SKYNEXT

Comprendre

User Manual · VST3 Compressor

Plugin version 1.0.3 · English edition

Contents

1. Welcome to Comprendre 8
A name with a mission 8
What you have in your hands 9
Every school, and everything in between 10
What you will hear 10
Who is it for? 11
Comprendre as an analysis tool 12
How to read this manual 12

2. A History of Compression 14
Protecting the transmitter: the birth of the limiter (1935-1950) 15
The golden age of tubes (1950-1960) 16
Light as the gain element: the LA-2A (1962) 17
The transistor: the 1176 (1967) 18
Rupert Neve's diodes (1969) 18
The VCA: precision (1970s and 1980s) 19
Analog revival and the first digital tools (1990s) 20
The 21st century: high-end mastering and plugins 21
The big ideas, and where to find them in Comprendre 21

3. The Interface 23
The main row 24
The knobs 24
Yellow: what has been changed 25
The tab bar 26
The glass pane 26

4. The Graphs 27
The vertical scale: linear or decibels 28
1. The graduated scale 29
2. The present moment 29
3. The level distribution 29
4. The transfer curve 30
The reference lines 31
5. The history 31
6 and 7. The attack and release shapes 35
8. The waveshaper curve 35
The glass pane 36
Starting ideas 36

5. The MAIN Tab and the Presets 37
Saving and loading a preset 38
Demo version and registration 40

6. Overview of the Signal Chain 41
Sample-by-sample processing 41
The complete path 41
What is measured, what is compressed 43
Inside the compressor 45
The tab names 46
What Comprendre does with its default settings 47
Future and present: the lookahead 48
Latency 48

7. Input, Output, Clippers, and Headroom 49
Oversampling 50
Input gain 51
Left/right or Mid/Side 51
Per-channel gains: the external headroom 52
The clippers 52
Headroom: the internal headroom 55
After the compressor: the OUTPUT tab 56
Makeup and post-clipper 58
Output gain 59
Linking or unlinking the channels 59
Dry/Wet and polarity 60
Hearing what's going on: the monitors 61

8. Auto-Volumes 63
The common principle 64
Out->In: automatic compensation 65
In->Out: the input follows the output 67
In->Sc and Out->Sc: following the sidechain 68
Summary: where each auto-volume acts 69

9. How Comprendre Compresses 70
The common engine 71
Two envelopes: before and after the curve 72
The default compressor 74
Sample-by-sample computation 79
One engine, several uses 80
What the graphs show during simple compression 82
Roadmap: from stage to chapter 83

10. The Transfer Curve 85
How it works 86
Threshold, ratio, and knee 92
The auto knee 95
Reduction limits: GR Max and GR Min 98
Ratios below 1 and negative ratios 100
The gate 102
Amplifying the reduction: Gr Amp 105

Example settings 106

11. The Envelope Follower 108

How it works: covering a fraction of the distance 109
Six envelopes in one 110
Linear or decibels 111
Branching or decoupled 112
The Mu envelope: a speed that depends on level 116
The Gamma envelope 121
Comparing the envelopes on the same signal 124
Blending the envelopes 124
What each envelope takes from the A/R settings 124
The four envelope followers 125
Practical workflows 127

12. Attack and Release 129

Time, coefficient and altitude 130
Automatic attack and release 134
The shape: Shape 137
The return: Ret From and Ret Spd 139
The modes: logarithmic, linear, zero 142
The shape graphs 146
The four attack and release sets 147
Practical workflows 148

13. Detection 150

How it works 151
The three streams 152
Dosing each stream: volumes and powers 157
Merging the streams: Det Mix 160
Formulas 162
A separate control: auto secs 165
Setting detection with the graphs 167
Example settings 167
Bias detection: DET (bs) 168
Starting ideas 168

14. What Is Measured, What Is Compressed: Feedback, Filters, and Target 170

How it works: two branches 171
The detection branch (INPUT tab) 173
The target branch (OUTPUT tab) 186
Linkwitz-Riley or FIR 190
Example settings 197
Summary 198

15. Gain Reduction and the Opto Designer 199

Where these settings act 200
Shaping the envelope 202
The opto designer 207

The opto's saturation product 216
Practical workflows 221

16. The Saturation Product 222
How it works: processing what the saturation adds 222
The processing chain 224
The amount: amp 226
The feedback delay 227
The three filter bands 230
The mix: src mix and band mixes 234
The limit: dst limit 236
Back into the signal 237
Practical workflows 238
The F (gr) and F (bs) tabs 238
Starting ideas 239

17. The Waveshaper (FET Designer) 240
How it works 241
The displayed curve 243
The amount of saturation: Fet gr and Fet env 243
The drive 246
Fet even: the second harmonic 247
Fet odd: the odd harmonics 248
Fet Thresh and Fet Knee: the clipper 249
Fet *: the multiplier 251
Fet sin * and Fet sin x: sine and DC bias 251
Fet Precision: bit crusher and spreader 254
Which harmonics? 256
The product delay 257
Listening to the product: Monitor Fet 259
Practical workflows 259
Starting ideas 260

18. The Dynamic EQs 261
How it works 262
Raising or lowering a band with compression 265
Where they sit in the chain 266
The sends: target, detection and waveshaper 267
The DEQ controls 269
DEQ 2: the EQ after the waveshaper 273
Practical workflows 276
Starting ideas 277

19. Introduction to the Bias 278
A real vari-mu compressor 279
The normal path and the inverse path 281
The bias height 284
Where the saturation comes from 284

The bias saturation product 287
Nothing runs while the five amounts are at 0 288
Recreating the article's waveform: starting settings 289
The bias over time 291
Starting ideas 291
What's next 291

20. The Bias Source 292

Computing the bias value 293
The inverse path 294
Where the bias comes from: the output or the gain reduction 294
The Feedback 299
The bias detector: DET (bs), ENV (bs), A/R (bs) 300
The bias curve 306
The detector input: Bias input add, mul and Fm In 308
Seeing the bias in the graphs 312
Example sources 313
Starting ideas 313

21. Destination and Fusion 315

The order of operations 316
Hearing what the bias does 318
The output height 318
Before or after the gain: pre and out 321
Bias Fm On: a drive before the limiter 321
The bias limiter 322
Fusion: unbiasing before recombining 325
The combinations 327
The bias saturation product 330
Practical workflows 334
Starting ideas 335

22. Lookahead 336

How it works: two chains offset in time 337
Looking into the future, or the past 340
The two schools 341
The hold: pk and min max 343
The dedicated follower: ENV (la) and A/R (la) 348
Lookahead in reverse 349
Latency 351
Practical workflows 352

23. Modulations 354

How it works 355
The sources 357
Shaping 360
The rest value 363
The targets 364

Reading the modulations on the graphs 369
Workflow: a frequency-sensitive ratio 369
Workflow: a frequency-dependent release 370
Starting ideas 370

24. Analyzing and Imitating a Compressor 372

How it works 372
The setup 373
The method 374
Precautions 375
Comprendre versus itself 376
From analysis to presets 376

25. The Presets: Eighty Compressors to Explore 378

How these presets were made 378
Loading a preset 379
The preset map 380
What the presets teach us 383
Light that compresses: the optos 385
Transistor speed: the FETs 395
The breath of tubes: vari-mu and tube compressors 406
The precision of the VCA: bus and console compressors 422
The chameleon: the Distressor family 436
British grain: the diode bridges 439
The mastering room 443
Digital inventions 452
Color without compression: saturators, transformers, clippers 465

26. Starting Points 474

Three habits before you start 474
By role 474
By source 477
Now it’s your turn 478

Appendix 1 — Settings Reference 479

Appendix 2 — Glossary 493

Points to Confirm (working document) 496

CHAPTER 1

Welcome to Comprendre

Congratulations, and thank you: you now own a compressor unlike any other.

A conventional compressor, hardware or software, is a series of choices frozen in place by its designer: a particular way of measuring the signal, a particular curve, a particular way of releasing, a particular grain of saturation. Those choices make up its personality, and that personality is why we love it. Comprendre turns the problem on its head. Each of those choices, usually soldered into a circuit or buried in an algorithm, has become a control you can turn, and most of the time blend: where other compressors give you a switch between two behaviors, Comprendre almost always gives you a knob that moves gradually from one to the other.

The result is an instrument in its own right. It can stay out of the way, with clean, transparent compression. It can also bite like a transistor compressor pushed to its limits, release with the smoothness of an optical cell, or thicken a sound the way tube compressors do. Above all, it knows how to **listen to others**: thanks to its analysis mode, it can display the gain curve of another compressor and help you imitate it.

Comprendre is a VST3 plugin. This manual describes version **1.0.3**, the version shown in its interface.

A name with a mission

Comprendre was designed and developed by François (Skynext), from mid-March 2023 to mid-July 2024. The name was not chosen at random. *Comprendre* is French for “to understand,” and that is its first meaning: understanding what a compressor does. The name also plays on its syllables, *com-* as in compressor and *prendre*, French for “to take”: the original ambition was to be able to reproduce the behavior of any compressor on the market, setting by setting, and to have the tools to check that you have succeeded.

> **A word from the author**
>
> François explains this at the start of his video on analysis: he named his compressor Comprendre because he wanted to be able to imitate every compressor on the market, and the analysis mode is the tool for that ambition. At the end of his long walkthrough video, he deliberately comes back to this mode, which he considers the product’s real added value.

To get there, he studied existing compressors, their schematics when they had been published, and the literature: the landmark study by Giannoulis, Massberg and Reiss on digital compressor design, the blog article by French manufacturer Pulsar Audio on how "vari-mu" tube compressors work, and a video by Dan Worrall on lookahead that inspired him a great deal. Every time he discovered a new "school" of design, he added it to Comprendre instead of picking one. That includes the ones he liked less: major manufacturers use them, so Comprendre had to be able to imitate them.

What you have in your hands

By default, Comprendre compresses in the simplest, cleanest way there is: peak detection, a linear transfer curve, a logarithmic envelope follower after the curve. It is an excellent everyday compressor, and chapters 3 to 10 are all you need to use it that way.

But every stage of this compressor has been opened up, taken apart and made adjustable:

- **detection** combines three measurements (peak, RMS, envelope), each with its own level and power, plus formulas to combine them, and moves gradually from feed-forward to feedback;
- the **transfer curve** is computed linearly, in decibels or anywhere in between, with an automatic knee that follows the level of the sound, gain reduction limits, a gate and negative ratios;
- the **envelope follower** comes in branching or decoupled form, linear or in decibels, with two additional algorithms ("Mu," whose speed varies with level, and "Gamma," a four-pole follower);
- **attack and release** have an adjustable shape, an automatic mode and three tracking modes;
- around the compressor sit three color "designers," each inspired by one of the great families of hardware: an **opto designer**, a **waveshaper** that imitates the saturation of FET transistor compressors, and a **Vari-Mu designer** (the bias) that simulates the biasing of a tube;
- on top of that come two **dynamic EQs**, a **lookahead** that can also look into the past, **modulations**, **filters** (all the way up to linear-phase FIRs) to choose what is detected and what is compressed, **Mid/Side** processing, **clippers** with anti-aliasing, **oversampling** up to 16 times and **auto-volumes** for level-matched comparisons.

All told, that is more than 360 parameters spread across 25 tabs, eight graphs that show what the compressor is doing, and **80 presets**, each named after the plugin François imitated to build it.

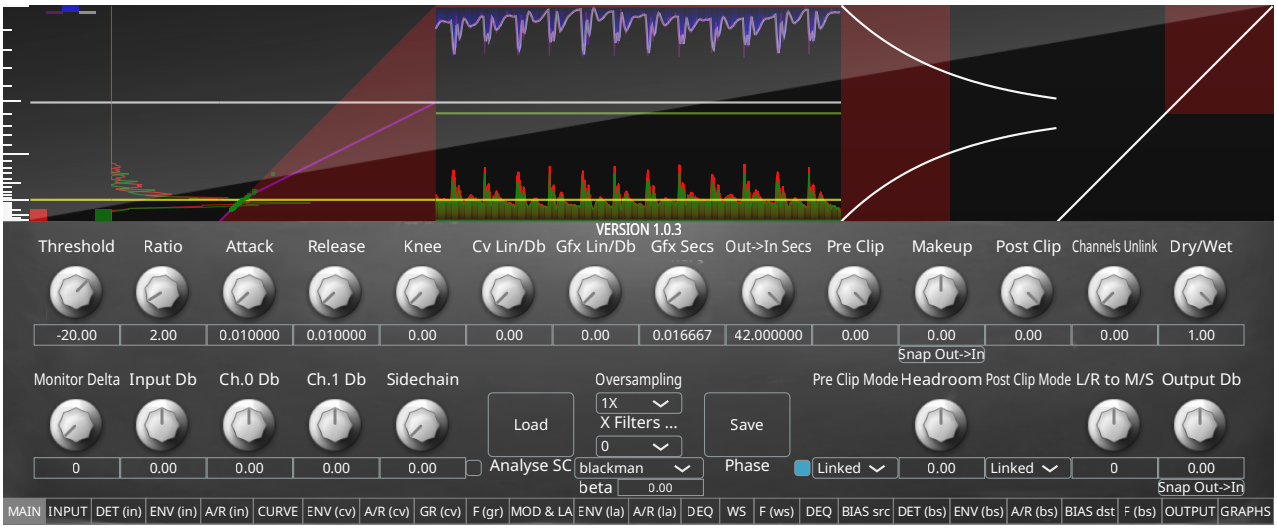


Figure 1 — Comprendre in full: the graphs, the main row and the MAIN tab.

Every school, and everything in between

In his videos, François often talks about “schools.” There are two schools for following the envelope: according to him, roughly half the compressors on the market “branch” it and the other half “couple” it. Some place the follower before the transfer curve, others after it; some compute in decibels, others in amplitude; some look ahead by delaying their detector, others by anticipating their envelope. Some measure the peak, others the average power. Some compress what they hear (feed-forward), others what they have just produced (feedback).

You can hear these choices, sometimes subtly, sometimes dramatically. Comprendre offers them all and, more importantly, makes them **blendable**: you can be at 30% feedback, halfway between the linear curve and the decibel curve, half in automatic release. Much of a compressor’s color lives in these in-between settings. That is what lets Comprendre get close to very different machines, and what will let you invent your own.

> **A word from the author**
>
> Faced with these choices, François’s advice is always the same: try it and listen. Some settings, he says, make only a small difference on their own, but can make a huge one once the rest of the setup changes.

What you will hear

Comprendre doesn’t have one sound: it has dozens. Here are a few examples of what this manual will teach you to get out of it:

- **transparent** compression that holds a vocal or a mix steady without ever being heard working, with the 200 ms automatic release that François considers the most natural;
- the **bite** of a fast compressor grabbing the transients of a drum kit, with saturation that grows as it compresses, just like on transistor machines;

- the **smoothness** of optical compression, whose gain trails behind the music like a cell that lights up and fades out;
- the **thickness** of a tube compressor, with asymmetric saturation that arises from the bias and evolves with the compression;
- drums that **breathe**, thanks to a dynamic EQ that boosts a band when they hit;
- compression that **anticipates** attacks, with a lookahead whose envelope you draw yourself;
- **never-before-heard** saturations for sound design: white-noise grain reserved for the peaks, bit crushing, bias fusion in absolute value, delay on the saturation product…

Who is it for?

Comprendre was designed for everyone who works with sound, from the first home studio to the mastering room. Everyone will find their own way in.

You compose, produce or make beats. Start with the presets: eighty different characters, built by imitating renowned compressors, ready to load and try on your tracks. The main row (threshold, ratio, attack, release, knee) is then all you need to adapt them. When you like a preset, the tabs that have turned yellow show you how it is built. Chapter 25 guides you through them.

You have a home studio and want to finally understand compression. The plugin's name is a promise. The eight graphs show in real time what the compressor is doing: the detected level, the curve, the gain reduction, the envelope, the saturation, on a linear or decibel scale. The analysis mode even lets you compare two Comprendre settings against each other, to see exactly what a parameter changes. This manual explains every stage, with diagrams and curves computed from the plugin's code.

You mix. You will appreciate the separation between what the compressor listens to (the detection) and what it processes (the target): compressing the highs based on the low end, or processing just one frequency band. You will also appreciate the automatic knee, which François recommends for vocals, the two dynamic EQs, the feed-forward/feedback blend and the Mid/Side mode. And the analysis mode lets you get close to the behavior of a reference compressor, even one you don't own: all it takes is an excerpt before and after processing.

You master. Comprendre gives you linear-phase FIR filters (up to 1025 coefficients, eight windows), oversampling up to 16 times in IIR or FIR, a pre-clipper to place before limiting, clippers with anti-aliasing, a lookahead with peak hold, and auto-volumes that make up for the loss of level so you can judge level-matched. The release can even be modulated by frequency, thanks to the dynamic EQs: François sees this as a way to achieve an overall coherence that borders on mastering.

You do sound design. The three saturation designers, negative ratios, the inverse lookahead (which makes the compressor act *after* the sound), modulations between modules and the delay on saturation products add up to an almost limitless playground. François himself has a lot of fun with them in his videos.

You teach or study audio. Few tools let you demonstrate, on a single plugin, the difference between peak and RMS detection, between feed-forward and feedback, between branching and decoupled envelopes, between a linear curve and a decibel curve. Comprendre does, setting by setting, with the graphs to see it and your ears to hear it.

Comprendre as an analysis tool

The second founding idea is **analysis**. Comprendre can receive the signal processed by another compressor on its sidechain input and display that compressor's gain curve on its own graph. You can then adjust Comprendre's settings until its envelope hugs the model's, and check the result by ear by listening to the difference between the two. That is how François built the included presets. Chapter 24 describes this method step by step.

How to read this manual

Chapter 2 tells the story of compression, from radio limiters to today's plugins, and shows where each of the great families of machines that shaped it lives on in Comprendre.

The manual then follows the order in which François presents the plugin in his demo video: the interface and the graphs, then the signal path around the compressor, then the compressor itself, stage by stage, then the color modules (opto, waveshaper, dynamic EQs, bias), and finally time (lookahead, modulations). Chapter 24 describes the analysis and imitation method, chapter 25 presents the 80 included presets, and chapter 26 suggests starting points based on your line of work and your sound source.

You don't need to read everything to use Comprendre. Chapters 3 to 10 are enough to use it as a very good classic compressor. The chapters that follow are for those who want to shape a specific behavior or imitate an existing unit. And if you can't wait to hear what Comprendre has under the hood, nothing stops you from starting with chapter 25.

Everything this manual says about how the plugin works has been checked against its source code. When the actual behavior differs from what you might expect from a setting's name, the manual describes the actual behavior. The descriptions of sound, on the other hand, are suggestions for listening: compression depends enormously on the source, and your ears will always have the final say.

Conventions

Each setting is introduced by a **parameter card** like this one, whose header is extracted automatically from the plugin's code: the name shown on screen, the tab, the range, the default value and the internal identifier (the one your host shows in its automation lists).

Threshold main row tab

Range: −60 to 0 dB **Default:** −20 dB **ID:** `seuil`

The text of the card explains what the setting does, how it interacts with the others and, often, how to use it.

Setting names are the ones shown in the interface, which is in English: **Threshold**, **Release Mode**, **Bias Fm On**… Tabs are referred to by their on-screen names: the **CURVE** tab, the **A/R (cv)** tab.

Boxes flag points of particular interest:

> **Tip**
> An effective way to use a setting.

> **Caution**
> A behavior that may surprise you, or a setting to handle with care.

> **♪ In the studio**
> An idea for musical use: on which source, with what intention, and what to listen for.

> **A bit of history**
> A machine, a technique or an era from the history of compression, and what remains of it in Comprendre.

> **A word from the author**
> A tip or an explanation given by François in his videos.

> **Going further**
> The mathematical or algorithmic detail, for those who are interested.

The diagrams and curves in this manual are computed from the formulas in the source code; the interface views are reconstructed from the description of the interface in that same code.

CHAPTER 2

A History of Compression

Every famous compressor is an answer to a problem, built with the components of its day. The tubes of the 1950s, the optical cell of the LA-2A, the transistor of the 1176, Rupert Neve's diodes, David Blackmer's VCA, the digital processors of the 1990s: each technology imposed its own way of measuring the signal, its own speed, its own curve, its own grain. Those constraints became sounds, and those sounds became references. When an engineer asks for "a little Fairchild on this vocal" or "the glue of an SSL on the bus," it is this whole history they are calling on.

Comprendre was born of a desire to understand these machines and to imitate them. This chapter tells their story, from radio limiters to today's plugins, and shows at each step where their legacy lives on in the plugin. You will see that almost every module in Comprendre answers an idea that first appeared in a piece of hardware, and that what used to be a choice soldered into a circuit has become a control.

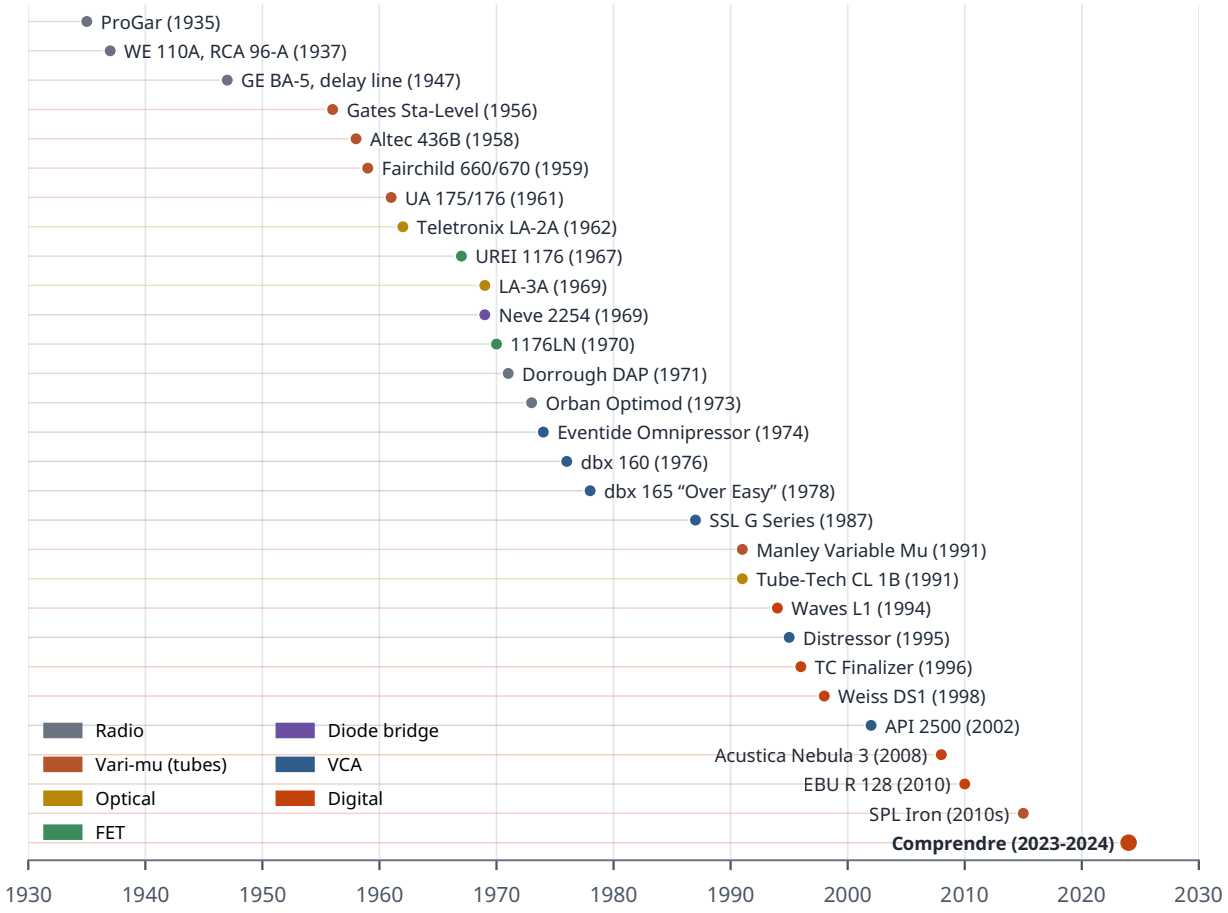


Figure 2 — A few milestones in the history of compression, grouped by the technology of the gain reduction element. The dates are those given by most sources; some are disputed (see the text).

Protecting the transmitter: the birth of the limiter (1935-1950)

Compression was not born in the studio, but in radio. In the 1930s, an overmodulated AM transmitter would cut off its carrier and go off the air: a device was needed to keep the peaks from going over the limit. In 1935, Al Towne, an engineer at station KSFO in San Francisco, designed the ProGar, perhaps the first true audio processor: an automatic gain control combined with a peak limiter. In 1937 came the first commercial peak limiters, the Western Electric 110A, followed by the RCA 96-A.

All of these devices relied on the same principle, one that would endure for decades: a tube whose gain (the "mu") varies with the DC voltage applied to its grid. The signal level is rectified into a control voltage, which changes the **bias** of the tube and therefore its gain. A push-pull circuit cancels the control voltage at the output. When the tubes age and drift out of balance, that cancellation becomes imperfect, and you hear "thumps" with every change in gain.

In 1947, the General Electric BA-5 broke new ground: it ran the audio through a delay line so that the control had time to act before the sound arrived. It sounded cleaner than anything before it, so much so that NBC, which was supposed to buy only RCA equipment, repainted BA-5s in RCA colors.

> **In Comprendre**
>
> So **lookahead** is not a digital invention: as early as 1947, the sound was being delayed so that the control would arrive in time. Comprendre's lookahead (chapter 22) goes further by letting you choose what is anticipated, the detection or the envelope. It can even look into the past. As for tube biasing and its imbalances, they are at the heart of the **Vari-Mu designer** (chapters 19 to 21).

The golden age of tubes (1950-1960)

The 1950s belonged to the 6386, a "variable-mu" (remote-cutoff) dual triode whose grid is wound non-linearly: its gain drops progressively as the signal rises, with far less distortion than sharp-cut-off tubes. It powered the great broadcast units, such as the Gates Sta-Level (1956), a feedback leveling amplifier with a soft knee and a ratio of about 3.3:1. Its **Single/Double** switch chose between a slow, fixed release and a program-dependent release: the 1956 manual recommended the first for classical music and the second for speech.

Altec Lansing built the 436 series, inexpensive tube compressors intended for telephone and public-address use. The 436A had no controls at all; the 436B (1958) added input gain, and the 436C a threshold and a release, probably because studio engineers were modifying the units themselves. Producer Joe Meek shortened the attack and release on his to get the pumping effects that defined his sound. At Abbey Road, EMI's technicians turned 436Bs into the **RS124**, with a six-position release selector and a **Hold** position that freezes the gain. Each unit had its own timings: the most aggressive ones were used for tracking, the others on the bus or for disc cutting. Geoff Emerick used them on Paul McCartney's bass.

> **The Fairchild 660 and 670, the Holy Grail**
>
> Rein Narma, an Estonian-born engineer, built consoles for Rudy Van Gelder and Les Paul. It was Les Paul who asked him for a limiter and introduced him to Sherman Fairchild. The Fairchild 660 (mono) and 670 (stereo) were manufactured from 1959 on. The first 660 went to Rudy Van Gelder, who cut the Blue Note records. The 670 weighs around thirty kilograms and has some twenty tubes: four 6386 dual triodes per channel, driven in push-pull. Its six **Time Constant** positions range from a 0.2 ms attack with a 0.3 s release to program-dependent releases that can reach 25 s in position 6. Its **Lateral/Vertical** mode compresses the sum and the difference of the channels separately: designed for vinyl cutting, it is the ancestor of Mid/Side processing. Abbey Road bought a dozen, mainly for the Beatles' vocals, and Geoff Emerick recalled that the drums run through the 660s "became the sound of *Revolver* and *Sgt. Pepper*." More recently, you can hear it on Amy Winehouse's "Rehab."

Two Fairchild controls, long hidden on the back or inside the unit, shed particular light on Comprendre. The **DC Bias** shifts the bias of the tubes: it acts on both the knee and the ratio, and takes the unit from a hard-knee, high-ratio limiter to a wide-knee, low-ratio compressor. The **Balance** control balances the two halves of the push-pull. Properly centered, it minimizes the "thud," a deflection added to the signal at every attack; off center, it produces a thud of one polarity or the other. This is the historical example of an **asymmetric** saturation born of biasing.

French manufacturer Pulsar Audio, which has modeled this type of compressor, devoted a blog article to this mechanism: the gain reduction serves as the bias voltage, the signal is added to it, and the harder you drive it, the more the compression blends with saturation, especially on transients. It was this article that gave François the idea for Comprendre's bias.

> **In Comprendre**
>
> The **Vari-Mu designer** (chapters 19 to 21) transposes this mechanism: a bias value computed from the output or from the gain reduction, proportional "heights" added to the signal, an inverse path that plays the role of the second half of the push-pull, then the recombination of the two paths, which gives rise to an asymmetric saturation. The **Mu** envelope follower (chapter 11), whose speed varies with level, was designed to imitate the curve of a vari-mu. Program-dependent releases live on in the **automatic mode** (chapter 12), and Lateral/Vertical in **L/R to M/S** (chapter 7). The presets in the vari-mu family (chapter 25) use all of this extensively.

In 1961, Bill Putnam, founder of Universal Audio, released the 175 and then the 176, two vari-mu compressors whose attack and release knobs already turned "backwards," faster clockwise. That ergonomic quirk would carry over to his next compressor, which would make history.

Light as the gain element: the LA-2A (1962)

In the early 1950s, James F. Lawrence II was an engineer at radio station KMGM in Los Angeles. Tired of "riding the faders," he went looking for an amplifier that would level the sound on its own. He founded Teletronix in Pasadena in 1958 and, after the LA-1 and the LA-2, released the **LA-2A**, in 1962 according to most sources.

Its gain element is the **T4** cell: inside a sealed housing, an electroluminescent panel lights up two photoresistive cells. The louder the signal, the brighter the panel glows, the lower the cell's resistance drops, and the more the gain goes down. The audio path sees nothing but a resistor: that is one of the reasons for the unit's renowned transparency. The circuit is a feedback design, and a pre-emphasis control on the back made the detector more or less sensitive to the voice band, a legacy of radio.

All of the LA-2A's charm comes from the physical behavior of the cell. The attack, about 10 ms, depends on frequency. The release happens **in two stages**: the cell recovers about half its resistance in some sixty milliseconds, then the rest in one to fifteen seconds according to the original manual (0.5 to 5 s according to the reissue's spec sheet). And the cell has a **memory**: after heavy or prolonged compression, the second stage is slower. So the compression is "program-dependent" without any circuit having been designed to make it so. The ratio in **Compress** mode, with its soft knee, is about 4:1, but it varies with the source. The LA-2A has been used on countless records, from Nirvana's *Nevermind* to Alanis Morissette's *Jagged Little Pill*.

It has many descendants: the LA-3A (1969), a solid-state version designed by Brad Plunkett, with a much faster attack; the Danish Tube-Tech CL 1B (1991), which adds adjustable attack, release, ratio and threshold, and whose **Fixed/Manual** mode chains a fast release with a slower, adjustable one; and the Shadow Hills, which places an optical stage ahead of a VCA stage.

> **In Comprendre**
>
> The **opto designer** (chapter 15) offers a model of this behavior: a second envelope follower whose times get shorter as the gain reduction increases (**Opto severity**), in steps (**Opto precision**, which François likens to the number of simulated cells), with a saturation that pulls it closer to the curve during attacks. The **detection filters** (chapter 14) play the role of the pre-emphasis. François adds one more ingredient: the envelope **offset**, his favorite setting for imitating compressors; on the LA-2A, he says, the effect is striking. Chapter 25 shows how the optical presets go about it, and you will see that not all of them go through the opto designer.

The transistor: the 1176 (1967)

In 1966, Bill Putnam replaced the tubes with field-effect transistors (FETs). His 175 and 176 became the **1176**, released in 1967: the first true all-transistor peak limiter. The FET acts as a voltage-controlled resistor. It is an inherently non-linear component, and Putnam's whole art lay in keeping it within its linear region. The "LN" (low noise) circuit of 1970 helped with that.

The 1176 has no threshold control: you set the amount of compression with the input level. Its ratios range from 4:1 to 20:1, its attack from 20 to 800 µs, its release from 50 ms to 1.1 s. It is a feedback design, and its class A output, with a transformer designed by Putnam, is "essential to the sound," according to its manufacturer. It also gave us a mode that has become legendary, **all buttons**: all four ratio buttons pushed in at once. The bias points of the entire circuit shift, the attack lags behind the first transients, and the distortion rises dramatically. It is used on drums, room mics, "dirty" bass and "in-your-face" vocals. Next to the soft, warm Fairchild, the 1176 brightens the tone and adds bite; its slight distortion helps the bass cut through, and it tops producer polls for bass. The 1178 is the stereo version.

> **In Comprendre**
>
> The **waveshaper** (chapter 17), which François calls the "FET designer," was designed to simulate the saturation of FET compressors. By default, its amount follows the gain reduction: the more the compressor compresses, the more it saturates, just like the hardware. Attacks of a few tens of microseconds are within reach with **Attack**, and the attack and release tracking modes (chapter 12) let you harden its character. Nine presets target FET compressors, most of them 1176s (chapter 25).

Rupert Neve's diodes (1969)

In 1969, Rupert Neve and David Rees designed the **2254**, one of the first dynamics processors to exploit the electrical properties of diodes. Arranged as a bridge, the circuit is symmetrical and distorts less, but it is very "lossy": the signal is attenuated by about 40 dB before the bridge, then brought back up. The knee is soft, spread over about ten decibels, the ratios range from 1.5:1 to 6:1, and an **Auto** mode combines fast times for isolated peaks with slow times for sustained levels: the program "rides on a slowly moving gain platform." The **33609** takes up the same design in stereo. Their sound is described as creamy, round, silky; Dave Berners, Universal Audio's chief scientist, notes that the distortion of a diode bridge can reach 10% under certain conditions, hence a distinctive sound.

> **In Comprendre**
>
> The color of a diode bridge is the job of the **waveshaper** (chapter 17) and the **dynamic EQs** (chapter 18); the gain platform is the job of the **automatic mode** (chapter 12). Three presets target this family (chapter 25).

The VCA: precision (1970s and 1980s)

In 1971, David Blackmer developed a voltage-controlled transistor gain cell, the **VCA**, while working on noise reduction for tape recorders. The **dbx 160** (1976) paired it with an **RMS** detector, which measures the power of the signal rather than its peaks, and with a **feed-forward** topology: the control signal is taken from the input and "fed forward" to the VCA, which allows fast attacks and infinite ratios without instability. The original 160 has a hard knee; the soft **Over Easy** knee arrived on the dbx 165 in 1978. Its sound is clean and predictable, and producer Glen Ballard considers the 160X the best drum compressor he has found.

The VCA changed the very nature of the compressor. Dave Berners sums it up this way: very linear, very flexible, potentially transparent, it brings none of the "good accidental properties" of tubes, optical cells or FETs. **All the character comes from the design of the timing and the curve.** That is exactly Comprendre's territory: it turns that design into a set of controls.

The 1970s also invented new uses. The Eventide Omnipressor (1972 to 1974), which its manufacturer presents as the invention of the side-chain, separates the detector from the VCA so that one source can drive another, turning dynamics into an effect. In radio, Mike Dorrough (1971) and Bob Orban (Optimod, 1973) split the signal into bands, because in wideband processing the bass "modulates everything else" and the listener hears the pumping.

> **The SSL bus compressor and its 'glue'**
>
> Solid State Logic built a bus compressor into its consoles as early as the 4000 B of 1976, according to the manufacturer; it remains associated with the G Series, introduced in 1987. Its VCA came from dbx. Its values are famous: attack from 0.1 to 30 ms; release of 0.1, 0.3, 0.6 or 1.2 s, or **Auto**; ratio of 2, 4 or 10:1. Its automatic release has **two stages**, 100 ms and 12 s: loud elements are released quickly, quieter ones slowly. Its topology is hybrid: according to SSL, "the side-chain itself is a feedback topology, but the signal that feeds it is taken from a feed-forward position." SSL presents it as the de facto choice for "gluing" a mix, and the classic recipe is to "tickle the VU meters at 2:1 with auto release." In 1996, 83% of *Billboard* number ones had been produced on an SSL console.

> **In Comprendre**
>
> The VCA is the main row and the **transfer curve** (chapter 10), especially in decibels, **RMS detection** (chapter 13), the gradual move from **feed-forward to feedback** (chapter 14), and the **automatic release** (chapter 12). While studying the SSL schematic, François noticed that its feedback "isn't really feedback": he added the **Bias input mul** setting to the bias (chapter 20) to imitate it more closely. Later, the API 2500 (2002) would offer an **Old/New** switch between feedback and feed-forward, three knees and a detection filter, the **Thrust**; Comprendre offers each of these choices as continuous controls.

Analog revival and the first digital tools (1990s)

The 1990s saw the return of tubes. Manley built its **Variable Mu** in mono from 1991 and in stereo from 1994: a 5670 dual triode that is "constantly re-biased," as in the Fairchild. The manual makes the case for feedback ("most modern compressors use a feed-forward circuit, which strikes us as not very musical") and describes the measurable signature of this re-biasing: **the ratio rises and the knee softens as the gain reduction increases**. The unit became a standard in mastering studios, known for giving a project "glue." Later, the SPL Iron would go as far as offering the **bias voltage** as a user control.

In 1995, Dave Derr, formerly of Eventide, sold the first **Distressors**. This analog VCA compressor, controlled by digital electronics, partly emulates the dbx 160, the LA-2A, the Fairchild and the 1176. Each "ratio" is a different curve, with its own knee; 10:1 is an **Opto** mode with long releases; **Nuke** is a brickwall limiter, with a logarithmic release that accounts for "a big part of the sound" of the unit. Its **Dist 2** and **Dist 3** modes add, respectively, mostly the 2nd harmonic, the warmth of tubes, and then the 3rd harmonic, which flattens the top and bottom of the waveform like magnetic tape. A single unit that imitates several families: it is the machine closest to Comprendre's philosophy.

Meanwhile, digital joined the party. In 1994, the **Waves L1** became the first mass-produced brickwall lookahead digital limiter. The TC Electronic **Finalizer** (1996) put multiband compression within reach of every studio. The **Weiss DS1** (1998) became the benchmark digital mastering compressor: its side-chain measures both peak and RMS and chooses its release based on the ratio between them, it doubles its sample rate to avoid aliasing, it has a soft knee and a lookahead (**Preview**), and it can compress just one band.

> **The loudness war**
>
> The CD, and then digital limiters, set off the race for loudness. Oasis's *(What's the Story) Morning Glory?* (1995) shows an average RMS level of −8 dBFS on many tracks; Red Hot Chili Peppers' *Californication* (1999) has audible clipping throughout the album; Metallica's *Death Magnetic* (2008) drew the general public's attention to clipped masters. Normalization has put the brakes on it: the EBU R 128 recommendation (2010) sets a target of −23 LUFS for broadcast, and streaming platforms now normalize loudness by default (Spotify aims for −14 LUFS). On those platforms, a crushed master no longer sounds louder: it simply gets turned down, and it has lost its dynamics.

> **In Comprendre**
>
> **Lookahead** (chapter 22), **oversampling** up to 16 times and **clippers** with anti-aliasing (chapter 7), **detection** that blends peak, RMS and envelope (chapter 13), **dynamic EQs** and band **filters** (chapters 14 and 18): all the tools of digital mastering are here. The Distressor has its own family of presets (chapter 25), and the **waveshaper** distinguishes even and odd harmonics, like its Dist 2 and Dist 3 modes.

The 21st century: high-end mastering and plugins

The elysia **alpha compressor** (2006) sums up many of this chapter's ideas all by itself: a **Feed Forward** / **Feedback** switch (feedback tops out at a 2.5:1 ratio, while feed-forward goes all the way to limiting and even to negative ratios), an attack that shortens by itself on strong impulses (**Auto Fast**), a soft clipper, 2nd and 3rd harmonics, an M/S matrix, a parallel mix and a side-chain filter. The Bettermaker combines a VCA limiter, a two-stage analog clipper and even and odd harmonic generators.

But the big change was happening elsewhere: the machines were moving into the computer. Plugin developers emulate them in several ways. **Circuit modeling**, practiced for example by Universal Audio, rebuilds every component digitally. **Measurement-based modeling** observes the input and output at many levels and settings, without rebuilding the components. Acustica Audio samples the behavior of hardware using **Volterra kernels**, tiny snapshots of its response that an engine steps through according to level. Neural networks learn the behavior of a unit. When the original machine is digital, its code can even be ported: that is how Softube ported the code of the Weiss DS1, line by line.

Comprendre takes a different path. It does not emulate a machine: it provides the **ingredients** (detection, topology, curve, envelope, optical element, bias, harmonics), and its **analysis mode** (chapter 24) puts the approach of the "measurement-based" modelers in your hands: compare Comprendre's output with that of another compressor and adjust until they match. That is how François built the 80 presets of chapter 25, whose names read like a catalog of this history.

The big ideas, and where to find them in Comprendre

Idea	Where it comes from	In Comprendre
Limiter, infinite ratio	Broadcast limiters (1937), Waves L1 (1994)	**Ratio** at 42 (chapter 10), clippers (chapter 7)
Lookahead	GE BA-5 delay line (1947), Waves L1, Weiss DS1 Preview	Lookahead on the detection or the envelope (chapter 22)
Feedback or feed-forward	Early feedback compressors; dbx 160 in feed-forward; API 2500 Old/New, elysia alpha FF/FB	**FF->FB**, gradual (chapter 14)
Peak or RMS	dbx 160 RMS detector; combined peak and RMS on the Weiss DS1	Blend of peak, RMS and envelope (chapter 13)
Soft knee	Vari-mu and opto units, dbx 165 "Over Easy," the API 2500's three knees	**Knee**, **Knee Auto** (chapter 10)
Program-dependent release	LA-2A T4 cell, Fairchild positions 5 and 6, SSL Auto, Neve 2254 Auto	Automatic release (chapter 12), opto designer (chapter 15), Mu follower (chapter 11)
"Branching" and "decoupled" detectors	Paper by Giannoulis, Massberg and Reiss (2012), cited by François	**EvMu Br/Cp** (chapter 11)

Idea	Where it comes from	In Comprendre
Tube biasing, asymmetric saturation	Fairchild (DC Bias, Balance), Manley Variable Mu, SPL Iron	Vari-Mu designer (chapters 19 to 21)
Even and odd harmonics	1176, Distressor (Dist 2, Dist 3), elysia, Bettermaker	Waveshaper (chapter 17)
Mid/Side compression	Fairchild 670 Lateral/Vertical; Manley, elysia, TC System 6000	**L/R to M/S** (chapter 7)
Side-chain, filtered detection, de-essing	LA-2A pre-emphasis; Omnipressor; API 2500 Thrust	Sidechain, detection and target filters (chapter 14)
Multiband compression	1970s radio (Dorrough, Orban), TC Finalizer	Dynamic EQs (chapter 18), band filters (chapter 14)
Parallel compression	Dolby A noise reduction (1965), Mike Beville (1977), Bob Katz	**Dry/Wet** (chapter 7)
Oversampling	Weiss DS1-MK2	**Oversampling** up to 16X (chapter 7)

About trademarks

The names of machines, manufacturers and plugins mentioned in this manual belong to their respective owners. They are used to tell the story of compression and to identify the models François imitated. Their mention implies no affiliation with these manufacturers and no endorsement on their part.

CHAPTER 3

The Interface

Open Comprendre and you see, all at once, what a studio usually spreads across several units and often hides inside their cases. At the top, the graphs show what the compression is doing while you hear it. In the middle, a single row of knobs is all it takes to set up a complete compressor, from the threshold to the parallel blend. At the bottom, twenty-five tabs open up the machine stage by stage: detection, curve, envelope, then the imitation of an optical cell, the grain of a FET transistor or the biasing of a tube. And one very simple color code, yellow, tells you at every moment what has been changed: enough to read a preset like a recipe, or to spot at a glance the setting that changed the sound.

The Comprendre window has a fixed size of 1175 × 480 pixels. It is made up of four horizontal strips.

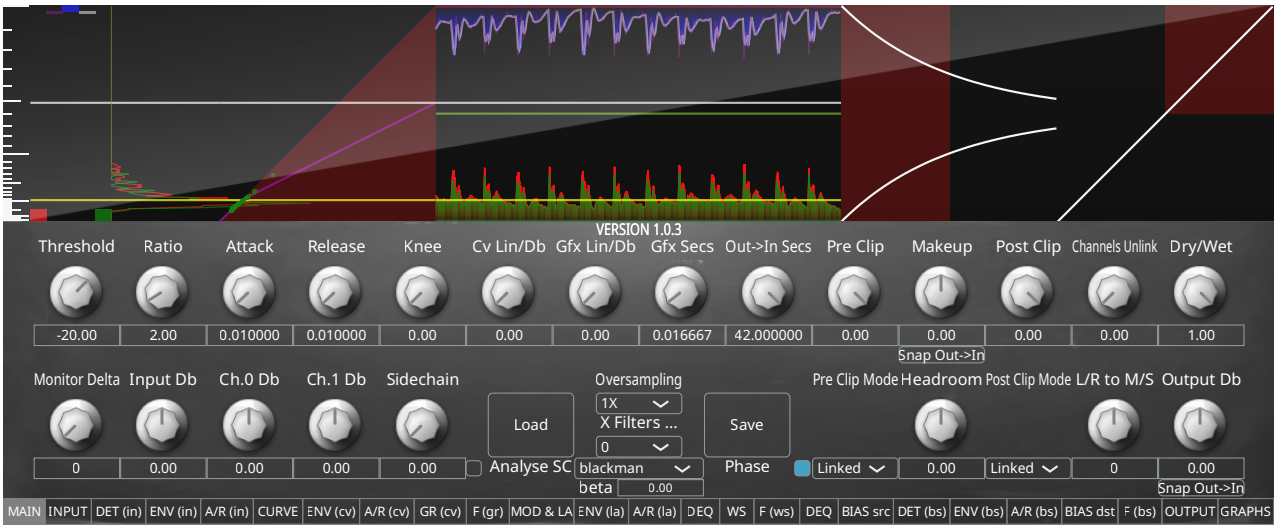


Figure 3 — The Comprendre window at rest, with the MAIN tab open. The graphs are simulated with the default settings.

1. **The graph area**, at the top (200 pixels high): eight displays that show in real time what the compressor is doing. They are described in chapter 4.
2. **The main row**, which holds the 14 most-used settings: Threshold, Ratio, Attack, Release, Knee, Cv Lin/Db, Gfx Lin/Db, Gfx Secs, Out->In Secs, Pre Clip, Makeup, Post Clip, Channels Unlink and Dry/Wet. It is always visible.
3. **The second row**, whose contents depend on the selected tab: up to 14 settings per tab.
4. **The tab bar**, at the very bottom: 25 tabs, from **MAIN** to **GRAPHS**.

Above the Gfx Secs column, the text **VERSION 1.0.3** shows the version of the plugin.

The main row

Setting	Role	Chapter
Threshold	threshold of the transfer curve	10
Ratio	ratio of the transfer curve	10
Attack	attack of the envelope follower after the curve	12
Release	release of that follower	12
Knee	width of the knee	10
Cv Lin/Db	curve computed linearly (0), in decibels (1), or a blend	10
Gfx Lin/Db	scale of the graphs: linear (0), decibels (1), or a blend	4
Gfx Secs	scrolling speed of the history	4
Out->In Secs	automatic level compensation	8
Pre Clip	threshold of the pre-clipper	7
Makeup	makeup gain, inside the clippers	7
Post Clip	threshold of the post-clipper	7
Channels Unlink	stereo linking of the compressor	7
Dry/Wet	final blend between input and output	7

Below the **Makeup** knob, the small **Snap Out->In** button transfers the compensation calculated by the Out->In auto-volume into the Makeup (chapter 8).

> **♪ In the studio: a complete compressor on a single row**
>
> The main row is not a stripped-down version of Comprendre: it is a complete compressor. Threshold, ratio, attack, release and knee for the character; pre-clipper, makeup and post-clipper to keep the peaks in check; **Out->In Secs** for level-matched comparisons; **Dry/Wet** for parallel compression. Several of the included presets live almost entirely on this row. **boz_manic** changes only two processing settings, **Ratio** at 4 and **Release** at 0.06 s. **cytomic_glue**, built by imitating a bus compressor plugin, sets **Attack** to 30 ms, **Ratio** to 4 and **Release** to 0.1 s there, and adds just two settings in the BIAS src tab. When you like a preset on a vocal or a bus, this is almost always where you will adapt it to your source: first **Threshold**, to get back the amount of compression you are after, then **Release**, so that the compression breathes with the tempo.

The knobs

Each knob has its name above it and a box showing its value below it.

- **Dragging** the mouse vertically or horizontally turns the knob; a travel of about 250 pixels covers the full range.
- **Double-clicking** the knob returns it to its default value.

- **Clicking in the value box** lets you type in a precise value.

The number of decimal places shown depends on how fine the setting is: two decimals for gains in dB (“-20.00”), six for times in seconds (“0.010000”), none for integer settings (“600”).

> **Knob position and travel**
>
> Many settings have a **progressive** travel: the mouse moves finely through the bottom of the range and faster through the top. This is the case for attack and release times, the ratio, the auto-volumes… The knob graphic, on the other hand, shows the value in proportion to the full range. That is why the **Attack** knob, set to 10 ms in a range of 0 to 2 s, looks almost all the way down.

Some settings are **drop-down menus** (for example **Pre Clip Mode**, **Oversampling**, **Channels Mix**) and a few are **checkboxes** (**Analyse SC**, **Phase**). Other knobs have only a few positions: **L/R to M/S** accepts only −1, 0 and 1, and the monitors only 0 and 1.

> **Controls once hidden on the back**
>
> On vintage hardware, much of what gives a compressor its character was not on the front panel. The Fairchild 670’s ratio was set with internal screws, from 2:1 to 30:1, and its **DC Bias**, which acts on both the knee and the ratio, was on the back or inside the unit. On the LA-2A, the **Compress/Limit** switch was originally on the back, as was the pre-emphasis control that made the detector more or less sensitive to the voice band. As for the Altec 436A, it had no controls at all. Comprendre goes the other way: more than 360 settings are available from the interface, and yellow flags the ones that have moved away from their starting value.

Yellow: what has been changed

By default, all setting names are written in white. As soon as a setting moves away from its default value, **its name turns yellow**. And if that setting belongs to a tab, **the tab’s name turns yellow too** in the bar at the bottom.

With more than 360 settings, this is an invaluable landmark: when you open a preset, you can see immediately which tabs have been touched, then, within each tab, which settings. To return a setting to its default value, just double-click it: its name turns white again.



Figure 4 — The INPUT tab of a preset: the modified settings and the tab name appear in yellow.

A few details:

- the 14 settings in the main row don’t color any tab: only their own name turns yellow;

- the FIR filter window menu (MAIN tab, below **X Filters Taps**) has no displayed name: changing it turns the **MAIN** tab yellow without any setting name in that tab being yellow;
- the comparison with the default value is exact: a setting brought back *by hand* very close to its default value, but not exactly on it, stays yellow. Double-clicking it, or typing the exact value into the box, takes care of that.

> **♪ In the studio: reading a preset like a recipe**
>
> Yellow turns every preset into a lesson. First load your "default" preset (see the tip below), then **kush_silka**: on the main row, **Ratio** (4) and **Release** (0.1 s) turn yellow; in the tab bar, **MAIN** (a **Headroom** of −7 dB) and **GR (cv)** (an **Env Offset** of −0.303). Four settings are all it takes to build this character. At the other extreme, **spl_iron** doesn't touch a single compression setting on the main row: everything happens in the **CURVE**, **WS** and **DEQ 2** tabs. To hear what a setting brings, note its value, double-click it to return it to its default, listen, then type the value back into the box. That is how François introduces the presets at the start of his long walkthrough video in French: you pick one at random, and the yellow tabs tell you right away what has been touched. Chapter 25 explains, preset by preset, why these settings were chosen.

> **Getting back to the starting point**
>
> Comprendre has no global reset button. The simplest approach is to save, the first time you use it, a preset containing the default settings (**Save** button in the MAIN tab), then reload it whenever you want to start from scratch. Removing the plugin and inserting it again on your track has the same effect.

The tab bar

The 25 tabs are arranged roughly in the order of the signal path. Chapter 6 explains the logic behind their names; in short:

- **MAIN** holds the global settings and preset management, and **GRAPHS** the display settings;
- **INPUT** and **OUTPUT** frame the compressor: what is measured, and what is compressed;
- in between, each module has one or more tabs, whose suffix tells you where it acts: (in) in the detection, (cv) after the curve, (gr) on the gain reduction, (la) for the lookahead, (ws) for the waveshaper, (bs) for the bias.

The selected tab has a gray background. At startup, that is the **MAIN** tab.

The glass pane

A semi-transparent image, depicting the reflection on a pane of glass, covers the graph area. It is purely decorative: the **Gfx Windows** setting in the GRAPHS tab lets you remove it so you can read the curves more easily (chapter 4).

CHAPTER 4

The Graphs

The graphs are Comprendre's great strength: they show, in real time, what the detector is measuring, what the transfer curve is asking for, what the envelope follower is doing and what actually comes out. Learning to read them is learning to understand what the compressor is doing.

Compression is probably the hardest effect to hear. It works a few decibels at a time, over a few milliseconds, and the ear rarely perceives it as a change in level: more as a snare that cracks less, a vocal that stays up front, a mix that tightens up or starts to breathe. The graphs make that work visible. You see the envelope dip with every snare hit, trail behind a bass note, come back up between two phrases. By seeing what you hear, you learn to hear what you see. It is a learning tool for students and teachers, a monitoring instrument for mixing and mastering engineers, and the foundation of the analysis mode, which overlays another compressor's curve on top of these (chapter 24).

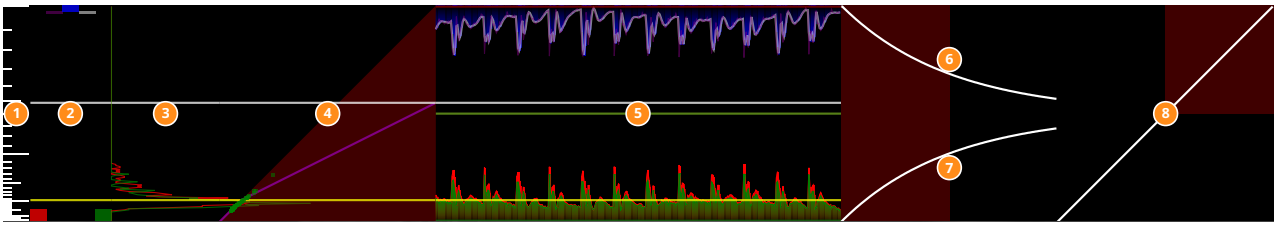


Figure 5 — The graph area, linear scale (the default setting), glass pane removed. The numbers refer to the sections of this chapter.

The graph area has eight displays, from left to right:

1. the graduated scale;
2. the present moment;
3. the level distribution;
4. the transfer curve;
5. the history;
6. the attack shape;
7. the release shape;
8. the waveshaper curve.

The vertical scale: linear or decibels

Displays 1 to 5 share the same vertical scale: the top corresponds to the maximum level (0 dBFS, or a gain of 1), the bottom to silence (or a gain of 0).

This scale can be **linear** or **in decibels**, as you prefer, using the **Gfx Lin/Db** knob on the main row.

- In **linear** (0, the default), height is proportional to amplitude. −6 dB sits at half height, −12 dB at a quarter, −20 dB at a tenth; all the low levels are squeezed together at the bottom.
- In **decibels** (1), height is proportional to the level in dB, over an adjustable range (**Gfx DB**, 60 dB by default): −20 dB sits one third of the way down from the top, −40 dB two thirds of the way down.
- In between, the scale is a blend of the two.

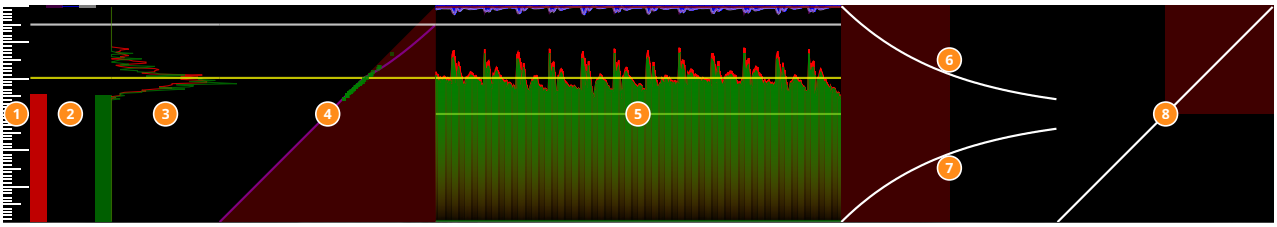


Figure 6 — The same scene as at the start of the chapter, on a decibel scale over 60 dB.

> **A word from the author**
>
> François prefers to read the curves in linear, where you really see the amplitudes. But most compressors display in decibels, and when you want to compare against them, the decibel scale is more practical.

Gfx Lin/Db

main row tab

Range: 0 to 1 **Default:** 0 **ID:** `decibels`

Scale of the graphs: 0 = linear, 1 = decibels, values in between = a blend. It applies to all the vertical displays, to the displayed transfer curve and to the scale markings.

The change applies only to new values: the part of the history that has already been drawn keeps the old scale until it has scrolled off.

Gfx DB

GRAPHS tab

Range: 1 to 60 dB **Default:** 60 dB **ID:** `graph_db`

Range of the decibel scale: the bottom of the graph corresponds to −Gfx DB. Lowering this value amounts to zooming in on the top of the dynamic range (for example 30 dB). This setting only has an effect if **Gfx Lin/Db** is greater than 0.

> **♪ In the studio: linear for peaks, decibels for nuances**
>
> The linear scale gives all the room to the top of the dynamic range: drum peaks and the attacks of an acoustic guitar stand out clearly, and you can see right away what the compressor is doing to them. The decibel scale, closer to the way the ear judges differences in level, makes the low levels visible: the ends of a singer's phrases, reverb tails, breaths, ghost notes on a snare. To set up a vocal or a bus, try **Gfx Lin/Db** at 1; to set the attack on percussion, go back to 0. The 80 included presets are all saved with the linear scale, the one François prefers: loading one switches the display back to linear.

1. The graduated scale

At the far left, a ruler gives the level markers: one tick per decibel, over the range of **Gfx DB**. The long ticks mark tens of decibels, the medium ones multiples of 5, the short ones each decibel. On the linear scale, the ticks bunch up toward the bottom; on the dB scale, they are evenly spaced.

2. The present moment

Five narrow columns show the most recent values:

Column	Quantity	Color	Drawn as
1	input level	red	bar rising from the bottom
2	gain requested by the transfer curve	purple	horizontal line
3	gain reduction (difference between output and input)	blue	bar coming down from the top
4	gain smoothed by the envelope follower	white	horizontal line
5	output level	green	bar rising from the bottom

> **From the needle to the history**
>
> On vintage hardware, gain reduction was read off the needle of a VU meter. In the LA-2A, the T4 optical module contains two photoresistive cells lit by the same luminous panel: one sets the gain, the other drives the VU meter in gain reduction mode. On the 1176, "all buttons" mode is famous for sending the needle into a frenzy. These five columns play the role of the needles of old, and the history, right next to them, also keeps a record of the last few seconds: you no longer see only how much the compressor is reducing the gain, but when, and how.

3. The level distribution

This display is a histogram turned on its side: for each height, the length of the line toward the right shows how much time the signal has spent at that level over the last few seconds. The **red** line is for the input, the **green** line for the output. Heavy compression tightens the green distribution toward the top, or shifts it downward, compared with the red one.

Gfx Cum Tmp GRAPHS tab

Range: 1 to 3720 **Default:** 600 **ID:** `graph_cumtmp`

Duration taken into account, in screen refreshes (60 per second): 600, the default, corresponds to about 10 s, and the maximum (3720) to about a minute. This setting also sets the length of the trail of dots on the transfer curve: one dot per refresh, over a tenth of this duration (60 dots, or about one second, by default).

Gfx Cum Amp GRAPHS tab

Range: 1 to 16 **Default:** 4 **ID:** `graph_cumamp`

Length added to the line, in pixels, each time the corresponding level occurs: this is the horizontal zoom of the distribution.

> **♪ In the studio: seeing density**
>
> The level distribution is a portrait of density. On a mix or a master, let a whole chorus play with **Gfx Cum Tmp** at maximum (about a minute) and compare the two lines: the harder the compression and limiting work, the more the green distribution tightens compared with the red. A green distribution shrunk to a narrow band is the signature of a very dense sound; if that is what you are after, great, and if not, it is time to raise the threshold. For teachers, it is also the most direct way to show what "reducing the dynamic range" means.

4. The transfer curve

The horizontal axis is the detected level and the vertical axis the output level, on the same scale. It shows:

- a **dark red triangle** below the diagonal, marking the zone where the output is lower than the input;
- the **transfer curve**, in **purple**: the output the curve would impose if the gain were applied instantly;
- a **trail of green dots**: the actual operating point over the last second. Horizontally, the level of the signal entering the compressor (the absolute value of the detection signal, before the settings of the DET (in) tab); vertically, that level multiplied by the smoothed gain. The dots fall exactly on the purple curve only if the attack and release are instantaneous; otherwise, they stray from it, and that gap shows the envelope follower at work;
- **orange dots**, when the bias is active (chapter 19).

The curve is **computed** in the domain chosen with **Cv Lin/Db** (linear or decibels) and **displayed** in the domain chosen with **Gfx Lin/Db**. So a curve computed linearly and displayed in decibels is not a straight line above the threshold: it bends. This is not a display error; it is exactly what the compressor does (chapter 10).

> **♪ In the studio: where your sound sits on the curve**
>
> The trail of green dots tells you where your sound really sits on the curve. On a vocal, set the **Threshold** so that the dots go past the bend of the curve only on the loud syllables: the rest of the phrase passes through untouched, and the vocal keeps its expression. If all the dots stay past the bend, the compressor is working all the time: that can suit a bus you want held very tight, but much less a vocal. And when the dots stray clearly from the purple curve, it means the attack and release are taking their time: that is often where a compressor's character is decided.

The reference lines

In displays 2 to 5, two horizontal lines serve as references:

- the **yellow** line is the **threshold** level;
- the **white** line is the output level that, according to the curve, an input signal at 0 dBFS reaches. It gives you an idea of the maximum reduction.

With the default settings (threshold −20 dB, ratio 2:1, linear curve), the yellow line is at 10% of the height and the white one at half height.

Gfx Lines GRAPHS tab

Range: 0 to 2 **Default:** 2 **ID:** `graph_lines`

0: no reference lines; 1: threshold only; 2: threshold and ratio.

5. The history

The history is the main display. It shows how all of the compressor's quantities have evolved over the last few seconds; time flows from right (most recent) to left. Each column of pixels is an average over a short duration, set by **Gfx Secs**; by default, the 375 columns cover 6.25 s.

The curves, in the order in which they are drawn:

Curve	Color	Drawn as
input level	red	bar from the bottom
peak, RMS and envelope detectors	semi-transparent red	thin lines
final detection	red	thin line
output level	green	bar from the bottom, and line
gain reduction (output relative to input)	semi-transparent blue	bar from the top
gain smoothed by the envelope follower	semi-transparent white	thick line
automatic release factor	pink	thick line
Mu factor	brown	thick line

Curve	Color	Drawn as
opto envelope	sky blue	thick line
bias detectors	semi-transparent orange	thin lines
final bias	orange	thin line
Out->In compensation	yellow-green	thick line
gain requested by the transfer curve	semi-transparent purple	thin line
sidechain analysis	gold	bar from the bottom

Then come the reference lines (threshold in yellow, ratio in white) and two lines that are always shown: **GR Min** in dark red and **GR Max** in dark green (CURVE tab, chapter 10). With the default settings, they hug the top and bottom edges.

A quantity whose module is inactive is 0: its line stays flat along the bottom of the graph.

Reading the history

- **Red above green**: the output is quieter than the input, so the compressor is reducing the level. The **blue** bar, coming down from the top, measures that reduction.
- The **purple** line is the gain the transfer curve asks for, sample after sample. The **white** line is the gain actually applied: it tries to follow the purple one, at the speed set by the attack and release. With short times, the two almost coincide; with long times, the white one lags behind and is smoothed out.
- With makeup, the green can rise back above the red: the blue bar disappears.
- The **yellow-green** line is the automatic Out->In compensation: at half height, it equals 0 dB (chapter 8).

> **Averages and peaks**
>
> Each column of the history is an average of absolute values over the duration of one column: a sine wave at 0 dBFS shows up around −3.9 dB, not at 0 dB. Short transients are therefore attenuated on the display.

Gfx Secs

main row tab

Range: 0 to 1 s **Default:** 0.016667 s **ID:** `samples`

Duration represented by one column of the history, and therefore the scrolling speed. The default value, 1/60 s, scrolls one column per screen refresh and covers 6.25 s. A larger value slows down the scrolling and smooths the curves; a smaller value lets you observe the attack and release in detail. Displays 2 and 3 also read the last column of the history, so they follow this setting too.

A word from the author

François considers **Gfx Secs** a very important setting: to see how the envelope behaves on a transient, you have to "slow down" the graph enormously, the way you would play a film in slow motion. With a very small value, each instant takes up many more columns, and the transient spreads out across the screen.

♪ In the studio: watching a snare drum

On a drum loop, set **Gfx Secs** to around 0.003 s: the history now covers only a little over a second, and each snare hit spreads across dozens of columns. With **Attack** at 1 ms, the white line dips almost at the same time as the purple one: the top of the hit gets knocked down, and the snare flattens out. Go to 20 or 30 ms: the white line comes down later, the green bar keeps its peak at the start of each hit, and the crack comes back. Then work the **Release**: if the white line doesn't have time to come back up before the next hit, the compression never lets go and the drums lose their life; if it comes back up in rhythm, you will often hear the drums breathe with the tempo. In fact, 71 of the 80 included presets were saved with a **Gfx Secs** shorter than the default, 59 of them between 2 and 5 ms: it is the setting that lets you watch attacks, so loading one of these presets also changes the time scale of the history.

Gfx Detects

GRAPHS tab

Range: −2 to 2 **Default:** 0 **ID:** `graph_detects`

Chooses which detection curves are shown in the history:

- **−2**: no detection curve;
- **−1**: the final detection only;
- **0** (default): the three detectors (peak, RMS, envelope) and the final detection, along with all the other curves;
- **1**: the three detectors and the final detection, **alone**: all the other curves are hidden;
- **2**: the final detection **alone**.

The positive values are "exclusive" modes, very handy for setting up the detection (chapter 13).

Gfx Detects Bias

GRAPHS tab

Range: −2 to 2 **Default:** 0 **ID:** `graph_detects_bias`

Same logic for the bias curves: −2 also hides the orange dots on the transfer curve, and 1 and 2 hide all the other curves (chapter 20).

Gfx Histos

GRAPHS tab

Choices: ____ · ___g · __o_ · __og · _u__ · _u_g · _uo_ · _uog · a___ · a__g · a_o_ · a_og · au__ · au_g · auo_ · auog **Default:** auog **ID:** `graph_gains`

Chooses which of four optional curves are shown. Each menu choice is a four-letter word, where a letter means “shown” and an underscore means “hidden”:

- **a**: automatic release factor (A/R), pink;
- **u**: M**u** factor, brown;
- **o**: **o**pto envelope, sky blue;
- **g**: **g**ain requested by the transfer curve, purple.

By default, “auog”: all four are shown. For example, “a_o_” shows only the A/R factor and the opto.

Gfx Make Up

GRAPHS tab

Range: 0 to 1 **Default:** 1 **ID:** `graph_make`

Display of the yellow-green Out->In compensation line: 0 = never, 0.5 = only when Out->In is active (Out->In Secs below 42), 1 = always.

The gradients

The bars of the history (input, output, analysis) are solid at the top and fade out toward the bottom; the blue reduction bar fades out toward the top. Four settings position these gradients, in pixels from the top of the graph (0 to 199), separately for the linear scale and the decibel scale (the two are blended according to **Gfx Lin/Db**).

Gfx Floor Lin

GRAPHS tab

Range: 0 to 199 **Default:** 150 **ID:** `graph_degrade_sol_lin`

Height from which the rising bars start to fade, on the linear scale. At 150 (default), they are solid up to 150 pixels and fade out over the last 50.

Gfx Ceil Lin

GRAPHS tab

Range: 0 to 199 **Default:** 50 **ID:** `graph_degrade_plafond_lin`

Height at which the descending blue bar becomes solid, on the linear scale. At 50 (default), it is transparent at the top and solid from 50 pixels on.

Gfx Floor Db GRAPHS tab

Range: 0 to 199 **Default:** 0 **ID:** `graph_degrade_sol_db`

Same as **Gfx Floor Lin**, for the decibel scale. At 0 (default), the bars fade over their full height.

Gfx Ceil Db GRAPHS tab

Range: 0 to 199 **Default:** 0 **ID:** `graph_degrade_plafond_db`

Same as **Gfx Ceil Lin**, for the decibel scale. At 0 (default), the blue bar is uniform.

Gfx Photo GRAPHS tab

Range: 0 to 1 s **Default:** 0 s **ID:** `graph_obturateur`

Makes the history advance **in jumps** rather than continuously: at 0 (default), the history scrolls at every refresh; with a value v, it stays frozen and only updates about once every v seconds, adding all the accumulated columns at once. This “snapshot” effect lets you examine a portion of the history at your leisure. The other graphs are not affected.

6 and 7. The attack and release shapes

These two small graphs show the **shape** of the attack curves (top) and release curves (bottom): how the envelope reaches its target over time. The horizontal axis covers twice the set time; the left half, tinted red, corresponds to the set time itself.

Four curves are superimposed, one per envelope follower: **white** for the follower after the curve (A/R (cv)), **red** for the detection envelope (A/R (in)), **orange** for the bias (A/R (bs)), **lavender** for the lookahead (A/R (la)). With the default settings, they are identical and only the white one, drawn last, is visible.

When the shape of an active follower is modified (**Shape** setting), a **vertical line** in the corresponding color marks the current position on the curve. Chapter 12 explains how to interpret it.

8. The waveshaper curve

The last graph shows the transfer function of the waveshaper (chapter 17): the input from −1 to +1 on the horizontal axis, the output on the vertical axis. The upper right quadrant, tinted red, corresponds to the positive values. With the default settings, the waveshaper is neutral and the curve is a diagonal line.

The glass pane

Gfx Windows GRAPHS tab

Range: 0 to 1 **Default:** 1 **ID:** `graph_fenetre`

Shows (1) or removes (0) the semi-transparent glass pane that covers the graph area. It is purely decorative; removing it makes the curves easier to read.

> **The graphs and the lookahead**
> With an active lookahead, the graphs always display the “present” chain, that is, the signal you hear.

Starting ideas

A few display settings, depending on what you want to observe:

- **An attack** (snare, kick, bass played with the thumb, acoustic guitar): **Gfx Secs** between 0.002 and 0.004 s, **Gfx Lin/Db** at 0, and **Gfx Detects** at −1 to keep only the final detection and declutter the graph.
- **A vocal or a bus**: **Gfx Secs** at its default value (6.25 s of history), **Gfx Lin/Db** at 1, and both reference lines (**Gfx Lines** at 2) to locate the threshold and the maximum reduction.
- **A master**: **Gfx Secs** around 0.05 s, or nearly 20 s of history, to follow the release over several bars; **Gfx Cum Tmp** at maximum for the distribution; **Gfx Make Up** at 1 to follow the Out->In compensation (chapter 8).
- **A precise moment**: **Gfx Photo** around 1 s freezes the history between two updates, long enough for you to examine it.
- **The detection**: **Gfx Detects** at 1 or 2 isolates the detection curves, so you can compare peak, RMS and envelope (chapter 13).

CHAPTER 5

The MAIN Tab and the Presets

The **MAIN** tab is the one shown when you open the plugin. It brings together preset management and the global settings, which act on the whole chain.

It is also the gateway to the 80 presets that ship with Comprendre. Each one is named after a commercial plugin that François imitated by comparing gain curves with the analysis mode of chapter 24: optical compressors, FETs, vari-mus, bus compressors, mastering tools, purely digital designs, and even saturators. For the composer in a hurry, it is a ready-to-use palette of characters. For anyone who wants to learn, it is a library of eighty ways to build a compressor, all of them easy to read thanks to the yellow highlighting (chapter 3). And every preset you save yourself, for a vocal, a drum bus or a master, will add to that library. Chapter 25 presents the included presets, family by family.



Figure 7 — The MAIN tab.

Setting	Role	Chapter
Monitor Delta	listen to the difference between output and input	7
Input Db	input gain	7
Ch.0 Db, Ch.1 Db	per-channel gains (external headroom)	7
Sidechain	share of the sidechain in the detection	14
Load	load a preset	this chapter
Analyse SC	display the gain of a compressor placed in the sidechain	24
Oversampling	oversampling	7
X Filters Taps, window, beta	FIR band filters for the detection and the target	14
Save	save a preset	this chapter

Setting	Role	Chapter
Phase	output polarity	7
Pre Clip Mode	pre-clipper mode	7
Headroom	internal headroom	7
Post Clip Mode	post-clipper mode	7
L/R to M/S	left/right or Mid/Side processing	7
Output Db	output gain	7
Snap Out->In (below Output Db)	transfer the Out->In compensation into Output Db	8

Saving and loading a preset

The **Save** button opens a file save dialog, and the **Load** button a file open dialog. Presets are files with the **.preset** extension. If the file you choose already exists, you are asked to confirm before it is replaced.

By default, these dialogs open in a **Comprendre** folder, created automatically in the user's application data folder:

System	Preset folder
Windows	`C:\Users\<name>\AppData\Roaming\Comprendre`
macOS	`~/Library/Comprendre`
Linux	`~/.config/Comprendre`

You can save your presets elsewhere: the Comprendre folder is only a suggestion.

What a preset contains

A preset contains the value of **all** the plugin's parameters, including the display settings (GRAPHS tab, **Gfx Lin/Db**, **Gfx Secs**). So loading a preset also changes how the graphs look.

It is an XML text file: a root element `<Parametres>` and one `<PARAM id="…" value="…"/>` line per setting, where the identifier is the one given on the parameter cards in this manual. You can therefore open it in a text editor to check a value.

> **Incomplete presets**
> If a preset file is missing some settings (for example, a file edited by hand), those settings keep their current value when it is loaded; they are not reset to their default value.

After loading, the names of the modified settings and tabs turn yellow (chapter 3): you can see at a glance what the preset changes compared with the default settings.

♪ In the studio: starting from a preset

A preset was built on a particular excerpt, by imitating a plugin set up a certain way. Your track is not at the same level, so almost always start by adjusting the **Threshold** until you get the amount of compression you are aiming for, before you judge the character. Then adjust the **Release** to the tempo. The included presets don't make up on their own for the level they take away or add: to compare them with each other, let the **Out->In** auto-volume work at one or two seconds (chapter 8), otherwise the loudest one will always seem the best. Finally, load your "default" preset before trying another one: since presets don't always contain every setting, that is the way to hear each one exactly as it was saved.

♪ In the studio: two readings of the same machine

Compare **slate_fg2a** and **waves_la2a**, imitations of two plugins that emulate the same optical compressor, the LA-2A. The first one fits in four settings: **Attack** at 7.5 ms, **Knee** at 12 dB, a barely negative **Fet even** (−0.032) in the WS tab and, above all, an **Env Offset** of −0.55 in the GR (cv) tab. This envelope offset is, as François says in his walkthrough in French, the setting he uses most to imitate compressors, and he finds its effect striking on the LA-2A. The second one uses thirteen, including a decibel curve, a 25 dB knee, an automatic release blended at half, a **Headroom** of −6 dB, half-unlinked channels and a high band of the dynamic EQ, at 7 kHz. Two plugins, two readings of the same machine, and two paths in Comprendre to get close to them: load them one after the other on the same vocal and listen to what sets them apart.

The settings in your project

Independently of presets, your host saves the complete state of Comprendre with your project (in binary form), and restores it when you reopen the project.

A 'default' preset

Comprendre has no reset button. Save a preset with all the default settings the first time you use it: reloading it will take you back to the initial state.

Measuring a machine, setting by setting

Developers who model a unit "by measurement" record its input and output for many levels, sources and settings, without rebuilding its components. The method has a limitation, summed up by Arvid Rosén of Softube: with twenty knobs, the number of combinations to measure becomes astronomical. Even at Universal Audio, which mostly practices circuit modeling, they often measure several restored units of a machine to derive a "reference unit." Comprendre's presets belong to the measurement approach, on a musician's scale: each one imitates a plugin at **one** setting of that plugin, the one François chose for the analysis. That is why a preset is a starting point, and not the whole machine.

Demo version and registration

Until the plugin has been registered, it runs in **demo mode**. The sound is not limited in any way: no added noise, no dropouts, no time limit. Only two features are missing:

- the **Save** button is replaced by a **DEMO** button, written in red: you can't save presets (loading them is still possible);
- the plugin's state is **not** saved in your project: when you reopen it, Comprendre goes back to the default settings.

The **DEMO** button opens the **Registration** window. Enter the email address and password of your account, then click **REGISTER**. The plugin contacts the Skynext server; if registration succeeds, the message "Registration Successful !" appears in green. The registration is stored on the computer, in the Comprendre folder described above.

> **Reopen the plugin**
>
> The registration only takes effect the **next time** the plugin is opened: remove Comprendre from your track and put it back, or reload your project. The **Save** button then appears in place of **DEMO**.

Registration is tied to the computer: on another machine, you need to register the plugin again.

CHAPTER 6

Overview of the Signal Chain

In a studio, the sound of a compressor doesn't depend on the compressor alone. It depends on what you patch in before and after it, on how hard you drive it, on what you send to its detector. Comprendre contains all that wiring: input gain, a Mid/Side matrix, a clipper, a compressor whose detection can listen to another track or to a single frequency band, three color modules inspired by the great families of hardware, a parallel blend and an output clipper. It is a small mixing and mastering studio, wired in a precise order, that you can walk through from input to output. Knowing that order means knowing where to act to get the sound you are after.

Before going into the details of the settings, it helps to have in mind the complete path the sound follows through Comprendre. Most of the surprises you may run into with the plugin come down to the order of operations: a gain applied before or after a clipper, a measurement taken before or after a filter, an envelope placed before or after the transfer curve.

Sample-by-sample processing

Comprendre processes sound **sample by sample**. Each sample goes through the entire chain before the next one comes in; there is no block processing and no windowed analysis. The compressor can therefore react with single-sample precision.

When **oversampling** is enabled (**MAIN** tab, **Oversampling** menu), the sound is first converted to a higher rate (×2 to ×16), and the **entire** chain runs at that rate: the detection, the envelopes, the saturations, the clippers, the auto-volumes and even the graphs. Times expressed in seconds (attack, release, auto-volume constants…) are recalculated for the internal rate, so they stay accurate whatever factor you choose.

The complete path

The following figure shows the exact order of operations, as programmed. Read it column by column, from top to bottom.

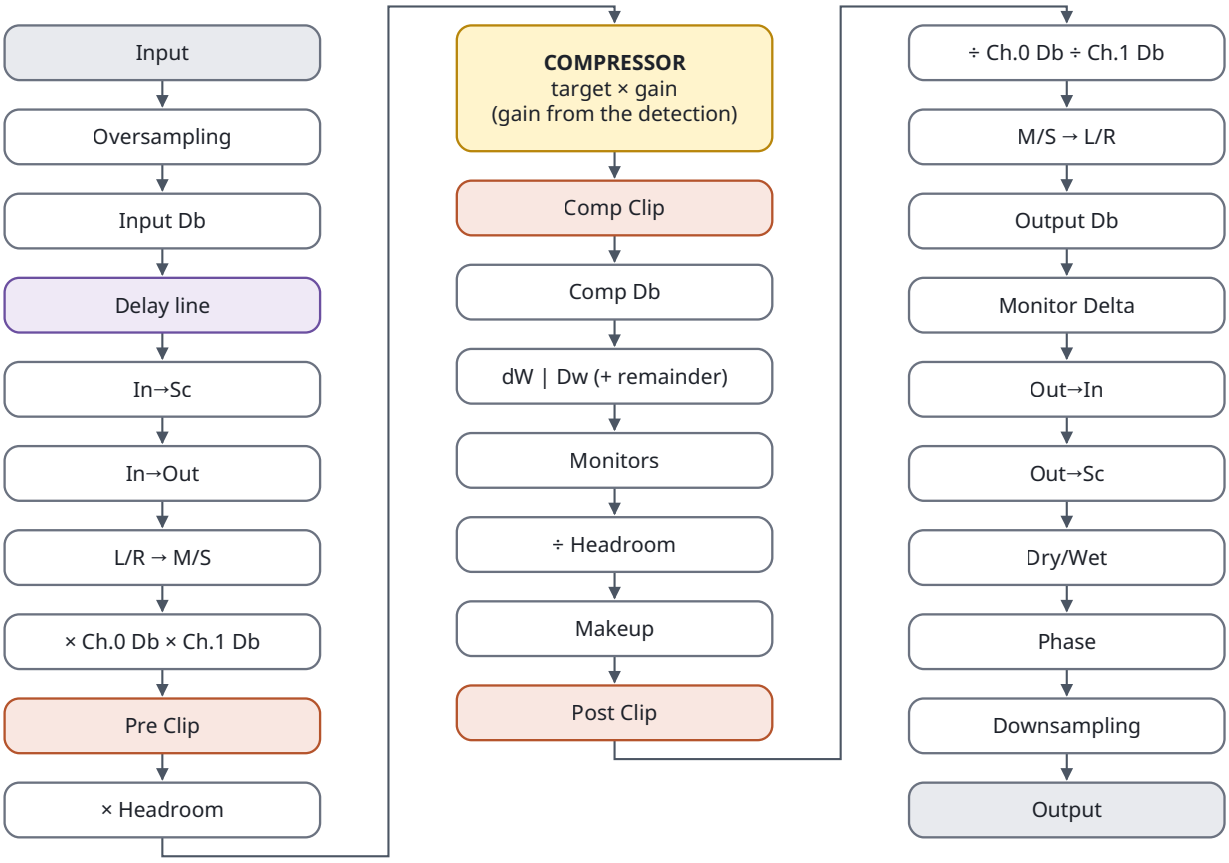


Figure 8 — The complete audible chain, from input to output.

It can be divided into three zones:

1. **Before the compressor**: input gain, input auto-volumes, optional Mid/Side conversion, per-channel gains, pre-clipper and headroom. These stages prepare the signal; they are described in chapter 7, and the auto-volumes in chapter 8.
2. **The compressor** itself, which computes a gain from a *detection* branch and applies it to a *target* branch.
3. **After the compressor**: clipper on the compressed signal, blends, listening monitors, makeup, post-clipper, output gain, output auto-volumes, dry/wet and polarity inversion.

> **The delay line**
>
> Right after the input gain, each sample is written into a delay line. This is what lets the lookahead "see the future": the audible chain reads the signal with a delay, while the look-ahead detection reads the more recent samples. The delay line is also used to align the signal with the delay of the FIR filters (chapter 14). Without lookahead or FIR filters, the delay is zero.

> **♪ In the studio: you can hear the order**
>
> The same setting, placed at two different points in the chain, doesn't sound the same. Try it on a drum loop that peaks near 0 dBFS. A **Pre Clip** at −3 dB shaves off the peaks **before** the compressor: it receives transients that are already flattened and reacts less to each hit. The same threshold on the **Post Clip**, on the other hand, lets the compressor react to the full peaks, then catches whatever still pokes out after the makeup. Listen to the difference on the snare attacks and on how the whole kit breathes. Same logic for gain: raising **Input Db** makes everything work harder, all the way to the output level; raising the **Headroom** makes only the inside of the chain work harder, compressor and saturations included, then gives the level back (chapter 7).

What is measured, what is compressed

In an ordinary compressor, the signal being measured and the signal being attenuated are one and the same. Comprendre splits them into two branches that can be set independently.

- The **detection branch** (settings in the **INPUT** tab) builds the signal the compressor will *measure*. You can mix in the sidechain, feed the output back into it (feedback), filter it, change its stereo image, and send the dynamic EQs and the waveshaper into it without hearing them.
- The **target branch** (settings in the **OUTPUT** tab) determines the part of the signal that will be *attenuated*. Filters can keep just one frequency band or one part of the stereo image: only this **target** receives the compressor's gain, while the **remainder** of the signal goes around it without being compressed, and then the two are summed.

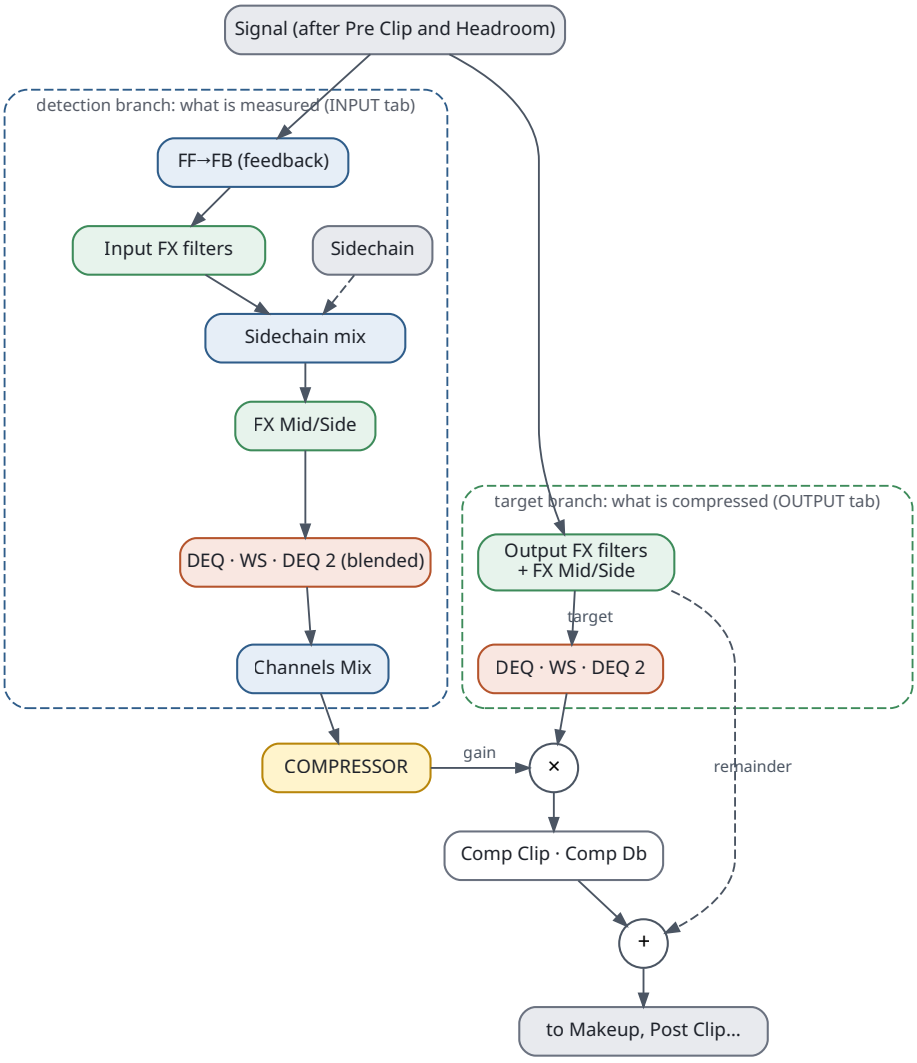


Figure 9 — The two branches of the compressor.

By default, no filter is engaged: the detection measures the input signal as is and the target is the whole signal (the remainder is zero). Comprendre then behaves like a classic compressor. Chapter 14 covers both branches in detail.

> **♪ In the studio: listen here, act there**
>
> Separating what the compressor listens to from what it processes opens up moves you would usually hand off to specialized units. A bass that ducks when the kick hits: the kick comes in through the sidechain, and the bass is the target. A de-esser: the detection listens only to the sibilants, and the compression touches only the top end of the vocal. A mix whose low end no longer makes everything else pump: you remove the low end from the detection, without removing it from the sound. In his video on input and output selection, François goes as far as making the highs compress according to what is in the low end, and he presents this separation between detection and target as one of Comprendre's distinctive features. The settings for these examples are given in chapter 14, in the "Example settings" section.

Inside the compressor

The heart of the compressor is one single function, which the code calls the **factory** (*usine*). It receives the absolute value of the detection signal and produces a gain, always going through the same steps:

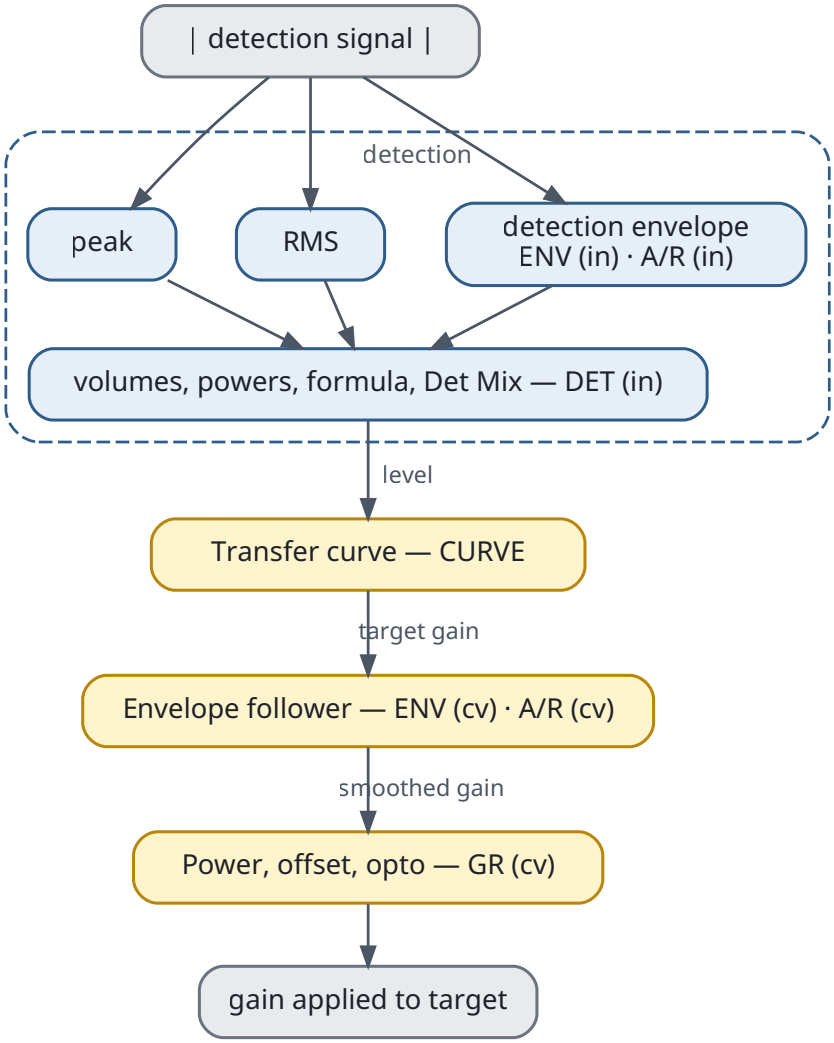


Figure 10 — The steps of the gain calculation.

1. **Detection** (**DET (in)** tab) turns the signal into a *level*. It combines three measurements: the instantaneous peak value, a root-mean-square (RMS) average and a detection envelope, which has its own settings in the **ENV (in)** and **A/R (in)** tabs (chapter 13).
2. **The transfer curve** (the Threshold, Ratio and Knee knobs on the main row, and the **CURVE** tab) turns that level into a *target gain*: the gain that would have to be applied to respect the ratio (chapter 10).
3. **The envelope follower** (**ENV (cv)** and **A/R (cv)** tabs) chases that target gain at a certain speed: this is what the Attack and Release knobs on the main row control (chapters 11 and 12).
4. **The gain reduction options** (**GR (cv)** tab) reshape this gain further: power, offset, opto designer (chapter 15).

The final gain multiplies the target.

> **Two envelopes in one compressor**
>
> So there can be two envelope followers in the same compressor: one **before** the transfer curve, in the detection ("(in)" tabs), and the other **after** it ("(cv)" tabs). By default, only the second one is active. Academic papers tend to recommend the position after the curve, but many commercial compressors smooth the signal before it; Comprendre offers both, and any blend of the two.

> **Several stages in one box**
>
> Chaining several processes in a single unit is not a new idea. The Focusrite Red 3, in the mid-1990s, generates separate compression and limiting voltages in its side-chain, which drive the same VCA: the result, according to its manufacturer, is "true compression followed by limiting," and not compression that turns into limiting. The Shadow Hills Mastering Compressor places two compressors in series in each channel, an optical stage followed by a discrete VCA-type stage. The Bettermaker Mastering Limiter (2017) follows its limiter with a two-stage analog clipper, for the transients that get past it. In Comprendre, the pre-clipper, the compressor, the opto designer, the compressor's clipper and the post-clipper form a chain of the same kind, in which you choose the active stages.

The tab names

The 25 tabs are arranged from left to right, roughly in the order of the signal path. Each name indicates the module and, in parentheses, where it acts:

Suffix	Meaning
(in)	in the detection, **before** the transfer curve
(cv)	**after** the transfer curve (cv for *curve*)
(gr)	on the gain reduction (gr for *gain reduction*), that is, the opto
(la)	for the lookahead
(ws)	for the waveshaper
(bs)	for the bias (Vari-Mu designer)

The **ENV** tabs always contain the same 14 envelope follower settings, and the **A**/**R** tabs the same 14 attack and release settings: only the follower they apply to changes. Likewise, the three **F** tabs each set the processing of a "saturation product."

Tab	Role	Chapter
MAIN	presets, oversampling, FIR filters, input and output gains, clipper modes, headroom, M/S, sidechain, analysis, polarity	5, 7
INPUT	input auto-volumes, feedback, detection filters and Mid/Side, sends from the dynamic EQs and the waveshaper to the detection, Channels Mix, detection monitoring	8, 14
DET (in)	detection: peak, RMS, envelope, formulas, blend	13

Tab	Role	Chapter
ENV (in), A/R (in)	detection envelope follower	11, 12, 13
CURVE	transfer curve, gate, reduction limits	10
ENV (cv), A/R (cv)	follower after the curve (the “classic” follower)	11, 12
GR (cv)	power, offset, opto designer	15
F (gr)	saturation product of the opto	15, 16
MOD & LA	modulations and lookahead	22, 23
ENV (la), A/R (la)	lookahead envelope follower	22
DEQ	first dynamic EQ (before the waveshaper)	18
WS	waveshaper (FET designer)	17
F (ws)	saturation product of the waveshaper	16, 17
DEQ (second)	second dynamic EQ (after the waveshaper)	18
BIAS src, DET (bs), ENV (bs), A/R (bs)	bias source	19, 20
BIAS dst, F (bs)	bias destination and its saturation product	21
OUTPUT	target, target clipper, blends, monitoring, Out→Sc	7, 8, 14
GRAPHS	display settings	4

> **Two ‘DEQ’ tabs**
>
> The tab bar shows the name **DEQ** twice. The first one, to the left of **WS**, is the dynamic EQ placed before the waveshaper; the second one, to the right of **F (ws)**, is the one placed after it. This manual calls them **DEQ** and **DEQ 2**.

What Comprendre does with its default settings

When you load it, Comprendre is a deliberately simple compressor:

- the detection is the **instantaneous peak value** of the signal: no RMS, no detection envelope, no smoothing;
- the transfer curve is computed **linearly**, with a threshold of −20 dB, a ratio of 2:1 and a zero knee;
- the envelope follower is **branching** and **linear**, with a 10 ms attack and release in **logarithmic** (exponential) mode;
- the channels are **linked**: a single gain for both channels, calculated from the average of their absolute values (**Channels Mix** set to *Avg*, chapter 14);
- no clipper, no auto-volume, no saturation, no filter.

All the additional modules are neutral until you set them, and several of them (bias, lookahead, detection envelope) aren’t even computed as long as they stay at their neutral setting.

> **♪ In the studio: starting from the simplest**
>
> With its default settings, Comprendre is a clean, uncolored compressor whose entire character comes from the threshold, the ratio and the times, as on VCA compressors (chapter 2). In his first video, François describes it as very clean and based on the simplest algorithm. It is the best starting point for building a sound: first set the threshold, ratio and times on your source, then add **one** color at a time (the opto designer, the waveshaper, the bias, a dynamic EQ), comparing level-matched (chapter 8). That way you will know what each module brings to your vocal, your bass or your bus, and you will avoid stacking colors that mask one another.

Future and present: the lookahead

When you set a lookahead (**MOD & LA** tab, negative **la delay** setting), Comprendre runs the compression chain **twice**: first on the "future" signal, undelayed, to find out what the detection and the envelope are about to do; then on the "present" signal, delayed, which is the one you hear. The future chain produces no sound: it only supplies its detection and its envelope, which the present chain uses according to the lookahead settings (chapter 22).

This duplication has two practical consequences: with an active lookahead, the CPU load roughly doubles, and the plugin reports latency to your host.

Latency

Comprendre reports to the host the sum of three delays:

- the **lookahead**, when **la delay** is negative;
- the delay of the **FIR filters**, equal to (X Filters Taps − 1) / 2 samples whenever this setting is not 0 (chapter 14);
- the delay inherent to **oversampling**, which depends on the factor and the filter type (IIR or FIR).

Without lookahead, FIR filters or oversampling, the latency is zero.

> **♪ In the studio: playing through Comprendre**
>
> If you record while monitoring through Comprendre (a singer on headphones, a bass plugged in direct), keep the latency at zero: no lookahead, no FIR filters, no oversampling. These three tools are made for mixing and mastering, where your host in principle compensates for the delay the plugin reports. For tracking, the main row, the opto designer, the waveshaper or the bias are enough to add character without adding latency.

> **Going further**
>
> When oversampling is active, the delay of the FIR filters is counted in internal samples: the plugin divides it by the oversampling factor and rounds it up to the next whole sample. Clippers in ADAA mode introduce a delay of a fraction of a sample, which is not reported.

CHAPTER 7

Input, Output, Clippers, and Headroom

This chapter covers everything that surrounds the compressor: the input and output gains, Mid/Side conversion, the per-channel gains, the three clippers, the two headrooms, makeup, channel linking, the dry/wet blends, and the listening monitors. These controls live on the main row and in the **MAIN** and **OUTPUT** tabs.

These are the mastering engineer's tools, and the tools of anyone who wants to stay in control of level. Here you'll find what you need to hit the compressor at the right level, clip peaks before limiting them (as François recommends), process the center and the sides of the stereo image separately, blend the compressed sound with the original for parallel compression, and even listen to what the compressor takes away. In a mastering room, these functions are spread across several units and a console; here, they surround the compressor in a precise order, detailed below.



Figure 11 — Input and output controls: main row and MAIN tab.

Their order in the chain is as follows (see also the figure in chapter 6):

1. oversampling;
2. **Input Db**;
3. input auto-volumes **In→Sc** and **In→Out** (chapter 8);
4. **L/R to M/S**;
5. **Ch.0 Db** and **Ch.1 Db**;
6. **Pre Clip**;
7. **Headroom**;
8. *the compressor*;
9. **Comp Clip**, **Comp Db**, **dW | Dw**;
10. monitors;
11. **Headroom** return;

12. **Makeup**;
13. **Post Clip**;
14. **Ch.0 Db** and **Ch.1 Db** return, Mid/Side back to left/right;
15. **Output Db**;
16. **Monitor Delta**;
17. output auto-volumes **Out→In** and **Out→Sc** (chapter 8);
18. **Dry/Wet**;
19. **Phase**.

> **0 dB means 'neutral'**
> All the gains in this chapter range from −24 to +24 dB. At 0 dB, the stage is simply bypassed. Likewise, the three clippers are disabled when their threshold is at 0 dB: there is no clipping at 0 dBFS by default.

Oversampling

Oversampling

MAIN tab

Choices: 1X · 2X IIR · 4X IIR · 8X IIR · 16X IIR · 2X FIR · 4X FIR · 8X FIR · 16X FIR **Default:** 1X **ID:** `oversampling`

Selects the oversampling factor (×2, ×4, ×8, ×16) and the type of conversion filters: **IIR** (polyphase half-band filters, low latency, nonlinear phase) or **FIR** (linear-phase equiripple half-band filters, higher latency). These are the oversampling filters from the JUCE library, set to maximum quality.

The whole Comprendre chain then runs at the oversampled rate, so the processing cost rises roughly in proportion to the chosen factor. Oversampling reduces the aliasing produced by the nonlinear stages: the clippers, the waveshaper, bias, and the compressor itself when its times are very short.

> **Oversampling and FIR filters**
> The size of the FIR band filters (**X Filters Taps**, chapter 14) is not adjusted automatically for oversampling. Since these filters run at the internal rate, you need to multiply the X Filters Taps value by the oversampling factor to keep the same selectivity: 65 at 1X corresponds roughly to 129 at 2X.

Input gain

Input Db

MAIN tab

Range: −24 to 24 dB **Default:** 0 dB **ID:** `entree_premier`

Gain applied to the signal as soon as it enters the plugin, before any other operation (right after oversampling). It affects everything downstream, including the red curve in the graphs (input level), the reference for the auto-volumes, and the dry part of the **Dry/Wet** blend.

It does **not** apply to the sidechain signal.

Left/right or Mid/Side

L/R to M/S

MAIN tab

Range: −1 to 1 **Default:** 0 **ID:** `lrms`

Selects the stereo representation the entire plugin works in. This knob has only three positions:

- **0**: left/right. Channel 0 is the left, channel 1 the right.
- **+1**: Mid/Side. Channel 0 becomes the Mid, M = (L + R) / 2, and channel 1 the Side, S = (L − R) / 2.
- **−1**: Side/Mid. The same signals with the channels swapped: channel 0 is the Side, channel 1 the Mid.

The inverse conversion takes place at the end of the chain, just before **Output Db**; it reconstructs left and right exactly.

This setting comes into its own as soon as any processing works channel by channel: the **Ch.0 Db** and **Ch.1 Db** gains, the clippers in *Unlink* or *Adaa* mode, unlinked compression (**Channels Unlink**), or a single channel chosen for detection (**Channels Mix**). For example, you can compress the Mid and the Side separately.

This setting has no effect on a mono signal.

> **The sidechain is not converted**
>
> The sidechain signal enters detection as is, in left/right. In Mid/Side mode with detection on the sidechain, channel 0 of the detector therefore mixes the Mid of the internal signal with the left channel of the sidechain.

> **From record cutting to mastering: Mid/Side**
>
> Mid/Side processing comes from record cutting. The Fairchild 670 could run as two independent limiters, or as a limiter of the "lateral and vertical" components, that is, of the sum and the difference of the channels. This **Lateral/Vertical** mode is used today for Mid/Side processing, and one mixing trick is to use it on a drum bus to compress the sides independently of the center. FM radio, for its part, transmits sum and difference, hence its mono compatibility. Mastering compressors later adopted the principle: the M/S modifications of the Manley Variable Mu, the built-in matrix of the elysia alpha, the MD3 algorithm of the TC Electronic System 6000, the Mid/Side mode of the Bettermaker Mastering Limiter.

Per-channel gains: the external headroom

Ch.0 Db

MAIN tab

Range: −24 to 24 dB **Default:** 0 dB **ID:** `lrms_a`

Gain applied to channel 0 (left, Mid, or Side depending on **L/R to M/S**), just before the pre-clipper. The inverse gain is applied at the end of the chain, after the post-clipper, so the output level is unchanged.

Ch.1 Db

MAIN tab

Range: −24 to 24 dB **Default:** 0 dB **ID:** `lrms_z`

Same thing for channel 1.

These two gains form what François calls the **external headroom**: they wrap around the entire chain, clippers included. Raising **Ch.0 Db** by 6 dB amounts to driving the pre-clipper, the compressor, the compressor clipper, and the post-clipper 6 dB harder on that channel, then coming back down 6 dB at the output. Unlike **Headroom** (below), they are set channel by channel, which makes them a tool for deliberate imbalance, especially in Mid/Side: pushing the Side into the clippers and the compressor without touching the Mid. They have no effect on a mono signal.

The clippers

Comprendre has three clippers:

Clipper	Threshold	Mode	Position
pre-clipper	**Pre Clip** (main row)	**Pre Clip Mode** (MAIN)	before the compressor
compressor clipper	**Comp Clip** (OUTPUT)	**Comp Clip Mode** (OUTPUT)	right after the compressor gain, on the target only
post-clipper	**Post Clip** (main row)	**Post Clip Mode** (MAIN)	after the makeup

The pre-clipper reflects a common practice: lightly clipping a signal before limiting it. Comprendre can work as a limiter (high ratio, zero attack), and it has this pre-clipper built in.

The modes

The mode menus offer four choices (three for **Comp Clip Mode**, which has no *Linked* mode):

- **Linked**: the loudest channel determines the clipping. If the largest absolute value across the channels exceeds the threshold, **all** channels are multiplied by the same ratio, threshold / value. The loudest channel is clipped exactly at the threshold and the others are lowered in the same proportion: the stereo image is preserved, but the quieter channel is subjected to the gain changes imposed by the other.
- **Unlink**: each channel is clipped on its own, with hard clipping (the signal is clamped between −threshold and +threshold).
- **Adaa 1**: hard clipping per channel, computed with first-order *antiderivative anti-aliasing*. Instead of clipping each sample, the clipper computes the average of the clipping function between two successive samples, using its antiderivative. Aliasing is greatly reduced, at the cost of a very slight softening of the highs.
- **Adaa 2**: the same technique at second order: even less aliasing, and slightly more attenuated highs.

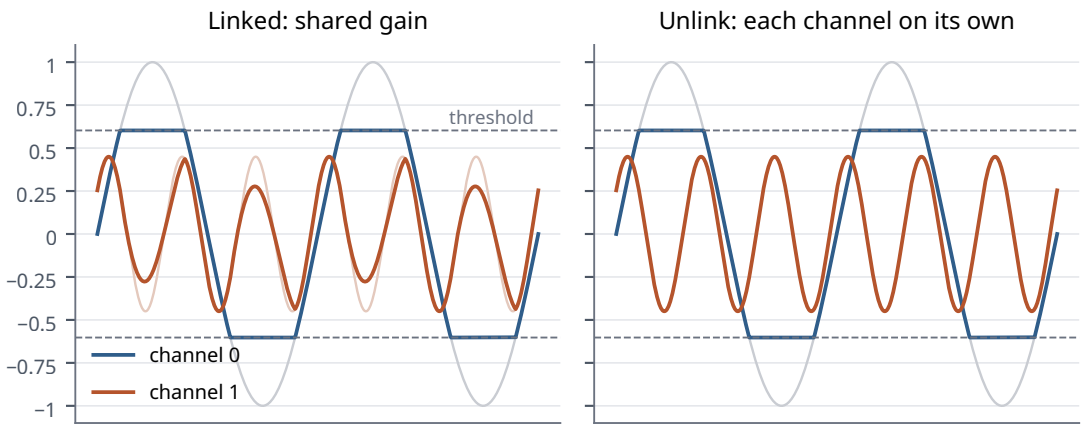


Figure 12 — Pre Clip in Linked mode and in Unlink mode: two channels with different amplitudes, threshold at −4.4 dB.

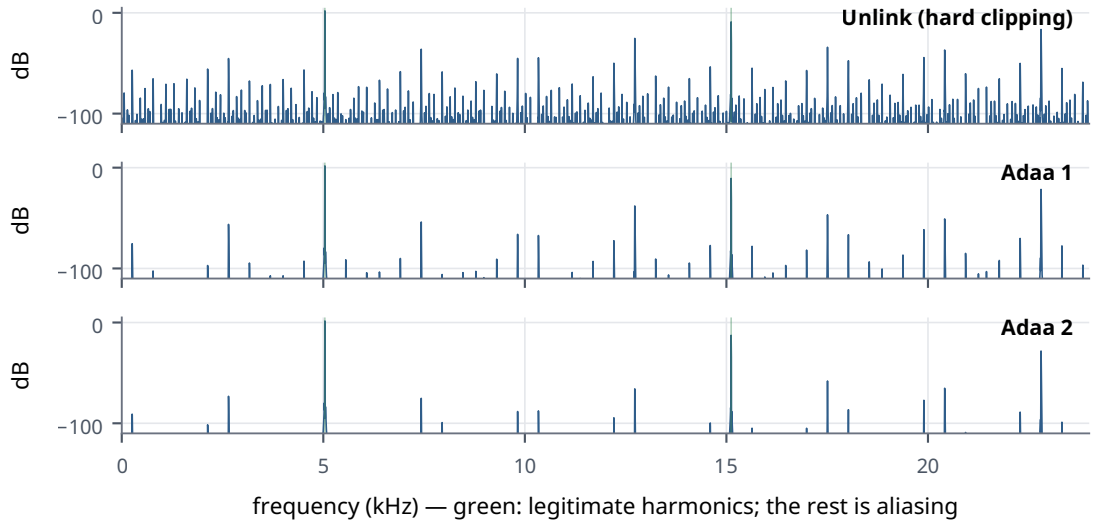


Figure 13 — Spectrum of a 5 kHz sine wave clipped by 9.5 dB, at 48 kHz: hard clipping, Adaa 1, and Adaa 2. Every line outside the odd harmonics (in green) is aliasing.

A word from the author

François describes ADAA as “poor man’s oversampling”: an inexpensive way to make clipping cleaner, comparable in effect to a one- or two-pole filter. Adaa 2 is the cleaner of the two, but it can sound a little duller.

Going further

First-order ADAA replaces f(x[n]) with $(F_1(x[n]) - F_1(x[n-1])) / (x[n] - x[n-1])$, where F_1 is the antiderivative of the clipping function; the second order uses the second antiderivative F_2 over three samples. When two successive samples are nearly equal, the computation falls back to the function itself. These two methods delay the signal by about half a sample (order 1) and by one sample (order 2); this delay is not compensated.

Pre Clip

main row tab

Range: −24 to 0 dB **Default:** 0 dB **ID:** `sol`

Pre-clipper threshold. At 0 dB, the pre-clipper is disabled. The threshold applies to the signal after **Input Db**, the input auto-volumes, the Mid/Side conversion, and the **Ch.0 Db**/**Ch.1 Db** gains.

Pre Clip Mode

MAIN tab

Choices: Linked · Unlink · Adaa 1 · Adaa 2 **Default:** Linked **ID:** `sol_adaa`

Pre-clipper mode: *Linked*, *Unlink*, *Adaa 1*, or *Adaa 2* (see above).

♪ In the studio: clip before you limit

François makes the point in his videos: when you use a limiter, people often say it's a good idea to clip a little beforehand, and that's why Comprendre includes a pre-clipper. The clipper takes care of the briefest peaks (the attack of a snare, the click of a kick), which the ear often barely notices when they lose 1 or 2 dB; the limiter, which no longer has to catch them, works less on everything else. To turn Comprendre into a master limiter, set **Ratio** to 42 and **Attack** to 0 (the lookahead in chapter 22 can round out the setup), then bring **Pre Clip** down until it touches only those peaks. Listen to the attacks: if they start to crackle or harden, raise the threshold, switch to **Adaa 1** or **Adaa 2**, which are cleaner but a bit duller, or increase the oversampling. **Linked** preserves the stereo image; the other modes clip each channel on its own. **Post Clip**, at the end of the chain, will catch whatever got past the limiter.

The compressor clipper, set in the **OUTPUT** tab, is described later along with the other controls in that tab. The post-clipper is covered together with the makeup.

Headroom: the internal headroom

Headroom MAIN tab

Range: −24 to 24 dB **Default:** 0 dB **ID:** `chambre`

Gain applied just after the pre-clipper, at the compressor input; the inverse gain is applied after the compressor and the monitors, just before **Makeup**.

This headroom therefore sits **inside** the pre- and post-clippers. Raising **Headroom** drives the signal harder into the compressor (which, as far as compression goes, amounts to lowering the threshold) and into the compressor clipper, without changing either the output level or what the pre- and post-clippers do.

A word from the author

Why two headrooms? Because there are clippers. The **Ch.0 Db**/**Ch.1 Db** gains act as far outside as possible: there's nothing before them except the input, and nothing after them except the output. **Headroom** acts inside the clippers. If the clippers aren't in use, the two amount to the same thing; if they are, the result is very different.

Headroom and sidechain

Headroom does not apply to the sidechain signal. When detection comes entirely from the sidechain (**Sidechain** at 1), Headroom therefore no longer has any effect on the compression itself: the input gain and the inverse gain cancel each other out around the compressor.

> **♪ In the studio: using Headroom to drive the inside**
>
> **Headroom** acts like a “drive” on the whole inside of the chain. Raised by a few decibels, it makes the compressor work harder, like a lower threshold, but it also drives the signal harder into the wave-shaper, which works on amplitudes (chapter 17), and into the compressor clipper; the inverse gain then brings everything back to the original level, compression aside. On a preset that saturates, it’s a simple way to thicken the grit without changing what the pre- and post-clippers do. Keep an eye on the dynamic EQs, which clip their signal at full internal level as soon as one of their bands is active (chapter 18). Eight of the included presets use it, from −15 dB (**acqua_titanium**) to +12 dB: **klang_mjuc_mk1** and **acqua_ash-ultra** pair a Headroom of +12 dB with a **Threshold** of 0 dB and a **Ratio** of 42, while **waves_la2a** and **kush_silka** lower it by 6 and 7 dB.

After the compressor: the OUTPUT tab



Figure 14 — The OUTPUT tab: monitoring, compressor clipper, blends, Out→Sc.

Remember that the compressor gain applies only to the **target**, that is, the part of the signal selected by the output filters (chapter 14). The **remainder** of the signal bypasses the compressor. By default, the target is the entire signal.

Comp Clip

OUTPUT tab

Range: −24 to 0 dB **Default:** 0 dB **ID:** `fauxplafond`

Threshold of the clipper placed right after the compressor gain. It acts only on the compressed target, before the remainder of the signal is added back. Because it sits inside **Headroom**, its effective threshold follows the Headroom setting. At 0 dB, it is disabled.

Comp Clip Mode

OUTPUT tab

Choices: Unlink · Adaa 1 · Adaa 2 **Default:** Unlink **ID:** `fauxplafond_adaa`

Mode of this clipper: *Unlink*, *Adaa 1*, or *Adaa 2*. There is no *Linked* mode for this clipper: each channel is always clipped on its own.

Comp Db

OUTPUT tab

Range: −24 to 24 dB **Default:** 0 dB **ID:** `sortie_mix`

Gain applied to the compressed target alone, right after **Comp Clip**. The remainder of the signal is not affected: this control lets you balance the compressed part against the untargeted part.

dW | Dw

OUTPUT tab

Range: −1 to 1 **Default:** 0 **ID:** `sortie_heros`

A second blend between compressed and uncompressed signal, done inside the chain. Let D be the uncompressed signal (the target before compression, plus the remainder) and W the compressed target. Then:

- at **0** (default), the output is W + remainder: normal operation;
- between **0 and +1**, the output is D + (1 − h)·W: you start from 100%/100% parallel compression and progressively pull out the compressed signal; at +1, only D remains, with no compression at all;
- between **−1 and 0**, the output is (1 + h)·D + W (h being negative): you progressively pull out the uncompressed signal; at −1, only W remains, the compressed target alone, **without** the remainder of the signal.

Unlike **Dry/Wet**, the “dry” of this blend is taken from inside the chain: after the pre-clipper, Headroom, and the output filters.

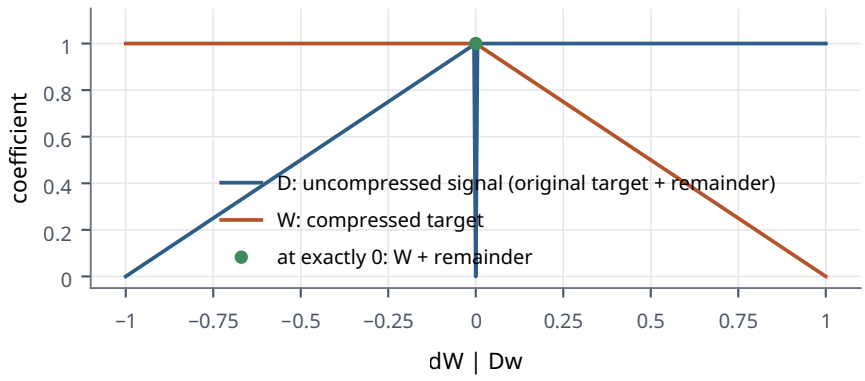


Figure 15 — dW | Dw blend coefficients as a function of the control position.

Watch out around zero

The transition through 0 is deliberately discontinuous, and you need to keep that in mind when using it. At exactly 0, the output is W + remainder; just next to 0, it's roughly D + W, that is, the uncompressed signal **plus** the compressed signal, which is noticeably louder. The knob is designed as two controls in one: one going "dry to wet," the other "wet to dry." At −1, the remainder of the signal (the part not targeted by the output filters) disappears: you hear only the compressed target. The default value, 0, is affected by neither of these two effects.

Makeup and post-clipper

Makeup

main row tab

Range: −24 to 24 dB **Default:** 0 dB **ID:** `maquillage`

Compensation gain applied after the **Headroom** return and **before the post-clipper**, so it sits "inside" the clippers. It also applies to the monitor signals.

Below this knob, the **Snap Out->In** button adds the compensation computed by the **Out→In** auto-volume (chapter 8) to Makeup.

A word from the author

There are two schools for compensating level: **Makeup**, inside the clippers, or **Output Db**, at the very end. If the clippers aren't in use, the two amount to the same thing.

Post Clip

main row tab

Range: −24 to 0 dB **Default:** 0 dB **ID:** `plafond`

Post-clipper threshold, placed right after **Makeup**. It applies to the signal while it is still in Mid/Side, if that mode is active, and still multiplied by the **Ch.0 Db**/**Ch.1 Db** gains. At 0 dB, it is disabled.

When a listening monitor is active (except **Monitor Delta**), the post-clipper is not applied to the monitored signal.

Post Clip Mode

MAIN tab

Choices: Linked · Unlink · Adaa 1 · Adaa 2 **Default:** Linked **ID:** `plafond_adaa`

Post-clipper mode: *Linked*, *Unlink*, *Adaa 1*, or *Adaa 2*, as for the pre-clipper.

Output gain

Output Db

MAIN tab

Range: −24 to 24 dB **Default:** 0 dB **ID:** `sortie_dernier`

Output gain, applied after the conversion back to left/right. It is not quite the last stage, though: **Monitor Delta**, the **Out→In** and **Out→Sc** auto-volumes, **Dry/Wet**, and **Phase** all come after it.

Two consequences: the dry part of **Dry/Wet** does not get Output Db, and the **Out→In** auto-volume, if active, ends up neutralizing Output Db (just as it does Makeup), since it brings the output back to the input level.

Below this knob, in the MAIN tab, a second **Snap Out->In** button adds the **Out→In** compensation to Output Db instead of Makeup.

Linking or unlinking the channels

Channels Unlink

main row tab

Range: 0 to 1 **Default:** 0 **ID:** `rupture`

Sets the compressor's stereo linking.

- At **0** (default), the channels are **linked**: the detection signals of the two channels are combined into one (according to the **Channels Mix** setting in the INPUT tab: average, channel 0, channel 1, minimum, or maximum, chapter 14), and a single compressor computes a single gain, applied to both channels.
- At **1**, the channels are **unlinked**: each channel has its own compressor, with its own detection and its own gain.
- In between, **both** computations run in parallel and the two gains are blended linearly: gain = linked gain × (1 − u) + channel gain × u. This is not a degree of detection linking, but a blend of the results. The processing load is then roughly doubled.

With **L/R to M/S**, unlinking the channels amounts to compressing the Mid and the Side separately.

♪ In the studio: linking, unlinking, and Mid/Side in mastering

With the channels linked, a snare panned slightly left pulls both sides down at once: the stereo image stays put, which makes this the safest choice for a bus or a master. With the channels unlinked, each side lives its own life: the whole can sound wider and more alive, but the image sometimes drifts toward the less compressed side. Intermediate values blend the two: **klang_mjuc_mk1** and **waves_la2a** set **Channels Unlink** to 0.5, while six presets, including **softube_weiss** and **massenbug_drc2_piano**, unlink completely. In Mid/Side (**L/R to M/S** at +1), unlinking the channels gives you two compressors, one for the center and one for the sides. In mastering, this lets you hold the vocal, the bass, and the kick in the center without the reverbs and wide guitars pumping along with them; or, conversely, compress the sides a little to stabilize the width. The per-channel gains round out the move: in Mid/Side, raising **Ch.1 Db** by 2 or 3 dB drives the clippers and the compressor harder on the Side only, without changing the output level. Chapter 14 offers another route: compressing only the Side using the target's Mid/Side.

Dry/Wet and polarity

Dry/Wet

main row tab

Range: 0 to 1 **Default:** 1 **ID:** `drywet`

Final blend between the input signal (dry) and the processed signal (wet): output = input × (1 − w) + processed × w.

The dry signal is the input multiplied by **Input Db**, delayed by the internal latency to stay aligned: it goes through no other stage. The wet signal includes everything else, including **Output Db** and the output auto-volumes.

Dry/Wet and filters

When Linkwitz-Riley output filters are engaged (chapter 14), the target + remainder sum has a flat frequency response but a rotated phase. Mixed with the dry signal, which hasn't gone through this rotation, it can create interference. François demonstrates this in his video on FIR filters.

♪ In the studio: parallel compression

Comprendre gives you two ways to blend the compressed sound with the original. **Dry/Wet** is a crossfade: at 50%, you hear half input, half processed output, and the dry is the untouched input, to which only **Input Db** applies. The **dW | Dw** control, between 0 and +1, does what engineers do on their consoles: it keeps the uncompressed sound at full level and **adds** the compressed sound to it. Just above 0, both are at full level; as you move toward +1, the compressed part shrinks until it disappears. This blend happens inside the chain, before Makeup and the post-clipper, which therefore process the sum, and **Comp Db** sets the level of the compressed part alone. On a drum bus, try some frankly exaggerated compression (**Ratio** from 8 to 10, **Attack** 1 ms, **Release** from 50 to 100 ms, around ten decibels of reduction), then tuck it under the original: the attacks remain those of the drums, while the compressed part often brings up the room, the tom sustain, and the ghost notes. Since the blend is louder than the original, compare level-matched (chapter 8).

> **'New York' compression**
>
> Parallel compression has a long history. The Dolby A noise reduction system (1965) already mixed two paths in its circuit, one of which was compressed. In October 1977, Mike Beville described in *Studio Sound* magazine a limiter-compressor placed in parallel with the direct signal, to compress the low levels of classical music effectively. It was Bob Katz who coined the term "parallel compression," which he describes as upward compression; the nickname "New York" comes from the New York engineers who used it heavily. Why it works: even a compressor slammed to the limit still sounds natural when it's mixed low under the direct signal, and, as Katz notes, the part of the signal that doesn't go through the processing reduces distortion. The Weiss DS1-MK2, the elysia alpha, and the SSL Bus Compressor 2 later built this blend in.

Phase

MAIN tab

Range: 0 to 1 **Default:** 1 **ID:** `phase`

Inverts the polarity of the entire output, dry included: it is the very last operation. When the button is **checked** (the default), the polarity is **normal**; unchecked, the output is inverted.

This setting is mainly used for the analysis method described in chapter 24: by inverting Comprendre's polarity and summing it with the signal of another compressor, you hear the difference between them.

Hearing what's going on: the monitors

Several controls replace the output signal with an internal signal so you can listen to it. They are **exclusive**: turning one on turns the others off. There is one exception: **Monitor Comp** works together with **Monitor Output**, which it turns on automatically.

Monitor Delta

MAIN tab

Range: 0 to 1 **Default:** 0 **ID:** `delta`

Plays the **difference** between output and input: output − input. What remains is exactly what Comprendre has changed: the compression, but also the level changes (Makeup, Output Db), the clipping, the saturation, and any phase shift from the output filters.

The difference is computed after **Output Db**. The output auto-volumes, **Dry/Wet**, and **Phase** are still applied after it.

Monitor Input

INPUT tab

Range: 0 to 1 **Default:** 0 **ID:** `fuite`

Plays the **detection signal**: what the compressor measures, after the blend with the sidechain, the input filters, the feedback, the input Mid/Side, and the sends from the dynamic EQs and the waveshaper. It is described in chapter 14.

Monitor Output

OUTPUT tab

Range: 0 to 1 **Default:** 0 **ID:** `sortie_cible`

Plays the **uncompressed target**: the part of the signal selected by the output filters and the output Mid/Side, after the target's dynamic EQs and waveshaper, but without the compressor gain and without the remainder of the signal.

Monitor Comp

OUTPUT tab

Range: 0 to 1 **Default:** 0 **ID:** `delta_cible`

Plays the **compressed target** (after **Comp Clip** and **Comp Db**), without the remainder of the signal. Despite its id, this is not a difference: it's the compressed counterpart of **Monitor Output**, which it turns on automatically. Turning Monitor Comp off also turns Monitor Output off.

Other monitors, specific to certain modules, let you hear an isolated saturation product: **Monitor opto** (GR (cv) tab, chapter 15), **Monitor Fet** (WS tab, chapter 17), and **Mon bias sat** (BIAS dst tab, chapter 21). They follow the same exclusivity rule.

> **Monitors and post-clipper**
> A monitor's signal still goes through the Headroom return, Makeup, the per-channel gain return, the inverse Mid/Side conversion, and Output Db, but not through the post-clipper.

> **♪ In the studio: hearing what the compressor takes away**
> **Monitor Delta** is the most honest way to listen in the plugin. Set **Makeup** and **Output Db** to 0 dB, then turn it on: now you hear only what the processing has changed. On a vocal, that should be the syllables that are too loud, and only those; if the whole phrase comes through in the difference, the threshold is probably too low. On a drum bus, you hear the attacks the compressor is shaving off: lengthen **Attack**, and the crack disappears from the difference, which means it's back in the output. Add some waveshaper or bias, and the difference also lets you hear the added harmonics. It's the best school for learning to recognize overly heavy compression by ear, and for checking that a subtle setting really is doing something.

CHAPTER 8

Auto-Volumes

Comprendre has four controls that automatically, and somewhat slowly, adjust one level to match another. François calls them the **auto-volumes**. The most useful one is **Out->In**, which automatically makes up for the level lost to compression; the other three allow more unusual pairings between the input, the output, and the sidechain.

These are honesty tools. When you listen, a slightly louder sound almost always seems better: the ear hears more low end, more presence, more energy. A compressor followed by generous makeup gain wins plenty of comparisons this way, even when it's damaging the sound. **Out->In** gets around this trap: it brings the output back to the average level of the input, so when you bypass Comprendre and then switch it back on, you no longer hear a difference in level but a difference in shape: in punch, density, color. It's essential for judging a subtle setting, an envelope mode, or a saturation designer, and invaluable for the mastering engineer, who needs to know whether the processing truly improves the track. The other three auto-volumes will interest the sound designer, and anyone imitating a compressor (chapter 24).

Control	Location	What gets adjusted	Toward what
Out->In Secs	main row	the output	the input level
In->Out Secs	INPUT tab	the input	the output level
In->Sc Secs	INPUT tab	the input	the sidechain level
Out->Sc Secs	OUTPUT tab	the output	the sidechain level



Figure 16 — The input auto-volumes in the INPUT tab, and Out->In on the main row.

The common principle

All four auto-volumes work the same way.

1. At every sample, each of the two signals being compared is measured as the average of the absolute values of its channels. This measurement is instantaneous: it is not smoothed.
2. The ratio between the target level and the level to be corrected is computed: this is the gain that would make them equal at that instant.
3. A correction gain chases this ratio with an adjustable time constant, like a one-pole envelope follower: at each sample, it covers a small fraction of the distance separating it from the measured ratio.
4. The signal to be corrected is multiplied by this gain.

The correction gain therefore changes slowly: it doesn't follow transients, but the average trend of the levels over the set time.

The time constant T is set in seconds, from 0 to 42:

- **42 s**, the default value, **disables** the auto-volume;
- in between, the gain covers about 63% of the gap in T seconds, and 95% in 3T;
- **0 s** makes the tracking instantaneous: at every sample, the corrected level is forced onto the target level.

The knob is very progressive: its position p (from 0 to 1) corresponds to $T = 42 \cdot p^4$. The middle of its travel is therefore about 2.6 s, and you have to go down to a quarter of the travel to reach 0.16 s.

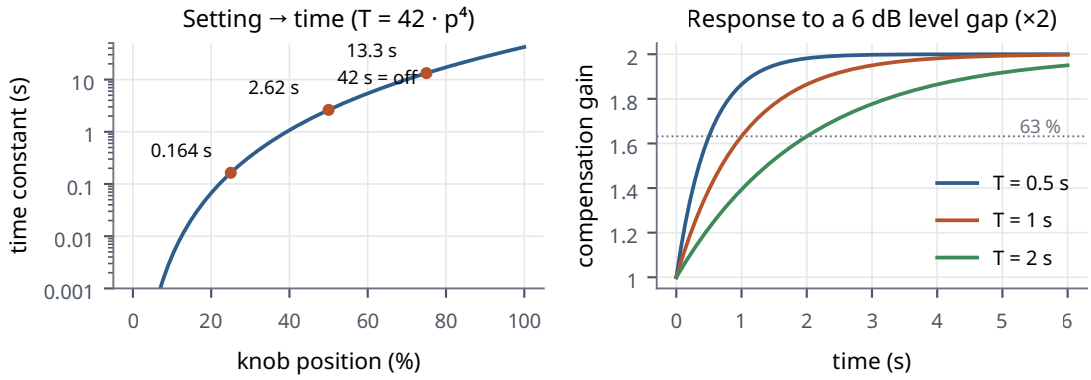


Figure 17 — Left: how the knob position maps to the time constant. Right: the response of the correction gain when the level gap suddenly jumps to 6 dB (×2).

> **Going further**
>
> The correction gain g follows the rule g ← g + (r − g) · k, where r is the ratio of the instantaneous levels and k = 1 − e^(−1/(T·fs)), fs being the internal sample rate (oversampling included). The computation is only carried out if both levels are strictly positive; otherwise, the sample passes through with a gain of 1, not with the last computed gain. During digital silence on either of the two signals, the correction therefore momentarily disappears.

Out->In: automatic compensation

Out->In Secs

main row tab

Range: 0 to 42 s **Default:** 42 s **ID:** `compensation`

Brings the **output** level toward the **input** level. This is automatic makeup gain: the more gain reduction the compressor applies, the more the compensation rises, so that on average the output has the same level as the input.

- The reference input level is the input multiplied by **Input Db** (the red curve in the graphs).
- The output level is measured after **Output Db**, just before this correction.
- The correction applies to the output after **Monitor Delta**, before **Out->Sc** and **Dry/Wet**.

At 0 s, the output is forced, sample by sample, onto the input level: the compression is then completely canceled, which is only useful for checking that the control works.

A word from the author

One to two seconds is a good setting: slow enough not to undo the dynamics created by the compressor, fast enough to follow the section changes in a song.

♪ In the studio: choosing the time scale

The **Out->In** time constant sets the time scale at which you hear the compression. Variations faster than it (the attack of a snare, the release of a bus, a syllable that's too loud) remain the compressor's work; slower variations, like going from a verse to a chorus, are brought back in line with those of the input. At one or two seconds, the value François recommends, a chorus stays louder than a verse much as it is in the original recording, and you judge what the compressor does to the notes and the bars. Go down toward a few tenths of a second, and the compensation starts undoing the compression itself; go up to ten seconds or more, and it corrects only the overall level of a long passage. Left on during mixing, it makes Comprendre a compressor that changes the texture of a track without changing, on average, its place in the mix.

Because **Out->In** measures the output after **Makeup** and **Output Db**, it ends up neutralizing both of these controls: if you raise Makeup by 3 dB, the compensation drops by 3 dB within a few time constants. That's what lets you use Out->In as a tool for measuring the makeup you need.

Reading the compensation on the graph

The compensation value is drawn on the history graph as a **yellow-green line**. Its scale is linear and different from that of the other curves:

- at **half height**, the compensation is 1 (0 dB): no correction;
- at the **top** of the graph, it is 2 (+6 dB); beyond that, the line stays pinned to the top;
- below the halfway mark, the output is louder than the input and the compensation brings it down.

Whether this line is displayed depends on the **Gfx Make Up** setting in the GRAPHS tab: at 0, it is never displayed; at 0.5, it is displayed only if Out->In is active; at 1 (default), it is always displayed. When Out->In is disabled, the compensation is 1 and the line stays in the middle.

Finding the right makeup

The method François suggests is to use **Out->In** to measure, then transfer the measurement to Makeup:

1. set **Out->In Secs** to one to two seconds;
2. play a representative passage, such as a chorus;
3. raise **Makeup** until the yellow-green line hovers around half height: Makeup alone now makes up for the lost level;
4. set **Out->In Secs** back to 42 to disable the automatic compensation.

The **Snap Out->In** button automates step 3.

> **♪ In the studio: comparing presets level-matched**
>
> None of the included presets sets **Makeup**, **Output Db**, or an auto-volume: depending on their threshold, ratio, and saturation, they change the level more or less, and the loudest one is likely to sound best to you. To choose between two presets, or between two positions of a control, leave **Out->In** active at one or two seconds throughout the comparison: each attempt comes out at the input's average level, and what you're comparing is the character. This is especially useful with the saturation modules, whose saturated signal can be louder than the original (chapter 17). Once you've made your choice, transfer the compensation with **Snap Out->In** and disable the auto-volume, as described below.

The Snap Out->In buttons

Two buttons share this name:

- the one **below the Makeup knob** (main row) adds the current compensation, expressed in decibels and rounded to the hundredth, to the **Makeup** value;
- the one **below Output Db** (MAIN tab) adds it to the **Output Db** value.

The button doesn't disable **Out->In**. If the auto-volume stays active, the compensation is applied twice right after the click (once through Makeup or Output Db, once through the auto-volume), until the auto-volume returns on its own toward 0 dB, within a few time constants.

> **Tip**
>
> Click **Snap Out->In** during a representative passage, then immediately set **Out->In Secs** back to 42. In the reverse order, the compensation would already be back to 0 dB when you click, and the button would add nothing.

Level matching, from the Finalizer to streaming

Automatic level compensation is not a new idea. The TC Electronic Finalizer (1996), a three-band compressor that had become a very widespread mastering tool by around 2000, already offered it, at a price that put it within reach of many studios. The question has since changed scale: most streaming platforms now normalize loudness by default. Spotify targets −14 LUFS integrated (with options at −11 and −19 LUFS), YouTube −14 LUFS, Apple Music −16 LUFS. A track mastered louder is simply turned down on playback: the loudness war recounted in chapter 2 loses some of its meaning there, and, according to at least one critic, it has shifted toward density, achieved through compression and EQ rather than by slamming the limiter. Judging level-matched means hearing your work the way the listener will hear it.

In->Out: the input follows the output

In->Out Secs

INPUT tab

Range: 0 to 42 s **Default:** 42 s **ID:** `int_a_out`

Pushes the **input** level toward the **output** level. The gain is applied to the signal right after the **In->Sc** auto-volume, before the Mid/Side conversion and the pre-clipper; it therefore affects everything downstream, detection included.

The ratio uses the output level of the previous sample and the **original** input level, that is, before **In->Sc** has modified it. This is a feedback loop: the input is corrected based on the output, which itself depends on the input.

Don't set In->Out to 0

At 0 s, the loop becomes instantaneous: at every sample, the input level is replaced by the level of the previous output. The behavior then degenerates (the envelope of the input signal is destroyed, and the level can collapse or run away). François himself flags this as a known issue: this control should not be set to zero.

In->Out and FIR filters

The **In->Out** correction is not applied to the copy of the signal that feeds the FIR band filters (**X Filters Taps** other than 0, chapter 14). With these filters, the target and the remainder are no longer computed from the same signal, and their sum no longer reconstructs the input correctly. In addition, the output used for the computation is delayed by the filter latency, which makes the control inconsistent. François advises against using In->Out with the FIR filters.

In->Sc and Out->Sc: following the sidechain

In->Sc Secs

INPUT tab

Range: 0 to 42 s **Default:** 42 s **ID:** `int_a_ext`

Pushes the **input** level toward the level of the **sidechain** signal. The correction is applied at the very beginning of the chain, right after the delay line: it affects everything downstream.

The sidechain level is measured on the raw signal, without any Comprendre gain; the input level includes **Input Db**.

Out->Sc Secs

OUTPUT tab

Range: 0 to 42 s **Default:** 42 s **ID:** `out_a_ext`

Pushes the **output** level toward the **sidechain** level. This is the last gain correction in the chain: it comes after **Out->In**, just before **Dry/Wet**.

These two controls only have an effect if the plugin receives a signal on its sidechain input. They work independently of the **Sidechain** setting in the MAIN tab, which only concerns detection.

> **Instantaneous tracking**
>
> Set to 0 s, In->Sc and Out->Sc impose the sidechain level on the signal at every sample: the waveform is preserved, but its envelope becomes that of the sidechain. Unless that's the effect you're after, this produces heavy distortion. Values on the order of a second simply align the average levels.

> **♪ In the studio: following a reference**
>
> Set to one or two seconds, **Out->Sc** aligns the average level of your track with that of a signal sent to the sidechain: a reference mix, another take, the original version of a song. It's a great ally for imitation: during an analysis (chapter 24), it keeps both signals at the same average level, so that their difference comes down to shape alone. **In->Sc** does the same at the input, hence before the compressor: takes whose level varies then hit the compressor at the reference level, and a single threshold suits them better. At very short settings, these two auto-volumes become sound design effects: the track takes on the level envelope of the sidechain, at the cost of heavy distortion. Turn your monitoring down before trying it, as the level swings can be brutal; with **In->Sc**, the clippers further down the chain can act as a safety net.

Summary: where each auto-volume acts

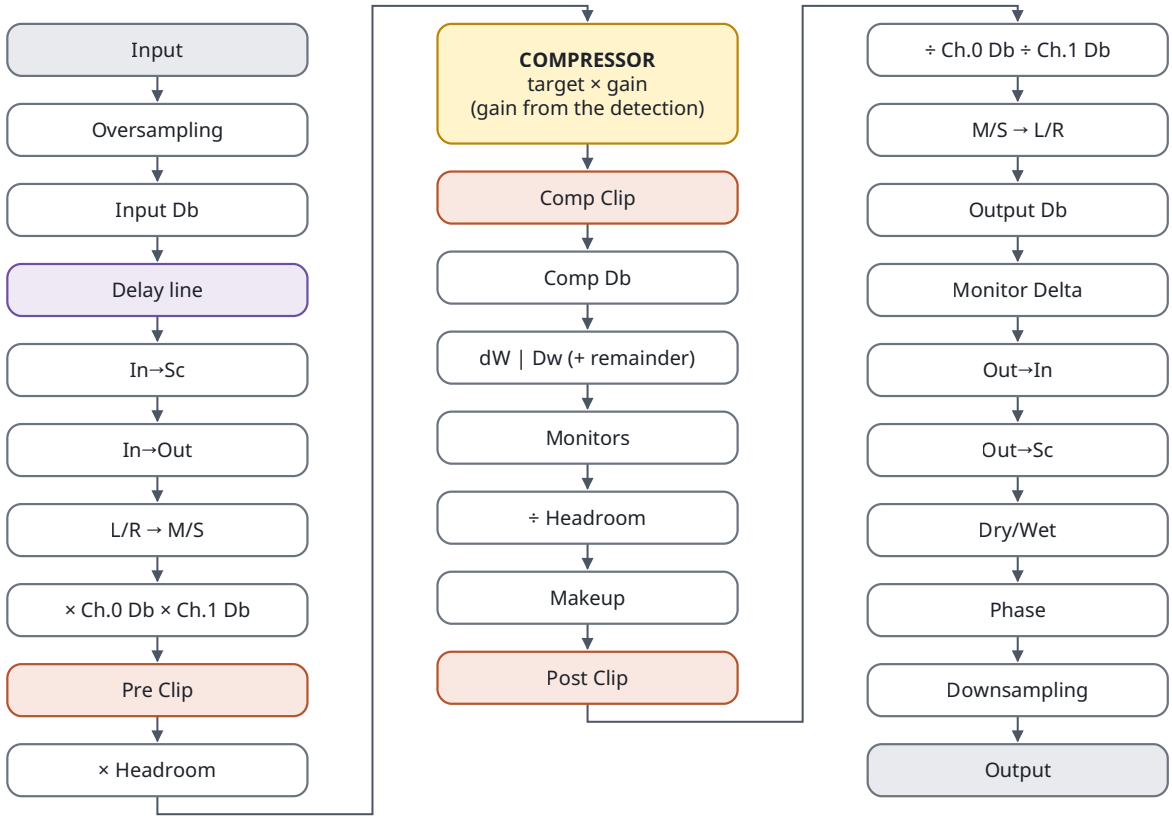


Figure 18 — Position of the auto-volumes in the chain: In→Sc and In→Out in the first column, Out→In and Out→Sc in the third.

- **In->Sc** then **In->Out**: at the beginning of the chain, before the switch to Mid/Side;
- **Out->In** then **Out->Sc**: at the end, after **Output Db** and **Monitor Delta**, before **Dry/Wet** and **Phase**.

The reference levels are always measured on the original signals: the input multiplied by **Input Db** only, and the raw sidechain.

CHAPTER 9

How Comprendre Compresses

This chapter opens the part of the manual devoted to the compressor. Before going into the details of the controls (chapters 10 to 15), it describes the **engine** shared by all of Comprendre's gain computations: what it receives, what it computes, in what order, and what it does with its default settings. It is the key to everything that follows: each compressor tab sets one specific stage of this engine.

People say a compressor "glues," "breathes," "bites," or disappears into the mix. All these words describe the same thing: the way the gain moves, faster or slower, deeper or shallower, in time with the music. Chapter 2 recalls Dave Berners's observation about VCA compressors: once the gain element no longer adds any color of its own, all of the character comes from how the timing and the curve are designed. That design is precisely this engine. In a piece of hardware, it's frozen into the circuit; in Comprendre, each of its stages has become a control, and the graphs show you every one of them at work.

This is the most "theoretical" chapter in the manual, but it's the one that lets you hear what the others describe. If you teach or study audio, you'll find what you need to show, graphs in hand, why very short times make a bass buzz or why a slow release makes a pad pump. If you mix, you'll understand why two compressors set to the same values don't sound the same. And if you're trying to imitate a piece of gear, you'll know at which stage to look for the difference.

Chapter 6 placed the compressor in the signal chain. It receives a **detection** signal, prepared by the INPUT tab, and applies a gain to a **target**, selected by the OUTPUT tab. This chapter is about what happens in between.



Figure 19 — The main row. The six circled knobs directly control the compressor engine.

The six circled knobs act directly on the engine: **Threshold**, **Ratio**, **Knee**, and **Cv Lin/Db** set the transfer curve, while **Attack** and **Release** set the envelope follower. All the other compressor controls are in the tabs, from **DET (in)** to **GR (cv)**.

The common engine

In Comprendre's code, the heart of the compressor is a single function, which François calls the **factory**. It is called once per sample. It receives a positive number, the absolute value of the detection signal, and returns a **gain**: a multiplier that equals 1 when there's nothing to do, about 0.5 when the signal needs to come down by 6 dB, 0.1 for 20 dB, and so on.

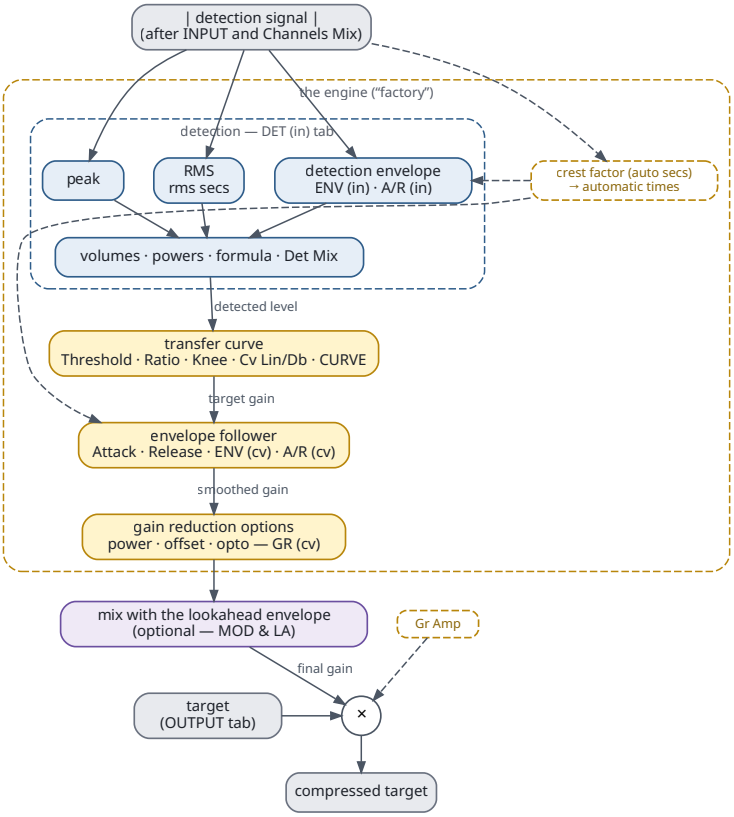


Figure 20 — The compressor engine, from detection to the gain applied to the target. Dotted: auxiliary measurements and controls.

The stages always run in the same order.

1. **Detection** turns the absolute value of the signal into a **detected level**. It computes three measurements in parallel: the peak, an RMS (root mean square) average, and a detection envelope. It sets the amount of each, can combine them with a formula, then merges them into a single level. Controls: the **DET (in)** tab, plus the **ENV (in)** and **A/R (in)** tabs for the detection envelope (chapter 13).
2. **The transfer curve** turns this level into a **target gain**: the gain that would have to be applied, at this precise instant, for the output to respect the threshold and the ratio. The curve is *static*: it only knows the present level and has no notion of duration. Controls: Threshold, Ratio, Knee, Cv Lin/Db, and the **CURVE** tab (chapter 10).
3. **The envelope follower** chases the target gain at a limited speed and produces the **smoothed gain**. This is what brings time into compression. Controls: Attack, Release, and the **ENV (cv)** and **A/R (cv)** tabs (chapters 11 and 12).

4. **The gain reduction options** in the **GR (cv)** tab can reshape this gain further: power, offset, opto designer (chapter 15).
5. If lookahead is active, the resulting gain can be blended with an envelope computed on the "future" signal (chapter 22).
6. The **final gain** multiplies the **target**. At this point, the **Gr Amp** control in the CURVE tab can amplify, reduce, or invert the gain reduction (chapter 10).

Alongside this chain, the engine measures the **crest factor** of the incoming signal: the ratio between its peak and its RMS value, at a speed set by **auto secs** (DET (in) tab). This measurement is used only by the automatic times, **Attack Auto** and **Release Auto** (chapter 12). It is taken on the signal as it enters the engine, before any of the detection settings.

> **Gain, gain reduction, and decibels**
>
> Inside the engine, the gain is a simple multiplier: 1 corresponds to 0 dB, 0.5 to −6 dB, 0.1 to −20 dB. **Gain reduction** is what gets taken away: a gain of 0.5 is a reduction of 6 dB. Unless otherwise stated (the **Cv Lin/Db** and **EvMu Lin/Db** controls), Comprendre computes with these linear values, not in decibels. This is one of its distinctive features, and the reason why its default curve is not a straight line on a decibel graph.

> **Attack and release: what are we talking about?**
>
> For the follower placed after the curve, there is **attack** when the target gain is lower than the smoothed gain (more compression is needed), and **release** when it is higher (the compression has to let go). For the detection envelope, placed before the curve, attack corresponds to a rising level and release to a falling level. In both cases, then, attack is the reaction to a rise in level, and release the reaction to a drop.

Two envelopes: before and after the curve

The engine can smooth at two different points.

- **After the curve**, the envelope follower smooths the **gain**. This is the most common position, and the only one active by default.
- **Before the curve**, the detection envelope smooths the **level**. The curve is then applied to this smoothed level: the target gain is already smooth and follows the shape of the curve exactly.

Both can be used together, each with its own settings: the ENV (in) and A/R (in) tabs for the first, ENV (cv) and A/R (cv) for the second. The two followers are built on the same code and offer the same possibilities (linear or decibels, branching or decoupled, Mu, Gamma, shapes, modes…), with one exception: the speed modulations (mod attack, mod release) exist only for the follower placed after the curve.

With equal times, they don't give the same result. The following figure compares the two positions on two fast-decaying notes, with the same attack (5 ms) and the same release (100 ms).

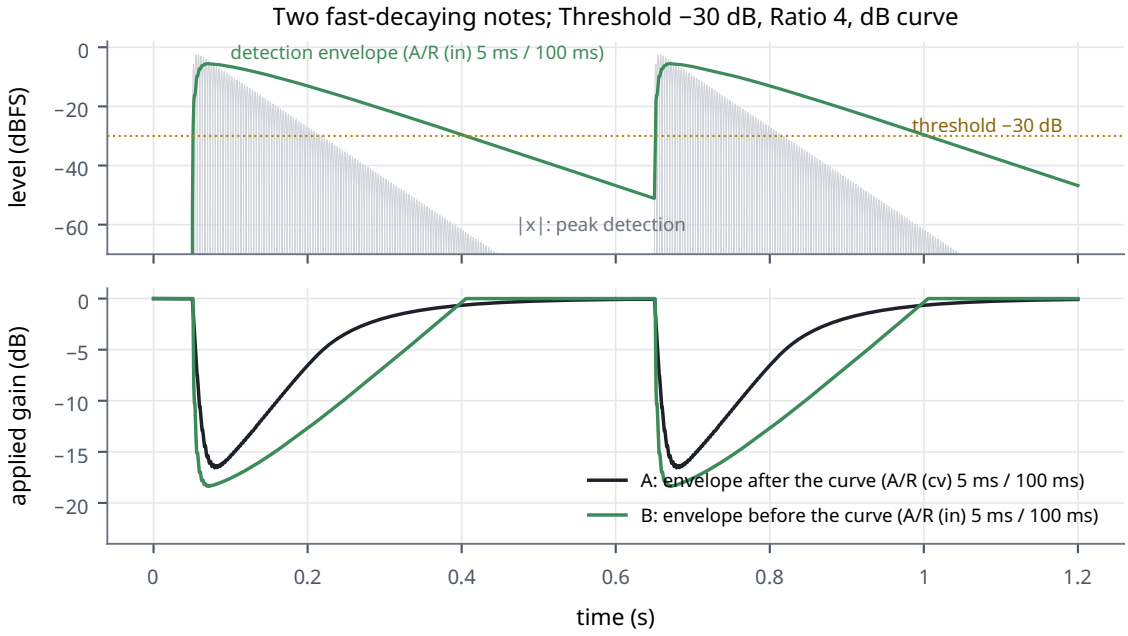


Figure 21 — Same signal, same times, two positions. A: peak detection and follower after the curve. B: detection envelope alone (peak db at −24, env db at 0 dB) and follower after the curve disabled (Attack and Release at 0). Curve in decibels (Cv Lin/Db = 1), Threshold −30 dB, Ratio 4.

- **In A**, it's the gain itself that recovers, along an exponential: very fast at first, then more and more slowly. Seen in decibels, the release stretches out into a long tail that approaches 0 dB without ever quite reaching it.
- **In B**, it's the smoothed level that falls along an exponential, which is a straight line in decibels. The curve turns this straight line into another straight line: the gain recovers at a constant speed in decibels, then stops dead as soon as the smoothed level drops back below the threshold.

Two practical differences follow. With the envelope before the curve, the length of the release depends on the level reached: the further the note exceeds the threshold, the longer it takes the smoothed level to fall back below it. And the detection envelope works at all levels, including below the threshold: a transient shorter than its attack may never push the smoothed level over the threshold. The follower after the curve, on the other hand, only starts acting when the target gain drops below 1, that is, when the signal exceeds the threshold.

> **A word from the author**
>
> Academic publications (Giannoulis, Massberg, and Reiss) consider the follower placed after the curve to be more transparent. Yet many commercial compressors, some of them highly regarded, smooth the level before the curve. Comprendre offers both. François regards the envelope placed before the curve as part of the detector, and for his part prefers the position after the curve.

> **♪ In the studio: smoothing the level or smoothing the gain**
>
> The two positions don't sound the same, and several presets have chosen to smooth only the level, before the curve. **korneff_comp2**, **melda_mturbo**, **tonelib_multicomp**, and **massenbug_drc2_mix** all set **Attack** and **Release** to 0: no smoothing after the curve, everything goes through the detection envelope (**env db** at 0 dB in the DET (in) tab, times in the A/R (in) tab). **korneff_comp2** sets an attack of 35 ms and a release of 180 ms there, **tonelib_multicomp** a release of 250 ms.
>
> When you listen, two traits often stand out. A transient shorter than the detection attack can get through the compressor without fully triggering it, which preserves the crack of a snare. And the release length follows the performance: a note that goes well over the threshold holds the compression longer than a note that just grazes it. On a vocal or a piano, compare the two positions with equal times (example B in the previous figure shows how): one may sound more alive to you, the other more even.

The default compressor

When it loads, Comprendre is a deliberately simple compressor. Here is exactly what it computes at every sample.

Detection: the instantaneous peak

In stereo, the absolute values of the two detection channels are first combined by **Channels Mix** (INPUT tab, chapter 14), set to *Avg* by default: the engine receives their average, $v = (|x_0| + |x_1|) / 2$. In mono, it simply receives $|x|$.

The DET (in) tab then lets this level through unchanged: **peak db** is at 0 dB, **rms db** and **env db** are at −24 dB, which excludes these two measurements, and **Det Mix** takes the maximum of the remaining measurements. The detected level is therefore the instantaneous absolute value of the signal, with no smoothing or hold whatsoever. On a sine wave, this level drops to zero twice per cycle.

The curve: linear, threshold at −20 dB, ratio 2

Cv Lin/Db is at 0: the curve is computed on amplitudes. The threshold, −20 dB, corresponds to the amplitude S = 0.1; the ratio of 2 corresponds to a slope of 1/2 above the threshold.

- If the detected level v does not exceed 0.1, the target gain is 1: no reduction.
- Above that, the target output is S + (v − S) / 2: only half of the amplitude excess above the threshold is kept. The target gain is the ratio between this target output and the level: g = (0.1 + (v − 0.1) / 2) / v.

Detected level	Target gain (linear curve, default)	Same setting in dB
−20 dBFS	0 dB	0 dB
−17 dBFS	−1.4 dB	−1.5 dB
−14 dBFS	−2.5 dB	−3 dB
−10 dBFS	−3.6 dB	−5 dB
−6 dBFS	−4.4 dB	−7 dB

Detected level	Target gain (linear curve, default)	Same setting in dB
−3 dBFS	−4.9 dB	−8.5 dB
0 dBFS	−5.2 dB	−10 dB

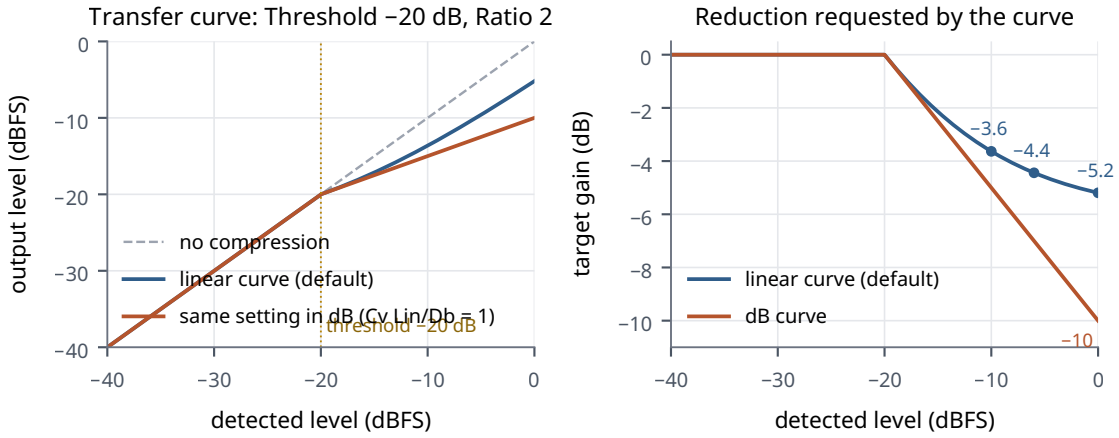


Figure 22 — Left: the default transfer curve on decibel axes, compared with the same setting computed in decibels. Right: the corresponding target gain.

Seen in decibels, the linear curve is not a straight line: it bends and compresses less and less as the level rises. With a threshold at −20 dB and a ratio of 2, the default compressor never takes more than 5.2 dB off a signal that stays below 0 dBFS, whereas a decibel curve with the same settings would take off 10. Chapter 10 details the two ways of computing the curve.

The other curve settings are neutral: zero knee (**Knee** at 0), no reduction limit (**GR Max** at −60 dB, **GR Min** at 0 dB), no gate (**Ratio Gate** at 1).

The follower: 10 ms, exponential, branching

The smoothed gain e chases the target gain g according to the simplest rule there is: at each sample, it covers a small fraction c of the distance separating it from its target.

$e \leftarrow e + (g - e) \cdot c$

The fraction c is the attack fraction when e is above g (the gain has to come down), and the release fraction otherwise. It is computed from the set time T and the internal sample rate fs: $c = 1 - \exp(-1/(T \cdot fs))$. With T = 10 ms at 48 kHz, c is about 0.0021: at each sample, the smoothed gain covers 0.2% of the remaining gap. After T, it has covered 63% of the way; after 3T, 95%; after 4.6T, 99%.

This is the default behavior of all the controls in the ENV (cv) and A/R (cv) tabs: a **linear** follower (EvMu Lin/Db at 0), **branching** (EvMu Br/Cp at 0), with no **Hold**, no **Mu**, and no **Gamma**; **automatic** times disabled; an **altitude** of 1/e, which gives the 63% rule; **shape** 1; **mode** 1, which François calls logarithmic and which is this exponential law.

Everything else is neutral

The options in the GR (cv) tab are at neutral (**Env Pow Unpow** at 1, **Gr|Env Pow** at 0, **Env Offset** at 0, **Opto** at 0), lookahead is disabled, **Gr Amp** is at 1, and the target is the entire signal. There is no makeup: the output is simply the input multiplied by the smoothed gain.

Response to a level step

The simplest way to watch the engine is to feed it a level that changes abruptly. With a square wave, the absolute value is constant: the detected level jumps from one plateau to the other without oscillating.

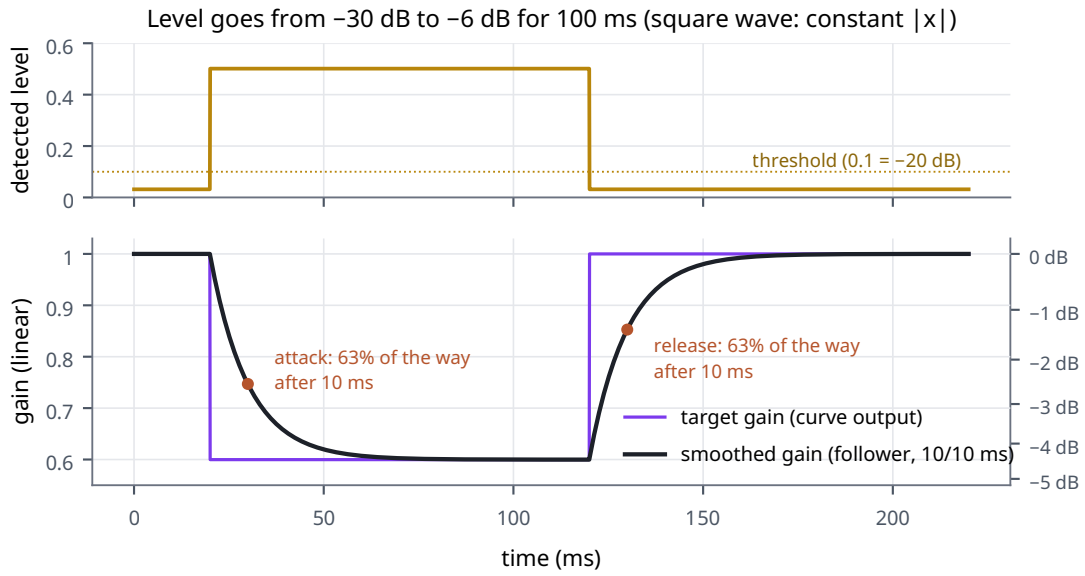


Figure 23 — Response of the default compressor to a level step from −30 dB to −6 dB, then back to −30 dB. The target gain jumps instantly; the smoothed gain catches up with it along an exponential with a 10 ms time constant.

As long as the level stays at −30 dB, below the threshold, the target gain is 1 and nothing happens. As soon as the level jumps to −6 dB, the curve calls for a gain of 0.60 (−4.4 dB): the target gain jumps instantly to this value. The smoothed gain catches up with it in 30 to 50 ms, having covered 63% of the way after 10 ms. When the level returns to −30 dB, the target gain instantly goes back up to 1 and the smoothed gain follows it with the same law, this time using the release time.

The exponential is computed on the linear gain, not in decibels. Seen in decibels (right-hand scale), attack and release are therefore no longer symmetrical, even with equal times: after 10 ms, the attack has covered 57% of the way in decibels, the release 69%.

Target gain versus smoothed gain

With a real signal, peak detection makes things less simple. The following figure repeats the same step, but with a 100 Hz sine wave.

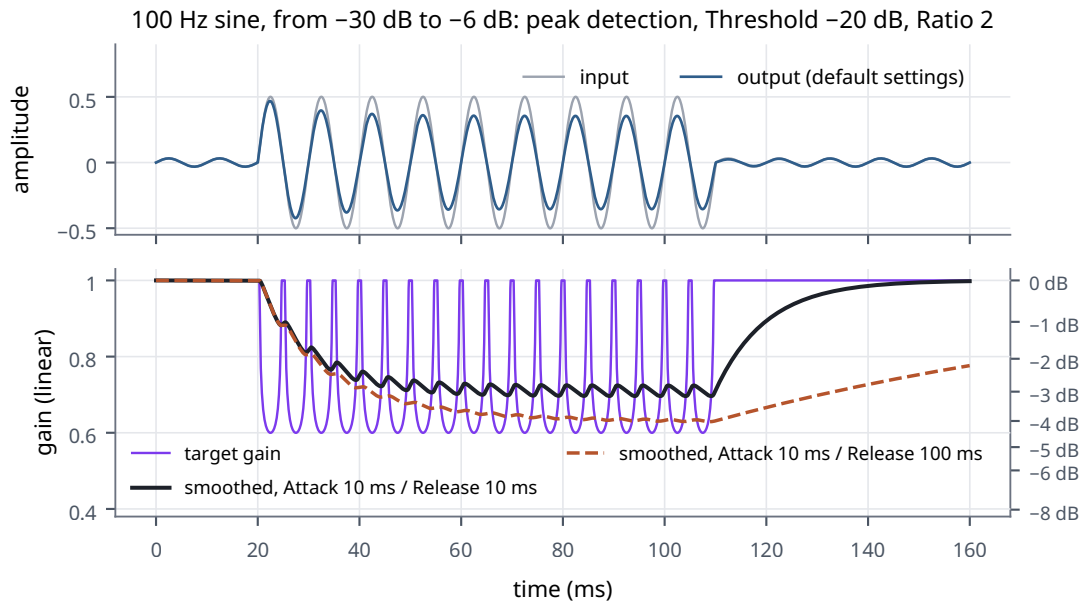


Figure 24 — The same level step on a 100 Hz sine wave. The target gain swings between 1 and 0.60 two hundred times per second; the smoothed gain ripples. A longer release reduces the ripple and increases the average reduction.

The detected level is the absolute value of the sine wave: it drops to zero at every zero crossing. The target gain therefore swings back and forth between 1 (at the zero crossings, below the threshold) and 0.60 (at the peaks), twice per cycle. The smoothed gain tries to follow: it goes down at each peak, with the attack time, and comes back up between peaks, with the release time. This has two important consequences.

- **Release acts during a sustained sound.** With peak detection, release doesn't just come into play after a sound ends: it works on every half cycle. The average reduction you get therefore depends as much on the ratio between attack and release as on the curve. With 10 ms / 10 ms, the gain settles around 0.71 (−3 dB), far from the −4.4 dB called for at the peak; with a 100 ms release, it goes down to around 0.63 (−4 dB).
- **Gain ripple is distortion.** The smoothed gain ripples at twice the signal frequency, and this ripple modulates the sound: it adds harmonics. The lower the frequency and the shorter the times, the more pronounced the effect.

> **Low frequencies and short times**
>
> If very short times make the low end buzz, there are several possible remedies: lengthen the release, measure the level with RMS or a detection envelope rather than the peak (chapter 13), or use a hold (**Hold branch**, chapter 11). Conversely, François points out that with zero attack and zero release, compression becomes very aggressive and turns into distortion: that's a possible effect, not a mistake.

♪ In the studio: when the compressor becomes a clipper

Take this logic all the way: **Attack** and **Release** at 0, **Ratio** at 42 (infinity), **Threshold** at 0 dB, and unlinked channels (**Channels Unlink** at 1, see below). The smoothed gain is then the target gain, sample by sample, and every sample that exceeds the threshold is brought back exactly to it: the compressor behaves like a clipper. That's the heart of the **acqua_ash-ultra** preset, built by imitating a clipper, the Acustica Ash Ultra: **Attack** and **Release** at 1 µs, **Ratio** 42, **Threshold** 0 dB, unlinked channels, and a **Headroom** of 12 dB that drives the signal into the engine (which amounts to lowering the threshold by the same amount); the Vari-Mu designer (chapters 19 to 21) adds its own color. On a kick or a synth bass, try the same idea while lowering the threshold: the lower it goes, the harder the clipping bites. If you bring the times back up a little, to a few tens of microseconds, the ear often perceives it as softer.

Simulation on a musical signal

The last figure shows the default compressor at work on a synthetic drum loop (kick, snare, hi-hat, over a pad and a bass).

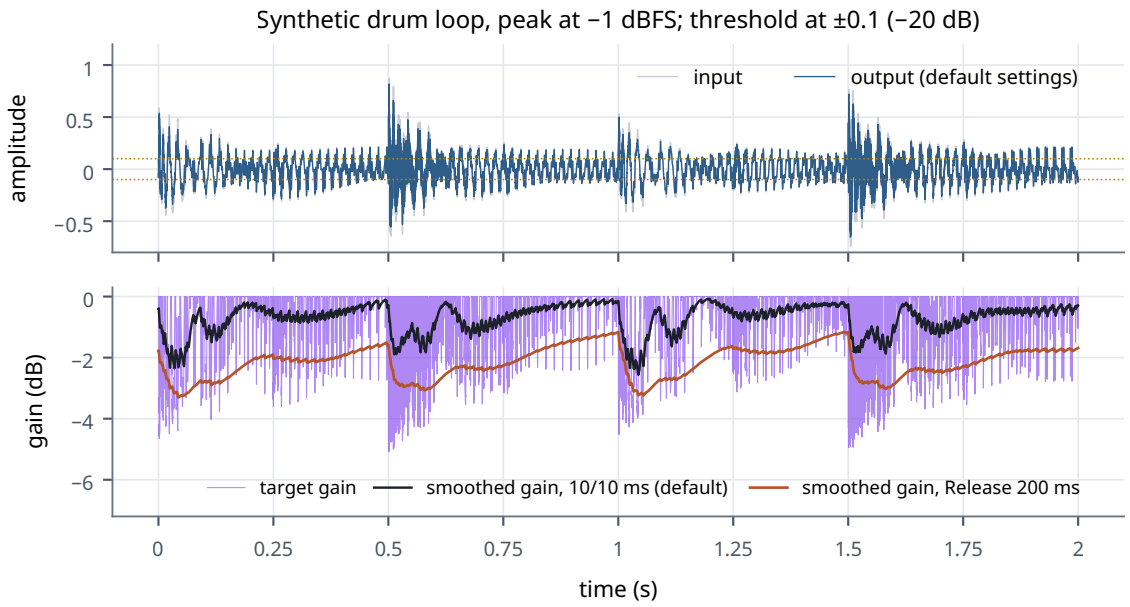


Figure 25 — The default compressor on a drum loop. Top: input and output. Bottom: the target gain (purple) and the smoothed gain, with the default times and with a 200 ms release.

With its default settings, Comprendre applies light, fast compression: the gain dips by about 2 to 2.5 dB on each kick or snare hit, then recovers within a few tens of milliseconds. The pad, which also exceeds the threshold, gets a constant small reduction, on the order of half a decibel, which ripples with its own peaks. With a 200 ms release, the gain no longer has time to recover between hits: it swings between −1.2 and −3.3 dB, and the pad follows these slow variations, like a gentle pumping.

> **♪ In the studio: starting from the default setting**
>
> From his very first video, François presents the default setting as the cleanest and simplest there is. It's an excellent starting point for learning to listen to a compressor. Put Comprendre on a drum bus or a loop, bring **Threshold** down toward −30 dB, and raise **Ratio** to 4. First listen with the default times (10 ms): each hit is grabbed and then released right away, the pad and the bass barely breathe, and the low end may take on a slight grit. Then raise **Release** to 200 ms: the gain no longer has time to recover between hits, the whole group starts moving together, and you can hear the compression tug slightly at the pad on every hit. The right setting for your track lies somewhere in between.

Sample-by-sample computation

Comprendre works neither in blocks nor with analysis windows. The engine is called **once per sample**, at the internal rate, that is, multiplied by the oversampling factor if oversampling is active (×2 to ×16, MAIN tab). The engine itself therefore introduces no latency.

What gives the compressor its time dimension is the engine's **memory**: from one sample to the next, it keeps its states (the smoothed gain, the RMS average, the detection envelope, the crest factor, the auto knee followers, and so on). Each sample moves these states forward by one step.

Within a sample, the order is always the same:

1. if an automatic time (Attack Auto, Release Auto) or a modulation source (MOD & LA tab, chapter 23) is active, the crest factor is updated and the attack and release speeds are recomputed; otherwise, they were computed once and for all at the last settings change;
2. detection (peak, RMS, detection envelope, volumes, powers, formula, Det Mix);
3. the transfer curve;
4. the envelope follower;
5. the gain reduction options and the opto.

Then, outside the engine, come the optional blend with the lookahead envelope and the application to the target.

Times are set in seconds and converted into per-sample fractions using the internal rate. They therefore stay accurate whatever the oversampling: 10 ms is 480 steps at 48 kHz and 1,920 steps at 48 kHz oversampled ×4, each time with a fraction c adjusted accordingly.

Unneeded stages are not computed. The detection envelope only exists if **env db** is above −24 dB and one of the A/R (in) times is not zero. The follower after the curve only exists if **Attack** or **Release** is not zero, or if an option in the GR (cv) tab requires it: with Attack and Release at 0, the smoothed gain is exactly the target gain, sample by sample.

> **Going further: the default compressor in five lines**
>
> For each sample, with x_0 and x_1 the two channels of the detection signal, e the smoothed gain kept from one sample to the next, and $c = 1 - \exp(-1/(0.01 \cdot fs))$:
>
> 1. $v = (|x_0| + |x_1|) / 2$ (Channels Mix on *Avg*, peak detection);
> 2. $g = 1$ if $v \leq 0.1$, otherwise $g = (0.1 + (v - 0.1) / 2) / v$ (linear curve, threshold −20 dB, ratio 2);
> 3. $e \leftarrow e + (g - e) \cdot c$ (the same c for attack and release, since both times are 10 ms);
> 4. channel 0 output = $target_0 \cdot e$, channel 1 output = $target_1 \cdot e$;
> 5. since the target is, by default, the entire signal: output = input · e.

One engine, several uses

The same engine is called at several places in the plugin, with other inputs, other settings, and stopping at different points in the chain. Each call keeps its own memory.

Use	What the engine receives	How far it goes	Controls used	When	Chapter
linked compressor	the detection of the channels, combined by Channels Mix	the whole engine	DET (in) to GR (cv)	**Channels Unlink** < 1 (default)	7, 14
per-channel compressor	each channel's detection, separately	the whole engine, once per channel	the same	**Channels Unlink** > 0	7
bias **inverse** path	the detection of the inverted signal	the whole engine (linked and per channel)	the same	bias active and **Bias +inverse** > 0	19
lookahead envelope	the anticipated envelope (a gain)	the follower only	ENV (la), A/R (la)	lookahead active	22
bias detector	the bias source signal	detection only	DET (bs), ENV (bs), A/R (bs)	bias active	20

Linked or per channel

As a linked compressor, a single engine computes a single gain, applied to all channels. As a per-channel compressor, each channel has its own engine, its own detection, and its own gain. In between, **both computations are carried out** and their gains are blended channel by channel, which roughly doubles the processing load.

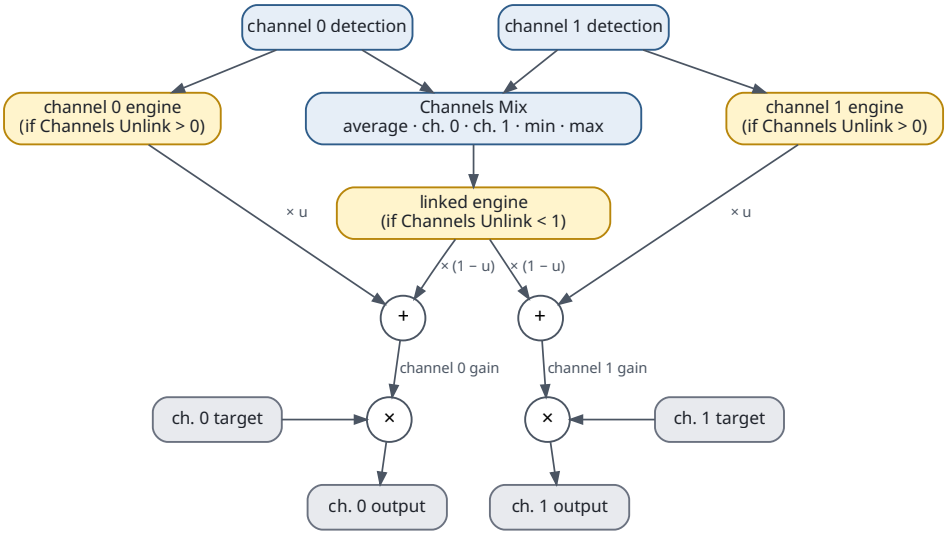


Figure 26 — Linked compressor and per-channel compressors. Each channel's gain blends the linked gain and the channel's own gain, in the proportion u = Channels Unlink.

All these engines share **the same settings**: threshold, ratio, times, detection, and so on. You can't set one channel's engine differently from the other's; each one does, however, keep its own memory and react to its own signal. Combined with the switch to Mid/Side (**L/R to M/S**, chapter 7), per-channel mode lets you compress the Mid and the Side separately.

Linking, unlinking, splitting center and sides

The Fairchild 670 (chapter 2) could already work in two ways: as two independent limiters, one per channel, or as a limiter of the "lateral and vertical" components, that is, of the sum and the difference of the channels. Designed for vinyl cutting, this mode is used today for Mid/Side processing, and a classic mixing trick is to compress the sides of a drum bus independently of the center. The Manley Variable Mu offers an M/S modification in which the first channel handles the Mid and the second the Side. The API 2500 (2002), for its part, made stereo linking a stepped control, from 50 to 100%, with a filter that selects which frequencies trigger the linking. Comprendre makes it a continuous control: **Channels Unlink** moves progressively from the linked compressor to the per-channel compressor, and, combined with **L/R to M/S**, it compresses the center and the sides separately.

♪ In the studio: linking or unlinking the channels

On a stereo bus or a mix, keep the channels linked (the default): a single gain for both sides, and the stereo image stays put, even when an instrument panned to one side triggers the compression. Unlinked, the channels each react to their own content: a loud hi-hat on the left no longer pulls down the guitar on the right, but the image can shift in time with the compression. An intermediate value keeps part of the linking. The presets show both uses: **acqua_tulip**, **ddmf_mde2**, **kiive_kc1**, **massenbug_drc2_piano**, and **softube_weiss** set **Channels Unlink** to 1, while **klang_mjuc_mk1** and **waves_la2a** stop at 0.5. In mastering, try switching to Mid/Side with unlinked channels: the center, where the vocal, the bass, and the kick often live, and the sides are then each compressed according to their own level.

The bias inverse path

When the Vari-Mu designer is active and **Bias +inverse** is above 0, Comprendre runs a complete copy of the processing on the inverted signal, in parallel with the normal path. The linked compressor and the per-channel compressor are then duplicated, each with its own memory, and the two paths are recombined (chapter 19).

The lookahead envelope

With lookahead (a negative **la delay** setting, MOD & LA tab), the chain is first computed on the undelayed signal: this is the "future" section described in chapter 6. The compressor engines (and the bias detector) run a first time there. The resulting envelope is then smoothed by an engine reduced to its follower alone, set by the **ENV (la)** and **A/R (la)** tabs, and blended with the gain of the present (chapter 22). The **auto secs** control in the DET (in) tab also sets the crest factor speed for this follower.

The bias detector

The bias uses the engine reduced to its detection alone, with its own set of controls: the **DET (bs)**, **ENV (bs)**, and **A/R (bs)** tabs, exact copies of DET (in), ENV (in), and A/R (in). It works channel by channel. Its output doesn't drive the compressor's transfer curve but the bias curve (chapter 20).

What the graphs show during simple compression

Comprendre's graphs show almost every stage of the engine. The following figure simulates compression that is a little heavier than the default, displayed in decibels with a slowed-down history so you can clearly see the attacks and releases.

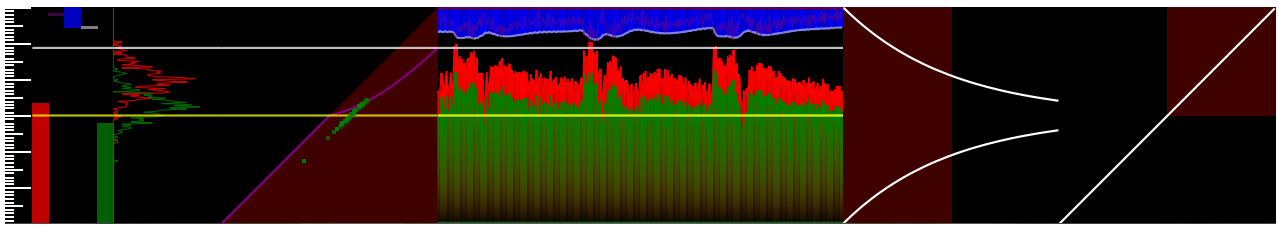


Figure 27 — The graphs during simple compression on the drum loop: Threshold −30 dB, Ratio 4 (linear curve), Attack 10 ms, Release 100 ms; decibel scale (Gfx Lin/Db = 1) and slowed-down history (Gfx Secs = 1/240 s, i.e., about 1.6 s displayed).

You'll find every stage of the engine in it (the colors are described in chapter 4).

- **The detected level.** The **red** bars in the history are the input level; the thin red lines are the detection measurements. With the default detection, the final detection is the peak of the detection signal, which is itself the input: the red line hugs the tops of the bars. The excluded measurements (RMS and envelope, at −24 dB) stay pinned to the bottom of the graph.
- **The target gain.** The thin **purple** line is the gain called for by the curve. Each column of the history is an average: the fast oscillation of the target gain under peak detection isn't visible as such, but it pulls the purple line upward.

- **The smoothed gain.** The thick **white** line is the output of the follower, which tries to follow the purple one. With short times, the two are close; with long times, the white line lags behind and is smoother.
- **The actual reduction.** The **blue** bars, which hang down from the top, measure the gap between the output (in **green**) and the input.
- **The transfer curve.** The **purple** curve in the curve graph is the static curve. Each **green dot** has as its x-coordinate the level entering the engine, before the detection settings, and as its y-coordinate that same level multiplied by the smoothed gain. A dot **below** the curve indicates more reduction than the curve calls for (the release is lagging); a dot **above** it, less reduction (the attack is lagging). With zero attack and release and the default detection, all the dots land on the purple curve.
- **The attack and release shapes.** The two small graphs on the right show the exponential shape of the follower after the curve; they change with the altitude, the shape, and the mode (chapter 12).
- **The reference lines.** The **yellow** line marks the threshold; the horizontal **white** line, the output level the curve gives a 0 dBFS signal.

> **Watching the engine at work**
>
> Set **Attack** and **Release** to 0: the white line merges with the purple one and the green dots land on the curve. Then raise the attack, then the release: you'll see the white line peel away from the purple one and the green dots move away from the curve, above it during attacks, below it during releases. Slow down the history with **Gfx Secs** to observe the details of a transient.

Roadmap: from stage to chapter

The following chapters take the stages of the engine one by one. Their order is that of François's walkthrough: the curve first, then the envelopes and their times, then detection, which reuses the envelope concepts. If you're new to compressors, read chapters 10, 11, and 12 before chapter 13.

Stage	What it produces	Controls	Chapter
preparing the detection signal and choosing the target	detection signal, target, and remainder	INPUT and OUTPUT tabs, Sidechain (MAIN)	14
transfer curve	target gain	Threshold, Ratio, Knee, Cv Lin/Db, CURVE tab, Gr Amp	10
envelope follower	smoothed gain	ENV (cv) tab, and its twins ENV (in), ENV (la), ENV (bs)	11
attack and release	follower speeds	Attack, Release, A/R (cv) tab and its twins	12
detection	detected level	DET (in) tab, with ENV (in) and A/R (in)	13
gain reduction options and opto	final gain	GR (cv) and F (gr) tabs	15

The modules that come next modify the sound itself rather than the gain: the saturation product, the waveshaper, and the dynamic EQs (Part IV), and the bias (Part V). Lookahead and the modulations (Part VI) act on the engine from the outside.

CHAPTER 10

The Transfer Curve

The transfer curve is the “static” part of the compressor: for each level measured by detection, it decides what gain should be applied. It’s what the **Threshold**, **Ratio**, and **Knee** knobs on the main row control.

The curve decides how much the compressor takes off, and from what point. A threshold and a ratio are enough to bring a runaway vocal back into the mix; a wide knee lets compression come in without you hearing it settle in; an infinite ratio turns the curve into a wall for the peaks of a master. These moves are universal, and mixers, mastering engineers, and sound designers alike use them every day. What’s far less common is being able to choose the very logic of the curve, to let the knee follow the music as on some tube compressors, or to push peaks back down below the threshold to invent new effects.

Comprendre adds several uncommon possibilities:

- computing on amplitudes (**linear** curve) or on decibels, and every value in between;
- a knee that widens by itself as the reduction increases;
- minimum and maximum reduction limits;
- ratios below 1, and even negative ones;
- a built-in gate, with its own threshold, ratio, and knee;
- a gain reduction amplifier.

In the compressor (chapter 6), the curve sits between detection and the envelope follower:

1. detection (chapter 13) provides a level at every sample;
2. the transfer curve turns this level into a **target gain**;
3. the envelope follower (chapters 11 and 12) chases this gain, with the attack and release times;
4. the gain reduction options (chapter 15) reshape it further, then it multiplies the target.

Apart from the auto knee and the modulations, the curve has no memory: the same detected level always gives the same target gain. All the dynamics (attack, release) come from the next stage.



Figure 28 — The transfer curve controls: Threshold, Ratio, Knee, and Cv Lin/Db on the main row, and the CURVE tab. The mod ratio and mod gate controls, which vary the ratios according to the frequency content, are described in chapter 23.

How it works

From detected level to target gain

Let v be the detected level. The curve computes a gain g(v). If this gain were applied instantly, the output level would be v × g(v): that's the level drawn by the **purple** curve in the transfer curve graph. The gain g(v) itself is drawn in purple in the history (chapter 4).

The target gain is the **product** of two curves:

- the **compressor**, which acts above its threshold (**Threshold**, **Ratio**, **Knee**);
- the **gate**, which acts below its own threshold (**Tresh Gate**, **Ratio Gate**, **Knee Gate**). By default, it is inactive.

Each of the two is clamped by **GR Max** and **GR Min**. Each exists in two versions: one computed on amplitudes, the other on decibels. The **Cv Lin/Db** control chooses the version, or blends the two.

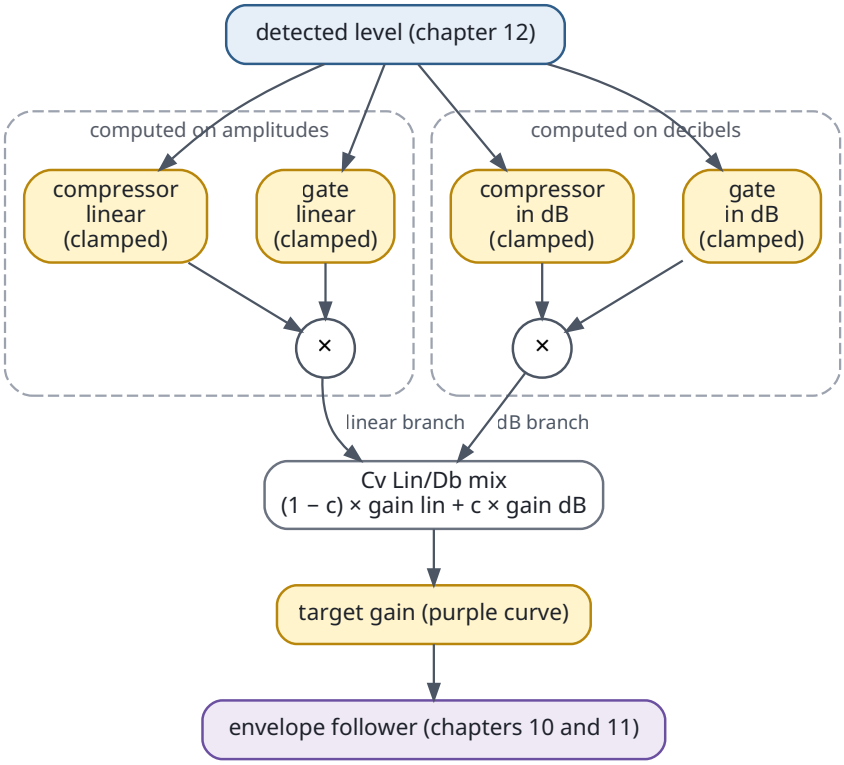


Figure 29 — How the transfer curve computation is organized. The compressor depends on Threshold, Ratio, Knee, Ratio Sens, Knee Auto, and mod ratio; the gate on Tresh Gate, Ratio Gate, Knee Gate, Knee Gate Auto, and mod gate; each gain is clamped by GR Max and GR Min.

> **What isn't computed**
>
> When **Ratio** is exactly 1, the compressor is not computed: its gain is 1, whatever the other settings, including **GR Min** and **GR Max**. Likewise, the gate is not computed as long as **Ratio Gate** is 1, its default value. Finally, only the version (linear or dB) that has a nonzero weight in **Cv Lin/Db** is computed.

Linear or decibels

This is the most important concept in this chapter, and the most original one in Comprendre.

In decibels, the curve follows the usual rule: above the threshold, each decibel of input gives 1/ratio decibel of output.

> output (dB) = threshold (dB) + (input (dB) − threshold (dB)) / ratio

With a threshold of −20 dB and a ratio of 2:1, a signal at 0 dBFS, which is 20 dB above the threshold, comes out at −10 dBFS. Drawn on decibel axes, the curve is a straight line with a break at the threshold.

In linear (the default), the same rule is applied to **amplitudes**: the threshold is converted to an amplitude (−20 dB becomes 0.1), and it's the amplitude excess above the threshold that is divided by the ratio.

> output = threshold + (input − threshold) / ratio, in amplitudes

Drawn on linear axes, this curve is a straight line with a break at the threshold. But seen in decibels, it bends: just above the threshold, it does have the slope of the ratio, then it gradually straightens up toward a 1:1 slope as the level rises. With a threshold of −20 dB and a ratio of 4:1, the “local” ratio at 0 dBFS is down to just 1.3:1. All in all, for the same settings, the linear curve compresses high levels **much less** than the dB curve.

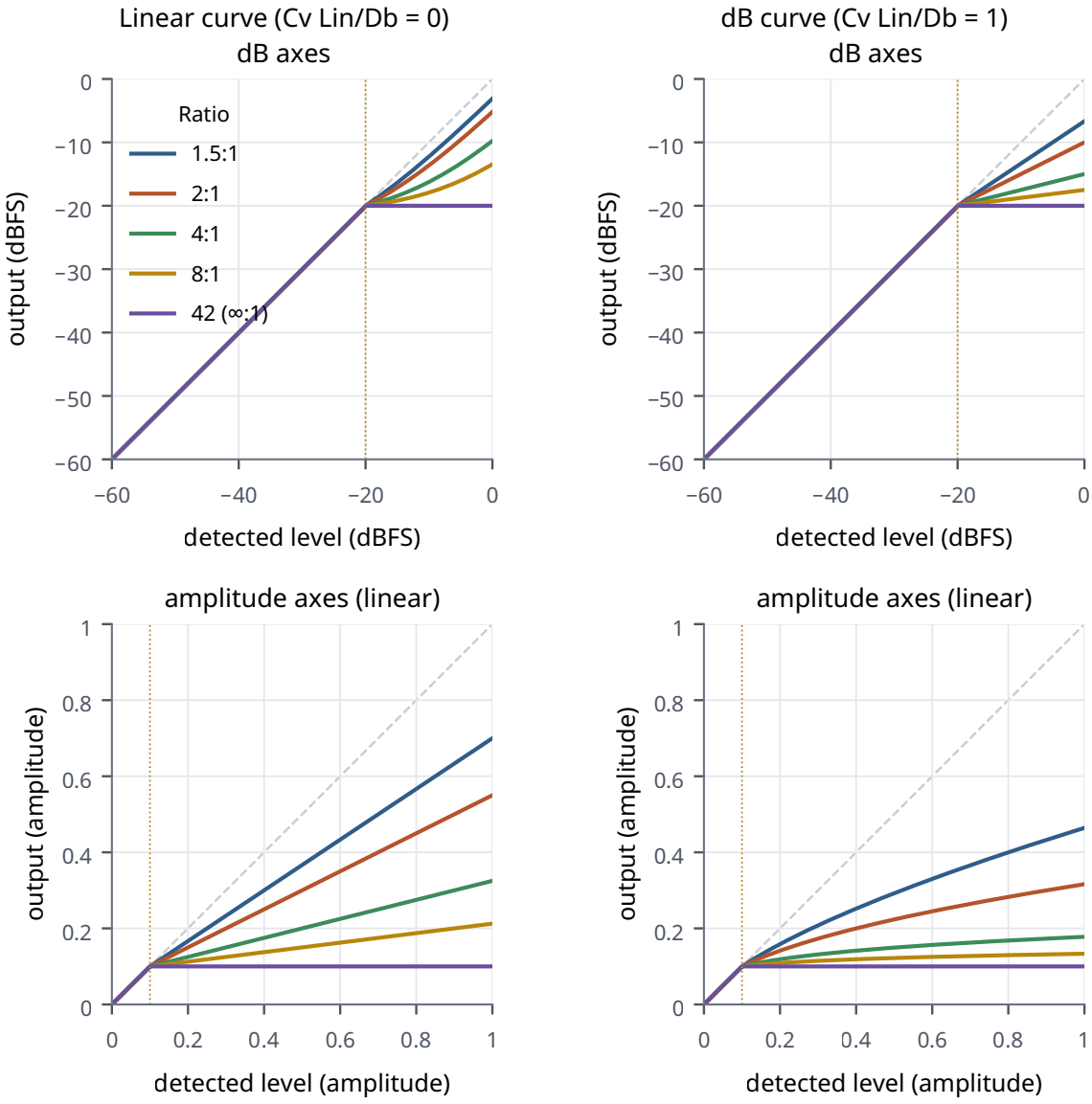


Figure 30 — The same ratios (threshold −20 dB) computed in linear (left) and in dB (right), seen on dB axes (top) and amplitude axes (bottom). Each curve is straight in the domain where it is computed, and curved in the other.

The following table gives the output level (dBFS) for a threshold of −20 dB and zero knee.

Ratio	Computation	−15 dBFS	−10 dBFS	−6 dBFS	0 dBFS
2:1	linear	−17.2	−13.6	−10.4	−5.2
2:1	dB	−17.5	−15	−13	−10
4:1	linear	−18.5	−16.3	−14	−9.8

Ratio	Computation	−15 dBFS	−10 dBFS	−6 dBFS	0 dBFS
4:1	dB	−18.8	−17.5	−16.5	−15
8:1	linear	−19.2	−17.9	−16.5	−13.5
8:1	dB	−19.4	−18.8	−18.3	−17.5

The two computations only coincide at the extremes: below the threshold, where they do nothing, and for a ratio of 42 (infinity), where the output is pinned to the threshold in both cases.

> **A word from the author**
>
> François likes to reason about curves and look at them in linear, where you really see the amplitudes; that's why the curve is linear by default. He acknowledges that most compressors work and display in decibels, and he recommends the dB curve as soon as you use a knee: the result is then what you'd expect from a conventional compressor.

> **Getting 'classic' compressor behavior**
>
> To get the behavior described in most compressor manuals (ratio expressed in dB, knee symmetrical in dB), set **Cv Lin/Db** to 1. If you keep the linear curve, expect to have to raise the ratio considerably to get the same reduction at high levels.

Cv Lin/Db

main row tab

Range: 0 to 1 **Default:** 0 **ID:** `gain`

Chooses the domain in which the transfer curve is computed: **0** (default) = linear, on amplitudes; **1** = in decibels.

In between, **both** curves are computed and their gains are blended linearly: gain = (1 − c) × linear gain + c × dB gain, where c is the control's value. So this is not a curve "halfway" in the decibel sense, but a weighted average of the two gains. In practice, the intermediate values do produce a family of curves that moves gradually from one to the other (figure below).

The control affects the whole computation: compressor, gate, knees, GR Min and GR Max limits, auto knees. The graph's reference lines (threshold, ratio, GR Min, GR Max) are blended in the same way.

Cv Lin/Db has nothing to do with **Gfx Lin/Db**, which only changes the display scale of the graphs (chapter 4), nor with **EvMu Lin/Db**, which chooses the domain of the envelope follower (chapter 11).

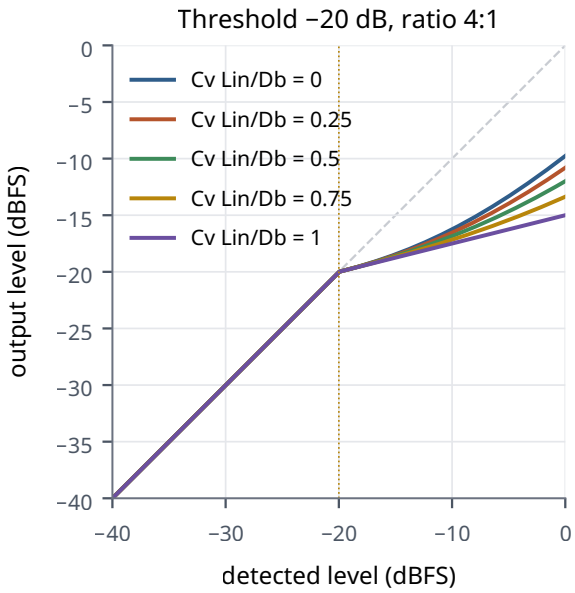


Figure 31 — Cv Lin/Db from 0 to 1: blending the gains moves gradually from the linear curve to the dB curve.

> **♪ In the studio: amplitudes or decibels?**
>
> The two curves only really differ above the threshold, and all the more so the higher the signal climbs there. On drums, at the same ratio, the linear curve acts firmly on the medium hits but leaves the loudest ones a good part of their height: the accents keep their contour. The decibel curve applies the same slope at all levels: the result is more even, more conventional.
>
> The included presets show that both logics are alive and well: 41 of them set **Cv Lin/Db** to 1, 37 keep the linear curve, and two stop in between (**pa_bettermaker** at 0.3, **rockruepel_comptwo_drums** at 0.5). The split doesn't follow hardware families: the 1176 imitations fall on both sides, as do the LA-2A ones (**waves_la2a** in decibels, **slate_fg2a** and **native_vc2a** in linear). Only the imitations of the SSL bus compressor form an almost solid block, nine out of ten in decibels (**ssl_1**, **tr_bus**, **bx_townhouse**, **acustica_sand3**…), with **cytomic_glue** remaining linear. The analysis made the call, preset by preset; it's up to you to hear which one serves your track.

Reading the curve on the graph

The transfer curve graph (chapter 4, display 4) shows the detected level on the horizontal axis and the output level on the vertical axis, with the same scale on both axes. This scale is set by **Gfx Lin/Db**, independently of the computation domain chosen by **Cv Lin/Db**. On it you can see:

- the **purple curve**, computed with exactly the same formulas as the audio: compressor, gate, lin/dB blend, limits, auto knee, and modulated ratios. When the auto knee or a modulation is active, it is constantly recomputed and you can see it move. It is capped at the top edge: a gain that would push the output past 0 dBFS is not visible;
- the **dark red triangle** below the diagonal, which is the zone where the output is quieter than the input;

- the **yellow** threshold line and the **white** line showing the output level reached at 0 dBFS (if **Gfx Lines** allows it). The white line is computed without the knee, without the gate, and without the GR Min and GR Max limits;
- the **green dots**: the detected level and the level actually obtained after the envelope follower, over roughly the last second. They take into account neither **Gr Amp** nor the opto designer (chapter 15).

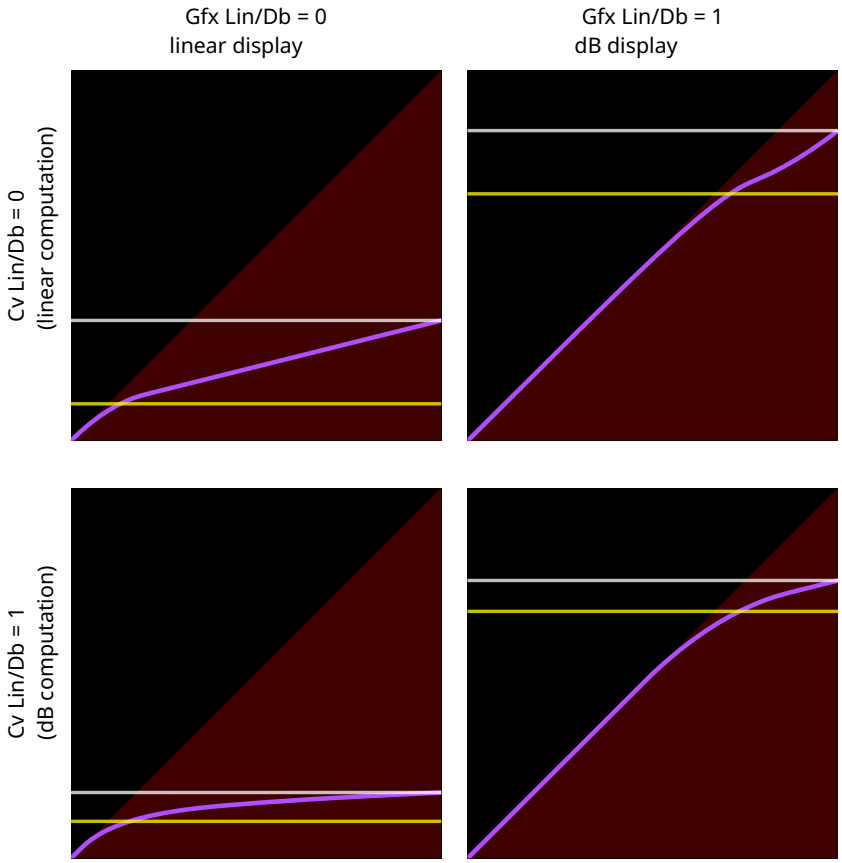


Figure 32 — The curve graph for a threshold of −20 dB, a ratio of 4:1, and a Knee of 24: top, the curve computed in linear; bottom, in dB; left, displayed in linear; right, in dB. The curve computed in linear and displayed in dB is not straight: it "bows" above the threshold.

In the history, two additional horizontal lines are always displayed: **GR Min** in dark red and **GR Max** in dark green. They are drawn at the height of the corresponding gain, on the same scale as the purple (target gain) and white (smoothed gain) lines. With the default settings (0 dB and −60 dB), they are pinned to the top and bottom edges.

Threshold, ratio, and knee

Threshold

main row tab

Range: −60 to 0 dB **Default:** −20 dB **ID:** `seuil`

Detected level above which the compressor starts reducing gain. It is compared to the level provided by detection (chapter 13): with the default detection, the instantaneous peak value of the signal; with RMS or smoothed detection, a lower level, which amounts to less compression for the same threshold.

In linear, the threshold is converted to an amplitude (−20 dB = 0.1; −6 dB ≈ 0.5); in dB, it is used as is.

Remember that **Headroom** (chapter 7) drives the signal harder into the compressor: raising Headroom by 6 dB amounts, as far as compression goes, to lowering the threshold by 6 dB.

In the graphs, the threshold is the **yellow** horizontal line.

Ratio

main row tab

Range: 0.25 to 42 **Default:** 2 **ID:** `ratio`

Compression ratio above the threshold. The code actually works with the **slope** 1/ratio: a ratio of 4 gives a slope of 0.25 (in dB or in amplitude, depending on **Cv Lin/Db**).

Special values:

- **42**, the maximum value, is treated as infinity: the slope is zero and the output stays pinned to the threshold. This is the setting for a limiter (with zero attack, chapter 12);
- **1** disables the compressor completely: its gain is 1 at all levels, even if **GR Min** is negative;
- **between 0.25 and 1**, the slope is greater than 1 (up to 4 for 0.25): above the threshold, the signal is **expanded upward**. This gain above 1 is only possible if **GR Min** is positive (see below); with GR Min at 0 dB, its default value, a ratio below 1 has **no effect**.

To get a negative ratio (the output decreases as the input increases), use **Ratio Sens** (CURVE tab).

In the graphs, the **white** horizontal line shows the output level a 0 dBFS signal would reach with this ratio.

The **knee** softens the curve's break at the threshold. In Comprendre, it's a parabolic transition (the classic "quadratic" knee), centered on the threshold: it starts at threshold − Knee/2 and ends at threshold + Knee/2. Within this zone, the slope moves gradually from 1 to that of the ratio; outside it, the curve is unchanged.

Knee

main row tab

Range: 0 to 240 **Default:** 0 **ID:** `genou`

Knee width, centered on the threshold. At 0 (default), the knee is hard: the curve breaks sharply at the threshold.

Its unit depends on the computation domain:

- **in dB**, it's a width in decibels. Knee = 12 with a threshold of −20 dB spreads the knee from −26 to −14 dB. The control's range (up to 240) goes far beyond the useful dynamic range: with this same threshold, a Knee of 40 already takes the knee all the way up to 0 dBFS, and a Knee of 80 takes it all the way down to −60 dBFS;
- **in linear**, it's a width in **amplitude**, equal to Knee/120: Knee = 120 corresponds to the entire amplitude range from 0 to 1. With a threshold of −20 dB (0.1), Knee = 6 spreads the knee from amplitude 0.075 to 0.125, that is, from −22.5 to −18.1 dB. The linear knee is therefore symmetrical in amplitude, but **not** in decibels: its lower part is longer than its upper part (2.5 dB below the threshold versus 1.9 dB above).

The two versions have the same name but very different effects for the same value: change **Cv Lin/Db** and you'll generally have to readjust the Knee.

> **The soft knee, from tubes to 'Over Easy'**
>
> On the first compressors, the soft knee wasn't a control: it came for free. Variable-mu tube compressors and optical compressors naturally ease into compression, like the 1956 Gates Sta-Level or the LA-2A, whose ratio, "fixed at 4:1," actually varies with the source. The Neve 2254 (1969), a diode-bridge design, spreads its transition over about ten decibels. With the VCA, which is very linear, the knee becomes a design choice: the 1976 dbx 160 has a hard knee, and it was in 1978, on the dbx 165, that the company marketed its soft knee under the name "Over Easy." The API 2500 (2002) would later offer three, hard, medium, and soft, and the Distressor gives its 2:1 and 3:1 ratios "parabolic" knees, 15 dB wide for the 2:1; that's also the shape of Comprendre's knee. As for the Manley Variable Mu, its manual describes a knee that softens as limiting increases: that's the idea behind the auto knee, presented later.

Knee "shaving" on the linear curve

In linear, the knee width is an amplitude, whereas the threshold can be very low. The knee starts at the amplitude threshold − Knee/240: as soon as Knee reaches 240 times the threshold (in amplitude), this starting point falls to zero. The knee then starts at silence, and its entire lower part is "shaved off."

Threshold	−6 dB	−10 dB	−20 dB	−30 dB	−40 dB	−60 dB
Knee at which the knee starts at silence	120	76	24	7.6	2.4	0.24

Beyond this value, the behavior changes radically. For the lowest levels, the knee formula gives a negative output: the gain collapses down to the **GR Max** floor (−60 dB by default). Low levels are then reduced **more** than those near the threshold, and the wider the knee, the higher this zone ex-

tends. With a threshold of −20 dB, a ratio of 4:1, and Knee = 48, everything detected below −38.6 dBFS is lowered by 60 dB: the compressor partly behaves like a gate. With a threshold of −40 dB, Knee = 6 is enough for everything below −53 dBFS to suffer the same fate.

In dB, nothing of the sort happens: the knee is symmetrical in decibels, and its reduction increases steadily from its starting point.

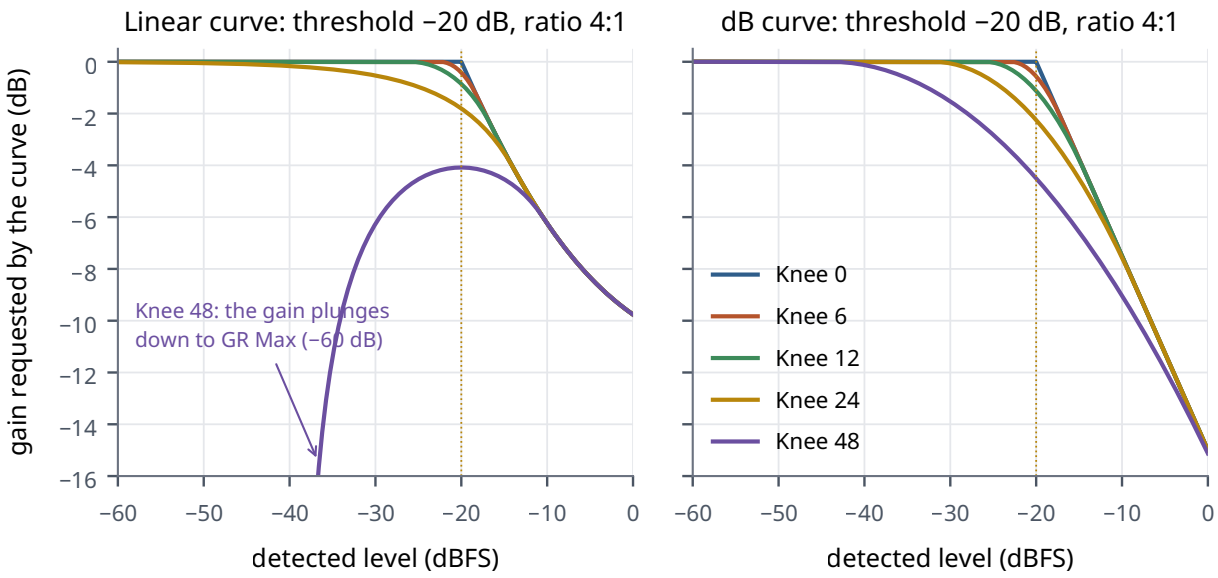


Figure 33 — Gain called for by the curve depending on Knee, in linear (left) and in dB (right). In linear, the lower part of the knee is shortened, and beyond Knee = 24 (for this −20 dB threshold) low levels get reduced in turn.

To be confirmed by François

On the linear curve, when Knee exceeds 240 times the threshold in amplitude (24 for a threshold of −20 dB, 2.4 for −40 dB), the knee formula gives a negative output at the lowest levels, and the gain drops to the GR Max floor: weak signals are then heavily attenuated, as if by a gate. Is this intended, and is this the “shaving” mentioned in the video?

Choosing the Knee

In dB, a Knee of 6 to 12 dB gives a conventional soft knee. In linear, keep Knee well below the value in the table above for your threshold, unless you’re specifically after the “shaving” effect.

Going further

In both domains, let x be the level (amplitude or dB), T the threshold, K the knee width, and r = 1/ratio the slope.

- If $x > T + K/2$: output = $T + (x - T) \cdot r$.
- If $x \leq T - K/2$: output = x.
- Otherwise: output = $x + (r - 1) \cdot (x - T + K/2)^2 / (2K)$.

In linear, the gain is output / x, with K = Knee/120; in dB, the reduction is x − output, with K = Knee. The gate uses the same knee, flipped: above $T + K/2$, output = x; below $T - K/2$, output = $T + (x - T) \cdot$ Ratio Gate; in between, output = $x + (1 - \text{Ratio Gate}) \cdot (x - T - K/2)^2 / (2K)$.

Internally, dB levels are offset by 60 dB and limited to −60 dBFS: a detected level below −60 dBFS is treated as −60 dBFS by the dB curve.

♪ In the studio: knee width

A hard knee brings compression in all at once, right at the threshold: that's what you want for a limiter or for compression that's meant to be heard. A wide knee eases it in, a few decibels before the threshold, and makes it far less noticeable on a vocal, an acoustic guitar, or a bus. On a vocal, with the dB curve, compare **Knee** at 0 and then at 12: the beginnings of phrases get squeezed less abruptly.

The presets go much further. **waves_la2a**, built by imitating Waves' CLA-2A (an emulation of the LA-2A), sets **Knee** to 25 dB. **arturia_tube-sta**, an imitation of Arturia's Comp TUBE-STA (the Gates Sta-Level), pairs a **Ratio** of 42 with a **Knee** of 38.29 dB: its curve moves gradually from 1:1 to infinity over nearly 40 dB. **klang_mjuc_mk1** (67.42 dB) and **waves_rvox** (88 dB) widen the knee so much that the ratio keeps increasing over the whole useful range. **ao_dbcomp**, based on a dbx 160–style plugin described as having an "Over Easy" soft knee, sets **Knee** to 25 on the linear curve, just below the shaving limit for its threshold of −19.28 dB. Load one of these presets and gradually lower **Threshold**: the compression sets in progressively, with no audible threshold.

The auto knee

The auto knee varies the knee width with the **gain reduction**: the harder the compressor works, the wider the knee gets (or the narrower, with a negative value). The reduction used is the one the curve would impose with a hard knee, smoothed by a follower whose time constant is set by **Knee Auto Secs**.

Below the threshold, this reduction is zero: the knee then returns to its base width (**Knee**) at the pace set by Knee Auto Secs. During a loud passage, the knee widens; when the level comes back down, it stays wide for a while. The quieter notes that follow a loud passage, around the threshold, are then compressed more gently but earlier (from the bottom of the knee).

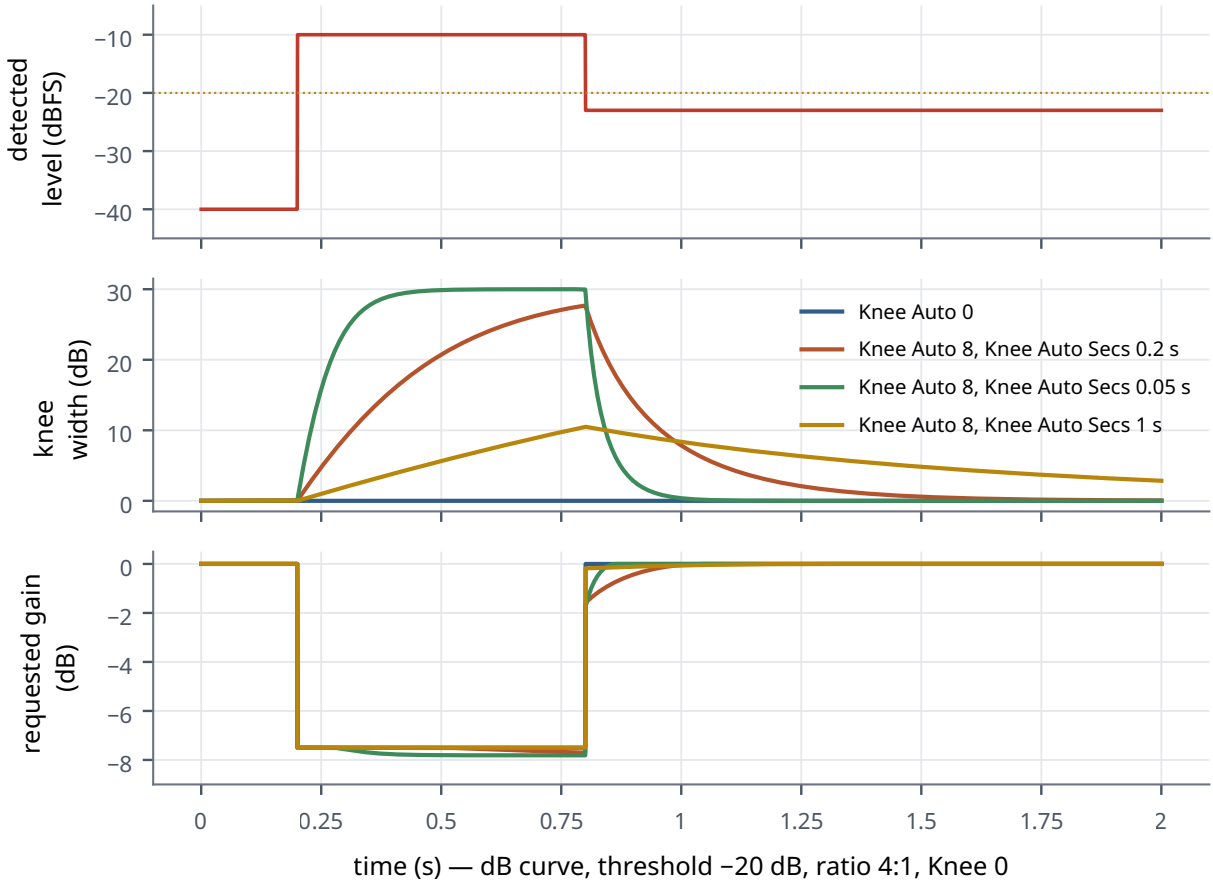


Figure 34 — Auto knee in dB (threshold −20 dB, ratio 4:1, Knee 0, Knee Auto 8). During the passage at −10 dBFS, the knee widens up to 30 dB; after the return to −23 dBFS, just below the threshold, it closes at the pace set by Knee Auto Secs, and the −23 dBFS level gets a slight reduction as long as the knee stays wide.

Knee Auto

CURVE tab

Range: −240 to 240 **Default:** 0 **ID:** `auto_genou`

Amount of automatic widening of the compressor's knee. At 0 (default), the knee is fixed.

The effect depends on the computation domain:

- **in dB**, the knee widens by Knee Auto/2 dB per decibel of smoothed reduction. Knee Auto = 1 and 6 dB of reduction add 3 dB of knee; Knee Auto = 8 and 7.5 dB of reduction add 30 dB;
- **in linear**, the knee widens in amplitude by Knee Auto/20 × (1 − g), where g is the smoothed gain (0.5 for 6 dB of reduction). Expressed in the unit of the Knee control, this amounts to adding 6 × Knee Auto × (1 − g): Knee Auto = 2 with 6 dB of reduction is equivalent to adding 6 to Knee.

A **negative** value narrows the knee as the reduction increases. If the total width becomes negative, the knee zone disappears and you're back to a hard knee (as long as **GR Min** stays at 0 dB; with a positive GR Min, a slight gain can appear just below the threshold).

In linear, watch out for the "shaving" described above: an overly generous auto knee can, during loud passages, exceed the limit of 240 times the threshold, and make the gain of the low levels that follow plummet.

The auto knee uses the ratio as modulated by **mod ratio** (chapter 23), if any.

Knee Auto Secs

CURVE tab

Range: 0 to 2 s **Default:** 0.2 s **ID:** `plie`

Time constant of the follower that smooths the reduction used by **Knee Auto**: the knee covers about 63% of the way to its new width in Knee Auto Secs seconds, both when it widens and when it closes. At 0, there is no smoothing: the knee follows the reduction instantly.

This time is independent of the attack and release settings (including their "altitudes," chapter 12). It only has an effect if **Knee Auto** is not 0.

> **A word from the author**
>
> The auto knee is one of the controls François uses most when imitating vocal compressors: it's often what got him close to the result he was after. He finds that a time constant of 200 ms, the default value, works well, and he reminds you to choose the domain (linear or dB) deliberately, because the effect is very different.

To be confirmed by François

In the video, the auto knee display seems incorrect when the channels are unlinked. The code displays the average of the two channels' knees only if Channels Unlink is exactly 1 (for an intermediate value, it displays the linked knee); moreover, on the dB curve, the per-channel auto knees start from an improperly initialized state after each plugin reset, which gives an almost hard knee for a few time constants. Is this the cause of the defect observed?

♪ In the studio: a knee that follows the vocal

This is François's advice for vocals: let the knee move with the singer's level. Start with **Cv Lin/Db** at 1, **Knee** at 6, **Knee Auto** at 2, and **Knee Auto Secs** at 0.2 s. During a chorus sung loud, the knee widens; when the voice drops back to a softer level, it stays wide for a while, and the syllables that follow go into compression earlier but more gently. The ear often perceives a more even vocal, one that stays up front without being squashed. Try lengthening **Knee Auto Secs** on a slow, legato vocal, and shortening it on a fast or highly articulated delivery. Among the presets, **bx_townhouse** uses a subtle auto knee (**Knee Auto** 0.2, with a time of 0.127 s) and **pa_elysia** a very pronounced one (24). **blackrooster_omnitec436**, an imitation of Black Rooster Audio's OmniTec-436C (an Altec 436C, a tube compressor), does the opposite: with **Knee Auto** at −14.53, its knee tightens as the compression intensifies, which hardens the curve on loud passages.

Reduction limits: GR Max and GR Min

These two controls clamp the gain of each curve (compressor and gate), before the envelope follower. They are expressed as gains in dB: 0 dB = unity gain, −6 dB = 6 dB of reduction, +6 dB = 6 dB of amplification.

- **GR Max** is the gain **floor**: the reduction can't exceed this value.
- **GR Min** is the gain **ceiling**: the gain can't exceed this value. At 0 dB, it prevents any gain above 1; when positive, it allows it; when negative, it imposes a minimum reduction.

GR Min is applied **after** GR Max: if the two contradict each other (GR Max higher than GR Min), GR Min wins.

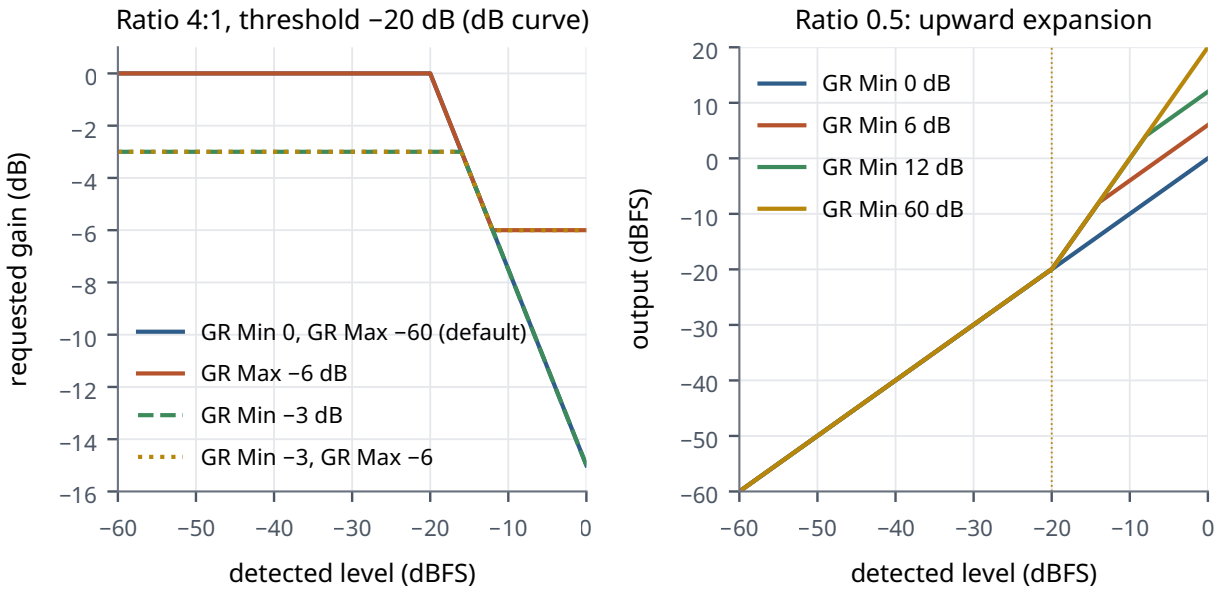


Figure 35 — Left: GR Max and GR Min on a 4:1 compressor in dB. Right: a ratio of 0.5 (upward expansion) only has an effect if GR Min is positive, and this gain is capped by GR Min.

GR Max

CURVE tab

Range: −60 to 60 dB **Default:** −60 dB **ID:** `gr_max`

Maximum gain reduction allowed by the curve: the gain of each curve never goes below this value. At −2 dB, the curve never calls for more than 2 dB of reduction, whatever the level. At −60 dB (default), the limit practically never gets in the way.

A positive value imposes a minimum gain above 0 dB, but only if **GR Min** is at least as high (otherwise GR Min wins).

On the dB curve, the −60 dB limit has a peculiarity: a reduction that reaches it gives a gain of zero, meaning complete silence, whereas in linear it does give −60 dB.

In the history, GR Max is the **dark green** horizontal line.

GR Min

CURVE tab

Range: −60 to 60 dB **Default:** 0 dB **ID:** `gr_min`

Maximum gain allowed by the curve: the gain of each curve never rises above this value.

- **0 dB** (default): no gain above 1 is possible. A ratio below 1 or a Ratio Gate below 1 then has no effect.
- **Positive**: allows amplification up to this value. This is essential for upward expansion (ratio between 0.25 and 1) and for the gate's upward compression (Ratio Gate below 1).
- **Negative**: imposes a **minimum** reduction, at all levels, even below the threshold. With GR Min = −3 dB, the curve calls for at least 3 dB of reduction at all times.

Two clarifications. First, GR Min only applies to an active curve: with Ratio = 1 (and Ratio Gate = 1), it has no effect. Second, it clamps the compressor and the gate separately: if both are active, a GR Min of −3 dB can give up to 6 dB of minimum reduction in total. For the same reason, the product of the two curves can go below GR Max.

In the history, GR Min is the **dark red** horizontal line.

A word from the author

GR Max is the "classic" maximum reduction control: at 2 dB, the curve will never reduce by more than 2 dB. GR Min is its counterpart on the other side: since it's at 0 dB by default, you have to make it positive for the curve to be allowed to exceed unity gain.

To be confirmed by François

In the videos, a positive GR Min is presented as necessary for negative ratios. In the code, a negative ratio (Ratio Sens at −1) never produces a gain above 1 and works with GR Min at 0; it's ratios below 1 (and Ratio Gate below 1) that need a positive GR Min. Were these the ratios being referred to?

♪ In the studio: capping bus compression

On a mix bus, "glue" often comes down to very little: a few decibels of reduction, no more. **GR Max** guarantees that. Set it to −2 or −3 dB, then lower the threshold without fear: the compressor works across the whole track, but the curve will never call for more, even on the loudest cymbal crash. In mastering, a **GR Max** of −1 to −2 dB gives the same assurance to density compression. A negative **GR Min** is mostly useful for imitation, imposing a permanent reduction: **softube_cl1b** sets it to −0.6 dB, **ircam_purecomp** to −6 dB.

Ratios below 1 and negative ratios

The **Ratio** knob goes from 0.25 to 42:

- from 1 to 42, it's ordinary compression;
- from 0.25 to 1, it's **upward expansion** above the threshold: a ratio of 0.5 puts out 2 dB for each decibel above the threshold (in dB). It only acts if **GR Min** is positive, and it is capped by it.

The **Ratio Sens** control can also reverse the direction of the slope: with a **negative** ratio, the output **decreases** as the input rises above the threshold. It's an "overcompression" effect, which pushes the peaks down below the threshold.

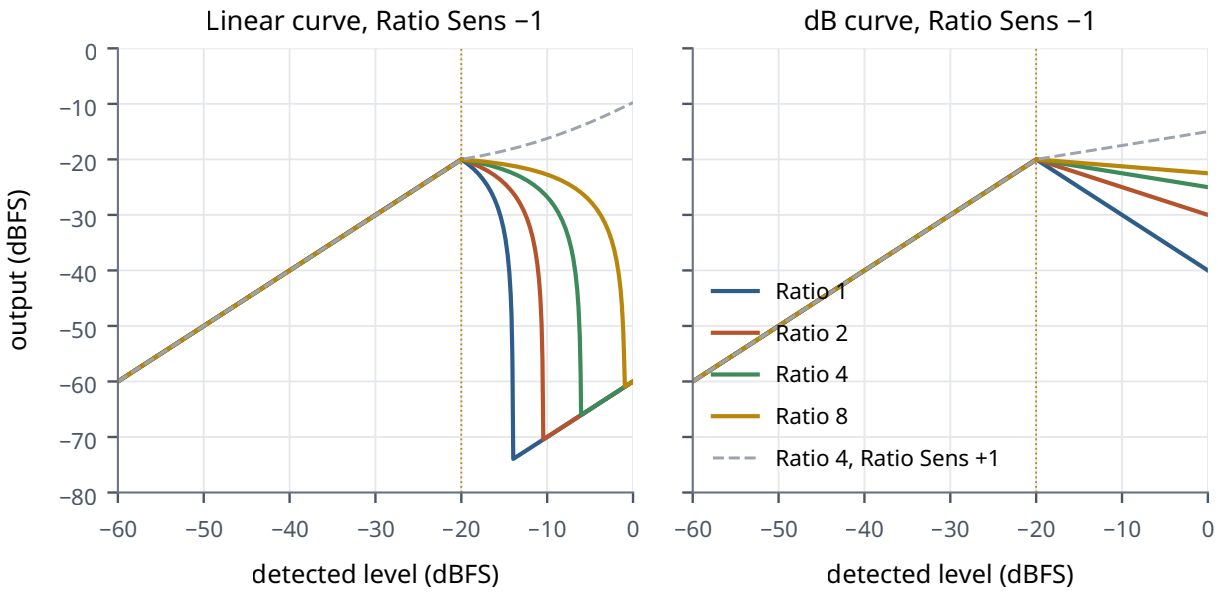


Figure 36 — Ratio Sens at −1: above the threshold, the output goes back down. In linear, the output amplitude reaches zero, then the gain stays on the GR Max floor; in dB, the output goes down in a straight line. Dotted gray: the ordinary ratio of 4.

Ratio Sens

CURVE tab

Range: −1 to 1 **Default:** 1 **ID:** `ratio_sens`

Direction of the compressor's slope. This control has only three positions:

- **+1** (default): positive slope, ordinary compression. If the ratio is modulated (**mod ratio**, chapter 23), the modulation can't make the slope negative: at worst, it brings it to zero (∞: 1);
- **0**: positive slope, like +1, but the modulation is allowed to make it negative (down to −4). Without modulation, 0 and +1 sound exactly the same;
- **−1**: negative slope, equal to −1/ratio. Above the threshold, the output decreases as the input increases.

With −1 on the dB curve, a threshold of −20 dB, and a ratio of 4, a signal at −10 dBFS comes out at −22.5 dBFS and a signal at 0 dBFS at −25 dBFS. On the linear curve, the output amplitude decreases until it reaches zero for an input equal to (1 + ratio) times the threshold (−6 dBFS for a threshold of −20 dB and a ratio of 4); beyond that, the gain stays on the **GR Max** floor. Note that here a ratio of 1 is not neutral: the slope is −1, and the output mirrors the input around the threshold.

A negative ratio never calls for a gain above 1: it doesn't need a positive **GR Min**.

A word from the author

Since the maximum ratio (42) already represents infinity, another control was needed to go beyond it: that's the role of Ratio Sens. François also designed it with modulation in mind: he wants to be able to decide whether or not a modulated ratio is allowed to go negative.

To be confirmed by François

With Ratio Sens at −1 and mod ratio active, the code computes the modulation from the positive slope 1/(Ratio + mod ratio): when the modulation is at its resting level, the slope becomes positive again (for example, Ratio 2 and mod ratio +2 give a slope of +1/4 instead of −1/4). Is this intended? It could explain the "weird" behavior noticed in the video.

From the limiter to 'overcompression'

Compression started out as limiting. In 1937 came the first commercial peak limiters, the Western Electric 110A, followed by the RCA 96-A: simple tools to keep radio transmitters from overmodulating. Today, limiting is generally said to start at a ratio of about 10:1, and "brickwall" limiting runs from 20:1 to infinity, with a very fast attack. You still have to be able to reach infinity: in theory, a feedback compressor, which measures its own output, can't, whereas a feed-forward compressor gets there easily and can even go beyond, all the way to the "overcompression" of negative ratios. A few units have put it within engineers' reach: the elysia alpha compressor (2006), whose feedback mode tops out at 2.5:1 but whose feed-forward mode goes as far as negative ratios, or the SSL The Bus+ (2022). In Comprendre, the **Ratio** of 42 is that infinity, and **Ratio Sens** opens the door to what lies beyond.

♪ In the studio: negative ratios for sound design

With **Ratio Sens** at −1, the harder a hit, the lower it comes out. On a drum loop, with the dB curve, set the threshold just below the medium hits, with a **Ratio** of 2 to 4 and short times: the hardest hits sink, while the ghost notes and the ring of the shells, which stayed below the threshold, come to the front. The loop takes on a sucked-in, almost inside-out feel. On a pad or a texture, a long release turns each accent into a dip. None of the 80 presets uses this control: it's still uncharted territory, waiting to be explored. Keep an eye on the level, and check on the curve graph that the output really does go back down above the threshold.

The gate

The gate (or, more precisely, downward expander) acts **below** its threshold, where the compressor doesn't. It has its own threshold, ratio, and knee, and its own auto knee. Its gain is multiplied with the compressor's **before** the envelope follower, so it uses the same attack and release times. When the gate closes, the gain goes down at the attack speed; when it reopens, it comes back up at the release speed.

Watch out: the gate's ratio convention is not the compressor's. **Ratio Gate** is directly the slope below the threshold. With Ratio Gate = 4, each decibel below the gate threshold becomes 4 dB below that threshold (in dB). Ratio Gate = 1 is neutral (gate inactive, the default). And 42 is **not** infinity: it's a slope of 42, which in practice still amounts to closing almost completely.

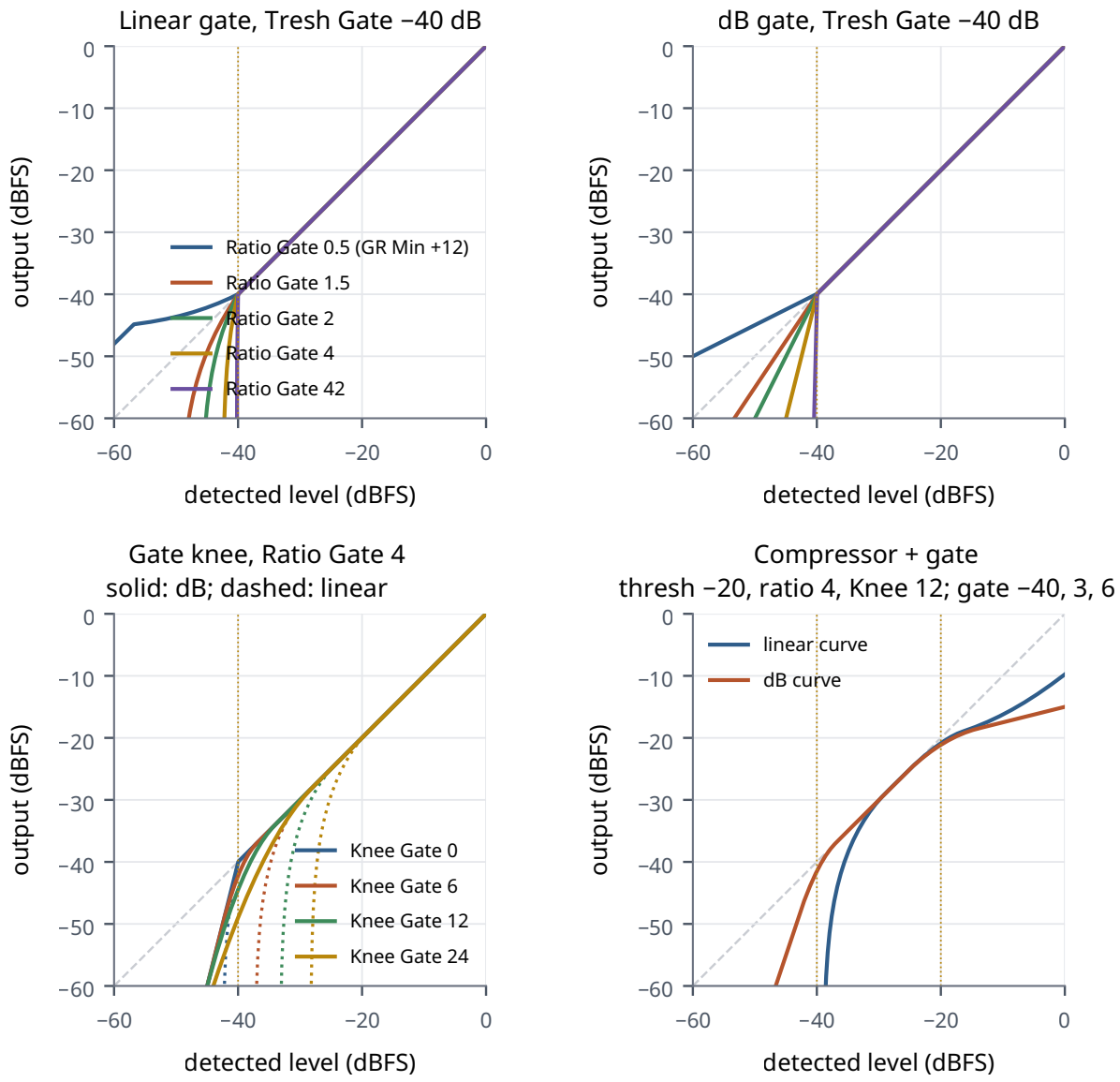


Figure 37 — The gate. Top: various Ratio Gate values, in linear and in dB (Tresh Gate −40 dB); in linear, the output drops to zero at some distance below the threshold, then the gain stays on the GR Max floor. Bottom left: the gate knee, in dB (solid line) and in linear (dotted). Bottom right: compressor and gate combined.

Tresh Gate

CURVE tab

Range: −60 to 0 dB **Default:** −20 dB **ID:** `seuil_porte`

Gate threshold: the gate acts on detected levels **below** this value. As with the compressor, it is converted to an amplitude in linear and used as is in dB.

The gate threshold is independent of the compressor threshold. It's usually set well below it, to deal with background noise or note tails, but nothing prevents you from setting it above.

Ratio Gate

CURVE tab

Range: 0.25 to 42 **Default:** 1 **ID:** `ratio_porte`

Slope of the gate below its threshold. Unlike **Ratio**, this is not an “input:output” ratio but the slope itself:

- **1** (default): gate inactive;
- **greater than 1**: downward expansion. In dB, 2 means that 10 dB below the threshold becomes 20 dB below the threshold. In linear, the output amplitude drops to zero for an input equal to (1 − 1/Ratio Gate) times the threshold: with a threshold of −40 dB, at −46 dBFS for 2, at −42.5 dBFS for 4. Below that, the gain stays on the **GR Max** floor: the linear gate is therefore a “hard” gate;
- **42**: a slope of 42, practically closed from the threshold down, without being infinite;
- **less than 1** (down to 0.25): **upward compression** below the threshold, which brings up low levels. Since it calls for a gain above 1, it only acts if **GR Min** is positive, and it is capped by it.

The **GR Max** and **GR Min** limits apply to the gate’s gain separately from the compressor’s.

Knee Gate

CURVE tab

Range: 0 to 240 **Default:** 0 **ID:** `genou_porte`

Width of the gate’s knee, centered on **Tresh Gate**, with the same units as **Knee**: a width in dB on the dB curve, a width in amplitude equal to Knee Gate/120 on the linear curve. The transition is the same parabolic knee, flipped.

In linear, the remark made for the compressor applies even more here, since the gate threshold is usually low: with Tresh Gate = −40 dB (amplitude 0.01), Knee Gate = 12 extends the top of the knee up to −24 dBFS and eliminates its lower part. For a well-controlled gate knee, use the dB curve.

Knee Gate Auto

CURVE tab

Range: −240 to 240 **Default:** 0 **ID:** `auto_genou_porte`

The gate’s auto knee: its width increases with the reduction **caused by the gate**, smoothed by **KG Auto Secs**. The principle is the same as **Knee Auto**, but with a sensitivity four times lower: in dB, the knee widens by Knee Gate Auto/8 dB per decibel of reduction; in linear, by Knee Gate Auto/80 × (1 − g) in amplitude.

A negative value narrows the knee.

KG Auto Secs

CURVE tab

Range: 0 to 2 s **Default:** 0.2 s **ID:** `plie_porte`

Time constant of the smoothing used by **Knee Gate Auto**, with the same definition as **Knee Auto Secs** (63% of the way in that time; 0 = no smoothing). No effect if Knee Gate Auto is 0.

mod gate

The **mod gate** control adds directly to Ratio Gate an amount that depends on the frequency content of the signal (chapter 23). It can therefore make the gate active even when Ratio Gate is 1.

A simple noise gate

Set **Cv Lin/Db** to 1, **Ratio** to 1 if you don't want to compress, **Tresh Gate** just above the noise floor (for example −50 dB), and **Ratio Gate** between 2 and 4. The gate closes at the speed of **Attack** and re-opens at the speed of **Release**. A **Knee Gate** of a few decibels softens the onset.

Amplifying the reduction: Gr Amp

Gr Amp lives in the CURVE tab, but it doesn't act on the curve: it acts at the very last moment, when the smoothed gain (the envelope) multiplies the target. It multiplies the **linear reduction**, that is, the gap between the envelope and 1:

$$\text{applied gain} = 1 + (\text{envelope} - 1) \times \text{Gr Amp}$$

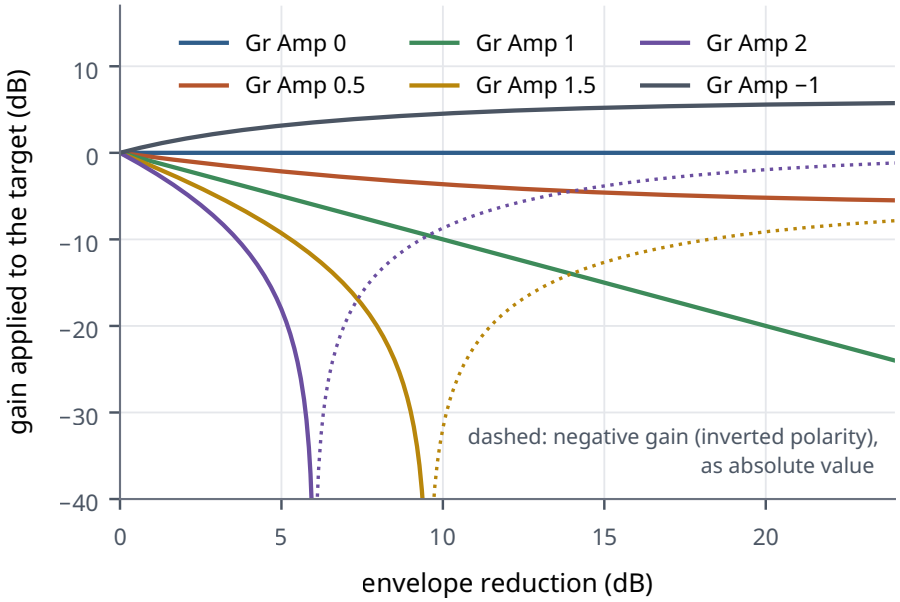


Figure 38 — Gain applied to the target as a function of the envelope's reduction, for various Gr Amp values. Beyond a certain amount of reduction, values above 1 give a negative gain (dotted: its absolute value).

Gr Amp

CURVE tab

Range: −8 to 8 **Default:** 1 **ID:** `env_amp`

Factor applied to the gain reduction at the moment it multiplies the target.

- **1** (default): normal operation.
- **0**: no compression at all: the applied gain is 1 at all times.
- **Between 0 and 1**: attenuated reduction. At 0.5, an envelope of −6 dB (0.5) gives a gain of 0.75, that is, −2.5 dB.
- **Greater than 1**: amplified reduction, very quickly. At 2, an envelope of −6 dB gives a gain of 0.002, that is, −52 dB. As soon as the reduction exceeds a certain value (6 dB for Gr Amp = 2, 9.5 dB for 1.5, 2.5 dB for 4), the gain becomes **negative**: the target's polarity is then inverted.
- **Negative**: the reduction is turned into amplification. At −1, a reduction of 6 dB becomes a gain of +3.5 dB.

Gr Amp applies separately to the linked gain and to the per-channel gains (**Channels Unlink**). It does **not** apply to the envelope as seen by the graphs (white envelope, green dots), the second dynamic EQ (chapter 18), the bias (chapter 19), or the opto designer (chapter 15): these use the envelope before Gr Amp.

To be confirmed by François

With Gr Amp above 1, the applied gain becomes negative as soon as the envelope drops below 1 − 1/Gr Amp (6 dB of reduction for Gr Amp = 2), which inverts the target's polarity; the code even carries the comment "sécurisation courant < 0 à faire" (a to-do note about guarding against a negative value). Is this intended, or should the gain be clamped at zero?

Example settings

Conventional compressor. Cv Lin/Db = 1, **Threshold** at −20 dB, **Ratio** 3 to 4, **Knee** 6 to 12. This is the behavior described in most manuals, with a soft knee that is symmetrical in dB.

Gentle linear compression. Leave **Cv Lin/Db** at 0 and compare by ear with the same ratio value in dB: the linear curve lets through more of the peaks that go well above the threshold, while acting firmly just above it. For comparable reduction at high levels, you need a considerably higher ratio.

Limiter. Ratio = 42, zero attack (chapter 12). The pre-clipper (chapter 7) can clip lightly beforehand.

Capped reduction. A **GR Max** of −3 to −6 dB guarantees that the compressor will never reduce by more than this value, even on the loudest peaks. It's a simple way to keep the character of light compression on a bus.

Upward expansion. Ratio = 0.5 and **GR Min** = +6 dB: above the threshold, the peaks are raised, by 6 dB at most.

Upward compression of low levels. Ratio = 1, **Ratio Gate** = 0.5, **Tresh Gate** at −40 dB, **GR Min** = +12 dB, in dB: below −40 dBFS, low levels are raised, all the more so the lower they are (10 dB at −60 dBFS), without ever exceeding GR Min's 12 dB.

Vocal with auto knee. Cv Lin/Db = 1, **Knee** 6, **Knee Auto** 1 to 4, **Knee Auto Secs** 0.2 s: the knee widens during loud phrases and softens the way the following syllables go into compression.

Starting ideas

- **Hear the two curve logics.** Load **sonible_smartcomp2**, one of the simplest presets in decibels, on a drum bus, then switch **Cv Lin/Db** from 1 to 0: with the same settings, the linear version lets the accents stand out more.
- **A hard gate on toms or a noisy guitar.** Start from **playfair_grading**, which pairs very gentle compression (**Ratio** 1.28, **Knee** 2.16 dB, threshold at −15 dB) with a gate that is almost closed below its threshold (**Ratio Gate** 42, **Knee Gate** 8.05 dB); adjust **Tresh Gate**, left at −20 dB, to suit your source.
- **A soft limiter.** Take **arturia_tube-sta** and lower **Threshold**: its 38.29 dB knee gradually raises the ratio all the way to infinity.
- **An effect.** Start from the default setting, set **Cv Lin/Db** to 1 and **Ratio Sens** to −1, and look for the threshold where the dynamics flip.

CHAPTER 11

The Envelope Follower

The transfer curve (chapter 10) tells you, at every sample, what gain should be applied: this is the **requested gain**, the purple line in the graphs. Applied as is, this gain would follow every half-cycle of the waveform and cause heavy distortion. The **envelope follower** smooths it: it chases the requested gain at a certain speed, faster or slower depending on whether it has to go down (the **attack**) or come back up (the **release**). The result is the gain actually applied, the white line in the graphs.

The envelope follower is the motion of compression: the way the gain dives on an attack, stays down, then recovers. It is what makes compression breathe with the music or pump against it, what makes a bus feel held by a gentle hand or clamped in a vise. Two compressors set to the same times can sound very different depending on how this motion is computed, and that is often where their personality hides. Here Comprendre brings together the major "schools" François came across while studying the compressors on the market and the literature, including an envelope whose speed varies with level, as on tube compressors, and makes them all blendable. If you are trying to emulate a unit, this is one of the first places to look; if you are mixing, it is a palette of ways to breathe.

In an ordinary compressor, the envelope follower boils down to two times, attack and release. Comprendre turns it into a full module, controlled from two tabs:

- the **ENV (cv)** tab, described in this chapter, chooses **which type** of envelope is used: linear or decibel computation, "branching" or "decoupled" law, hold, the Mu envelope with variable speed, the Gamma envelope, and blends of these;
- the **A/R (cv)** tab and the **Attack** and **Release** knobs on the main row set **the times** and the shape of the curves (chapter 12).

The "(cv)" suffix means that this follower sits **after the curve**, the transfer curve: it is the compressor's "classic" follower (see the figure of the computation stages in chapter 6). Comprendre contains three other followers built exactly on the same model, with the same 14 settings, in the **ENV (in)**, **ENV (la)** and **ENV (bs)** tabs. This chapter describes the settings for the ENV (cv) tab; the section "The four envelope followers" explains what changes for the other three.



Figure 39 — The ENV (cv) tab: the 14 settings of the envelope follower placed after the transfer curve. The times are those of the Attack and Release knobs on the main row.

With the default settings (all of the tab's knobs at 0, except the Mu settings at 1), the follower is as simple as it gets: a **classic** envelope, computed **linearly**, **branching**, with no hold. Comprendre then behaves like most digital compressors.

How it works: covering a fraction of the distance

At every sample, the envelope compares its current value with the requested gain:

- if it is **above** the requested gain, the compressor must reduce further: this is an **attack**;
- otherwise, it rises back toward the requested gain: this is a **release**.

In both cases, the envelope covers a **fixed fraction of the distance** separating it from its target. A fraction close to 1 gives nearly instantaneous tracking; a tiny fraction, very slow tracking. This fraction, called the **coefficient**, is computed from the attack or release time and the **altitude** (chapter 12). Covering the same fraction of the remaining distance each time produces an exponential curve: fast at first, then slower and slower as it nears the target.

> **Going further**
>
> In logarithmic mode (the default), the classic envelope follows the law e ← e + (g − e) · c, where g is the requested gain and c the coefficient, c = 1 − a^(1/(T · fs)); T is the set time (attack or release), a the altitude (1/e by default) and fs the internal sample rate, oversampling included. After time T, exactly the fraction a of the distance remains to be covered: 36.8% by default; in other words, 63.2% of the way has been covered. Times therefore stay accurate regardless of oversampling. Chapter 12 details this calculation.

The ENV (cv) follower is only active if **Attack** or **Release** is greater than 0 (or if one of the Env Offset, Env Pow Unpow or Gr|Env Pow settings in the GR (cv) tab is not at its neutral value, chapter 15). With Attack = Release = 0, the requested gain is applied directly, with no smoothing at all.

Six envelopes in one

Comprendre's follower is not one envelope but **up to six envelopes computed in parallel** on the same target, then blended:

- three families: the **classic** envelope, the **Mu** envelope (whose speed depends on level) and the **Gamma** envelope (an algorithm with an S-shaped transition);
- each computed in two domains: **linear** (the gain, from 0 to 1) and **decibels**.

Three knobs **crossfade** between the outputs of these envelopes: **Env->Mu** between classic and Mu, **Env/Mu->Gamma** between the previous result and Gamma, **EvMu Lin/Db** between the linear domain and the decibel domain. Two other settings change **how the classic and Mu envelopes evolve**: **EvMu Br/Cp** (branching or decoupled law) and **Hold branch** (hold).

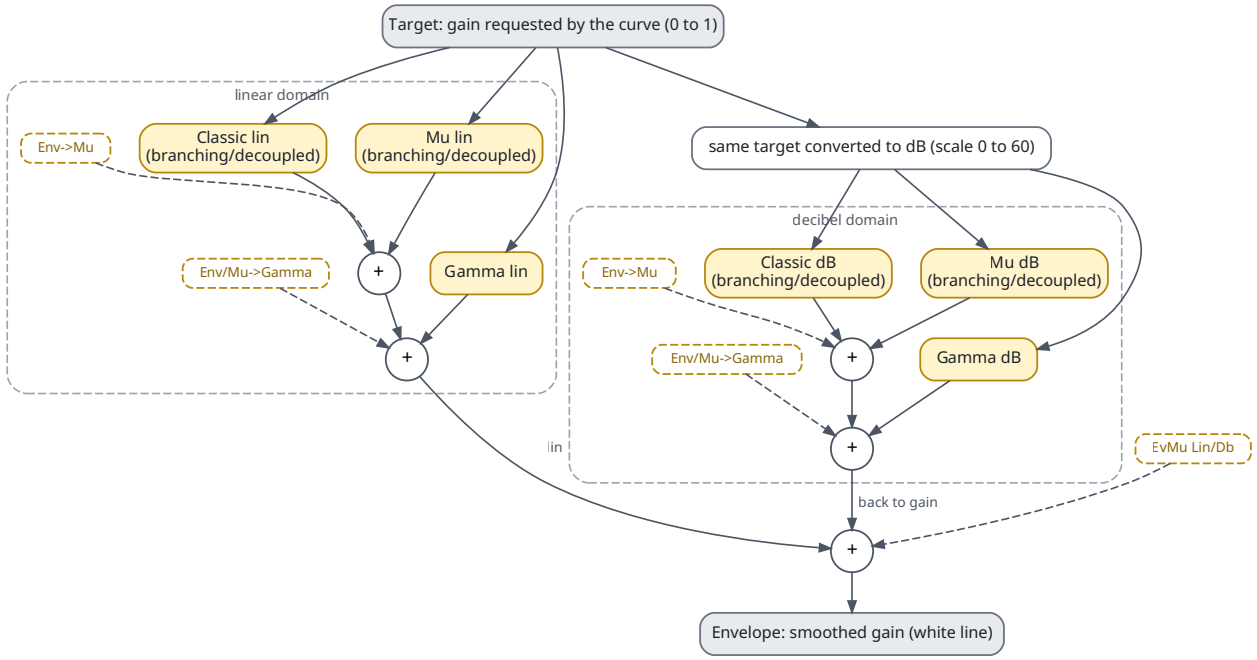


Figure 40 — The six envelopes of the follower and the three blends. The dashed settings weight the outputs; EvMu Br/Cp and Hold branch act inside the classic and Mu envelopes.

> **Going further**
>
> Writing E, M and Γ for the classic, Mu and Gamma outputs in linear, E', M' and Γ' for those of the decibel domain (converted back to gain), and d, m, γ for the values of EvMu Lin/Db, Env->Mu and Env/Mu->Gamma, the final envelope is: $(1 - d) \cdot [\, (1 - \gamma) \cdot ((1 - m) \cdot E + m \cdot M) + \gamma \cdot \Gamma \,] + d \cdot [\, (1 - \gamma) \cdot ((1 - m) \cdot E' + m \cdot M') + \gamma \cdot \Gamma' \,]$. The weights always add up to 1. A negative result is brought back to 0.

> **An envelope with zero weight is not computed**
>
> To save CPU, an envelope whose weight is zero is not computed at all: with the default settings, only the linear classic envelope runs. An envelope that is not computed keeps the last value it had. If you reactivate it during playback, it restarts from that stale value, which can cause a brief jump in gain while it catches up with the target.

Linear or decibels

The envelope can chase the requested gain as a linear value (a gain from 0 to 1) or in decibels. In the second case, Comprendre converts the requested gain to a scale from 0 to 60, where 60 corresponds to 0 dB (no reduction) and 0 to −60 dB, smooths this value, then converts it back to gain.

The difference comes from the fact that "covering a fraction of the distance" does not mean the same thing in the two domains. Take a 20 dB reduction (gain 0.1) with the default settings. After the attack time:

- in **linear**, the envelope has covered 63% of the way between 1 and 0.1: it is at 0.43, or −7.3 dB;
- in **decibels**, it has covered 63% of the 20 dB: it is at −12.6 dB.

On release, it is the opposite: starting from −20 dB, the linear envelope has risen to −3.5 dB after the release time, the decibel envelope only to −7.4 dB. In other words, with equal settings, decibel computation **attacks more firmly** on large reductions and **releases more slowly** at first; linear computation recovers the first few decibels quickly, then lingers at the end.

For small reductions (a few decibels), the two computations are nearly identical. The difference grows with the depth of the reduction: it becomes clear with high ratios, low thresholds or negative ratios.

EvMu Lin/Db

ENV (cv) tab

Range: 0 to 1 **Default:** 0 **ID:** `db`

Chooses the domain in which the envelope is computed: **0** (default) = linear, **1** = decibels. Intermediate values do not compute the envelope in an "intermediate" domain: they compute **both** envelopes, each with its own state, and blend their outputs (at 0.5: half of one, half of the other).

The setting applies to all three envelope families, Gamma included, despite the "EvMu" in its label.

The decibel domain also affects the linear mode of attack and release (chapter 12): a constant-step progression **in decibels** gives a rounded curve in gain. This is one of the ways to "curve" a release.

Key takeaway: a decibel envelope is more "true to the ear" on large reductions; a linear envelope releases the first few decibels faster. If you are emulating a compressor, try both: some commercial compressors smooth in linear, others in decibels.

> **A word from the author**
>
> With modest settings, switching from linear to decibels doesn't change much. But as soon as the rest of the setup gets more extreme, the difference can be huge. On some of the compressors you may try to emulate, it is glaring.

> **♪ In the studio: linear or decibels on heavy reductions**
>
> The difference becomes audible as soon as the compressor works hard: a limiter on a drum room mic, a heavily compressed vocal, a bus pushed to its limits. In decibels, the envelope grabs heavy reductions more firmly and lets go of them more calmly at first; in linear, it gives back the first few decibels quickly, then lingers. Fifteen presets set **EvMu Lin/Db** to 1, including five emulations of the SSL bus compressor (**ssl_1**, **ssl_1_f**, **ssl_2_b_**, **tr_bus**, **acustica_sand3**), **waves_la2a**, **dmg_trackcomp_76a** and **softube_weiss**. The setting is independent of **Cv Lin/Db**: **ao_dbcomp** keeps a linear curve but smooths its envelope in decibels. To hear it, load one of these presets, push the reduction past 10 dB and switch **EvMu Lin/Db** between 0 and 1 while listening to the ends of the notes.

Branching or decoupled

There are two main ways to build an envelope follower, described notably by Giannoulis, Massberg and Reiss in their landmark paper on digital compressor design (*Journal of the Audio Engineering Society*, 2012).

The branching envelope is the simpler one: at each sample, one branch is chosen. If the envelope is above the requested gain, it attacks; otherwise, it releases. Attack and release never apply at the same time.

The decoupled envelope chains two stages:

1. a first stage **releases continuously** toward the requested gain, with the release time; when the requested gain drops below it, it jumps to it instantly;
2. a second stage follows the first with the **attack** time, in either direction.

The difference is mostly audible at the end of attacks. When the requested gain rises abruptly, the branching envelope starts releasing right away. The decoupled envelope keeps going down for a moment (its second stage is catching up with the first, which has only just started to rise), then sets off more gently: the gain dip is rounder, a little deeper and a little later. It is a kind of natural hold, subtler than Hold. The longer the attack, the longer the apparent release time.

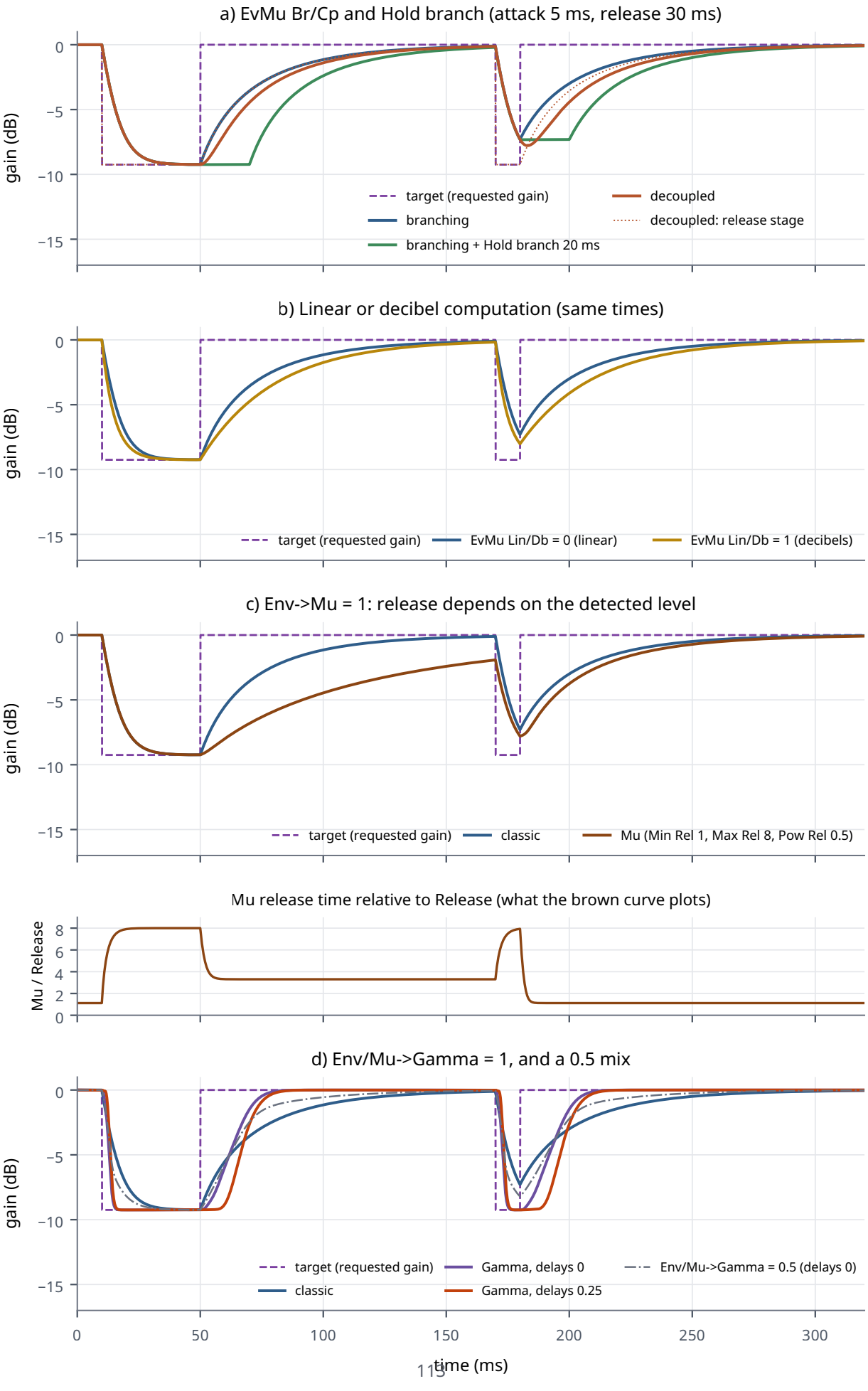

a) EvMu Br/Cp and Hold branch (attack 5 ms, release 30 ms)
gain (dB)
0
−5
−10
−15
target (requested gain)
branching
branching + Hold branch 20 ms
decoupled
decoupled: release stage
b) Linear or decibel computation (same times)
gain (dB)
target (requested gain)
EvMu Lin/Db = 0 (linear)
EvMu Lin/Db = 1 (decibels)
c) Env->Mu = 1: release depends on the detected level
gain (dB)
target (requested gain)
classic
Mu (Min Rel 1, Max Rel 8, Pow Rel 0.5)
Mu release time relative to Release (what the brown curve plots)
Mu / Release
8
6
4
2
0
d) Env/Mu->Gamma = 1, and a 0.5 mix
gain (dB)
target (requested gain)
classic
Gamma, delays 0
Gamma, delays 0.25
Env/Mu->Gamma = 0.5 (delays 0)
0
50
100
150
200
250
300
time (ms)

Figure 41 — The same signal processed by different envelopes. The purple target is the gain requested by a linear curve (threshold −12 dB, ratio 8:1) for a detected level that steps from −60 dB to 0 dB, −14 dB, 0 dB (10 ms), then −60 dB. a) branching, branching with hold, decoupled; b) linear or decibels; c) Mu envelope and its time ratio (brown curve); d) Gamma, with and without delays, and a 50% blend.

In the figure, part a) shows the three behaviors. On the long reduction (from 10 to 50 ms), branching and decoupled attack the same way; on release, the decoupled envelope sets off more gently. On the 10 ms peak (at 170 ms), it goes lower than the branching envelope and reaches its lowest point after the peak has ended. The thin dotted line is its first stage: it rises as soon as the peak ends, and the envelope then catches up with it.

> **Branching and decoupled: a vocabulary from research**
>
> The terms “branching” and “decoupled” come from the paper by Giannoulis, Massberg and Reiss on digital compressor design, published in June 2012 in the *Journal of the Audio Engineering Society*. The authors describe two detectors in it. The **decoupled** detector catches peaks instantly, then smooths them: attack and release are separate, but the attack “prints” onto the release, whose measured duration is roughly the sum of the two times. The **branching** detector applies the release only when the level comes back down: the release time you get is exactly the one you set. The same paper also compares detection in the linear domain and in the logarithmic domain, which echoes the choice offered by **EvMu Lin/Db**.

EvMu Br/Cp

ENV (cv) tab

Range: 0 to 1 **Default:** 0 **ID:** `indirect`

Chooses the envelope law: **0** (default) = **branching** (*Br*), **1** = **decoupled** (*Cp*, for *coupled*).

In between, Comprendre does not add up two envelopes: it applies both laws **in succession** to the **same** state, each one weighted. At 0.5, each sample undergoes half a decoupled step, then half a branching step. The behavior is therefore a true intermediate between the two dynamics.

This setting concerns the classic envelope and the Mu envelope. It has no effect on the Gamma envelope.

In decoupled mode, some settings of the A/R tab behave differently: **Release Mode** works (on the first stage), but **Attack Mode** has no effect; **Hold branch** does not apply; **Release Auto** subtracts the automatic attack time from the release time, to compensate for the lengthening inherent to the decoupled law (chapter 12).

> **A word from the author**
>
> In the compressor world, these really are two schools: roughly half the compressors on the market are branching, the other half decoupled. When you are trying to emulate a compressor, it is one of the first settings to test. The side effect of the decoupled law is that after an attack of a certain length, the release waits a little before coming back.

♪ In the studio: two schools, two ways to breathe

On a bass or a vocal, set a fairly long attack (20 to 50 ms) and a medium release, then switch **EvMu Br/Cp** between 0 and 1. In branching mode, the compression lets go as soon as the note weakens. In decoupled mode, it stays pressed down for a brief moment after each attack, then comes back more roundly: a kind of natural hold. In his French-language presentation, François compares it to **Hold**, which gives a clean, somewhat abrupt plateau; the decoupled law is subtler, but the difference is still huge. Five presets use decoupled mode: **acqua_magenta** (an emulation of the Acustica Magenta, vari-mu inspired, with **Attack** at 50 ms and **Release** at 1 s), **acqua_titanium**, **dmg_trackcomp_76a** (a revision A 1176), **ssl_1** and **voxengo_polysquasher3**. Load one and set **EvMu Br/Cp** to 0: you will hear what this school brings to the character of the model.

Hold

Hold is a more direct process: after an attack, the envelope stays **frozen** for a fixed duration before releasing.

Hold branch

ENV (cv) tab

Range: 0 to 2 s **Default:** 0 s **ID:** `pause`

Duration of the envelope hold after an attack, from 0 to 2 s. At **0** (default), there is no hold.

A counter restarts from zero at each attack sample. Until it reaches the set duration, the envelope stays put, even if the requested gain rises. If, during that time, the requested gain drops back below the envelope, the attack resumes immediately (and the counter restarts from zero). Hold therefore only delays the **release**.

As its name suggests (*branch*), hold only acts on the **branching** law, for the classic envelope and the Mu envelope, in linear as in decibels. It has no effect in decoupled mode or on the Gamma envelope. With **EvMu Br/Cp** between 0 and 1, only the branching part is frozen: the decoupled part keeps acting on the same envelope, and the hold "leaks."

The duration is expressed in real seconds: it does not depend on oversampling.

In the previous figure (part a), the 20 ms hold shifts the whole release: the green curve stays flat, then sets off with the same release time as the branching envelope. It is a "clean" hold, whereas the decoupled law rounds things off.

Hold against low-end distortion

With the default detection (instantaneous peak value), the requested gain follows every half-cycle of the signal: on a bass at 50 Hz, it rises between two peaks, every 10 ms. With short times, the envelope follows these ripples and the compressor distorts. A hold slightly longer than half a period (here about ten milliseconds) freezes the gain between two peaks and removes most of these ripples, without slowing down the attack.

> **Hold, from Abbey Road to digital mastering**
>
> Freezing the gain is not a new idea. When EMI's engineers at Abbey Road turned Altec 436B units into RS124 compressors, they fitted them with a six-position release selector, including a **Hold** position that kept the gain from coming back up: it prevented the room ambience from swelling at the end of a song. In radio, CBS's Audimax II added a gate that froze the gain below a threshold. And the Weiss DS1, which became the reference digital mastering compressor, offers a release delay to counter the "hiss" of noisy sources. Comprendre's **Hold branch** is a hold of this kind, but measured: it freezes the envelope after each attack, for a duration you choose, up to 2 s.

The Mu envelope: a speed that depends on level

In vari-mu tube compressors, attack and release times are not fixed: they vary with the signal level. Comprendre's **Mu** envelope borrows this idea: its attack and release speeds **glide continuously** between two bounds, according to the detected level.

It works in two steps:

1. a **speed controller** computes, at each sample, an attack speed and a release speed. Each one aims for a value between a "Min" speed and a "Max" speed, according to the detected level raised to a power (**Mu Pow Att**, **Mu Pow Rel**). The controller does not jump from one speed to another: it glides toward the target speed, more or less quickly depending on **Mu Speed**;
2. the **envelope** itself chases the requested gain, like the classic envelope, but with these variable speeds.

The Min and Max bounds are **multipliers of the time** set by Attack or Release. With **Mu Min Rel** at 1 and **Mu Max Rel** at 4, the release lasts 1 times the Release time at one extreme, 4 times at the other, and every value in between.

For the ENV (cv) tab, the quantity that drives the controller is the **detected level**, that is, the output of the detection stage (DET (in) tab, chapter 13), before the transfer curve, **as a linear value**:

- zero level (silence) → **Min** speeds;
- level of 1 (0 dBFS) or more → **Max** speeds.

Because this level is linear, it varies mostly near the top of the scale: −6 dB corresponds to 0.5, −20 dB to 0.1, −40 dB to 0.01. With **Mu Pow** at 1, a signal detected around −20 dB therefore stays very close to the Min speeds; you need to get close to 0 dBFS to reach the Max speeds. Powers below 1 spread the effect toward lower levels; powers above 1 concentrate it near 0 dBFS.

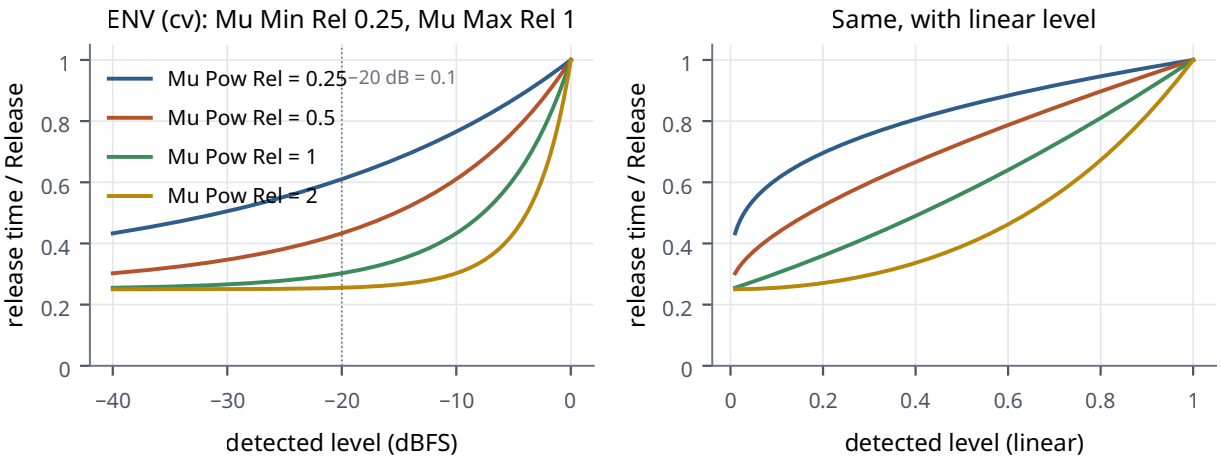


Figure 42 — Release time of the Mu envelope (relative to Release) versus detected level, for Mu Min Rel = 0.25 and Mu Max Rel = 1, once the speed controller has settled. Left, in dBFS; right, as a linear value.

In the comparison figure (part c), the Mu envelope is set with Mu Min Rel = 1, Mu Max Rel = 8 and Mu Pow Rel = 0.5. During the −14 dB plateau (from 50 to 170 ms), the detected level is 0.2: the release lasts about 3.3 times the set time, and the recovery is much slower than that of the classic envelope. After the second peak, the level drops back to −60 dB: the release returns almost to the set time. The release therefore depends on the level **during** the release: a passage that stays loud holds the compression back; silence frees it.

The brown curve

When **Env->Mu** is greater than 0, the history graph draws a **brown line**: it is the current release time of the Mu envelope divided by the **Release** time. It is drawn on the same scale as the gains: at the top of the graph, the Mu release equals Release (ratio 1); on a linear scale, at mid-height, it is half as long (ratio 0.5). A ratio greater than 1 stays pinned to the top edge.

This line only concerns the **release** of the ENV (cv) tab: the attack speed varies too, but is not displayed, and the Mu envelopes of the other three ENV tabs have no display. Whether it is shown also depends on the **Gfx Gains** setting in the GRAPHS tab (letter “u”, chapter 4).

> **Reading the brown curve**
>
> To keep the brown curve readable, keep both release bounds between 0 and 1 (for example Mu Min Rel = 0.25 and Mu Max Rel = 1): the line then wanders across the height of the graph. With bounds above 1, it stays pinned to the top.

The Mu settings

Env->Mu

ENV (cv) tab

Range: 0 to 1 **Default:** 0 **ID:** `mu`

Crossfade between the **classic** envelope (0, default) and the **Mu** envelope (1). Intermediate values blend the outputs of the two envelopes, each computed on its own. The speed controller only runs if this setting is greater than 0; at 1, the classic envelope is no longer computed.

With the default bounds (**Mu Min** and **Mu Max** at 1), the Mu envelope has practically the same times as the classic envelope: setting Env->Mu to 1 changes almost nothing. You need to spread the Min and Max bounds apart to hear a difference. Three differences from the classic envelope remain:

- the Mu envelope ignores the **linear** part of Attack Mode and Release Mode (chapter 12); it does follow the logarithmic mode, the zero mode and their blends;
- it is clamped between 0 and 1;
- with a zero attack time, it is not quite instantaneous: it covers half the distance at each sample.

The Mu envelope accepts the branching law and the decoupled law (**EvMu Br/Cp**), hold (in branching mode), linear or decibel computation, the curve shape (**Shape**) and automatic times.

Mu Min Att

ENV (cv) tab

Range: 0 to 421 **Default:** 1 **ID:** `mu_min_a`

Multiplier of the **attack** time when the detected level is zero. At 1 (default), the attack time is the one set by the **Attack** knob; at 2, it is twice as long; at 0.5, half as long.

At 0, the time becomes zero: the Mu envelope then covers half the distance at each sample, which is very fast without being instantaneous.

You can set **Mu Min Att** above **Mu Max Att**: the direction of the variation is simply reversed (the attack becomes faster on loud signals).

The knob is very progressive: the value 1 sits about two-thirds of the way through its travel, and the rest of the travel goes up to 421. For a precise setting, type the value on the keyboard (chapter 3).

Mu Max Att

ENV (cv) tab

Range: 0 to 421 **Default:** 1 **ID:** `mu_max_a`

Multiplier of the **attack** time when the detected level reaches 0 dBFS (linear value 1). With **Mu Min Att** = 1 and **Mu Max Att** = 3, the attack lasts the set time on a quiet signal and up to three times that time on a signal close to 0 dBFS. Same range and remarks as **Mu Min Att**.

Mu Pow Att

ENV (cv) tab

Range: 0 to 8 **Default:** 1 **ID:** `mu_a`

Power applied to the detected level before choosing the **attack** speed between Min and Max.

- At **1** (default), the speed varies in proportion to the linear level.
- **Below 1**: medium and low levels already move clearly away from the Min speed; the effect spreads over a larger part of the dynamic range.
- **Above 1**: the speed stays close to Min until the level gets close to 0 dBFS; the effect is reserved for peaks.
- At **0**, the level no longer has any influence: the speed is always the **Max** speed.

Mu Min Rel

ENV (cv) tab

Range: 0 to 421 **Default:** 1 **ID:** `mu_min`

Multiplier of the **release** time when the detected level is zero. Same principle as **Mu Min Att**, applied to the time of the **Release** knob.

At 0, the release becomes instantaneous.

Mu Max Rel

ENV (cv) tab

Range: 0 to 421 **Default:** 1 **ID:** `mu_max`

Multiplier of the **release** time when the detected level reaches 0 dBFS. This is the most “vari-mu” setting in the tab: with **Mu Min Rel** = 1 and **Mu Max Rel** = 4, the compressor releases at the set rate when the music calms down, and up to four times more slowly when it stays loud. The brown curve shows the current ratio.

Mu Pow Rel

ENV (cv) tab

Range: 0 to 8 **Default:** 1 **ID:** `mu_r`

Power applied to the detected level before choosing the **release** speed between Min and Max. Same principle as **Mu Pow Att**. It decides from what level the release starts to change: the previous figure shows its effect.

Mu Speed

ENV (cv) tab

Range: 0.000001 to 1 **Default:** 1 **ID:** `mu_sqrt`

How fast the **speed controller** moves, that is, how quickly the attack and release speeds glide toward their new value when the level changes. Range: from 0.000001 to 1.

- At **1** (default), the controller is very fast: on the order of a millisecond for a 100 ms release. The speeds follow the detected level almost instantly, peak by peak.
- Lower values slow the controller down: about ten times slower at 0.1, a hundred times at 0.01, a thousand times at 0.001. The speeds then follow the trend of the level rather than each peak, and the envelope "moves" less.

The controller is also slower the longer the envelope time: it slows down with the square root of the attack or release time.

Going further

The Mu envelope moves by $(g - e) / v^2$ at each sample, where v is the current speed ($v \geq 1$). For a time T, v is about $\sqrt{(T \cdot fs / (-\ln a))}$: the quantity the controller glides is therefore proportional to the **square root** of the time. The target speed is v* = x^p · (v_max − v_min) + v_min, where x is the detected level, p the power (Mu Pow), and v_min and v_max the speeds corresponding to the times T × Mu Min and T × Mu Max; it is clamped between these two speeds. The controller follows the law $v \leftarrow v + (v^* - v) \cdot s / v$, where s is Mu Speed × 44,100 / (host sample rate). The release time therefore varies as the square of v: with Mu Pow Rel = 1, Mu Min Rel = 1 and Mu Max Rel = 4, a level of 0.5 gives a release of 2.25 times the set time, not 2.5 times.

To be confirmed by François

In the videos, a power of 2 is presented as making the variation "linear." In the code, the interpolated speed is proportional to the square root of the time, so no value of Mu Pow makes the time proportional to the level when Mu Min is not zero; it is rather a power of 0.5 that comes close when Mu Min is small. The manual describes the code: is this really the intent?

To be confirmed by François

The Mu Speed compensation (× 44,100 / host sample rate) does not make the setting independent of the sample rate: the controller's actual time constant varies as the square root of the host sample rate divided by the square root of the oversampling factor (about 1.5 times slower at 96 kHz, twice as fast at ×4). Is this intended?

A word from the author

The Mu envelope creates movement: the release speed varies with level. The differences often remain subtle; to get big differences, you have to go to fairly extreme settings. **Mu Speed** lets you make the speeds move more slowly, and the power lets you amplify their variation.

♪ In the studio: a release that moves with the music

In his video on envelopes, François presents the Mu envelope as a way to make the curve more human; in his French-language presentation, as a way to create movement. On a drum bus or a string section, try the workflow “A release that holds back loud passages” at the end of the chapter: busy choruses keep the compression in place, calmer passages release it faster. Six presets enable **Env->Mu**, each in its own way.

- **rockruepel_comptwo_drums** takes the idea to the extreme, with **Mu Min Rel** at 0.072 and **Mu Max Rel** at 421: a release of a few milliseconds in quiet passages, stretching to several seconds when the detected level approaches 0 dBFS.
- **acustica_amber3** does almost the opposite: its release gets shorter on loud signals (**Mu Min Rel** 1.734, **Mu Max Rel** 0.754) and its attack slows down sharply near 0 dBFS (**Mu Max Att** 9.584).
- **uhe_presswerk** halves its times in quiet passages and makes them almost instantaneous on peaks (**Mu Min** at 0.5 and **Mu Max** at 0, for attack and release alike).
- **acqua_opal** (0.565), **acqua_titanium** (2) and **bx_townhouse** (0.685) only change **Mu Min Rel**, that is, the release in quiet passages.

The Gamma envelope

The **Gamma** envelope is a free algorithm, *gammaenv*, published by Aleksey Vaneev (Voxengo) under the MIT license. Where the classic envelope is a simple one-pole filter (an exponential), Gamma combines five sets of first-order low-pass filters, four of them as cascades of four filters. The result is an **S-shaped transition**: a gradual start, a steep rise, a decisive arrival, close to a slightly hardened half sine wave.

For the same set time, Gamma starts more slowly than the exponential but **finishes its transition much earlier**: it gets halfway at about 0.5 times the set time and almost all the way (99%) at about 1.1 times that time, whereas the classic exponential only reaches 99% after 4.6 times its time constant. It therefore seems both a little late at the start and much more “violent.” It can also slightly overshoot its target.

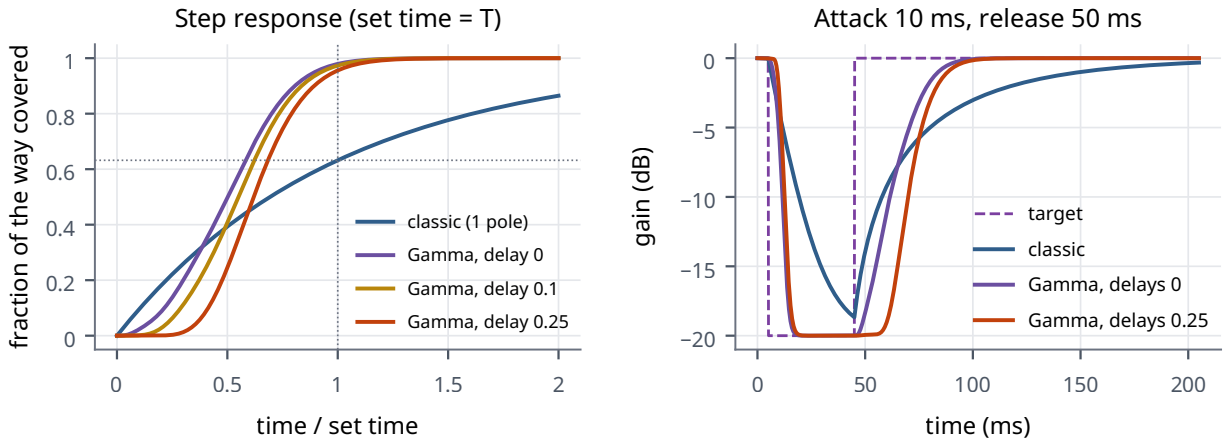


Figure 43 — Left: step response of the classic envelope and of the Gamma envelope for different delays, with time expressed in multiples of the set time. Right: a 20 dB, 40 ms reduction with Attack = 10 ms and Release = 50 ms.

The time Comprendre gives Gamma is the set time (Attack or Release) multiplied by −1/ln(altitude); with the default altitude (1/e), it is exactly the set time. The Gamma envelope follows the same target as the other envelopes, **in parallel**: it does not smooth their output.

Delays

Gamma can devote up to 25% of its time to a **delay stage**, built with the same filters (no buffer memory). It is not a pure delay: the start of the S-curve is pushed back, and the transition becomes steeper.

Delay	1% of the way	10%	50%	90%	99%
0	0.07 T	0.22 T	0.50 T	0.81 T	1.08 T
0.10	0.16 T	0.29 T	0.55 T	0.84 T	1.11 T
0.25	0.29 T	0.41 T	0.62 T	0.89 T	1.17 T
classic exponential	0.01 τ	0.11 τ	0.69 τ	2.3 τ	4.6 τ

What Gamma ignores

Gamma has its own way of handling attack and release: a fast envelope runs all the time; as long as it moves in the attack direction, it provides the output; otherwise, a slow envelope takes over. As a result:

- Gamma has **neither a branching nor a decoupled law** (EvMu Br/Cp has no effect on it) and **no hold**;
- it completely ignores **Attack Mode** and **Release Mode** (neither zero mode nor linear mode);
- on the other hand, it does accept decibel computation (**EvMu Lin/Db**), altitude, the curve shape (**Shape**) and automatic times.

> **Attack longer than release**
>
> The Gamma algorithm always gives the shorter time to the attack direction. If **Attack** is longer than **Release**, Gamma behaves exactly as if the two times were swapped, **delays included**: with Attack = 1 s and Release = 10 ms, the Gamma part attacks in 10 ms and releases in 1 s, and it is **Gamma R delai** that applies to the attack.

Env/Mu->Gamma

ENV (cv) tab

Range: 0 to 1 **Default:** 0 **ID:** `gamma`

Crossfade between the classic/Mu envelope (0, default) and the **Gamma** envelope (1), performed separately in the linear domain and in the decibel domain. The Gamma envelope is only computed if this setting is greater than 0; at 1, the classic and Mu envelopes are no longer computed.

Intermediate values are very useful: a little Gamma blended with the classic envelope shortens the tail of the exponential without the abrupt transition of Gamma alone (part d of the comparison figure, at 0.5).

Gamma A delai

ENV (cv) tab

Range: 0 to 0.25 **Default:** 0 **ID:** `gamma_attaque_delai`

Share of Gamma's attack time devoted to the delay stage, from 0 to 0.25 (that is, 0 to 25%). At 0 (default), there is no delay. The higher the value, the further the start of the attack is pushed back and the steeper the transition that follows. No effect if **Env/Mu->Gamma** is at 0.

Gamma R delai

ENV (cv) tab

Range: 0 to 0.25 **Default:** 0 **ID:** `gamma_relachement_delai`

Same thing for Gamma's release. A release delay holds the compression for a brief moment before letting it go decisively: it is a rounded alternative to **Hold branch**, which Gamma does not have.

A word from the author

Gamma is pretty violent: it's like a four-pole filter, with recursion between the stages. Like any multi-pole filter, it introduces a bit of delay, but the transition is much more abrupt. It's also a good technique for giving the release some curvature.

To be confirmed by François

The Gamma envelopes are initialized to 0 (full reduction) when the host prepares the plugin, whereas the other envelopes start at 1; the code carries the note "BUG." With Env/Mu->Gamma above 0, the very beginning of playback therefore gets a fade-in lasting as long as Gamma's release. Should they be initialized to 1?

To be confirmed by François

With lookahead active (la delay other than 0), the Gamma envelopes of the "future" chain only receive their times, delays and direction when the plugin is prepared: a change to Attack, Release, altitude or a Gamma delay during playback does not reach them (unless Auto or Shape is active). Is this intended?

♪ In the studio: Gamma, a release that ends cleanly

Where the classic exponential lingers a long time before reaching its target, Gamma finishes its course decisively. François finds it rather violent, and that is exactly what makes it interesting. On a drum bus or a loop, a little Gamma (**Env/Mu->Gamma** between 0.2 and 0.4) shortens the release tail: the compression lets go more cleanly between hits instead of fading out slowly, and the whole thing often feels tighter, more rhythmic. With **Gamma R delai**, the compression stays in place for a moment before letting go, a rounder alternative to **Hold branch**. None of the included presets enables the Gamma envelope: it is one of the follower's colors that remain entirely to be explored.

Comparing the envelopes on the same signal

The large comparison figure above sums up the differences between the envelope families, all with Attack = 5 ms and Release = 30 ms:

- **a)** the branching law releases as soon as the target rises; hold shifts the release as a block; the decoupled law rounds off the dip and delays the recovery;
- **b)** in decibels, the attack on the short peak goes deeper and the release starts more slowly than in linear;
- **c)** the Mu envelope attacks like the classic one (its attack bounds are at 1), but releases slowly as long as the detected level stays high and returns to the nominal time in silence;
- **d)** Gamma attacks in an S-curve, faster than the exponential over the whole transition, and finishes its release well before it; delays of 0.25 visibly push back the start of each transition. The 0.5 blend sits between the two.

Blending the envelopes

All the blend settings are continuous: you can dose each family. A few useful combinations:

- **classic** + **a little Gamma** (Env/Mu->Gamma between 0.2 and 0.5): the attack keeps its immediate start, but the long tail at the end of the exponential partly disappears;
- **Mu** + **Gamma**: Env->Mu acts inside the non-Gamma part; you get an envelope part of which varies with level while the other part has an S-shaped transition;
- **linear** + **decibels** (EvMu Lin/Db at 0.5): an intermediate release, with neither the fast recovery of linear nor the tail of decibel computation;
- **branching** + **decoupled** (EvMu Br/Cp at 0.5): a lighter rounding than pure decoupled.

Each combination multiplies the number of envelopes computed (up to six), per channel if the channels are unlinked (Channels Unlink), and twice over if lookahead is active. In a heavy session, only enable what you can hear.

What each envelope takes from the A/R settings

The settings of the A/R (cv) tab (chapter 12) do not apply the same way to the different envelopes. This table sums up what the code does:

A/R setting	Classic branching	Classic decoupled	Mu branching	Mu decoupled	Gamma
Times, altitude, Auto, mod	yes	yes	yes	yes	yes
Shape, Ret From, Ret Spd	yes	yes (different return)	yes	yes (different return)	yes (different return)
Attack Mode: logarithmic, zero and their blend	yes	**no**	yes	**no**	**no**
Attack Mode: linear part	yes	**no**	**no**	**no**	**no**

A/R setting	Classic branching	Classic decoupled	Mu branching	Mu decoupled	Gamma
Release Mode: logarithmic, zero and their blend	yes	yes	yes	yes	**no**
Release Mode: linear part	yes	yes	**no**	**no**	**no**
Hold branch (ENV tab)	yes	no	yes	no	no
EvMu Lin/Db (ENV tab)	yes	yes	yes	yes	yes

In linear mode with decibel computation, the constant step is expressed in decibels: the straight line becomes a curve in gain (chapter 12). For the Mu envelope, only the "toward the target or toward the extreme" part of the mode counts: any negative value acts like zero mode, any value above 1 like logarithmic mode. "Different return" refers to the particular way Ret Spd works in decoupled mode and in Gamma (chapter 12).

> **To be confirmed by François**
>
> In the walkthrough, zero mode is presented as also working with Gamma, and the decoupled linear mode as acting only on the attack. In the code, Gamma completely ignores Attack Mode and Release Mode, and in decoupled mode it is Release Mode that acts (linear and zero included) while Attack Mode has no effect (its version is commented out). The manual describes the code: is this the intended behavior?

The four envelope followers

The **ENV (in)**, **ENV (cv)**, **ENV (la)** and **ENV (bs)** tabs contain the same 14 settings, with the same ranges and the same default values. Their ids differ only by a prefix: none for ENV (cv), **var_** for ENV (in), **la_** for ENV (la), **bias_** for ENV (bs) (for example **mu_max**, **var_mu_max**, **la_mu_max**, **bias_mu_max**). Each one comes with a twin A/R tab that sets its times (chapter 12).

The engine is the same; what changes is what each follower smooths and when it is active.

	ENV (in)	ENV (cv)	ENV (la)	ENV (bs)
Id prefix	var_	none	la_	bias_
Where it acts	in the detection, **before** the curve: it is the "env" detector of DET (in)	**after** the curve: the compressor's classic follower	in the lookahead: it smooths the "future" envelope before applying it to the present	in the bias detection: the "env" detector of DET (bs)
What it follows	the level of the detection signal	the gain requested by the curve	the sliding minimum (or maximum) of the future envelope, a gain	the bias source signal
An attack is…	the level going up	the gain going down	the gain going down	the bias going up
Active when	**env db** (DET (in)) > −24 dB and an A/R (in) time > 0	**Attack** or **Release** > 0 (or a non-neutral setting in GR (cv))	envelope lookahead in use (MOD & LA, chapter 22) and an A/R (la) time > 0	**env db** (DET (bs)) > −24 dB and an A/R (bs) time > 0

	ENV (in)	ENV (cv)	ENV (la)	ENV (bs)
Active by default	no	**yes**	no	no
Mu: “Max” speeds when…	the level is zero (silence)	the detected level reaches 0 dBFS	the future envelope is not reducing (gain 1)	the bias is zero
Mu: “Min” speeds when…	the level reaches 0 dBFS	the level is zero	the future envelope is fully reducing (gain 0)	the bias is 1
Brown curve	no	yes	no	no
Chapter	13	this chapter	22	20

A few details:

- **ENV (in)** and **ENV (bs)** work “upside down”: so that the attack corresponds to a rise in level, they smooth the quantity 1 − level (clamped between 0 and 1), then flip the result back. A level above 0 dBFS (possible with a high input gain or a hot sidechain) is therefore capped at 0 dBFS in this detector.
- In these two tabs, decibel computation (EvMu Lin/Db) applies to this 1 − level quantity: it gives high resolution near 0 dBFS, not a decibel scale of the level itself.
- The driver of the Mu envelope therefore changes direction from one tab to another: in ENV (cv), “Max” corresponds to loud signals; in the other three, “Max” corresponds to quiet (silence, gain with no reduction, zero bias).
- **ENV (la)** only exists in the “present” chain, and only when lookahead acts on the envelope; its times (A/R (la) tab) are 0 by default: the lookahead envelope then follows its target instantly.

To be confirmed by François

The direction of Mu Min and Mu Max is opposite between ENV (cv) (“Max” = loud detected level) and the other three tabs (“Max” = silence for ENV (in), gain with no reduction for ENV (la), zero bias for ENV (bs)). The code carries the comment “contre détecteur” (counter-detector): is this inversion intended?

To be confirmed by François

In ENV (in) and ENV (bs), decibel computation smooths the logarithm of (1 − level), not the level in decibels. Is that the intent, or was decibel computation supposed to apply to the level itself?

To be confirmed by François

In the walkthrough, it is said that for the envelope placed before the curve “the attack is the release.” In the current code, the inversion is forced, so the **Attack** of A/R (in) does set the **rise** of the detected level (and thus the onset of compression). The manual describes this behavior: is this really the intended current behavior?

To be confirmed by François

For ENV (la), the Gamma envelope is set up as if the signal being followed were a level (“non-inverted” direction), whereas it is a gain: for the Gamma part of this tab only, the attack (gain going down) takes the longer of the two A/R (la) times and the release the shorter, the opposite of the classic and Mu envelopes in the same tab. Is this intended?

Practical workflows

Emulating a compressor

When you are trying to reproduce the behavior of an existing compressor (chapter 24), the envelope follower is often the key:

1. first set the threshold, ratio and times with a classic branching envelope;
2. compare the behavior at the end of the attack: if the compressor you are emulating “holds” its reductions before releasing, try **EvMu Br/Cp** at 1, then intermediate values;
3. on heavy reductions, compare **EvMu Lin/Db** at 0 and at 1: the difference shows mostly at the start of releases;
4. if the release seems to end “too abruptly” for an exponential, add a little **Env/Mu->Gamma**;
5. if the release speed seems to depend on program level, try the Mu envelope.

A release that holds back loud passages

For “vari-mu”-style behavior (slow release as long as the program is loud, fast when it calms down):

- **Env->Mu** at 1;
- **Mu Min Rel** at 1 and **Mu Max Rel** between 2 and 8;
- **Mu Pow Rel** between 0.25 and 0.5, so that the effect is not limited to peaks close to 0 dBFS;
- **Mu Speed** around 0.01, so that the speed follows the trend of the program rather than each peak.

To visualize the effect, temporarily set Mu Min Rel to 0.25 and Mu Max Rel to 1 and watch the brown curve, then go back to your values.

Curving a release

There are several ways to give the release a shape other than a simple exponential:

- compute the envelope in decibels (**EvMu Lin/Db** at 1) and switch **Release Mode** to linear (chapter 12): the release then progresses in constant steps in decibels;
- add some Gamma envelope, possibly with a **Gamma R delai**;
- change the **Release Shape** (chapter 12), which acts directly on the shape of the curve.

Starting ideas

A few presets make good starting points for hearing what this chapter describes, as long as you listen to what one single setting changes at a time.

- **Sustained sounds: vocals, bass, pads. acustica_cobalt**, an emulation of Acustica's Cobalt (a feedback tube compressor in the spirit of the Altecs), combines a 12.3 ms **Hold branch** with a 30 ms attack and a 250 ms release. Bring **Hold branch** back to 0 to hear what the hold contributes to the sustain of low notes.
- **A vari-mu that breathes. acqua_magenta** combines the decoupled envelope, feedback detection (chapter 14) and a one-second release: a starting point for a bus or a laid-back vocal.
- **Drums that move. rockruepel_comptwo_drums** and its extreme Mu envelope: compare with **Env->Mu** at 0 to gauge what the variable speed brings.
- **The bus, SSL style. ssl_1** combines the decibel envelope, the decoupled envelope and automatic times (chapter 12): a good example of how these choices work together.

CHAPTER 12

Attack and Release

The envelope follower (chapter 11) chases the gain requested by the transfer curve. This chapter describes **how fast and in what way** it chases it: the attack and release times, their automatic calculation, the altitude, the shape of the curves, the "return" of the position on these curves, and the modes (logarithmic, linear, zero).

Attack and release are probably the two most musical controls on a compressor. Attack decides the fate of transients: letting through the crack of a snare, the pluck of a string, the consonants of a vocal, or flattening them. Release decides the breathing: compression that melts into the tempo, that "glues" a bus, or that pumps conspicuously. On hardware compressors, these two times are often switches with a handful of positions, and their shape is fixed by the circuit. Comprendre turns them into a real workshop: times from 0 to 2 s, automatic times that adapt to the character of the signal, and the ability to draw the very shape of the attack and the release. Drummers, mix engineers and mastering engineers will find what they need to dial in their compression precisely; those who emulate a unit, what they need to reproduce the way it moves.

For the follower placed after the curve, these settings live in two places:

- the **Attack** and **Release** knobs on the main row, which set the times;
- the **A/R (cv)** tab, which contains the other twelve settings, on two rows: the first for the attack, the second for the release.

In this tab, the first column does not contain times but **mod attack** and **mod release**: these two settings add to the times a modulation driven by the dynamic EQs or the waveshaper. They are described in chapter 23, with the other modulations.

Comprendre contains three other A/R tabs, twins of this one: **A/R (in)**, **A/R (la)** and **A/R (bs)**. They control the ENV (in), ENV (la) and ENV (bs) followers; their differences are summarized at the end of the chapter.



Figure 44 — The attack and release settings of the follower placed after the curve: Attack and Release on the main row, the other twelve in the A/R (cv) tab. The two settings that are not circled, mod attack and mod release, are described in chapter 23.

A quick reminder of the vocabulary: for the follower placed after the curve, an **attack** is a drop in gain (the compressor reduces further), a **release** a rise in gain. At each sample, the follower picks one or the other depending on whether it is above or below the requested gain.

Time, coefficient and altitude

At each sample, the envelope covers a fraction of the distance separating it from the requested gain. This fraction, the **coefficient**, is computed so that **after the set time, the "altitude" fraction of the distance remains**. With the default altitude, 1/e ≈ 0.368, 36.8% of the way remains after the attack time: the envelope has covered 63.2% of it. This is the usual compressor convention, where the set time is the time constant of an exponential.

The **altitude** lets you change this convention. With an altitude of 0.1, the set time becomes the time needed to cover 90% of the way; with 0.01, 99% of the way. For the same set time, the envelope is therefore faster the lower the altitude.

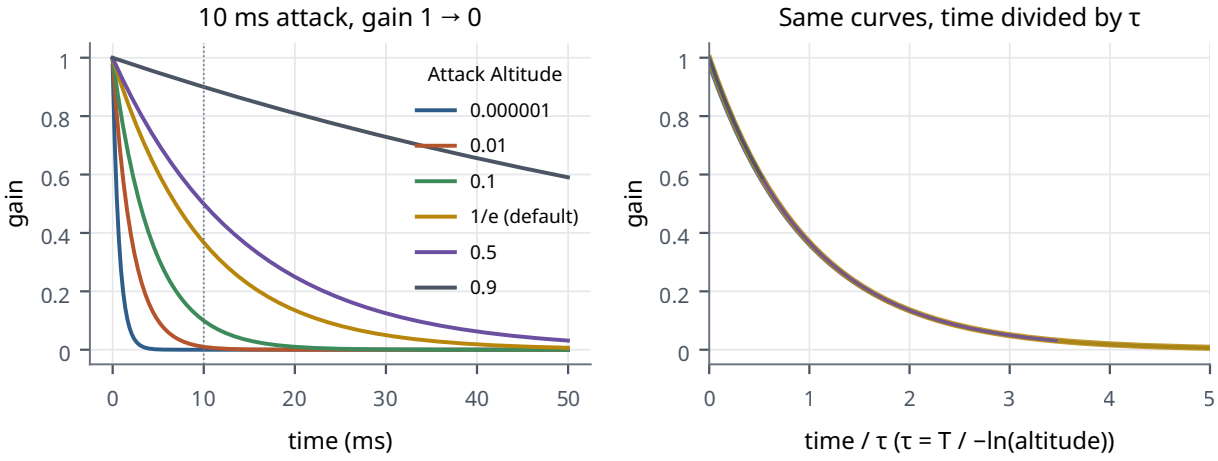


Figure 45 — Left: a 10 ms attack (requested gain going from 1 to 0) for different altitudes: after 10 ms, the envelope is exactly at the altitude. Right: the same curves with time divided by the time constant τ = T / (−ln altitude): they overlap.

The figure shows it: changing the altitude does not change the **shape** of the curve, only its **time scale**. All the curves are the same exponential, stretched to a greater or lesser degree. Altitude is therefore another way of setting the time: setting Attack = 10 ms with an altitude of 0.1 amounts to setting Attack = 4.3 ms (10 × 0.434) with the default altitude, for the classic envelope in logarithmic mode.

> **Going further**
>
> The coefficient is c = 1 − a^(1/(T · fs)), where T is the set time, a the altitude and fs the internal sample rate (oversampling included). With a constant target, the remaining distance at time t is a^(t/T) = e^(−t/τ), with τ = T / (−ln a). Examples: τ = T for a = 1/e; τ = 0.434 T for a = 0.1; τ = 1.44 T for a = 0.5; τ = 9.5 T for a = 0.9; τ = T / 13.8 for the minimum altitude (0.000001). Since fs includes oversampling, times stay accurate whatever factor you choose.

When altitude also changes the shape

The equivalence “altitude changes the time, not the shape” holds exactly for the classic envelope in logarithmic mode, in pure linear mode and in pure zero mode. It no longer holds in the following cases:

1. **Intermediate modes** (between logarithmic and linear, or between zero and linear; see below): the exponential part and the linear part do not depend on altitude in the same way. Changing the altitude then changes the **balance** between the two, and therefore the shape.
2. **Shape other than 1**: the curve remains a simple stretch of the same shape, but the stretch factor depends on the Shape value (for an altitude of 0.1: 0.434 with Shape 1, 0.659 with Shape 2).
3. **Shape = 0**: the altitude then sets the height of a step, not a time (see the section on shape).
4. **Ret From** is expressed in multiples of the set time, not of the time constant: the same Ret From does not correspond to the same point on the curve if the altitude changes.
5. **Altitude = 1** (the maximum): the coefficient becomes zero, and the classic or Mu envelope **no longer moves at all**. Only the Gamma part keeps evolving.

Attack

main row tab

Range: 0 to 2 s **Default:** 0.01 s **ID:** `attaque`

Attack time of the follower placed after the curve: the time after which, when the requested gain drops, only the **Attack Altitude** fraction of the way remains (36.8% by default). Range: 0 to 2 s; default: 10 ms. The knob has a progressive taper: the middle of its travel corresponds to 125 ms, the quarter point to 8 ms.

- At **0**, the attack is **instantaneous**: the envelope reaches the requested gain on the very next sample (even if Attack Shape is not 1). For the Mu envelope, it covers half the distance at each sample.
- If **Attack** and **Release** are both at 0 (and the GR (cv) tab is neutral), the follower is disabled: the requested gain is applied as is.

The time actually used can be modified on the fly by **Attack Auto** (below) and by **mod attack** (chapter 23). It is used by all the envelopes chosen in the ENV (cv) tab: classic, Mu (multiplied by Mu Min Att or Mu Max Att) and Gamma. It is also used by the opto designer (chapter 15).

For the decoupled envelope (chapter 11), the attack time is that of the second stage, which follows the first in both directions: a long attack time also slows down the start of releases.

Release

main row tab

Range: 0 to 2 s **Default:** 0.01 s **ID:** `relachement`

Release time of the follower placed after the curve: the time after which, when the requested gain rises, only the **Release Altitude** fraction of the way remains. Range: 0 to 2 s; default: 10 ms; same progressive taper as **Attack**.

At **0**, the release is instantaneous. The time actually used can be modified by **Release Auto** and by **mod release** (chapter 23); it is used by the classic, Mu (multiplied by Mu Min Rel or Mu Max Rel) and Gamma envelopes, as well as by the opto designer.

A very short release (a few milliseconds) lets the gain follow the half-cycles of the signal, especially in the low end: this is a source of distortion, intended or not. **Hold branch** (chapter 11) or RMS detection (chapter 13) can limit it.

> **♪ In the studio: attack decides the transient**
>
> On a snare, attack is the setting that changes everything. With a slow attack, from 20 to 40 ms, the hit gets through before the compression sets in: the crack stands out, often even more than before, and the compression takes care of the body and the ring. With a very fast attack, from 0.1 to 1 ms, the peak is grabbed right away: the snare flattens, gains density and sits further back in the mix. The same logic applies to a bass played with a pick, an acoustic guitar or the consonants of a vocal. The presets show the range of choices. The ones emulating Distressor-inspired plugins attack very fast: 0.5 ms for **kiive_xstressor**, 0.589 ms for **slate_fgstress**, 0.837 ms for **eli_arouser**. At the other end, **bx_townhouse** takes 127 ms, **arturia_tube-sta** and **softube_germanium** more than 180 ms, and they let almost all the transients through.

♪ In the studio: release decides the breathing

A short release gives the gain back immediately: the compression follows each note, and the sound gains density and energy, but with peak detection it can buzz in the low end. A medium release, timed to the tempo, brings the gain back up just before the next hit: this is often where compression melts into the groove. A long release keeps the gain down from one hit to the next: the dynamics flatten out, and you may hear the pad or the reverb pumping behind the drums, an effect people sometimes go for. Listen to the ends of the notes and the space between two hits rather than the attacks. The presets cover the whole range, from 7.3 ms for **playfair_grading** to 2 s for **acustica_ultramarine4**, an emulation of Acustica's Ultramarine 4 (a Fairchild 670), by way of 1.2 s for **acqua_eminence** and 1.5 s for **caelumçsmmov**.

Attack Altitude

A/R (cv) tab

Range: 0.000001 to 1 **Default:** 0.367879 **ID:** `altitude_attaque`

Fraction of the way that **remains** to be covered after the attack time. Range: 0.000001 to 1; default: 1/e ≈ 0.367879.

- **Lower** than 1/e, the attack is faster for the same set time: at 0.1, the envelope has covered 90% of the way after the attack time.
- **Higher**, it is slower: at 0.9, it has covered only 10% of the way.
- At **1**, the attack is frozen: avoid this.

Altitude also plays a part in the step of linear mode (in one attack time, the envelope covers 1 − altitude of full scale), in the shape drawn by the attack graph, in the speeds of the Mu envelope and in the time of the Gamma envelope (set time × −1/ln altitude). It is also used by the opto designer (chapter 15).

The attack graph in the main window shows the effect of altitude directly: at the right edge of the red zone (the set time), the curve passes at the height of the altitude.

Release Altitude

A/R (cv) tab

Range: 0.000001 to 1 **Default:** 0.367879 **ID:** `altitude_relachement`

Same thing for the release: the fraction of the way remaining after the release time. At the default, 63.2% of the recovery is done after the **Release** time. At 0.01, the Release time becomes the time for 99% of the recovery: the setting then reads as a "full return" duration.

A word from the author

A lot of people think altitude changes the shape of the curve. In reality, it only changes the proportion, and therefore the timing: it's another way of saying "for this many seconds, I want this much reaction." The shape is the next setting, Shape.

To be confirmed by François

With a Shape other than 1, the time passed to the Gamma envelope is recalculated at each sample without the −1/ln(altitude) factor applied in the normal case: with an altitude other than 1/e, the Gamma part therefore changes time as soon as Shape moves away from 1. Is this intended?

The times of the classics

Every great unit has its own territory. The 1176 attacks in 20 to 800 µs and releases in 50 ms to 1.1 s. The Fairchild attacks in 0.2 to 0.8 ms and releases in 0.3 to 25 s. The LA-2A attacks in about ten milliseconds on average, depending on frequency. The SSL bus compressor offers attacks of 0.1, 0.3, 1, 3, 10 and 30 ms and releases of 0.1, 0.3, 0.6 and 1.2 s, or Auto; the Manley Variable Mu, attacks from 25 to 70 ms; the Distressor, from 50 µs to 30 ms according to its manual. Some shapes have become famous. The release of the Distressor's **Nuke** mode, fast at first, then slower and slower, makes up "a big part of the sound" of the unit, according to that same manual: that is the profile of Comprendre's default exponential, which a **Release Shape** below 1 accentuates. And the 1176's "all buttons" mode owes part of its distortion to a delayed attack on the first transients; in Comprendre, it is an **Attack Shape** above 1 that holds back the start of the attack (see below). With times from 0 to 2 s, Comprendre covers all these territories.

Automatic attack and release

The **Attack Auto** and **Release Auto** settings adapt the times to the signal, following the method proposed by Giannoulis, Massberg and Reiss in their work on automating compressor settings. The idea: a very transient signal (percussion) calls for short times, a steady signal (pad, sustained vocal) for longer times.

To do this, Comprendre measures the **crest factor** of the detection signal, that is, the ratio between its peak level and its average (RMS) level. It uses two measurements with the same time constant, set by **auto secs**:

- a peak detector on the power, which rises instantly and falls back with the **auto secs** constant;
- a root mean square (RMS) average with the same constant.

The ratio of the two is the square of the crest factor, written C^2. It is 1 for a constant signal (or a square wave), 2 for a pure sine wave, and much more for a percussive signal.

The automatic times are then:

- automatic attack = 2 × Attack / C^2;
- automatic release = 2 × Release / C^2, minus the automatic attack if the envelope is decoupled.

For a pure sine wave ($C^2 = 2$), the automatic times equal the set times. For drums, they are shorter; for a very steady signal, they can go up to double.

Attack Auto and **Release Auto** then set the blend between the set time and the automatic time: at 0, the set time alone; at 1, the automatic time alone; at 0.5, the average of the two times. The calculation is redone at each sample.

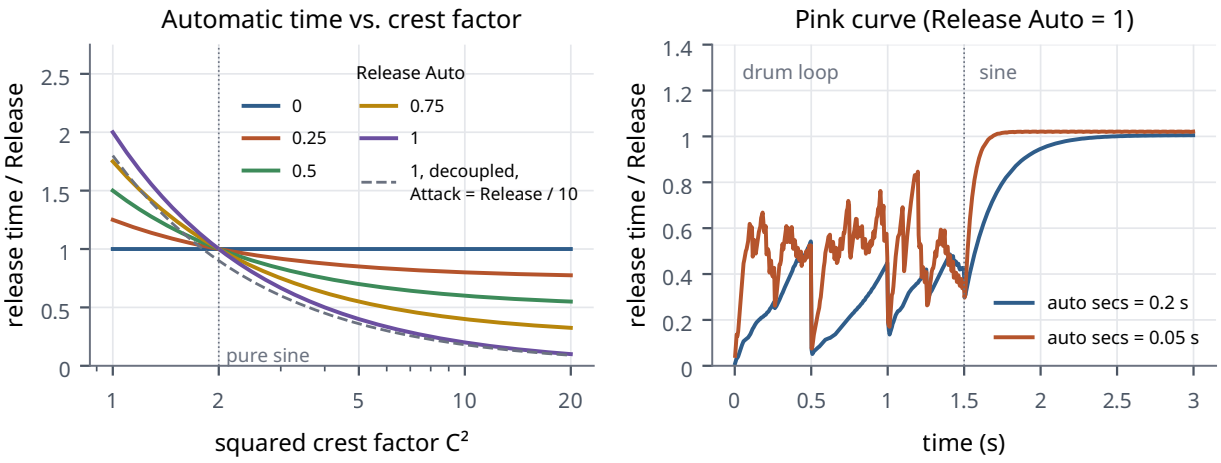


Figure 46 — Left: resulting release time (relative to Release) versus the square of the crest factor, for different values of Release Auto. Right: this ratio over time (the pink curve in the graphs) with Release Auto = 1, on a drum loop, then on a sine wave, for two values of auto secs.

Releases that adapt to the music

Release quickly after an isolated transient, slowly when compression is sustained: many units have sought this compromise. The Gates Sta-Level (1956) already offered, in **Double** mode, a program-dependent release. The Fairchild, in position 6 of its selector, releases in 0.3 s after a transient, in 10 s after repeated peaks and in 25 s on a constantly high level. The LA-2A's optical cell gets this behavior without even trying: it completes half of its release in about sixty milliseconds, then finishes in one to fifteen seconds according to the original manual (0.5 to 5 s according to the reissue's spec sheet), all the more slowly as the compression has been heavy and long. The **Auto** release of the SSL bus compressor stacks two time constants, 100 ms and 12 s. Finally, the Weiss DS1 measures both the peak and the RMS level, and chooses between a fast release and a slow release based on their ratio. **Attack Auto** and **Release Auto** rely on a measurement of the same kind, the crest factor; the **Mu** envelope (chapter 11) and the opto designer (chapter 15) offer other ways to make the release program-dependent.

The "auto secs" setting

The **auto secs** constant is not in the A/R tab but in the **DET (in)** tab (chapter 13); its default value is 200 ms. A short value makes the crest factor jumpy, very sensitive to each hit; a long value makes it more stable, more representative of the overall character of the passage. At 0, the crest factor is always 1 and the automatic times are double the set times.

This DET (in) setting is used for the automatic times of A/R (cv), A/R (in) and A/R (la). A/R (bs) has its own setting, **b auto secs**, in the DET (bs) tab.

The crest factor for A/R (cv) is measured on the **detection signal** as it enters the detector (after the mix with the sidechain, the filters and the feedback, chapter 14), before the peak, RMS and envelope detectors of the DET (in) tab.

Attack Auto

A/R (cv) tab

Range: 0 to 1 **Default:** 0 **ID:** `auto_attaque`

Blend between the set **Attack** time (0, default) and the automatic attack time (1), computed from the crest factor of the detection signal. Intermediate values interpolate the times in seconds: at 0.5, the time used is the average of the set time and the automatic time.

On a very percussive signal, the automatic attack becomes very short: peaks are caught faster. On a steady signal, it gets longer.

Release Auto

A/R (cv) tab

Range: 0 to 1 **Default:** 0 **ID:** `auto_relachement`

Blend between the set **Release** time (0, default) and the automatic release time (1).

With the **decoupled** envelope (EvMu Br/Cp at 1, chapter 11), the automatic attack time is subtracted from the automatic release (in proportion to EvMu Br/Cp): this compensates for the lengthening of the release inherent to the decoupled law. This subtraction uses the automatic attack time even if **Attack Auto** is 0. The resulting time never goes below 0.

When **Release Auto** is greater than 0, the history graph draws a **pink line**: the release time in use, divided by the Release time. It is drawn on the gain scale: at the top, the ratio is 1 (or more, the line then staying pinned to the edge); on a linear scale, at mid-height, the release is half the set time. Whether it is shown also depends on **Gfx Gains** (letter “a”, chapter 4). This line also exists when **mod release** is active; it only concerns A/R (cv).

A word from the author

The auto-release is the classic academic auto-release. The 200 ms value is very natural: the brain takes about 170 ms to react, plus a little margin, and it’s also the value many compressors stick with. Release Auto at 1 gives a result that’s a bit too square: François generally prefers 0.5, which works very well.

♪ In the studio: a release you can forget about

Automatic release is invaluable when the music changes character over the course of a song: an airy verse, then a dense chorus, a vocal that moves between speaking and singing, a mix that alternates between breaks and full-band sections. Start from the setting François considers the most natural: **auto secs** at 200 ms and **Release Auto** at 0.5. On percussive passages, the release gets shorter and the compression quickly makes room for the next attacks; on sustained passages, it gets longer and the gain stays steadier in the middle of the notes. Several presets use it this way. **waves_la2a** and **native_vc2a**, both imitations of LA-2A emulations, set **Release Auto** to 0.5, as do **softube_ds1mk3** (an imitation of Softube’s Weiss DS1-MK3) and **uhe_presswerk**. **softube_weiss**, **acqua_tulip**, **bx_townhouse**, **massenbug_drc2_piano**, **voxengo_polysquasher3** and **waves_rvox** go all the way to 1. **waves_la2a** also shortens **auto secs** to 75 ms, so that the measurement tracks the signal more closely.

Going further

With x the detection signal, $p \leftarrow \max(x^2, \alpha \cdot p + (1 - \alpha) \cdot x^2)$ and $r \leftarrow \alpha \cdot r + (1 - \alpha) \cdot x^2$, where $\alpha = e^{(-1/(\text{auto secs} \cdot fs))}$. Then $C^2 = p / r$ (1 if either is zero). With u_a and u_r the values of Attack Auto and Release Auto and b that of EvMu Br/Cp: $A' = A \cdot (1 - u_a) + (2A / C^2) \cdot u_a$ and $R' = R \cdot (1 - u_r) + (2R / C^2 - (2A / C^2) \cdot b) \cdot u_r$, clamped at 0. For A/R (cv), A and R already include the modulation (mod attack, mod release).

The shape: Shape

With the default settings, the envelope follows an exponential: it starts fast and slows down as it approaches its target. The **Shape** setting changes this curve by applying a power to the time axis. The remaining distance is no longer altitude^(t/T) but altitude^((t/T)^Shape).

- **Shape > 1**: slow start, then acceleration. On the attack, the first milliseconds of a transient get through with almost no reduction, then the gain drops decisively: the curve takes on an S shape.
- **Shape < 1**: very fast start, then a long tail. The envelope covers most of the way right away, then finishes slowly.
- **Shape = 1** (default): the usual exponential.

All these curves go through the same point after the set time: the altitude fraction of the way remains.

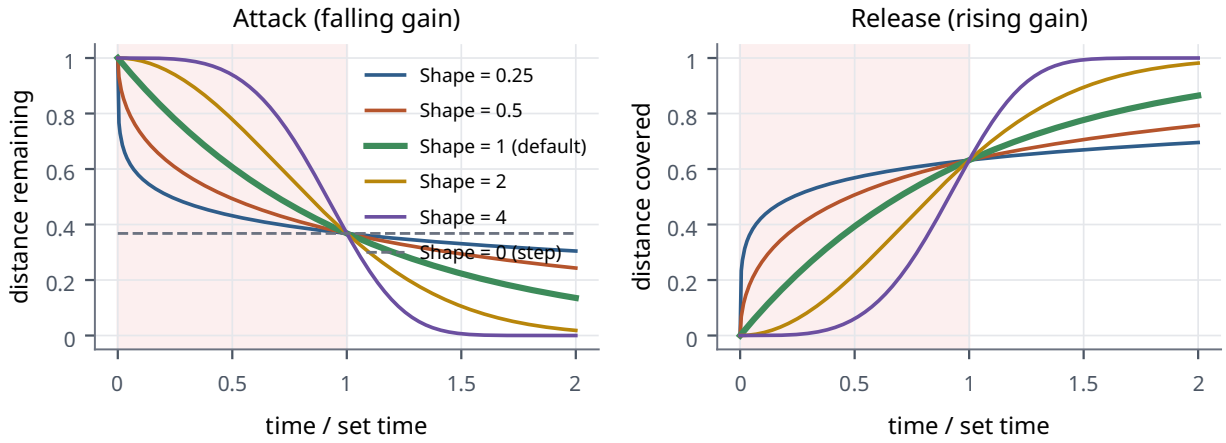


Figure 47 — The distance remaining to be covered (attack) and the distance covered (release) according to the shape, with time expressed in multiples of the set time. The red-tinted zone corresponds to the set time, as in the shape graphs of the main window.

To get this curve, Comprendre keeps track, for each direction, of a **position** on the curve: the time elapsed since the start of the phase, in multiples of the set time. At each sample of the phase, the coefficient is recalculated from this position. Two important consequences:

- **the longer a phase lasts, the more it speeds up** (Shape > 1) or **slows down** (Shape < 1). With Shape 2, beyond 3.7 times the set time, the envelope even follows the target instantly until the end of the phase;

- what happens at the **start** of a phase depends on the starting position: this is the role of the return settings (next section).

> **Shape = 0**
>
> At 0, the power gives a constant curve: on the first sample of a phase, the envelope covers the fraction 1 − altitude of the way in one go, then freezes until the position has returned to the start. The result looks more like a sample-and-hold than an envelope. Values close to 0 (0.05, 0.1) give a very large initial jump followed by a very slow tail.

> **To be confirmed by François**
>
> With Shape = 0, the envelope makes a single jump of (1 − altitude) at the start of each phase, then stays frozen as long as the position has not returned to the start; the shape graph shows a step. Should this case be presented as a usable effect, or discouraged?

Attack Shape

A/R (cv) tab

Range: 0 to 8 **Default:** 1 **ID:** `attaque_forme`

Shape of the attack curve, from 0 to 8; 1 (default) = the usual exponential.

- Above 1, the attack starts slowly, then speeds up: transients keep their bite, and the reduction then arrives all at once. It is also a way to get an S-shaped attack without using the Gamma envelope.
- Below 1, the attack immediately grabs a large part of the reduction, then finishes gently.

The shape applies to all the envelopes of the ENV (cv) tab: classic, Mu and Gamma. With a zero attack time, it has no effect (the attack remains instantaneous). When it is not 1, a white vertical line appears in the attack graph (see “The shape graphs”), and the **Att Ret From** and **Att Ret Spd** settings become active.

Release Shape

A/R (cv) tab

Range: 0 to 8 **Default:** 1 **ID:** `relachement_forme`

Shape of the release curve, from 0 to 8; 1 (default) = the usual exponential.

- Above 1, the gain first stays low, then rises faster and faster: a “held-back” release that lets go decisively.
- Below 1, the gain rises quickly at first, then lingers for a long time near 0 dB.

The **Rel Ret From** and **Rel Ret Spd** settings are only active if this shape is not 1.

> **A word from the author**
>
> Shape is where you really change the shape: either a release that’s slow at first and becomes fast, or the opposite. Changing the shape can put more pressure on one direction or the other: if the release is faster, the compression can attack again sooner, and there are real side effects between the two.

> **♪ In the studio: keep the crack without slowing the attack**
>
> With **Attack Shape** above 1, the attack starts slowly, then speeds up: the first milliseconds of a snare hit or a bass note get through almost intact, then the reduction drops all at once. You keep the crack while holding the body of the note firmly. **dmg_trackcomp_76a**, an imitation of TrackComp's 76A model (a revision A 1176), sets **Attack Shape** to 3.08, and **massenbug_drc2_piano**, whose name suggests a setting for piano, to 3.4. Conversely, **softube_cl1b** (0.328) grabs most of the reduction right away, then finishes it gently.
>
> On the release side, **massenbug_drc2_groove** combines a **Release Shape** of 2.088 with a **Rel Ret From** of 0: each release starts again from the beginning of its curve, where it is flat, and the faster the attacks follow one another, the less the gain recovers, which helps hold a tight groove. **ssl_1** does the opposite with a **Release Shape** of 0.333: the gain rises very quickly, then lingers near 0 dB. As François advises, check on the shape graphs that the vertical line really moves along the curve: otherwise, the shape does not have time to act.

The return: Ret From and Ret Spd

With the default detection (instantaneous peak value), the requested gain changes with every half-cycle of the signal: the envelope constantly switches between attack and release. If each phase restarted from the beginning of its curve, a shape that is slow at the start (Shape > 1) would never have time to get past that start, and the Shape setting would have almost no effect.

So Comprendre treats the position on each curve like a **capacitor**: it "charges" during its phase and gradually "discharges" during the opposite phase. When the phase resumes, it starts again from wherever the position has come down to, not from the beginning.

- During an **attack**, the position on the attack curve advances; the position on the release curve is first brought down to at most **Rel Ret From**, then it falls back toward the start at the **Rel Ret Spd** speed.
- During a **release**, it is the opposite, with **Att Ret From** and **Att Ret Spd**.
- During a **hold** (Hold branch), neither position moves.

Positions are expressed in multiples of the set time. A return speed of 1 brings the position down as fast as it went up.

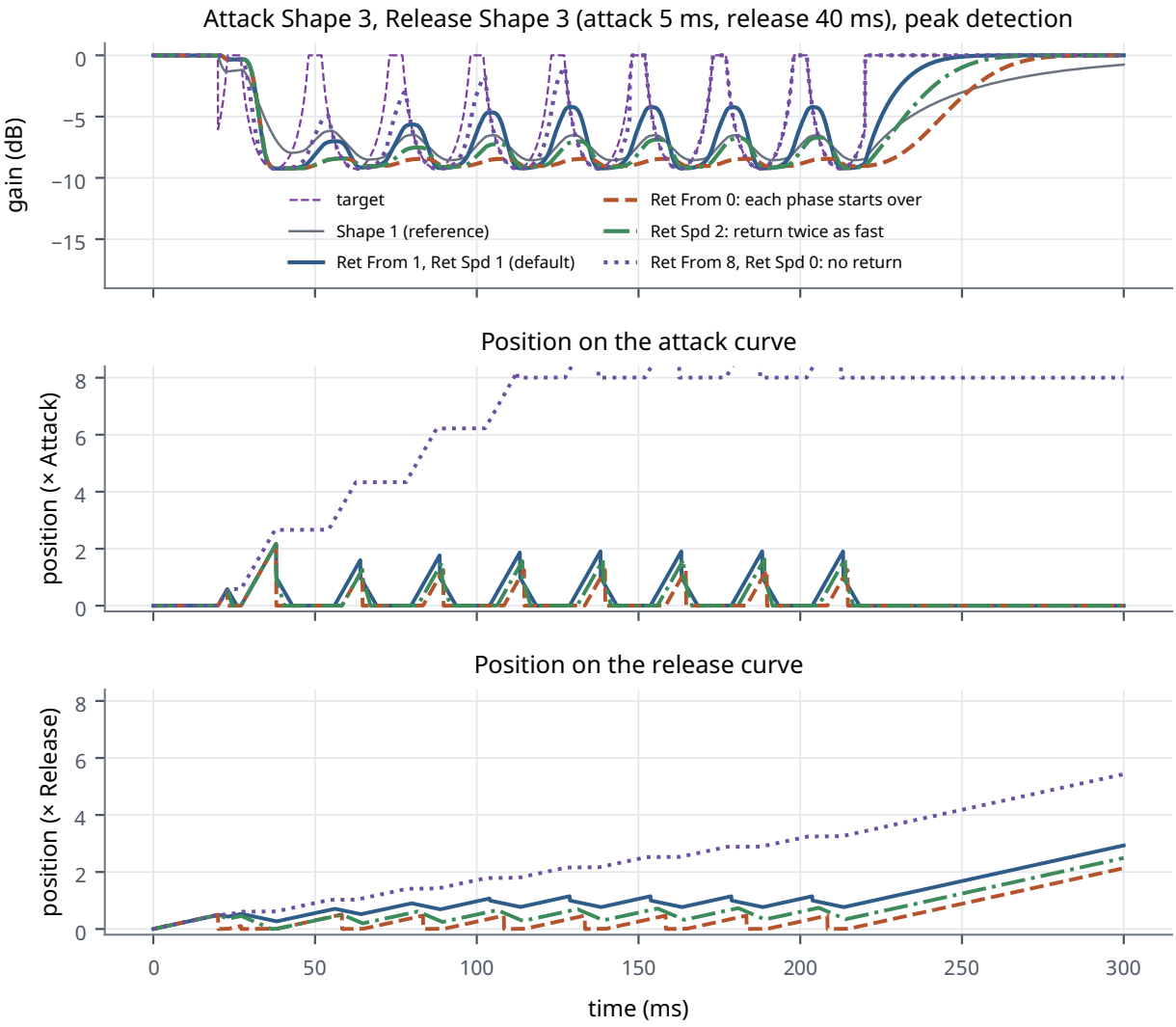


Figure 48 — A target that alternates at each half-period of a 20 Hz sine wave (peak detection, threshold −12 dB, ratio 8:1), with Attack Shape and Release Shape at 3. Top: the resulting gain for different return settings; below: the positions on the attack curve and on the release curve.

The figure shows what is at stake. With the default settings (Ret From 1, Ret Spd 1), the release position stays around 1: each release resumes around the middle of its curve, where it is steep, and the gain ripples noticeably. With **Ret From at 0**, each phase restarts from the beginning, where the curve is flat: the gain ripples much less. **Ret Spd at 2** gives an intermediate result. Finally, with **Ret Spd at 0** (and Ret From at maximum), the positions never come down: they accumulate, the curves reach their steepest part, and the envelope ends up following the target almost instantly.

Att Ret From

A/R (cv) tab

Range: 0 to 8 **Default:** 1 **ID:** `attaque_retour_depart`

Maximum position (in multiples of the attack time) that the **attack** curve keeps when a release begins. Range: 0 to 8; default: 1.

- At **0**, each new attack starts again from the **beginning** of its curve.
- At **1** (default), it restarts at most from the point located at the attack time, that is, from the middle of the attack graph.
- At **8**, there is practically no ceiling: the position kept is the one reached at the end of the attack.

Only acts if **Attack Shape** is not 1.

Att Ret Spd

A/R (cv) tab

Range: 0 to 8 **Default:** 1 **ID:** `attaque_retour_vitesse`

Speed at which the position on the **attack** curve returns toward the start during releases, in multiples of its forward speed. Range: 0 to 8; default: 1.

- At **1**, the position comes down as fast as it went up.
- At **0**, it does not come down: it stays stuck at the value set by **Att Ret From**.
- At **8**, it returns eight times faster.

Only acts if **Attack Shape** is not 1.

Rel Ret From

A/R (cv) tab

Range: 0 to 8 **Default:** 1 **ID:** `relachement_retour_depart`

Maximum position (in multiples of the release time) that the **release** curve keeps when an attack begins. Works the same way as **Att Ret From**. At 0, each release restarts from the beginning of its curve: with a Release Shape above 1, this strongly holds back the recovery of the gain when attacks are frequent.

Only acts if **Release Shape** is not 1.

Rel Ret Spd

A/R (cv) tab

Range: 0 to 8 **Default:** 1 **ID:** `relachement_retour_vitesse`

Return speed of the position on the **release** curve during attacks. Works the same way as **Att Ret Spd**. Only acts if **Release Shape** is not 1.

> **A word from the author**
>
> This position tracker works like capacitors. You can tell it to come back faster, but finding the right speed is tricky, because it depends on the audio material. When you change the shapes, it's better for detection not to be peak but rather RMS (chapter 13): the position is no longer reset at each half-cycle, and the envelope works more easily in the right part of the curve. The vertical line in the shape graphs lets you check where you are.

> **To be confirmed by François**
>
> In the decoupled envelope and in Gamma, the curve for both directions is advanced at each sample, before the return of the inactive direction: the effective return speed there is Ret Spd − 1, so at the default setting (1) the position never comes down (it stays capped at Ret From), whereas in branching mode it returns to the start. In decoupled mode, the attack position also keeps advancing during releases before being capped. Is this intended?

The modes: logarithmic, linear, zero

The **Attack Mode** and **Release Mode** settings choose how the envelope reaches its target. They combine two ideas.

Toward the target or toward the extreme. In **logarithmic** mode (1, default), the envelope aims at the requested gain and covers a fraction of the distance to it each time: this is the usual exponential. In **zero** mode (0), it aims at the **extreme**: 0 (full reduction) for an attack, 1 (no reduction) for a release, ignoring the value of the requested gain. So it rushes toward its target without slowing down, reaches it in a finite time, overshoots it by at most one sample, and the opposite phase brings it back. Between 0 and 1, the envelope aims at a point between the target and the extreme.

Exponential or constant step. In **linear** mode (2), the envelope no longer advances by a fraction of the distance but by a **constant step**: it moves in a straight line. The step is such that in one set time, the envelope covers the fraction 1 − altitude of **full scale** (63.2% by default), whatever the distance to be covered. A linear attack from 1 to 0.5 therefore takes 0.5 / 0.632 = 0.79 times the attack time.

Mode value	Toward…	Linear part	Behavior
1 (default)	the target	0	logarithmic (classic exponential)
between 1 and 2	the target	mode − 1	logarithmic / linear blend
2	the target	1	pure linear
between 0 and 1	between the target and the extreme	0	exponential aiming at a point beyond the target
0	the extreme	0	zero
between −1 and 0	the extreme	−mode	zero / linear blend
−1	—	1	pure linear, identical to 2
between −2 and −1	the extreme	2 + mode	zero / linear blend, mirrored

Mode value	Toward…	Linear part	Behavior
−2	the extreme	0	identical to 0

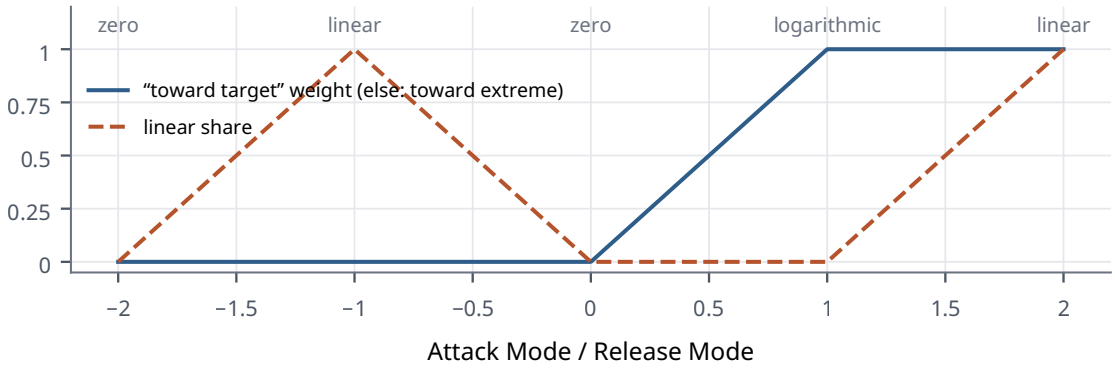


Figure 49 — The two components of the mode according to its value. The three pure modes blend two at a time, never all three at once.

The range from −2 to 2 thus lets you move continuously from one mode to another: from zero to logarithmic (0 → 1), from logarithmic to linear (1 → 2), and from zero to linear through the negative values (0 → −1). The part from −1 to −2 goes back toward zero mode; it adds nothing new, but saves you from having to go back through logarithmic to return to zero mode.

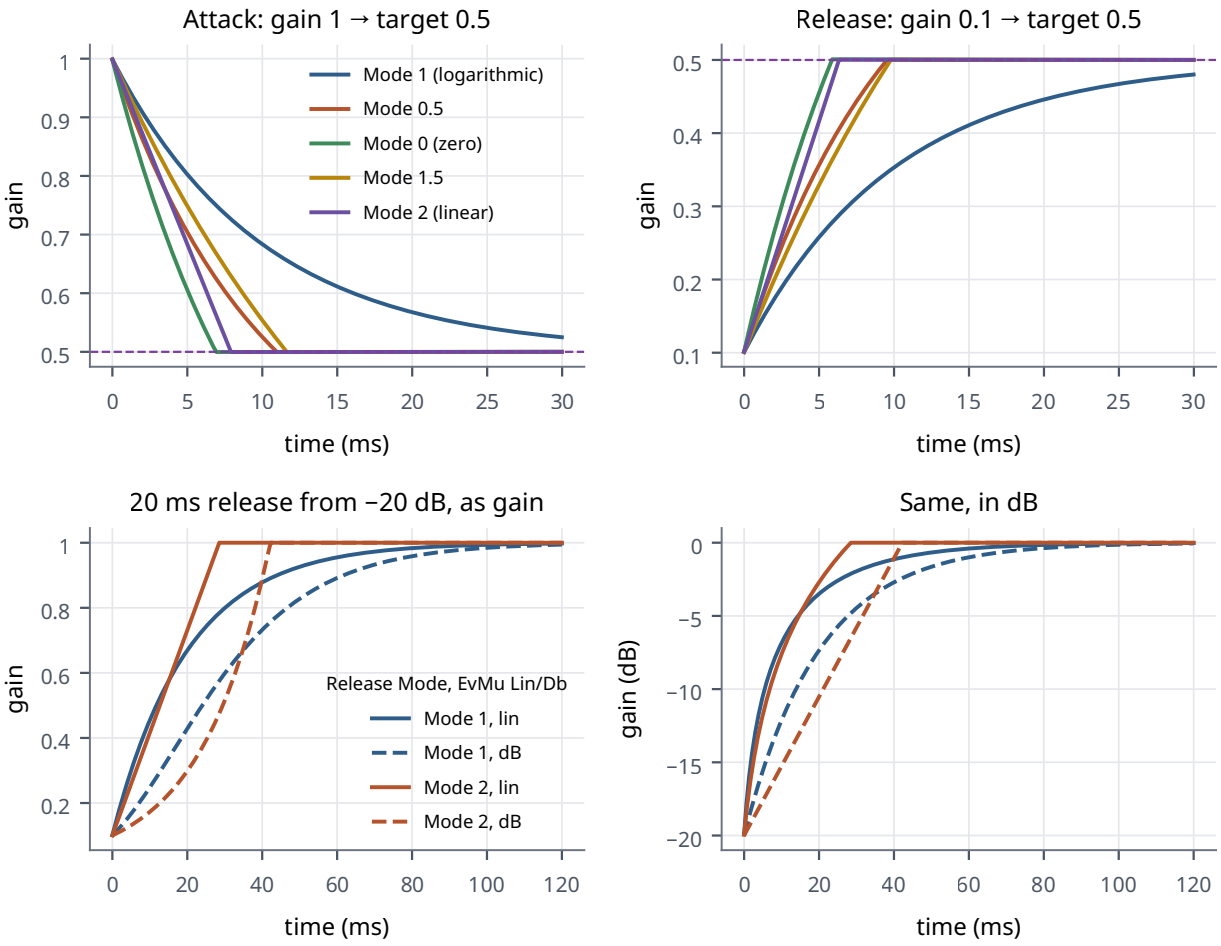


Figure 50 — Top: a 10 ms attack (requested gain 1 → 0.5) and release (0.1 → 0.5) for different modes. Bottom: a 20 ms release from −20 dB in logarithmic and linear mode, with the envelope computed in linear or in decibels (EvMu Lin/Db, chapter 11), plotted in gain, then in decibels.

In the figure, in zero mode, the 10 ms attack reaches 0.5 in 6.9 ms; in linear mode, in 7.9 ms; in logarithmic mode, it still hasn't gotten there after 30 ms.

Linear mode in decibels

When the envelope is computed in decibels (**EvMu Lin/Db** at 1), the constant step of linear mode is a step **in decibels**: 15 × (1 − altitude) dB per set time, or 9.48 dB per set time with the default altitude. A linear release in decibels with Release = 1 s therefore rises by 9.48 dB per second, at constant speed. It is the equivalent of a release expressed in decibels per second. Plotted in gain, this release is a curve that starts slowly and finishes fast (figure, bottom left, dashed orange curve).

Modes by envelope type

The modes do not apply to all the envelopes of the ENV tab (detailed table in chapter 11):

- the **classic branching** envelope accepts everything;
- the **classic decoupled** envelope applies Release Mode (on the first stage) but ignores Attack Mode;

- the **Mu** envelope ignores the linear part: any negative value acts like zero mode, any value above 1 like logarithmic mode; in decoupled mode, it also ignores Attack Mode;
- the **Gamma** envelope ignores both modes entirely.

Attack Mode

A/R (cv) tab

Range: −2 to 2 **Default:** 1 **ID:** `attaque_douceur`

Attack mode, from −2 to 2; 1 (default) = logarithmic. See the table above.

- **2** (or −1), linear: the attack goes down at constant speed. It reaches its target sooner than an exponential with the same time, without slowing down on arrival.
- **0** (or −2), zero: the attack rushes toward full reduction and stops when it crosses the requested gain. It is the most aggressive attack for a given time; it can overshoot the target by one step.
- Intermediate values blend these behaviors.

No effect with the decoupled envelope or on the Gamma envelope; partial effect on the Mu envelope (see above).

Release Mode

A/R (cv) tab

Range: −2 to 2 **Default:** 1 **ID:** `relachement_douceur`

Release mode, from −2 to 2; 1 (default) = logarithmic.

- **2** (or −1), linear: the gain rises at constant speed, in gain or in decibels depending on **EvMu Lin/Db**.
- **0** (or −2), zero: the release aims at 0 dB of reduction, regardless of the requested gain, and so rises much more decisively; it slightly overshoots the requested gain, then the attack brings it back.
- Values between 0 and 1 give a release “pulled” toward 0 dB, more energetic than logarithmic without being as brutal as zero mode.

Release Mode works with the branching envelope as well as the decoupled one; the Mu envelope only keeps its zero/logarithmic part; the Gamma envelope ignores it.

> **Both modes at zero**
>
> If Attack Mode and Release Mode are both at 0 with short times, the envelope overshoots its target at each sample in one direction, then in the other: it oscillates around the requested gain (for example between 0.448 and 0.552 for a target of 0.5, with times of 0.1 ms). This oscillation at the sample rate is a form of distortion. With longer times, the one-step overshoot becomes negligible.

A word from the author

Linear mode is a static addition or subtraction at each iteration: the curve really is straight. Zero mode, instead of going for the gain value, goes for the extreme: on the attack it aims for 0, on the release it aims for 1. It's much more aggressive: the compression tries to reach its target at all costs. The three modes blend with each other, two at a time.

♪ In the studio: choosing how it comes back

Logarithmic mode is a compressor's usual way of breathing: lively at first, settling down at the end. Linear mode, with a decibel envelope, gives a release at constant speed, in decibels per second, which the ear often perceives as more even on the long reductions of a bus or a master. **softube_weiss** uses it (**Attack Mode** and **Release Mode** at 2, **EvMu Lin/Db** at 1), as does **ao_dbcomp** for its release. Eight other presets set **Release Mode** to 2 with a linear envelope, including **waves_76**, **tr_670** and **acqua_eminence**; others stop halfway, like **acustica_cobalt** and **eli_arouser** at 1.5. Zero mode, for its part, is a character tool: a **Release Mode** between 0.5 and 0 makes the release more energetic and more abrupt, and the compression edgier. **waves_rvox** grazes this zone with both its modes at 0.94.

Recognizing a compressor's mode is part of the emulation work. In his video on analysis, François notes that the release of the compressor he is emulating seems linear, whereas Comprendre's is logarithmic, and he adjusts his setting accordingly (chapter 24).

The shape graphs

To the right of the history graph, two small graphs show the shape of the attack curve (top) and of the release curve (bottom):

- the horizontal axis covers **twice the set time**; the left half, tinted red, corresponds to the set time itself;
- the attack curve starts at the top and goes down (the distance remaining to be covered), the release curve starts at the bottom and goes up (the distance covered);
- at the right edge of the red zone, the attack curve passes at the height of the altitude, the release curve at the height 1 − altitude (36.8% and 63.2% by default).

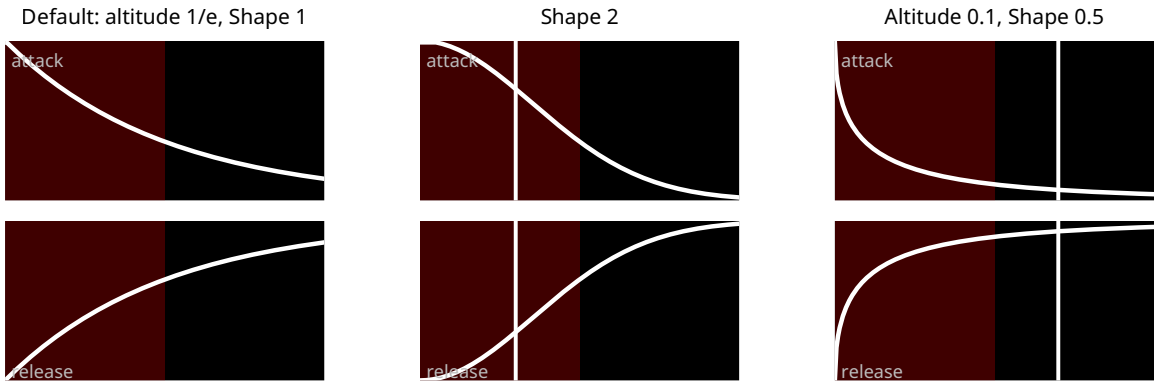


Figure 51 — The shape graphs for three settings. The vertical lines indicate a position on the curve (chosen here for the example).

Since the axis is in multiples of the set time, these graphs do not change when you turn **Attack** or **Release**: they only represent the **altitude** and the **shape**. They show neither the mode, nor the automatic times, nor the return, nor the envelope type.

Four curves are overlaid there, one per A/R set: **orange** for A/R (bs), **red** for A/R (in), **lavender** for A/R (la) and **white** for A/R (cv), drawn last. With the default settings, they are identical and only the white one is visible.

When the shape of an active follower is not 1, a **vertical line** in its color shows the envelope's current position on the curve: it crosses the curve where the envelope actually is. If several envelopes are blended in the ENV tab, the position shown is the average of their positions, weighted by the blend settings. The line stays at the right edge beyond twice the set time.

This line is the best tool for setting **Ret From** and **Ret Spd**: if it stays stuck to the left, each phase restarts from the beginning and the shape barely acts; if it stays stuck to the right, the envelope is permanently in the end of its curve. A line that roams across the whole width shows that the shape is really working.

> **To be confirmed by François**
>
> With Channels Unlink above 0, the calculation of the displayed position adds each channel's contribution twice: at 1, the vertical line sits at double the actual position (and always at double for A/R (bs)). For A/R (bs), whether the line is displayed depends on the bias master switch, not on whether this follower is active. Is this intended?

The four attack and release sets

The **A/R (in)**, **A/R (cv)**, **A/R (la)** and **A/R (bs)** tabs have the same layout and the same settings, with two differences: A/R (cv) has no times (those are the knobs on the main row) but has the two modulation settings instead; A/R (la) has zero times by default. The ids of the other three sets carry a prefix: **var_** for A/R (in), **la_** for A/R (la), **bias_** for A/R (bs) (for example **relachement_forme**, **var_relachement_forme**, **la_relachement_forme**, **bias_relachement_forme**).

	A/R (in)	**A/R (cv)**	**A/R (la)**	**A/R (bs)**
Id prefix	var_	none	la_	bias_
Follower controlled	ENV (in): "env" detector before the curve	ENV (cv): follower after the curve	ENV (la): lookahead envelope	ENV (bs): bias "env" detector
Times	**Attack**, **Release** in the tab, 10 ms	**Attack**, **Release** on the main row, 10 ms	**Attack**, **Release** in the tab, **0**	**Attack**, **Release** in the tab, 10 ms
First column	Attack, Release	mod attack, mod release	Attack, Release	Attack, Release
An attack is…	the detected level going up	the gain going down	the gain going down	the bias going up
Zero mode: extreme aimed at	0 dBFS on attack, silence on release	full reduction, then 0 dB	full reduction, then 0 dB	bias 1, then bias 0

	A/R (in)	A/R (cv)	A/R (la)	A/R (bs)
Active when	**env db** of DET (in) > −24 dB and a time > 0	Attack or Release > 0	envelope lookahead in use and a time > 0	**env db** of DET (bs) > −24 dB and a time > 0
Automatic-time constant	auto secs (DET (in))	auto secs (DET (in))	auto secs (DET (in))	b auto secs (DET (bs))
Curve in the shape graphs	red	white	lavender	orange
Pink line, modulation, opto designer	no	yes	no	no
Chapter	13	this chapter	22	20

All the other ranges and default values are identical from one set to another: altitudes at 1/e, shapes, returns and modes at 1, auto settings at 0.

For A/R (la), the zero default times mean that the lookahead envelope follows its target (the sliding minimum of the future envelope) instantly: this is the behavior of a classic lookahead. Setting a time in this tab adds an envelope follower on this anticipated envelope (chapter 22). Its automatic times are computed on this envelope, which is a gain, not a level.

Practical workflows

A natural automatic release

1. Leave **auto secs** (DET (in) tab) at 200 ms.
2. Set **Release** as usual, on a representative passage.
3. Raise **Release Auto** to 0.5: the release gets shorter on percussive passages and longer on sustained passages.
4. Turn on the pink line display (**Gfx Gains**, letter “a”) to see the ratio being applied.

Letting transients through

To make a fast compressor respect the attack of notes without lengthening the attack time:

- raise **Attack Shape** to between 2 and 4: the start of the attack is held back, and the reduction then arrives all at once;
- set **Att Ret From** to 0 if each hit should be treated the same way, regardless of the previous hit;
- switch detection to RMS or dial in a little detection envelope (chapter 13), so that the attack is not retriggered at each half-cycle; check with the white vertical line.

A more energetic release

- **Release Mode** between 0.5 and 0: the release aims at 0 dB of reduction and rises more decisively, with an abrupt ending;
- or **Release Mode** at 2 with **EvMu Lin/Db** at 1: a recovery at constant speed in decibels, whose duration is easy to calculate (9.48 dB per Release time, at the default);

- or a **Release Shape** above 1, to hold back the start of the recovery and then let go.

Choosing the time convention

If you are reproducing a compressor whose documentation gives a “90%” or “full” release time, set **Release Altitude** to 0.1 or 0.01 and enter the stated value directly in **Release**.

Starting ideas

These values are starting points, to be adjusted by ear to suit the tempo and the source.

- **Snare that cracks. Attack** around 25 ms, **Release** from 80 to 120 ms, **Ratio** 4; if the hit still sounds dull, raise **Attack Shape** toward 2.
- **Flattened snare, smashed drum room. Attack** from 0.1 to 0.5 ms, **Release** around 50 ms, a high ratio; add a little **Hold branch** (chapter 11) if the low end buzzes.
- **Mix bus. Attack** 30 ms and **Release** 100 ms, two positions of the SSL bus compressor: this is the pair chosen by six presets that target this unit or its spirit (**ssl_1_f**, **ssl_2_b_**, **tr_bus**, **cytomic_glue**, **ao_buster**, **acustica_sand3**). Keep a moderate ratio and a few decibels of reduction, and add a little **Release Auto**.
- **Vocals.** An attack of a few milliseconds, a release of 100 to 200 ms, then **Release Auto** at 0.5 with **auto secs** at 200 ms, the setting François considers the most natural.
- **Master.** An attack of 20 to 30 ms to let transients through, a long or automatic release, and possibly **Release Mode** at 2 with **EvMu Lin/Db** at 1 for a constant-speed recovery.

CHAPTER 13

Detection

Detection is the first stage of the compressor engine (chapter 9). It receives the absolute value of the detection signal and derives a **detected level** from it: a single number, which the transfer curve then turns into gain. Everything the curve "sees" goes through it.

So this is where you decide how the compressor **listens** to the music. A peak detector hears every snare crack and every pick attack, and reacts instantly. An RMS detector hears energy instead: the density of a phrase or a groove, a bit like the ear judging loudness. The classic hardware units settled the question once and for all: the dbx 160, for example, listened in RMS. Comprendre lets you blend the two, add a third measurement, and dose each one: a detection that lets the body of a bass drive the compression while keeping an eye on its attacks, or one that lets a drum bus breathe without letting the hardest hits slip through. Mix engineers will find a way to make the compressor react to the body of a vocal rather than to its attacks; mastering engineers, a way to follow the density of a track rather than its peaks; teachers, a way to show, on a single plugin, what separates peak detection from RMS detection; and anyone who imitates compressors, formulas born one by one from a specific imitation problem.

By default, detection is as simple as it gets: the instantaneous peak value of the signal, with no smoothing. The **DET (in)** tab lets you transform it profoundly. It computes three measurements in parallel, which François calls **streams**: a peak measurement (*peak*), a root mean square (*RMS*) and a detection envelope (*env*). Each stream can be dosed and reshaped, two streams can be combined by a formula, and then all three are merged into a single level.



Figure 52 — The DET (in) tab and its fourteen controls.

The signal arriving at detection has already been prepared by the INPUT tab: feedback, detection filters, sidechain, Mid/Side, sends from the dynamic EQs and the waveshaper, then the merging of the channels by **Channels Mix** for the linked compressor (chapter 14). The lookahead can also replace it with its anticipated version before it enters detection (chapter 22). The bias has its own detection, the **DET (bs)** tab, identical to this one in every respect (chapter 20).

How it works

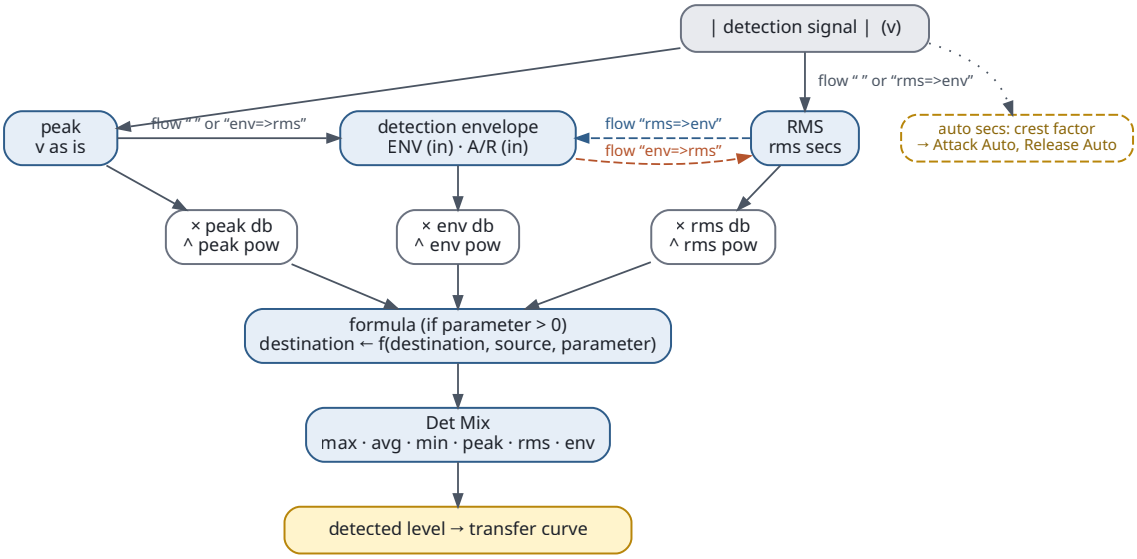


Figure 53 — The three detection streams, their controls and how they are merged.

At each sample, detection proceeds in this order:

1. it receives the absolute value v of the detection signal;
2. it computes the three streams: the **peak**, which is v itself; the **RMS**, a root mean square of v whose speed is set by **rms secs**; and the **detection envelope**, an envelope follower set by the ENV (in) and A/R (in) tabs. The **flow** control lets one feed into the other: the RMS into the envelope, or the envelope into the RMS;
3. each stream is multiplied by its **volume** (**peak db**, **rms db**, **env db**), then raised to its **power** (**peak pow**, **rms pow**, **env pow**);
4. if **parameter** is not zero, a **formula** replaces one stream (the *destination*) with a combination of itself, another stream (the *source*) and **parameter**;
5. **Det Mix** merges the streams into a single level: their maximum, their average, their minimum, or just one of them;
6. this level goes on to the transfer curve.

All these computations work on linear amplitudes: 1 corresponds to 0 dBFS, 0.5 to −6 dBFS, 0.1 to −20 dBFS. Nothing prevents a stream from exceeding 1 after its volume: the transfer curve then treats it as a signal louder than 0 dBFS.

With the default settings, **peak db** is at 0 dB, **rms db** and **env db** are at −24 dB, a value that excludes a stream, and **Det Mix** is set to *max*. The detected level is therefore exactly the instantaneous peak value v.

Peak or RMS: two ways of listening

Peak detection controls peaks better, but does not track perceived loudness. RMS detection, which measures power, gives a more relaxed compression, closer to perception. Each manufacturer picked a side. In 1976, David Blackmer's dbx 160 paired its VCA with a discrete-transistor RMS detector, in a feed-forward design; the API 2500 (2002) also uses an RMS detector. In the mid-1990s, the Focusrite Red 3, on the contrary, claimed a "true peak, not RMS averaging" response. The Weiss DS1 (1998), a digital mastering compressor, refuses to choose: its side-chain measures both peak **and** RMS, and picks its release, fast or slow, based on their ratio. Even the VU meter is close to an RMS detector, without being a true one. Comprendre inherits from all these schools: its three streams can be blended, and its **auto secs** control, which measures the crest factor to drive the automatic times (see the end of this chapter), echoes the Weiss idea.

The three streams

The peak: peak

The **peak** stream is the absolute value of the detection signal, sample by sample, with no hold and no smoothing. It is the fastest possible measurement: it follows every oscillation of the signal, drops back to zero at every zero crossing, and never misses a transient. It has no time control; only its volume (**peak db**) and its power (**peak pow**) modify it.

With peak detection, all the smoothing is left to the envelope follower placed after the curve. Chapter 9 shows what this implies on a sine wave: the target gain oscillates twice per cycle, and the release keeps working even during a sustained sound.

The root mean square: RMS

The **rms** stream is the square root of an average of the squared signal. Comprendre computes it as an **exponential average**: at each sample, the mean of the squares moves a small fraction of the way toward the square of the current signal, and then its square root is taken. It is not a sliding-window average: there is no duration beyond which the past is suddenly forgotten; it fades away gradually. The speed is the same going up and coming down.

rms secs

DET (in) tab

Range: 0 to 2 s **Default:** 0 s **ID:** `det`

Time constant τ of the root mean square.

- At **0** (default), the RMS is not computed: the rms stream is simply a copy of the peak stream.
- Above that, the larger the value, the slower and smoother the RMS. On a level step, the RMS reaches 63% of its final value in 0.5 τ and is within 3 dB of its final value after 0.7 τ. When the signal stops, it decays by roughly 4.3 dB every τ seconds (43 dB per second for τ = 100 ms).
- A short value (a few milliseconds) lets low-frequency ripple through, as the peak does: at 100 Hz, with τ = 10 ms, the RMS of a sine wave still ripples between −3.4 and −2.7 dB below its peak; with τ = 100 ms, the ripple becomes negligible.

On a sustained sound, the RMS of a sine wave is 3 dB below its peak: at the same threshold, RMS detection therefore triggers compression about 3 dB later than peak detection, and much later on transients, whose peak far exceeds the RMS.

The RMS is computed as soon as rms secs is greater than 0, even if **rms db** is at −24 dB: it can thus feed the detection envelope (**flow** control) without taking part in the mix itself. The time is expressed in real seconds and does not depend on oversampling. The knob is progressive: its midpoint corresponds to 0.125 s.

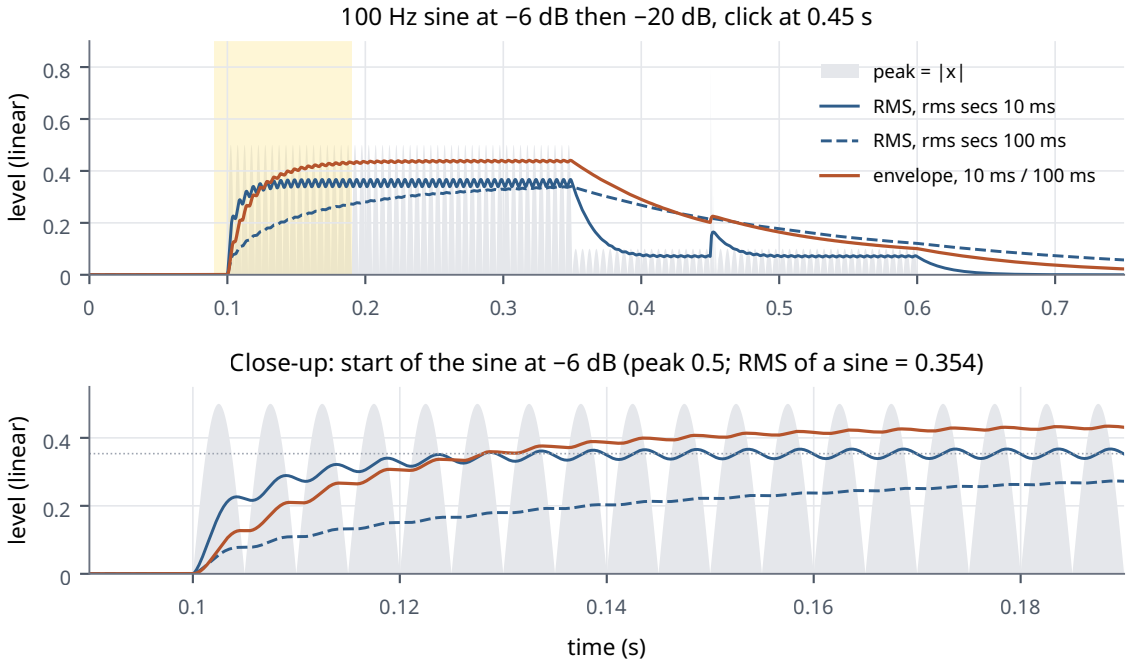


Figure 54 — The three streams on a test signal: a 100 Hz sine wave at −6 dB, then at −20 dB, with a short click. Bottom: close-up of the start of the sine wave.

The figure shows the three behaviors. The peak (in gray) follows every half-cycle. The fast RMS (10 ms) rises in about ten milliseconds toward 0.354, the RMS value of a sine wave with a peak of 0.5, and ripples slightly around it; the click only adds a small bump. The slow RMS (100 ms) takes several hundred milliseconds to settle, nearly ignores the click and keeps a long memory of the loud passage. The detection envelope (10 ms attack, 100 ms release) sits close to the peaks: it rises fast, with its attack, and falls slowly between two peaks, with its release; it settles slightly below the peak, at a height that depends on the ratio between its two times.

> **A word from the author**
>
> François isn't much of an RMS fan, for a simple reason: there is no point setting an attack faster than the RMS, since the whole system can never react faster than the RMS does. The transfer curve only transforms the detected level, moment by moment: if detection takes 20 ms to rise, the gain will take at least 20 ms to come down, whatever attack you set afterward.

> **♪ In the studio: listening to energy rather than peaks**
>
> Peak detection reacts to the slightest spike: that's what you need to catch drum hits or protect a maximum level. RMS detection reacts to energy: on a bass, a vocal or a mix bus, it lets the attacks through and works on the body of the notes, and you often get a more even compression that holds the level without jumping on every transient. To hear it, set **rms secs** between 20 and 50 ms, **rms db** to +3 dB and **peak db** to −24 dB, then compare with peak detection, listening mostly to the releases. Despite his reservations, François acknowledges it in his video on attack and release: a little RMS makes the motion of the compression more natural, especially when you reshape the release (chapter 12). Two presets listen in RMS only: **uhe_presswerk** (**rms secs** at 0.01 s, **rms db** at 0 dB, **peak db** at −24 dB) and **voxengo_polysquasher3** (**rms secs** at 0.006204 s, i.e. 6.2 ms, peak excluded). **massenbug_drc2_groove** also sets the peak aside, but pairs its RMS (**rms secs** at 0.01895 s, raised to +3 dB) with a detection envelope at 0 dB.

The detection envelope: env

The third stream is a full envelope follower, placed **before** the transfer curve: it smooths the level, not the gain. It has the same capabilities as the follower placed after the curve, except for the speed modulations, and its controls live in separate tabs: **ENV (in)** for the envelope type (linear or decibels, branching or decoupled, hold, Mu, Gamma) and **A/R (in)** for the times, shapes and modes. These controls are described in chapters 11 and 12; chapter 9 compares the two positions of the envelope.

A few points are specific to the detection envelope.

- **It only exists if you call on it.** It is computed only if **env db** is above −24 dB **and** the attack or the release of the A/R (in) tab is not zero. Otherwise, the env stream is simply a copy of the peak stream. Since A/R (in) defaults to 10 ms / 10 ms, just raising **env db** gives you a detection envelope with a 10 ms attack and a 10 ms release.
- **Its input** is the peak, or the RMS if **flow** is set to "rms=>env." It receives that stream before its volume and power.
- **Attack and release.** The attack of the A/R (in) tab sets how fast the envelope follows a rising level, and the release how fast it follows a falling level, as in a classic detector. To achieve this,

the follower works internally on the complement of the level (1 − level), which behaves like a gain, and the result is flipped back at the output.

- **It is clamped to 1**, that is, to 0 dBFS: a higher level is seen as 0 dBFS. The peak and the RMS, on the other hand, are not clamped. The **env db** volume, applied afterward, can then carry the stream above 1.

> **To be confirmed by François**
>
> In the walkthrough, you explain that for the envelope placed before the curve, attack and release are "reversed." In the current code, the internal inversion is compensated for: Attack in A/R (in) sets the rise of the detected level and Release its fall, as described above. Is that the intended behavior?

> **To be confirmed by François**
>
> The detection envelope is clamped to 0 dBFS (before its volume), whereas the peak and the RMS are not. With a high input gain or Headroom, or a hot sidechain, it can therefore no longer follow the level above 0 dBFS. Is this intended?

Who feeds whom: flow

By default, the three streams are computed side by side, all from the signal. The **flow** control lets you chain them.

flow — DET (in) tab

Choices: (empty) · rms=>env · env=>rms **Default:** (empty) **ID:** `pik_nik`

Chooses what feeds the RMS and the detection envelope.

- **(empty)**, the default: the RMS is computed on the signal, and the detection envelope follows the peak. The three streams are independent.
- **rms=>env**: the detection envelope follows the **RMS** instead of the peak. It receives the RMS before its volume and power: **rms db** therefore doesn't change what the envelope receives, and the RMS doesn't even need to take part in the mix (rms db can stay at −24 dB). If **rms secs** is 0, the RMS is a copy of the peak and this choice amounts to the empty choice.
- **env=>rms**: the RMS is computed on the **output of the detection envelope** (before its volume and power), and the envelope follows the peak. If the envelope is not active (env db at −24 dB, or A/R (in) times at zero), the env stream is a copy of the peak and this choice, too, amounts to the empty choice.

In every case, the stream that feeds another one remains available in the mix, with its own volume and its own power.

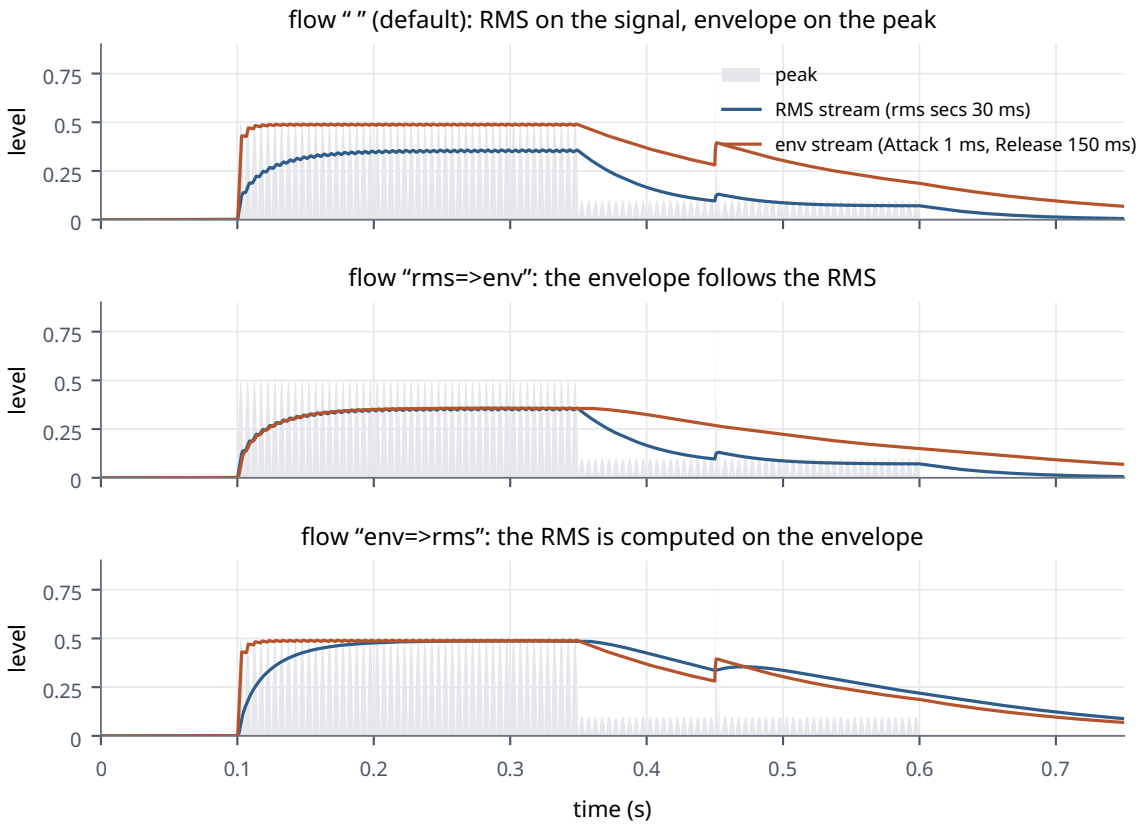


Figure 55 — The three flow choices on the same signal, with the same settings: a 30 ms RMS, and an envelope with a 1 ms attack and a 150 ms release.

With "rms=>env," the envelope cannot rise faster than the RMS feeding it: despite its 1 ms attack, it hugs the rise of the RMS. It does, however, fall with its own release, slower than the RMS. You thus get an RMS with a medium rise and a slow fall, something an RMS alone can't do, since its speed is the same in both directions.

With "env=>rms," it's the other way around: the envelope follows the peaks with its fast attack, and the RMS computed on it smooths the result further. The rms stream becomes a softened version of the envelope that keeps none of the signal's ripple.

> **A word from the author**
>
> François demonstrates the effect of "rms=>env" with a very slow RMS: the envelope, which follows the RMS, becomes very slow itself, whatever its attack.

> **♪ In the studio: smoothing the level rather than the gain**
>
> With the default peak detection, setting **Attack** and **Release** to 0 makes the gain follow every half-cycle of the signal: as François points out at the start of his video on detection, the compression then becomes very aggressive; it's distortion. Hand the smoothing over to the detection envelope and everything changes: the gain follows the curve exactly, applied to a level that is already smooth. This is another school, found in many commercial compressors (chapter 9), with its own character: the release time then depends on how far the note went above the threshold, so the loudest notes are held longer. Three presets take it all the way, with **Attack** and **Release** at 0 in the main row: **korneff_comp2** (**Det Mix** set to *env*, **env db** at 0 dB, **env pow** at 0.907, a 35 ms attack and a 180 ms release in the A/R (in) tab), **tonelib_multicomp** (envelope only, computed in decibels, **env pow** at 0.707, a 0.25 s release in the A/R (in) tab) and **massenbug_drc2_mix** (a 0.05 s RMS at −11 dB with **rms pow** at 0.571, and the detection envelope at 0 dB, with a 0.2538 s release and **Release Auto** at 0.5 in the A/R (in) tab).

Dosing each stream: volumes and powers

Each stream gets a volume, then a power, before the formula and the mix. These controls don't change what the streams pass to each other through **flow**: they only apply to each stream's contribution to the detected level.

Volumes

A stream's volume is an ordinary gain, in decibels. Raising a stream's volume by G dB amounts, for that stream, to lowering the threshold by G dB: it reaches the threshold G dB sooner. In a *max* mix, it is also how you decide which stream wins: you "push" a stream so that it rises above the others.

The lowest value, **exactly −24 dB**, has a special meaning: the stream is **excluded**. It is then 0, and it no longer takes part in the *max*, *avg* and *min* mixes of Det Mix. Just above, at −23.99 dB, the stream is included with a gain of 0.063.

peak db — DET (in) tab

Range: −24 to 24 dB **Default:** 0 dB **ID:** `pik_mix`

Volume of the peak stream. At 0 dB (default), the peak passes through unchanged; at −24 dB, it is excluded from the mix.

Lowering the peak by a few decibels in a *max* mix with the RMS or the envelope lets it "tickle" the detection: it only wins on transients whose peak exceeds the other streams by more than the reduction you set.

rms db

DET (in) tab

Range: −24 to 24 dB **Default:** −24 dB **ID:** `det_mix`

Volume of the rms stream. At −24 dB (default), the RMS is excluded from the mix, but it is still computed if **rms secs** is greater than 0, so that it can feed the detection envelope if **flow** calls for it.

A volume of +3 dB brings the RMS of a sine wave up to the level of its peak: a good starting point for moving from peak detection to RMS detection without changing the threshold.

env db

DET (in) tab

Range: −24 to 24 dB **Default:** −24 dB **ID:** `var_mix`

Volume of the env stream. At −24 dB (default), the detection envelope is excluded from the mix **and not computed at all**: this control also acts as an on/off switch. As soon as it goes above −24 dB, the detection envelope is computed, provided the attack or the release of A/R (in) is not zero.

Powers

The power is applied after the volume: the stream becomes (gain × stream)^p. In decibels, this amounts to **multiplying the level by p**: with p = 2, a stream at −10 dBFS becomes −20 dBFS; with p = 0.5, it becomes −5 dBFS. 0 dBFS (the value 1) is a fixed point.

- **p less than 1** brings low levels closer to 0 dBFS: detection becomes more sensitive, and compression starts lower and gets gentler. François talks about “softening” the detection.
- **p greater than 1** pushes low levels further away from 0 dBFS: compression starts higher and gets steeper, all the way to over-compression, where the output drops as the input rises.
- Above 0 dBFS (after a positive volume, for example), the effect is reversed: p greater than 1 makes the stream bigger.

Because it multiplies decibels, the power also acts on the stream’s dynamics: an envelope release falling at 40 dB per second falls at 80 dB per second with a power of 2. François sees it as a way to play with the nonlinearity of the envelope “vertically.”

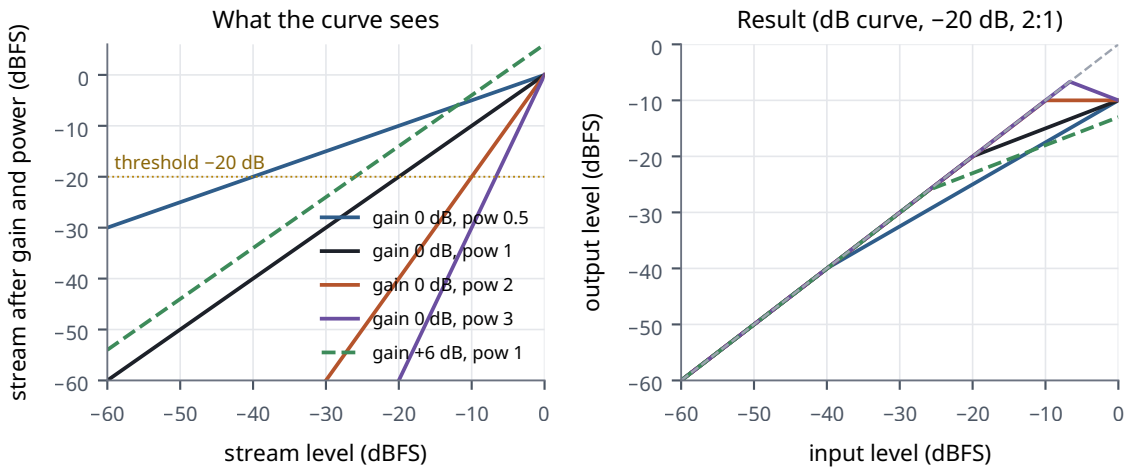


Figure 56 — Left: what the transfer curve sees, depending on a stream's volume and power. Right: the resulting curve with a dB curve (Threshold −20 dB, Ratio 2) when this stream is alone in the detection.

peak pow

DET (in) tab

Range: 0.125 to 8 **Default:** 1 **ID:** `pik_pow`

Power of the peak stream, from 0.125 to 8. At 1 (default), the stream is unchanged; values between 0.999 and 1.001 are snapped to 1. The value 1 sits a little past the midpoint of the knob's travel.

On a peak detection, a power greater than 1 widens the gap between the peaks and the troughs of each half-cycle; a power less than 1 narrows it.

rms pow

DET (in) tab

Range: 0.125 to 8 **Default:** 1 **ID:** `det_pow`

Power of the rms stream, with the same effects and the same range as **peak pow**. It is applied after **rms db** and does not change the RMS passed to the detection envelope by **flow**.

env pow

DET (in) tab

Range: 0.125 to 8 **Default:** 1 **ID:** `var_pow`

Power of the env stream, with the same effects and the same range as **peak pow**. It is applied after **env db** and does not change the envelope passed to the RMS by **flow**.

Going further: effective threshold and ratio

Suppose a single stream in the detection, with a volume of G dB and a power p, and a curve computed in decibels with a threshold T and a ratio R. A signal of level L (in dBFS) reaches the curve at the level $p \cdot (L + G)$. Compression therefore starts at $L = T/p - G$, and the slope of the output above that point is $1 - p \cdot (1 - 1/R)$, i.e. an effective ratio of $1 / (1 - p \cdot (1 - 1/R))$. With T = −20 dB and R = 2: p = 0.5 gives an effective threshold of −40 dB and a ratio of 1.33; p = 1.5, a threshold of −13.3 dB and a ratio of 4; p = 2, a threshold of −10 dB and an infinite ratio; p = 3, a negative slope of −0.5, so the output decreases as the input increases. With the default linear curve, the values differ but the trends are the same.

♪ In the studio: 'softening' or hardening

Powers change the compressor's temperament without touching the threshold or the ratio. A power below 1 brings the compression in earlier and more gradually: that's what François calls "softening" the detection. Try it on a vocal or a piano you want to hold without hearing the compressor kick in. A power above 1, on the other hand, makes the compression steeper above the threshold, all the way to limiting: enough to hold a bus or a bass firmly in place. Two imitations of plugins that emulate the Fairchild 670 use it boldly: **tr_670** sets **peak pow** to 1.987 and **acustica_ultramarine4** to 2.03, both in feedback (chapter 14) and with **Ratio** at 42. **softube_ds1mk3** settles for a touch, with **peak pow** at 1.101. In the other direction, **massenbug_drc2_mix** softens its RMS (**rms pow** at 0.571), **tonelib_multicomp** its envelope (**env pow** at 0.707), and **ao_buster** its slow RMS (**rms pow** at 0.822).

Merging the streams: Det Mix

Det Mix

DET (in) tab

Choices: max · avg · min · peak · rms · env **Default:** max **ID:** `mmm`

Chooses how the three streams, after volume, power and formula, are merged into a single detected level.

- **max** (default): the largest of the included streams. Each stream can take over in turn, depending on the moment: the peak on transients, the RMS or the envelope on sustained sounds.
- **avg**: the average of the included streams. It is an average of amplitudes, not of decibels.
- **min**: the smallest of the included streams, capped at 1 (0 dBFS).
- **peak**, **rms**, **env**: a single stream, whatever its volume. If that stream is at −24 dB, it is 0 and there is no compression anymore.

A stream is included in *max*, *avg* and *min* if its volume is above −24 dB. If all three streams are excluded, *max* and *avg* give 0: there is no compression at all.

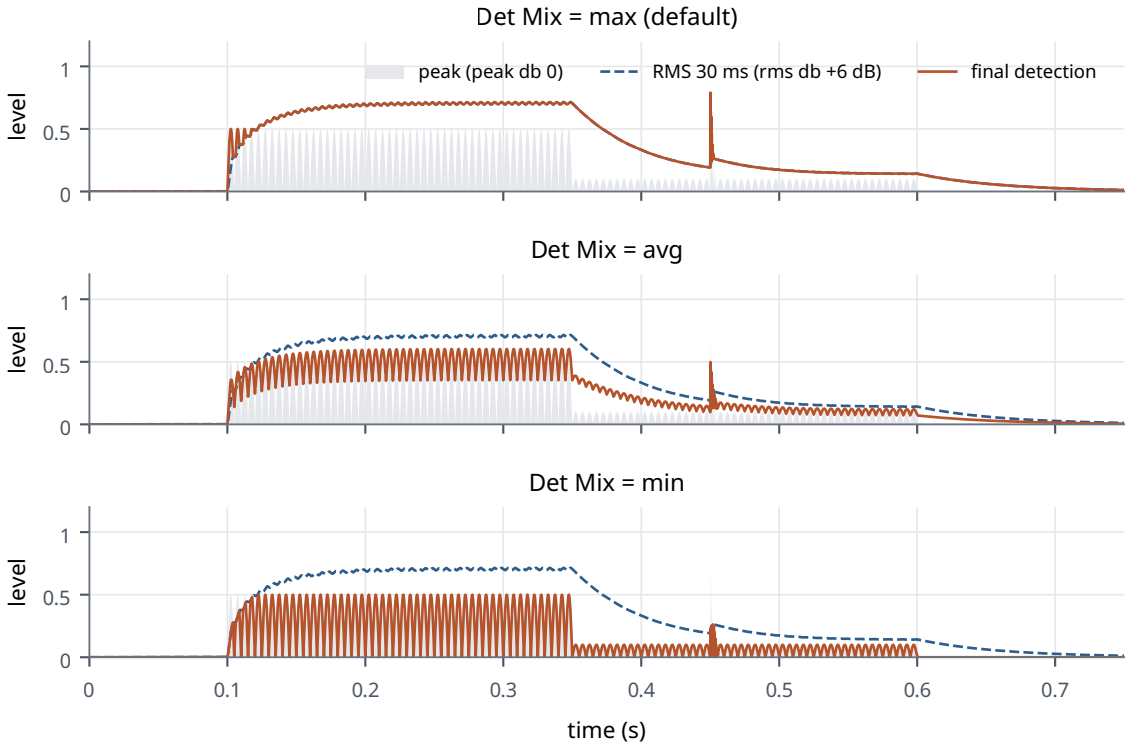


Figure 57 — Det Mix with the peak at 0 dB and a 30 ms RMS raised by 6 dB. In max, the RMS leads and the peak only takes over on the click; in avg, the detection ripples halfway between the two; in min, the peak leads, capped by the RMS.

To be confirmed by François

With **Det Mix** set to *min*, the detection starts from 1: it never exceeds 0 dBFS, and if all three streams are at −24 dB, it stays at 1 permanently, which forces constant compression, as for a signal at 0 dBFS. Is this intended?

A word from the author

In his demo, François sets a very fast RMS and a slower envelope, then raises the volume of the RMS. In *max*, the RMS takes over at times; at other times, it's the envelope that smooths the detection again. You can thus "play with the volumes" of the three detectors, and bring the peak back up a little so it joins in.

> **♪ In the studio: peak for punch, RMS for body**
>
> Blending the streams gives the compressor two pairs of ears. On a drum bus or a full mix, an RMS of a few tens of milliseconds lets the compressor follow the density of the playing, while a slightly lowered peak only takes over on the hits that really stand out: the snare keeps its attack, but a hit harder than the others gets caught. Several presets rely on this division of labor. **tdr_kotelnikov**, an imitation of a digital compressor that itself combines a peak path and an RMS path, doses all three streams: **peak db** at −1 dB, **rms db** at −1 dB with **rms secs** at 0.0212 s, and **env db** at −6 dB. **slate_fgstress** lets a slow RMS lead (**rms secs** at 0.06966 s, **rms db** at +1.92 dB, **rms pow** at 0.976) and keeps the peak at −1.72 dB, ready to take over on transients. **ao_buster** pairs a peak raised to +1.92 dB, a very slow RMS (**rms secs** at 0.3526 s, **rms db** at −10.95 dB) and a formula (**parameter** at 1, with **source** set to *rms*). Load them, set **Gfx Detects** to 1 and watch, hit after hit, which stream wins.

Formulas

Formulas modify one stream, the **destination**, based on another one, the **source**, and on an adjustable value, **parameter**. They act after the volumes and powers, just before Det Mix. They are only active if **parameter** is greater than 0.

In what follows, d stands for the value of the destination stream, s for the value of the source stream (both after their volume and power) and p for the value of **parameter**. The result cannot be negative (it is brought back to 0 if it is), but it is not capped. It replaces the destination stream, which then takes part in Det Mix like the others.

> **A word from the author**
>
> François added the formulas over time, whenever he needed one to imitate a particular compressor: each one solved a specific problem. He nonetheless regards them as a tool for specialists, one he tries to do without as much as possible. You really have to know what you're doing.

destination

DET (in) tab

Choices: peak · rms · env **Default:** peak **ID:** `op_dst`

Stream modified by the formula: **peak** (default), **rms** or **env**.

The destination keeps its status in the mix: if its volume is at −24 dB, the result of the formula is excluded from *max*, *avg* and *min*. For a formula to have any effect, the destination must therefore take part in the mix (volume above −24 dB) or be selected on its own by **Det Mix**.

source

DET (in) tab

Choices: peak · rms · env **Default:** peak **ID:** `op_src`

Stream used as the source of the formula: **peak** (default), **rms** or **env**. It is taken after its volume and power: a source whose volume is at −24 dB is 0. If the source and the destination are the same stream, s = d.

The source is not modified by the formula: it keeps taking part in the mix according to its own volume.

parameter

DET (in) tab

Range: 0 to 1 **Default:** 0 **ID:** `op_par`

Value p used by the formula, from 0 to 1.

- At **0** (default), the formula is disabled, whichever formula is selected.
- In the formulas that compare or shift (7 to 12), p is an amplitude: 0.5 corresponds to −6 dBFS, 0.1 to −20 dBFS, 0.01 to −40 dBFS.
- In formulas 5 and 6, the exponent uses not p but $P = (p^2 + 1)^8$, which grows very fast: 1.08 for p = 0.1; 1.62 for 0.25; 5.96 for 0.5; 35.5 for 0.75; 256 for 1.

Since p cannot be zero without disabling the formula, formulas 11 and 12 cannot copy the source exactly: with p = 0.001, “d=max(s,p)” copies the source with a floor of −60 dBFS.

formula

DET (in) tab

Choices: d-=sp · d*=sp · d=dp/s · d+=sp · d=1-d^sp · d=d^sp · d=s<p?s:0 · d=s>p?s:O · d+=p · d-=p · d=max(s,p) · d=min(s,p) · d=ssp+d(1-s)(1-p) · d=min(s,p)s+(1-m)d · d=max(s,p)s+(1-m)d · d=1-sp **Default:** d-=sp **ID:** `op_sig`

Chooses the formula, out of sixteen. The table below gives, for each one, the menu label, the computation actually performed and its effect. By default, the menu is set to the first formula, but it does nothing as long as **parameter** is 0.

No.	Menu	Actual computation	Effect
1	`d-=sp`	d ← d − s·p	subtracts the weighted source: the destination dips as the source rises
2	`d*=sp`	d ← d · s · p	product: in decibels, the levels add up; the detection becomes much lower, unless the volumes are high
3	`d=dp/s`	d ← d · p / s (d · p if s = 0)	destination/source ratio: for example, peak divided by rms measures the instantaneous crest factor
4	`d+=sp`	d ← d + s·p	adds the weighted source
5	`d=1-d^sp`	d ← 1 − d^(s·P) if s > 0	inverted curvature of the destination, driven by the source

No.	Menu	Actual computation	Effect
6	`d=d^sp`	d ← d^(s·P) if s > 0	curvature of the destination, driven by the source
7	`d=s<p?s:0`	d ← d if s < p, otherwise 0	gate: the destination is kept only while the source stays below p
8	`d=s>p?s:0`	d ← d if s > p, otherwise 0	gate: the destination is kept only if the source exceeds p
9	`d+=p`	d ← d + p	constant upward offset
10	`d-=p`	d ← d − p	constant downward offset (without going below 0)
11	`d=max(s,p)`	d ← max(s, p)	the destination is replaced by the source, with a floor p
12	`d=min(s,p)`	d ← min(s, p)	the destination is replaced by the source, with a ceiling p
13	`d=ssp+d(1-s)(1-p)`	d ← s^2·p + d·(1 − s)·(1 − p)	driven crossfade: toward d·(1 − p) when the source is weak, toward p as it approaches 1
14	`d=min(s,p)s+(1-m)d`	m = min(s, p); d ← s·m + d·(1 − m)	crossfade from the destination toward the source, more pronounced the stronger the source, limited by p
15	`d=max(s,p)s+(1-m)d`	m = max(s, p); d ← s·m + d·(1 − m)	same crossfade, with at least the proportion p
16	`d=1-sp`	d ← 1 − s·p	inversion: the lower the source, the higher the detection

A few words of caution:

- formulas 5, 6 and 16 can produce a detection that is **high when the signal is low**: in formula 6, a source close to zero (but not zero) gives an exponent close to zero, and d^0 = 1, i.e. 0 dBFS; formula 16 equals 1 when the source is zero. The compressor then compresses the quiet passages;
- formulas 7 and 8 turn the detection into an all-or-nothing signal: the destination disappears abruptly when the condition is no longer met, which makes the compressor release;
- formulas 9 and 10 add or subtract a fixed amplitude: with “d+=p” and p = 0.1, the detection never drops below a threshold of −20 dB.

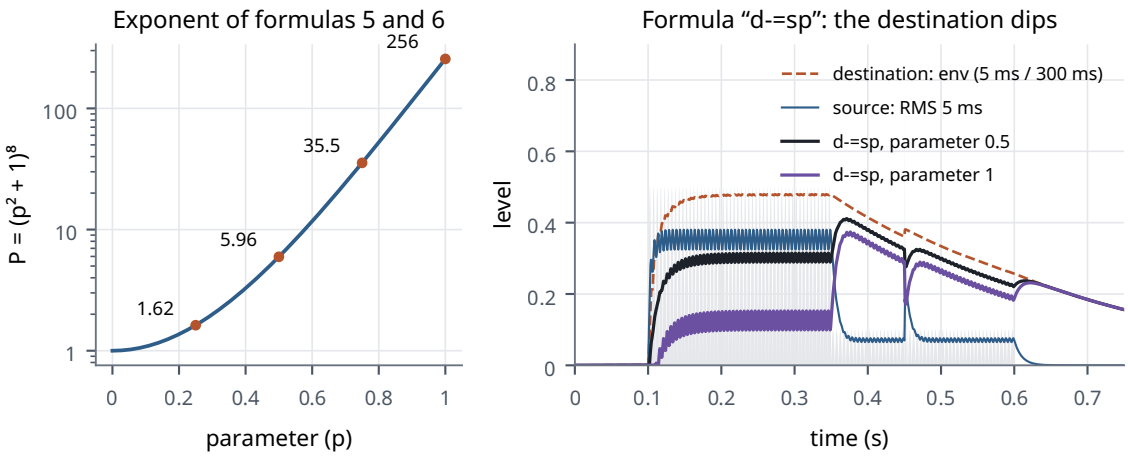


Figure 58 — Left: the exponent P used by formulas 5 and 6. Right: the “d-=sp” formula with the detection envelope as the destination and a fast RMS as the source: the envelope dips during the sustained sound and comes back up when the source dies away.

To be confirmed by François

The labels of four formulas don’t match the computation: “d=sp?s:O” (with the letter O) keep the destination d, not the source s; “d=1-d^sp” and “d=d^sp” use the exponent $s\cdot(p^2 + 1)^8$, not $s\cdot p$. The manual describes the actual computation. Should the labels be fixed, or the computation?

A separate control: auto secs

The first control of the tab doesn’t belong to detection. It sets the speed of an auxiliary measurement made by the engine, the **crest factor**, which is used only by the automatic times **Attack Auto** and **Release Auto** (chapter 12).

auto secs

DET (in) tab

Range: 0 to 2 s **Default:** 0.2 s **ID:** `rms`

Time constant of the crest-factor measurement, which drives **Attack Auto** and **Release Auto**. Despite its identifier, this control has nothing to do with the RMS stream of the detection, and it doesn't change the detected level. Nor is it the speed of the auto knee (**Knee Auto Secs**, chapter 10).

The engine tracks two quantities, both computed on the square of the signal entering the engine (before any of the detection settings): a **peak** that rises instantly and falls with the set time constant, and a **root mean square** with the same constant. Their ratio is the square of the crest factor, written C^2. When Release Auto is at 1, the release time becomes $2 / C^2$ times the set time (for a branching follower; chapter 12 describes the decoupled case); likewise for the attack with Attack Auto. Between 0 and 1, the time is a blend of the two.

- On a sine wave, C^2 is 2: the automatic time equals the set time.
- On a very steady signal, such as a square wave, C^2 is 1: the time is doubled.
- On a percussive signal, C^2 is large: the times are shortened considerably.
- At **0**, the peak and the average are instantaneous and equal: C^2 is 1, and the automatic times are twice the set times.

On a dynamic signal, the longer the value, the longer the peak of a hit is held relative to the average: C^2 grows and the automatic times get shorter.

The DET (in) setting serves three followers: the detection envelope (A/R (in)), the follower after the curve (A/R (cv)) and the lookahead envelope (A/R (la)). The bias has its own setting in DET (bs). This control only has an effect if one of the relevant Attack Auto or Release Auto controls is above 0. The knob is progressive: its midpoint corresponds to 0.125 s.

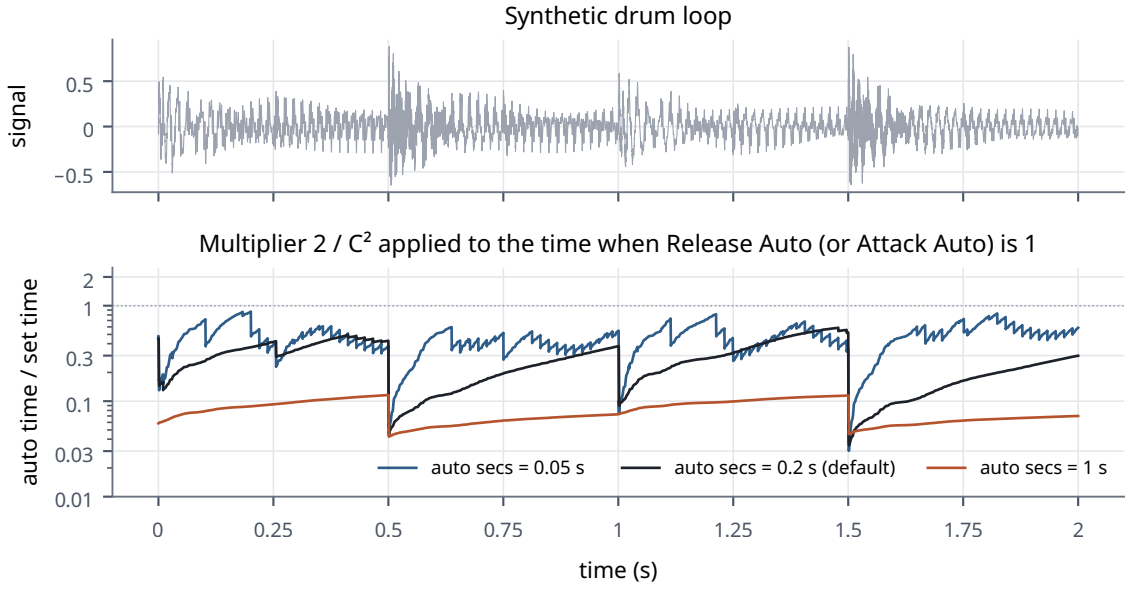


Figure 59 — The multiplier applied to the times (Auto at 1) on the drum loop, for three values of auto secs. Each hit makes the multiplier drop; it then recovers all the more slowly as auto secs gets longer.

> **A word from the author**
>
> The default value, 200 ms, is the one you find in many compressors. For François, it is the most natural setting: the brain takes about 170 ms to integrate a sound, and the remaining 30 ms leave a small margin.

Setting detection with the graphs

You can't hear detection: you only hear what the compression does with it. To set it, the safest approach is to look at it. The graph history draws the three streams (after their volume, power and formula) as thin, semi-transparent red lines, and the final detection as an opaque red line. The **Gfx Detects** control in the GRAPHS tab (described in chapter 4) chooses what is displayed:

- at **1**, the history shows only the three streams and the final detection: all the other curves disappear. This is the ideal position for setting up the DET (in) tab;
- at **2**, it shows only the final detection: you can see at a glance what goes on to the curve;
- at **0** (default), the streams and the final detection are shown along with all the other curves;
- at **−1**, only the final detection remains among the detection curves; at **−2**, none of them is shown.

A few tips for reading them:

- the three streams are the **same color**. To tell one from another, go by its shape (the peak is jagged, the RMS is smooth, the envelope rises fast and falls slowly), or temporarily exclude the others by lowering their volume to −24 dB: an excluded stream stays pinned to the bottom of the graph;
- the final detection has the same color as the input level bars;
- each column of the history is an **average** over the duration set by **Gfx Secs**. The peak curve therefore shows the average of the absolute values, which is lower than the peak: on a sine wave, about 3.9 dB below its peak, versus 3 dB for the RMS. The RMS curve can thus appear **above** the peak curve. Slow the history down with **Gfx Secs** to see the rises in detail;
- on the transfer curve graph, the x-coordinate of the **green dots** is the level entering detection, not the final detected level. As soon as the detection is smoothed or blended, the dots drift away from the purple curve, even with zero attack and release.

To **hear** the signal entering detection (before the DET (in) tab), use **Monitor Input**, in the INPUT tab (chapter 14).

Example settings

An RMS detection

1. Set **rms secs** between 10 and 50 ms: any shorter and the RMS ripples on the low end; any longer and it gets sluggish.
2. Raise **rms db** to 0 dB (or to +3 dB to make up for the gap between the peak and the RMS of a sine wave) and lower **peak db** to −24 dB. You can also leave the volumes alone and set **Det Mix** to *rms*.

3. If you stayed at 0 dB, lower the **Threshold** by about 3 dB to get the same amount of compression on sustained sounds.
4. There's no point setting an **Attack** shorter than the rise time of the RMS (see the word from the author above).

Peak for transients, RMS for body

1. Set a 20 to 50 ms RMS at 0 dB, and leave the peak at 0 dB.
2. Lower **peak db** by a few decibels (−6 dB, for example), leaving **Det Mix** on *max*.
3. On sustained sounds, the RMS leads the detection; the peak only takes over on transients where it exceeds the RMS by more than 6 dB. Set **Gfx Detects** to 1 to see which one wins, and when.

Smoothing the level before the curve

1. Lower **peak db** to −24 dB and raise **env db** to 0 dB.
2. Set the attack and the release in the **A/R (in)** tab, for example 5 ms and 100 ms.
3. To do all the smoothing before the curve, set **Attack** and **Release** in the main row to 0: the gain then follows the curve exactly, applied to the smoothed level. You can also keep both followers, with different times.

Chapter 9 compares the two positions of the envelope and their effect on the release.

A softer or harder detection

To extend the compression zone downward without touching the threshold, lower the power of the main stream (to 0.7, for example): the compression starts earlier and becomes more gradual. To harden the compression above the threshold, raise it (1.5 to 2): with a dB curve, a power of 2 turns a ratio of 2 into a limiter beyond the effective threshold.

Bias detection: DET (bs)

The **DET (bs)** tab contains the same fourteen controls, with the same ranges, the same default values and exactly the same computation. It applies to the bias detector, which works on the bias source signal, channel by channel; its detection envelope is set by the ENV (bs) and A/R (bs) tabs, and its **auto secs** only drives the automatic times of A/R (bs). Its display is set by **Gfx Detects Bias**. Chapter 20 describes its role.

Starting ideas

To learn to recognize detection types by ear, nothing beats the supplied presets: load one, open the **DET (in)** tab (it turns yellow when the preset modifies it), set **Gfx Detects** to 1, and listen while you watch. Chapter 25 presents these presets family by family.

- **A pure RMS detection**: **uhe_presswerk** or **voxengo_polysquasher3**. Compare with a preset that leaves detection alone, such as **softube_fet**.
- **Peak and RMS blended**: **tdr_kotelnikov**, **slate_fgstress**, **ao_buster**.

- **All the smoothing before the curve**: **korneff_comp2**, **tonelib_multicomp**, **massenbug_drc2_mix**.
- **A detection hardened by the power**: **tr_670**, **acustica_ultramarine4**.
- **Automatic times set differently**: **softube_weiss** sets **auto secs** to 0.1 s, with **Attack Auto** at 0.5 and **Release Auto** at 1; **bx_townhouse** takes it up to 0.4 s, with **Attack Auto** and **Release Auto** at 1; **waves_la2a** brings it down to 0.075 s, with **Release Auto** at 0.5.
- **On a vocal**, start from a 20 to 30 ms RMS raised to +3 dB and a peak at −6 dB, with **Det Mix** on *max*: the body of the voice leads, and the peak keeps watch over the attacks.
- **On a mix or a master**, try a slower RMS (50 to 100 ms) and a peak set well back, so that the compressor follows the density of the track rather than its peaks.
- **For teaching**, set **rms secs** to 30 ms and **rms db** to 0 dB, then play the same excerpt with **Det Mix** on *peak*, then on *rms*, at the same threshold: the difference in behavior, which the graphs show at the same time, often speaks louder than any diagram.

CHAPTER 14

What Is Measured, What Is Compressed: Feedback, Filters, and Target

In an ordinary compressor, the signal you measure and the signal you attenuate are one and the same. Comprendre splits them into two branches, introduced in chapter 6: the **detection branch** builds the signal the compressor *measures*; the **target branch** chooses the part of the sound that *receives* the gain reduction. This chapter describes them in detail.

This separation is one of Comprendre's great strengths: in his video on input and output selection, François presents it as one of the plugin's distinctive features. It lets you choose what the compressor **listens to** in order to make its decisions, independently of what it **processes**. Musically, this opens doors you would usually look for across several different units: a de-esser, which listens only to the sibilants and attenuates only the top end of a vocal; a bus compressor that the kick no longer makes pump; cymbals that duck on every kick hit; compression that affects only the sides of the stereo image, or only the center. Add to that the choice between **feed-forward** and **feedback**, two philosophies that have divided the history of compressors and that Comprendre lets you blend, and linear-phase **FIR** filters for those who do mastering.

The relevant controls are spread over three tabs:

- the **INPUT** tab sets up the detection: feedback, filters, Mid/Side, sends from the dynamic EQs and the waveshaper, and the merging of the channels;
- the **OUTPUT** tab sets up the target: filters, Mid/Side, and the amounts of the dynamic EQs and the waveshaper;
- the **MAIN** tab holds the share of the **Sidechain** in the detection and the choice of filter type (Linkwitz-Riley or FIR), common to both branches.

The other controls in these tabs are described elsewhere: the auto-volumes **In->Sc**, **In->Out** and **Out->Sc** in chapter 8; **Comp Clip**, **Comp Clip Mode**, **Comp Db**, **dW | Dw** and the monitors **Monitor Input**, **Monitor Output** and **Monitor Comp** in chapter 7. This chapter does, however, explain how to use these monitors to set up the two branches.



Figure 60 — The INPUT tab: the controls of the detection branch.



Figure 61 — The OUTPUT tab: the controls of the target branch.



Figure 62 — The MAIN tab: the share of the sidechain in the detection, and the three FIR filter controls (number of taps, window, beta).

How it works: two branches

Both branches start from the same signal, which this chapter calls the **internal signal**: the input after **Input Db**, the input auto-volumes, the **L/R to M/S** conversion, the **Ch.0 Db** and **Ch.1 Db** gains, the **Pre Clip** and the **Headroom** (chapter 7). The following figure shows, in the program's exact order, what each branch does to it.

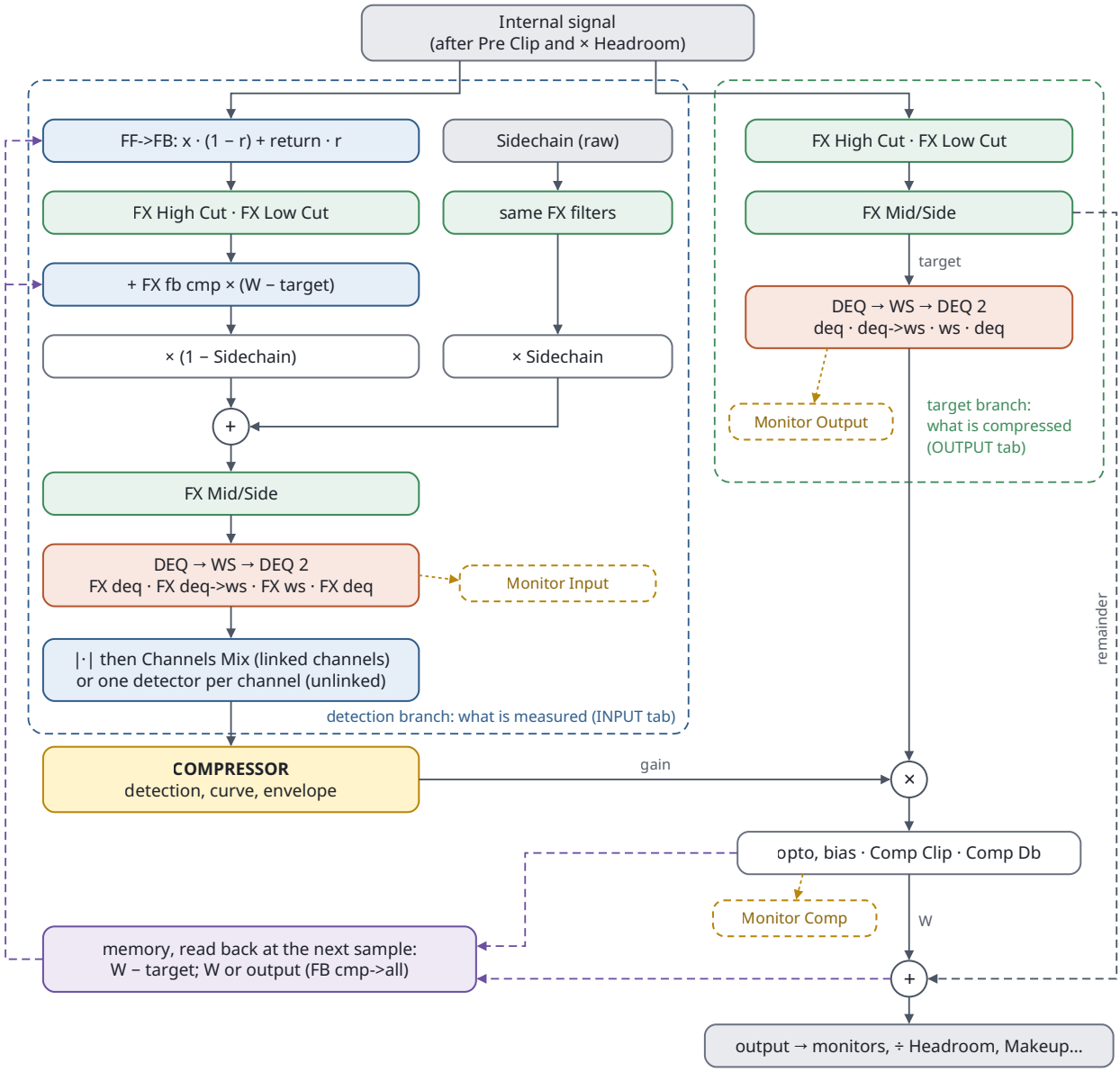


Figure 63 — The two branches in detail. W denotes the compressed target. In purple, the returns from the output to the detection (feedback), read back at the next sample; in dotted gold, the monitors' listening points.

The detection branch chains, for each channel:

1. the mix between the internal signal and the output of the previous sample (**FF->FB**);
2. the band filters (**FX High Cut** and **FX Low Cut**);
3. the optional addition of the “compression feedback” (**FX fb cmp**);
4. the mix with the sidechain signal, run through the same filters (**Sidechain**, MAIN tab);
5. the detection Mid/Side (**FX Mid/Side**);
6. the sends to the dynamic EQ **DEQ**, the waveshaper **WS** and the second dynamic EQ **DEQ 2**;
7. finally, if the channels are linked, taking the absolute value and merging the channels into a single signal (**Channels Mix**); if they are unlinked, each channel feeds its own detector.

The result enters the compressor proper (detection, transfer curve, envelope: chapters 9 to 13), which produces the gain. Just before step 7, the bias can still modify the detection (part V).

The target branch chains:

1. the band filters of the OUTPUT tab, which split the internal signal into a **target** and a **remainder**;
2. the target Mid/Side, which removes part of the stereo image from the target as well; that part joins the remainder;
3. the **DEQ**, the **WS** and the **DEQ 2**, dosed by the **deq**, **deq->ws**, **ws** and **deq** knobs of the OUTPUT tab;
4. multiplication by the compressor's gain, then the opto and the bias if they are engaged (chapters 15 and 21), **Comp Clip** and **Comp Db** (chapter 7): the result is the **compressed target**, labeled W in the figure;
5. adding the remainder back (or the **dW | Dw** mix, chapter 7): this is the compressor's **output**, which continues on to the monitors, the Headroom return, the Makeup, and so on.

Finally, two values are stored for the next sample: the compressed target (or the full output), and the difference between the compressed target and the uncompressed target. These are the two **returns** that the detection uses in feedback mode.

> **The default settings**
>
> By default, none of these processes is engaged: the detection measures the internal signal as is, the target is the entire signal and the remainder is zero. Comprendre then behaves like a classic compressor. The amounts in the OUTPUT tab are at 1, but they only have an effect if the DEQ, the WS or the DEQ 2 are themselves set up.

> **Computed only when needed**
>
> The plugin runs the full detection path (filters, FX fb cmp, Mid/Side, sends) only if one of these processes is engaged: FX High Cut above 0 Hz, FX Low Cut below 20,000 Hz, FX Mid/Side other than 0.5, FX fb cmp other than 0, an active send to the DEQ, the WS or the DEQ 2, or a modulation in the MOD & LA tab that relies on them. Otherwise, the detection boils down to the FF->FB mix and the mix with the sidechain. Likewise, the split between target and remainder is only computed if a filter or the Mid/Side of the OUTPUT tab is engaged.

The detection branch (INPUT tab)

Feed-forward and feedback

A **feed-forward** compressor measures the signal coming in. A **feedback** compressor measures the signal going out, that is, the signal it has already compressed. Both topologies exist, and they don't behave the same way.

In feedback, the detector sees a signal that the compression has already reduced. With identical settings, it therefore measures a lower level, and the gain reduction is smaller than in feed-forward: the compression is gentler, and the ratio actually obtained between input and output is lower than the displayed ratio. To get the same amount of compression back, you have to lower the threshold or raise the ratio.

A program can't measure the output of a sample before it has computed it. In Comprendre, the "output" measured in feedback is therefore that of the **previous sample**, at the internal rate (oversampling included). This one-sample delay is imperceptible.

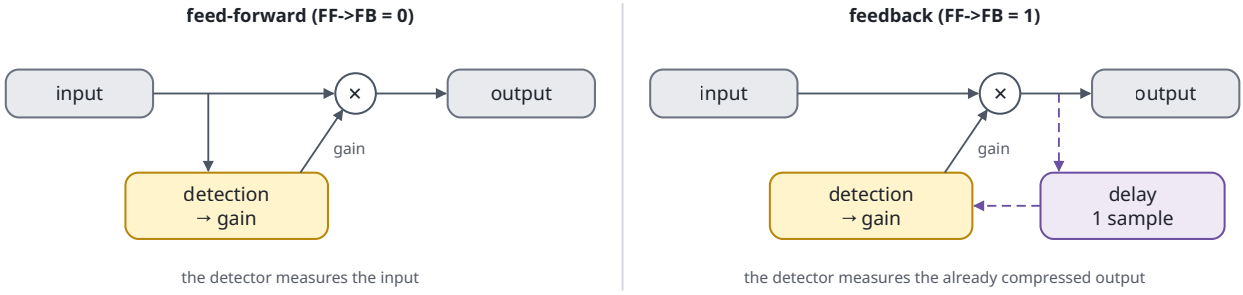


Figure 64 — Feed-forward and feedback. In feedback, the detector receives the output of the previous sample.

FF->FB

INPUT tab

Range: 0 to 1 **Default:** 0 **ID:** `retour`

Moves the detection gradually from feed-forward to feedback. The detection receives the mix x · (1 − r) + return · r, where x is the internal signal, r the value of **FF->FB** and "return" the output of the previous sample, as selected by **FB cmp->all**.

- At **0** (default): pure feed-forward; the detection measures the internal signal.
- At **1**: pure feedback; the detection measures the previous output.
- In between: the mix applies to the **signals** themselves, sample by sample, before the absolute value is taken. It is not a mix of two levels or of two envelopes.

The mix takes place **before** the detection filters: with the Linkwitz-Riley filters, it is the mix that gets filtered. It is then multiplied by (1 − **Sidechain**): with Sidechain at 1, FF->FB no longer has any effect.

The return contains the compressed target **after Comp Clip and Comp Db**, but neither the Makeup, nor the Post Clip, nor the Output Db, nor the Dry/Wet, which come later in the chain. It is expressed in the same level domain as the internal signal (Headroom included). In practice, this means that in feedback, raising **Comp Db** makes the detector measure a louder signal, and therefore increases the compression.

FB cmp->all

INPUT tab

Range: 0 to 1 **Default:** 0 **ID:** `cible_retour`

Chooses what the return used by **FF->FB** contains:

- at **0** (default), the **compressed target** only, after Comp Clip and Comp Db (*cmp*);
- at **1**, the compressor's **full output**: the compressed target plus the remainder, or the **dW | Dw** mix if it is engaged (*all*);
- in between, a linear crossfade between these two signals.

As long as the target is the entire signal (no filter or Mid/Side in the OUTPUT tab) and **dW | Dw** is at 0, the two signals are identical and this control has no effect. In any case, it only acts if **FF->FB** is above 0.

Example: if the target is limited to the top end, position 0 makes the detector measure only the compressed highs; position 1 makes it measure the whole output sound, uncompressed lows included.

FX fb cmp

INPUT tab

Range: −16 to 16 **Default:** 0 **ID:** `retour_moymax_fbs`

Adds to the detection the **difference** between the compressed target and the target before compression, taken at the previous sample and multiplied by the value of the control. This difference is exactly what the compression (with the opto, the bias, Comp Clip and Comp Db) changed in the target: François calls it another kind of feedback.

- At **0** (default), nothing is added.
- At **+1**, if the detection and the target are the same signal, the detection becomes roughly the compressed target: you get a behavior close to feedback.
- **Negative** values give an "inverted feedback": when the compressor reduces the gain, the detection increases instead of decreasing, which reinforces the compression even further.
- Beyond ±1, the effect is amplified; the extremes (±16) are very aggressive and should be set by ear.

Unlike **FF->FB**, the difference is added **after** the detection filters: it is not filtered. It is then multiplied, like the rest of the internal signal, by (1 − **Sidechain**).

The knob is very progressive around its center position: the value is $\pm 16 \cdot |2p - 1|^8$, p being the position from 0 to 1. At three quarters of its travel, it is only 0.06; at 90%, 2.7. For a precise value, type it into the field.

To be confirmed by François

In FIR mode (**X Filters Taps** other than 0), as soon as a detection filter is engaged, the code feeds the filter with the internal signal alone: **FF->FB** then no longer has any effect on the detection, while **FX fb cmp** stays active. Is this intended?

A word from the author

In feedback, the detector's input is the previous output, and the result is gentler; the gradual transition of FF->FB lets you dose that gentleness. François does, however, advise against combining the two feedback options (FF->FB and FX fb cmp) with the FIR filters, because they rely on the last output sample.

Going further

Let x be the internal signal, s the sidechain, C the target before compression, W the compressed target and S the output (W + remainder). At the end of each sample, the plugin stores return = W · (1 − a) + S · a (a = **FB cmp->all**) and diff = W − C. At the next sample, the detection of a channel is:

d = [F(x · (1 − r) + return · r) + k · diff] · (1 − e) + F(s) · e,

where F is the detection filter (the identity if no filter is engaged), r = **FF->FB**, k = **FX fb cmp** and e = **Sidechain**. With a lookahead (chapter 22), the "future" chain has its own memories, fed by its own output.

Feedback or feed-forward: two schools

The first compressors were feedback designs. The Gates Sta-Level, the Altec 436, the LA-2A, the 1176 and, later, the Manley Variable Mu all measure the already-reduced signal; variable-mu tube compressors, such as the Fairchild 670, also derive their control signal from the output. Among the pioneers of feed-forward is the dbx 160 (1976): the control signal is taken from the input and "fed forward" to the VCA, which allows faster attacks and infinite compression without excessive distortion or oscillation. It is the dominant topology today; feedback, for its part, theoretically cannot reach an infinite ratio. The difference can be heard, and manufacturers that offer both describe it in similar terms. On the API 2500 (2002), which has an **Old/New** switch, feedback is "smoother" and feed-forward "raw, aggressive." On the elysia alpha compressor, feedback is "soft and even" and tops out at a ratio of 2.5:1, while feed-forward is "louder and harder." The SSL bus compressor is a hybrid: its side-chain has a feedback topology, but it is fed from a feed-forward position. The Manley manual, for its part, takes sides: "most modern compressors use a feed-forward circuit, which we find not very musical."

♪ In the studio: dosing the gentleness of feedback

Feedback lends itself to situations where you want to hold a sound without constraining it: a vocal you level gently, a bass you want even, a bus that has to stay supple. Raise **FF->FB** gradually while listening: as the detector hears a signal that has already been reduced, the compression relaxes, and you have to lower the threshold or raise the ratio to get the same reduction back. Between 0 and 1, you dose this gentleness, something the switches on hardware units don't let you do. Nine presets use it. Two imitations of plugins that emulate the Fairchild 670, **tr_670** and **acustica_ultramarine4**, are pure feedback (**FF->FB** at 1), as are **acqua_magenta**, **acqua_opal**, **acqua_titanium**, **waves_rvox** and **ssl_1_f**; **waves_76** and **caelumçsmmov** stop halfway, at 0.5. Notice that several of them set **Ratio** to 42: in feedback, the ratio actually obtained is lower than the displayed ratio. The SSL bus compressor presets, incidentally, show two approaches: **ssl_1_f** switches to feedback (with a light **Bias input mul** of 6.5), while **ssl_1_b**, **ssl_2_b** and **ssl_2_b_** stay feed-forward and rely on **Bias input mul** (75, 38.57 and 30.52), the control François added to the bias after studying the SSL schematic (chapter 20).

The sidechain

The **Sidechain** control sits in the **MAIN** tab, but it acts at the heart of the detection branch.

Sidechain

MAIN tab

Range: 0 to 1 **Default:** 0 **ID:** `externe`

Doses the share of the **sidechain** signal in the detection. The detection receives the internal signal (with its feedbacks) multiplied by (1 − e), plus the sidechain multiplied by e, e being the value of the control:

- at **0** (default), the sidechain is not used for detection;
- at **1**, the compressor measures only the sidechain;
- in between, the two signals are weighted and then added together, sample by sample.

The sidechain enters the detection **raw**: it goes through neither **Input Db**, nor the **L/R to M/S** conversion, nor the **Ch.0 Db** and **Ch.1 Db** gains, nor the **Pre Clip**, nor the **Headroom**, nor the feedbacks. It does, however, go through the same detection filters as the internal signal (with their own memories), and then, mixed with the internal signal, through the FX Mid/Side, the DEQ, WS and DEQ 2 sends and the Channels Mix. It is never audible, except through **Monitor Input**.

With no signal on the sidechain input (bus not connected or silent), raising this control therefore **weakens** the detection: at 1, the compressor no longer measures anything and stops compressing.

This control only concerns detection. The auto-volumes **In->Sc** and **Out->Sc** (chapter 8) and **Analyse SC** (chapter 24) use the sidechain whatever its value.

Since the **Headroom** doesn't apply to the sidechain, it changes the balance between the internal signal and the sidechain when **Sidechain** is between 0 and 1; at 1, it no longer has any effect on the compression (chapter 7). Likewise, in **L/R to M/S** mode, the sidechain stays in left/right: channel 0 of the detection then mixes the Mid of the internal signal with the left channel of the sidechain.

> **Going from 0 to a positive value**
>
> When a sidechain bus is connected, moving **Sidechain** from 0 to a positive value (or back) changes the configuration of the internal oversampler, which resets the plugin's filters. Avoid automating this control around 0 during playback.

> **To be confirmed by François**
>
> With an active lookahead (negative **la delay**) and **Sidechain** above 0, the "future" detection reads the sidechain ahead of time only if a detection process is engaged (filter, Mid/Side, FX fb cmp, send to the DEQ/WS); otherwise, it reads the sidechain without any advance. The sidechain's behavior therefore changes as soon as you engage a detection filter. Is this intended?

The side-chain: listening to one source, processing another

In 1972, at the Sound Exchange studio in New York, Eventide showed the prototype of the Omnipressor, designed by Richard Factor; the models followed in 1973 and 1974. Its idea: separate the level detector from the VCA, so that one source can control the gain of another. Eventide presents it as "the invention of the side chain"; that is the manufacturer's claim, and the unit relied on dbx VCAs and RMS detectors. The uses multiplied: DJs send their mic into the side-chain to duck the music when they talk; in dance music, the bass is "side-chained" by the kick. De-essing relies on the same separation, with an equalized side-chain that makes the compressor more sensitive to sibilants. The Weiss DS1 (1998), whose "DS" stands precisely for de-essing, extends it to compression limited to one band, in high-pass, low-pass or band-pass. Comprendre brings these ideas together in a single plugin: a sidechain input, detection filters, and a filtered target.

♪ In the studio: making room

The sidechain is first and foremost a way to make room. Put Comprendre on the bass, send the kick to its sidechain input and set **Sidechain** to 1: the bass ducks on every hit, for the length of the release, and the two instruments stop fighting over the low end. The same setup on a music bed, with a voice-over in the sidechain, ducks the music when the voice speaks; a fairly long release brings it back up smoothly between phrases. Since the sidechain goes through the detection filters, an **FX Low Cut** around 120 Hz leaves it only the low end of the kick, even if the send also carries the slap of the drumhead or the cymbal bleed from the recording. And since **Sidechain** is a blend control, intermediate values mix the two ways of listening: the bass reacts both to its own level and to the kick. To set all this up, listen to **Monitor Input**: it's the only way to hear the sidechain as the compressor measures it.

Detection filters

The two controls **FX High Cut** and **FX Low Cut** define a frequency band. You'll find them again, working the same way, in the OUTPUT tab for the target. Their names call for careful reading:

- **FX High Cut** is the cutoff frequency of a **high-pass** filter: it removes what lies **below** the chosen frequency. It is the **lower** limit of the band that is kept.
- **FX Low Cut** is the cutoff frequency of a **low-pass** filter: it removes what lies **above**. It is the **upper** limit of the band that is kept.

In everyday vocabulary, a "high cut" cuts the highs; here, it's the other way around. The simplest approach is to read them as "high-pass cutoff" and "low-pass cutoff."

Each one is disabled at its default value: 0 Hz for FX High Cut, 20,000 Hz for FX Low Cut. Their combinations give five configurations:

Settings (f_1 = FX High Cut, f_2 = FX Low Cut)	Part kept
both at their default value	the whole signal: no filter
FX High Cut alone	what lies above f_1 (high-pass)
FX Low Cut alone	what lies below f_2 (low-pass)
both, f_1 lower than f_2	the band between f_1 and f_2 (band-pass)

Settings (f_1 = FX High Cut, f_2 = FX Low Cut)	Part kept
both, f_1 higher than f_2 (or equal)	**two bands**: below f_2 and above f_1; the zone between f_2 and f_1 is removed

The last configuration is intentional: placing FX High Cut above FX Low Cut inverts the selection and gives two bands. When the two frequencies are equal, the plugin applies this configuration; the removed zone is then empty, and the filtering becomes almost transparent (a slight dip of about 2.5 dB at that frequency with the Linkwitz-Riley filters).

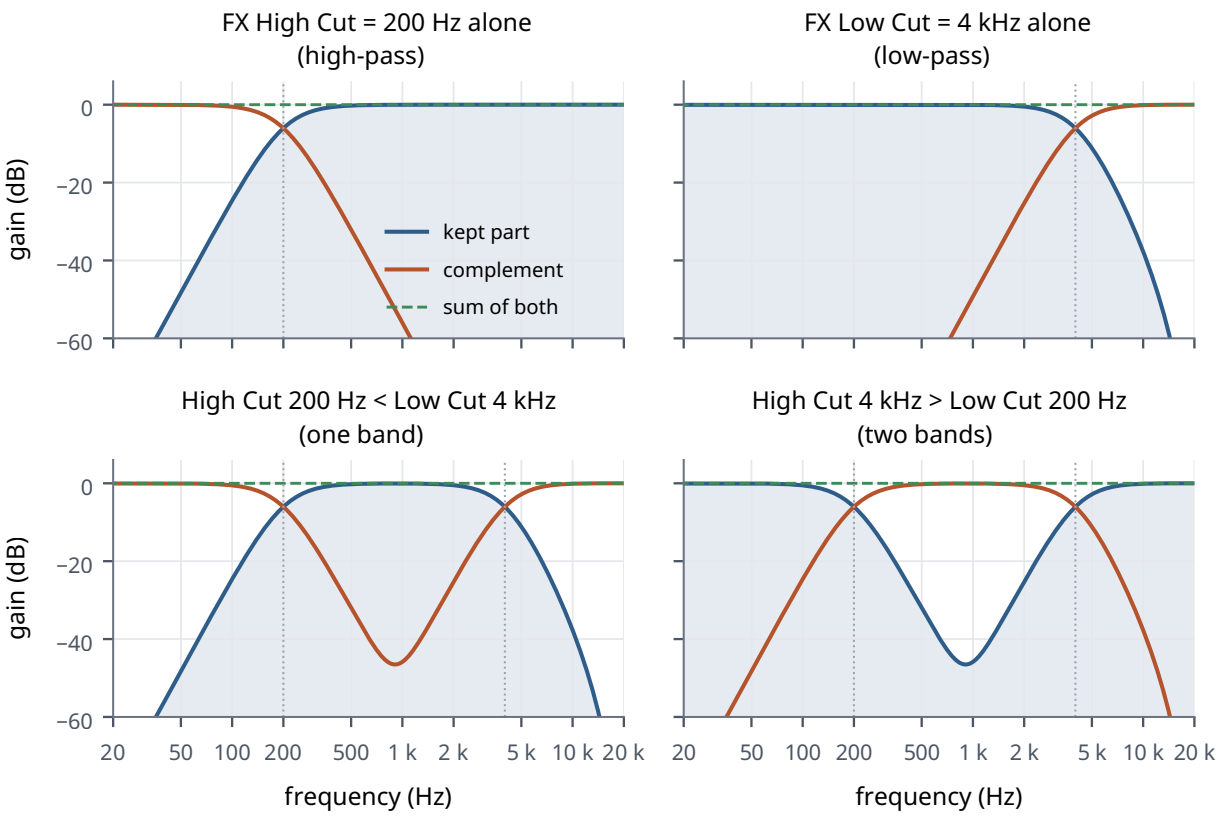


Figure 65 — The four active configurations, with the default Linkwitz-Riley filters. In blue, the part kept; in orange, its complement; in dotted green, their sum. For the target, the complement is the "remainder," which passes through uncompressed.

By default, these filters are 4th-order **Linkwitz-Riley** filters (24 dB per octave slope, −6 dB at the cut-off frequency). The **X Filters Taps** control in the MAIN tab replaces them with linear-phase **FIR** filters. The two families are compared later in this chapter.

The frequency knobs are very progressive: the frequency is $20{,}000 \cdot p^4$ Hz, p being the position from 0 to 1. A quarter of the travel corresponds to 78 Hz, halfway to 1,250 Hz, and three quarters to 6,328 Hz. For a precise frequency, type it into the field.

FX High Cut

INPUT tab

Range: 0 to 20000 Hz **Default:** 0 Hz **ID:** `ciseau_bas`

Cutoff frequency of the detection **high-pass** filter: the compressor no longer measures what lies below this frequency. At 0 Hz (default), this filter is disabled.

Combined with **FX Low Cut**, it forms one band (if it is lower) or two bands (if it is higher), as described above.

The same filter is applied to the sidechain signal before it is mixed in. The **FF->FB** mix is filtered too (with the Linkwitz-Riley filters); the difference added by **FX fb cmp** is not.

This is the classic setting to keep the low end from making a compressor "pump": with a high-pass at 100 or 150 Hz in the detection, a kick or a bass no longer triggers the compression on its own, while the compressor keeps acting on the whole signal. Since this filter is never heard (except with Monitor Input), it doesn't change the tone.

FX Low Cut

INPUT tab

Range: 0 to 20000 Hz **Default:** 20000 Hz **ID:** `ciseau_top`

Cutoff frequency of the detection **low-pass** filter: the compressor no longer measures what lies above this frequency. At 20,000 Hz (default), this filter is disabled.

Combined with **FX High Cut**, it forms one band or two bands. It also applies to the sidechain.

Example: to make the compressor react only to the low end (the kick in a mix, or a kick received through the sidechain), set FX Low Cut around 120 to 150 Hz. The compression can then be applied to something other than what is measured, for example to the highs only (OUTPUT tab).

> **♪ In the studio: a bus that no longer pumps**
>
> On a mix bus or a master, the kick and the bass carry most of the energy: with full-band detection, they are the ones that trigger the compression, and everything else drops on every hit. An **FX High Cut** around 80 to 150 Hz in the INPUT tab makes the compressor deaf to that energy: it keeps processing the whole mix, but reacts mostly to what happens above it, the vocals, the guitars, the snare. Listen to **Monitor Input** to hear what it measures, then compare with and without the filter on a busy chorus: the mix often stops breathing to the rhythm of the kick. The **toneproject_unisum** preset cuts its detection below 85 Hz in this way (**FX High Cut** at 85 Hz); it is the only supplied preset that filters its detection, and on top of that it makes its detector hear a dynamic EQ that is never applied to the sound (chapter 18). The idea isn't new: chapter 18 tells how radio and the API 2500 fought the bass that decides everything, and how other units, from the LA-2A to the Distressor, equalized what their detector hears. Conversely, if you want deliberate pumping, the kind heard in electronic productions where everything ducks under the kick, let the detection hear the low end, or even have it listen to the kick alone through the sidechain (see the examples at the end of the chapter).

Detection Mid/Side

The **FX Mid/Side** control, present in both tabs, acts on the stereo image. It computes the **Mid** M, the average of the channels, and for each channel its deviation from the Mid, which is that channel's **Side**. It then rebuilds each channel with two weights: channel = M · (Mid weight) + (channel − M) · (Side weight). In stereo, the deviation of the left channel is S = (L − R)/2 and that of the right channel is −S.

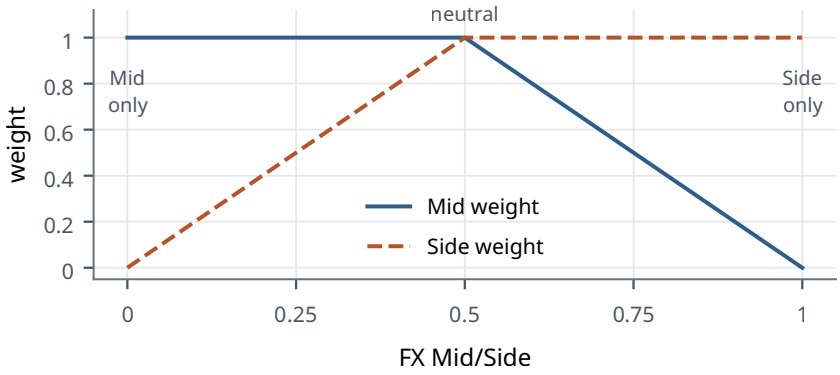


Figure 66 — Mid and Side weights as a function of the FX Mid/Side value.

FX Mid/Side INPUT tab

Range: 0 to 1 **Default:** 0.5 **ID:** `cotes`

Chooses which part of the stereo image the compressor measures.

- **0**: Mid only. Both detection channels equal (L + R)/2.
- **0.5** (default): neutral.
- **1**: Side only. The first channel equals (L − R)/2, the second its opposite; their absolute values are equal.
- Between 0 and 0.5, the Side is progressively attenuated: the detection image narrows toward the center. Between 0.5 and 1, it's the Mid that is attenuated.

It acts after the filters, the feedbacks and the mix with the sidechain, on signals that are still signed.

Use: make the compressor react to what's in the center (a vocal, a bass) or to what's on the sides (reverb, wide instruments). The compression itself always applies to the target defined in the OUTPUT tab.

Special cases:

- On a **mono** signal, the Side is zero. Below 0.5, the control has no effect; above, the detection is attenuated; at 1, it is zero and the compressor no longer compresses.
- If **L/R to M/S** (MAIN tab) is engaged, the channels are already the Mid and the Side, and FX Mid/Side mixes them: at 0, both channels equal (M + S)/2, that is, half of the left channel. In this mode, to measure only the Mid or only the Side, use **Channels Mix** on Ch0 or Ch1 instead.

> **Going further**
>
> For a value c of the control, the Mid weight is min(1, 2 · (1 − c)) and the Side weight is min(1, 2c). Both equal 1 at c = 0.5.

The DEQ, WS and DEQ 2 sends to the detection

The dynamic EQs **DEQ** and **DEQ 2** (chapter 18) and the waveshaper **WS** (chapters 16 and 17) exist in each of the two branches, with the same settings: those of their own tabs. Only the amounts differ; they are specific to each branch. In both branches, the order is the same: DEQ, then WS, then DEQ 2, after the Mid/Side.

In the detection, they are disabled by default. Enabling them lets the compressor measure an equalized or saturated signal without your hearing it: the detection reacts to what the dynamic EQ *would do*, or to what the saturation would add. A negative amount inverts their effect on the detection.

These sends only have an effect if the module concerned is itself active: at least one band set up in the DEQ or the DEQ 2, or a non-neutral setting in the WS (see the corresponding chapters).

> **Two FX deq knobs**
>
> The INPUT tab shows the name **FX deq** twice: the first one, to the left of **FX deq->ws**, doses the DEQ; the second, to the right of **FX ws**, doses the DEQ 2. Likewise, the OUTPUT tab shows **deq** twice, in the same order.

FX deq

INPUT tab

Range: −1 to 1 **Default:** 0 **ID:** `retour_moymax_deq`

The first **FX deq**: doses the effect of the **DEQ** (DEQ tab, chapter 18) on the detection. If d_0 is the detection entering the DEQ and a the value of the control, the detection becomes $d_0 + (DEQ(d_0) - d_0) \cdot a$.

- **0** (default): the detection is not equalized.
- **1**: the detection goes entirely through the DEQ.
- **−1**: inverted effect: the bands the DEQ would boost are cut by the same amount, and vice versa.
- Intermediate values: proportional amount.

The knob is extremely progressive around its center position: the value is $\pm|2p - 1|^8$. At three quarters of its travel, it is only 0.004; at 90%, 0.17. For a precise value, type it into the field.

FX deq->ws

INPUT tab

Range: 0 to 1 **Default:** 0 **ID:** `ideqfet`

Chooses what **enters** the detection waveshaper:

- **0** (default): the signal as it was before the DEQ;
- **1**: the signal after the full DEQ, whatever the value of the first **FX deq**;
- in between: a linear mix of the two.

This control lets the WS react to the DEQ without applying the DEQ itself to the detection: FX deq at 0, FX deq->ws at 1. Without an active DEQ, it has no effect. It only affects the detection if **FX ws** is other than 0.

FX ws

INPUT tab

Range: −1 to 1 **Default:** 0 **ID:** `retour_moymax_fet`

Doses the waveshaper's **saturation product** (WS tab) added to the detection. The WS is not inserted as is: the plugin computes the difference between the WS output and its input, that is, what the saturation adds, processes it according to the settings of the F (ws) tab (chapter 16), then adds it to the detection, multiplied by this control.

- **0** (default): nothing is added.
- **1**: the product is added in full.
- **−1**: it is subtracted.
- Intermediate values: proportional amount.

The product is added to the detection as it comes out of the DEQ (dosed by the first FX deq), even if what enters the WS is different (FX deq->ws control). The knob's travel is just as progressive as that of FX deq.

FX deq

INPUT tab

Range: −1 to 1 **Default:** 0 **ID:** `retour_moymax_dep`

The second **FX deq**: doses the effect of the **DEQ 2** (DEQ 2 tab, chapter 18) on the detection, after the DEQ and the WS. Same formula and same values as the first FX deq: 0 (default) = not applied, 1 = applied, −1 = inverted.

There is no equivalent of FX deq->ws for the DEQ 2, which comes after the WS.

A word from the author

At the moment the detection is computed, the compressor's envelope isn't known yet. The detection's DEQ and WS therefore use the envelope of the previous sample. François accepts this one-sample offset, which causes no trouble in practice; he likens it to feedback, with extra latitude.

Going further

With d_0 the detection after the Mid/Side, the sends are chained as follows:

- $d_1 = d_0 + (\mathrm{DEQ}(d_0) - d_0) \cdot a_1$;
- WS input: $w = d_0 \cdot (1 - i) + \mathrm{DEQ}(d_0) \cdot i$;
- $d_2 = d_1 + a_2 \cdot P(w)$, where $P(w)$ is the saturation product of w processed by F (ws);
- $d_3 = d_2 + (\mathrm{DEQ2}(d_2) - d_2) \cdot a_3$,

with a_1, a_2, a_3 the three amounts (FX deq, FX ws, second FX deq) and i = FX deq->ws. The modulations of the MOD & LA tab can cause the detection's DEQ, WS and DEQ 2 to be computed even when these amounts are at 0 (chapter 23).

Merging the channels: Channels Mix

Channels Mix

INPUT tab

Choices: Avg · 0 · 1 · Min · Max **Default:** Avg **ID:** `sss`

When the channels are **linked** (**Channels Unlink** below 1), the compressor computes a single detection for all channels. This menu chooses how to merge the absolute values of the channels:

- **Avg** (default): the average of the absolute values, $(|c_0| + |c_1|)/2$;
- **Ch0**: channel 0 alone (the left, or the Mid in **L/R to M/S** mode);
- **Ch1**: channel 1 alone (the right, or the Side);
- **Min**: the weaker of the two;
- **Max**: the stronger of the two.

Avg is not the absolute value of the Mid: two channels in opposite polarity give a strong detection in Avg, whereas their Mid is zero. **Max** makes the compressor react to the louder channel, which protects each channel from overshooting. **Min** makes it react only when both channels are loud.

This menu doesn't act on the unlinked part: with Channels Unlink at 1, each channel has its own detector and the menu has no effect; between 0 and 1, it only concerns the linked part of the mix (chapter 7). The same choice is used by the modulations of the MOD & LA tab (chapter 23). In mono, Ch0 and Ch1 refer to the same channel.

In your host's automation lists, Ch0 and Ch1 appear under the names "0" and "1."

To be confirmed by François

In **Min** mode, the code starts from the value 1 before looking for the minimum of the channels: if all the channels exceed 1 (0 dBFS in the Headroom domain, which happens with a positive Headroom or a very hot signal), the detection stays stuck at 1. Is this intended?

Listening to the detection: Monitor Input

The **Monitor Input** monitor (INPUT tab, card in chapter 7) replaces the output with the detection signal, channel by channel and with its sign: you hear the internal signal and the sidechain mixed, filtered, with the feedbacks, the Mid/Side and the DEQ, WS and DEQ 2 sends. The height added by the bias is taken out of it. The **Channels Mix**, which comes afterward, is not audible.

This is the tool for setting up the detection filters and Mid/Side: you directly hear what will trigger the compression.

The target branch (OUTPUT tab)

Target and remainder

The **target** is the part of the signal that receives the compressor's gain; the **remainder** is everything that isn't the target. The remainder bypasses the compressor and is added back at the end:

output = compressed target + remainder.

Only the target goes through the gain reduction, the opto, the bias, **Comp Clip** and **Comp Db**. The remainder passes through untouched. When the compressor isn't reducing the gain (and the target's DEQ and WS are neutral), the sum rebuilds the input signal: exactly (apart from the delay) with the FIR filters, and with a phase rotation with the Linkwitz-Riley filters (see below).

Comprendre is therefore not a multiband compressor: it compresses **one** target, which can be one band, two bands or part of the stereo image, and leaves the remainder untouched. To process several bands differently, you need several instances of the plugin.

> **Comp Db and dW | Dw**
>
> **Comp Db** (chapter 7) applies only to the compressed target: it is the way to balance the compressed part against the remainder. The **dW | Dw** mix also uses the notion of remainder; at its −1 position, the remainder disappears and you hear only the compressed target.

Target filters

The filters of the OUTPUT tab work exactly like those of the detection (see the table and the figure of the configurations, above): what they keep is the target, and their complement is the remainder.

FX High Cut

OUTPUT tab

Range: 0 to 20000 Hz **Default:** 0 Hz **ID:** `ciseau__bas`

Cutoff frequency of the target **high-pass** filter: only the part of the signal above this frequency is compressed; what lies below joins the remainder and passes through uncompressed. At 0 Hz (default), this filter is disabled.

Combined with **FX Low Cut** (OUTPUT tab), it forms one band or two bands, depending on whether its frequency is lower or higher.

Example: with FX High Cut at 3 kHz, only the highs are compressed. Paired with a detection on the highs, the compressor becomes a kind of de-esser or dynamic EQ for the top end.

With the Linkwitz-Riley filters, the output has its phase rotated around the cutoff frequency, even without compression.

FX Low Cut

OUTPUT tab

Range: 0 to 20000 Hz **Default:** 20000 Hz **ID:** `ciseau__top`

Cutoff frequency of the target **low-pass** filter: only the part of the signal below this frequency is compressed; what lies above joins the remainder. At 20,000 Hz (default), this filter is disabled.

Example: with FX Low Cut at 150 Hz, only the lows are compressed, which lets you hold a bass steady without touching the rest of the spectrum.

Placed **below** FX High Cut, it gives two bands: with FX High Cut at 5 kHz and FX Low Cut at 120 Hz, the lows and the highs are compressed, and the midrange passes through untouched.

> **♪ In the studio: listen to the lows, process the highs**
>
> Crossing the two branches gives you tools an ordinary compressor can't build. On a vocal, an **FX High Cut** around 5 kHz in both tabs makes a de-esser: only the sibilants trigger the compression, and only the highs are attenuated, while the body of the voice stays untouched. On a drum loop or an electronic track, cross the branches: the detection listens only to the lows (**FX Low Cut** in the INPUT tab around 150 Hz) and the target contains only the highs (**FX High Cut** in the OUTPUT tab around 3 kHz). The cymbals and hi-hats duck on every kick hit and come back with the release, a breathing motion you can sculpt with the attack and release. This is the example François picks in his video on input and output selection, and he also lets you hear the limits of the exercise: overcompressed highs dull the sound. Dose the reduction while listening to **Monitor Comp**, then always judge the result in the mix.

Target Mid/Side

FX Mid/Side

OUTPUT tab

Range: 0 to 1 **Default:** 0.5 **ID:** `cotes_`

Chooses which part of the stereo image is compressed. The formula is the same as for the detection FX Mid/Side, but the part set aside is not lost: it joins the **remainder** and passes through uncompressed.

- **0**: only the Mid is compressed; the Side passes through untouched.
- **0.5** (default): neutral; the whole image is compressed.
- **1**: only the Side is compressed; the Mid passes through untouched.
- Intermediate values: for example, at 0.75, half of the Mid is in the target and the other half in the remainder, while the Side is entirely in the target.

It is applied **after** the target filters: combined with FX High Cut, it lets you compress, for example, only the Side of the highs.

In mono, the Side is zero: at 1, the target is empty and no compression is audible. With **L/R to M/S** engaged, the same remark as for the detection applies; to compress the Mid and the Side separately, use **L/R to M/S** with **Channels Unlink** instead (chapter 7).

The target's DEQ, WS and DEQ 2 amounts

On the target, the DEQ, the WS and the DEQ 2 are applied **by default**: their four amounts are at 1. They are what you hear when you set up these modules. They act on the target **before** the multiplication by the compressor's gain, and they use the envelope of the current sample.

Unlike in the detection, these amounts have no negative values. Their knob follows the law $a = p^2$ (except deq->ws, which is linear): the center position corresponds to 0.25.

deq

OUTPUT tab

Range: 0 to 1 **Default:** 1 **ID:** `retour_cible_deq`

The first **deq**: doses the effect of the **DEQ** on the target. If c_0 is the target entering the DEQ, it becomes $c_0 + (\mathrm{DEQ}(c_0) - c_0) \cdot a$.

- **1** (default): the DEQ is fully applied.
- **0**: the DEQ is not heard on the target; it can, however, keep feeding the WS (deq->ws control).
- Intermediate values: proportional amount, like a dry/wet mix specific to the DEQ.

deq->ws

OUTPUT tab

Range: 0 to 1 **Default:** 1 **ID:** `odeqfet`

Chooses what **enters** the target waveshaper:

- **1** (default): the target after the full DEQ, whatever the value of the first deq;
- **0**: the target before the DEQ;
- in between: a linear mix.

With the first **deq** at 0 and **deq->ws** at 1, you feed the DEQ into the saturation without hearing the DEQ itself: the WS reacts to the dynamic equalization, but that equalization is not applied to the sound. This control and its counterpart in the INPUT tab are independent. Without an active DEQ, it has no effect.

ws OUTPUT tab

Range: 0 to 1 **Default:** 1 **ID:** `retour_cible_fet`

Doses the waveshaper's **saturation product** added to the target (same principle as FX ws in the detection: the difference between the WS output and input, processed by F (ws), then added).

- **1** (default): the product is added in full: you hear the WS.
- **0**: the WS is not heard on the target.
- Intermediate values: proportional amount.

It is the natural switch for making the WS work in the detection only: ws at 0 here, FX ws other than 0 in the INPUT tab. The **Monitor Fet** monitor (chapter 17) lets you hear this product alone, after this amount.

deq OUTPUT tab

Range: 0 to 1 **Default:** 1 **ID:** `retour_cible_dep`

The second **deq**: doses the effect of the **DEQ 2** on the target, after the DEQ and the WS. 1 (default) = fully applied, 0 = not heard, intermediate values = proportional amount.

Since the WS is nonlinear, its position before or after the compressor's gain changes the result: in Comprendre, the compressor attenuates the already-saturated target.

To be confirmed by François

The code applies the target's DEQ, WS and DEQ 2 **before** multiplying the target by the compressor's gain. In video 24, you said you thought they were "normally after." Is the order in the code the one you want?

Listening to the target: Monitor Output and Monitor Comp

Two monitors in the OUTPUT tab (cards in chapter 7) let you hear the target without the remainder:

- **Monitor Output**: the **uncompressed** target, after the target's filters, Mid/Side, DEQ, WS and DEQ 2;
- **Monitor Comp**: the **compressed** target, after Comp Clip and Comp Db. It automatically turns on Monitor Output.

Together with **Monitor Input** and **Monitor Delta** (the difference between output and input, chapter 7), you thus have a listening point at every stage. A handy workflow:

1. turn on **Monitor Output** and set the filters and the Mid/Side of the OUTPUT tab until you hear only the part to be compressed;
2. turn on **Monitor Input** and set those of the INPUT tab until you hear what should trigger the compression;

3. turn on **Monitor Comp** and set the threshold, ratio, attack and release while listening to the compressed part alone;
4. turn the monitors off to hear the result, and use **Monitor Delta** to hear what has been removed.

Linkwitz-Riley or FIR

The band filters of both branches (detection, sidechain and target) are all of the same type, chosen in the MAIN tab with **X Filters Taps**: **Linkwitz-Riley** filters when this control is at 0 (default), **FIR** filters otherwise. This choice only concerns these filters: those of the dynamic EQs, the waveshaper, the opto and the bias don't change.

Linkwitz-Riley filters

A 4th-order Linkwitz-Riley filter splits the signal into a low output and a high output, with 24 dB per octave slopes that cross at −6 dB at the cutoff frequency. Their remarkable property: **the sum of the two outputs has a perfectly flat amplitude**. Splitting the signal into target and remainder and then adding them back together therefore doesn't change the spectrum.

This sum is not the original signal, however: it is an **all-pass**, whose phase rotates through a full turn (360°) around the cutoff frequency. With two frequencies (one band or two bands), the rotation is doubled.

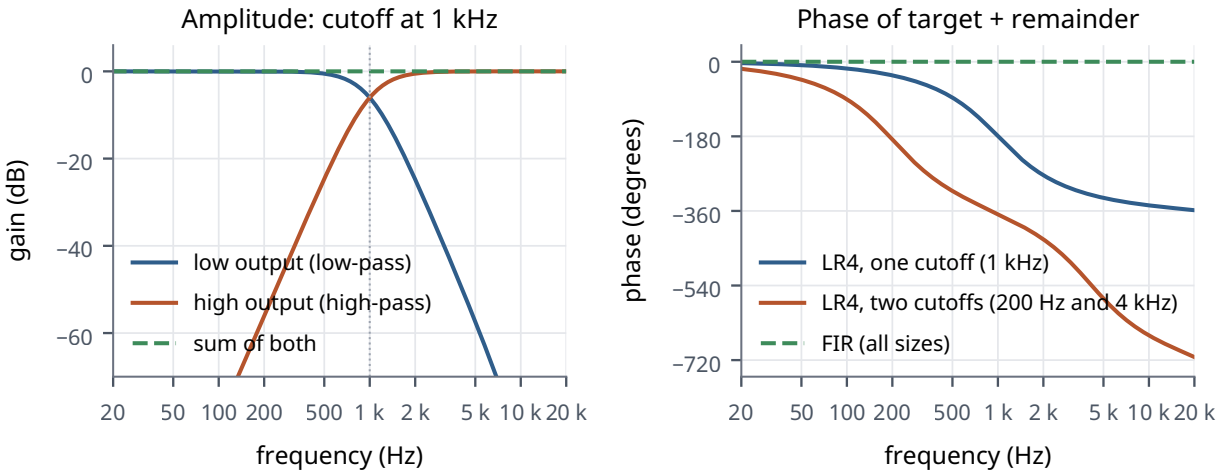


Figure 67 — Comprendre's Linkwitz-Riley filters. Left: the two outputs of a 1 kHz filter and their flat sum. Right: the phase of the target + remainder sum: it rotates with the Linkwitz-Riley filters and stays at zero with the FIR filters.

The consequences depend on the branch:

- **In the detection**, the phase rotation is never heard: only the level of the filtered signal matters.
- **In the target**, the output is the phase-shifted version of the input, even when the compressor isn't reducing the gain. On its own, it sounds like the original, but it no longer cancels against it: **Monitor Delta** isn't silent, and the **Dry/Wet**, or a parallel mix in your host, can create interference (chapter 7).

The Linkwitz-Riley filters add no latency and cost almost nothing in CPU.

A word from the author

For François, the strength of Linkwitz-Riley filters is their perfect frequency response once the bands are summed, and their weakness is phase, which is considered troublesome in mastering. That is why the FIR filters exist.

Going further

A 4th-order Linkwitz-Riley is two 2nd-order Butterworth filters of the same frequency in series. With u = f / fc, the amplitudes are $1/(1 + u^4)$ for the low output and $u^4/(1 + u^4)$ for the high output. Their sum is a 2nd-order all-pass. With two frequencies, Comprendre runs the low band through the all-pass of the second frequency, so that the three bands stay in phase: the sum is then the product of the two all-passes. These are JUCE's filters, in a TPT structure, recomputed at the internal rate (over-sampling included).

FIR filters

When **X Filters Taps** is other than 0, the band filters become **FIR** (finite impulse response) filters. Each one is a low-pass designed with the **window method**: an ideal impulse response, truncated to a finite number of coefficients and softened by a window. The other shapes are obtained by subtraction: the high-pass is the signal minus the low-pass, the band is the difference between two low-passes, and the remainder is the signal minus the target. The main signal is delayed by the filter's latency, to align these subtractions.

The first advantage comes from this construction: the target and the remainder recombine exactly into the original signal, simply delayed. There is no more phase rotation (previous figure, right).

The price to pay is threefold:

- **latency**: (taps − 1)/2 samples, reported to the host;
- **selectivity**: for a given number of taps, the width of the transition is fixed in hertz. A low cutoff requires a lot of taps;
- **CPU load**: the cost is proportional to the number of taps, for each channel and each engaged filter.

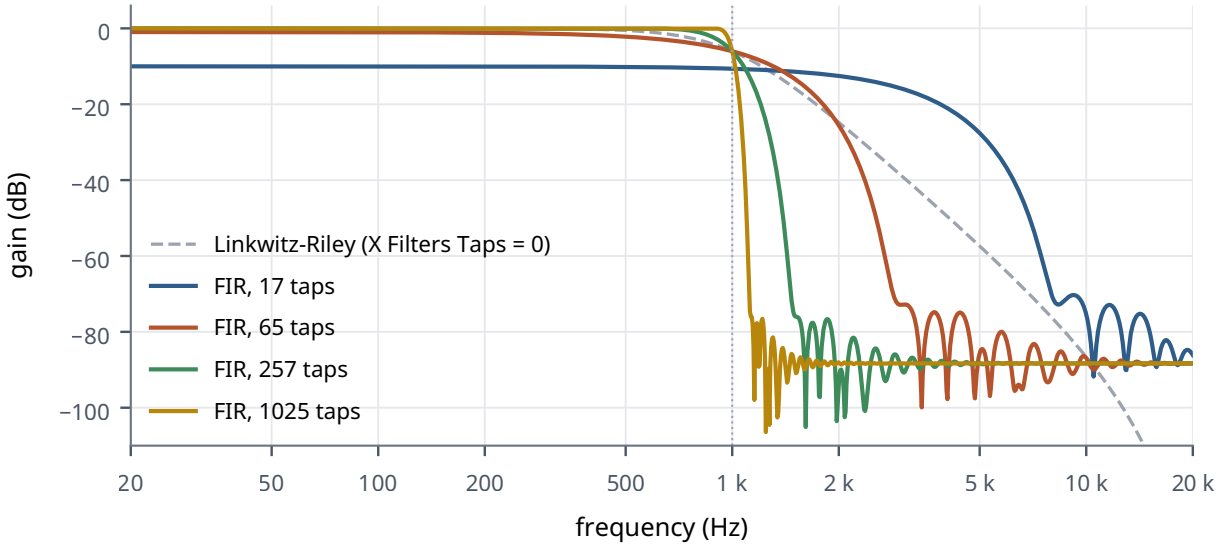


Figure 68 — FIR low-pass at 1 kHz (FX Low Cut alone), blackman window, at 44.1 kHz, for four values of X Filters Taps; dotted gray, the Linkwitz-Riley. With 17 taps, the transition is so wide that the filter doesn't even reach 0 dB in the low end.

X Filters …

MAIN tab

Choices: 0 · 17 · 33 · 49 · 65 · 97 · 129 · 193 · 257 · 385 · 513 · 769 · 1025 **Default:** 0 **ID:** `fir_taps`

Chooses the type of the band filters for the detection, the sidechain and the target:

- **0** (default): Linkwitz-Riley filters;
- **17 to 1025**: FIR filters of that size.

The larger the value, the narrower the transition. For a 5 kHz low-pass at 44.1 kHz with the blackman window, the transition from −1 dB to −40 dB spans 8.5 kHz with 17 taps, 2.2 kHz with 65 taps, 565 Hz with 257 taps and 141 Hz with 1025 taps. As a rule of thumb, this width is 3.3 · fs / (taps + 1) with the blackman window (fs being the internal rate), about 4 · fs / (taps + 1) with blackmanHarris and 2.5 · fs / (taps + 1) with hann or hamming. A precise cutoff around 100 or 200 Hz therefore requires more than the 1025 taps available at 44.1 kHz.

The **latency** is (taps − 1)/2 samples: at 44.1 kHz, 8 samples (0.18 ms) for 17 taps, 32 (0.73 ms) for 65, 128 (2.9 ms) for 257 and 512 (11.6 ms) for 1025. It is reported to your host, and it applies **as soon as this control is other than 0**, even if no filter is engaged: leave it at 0 if you don't use the band filters.

The number of taps is **not adjusted for oversampling**. The filters run at the internal rate: with ×F oversampling, you have to multiply the value by F to keep the same selectivity (65 without oversampling corresponds to 129 at ×2). The maximum value, 1025, therefore limits the achievable selectivity: at ×16, it is equivalent to about 64 taps without oversampling. The reported latency is then rounded up to the next sample, at the base rate.

Every change of frequency, size, window or beta recomputes the filters concerned: during the computation, the signal passes through unfiltered, then the filter's memory is cleared. In FIR mode, avoid automating these controls during playback.

The **In->Out** auto-volume is not compatible with the FIR filters (chapter 8).

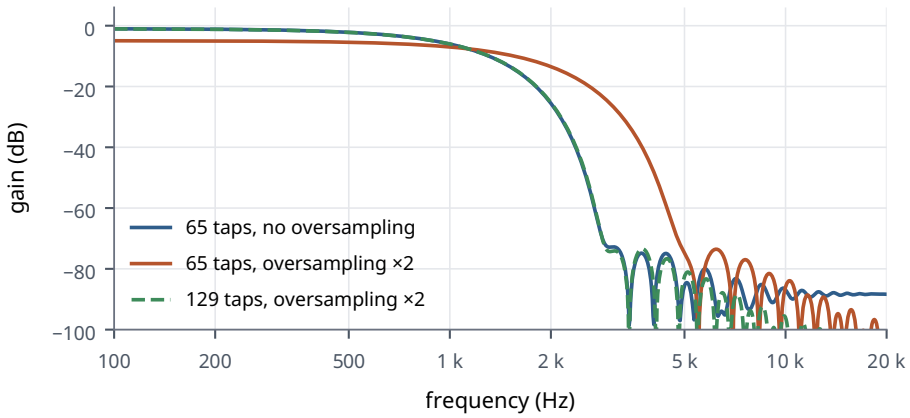


Figure 69 — Effect of oversampling on a 1 kHz FIR low-pass (blackman window). With the same number of taps, ×2 oversampling widens the transition; doubling the number of taps restores the curve exactly.

A word from the author

François sums it up this way: you need a large number of taps for the filter to be effective, but the more taps there are, the greater the latency; and when you oversample, you have to multiply the number of taps by the oversampling factor to get the same filtering. He acknowledges that a small phase effect remains, and considers this compromise the least bad one he has found.

To be confirmed by François

JUCE interprets the value of **X Filters Taps** as an order: the filter has taps + 1 coefficients and a delay of taps/2 samples, whereas the plugin compensates for (taps − 1)/2. According to the calculation, this half-sample offset limits the attenuation of the filters obtained by subtraction (with FX High Cut, at 44.1 kHz: about −43 dB at 100 Hz, and only −17 dB at 2 kHz for a 5 kHz cutoff), and the center coefficient, placed next to the actual center, creates a floor that rises with the cutoff frequency (about −46 dB for a 5 kHz low-pass). Should taps − 1 be passed to JUCE?

To be confirmed by François

In FIR mode, a filter is only recomputed when "its" frequency changes (or the taps, the window, the beta, the oversampling). According to the code, setting FX Low Cut **before** FX High Cut leaves the second filter of the band uncomputed, and setting FX High Cut back to 0 while FX Low Cut is active leaves the filter on the old FX High Cut frequency; touching the other frequency afterward, or X Filters Taps, fixes the situation. Can you check this by ear?

Windows

The impulse response of an ideal low-pass is infinite. The FIR filter keeps only a finite number of its points; cutting it off abruptly at the ends creates ripples and limits the attenuation outside the band. The **window** is a curve that softens these ends. The choice is a compromise:

- a window that softens little gives a **steep** transition, but **weak attenuation** outside the band;
- a window that softens a lot gives **strong attenuation**, but a **wider** transition.

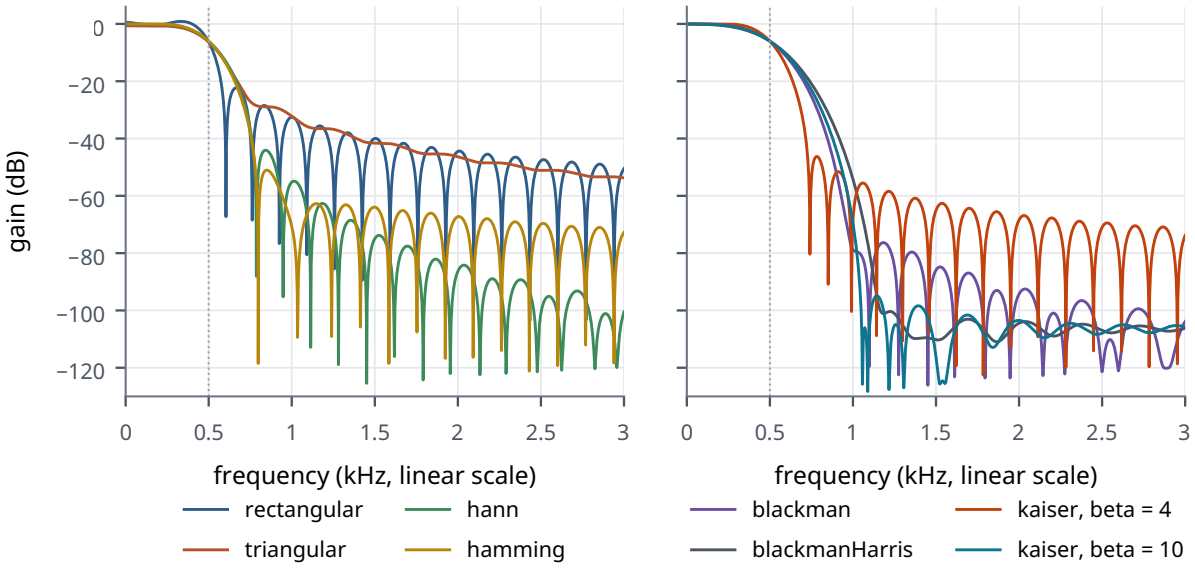


Figure 70 — FIR low-pass at 500 Hz, 257 taps, at 44.1 kHz: effect of the window, on a linear frequency scale.

In the example in the figure, the attenuation outside the band reaches about 20 to 40 dB with the rectangular window, 45 to 70 dB with hann, 50 to 65 dB with hamming, 75 dB with blackman and 100 dB with blackmanHarris; the transition widens in the same order. The triangular window combines a wide transition with weak attenuation.

X Filters Window

MAIN tab

Choices: blackman · blackmanHarris · flatTop · hamming · hann · kaiser · rectangular · triangular **Default:** blackman
ID: `fir_fenetre`

Chooses the window of the FIR filters: **blackman** (default), **blackmanHarris**, **flatTop**, **hamming**, **hann**, **kaiser**, **rectangular** or **triangular**. This menu, located under X Filters Taps in the MAIN tab, has no displayed name. It has no effect as long as X Filters Taps is 0.

- **rectangular**: no softening; the steepest transition, the weakest attenuation, and a slight ripple in the passband.
- **hann** and **hamming**: middle-of-the-road compromises.
- **blackman**: good attenuation, a slightly wider transition; a good default choice.
- **blackmanHarris**: the strongest attenuation, the widest transition.
- **kaiser**: adjustable with the **beta** parameter, from the equivalent of rectangular (beta = 0) to the equivalent of blackmanHarris (beta = 12).
- **triangular**: of little interest here.
- **flatTop**: see the box below.

beta

MAIN tab

Range: 0 to 12 **Default:** 0 **ID:** `fir_beta`

Parameter of the **kaiser** window, from 0 to 12; the other windows ignore it. At 0 (default), the kaiser window is exactly the rectangular window. As beta goes up, the attenuation outside the band increases and the transition widens: in the example in the figure, about 50 dB of attenuation for a beta of 4, 80 dB for 8 and 100 dB for 12, a value that matches blackmanHarris.

A word from the author

François's favorite window is blackmanHarris: it resembles a kaiser at maximum beta, whereas a kaiser with a low beta resembles the rectangular window.

To be confirmed by François

With the **flatTop** window, JUCE's window is not normalized (its center value is 4.6): according to the calculation, the low-pass has a gain of up to +13 dB in its passband, and the filters obtained by subtraction (FX High Cut, bands) boost instead of cutting. Should this window be removed from the menu?

To be confirmed by François

The code truncates **beta** to an integer: only the values 0, 1, 2… 12 have an effect, whereas the knob moves in steps of 0.01. Is this intended?

Going further

For a cutoff frequency fc, an internal rate fs and a value N of X Filters Taps, JUCE computes N + 1 coefficients $h[i] = w[i] \cdot \sin(2\pi \cdot fn \cdot (i - N/2)) / (\pi \cdot (i - N/2))$, with fn = fc / fs, except for the coefficient of index i = (N − 1)/2, which receives 2 · fn (see the first "To be confirmed" box in the FIR section); the resulting low-pass passes through −6 dB at fc. The windows w have N + 1 points; with $c_k = \cos(k \cdot \pi \cdot i / N)$:

- rectangular = 1; triangular = $1 - |i - N/2| / (N/2)$;
- hann = $0.5 - 0.5\ c_2$; hamming = $0.54 - 0.46\ c_2$;
- blackman = $0.42 - 0.5\ c_2 + 0.08\ c_4$;
- blackmanHarris = $0.35875 - 0.48829\ c_2 + 0.14128\ c_4 - 0.01168\ c_6$;
- kaiser = $I_0(\beta \cdot \sqrt{1 - ((i - N/2)/(N/2))^2}) / I_0(\beta)$, I_0 being the modified Bessel function of order 0.

With L(f) the low-pass at f and z the delayed signal: FX High Cut alone gives $z - L(f_1)$; FX Low Cut alone, $L(f_2)$; one band, $L(f_2) - L(f_1)$; two bands, $z - L(f_1) + L(f_2)$. The remainder is always z − target.

> **♪ In the studio: for mastering**
>
> On a master, two controls from this chapter really come into their own. The **FX Mid/Side** of the OUTPUT tab compresses only the sides (at 1) or only the center (at 0): a way to tame overly wide cymbals or reverb without touching the vocal and the bass, which live in the center, or conversely to hold the center while letting the image breathe. The trick isn't new: chapter 7 traces it back to the Lateral/Vertical mode of the Fairchild 670. As soon as the target is filtered, switch to the FIR filters (**X Filters Taps** at 257 or more, **blackmanHarris** window, François's favorite): the target and the remainder recombine without phase rotation. François designed them for those who, in mastering especially, dislike the phase shift of conventional filters; their latency of a few milliseconds doesn't matter in mastering, but it gets in the way if you play live through the plugin. Finally, François sees the dynamic EQ placed in the detector, combined with the frequency-driven release modulations (chapter 23), as a way to achieve an overall coherence that is almost a mastering matter.

Example settings

The following examples start from the default settings. The frequencies are only a guide.

Keep the low end from causing pumping. INPUT tab, **FX High Cut** around 100 to 150 Hz. The target remains the whole signal: the entire mix is compressed, but the kick and the bass no longer trigger the compression on their own. Since only the detection is filtered, no phase rotation is audible.

A de-esser. INPUT tab, **FX High Cut** around 5 kHz: the compressor reacts only to the sibilants. OUTPUT tab, **FX High Cut** at the same frequency: only the highs are attenuated when the sibilants exceed the threshold. Short attack and release times work well (chapter 12). With the whole target (no filter in OUTPUT), you instead get a "wideband" de-esser, which turns down the whole signal on every sibilant.

Compress the highs based on the lows. INPUT tab, **FX Low Cut** around 150 Hz; OUTPUT tab, **FX High Cut** around 3 kHz. The highs (cymbals, for example) are turned down on every kick hit. Set both filters while listening to **Monitor Input**, then **Monitor Output**.

Make the bass give way to the kick. Send the kick to the sidechain input of Comprendre, placed on the bass. MAIN tab, **Sidechain** at 1; INPUT tab, **FX Low Cut** around 120 Hz (this filter also applies to the sidechain); OUTPUT tab, **FX Low Cut** around 200 Hz. Only the low end of the bass is turned down when the kick hits; its harmonics stay untouched.

Hold the sides of the image. OUTPUT tab, **FX Mid/Side** at 1: only the Side is compressed, and the Mid passes through untouched. INPUT tab, **FX Mid/Side** at 1 so that the detection measures the Side as well. **Comp Db** then lets you bring up the compressed part alone, and thus widen the image in a controlled way. Add **FX High Cut** in the OUTPUT tab, and you process only the Side of the highs.

Compress the extremes of the spectrum. OUTPUT tab, **FX High Cut** at 5 kHz and **FX Low Cut** at 120 Hz: the lows and the highs form the target, and the midrange passes through untouched.

A feedback compressor. INPUT tab, **FF->FB** at 1. With the same settings, the compression is gentler; lower the threshold or raise the ratio to get the same amount of reduction back. If the target is filtered, **FB cmp->all** chooses whether the detector measures only the compressed band or the whole output.

Filter the target without touching the phase. MAIN tab, **X Filters Taps** at 257 or more, **blackman** or **blackmanHarris** window; multiply the value by the oversampling factor if oversampling is active. The target and the remainder recombine exactly, at the cost of a few milliseconds of latency.

Summary

Control	Tab	Neutral value	Role
FF->FB	INPUT	0	feed-forward (0) or feedback (1)
FB cmp->all	INPUT	0	return taken from the compressed target (0) or the whole output (1)
FX High Cut	INPUT	0 Hz	detection high-pass
FX Low Cut	INPUT	20,000 Hz	detection low-pass
FX fb cmp	INPUT	0	adds the compressed − uncompressed difference
FX Mid/Side	INPUT	0.5	Mid (0) or Side (1) of the detection
FX deq, FX deq->ws, FX ws, FX deq	INPUT	0	DEQ, WS and DEQ 2 sends to the detection
Channels Mix	INPUT	Avg	merging of the linked channels
Sidechain	MAIN	0	share of the sidechain in the detection
FX High Cut	OUTPUT	0 Hz	target high-pass
FX Low Cut	OUTPUT	20,000 Hz	target low-pass
FX Mid/Side	OUTPUT	0.5	Mid (0) or Side (1) of the target
deq, deq->ws, ws, deq	OUTPUT	1	DEQ, WS and DEQ 2 amounts on the target
X Filters Taps	MAIN	0	Linkwitz-Riley (0) or FIR of N taps
window (unnamed)	MAIN	blackman	FIR filter window
beta	MAIN	0	kaiser window parameter

CHAPTER 15

Gain Reduction and the Opto Designer

Once the envelope follower has smoothed the target gain set by the transfer curve (chapters 10 to 12), Comprendre can still reshape it before applying it to the signal. That is the job of the two tabs described in this chapter:

- the **GR (cv)** tab bends the envelope with powers and an offset, then hosts the **opto designer**, which mimics the behavior of optical compressors: reaction times that depend on the amount of reduction;
- the **F (gr)** tab processes the opto's "saturation product," that is, the difference the opto introduces into the signal: you can delay it, filter it, dose it and limit it.

The (cv) suffix stands for *curve*, a reminder that these settings act after the curve; the (gr) suffix stands for *gain reduction*, since the product comes from the gain reduction.

These two tabs touch on what gives a compressor its personality. First, the shape of the envelope: its resting point, its depth, its curvature. The **Env Offset** setting, which shifts the whole envelope, is the one François uses most to imitate other compressors: 21 of the 80 presets use it. Then the **opto designer**, which puts within reach one of the most sought-after feels in the history of compression: the breathing of an optical cell, which follows the music a beat behind, lets go quickly and then lingers, and gives vocals, bass and piano the suppleness that made the LA-2A's reputation. On hardware, this behavior is frozen into the physics of a cell. Here, it becomes a set of controls you can dose, speed up, slow down, invert, fold into the envelope or treat as a separate color. Mixers will find compression you forget is there; imitators, tools to recover the exact shape of a gain curve; explorers, what François calls "negative opto."



Figure 71 — The GR (cv) tab: envelope shaping and the opto designer.



Figure 72 — The F (gr) tab: processing the opto's saturation product.

Where these settings act

Here is the exact order of operations, from the transfer curve to the application of the gain:

1. the transfer curve supplies the target gain (purple curve);
2. **Env Pow Unpow** raises this gain to a power p;
3. the envelope followers smooth it (**ENV (cv)** and **A/R (cv)** tabs);
4. **Env Pow Unpow** applies the inverse power 1/p;
5. **Gr|Env Pow** applies a second power, to the envelope or to the reduction;
6. **Env Offset** adds an offset;
7. the envelope is clamped at 0 from below, but **not** from above: it can exceed 1, that is, become a gain;
8. the **opto designer** computes its own gain, if it is active (**Opto** other than 0);
9. depending on **Opto Env**, this optical gain is either folded into the envelope or kept separate;
10. the envelope multiplies the target, through **Gr Amp** (chapter 10);
11. if the opto is kept separate, the difference it introduces (the saturation product) is processed by the **F (gr)** tab, then added to the signal.

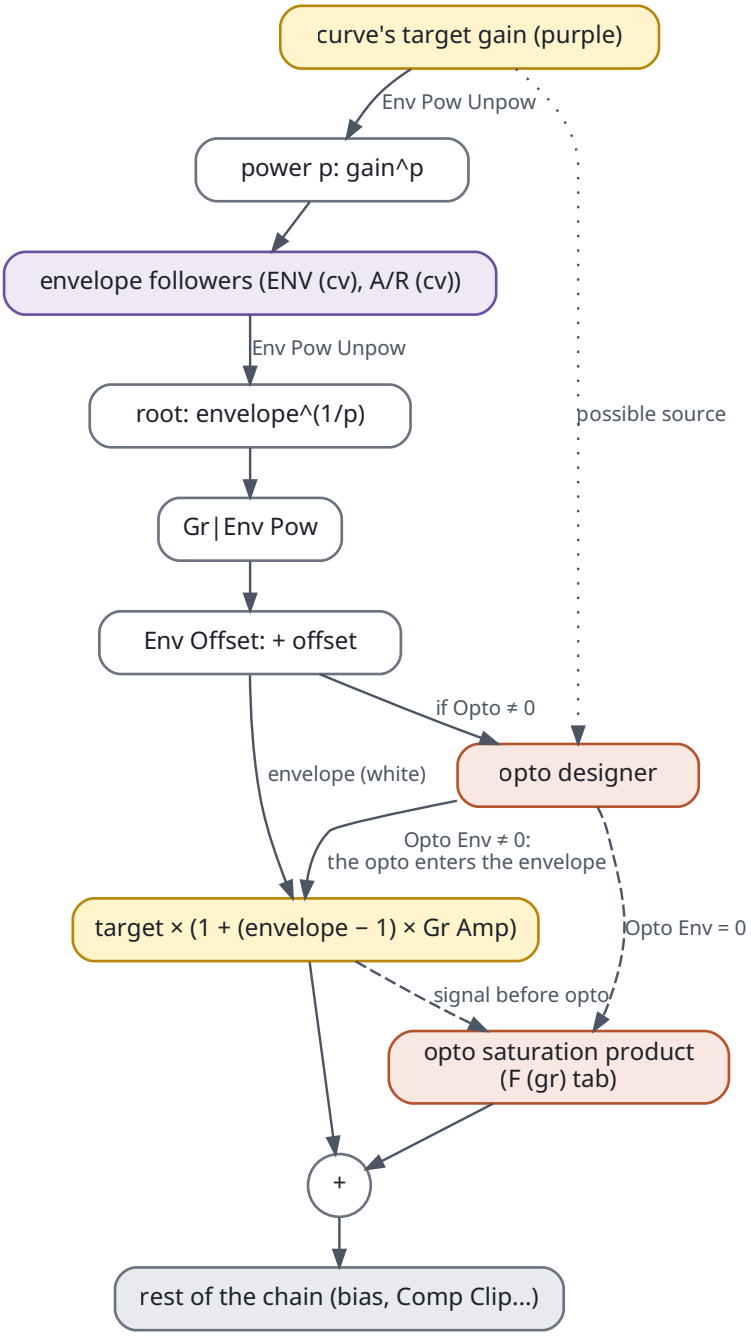


Figure 73 — The gain reduction chain, from the curve to the signal.

> **These settings turn on the follower**
>
> The envelope follower and its post-processing are computed only when they serve a purpose: a non-zero attack or release, or one of the **Env Pow Unpow**, **Gr | Env Pow** or **Env Offset** settings away from its neutral value. When they are not computed, the envelope simply equals the target gain from the curve.

Shaping the envelope

The first three settings in the tab modify the envelope itself. Their roles are quite distinct:

- **Env Pow Unpow** does not change the reduction reached on a steady signal, only the way the envelope gets there: it is a **dynamics** setting;
- **Gr|Env Pow** changes the amount of reduction, at every level: it is a **curve** setting, applied after smoothing;
- **Env Offset** shifts the whole envelope, all the time.

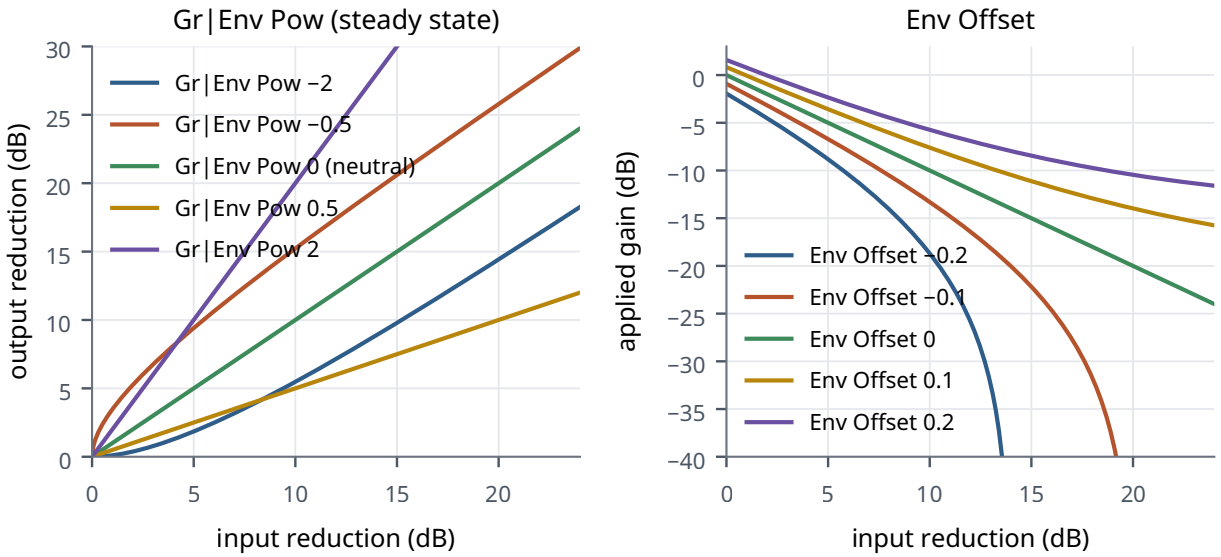


Figure 74 — Left, the effect of Gr|Env Pow on steady-state reduction. Right, the effect of Env Offset: the offset is additive in amplitude, so its effect in dB grows with the reduction.

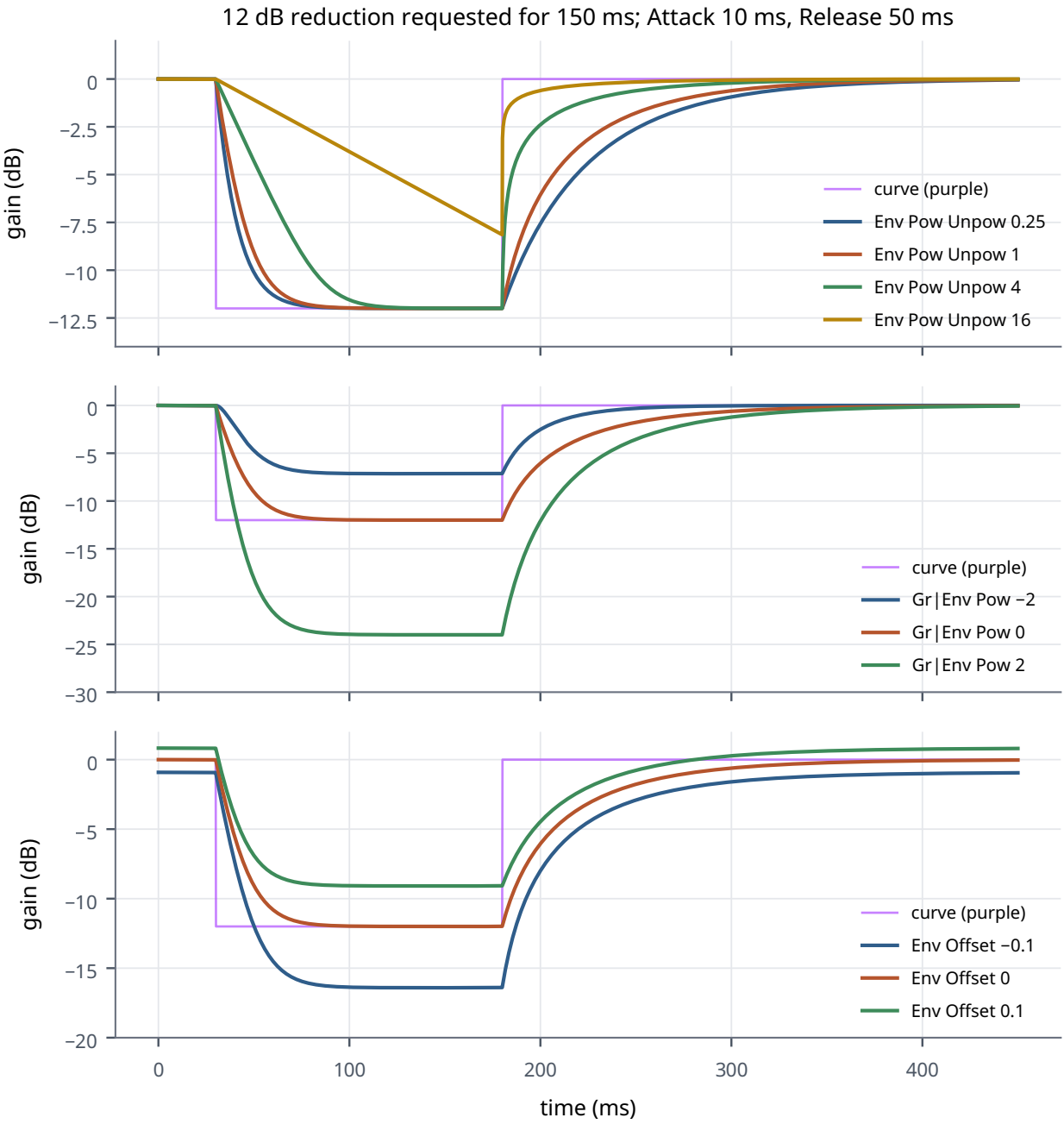


Figure 75 — The three settings on the same reduction request (12 dB for 150 ms), with the default follower (linear, Attack 10 ms, Release 50 ms).

Env Pow Unpow

GR (cv) tab

Range: 0.000001 to 64 **Default:** 1 **ID:** `env_puissance`

Raises the target gain to the power p **before** the envelope followers, then applies the inverse power 1/p to their output. Raising a gain to the power p amounts to multiplying its reduction in dB by p: with p = 2, a 6 dB request becomes 12 dB for the followers, which smooth it, then the root brings the result back to the original scale.

On a steady signal, the two operations cancel out: the final reduction is exactly the same. Only the **dynamics** change, because the followers work on an expanded (p > 1) or contracted (p < 1) reduction. With the default linear follower:

- **p > 1** slows down the attack and speeds up the release, all the more as p gets larger. For large values, the descent becomes almost a straight line in dB and stops dead on the target, while the release climbs back almost instantly;
- **p < 1** speeds up the attack a little and slows down the release a little. As p tends toward 0, the linear follower behaves exactly like a dB follower;
- **p = 1** (default): no effect.

With the dB followers (**EvMu Lin/Db** at 1, chapter 11), multiplying the reduction in dB before smoothing and then dividing it afterward comes to almost the same thing: the effect is limited to the nonlinear cases (−60 dB floor, attack and release modes).

A word from the author

Some envelope followers use this algorithm (a power at the input, the inverse power at the output) to get different properties. Depending on the value, it can “soften” or “rough up” the envelope.

To be confirmed by François

Env Pow Unpow also raises to the power p the target gain as it is displayed (the purple line in the history and the graph’s curve during operation), and as it is received by the opto designer (“curve” source) and by the bias. Is this intended, or was only the input of the followers meant to be modified?

Gr|Env Pow

GR (cv) tab

Range: −8 to 8 **Default:** 0 **ID:** `env_diode`

Applies a power d to the envelope or to the reduction, depending on the sign of the setting. Unlike **Env Pow Unpow**, it is not compensated afterward: it changes the amount of reduction.

- **Positive d**: the envelope is raised to the power d, which multiplies the reduction in dB by d. With d = 2, 6 dB of reduction becomes 12 dB; with d = 0.5, it becomes 3 dB. On a steady signal and a dB curve, it is as if the reduction slope were multiplied by d: 2 turns a 2:1 into ∞:1.
- **Negative d**: the power |d| is applied to the **linear** reduction, that is, to the gap between the envelope and 1. With |d| > 1, small reductions are strongly softened and large ones less so: for d = −2, 6 dB becomes 2.5 dB, but 20 dB still becomes 14.4 dB. With |d| < 1, it's the opposite: for d = −0.5, 6 dB becomes 10.6 dB. If the envelope exceeds 1 (gain), the power applies in the same way to the excess above 1.
- **0, 1 and −1**: no effect. The default value is 0.

Gr|Env Pow acts after smoothing: it bends an envelope that has already been smoothed, with its attack and release times (figure above).

A word from the author

Applying a power to the envelope or to the gain reduction: François didn't see the point of doing both at once, hence a single setting, whose sign picks the variant.

♪ In the studio: soften or rough up the envelope

These two powers act on character more than on amount. **Env Pow Unpow** does not change the reduction on a sustained sound, only the way it gets there: above 1, with the default linear follower, the attack slows down and the release speeds up, to the point of snapping back almost at once for large values. Try it on drums, listening to the bounce between hits. **Gr|Env Pow** reworks the reduction law itself: a negative value between −1 and 0 mainly emphasizes small reductions, and makes the compression present from its very first decibels. Several presets use it this way: **acustica_sand3** (−0.89), **softube_ds1mk3** (−0.8), **toneproject_unisum** (−0.6888), **tr_670** (−0.6666). On the **Env Pow Unpow** side, **ssl_1_b** uses 1.173, **tc_dyn3000** 1.414, **acqua_titanium** 2, and **acustica_ivory** goes as far as 7. Finally, **overloud_670** pushes **Gr|Env Pow** to 8: the reduction in decibels is multiplied by eight.

Env Offset

GR (cv) tab

Range: −1 to 1 **Default:** 0 **ID:** `env_offset`

Adds a constant value to the envelope, after the powers. The addition is done in amplitude, on an envelope that equals 1 at rest and falls toward 0 as the reduction increases. There is no ceiling: the envelope can exceed 1.

- **At rest** (no reduction), a positive offset becomes a permanent gain: +0.1 gives +0.83 dB; a negative offset becomes a permanent attenuation: −0.1 gives −0.92 dB.
- **During reduction**, since the envelope is smaller, the same offset weighs more in dB: with 6 dB of reduction, +0.1 brings it back to 4.4 dB and −0.1 takes it to 7.9 dB.
- **A negative offset can cancel the envelope**: with −0.1, the target goes silent as soon as the reduction reaches 20 dB (envelope at 0.1). The envelope is indeed clamped at 0.

The offset applies before the opto designer: it also changes the reduction that sets the opto's speed.

A word from the author

The offset is the setting François uses most to imitate compressors: it has "saved his life" many times. On an LA-2A, for example, you can clearly feel an offset of this kind at work.

Compensating for a positive offset

A positive **Env Offset** raises the level even when the compressor is not working. To compare level-matched, compensate with **Makeup** or **Output Db** (chapter 7), or let the **Out->In** auto-volume do it for you (chapter 8).

♪ In the studio: the offset, the go-to tool for imitation

When you imitate a compressor in analysis mode (chapter 24), the gap that remains between the two gain curves is sometimes a simple offset: a reduction slightly too deep or not deep enough, a different resting level. **Env Offset** fixes it in one move, and you can see why François made it his favorite imitation tool. The presets bear this out: 21 out of 80 use it, almost all with a negative value, which lowers the resting level and deepens the reduction all the more as the reduction gets heavier. **slate_fg2a**, an imitation of a plugin inspired by the LA-2A, uses −0.55, the strongest value in the collection, which echoes François's remark about the LA-2A. Next come **rockruepel_comptwo_drums** (−0.5), **softube_germanium** (−0.455), **pulsar_1178** (−0.4), **ddmf_mde** (−0.39), **kush_silka** and **voxengo_polysquasher3** (−0.303), and **massenbug_drc2_groove** (−0.3). Only three presets raise it: **acustica_sand3** and **pa_elysia** (+0.173), and **tc_dyn3000** (+0.05). Just like a positive offset, a negative offset changes the level even at rest: compensate for it before comparing, or your ear will most often prefer the louder version.

The opto designer

How it works

Optical compressors are known for reaction times that depend on what they are compressing. Comprendre's opto designer offers a simple model of this: a **second follower**, which works in decibels and follows the envelope (or the curve directly), with attack and release times that get **shorter as the reduction gets heavier**.

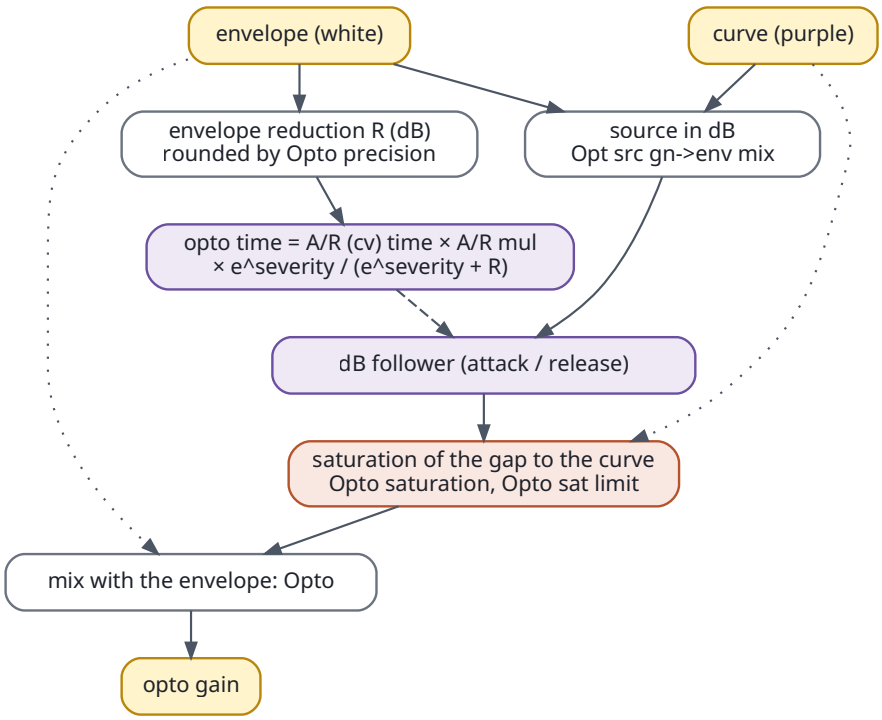


Figure 76 — The opto designer's computation.

At each sample:

1. **R**, the envelope's reduction in dB (after **Env Offset**), is measured and rounded by **Opto precision**;
2. the opto's times are computed from the attack and release times of the main follower:

   opto time = A/R time × Opto A/R mul × e^severity / (e^severity + R)

3. the optical follower, in dB, moves toward its **source** (the envelope, the curve, or a blend of the two, depending on **Opt src gn->env**): with the attack time when it has to go down, with the release time when it has to come back up;
4. if the opto lags behind the reduction requested by the curve, **saturation** can shrink that lag (**Opto saturation**, **Opto sat limit**);
5. the resulting optical gain is mixed with the envelope according to **Opto**.

The “A/R times” are the attack and release times actually used by the main follower: the **Attack** and **Release** knobs, after any automatic settings and modulations (chapters 12 and 23). Their precise meaning (the percentage of the way covered in that time) is the one set by the altitude settings of the **A/R (cv)** tab; by default, 63%.

With no reduction (R = 0), the opto’s time is therefore A/R time × Opto A/R mul. The more the reduction increases, the shorter it gets, and **Opto severity** sets the strength of this effect.

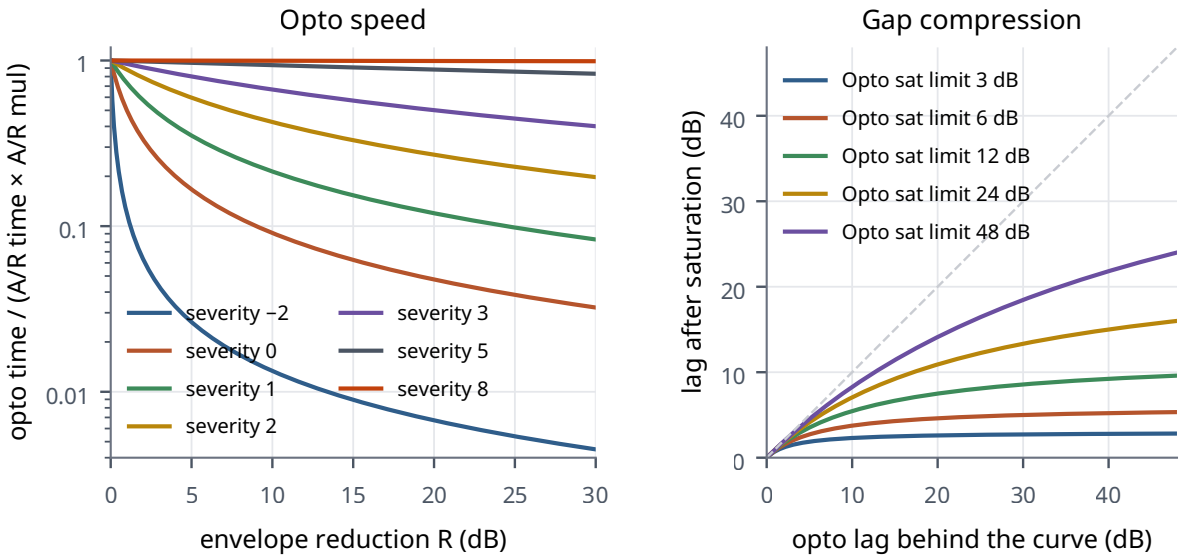


Figure 77 — Left, the opto's time (relative to A/R time × A/R mul) as a function of the envelope's reduction, for different severity values. Right, the compression of the gap by Opto sat limit.

A second smoothing stage

The optical follower sits **after** the main follower and tracks its output: it adds a second smoothing stage, and therefore a lag. But this lag depends on the reduction:

- at the start of an attack, the reduction is still small, so the opto is slow; as soon as the reduction grows, it speeds up and catches up with the envelope;
- during the release, it’s the reverse: at first, the reduction is still heavy and the opto tracks the envelope closely; as the reduction decreases, the opto slows down, and the end of the release stretches out into a long tail.

This two-stage release, fast then slow, is what you can see in the next figure.

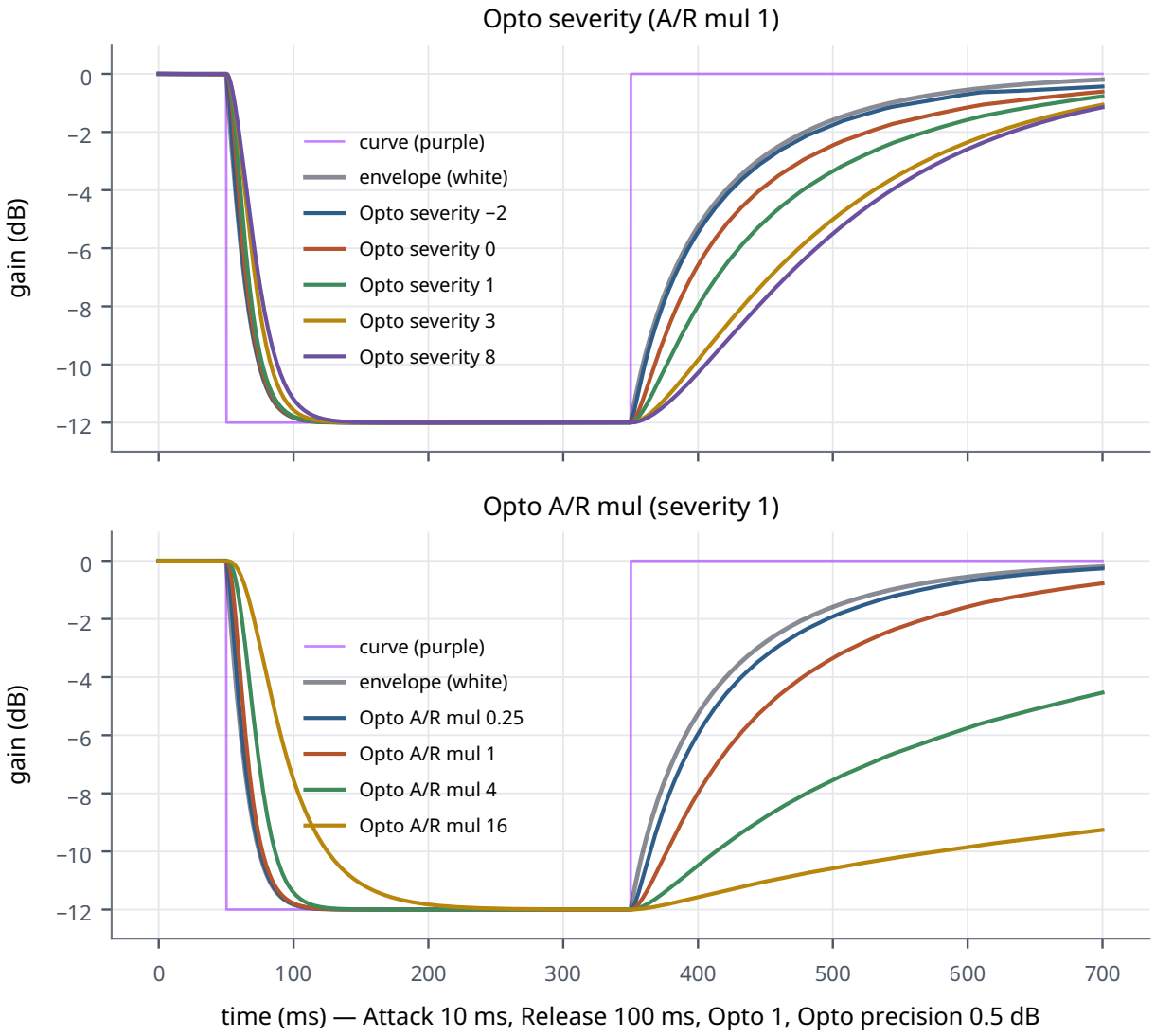


Figure 78 — Response of the opto to a 12 dB reduction request lasting 300 ms (Attack 10 ms, Release 100 ms, Opto 1). Top, by Opto severity; bottom, by Opto A/R mul. The small kinks visible on the slowest curves come from the rounding by Opto precision.

The T4 cell and its two-stage release

In the LA-2A, gain is controlled by the T4 cell: an electroluminescent panel lights up cadmium sulfide photoresistive cells, more brightly as the signal gets louder. This panel actually brought a faster attack than earlier optical compressors, whose neon or incandescent lamps were slow to light up. Chapter 2 covers the essentials: an attack of about 10 ms that depends on frequency; a two-stage release, fast over the first half (40 to 80 ms) and then slow over the rest, up to several seconds; and a memory that slows down this second stage after heavy or prolonged compression. According to Will Shanks of Universal Audio, the T4 in practice has "a time constant of several minutes." This program-dependent release, which other families build with circuitry, comes "for free" with optical designs, from the physics of the cell alone. Other units have borrowed it in their own way: the Distressor's 10:1 ratio is an **Opto** mode that emulates optical-cell compressors, with releases of up to 20 s, and the discrete VCA-type stage of the Shadow Hills has a "Dual" mode that imitates the dual optical release. Comprendre's opto designer does not imitate the physics of the cell: it offers a simple model, centered on that fast-then-lingering release, with times that get shorter as the reduction increases.

Reading the opto in the graph

As soon as **Opto** is not 0, a **sky-blue** line appears in the history (chapter 4). It shows the ratio between the optical gain and the envelope. On a linear scale, it sits at **mid-height** when the opto has no effect; higher up, the opto lets more signal through than the envelope (it is lagging during an attack); lower down, it reduces more (it is lagging during a release). With **opto amp** at 1, its height equals this ratio minus 0.5: a ratio of 1.25 puts it at three quarters of the height. More generally, its distance from mid-height is proportional to 1 + opto amp.

The **white** envelope and the green dots on the transfer curve show the opto's effect only if **Opto Env** is not 0.

The designer's settings

Opto

GR (cv) tab

Range: −1 to 1 **Default:** 0 **ID:** `opto`

Turns on the opto designer and sets its blend with the envelope. At **0** (default), the opto is not computed.

The opto produces an optical gain O; let E be the envelope. The setting determines the opto's share in the applied gain: E + (O − E) × Opto.

- **1**: the envelope is entirely replaced by the optical gain;
- **between 0 and 1**: a blend, in amplitude;
- **negative**: the difference is flipped. At −1, the gain equals E − (O − E): where the opto lags (less reduction than the envelope), you get more reduction instead, and vice versa.

This gain is applied as a multiplier on the already compressed signal (after **Gr Amp**). If the envelope is 0, the opto no longer has any effect.

Since the opto's times are proportional to those of the main follower, with the **Attack** and **Release** times at zero the opto follows its source instantly: if that source is the envelope, the opto then has no effect at all.

> **A word from the author**
>
> You can do "negative opto." Above all, François recommends trying things out: the designer is experimental, and listening and watching the blue line tells you more than reasoning about the formulas.

Opt src gn->env

GR (cv) tab

Range: 0 to 1 **Default:** 1 **ID:** `opto_source`

Chooses what the optical follower tracks:

- **1** (default): the **envelope**, that is, the output of the main follower (white line). The opto then adds a second smoothing stage;
- **0**: the **target gain from the curve** directly (purple line), bypassing the main follower. With **Opto** at 1, the opto alone then smooths the curve, with its variable times: the result is more aggressive;
- **in between**: a blend of the two gains.

Whatever the source, the opto's speed is always computed from the reduction of the **envelope**.

Opto A/R mul

GR (cv) tab

Range: 0 to 421 **Default:** 1 **ID:** `opto_ar`

Multiplies both of the opto's times (attack and release). At 1 (default), with no reduction, the opto has the same times as the main follower. Above that, it is slower and its lag behind the envelope grows; at 16, the end of the release stretches over several hundred milliseconds with the settings used in the figure.

At **0**, the times are zero: the opto follows its source instantly. If the source is the envelope, the opto no longer has any effect; if it is the curve, the envelope is replaced by the raw target gain, with no smoothing at all.

Opto severity

GR (cv) tab

Range: −8 to 8 **Default:** 1 **ID:** `opto_severite`

Sets how much the reduction speeds up the opto. The opto's time is multiplied by e^severity / (e^severity + R), where R is the envelope's reduction in dB.

- **Small values** (negative): the slightest reduction makes the opto very fast. At −2, it keeps its base time only when there is no reduction at all; from 3 dB on, its time is divided by more than twenty, and it sticks to the envelope.
- **Default value, 1**: with a base time of 10 ms, the opto takes 10 ms with no reduction, 4.8 ms at 3 dB, 3.1 ms at 6 dB, 2.1 ms at 10 dB, and 1.2 ms at 20 dB.
- **Large values**: the reduction has almost no influence anymore. At 3, the time only drops by about half at 20 dB; at 8, it stays practically constant. The opto then becomes a simple second follower, which adds a constant lag to the envelope.

The higher the severity, the slower the opto and the more pronounced its effect.

> **A word from the author**
>
> According to François, severity is the designer's most important setting: the higher you set it, the more "violent" the effect.

> **♪ In the studio: the breathing of an optical compressor**
>
> On a vocal, a bass or a piano, the designer's two-stage release often gives the impression of a compressor that follows the performer: after a loud phrase, the gain comes back up quickly, then the end of the recovery stretches out and melts into the music. To get close to it, start with a few decibels of compression, with **Attack** around 10 ms and **Release** around 100 ms, then set **Opto** to 1, **Opto severity** between 1 and 3, and **Opto A/R mul** between 2 and 4 (see also the workflow "An 'optical' release" at the end of the chapter). Listen to the ends of phrases, the breaths, the note tails: that's where the opto makes itself heard. For a more pronounced effect, raise the severity first, as François advises; for a lazier opto, raise **Opto A/R mul**. And if you want the rest of the plugin to see this behavior (graph, DEQ 2, bias), fold it into the envelope with **Opto Env**.

Opto precision

GR (cv) tab

Range: 0 to 60 dB **Default:** 0.5 dB **ID:** `opto_precision`

Rounds down the reduction R used to compute the opto's speed, in steps of this value. With 0.5 dB (default), R takes the values 0, 0.5, 1, 1.5…: the speed changes in steps instead of varying continuously.

The rounding only affects the computation of the times, not the applied gain: you don't see a staircase in the reduction itself, but small kinks in its speed (visible on the slowest curves of the previous figure).

- **0**: no rounding;
- **a value larger than the maximum reduction**: R is always rounded to 0, and the opto keeps a constant time (A/R time × A/R mul), whatever the reduction.

The rounding has another consequence: as long as the reduction stays below the step, the opto keeps its base time, which is the slowest.

> **A word from the author**
>
> This setting simulates the number of cells in an optical compressor: one cell every 0.5 dB by default; a larger value corresponds to fewer cells. Its effect is subtle.

Opto saturation

GR (cv) tab

Range: 0 to 1 **Default:** 0 **ID:** `opto_saturation`

Amount of opto "saturation." This setting acts only when the opto **lags behind** the reduction requested by the **curve** (the purple line), that is, during attacks. The gap between the two, in dB, is first compressed by **Opto sat limit**; the opto is then pulled toward the curve, at most until only that compressed gap remains:

- **0** (default): no effect; **Opto sat limit** then has no influence;
- **1**: the opto cannot stay farther from the curve than the compressed gap;
- **in between**: a partial pull.

Since the curve reacts instantly, saturation makes the opto react **right away** at the start of an attack, with a jump in gain, before the smoothing takes over. The larger the gap, the bigger the jump: it is this abrupt reaction, tied to the size of the reduction, that produces the saturation effect.

Saturation only changes the gain produced by the opto, not its internal follower: as soon as the gap disappears, the opto resumes its course.

Opto sat limit

GR (cv) tab

Range: 0.000001 to 480 dB **Default:** 24 dB **ID:** `opto_limite`

Sets the gap compression used by **Opto saturation**. A gap d is brought down to L × d / (L + d), where L is the value of the setting: a small gap passes through almost unchanged, while a large gap is pulled toward L without ever reaching it. With 24 dB (default), a 6 dB gap becomes 4.8 dB, 24 dB becomes 12 dB, and 48 dB becomes 16 dB.

With a small value, the saturated opto can never lag behind the curve by more than L dB: the saturation is strong. With a large value, the gap is barely compressed and the saturation has little effect. No effect if **Opto saturation** is 0.

Going further

Let E be the envelope, R_o the reduction of the optical follower and R_c the reduction requested by the curve, in dB. If R_o < R_c, the gap d = R_c − R_o is compressed to d' = L · d / (L + d), then the reduction produced by the opto becomes R_o + (R_c − d' − R_o) × Opto saturation. Converted to amplitude, this result gives the optical gain O, and the compressed signal is multiplied by 1 + (O / E − 1) × Opto. At each sample, the optical follower advances by a fraction 1 − a^(1/(T · fs)) of its distance to its source, where T is the opto time, a is the altitude of the A/R (cv) tab (1/e by default) and fs is the internal sample rate.

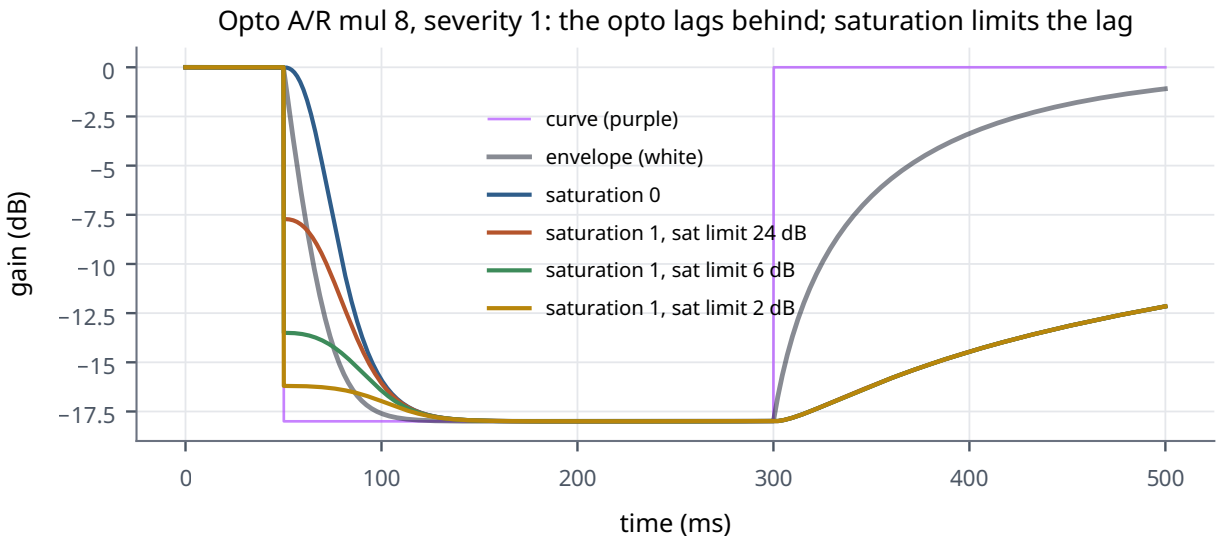


Figure 79 — Opto saturation. With A/R mul at 8, the opto lags far behind an 18 dB request. With Opto saturation at 1, it immediately jumps to a fraction of the requested reduction, and the smaller Opto sat limit is, the larger that fraction. The release is not affected.

Opto Env

GR (cv) tab

Range: −8 to 8 **Default:** 0 **ID:** `opto_onenv`

Chooses where the opto's effect goes.

- **0** (default): the opto stays **separate**. The envelope is not modified; the opto's effect is applied to the signal in the form of a **saturation product**, processed by the **F (gr)** tab (delay, filters, mix, limit).
- **Any other value**: the opto is **folded into the envelope**, with this factor: envelope + (optical gain − envelope) × Opto × Opto Env. At 1 (with Opto at 1), the envelope becomes the optical gain; at 2, the difference is doubled; a negative value flips it.

Once folded in, the opto is part of the envelope: it shows up in the white line and the green dots, it is amplified by **Gr Amp**, and both the second dynamic EQ and the bias see it. On the other hand, there is no saturation product anymore: the settings of the **F (gr)** tab, the delay and **Monitor opto** have nothing left to process (at most, the tail of an echo already in progress may remain). This is the case as soon as Opto Env is not 0, even for a very small value.

A word from the author

You can't have your cake and eat it too: either the envelope becomes the result of the opto, or you keep a saturation product that you can delay and filter.

Monitor opto

GR (cv) tab

Range: 0 to 1 **Default:** 0 **ID:** `delta_opto`

Lets you hear the **opto's saturation product**, after processing by the **F (gr)** tab: it is the difference between the signal with and without the opto. It is the best way to hear what the opto is doing, and to set its filters.

Like the other monitors, it is exclusive: turning it on turns the others off (chapter 7). The signal you hear still goes through the end of the output chain (Headroom return, Makeup, Output Db…), but not through the post-clipper.

The monitor is computed as the compressor's output minus the signal before the opto. It gives exactly the saturation product only if **Comp Clip** and **Comp Db** are neutral, **dW | Dw** is at 0 and the target is the whole signal (no output filters); otherwise, it also contains the differences caused by these settings, or the untargeted remainder of the signal. With **Opto Env** other than 0, it is silent, apart from the tail of any delay echo.

To be confirmed by François

The opto's time is divided by e^severity + R. When the envelope goes above 0 dB (positive Env Offset, positive GR Min…), R becomes negative; if e^severity + R becomes negative, the time becomes negative and the optical follower diverges. In a faithful simulation of the code (severity 1, envelope at +3.5 dB, Opto 1), the target goes silent in less than 0.1 s. Was this case anticipated?

♪ In the studio: the opto as the presets use it

The presets hold a few surprises. None of the three imitations of LA-2A-inspired plugins (**waves_la2a**, **native_vc2a**, **slate_fg2a**) goes through the opto designer: they rely on other settings, such as the **Env Offset** of **slate_fg2a** (−0.55) or the automatic release of **waves_la2a** and **native_vc2a** (**Release Auto** at 0.5). Conversely, **pulsar_mu**, an imitation of a vari-mu plugin, makes full use of it: **Opto** at 1, **Opto saturation** at 0.9 with **Opto sat limit** at 5.979 dB, **Opto A/R mul** at 0.2247, **Opto precision** at 0.6788, a source taken almost entirely from the envelope (**Opt src gn->env** at 0.96) and a product softened by **opto amp** at 0.5287 (F (gr) tab). The opto designer is a shaping tool, not a label. More surprising still, four of the six presets that turn on the opto set it negative, at −1: **waves_76** (**Opto A/R mul** at 2.462, **Opto severity** at −2.97), **ao_britpressor** (**Opto severity** at 0, folded into the envelope by **Opto Env** at 1), **rockruepel_comptwo_drums** (**Opto severity** at −7.96, **Opto saturation** at 1) and **ssl_2_b_** (**Opto A/R mul** at 8). **acqua_eminence** sets it to −0.2, with a very slow opto (**Opto A/R mul** at 421, **Opto severity** at 2.66) folded into the envelope. When it is negative, wherever the opto would lag behind the envelope, the reduction is accentuated instead: the attack digs in faster and the release comes back up sooner. Try it on drums while listening to **Monitor opto**: it's a way to add bite without touching the times.

The opto's saturation product

When **Opto Env** is 0, the opto is not folded into the envelope: it acts as a "saturator" of the compressed signal. Comprendre computes the signal with the opto, subtracts the signal without the opto, and gets a difference: the **saturation product**. It is this product, and this product alone, that is then processed and added back to the signal:

1. it is multiplied by **opto amp**;
2. it goes through a feedback delay (**Opto Delai Secs**, **Opto Delai FB**, GR (cv) tab);
3. three filters extract a low band, a mid band and a high band from it;
4. the unfiltered product (dosed by **opto src mix**) and the three bands (dosed by the **mix** settings) are summed;
5. a clipper limits the result (**opto dst limit**);
6. the result is added to the signal without the opto.

With the default settings (amp 1, src mix 1, bands off, no delay, no limit), you get exactly the signal with the opto. Everything that follows is there to **color** the opto's effect: keep only its low end, delay it, exaggerate it…

This chain is the same for all three of Comprendre's saturation products: the opto's, the waveshaper's (**F (ws)** tab) and the bias's (**F (bs)** tab). Chapter 16 describes it in detail, in particular how the filters and the delay work. Here we only give the essentials.

With the bias

When the bias is active (chapter 19), the whole gain application (Gr Amp, opto and saturation product) is done a second time after the bias, with a second set of filters and delay, independent of the first but with the same settings.

The delay

Opto Delai Secs

GR (cv) tab

Range: 0 to 2 s **Default:** 0 s **ID:** `opto_delai_nb`

Length of the delay applied to the opto's saturation product. This delay acts **only if Opto Delai FB is not 0**: with FB at 0, this setting has no effect.

The actual delay equals this length, rounded down to the sample (at the internal rate, oversampling included), plus one sample. With 0 s and a non-zero FB, the delay is therefore one sample long: the "delay" becomes a simple filter.

This is not a reverb: the idea is to use very short delays to simulate a new harmonic excitation. Longer delays turn it into a creative delay that repeats only the opto's coloration (chapter 16).

Opto Delai FB

GR (cv) tab

Range: −64 to 64 **Default:** 0 **ID:** `opto_delai_fb`

Feedback of the saturation product's delay. The delayed product is mixed with the current product: the share of the current product is 1/(|FB| + 1), the share of the feedback 1 − 1/(|FB| + 1). It is a recursive comb filter: each repeat is weaker than the previous one, and the larger |FB|, the longer the tail.

- **0** (default): no delay;
- **positive**: repeats with the same sign. The very low components of the product are preserved (a gain of 1 at DC);
- **negative**: repeats with alternating signs. The very low components are attenuated.

With **Opto Delai Secs** at 0, the one-sample delay gives a low-pass filter (positive FB) or a filter that attenuates the low end (negative FB), all the more pronounced as |FB| grows.

Gain and filters (F (gr) tab)

opto amp

F (gr) tab

Range: −42 to 42 **Default:** 1 **ID:** `opto_amp`

Gain of the opto's saturation product, applied before the delay and the filters.

- **1** (default): the opto's effect as is;
- **more than 1**: an exaggerated effect; **between 0 and 1**: an attenuated one;
- **0**: the opto is no longer applied to the signal at all, even if **Opto** is not 0 (the computation goes on, and the sky-blue line remains on screen);
- **negative**: the opto's effect is inverted.

This gain also changes the distance of the sky-blue line from mid-height in the history (see above). It has no effect when **Opto Env** is not 0, since there is no longer any product to amplify.

opto src mix

F (gr) tab

Range: 0 to 1 **Default:** 1 **ID:** `opto_src_mix`

Share of the **unfiltered** product in the final mix. At 1 (default), the full product is kept, and the filtered bands are **added** to it. At 0, only the filtered bands are kept: for example, with only the low band active, all you hear is the low end of the product.

At 0 with all bands off, the product is removed: the opto no longer has any audible effect.

> **A word from the author**
>
> There are two schools of thought. Either you start from nothing (src mix at 0) and build the product with a filter, for example a low-pass on the full product; or you keep the whole product (src mix at 1) and the filtered bands are added to it.

All three bands work the same way. Each one is **off** as long as its frequency is 0 (default); it is then not computed. The filters are "zero-delay" digital filters (TPT structure):

- the **low** band is a low-pass, a blend of a first-order filter (6 dB/octave, labeled "zdf") and a second-order state-variable filter (12 dB/octave, adjustable resonance, labeled "svf");
- the **mid** band is always a second-order band-pass (svf);
- the **high** band is a high-pass, a blend of a first-order and a second-order filter, like the low band.

opto low herz

F (gr) tab

Range: 0 to 16000 Hz **Default:** 0 Hz **ID:** `opto_bas_coupe`

Cutoff frequency of the low band's low-pass. At 0 (default), the band is off.

opto low reso

F (gr) tab

Range: 0.000001 to 16 **Default:** 0.707107 **ID:** `opto_bas_resonance`

Resonance (quality factor) of the second-order (svf) part of the low-pass. The default value, $1/\sqrt{2} \approx 0.707$, gives a response with no bump; above that, a bump appears at the cutoff frequency. No effect on the first-order part.

opto low mix

F (gr) tab

Range: −8 to 8 **Default:** 0 **ID:** `opto_bas_mix`

Amount of the low band in the final mix: the filtered band, multiplied by this value, is added to the product. At 0 (default), it is not added. A negative value **subtracts** it: with **opto src mix** at 1 and this setting at −1, you remove the low end from the product, which is roughly equivalent to a high-pass.

opto low zdf->svf

F (gr) tab

Range: 0 to 1 **Default:** 1 **ID:** `opto_bas_type`

Blend between the two low-pass filters: **0** = first order only (6 dB/octave, no resonance), **1** (default) = second order only (12 dB/octave, with the resonance as set), intermediate values = a blend of the two outputs.

opto mid herz

F (gr) tab

Range: 0 to 16000 Hz **Default:** 0 Hz **ID:** `opto_mid_coupe`

Center frequency of the mid band's band-pass. At 0 (default), the band is off.

opto mid reso

F (gr) tab

Range: 0.000001 to 16 **Default:** 0.707107 **ID:** `opto_mid_resonance`

Resonance of the band-pass: it sets the **width** of the band (the higher it is, the narrower the band) and also its **gain** at the center, which is equal to it. At $1/\sqrt{2}$ (default), the center is at −3 dB; at 4, it is at +12 dB.

opto mid mix

F (gr) tab

Range: −8 to 8 **Default:** 0 **ID:** `opto_mid_mix`

Amount of the mid band in the final mix, as with **opto low mix** (negative = subtraction).

opto hgh zdf->svf

F (gr) tab

Range: 0 to 1 **Default:** 1 **ID:** `opto_top_type`

Blend between the two high-pass filters of the high band: **0** = first order (6 dB/octave), **1** (default) = second order (12 dB/octave, with resonance), intermediate values = a blend.

opto hgh herz

F (gr) tab

Range: 0 to 16000 Hz **Default:** 0 Hz **ID:** `opto_top_coupe`

Cutoff frequency of the high band's high-pass. At 0 (default), the band is off.

opto hgh reso

F (gr) tab

Range: 0.000001 to 16 **Default:** 0.707107 **ID:** `opto_top_resonance`

Resonance of the second-order part of the high-pass, as for the low band.

opto hgh mix

F (gr) tab

Range: −8 to 8 **Default:** 0 **ID:** `opto_top_mix`

Amount of the high band in the final mix, as with **opto low mix** (negative = subtraction).

opto dst limit

F (gr) tab

Range: 0 to 42 **Default:** 42 **ID:** `opto_dst_limit`

Clipper applied to the processed saturation product, just before it is added to the signal. The value is an **amplitude** (not decibels): the product is clamped between −limit and +limit.

- **42** (default): clipper off;
- **small values**: the product is clipped, which adds distortion to the product alone;
- **0**: the product is removed, and the opto no longer has any audible effect.

It acts as a safety net when the product gets too loud, for example with a high **opto amp** or a delay with heavy feedback.

Practical workflows

Discovering the opto

1. Start from audible compression, for example Attack 10 ms, Release 100 ms and a few decibels of reduction.
2. Set **Opto** to 1: the sky-blue line appears in the history.
3. Turn on **Monitor opto** to hear the saturation product on its own; if needed, raise **opto amp** to exaggerate it while you adjust.
4. Vary **Opto severity**, then **Opto A/R mul**: the higher they are, the more the opto lags behind the envelope, and the farther the blue line moves away from mid-height.
5. Turn off the monitor, set **opto amp** back to 1, and compare with **Opto** at 0.

An “optical” release

The two-stage release (fast at first, then a long tail) appears as soon as the opto follows the envelope with a medium severity (1 to 3) and an **Opto A/R mul** of 2 to 4. To get this behavior in the envelope itself, and therefore in everything that depends on it (graph, second dynamic EQ, bias), set **Opto Env** to 1; the saturation product then disappears.

Coloring the opto’s effect

With **Opto Env** at 0, the saturation product can be treated as a signal in its own right: **opto src mix** at 0 and a low band around 200 Hz keep only the low end of the optical effect; a very short delay with feedback thickens the effect; **opto dst limit** keeps it from getting too loud. Chapter 16 gives more examples.

Shaping the envelope to imitate a compressor

To get close to an existing compressor (chapter 24), start with the curve and the times, then use:

- **Env Offset**, slightly positive or negative, to recover a resting level or a reduction depth;
- **Gr|Env Pow** below −1 to soften small reductions without touching the large ones much, above 1 to accentuate the whole reduction;
- **Env Pow Unpow** to change the shape of the attack and release without changing the steady-state reduction.

CHAPTER 16

The Saturation Product

Comprendre contains three saturators: the **opto designer** (chapter 15), the **waveshaper** or FET designer (chapter 17) and the **bias** or Vari-Mu designer (chapters 19 to 21). None of the three directly replaces the signal with its saturated version. Each one first computes what the saturation **adds** to the signal: the difference between the saturated signal and the original signal. This difference is called the **saturation product**. It is what gets processed (amplification, delay, filters, limit) and then fed back into the original signal.

Why pay so much attention to saturation in a compressor? Because it is part of its character. A hardware unit doesn't just turn the level down: its tubes, its transistors and its transformers color the sound, often all the more as it compresses harder. That is the idea behind Comprendre's designers. In his video on the FET designer, François puts it this way: by default, it is the gain reduction that drives the saturation, just as on a real compressor, which saturates more when it compresses more. Working on the saturation product lets you go further than any hardware unit allows: keep only the low end of the harmonics to thicken a bass, keep only their top end to give a snare some bite, delay them by a few milliseconds to "multiply" the saturation, or clip them so they don't bring back up the level the compressor has just brought down. Mixers will find a color control of great finesse; sound designers, a playground; imitators, a way to recover a unit's grain on top of its movement.

This chapter describes this shared mechanism. Each of the three saturators has its own tab for processing its product: **F (gr)** for the opto, **F (ws)** for the waveshaper, **F (bs)** for the bias. These tabs have exactly the same layout and work exactly the same way. The **F (ws)** tab serves here as the reference example: all of its cards are in this chapter. The cards for **F (gr)** are in chapter 15, those for **F (bs)** in chapter 21.



Figure 80 — The F (ws) tab: processing the waveshaper's saturation product. F (gr) and F (bs) have the same layout.

How it works: processing what the saturation adds

Let x be the signal entering a saturator and S(x) the signal coming out of it. The saturation product is:

$P = (S(x) - x) \times \text{amount}$

The processed signal is then x + P, once P has gone through the processing chain described below. With its default settings, this chain is neutral. If, in addition, the amount equals 1, you get exactly x + S(x) − x = S(x), the saturated signal. For the opto and the bias, the amount is 1 by default; for the waveshaper, it follows the compressor's gain reduction by default (chapter 17).

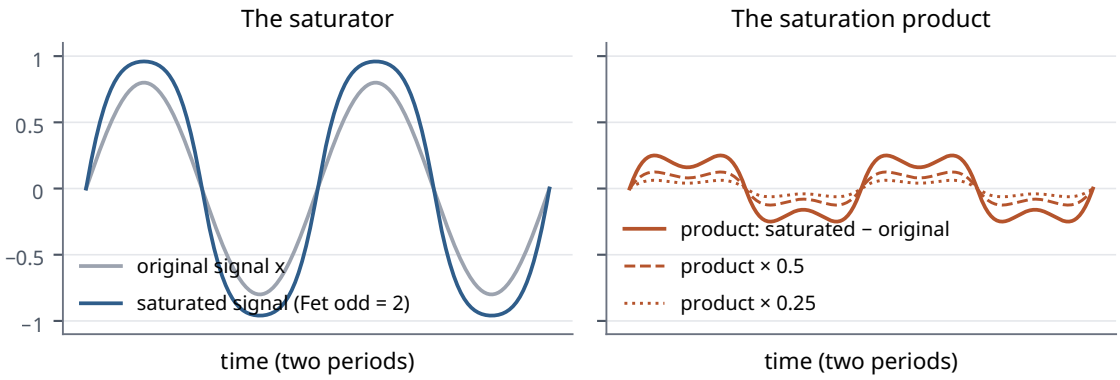


Figure 81 — Left, a sine wave and its version saturated by the waveshaper (Fet odd = 2). Right, the saturation product: the difference between the two. Multiplying the product by 0.5 or 0.25 amounts to dosing the saturation.

Working on the difference rather than on the saturated signal has several advantages:

- **Dosing** the saturation is a matter of multiplying the product: at 0, there is no saturation anymore; at 0.5, you are halfway between the original signal and the saturated signal; above 1, you exaggerate the effect; with a negative value, you flip it.
- **Filtering** the product lets you keep only some of the harmonics it creates (only the lows, only the highs), without touching the original signal.
- **Limiting** the product keeps the saturation from taking over when it gets too strong.
- **Delaying** the product, with feedback, colors the saturation without delaying the original signal.
- **Listening** to the product alone becomes possible: each saturator has a monitor that lets you hear it in isolation.

> **Saturation can also add fundamental**
>
> The product does not contain only new harmonics. A curve that boosts low levels (for example **Fet odd** above 1) also produces a copy of the original signal: the saturated signal is louder than the original, and the product reflects that. This is one of the reasons for the **dst limit** clamp, described further on.

The three saturators

The principle is the same for all three, but the "before" signal, the "after" signal and the amount differ:

	Opto	Waveshaper	Bias
Original signal	the compressed target (after the compressor's gain)	the waveshaper's input, **before** the compressor's gain	the compressor's output without the bias

	Opto	Waveshaper	Bias
Saturated signal	the compressed target multiplied by the opto envelope	the output of the waveshaper's curve	the compressor's output with the bias
Amount	**opto amp**	**Fet gr** and **Fet env** (which dose the gain reduction and the envelope), multiplied by **fet amp**	**bias amp**
Delay	**Opto Delai Secs**, **Opto Delai FB** (GR (cv) tab)	**Fet Delai Secs**, **Fet Delai FB** (WS tab)	**Bias Delay Secs**, **Bias Delay FB** (BIAS dst tab)
Filters, mix, limit	**F (gr)** tab	**F (ws)** tab	**F (bs)** tab
Final dosing	none	**ws** (OUTPUT) and **FX ws** (INPUT)	none
Monitoring	**Monitor opto** (GR (cv))	**Monitor Fet** (WS)	**Mon bias sat** (BIAS dst)
Chapters	15	17	19 to 21

The waveshaper has two particularities. It sits **before** the compressor's gain, whereas the opto and the bias sit after it (chapter 18). And it can act in two places: on the target (what you hear) and on the detection (what the compressor measures, chapter 14). Both use the same settings from the **F (ws)** tab, but each has its own filters and its own delay memory.

> **A word from the author**
>
> François presents the three F tabs as one and the same toolbox: a filter and a delay that act only on the saturation product, identical for the optical saturator, the waveshaper and the bias. Once you understand one, you understand the other two.

> **When color is part of the compressor**
>
> Engineers have always known that their compressors color the sound, and they put that to use. The Fairchild 670 is used as much to enhance as to reduce gain: according to Jack Joseph Puig, it makes notes "long and lush and beautiful." Pushed all the way, the LA-2A distorts through its last tube: Joe Chiccarelli overdrove it this way for the vocals of Jack White, of the White Stripes. On the Neve 2254, the ultra-fast attacks produce obvious transient distortion. SSL 4000 consoles are, according to Jason Goldstein, "always just shy of distorting," and the **4K** mode of the SSL Bus+ deliberately runs its VCA unbalanced, as in those consoles, and adds an adjustable amount of harmonic distortion for a more "colored" sound. The Distressor even goes as far as offering a 1:1 ratio that doesn't compress at all and keeps only its "warming" circuits. Chapter 17 completes this picture with the 1176 and the Distressor's Dist 2 and Dist 3 modes. Comprendre takes these colors, usually welded to the circuit, and turns them into products you can dose, filter and delay.

The processing chain

In all three saturators, the product goes through the same stages, in this order:

1. **Amount**: the product is multiplied by the saturator's **amp** setting (and, for the waveshaper, by the dynamic amount **Fet gr** / **Fet env**);

2. **Feedback delay**: a comb filter, active only if the delay's FB setting is not 0;
3. **Three filter bands** computed on the product: a low band (low-pass), a mid band (band-pass) and a high band (high-pass);
4. **Mix**: the unfiltered product, multiplied by **src mix**, plus each band multiplied by its **mix**;
5. **Limit**: a hard clip of the processed product at ± **dst limit**;
6. for the waveshaper only, **dosing** by **ws** (target) or **FX ws** (detection);
7. **Addition** to the original signal.



Figure 82 — The processing chain of the saturation product, shared by all three saturators.

Going further

With $P_0 = (S(x) - x) \times$ amount:

- delay (if FB ≠ 0): $P_1[n] = s \cdot P_0[n] + c \cdot P_1[n - D]$, with $s = 1/(1 + |FB|)$ and $c = \pm(1 - s)$, with the sign of FB;
- bands: $B = t_b \cdot LP_2(P_1) + (1 - t_b) \cdot LP_1(P_1)$; $M = BP_2(P_1)$; $H = t_h \cdot HP_2(P_1) + (1 - t_h) \cdot HP_1(P_1)$, where LP_2, BP_2 and HP_2 are the outputs of the state-variable filter (2 poles), LP_1 and HP_1 those of the one-pole filter, and t_b and t_h the **zdf->svf** settings;
- mix: $P_2 = P_1 \cdot$src mix + B·low mix + M·mid mix + H·hgh mix;
- limit: $P_3 = P_2$ clamped to [−dst limit, +dst limit] if dst limit < 42;
- output: $x + P_3$ (× dosing for the waveshaper).

A band whose frequency is 0 is not computed.

All of this processing is done sample by sample, at the plugin's internal rate (oversampling included). It adds no latency.

The amount: amp

Each F tab starts with an **amp** setting that multiplies the product before any other operation. For the opto and the bias, it is the only amount. For the waveshaper, **fet amp** multiplies the dynamic amount set in the WS tab (**Fet gr**, **Fet env**, chapter 17).

fet amp

F (ws) tab

Range: −42 to 42 **Default:** 1 **ID:** `fet_amp`

Multiplies the waveshaper's saturation product, right after it is computed and before the delay, the filters and the limit. It acts on the target's waveshaper as well as on the detection's.

- **1** (default): the product is left intact; the waveshaper's curve is applied as drawn, in the proportion set by **Fet gr** and **Fet env**.
- **Between 0 and 1**: the saturation is attenuated. The signal is a blend of the original signal and the saturated signal.
- **0**: the product is zero. The waveshaper is still computed, but it no longer has any audible effect.
- **Above 1**: the effect of the curve is exaggerated. At 2, the signal departs from the original twice as much as the drawn curve would make it.
- **Negative**: the product is subtracted instead of added. A curve that compresses the peaks then starts accentuating them, and vice versa: it is an "anti-saturation." At −1, the signal departs from the original as much as with the curve, but in the other direction.

Since **fet amp** multiplies the dynamic amount, it changes the intensity of the saturation without changing its law: with the default settings, the saturation stays proportional to the gain reduction.

The knob goes from −42 to +42 and is very tightly packed around the center: the value 1 sits at about four fifths of its travel. For precise values, type them in.

> **♪ In the studio: dosing the color**
>
> The **amp** setting is each saturator's "color" knob: it doses the grain without touching the compression. Several presets use it to keep just a touch of saturation: **kiive_xstressor**, an imitation of a Distressor-inspired plugin, sets **fet amp** to 0.0625; **mixland_vacattack** sets it to 0.5 and **blackrooster_omnitec436** to 0.5287. On the opto's product, **pulsar_mu** sets **opto amp** to 0.5287; on the bias's product, **pulsarmodular_p42** sets **bias amp** to 0.475, and **softube_germanium** takes it negative, to −0.5: the bias saturation is flipped. At the other extreme, **rockruepel_comptwo_drums** pushes **fet amp** to 42, the maximum, but feeds back only a filtered, inverted band of the product (see below). To find your amount, listen to the product alone with the saturator's monitor, bring **amp** up from 0 until you can feel the grain in the mix, then back off a notch: a saturation you barely notice is often missed as soon as you take it away.

The feedback delay

Each saturator has a delay that acts only on its product. For the waveshaper, its two settings, **Fet Delai Secs** and **Fet Delai FB**, are in the **WS** tab and have their cards in chapter 17. For the opto, they are in the **GR (cv)** tab (chapter 15); for the bias, in the **BIAS dst** tab (chapter 21). All three work the same way.

A comb filter, not an echo

Contrary to what its name suggests, this delay does not add an echo to the product. It is a **feedback comb filter**:

- the incoming product is first attenuated by the factor s = 1/(1 + |FB|);
- the output of the delay, delayed by **Delai Secs** and multiplied by c = 1 − s (or −(1 − s) if FB is negative), is added to it;
- the result is both the output and what is written back into the delay line.

Two consequences follow.

1. **At FB = 0, the delay is off**, whatever the value of **Delai Secs**. There is no “single echo, no feedback” position.
2. **The larger FB, the more the direct product is attenuated** and the stronger the feedback. The total gain stays at 1 for the frequencies that “fit” the loop exactly (the multiples of 1/delay), but it drops everywhere else: the product concentrates on a series of evenly spaced frequencies.

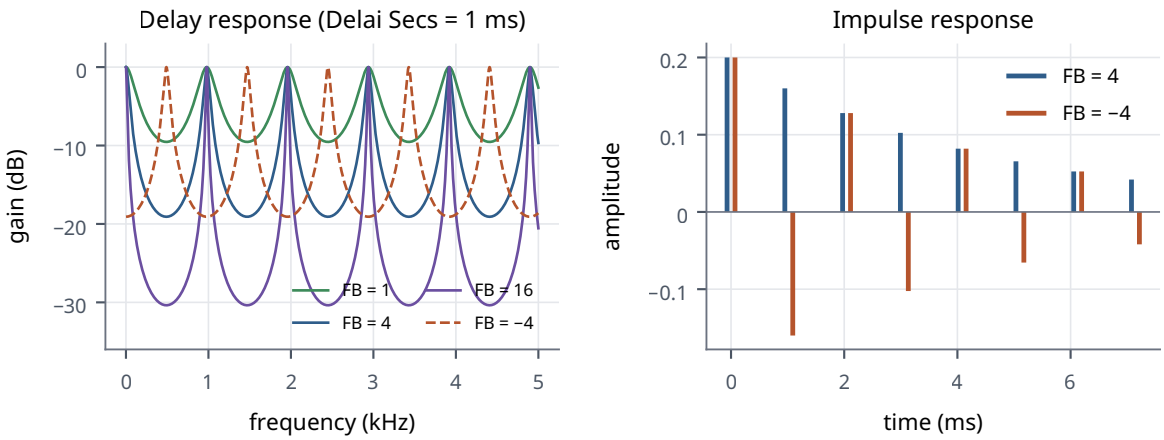


Figure 83 — Left, the frequency response of the delay for a 1 ms delay: with positive FB, the peaks sit at multiples of about 1 kHz; with negative FB, they are shifted by half an interval. Right, the impulse response: decaying repeats, with the same sign (FB = 4) or alternating (FB = −4).

Between the peaks, the gain drops to 1/(1 + 2·|FB|): −9.5 dB for FB = ±1, −19 dB for FB = ±4, −30 dB for FB = ±16. With a negative FB, the peaks are at the **odd** multiples of half of 1/delay, and DC (0 Hz) is attenuated instead of preserved.

Over time, each repeat equals the previous one multiplied by c: 0.5 for FB = 1, 0.8 for FB = 4, about 0.94 for FB = 16 and 0.985 for FB = 64. With a high FB and a long delay, the tail gets very long: at FB = 64, it takes about 65 repeats for the product to fall back to a third of its value.

Going further

The delay is counted in internal samples: N = integer part of (Delai Secs × internal rate), and the read happens before the write, so the effective delay is D = N + 1 samples. The transfer function is H(z) = s / (1 − c·z^(−D)), a recursive comb filter that is stable since |c| < 1. For FB > 0, its gain is 1 at the frequencies k·fs/D and s/(1 + c) between two peaks; for FB < 0, the peaks are at the frequencies (k + ½)·fs/D and the gain at DC is 1/(1 + 2·|FB|).

With zero delay: a one-pole filter

When **Delai Secs** is 0 and FB is not 0, the effective delay is a single sample. The comb becomes a simple one-pole filter that acts on the whole spectrum:

- with a **positive FB**, a low-pass: the lows are preserved and the highs are attenuated, starting lower in frequency as FB gets larger;
- with a **negative FB**, the reverse: the whole product is attenuated by 1/(1 + 2·|FB|), except the extreme highs, which climb back toward 0 dB at half the sample rate.

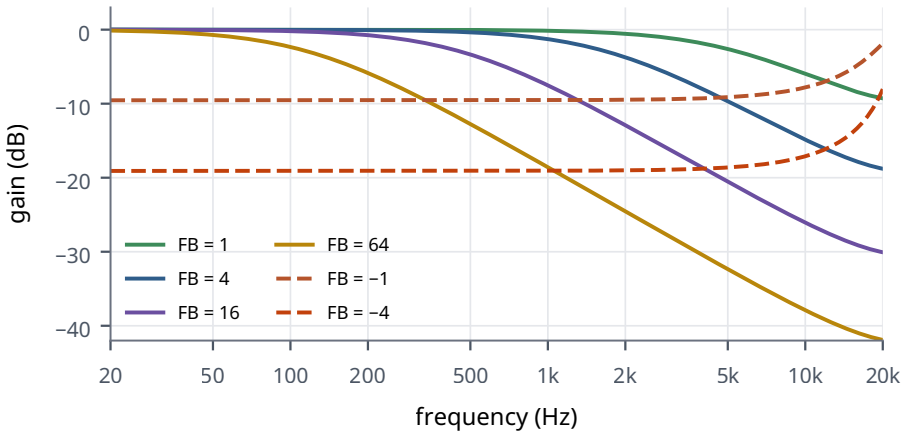


Figure 84 — Delay set to 0 s: response of the resulting one-pole filter, at 48 kHz without oversampling.

The one-pole filter depends on oversampling

This one-sample delay is one **internal** sample. With ×4 oversampling, it is four times shorter, and all the frequencies in the figure above are multiplied by 4. If you use the delay at 0 s as a filter, set it after choosing the oversampling. For a non-zero delay, the length in seconds is preserved, to within one sample.

A word from the author

This delay is not meant for creating reverb. François designed it for very short delays, which simulate a new excitation of the harmonics and “multiply” the saturation. Nothing stops you, though, from using it as a creative delay on the product alone, subtler than a delay on the whole signal.

To be confirmed by François

The delay memory holds 16 seconds at the host's sample rate, but the delay is counted at the over-sampled rate. With ×8 oversampling, a 2 s delay comes down to a delay of a single sample; with ×16, any delay longer than 1 s is shortened. Is this intended?

♪ In the studio: multiplying the saturation

The product delay is one of Comprendre's most unusual ideas. With a delay of a few milliseconds and moderate feedback, the product concentrates on a series of evenly spaced frequencies: the saturation seems to resonate, as if its harmonics were being excited a second time, while the original signal is not delayed. This is the use François designed it for. Try it on a guitar, a synth or a vocal that the compressor drives into saturation: **Fet Delai Secs** between 0.5 and 5 ms, **Fet Delai FB** between 1 and 4, while listening to **Monitor Fet**, then reduce **fet amp** if the product becomes too present. For sound design, lengthen the delay and raise the feedback: you get an echo that repeats only the color, never the sound itself. Among the included presets, only one uses a product delay: **acqua_ash-ultra**, an imitation of a clipping plugin, on the bias product (**Bias Delay Secs** at 0.01454 s, **Bias Delay FB** at 0.1854).

The three filter bands

After the delay, the product feeds three filters in parallel:

Band	Filter	Frequency	Resonance	Type
low	low-pass	**low herz**	**low reso**	**low zdf->svf**
mid	band-pass	**mid herz**	**mid reso**	(always SVF)
high	high-pass	**hgh herz**	**hgh reso**	**hgh zdf->svf**

A band is **off** as long as its frequency is 0 (the default setting): it is then not computed, and its **mix** setting has no effect.

ZDF and SVF

The filters come from the JUCE library and belong to the so-called "zero-delay feedback" family (TPT topology). They react without delay and remain stable when you sweep the frequencies during playback. Comprendre uses two kinds, which it calls ZDF and SVF:

- the **ZDF** is a **one-pole** filter: a slope of 6 dB per octave, no resonance;
- the **SVF** (*state variable filter*) is a **two-pole** filter: a slope of 12 dB per octave, with adjustable **resonance**.

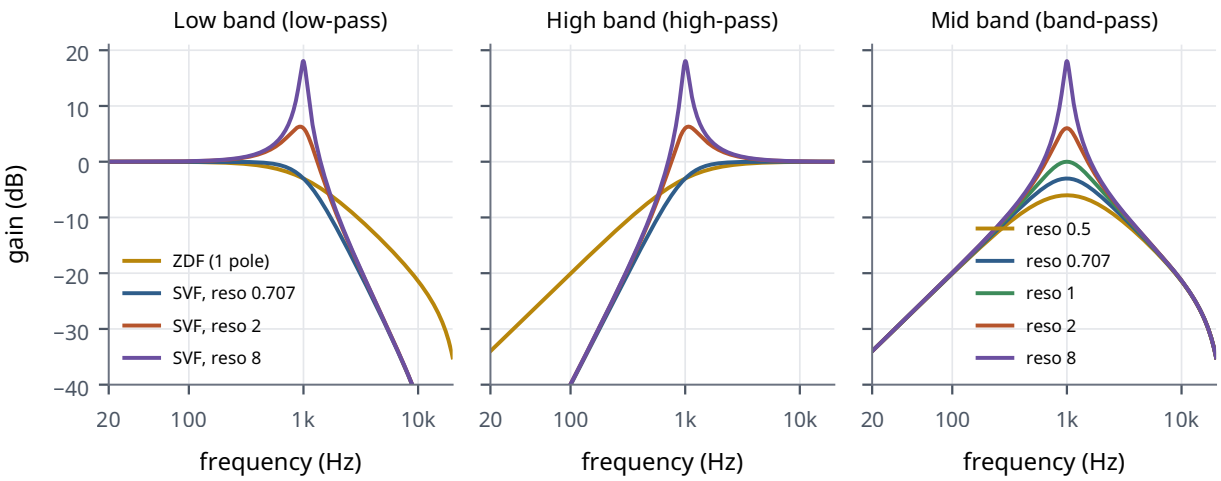


Figure 85 — Responses of the three bands for a frequency of 1 kHz, at 48 kHz. Low and high bands: ZDF, then SVF for three resonance values. Mid band: SVF band-pass for five resonance values.

The resonance is the quality factor Q of the SVF filter. Its default value, $1/\sqrt{2} \approx 0.707$, gives a response with no bump (a Butterworth filter): the low-pass and the high-pass are at −3 dB at the set frequency. Above that, a bump appears around the frequency, all the higher and narrower as the resonance increases (about +6 dB for a resonance of 2, +18 dB for 8). Below it, the filter gets gentler and the cutoff spreads out.

For the **mid** band, the band-pass is **not normalized**: its gain at the center equals the resonance. At 0.707, the center is at −3 dB; at 1, at 0 dB; at 2, at +6 dB; at 16, at +24 dB. The resonance therefore sets both the **width** of the band (the larger it is, the narrower the band) and its **level**. For a narrow band that isn't louder, compensate with **mid mix**.

The zdf->svf blend

For the low and high bands, the **zdf->svf** setting blends the **outputs** of the two filters: at 0, you hear only the ZDF; at 1 (default), only the SVF; in between, a weighted sum of the two. There is no type setting for the mid band, which is always an SVF band-pass.

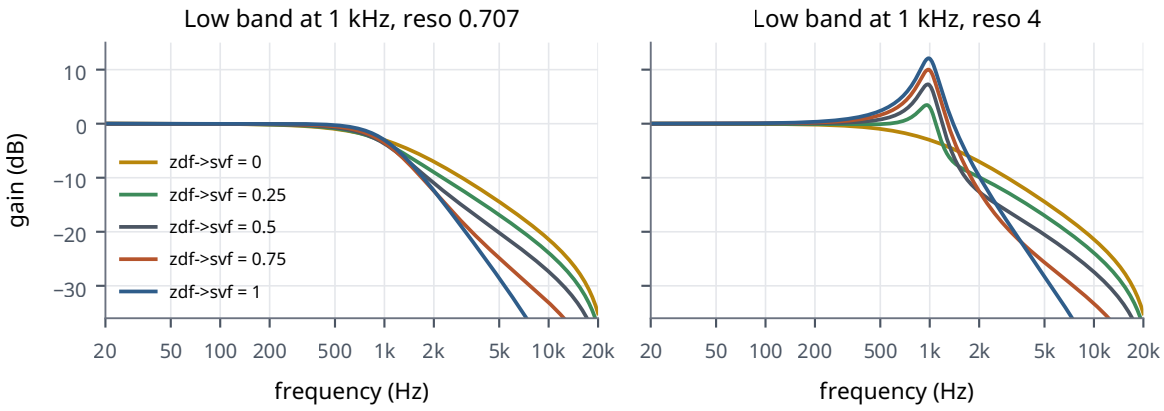


Figure 86 — Effect of the zdf->svf setting on the low band at 1 kHz, with the default resonance of 0.707 (left) and a resonance of 4 (right).

The blend gives intermediate slopes: the cutoff starts like the SVF's, then eases into 6 dB per octave in the extreme highs (or lows). The resonance, which exists only in the SVF, decreases in proportion to the SVF's share.

> **A word from the author**
>
> The SVF lets you use resonance, the ZDF doesn't; the two can be blended. For the mid band, it is necessarily an SVF, and its resonance is used to set the width of the band.

Band settings

fet low herz

F (ws) tab

Range: 0 to 16000 Hz **Default:** 0 Hz **ID:** `fet_bas_coupe`

Cutoff frequency of the low-pass of the product's low band. At **0** (default), the band is off and is not computed.

The low band contains only the low end of the product, below this frequency. What it brings to the result depends on **fet low mix**: you can add it (to reinforce the low harmonics), subtract it (to remove them) or use it on its own (with **fet src mix** at 0).

The knob is progressive: the middle of its travel is 1,000 Hz, the quarter mark 62.5 Hz and the three-quarter mark about 5 kHz. The maximum frequency is 16 kHz.

fet low reso

F (ws) tab

Range: 0.000001 to 16 **Default:** 0.707107 **ID:** `fet_bas_resonance`

Resonance (Q factor) of the low band's SVF low-pass. It acts only on the SVF part of the filter: with **fet low zdf->svf** at 0, it has no effect.

- **0.707** (default): a clean cutoff, with no bump.
- **Higher**: a bump around the cutoff frequency. It lets you bring out a precise area of the product's low end, much like the resonance of a synthesizer filter.
- **Lower**: a very gentle cutoff, which starts well before the set frequency.

fet low zdf->svf

F (ws) tab

Range: 0 to 1 **Default:** 1 **ID:** `fet_bas_type`

Blend between the one-pole low-pass (ZDF, 6 dB per octave, at **0**) and the two-pole low-pass (SVF, 12 dB per octave, at **1**, default) of the low band. Intermediate values add the two outputs in the chosen proportion.

The ZDF is useful when you want an exact subtraction: a product minus its ZDF low-pass gives exactly its ZDF high-pass (see **fet low mix**).

fet mid herz

F (ws) tab

Range: 0 to 16000 Hz **Default:** 0 Hz **ID:** `fet_mid_coupe`

Center frequency of the mid band's band-pass. At **0** (default), the band is off.

The mid band is always an SVF band-pass: it has no type setting. The knob has the same progressive taper as the low band's (1,000 Hz in the middle).

fet mid reso

F (ws) tab

Range: 0.000001 to 16 **Default:** 0.707107 **ID:** `fet_mid_resonance`

Resonance of the mid band-pass. It sets the **width** of the band: the larger it is, the narrower the band around the center frequency.

Caution: the gain at the center of the band equals this value. At **0.707** (default), the center is at −3 dB; at **1**, at 0 dB; at **4**, at +12 dB; at **16** (maximum), at +24 dB. A high resonance therefore makes the band both narrower and much louder; adjust **fet mid mix** accordingly.

fet hgh herz

F (ws) tab

Range: 0 to 16000 Hz **Default:** 0 Hz **ID:** `fet_top_coupe`

Cutoff frequency of the high band's high-pass. At **0** (default), the band is off.

The high band contains only the top end of the product, above this frequency. It is typically used to dose the high-frequency "grain" of the saturation separately, which is often the most aggressive part.

fet hgh reso

F (ws) tab

Range: 0.000001 to 16 **Default:** 0.707107 **ID:** `fet_top_resonance`

Resonance of the high band's SVF high-pass. As for the low band, 0.707 (default) gives a cutoff with no bump, a higher value creates a bump around the cutoff frequency, and it has no effect on the ZDF part.

fet hgh zdf->svf

F (ws) tab

Range: 0 to 1 **Default:** 1 **ID:** `fet_top_type`

Blend between the one-pole high-pass (ZDF, at **0**) and the two-pole high-pass (SVF, at **1**, default) of the high band. Works the same way as **fet low zdf->svf**.

The mix: src mix and band mixes

The processed product is the sum of four terms: the unfiltered product, multiplied by **src mix**, and each of the three bands, multiplied by its **mix**. The band **mix** settings are not gains in decibels but coefficients: the band is **added** to the product in that proportion, positively or negatively.

With the default settings (**src mix** at 1, the three **mix** settings at 0, the three frequencies at 0), the product passes through intact. The useful combinations fall into two families, which François calls the “two schools”:

- **replace** the product with one or more of its bands: **src mix** at 0, and the **mix** of the bands you want at 1. For example, src mix 0 and low mix 1: only the low part of the saturation is fed back;
- **correct** the product: **src mix** at 1, with positive **mix** values (to reinforce an area) or negative ones (to weaken it).

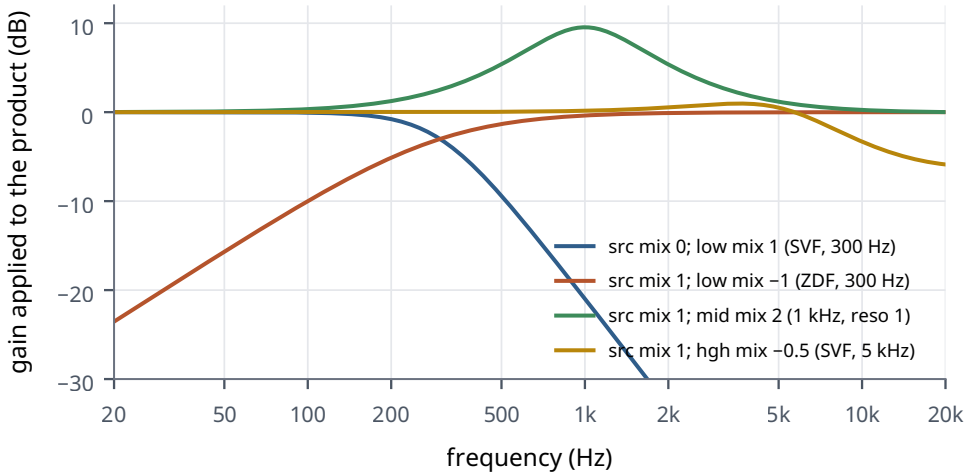


Figure 87 — Gain applied to the product for a few mix combinations, at 48 kHz.

A few reference points for a band whose filter passes the signal with a gain of 1 (the lows for a low-pass, the center for a band-pass with a resonance of 1):

- **mix** = **1** adds the band once: the product is doubled in that area, which is +6 dB;
- **mix** = **−1** removes the band: with a ZDF filter, the result is exactly the complement (the product minus its ZDF low-pass is its ZDF high-pass); with the mid band-pass at a resonance of 1, you get a complete notch at the center;
- **mix** < **−1** removes more than the band: it comes back with inverted polarity;
- **mix** = **2** on a band-pass with a resonance of 1 gives +9.5 dB at the center (1 + 2 = 3 times).

With an SVF, the filter’s phase shifts make the sum less simple: small bumps or dips appear around the cutoff frequency.

fet src mix

F (ws) tab

Range: 0 to 1 **Default:** 1 **ID:** `fet_src_mix`

Proportion of the **unfiltered** product in the processed product. It ranges from 0 to 1.

- **1** (default): the whole product is kept; any active bands are added to it or subtracted from it.
- **0**: only what the bands supply is kept. If no band is active, the product is removed and the saturator no longer has any effect.
- **In between**: a blend of the full product and its filtered versions.

This setting cannot be negative; to remove an area from the product, use a negative band **mix**.

fet low mix

F (ws) tab

Range: −8 to 8 **Default:** 0 **ID:** `fet_bas_mix`

Coefficient of the low band in the processed product. It ranges from −8 to +8; at **0** (default), the band is not used. It has an effect only if **fet low herz** is not 0.

- **Positive**: the product's low end is reinforced (or makes up the whole product on its own, with **fet src mix** at 0).
- **Negative**: the product's low end is removed. At −1 with a ZDF filter (**fet low zdf->svf** at 0), the result is an exact 6 dB per octave high-pass.

The knob is very tightly packed around 0; type in precise values. Despite its name on the host side ("fet low env mix"), this coefficient is constant: it does not depend on any envelope.

fet mid mix

F (ws) tab

Range: −8 to 8 **Default:** 0 **ID:** `fet_mid_mix`

Coefficient of the mid band in the processed product, from −8 to +8; at **0** (default), the band is not used. It has an effect only if **fet mid herz** is not 0.

Don't forget that the band-pass has a center gain equal to its resonance: with a resonance of 4 and a mix of 1, the center of the band is multiplied by 1 + 4 = 5 (+14 dB). To notch out the center exactly, set a resonance of 1 and a mix of −1.

fet hgh mix

F (ws) tab

Range: −8 to 8 **Default:** 0 **ID:** `fet_top_mix`

Coefficient of the high band in the processed product, from −8 to +8; at **0** (default), the band is not used. It has an effect only if **fet hgh herz** is not 0.

A negative mix softens the saturation by removing its high harmonics; a positive mix makes it brighter. At −1 with a ZDF filter, the product becomes exactly its ZDF low-pass.

> **♪ In the studio: choosing which harmonics to keep**
>
> The presets show two opposite ways of sculpting the saturation. Some keep only its **low end**, to thicken without adding grain: **softube_germanium** (**fet src mix** at 0, **fet low herz** at 450 Hz, **fet low mix** at 1, at 6 dB per octave with **fet low zdf->svf** at 0) or **pulsar_1178** (**fet src mix** at 0, low band at 100 Hz, **fet low mix** at 4, **fet low reso** at 0.25). **ao_dbcomp** removes the top end from a full product (**fet hgh herz** at 1,000 Hz, **fet hgh mix** at −1, in ZDF): all that remains is its low-pass at 1 kHz. Others keep only the **top end**, the grain with the most bite: **waves_76** (**fet src mix** at 0, **fet hgh herz** at 5,566.29 Hz, **fet hgh mix** at 1.179, in ZDF) and **softube_cl1b** (**fet src mix** at 0, **fet hgh herz** at 6,000 Hz, **fet hgh mix** at 0.75). Notice that **waves_76** and **pulsar_1178**, two imitations of plugins from the 1176 family, make opposite choices: an imitation is tuned to the plugin being imitated, by ear and in analysis mode, not by following a recipe. On your own tracks, start from the same question: should the saturation thicken the bottom, or brighten the top? On a bass, the low end of the product often adds weight without muddying the attack; on a snare or a guitar, the top end of the product adds bite without thickening the bottom.

The limit: dst limit

The last stage before the product is fed back, the limit is a hard clip of the processed product: each sample of the product is clamped between −**dst limit** and +**dst limit**.

fet dst limit

F (ws) tab

Range: 0 to 42 **Default:** 42 **ID:** `fet_dst_limit`

Maximum value, in linear amplitude, of the waveshaper's processed product. The value 1 corresponds to full internal level (0 dBFS after the Headroom).

- **42** (default): the limit is off.
- **Between 0 and 42**: the product is clipped at ± this value. For example, 0.1 limits the product to −20 dBFS, and 0.01 to −40 dBFS.
- **0**: the product is removed entirely; the waveshaper no longer has any effect.

The clipping applies to the product alone, not to the signal: the original signal always passes through in full, and only the part added by the saturation is capped. On the waveshaper, the limit acts before the **ws** / **FX ws** dosing and before the compressor's gain.

The knob is extremely progressive: the value 1 sits at about 80% of its travel, and the entire first half corresponds to values below 0.001. Typing the value in is easier.

> **A word from the author**
>
> The limit addresses a paradox of dynamic saturation: the more the compressor reduces the gain, the more the waveshaper distorts the signal, but distorting often means amplifying. Without a safeguard, the saturation can then bring back up the level the compressor has just brought down. The limit, a clipper that acts only on the saturation product, lets you keep the color without that rebound in level.

Since it is a hard clip, a low limit itself adds harmonics to the product. That doesn't matter when the product is small compared to the signal, but it becomes audible if you clip a lot.

> **♪ In the studio: color without raising the level**
>
> On a bus or a master, the paradox François describes shows up quickly: the harder the compressor works, the more the waveshaper distorts, and the more the product pushes back up the peaks the compression has just held down. **dst limit** lets you keep the grain while capping what it adds. Listen to the product alone with **Monitor Fet**, spot the height of its peaks on the loudest passages, then set the limit a little below that: the color stays, the excess goes away. If the clipping of the product becomes audible in itself, lower **fet amp** instead, which scales the product down proportionally rather than shaving it off.

Back into the signal

Finally, the processed product is added to the saturator's original signal:

- for the **opto** and the **bias**, directly;
- for the target's **waveshaper**, after multiplication by the **ws** setting of the OUTPUT tab (from 0 to 1, default 1);

- for the detection's **waveshaper**, after multiplication by the **FX ws** setting of the INPUT tab (from −1 to 1, default 0: by default, the waveshaper does not act on the detection).

These dosing settings are described in chapter 14.

To hear what the chain is doing, the simplest way is to listen to the product alone. The **Monitor Fet** monitor (WS tab, chapter 17) lets you hear the product of the target's waveshaper, after all the processing of the F (ws) tab and the **ws** dosing, without the compressor's gain. **Monitor opto** and **Mon bias sat** play the same role for the other two saturators (chapters 15 and 21).

Practical workflows

Keep only the low end of the saturation. Set **fet src mix** to 0, **fet low herz** to around 200 to 400 Hz and **fet low mix** to 1. Only the low harmonics are fed back: the saturation thickens the bottom of the spectrum without adding grain up top.

Remove the low end of the saturation. Keep **fet src mix** at 1, set **fet low herz** to the frequency you want, **fet low zdf->svf** to 0 and **fet low mix** to −1. The product then goes through an exact 6 dB per octave high-pass. With a low frequency (a few tens of hertz), this combination also removes the DC component that some waveshaper settings can create (**Fet even**, DC bias: chapter 17).

Bring out an area. With **fet src mix** at 1, turn on the mid band over the area you want, with a resonance between 1 and 4 and a positive **fet mid mix**. The saturation then seems to "sing" at that frequency.

Soften a harsh saturation. Turn on the high band around 3 to 6 kHz and give it a negative **fet hgh mix**, between −0.5 and −1.

Tame a product that is too loud. Listen to the product with **Monitor Fet**, then lower **fet dst limit** until the product's peaks are contained. Another approach is to reduce **fet amp**, which lowers the product proportionally instead of clipping it.

Add a resonance to the saturation. Set **Fet Delai Secs** to a few milliseconds and **Fet Delai FB** between 2 and 8. The product concentrates on a series of frequencies spaced 1/delay apart (1 kHz for 1 ms): the saturation takes on a metallic color, without the original signal being delayed.

The F (gr) and F (bs) tabs

The **F (gr)** and **F (bs)** tabs have the same fourteen settings as **F (ws)**, in the same order, with the prefixes "opto" and "bias": **amp**, **src mix**, the three bands (**herz**, **reso**, **mix**, plus **zdf->svf** for the low and high bands) and **dst limit**. They work as described in this chapter. The only things that change are what enters the chain and the meaning of **amp**:

- **F (gr)** processes the opto's product, computed after the compressor's gain. Its cards and its particularities (in particular the effect of the **Opto Env** setting, which folds the opto into the envelope and makes the product zero) are in chapter 15.
- **F (bs)** processes the bias product, the difference between the output with the bias and the output without it. Its cards and its particularities are in chapter 21.

Starting ideas

These starting points apply to all three F tabs: just replace “fet” with “opto” or “bias.”

- **Thicken a bass or a kick drum**: keep only the low end of the product (**fet src mix** at 0, **fet low herz** around 200 to 450 Hz, **fet low mix** at 1), like **softube_germanium**.
- **Give a snare or a guitar some bite**: keep only the top end of the product (**fet src mix** at 0, **fet hgh herz** around 5 to 6 kHz, **fet hgh mix** between 0.75 and 1.2), like **waves_76** and **softube_cl1b**.
- **Soften a saturated vocal**: keep the full product and remove its top end, with a high band around 3 to 6 kHz and a negative **hgh mix**; or, like **ao_dbcomp**, a **hgh mix** at −1 in ZDF.
- **Color a master without raising its level**: a modest **amp** (0.1 to 0.5) and a **dst limit** just below the product’s peaks.
- **Make the saturation sing**: a narrow mid band (resonance of 2 to 4) over the area you want, with a positive **mid mix**, and the delay set to a few milliseconds.
- **Sound design**: a negative **amp** for anti-saturation, which accentuates the peaks instead of rounding them off; a delay of several tens of milliseconds, with heavy feedback, for an echo of the color alone; a very low **dst limit** for a clipped grain that never touches the original signal.
- **To understand**, finally: listen to the product alone with the saturator’s monitor, then turn the settings in this chapter one by one. Hearing in isolation what a saturation adds is a rare experience, and one of the best schools for the ear.

CHAPTER 17

The Waveshaper (FET Designer)

Push a 1176 on a bass or a snare: as it clamps down, its slight transistor distortion blends into the sound, and it is said to brighten the tone and add bite. On the hardware, this color is welded to the circuit: you take it as it is, or you switch units. In Comprendre, it becomes an instrument. You choose which harmonics are born (the second alone, the odd ones, or all of them), when (only when the compressor is working, all the time, or, on the contrary, in the quiet passages) and with what grain: a soft curve, hard clipping, white noise reserved for the peaks, the stair steps of a bit crusher. The curve is drawn before your eyes in the small graph at the top right of the window, and **Monitor Fet** lets you hear the saturation on its own.

For mixing, it's the bite of a FET compressor on a bass or a drum bus, or a hint of second harmonic that thickens a vocal. For mastering, a barely perceptible coloration, dosed by the compression itself. For sound design, a six-stage saturator that can react to dynamics. The waveshaper is one of the modules most used by the included presets: 26 of the 80 presets change the **WS** tab, a sign that saturation is part of the character of the units François has imitated.

A field-effect transistor (FET) compressor colors the sound more when it compresses hard. Comprendre's **waveshaper**, which François also calls the **FET designer**, is there to reproduce this kind of behavior, and many others. It is a transfer curve: each input sample is turned into an output sample according to a function that you draw with the settings of the **WS** tab. As with the plugin's other saturators, it is not the output of the curve that is fed back, but the **saturation product**, the difference between the output and the input (chapter 16). By default, the amount of this product follows the gain reduction: the more the compressor reduces the gain, the more the signal is distorted.

The waveshaper sits in the target **before** the compressor's gain, between the two dynamic EQs: DEQ, then WS, then DEQ 2 (chapter 18). It can also be sent into the detection (chapter 14). Its product is processed by the **F (ws)** tab, described in chapter 16.

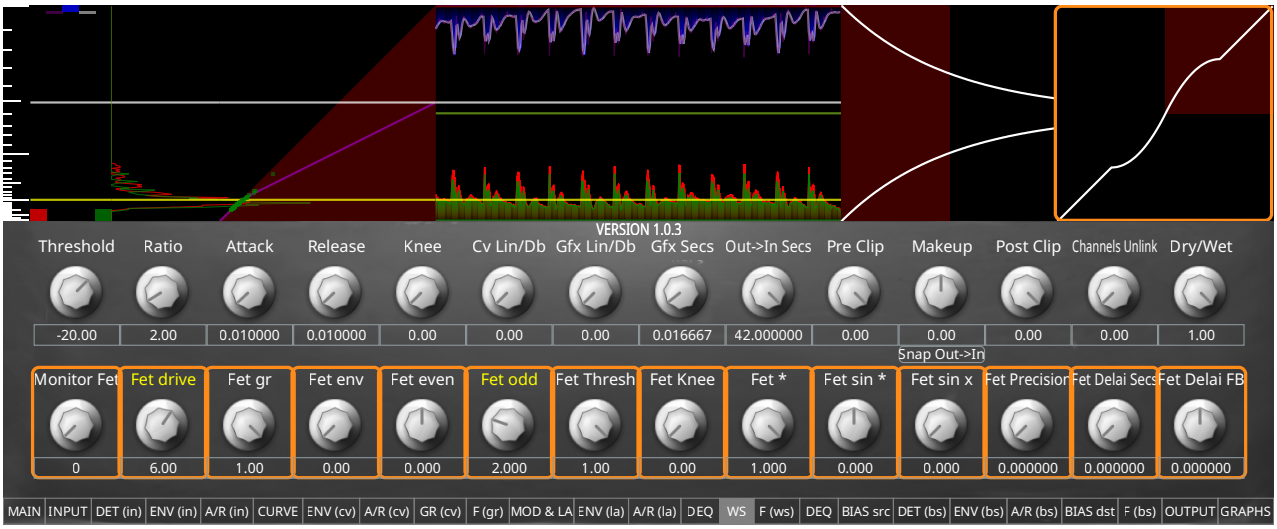


Figure 88 — The WS tab and its fourteen settings. At the top right (boxed), the waveshaper's curve, here for Fet odd = 2 and Fet drive = +6 dB.

How it works

Six stackable stages

The waveshaper’s curve is made of six stages chained end to end, always in the same order. Each one is neutral at its default setting, and you can combine them freely:

1. **Fet even**: a blend of the signal and its square, which creates the second harmonic;
2. **Fet odd**: a symmetrical curve that creates odd harmonics;
3. **Fet Thresh** and **Fet Knee**: a clipper, hard or with a knee;
4. **Fet ***: a simple multiplication;
5. **Fet sin *** and **Fet sin x**: the addition of a sine function of the input (or of a constant offset, the DC bias);
6. **Fet Precision**: quantization, in the manner of a bit crusher.

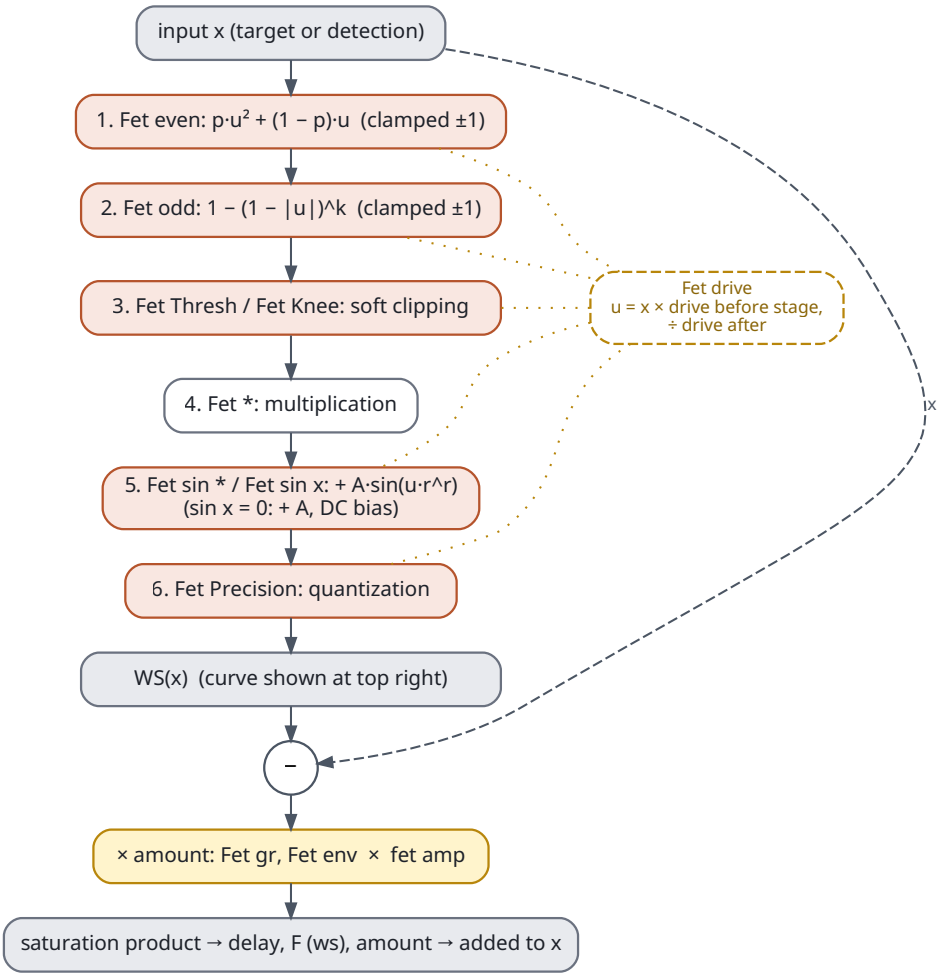


Figure 89 — The waveshaper's six stages. The Fet drive wraps around stages 1, 2, 3, 5 and 6: the signal is multiplied by the drive before the stage and divided by it afterward.

The waveshaper turns on only if at least one stage departs from its neutral setting. **Fet drive** alone does not turn it on: it changes nothing as long as no stage is engaged.

Input level

The curve works on amplitudes: ±1 corresponds to full internal level, that is, 0 dBFS **after the Headroom** (chapter 7). The threshold, precision and sine settings are therefore read relative to that level. A signal peaking at −12 dBFS uses only the central part of the curve, between −0.25 and +0.25.

Raising the **Headroom** drives the signal harder into the waveshaper. Since the Headroom gain is removed right after the compressor, the effect resembles that of **Fet drive**; but the Headroom also makes the compressor work harder, and therefore, with the default settings, saturate more.

From signal to saturation

The waveshaper computes the curve WS(x), then the product:

P = (WS(x) − x) × amount × **fet amp**

The amount, set by **Fet gr** and **Fet env**, equals the compressor's gain reduction by default, between 0 (no compression) and 1 (total reduction). The product then goes through the **F (ws)** tab (delay, filters, limit), and is then added to the original signal.

With neutral F (ws) processing, the signal you hear therefore follows a curve **in between** the diagonal (untouched signal) and the drawn curve: x + amount × (WS(x) − x). The displayed curve is the one you get when the amount equals 1. With the default settings, at 6 dB of gain reduction, the amount is 0.5 and the signal follows a curve halfway between the diagonal and the displayed curve.

> **Saturation depends on the uncompressed level**
>
> The waveshaper acts before the multiplication by the compressor's gain: the curve sees the **uncompressed** signal. The amount of saturation, however, depends on the gain reduction of the current sample. On the detection, where the gain of the current sample has not been computed yet, the gain of the previous sample is used.

The displayed curve

The small square graph at the top right of the window (graph 8, chapter 4) shows the curve WS(x): horizontally, the input from −1 to +1; vertically, the output, on the same scale. The upper right quadrant is tinted red. With the default settings, the curve is the diagonal.

The curve is computed with exactly the same formulas as the audio processing, drive included, and it updates continuously as you turn the controls. It only shows the curve itself, however:

- it does not take into account the amount (**Fet gr**, **Fet env**), nor **fet amp**: it is the curve for an amount of 1;
- it ignores all the processing of the product (delay, filters, limit) and the **ws** and **FX ws** dosing;
- it is drawn even when the waveshaper is off (**Fet gr** and **Fet env** at 0);
- output values beyond ±1 go off the frame and are drawn on its edge.

The amount of saturation: Fet gr and Fet env

The amount of saturation is computed from the compressor's gain envelope, written e: e = 1 when there is no gain reduction, e = 0.5 at −6 dB, e = 0 at total reduction. The gain reduction, in linear terms, is 1 − e. The two settings dose each of these two quantities:

amount = **Fet gr** × (1 − e) + **Fet env** × e

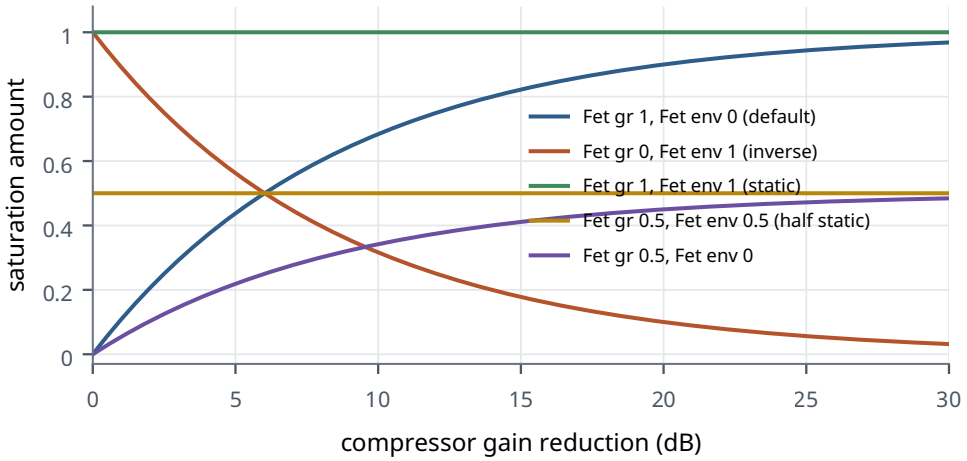


Figure 90 — Amount of saturation as a function of gain reduction, for a few combinations of Fet gr and Fet env.

This gives three ways of working:

- **dynamic waveshaping** (default: Fet gr 1, Fet env 0): the saturation follows the gain reduction. It is zero without compression, 0.5 at 6 dB of reduction, 0.75 at 12 dB, 0.9 at 20 dB. This is the behavior of a real compressor, which saturates more when it compresses;
- **inverted dynamic waveshaping** (Fet gr 0, Fet env 1): the saturation is at its maximum when the compressor isn't compressing, and decreases when it compresses;
- **static waveshaping** (Fet gr and Fet env equal): the amount no longer depends on the compression. At 1 and 1, the curve is applied in full all the time; at 0.5 and 0.5, at half strength.

Intermediate values blend these behaviors. When the channels are unlinked (**Channels Unlink**, chapter 7), the amount uses each channel's envelope, in proportion to the unlinking.

Fet gr

WS tab

Range: 0 to 1 **Default:** 1 **ID:** `fet_gr`

Share of the **gain reduction** in the amount of saturation, from 0 to 1.

- **1** (default): with **Fet env** at 0, the amount equals the linear gain reduction. Without compression, the waveshaper has no effect; the more the compressor reduces the gain, the more the curve is applied.
- **0**: the gain reduction no longer plays a part. If **Fet env** is also at 0, the waveshaper is completely off: the curve stays on screen, but no processing is done and **Monitor Fet** is silent.
- **Intermediate value**: the saturation follows the gain reduction, but more weakly. At 0.5, it tops out at 0.5 at total reduction.

If **Fet gr** and **Fet env** are both 1, the amount is constant and equal to 1: this is static waveshaping.

Fet env

WS tab

Range: 0 to 1 **Default:** 0 **ID:** `fet_env`

Share of the **gain envelope** in the amount of saturation, from 0 to 1. The envelope is 1 without compression and decreases as the compressor reduces the gain: the saturation it controls is therefore at its maximum **when the compressor isn't compressing**.

- **0** (default): the envelope plays no part.
- **1**, with **Fet gr** at 0: "upside-down" saturation. The curve is applied in full at rest and fades away as the compression increases.
- **Equal to Fet gr**: the two shares offset each other and the amount becomes constant (static waveshaping).

This setting makes it possible, for example, to have a saturation that colors the quiet passages and leaves the compressed passages cleaner.

A word from the author

By default, waveshaping is dynamic and based on gain reduction, because that's what makes the most sense: a real compressor saturates more when it compresses. You can then ask it to do the opposite and saturate when the envelope is high, or to saturate constantly, with static waveshaping.

Finally, the amount is multiplied by **fet amp**, the first setting of the F (ws) tab (chapter 16).

Setting the curve by ear

With the default settings, the waveshaper does nothing as long as the compressor isn't compressing. To shape the curve on a lightly compressed passage, temporarily set **Fet gr** and **Fet env** to 1 (static waveshaping), listen with **Monitor Fet**, then go back to dynamic mode.

The 1176 and FET nonlinearity

In 1966, Bill Putnam replaced the tubes of his 175 and 176 compressors with field-effect transistors: the 1176, released in 1967, uses a FET as a voltage-controlled resistor. As its manual acknowledges, FETs are inherently nonlinear components, and Putnam's whole challenge was to keep this one in its linear region to limit distortion; the 1970 "LN" circuit was still working on it. What remains is a color: next to the soft, warm Fairchild, the 1176 brightens the tone and adds bite, and, outside its fastest settings, its subtle FET distortion helps the bass cut through. Even the attack set to **Off** has its use: the unit no longer compresses, but the signal still goes through the electronics, which lets you add the color alone. The waveshaper's dynamic mode follows François's intuition that a real compressor saturates more when it compresses; the static mode recalls the attack set to Off: color without compression. Chapter 2 tells the story of the unit.

♪ In the studio: saturation that grows with compression

On a bass, a snare or a drum bus, leave **Fet gr** at 1, raise **Fet odd** to around 1.5 and set the compressor so that it works mostly on the attacks. Sustained notes and quiet passages stay clean; the hits that trigger gain reduction take on harmonics, at the very moment their level drops. The ear often perceives the compression as denser, more "locked in," without it being any louder. Switch to **Monitor Fet**: you hear the saturation appear and disappear in time with the gain reduction. This is the setting kept by almost all the presets that change the WS tab.

Two presets choose static mode instead. **pa_hg2**, an imitation of a tube saturator, doesn't compress at all (**Ratio** at 1, **Threshold** at 0 dB): with **Fet env** at 1, its curve (**Fet odd** at 0.907, **Fet Thresh** at 0.22, **Fet Knee** at 27.92) applies all the time, and the waveshaper becomes a saturator in its own right. **arturia_diode**, an imitation of an emulation of a diode-bridge compressor, also sets **Fet env** to 1, with **Fet odd** at 1.56: its color no longer depends on the compression.

The drive

Fet drive

WS tab

Range: −24 to 24 dB **Default:** 0 dB **ID:** `fet_input`

Gain applied to the signal before each shaping stage, and removed after it. It ranges from −24 to +24 dB; at **0 dB** (default), it is neutral.

The drive wraps around the stages **Fet even**, **Fet odd**, **Fet Thresh**/**Fet Knee**, **Fet sin *** and **Fet Precision**: the signal is multiplied by the drive, shaped, then divided by the drive. It acts neither on **Fet *** nor on the DC bias (**Fet sin x** at 0).

So the drive does not raise the level. It acts like a **magnifying glass** centered on the origin of the curve: with +6 dB, the signal covers a portion of each stage twice as large, and the result is brought back to the original scale. Small signals keep the same slope, but the shaping starts earlier. In practice:

- **Fet even** produces more second harmonic (the squared term is proportional to the drive);
- **Fet odd** shapes only levels below 1/drive (see its card);
- the effective threshold of **Fet Thresh** is divided by the drive: a threshold of 0.5 with +6 dB of drive clips at 0.25;
- the sine of **Fet sin x** oscillates faster over the input range;
- the steps of **Fet Precision** are finer and the dead zone around zero is narrower.

A negative drive does the opposite: it pushes the shaping away, so that it affects only high levels, or none at all.

The drive alone does not turn on the waveshaper: at least one stage has to be engaged.

A word from the author

François points out that the drive is a true drive in decibels, whose effect you can see directly on the displayed curve.

Fet even: the second harmonic

Fet even

WS tab

Range: −1 to 1 **Default:** 0 **ID:** `fet_pair`

Blends the signal and its square. The setting p ranges from −1 to +1; at **0** (default), the stage is inactive. With no drive, the output of the stage is:

output = $p \times x^2 + (1 - p) \times x$

then it is clamped between −1 and +1.

The square of a sine wave contains only two things: a DC component and the second harmonic. **Fet even** therefore creates the 2nd harmonic alone, without the higher even harmonics (4, 6, 8…). Since the curve is not symmetrical, it also creates a **DC component**, and it **attenuates the fundamental** in the proportion 1 − p.

- **Between 0 and 1**: the second harmonic and the DC increase, and the fundamental decreases. At 1, the curve is a parabola: the output of the stage no longer contains the original signal at all, only its square.
- **Between −1 and 0**: the second harmonic has inverted polarity, and the fundamental is **reinforced** (× (1 + |p|)). High negative levels quickly hit the −1 bound, which adds other harmonics.

Fet drive increases the share of the square: at drive d, the squared term becomes $p \times d \times x^2$.

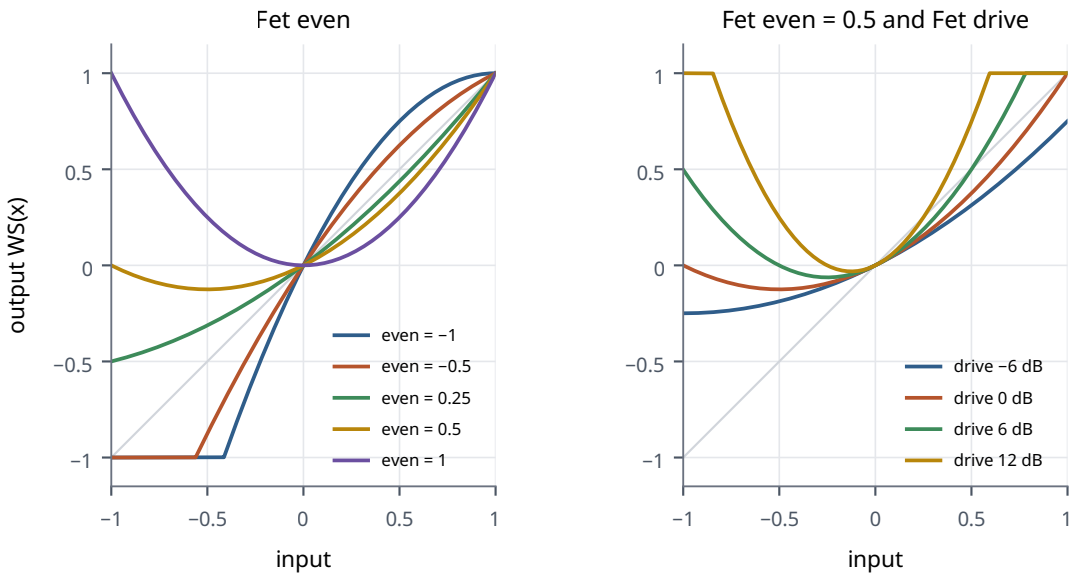


Figure 91 — Curves of Fet even alone (left) and effect of the drive on Fet even = 0.5 (right).

A word from the author

The even generator produces only the second harmonic. A formula that brought out harmonics 4, 6 and 8 would be much more complicated, and it isn't necessary: just add a little **Fet odd**. The odd curve, applied to a signal that has been made asymmetrical, brings out the other even harmonics.

To be confirmed by François

Fet even and the DC bias (**Fet sin x** at 0) add a DC component to the saturation product, and no filter in the chain seems to remove it afterward. Since this product is dosed by the gain reduction, this DC varies with the compression. Is this intended, or should users be advised to remove it with the low band of F (ws)?

Fet odd: the odd harmonics

Fet odd

WS tab

Range: 0.125 to 8 **Default:** 1 **ID:** `fet_impair`

Exponent k of a symmetrical curve that creates odd harmonics. It ranges from 0.125 to 8; at **1** (default), the stage is inactive (values between 0.999 and 1.001 are snapped to 1). With no drive, for an input x between 0 and 1:

output = 1 − (1 − x)^k

and symmetrically for negative values.

- **k > 1**: the curve rises quickly, then flattens out as it approaches ±1. This is soft saturation. Its slope at the origin is k: low levels are **amplified** k times (×2, or +6 dB, for k = 2). The saturated signal is therefore louder than the original, which the product reflects.
- **k < 1**: the curve starts slowly, then rises steeply as it approaches ±1. Low levels are **attenuated** (slope k at the origin) and the peaks are preserved: this is an expansive shape.

The output is clamped between −1 and +1.

With a positive **Fet drive**, the stage shapes only inputs below 1/drive in absolute value; beyond that, it lets the signal through unchanged. The displayed curve then shows a plateau, then re-joins the diagonal with a kink. With a negative drive, the whole range is shaped; for k > 1, the amplified output then hits the ±1 bound before the end of the range: the soft saturation ends in hard clipping.

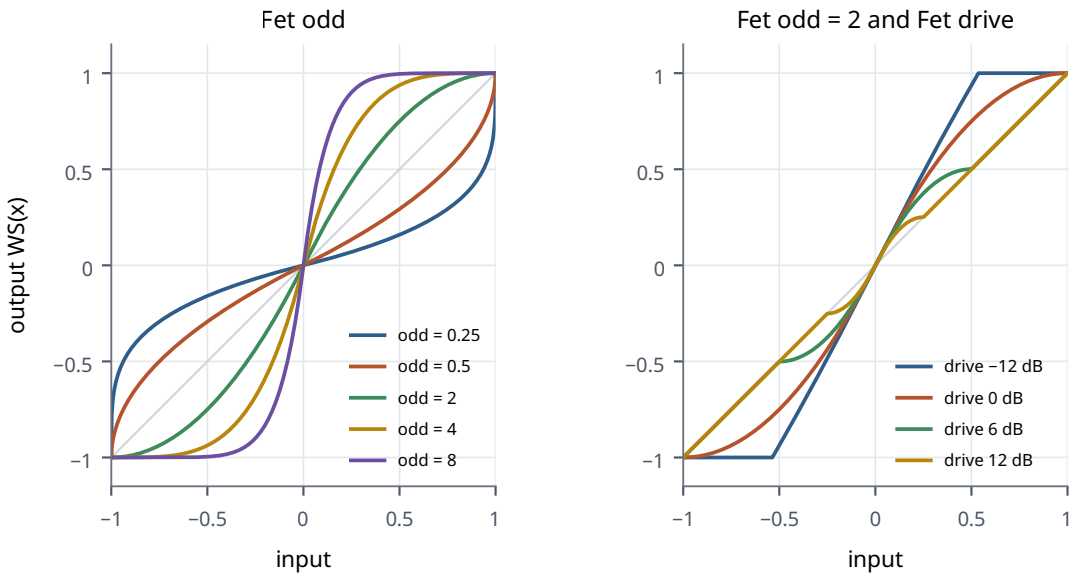


Figure 92 — Curves of Fet odd alone (left) and effect of the drive on Fet odd = 2 (right).

To be confirmed by François

With a positive **Fet drive**, **Fet odd** distorts only levels below 1/drive: beyond that, the curve returns to the diagonal, with a break in slope visible on the graph (for example above 0.5 with +6 dB of drive). Is this the intended behavior?

Fet Thresh and Fet Knee: the clipper

Stage 3 is a clipper: below the threshold, the signal passes through untouched; above it, it is capped at the threshold. The knee rounds off the transition.

Fet Thresh

WS tab

Range: 0 to 1 **Default:** 1 **ID:** `fet_seuil`

Threshold of the clipper, as a **linear** amplitude from 0 to 1. At **1** (default), there is no clipping (as long as **Fet Knee** is at 0).

The threshold is read relative to full internal level: 0.5 corresponds to −6 dBFS, 0.25 to −12 dBFS, 0.1 to −20 dBFS. Above the threshold, the output is constant, equal to the threshold: without a knee, this is hard clipping.

Fet drive divides the effective threshold: with a threshold of 0.5 and +6 dB of drive, the clipping happens at 0.25, at the input as well as at the output.

Fet Knee

WS tab

Range: 0 to 240 **Default:** 0 **ID:** `fet_genou`

Width of the clipper's knee. Although its knob looks like the one for the main knee (in decibels), this width is **linear**: it equals **Fet Knee** / 120, from 0 to 2. At **0** (default), the clipping is hard.

The knee is centered on the threshold: the shaping starts at threshold − width/2 and the ceiling is reached at threshold + width/2. In between, the curve is a parabolic arc that joins the diagonal to the ceiling without a kink. For example, with a threshold of 0.5 and **Fet Knee** at 60 (width 0.5), the rounding runs from 0.25 to 0.75.

A non-zero knee turns on the stage even if **Fet Thresh** is 1: it then rounds off the peaks from 1 − width/2 up.

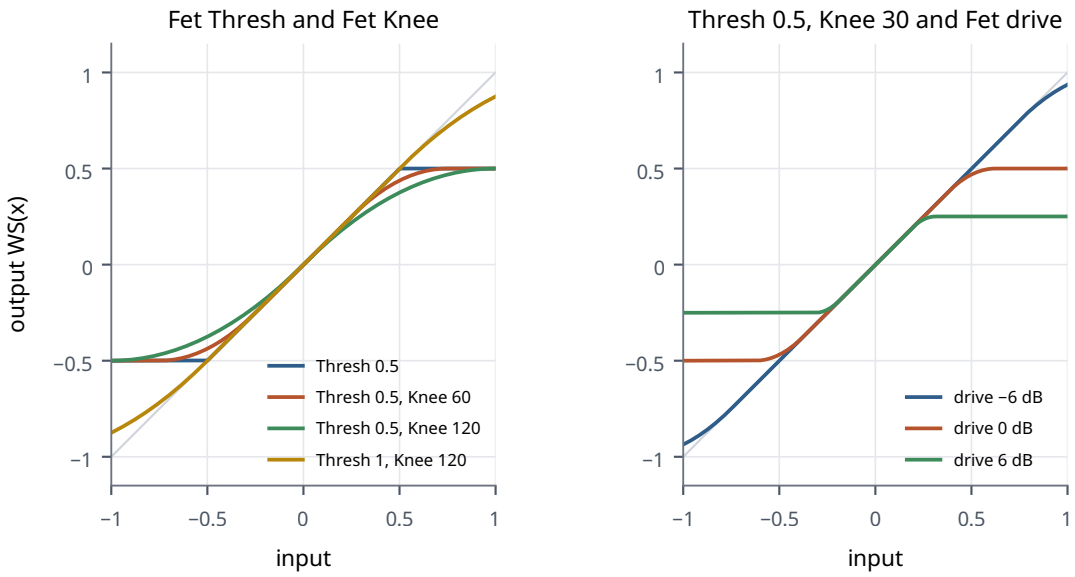


Figure 93 — Clipper curves: threshold and knee (left), effect of the drive on a threshold of 0.5 with a knee of 30 (right).

A word from the author

Threshold and knee are both linear, and all the stages can be stacked: you can clip after creating even and odd harmonics.

To be confirmed by François

When half the knee width exceeds the threshold (**Fet Knee** / 240 > **Fet Thresh**, for example a threshold of 0.2 with a knee of 120), the knee zone includes zero and the curve shows a small jump at the zero crossing. Is this intended?

Fet *: the multiplier

Fet * WS tab

Range: −1 to 1 **Default:** 1 **ID:** `fet_rebond_min`

Multiplies the signal at the output of stage 3, without going through the drive. It ranges from −1 to +1; at **1** (default), it is neutral.

This simple setting has strong effects, because the waveshaper feeds back a **difference**:

- **Between 0 and 1**: the output of the curve is attenuated; the product subtracts part of the signal from the signal itself. The waveshaper becomes an extra gain control, driven by the amount of saturation.
- **0**: the curve is zero and the product is exactly −x × amount. With the other settings at their defaults, the target is then multiplied by the compressor's envelope **before** being multiplied a second time by the compressor's gain: in decibels, the gain reduction is **doubled**.
- **Negative**: the curve inverts the polarity. At −1, with the default settings, the target equals x × (2e − 1): untouched without compression, zero at 6 dB of gain reduction, then with inverted polarity beyond that.

Fet * also acts as a switch: when it is exactly **0** and **Fet Precision** is above 0, the waveshaper switches to **spreader** mode (see below). In this mode, the multiplication is not performed.

> **Fet * at 0 and Fet Precision**
>
> With **Fet *** at 0, raising **Fet Precision** above 0 completely changes the behavior: you go from a gain control to the spreader. Keep this in mind if the sound changes abruptly when you touch either of these two settings.

Fet sin * and Fet sin x: sine and DC bias

Stage 5 adds to the signal a sine function of the signal itself: output = x + A × sin(x × r^r), where A is **Fet sin *** and r is **Fet sin x** (with no drive). When **Fet sin x** is 0, the stage simply adds the constant A: this is the **DC bias**.

The frequency of the sine varies enormously with **Fet sin x**, because the setting is raised to its own power:

Fet sin x	0.25	0.5	1	2	3	4	8
r^r	0.71	0.71	1	4	27	256	16,777,216

The value r^r goes through a minimum (about 0.69) around r = 0.37: between 0 and 1, the setting is not monotonic, but the effect stays gentle. From 3 up, the sine makes several oscillations over the input range and creates high harmonics; from 4 up, it varies so fast from one sample to the next that its result is practically random: the waveshaper adds a **noise** whose amplitude follows the signal.

Fet sin *

WS tab

Range: −1 to 1 **Default:** 0 **ID:** `fet_rebond_max`

Amplitude A of the sine added by stage 5, or value of the DC bias if **Fet sin x** is 0. It ranges from −1 to +1; at **0** (default), the stage is inactive.

- With **Fet sin x** other than 0, the signal receives A × sin(x × r^r). A negative value inverts the sine. For **Fet sin x** less than or equal to 1, the effect resembles a slight amplification (A > 0) or attenuation (A < 0) of low levels, with a gentle curvature; for higher values, it creates ripples, then noise.
- With **Fet sin x** at 0, the stage adds the constant A to the signal: this is the **DC bias**. It is not multiplied by the drive.

The DC bias is added **after** the shaping stages: on its own, it creates no harmonics, only a DC offset in the product (which the amount of saturation makes vary with the compression). The only stage it distorts is **Fet Precision**, which follows it: the quantization then becomes asymmetrical and creates even harmonics.

Fet sin x

WS tab

Range: 0 to 8 **Default:** 0 **ID:** `fet_rebond_ratio`

Frequency r of the stage 5 sine. It ranges from 0 to 8; the argument of the sine is the input multiplied by r^r. At **0** (default), the stage adds a constant (the DC bias) instead of a sine.

- **Up to 1**: a slow variation, which gently bends the characteristic.
- **2 to 3**: ripples over the input range (about one across the whole range at 2, eight at 3); high odd harmonics.
- **4 and above**: a pseudo-random result, a noise that follows the amplitude of the signal.

Fet drive speeds up the sine, since the signal is multiplied by the drive before the stage. The knob is progressive: the value 1 is at about 60% of its travel, the value 2 at about 71%.

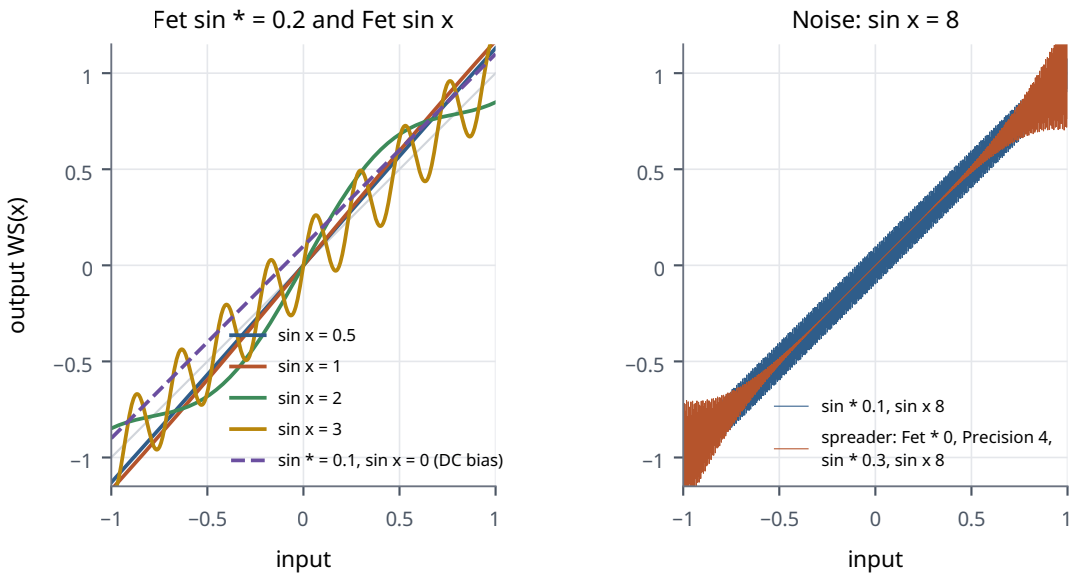


Figure 94 — Sine and DC bias (left); right, the noise obtained with Fet sin x = 8, and the same noise in spreader mode, confined to high levels.

A word from the author

François added the sine because he wanted a saturation close to white noise: set very high, it gives a grainy saturation. Its drawback is that it is everywhere, even on low levels; that's what he solved with the spreader mode.

Fet Precision: bit crusher and spreader

Fet Precision

WS tab

Range: 0 to 256 **Default:** 0 **ID:** `fet_precision`

Depending on the setting of **Fet ***, this setting has two roles.

Bit crusher (the general case, **Fet *** other than 0). The signal is quantized in steps of 1/P, where P is the value of the setting, rounding toward zero. For example, at 4, the only possible values are 0, ±0.25, ±0.5, ±0.75 and ±1. At **0** (default), the stage is inactive.

- The number of levels over the −1 to +1 range is 2P + 1: a precision of 128 corresponds roughly to 8-bit quantization, and 8 to 4 bits.
- Rounding toward zero creates a **dead zone**: any input smaller than 1/P in absolute value becomes 0. Since the waveshaper feeds back a difference, low levels are then removed from the signal, in proportion to the amount of saturation. Below 1, the curve is zero over the whole range (with no drive): the effect is that of **Fet *** at 0.
- The quantization is symmetrical: it creates only odd harmonics, unless an asymmetrical stage comes before it (**Fet even** or the DC bias).
- **Fet drive** makes the steps finer: with a drive d, they are 1/(P × d) at the input.

Spreader (**Fet *** exactly at 0). Quantization is no longer performed. **Fet Precision** becomes the exponent of a weighting of the stage 5 sine: its amplitude is no longer A but A × |x|^P. The sine, or the noise, now touches only high levels, and leaves the center of the signal clean. The higher the value, the more the noise is confined to the peaks. Without a sine (**Fet sin *** at 0), this mode adds nothing to stages 1 to 3; with **Fet sin x** at 0, stage 5 goes back to being a simple DC bias.

The knob is extremely progressive: the value 1 sits at about 71% of its travel, the value 16 at about 84%. Typing the value in is often easier.

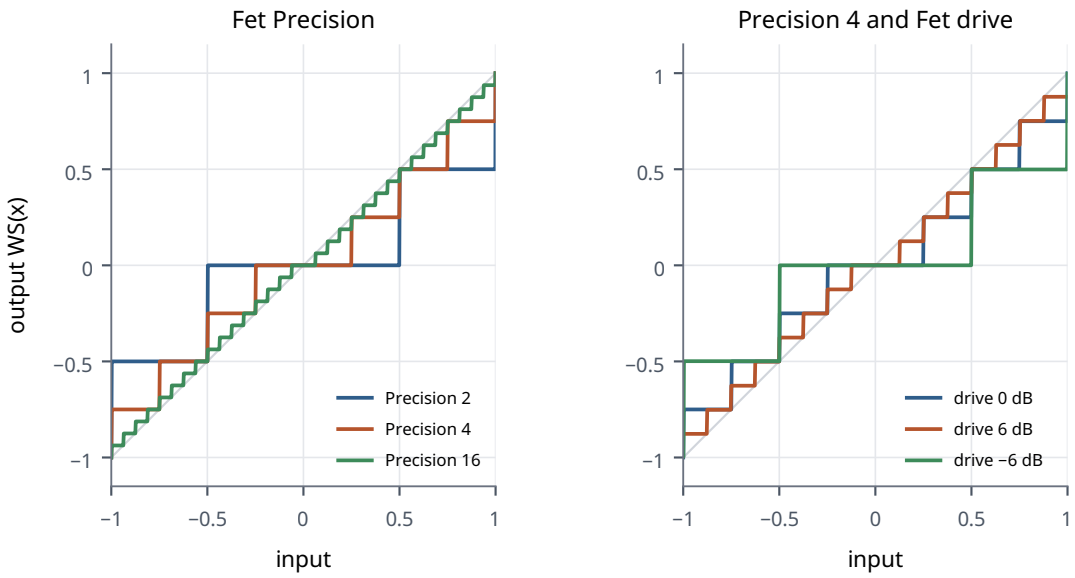


Figure 95 — Quantization by Fet Precision (left) and effect of the drive on a precision of 4 (right).

A word from the author

The problem with the sine is that it's everywhere. François wanted the middle of the signal to stay normal and the grain to appear only at the edges: hence the idea of setting **Fet *** to 0 and engaging the precision, which then becomes a "spreader." This recovers the character of certain saturators (François mentions germanium) whose noise appears only when the signal is loud, which is useful for simulating the saturation of certain compressors.

To be confirmed by François

In spreader mode, the amplitude of the sine is multiplied by $|x \times \text{drive}|$ raised to the power **Fet Precision**. With a positive drive and a high precision (for example +24 dB and 40), this factor exceeds the range of floating-point arithmetic and the result becomes infinite. Should this computation be clamped?

♪ In the studio: grain only on the peaks

This is the tone François was after to imitate the saturation of certain compressors: white noise that appears only when the signal is loud, and a center that stays clean. Set **Fet *** to 0, **Fet sin *** to around 0.2, **Fet sin x** to 8 and **Fet Precision** between 2 and 8. On a snare or a rhythm guitar, the grain settles on the attacks and the peaks, and only when the compressor is reducing gain: the saturation seems to switch on with the hit, then switch off. Raise **Fet Precision** to push it toward the loudest peaks, and listen to it alone with **Monitor Fet**.

Two presets rely on this mode. **waves_76**, an imitation of a 1176 emulation, sets **Fet *** to 0, **Fet Precision** to 2.677, **Fet sin *** to 1 and **Fet sin x** to 8, adds a rounded clipper (**Fet Thresh** at 0.74, **Fet Knee** at 240) and keeps, in the F (ws) tab, only the high part of the product, above about 5.5 kHz. **rockruepel_comptwo_drums**, whose name points to drums, uses **Fet *** at 0, **Fet Precision** at 7.804, **Fet sin *** at 1 and **Fet sin x** at 4, then heavily reshapes the product in the F (ws) tab.

Which harmonics?

The next figure shows the spectrum of the curve's output for a sine wave with an amplitude of 0.7, in static waveshaping (amount 1), with one stage at a time, then a combination.

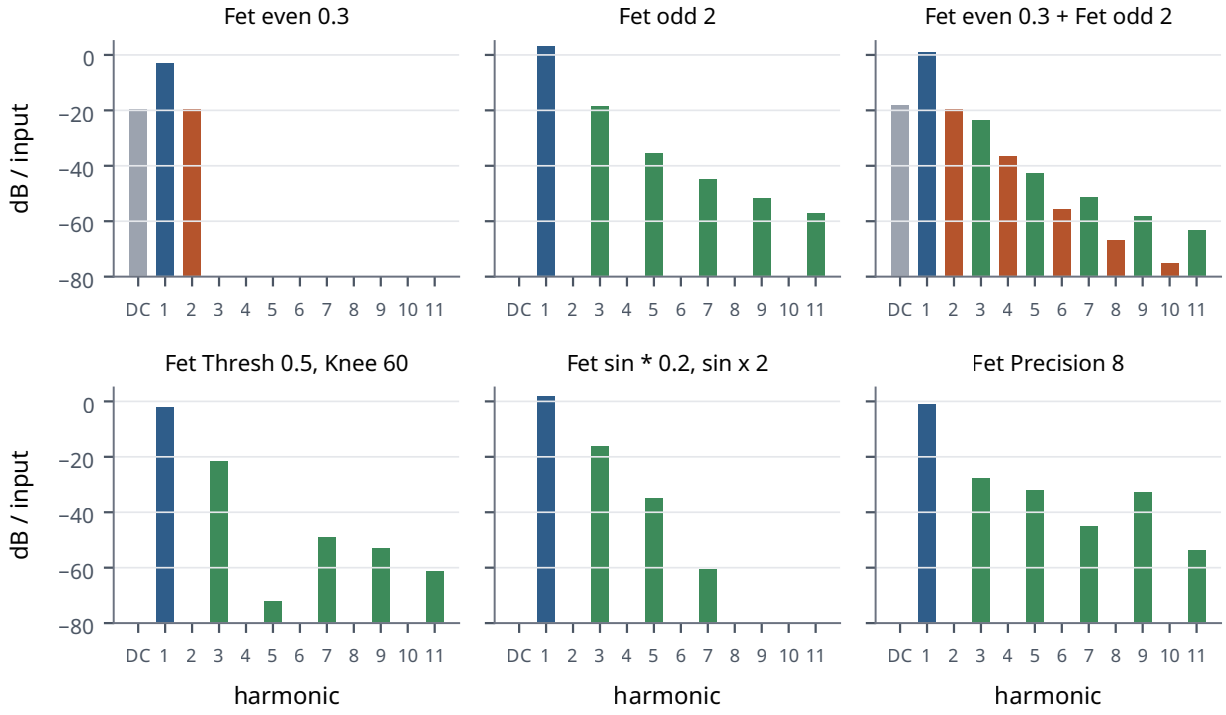


Figure 96 — Spectrum of the waveshaper's output for a sine wave with an amplitude of 0.7, in dB relative to the input amplitude. Blue: fundamental; orange: even harmonics; green: odd harmonics; gray: DC component.

Key takeaways:

- the symmetrical stages (**Fet odd**, the clipper, the sine, the precision) create only **odd** harmonics;
- **Fet even** creates the DC component and the 2nd harmonic alone, and attenuates the fundamental;
- combining **Fet even** with a symmetrical stage brings out **all** the harmonics, even ones included;
- **Fet odd** above 1 and the sine also reinforce the fundamental: the saturated signal is louder than the original.

These spectra are those of the full curve. In default operation, the product is multiplied by the gain reduction, then filtered and limited by the F (ws) tab: the level of the harmonics therefore varies with the compression.

Dist 2, Dist 3: choosing your harmonics

In 1995, the first Empirical Labs Distressors went on sale. Designed by Dave Derr, an Eventide veteran, this VCA compressor borrows from several classic units, from the 1176 to the LA-2A. Two of its distortion modes speak directly to the waveshaper. **Dist 2** brings a "class A" warmth, mostly 2nd harmonic, when the unit is compressing; its manual points out that tube distortion is known for its 2nd harmonic. **Dist 3** adds the 3rd harmonic to the second and progressively flattens the top and bottom of the waveform, like magnetic tape. The manual even offers a recipe for giving a digital recording the character of tape: a 2:1 ratio, Dist 3, and 1 to 3 dB of compression. Other units also separate even and odd: the gain element of the elysia alpha compressor enriches the signal with 2nd and 3rd harmonics, and the **Colour** section of the Bettermaker Mastering Limiter has two harmonic generators, one odd, the other even. In Comprendre, **Fet even** plays the role of the 2nd-harmonic generator, **Fet odd** that of the odd one, and the clipper flattens the top and bottom of the waveform; the three combine freely.

♪ In the studio: the second, the odds, or both

Start by listening to each stage alone, in static waveshaping and with **Monitor Fet**. **Fet even** between 0.1 and 0.2 creates only one harmonic, at the octave: the ear often perceives it as added body rather than as distortion, which makes it a good candidate for a vocal, a bass or a mix bus. **Fet odd** above 1 brings in the odd harmonics: the sound gains edge and presence, and the saturation becomes more recognizable. The two together give all the harmonics; finally, add a little clipper to flatten the peaks.

The presets show the amounts François chose. In the Distressor family, **kiive_xstressor** combines **Fet even** at 0.168 and **Fet odd** at 1.5, with a clipper (**Fet Thresh** at 0.4, **Fet Knee** at 50), all fed back sparingly (**fet amp** at 0.0625); **slate_fgstress** pushes **Fet odd** to 1.299 with +6 dB of **Fet drive**. On the 1176 side, **pulsar_1178** relies on the second harmonic (**Fet even** at 0.2, a drive of +1 dB). Many presets settle for a touch: **Fet odd** at 1.269 for **native_vc2a** and **spl_iron**, at 1.299 for **blackrooster_omnitec436**, at 1.706 with **Fet even** at 0.072 for **tr_670**. **ao_dbcomp** combines **Fet odd** at 1.24 and **Fet even** at 0.088, then attenuates, in the F (ws) tab, the part of the saturation above about 1 kHz: the color stays in the lows and mids. At the other extreme, **kiive_filk2_saturation**, from the saturator family, goes as far as **Fet odd** at 8 and **Fet even** at −1, with a clipper at 0.6 and +4.5 dB of drive.

The product delay

These two settings control the feedback delay specific to the waveshaper's product. How it works, which is common to all three saturators, is described in detail in chapter 16: it is a comb filter, not a simple echo.

Fet Delai Secs

WS tab

Range: 0 to 2 s **Default:** 0 s **ID:** `fet_delai_nb`

Length of the feedback delay applied to the waveshaper's saturation product, from 0 to 2 s. At **0** (default), the delay is a single internal sample.

This setting has **no effect as long as Fet Delai FB is 0**. The actual delay is rounded down to the internal sample, plus one sample. The original signal is never delayed: only the product goes through the delay.

- **0 s** (with FB other than 0): a one-pole filter, low-pass if FB is positive; its frequency depends on the oversampling factor (chapter 16).
- **A few milliseconds**: the product concentrates on a series of frequencies spaced 1/delay apart, which colors the saturation.
- **Several tens of milliseconds or more**: you hear repeats of the product alone.

The knob is progressive: the middle of its travel is 0.125 s, the quarter mark about 8 ms. The delay acts separately on the target's waveshaper and on the detection's, each with its own memory.

Fet Delai FB

WS tab

Range: −64 to 64 **Default:** 0 **ID:** `fet_delai_fb`

Feedback of the product delay, from −64 to +64. At **0** (default), the delay is off.

The incoming product is multiplied by 1/(1 + |FB|), and the delayed output is fed back with the coefficient 1 − 1/(1 + |FB|), with the sign of FB:

- **positive**: repeats with the same sign; the frequencies that are multiples of 1/delay are preserved, the others attenuated;
- **negative**: repeats with alternating signs; the preserved frequencies are shifted by half an interval, and the low end is attenuated.

The larger |FB|, the longer the tail and the more the product is attenuated outside the preserved frequencies: at FB = 1, each repeat is half the previous one; at FB = 4, 80%; at FB = 64, more than 98%. The knob is symmetrical and tightly packed around 0: at three quarters of its travel, the value is 4.

> **A word from the author**
>
> This delay is primarily meant to multiply the saturation with very short times, like a new excitation of the harmonics. You can also use it as a creative delay that acts only on the saturation product.

♪ In the studio: the waveshaper for sound design

On a drum loop or a synth, **Fet Precision** between 8 and 32 in static waveshaping gives you a bit crusher: the quantization grain blends into the sound, and the ends of notes, quieter than the first step, get trimmed. In dynamic mode, this grain appears only with compression: a lo-fi texture that breathes with the hits. In inverted mode (**Fet gr** at 0, **Fet env** at 1), it's the reverse: the quiet passages and note tails degrade, while the hits stay clean.

Fet sin x around 3, with a little **Fet sin ***, adds ripples that give rise to high harmonics, which the ear often hears as metallic. Finally, the product delay, set to a few milliseconds with a **Fet Delai FB** of a few units, makes the saturation "sing" on a series of frequencies, without delaying the original sound; with several tens of milliseconds, only the saturation repeats, like an echo of grain. Always listen to the result with **Monitor Fet** before blending it into the sound.

Listening to the product: Monitor Fet

Monitor Fet WS tab

Range: 0 to 1 **Default:** 0 **ID:** `delta_fet`

Replaces the output with the **saturation product** of the target's waveshaper, so you can listen to it alone. It is a switch: 0 (default) or 1.

You hear the product after all the processing of the F (ws) tab (delay, filters, limit) and after the **ws** dosing of the OUTPUT tab, but **without** the compressor's gain, without DEQ 2 and without the original signal. Like the other monitors (chapter 7), it still goes through the Headroom return, the Makeup and the Output Db, but not through the post-clipper, and it is mutually exclusive with the other monitors.

If the target's waveshaper is inactive, **Monitor Fet** gives silence. This is also the case, with the default settings, as long as the compressor isn't reducing gain: the amount of saturation is then zero.

A word from the author

Listening to the saturation alone is very handy for knowing exactly what the waveshaper is doing: it's the best way to set the curve and the processing of the product.

Practical workflows

A FET compressor saturation. Leave **Fet gr** at 1 and **Fet env** at 0. Raise **Fet odd** to between 1.5 and 3, and adjust **Fet drive** if needed. Listen to the product with **Monitor Fet** during a passage that compresses, then go back to normal listening. If the level climbs too much when the compression is heavy (which is the case with **Fet odd** above 1, since it amplifies), lower **fet dst limit** or **fet amp** in the F (ws) tab.

Just a second harmonic. Set **Fet even** between 0.1 and 0.3. To remove the DC component it creates, set **fet low herz** to around 20 Hz, **fet low zdf->svf** to 0 and **fet low mix** to −1 in the F (ws) tab. To also get harmonics 4, 6 and 8, add a little **Fet odd**.

Noisy grain only on the peaks (spreader). Set **Fet *** to 0, **Fet Precision** between 2 and 8, **Fet sin *** between 0.1 and 0.3 and **Fet sin x** to 8. The noise appears only on high levels, and only when the compressor is reducing gain. Increase **Fet Precision** to confine it more to the peaks.

A bit crusher that follows the compression. Set **Fet Precision** between 8 and 32: the quantization grain blends into the signal as the compressor works. In static mode (**Fet gr** and **Fet env** at 1), it is a permanent bit crusher.

Extra gain reduction. With **Fet *** between 0 and 1, the waveshaper subtracts part of the signal from the signal itself, in proportion to the gain reduction. At 0, the gain reduction is doubled in decibels. This is not saturation, but a way to make the compression deeper without touching the transfer curve.

Saturating the detection. The **FX ws** setting of the INPUT tab (chapter 14) adds the waveshaper's product to the detection signal. The compressor then reacts to a signal richer in harmonics, without the saturation being heard. The detection's waveshaper uses the same settings from the WS and F (ws) tabs.

Starting ideas

- **Bass. Fet odd** between 1.3 and 1.8, **Fet gr** at its default, a fast attack: the bass gains presence on the notes that trigger compression. For comparison, load a preset from the FET family (chapter 25), such as **waves_76** or **pulsar_1178**, and listen to its WS tab with **Monitor Fet**.
- **Vocals. Fet even** between 0.05 and 0.15, possibly with a reduced **fet amp**: a discreet thickness that appears only when the voice pushes.
- **Drum bus.** Grain on the peaks (spreader mode), or **Fet odd** with a few decibels of **Fet drive**, as in **slate_fgstress**. The **rockruepel_comptwo_drums** preset is a good starting point.
- **Mixing and mastering.** A touch of **Fet odd** (around 1.25, like **spl_iron** and **native_vc2a**), then, in the F (ws) tab, keep only the part of the saturation you're interested in, the way **ao_dbcomp** does.
- **Saturation without compression. Ratio** at 1, **Fet env** at 1: the waveshaper becomes a static saturator, as in **pa_hg2**.
- **Sound design.** A bit crusher in static or inverted mode, a high **Fet sin x**, the product delay: chapter 26 offers more starting points depending on your line of work.
- **Without hearing it.** The waveshaper can also serve purely as a modulation source: **pa_elysia** sets the **ws** dosing of the OUTPUT tab to 0 and uses its clipper (**Fet Thresh** at 0.11, **Fet Knee** at 32.8) as a **ws mod** source (at −1) to vary the ratio (**mod ratio** at 1.75, chapter 23).

CHAPTER 18

The Dynamic EQs

Compression doesn't just change the level: it changes the tone as well. When the compressor clamps down on kick drum hits, the body of the drums can flatten out; when it tightens up a vocal, the low end drains away or the harshness comes forward. Comprendre's dynamic EQs tackle this problem in an original way: instead of listening to the signal with a detector of their own, they follow the **movement of the compressor itself**. Each band moves exactly in step with the gain reduction. That lets you make a drum kit **breathe**, with a bump of body that appears on every hit, give back to the low end what compression takes away, soften the top end when the compressor tightens up, feed the saturation with a single region of the spectrum, or make the compressor more or less sensitive to certain frequencies.

These EQs are among the most used modules in the bundled presets: 24 of the 80 presets change at least one of the two, most often for a subtle correction that goes along with the compression, sometimes as a simple fixed EQ, to imitate the color of a saturator or a transformer. They will speak to mixing engineers and mastering engineers alike, and to the producer who wants heavy compression to keep its body.

Comprendre contains two three-band dynamic EQs. They have no detector of their own: their bands follow the **compressor's gain envelope**. A band can thus rise when the compressor reduces gain and fall back when it releases, or the reverse. You use them to compensate for what compression takes away (low end, for example), to accentuate what it does, or to make the saturation and the detection frequency-dependent.

The two EQs are identical. Only their position differs:

- **DEQ** (tab **DEQ**, to the left of **WS**) sits **before** the waveshaper;
- **DEQ 2** (its tab is also labeled **DEQ**, to the right of **F (ws)**) sits **after** the waveshaper.



Figure 97 — The DEQ tab, with an example setting: the low band at 300 Hz, resonance 2, min 1, max 0. Modified labels and the tab name turn yellow.



Figure 98 — The DEQ 2 tab: same settings, with the tab labeled “DEQ” to the right of F (ws).

How it works

Three bands added to the signal

A Comprendre dynamic EQ runs the signal through three filters in parallel, the same ones used for the saturation product (chapter 16):

- a **low** band: a low-pass, with an adjustable blend of a one-pole filter (ZDF) and a two-pole filter with resonance (SVF);
- a **mid** band: an SVF band-pass, whose resonance sets the width;
- a **high** band: a high-pass, with an adjustable ZDF/SVF blend.

Each band is then **added** to the signal, multiplied by a coefficient:

output = signal + c_low × low band + c_mid × mid band + c_high × high band

What makes the EQ dynamic is that each coefficient varies with the compressor’s gain envelope, written e:

c = **min** + (**max** − **min**) × e

The envelope is 1 when the compressor isn’t reducing gain and 0 at full reduction. So:

- **max** is the band’s coefficient **at rest**, when the compressor isn’t compressing;
- **min** is the band’s coefficient at **full gain reduction**;
- in between, the coefficient varies in a straight line with the envelope.

A band is off as long as its frequency is 0. The EQ does nothing as long as no band has both a nonzero frequency and a **min** or **max** other than 0.

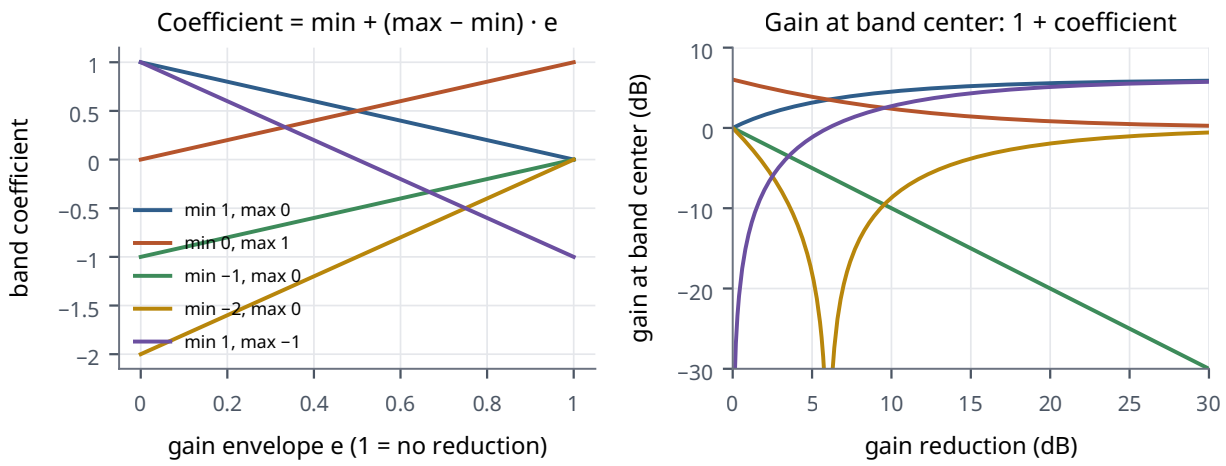


Figure 99 — Left: a band's coefficient as a function of the envelope, for a few min/max pairs. Right: the resulting gain at the heart of the band (where the filter passes the signal with a gain of 1), as a function of the compressor's gain reduction.

> **Halfway there at 6 dB**
>
> The coefficient varies linearly with the envelope, not with the gain reduction in decibels. At 6 dB of reduction, the envelope is 0.5, so the band is already halfway between **max** and **min**. At 12 dB, it has covered three quarters of the way; at 20 dB, 90%. Most of the movement therefore happens in the first few decibels of compression.

A coefficient, not a gain in decibels

The coefficient tells you how many times the filtered band is added to the signal. At the heart of the band (the low end for the low band, the center for a mid band with a resonance of 1, the top end for the high band), the resulting gain is 1 + c:

Coefficient	Gain at the heart of the band
3	+12 dB
1	+6 dB
0.41	+3 dB
0	0 dB (the band has no effect)
−0.5	−6 dB
−1	the band is removed
−2	the band comes back, with inverted polarity

With a ZDF filter, a coefficient of −1 removes the band exactly: the signal minus its ZDF low-pass is its ZDF high-pass. With the mid band-pass at a resonance of 1, a coefficient of −1 notches out the center completely. With the SVF filters, phase shifts create small bumps or dips around the cutoff frequency; the blend figure in chapter 16 shows some examples.

For the mid band, remember that the band-pass gain at the center equals its resonance: with a resonance of 4, a coefficient of 1 gives 1 + 4 = 5 times the signal at the center, or +14 dB.

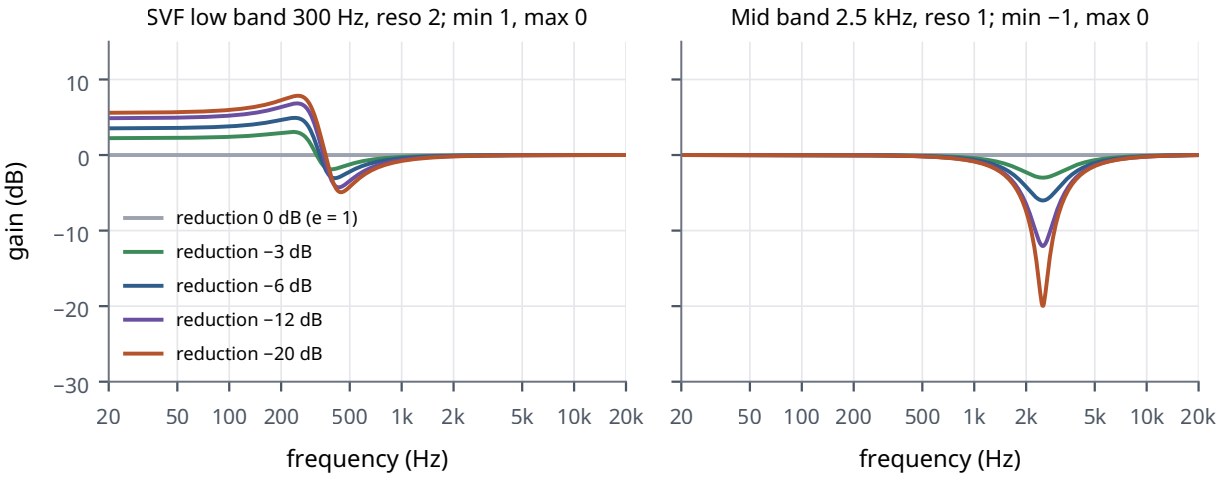


Figure 100 — Frequency response of a dynamic EQ for several amounts of gain reduction, at 48 kHz. Left: more low end when the compressor reduces gain. Right: a dip in the mids that deepens with compression.

In the first example, the low band's resonance creates a small bump just below the cutoff frequency, then a dip just above it: that's the effect of the filter's phase shift when it is added to the signal.

> **Going further**
>
> For a given envelope e, the frequency response is H(f) = 1 + Σ c_b(e) × H_b(f), where H_b is the complex response of the filter of band b (the sums are complex: phases matter). The filters are JUCE's TPT filters (chapter 16). For example, at 44.1 kHz, an SVF low band at 300 Hz (resonance 0.707) with a coefficient of 1 gives +6 dB at 50 Hz, +1.8 dB at 300 Hz and −0.7 dB at 1 kHz.

Safety clipping

As soon as at least one band has a nonzero frequency, the signal coming out of the EQ is **clipped** at ±1, that is, at full internal level (0 dBFS after the Headroom). This hard clipping applies to the whole signal, and it happens before the compressor's gain. With a positive Headroom or a signal already close to 0 dBFS, a band that rises by a few decibels can therefore cause clipping.

> **To be confirmed by François**
>
> As soon as a band of a dynamic EQ is active, the signal is clipped at ±1 (internal 0 dBFS), before the compressor's gain. With a positive Headroom or a band that rises a lot, this clipping is audible. Is it intended as a safety measure?

Raising or lowering a band with compression

The direction of the effect depends on the order of **min** and **max**:

- **min** > **max**: the band **rises** with compression. With min 1 and max 0, the band is neutral at rest and rises up to +6 dB at full reduction (+3.5 dB at 6 dB of reduction). This is the most natural setting for making up for what compression takes away.
- **max** > **min**: the band **drops** with compression. With min 0 and max 1, the band is at +6 dB at rest and returns to 0 dB when the compressor reduces gain heavily.
- **negative values**: the band is cut instead of boosted. With min −1 and max 0, the band is neutral at rest and disappears at full reduction (−6 dB at 6 dB of reduction).
- **min** = **max**: the coefficient is constant; the band is **static**. The DEQ becomes a fixed EQ, useful for correcting the target or the detection.

> **Don't go past −1 by accident**
>
> A coefficient below −1 doesn't remove the band "even more": it brings it back with inverted polarity. With min −2 and max 0, the band dips until it disappears at 6 dB of reduction, then **comes back up** beyond that. To cut a band with compression, keep the coefficient it reaches between −1 and 0.

Yet the **min** and **max** controls range from −8 to +8. That's deliberate: if the compression is moderate, the envelope never gets anywhere near 0, and you need stronger values to get the effect you want. For example, if gain reduction never exceeds 6 dB (envelope 0.5), a **min** of −2 with a **max** of 0 gives exactly −1 at the peak of compression.

> **A word from the author**
>
> **max** says where the amplitude of the filtered signal should sit when the envelope is at 1, and **min** where it should sit when the envelope is at 0. The most natural move is to set **min** to 1: the more gain reduction there is, the more the band is boosted, and with no gain reduction, nothing happens. You can do the opposite and subtract the band when the compressor is working, but that coefficient must not go past −1, or the band starts rising again. The range goes up to ±8 in case the envelope doesn't go down far enough.

> **♪ In the studio: drums that breathe**
>
> This is the example François plays in his video on the dynamic EQs: on compressed drums, a low band around 300 Hz, with a little resonance, **Deq low min** at 1 and **Deq low max** at 0. On every hit, when the compressor reduces gain, a resonant low-end bump appears below 300 Hz; between hits, the sound returns to its original balance. The drums keep their body despite the compression, and the movement of the band gives a sense of breathing. Set up the compressor first, then raise the resonance until you hear the bump, and dial in the amount with **Deq low min** (around 0.5 for a subtle effect).
>
> The same principle works for the top end. The **waves_la2a** preset, an imitation of an LA-2A emulation, sets the DEQ's high band at 7 kHz with **Deq hgh min** at 0.5: the extreme top end opens up as the compressor works, by about 2 dB at 6 dB of gain reduction and up to 3.5 dB at full reduction. On a vocal, it's a way to keep the air when the compression bears down.

Where they sit in the chain

The two dynamic EQs and the waveshaper form a trio, always in the same order: **DEQ**, then **WS**, then **DEQ 2**. This trio exists in two places:

- on the **target**, that is, the part of the signal the compressor is going to attenuate (chapter 14);
- on the **detection**, that is, the signal the compressor measures (disabled by default).

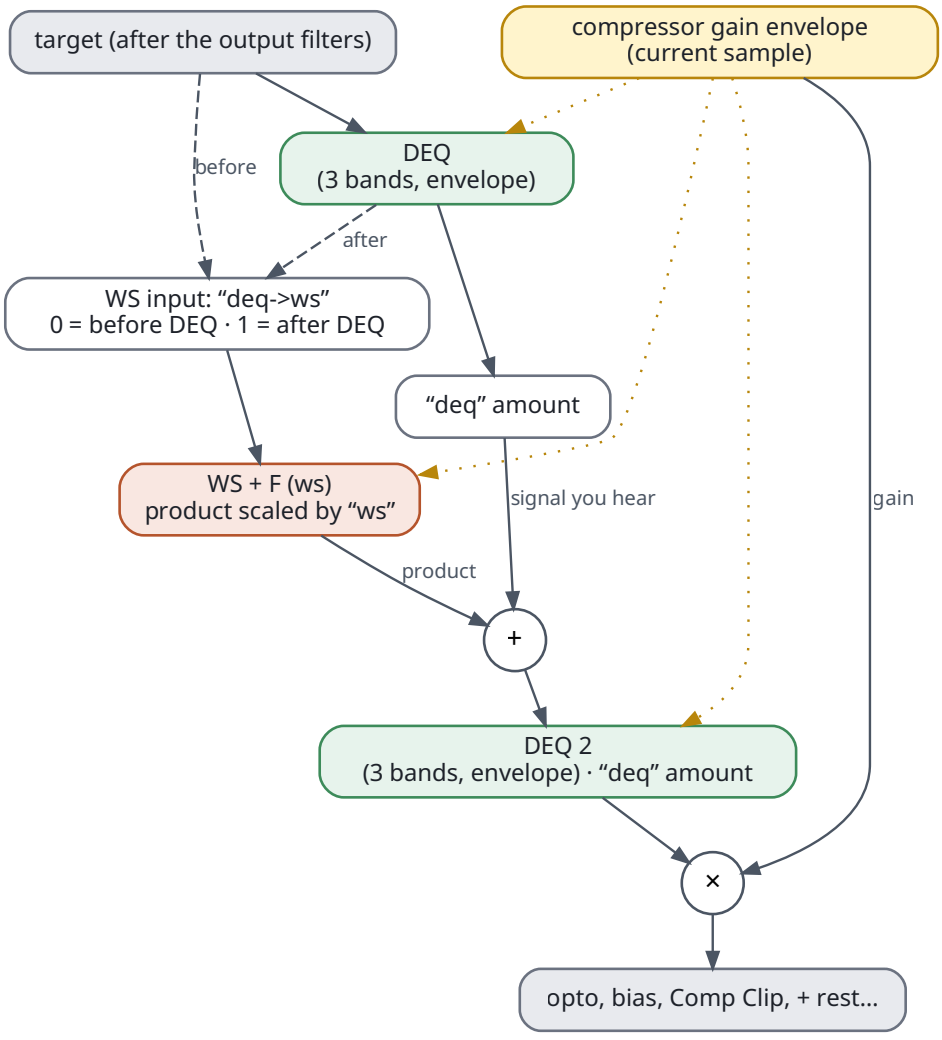


Figure 101 — The DEQ, WS, DEQ 2 trio on the target. The dotted lines show what the compressor's gain envelope controls.

On the target, the trio acts **after** the gain envelope is computed, but **before** the target is multiplied by the compressor’s gain. Then come the opto, the bias, the compressor’s clipper, and the addition of the remainder of the signal.

Filtering and multiplying by the gain are both linear operations: as long as the gain changes slowly, placing the EQs before or after the multiplication amounts to nearly the same thing. The difference shows up with what isn’t linear: the EQs’ clipping at ±1 and the waveshaper see the **uncompressed** level of the signal.

> **A word from the author**
>
> Since compression is a multiplication, François points out that applying the dynamic EQ just before or just after the compressor makes little difference to the ear.

The envelope used

The envelope e is the compressor's final gain envelope, the one drawn by the white line in the history graph (chapter 4).

- On the **target**, it's the envelope of the current sample.
- On the **detection**, the current sample's gain isn't known yet when the measured signal is being prepared, so the envelope of the **previous sample** is used. This one-sample delay is tiny; it makes the detection DEQ a small feedback loop, comparable to feedback compression (chapter 14).
- When the channels are unlinked (**Channels Unlink**, chapter 7), each channel's envelope contributes in proportion to the unlink amount.
- The **Gr Amp** control (CURVE tab, chapter 10) doesn't change the envelope the dynamic EQs see: it only affects the gain applied to the target.

> **To be confirmed by François**
>
> The gain envelope can exceed 1 (for example with a positive **Env Offset** or a **GR Min** above 0 dB). The dynamic EQs' coefficients are then extrapolated beyond **max**, and the waveshaper's **Fet gr** amount becomes negative (the saturation product is inverted). Is this an intended use case?

The sends: target, detection and waveshaper

Several controls in the **OUTPUT** and **INPUT** tabs set how much of the trio is used. Their cards are in chapter 14; here is what they do from the EQs' point of view.

Tab	Label	Default	Role
OUTPUT	**deq** (the first one)	1	amount of DEQ applied to the target
OUTPUT	**deq->ws**	1	input of the target's waveshaper: 0 = signal before the DEQ, 1 = signal after the DEQ
OUTPUT	**ws**	1	amount of waveshaper product added to the target
OUTPUT	**deq** (the second one)	1	amount of DEQ 2 applied to the target
INPUT	**FX deq** (the first one)	0	amount of DEQ applied to the detection (from −1 to 1)
INPUT	**FX deq->ws**	0	input of the detection's waveshaper: before or after the DEQ
INPUT	**FX ws**	0	amount of waveshaper product added to the detection (from −1 to 1)
INPUT	**FX deq** (the second one)	0	amount of DEQ 2 applied to the detection (from −1 to 1)

So by default, the dynamic EQs act fully on the target and not at all on the detection. On the detection, a negative amount inverts the EQ's effect: the bands it would raise are lowered, and vice versa.

Saturating the EQ without hearing it

The **deq->ws** control picks what goes into the waveshaper, independently of how much of the DEQ you hear. With **deq** at 0 and **deq->ws** at 1, the DEQ isn't heard, but the waveshaper saturates the equalized signal: you hear the original signal plus a saturation product computed from the equalized version. This lets you, for example, saturate mostly the low end when the compressor is working, without hearing any extra low end. DEQ 2 has no equivalent, since it sits after the waveshaper.

> **A word from the author**
>
> That's the whole point of having a dynamic EQ before the waveshaper: you drive into the saturation with the dynamic EQ, without hearing the dynamic EQ. And that choice is made independently for the output and for the detector.

> **♪ In the studio: saturating the low end without hearing it**
>
> On a bass or a kick drum, this trick gives you harmonics that are born from the low end, and so follow the notes, without bloating the bottom of the spectrum. The **softube_germanium** preset, an imitation of a FET compressor emulation, uses it: in the DEQ tab, a ZDF low band around 240 Hz, heavily boosted (**Deq low herz** at 239.86, **Deq low max** at 4.713); in the OUTPUT tab, the first **deq** at 0, so that this boost isn't heard but feeds the waveshaper (**Fet odd** at 0.863); finally, in the F (ws) tab, only the low part of the saturation is fed back in. Listen with **Monitor Fet**: the saturation follows the low end, while the EQ itself stays inaudible.

The DEQ in the detection

With **FX deq** at 1, the compressor measures the equalized signal. Since the EQ itself follows the gain reduction, the compressor's sensitivity to each frequency region changes with the compression: for example, a low band that rises with gain reduction makes the compressor more and more sensitive to the low end as it compresses.

A modulation source

The two detection dynamic EQs can also serve as a **modulation source** for the ratio, the gate, the attack and the release (the **deq mod** controls in the MOD & LA tab, chapter 23). The modulation measures, on every sample, how much the EQ changes the level of the detection signal. The detection DEQ is then computed even if **FX deq** is 0: it can modulate the compressor without changing what it measures.

When the bass calls the shots

In radio, broadcasters soon found that with wideband compression, the bass modulates everything else, and the listener hears the pumping. Hence the first multiband compressors: a two-band Altec unit in the 1960s, Mike Dorrough's processor in 1971, from which the three-band DAP 310 descends, then Bob Orban's Optimod 8000A in 1973. Other designers preferred to act on what the compressor hears. The LA-2A already had, on its back, a pre-emphasis that made its detector more or less sensitive to the vocal range. The Distressor (1995) has its "band emphasis," an EQ in the detector. The **Thrust** feature of the API 2500 (2002) puts a filter in front of the RMS detector that reduces the lows and boosts the highs, because music has more energy in the low end: less pumping, more punch. Placed in the detection, Comprendre's dynamic EQs carry this idea further, with a twist: the equalization of what the compressor hears itself moves with the compression.

♪ In the studio: changing what the compressor hears

With the DEQ in the detection, you don't touch the tone: you change what makes the compressor react, and François points out in his video that the compressor then reacts very differently. Two presets use it without letting you hear the EQ (second **deq** in the OUTPUT tab at 0, second **FX deq** in the INPUT tab at 1). **toneproject_unisum**, an imitation of a mastering compressor with multiband detection, strongly boosts the 3.5 kHz region in DEQ 2 (**Deq mid herz** at 3502.22, **Deq mid max** at 5.771): at rest, the compressor is very sensitive to presence, and that sensitivity drops as it compresses. **massenbug_drc2_piano**, whose name points to the piano, does the same in the low end (**Deq low herz** at 280.9, **Deq low max** at 8, **Deq low reso** at 0.35): the low register weighs much more heavily in the decision to compress, especially at rest.

For mastering, François goes further. By placing the dynamic EQ in the detector and using it to modulate the release according to frequency (chapter 23), he feels you reach a kind of overall coherence that is almost more a matter of mastering than of compression.

The DEQ controls

The fourteen controls fall into three groups, one per band. For each band: a frequency (**herz**), a resonance (**reso**), a coefficient at rest (**max**), a coefficient at full reduction (**min**) and, for the low and high bands, the filter type (**zdf->svf**). How the filters work (slopes, resonance, ZDF/SVF blend) is covered in detail in chapter 16.

Low band

Deq low herz DEQ tab

Range: 0 to 16000 Hz **Default:** 0 Hz **ID:** `deq_bas_coupe`

Cutoff frequency of the low band's low-pass. At **0** (default), the band is off and isn't computed. It goes up to 16 kHz; the midpoint of the knob's travel is 1,000 Hz.

The low band contains everything below this frequency. To work on the low end of a mix or a drum kit, values between 80 and 300 Hz are typical.

Deq low reso

DEQ tab

Range: 0.000001 to 16 **Default:** 0.707107 **ID:** `deq_bas_resonance`

Resonance (Q factor) of the low band's SVF low-pass. At **0.707** (default), the cutoff has no bump. Above that, a bump appears around the cutoff frequency: the low band focuses on one region, like an adjustable low-end "bump." The resonance has no effect on the ZDF part of the filter.

With a high resonance, the DEQ also creates a dip just above the cutoff frequency, caused by the phase shift of the filter added to the signal.

Deq low min

DEQ tab

Range: −8 to 8 **Default:** 0 **ID:** `deq_bas_min`

Coefficient of the low band when the gain envelope is **0** (full reduction). It ranges from −8 to +8; default **0**.

The actual coefficient is **min** + (**max** − **min**) × envelope: at 6 dB of gain reduction, it is halfway between **min** and **max**.

- **min** greater than **max**: the band rises with compression (min 1, max 0: up to +6 dB in the low end at full reduction).
- **min** less than **max**: the band drops with compression.
- A coefficient of −1 removes the band; below that, it comes back with inverted polarity.

The knob is very tightly packed around 0; type in precise values with the keyboard.

Deq low max

DEQ tab

Range: −8 to 8 **Default:** 0 **ID:** `deq_bas_max`

Coefficient of the low band when the gain envelope is **1**, that is, **with no gain reduction**. It ranges from −8 to +8; default **0**.

This is the band's "at rest" EQ. At 0, the band has no effect as long as the compressor isn't compressing. A nonzero value acts even without compression: with max 1 and min 0, the low end is boosted by 6 dB at rest and this correction fades away as the compressor reduces gain. With **min** equal to **max**, the band is static.

Deq low zdf->svf

DEQ tab

Range: 0 to 1 **Default:** 1 **ID:** `deq_bas_type`

Blend between the one-pole low-pass (ZDF, 6 dB per octave, at **0**) and the two-pole low-pass (SVF, 12 dB per octave, with resonance, at **1**, the default). In-between values add the two outputs together.

The ZDF gives an exact subtraction: with a coefficient of −1, the signal becomes exactly its ZDF high-pass, with no bump or dip.

Mid band

Deq mid herz

DEQ tab

Range: 0 to 16000 Hz **Default:** 0 Hz **ID:** `deq_mid_coupe`

Center frequency of the mid band's band-pass. At **0** (default), the band is off. The mid band is always an SVF band-pass: it has no type control.

Deq mid reso

DEQ tab

Range: 0.000001 to 16 **Default:** 0.707107 **ID:** `deq_mid_resonance`

Resonance of the mid band-pass. It sets the **width** of the band (the higher it is, the narrower the band) and also its **level**, since the band-pass gain at the center equals the resonance.

- **0.707** (default): wide band, center at −3 dB before the coefficient.
- **1**: center at 0 dB; a coefficient of −1 then notches out the center completely, and a coefficient of 1 raises it by 6 dB.
- **4**: narrow band, center at +12 dB; a coefficient of 1 gives +14 dB at the center.

For a narrow but moderate correction, raise the resonance and scale the coefficients down accordingly.

Deq mid min

DEQ tab

Range: −8 to 8 **Default:** 0 **ID:** `deq_mid_min`

Coefficient of the mid band at full gain reduction (envelope 0), from −8 to +8; default **0**. Works like **Deq low min**.

With a resonance of 1, min −1 and max 0, the band dips progressively with compression until it disappears at full reduction (−6 dB at the center at 6 dB of reduction).

Deq mid max

DEQ tab

Range: −8 to 8 **Default:** 0 **ID:** `deq_mid_max`

Coefficient of the mid band with no gain reduction (envelope 1), from −8 to +8; default **0**. Works like **Deq low max**.

High band

Deq hgh herz

DEQ tab

Range: 0 to 16000 Hz **Default:** 0 Hz **ID:** `deq_top_coupe`

Cutoff frequency of the high band's high-pass. At **0** (default), the band is off. The high band contains everything above this frequency: brilliance, sibilance, the attack of percussion.

Deq hgh reso

DEQ tab

Range: 0.000001 to 16 **Default:** 0.707107 **ID:** `deq_top_resonance`

Resonance of the high band's SVF high-pass. At **0.707** (default), the cutoff has no bump; above that, a bump appears around the cutoff frequency. No effect on the ZDF part.

Deq hgh min

DEQ tab

Range: −8 to 8 **Default:** 0 **ID:** `deq_top_min`

Coefficient of the high band at full gain reduction (envelope 0), from −8 to +8; default **0**. Works like **Deq low min**.

With a negative coefficient, the top end softens when the compressor works; with a positive coefficient, it gets brighter.

Deq hgh max

DEQ tab

Range: −8 to 8 **Default:** 0 **ID:** `deq_top_max`

Coefficient of the high band with no gain reduction (envelope 1), from −8 to +8; default **0**. Works like **Deq low max**.

Deq hgh zdf->svf

DEQ tab

Range: 0 to 1 **Default:** 1 **ID:** `deq_top_type`

Blend between the one-pole high-pass (ZDF, at **0**) and the two-pole high-pass (SVF, at **1**, the default) of the high band. Works like **Deq low zdf->svf**.

> **♪ In the studio: a color at rest that fades away as it compresses**
>
> Many presets set a band with a nonzero **max** and a **min** at or near 0: the correction is there at rest and fades away as the compressor works. With a negative **max** in the low end, you hear it as a low end that sits slightly back in the quiet passages and returns when the music pushes and the compressor bears down: the compression seems to open up the bottom of the spectrum instead of flattening it. **bx_townhouse** does just that, cutting the low end below 100 Hz at rest (**Deq low max** at −0.33). **softube_fet** does the same (−0.5 below 100 Hz) and adds a little air above 5 kHz (**Deq hgh max** at 0.5), which fades away when the compressor clamps down. **arturia_fet** sets both bounds in DEQ 2: below 65 Hz, **Deq low max** at −0.75 and **Deq low min** at −0.2; above 9 kHz, −0.5 and −0.1. The low end and the extreme top end, pulled back at rest, partly return with compression.
>
> To try it on a drum bus or a mix: a low band around 100 Hz, **max** between −0.3 and −0.5, **min** at 0, then compare by bringing **max** back to 0. The movement is subtle; you mostly hear it in passages where the dynamics change a lot.

DEQ 2: the EQ after the waveshaper

The **DEQ 2** tab has exactly the same controls as **DEQ**, with the same labels, and it works the same way. There are only three differences:

- it sits **after** the waveshaper: it equalizes the already saturated signal, saturation product included;
- it has no equivalent of the **deq->ws** control: nothing links it to the waveshaper's input;
- its amounts are the second **deq** control in the OUTPUT tab and the second **FX deq** in the INPUT tab; its modulation source is the second **deq mod** in the MOD & LA tab.

Each EQ has its own filters: you can, for example, boost the low end before the saturation with DEQ and cut it after with DEQ 2.

> **Duplicate names**
>
> In your host's automation list, the parameters of the two EQs have the **same names** ("deq low Herz," "deq low env min," and so on), and the amounts in the INPUT tab are both called "FX deq." This manual tells them apart by their ids: `deq_…` for DEQ, `dep_…` for DEQ 2. To find out which one you're automating, touch the control in the interface and watch which parameter moves.

> **♪ In the studio: a fixed EQ for color**
>
> When the compressor isn't reducing gain, the envelope is 1 and each band keeps its **max** coefficient: the dynamic EQ becomes a fixed EQ. Three presets from the saturator family (chapter 25) use it this way, with no compression at all (**Ratio** at 1, **Threshold** at 0 dB). **kazrog_true-iron**, an imitation of transformer saturation, lowers the low end below 50 and 60 Hz in both EQs, lifts the top end above roughly 6.9 kHz in DEQ (**Deq hgh max** at 0.2744), then, in DEQ 2, sets a ZDF high band at 9 kHz with **Deq hgh max** at −1: the signal then keeps only its content below 9 kHz, with a gentle 6 dB per octave slope. **pa_hg2** adds a little presence above 3 kHz (**Deq hgh max** at 0.1207 in DEQ 2), after the waveshaper's saturation. **dafalife_sausage** scoops out the low end below 80 Hz in both EQs.
>
> Since it sits after the waveshaper, DEQ 2 is also the natural place to correct the color of a saturation: **kiive_xstressor**, for example, cuts the top end above 7 kHz at rest (**Deq hgh max** at −0.33 in DEQ 2).

Deq low herz

DEQ 2 tab

Range: 0 to 16000 Hz **Default:** 0 Hz **ID:** `dep_bas_coupe`

Cutoff frequency of the low-pass of DEQ 2's low band. At 0 (default), the band is off. Works like **Deq low herz** in DEQ.

Deq low reso

DEQ 2 tab

Range: 0.000001 to 16 **Default:** 0.707107 **ID:** `dep_bas_resonance`

Resonance of the SVF low-pass of DEQ 2's low band (0.707 by default, no bump). Works like **Deq low reso** in DEQ.

Deq low min

DEQ 2 tab

Range: −8 to 8 **Default:** 0 **ID:** `dep_bas_min`

Coefficient of DEQ 2's low band at full gain reduction (envelope 0), from −8 to +8, default 0. Works like **Deq low min** in DEQ.

Deq low max

DEQ 2 tab

Range: −8 to 8 **Default:** 0 **ID:** `dep_bas_max`

Coefficient of DEQ 2's low band with no gain reduction (envelope 1), from −8 to +8, default 0. Works like **Deq low max** in DEQ.

Deq low zdf->svf

DEQ 2 tab

Range: 0 to 1 **Default:** 1 **ID:** `dep_bas_type`

ZDF (0) / SVF (1, default) blend of DEQ 2's low band. Works like **Deq low zdf->svf** in DEQ.

Deq mid herz

DEQ 2 tab

Range: 0 to 16000 Hz **Default:** 0 Hz **ID:** `dep_mid_coupe`

Center frequency of DEQ 2's mid band-pass. At 0 (default), the band is off. Always an SVF band-pass.

Deq mid reso

DEQ 2 tab

Range: 0.000001 to 16 **Default:** 0.707107 **ID:** `dep_mid_resonance`

Resonance of DEQ 2's mid band-pass: the width of the band and the gain at the center (equal to the resonance). Works like **Deq mid reso** in DEQ.

Deq mid min

DEQ 2 tab

Range: −8 to 8 **Default:** 0 **ID:** `dep_mid_min`

Coefficient of DEQ 2's mid band at full gain reduction, from −8 to +8, default 0. Works like **Deq mid min** in DEQ.

Deq mid max

DEQ 2 tab

Range: −8 to 8 **Default:** 0 **ID:** `dep_mid_max`

Coefficient of DEQ 2's mid band with no gain reduction, from −8 to +8, default 0. Works like **Deq mid max** in DEQ.

Deq hgh herz

DEQ 2 tab

Range: 0 to 16000 Hz **Default:** 0 Hz **ID:** `dep_top_coupe`

Cutoff frequency of the high-pass of DEQ 2's high band. At 0 (default), the band is off. Works like **Deq hgh herz** in DEQ.

Deq hgh reso

DEQ 2 tab

Range: 0.000001 to 16 **Default:** 0.707107 **ID:** `dep_top_resonance`

Resonance of the SVF high-pass of DEQ 2's high band. Works like **Deq hgh reso** in DEQ.

Deq hgh min

DEQ 2 tab

Range: −8 to 8 **Default:** 0 **ID:** `dep_top_min`

Coefficient of DEQ 2's high band at full gain reduction, from −8 to +8, default 0. Works like **Deq hgh min** in DEQ.

Because it sits after the waveshaper, it lets you, for example, soften the top end that the saturation adds when the compressor is working hard.

Deq hgh max

DEQ 2 tab

Range: −8 to 8 **Default:** 0 **ID:** `dep_top_max`

Coefficient of DEQ 2's high band with no gain reduction, from −8 to +8, default 0. Works like **Deq hgh max** in DEQ.

Deq hgh zdf->svf

DEQ 2 tab

Range: 0 to 1 **Default:** 1 **ID:** `dep_top_type`

ZDF (0) / SVF (1, default) blend of DEQ 2's high band. Works like **Deq hgh zdf->svf** in DEQ.

Practical workflows

More low end when it compresses. This is the example from François's video on the dynamic EQs. In the DEQ tab, set **Deq low herz** to around 300 Hz, **Deq low reso** between 1 and 2, **Deq low min** to 1 and **Deq low max** to 0. At rest, nothing changes; when the drum hits push the compressor into gain reduction, a low-end bump appears below 300 Hz and makes up for the low end that compression takes away.

Taking out harshness when it compresses. Turn on the mid band around 2 to 4 kHz, with a resonance of 1, **Deq mid min** at −1 and **Deq mid max** at 0. The harder the compressor works, the deeper the upper mids dip. If gain reduction stays moderate, lower **Deq mid min** toward −2 to reach the full notch at the peak of compression, without going past it.

An at-rest EQ that fades away with compression. Set **max** to the value you want at rest (for example 0.5 on the high band for a little air) and **min** to 0: the correction gradually fades away as the compressor reduces gain.

Saturating the low end without hearing it. Set up the DEQ's low band (min 1, max 0, or static with min = max = 2), then, in the OUTPUT tab, set the first **deq** to 0 while leaving **deq->ws** at 1. Engage the waveshaper (chapter 17). The equalized low end isn't heard, but it feeds the saturation: the harmonics come mostly from the low end.

Making compression frequency-sensitive. In the INPUT tab, raise the first **FX deq** toward 1 and, in the OUTPUT tab, set the first **deq** to 0 if you don't want to hear the EQ. The compressor then measures the equalized signal: a boosted high band (for example **min** = **max** = 1 for a static correction) makes it more sensitive to sibilance.

Correcting after the saturation. In the DEQ 2 tab, a high band with a negative **min** softens the extra top end that the waveshaper produces when the compressor reduces gain heavily.

Starting ideas

- **Drums.** The 300 Hz example, to keep the body on every hit; or a mid band around 3 to 5 kHz with a negative **min**, so the cymbals get less harsh when the compressor clamps down.
- **Vocals.** A high band with a positive **min**, as in **waves_la2a**, to keep the air when the compression bears down; or a mid band around 2 to 4 kHz with **min** at −0.5, to soften the harshness that comes out under heavy compression.
- **Bass.** Saturate the low end without hearing it, the way **softube_germanium** does, so the bass stands out in the mix without gaining low end.
- **Mix bus and mastering.** An at-rest correction that fades away with compression (**bx_townhouse**, **softube_fet**), or a dynamic EQ placed in the detection (**toneproject_unisum**, **massenbug_drc2_piano**) to choose what makes the compressor react.
- **Color without compression. Ratio** at 1 and static bands, as in **kazrog_true-iron**.

Chapter 25 presents these presets within their family, and chapter 26 offers other starting points depending on your line of work.

CHAPTER 19

Introduction to the Bias

The **bias**, which François also calls the **Vari-Mu designer**, is the richest and most baffling module in Comprendre. It isn't just one more control on the compressor, but a second architecture grafted around it: an inverted copy of the entire compressor, a "bias" value computed continuously, "heights" added to the signal, and then a recombination of the two copies. The result is a family of asymmetric saturations and compressions that no other module in the plugin produces.

What the bias lets you get close to is the most elusive part of tube compressors. On a vari-mu, compression and saturation come from the same gesture: the control voltage shifts the bias of the tubes, and the harder you drive it, the more the compression turns into saturation, especially on transients. On the hardware, this mechanism is frozen into the circuit. In Comprendre, each of its ingredients becomes adjustable: where the bias comes from, how fast it moves, where it is added to the signal, how the two sides of the push-pull recombine. The principle of the Fairchild or the Manley Variable Mu is no longer a black box: you can explore it piece by piece, then invent variations of your own.

It all started with something François read: the article that the French manufacturer Pulsar Audio devoted to how variable-bias compressors work. François says it plainly in his French walkthrough: without that article, he would never have written the bias, and he thanks Pulsar for it. The module is aimed at mixing and mastering engineers looking for the thickness and "glue" of tube compressors, at those who want to imitate a specific unit, and at sound designers, for whom it is a generator of saturations you won't find anywhere else. Twenty of the eighty bundled presets turn it on, from imitations of Fairchild 670 emulations to imitations of SSL-style bus compressors.

The bias spans six tabs:

Tab	Role	Chapter
BIAS src	where the bias value comes from and how it is shaped; height added at the detector input; inverse path	20
DET (bs), **ENV (bs)**, **A/R (bs)**	the bias's own detector, which smooths or transforms its value	20
BIAS dst	height added to the audio signal, limiter, fusion of the two paths, product delay, monitoring	21
F (bs)	processing of the bias saturation product	21

This chapter has no parameter cards: it explains the principle, shows where the saturation comes from, and gives starting settings so you can hear the bias at work. Chapters 20 and 21 cover each control in detail.

In the signal chain (chapter 6), the bias comes in at two points:

- **at the detector input**, between the input effects (dynamic EQs and waveshaper sent to the detection) and the detection itself;
- **on the audio**, after the compressor gain and the opto are applied, before the compressor's clipper (**Comp Clip**), **Comp Db**, the addition of the remainder of the signal, and the monitors.

> **A word from the author**
>
> François admits that this part of the plugin is so complex that it would need a manual of a hundred pages or so all by itself, and that it will probably take a year and a lot of users to find out everything it can do. All its options were born from research aimed at reproducing a specific behavior, described below; many of them turned out to be interesting in their own right.

A real vari-mu compressor

The module is inspired by a post on the blog of Pulsar Audio, the maker of the Pulsar Mu, about how variable-bias compressors work. François recommends reading it: in his view, once you understand that mechanism, you understand most of the bias options.

The article describes how a "variable-bias" tube compressor, better known as a *vari-mu*, works:

1. **A control loop.** The compressor's output feeds a peak detector, with its own attack and release. The detector produces a gain reduction signal. This is a feedback structure: you measure what comes out.
2. **The bias.** This gain reduction signal serves as the bias voltage for a remote-cutoff tube. The audio signal is added to this bias, and the tube's gain depends on the resulting operating point. The louder the signal, the more it interacts with the bias, and the more the compression turns into saturation.
3. **The push-pull circuit.** Two branches receive the signal in opposite phase: the audio is inverted from one to the other, but the bias has the same sign in both. Since the bias is added to audio of opposite sign in each branch, the two branches don't saturate the same way.
4. **Recombination.** The output transformer takes the difference of the two branches. The bias, identical on both sides, cancels out; the audio, opposite on the two sides, adds up. The saturations of the two branches combine into a very particular shape, most visible on transients.

Comprendre transposes each of these elements:

Vari-mu compressor	In Comprendre
bias voltage from the detector	the **bias value**, written β, between 0 and 1, computed from the output or from the gain reduction (**BIAS src** tab), then smoothed by its own detector (**DET (bs)**, **ENV (bs)**, **A/R (bs)**)
the signal is added to the bias	a **height** proportional to β added to the signal: at the detector input (**Bias input add**) and/or to the audio signal (**Bias x base**, **Bias x env**, **Bias x gr**)

Vari-mu compressor	In Comprendre
two branches in opposite phase	the **normal path** and the **inverse path** (**Bias +inverse**)
the tube saturation	the compressor itself, when its times are very short, and the **bias limiter** (**Bias on limit** and the controls that follow it)
the output transformer	the **recombination** of the two paths, direct or after "unbiasing" (**Bias <> bs->ub**)
(no equivalent)	the bias **saturation product**, processed by the **F (bs)** tab

Notation used in these three chapters

- d: the detection signal (what the compressor measures); x: the target (what the compressor attenuates, chapter 14).
- e and e′: the gain of the normal compressor and that of the inverse compressor (1 = no reduction).
- β: the bias value, generally between 0 and 1.
- B: a height, that is, β multiplied by one of the amounts (**Bias input add** for the input, **Bias x base/env/gr** for the output).
- s and s′: the output of the normal path and that of the inverse path, with the bias.
- s_pre: the output of the normal path **without** any height on the audio. It's the reference against which the saturation product is measured.
- O: the recombined output; P = O − s_pre: the saturation product.

The Manley Variable Mu, a vari-mu revival

Manley Laboratories has built its Variable Mu in mono since 1991, and in stereo since 1994. Its 5670 dual triode is "constantly re-biased" by a control voltage, using the same technique as the Fairchild 670, and its control chain is feedback: the manual even finds the feed-forward design of most modern compressors rather unmusical. In **Compress** mode, the ratio is 1.5:1 with a soft knee; in **Limit** mode, it starts at 4:1 and rises up to 20:1 beyond 12 dB of reduction, while the knee gets softer as the limiting increases. Originally named the "10 dB Limiter," the unit was meant to stay mostly inaudible up to about ten decibels of reduction. It became a staple of mastering studios, known for giving a project "glue." This ratio that grows and this knee that softens with gain reduction are the measurable signature of re-biasing.

> **♪ In the studio: the vari-mu family in the presets**
>
> To hear what the bias brings, the easiest way is to load a preset from the vari-mu family (chapter 25), then bring its five amounts back to 0 to compare. **pulsar_mu**, an imitation of a Manley Variable Mu emulation, sets an input height (**Bias input add** at 1) and an output height (**Bias x base** at 1), a bias limiter (**Bias on limit** at 4, **Bias on ratio** at 1.5) and "unbiased" fusion (**Bias <> bs->ub** at 1). The imitations of Fairchild 670 emulations each go about it in their own way: **tr_670** combines **Bias input add** at 2.534 with **Bias x gr** at 0.3333; **overloud_670** pushes **Bias input add** to 25 with **Bias x env** at −1; **acustica_ultramarine4** goes as far as **Bias input add** at 42, with **Bias x gr** at 2.5. Listen above all to the attacks, where, according to the Pulsar article, a vari-mu's compression turns into saturation. These presets are imitations built by ear and with the analysis mode (chapter 24): starting points to adjust to your own music.

The normal path and the inverse path

When the bias is active and **Bias +inverse** is greater than 0, Comprendre runs **two complete compressors** in parallel. The first, the normal path, receives the signal as is. The second, the inverse path, receives the signal with its sign flipped: −d for the detection, −x for the target.

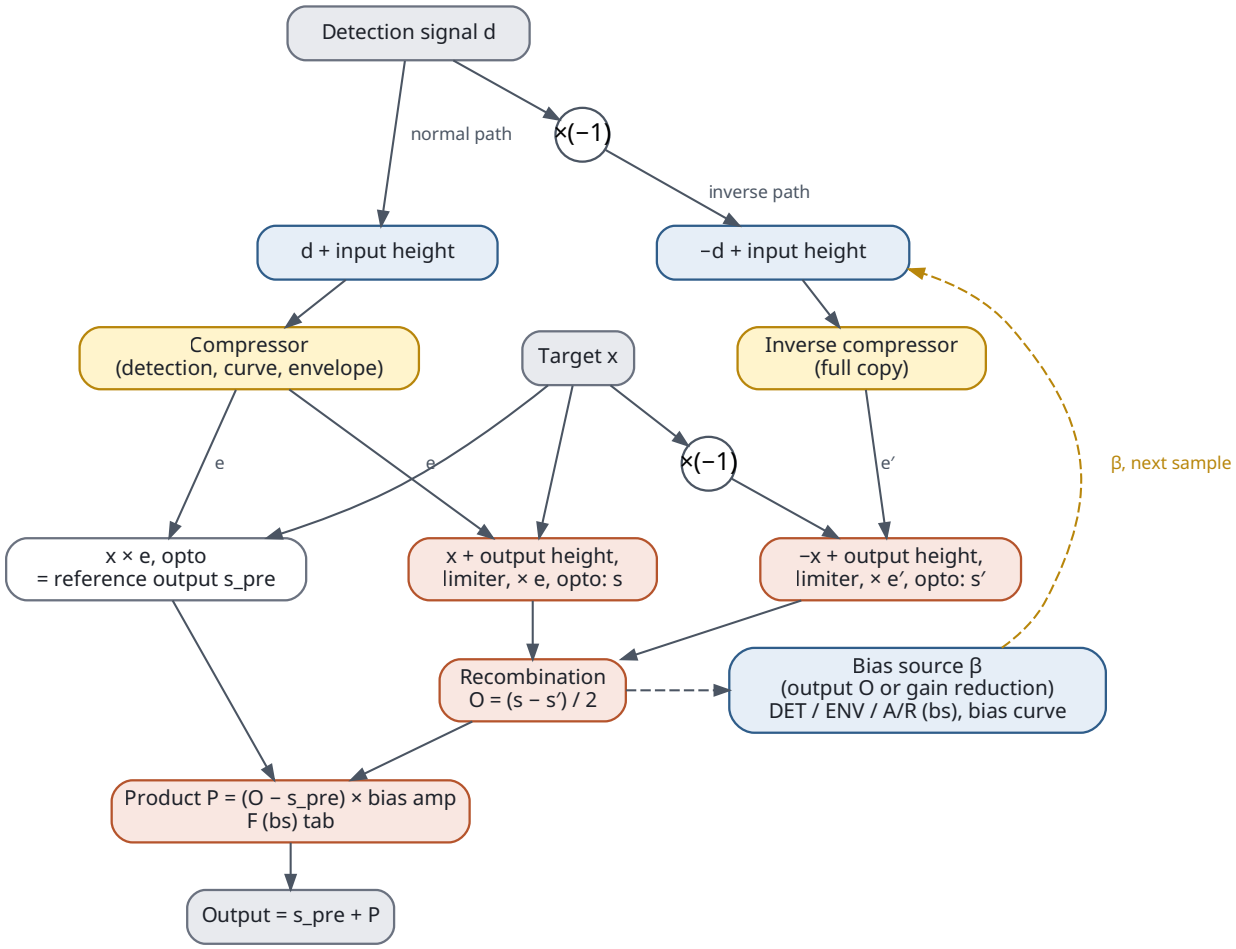


Figure 102 — The two bias paths. The detection signal and the target are duplicated and inverted; each path has its own compressor; the two outputs are recombined; the difference from the reference s_pre is the saturation product. The bias value β, computed at the end of a sample, sets the input and output heights of both paths for the next sample (dotted arrows).

What gets duplicated

All of the compressor's processing is copied onto the inverse path, with its own memories (filters, envelopes, delay lines):

- the effects sent to the detection: **DEQ**, waveshaper (**WS**), **DEQ 2** (chapters 17 and 18);
- the detection (**DET (in)**, **ENV (in)**, **A/R (in)**), the transfer curve, the envelope follower (**ENV (cv)**, **A/R (cv)**) and the gain reduction options (**GR (cv)**), including in unlinked mode (**Channels Unlink**);
- the **lookahead** and its envelope (**ENV (la)**, **A/R (la)**, chapter 22);
- the target effects (**DEQ**, **WS**, **DEQ 2**) and the **opto** (chapter 15).

The compressor's CPU load is therefore roughly doubled.

What doesn't

- **The bias value**: there is only one β per channel, computed once and shared by both paths.
- **The heights**: the input height and the output height are the same on both paths. The output height is computed with the normal path's gain only (chapter 21).

- **The saturation product and the F (bs) tab**: they act after the recombination, only once.
- **Everything that comes before and after the compressor**: gains, clippers, headroom, auto volumes, makeup, mixes (chapters 7 and 8).

> **Going further**
>
> The inversion is applied to the detection signal and to the target, before the input and output effects. If those effects are neutral (or symmetric), the inverse path processes exactly −d and −x. When the detection is fed from the output (**FF->FB** control, chapter 14), the feedback share of the inverse path is the opposite of the **recombined** output, not the output of the inverse path itself.
>
> When the bias is active, the main curves in the graphs (detection, gain, envelope, and so on) show the weighted average of the two paths: (X + i·X′)/(1 + i), where i is the value of **Bias +inverse**.

> **To be confirmed by François**
>
> There are two small differences in the code between the normal and inverse paths. The **Fet gr** modulation of the waveshaper sent to the detection receives, on the inverse path, the envelope instead of the gain reduction (the target side is correct). And in linked stereo, with an active modulation (MOD & LA tab, chapter 23) and the **Channels Mix** modes “channel 0,” “channel 1” or “max,” the inverse path’s modulation value is accumulated instead of being copied. These two discrepancies can create a saturation product even without an input height; are they intended?

Recombination

At the output, the two paths are brought back together with the formula:

$$O = s \times (1 - i) + (s - s') \times i / (1 + i)$$

where i is the value of **Bias +inverse**:

- at **i = 1** (default), O = (s − s′) / 2: this is the difference of the two paths, as in the transformer of a push-pull circuit;
- at **i = 0**, O = s: only the normal path counts, and the inverse path isn’t even computed.

If the two paths are perfectly symmetric, s′ = −s and the recombination gives back exactly s: the bias changes nothing. All the interest comes from what **differs** between the two paths, that is, from the height, which has the same sign on both sides while the signal changes sign.

Recombination has a simple property that is very useful for understanding what you hear:

- everything that is **identical** on both paths disappears: the height itself and, for a periodic signal, the DC component and the **even harmonics**;
- everything that is **opposite** is kept: the fundamental and the **odd harmonics**.

In other words, with **Bias +inverse** at 1 and neutral effects, the recombined output is the **odd part** of the normal path. The saturation product P = O − s_pre is then exactly the opposite of its even part (DC component and even harmonics).

Push-pull, the 'thud' and the Balance control

From the very first radio limiters, in the late 1930s, tube units mixed a DC control voltage with the audio on the grid of a variable-mu tube, and it was the push-pull circuit that canceled this control voltage at the output. When the tubes aged and drifted out of balance, the cancellation was no longer perfect and "thumps" appeared. The Fairchild 670, built from 1959 on, drives four 6386 dual triodes per channel in push-pull through its input transformer. Its **Balance** control adjusts the balance of the bias currents: well centered, it minimizes the "thud," a deflection added to the signal on every attack; off center, it produces one of one polarity or the other. It's the historic example of an asymmetric saturation born from the bias. In Comprendre, the recombination of the two paths plays the role of the push-pull: with **Bias +inverse** at 1, whatever is common to both paths disappears; below 1, part of the normal path, with its asymmetry and its DC component, remains audible (chapter 20). Chapter 2 tells the story of the Fairchild.

The bias height

The word "bias" refers here to a **height**: an offset added to the signal, proportional to the bias value β. Comprendre offers two of them, independent of each other, which François calls the "bias height for the input" and the "bias height for the output":

- the **input height** $B = \beta \times$ **Bias input add** is added to the detection signal of both paths. It doesn't touch the audio: it only changes what the two compressors measure. The normal path measures $|d + B|$, the inverse path $|-d + B|$, so they no longer compress the same way (chapter 20);
- the **output height** $B = \beta \times$ (**Bias x base** + **Bias x env** $\times e +$ **Bias x gr** $\times (1 - e)$) is added to the audio signal of both paths, before or after the compressor gain. It shifts the signal upward on both paths, which makes it hit the bias limiter sooner on one side than on the other (chapter 21).

A third way of using β, **Bias input mul**, isn't a height: it **multiplies** the detection signal. Since it acts the same way on d and on −d, it creates no difference between the two paths (chapter 20).

Where the saturation comes from

The bias saturation can arise in three places.

In the detector

With an input height, the two compressors no longer see the same level: on a positive half-cycle, the normal path measures $d + B$, which is larger, and compresses more; the inverse path measures $-d + B$, which is smaller, and compresses less. On a negative half-cycle, it's the other way around. The gains e and e′ differ.

If the compressor's attack and release are very short, the gain follows the waveform itself, within each cycle: the compressor then behaves like a **waveshaper**, asymmetric on each path. That's what François shows in his video on the inverse path, with attack and release at zero.

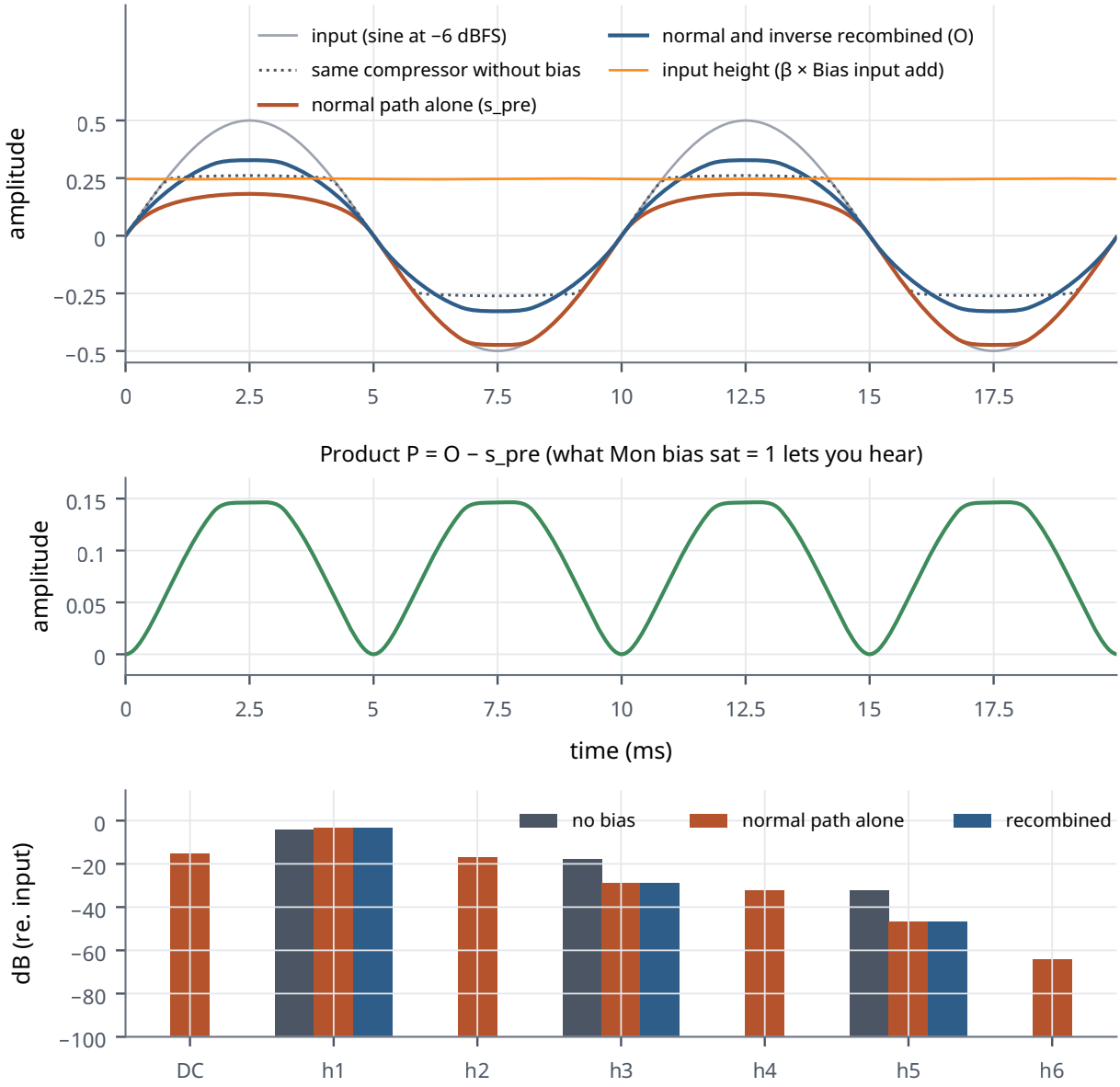


Figure 103 — Simulation of the code with the starting settings described below (Threshold −12 dB, Ratio 25, Knee 12, Attack and Release at 0, Bias input add = 1, bias detector in RMS over 50 ms), for a 100 Hz sine wave at −6 dBFS, in steady state. Top: the normal path alone is asymmetric; recombined with the inverse path, it becomes symmetric again. Middle: the saturation product. Bottom: level of the DC component and of the first six harmonics; the normal path alone contains DC and even harmonics, while the recombination removes them and keeps its odd harmonics exactly.

In this figure, notice that:

- the input height (in orange) is almost constant: the bias detector, set to RMS, makes it “flat”;
- the normal path alone compresses the positive half-cycle more than the negative one: it is shifted downward (negative DC) and rich in even harmonics;
- the recombined output is symmetric; its shape isn’t that of the same compressor without the bias: the bend is more gradual and the peak less squashed;
- the product P is always positive here, at twice the frequency of the signal: it’s the opposite of the DC component and even harmonics of the normal path.

In the limiter

With **Bias +inverse** at 1 and no limiter, the output height has no audible effect in itself as long as the two compressors have the same gain: it is identical on both paths and the recombination cancels it. It gets interesting with the **bias limiter**: the shifted signal x + B hits the ceiling sooner on positive half-cycles, and the signal −x + B on negative ones. François sums it up this way: when you put height on the signal, at some point it “bumps against the top.”

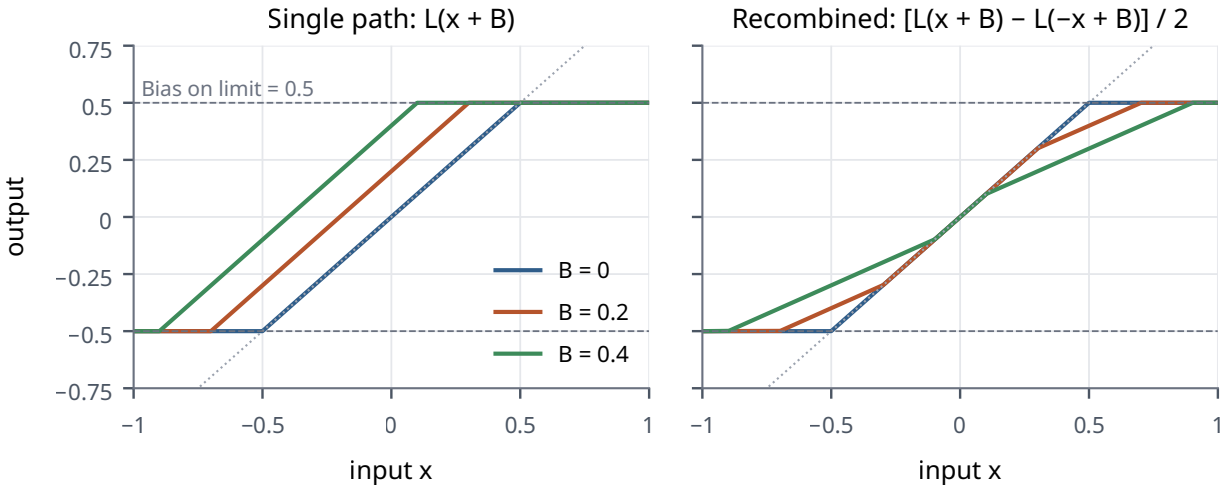


Figure 104 — Output height and bias limiter (hard ceiling at 0.5, in the “pre” position, no compression). Left, a single path: the height shifts the curve and the ceiling is reached on one side only. Right, after recombining the two paths: the curve is symmetric again, with a two-slope bend.

After recombination, with a hard ceiling T and a height B:

- up to T − B, the curve is a straight line with a slope of 1: nothing happens;
- between T − B and T + B, only one of the two paths is clipped: the slope drops to ½;
- beyond T + B, both paths are clipped: the output tops out at T.

So the height sets the **width of the bend**, and the limiter sets its **position**. This is the “push-pull” bend: a saturation that builds in two stages, and stays symmetric.

> **♪ In the studio: saturation that builds in two stages**
>
> The push-pull bend rounds off the peaks in stages: nothing happens below T − B, the slope is halved between T − B and T + B, then the output tops out. On a bass, a vocal or a bus, you often get a more gradual rounding than with a simple clipper, and you set its width (the height) and its position (the limiter) separately. Start with a bias limiter placed just above the peaks, gradually add output height and listen to **Mon bias sat**: start subtle, the color should come before the distortion. Chapter 21 covers these controls in detail.
>
> Some presets use the bias this way, as a saturator with no compressor. **pulsarmodular_p42**, an imitation of an overdriven line amplifier, doesn’t compress (**Ratio** at 1, **Threshold** at 0 dB) and relies mainly on the bias limiter (**Bias on limit** at 0.09665, **Bias on ratio** at 3.91, **Bias Fm On** at 1.525), with the amount set by **bias amp** at 0.475.

At the recombination

The recombination doesn't create saturation, but it sorts: it cancels what is even and keeps what is odd. That explains two practical rules:

- **without an inverse path** (**Bias +inverse** at 0), nothing is canceled. The biased normal path is heard as is, with its DC component and its even harmonics. The bias then becomes, in François's words, "something rather feedback-oriented": a compressor whose detection or signal is offset by its own output;
- **with an input height alone and no inverse path**, the saturation product is zero: the reference s_pre is itself computed with the biased detection. The asymmetric compression is audible, but it's part of the reference, and **Mon bias sat** stays silent. With an output height and no inverse path, on the other hand, there is indeed a product: the height itself (multiplied by the compressor gain in the "pre" position), which you hear as a slow shift of the signal, and the asymmetric clipping of the limiter.

The bias saturation product

Like the opto and the waveshaper (chapter 16), the bias doesn't directly replace the signal with its saturated version. It computes the difference between the recombined output O and the reference s_pre, the output of the normal path **without** any height on the audio:

P = (O − s_pre) × **bias amp**

This product goes through the delay (**Bias Delay Secs**, **Bias Delay FB**, **BIAS dst** tab), then the filters and the limit of the **F (bs)** tab, and is then added back to s_pre. With the default settings (**bias amp** at 1, filters off), the output is exactly O.

- **bias amp at 1**: you hear the recombined output O. With **Bias +inverse** at 1, that's the odd part of the biased path.
- **bias amp at 0**: you hear s_pre, the normal path alone. The bias's effects **on the detector** (input height, **Bias input mul**, **Bias Fm In**) remain, since they determine the compressor's gain.
- **bias amp at −1**: you hear 2 × s_pre − O, so the even part of the normal path is doubled instead of removed.

The **Mon bias sat** monitor (**BIAS dst** tab) lets you hear P alone, after F (bs) (chapter 21).

> **The bias value doesn't see F (bs)**
>
> When the bias value is taken from the output, it is measured on the recombined output **before** the product and the F (bs) tab. The F (bs) settings change what you hear, not the bias value.

> **♪ In the studio: a saturation generator**
>
> The bias isn't just an imitation tool. By combining the source, the heights, the inverse path and the fusion, François has seen new saturations emerge, all the way to a never-before-heard kind of white noise that he plays in his video on fusion. He has listed the combinations in a table: out of 32, about ten give, in his view, truly distinct types of saturation products. To explore, begin with the starting settings below, set **Mon bias sat** to 1 and change one setting at a time: **Bias +inverse** to 0, then the source, then the bias detector. Turn up your monitoring level, as François advises: some effects are subtle and only reveal themselves at high levels.
>
> In the saturator family (chapter 25), **acqua_ash-ultra**, an imitation of a clipper, combines an input height (**Bias input add** at 0.4062) taken from the gain reduction (**Bias g.e->out** at 0), an output height that follows the gain reduction (**Bias x gr** at −4.675) and a delay on the saturation product (**Bias Delay Secs** at 0.01454 s, **Bias Delay FB** at 0.1854). **kiive_filk2_saturation** adds to its waveshaper an input height of 5 and a bias limiter at 0.25.

Nothing runs while the five amounts are at 0

The bias turns on as soon as **one** of the following five amounts is not 0:

Amount	Tab	What it does
Bias input add	BIAS src	height added to the detection signal
Bias input mul	BIAS src	multiplies the detection signal by the bias
Bias x base	BIAS dst	height added to the audio, proportional to β
Bias x gr	BIAS dst	height added to the audio, proportional to β and to the gain reduction
Bias x env	BIAS dst	height added to the audio, proportional to β and to the gain



Figure 105 — The five amounts that turn on the bias: two in the BIAS src tab…



Figure 106 — … and three in the BIAS dst tab.

As long as these five amounts are at 0 (which is the default):

- no bias computation is done, and the inverse path doesn't exist, even though **Bias +inverse** defaults to 1;
- **all the other controls** in the six bias tabs have no effect: sources, **Bias Feedback**, bias detector, bias curve, **Bias Fm In**, **Bias Fm On**, limiter, fusion, delay, F (bs) tab. **Mon bias sat** then plays silence, at least if **Comp Clip**, **Comp Db** and the target filters are neutral (chapter 21);
- the compressor behaves exactly as if the module didn't exist, with no extra CPU load.

As soon as a single amount is not 0, the **whole** module wakes up: the inverse path is computed (if **Bias +inverse** is greater than 0), and the controls that were ignored until then take effect. For example, with only **Bias x base** set, **Bias Fm In** starts acting as a gain on the detection, even without an input height.

> **Turning the bias off in one move**
>
> To compare with and without the bias, just bring the five amounts back to 0. The module's other settings are kept and will take effect again as soon as an amount is active once more.

Recreating the article's waveform: starting settings

In his introduction video on the bias, and then in his French walkthrough, François shows that with very few settings you can get, on an oscilloscope and with a sine wave, a waveform like the one in the article. The settings he mentions are a ratio of 25, a little knee, an input height (**Bias input add**), and above all **RMS detection in the bias detector**, not in the main detector. The rest (inverse path, source taken from the output, classic fusion) matches the default values.

Here's one way to go about it, starting from the default preset. The threshold, knee and RMS time values below are an example: they're the ones used in the simulation shown in the previous figure.

1. On the main row: **Ratio** at 25, **Knee** at 12, **Threshold** around −12 dB for a test sine at −6 dBFS. The compressor should be clearly compressing.
2. **Attack** and **Release** at 0: the gain then follows the waveform, which makes the saturation visible on an oscilloscope.
3. **BIAS src** tab: **Bias input add** at 1. Leave **Bias +inverse** at 1, **Bias g.e->out** at 1 (the source is the output) and **Bias out atom** at 0 (classic measurement).

4. **DET (bs)** tab: **rms secs** at 0.05 s, **rms db** at 0 dB and **Det Mix** on *rms*. The bias becomes a slow value, almost a flat line on the graph, instead of following every half-cycle.
5. Watch the output on the oscilloscope, then set **Mon bias sat** (**BIAS dst** tab) to 1 to hear the saturation product alone.



Figure 107 — Starting settings, main row and BIAS src tab (steps 1 to 3).



Figure 108 — Starting settings, DET (bs) tab: the bias detection switches to RMS (step 4).

Then a few variations let you get a feel for the mechanism:

- **raise Bias input add**: the input height grows. As long as it stays well below the signal level, it modulates the compression; once it exceeds it, both detectors mostly see the height, the compression is driven by the bias itself, and the shape breaks down;
- **bring Bias +inverse back to 0**: the output becomes the normal path alone again, asymmetric, with DC and even harmonics, but the product (and so **Mon bias sat**) drops to zero;
- **set Attack and Release back to usual values**: the gain no longer follows the waveform, the saturation disappears, and what's left is a compression whose detection is offset by the output, like a feedback compressor.

> **A word from the author**
>
> François advises setting the input height high enough to influence the detection, but not so high that it takes over: if you add too much bias, what you get is the bias itself. He adds, not without humor, that he couldn't get this shape out of the Pulsar Mu itself, which must therefore have other ingredients.

To be confirmed by François

The videos don't give all the values used in the demonstration (threshold, knee, RMS time of the bias detector, attack and release, sine level). Could you provide the exact settings from the introduction video, so they can be turned into an example preset? And could you confirm that the article cited is indeed "Variable-bias compression explained" from the Pulsar Audio blog, and give its link?

The bias over time

The bias value is computed at the **end** of the processing of each sample, from what has just been produced (the output or the gain reduction). It's used at the start of the **next** sample: for the input height, the multiplication of the detection and the output height. The bias is therefore always a one-sample **feedback loop**, whatever source you choose. On load, β is 0.

When the lookahead is active (chapter 22), the plugin's "future" section contains its own complete copy of the bias, with its own β value. There are then two independent bias loops: one on the future signal, used only for the lookahead analysis, and the other on the present signal, the one you hear.

Starting ideas

- **Glue for a bus or a mix.** Load **pulsar_mu** or **tr_670**, adjust the threshold for a few decibels of reduction, then compare with the five bias amounts at 0.
- **Denser drums. rockruepel_comptwo_drums** combines an input height (**Bias input add** at 7.63) with the waveshaper's grain on the peaks (chapter 17).
- **SSL-style bus compression. cytomic_glue** and the presets **ssl_1_b**, **ssl_1_f**, **ssl_2_b** and **ssl_2_b_** add no height: they multiply the detection by the bias value (**Bias input mul**, chapter 20).
- **Imitating a specific unit.** Start from the article's settings (above), then fine-tune with the analysis mode (chapter 24); chapter 25 presents the imitations in the vari-mu family.
- **Sound design.** Attack and release at 0, **Bias input add** around 1, RMS in the bias detector, then **Bias +inverse** at 0, a negative **Bias Feedback** (chapter 20) or a delay on the product (chapter 21).

What's next

Chapter 20 describes the bias **source**: how its value β is obtained (output or gain reduction, classic or "atom" measurement, Feedback, dedicated detector, curve), and how it acts on the detector input (**Bias input add**, **Bias input mul**, **Bias Fm In**). Chapter 21 describes its **destination**: the height added to the audio, **Bias Fm On**, the pre or out position, the limiter, the fusion of the two paths, the saturation product and how to monitor it, along with tables that sum up the possible combinations.

CHAPTER 20

The Bias Source

This chapter describes how Comprendre builds the **bias value**, written β, and how that value acts on the detector input. All the controls involved are in the **BIAS src** tab and in the three tabs of the bias detector, **DET (bs)**, **ENV (bs)** and **A/R (bs)**. The general principle of the bias, the inverse path and the notation (d, x, e, β, B, s, O, s_pre, P) are introduced in chapter 19.

On a vari-mu compressor, the bias of the tubes comes from the output: what the unit has just produced decides how it handles what comes next. This chapter puts you in control of that loop. You choose what the "bias" listens to (the output, as on the hardware, or the compressor's work), how fast it charges and discharges, like a control voltage going through a rectifier and a capacitor, and what it does to the detector: offset it, the way a vari-mu does, or multiply it, as François observed in the schematic of SSL compressors. These choices shape the character. Smoothed in RMS, the bias becomes an almost flat line and lets you recreate the shape described in the Pulsar article (chapter 19); left at peak, it follows every half-cycle, and the effect becomes, as François puts it in his French walkthrough, "ultra violent."

These tabs are where most of the vari-mu and bus compressor imitations bundled with Comprendre do their work: their presets set the input height, the multiplication of the detection and the speed of the bias detector here. If you're trying to reproduce the behavior of a specific unit, this is where you'll spend the most time, with the analysis mode (chapter 24) close at hand.

Remember that nothing described here works as long as the five bias amounts are at 0: **Bias input add** and **Bias input mul** (in this tab), **Bias x base**, **Bias x gr** and **Bias x env** (**BIAS dst** tab, chapter 21).



Figure 109 — The BIAS src tab: inverse path, origin of the bias value, Feedback and bias curve.

The tab reads from left to right, roughly in the order of the computation:

- **Bias +inverse**: the weight of the inverse path;

- **Bias g.e g->e**, **Bias out atom**, **Bias out unlink**, **Bias out bs->ub**, **Bias g.e->out**: where the bias value comes from;
- **Bias Feedback**: its scaling;
- **Bias thresh**, **Bias ratio**, **Bias knee**: the bias curve, applied after its detector;
- **Bias input add**, **Bias Fm In**, **Bias input mul**, **Bias mul %**: what the bias value does to the detector input.

Computing the bias value

The bias value is recomputed at the end of each sample, for each channel, and used for the next sample. It always follows the same path:

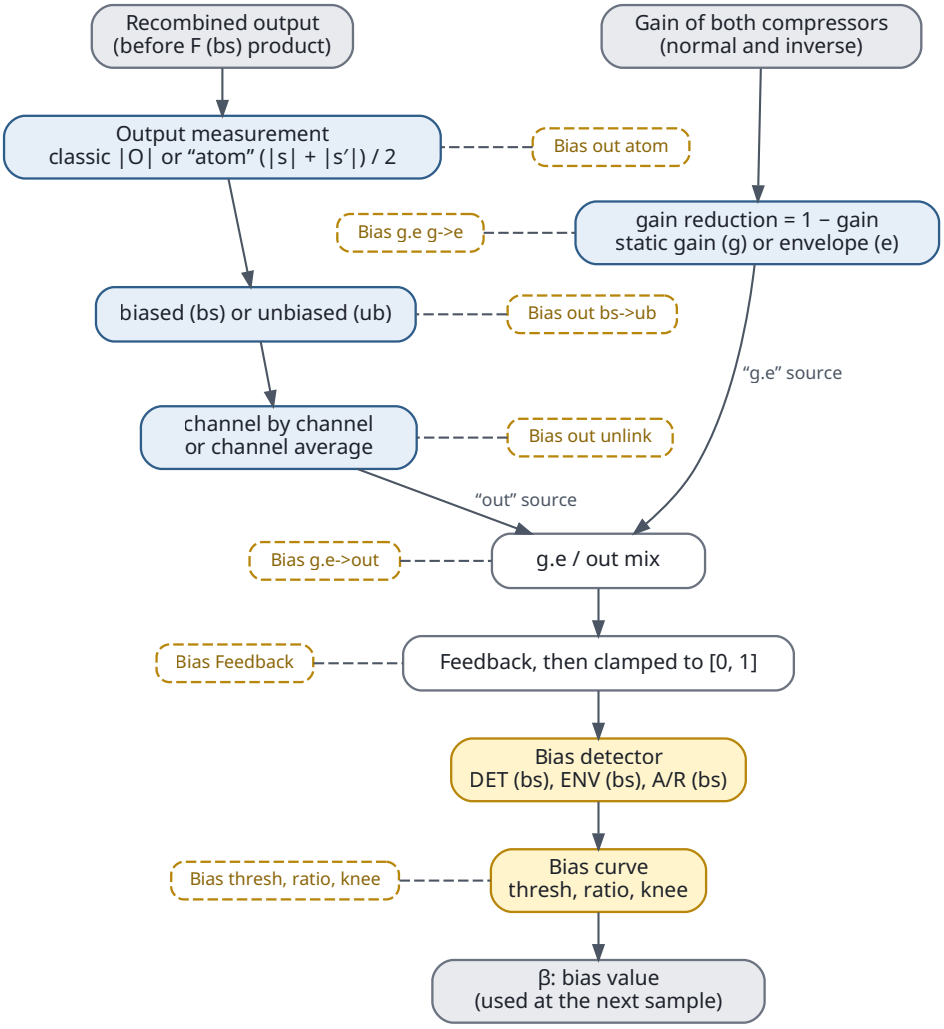


Figure 110 — Computing the bias value, from the source to β. The dotted labels show the control that acts at each step.

1. **A source**, between 0 and 1: either the recombined output or the compressors' gain reduction;
2. **the Feedback**, which multiplies it or reverses its direction, followed by clamping between 0 and 1;
3. **the bias detector (DET (bs)**, **ENV (bs)**, **A/R (bs)** tabs), which can smooth or transform it;
4. **the bias curve** (threshold, ratio, knee), which can compress it.

The result, β, is used for the next sample by the input height (**Bias input add**), the multiplication of the detection (**Bias input mul**) and the output height (**BIAS dst** tab).

The inverse path

Bias +inverse

BIAS src tab

Range: 0 to 1 **Default:** 1 **ID:** `bias_inverse`

Weight of the **inverse path**, the complete copy of the compressor that processes the sign-flipped signal (chapter 19).

- **1** (default): the two paths are recombined by their difference, O = (s − s′) / 2. This is the "push-pull" mode of operation: what is identical on both paths (the height, the DC component, the even harmonics) cancels out, and what is opposite (the fundamental, the odd harmonics) is kept.
- **0**: the inverse path isn't computed, which roughly halves the compressor's CPU load. The output is that of the biased normal path, with its asymmetry, its DC component and its even harmonics. The bias then works as a feedback mechanism rather than as a push-pull.
- **In between**: the output is s × (1 − i) + (s − s′) × i / (1 + i). This blend doesn't keep a constant level: when the two paths are symmetric (s′ = −s), it equals s × (1 + 2i − i²) / (1 + i), which means a gain of about +1.2 dB at 0.25, +1.4 dB at most (around 0.41), +1.3 dB at 0.5 and +0.9 dB at 0.75.

The same weight is used for the "g.e" source (weighted average of the gains of the two paths) and for the display: the main curves in the graphs show the weighted average of the two paths.

This control defaults to 1, but it has no effect as long as the bias hasn't been activated by one of the five amounts.

To be confirmed by François

Intermediate values of **Bias +inverse** produce a level bump of up to +1.4 dB, and the label "+inverse" suggests a switch. Are the intermediate values meant to be used, or should users be advised to stay at 0 or 1?

Where the bias comes from: the output or the gain reduction

As in a vari-mu compressor, the bias value can be taken from the compressor's **output**: that's the default setting. It can also be taken from the **gain reduction** computed by the compressor. François considers this choice the most important one for the source.

Bias g.e->out

BIAS src tab

Range: 0 to 1 **Default:** 1 **ID:** `bias_sortie`

Chooses the origin of the bias value.

- **1** (default): the "**out**" source, the compressor's recombined **output**, measured according to **Bias out atom**, **Bias out bs->ub** and **Bias out unlink**.
- **0**: the "**g.e**" source, the compressors' **gain reduction**, computed according to **Bias g.e g->e**.
- **In between**: linear blend of the two sources; both are then computed.

A branch with zero weight isn't computed: at 1, the **Bias g.e g->e** setting is ignored; at 0, it's **Bias out atom**, **Bias out bs->ub** and **Bias out unlink** that are ignored.

With the output as the source, the bias forms a feedback loop similar to that of a real vari-mu: the output determines the bias, which modifies the detection or the signal, which modifies the output. With the gain reduction, the bias follows the compressor's work instead.

The "g.e" source: the gain reduction

Bias g.e g->e

BIAS src tab

Range: 0 to 1 **Default:** 0 **ID:** `bias_source`

For the "g.e" source, chooses which compressor gain is used. It only acts if **Bias g.e->out** is less than 1.

- **0** (default): the **static** gain g, that is, the gain requested by the transfer curve, before attack and release. It follows the signal with no inertia and gives a "rough" bias.
- **1**: the **envelope** e, the gain smoothed by the envelope follower (**ENV (cv)** and **A/R (cv)** tabs). The bias is smoother.
- **In between**: linear blend of the two gains.

The value passed on is the **gain reduction**: 1 − gain, clamped between 0 and 1. It is 0 with no compression, 0.5 for 6 dB of reduction, 0.9 for 20 dB, and tends toward 1 as the reduction gets very heavy. With the inverse path, the weighted average of the gains of the two paths is used: $1 - (G + i \times G') / (1 + i)$.

In stereo, this source follows the **Channels Unlink** setting on the main row (linked gain, per-channel gains, or a blend of the two), not **Bias out unlink**.

With **Attack** and **Release** at 0 and the **GR (cv)** options at their neutral values, g and e are identical and this control no longer has any effect.

> **A word from the author**
>
> Taking the envelope gives a fairly smooth bias, taking the gain a very rough one. François stresses one point: what you need to feed the bias isn't the envelope itself, which is 1 at rest and goes down when you compress, but the gain reduction, which starts at 0 and rises with compression. That's why the code uses 1 − gain.

The "out" source: the output

The output is measured after the two paths are recombined, but **before** the saturation product and the **F (bs)** tab: neither **bias amp**, nor the delay, nor the filters, nor the product limit change the bias value. Three controls refine the measurement.

Bias out atom

BIAS src tab

Range: 0 to 1 **Default:** 0 **ID:** `bias_source_brute`

Chooses how the output is measured, for the "out" source. It only changes the **measurement**: the audio signal itself is always recombined normally.

- **0** (default): **classic** measurement, the absolute value of the recombined output, $|O|$. With **Bias +inverse** at 1, that's $|s - s'| / 2$: the height, identical on both paths, doesn't show up in it.
- **1**: "**atom**" measurement: the absolute value of **each** path, then their average, $(|s| + |s'|) / 2$ with **Bias +inverse** at 1. If an output height shifts both paths, this measurement "sees" it: it never drops below the height.
- **In between**: the blend is done **before** the absolute value. With symmetric paths, the value 0.5 gives half-wave rectification: the measurement follows the positive half-cycles of the output and is 0 during the negative ones.

Without an inverse path (**Bias +inverse** at 0), the classic and atom measurements are identical; only the intermediate values differ.

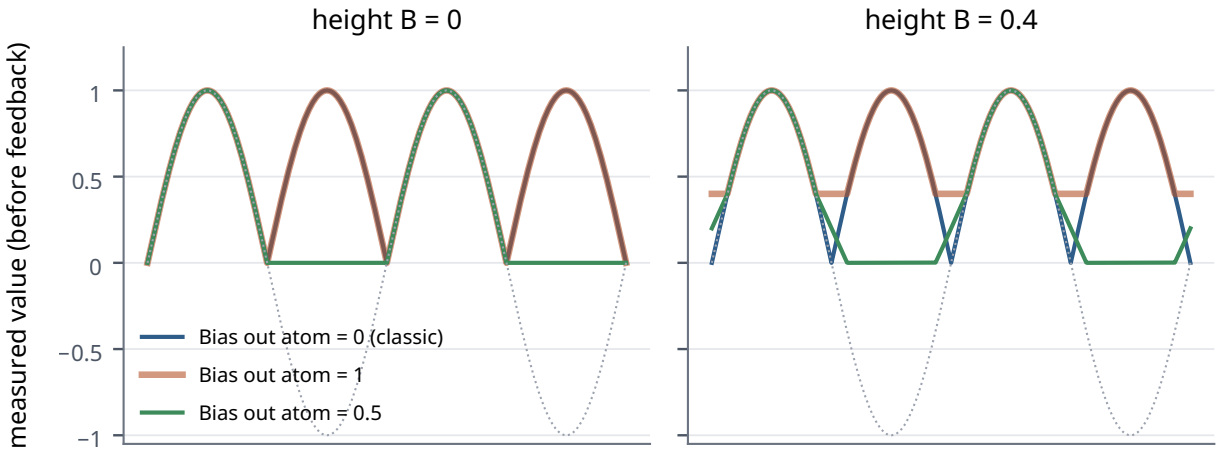


Figure 111 — The three output measurements for a sine wave with an amplitude of 1, Bias +inverse at 1. Left, with no height: classic and atom coincide, and the value 0.5 gives half-wave rectification. Right, with an output height of 0.4 in the "out" position: the classic measurement ignores it, while the atom measurement doesn't drop below it.

A word from the author

There are two ways to bring the two paths together to measure the output: "this minus that," as for the audio, or the absolute value of each path, then their average. François called this second way "atom," for its atomic side, and finds that it gives some really nice effects.

Bias out bs->ub

BIAS src tab

Range: 0 to 1 **Default:** 0 **ID:** `bias_sortie_pp`

For the "out" source, chooses whether the output is measured **biased** (bs) or **unbiased** (ub).

- **0** (default, "bs"): the output is measured as it comes out of the two paths, with its output heights and the **Bias Fm On** factor.
- **1** ("ub"): the output is measured after removing the output heights and dividing by **Bias Fm On**, on each path, before bringing them together (the same "unbiasing" as the fusion, chapter 21).
- **In between**: linear blend of the two measurements.

With **Bias +inverse** at 1 and the classic measurement, the height cancels out anyway in the difference between the two paths: the two measurements differ only by the **Bias Fm On** factor, and are **identical if Bias Fm On is 1**. They really differ when **Bias +inverse** is less than 1, when **Bias Fm On** is not 1, or with the atom measurement, which sees the height.

This control is independent of **Bias <> bs->ub** (**BIAS dst** tab), which makes the same choice for the audio signal.

A word from the author

François notes that unbiasing the source "sometimes does the same thing and sometimes doesn't." The previous paragraph explains why: with the full inverse path, a classic measurement and **Bias Fm On** at 1, the two measurements are identical; in every other case, they differ.

Bias out unlink

BIAS src tab

Range: 0 to 1 **Default:** 1 **ID:** `bias_rupture`

Stereo linking of the "out" source.

- **1** (default): each channel has its own bias, measured on its own output ("dual mono" operation).
- **0**: all channels receive the same bias, the average of the channels' measurements.
- **In between**: linear blend of the two.

This control only concerns the "out" source: the "g.e" source follows **Channels Unlink**. It has no effect on a mono signal. Since the Mid/Side conversion in the MAIN tab is done before the compressor, with **L/R to M/S** active the "channels" are the Mid and the Side.

♪ In the studio: listening to the output or to the compressor's work

With the output as the source (the default setting), the bias follows what you hear: once smoothed by its detector, it rises with dense passages and falls back in the gaps, as on a vari-mu. With the gain reduction as the source, it only moves when the compressor is working: the color of the bias arrives with the compression, like the dynamic saturation of the waveshaper. Between the smooth envelope and the rough static gain, **Bias g.e g->e** sets the grain.

klang_mjuc_mk1, an imitation of a software compressor model inspired by the first vari-mu units of the 1950s, chooses the smoothed gain reduction (**Bias g.e->out** at 0, **Bias g.e g->e** at 1), attenuated by **Bias Feedback** at 0.3162, with **Bias input add** at 1 and **Bias Fm In** at 0.25. At the other extreme, **acqua_ash-ultra**, an imitation of a clipper, takes the gain reduction without smoothing (**Bias g.e->out** at 0, **Bias g.e g->e** left at 0): the roughest bias of all. The other presets that activate the bias keep the output as the source.

The Feedback

Bias Feedback

BIAS src tab

Range: −42 to 42 **Default:** 1 **ID:** `bias_feedback`

Scales the source value, before the bias detector. The name is a reminder that this value is fed back into the processing on the next sample.

- **1** (default): the source passes through unchanged.
- **Between 0 and 1**: it is attenuated.
- **Above 1**: it is amplified, then capped at 1. At 4, for example, any source above 0.25 gives a maximum bias.
- **0**: the bias is zero. The module nonetheless stays active (inverse path, **Bias Fm In**, **Bias Fm On**, and so on).
- **Negative**: the direction is **reversed**. The value becomes 1 − |Feedback| × source, then is clamped between 0 and 1. At −1, a gain reduction becomes a gain, and an output level becomes “1 minus the level”: the bias is at its maximum in silence and decreases as the signal rises. Values more negative than −1 accentuate this inversion: at −2, the bias drops to 0 as soon as the source reaches 0.5.

In every case, the result is clamped between 0 and 1 before entering the bias detector.

The knob goes from −42 to +42, with its travel very tightly packed around 0: for precise values, type them in with the keyboard.

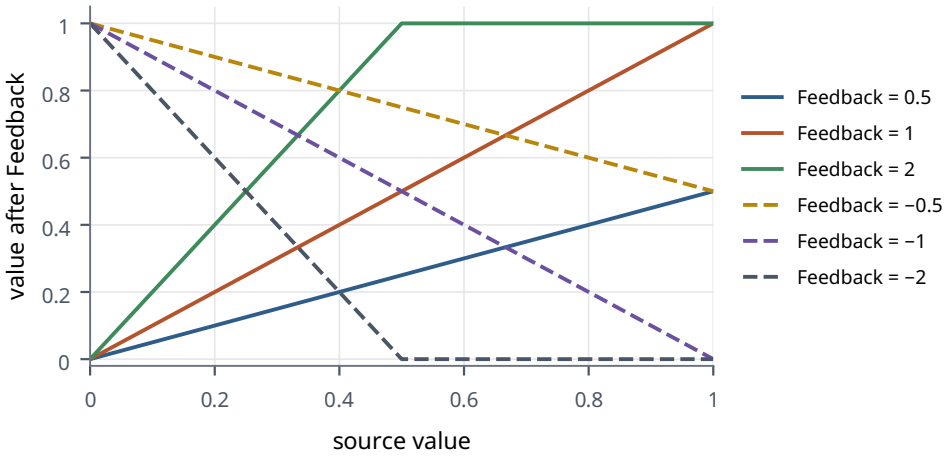


Figure 112 — Effect of the Feedback on the source value. Solid lines, positive values: attenuation or amplification, then capping at 1. Dotted lines, negative values: the direction is reversed.

> **A word from the author**
>
> With a negative value, you get an inversion of the bias, and you can then amplify that inversion by going further into negative values.

The bias detector: DET (bs), ENV (bs), A/R (bs)

After the Feedback, the bias value goes through a detector of its own. It's the same mechanism as the compressor's detection (chapter 13), with its three measurements (peak, RMS, detection envelope), their gains, their powers, the formula and the **Det Mix** blend, and the same detection envelope, set by the **ENV (bs)** and **A/R (bs)** tabs instead of **ENV (in)** and **A/R (in)**.

The differences from the compressor's detection are as follows:

- **the input** is the bias value itself, already between 0 and 1 and positive, channel by channel. There's no linked version: each channel has its own detector;
- **the output** is the result of the **Det Mix** blend, nothing more: no transfer curve, no envelope follower after the curve, no opto. It then goes through the bias curve (threshold, ratio, knee) described further on;
- **the gains in dB** can push the value above 1, and nothing brings it back to 1 afterward (except the bias curve): a **peak db** of +6 dB doubles the bias;
- **auto secs** only sets the automatic times of **A/R (bs)**.

With the default settings (peak at 0 dB, RMS and envelope excluded, **Det Mix** on *max*), this detector is **transparent**: it passes the bias value through unchanged. The value then follows the source sample by sample; with the "out" source, it follows the waveform of the output. To get a slow, almost constant bias, as in the demonstration in chapter 19, you need to smooth it in this detector, with the RMS or the envelope.



Figure 113 — The DET (bs) tab.

> **A word from the author**
>
> To smooth the bias, François puts RMS in the **bias** detector, not in the main detector: that's what turns the bias value, which was following the waveform, into an almost flat line.

The DET (bs) controls

Each of these controls works exactly like its counterpart in the **DET (in)** tab, described in chapter 13. The cards below only cover the essentials and what changes for the bias.

auto secs

DET (bs) tab

Range: 0 to 2 s **Default:** 0.2 s **ID:** `bias_rms`

Time constant of the **crest factor** measurement used by the **Attack Auto** and **Release Auto** controls of the **A/R (bs)** tab, and by them alone (chapters 12 and 13). The measurement is made on the bias value at the detector input.

This control has no effect on the detection itself, nor when both Auto controls of **A/R (bs)** are at 0, nor when the bias detection envelope is inactive.

rms secs

DET (bs) tab

Range: 0 to 2 s **Default:** 0 s **ID:** `bias_det`

Time constant of the bias **RMS** measurement: an exponential root mean square, with the same time constant for rise and fall (chapter 13).

- **0** (default): no RMS; the “rms” measurement is a simple copy of the peak.
- **A few tens of milliseconds**: the bias no longer follows the waveform but the average level; on a sustained sound, it becomes almost constant. This is the setting to use to recreate the behavior of a vari-mu.
- **Longer**: the bias reacts slowly to level changes, like a bias that charges and discharges.

The RMS measurement only counts if **rms db** is above −24 dB (0 dB, for example).

flow

DET (bs) tab

Choices: (empty) · rms=>env · env=>rms **Default:** (empty) **ID:** `bias_pik_nik`

Order in which the measurements are chained (chapter 13): *empty* (default), the measurements are independent; *rms=>env*, the detection envelope follows the RMS instead of the peak; *env=>rms*, the RMS is computed on the output of the envelope.

peak db

DET (bs) tab

Range: −24 to 24 dB **Default:** 0 dB **ID:** `bias_pik_mix`

Gain of the bias **peak** measurement. At **0 dB** (default), the bias passes through unchanged; at **−24 dB**, the measurement is excluded from the blend.

Above 0 dB, the bias is amplified and can exceed 1: the resulting height will be larger than the amount you set. Below that, it is attenuated, which amounts to reducing all the bias amounts at once.

rms db

DET (bs) tab

Range: −24 to 24 dB **Default:** −24 dB **ID:** `bias_det_mix`

Gain of the bias **RMS** measurement. It defaults to **−24 dB**, which excludes the measurement. Raise it (to 0 dB, for example) to use the RMS. Unlike **env db**, the value −24 doesn't stop the RMS from being computed, so it can still feed the envelope (**flow** control).

env db

DET (bs) tab

Range: −24 to 24 dB **Default:** −24 dB **ID:** `bias_var_mix`

Gain of the bias **envelope** measurement. At **−24 dB** (default), the bias detection envelope isn't even computed, and the **ENV (bs)** and **A/R (bs)** tabs have no effect. Above −24 dB, it is computed as soon as the attack or release of **A/R (bs)** is greater than 0; since they default to 10 ms, raising **env db** is enough to activate it.

The detection envelope tops out at 1, unlike the peak and RMS measurements.

peak pow

DET (bs) tab

Range: 0.125 to 8 **Default:** 1 **ID:** `bias_pik_pow`

Power applied to the bias peak measurement, after its gain. Since the bias value is generally less than 1, a power **less than 1** pulls small values toward 1 (the bias becomes present even when the source is weak); a power **greater than 1** crushes them (the bias only shows up when the source is strong). The value 1 is neutral, and a bias value equal to 1 is left unchanged.

rms pow

DET (bs) tab

Range: 0.125 to 8 **Default:** 1 **ID:** `bias_det_pow`

Power applied to the bias RMS measurement, after its gain. Same effect as **peak pow**.

env pow

DET (bs) tab

Range: 0.125 to 8 **Default:** 1 **ID:** `bias_var_pow`

Power applied to the bias envelope measurement, after its gain. Same effect as **peak pow**.

destination

DET (bs) tab

Choices: peak · rms · env **Default:** peak **ID:** `bias_op_dst`

The **destination** measurement of the formula: the one that is replaced by the result (chapter 13).

formula

DET (bs) tab

Choices: d-=s*p · d*=s*p · d=(d*p)/s · d+=s*p · d=1-d^(s*p) · d=d^(s*p) · d=s<p?s:0 · d=s>p?s:O · d+=p · d-=p · d=max(s,p) · d=min(s,p) · d=ssp+d(1-s)(1-p) · d=min(s,p)s+(1-m)d · d=max(s,p)s+(1-m)d · d=1-sp **Default:** d-=s*p **ID:** `bias_op_sig`

The **formula** combining the destination d, the source s and the parameter p (chapter 13, where the sixteen formulas are covered in detail). The tab's menu shows the same labels as the one in **DET (in)** (for example "d-=sp"); the names shown in your host's automation are written "d-=s*p". These are exactly the same calculations.

The result is clamped at 0 from below, but not from above: some formulas (for example "d+=p") can push the bias above 1.

source

DET (bs) tab

Choices: peak · rms · env **Default:** peak **ID:** `bias_op_src`

The **source** measurement of the formula (chapter 13).

parameter

DET (bs) tab

Range: 0 to 1 **Default:** 0 **ID:** `bias_op_par`

The **parameter** p of the formula. At **0** (default), no formula is applied, whatever the **formula** menu is set to (chapter 13).

Det Mix

DET (bs) tab

Choices: max · avg · min · peak · rms · env **Default:** max **ID:** `bias_mmm`

Final blend of the measurements (chapter 13): *max* (default), *avg*, *min*, or a single measurement (*peak*, *rms*, *env*). Only the measurements whose gain is above −24 dB take part in *max*, *avg* and *min*.

For the bias, two cases deserve attention:

- with *min*, the bias is capped at 1; and if all measurements are excluded, it's 1 all the time: constant and at its maximum;
- with *max* or *avg*, if all measurements are excluded, the bias is zero.

Choosing *rms* (with **rms db** above −24 dB) is the most direct way to get a smoothed bias.

ENV (bs) and A/R (bs)

The **ENV (bs)** and **A/R (bs)** tabs set the bias detection envelope, that is, the third measurement of **DET (bs)**. They have exactly the same controls as the other ENV and A/R tabs, described in chapters 11 and 12, and are only active if **env db** (**DET (bs)** tab) is above −24 dB and the attack or release of **A/R (bs)** is greater than 0.



Figure 114 — The ENV (bs) tab.



Figure 115 — The A/R (bs) tab.

Control	Id (bs)	Reference control	Chapter
EvMu Lin/Db	`bias_db`	EvMu Lin/Db (`db`)	11
EvMu Br/Cp	`bias_indirect`	EvMu Br/Cp (`indirect`)	11

Control	Id (bs)	Reference control	Chapter
Hold branch	`bias_pause`	Hold branch (`pause`)	11
Env->Mu	`bias_mu`	Env->Mu (`mu`)	11
Mu Min Att, Mu Max Att, Mu Pow Att	`bias_mu_min_a`, `bias_mu_max_a`, `bias_mu_a`	`mu_min_a`, `mu_max_a`, `mu_a`	11
Mu Min Rel, Mu Max Rel, Mu Pow Rel	`bias_mu_min`, `bias_mu_max`, `bias_mu_r`	`mu_min`, `mu_max`, `mu_r`	11
Mu Speed	`bias_mu_sqrt`	Mu Speed (`mu_sqrt`)	11
Env/Mu->Gamma	`bias_gamma`	Env/Mu->Gamma (`gamma`)	11
Gamma A delai, Gamma R delai	`bias_gamma_attaque_delai`, `bias_gamma_relachement_delai`	`gamma_attaque_delai`, `gamma_relachement_delai`	11
Attack, Release	`bias_attaque`, `bias_relachement`	Attack and Release knobs on the main row (`attaque`, `relachement`)	12
Attack Auto, Release Auto	`bias_auto_attaque`, `bias_auto_relachement`	`auto_attaque`, `auto_relachement`	12
Attack Altitude, Release Altitude	`bias_altitude_attaque`, `bias_altitude_relachement`	`altitude_attaque`, `altitude_relachement`	12
Attack Shape, Release Shape	`bias_attaque_forme`, `bias_relachement_forme`	`attaque_forme`, `relachement_forme`	12
Att Ret From, Att Ret Spd	`bias_attaque_retour_depart`, `bias_attaque_retour_vitesse`	`attaque_retour_depart`, `attaque_retour_vitesse`	12
Rel Ret From, Rel Ret Spd	`bias_relachement_retour_depart`, `bias_relachement_retour_vitesse`	`relachement_retour_depart`, `relachement_retour_vitesse`	12
Attack Mode, Release Mode	`bias_attaque_douceur`, `bias_relachement_douceur`	`attaque_douceur`, `relachement_douceur`	12

What else you need to know for the bias:

- **The direction of attack and release.** Like the compressor's detection envelope, the bias envelope works on "1 minus the value": **Attack** sets how fast the bias **rises**, and **Release** how fast it **falls back**. Both default to 10 ms.
- **The automatic times** use the crest factor measured with **auto secs** in the **DET (bs)** tab.
- **The direction of Mu Min and Mu Max** is the same as in the (in) and (la) tabs, and opposite to that of **ENV (cv)**: the "Max" speed applies when the bias is zero, the "Min" speed when it is at its maximum (chapter 11).

- **Decibel mode** (**EvMu Lin/Db**) smooths the logarithm of "1 minus the bias value," as for **ENV (in)**: its precision is focused on bias values close to 1.
- **On the attack and release shapes graph** (chapter 12), the **A/R (bs)** curve is orange.

Shaping the control voltage

On tube compressors, the speed of the bias is a matter of components. The Gates Sta-Level (1956) rectifies a sample of its output with a dual diode, then sends that DC voltage to the grids of its 6386 tube through a network of resistors and capacitors. The Fairchild 670 offers six **Time Constant** positions: the first pairs a 0.2 ms attack with a 0.3 s release, and the last ones make the release program-dependent, up to 25 s for a constantly high level in position 6. Closer to our time, the SPL Iron makes the bias a control in its own right: a choice of three bias voltages (more bias, more gain reduction) and six diode-capacitor rectifiers, germanium, silicon or LED, which set the shape and speed of the control signal, from 0.1 ms to 5 s. In Comprendre, the DET (bs), ENV (bs) and A/R (bs) tabs play that role: they decide how fast, and in what shape, the bias follows the music.

♪ In the studio: the speed of the bias

The bias detector sets how fast the "bias" reacts, and the ear quickly picks up the difference. Smoothed in RMS over a few tens of milliseconds, it becomes almost constant on a sustained sound: the saturation is steady. Left at peak, it follows every half-cycle, with the "ultra violent" effect François talks about. In between, the detection envelope and its attack and release times let you dial it in, and give the bias the inertia of a capacitor charging and discharging.

The imitations of Fairchild 670 emulations play on this. **overloud_670** runs the bias detection through the envelope (**peak db** at −24, **env db** at 0, **env pow** at 0.5) and sets its attack and release (**A/R (bs)** tab) to 0.2 ms and 0.3 s, the values of the first position of the Time Constant selector; it adds the Mu follower on top, whose speed varies with level (**Env->Mu** at 1, **Mu Min Rel** at 421, **Mu Max Rel** at 0.6226, **Mu Pow Rel** at 1.704 in the ENV (bs) tab). **tr_670** opts for a slower bias (20 ms attack, 1 s release); **acustica_ultramarine4**, for a 2 s release in automatic mode (**Release Auto** at 1, **Release Mode** at 2), with **env pow** at 4.

The bias curve

After its detector, the bias value can also go through a small static compression curve, with no attack or release. It is computed in linear, in decibels, or as a blend of the two, depending on the **Cv Lin/Db** setting on the main row, like the compressor's transfer curve (chapter 10).

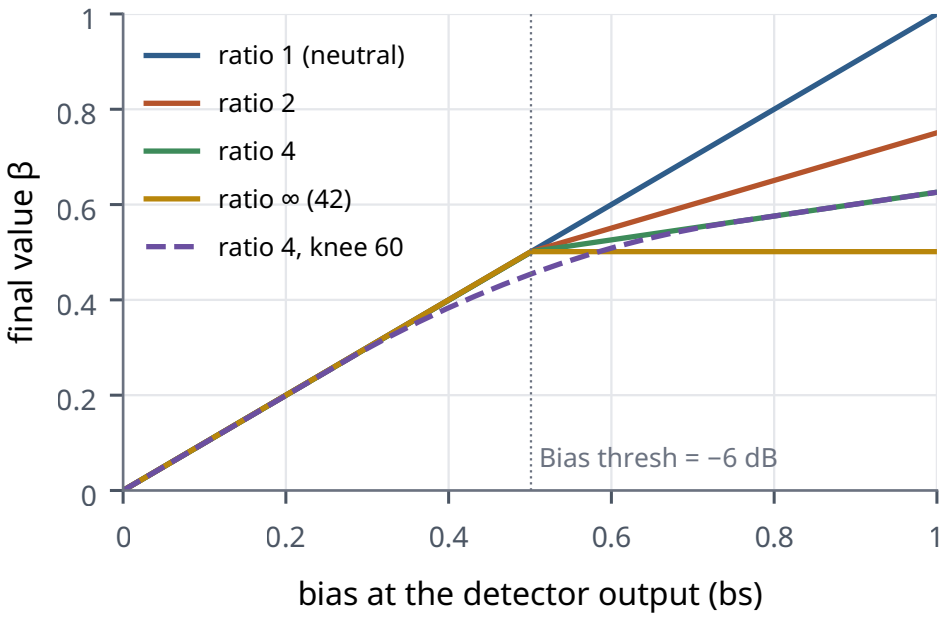


Figure 116 — Bias curve in linear mode, with a threshold of −6 dB: it limits the bias value above the threshold, more or less strongly depending on the ratio, with a gradual bend if the knee is not 0.

Bias thresh

BIAS src tab

Range: −60 to 0 dB **Default:** 0 dB **ID:** `bias_seuil`

Threshold of the bias curve, expressed in dB relative to a bias value of 1 (0 dB). At −6 dB, the curve starts acting from a value of 0.5; at −20 dB, from 0.1.

With the default threshold (0 dB), the curve only acts if the bias detector has pushed the value above 1 (positive gains in **DET (bs)**), or, with a knee, on the part of the bend that lies below 1. To make it act on an ordinary bias, you therefore need to lower the threshold.

Bias ratio

BIAS src tab

Range: 1 to 42 **Default:** 1 **ID:** `bias_ratio`

Ratio of the bias curve. Above the threshold, the amount by which the bias value exceeds it is divided by the ratio.

- **1** (default): the curve isn't computed, whatever the threshold and knee.
- **From 1 to 42**: compression of the bias value. At 2, in linear mode, a value that exceeds the threshold by 0.4 now exceeds it by only 0.2.
- **42**: infinite ratio; the bias value is capped at the threshold.

It's a way to bound the bias: the height stays proportional to the source as long as the source is weak, then levels off.

Bias knee

BIAS src tab

Range: 0 to 240 **Default:** 0 **ID:** `bias_genou`

Width of the bias curve's knee. In linear mode, it is counted in bias value: the displayed value divided by 120 (60 gives a bend 0.5 wide, centered on the threshold). In decibel mode, it is counted in dB. At 0 (default), the bend is sharp.

To be confirmed by François

In linear mode (**Cv Lin/Db** less than 1), with a ratio other than 1, when the bias value is exactly 0 and the bend extends down to 0 (for example a knee above 120 with a threshold of −6 dB), the code divides by 0. The bias value becomes undefined and can spread to the detection and the sound. A zero value is common (silence, "g.e" source with no compression). Is a safeguard needed?

The detector input: Bias input add, mul and Fm In

The last four controls in the **BIAS src** tab describe what the bias value does to the **detection signal**, at the start of the next sample. They don't act on the audio directly, but on what the normal and inverse compressors measure; so they also act on the reference s_pre.



Figure 117 — The detector input controls, in the BIAS src tab.

The operations are carried out in this order, on each path:

1. multiplication by the bias (**Bias input mul**, with the amount set by **Bias mul %**);
2. multiplication by **Bias Fm In**;
3. addition of the input height, β × **Bias input add**, the **same** on both paths;
4. absolute value, then each compressor's detection, curve and envelope.

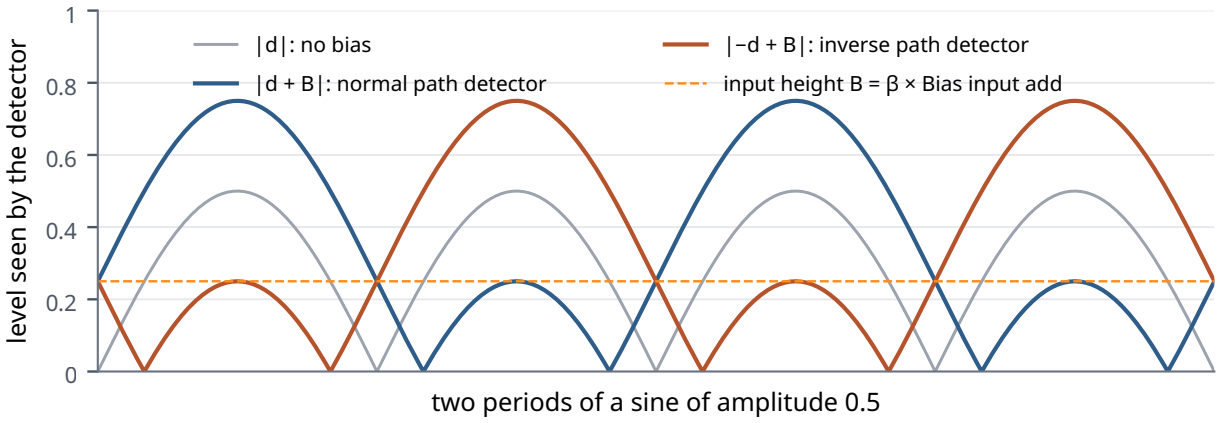


Figure 118 — What the two detectors see for a sine wave with an amplitude of 0.5, with a constant input height of 0.25. The normal path's detector sees the positive half-cycles enlarged and the negative ones reduced; the inverse path's detector sees the opposite.

Bias input add

BIAS src tab

Range: −42 to 42 **Default:** 0 **ID:** `bias_input`

Amount of the **input height**: the bias value, multiplied by this control, is added to the detection signal of both paths. It's one of the five amounts that activate the bias.

- **0** (default): no input height.
- **Positive**: the normal path's detector sees $|d + B|$ and the inverse path's detector $|-d + B|$. On a positive half-cycle, the normal path measures more and compresses more than the inverse path; on a negative half-cycle, it's the other way around. The two compressors no longer give the same gain: this is the source of the saturation "in the detector" (chapter 19).
- **Negative**: the roles of the two paths are swapped.

The height is expressed in linear amplitude, like the signal (1 corresponds to 0 dBFS). With the "out" source, β roughly follows the output level: an amount of 1 adds a height on the order of the output level.

As long as the height stays well below the level of the detection signal, it modulates the compression. If it greatly exceeds it, the two detectors mostly see the height: the compression is then driven by the bias itself, and the signal plays almost no part in it.

This control also multiplies the bias display in the graphs (see below).

A word from the author

The input height has to be strong enough to influence the detection, but not so strong that it takes its place. If you add too much bias, what you get is the bias itself.

Bias Fm In

BIAS src tab

Range: 0.000001 to 8 **Default:** 1 **ID:** `bias_fm_in`

Multiplies the detection signal of both paths **before** the input height is added. So it's a gain on the signal **relative** to the height:

- **1** (default): neutral;
- **small** (down to 0.000001): the signal almost disappears from the detection, which then sees only the height;
- **large** (up to 8): the signal dominates the height.

This control acts as soon as the bias is active, even if **Bias input add** is 0 (for example when only an output height is set). It is then a simple gain on the detection, which amounts to moving the threshold: at 2, the compressor reacts as if the signal were 6 dB louder. Since the detection also determines the reference s_pre, this change is audible even with **bias amp** at 0.

To be confirmed by François

Bias Fm In changes the detection as soon as one of the five amounts is active, including when only an output height (**Bias x base**, **x gr**, **x env**) is used. A control in the BIAS src tab then changes the compression with no connection to the input height. Is that intended?

Bias input mul

BIAS src tab

Range: −168 to 168 **Default:** 0 **ID:** `bias_input_ssl`

Amount of **multiplication** of the detection signal by the bias, on both paths. It's one of the five amounts that activate the bias.

When it is **positive** and **Bias mul %** is greater than 0, the detection signal d becomes $d \times (1 - m) + d \times \beta \times k \times m$, where k is this control and m is **Bias mul %**. At **Bias mul %** = 1, the detection is simply $d \times \beta \times k$: with the "out" source, the compressor measures the product of the signal and the level of its own output.

A few consequences:

- when the bias is zero (at startup, in silence, or with the "g.e" source as long as there is no gain reduction), the detection is zero and the compressor doesn't compress. With **Bias mul %** at 1 and the "g.e" source, the compressor therefore never gets going: with no compression, the bias is zero, and with no bias, the detection is zero. A **Bias mul %** below 1, or a negative **Bias Feedback**, breaks the deadlock;
- the multiplication is the same on d and on −d: it creates no difference between the normal and inverse paths. Used on its own, it produces no saturation product; it changes the compression itself;
- it is applied before **Bias Fm In** and before the input height.

Negative values activate the bias, but apply no multiplication.

A word from the author

François added this option while studying the schematic of SSL compressors: in his view, their "feedback" isn't really feedback, because you don't make a copy of the signal; you take the output and multiply. To get close to an SSL compressor, he advises leaving **Bias input add** at 0 and using **Bias input mul**. The result has a very particular character.

To be confirmed by François

Bias input mul goes from −168 to +168, but only positive values multiply the detection: a negative value activates the bias (inverse path included) with no other effect. Is that intended, or should the absolute value be used?

Bias mul %

BIAS src tab

Range: 0 to 1 **Default:** 1 **ID:** `bias_ssl`

Amount of the multiplication by **Bias input mul**: a blend between the unmultiplied detection (at 0) and the multiplied detection (at 1, the default). At 0, the multiplication is off, but a nonzero **Bias input mul** still activates the bias.

The SSL bus compressor, between feed-forward and feedback

The bus compressor of SSL consoles, present according to the manufacturer as early as the 4000 B of 1976 and associated above all with the G Series of 1987, has a hybrid topology. SSL describes it this way: the side-chain itself has a feedback topology, but the signal that feeds it is taken from a feed-forward position. The paper by Giannoulis, Massberg and Reiss, which François likes to cite, describes a "slave" VCA placed in the side-chain, which reproduces the action of the main VCA. So the detector doesn't listen to the output itself, but to the input passed through a copy of the gain element. That's what François noticed while studying the schematic, and it's the idea that **Bias input mul** transposes: the detection is multiplied by a value taken from the output, instead of being replaced by it. Chapter 2 comes back to the "glue" of this compressor.

♪ In the studio: SSL-style compression

On a drum bus or a mix, the multiplication gives a compression with a very particular character; in his French walkthrough, François says that with it, "it heads off into a whole other world." Start from the recipe in the card: **Bias input add** at 0, **Bias input mul** positive, "out" source, then compare **Bias mul %** at 0 and at 1. The difference lies in the movement of the compression, in the way it grabs the attacks and lets go, rather than in any added saturation.

Seven presets use this control, with very different amounts: **cytomic_glue** (**Bias input mul** at 8), **ssl_1_f** (6.5), **ssl_2_b_** (30.52), **ssl_2_b** (38.57) and **ssl_1_b** (75), all imitations of SSL-style bus compressors; all except **ssl_2_b** also turn off the inverse path (**Bias +inverse** at 0), which lightens the CPU load. **ao_buster**, another bus compressor imitation, blends the two detections (**Bias input mul** at 13.25, **Bias mul %** at 0.308, **Bias Fm In** at 1.16). The control is also used outside this family: **kiive_kc1**, from the optical family, sets **Bias input mul** to 4.25.

Monitor Input and the bias

The **Monitor Input** monitor (INPUT tab, chapter 14) lets you hear the detection signal. When the bias is active, Comprendre removes the input height and divides by **Bias Fm In** before sending the signal to the monitor, so that you hear the detection without the bias.

> **To be confirmed by François**
>
> For **Monitor Input**, the height removed is that of the last channel, applied to all channels, and the multiplication by **Bias input mul** isn't undone. In stereo, with a per-channel bias (**Bias out unlink** at 1), the monitor therefore doesn't play exactly the detection without the bias. Is that acceptable?

Seeing the bias in the graphs

When the bias is active, the history graph (chapter 4) shows **orange** curves:

- in semi-transparent orange, the three measurements of the bias detector (peak, RMS, envelope), after their gains and powers;
- in solid orange, the final bias value.

On the transfer curve graph, **orange dots** plot the input level on the horizontal axis and the bias value on the vertical axis.

All these curves show the bias value **multiplied by |Bias input add|**, averaged over the channels: they represent the input height, on the same scale as the detection. The practical consequence: **if Bias input add is 0, they stay at zero**, even if the bias is used by **Bias input mul** or by an output height.

The **Gfx Detects Bias** control in the GRAPHS tab (chapter 4) chooses which bias curves are displayed, with the same logic as **Gfx Detects** for the detection: −2 hides them all (orange dots included), −1 keeps only the final value, 0 (default) shows everything; 1 (the three measurements and the final value) and 2 (the final value only) are **exclusive** modes, which hide the input, the output, the gain and the other curves. The compressor's detection curves, for their part, are still controlled by **Gfx Detects**. To compare the input height with the detection, set for example both **Gfx Detects** and **Gfx Detects Bias** to 2: all that's left is the final detection, in red, and the final bias, in orange.

> **A word from the author**
>
> François gives a visual rule: the orange curve shouldn't be stronger than the red curve, or the effect loses its point. It has to stay below it, and you can hear the difference.

> **To be confirmed by François**
>
> In the rule "the orange curve shouldn't be stronger than the red one," is it indeed the input height (final bias value, in orange) compared with the final detection (in red)? And should the graph also show the bias when it is used only at the destination (**Bias input add** at 0), where it currently stays at zero?

> **♪ In the studio: dialing in the input height**
>
> To dial in the input height, use both the graph and your ears. Set **Gfx Detects** and **Gfx Detects Bias** to 2, raise **Bias input add** until the orange curve approaches the red one while staying below it, then listen with **Mon bias sat** (BIAS dst tab), turning up the monitoring level if the effect is subtle. Too little, and the effect stays imperceptible; too much, and the bias is what drives the compression instead of the music.
>
> The preset values vary a lot, because the height also depends on everything that shapes the bias value (Feedback, detector gains and powers): **Bias input add** is 0.9023 in **softube_cl1b**, 1 in **pulsar_mu** and **softube_germanium**, 2.534 in **tr_670**, 7.63 in **rockruepel_comptwo_drums**, 25 in **overloud_670** and 42 in **acustica_ultramarine4**. So don't copy a number from one preset to another: look at the curves.

Example sources

The source combinations are summarized in a table in chapter 21. Here are a few starting points.

A feedback vari-mu. "out" source (**Bias g.e->out** at 1), classic measurement, positive **Bias input add**, RMS in **DET (bs)**. This is the configuration from chapter 19: the smoothed output offsets the detection; with short compressor times, the saturation shows up in the waveform.

SSL-style compression. Bias input add at 0, positive **Bias input mul**, **Bias mul %** at 1, "out" source. The detection becomes the product of the signal and the output level. Then adjust **Bias mul %** to blend this detection with the ordinary detection.

A bias that follows the compression. "g.e" source (**Bias g.e->out** at 0), envelope (**Bias g.e g->e** at 1). The bias value is the gain reduction: zero with no compression, and larger the harder the compressor works. Paired with an output height (chapter 21), it gives a saturation that only appears when the compressor reduces gain.

An inverted bias. Same configuration, with **Bias Feedback** at −1: the bias value is then the gain itself, at its maximum with no compression and decreasing as the compressor works. With the "out" source, −1 makes the bias maximal in quiet passages.

A bounded bias. To keep the height from getting too strong on the loudest passages, lower **Bias thresh** (−6 dB, for example) and set **Bias ratio** between 2 and 42: the bias value is then gently capped.

Starting ideas

- **Imitating a specific vari-mu.** Load **overloud_670** or **tr_670**, then adjust the attack and release in the A/R (bs) tab, comparing with your reference using the analysis mode (chapter 24).
- **An SSL-style mix bus.** Start from **cytomic_glue** (**Bias input mul** at 8, **Bias +inverse** at 0) and adjust **Bias mul %** to blend the multiplied detection and the ordinary detection.
- **A color that only arrives with compression.** "g.e" source with the envelope, as in **klang_mjuc_mk1**, paired with an output height (chapter 21).

- **Sound design. Bias Feedback** at −1: the bias is at its maximum in silence and fades away as the signal rises. Try it with an output height and the bias limiter (chapter 21): the height, and therefore the width of the bend, is then greatest in quiet passages. Listen to the result with **Mon bias sat**.

CHAPTER 21

Destination and Fusion

This is where the bias becomes sound. The principle fits in an image François uses in his introduction video: you add “height” to the signal, and at some point it bumps against the top. Because the signal and its inverted copy don’t bump into it on the same side, each one saturates in its own way, and recombining them produces a very particular kind of saturation, which can grow with the compression or, on the contrary, fade away as the compression sets in. This simple mechanism yields a surprisingly wide palette: François reckons that about ten truly distinct types of saturation can be drawn from it, not counting the variants of the “atom” measurement (chapter 20). Among them you’ll find the grain of a tube compressor that only shows up when the unit is working, a saturation stage that thickens a signal without compressing it, even- or odd-order clipping, and new textures for sound design.

Eleven of the bundled presets set the **BIAS dst** tab. Six of them imitate vari-mu tube compressor plugins: three emulations of the Fairchild 670, one of the Manley Variable Mu, one of the Gates Sta-Level, and one inspired by the vari-mu units of the 1950s. The other five imitate a clipper, two saturators (one of which doesn’t compress at all), an optical tube limiter and a FET compressor. In mixing, this tab gives a vocal, a bass or a bus the density of a tube compressor, dialed in with a knob; in mastering, a compensated drive and a soft limiter add body without changing the level; in sound design, the fusion, the inverse path and the product delay open up sounds you won’t get anywhere else in the plugin.

Chapter 20 showed how Comprendre computes the bias value β and how it acts on the detection. This chapter describes what it does to the **audio signal**: the **output height**, its position before or after the compressor gain, the **Bias Fm On** factor, the bias **limiter**, the **fusion** of the normal and inverse paths, and then the **saturation product** and its processing in the **F (bs)** tab. It ends with tables that sum up the possible combinations, on the source side as well as on the destination side.

The notation (x, e, e′, β, B, s, s′, O, s_pre, P) is the one from chapter 19. Remember that everything that follows is inactive as long as the five bias amounts are at 0.



Figure 119 — The BIAS dst tab: monitor, output height, Fm On, position, limiter, fusion and product delay.

The order of operations

On each path, the destination runs through the following steps:

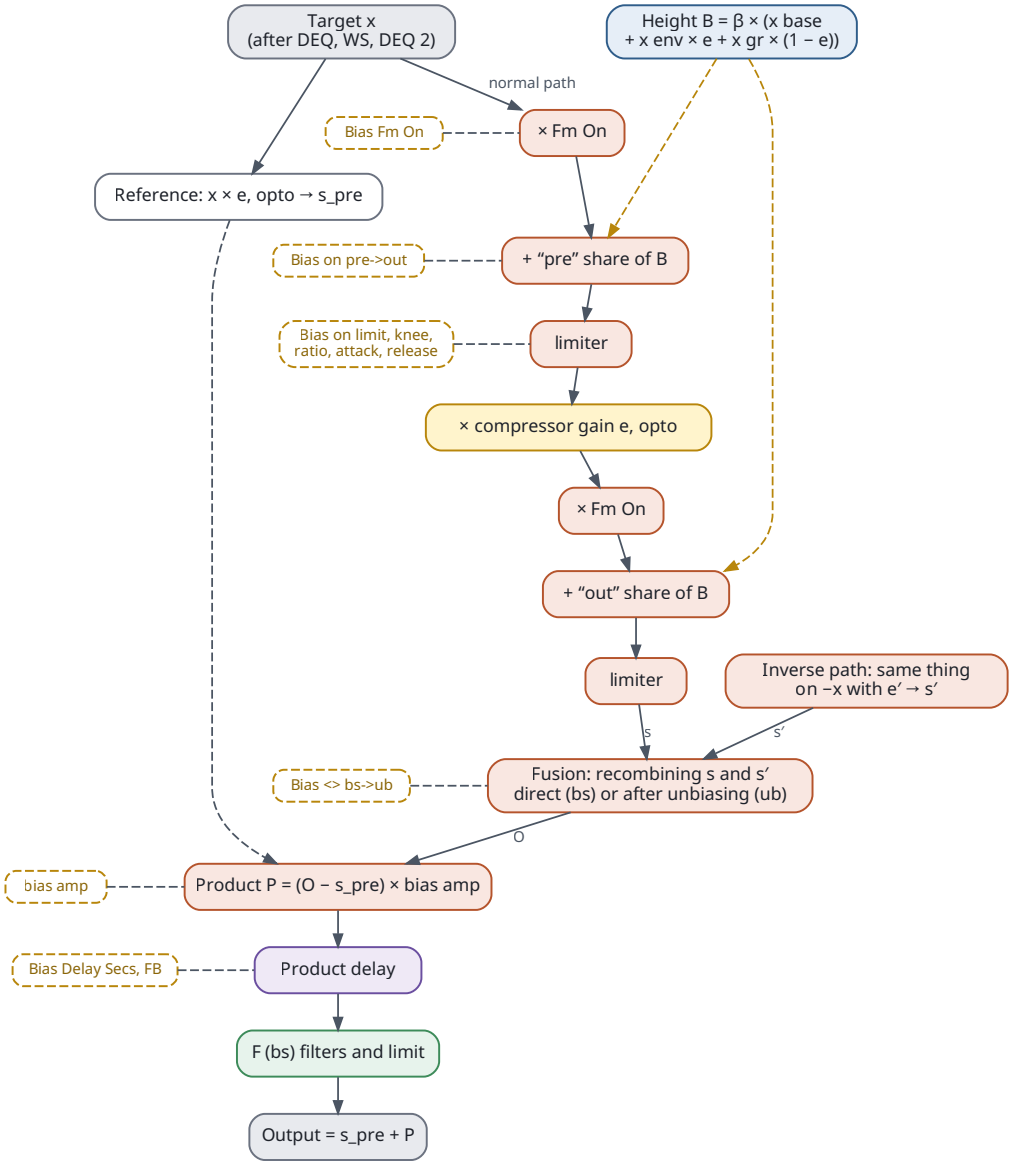


Figure 120 — The bias destination block. The height B is split between the “pre” position (before the compressor gain) and the “out” position (after the gain and the opto); each position has its own Fm On factor and its own limiter. The two paths are then fused; the difference from the reference s_pre is the saturation product, which is processed by F (bs) and then added back to s_pre.

1. Computing the **output height** B = β × (**Bias x base** + **Bias x env** × e + **Bias x gr** × (1 − e)).
2. **“Pre” part**, on the target, before the compressor gain: multiplication by **Bias Fm On**, addition of the “pre” share of the height, limiter.
3. Application of the **compressor gain** and the **opto**.
4. **“Out” part**, on the output: multiplication by **Bias Fm On**, addition of the “out” share of the height, limiter.
5. The same thing on the **inverse path**, with the gain e′ and the **same** height B.
6. **Fusion**: recombining the two paths, either directly (“bs”) or after removing the heights and dividing by **Bias Fm On** (“ub”).
7. **Product**: P = (O − s_pre) × **bias amp**, then the delay, filters and limit of **F (bs)**, and finally output = s_pre + P.

The output then goes, just as it does without bias, through **Comp Clip**, **Comp Db**, the addition of the remainder of the signal, the monitors and the rest of the chain (chapter 7).

The reference s_pre is the output of the normal path, gain and opto included, but **without** any of steps 2 and 4. That doesn't make it "bias-free," though: its gain e was computed with the biased detection (input height, **Bias input mul**, **Bias Fm In**).

Hearing what the bias does

The first control in the tab is a monitor. It's useful from the very start of your adjustments: most of the bias's effects are subtle, and setting it up is much easier when you listen to the product on its own.

Mon bias sat

BIAS dst tab

Range: 0 to 256 **Default:** 0 **ID:** `delta_bias`

Lets you hear the bias's **saturation product**, that is, the difference between the output and the reference s_pre.

- **0** (default): normal listening.
- **1**: you hear the output minus s_pre, in other words the product P after all the processing of the **F (bs)** tab. If **Comp Clip**, **Comp Db**, **dW | Dw** and the target filters are neutral, this is exactly P; otherwise, their effects and the remainder of the signal (the untargeted part, chapter 14) are added to it.
- **From 2 to 256**: the same thing, amplified N times: 2 gives +6 dB, 4 gives +12 dB, 16 gives +24 dB, 256 gives +48 dB. The product is often faint, and this amplification lets you hear it.

The control only takes whole-number values. Like the other monitors, it's exclusive (turning one on turns the others off), and it disables the post-clipper while you listen (chapter 7).

This monitor doesn't let you hear everything the bias changes, only the saturation it produces: the effect of the input height on the compression itself is already in s_pre. With **Bias +inverse** at 0 and an input height alone, the monitor stays silent even though the sound has clearly changed (chapter 19).

> **A word from the author**
>
> François is emphatic: this monitor doesn't play the overall difference caused by the bias, only the saturation generated by the bias. To judge the overall effect, compare by ear against the five amounts set to 0.

The output height

The output height is an offset added to the audio signal of both paths. It's the sum of three terms, all proportional to the bias value β:

B = β × **Bias x base** + β × **Bias x env** × e + β × **Bias x gr** × (1 − e)

where e is the compressor gain of the **normal** path (1 = no reduction). In stereo, e is the linked gain, the channel's own gain, or a mix of the two, depending on **Channels Unlink**. The three terms add up, and the same height is added to both paths.

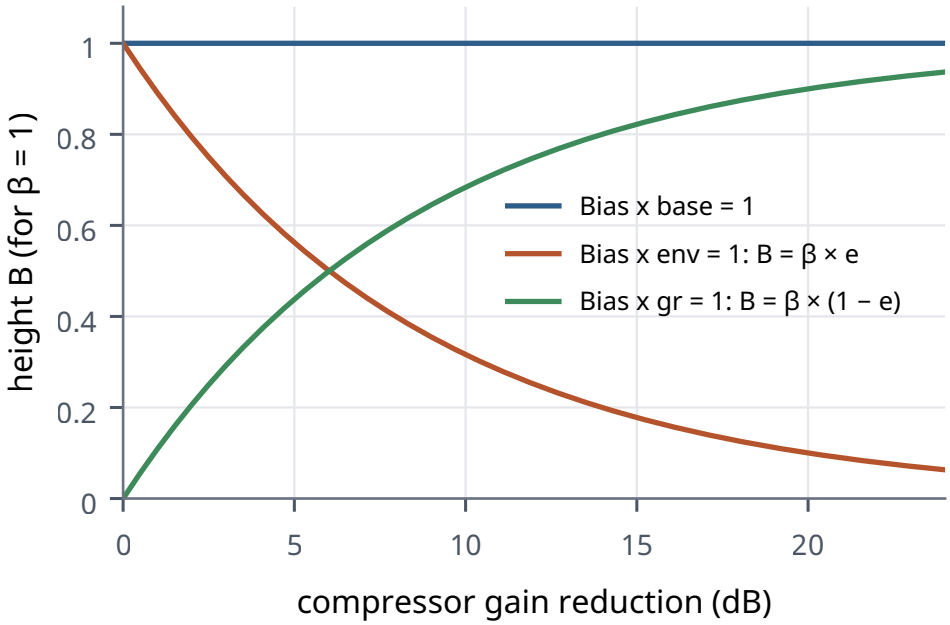


Figure 121 — Height produced by each of the three controls, set to 1, for a bias value β = 1, as a function of the compressor's gain reduction.

The height is expressed in linear amplitude, like the signal: 1 corresponds to 0 dBFS. With β = 0.3 and **Bias x base** at 1, the signal is offset by 0.3, or about −10 dBFS.

Bias x base

BIAS dst tab

Range: −42 to 42 **Default:** 0 **ID:** `bias_output_base`

Output height proportional to the bias value alone: B = β × this control. It's one of the five amounts that activate the bias.

- **0** (default): no "base" term.
- **Positive**: the signal of both paths is shifted upward; the positive peaks of the normal path and the negative peaks of the inverse path move closer to the limiter.
- **Negative**: the signal is shifted downward; the roles of the half-cycles are swapped.

Like the other two terms, with **Bias +inverse** at 1 it only has an audible effect if the limiter is active or if the two paths have different gains (input height): otherwise the height, identical on both sides, cancels out in the recombination.

Bias x gr

BIAS dst tab

Range: −42 to 42 **Default:** 0 **ID:** `bias_output_gr`

Output height proportional to the bias value **and to the compressor's gain reduction**: B = β × this control × (1 − e). It's one of the five amounts that activate the bias.

Without compression, this term is zero; it grows as the compressor reduces the gain: 0.5 × β × control for 6 dB of reduction, 0.9 × β × control for 20 dB. This is the closest picture of a tube whose biasing is driven by the control voltage: the saturation only appears when the compressor is working, and the harder it works, the more saturation you get.

Bias x env

BIAS dst tab

Range: −42 to 42 **Default:** 0 **ID:** `bias_output_env`

Output height proportional to the bias value **and to the compressor's gain**: B = β × this control × e. It's one of the five amounts that activate the bias.

This is the reverse of **Bias x gr**: the height is at its maximum without compression and decreases as the compressor reduces the gain (0.5 × β × control for 6 dB of reduction). The saturation is then more present in quiet passages than in compressed ones.

A word from the author

The destination height is a nominal value, multiplied "by nothing, by the gain reduction, or by the envelope." The three modes can be combined: you can, for example, keep a small fixed height with **x base** and add a share on top that follows the compression with **x gr**.

♪ In the studio: saturation that follows the compression

The three height terms serve three intentions. With **Bias x gr**, the height only exists when the compressor is working: a vocal or a bass stays clean in quiet passages and picks up grain when it pushes, like a tube compressor you're working hard. With **Bias x env**, it's the opposite: the grain dresses up the quiet passages and fades away when the compression sets in. Three presets imitate emulations of the Fairchild 670, and each one goes about it differently. **tr_670** sets **Bias x gr** to 0.3333 and puts the limiter at 1 (0 dBFS); **acustica_ultramarine4** raises **Bias x gr** to 2.5, with no limiter, relying on a strong input height; **overloud_670** instead opts for **Bias x env** at −1, in the "out" position, with a very soft limiter (**Bias on limit** at 1, **Bias limit knee** at 6.92, **Bias on ratio** at 1.3). For your own experiments, start with **Bias x gr** between 0.3 and 1 and **Bias on limit** at 1, as in **tr_670**, listen to the product with **Mon bias sat** on a passage that compresses, then dial it in.

Before or after the gain: pre and out

Bias on pre->out

BIAS dst tab

Range: 0 to 1 **Default:** 0 **ID:** `bias_cible`

Chooses where the output height, **Bias Fm On** and the limiter are applied.

- **0** (default, “pre”): on the **target**, before the compressor gain. The shifted signal goes through the limiter, then it’s multiplied by the gain e and goes through the opto. The height itself is therefore multiplied by the gain: it becomes B × e at the output. The limiter sees the uncompressed signal.
- **1** (“out”): on the **output**, after the compressor gain and the opto. The height is added as is, and the limiter sees the compressed signal.
- **In between**: the height and the deviation of **Bias Fm On** from 1 are split between the two positions, and the limiter acts in both places, each with its own smoothing state.

The choice changes the character of the saturation: in “pre,” it depends on the input level and the compressor then acts on the result; in “out,” it depends on the output level, after compression.

Bias Fm On: a drive before the limiter

Bias Fm On

BIAS dst tab

Range: 0.000001 to 8 **Default:** 1 **ID:** `bias_fm_out`

Multiplies the signal, on both paths, just **before** the output height is added, in the position or positions chosen by **Bias on pre->out**. At an intermediate position p, the factor applied in “pre” is 1 + (Fm On − 1) × (1 − p) and the one applied in “out” is 1 + (Fm On − 1) × p.

- **1** (default): neutral.
- **Above 1**: the signal is amplified before the limiter and reaches it sooner: this is a **drive**.
- **Below 1**: the signal is attenuated, and the limiter steps in less.

This control acts as soon as the bias is active, even if the height is zero. Its effect on the level depends on the fusion:

- with direct fusion (**Bias** <> **bs->ub** at 0), the signal stays multiplied: the output is roughly Fm On times louder;
- with unbiased fusion (**Bias** <> **bs->ub** at 1), the signal is divided by Fm On again before the recombination: the level is compensated, and only the nonlinear effect remains (the limiter being hit harder).

The same factor enters into the “ub” measurement of the source (**Bias out bs->ub**, chapter 20).

A word from the author

In his demonstration of the fusion, François raises the output saturation, then plays with **Bias Fm On**: with direct fusion, the level changes dramatically; by unbiasing before the fusion, you get your sound back, but with the saturation.

To be confirmed by François

Bias Fm On goes down to 0.000001. With unbiased fusion, the signal is divided by this value: whatever is left of the height after unbiasing (for example with **Bias** +**inverse** at 0, or with different gains on the two paths) can then be amplified enormously, and **bias dst limit** doesn't limit anything by default. Is a safeguard needed?

♪ In the studio: the bias as a saturation stage

Bias Fm On, the limiter and the fusion turn the bias into a genuine saturation stage, one you can use without compressing anything. The **pulsarmodular_p42** preset, an imitation of the Pulsar Modular P42 Climax, a saturating line amplifier, shows this well: the compressor is neutralized (**Ratio** at 1, **Threshold** at 0 dB) and everything happens in the **BIAS dst** and **F (bs)** tabs. The signal is driven by a factor of 1.525 (**Bias Fm On**) into a soft limiter placed after the gain (**Bias on pre->out** at 1, **Bias on limit** at 0.097, or about −20 dBFS, **Bias on ratio** at 3.91), then the effect is blended in with **bias amp** at 0.475. Notice **Bias x base** at 0.000019: a tiny value whose only purpose is to wake the module up (chapter 19). The **pulsar_mu** preset shows the other half of the idea: it doubles the signal (**Bias Fm On** at 2) and then divides it back down through unbiased fusion (**Bias <> bs->ub** at 1), so the level barely moves. On a bus or a master, this compensation lets you judge the color on its own, without being fooled by a jump in volume.

The bias limiter

The bias limiter is the piece that turns the height into saturation. It's applied at each active position ("pre" and/or "out") and on each path. It acts on the absolute value of the signal while preserving its sign, so its ceiling sits at ±**Bias on limit**, symmetrical around 0. But because the signal has been shifted by the height, it reaches this ceiling sooner on one side than on the other: on the normal path for the positive half-cycles, on the inverse path for the negative ones. After recombination, you get the "push-pull" bend described in chapter 19: a slope of 1 up to T − B, a slope of ½ up to T + B, and the ceiling beyond that.

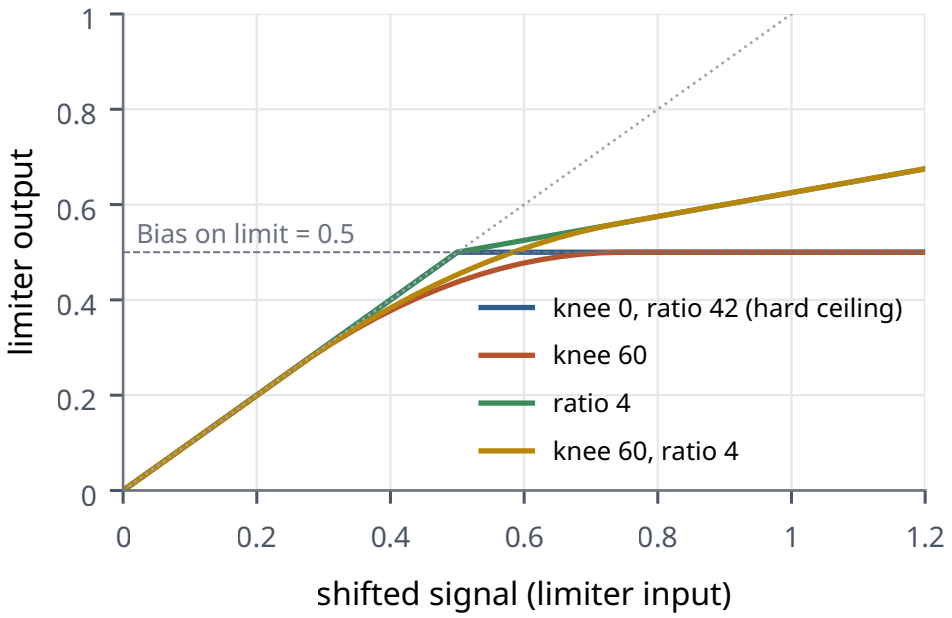


Figure 122 — Bias limiter curves for a threshold of 0.5: hard ceiling, knee, ratio, and the two combined.

The limiter is active as soon as **Bias on limit** or **Bias limit knee** is greater than 0. With **Bias on attack** and **Bias on release**, it becomes a compressor.

Bias on limit

BIAS dst tab

Range: 0 to 42 **Default:** 0 **ID:** `bias_limite`

Threshold of the bias limiter, in **linear amplitude** (not in dB): 1 corresponds to 0 dBFS, 0.5 to about −6 dBFS, 0.1 to −20 dBFS.

- **0** (default): the limiter is disabled, unless **Bias limit knee** is greater than 0.
- **Above 0**: the shifted signal is limited to ± this value, according to the knee and the ratio.

In the “pre” position, the threshold applies to the uncompressed signal (plus the height); in the “out” position, to the compressed signal. With a height B, the saturation starts around **Bias on limit** − B and becomes complete around **Bias on limit** + B.

Bias limit knee

BIAS dst tab

Range: 0 to 240 **Default:** 0 **ID:** `bias_limite_genou`

Width of the limiter’s knee, in amplitude: the displayed value divided by 120. At 60, the bend spans 0.5, from 0.25 below the threshold to 0.25 above it. At **0** (default), the bend is sharp.

A nonzero knee activates the limiter even if **Bias on limit** is 0.

To be confirmed by François

When half the knee width exceeds the threshold (**Bias limit knee** / 240 greater than **Bias on limit**, for example **Bias on limit** at 0 and **Bias limit knee** nonzero), the limiter no longer returns 0 for a zero signal: its output jumps abruptly at the zero crossing (±0.0625 for a threshold of 0 and a knee of 60). This produces a buzz or noise. Is this intended? Is it the source of the "new kind of white noise" in the fusion video?

Bias on ratio

BIAS dst tab

Range: 1 to 42 **Default:** 42 **ID:** `bias_limite_ratio`

Ratio of the limiter. Above the threshold (and the knee), the part of the signal that exceeds it is divided by this ratio.

- **42** (default): infinite ratio; the signal is capped at the threshold (hard clipping, or rounded off by the knee).
- **Between 1 and 42**: soft limiting; the signal keeps rising above the threshold, but more slowly.
- **1**: the limiter is transparent.

Bias on attack

BIAS dst tab

Range: 0 to 2 s **Default:** 0 s **ID:** `bias_limite_attaque`

Attack time of the limiter. When **Bias on attack** or **Bias on release** is greater than 0, the limiter no longer applies instantaneously: its **gain** (the ratio between the limited signal and the original signal) is smoothed, and the limiter becomes a small compressor, with **Bias on limit** as its threshold and **Bias on ratio** as its ratio, applied to the shifted signal.

The attack sets how fast this gain **falls**: the gain covers about 63% of the gap in this time. During the attack, the peaks overshoot the threshold. At **0** (default), the fall is instantaneous.

Each position and each path has its own smoothing state, which starts at 1 (no reduction).

Bias on release

BIAS dst tab

Range: 0 to 2 s **Default:** 0 s **ID:** `bias_limite_relachement`

Release time of the limiter: how fast its gain **rises** again when the signal drops back below the threshold. At **0** (default), the rise is instantaneous. Either of the two times being greater than 0 is enough to activate the smoothing.

A word from the author

You can turn this limiter into a compressor with the attack and release, which lets you tame saturation that gets too intense.

> **♪ In the studio: from a clean shave to a gentle touch**
>
> The bias limiter has several faces. With no knee and no ratio, it shaves peaks clean off: **kive_filk2_saturation**, an imitation of a console channel-strip saturation module, makes do in this tab with **Bias on limit** at 0.25 (about −12 dBFS) and no output height. At the other extreme, **klang_mjuc_mk1**, an imitation of a plugin inspired by the vari-mu units of the 1950s, puts the ceiling very high (**Bias on limit** at 8) and pushes the height very far (**Bias x base** at −30.3). In between, the knee rounds off the bend, a ratio of 2 to 4 lets the signal keep rising above the threshold, and the attack turns the limiter into a small compressor. On drums, try an attack of a few milliseconds: during that time, the peaks overshoot the threshold, and the snap of the transients gets through before the limiter rounds it off.

Fusion: unbiasing before recombining

When the two paths are brought back together, there are two possible choices:

- recombine them **directly** (“bs,” for *biased*): O = (s − s′) / 2 with **Bias** +**inverse** at 1;
- **unbias** them first (“ub,” for *unbiased*): on each path, the “out” share of the height is removed and the result is divided by the “out” factor of **Bias Fm On**, then the “pre” share of the height is removed and the result is divided by the “pre” factor; only then are the paths recombined.

Unbiasing doesn’t undo the saturation: what the limiter clipped stays clipped. It only removes the offset and the scale factor that were deliberately added.

Bias <> bs->ub

BIAS dst tab

Range: 0 to 1 **Default:** 0 **ID:** `bias_fusion`

Chooses how the normal and inverse paths are fused.

- **0** (default, “bs”): direct recombination of the biased paths.
- **1** (“ub”): recombination after unbiasing.
- **In between**: a linear mix of the two results.

Its effect depends heavily on **Bias +inverse**:

- **with Bias +inverse at 1**, the height cancels out anyway in the difference between the two paths: unbiased fusion merely divides the result by **Bias Fm On**. With **Bias Fm On** at 1, this control has **no audible effect**;
- **with Bias +inverse at 0**, unbiased fusion removes the height. In the “out” position, the output goes back to being the compressed signal, and all that remains is the limiter’s asymmetric clipping (nothing at all without the limiter). In the “pre” position, a residue equal to −B × (1 − e) / Fm On remains: an offset that follows the gain reduction;
- **in between**, it removes the share of the height that doesn’t cancel out, B × (1 − i).

The fusion applies to the audio signal, but also to the target and to the pre-opto signal, as heard through **Monitor Output** and **Monitor opto**, and to the compression difference sent back to the detection by **FX fb cmp** (INPUT tab, chapter 14). The bias value itself has its own equivalent control, **Bias out bs->ub** (chapter 20).

A word from the author

The fusion question is simple: do you recombine the two paths directly, or do you unbias them first? You might think it amounts to the same thing, but that isn’t always the case, because the two paths have gone through processing between the moment they were biased and the moment they’re recombined.

To be confirmed by François

In the “pre” position, unbiasing removes the height B, whereas the output contains B × e (the height has been multiplied by the compressor gain). With **Bias +inverse** at 0, or with different gains on the two paths, an offset of −B × (1 − e) / Fm On therefore remains, and it follows the gain reduction. Is this intended, as part of the saturation product, or should B × e be removed instead?

The ‘thump’ of unbalanced push-pull stages

As early as the first radio peak limiters, in 1937, the control voltage was mixed with the audio on the grid of a variable-mu tube, and the push-pull circuit canceled it out at the output. But as the tubes aged and drifted out of balance, the cancellation was no longer perfect and “thumps” appeared. The Fairchild 670 even had a **Balance** control to balance its two halves (chapter 2). In Comprendre, this balance becomes a control like any other: with **Bias +inverse** below 1, part of the height no longer cancels out and, with direct fusion, passes into the output; unbiased fusion removes it (completely in the “out” position). So you can keep a perfect push-pull, or deliberately dial in some imbalance.

The combinations

The bias controls combine into a very large number of configurations, but many of them are equivalent. The following three tables sum up what each combination does, within a simplified framework: a single channel (or linked channels), neutral opto, dynamic EQs and waveshaper, and paths that are otherwise symmetrical (the inverse path processes exactly −x). The formulas are exact within this framework.

Source side: what becomes of the bias value

Value passed to the Feedback, then to the bias detector, for each channel. ŝ and ŝ′ denote the outputs of the two paths after unbiasing; G and G′ the gain (static or envelope) of the two compressors; "–" marks a control that has no effect in that row.

Bias g.e->out	Bias g.e g->e	Bias +inverse	Bias out atom	Bias out bs->ub	Value passed	Notes
0	0 (g)	0	–	–	1 − g	static gain reduction, with no attack or release: rough
0	1 (e)	0	–	–	1 − e	smoothed gain reduction: smooth
0	0 or 1	1	–	–	1 − (G + G′) / 2	average of the two paths; G and G′ only differ with an input height
1	–	0	0 or 1	0 (bs)	|s|	contains the height (B × e in "pre," B in "out"): the bias feeds itself back
1	–	0	0 or 1	1 (ub)	|ŝ|	unbiased output
1	–	1	0	0 or 1	|s − s′| / 2	the height cancels out; bs and ub identical if **Bias Fm On** = 1
1	–	1	1	0 (bs)	(|s| + |s′|) / 2	"atom": sees the height
1	–	1	1	1 (ub)	(|ŝ| + |ŝ′|) / 2	unbiased atom
1	–	0 or 1	0.5	–	max(O, 0)	half-wave rectification (symmetrical paths)

With **Bias out unlink** at 0, the value of each "out" row is replaced by its average across the channels. Not counting intermediate values, the "g.e" source offers four distinct configurations (static gain or envelope, with or without the inverse path). For the "out" source, several combinations collapse into one: without the inverse path, the classic and atom measurements are identical; with the inverse path and the classic measurement, bs and ub only differ if **Bias Fm On** is something other than 1.

Detector side: input height and multiplication

Control	Bias +inverse = 0	Bias +inverse = 1
Bias input add alone	detection of |d + B|: asymmetric compression; the product P is **zero**, since the asymmetry is part of s_pre	e ≠ e′: P = x × (e′ − e) / 2; the DC component and the even harmonics of the normal path are removed
Bias input mul alone	detection of d × (1 − m + m × β × k): multiplicative compression; P is zero	the multiplication is the same on both paths: e = e′ and **P is zero**
add and mul	the multiplication is done before the height is added	same, with the asymmetry due to the height
Bias Fm In ≠ 1	gain of the detection signal relative to the height	same

Destination side: the output by inverse, position and fusion

F denotes **Bias Fm On**, L the limiter, B the output height. The “Without limiter” column gives the output O when the limiter is off.

Bias +inverse	Position	Fusion	Output O	Without limiter	What you get
0	pre	bs	L(F·x + B) · e	F·x·e + B·e	gain ×F, offset B·e that follows the gain, asymmetric clipping with L
0	pre	ub	(L(F·x + B) · e − B) / F	x·e − B·(1 − e) / F	clean signal, plus a residue that follows the gain reduction; asymmetry with L
0	out	bs	L(F·x·e + B)	F·x·e + B	offset B, asymmetric clipping after the gain
0	out	ub	(L(F·x·e + B) − B) / F	x·e	only the asymmetric clipping (nothing without a limiter)
1	pre	bs	[L(F·x + B)·e − L(−F·x + B)·e′] / 2	F·x·(e + e′)/2 + B·(e − e′)/2	push-pull bend with L; B term only if the detectors differ
1	pre	ub	same value, divided by F	x·(e + e′)/2 + B·(e − e′)/(2F)	identical to the previous row if F = 1
1	out	bs	[L(F·x·e + B) − L(−F·x·e′ + B)] / 2	F·x·(e + e′)/2	the height disappears without a limiter; push-pull after the gain with L
1	out	ub	same value, divided by F	x·(e + e′)/2	identical to the previous row if F = 1

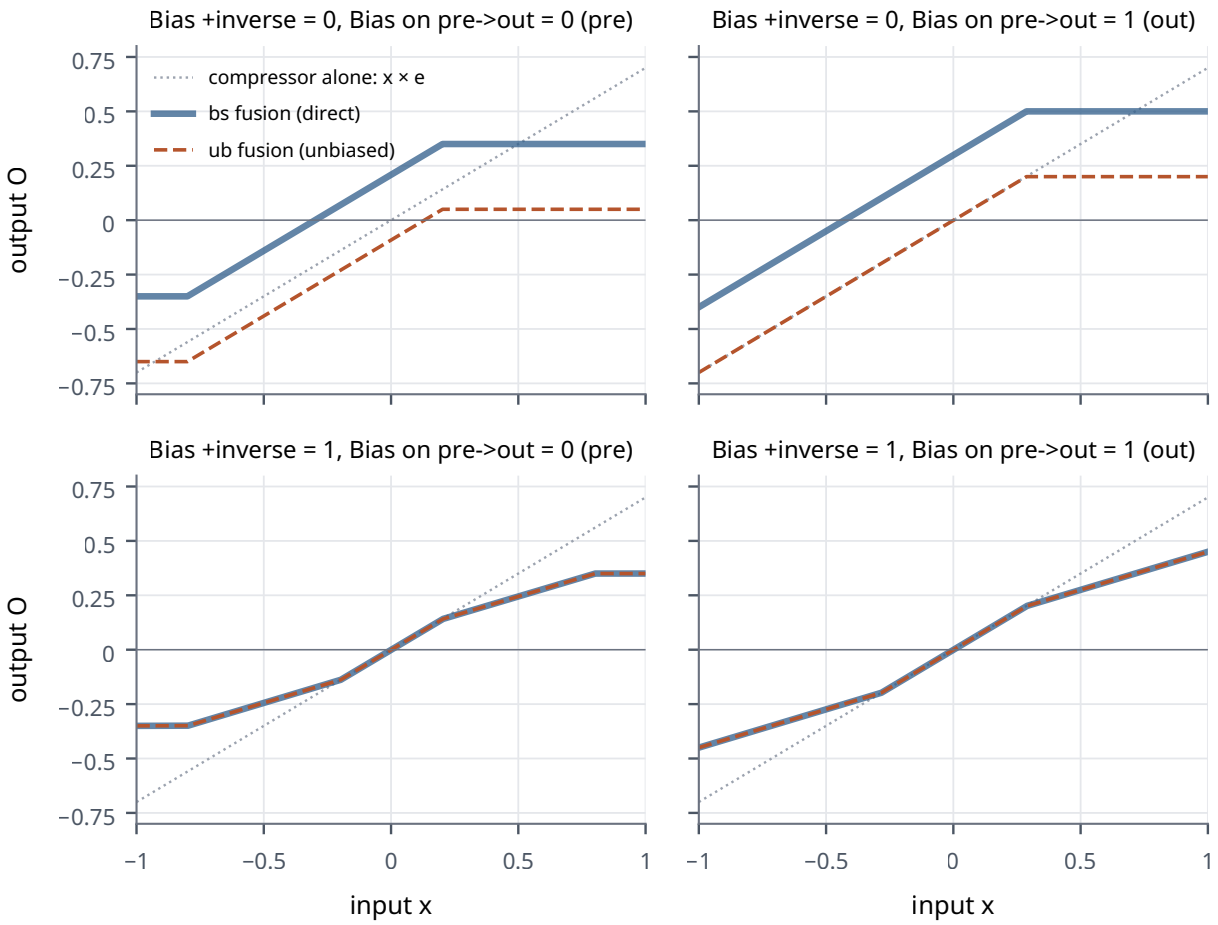


Figure 123 — The destination combinations, for a compressor gain fixed at 0.7 on both paths, a height of 0.3, a hard ceiling at 0.5 and Bias Fm On at 1. Top, without the inverse path: direct fusion shifts the curve, unbiased fusion recenters it (in "out") or leaves a residue (in "pre"). Bottom, with the inverse path: both fusions give exactly the same symmetrical curve, with its push-pull bend.

Key takeaways:

1. With **Bias +inverse** at 1, bs or ub fusion is nothing more than a 1/F factor: it serves to compensate for the **Bias Fm On** drive.
2. With **Bias +inverse** at 1 and no limiter, an output height only does something if the two paths have different gains, that is, with an input height (or with effects that differ between the paths).
3. The product due to the input height alone, with **Bias +inverse** at 1, is $P = x \times (e' - e) / 2$: it's the opposite of the DC component and even harmonics of the normal path (chapter 19).
4. When the output height is used alone, the **limiter** is the only source of "height" saturation.
5. Without the inverse path and with direct fusion, the height ends up in the output as an offset. To get rid of it, use the inverse path, unbiased fusion (which removes it completely in the "out" position) or a high-pass filter on the product.

A word from the author

François drew up a table of the destination combinations: depending on the base, the multiplication by the gain reduction, unbiasing before the fusion, **Bias Fm In** and the source, he counts 32 combinations, of which only about ten truly stand apart in the type of saturation product they produce, not even counting the atom, which adds more.

To be confirmed by François

The "table of 32 combinations" announced in the video isn't in the plugin. Could you provide a link to it or its content, and specify the five binary choices that make it up? The tables above show, for example, that with **Bias +inverse** at 1 and **Bias Fm On** at 1, bs or ub fusion changes nothing: that would explain why only about ten of the 32 combinations are useful.

The bias saturation product

The bias saturation product is the difference between the recombined output and the reference:

P = (O − s_pre) × **bias amp**

It then goes through the chain shared by the three saturators (chapter 16): a feedback delay (set in the **BIAS dst** tab), three filter bands, a mix and a limit (**F (bs)** tab). Finally, it's added back to s_pre. With the default settings, this chain is neutral and the output is exactly O.

When the bias value is taken from the output, it's measured **before** this chain: no product control changes β.

The product delay

The two delay controls are in the **BIAS dst** tab. They work exactly like those of the waveshaper and the opto: it's a feedback comb filter, described in detail in chapter 16.

Bias Delay Secs

BIAS dst tab

Range: 0 to 2 s **Default:** 0 s **ID:** `bias_delai_nb`

Delay time of the bias product's feedback loop. It only acts if **Bias Delay FB** is something other than 0. The effective delay is one internal sample longer than the set value; at 0, only one sample remains and the delay becomes a one-pole filter (chapter 16).

Bias Delay FB

BIAS dst tab

Range: −64 to 64 **Default:** 0 **ID:** `bias_delai_fb`

Feedback of the product delay. At **0** (default), the delay is disabled. Otherwise, the incoming product is multiplied by 1/(1 + |FB|) and the delayed output, fed back, by 1 − 1/(1 + |FB|), with the sign of FB. For example, at 1: 0.5 and 0.5; at −3: 0.25 and −0.75. Positive values reinforce the multiples of 1/delay, negative values the frequencies in between (chapter 16).

The F (bs) tab



Figure 124 — The F (bs) tab.

The **F (bs)** tab has the same controls as **F (ws)**, described in detail in chapter 16 (cards **fet amp**, **fet src mix**, **fet low herz**…). The cards below only spell out what's specific to the bias.

bias amp

F (bs) tab

Range: −42 to 42 **Default:** 1 **ID:** `bias_amp`

Multiplies the bias product before the delay, the filters and the limit.

- **1** (default): you hear the recombined output O. With **Bias +inverse** at 1, this is the odd part of the biased path.
- **0**: you hear s_pre, the normal path without any height on the audio. The bias's effects on the detection (input height, **Bias input mul**, **Bias Fm In**) are still there, since they determine the compressor gain.
- **Between 0 and 1**: sets the amount of bias saturation.
- **−1**: you hear 2 × s_pre − O: the bias saturation is flipped. With an input height alone and **Bias +inverse** at 1, the DC component and even harmonics of the normal path are then doubled instead of being removed.
- **Beyond 1, or more negative than −1**: the effect is exaggerated.

The value 1 sits at about four-fifths of the knob's travel: for precise values, type them in.

bias src mix

F (bs) tab

Range: 0 to 1 **Default:** 1 **ID:** `bias_src_mix`

Proportion of the unfiltered product in the processed product (chapter 16, **fet src mix**). At **0**, with no active band, the product is removed and the output reverts to s_pre.

bias low herz

F (bs) tab

Range: 0 to 16000 Hz **Default:** 0 Hz **ID:** `bias_bas_coupe`

Cutoff frequency of the low band (low-pass) of the bias product. At **0** (default), the band is off (chapter 16, **fet low herz**).

Without the inverse path, the bias product often contains the height itself, in the form of a slow offset. A low band at a few tens of hertz, in ZDF (**bias low zdf->svf** at 0) with **bias low mix** at −1, removes it while letting the rest of the product through.

bias low reso

F (bs) tab

Range: 0.000001 to 16 **Default:** 0.707107 **ID:** `bias_bas_resonance`

Resonance of the low band's SVF low-pass; no effect on the ZDF part (chapter 16, **fet low reso**).

bias low mix

F (bs) tab

Range: −8 to 8 **Default:** 0 **ID:** `bias_bas_mix`

Coefficient with which the low band is added to the product, from −8 to +8 (chapter 16, **fet low mix**). At 1, the band is added once; at −1, it's subtracted.

bias low zdf->svf

F (bs) tab

Range: 0 to 1 **Default:** 1 **ID:** `bias_bas_type`

Blend between the one-pole ZDF low-pass (at 0) and the two-pole SVF low-pass (at 1, default) of the low band (chapter 16, **fet low zdf->svf**).

bias mid herz

F (bs) tab

Range: 0 to 16000 Hz **Default:** 0 Hz **ID:** `bias_mid_coupe`

Center frequency of the mid band (SVF band-pass) of the product. At **0** (default), the band is off (chapter 16, **fet mid herz**).

bias mid reso

F (bs) tab

Range: 0.000001 to 16 **Default:** 0.707107 **ID:** `bias_mid_resonance`

Resonance of the mid band-pass: it sets the band's width, and also its gain at the center frequency, which is equal to it (chapter 16, **fet mid reso**).

bias mid mix

F (bs) tab

Range: −8 to 8 **Default:** 0 **ID:** `bias_mid_mix`

Coefficient with which the mid band is added to the product, from −8 to +8 (chapter 16, **fet mid mix**).

bias hgh zdf->svf

F (bs) tab

Range: 0 to 1 **Default:** 1 **ID:** `bias_top_type`

Blend between the ZDF high-pass (at 0) and the SVF high-pass (at 1, default) of the high band (chapter 16, **fet hgh zdf->svf**).

bias hgh herz

F (bs) tab

Range: 0 to 16000 Hz **Default:** 0 Hz **ID:** `bias_top_coupe`

Cutoff frequency of the high band (high-pass) of the product. At **0** (default), the band is off (chapter 16, **fet hgh herz**).

bias hgh reso

F (bs) tab

Range: 0.000001 to 16 **Default:** 0.707107 **ID:** `bias_top_resonance`

Resonance of the high band's SVF high-pass; no effect on the ZDF part (chapter 16, **fet hgh reso**).

bias hgh mix

F (bs) tab

Range: −8 to 8 **Default:** 0 **ID:** `bias_top_mix`

Coefficient with which the high band is added to the product, from −8 to +8 (chapter 16, **fet hgh mix**). A negative value softens the high-end grain of the bias saturation.

bias dst limit F (bs) tab

Range: 0 to 42 **Default:** 42 **ID:** `bias_dst_limit`

Hard clipping of the processed product, at ± this value in linear amplitude (chapter 16, **fet dst limit**). At **42** (default), the limit is disabled; at **0**, the product is removed and the output reverts to s_pre.

For the bias, this limit is also a **safeguard**: the amounts go up to 42, and a poorly controlled height (without the inverse path, or with unbiased fusion and a very small **Bias Fm On**) can produce very high levels. A limit of 1, or even less, protects your monitoring while you explore.

Watch your levels

Bias heights are expressed in linear amplitude and can reach 42 times the bias value. Without the inverse path, or when you play with **Bias Fm On**, the output can contain an offset or a level far above 0 dBFS. Start with small amounts (0.1 to 1), listen with **Mon bias sat**, and set **bias dst limit** to 1 while you explore.

♪ In the studio: sculpting the grain

The bias product can be shaped like raw material. To hear it on its own, set **Mon bias sat** to 16 (+24 dB): many subtle settings then become obvious. The **F (bs)** filters then decide its color. **softube_cl1b** adds the product's own top end back into it, above 3 kHz (**bias hgh herz** at 3,000 Hz, **bias hgh mix** at 0.75); **arturia_tube-sta** keeps only three bands of the product (**bias src mix** at 0; bands at 80 Hz, 160 Hz and 8 kHz, weighted 1, 0.5 and 0.2), favoring the low end; **softube_germanium** flips the product and halves it (**bias amp** at −0.5). Finally, the product delay opens the door to sound design: **acqua_ash-ultra**, an imitation of a clipper, is the only preset that uses it (**Bias Delay Secs** at about 14.5 ms, **Bias Delay FB** at 0.19), which reinforces multiples of about 69 Hz in the product.

Practical workflows

A push-pull bend, without a detector. To hear the effect of the output height and the limiter alone, you need a constant bias value. A simple way to get one: in **DET (bs)**, set **peak db**, **rms db** and **env db** to −24 dB and **Det Mix** to *min*; the bias value is then 1 at all times (chapter 20). Next, set **Bias x base** to 0.2 (the height is then exactly 0.2) and **Bias on limit** to 0.5, and leave **Bias +inverse** at 1 and **Bias on pre->out** at 0. To isolate the effect, neutralize the compressor with a **Ratio** of 1. On a signal that goes above 0.3, the slope is halved from 0.3 upward, and the output caps at 0.5 beyond 0.7: this is the right-hand curve in the second figure of chapter 19. Listen to the product with **Mon bias sat**, then vary **Bias x base** to widen or tighten the bend.

Saturation that only appears when compressing. Replace **Bias x base** with **Bias x gr**: the height only exists when the compressor is reducing gain. Set **Bias on pre->out** to 1 so the limiter acts on the compressed signal.

A compensated drive. With an active limiter, raise **Bias Fm On** (2 to 4) to hit the limiter harder, then set **Bias <> bs->ub** to 1: the level comes back to roughly where it started, and the saturation stays.

Even-order saturation. Bring **Bias +inverse** back to 0 and set **Bias <> bs->ub** to 1 in the "out" position: the offset is removed, and what remains is clipping that starts on one side only, rich in even harmonics. Alternatively, with an input height and the inverse path at 1, set **bias amp** to −1 to double the even part of the normal path.

Taming saturation that's too intense. Give the limiter an attack and release of a few milliseconds (**Bias on attack**, **Bias on release**): it becomes a compressor, and the peaks are no longer clipped hard. A negative high band in **F (bs)** also softens the product's high-end grain.

> **A word from the author**
>
> By combining the fusion, the input height, the output height, the pre or out position, **Bias Fm On** and **Bias Fm In**, you get very different types of saturation, without even using the output compressor. François admits that his demonstrations only show a small part of these possibilities: you have to experiment, because the results vary enormously from one combination to the next.

Starting ideas

- **A vocal or a bass that picks up grain as it pushes.** Load **tr_670** and listen to its destination, or start from the default settings with **Bias x gr** between 0.3 and 1 and **Bias on limit** at 1. Then dial it in with **bias amp**.
- **Density on a bus or a master, level-matched.** A compensated drive (**Bias Fm On** from 1.5 to 2, **Bias <> bs->ub** at 1) feeding a soft limiter (**Bias on limit** around 0.5, **Bias limit knee** from 30 to 60, **Bias on ratio** from 2 to 4). Don't forget that one of the five amounts must be something other than 0: a tiny value of **Bias x base** is enough, as in **pulsarmodular_p42**.
- **Even-order saturation on drums or guitar.** Follow the "Even-order saturation" workflow above, then give the limiter an attack of a few milliseconds to preserve the transients.
- **Textures for sound design. Bias +inverse** at 0 with direct fusion, to let the height through (remove the offset with the **F (bs)** low band at −1), the product delay with negative feedback, and **bias amp** beyond ±1. Keep **bias dst limit** at 1 to protect your monitoring.

CHAPTER 22

Lookahead

The snap of a snare that gets through before the compressor reacts, the consonant that sticks out of an otherwise well-controlled vocal, the peak that forces the master limiter to clip: these three situations share the same cause. An ordinary compressor always reacts **after** the sound: a transient has to arrive before the detector can measure it, and then the envelope needs time to come down. Lookahead, as its name suggests, solves this problem by delaying the signal you hear: the compressor measures the sound a little before processing it, so it can start reducing the gain before the transient arrives.

Comprendre takes this idea much further than usual:

- it doesn't just shift the detection: it runs a **complete copy** of the compressor on the "future" signal, and can anticipate either the **detection** or the gain **envelope** (the "two schools");
- an adjustable **hold** (peak hold) lets you bring the attack forward without bringing the release forward;
- a **dedicated envelope follower**, set in the **ENV (la)** and **A/R (la)** tabs, can smooth the anticipated envelope;
- finally, the lookahead can run **in reverse**: the compressor then looks into the past and compresses later.

To design this module, François drew in particular on a video by Dan Worrall, *The Least Exciting Compressor Feature*: he counts it among the videos that inspired him most, and recommends watching it. Where an ordinary lookahead simply shifts the compressor in time, Comprendre's lets you shape the way the compression arrives: François speaks of customizable anticipation, and ends his video by stating, with unabashed pride, that he considers this lookahead the best you can find in a compressor. Mastering engineers will find a limiter that sees peaks coming; mix engineers, a way to handle the attacks of a vocal or a piano without arriving late, or, on the contrary, to let the attacks through and compress only the sustained notes; sound designers, a compressor that can, quite literally, compress late.

The seven lookahead controls take up the right half of the **MOD & LA** tab; the left half is devoted to modulations (chapter 23).



Figure 125 — The lookahead controls in the MOD & LA tab: la delay, la det pk, la det min max, det->la.det, la env pk, la env min max, env->la.env.

> **Copy the plugin state before you touch the lookahead**
>
> François puts it bluntly: the lookahead controls are extremely sensitive, starting with **la delay**. Before exploring them, save a preset or copy the plugin state in your host, so you can easily go back. **la delay** changes the plugin's latency, and **la delay**, **la det pk** and **la env pk** rebuild internal buffers every time they change; a small move can completely transform the compressor's behavior.

How it works: two chains offset in time

As chapter 6 pointed out, as soon as **la delay** is set to anything other than 0, Comprendre runs its entire processing chain **twice**.

- The **present chain** is the one you hear. With lookahead, it reads the input signal with a delay Δ equal to the set duration.
- The **future chain** is a complete copy of the present chain: same input filters, same dynamic EQs, same detection, same transfer curve, same envelope follower, and the same everything else, with its own memory. It reads the signal **without** this delay, so it runs Δ ahead of the present chain. Its sound is **never** heard.

Both chains are fed by the same delay line, placed just after **Input Db** (chapter 6). The future chain supplies the present chain with two quantities:

1. its **detection signal**, that is, the instantaneous level entering the compressor, before any of the detection in the **DET (in)** tab;
2. its **final gain envelope**, that is, the gain coming out of the compressor, after the transfer curve and the **A/R (cv)** follower.

The present chain can use one, the other, or both, at two different points:

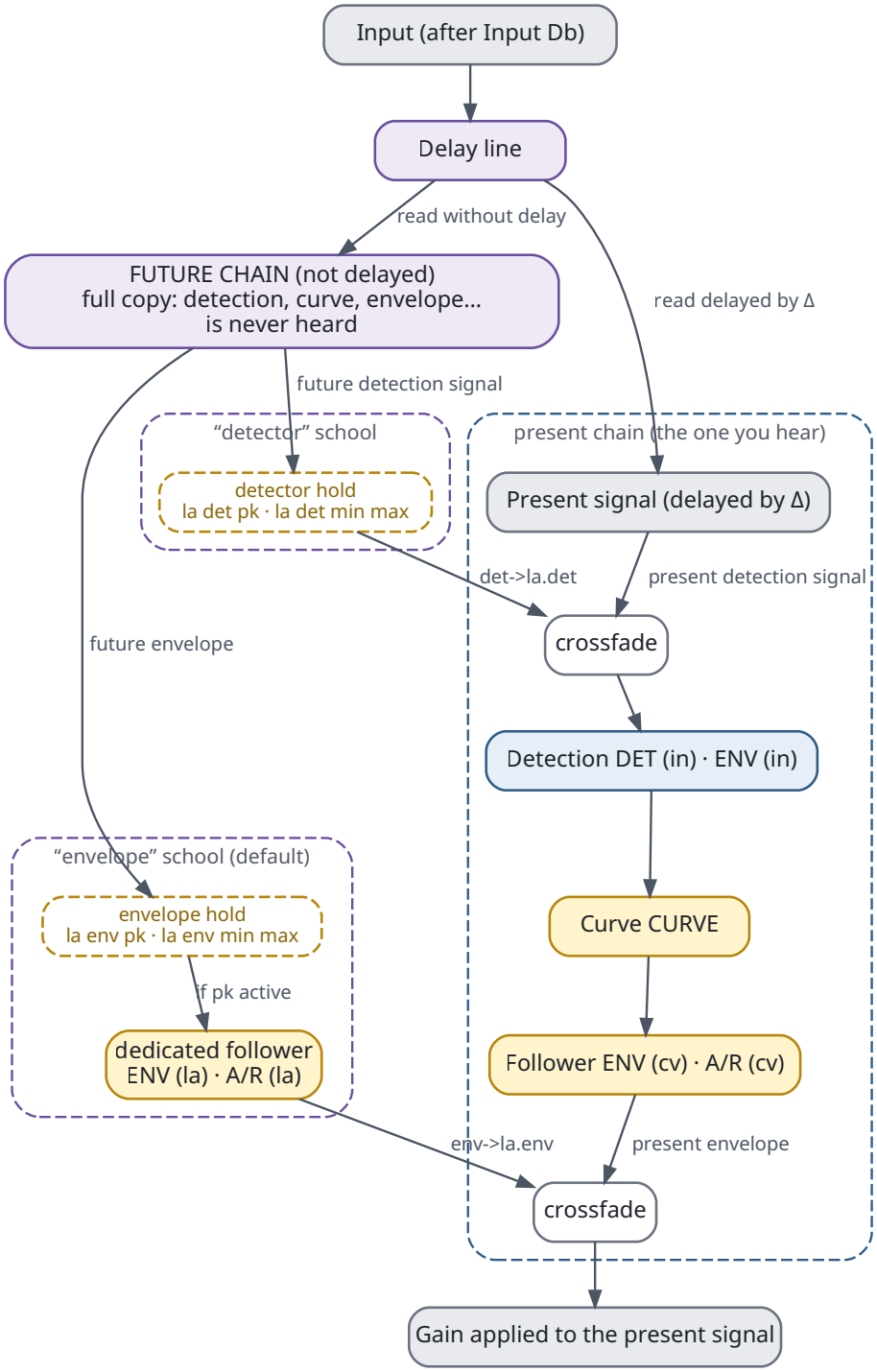


Figure 126 — The two chains and the two lookahead injection points. On the left, what the future chain provides; on the right, the present chain, the only one you hear.

- the future detection signal can replace the present one **just before** the detection: this is the “detector” school, with the amount set by **det->la.det**;
- the future envelope can replace the present one **just after** the compressor: this is the “envelope” school, with the amount set by **env->la.env**, and it’s active by default.

In both cases, the future quantity can first go through a **hold** (**pk** and **min max** controls) and, on the envelope side only, through a **dedicated envelope follower** (**ENV (la)** and **A/R (la)** tabs).

The next figure sums up what happens at any given moment. While the present chain processes and plays sample $t - \Delta$, the future chain measures sample t, the most recent one the plugin has received. The value applied at $t - \Delta$ is therefore known Δ seconds "in advance." With a hold, it's not just the future value at t that counts, but the extremum of the future values over a window that ends at t.

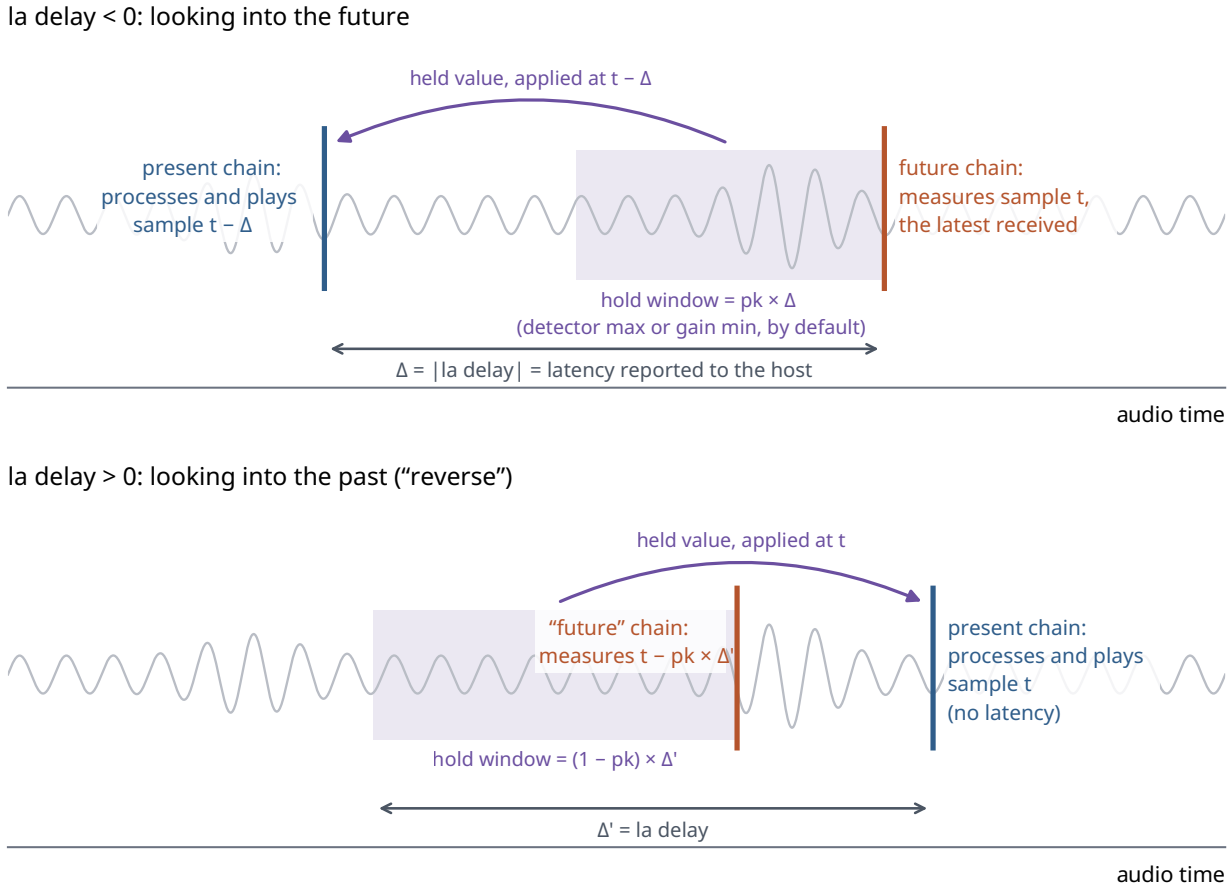


Figure 127 — What is measured and what is applied. Top: lookahead (negative la delay). Bottom: "reverse" (positive la delay), described later.

> **A complete copy, so twice the computation**
>
> The future chain runs as soon as **la delay** is other than 0, even if **det->la.det** and **env->la.env** are both at 0. It redoes all the work of the present chain, saturation stages included: the CPU load roughly doubles. If the bias inverse circuit is active (chapter 19), the lookahead is applied to it in the same way, with the same settings.

> **What the graphs show**
>
> The graphs display the present chain, that is, the sound you hear (chapter 4). With lookahead, you therefore see the gain reduction start **before** the transient that triggers it.

From delay line to digital limiter

The idea is almost as old as compression itself: as early as 1947, the General Electric BA-5 limiter ran the audio through a delay line so that the control had time to act (chapter 2). Digital technology made it commonplace. In 1994, the Waves L1 became the first mass-produced digital "brickwall" lookahead limiter, able, according to the trade press, to shave the top few decibels off the peaks with no audible effect. The feature was then picked up by most mastering limiters and maximizers, and the L1 became one of the weapons of the loudness war of the 1990s. The Weiss DS1, a digital mastering compressor introduced in 1998, also has its own lookahead, called **Preview**. The price has never changed: latency. What Comprendre adds is everything else: the choice of what is anticipated, the hold, the dedicated follower, and even the direction of time.

Looking into the future, or the past

la delay

MOD & LA tab

Range: −1 to 1 s **Default:** 0 s **ID:** `la_delay`

Length of the offset between the two chains, and its direction. This is the control that switches the whole module on.

- **0** (default): the lookahead is off. Nothing is computed and no latency is added. A value whose absolute value is less than one internal sample period (about 21 µs at 48 kHz without oversampling) amounts to the same thing.
- **Negative**: **lookahead**, looking into the future. The present chain is delayed by |la delay|; the future chain sees the signal |la delay| earlier. The plugin reports a latency of |la delay| to the host, which compensates for it: when you listen, the sound stays aligned with the other tracks, but the compression arrives early.
- **Positive**: **reverse** (look-behind), looking into the past. The present chain isn't delayed and no latency is added; it's the "future" chain that gets delayed. The compressor then reacts later than usual (see "Lookahead in reverse").

The range runs from −1 to +1 s. The knob's travel is very gradual around the center: halfway between the center and one end stop, the value is only about 4 ms; 0.1 s is reached at three-quarters of this half-travel, and 0.25 s at around 84%. For a precise value, type it into the box.

Typical uses range from a few milliseconds (anticipating transients, as in a limiter) to a few hundred milliseconds. François uses −0.25 s in his demonstration to exaggerate the effect, which then becomes very visible on the graphs, and very audible.

When la delay goes to a positive value, **la det pk** automatically takes on the value of **la env pk** (see below).

> **♪ In the studio: how much lead time?**
>
> The right value for **la delay** depends on what you want to catch. For a master limiter, or for drums whose peaks need to be held in check, a few milliseconds are enough (−1 to −5 ms): the **acqua_taupe** preset makes do with −4 ms. On a piano or a vocal, a few tens of milliseconds give the compressor time to settle in before the note: **massenbug_drc2_piano**, an imitation of the Massenburg DesignWorks MDWDRC2 whose name suggests a piano setting, looks 44 ms ahead. Beyond 100 ms, you're into effect territory: the reduction starts well before the sound, and you often hear the level dip before each attack, like an intake of breath. Don't forget that every millisecond of lead time adds to the latency (see "Latency").

The two schools

François talks about two "schools" of lookahead. They differ in **what** gets anticipated.

The "envelope" school, active by default, anticipates the compressor's **result**. The future chain computes the complete gain envelope Δ ahead of time (detection, transfer curve, **A/R (cv)** attack and release, **GR (cv)** tab options); the present chain applies this envelope instead of its own. This is the school that makes the dedicated **ENV (la)** / **A/R (la)** follower possible.

The "detector" school anticipates the **measurement**. The future detection signal enters the present chain's compressor in place of (or mixed with) the present signal. Everything downstream, from the **DET (in)** detection to the **A/R (cv)** follower, then works normally on this anticipated signal. This is the most conventional approach to lookahead.

env->la.env

MOD & LA tab

Range: 0 to 1 **Default:** 1 **ID:** `la_env2env`

Amount of the "envelope" school: a linear mix between the future gain envelope (processed by the hold and the dedicated follower) and the present envelope.

- **1** (default): the applied gain comes entirely from the future chain. The envelope computed by the present chain is no longer used for the gain.
- **0**: the "envelope" school is off. The hold and the dedicated follower aren't even computed.
- **In between**: gain = future envelope × x + present envelope × (1 − x). This way you can take just part of the anticipation.

With the default settings (**la env pk** at 1, **la env min max** at −1, **env->la.env** at 1), all it takes is a negative **la delay** to get full lookahead: the gain reduction starts Δ earlier and the original release is preserved.

This mix happens at the very end of the compressor: the resulting gain is the one then used by the target gain, the dynamic EQs, the waveshaper and the bias.

det->la.det

MOD & LA tab

Range: 0 to 1 **Default:** 0 **ID:** `la_det2det`

Amount of the “detector” school: a linear mix, at the input of the present chain’s compressor, between the future detection signal (possibly held by **la det pk**) and the present detection signal.

- **0** (default): the “detector” school is off.
- **1**: the present compressor only sees the future detection signal.
- **In between**: detected level = future × x + present × (1 − x). This is a mix of levels, before the detection: the transfer curve is then applied to the result.

This control only has an effect if **la delay** is other than 0. To hear the “detector” school, you also need to **lower env->la.env**, which is 1 by default (see below).

> **The envelope takes priority**
>
> The two schools can be active at the same time, but the “envelope” school wins out. With **env->la.env** at 1, the applied gain is entirely the future envelope, which the future chain computes from its own detection: the anticipated detection signal injected into the present chain then has no effect at all on the gain. To hear the “detector” school, set **env->la.env** to 0; to mix the two, set it between 0 and 1.

On a simple signal, the two schools give very similar results. The next figure compares the “envelope” school and the “detector” school, both with a 50% hold, on a 1 s burst; the two gain curves almost overlap. The differences show up elsewhere: the “detector” school lets the present chain’s detection, curve and **A/R (cv)** follower process the anticipated signal, with all their settings; the “envelope” school anticipates a gain that has already been computed, which it can hold and then smooth with its own follower.

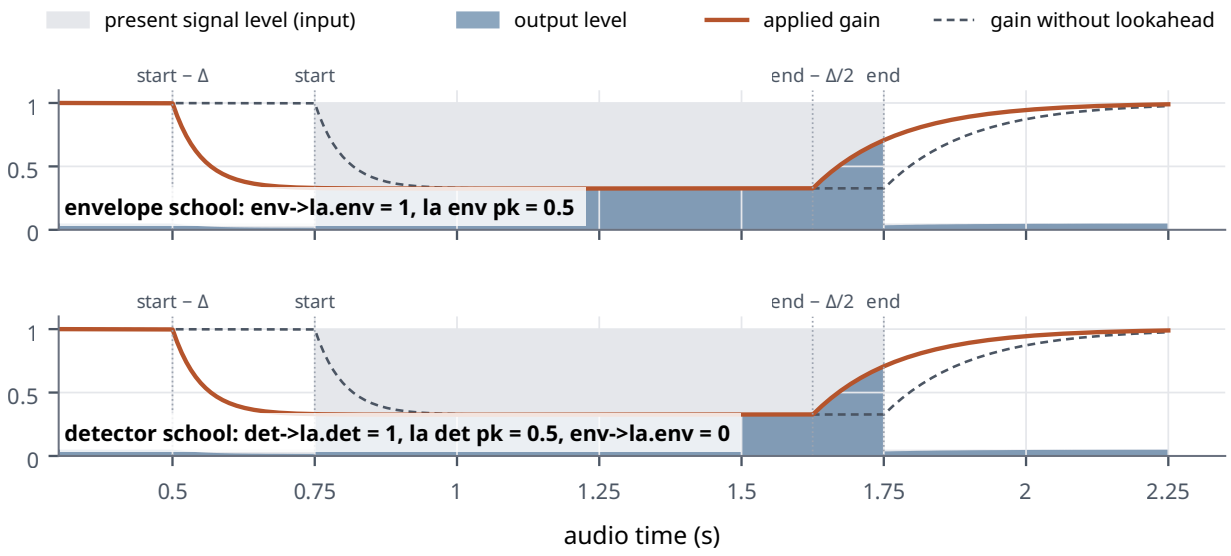


Figure 128 — "Envelope" school and "detector" school, same 50% hold (Threshold −20 dB, Ratio 4, Attack 50 ms, Release 150 ms, la delay −0.25 s). On a burst, the two gains are almost identical.

> **A word from the author**
>
> François clearly prefers the "envelope" school, which he considers more powerful: it's the one that lets you hold the anticipated envelope and then smooth it with the dedicated follower. The "detector" school is more conventional and less powerful, but it has its uses. His advice: if you don't want the detection to be altered by the lookahead, work on the envelope; if you don't want to touch the envelope, work on the detection, and you'll keep the compressor's "natural" envelope.

The hold: pk and min max

If you simply shift the envelope (or the detection) by Δ, **everything** gets shifted: the attack, but also the release. The compressor starts releasing Δ before the end of the sound, and the end of the sound is no longer compressed. This is the flaw François shows at the start of his lookahead video.

The **hold** fixes this flaw. Instead of taking the future value at a single instant, it takes the **extremum** (the minimum or the maximum) of the future values over a sliding window. The length of this window is a fraction of Δ, set by **pk** (peak hold):

- **pk** = **0**: no hold; the future value is simply shifted by Δ;
- **pk** = **1**: the window covers all of Δ. The attack is moved Δ earlier, but the release starts at the same moment as without lookahead;
- **in between**: the attack is moved Δ earlier, and the release Δ × (1 − pk) earlier.

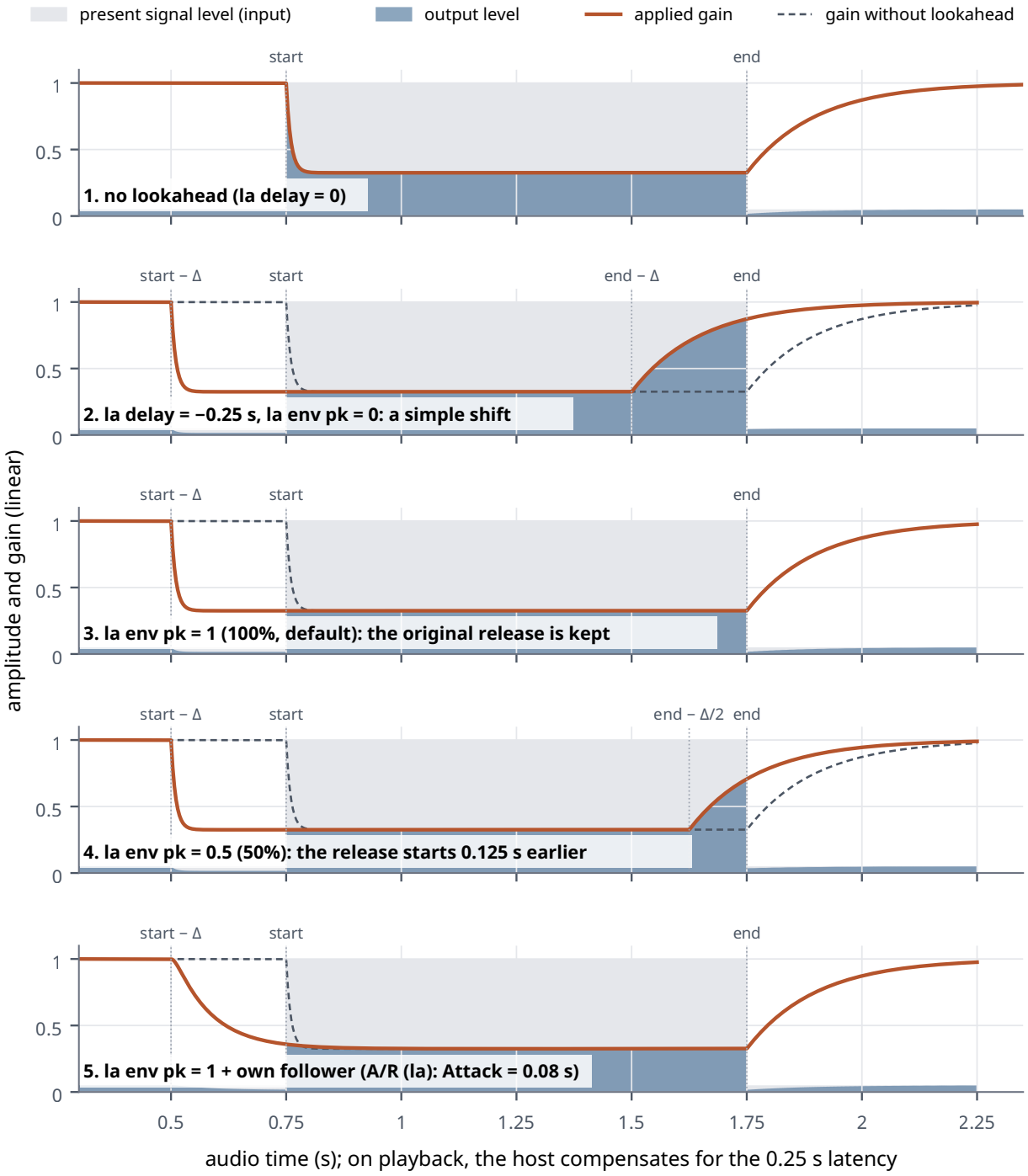


Figure 129 — Simulation on a level burst (a square wave that goes from −26 dB to 0 dB for 1 s), with Threshold −20 dB, Ratio 4, Attack 10 ms, Release 150 ms. The gain is plotted in audio time, as you hear it. 1: no lookahead. 2: la delay −0.25 s with no hold; the release is moved earlier too, and the end of the burst escapes compression. 3: 100% hold (default setting). 4: 50% hold. 5: 100% hold plus the dedicated follower (Attack (la) = 80 ms), which turns the anticipated step into a gradual slope.

The direction of the extremum is chosen with **min max**. By default, it goes in the direction of **more compression**: the **maximum** of the detection signal (the loudest level to come), the **minimum** of the gain envelope (the lowest gain to come, hence the strongest reduction). That's why the defaults of the two controls are opposite: the envelope is a gain, which is 1 with no reduction and drops as you compress.

> **Going further**
>
> Let Δ be the offset and p the pk setting. With lookahead, the value applied at time u is the extremum of the future values between $u + \Delta(1 - p)$ and $u + \Delta$. For compression that, without lookahead, would run from time a to time b:
>
> | Hold | Resulting compression |
> |---|---|
> | none (pk = 0) | from $a - \Delta$ to $b - \Delta$ (simple shift) |
> | in the direction of more compression (default) | from $a - \Delta$ to $b - \Delta(1 - p)$ |
> | in the opposite direction | from $a - \Delta(1 - p)$ to $b - \Delta$, and nothing if $b - a$ is shorter than $p\Delta$ |
>
> The hold acts on the value itself. Downstream, the dedicated follower (envelope school) or the present chain's detection and follower (detector school) then round off the edges. The sliding maximum or minimum is computed exactly, sample by sample.

la env pk

MOD & LA tab

Range: 0 to 1 **Default:** 1 **ID:** `la_env_pk`

Length of the hold window applied to the future **gain envelope**, as a fraction of Δ.

- **1** (default): hold over all of Δ. The reduction starts Δ before the transient, and the release starts at the same moment as without lookahead: this is what François calls preserving the release.
- **0**: no hold. The envelope is simply shifted by Δ: the compressor releases Δ before the end of the sound.
- **In between**: the release starts Δ × (1 − pk) before the original moment. At 0.5 with Δ = 0.25 s, it starts 0.125 s earlier.

This control also governs the **dedicated follower**: the **ENV (la)** and **A/R (la)** tabs only act if the hold is active, that is, if **la env pk** is greater than 0 in lookahead (less than 1 in reverse). Without a hold, the future envelope is applied as is.

It only has an effect if **env->la.env** is greater than 0. In reverse (positive **la delay**), it's linked to **la det pk**: changing one changes the other.

la env min max

MOD & LA tab

Range: −1 to 1 **Default:** −1 **ID:** `la_env_mm`

Chooses which extremum the envelope hold keeps. The knob has only two positions.

- **−1** (default): **minimum** of the gain over the window, that is, the strongest reduction to come. This is the “classic” hold: any compression expected within the window is applied right away, and held until the end.
- **+1**: **maximum** of the gain over the window, that is, the weakest reduction to come. The compressor only reduces the gain if the reduction is going to last for the **entire** window. Events shorter than the window are largely ignored, and the compression of long events starts later and ends sooner.

The +1 position is an unusual tool: it lets you, for example, let short transients through while compressing sustained notes.

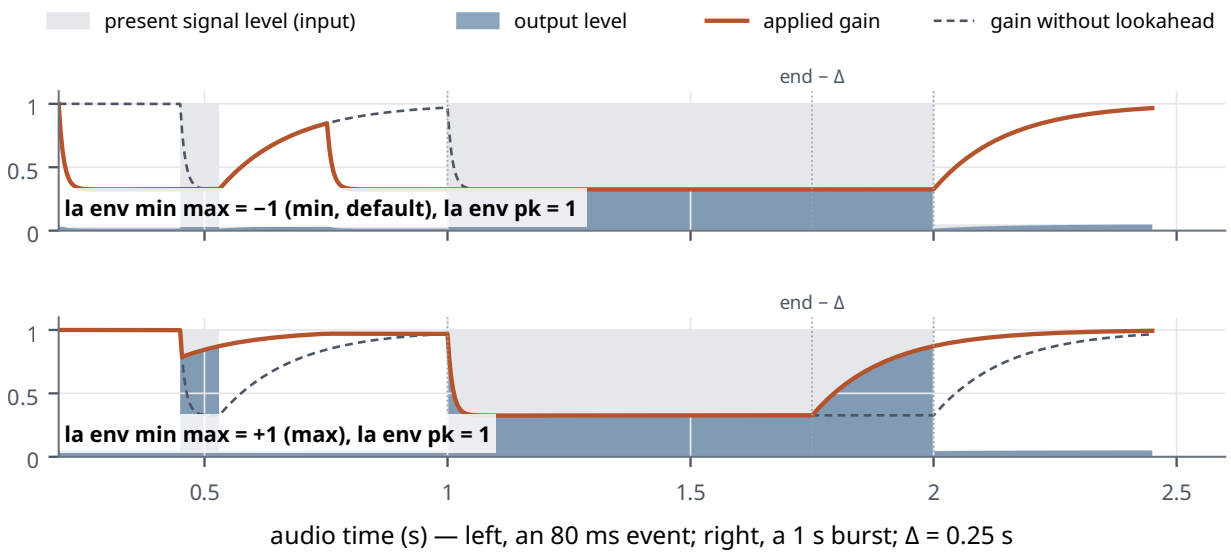


Figure 130 — Effect of la env min max on a short event (80 ms) followed by a 1 s burst, with la delay −0.25 s and la env pk = 1. Top: the minimum (default); each reduction is moved 0.25 s earlier and held. Bottom: the maximum; the short event leaves only a slight reduction, and the burst is only compressed from its start until 0.25 s before its end.

♪ In the studio: let the attacks through, hold the notes

The +1 position of **la env min max** turns the usual logic of lookahead on its head: instead of anticipating the peaks, the compressor largely ignores anything too short to fill the window. With **la delay** between −20 and −50 ms, the pluck of an acoustic guitar, the hammer of a piano or the slap of a bass get through almost intact, while the sustained notes are compressed. It’s a way to keep a performance’s articulation while evening out its sustain; the “Compressing only sustained sounds” workflow at the end of the chapter gives the full settings.

la det pk

MOD & LA tab

Range: 0 to 1 **Default:** 0 **ID:** `la_det_pk`

Length of the hold window applied to the future **detection signal**, as a fraction of Δ. It plays the same role as **la env pk**, but for the "detector" school: it only has an effect if **det->la.det** is greater than 0.

- **0** (default): no hold. The detection signal is simply shifted by Δ: this is the most conventional lookahead, where both the attack and the release are anticipated.
- **1**: the detected level rises Δ before the transient and only comes back down when it would have come back down without lookahead.
- **In between**: the detection comes back down Δ × (1 − pk) before the original moment.

The detection signal is an instantaneous level that follows the waveform. A sliding maximum over a window longer than one period therefore also replaces the waveform's oscillations with a held peak level: the DET (in) detection then receives a much smoother signal.

In reverse (positive **la delay**), **la det pk** and **la env pk** are linked: changing one changes the other. When **la delay** becomes positive, **la det pk** takes on the value of **la env pk**.

la det min max

MOD & LA tab

Range: −1 to 1 **Default:** 1 **ID:** `la_det_mm`

Chooses which extremum the detection signal hold keeps. The knob has only two positions.

- **+1** (default): **maximum**, the loudest level to come. This is the usual peak hold.
- **−1**: **minimum**. The detection only keeps what stays loud for the entire window: short peaks disappear from the detection.

Caution: the detection signal follows the waveform and crosses zero on every half-cycle. As soon as the window is longer than half a period of the signal, its minimum is therefore almost zero: on a music signal, the −1 position practically cancels out the anticipated detection. It only makes sense for very short windows, or for signals that don't cross back through zero.

The defaults of **la det min max** (+1) and **la env min max** (−1) are opposite, but both go in the direction of more compression: a louder detected level, or a lower gain.

To be confirmed by François

With **Channels Unlink** greater than 0, the per-channel detector hold is applied to the unrectified signal: the maximum only keeps the positive peaks and the minimum the negative peaks, unlike in linked mode. Moreover, in reverse with a nonzero hold window, the memory for the per-channel detector hold isn't created, and the per-channel anticipated detector is 0. Is this intended?

The dedicated follower: ENV (la) and A/R (la)

The envelope hold moves the attack forward as a block: the future reduction arrives Δ earlier, but just as abruptly as the original attack. With a Δ of a few milliseconds and zero attack, that's often what you're after (a limiter that doesn't let transients through). With a longer Δ, you may prefer the gain to **glide** gradually toward the reduction over the whole anticipation time: that's the job of the **dedicated follower**.

This follower is a full envelope follower, built on the same engine as the other three (chapters 11 and 12). It sits just after the envelope hold, and follows the held **gain**:

- **Attack** in the **A/R (la)** tab smooths the **arrival** of the reduction (the gain going down);
- **Release** in the **A/R (la)** tab smooths its **return** (the gain going back up).

In the burst figure (panel 5), an 80 ms attack turns the anticipated step into a gradual slope that starts 0.25 s before the sound and reaches almost the full reduction by the time the sound arrives. François describes this behavior as a follower gliding inside the held envelope.

> **Rule of thumb**
>
> In logarithmic mode and with the default altitude, a follower covers 63% of the way in a time equal to the attack, and 95% in three times that (chapter 12). An **Attack** in the **A/R (la)** tab equal to a third of |**la delay**| therefore brings the gain to 95% of the reduction by the time the transient arrives.

The dedicated follower only acts if **all** of the following conditions are met:

1. **la delay** is other than 0;
2. **env->la.env** is greater than 0;
3. the envelope hold is active: **la env pk** greater than 0 in lookahead, less than 1 in reverse;
4. **Attack** or **Release** in the **A/R (la)** tab is greater than 0. Both of these times are **0 by default**, in which case the follower simply copies the held envelope.

The other **ENV (la)** and **A/R (la)** controls play the same role as those in the **ENV (cv)** and **A/R (cv)** tabs, described in chapters 11 and 12. The following table lists the correspondences and the specifics.

(la) tab	Controls	Where to read about them	Specifics of the dedicated follower
A/R (la)	**Attack**, **Release**	chapter 12 (Attack and Release in the main row)	They're in the tab itself, in place of **mod attack** and **mod release**. **0 by default**: with both at 0, the follower is disabled. No modulation possible (chapter 23).
A/R (la)	**Attack Auto**, **Release Auto**	chapter 12	They use the **auto secs** time constant from the **DET (in)** tab.
A/R (la)	**Attack Altitude**, **Attack Shape**, **Att Ret From**, **Att Ret Spd**, **Attack Mode**, and their release equivalents	chapter 12	Identical. On the small attack and release graphs, the dedicated follower is drawn in lavender (chapter 4).

(la) tab	Controls	Where to read about them	Specifics of the dedicated follower
ENV (la)	**EvMu Lin/Db**, **EvMu Br/Cp**, **Hold branch**	chapter 11	Identical.
ENV (la)	**Env->Mu**, **Mu Min Att**, **Mu Max Att**, **Mu Pow Att**, **Mu Min Rel**, **Mu Max Rel**, **Mu Pow Rel**, **Mu Speed**	chapter 11	The Mu speeds are driven by the followed gain, which is 1 with no reduction: "Max" therefore corresponds to no reduction, the reverse of **ENV (cv)**, where "Max" corresponds to a loud signal.
ENV (la)	**Env/Mu->Gamma**, **Gamma A delai**, **Gamma R delai**	chapter 11	For the Gamma part alone, the code treats the followed signal as a level rather than a gain: the arrival of the reduction takes the longer of the two **A/R (la)** times, and its return the shorter one (point to be confirmed, see chapter 11).

> **♪ In the studio: shaping how the compression arrives**
>
> This is where Comprendre's lookahead becomes an expressive tool. Without the dedicated follower, the reduction drops in as a block, Δ before the sound. With an **Attack** in the **A/R (la)** tab equal to a third of |**la delay**|, the gain glides toward the reduction over the whole anticipation time and nearly reaches it when the note arrives: on a vocal or an acoustic guitar, the attack is held in check without you hearing the compressor "clamp down." Lengthen this attack, and the compression arrives like a fade still in progress when the transient hits, so the transient keeps a bit of its bite. The **Release** in the same tab rounds off the return of the gain in the same way. Try it on a vocal with **la delay** at −20 ms and an **Attack** in the **A/R (la)** tab of 7 ms, then compare with the follower turned off (both of its times at 0).

Lookahead in reverse

With a positive **la delay**, Comprendre looks into the **past**: François calls it "reverse" or "look-behind." The present chain is no longer delayed, and no latency is added; it's the "future" chain that reads the signal late, by **la delay** × **pk**, and the hold window covers the rest, **la delay** × (1 − **pk**). The whole toolkit is still available (detector or envelope school, hold, dedicated follower), but in reverse.

In reverse, there's only one "future" chain, and its delay depends on **pk**: the two controls **la det pk** and **la env pk** are therefore linked, and moving one moves the other. Their meaning is reversed too (Δ' here denotes the value of **la delay**):

- **pk** = **1** (default for **la env pk**): no hold; the compression is simply **delayed** by Δ'. The start of the sound gets through uncompressed, and the reduction carries on for Δ' after the end of the sound: "my compressor compresses later," as François puts it.
- **pk** = **0**: no delay, but a hold over the entire Δ' window. The attack arrives on time, and the reduction is **held** for Δ' after the end of the sound before the release begins: in other words, a hold.
- **In between**: the attack is delayed by Δ' × pk, and the release by Δ'.

As with lookahead, the default hold goes in the direction of more compression. The dedicated follower only acts with **la env pk** less than 1.

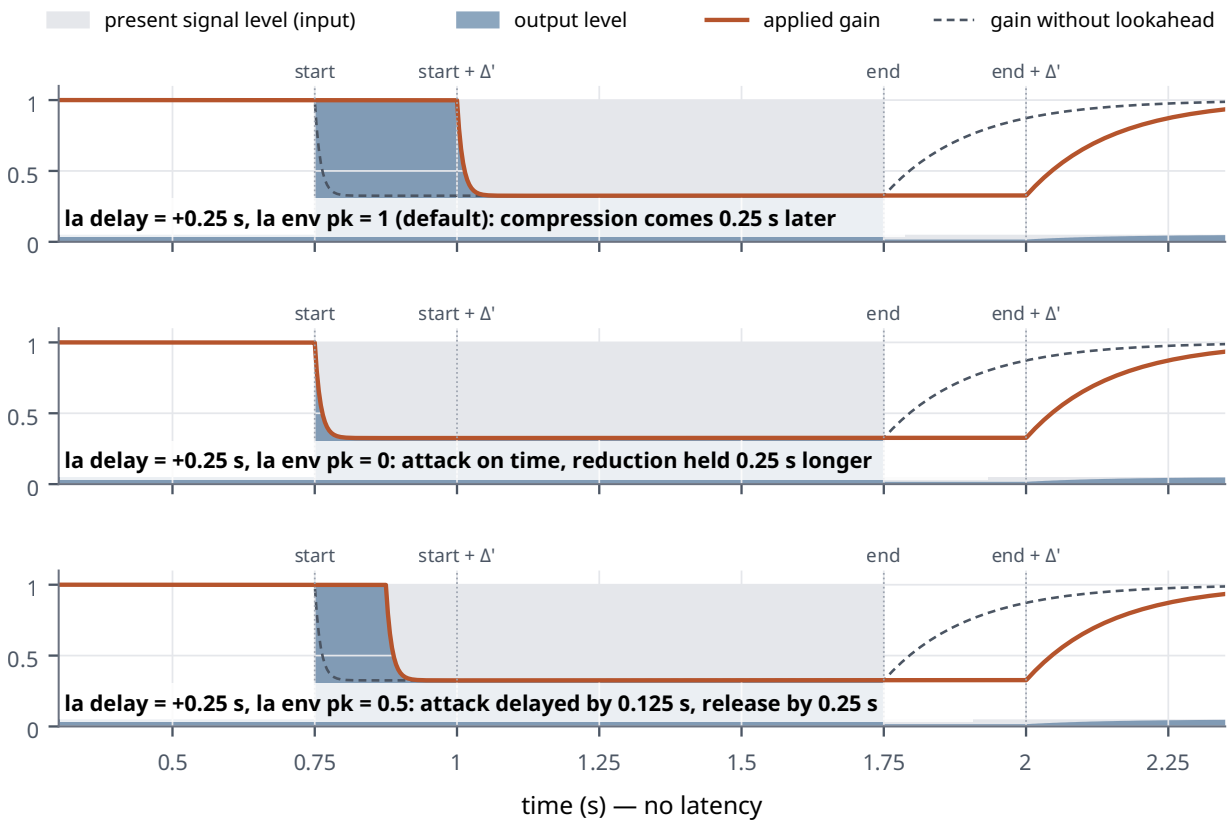


Figure 131 — Reverse, la delay +0.25 s, same settings as the previous simulation. Top: pk = 1, the compression is delayed. Middle: pk = 0, the reduction is held 0.25 s longer. Bottom: pk = 0.5.

A word from the author

François introduces the reverse mode with a touch of humor: it's "a bit ridiculous," but "it can be fun." Deliberately delaying the compression lets the attack of sounds through and extends their reduction, a bit like a compressor that reacts a beat late.

A 'reverse lookahead' in the 1176

The idea of compression that arrives late isn't just a joke. The 1176 manual describes its famous "all buttons" mode, with all four ratio buttons pushed in together, as a kind of "reverse lookahead": the attack lags behind the first transients, and distortion increases dramatically (chapter 2). Comprendre doesn't reproduce this mode, but its reverse lets you explore, in its own way, what you gain by letting the start of a sound through.

♪ In the studio: compressing late

On drums, reverse with **la env pk** at 1 lets the attack of each hit through intact, then compresses the body of the sound and extends the reduction by **la delay** after it ends: the crack stands out, and what follows is held in check. With **la env pk** at 0, the attack arrives on time, but the reduction stays in place for **la delay** after the end of the sound, with no latency at all: a way to keep the ring of a room or a tom tucked back after each hit. Only one preset looks into the past: **acqua_titanium**, an imitation of Acustica Audio's Titanium, a suite of processors sampled from tube hardware, sets **la delay** to +0.032 s with **la env pk** at 1, so its compression arrives 32 ms later than usual.

Latency

With lookahead (negative **la delay**), the sound you hear is delayed by |la delay|. Comprendre reports this delay to the host as **latency**, which the host compensates for automatically (latency compensation, or PDC). When you listen, the track stays aligned with the others; it's the compression that arrives early.

- The added latency equals |la delay| × sample rate, rounded down to the nearest sample: 12,000 samples for −0.25 s at 48 kHz, 240 samples for −5 ms.
- It adds to the latency of the FIR filters (chapter 14) and of oversampling (chapter 7). For example, at 48 kHz without oversampling, a −5 ms lookahead and 65-tap FIR filters give 240 + 32 = 272 samples.
- In reverse (positive **la delay**), no latency is added.
- The dry signal of **Dry/Wet** and the reference for **Monitor Delta** are taken from the present signal, so they stay aligned with the processed signal.

Latency and live playing

A lookahead of several tens of milliseconds delays everything going through the track by that much. That doesn't matter when mixing, but it gets in the way when recording or playing live through the plugin.

To be confirmed by François

The delay line holds sixteen times the host sample rate, which is only one second at ×16 oversampling. With ×16 and a **la delay** close to ±1 s (plus the FIR filter delay), the requested delay exceeds the line, and the delay actually read is wrong. Should **la delay** be limited in this case?

To be confirmed by François

Changing **la delay**, **la det pk** or **la env pk** rebuilds the hold buffers while the audio is being processed. Could this cause clicks, or even a crash, during playback? Should we advise setting these three parameters with playback stopped?

To be confirmed by François

With the “envelope” school, only the final gain is replaced by the future gain. The opto designer’s envelope (**GR (cv)** tab, chapter 15) is still computed by the present chain, relative to the present gain, and is therefore not anticipated. Is this intended?

Practical workflows

A limiter-style lookahead

This is the most common use: keeping transients from getting through before the compressor reacts.

1. Set the compressor up as a limiter: high ratio, **Attack** at or near 0, **Release** to suit the program material.
2. Set **la delay** between −1 and −5 ms. The other lookahead controls stay at their default values: “envelope” school, 100% hold, minimum.
3. The gain reduction now starts |la delay| before each transient and releases as before.
4. If the abrupt drop into reduction is audible (a click on each transient), give **Attack** in the **A/R (la)** tab a value of around a third of |**la delay**|: the gain then comes down gradually over the anticipation time.

Anticipating without changing the release

If you’re happy with your compressor setting but the attack arrives too late, a lookahead with the default settings moves the attack earlier without touching the release. Start with a few milliseconds and increase |**la delay**| until the attack lands where you want it; reduce **Gfx Secs** (chapter 4) to clearly see the reduction come before the transient.

Releasing before the end

To have the compressor start releasing a little before the end of sounds (for example, to keep a slow release from “eating” the next note), lower **la env pk**: at 0.5, the release starts |**la delay**|/2 earlier; at 0, it starts |**la delay**| earlier.

Classic lookahead, on the detection

To get back to conventional lookahead, where you simply move the detection earlier:

1. **la delay** negative;
2. **env->la.env** at 0;
3. **det->la.det** at 1;
4. **la det pk** depending on the release you want: 0 to shift everything, 1 to keep the original release.

The present chain’s detection, curve and **A/R (cv)** follower then process the anticipated signal with all their settings.

This is exactly the choice made in the **massenbug_drc2_piano** preset: **la delay** at −0.04436 s, **env->la.env** at 0, **det->la.det** at 1 and **la det pk** at 1, to keep the original release.

Compressing only sustained sounds

With **la env min max** at +1 and **la env pk** at 1, the compressor only reduces the gain if the reduction is going to last at least |**la delay**|. With **la delay** between −20 and −50 ms, events shorter than |**la delay**| are largely ignored, while sustained sounds are compressed. The compression of sustained sounds then starts when the sound starts (without anticipation) and releases |**la delay**| before they end.

A “hold” with no latency

With a positive **la delay** and **la env pk** at 0, the reduction is held for **la delay** after the end of each sound, with no latency at all. The effect is comparable to that of the **Hold branch** control (chapter 11), but it applies to the final envelope.

CHAPTER 23

Modulations

Imagine a compressor that clamps down harder on a vocal when it turns harsh in the upper mids, that releases faster on a drum bus every time the kick hits, or whose release follows the brightness of an entire mix. Normally, frequency content only changes what the detector hears: that's the principle behind a sidechain filter or a de-esser. With modulations, it changes **how** the compressor compresses. Taken all the way, François sees in this a form of overall coherence that's almost closer to mastering than to compression (see the end of the chapter). It's also one of the least explored areas of the plugin: only one of the 80 bundled presets uses it.

Modulations make some of the compressor's settings **depend on the signal's content**, and in particular on its frequency content. The principle is simple: sample by sample, Comprendre measures how much the dynamic EQs and the waveshaper of the **detection branch** change the signal; this measurement, once shaped and smoothed, varies four compressor settings:

- the **ratio slope** of the transfer curve;
- the **gate ratio**;
- the **attack** time and the **release** time of the envelope follower placed after the curve.

This gives you what François calls a "frequency-sensitive" ratio: a compressor that compresses harder, or releases faster, when a given frequency band is present in the signal.

Modulations are set in three places:

- the left half of the **MOD & LA** tab contains the three **sources** and the four **shaping** controls;
- the **CURVE** tab contains the **mod ratio** and **mod gate** targets;
- the **A/R (cv)** tab contains the **mod attack** and **mod release** targets.



Figure 132 — The modulation sources and shaping controls, in the MOD & LA tab.



Figure 133 — The mod ratio and mod gate targets, in the CURVE tab.



Figure 134 — The mod attack and mod release targets, in the A/R (cv) tab.

How it works

At every sample, Comprendre computes a **modulation value**, which we'll call *m*. It's **1 at rest**, that is, when the source modules aren't changing the signal, and it moves away from that value when they do. The next figure shows the complete path.

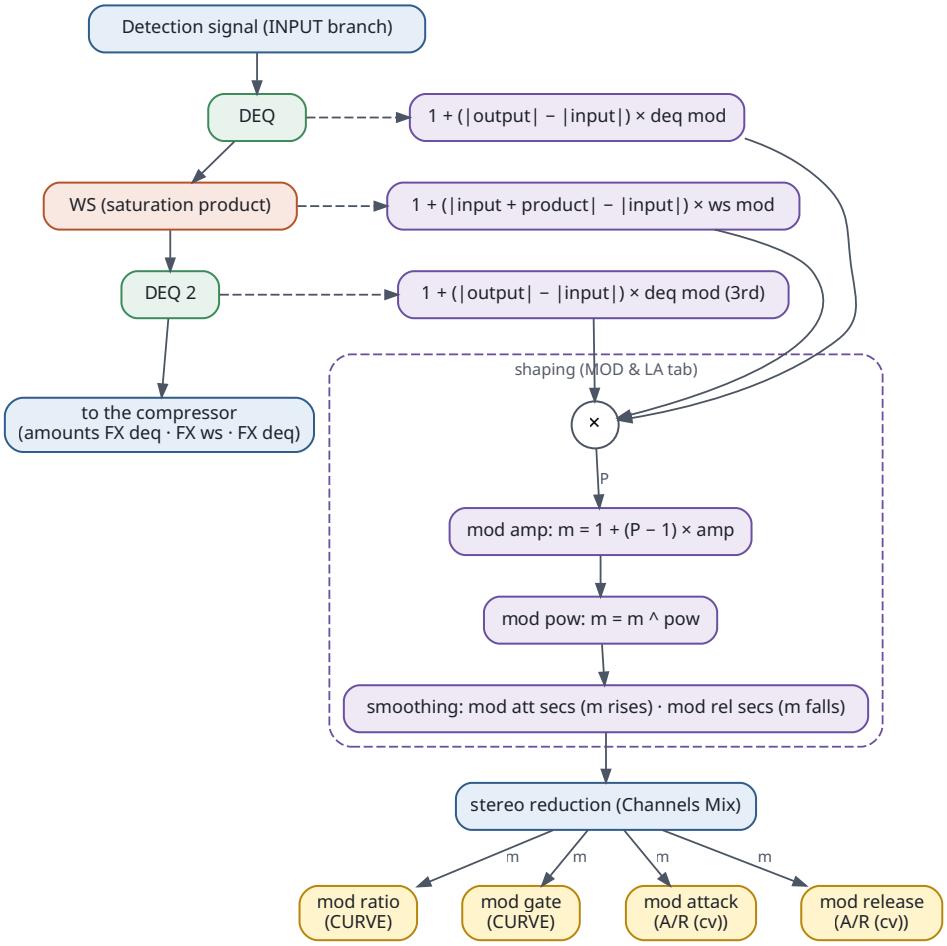


Figure 135 — Computing the modulation value m: three sources measured on the DEQ, WS, DEQ 2 trio of the detection branch, multiplied, shaped, smoothed, then applied to the four targets.

1. **The sources.** In the detection branch, the signal passes through the **DEQ**, the **WS** and the **DEQ 2** in turn (chapters 17 and 18). For each of these three stages, Comprendre measures the difference between the amplitude of its output and that of its input, multiplies it by the corresponding source control, and adds 1.
2. **Combining.** The three resulting factors are **multiplied** together.
3. **Shaping.** The deviation from 1 is amplified by **mod amp**, the result is raised to the power **mod pow**, then smoothed by **mod att secs** and **mod rel secs**.
4. **Stereo reduction.** In linked mode, the channel values are combined according to the **Channels Mix** control (chapter 14): the average in the *Avg* position, the value of a single channel in position *0* or *1*, the value of the channel with the lowest or highest detection in the *Min* or *Max* position. In unlinked mode (**Channels Unlink**, chapter 7), each channel keeps its own value.
5. **The targets.** Each target adds an amount proportional to *m* to the setting it modulates.

Modulations only act on the main compressor: not on the bias detection and envelope (chapter 20), nor on the lookahead's dedicated follower (chapter 22). They are computed in the present chain and, when a lookahead is active, in the future chain, with separate memories, so the anticipated envelope is modulated as well.

> **When the spectrum drives the dynamics**
>
> Making compression depend on frequency content is a long-standing concern. In radio, in the early 1970s, Mike Dorrough and then Bob Orban split the signal into bands, because in wideband processing the bass "modulates everything else" and the listener hears the pumping. De-essers, for their part, EQ the sidechain to make the compressor more sensitive to sibilance. In 1998, the Weiss DS1 measured both peak and RMS and chose between a fast and a slow release based on their ratio: the release time already depended on the character of the signal. Comprendre's modulations extend these ideas: it's no longer just the detected level that depends on timbre, but the ratio, the gate and the times themselves.

The sources

What gets measured

Each source measures, **at every sample**, how much a stage of the detection branch changes the signal's **instantaneous amplitude**:

- for the **DEQ** and the **DEQ 2**: |output| − |input|;
- for the **WS**: |input + saturation product| − |input|, where the saturation product is the one coming out of the **F (ws)** tab processing (chapter 16).

Each source's factor is then 1 + (this difference) × (source control). An EQ that **adds** energy in a band where the signal has some gives, on average, a positive difference; an EQ that **removes** energy gives a negative difference. The sign of the source control lets you reverse the direction of the modulation.

There are five consequences you should know about.

- **It's the detection-branch modules** (**INPUT** tab) that serve as sources, not the target's modules (**OUTPUT** tab). Their frequency, gain and saturation settings are those of the **DEQ**, **WS**, **F (ws)** and **DEQ 2** tabs, shared by both branches.
- **You don't have to hear them.** Setting a source to a nonzero value activates the corresponding module in the detection branch, even if its amount in the **INPUT** tab (**FX deq**, **FX ws**, **FX deq**) is at 0. The module then serves purely as a modulation source, without changing anything in what the compressor measures. That's what François does in his demonstration: he doesn't want to hear the EQ, just use it to modulate.
- **But watch out for the target.** The settings in the **DEQ**, **WS** and **DEQ 2** tabs also apply to the target branch, whose amounts (**deq**, **ws**, **deq** in the **OUTPUT** tab) are 1 by default: as soon as a DEQ band is set, you hear it on the compressed signal. If the module is only meant to serve as a source, set its amount to 0 in the **OUTPUT** tab: the first **deq** for the DEQ, **ws** for the WS, the second **deq** for the DEQ 2 (chapter 14).
- **The module has to be set up.** A source only has an effect if its module is actually doing something: at least one DEQ band with a nonzero frequency and a nonzero **min** or **max**; for the WS, non-neutral shaping and a nonzero amount (**Fet gr** or **Fet env**). Otherwise, the factor stays at 1.
- **The source is itself dynamic.** The gain of the DEQ bands varies between **min** and **max** according to the compressor's envelope (chapter 18), and the amount of WS saturation depends,

by default, on the gain reduction (chapter 17). With different **min** and **max** values, the source therefore changes with the compression, which itself changes with the modulation. For a source that depends only on frequency, give **min** and **max** the same value.

An absolute, instantaneous measurement

The difference is computed in linear amplitude (1 = 0 dBFS), sample by sample. It isn't relative to the signal level: for the same frequency content, a loud signal gives a larger modulation than a quiet one. And because it follows the waveform, it **oscillates at audio frequency**: without smoothing, the modulation jitters on every half-cycle. Smoothing (**mod att secs**, **mod rel secs**) is therefore almost always necessary.

The next figure simulates this measurement on a simple example. The DEQ has a single band, the mid band (a band-pass) centered on 2 kHz, with a resonance of 1 and **min** = **max** = 1: at the center of the band, it doubles the amplitude. The signal is a sine wave with an amplitude of 0.4, first at 200 Hz, then at 2 kHz, then at 200 Hz again; **deq mod** is 1.

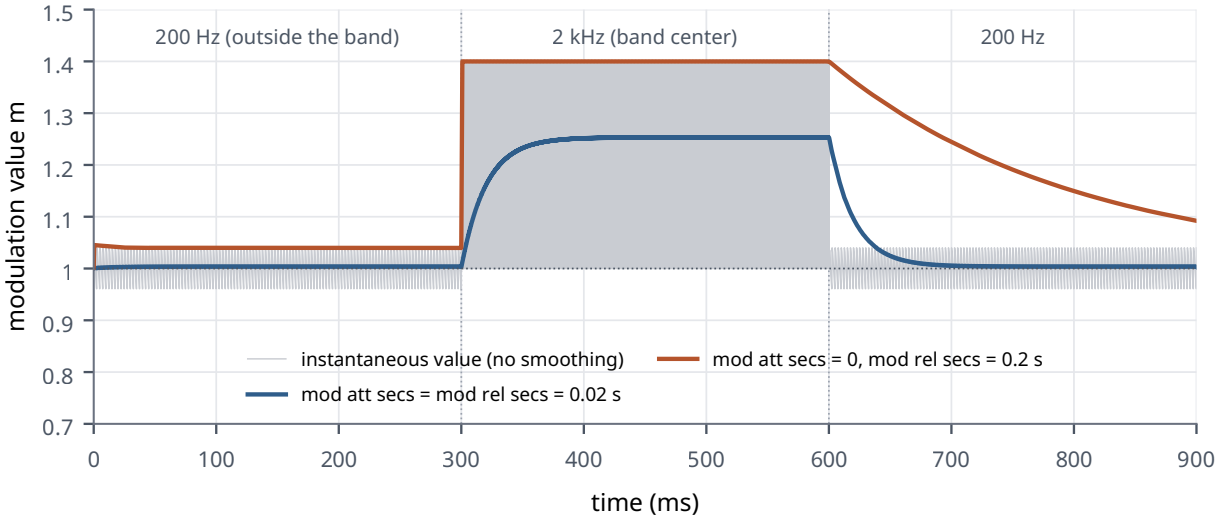


Figure 136 — Modulation value measured on the DEQ (mid band at 2 kHz). At 200 Hz, outside the band, the instantaneous value (gray) oscillates slightly around 1. At 2 kHz, it oscillates between 1 and 1.4, in step with the waveform. Smoothed with equal times of 20 ms (blue), it settles around 1.25; with an instantaneous rise and a 0.2 s fall (orange), it follows the peaks and comes back down slowly.

Combining the sources

The three factors are **multiplied**: P = (1 + a) × (1 + b) × (1 + c), where a, b and c are the deviations measured on the DEQ, the WS and the DEQ 2. For small deviations, this is almost the same as adding them (P ≈ 1 + a + b + c); for large deviations, the product drifts away from the sum. For example, two deviations of +0.5 give 2.25, not 2. A factor of zero (a deviation of −1) cancels the whole product.

deq mod

MOD & LA tab

Range: −1 to 1 **Default:** 0 **ID:** `freq_deq`

Control for the first source: the amplitude difference produced by the detection branch's **DEQ** (the dynamic EQ placed before the waveshaper, **DEQ** tab).

- **0** (default): source inactive.
- **Positive**: when the DEQ increases the amplitude, the modulation value rises above 1; when it decreases the amplitude, the value drops.
- **Negative**: the direction is reversed.
- Values of ±1 give the factor 1 ± (|output| − |input|): the amplitude difference is passed on as is.

A nonzero value activates the detection branch's DEQ (if it has at least one band set), even if **FX deq** is at 0 in the **INPUT** tab. The knob's travel is very gradual near 0 (about 0.004 halfway between the center and one end stop), so type precise values into the box.

To amplify the effect beyond ±1, use **mod amp**.

ws mod

MOD & LA tab

Range: −1 to 1 **Default:** 0 **ID:** `freq_fet`

Control for the second source: the saturation produced by the detection branch's **WS**. The measured difference is |input + saturation product| − |input|, where the saturation product is the one coming out of the **F (ws)** tab (filters and limiter included), before it's blended in by **FX ws**.

- **0** (default): source inactive.
- **Positive** or **negative**: same logic as for **deq mod**.

The waveshaper must be configured (non-neutral shaping, and a nonzero **Fet gr** or **Fet env**); a nonzero value then activates it in the detection branch, even if **FX ws** is at 0. Since the amount of saturation depends by default on the gain reduction (**Fet gr**), this source is zero as long as the compressor isn't reducing gain.

Like the DEQ, the waveshaper set in the **WS** tab also acts, by default, on the compressed signal (**ws** amount in the **OUTPUT** tab): set this amount to 0 if the waveshaper should only serve as a source.

This source reacts to the way saturation changes the waveform, and therefore to both the level and the harmonic content of the signal. François presents it as a more extreme use than the DEQ, but an entirely viable one.

deq mod

MOD & LA tab

Range: −1 to 1 **Default:** 0 **ID:** `freq_dep`

Control for the third source: the amplitude difference produced by the detection branch's **DEQ 2** (the dynamic EQ placed after the waveshaper). It works the same way as **deq mod**: 0 disables the source; a nonzero value activates the detection branch's DEQ 2 (if it has at least one band set), even if its **FX deq** amount in the **INPUT** tab is at 0.

Since the DEQ 2 sits after the waveshaper, its measurement applies to the already saturated signal, if the WS is blended into the detection.

This knob carries the same "deq mod" label as the first one: it's the third knob in the tab, to the right of **ws mod**.

To be confirmed by François

The two DEQ sources carry the same label, "deq mod," in the interface and in the host's automation. Would a distinct label for the third one (for example "deq2 mod") be preferable?

♪ In the studio: choosing the band in charge

The source decides what the compressor will be sensitive to. On a drum bus, a low DEQ band around 60 to 80 Hz makes the modulation react to every kick hit; on a vocal, a band around 3 to 5 kHz makes it react to harshness and sibilance; on a mix, a band around 200 to 300 Hz follows the thickness of the low mids. In every case, the EQ can be nothing more than an ear: leave **FX deq** at 0 in the **INPUT** tab, set the corresponding **deq** in the **OUTPUT** tab to 0, and it will only be used for modulation. The **ws mod** source, which reacts to the waveshaper's saturation, and therefore to both the level and the harmonic richness of the signal, is a more extreme tool, as François says. The **pa_elysia** preset uses it this way, with **ws mod** at −1 and a waveshaper you never hear (**ws** at 0 in the **OUTPUT** tab).

Shaping

The raw value P then goes through three stages, in this order: amplitude, power, smoothing.

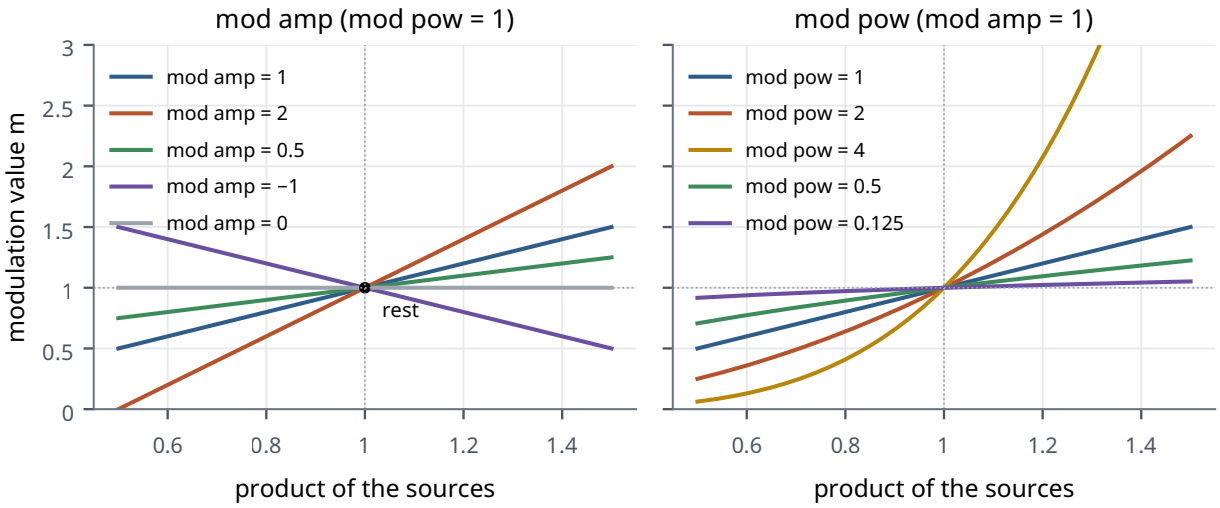


Figure 137 — Left: the effect of mod amp; the value 1 stays fixed and the deviation from 1 is multiplied. Right: the effect of mod pow; the value 1 stays fixed, and the curve sags or bulges on either side of it.

mod amp

MOD & LA tab

Range: −421 to 421 **Default:** 1 **ID:** `freq_amp`

Multiplies the modulation value's **deviation from 1**: m = 1 + (P − 1) × amp. The rest point, 1, doesn't move.

- **1** (default): the source value is used as is.
- **Greater than 1**: the effect is amplified. Since the measured amplitude differences are often small, values in the single digits, or even in the tens, may be needed.
- **Between 0 and 1**: the effect is attenuated.
- **0**: the modulation value stays at 1 permanently. The targets then sit at their rest value (see below), which is handy for setting that value.
- **Negative**: the direction of the modulation is reversed.

The range runs from −421 to 421, with very gradual travel: the default value, 1, sits at around three-quarters of the knob's travel.

François stresses this point: when you set the amplitude, the modulation always starts from 1, not from 0.

Beware of negative values

With a high amplitude, *m* can become negative (for example, P = 0.9 and amp = 20 give m = −1). Depending on the target, this can produce unexpected effects: a ratio slope that heads the other way, a time pulled down to 0. With **mod pow** other than 1, this is something to avoid at all costs (see the next box).

mod pow

MOD & LA tab

Range: 0.125 to 8 **Default:** 1 **ID:** `freq_puissance`

Raises the modulation value to a power: m = m^pow. The value 1 stays fixed.

- **1** (default): no change.
- **Greater than 1**: deviations are accentuated asymmetrically. Above 1, the value climbs faster; below 1, it gets squashed toward 0. For example, with pow = 2, m = 1.2 becomes 1.44 and m = 0.8 becomes 0.64.
- **Less than 1** (down to 0.125): deviations are softened, and values below 1 are pulled back toward 1.

The power is applied after the amplitude and before the smoothing.

To be confirmed by François

If the modulation value is negative (high amplitude) and **mod pow** is other than 1, the power calculation doesn't produce a number (NaN), and this value then stays trapped in the smoothing. Should the modulation value be clamped to 0 before the power?

mod att secs

MOD & LA tab

Range: 0 to 2 s **Default:** 0 s **ID:** `freq_vitesse_a`

Smoothing time constant when the modulation value **rises**. The smoothing is a one-pole filter: the smoothed value covers 63% of the gap in this time, and 95% in three times this time.

- **0** (default): the rise is instantaneous.
- **A few milliseconds to a few tens of milliseconds**: the value follows the envelope of the measurement without jittering at audio frequency.

Mind the vocabulary: "att" here refers to the **rise of the modulation value**, not to the compressor's attack. Depending on the target and the sign of the settings, a rise in *m* can just as well increase the compression as decrease it.

If **mod att secs** and **mod rel secs** are both 0, there's no smoothing at all: the modulation value follows the instantaneous measurement.

mod rel secs

MOD & LA tab

Range: 0 to 2 s **Default:** 0 s **ID:** `freq_vitesse_r`

Smoothing time constant when the modulation value **falls**. It works the same way as **mod att secs**.

With a fast rise and a slow fall, the smoothed value follows the **peaks** of the measurement and holds them: it behaves like a peak detector for the monitored band. With equal times, it tends to follow the **average** instead. The figure in the “The sources” section shows both cases.

> **A word from the author**
>
> François recommends always putting a little time into the smoothing: it’s the best way to see what the modulation is doing, instead of a value that jitters.

The rest value

All the targets work the same way: they **add** an amount proportional to *m* to the setting they modulate. And *m* is 1 at rest. This leads to an important rule, which François points out in his video: **the modulation isn’t centered on the main control**.

- With m = 0, the target adds nothing: you get the value of the main control (**Ratio**, **Ratio Gate**, **Attack** or **Release**).
- At rest (m = 1), you get the main control **plus** the target’s setting.
- Beyond that, the deviation keeps going in the same direction, in proportion to *m*: at m = 2, the target’s setting has been added twice. For the ratio, this proportionality applies to the slope (1/ratio), not to the ratio itself (see **mod ratio**).

So to get a given value at rest, you have to split it between the two: for example, for a 1 s release at rest that gets shorter when the monitored band is present, set **Release** to 1.5 s and **mod release** to −0.5 s. At rest, 1.5 − 0.5 = 1 s; at m = 1.5, 1.5 − 0.75 = 0.75 s; at m = 3, the release is zero.

> **Seeing the rest value**
>
> With a source active, temporarily set **mod amp** to 0: *m* stays at 1, and the transfer curve graph (**CURVE** tab) shows the curve at rest.

The targets

mod ratio: a signal-dependent ratio

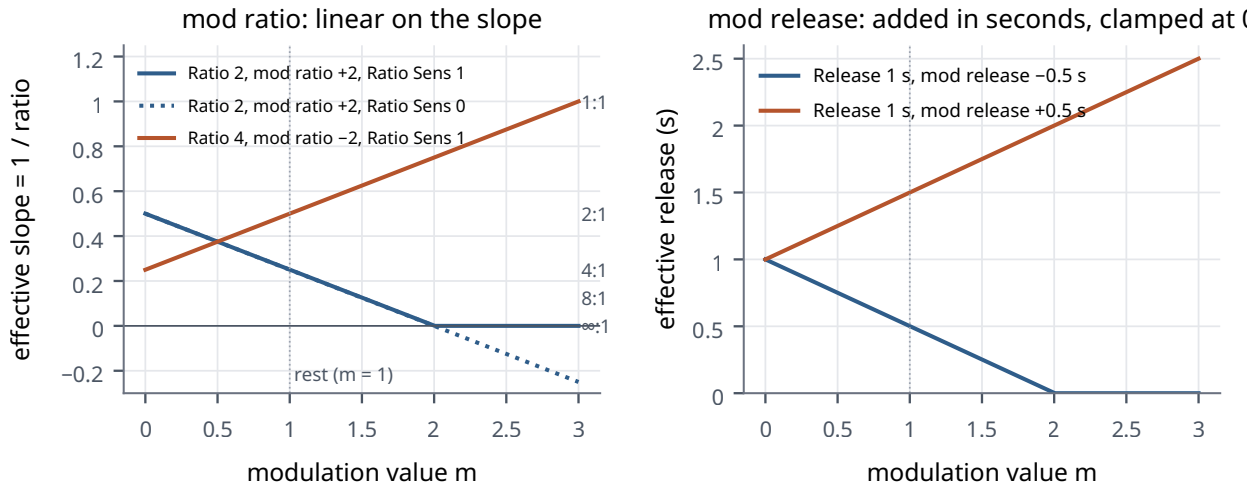


Figure 138 — Left: the effective slope (1/ratio) as a function of the modulation value; it varies in a straight line. With Ratio Sens at 1, it doesn't go below 0 (∞:1); with Ratio Sens at 0, it can become negative. Right: the effective release with mod release.

mod ratio

CURVE tab

Range: −42 to 42 **Default:** 0 **ID:** `freq_ratio`

Modulates the **slope** of the transfer curve (chapter 10). The control is expressed in ratio units: it's the amount added to the main **Ratio at rest**.

- **0** (default): no ratio modulation.
- **Positive**: at rest, the ratio is **Ratio** + **mod ratio**. For example, **Ratio** 2 and **mod ratio** +2 give 4:1 at rest.
- **Negative**: at rest, the ratio is reduced by that amount.

If **Ratio** + **mod ratio** reaches 42, the slope at rest is zero (infinite ratio); if it drops to 0.25 or below, the slope is 4.

The calculation isn't done on the ratio itself, but on the **slope**, which is 1/ratio. The slope varies in a straight line with *m*: it's 1/**Ratio** at m = 0 and 1/(**Ratio** + **mod ratio**) at m = 1, and it keeps going in the same direction beyond that. With **Ratio** 2 and **mod ratio** +2: 2:1 at m = 0, 4:1 at rest, 8:1 at m = 1.5, and infinity at m = 2.

The limits depend on **Ratio Sens** (**CURVE** tab, chapter 10):

- **Ratio Sens at +1** (default): the slope stays between 0 (infinite ratio, "42") and 4 (a ratio of 0.25:1). The modulation can't go beyond infinity: a modulated ratio never becomes negative.
- **Ratio Sens at 0**: the slope can go down to −4. The modulation can then make the ratio **negative** (the output decreases as the input increases), even if the main **Ratio** is positive.

A slope greater than 1 (a ratio below 1:1) gives a gain greater than 1, which only has an effect if **GR Min** allows it (chapter 10).

The modulation can make the compressor active even with **Ratio** at 1, a value at which it's normally bypassed: with **Ratio** 1 and **mod ratio** +1, the ratio is 2:1 at rest.

The transfer curve graph follows the modulation: you can watch it move in real time. The automatic knee (**Knee Auto**) also uses the modulated ratio.

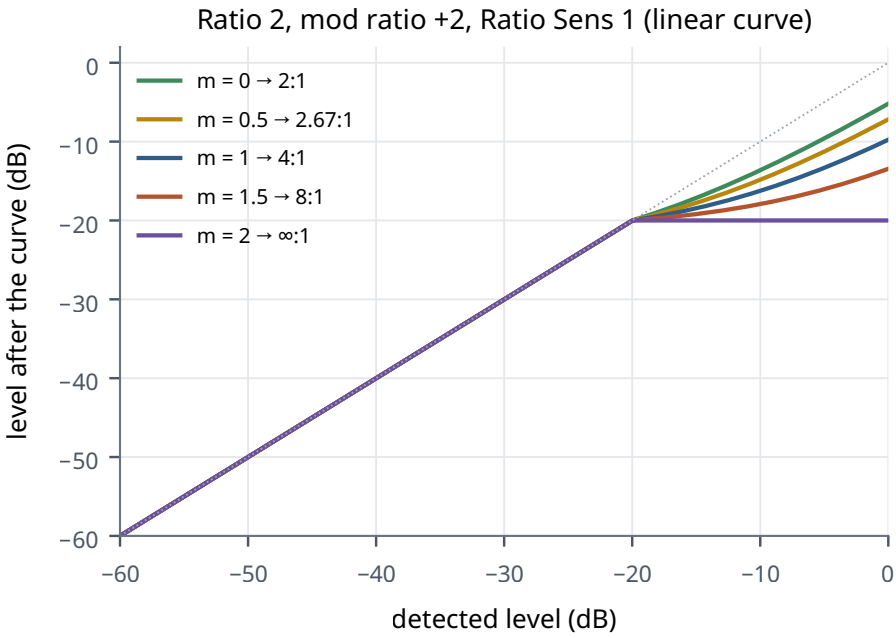


Figure 139 — The transfer curve for different values of m, with Ratio 2, mod ratio +2, Threshold −20 dB (linear curve, the default setting). The graph in the CURVE tab moves like this during playback.

A word from the author

François created the **Ratio Sens** control for the modulations: he wanted to be able to decide whether or not a modulated ratio is allowed to become negative. With the default value, the modulated ratio can't go beyond 42, that is, infinity; at 0, the modulation can push it negative.

To be confirmed by François

With **Ratio Sens** at −1, the code computes the ratio modulation even when **mod ratio** is 0: the added amount is then 2/**Ratio** × *m*, so that at rest (m = 1) the slope becomes positive again (+1/**Ratio**) and, with no source active, it depends on a modulation value that isn't computed. Shouldn't the negative ratio obtained with Ratio Sens −1 be independent of the modulations?

mod gate

CURVE tab

Range: –42 to 42 **Default:** 0 **ID:** `freq_porte`

Modulates the gate's **Ratio Gate** (chapter 10). Unlike **mod ratio**, the modulation is added directly to the gate ratio, without going through the slope: effective gate ratio = **Ratio Gate** + **mod gate** × *m*.

- **0** (default): no gate modulation.
- **At rest**, the gate ratio is **Ratio Gate** + **mod gate**.
- The result is clamped between **0.25** and **42**.

Since **Ratio Gate** is directly the slope below the gate threshold, a value greater than 1 closes the gate (downward expansion), and a value less than 1 brings up low levels, if **GR Min** allows it. The modulation can activate the gate even when **Ratio Gate** is 1 (its default value, at which the gate is normally inactive): with **mod gate** +1, the gate has a ratio of 2 at rest.

This way you can get a gate that closes further, or on the contrary opens up, depending on the signal's frequency content. Here too, the transfer curve graph follows the modulation.

> **♪ In the studio: a ratio that hardens with the timbre**
>
> On a vocal, use a DEQ band around 3 kHz as the source, set up as in the workflow at the end of the chapter, with **Ratio** at 2 and **mod ratio** at +1: the vocal is compressed at 3:1 at rest, and more firmly when it gets edgy in the upper mids. Soft, dark phrases breathe, and shouted passages are held in check. The **pa_elysia** preset, whose name probably refers to the emulation of the elysia alpha compressor, starts from a **Ratio** of 0.7 and a **mod ratio** of 1.75, so its ratio is 2.45:1 at rest. Its source is the saturation of the detection waveshaper (**Fet Thresh** at 0.11, **Fet Knee** at 32.8), whose amount follows the gain reduction: with a negative **ws mod**, the modulation value, and therefore the ratio, rise as the compressor works harder. The smoothing is fast on the rise (**mod att secs** at 0.002952 s, about 3 ms) and slower on the fall (**mod rel secs** at 0.1418 s, about 140 ms), so it follows the peaks. With **mod gate**, the same logic applies to the gate, for example to close a tom mic further when the cymbals dominate it.

mod attack and mod release: signal-dependent times

These two targets modulate the times of the envelope follower placed after the transfer curve, that is, the **Attack** and **Release** knobs in the main row (chapter 12). These are the only times that can be modulated: neither the detection envelope (**A/R (in)**), nor the bias (**A/R (bs)**), nor the lookahead's dedicated follower (**A/R (la)**) can be.

mod attack

A/R (cv) tab

Range: −2 to 2 s **Default:** 0 s **ID:** `freq_attaque`

Adds to the **Attack** time a duration proportional to the modulation value: effective attack = **Attack** + **mod attack** × *m*, brought back to 0 if it becomes negative.

- **0** (default): no attack modulation.
- **At rest**, the attack is **Attack** + **mod attack**. A negative setting shortens the attack as *m* rises; a positive setting lengthens it.

The modulated attack is recalculated at every sample. It's the value then used to compute the automatic attack (**Attack Auto**, applied after the modulation), by all the follower variants in the **ENV (cv)** tab, and for the opto designer's times (chapter 15).

This control only has an effect if a **source** is active (**deq mod**, **ws mod** or **deq mod** other than 0) and if the envelope follower is active (in practice, **Attack** or **Release** greater than 0) or the opto designer is engaged. No history curve displays the modulated attack.

mod release

A/R (cv) tab

Range: −2 to 2 s **Default:** 0 s **ID:** `freq_relachement`

Adds to the **Release** time a duration proportional to the modulation value: effective release = **Release** + **mod release** × *m*, brought back to 0 if it becomes negative. The automatic release (**Release Auto**) is applied afterward.

- **0** (default): no release modulation.
- **At rest**, the release is **Release** + **mod release**. With **Release** 1 s and **mod release** −0.5 s: 0.5 s at rest, 0 s at m = 2, and 1 s if *m* drops back to 0.

Same conditions as **mod attack**: an active source and an active envelope follower.

This is the easiest target to observe, because the modulated release has its own curve in the history (the pink curve, see below). François recommends starting with it, with a long release (1 to 2 s), so you can clearly hear and see the effect.

To be confirmed by François

With no source active, **mod ratio**, **mod gate** (and **mod attack** or **mod release** if an Auto control in the **A/R (cv)** tab is active) use a modulation value that isn't computed: in linked mode, it's undefined; in unlinked mode, it stays frozen at its last value. Should these targets be neutralized when no source is active?

For predictable behavior, then, always set at least one source (**deq mod**, **ws mod** or the second **deq mod**) to a nonzero value before using a target.

> **♪ In the studio: times that follow the music**
>
> On a drum bus, use a low band around the kick as the source and set a negative **mod release**: the compressor releases faster right after each kick hit, and more slowly when only the cymbals and the room are ringing. The bus quickly gets back up to level between hits, and keeps a gentler release when the kick drops out. On a snare or a guitar, a positive **mod attack**, driven by the band where the attack lives and with a fast smoothing rise (**mod att secs** on the order of a millisecond), lengthens the attack at the precise moment the transient arrives: the crack gets through, and the rest is held in check.

Reading the modulations on the graphs

Two displays let you follow the modulations.

- **The transfer curve graph** (**CURVE** tab) follows **mod ratio** and **mod gate**: the curve changes shape in real time, in step with the modulation value.
- **The pink curve in the history** shows the modulated release. It plots the ratio between the effective release (after modulation and automatic release) and the **Release** setting, averaged over each column of the history.

In linear display (**Gfx Lin/Db** at 0, the default), the pink curve sits at the **top** of the graph when the release is unchanged (a ratio of 1) and at **half height** when it's halved. A lengthened release (a ratio greater than 1) goes off the top of the graph and can't be read. In decibel display, the ratio is converted to dB: 0 dB is at the top, and half the release sits 6 dB lower, that is, at 90% of the height with the default 60 dB scale (**Gfx DB** control).

The pink curve is only computed if **mod release** is other than 0 or **Release Auto** is greater than 0, and if the envelope follower is active; otherwise it stays at the bottom of the graph. It's displayed if **Gfx Histos** (**GRAPHS** tab) contains the letter **a** (chapter 4). With **Release** at 0, the ratio is meaningless and the curve is unusable.

There's no curve for the modulated attack, nor for the modulation value itself. As with all the graphs, it's the present chain that's displayed.

Workflow: a frequency-sensitive ratio

This workflow is based on the one François uses in his videos. The goal: a compressor whose ratio increases when a frequency band is present in the signal, without that band being emphasized in the detection.

1. **Build the source.** In the **DEQ** tab, set a band on the frequency range that should trigger the modulation: for example the mid band, with **Deq mid herz** on the center frequency and **Deq mid reso** set for the width you want. Give **Deq mid min** and **Deq mid max** the same positive value (for example 1): the band is then boosted at all times, and the source depends only on the frequency content. Leave **FX deq** at 0 in the **INPUT** tab, so the detection doesn't hear this EQ. Also set the first **deq** in the **OUTPUT** tab to 0; otherwise, you'd hear this band on the compressed signal.
2. **Activate the source.** In the **MOD & LA** tab, set the first **deq mod** to 1.

3. **Smooth.** Set **mod att secs** and **mod rel secs**, for example to 20 ms each to follow the band's average, or a fast rise and a slower fall to follow its peaks.
4. **Choose the target.** In the **CURVE** tab, set **mod ratio**. Remember that the ratio at rest is **Ratio** + **mod ratio**: for 4:1 at rest, use for example **Ratio** 2 and **mod ratio** +2. The ratio then increases when the band is present, and becomes infinite when *m* reaches 2.
5. **Dial it in.** Play the signal and watch the transfer curve graph move. If it barely moves, increase **mod amp**; if it moves too abruptly, lengthen the smoothing.
6. **Fine-tune. mod pow** makes the response asymmetrical; **Ratio Sens** at 0 allows the modulation to make the ratio negative.

> **A word from the author**
>
> In his demonstration, François first tries letting the ratio go negative, observes behavior that surprises him, then goes back to a more conventional setting (Ratio Sens at 1). He then finds that the ratio moves too abruptly, and softens it by adjusting the smoothing speed.

Workflow: a frequency-dependent release

1. Build a source as above (a DEQ band, the first **deq** in the **OUTPUT** tab at 0 if you don't want to hear it, **deq mod** at 1, smoothing of a few tens of milliseconds).
2. Set a long **Release**, for example 1 to 2 s: the effect of the modulation will be easier to hear and see.
3. In the **A/R (cv)** tab, set a negative **mod release**: the release gets shorter when the band is present. With **Release** 1 s and **mod release** −0.5 s, it's 0.5 s at rest and gets shorter still as *m* rises.
4. Watch the pink curve in the history: in linear display, it should hover around half height at rest (release halved), and drop when the band is present. If it doesn't move, check that the source is active (module set up, nonzero **deq mod**) and that **mod release** really is other than 0.
5. Adjust **mod amp** and the smoothing until you get the range of variation you want.

> **A word from the author**
>
> François acknowledges that this setting takes some practice, but it lets you modulate the release according to frequency in a very flexible way, and even according to a frequency that is itself dynamic. By also engaging the DEQ in the detection (nonzero **FX deq**), he feels you end up with a kind of overall coherence that's almost closer to mastering than to compression.

Starting ideas

- **A vocal that holds steady when it gets harsh.** DEQ mid band around 3 kHz (**Deq mid min** and **Deq mid max** at 1), without hearing it (**FX deq** in the **INPUT** tab and the first **deq** in the **OUTPUT** tab at 0), **deq mod** at 1, 20 ms smoothing, **Ratio** 2 and **mod ratio** +1: 3:1 at rest, more when the vocal gets edgy.

- **A drum bus that breathes with the kick.** DEQ low band around 60 Hz (**Deq low min** and **Deq low max** at 1, without hearing it), **deq mod** at 1, **Release** 1 s and **mod release** −0.5 s: 0.5 s at rest, shorter on each hit.
- **Toward overall coherence, on a mix or a master.** The path François describes: a release modulated by a DEQ band, that same DEQ engaged in small doses in the detection (**FX deq**), slow smoothing (50 to 200 ms) and small amplitudes. Proceed in small steps, comparing level-matched.
- **For sound design.** A **mod amp** in the single digits and **Ratio Sens** at 0: the ratio can turn negative in time with the timbre. In that case, leave **mod pow** at 1 (see the “Beware of negative values” box) and keep an eye on your levels.
- **To see an example.** Load **pa_elysia** and watch the curve in the **CURVE** tab move during playback.

CHAPTER 24

Analyzing and Imitating a Compressor

For François, analysis is the product's real added value. At the end of his long presentation video, he deliberately comes back to analysis, which he had forgotten along the way, because it is the plugin's very reason for being: its name itself promises to understand compressors (*comprendre*) and to take them (*prendre*).

Comprendre was designed from the outset to be able to imitate other compressors. For this, it has an analysis tool: it displays another compressor's gain curve on its own history graph. You then adjust Comprendre until its curve lines up with the model's, and you check the result by ear by listening to the difference between the two.

This approach also helps you get to know Comprendre itself better: you can analyze one instance of Comprendre with another, to see precisely what a setting does.

What makes the tool so valuable is that you don't need to own the compressor being imitated: a dry excerpt and its processed version are enough. François himself finds it a little crazy that you can analyze a compressor you don't have. A mix engineer can recapture the behavior of a unit heard on a session whose tracks were kept before and after processing, or that of their own hardware compressor; teachers and students can finally see, on a graph, what sets apart two compressors the ear struggles to tell apart; and anyone can build their own presets, just as François built the eighty that are included.

How it works

The compressor to be analyzed receives the same signal as Comprendre. Its output is sent to Comprendre's **sidechain** input. For each sample, Comprendre divides the sidechain level by the level of its own input: if the two signals are identical apart from a gain, this ratio is exactly the **gain applied by the other compressor**. This ratio, averaged over each column of the history graph, is drawn in **gold** (semi-transparent bars rising from the bottom), on the same scale as the **white** envelope, Comprendre's gain.

Analyse SC

MAIN tab

Range: 0 to 1 **Default:** 0 **ID:** `analyse`

Turns on analysis of the sidechain signal: the gold curve in the history graph then shows the average ratio between the sidechain level and Comprendre's input level. When analysis is turned off, the calculation is reset and the gold curve disappears.

Analysis has **no effect on the sound**: it is for display only. It is independent of the **Sidechain** setting, which feeds the sidechain into the detection (chapter 14). For an analysis, leave **Sidechain** at 0: Comprendre must detect its own input signal.

> **Measuring rather than rebuilding**
>
> Developers who emulate hardware as plugins have two main methods. Circuit modeling rebuilds each component digitally. Measurement-based modeling, known as "black box" modeling, simply observes what goes in and what comes out, at many levels, on many sources and for many settings. This second method has a limit, which Arvid Rosén of Softube sums up this way: with twenty knobs, the number of combinations to measure becomes astronomical. Most developers therefore combine both approaches, with a lot of listening. Comprendre's analysis mode puts the "black box" approach in your hands, at your own scale: one setting of the model at a time, your own sources, and your ears as the judge.

The setup

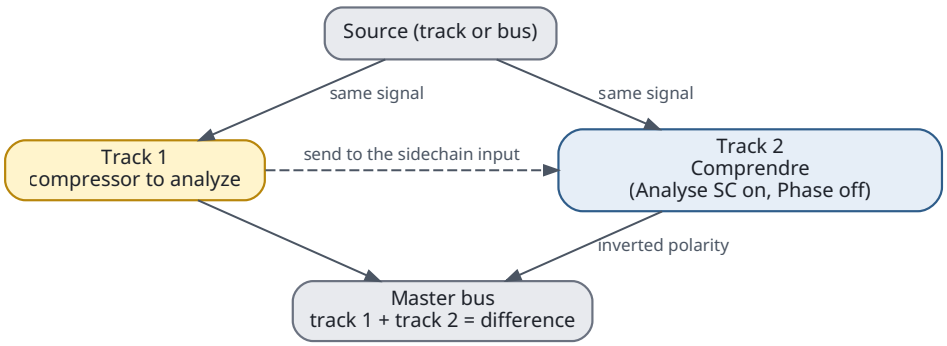


Figure 140 — Setup for analyzing a compressor.

1. Put the source to be analyzed (a vocal track, a drum track…) on **two tracks**, or send it to two buses. Both must receive exactly the same signal.
2. On **track 1**, insert the compressor to be analyzed, with the settings you are interested in. Send the output of this track to Comprendre's **sidechain** input, while also letting it go out normally to the master.
3. On **track 2**, insert Comprendre. In the **MAIN** tab, check **Analyse SC**.
4. Still in the MAIN tab, **uncheck Phase**: Comprendre's output is then inverted. At the master, the sum of the two tracks lets you hear only their **difference**.

5. Turn down the master level so that the difference, when it is large, isn't deafening: at first, the two compressors are not doing the same thing and the difference can be considerable.

> **A word from the author**
>
> In his demonstration, François turns the master down by about a dozen decibels before starting, so that the difference doesn't drown out everything else.

> **Analyzing a compressor you don't own**
>
> The compressor on track 1 is not essential. If you have a **dry** recording and its **compressed** version (for example the tracks of a mix for which you have the versions before and after processing), put the dry version on Comprendre's track and the compressed version on the sidechain. The two must be perfectly aligned in time.

> **♪ In the studio: imitating your own hardware**
>
> If you own a hardware compressor, the previous tip lets you turn it into a preset. Record the dry signal and its version passed through the unit at the same time, on two tracks (through an insert send and return, for example), align them to the sample by compensating for the converters' latency, then analyze the processed track. You will then get its behavior back without the unit, on any track, and can even push it further. Just remember that a compressor does not react the same way at every setting or at every level: the preset will imitate the unit at the setting you recorded. Note that setting in the preset's name.

The method

Play a representative passage, then:

1. **Compare the curves.** The gold curve shows what the model does; the white envelope, what Comprendre does. Slow down the history graph with **Gfx Secs** to see the attacks and releases clearly (chapter 4).
2. **Set the threshold and ratio first** so that the two curves have the same depth of reduction on loud passages and the same moments of rest.
3. **Then set the attack and release**, followed by their **shape**: does the gold curve come back up exponentially (logarithmic mode, chapter 12), in a straight line (linear mode), or with a plateau (decoupled envelope or hold, chapter 11)?
4. **Refine** with the other modules: RMS detection or detection envelope, curve in dB or linear, knee, automatic release, offset…
5. **Listen to the difference**: the quieter it is, the more faithful the imitation. A difference that contains nothing but saturation artifacts shows that the gain is well reproduced and that the color remains to be worked on (opto, waveshaper, bias…).

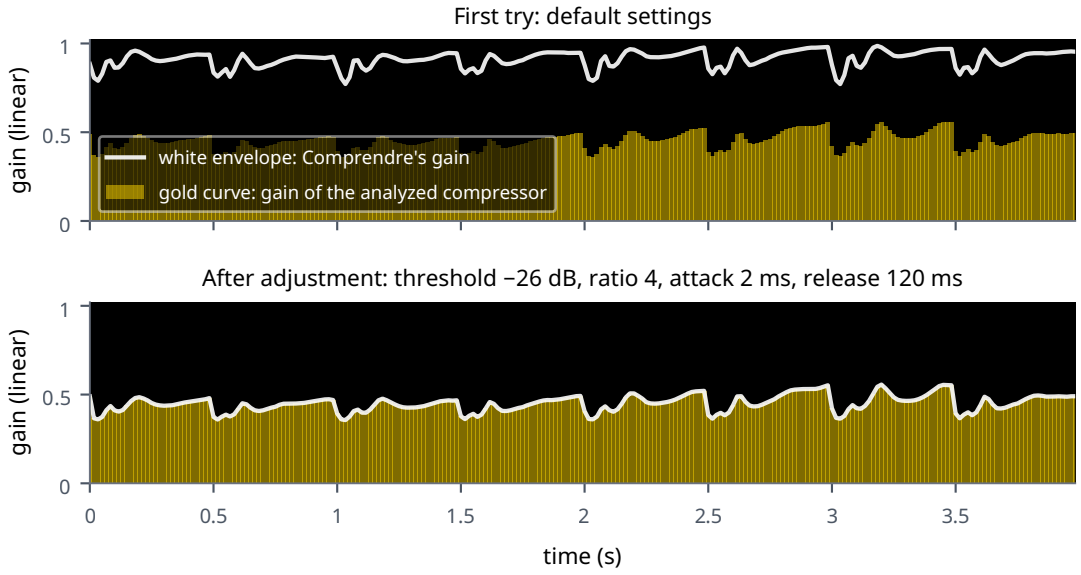


Figure 141 — The gold curve (the analyzed compressor) and the white envelope (Comprendre), before and after adjustment. Simulation.

A word from the author

In his demonstration video, François imitates the compressor that comes with his DAW. He copies the threshold and times displayed by that compressor, finds that the threshold seems a little too low, then notices that the model's release seems linear whereas Comprendre's is logarithmic: he then switches the release mode to linear and gets noticeably closer to the model. You don't always reach a perfect match (some compressors have undocumented internal behaviors), but you get close enough to build faithful presets.

♪ In the studio: choosing the right excerpt

The analyzed excerpt must make the compressor work in all its dimensions. A drum loop clearly shows the attack and release: each hit digs into the gold curve, which comes back up between two hits. A vocal with phrases that are sometimes loud, sometimes soft reveals the threshold, the knee and any program-dependent release. A sustained bass brings out the shape of the release and, in the difference, the grain of the saturation. In his videos, François also likes to use a square wave or a test sine wave, which make the times very easy to read. Analyze several excerpts: a setting that matches perfectly on drums may drift on vocals.

Precautions

- **Reference levels.** The ratio is calculated between the raw sidechain and Comprendre's input multiplied by **Input Db**. Leave Input Db at 0 dB during an analysis; otherwise the gold curve will be offset by the same amount.
- **The analyzed compressor must not change the average level in its own way.** Makeup gain on the model shifts the gold curve upward. Turn it off, or take it into account: the shape of the curve matters more than its absolute height.

- **Silences.** Samples where Comprendre's input is zero do not contribute to the ratio but do count in the column's average: during digital silence, the gold curve drops toward 0. This is not gain reduction by the model.
- **Timing.** If the analyzed compressor has latency (lookahead, linear-phase oversampling…) that your host does not compensate properly, the two signals are no longer aligned: the ratio loses its meaning and the difference you hear never cancels out. Check your host's latency compensation.
- **The level of the difference.** For the difference to cancel out, both tracks must have the same average level. The **Out->Sc** auto-volume (OUTPUT tab, chapter 8) can automatically align the level of Comprendre's output with that of the sidechain; remember to turn it off (42 s) to judge the work of the compressor itself.

Comprendre versus itself

By placing a second instance of Comprendre on track 1, you get an ideal test bench: you change a single setting on one of the two instances and watch, on the other, the gap between the gold curve and the white envelope, while hearing their difference. This is how François presents several subtle settings in his videos, for example linear or decibel tracking of the envelope (chapter 11).

> **♪ In the studio: a listening bench for learning**
>
> For teachers and students alike, two instances of Comprendre in opposite polarity make a valuable teaching tool. Set identically, they cancel each other out: the sum is silent, which François actually checks in his video on envelopes. Change a single setting on one of them, and you hear only what that setting changes: peak detection versus RMS detection, branching envelope versus decoupled envelope, linear curve versus decibel curve. The gold curve shows the gap; the sum lets you hear it. Nuances that are almost imperceptible in the mix then become obvious, and the ear gradually learns to recognize them without this aid.

From analysis to presets

The 80 presets included with Comprendre were born from this method. Each one bears the name of the plugin François took as a model: he placed it on the analyzed track, adjusted Comprendre until the white envelope hugged the gold curve, then reduced the difference by ear. The approach shows in the settings. Twenty-one presets offset the envelope with **Env Offset** (chapter 15), the setting François says he uses most for imitating, the one that has gotten him out of trouble many times: **slate_fg2a**, an imitation of an LA-2A emulation, sets it to −0.55, for example. Sixteen presets change the **Release Mode**, ten of which set it to pure linear, the setting that, in his video on analysis, brings him noticeably closer to the compressor he is imitating.

Keep two caveats in mind. A preset imitates a plugin at one setting of that plugin, which we don't always know, and at levels that are not necessarily yours: adjust the **Threshold** first. And when the imitated plugin itself emulates a piece of hardware, the preset is an imitation once removed. Chapter 25 presents these presets family by family and shows how their settings match the known be-

havior of their model; chapter 26 offers starting points depending on your line of work. Finally, nothing stops you from expanding the collection: with a dry excerpt, its processed version and a little patience, the next preset will be yours.

CHAPTER 25

The Presets: Eighty Compressors to Explore

Comprendre ships with 80 presets. Each one bears the name of a commercial plugin that François took as his model, and each one is the result of an imitation exercise: getting Comprendre to do what that plugin does, using nothing but its own settings. Taken together, they tell a short history of compression: the optical compressors of the 1960s, the transistors of the 1176, the tubes of the Fairchild, the SSL console, the Distressor, mastering processors, today's purely software inventions, and even a few saturators that don't compress at all.

This chapter takes you through the collection family by family. For each preset, it introduces the target, gives the complete list of modified settings, and shows how those settings answer the known behavior of the hardware or plugin being imitated. It is the best demonstration of what Comprendre can do: very different characters from one and the same plugin. It is also an excellent school, because each preset is an annotated example of how to use the modules described in the previous chapters.

How these presets were made

François built his presets with the method described in chapter 24. The plugin to imitate goes on a track, its signal is sent to Comprendre's sidechain input, and the **Analyse SC** mode displays its gain curve in gold next to Comprendre's envelope. He then adjusts the settings until the two curves line up, and listens to the difference between the two signals, with polarity inverted, to reduce it as much as possible.

> **A word from the author**
>
> In his video on analysis, François introduces his presets this way: settings that he believes come as close as possible to other compressors. He stays honest about the exercise: imitating the stock compressor of his DAW, he shows that he gets very close without quite matching it, and suspects some processing that the manufacturer doesn't document.

Two clarifications are therefore in order.

- **A preset imitates a plugin, at one setting of that plugin.** A compressor reacts differently depending on its threshold, its ratio and its times. The preset reproduces the behavior observed

under the conditions of the analysis, which we don't always know. When that plugin is itself the emulation of an analog unit, the preset is an imitation at one remove.

- **The names are pointers, not promises.** The file names keep the spelling François chose (`massenbug_drc2_mix`, `eli_arouser`…); the text gives the exact product names. All trademarks mentioned are the property of their respective owners. They are used solely to identify the models imitated, and their mention implies no affiliation with these manufacturers and no endorsement by them.

Loading a preset

Presets are loaded with the **Load** button on the MAIN tab (chapter 5). As soon as a preset loads, the names of the modified settings and the titles of the tabs involved turn yellow: you see at a glance what the preset has touched, and therefore where to look to understand its character. François shows this at the very start of his introduction video, on a preset picked at random.

> **Always start from a 'fresh' Comprendre**
>
> Only 15 of the 80 presets contain all the settings of version 1.0.3. The other 65 were saved with earlier versions of the plugin, and they lack the settings added since: depending on the preset, **Input Db**, **Output Db**, **L/R to M/S**, the clipper modes, the FIR filters and the lookahead settings, among others. And a setting that is missing from a preset **keeps its current value** on loading (chapter 5). If you load such a preset after loading another one, or after changing those settings, it will inherit their values. So before trying a preset, load your "default" preset (see the tip in chapter 5), or insert a new instance of Comprendre.

A few presets also contain settings that no longer exist under that name in version 1.0.3 and that the plugin ignores on loading. In most cases, these settings were at their neutral value; in the following presets, they had a different value: acqua_magenta, ao_buster, arturia_diode, arturia_tube-sta, blackrooster_omnitec436, bx_townhouse, caelumçsmmov, cytomic_glue, ddmf_mde, korneff_comp2, melda_mturbo, playfair_grading, spl_iron, ssl_1, ssl_1_b, ssl_1_f, ssl_2_b_, tdr_kotelnikov, tr_bus, tr_vintage and waves_rvox.

> **To be confirmed by François**
>
> These 21 presets contain settings from earlier versions (for example `mu_reverse`, `var`, `fet_ciseau`, `courbe_mix`) at a non-neutral value, which version 1.0.3 ignores: do they still sound as intended? In addition, 65 presets are incomplete and therefore inherit values from whatever was set before: would you like to re-save them with version 1.0.3, so that each one contains every setting?

Once the preset is loaded, adjust the **Threshold** first: François set each preset on a source and at a level that are not yours. Then compare level-matched, using the **Makeup** or the auto-volumes (chapter 8): heavier compression always sounds more impressive when it is also louder. No preset changes the oversampling, which stays at **1X**; on the presets that saturate heavily, try **2X** or **4X** (chapter 7) if you hear aliasing in the highs.

The preset map

The following figure gives an overview: for each preset, grouped by family, a yellow cell marks every tab in which at least one setting differs from its default value. These are exactly the tabs that turn yellow when you load the preset. Since the main row has no tab, it forms the first column. The display settings (GRAPHS tab, **Gfx Lin/Db**, **Gfx Secs**) are not counted.

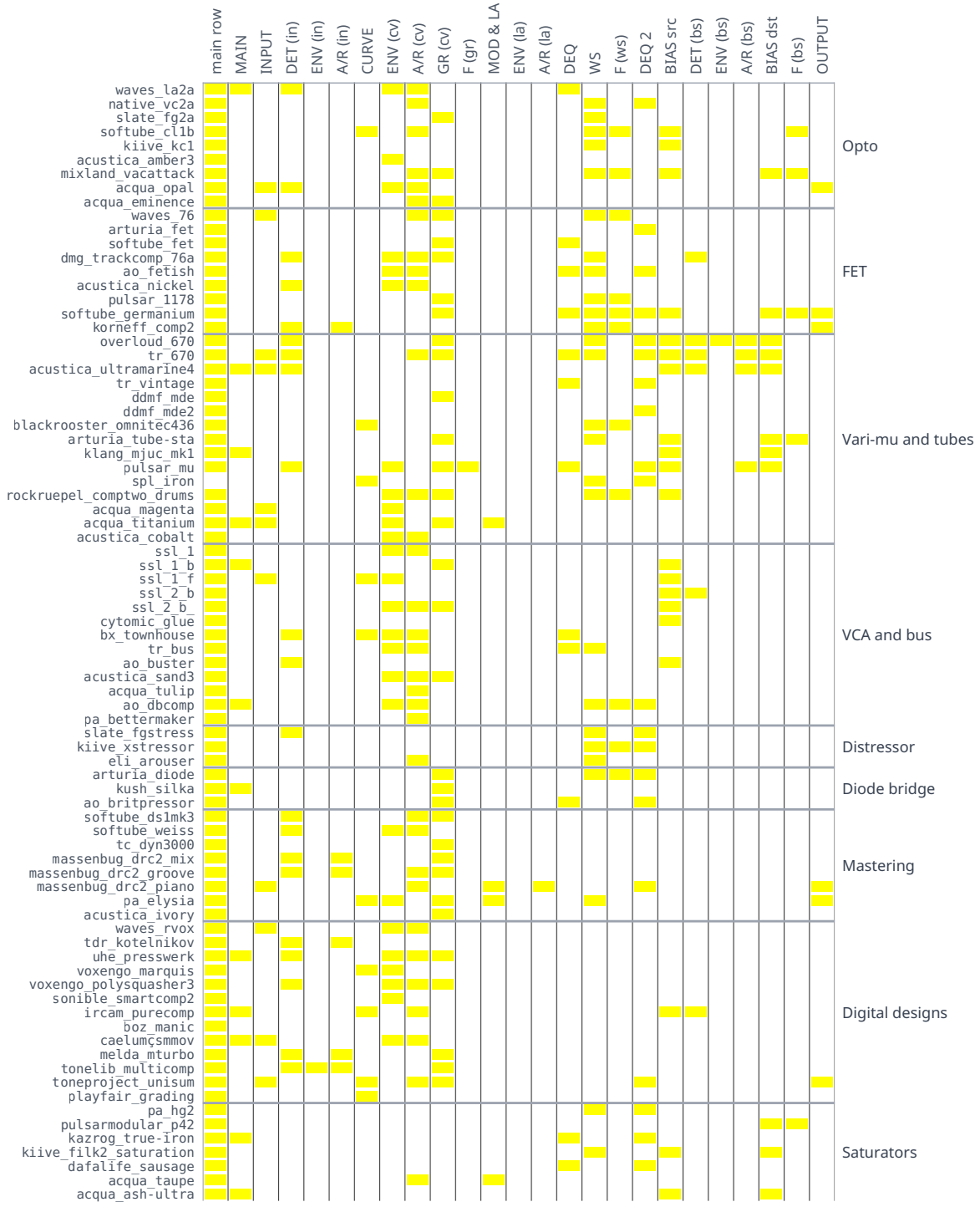


Figure 142 — The 80 presets and the tabs they modify, family by family. A yellow cell indicates that at least one setting in the tab differs from its default value.

A few numbers stand out. All the presets but one adjust the main row: threshold, ratio, attack and release remain the heart of any imitation. Next come the **A/R (cv)** tab (shape and automatic behavior of the attack and release, 33 presets), **GR (cv)** (powers, offset and opto designer, 32 presets), and **ENV (cv)** and the waveshaper (**WS**), with 26 presets each.

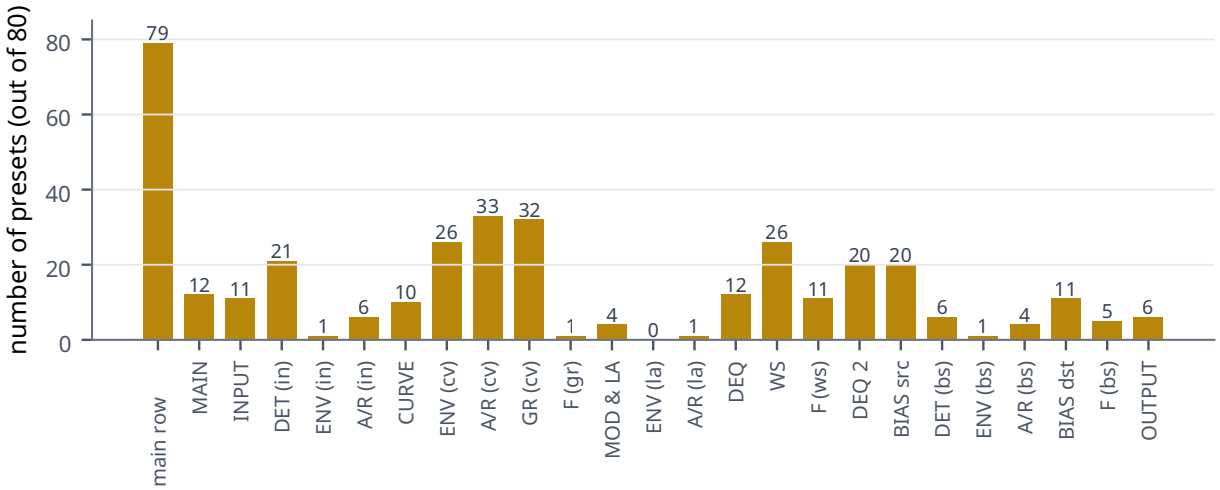


Figure 143 — Number of presets that modify each tab.

In detail:

- **41 presets** switch the transfer curve to decibels (**Cv Lin/Db** at 1), 37 keep the default linear curve, and two stop somewhere in between;
- **21 presets** shift the envelope with **Env Offset**, the setting François says he uses most to imitate compressors, the one that has “saved his life” many times;
- **26 presets** bring in the waveshaper, **20** the bias, and **6** the opto designer proper (nonzero **Opto**);
- **19 presets** have a knee (nonzero **Knee**), and **10** push the ratio to 42, that is, to infinity: they limit;
- **12 presets** use automatic release, **11** automatic attack, and **9** introduce feedback (**FF->FB**); only **3** touch the lookahead shift (**la delay**), one of them toward the past.

The transfer curve doesn’t tell the whole story of a compressor’s character, but it already tells a lot. The following figure overlays the static curves of the presets in each family: output level as a function of detected level, for the curve alone (main row and CURVE tab), before the envelope follower and the color modules. The four presets that use the automatic knee are left out, since their curve changes with the level of the sound. Be aware, too, of what the figure doesn’t show: the **Headroom**, the feedback, the powers and the envelope offset (GR (cv) tab) all change the compression you actually get. For several presets, the real curve is therefore quite different; each family’s section points this out.

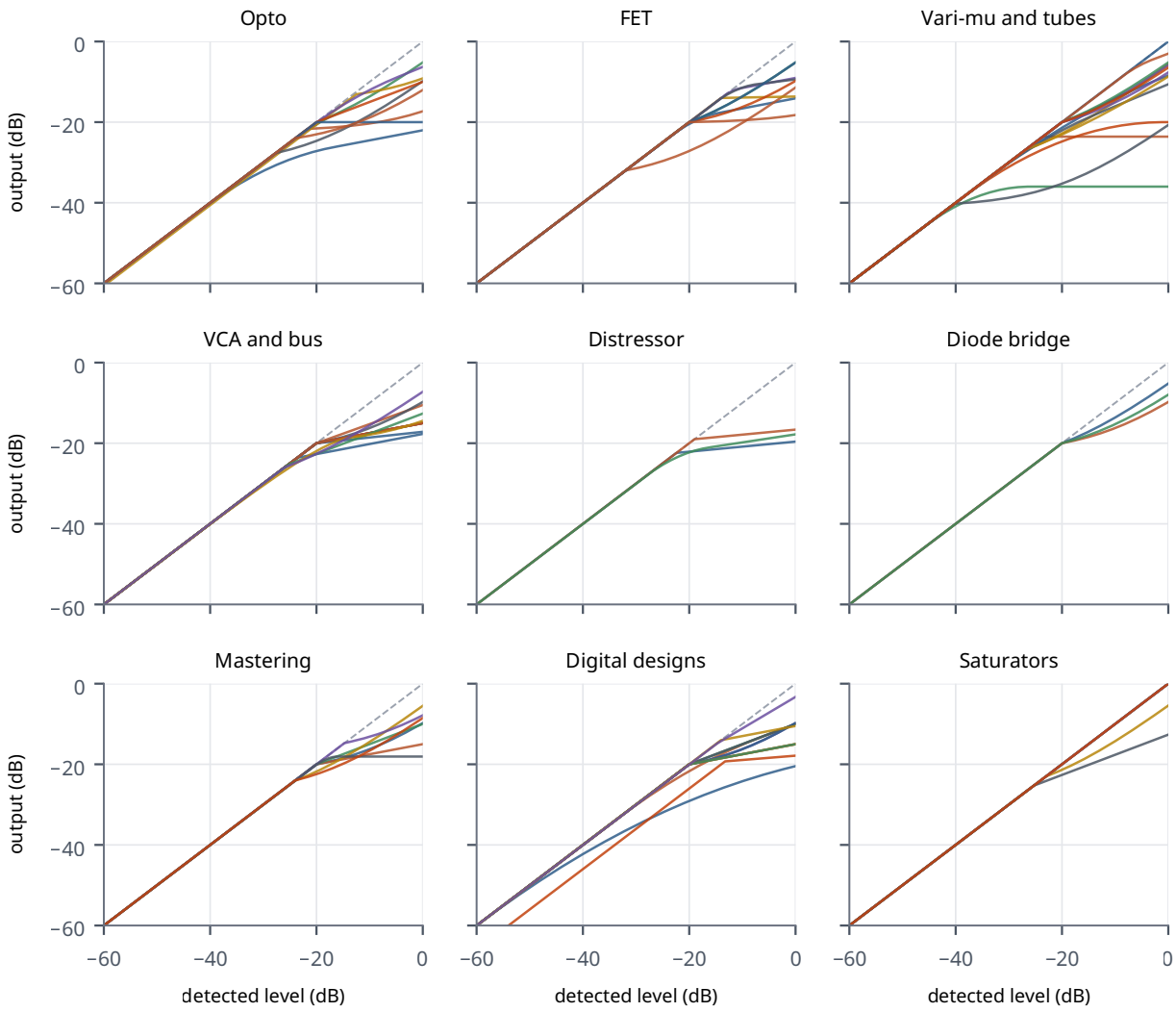


Figure 144 — Static transfer curves of the presets, by family (output versus detected level, in dB; dotted lines: no compression).

What the presets teach us

Reading the 80 presets setting by setting, and comparing them with the documented behavior of the hardware and plugins they imitate, gives you a better understanding of how François works, and of how you can work with Comprendre.

François imitates the behavior, not the mechanism. You might expect each family to go through "its" module: the opto designer for the optos, the waveshaper for the FETs, the bias for the tubes. That's only partly true. Only one of the nine opto presets uses the opto designer, and it uses it backward. The "optical" release, which is program-dependent, is obtained by other means: automatic release, release shape, the Mu follower, the linear modes. Likewise, six of the nine FET presets set up the waveshaper, but the speed of the transistor often comes from the shape of the attack rather than from the **Attack** knob. The analysis mode judges the result, the gain curve, not the path that leads to it.

The bias, right where you'd expect it. The vari-mu family is the exception: seven of its fifteen presets use the bias, and the three Fairchild 670 imitations all do it the same way, with a large height added at the detector input and a bias value taken from the output, smoothed by its own detector. This is the control loop described in the Pulsar Audio article that inspired the module. In overloud_670, the whole compressor is even built inside the bias: without it, the preset doesn't compress. And, as on a vari-mu, the ratio of these presets rises with level.

The offset is everywhere. Env Offset, François's favorite setting, appears in 21 presets. All three diode-bridge presets use it. The strongest, −0.55, belongs to slate_fg2a, an imitation of the LA-2A: François's remark about the LA-2A's offset holds true there, and not in waves_la2a, which has none.

The displayed ratio isn't always the ratio you hear. With a linear curve, in feedback, with an offset or a **Headroom**, the compression you actually get departs from the number on the **Ratio** knob. The ratio of 17 in native_vc2a behaves like roughly 4:1 on loud passages; the infinite ratio of acqua_opal, in full feedback, gives about 2:1 at steady state. Each family's section gives these equivalences.

The hardware's positions show up. When the imitated unit has fixed values, François often lands right on them: the 4:1, 30 ms, 0.1 s setting of the SSL bus compressor turns up as is in five presets; the threshold and ratio of waves_76 fall on the 4:1 position of the CLA-76; the three LA-2A imitations attack between 7.5 and 13.4 ms, around the documented 10 ms. To imitate the SSL bus compressor, six presets use the bias detection multiplication (**Bias input mul**), which François added precisely for that purpose.

Dynamics first, color second. Many presets reproduce the way the gain moves without trying to reproduce the grain of the hardware. None of the thirteen digital designs uses a color module; the diode-bridge presets barely imitate the saturation; two of the "saturators" are actually fixed EQs. There is room for you here: add the waveshaper or the bias to a preset whose dynamics you like.

What the presets don't use. No preset touches the three clippers, the FIR filters, the **Makeup**, **Output Db** or the auto-volumes, and all of them stay at **1X** oversampling. Modulations are used in only one preset, the lookahead in three. These tools remain entirely at your disposal for adapting a preset to your production.

Traces of exploration. A few modified settings have no effect in version 1.0.3: a Mu follower setting while that follower is disabled, a power left at its neutral value, bias settings while no amount activates the bias. They show up in yellow but change nothing in the sound; they are probably traces of the experiments made during the analysis. Each family's section points them out.

> **Three presets invert polarity**
>
> The presets ao_dbcomp, ssl_1_b and kazrog_true-iron are saved with the **Phase** checkbox unchecked: Comprendre's output is polarity-inverted (chapter 7). The first two also enable the **Analyse SC** mode. On a track heard on its own, a polarity inversion is barely audible, if at all, but it matters as soon as the signal is mixed with another version of itself (parallel compression on a bus, multi-mic recordings). Check **Phase** after loading if you don't want it.

Light that compresses: the optos

In an optical compressor, gain reduction is done with light. A light source (an electroluminescent panel, later an LED) shines on a photoresistive cell, which sits in the audio path on the LA-2A as on the CL 1B: the louder the signal, the brighter the source, the lower the cell's resistance, and the more the signal is attenuated. The control is thus isolated from the signal, hence very low distortion. Above all, the cell provides "for free" a behavior that other technologies have to build: it reacts with a slight delay (an attack of around ten milliseconds on the LA-2A, which varies with frequency), then releases in two stages, quickly for the first half of the way, then slowly, over several seconds, and all the more slowly as the compression was heavy or long. According to Dave Berners (Universal Audio), the flip side is a rather slow attack, linked attack and release times, and difficult stereo matching.

The reference of the family is the **Teletronix LA-2A** (1962 according to most sources), designed by James F. Lawrence II around his T4 cell. It works in feedback, with a soft knee and a nominal 4:1 ratio that actually varies with the source (Compress position), or an infinite ratio (Limit position). Lydkraft's **Tube-Tech CL 1B** (CL 1A in 1987, CL 1B since 1991) took up the principle, with full attack, release, ratio and threshold controls.

The nine presets in the family target three LA-2A emulations (**waves_la2a**, **native_vc2a**, **slate_fg2a**), two CL 1B emulations (**softube_cl1b** and, probably, **kiive_kc1**) and four plugins that reproduce lesser-known optos, often hybrids (**acustica_amber3**, **mixland_vacattack**, **acqua_opal**, **acqua_eminence**). What they have in common is not what you'd expect (see the preset map at the start of the chapter):

- after the main row, the most frequently modified tab is **A/R (cv)** (six presets out of nine), ahead of **WS** (five), then **GR (cv)**, **ENV (cv)** and **BIAS src** (three each);
- **only one preset out of nine uses the opto designer** from chapter 15: **acqua_eminence**, and even then in a very unusual way;
- the "optical" release is almost always obtained by other means: automatic release (**waves_la2a**, **native_vc2a**), the Mu envelope from chapter 11 (**acustica_amber3**, **acqua_opal**), the linear modes from chapter 12 (**mixland_vacattack**, **acqua_eminence**) or the attack and release shapes (**native_vc2a**, **softube_cl1b**, **acqua_opal**);
- **Env Offset**, the setting François says he uses most to imitate a compressor, appears in three presets (**slate_fg2a**, **mixland_vacattack**, **acqua_eminence**). François adds that on an LA-2A you can clearly feel an offset of this kind: that holds for **slate_fg2a**, which carries the strongest offset of all 80 presets (−0.55), but **waves_la2a** has none;
- five presets color the sound with the waveshaper (**native_vc2a**, **slate_fg2a**, **softube_cl1b**, **kiive_kc1**, **mixland_vacattack**), usually subtly, and three activate the bias (**softube_cl1b**, **kiive_kc1**, **mixland_vacattack**, the last one with no audible effect; see below);
- the three LA-2A imitations have an attack between 7.5 and 13.4 ms, around the 10 ms documented for the hardware;
- six presets keep the default linear curve, and three switch to decibels (**waves_la2a**, **kiive_kc1**, **mixland_vacattack**). On a linear curve, the displayed ratio is not the ratio in decibels: the Ratio of 17 in **native_vc2a** gives roughly 4:1 at high levels (chapter 10).

The curves figure at the start of the chapter shows the curve of the main row and the CURVE tab. It takes into account neither the **Headroom**, nor **Env Offset**, nor **Gr|Env Pow**, nor feedback: for the presets that use them, the actual static curves are described in their section.

waves_la2a: Waves CLA-2A

The target. The Waves CLA-2A (CLA Classic Compressors series, 2009) models Chris Lord-Alge's LA-2A, a Teletronix tube compressor with a T4 optical cell. The plugin has no time controls. Its manual describes a multistage, program-dependent release that can last several seconds after a loud signal, an attack and response speed that depend on frequency, a ratio of about 3:1 in Compress (100:1 in Limiter), and a HiFreq control that makes the reduction more sensitive above 1 kHz.

The settings.

- Main row: **Threshold** −29.36 dB; **Ratio** 4; **Attack** 8 ms; **Release** 120 ms; **Knee** 25 dB; **Cv Lin/Db** 1 (decibel curve); **Channels Unlink** 0.5.
- **MAIN** tab: **Headroom** −6 dB.
- **DET (in)** tab: **auto secs** 75 ms.
- **ENV (cv)** tab: **EvMu Lin/Db** 1 (envelope in decibels).
- **A/R (cv)** tab: **Release Auto** 0.5.
- **DEQ** tab: **Deq hgh herz** 7,000 Hz; **Deq hgh min** 0.5.



Figure 145 — The waves_la2a preset, DEQ tab. The modified settings and the tabs that contain them are shown in yellow: the main row (decibel curve, Knee of 25 dB, Channels Unlink at 0.5), and in the DEQ tab the high band at 7 kHz, which rises as the compressor reduces the gain.

Analysis. The heart of the preset is its curve. The LA-2A is known for its soft knee and for a ratio that is never quite fixed: François chose a decibel curve, a ratio of 4 and a **Knee** of 25 dB. The **Headroom** of −6 dB brings the signal into the compressor 6 dB lower, which amounts to raising the threshold to −23.4 dBFS: the knee then spreads from about −36 to −11 dBFS. Compression therefore starts very early and very gradually, and reaches 4:1 only on loud passages. This is the most direct translation, in this family, of a ratio that "varies with the source."

The times follow the hardware: 8 ms of attack, for the LA-2A's roughly 10 ms. The 120 ms **Release** is made program-dependent by **Release Auto** at 0.5: the time used is half the set time and half an automatic time derived from the crest factor, shorter on a percussive signal, longer on a sustained sound (chapter 12). With **auto secs** at 75 ms instead of 200 ms, this crest factor is measured over a shorter window. The envelope is computed in decibels, which makes it truer to the ear on heavy re-

ductions (chapter 11). On the other hand, nothing reproduces the long tail of several seconds: no opto designer, no **Env Offset**. Despite François's remark about the LA-2A's offset, it's in **slate_fg2a** that you'll find it, not here.

One last setting, found through analysis: the high band of the **DEQ**. Its coefficient is 0 at rest and rises toward 0.5 as the compressor reduces the gain. The highs above 7 kHz are therefore reduced a little less than the rest: at 6 dB of reduction, they lose only about 4 dB. The compression becomes frequency-dependent, like the hardware's, but through a different mechanism. This is not an imitation of HiFreq, which would do the opposite by compressing the highs more. Finally, **Channels Unlink** at 0.5 blends the common gain and each channel's own gain in equal parts.

> **♪ In the studio: waves_la2a**
>
> Try it first on a vocal or a bass, the two classic uses of the LA-2A. Set the **Threshold** until you see 3 to 6 dB of reduction on the loud passages: thanks to the very wide knee, the vocal settles in without its attacks being crushed. Then compare **Release Auto** at 0 and at 1 to hear what the program-dependent release brings, and listen to the brightness that holds up while the compressor is working.

native_vc2a: Native Instruments VC 2A

The target. The VC 2A is part of Native Instruments' Vintage Compressors collection (2011), produced with Softube. Its manual describes an early-1960s tube compressor with electroluminescent optical gain reduction: the LA-2A. It has no time controls; a COMPRESS (softer curve, lower ratio) or LIMIT (higher ratio) selector and the PEAK REDUCTION knob, which acts on both the ratio and the threshold, set the compression. It adds a LOW CUT on the detector and a DRY LEVEL. Recommended uses: guitars, bass and vocals.

The settings.

- Main row: **Threshold** −21.68 dB; **Ratio** 17; **Attack** 13.4 ms; **Release** 115 ms.
- **A/R (cv)** tab: **Attack Shape** 0.68; **Release Shape** 0.68; **Release Auto** 0.5.
- **WS** tab: **Fet odd** 1.27; **Fet even** 0.008.
- **DEQ 2** tab: **Deq low herz** 5 Hz; **Deq low max** −1; **Deq low zdf->svf** 0 (first-order filter).

Analysis. The **Ratio** of 17 suggests the LIMIT position. But the curve has stayed linear: the ratio applies to amplitudes, and, seen in decibels, the curve dips. It compresses very hard just above the threshold, then the slope rises again: between −16 and 0 dBFS, you get roughly 4:1 (steady-state calculation, curves figure at the start of the chapter). At high levels, this displayed ratio of 17 thus comes back to the same order of magnitude as the hardware's average ratio (about 3:1).

The release is the most telling setting. **Release Shape** at 0.68 makes the gain come back quickly at first, then lets it linger near 0 dB: that's the shape of the T4 cell's two-stage release, obtained without the opto designer. **Release Auto** at 0.5 makes the 115 ms time program-dependent, like the hardware's. The 13.4 ms attack stays close to the LA-2A's 10 ms, and **Attack Shape** at 0.68 makes it grab a large part of the reduction right away, then finish gently.

For color, **Fet odd** at 1.27 adds a soft, odd-harmonic saturation that follows the gain reduction (chapter 17). The plugin's harmonic character isn't documented: this is an adjustment found through analysis, consistent with the hardware's tube stage. **Fet even** at 0.008 adds only a trace of second harmonic, plus a little DC. The low band of the **DEQ 2**, at 5 Hz, as a first-order filter and with

a coefficient of −1, turns the EQ into an exact first-order high-pass at 5 Hz when the compressor isn't working: nothing audible, but DC and subsonic content disappear. The effect fades as the reduction increases, since the coefficient follows the envelope (chapter 18).

> **♪ In the studio: native_vc2a**
>
> On an electric guitar or a bass, this preset holds the level firmly without crushing the top of the dynamic range. Adjust the **Threshold** first; if the compression feels too much like a limiter just above the threshold, lower the **Ratio** to get closer to the spirit of COMPRESS. Listen to the ends of the notes: the release, fast and then lingering, lets the sustain breathe.

slate_fg2a: Slate Digital FG-2A

The target. The FG-2A is a module of Slate Digital's Virtual Mix Rack (introduced in late 2020, reviewed by Sound On Sound in 2021). It emulates the LA-2A and its T4 cell. It has no time controls; Slate modeled the "tapered" release, which slows down near 0 dB of reduction and is program-dependent. According to one review, the Compress/Limit switch gives about 4:1 and an infinite ratio. The plugin claims a "unique saturation characteristic" and adds an HF/Flat sidechain pre-emphasis, a Mix and a Stereo Link.

The settings.

- Main row: **Attack** 7.5 ms; **Knee** 12 (on a linear curve, a width of 0.1 in amplitude).
- **GR (cv)** tab: **Env Offset** −0.55.
- **WS** tab: **Fet even** −0.032.

Analysis. Only four settings, but one of them makes the whole preset: **Env Offset** at −0.55, the strongest offset of the 80 presets. This is where you find the offset that François says he clearly hears on the LA-2A. The rest of the curve has stayed at its defaults: **Ratio** 2, **Threshold** −20 dB, linear curve. On a linear curve, a ratio of 2 cannot reduce the gain by more than half (6 dB). The offset subtracts 0.55 from the envelope, which is 1 at rest: 0.45 remains, a permanent attenuation of 6.9 dB. After that, each decibel of reduction weighs much more heavily (chapter 15). At steady state, and relative to the resting level, the reduction is 1.3 dB for a detected level of −20 dBFS, 4.6 dB at −16 dBFS, 9.6 dB at −12 dBFS and 19 dB at −6 dBFS, and the target goes silent when the detected level reaches 0 dBFS. The output tops out for an input around −17 dBFS, then **decreases** as the input increases: this is overcompression, which a feed-forward topology makes possible. The curves figure at the start of the chapter, which ignores the offset, doesn't show this behavior.

This curve, which hardens with level to the point of making the output come back down, is an adjustment found through analysis. It suggests the Limit position. The linear **Knee** of 12 rounds off the transition between about −26 and −16.5 dB. The 7.5 ms attack is close to the LA-2A's 10 ms, but the **Release** has stayed at its default, 10 ms, much shorter than the roughly 60 ms the hardware takes for half of its release. With the default peak detection, such a short release lets the gain follow the individual cycles of the low end (chapter 12): a source of distortion, perhaps part of the saturation being sought. **Fet even** at −0.032 adds, in proportion to the gain reduction, a touch of second harmonic in inverted polarity and very slightly reinforces the fundamental.

> **♪ In the studio: slate_fg2a**
>
> Before anything else, raise the **Makeup** by about 7 dB so you can compare level-matched. Then set the **Threshold** on a vocal or a bass: with this offset, a few decibels of threshold are enough to go from gentle leveling to heavy squashing, where the loudest passages drop below the others. Listen to the low end: if the 10 ms **Release** makes it buzz, lengthen it.

> **To be confirmed by François**
>
> **slate_fg2a** carries the strongest **Env Offset** of the 80 presets (−0.55), while **waves_la2a** has none: is the LA-2A you talk about in the French video the FG-2A? With the default threshold, ratio and curve, this offset attenuates the signal by 6.9 dB at rest and makes the target go silent when the detected level reaches 0 dBFS: is that really the intended effect?

softube_cl1b: Softube Tube-Tech CL 1B

The target. Softube's Tube-Tech CL 1B (released in late 2007, now the "Mk II," which also includes the Mk I) emulates Lydkraft's CL 1B, designed by John G. Petersen: a mono optical compressor with an all-tube push-pull amplifier. Ratio from 2:1 to 10:1; FIXED mode (1 ms attack, 50 ms release), MANUAL mode (attack from 0.5 to 300 ms, release from 0.05 to 10 s) and FIX./MAN. mode, which chains a fast release followed by the manual release. The gain-reduction element has very low distortion. We don't know which generation of the plugin served as the model.

The settings.

- Main row: **Threshold** −13.2 dB; **Ratio** 6; **Attack** 5.8 ms; **Release** 138 ms.
- **CURVE** tab: **GR Min** −0.6 dB.
- **A/R (cv)** tab: **Attack Shape** 0.33; **Release Shape** 1.67; **Rel Ret From** 0.8.
- **WS** tab: **Fet odd** 2.
- **F (ws)** tab: **fet src mix** 0; **fet hgh herz** 6,000 Hz; **fet hgh mix** 0.75.
- **BIAS src** tab: **Bias input add** 0.90.
- **F (bs)** tab: **bias hgh herz** 3,000 Hz; **bias hgh mix** 0.75.

Analysis. The main row stays within the ranges of the CL 1B in MANUAL mode: a ratio of 6, a 5.8 ms attack, a 138 ms release. The curve is linear: seen in decibels, this ratio of 6 gives about 3:1 between −12 and 0 dBFS. **GR Min** at −0.6 dB imposes a permanent minimum reduction, even below the threshold: a small fixed attenuation which, unlike an offset, doesn't grow with compression. This is an adjustment found through analysis.

The shapes give the dynamic character. **Attack Shape** at 0.33 grabs almost all of the reduction right at the start of the attack, then finishes it slowly, which is reminiscent, though we can't say for sure, of a fast circuit combined with a slower one. **Release Shape** at 1.67 first holds back the gain's return, then lets it go decisively; **Rel Ret From** at 0.8 limits the point on the curve where each release resumes. So the preset isn't going for the "fast then slow" release of the FIX./MAN. mode.

The color comes from two saturators, both steered toward the top end. The waveshaper (**Fet odd** 2) saturates with odd harmonics, in proportion to the gain reduction, but its product keeps only what lies above 6 kHz, mixed at 0.75: you hear only the high-frequency grain of the saturation (chapter 16). The bias, activated by **Bias input add**, adds to the detection a height that follows the output lev-

el. With ordinary times like these, the gain doesn't follow the waveform: the result is mainly a compression whose detection is offset by the output, in the manner of a feedback compressor. Whatever saturation product remains goes through the **F (bs)** tab, which boosts its highs above 3 kHz (chapters 19 to 21). Comprendre's bias transposes a push-pull circuit, whose recombination cancels the even harmonics; the CL 1B happens to have a push-pull tube output amplifier. The connection is tempting, but the bias models a compression stage, not an output amplifier.

> **♪ In the studio: softube_cl1b**
>
> The CL 1B is first and foremost a vocal compressor: start there, then try a bass or an acoustic guitar. Set the **Threshold** first; the fast-grabbing attack holds the consonants, and the held-back release keeps the vocal from "pumping." To hear what the two saturations do, listen to them on their own with **Monitor Fet** and **Mon bias sat**: they sit in the air of the voice.

kiive_kc1: Kiive Audio Tube KC1

The target. Kiive Audio's Tube KC1 (May 2024) presents itself as "inspired by a legendary studio staple." One review describes it as an emulation of the Tube-Tech CL 1B, but that source could not be verified: the name probably refers to an imitation of the CL 1B. The developer promises "opto-style" compression, tube warmth, and use mainly on vocals, but also on drums.

The settings.

- Main row: **Threshold** −13.52 dB; **Ratio** 1.87; **Attack** 1.5 ms; **Release** 236 ms; **Knee** 18.09 dB; **Cv Lin/Db** 1 (decibel curve); **Channels Unlink** 1.
- **WS** tab: **Fet odd** 0.91; **Fet drive** 2 dB.
- **BIAS src** tab: **Bias input mul** 4.25.

Analysis. The decisive setting is **Bias input mul**. With **Bias mul %** at 1 (default) and the source set to "out" (default), the detection signal is multiplied by the bias value, which follows the output, and by 4.25: the compressor measures the product of the signal and its own output level (chapter 20). The more it compresses, the less it measures: the detection regulates itself, in the manner of a feedback compressor. The LA-2A, which inspired the CL 1A, is a feedback compressor; the topology of the CL 1B isn't documented in our sources. François created this multiplicative mode to imitate the SSL compressor; here he uses it for an opto.

The practical consequence: the **Threshold** and the **Knee** apply to this product, not to the signal alone. Below the threshold, when the output follows the input, the detection varies in decibels twice as fast as the signal. The 18 dB knee therefore corresponds to a narrower input range, and compression starts around −13 dBFS. With a ratio of 1.87, it stays gentle. The 1.5 ms attack is close to that of the CL 1B's FIXED mode (1 ms), and the 236 ms release is within the range of its MANUAL mode. **Channels Unlink** at 1 gives one compressor per channel, like two mono units.

The waveshaper is very discreet: **Fet odd** at 0.91, slightly below 1, gives a barely expansive curve, which slightly attenuates the low levels of each half-cycle and preserves the peaks, in proportion to the gain reduction; **Fet drive** at 2 dB restricts this shaping to levels below about 0.79 (chapter 17). The advertised "warmth" is thus only hinted at.

> **♪ In the studio: kiive_kc1**
>
> Try it on a lead vocal or a drum bus, for supple leveling that follows the singing. The **Threshold** is the first thing to set, but it acts about half as fast as usual: 2 dB of threshold moves the onset of compression by roughly 1 dB. Listen to how the compression seems to adapt to what it produces, without ever clamping down on the attacks.

acustica_amber3: Acustica Audio Amber 3

The target. Acustica Audio's Amber 3 (February 2019) adds a compressor and a preamp to the Amber 2 EQ. Its interface follows that of the Avalon Designs AD2044, and Acustica presents it as the reproduction of an American class A optical compressor. Like all of the developer's plugins, it relies on dynamic convolution. Ratios and times are not documented.

The settings.

- Main row: **Threshold** −27.68 dB; **Ratio** 3.41; **Attack** 17.5 ms; **Release** 48 ms.
- **ENV (cv)** tab: **Env->Mu** 1; **Mu Pow Att** 1.16; **Mu Max Att** 9.58; **Mu Min Rel** 1.73; **Mu Max Rel** 0.754; **Mu Pow Rel** 0.648.

Analysis. No opto designer, no saturation: this opto is imitated solely by the **Mu envelope**, the "varimu" tool from chapter 11, whose speeds slide with the detected level. For the attack, the time is 17.5 ms on a quiet signal and up to 9.58 times longer (about 170 ms) near 0 dBFS; **Mu Pow Att** at 1.16 reserves this slowdown for the loudest levels. The attack thus slows down on loud passages and lets their transients through. For the release, the time is 1.73 times the Release (about 83 ms) on a quiet signal and 0.754 times (about 36 ms) near 0 dBFS; **Mu Pow Rel** at 0.648 spreads this variation toward the medium levels. As long as the music is loud, the gain comes back fast; when it calms down, it comes back more slowly.

This fast-then-slow release is reminiscent of an opto, but it depends on the detected level and not on the reduction, as the opto designer's would. The attack that slows down at high levels even runs opposite to the designer, which speeds up as the reduction grows: this is an adjustment found through analysis. The linear curve, with a ratio of 3.41, compresses firmly between the threshold (−27.7 dB) and about −16 dBFS, then hardly reduces any further: the reduction cannot exceed 10.7 dB.

> **♪ In the studio: acustica_amber3**
>
> On a vocal or a solo instrument, this preset levels out the medium-level phrases and lets the outbursts live. Set the **Threshold** to place the compression zone where most of the playing happens. Watch the brown line in the history graph: it drops on loud passages, when the release speeds up.

mixland_vacattack: Mixland Vac Attack

The target. Vac Attack, by Mixland (mixer Jesse Ray Ernster and programmer Eddie Lucciola), distributed by Plugin Alliance since 2023, emulates the Inward Connections Vac Rac TSL-4, a tube optical limiter discontinued in 2020, whose detector is preceded by a 100 Hz high-pass filter. The plugin models the light source and the photoresistor. It has no time controls: its times are automatic and the release is program-dependent. A Knee control acts on how the ratio is applied and on the times; the ratio runs from 3.52:1 upward. A tube saturation (THD control), two EQs and a Mix round it out.

The settings.

- Main row: **Attack** 6.6 ms; **Release** 400 ms; **Cv Lin/Db** 1 (decibel curve).
- **A/R (cv)** tab: **Attack Mode** 2 (linear); **Release Mode** 2 (linear).
- **GR (cv)** tab: **Env Offset** −0.125.
- **WS** tab: **Fet odd** 3; **Fet drive** 3.04 dB; **Fet Thresh** 0.8; **Fet sin x** 2.
- **F (ws)** tab: **fet amp** 0.5.
- **BIAS src** tab: **Bias +inverse** 0.5.
- **BIAS dst** tab: **Bias on limit** 1; **Bias x gr** 3.03.
- **F (bs)** tab: **bias amp** 0; **bias src mix** 0; **bias hgh herz** 257 Hz; **bias hgh mix** 0.97; **bias hgh zdf->svf** 0.

Analysis. The curve is in decibels, with the default ratio (2) and threshold (−20 dB). **Env Offset** at −0.125 attenuates the signal by 1.2 dB at rest, then deepens the reduction all the more as the compression gets heavier: at steady state, the effective ratio goes from about 2.4:1 just above the threshold to more than 4:1 near 0 dBFS. A ratio that rises with level: one review of the plugin speaks precisely of a ratio of 3.52:1 and up.

With no adjustable times on the target, François chose the **linear modes** for the attack and release: the gain changes at a constant rate (chapter 12). A linear release therefore takes longer the farther it has to travel: about 0.3 s to recover from 6 dB of reduction, nearly 0.5 s for 12 dB. The recovery time depends on the amount of compression, which is a form of program-dependent release.

The waveshaper provides the tube saturation. **Fet odd** at 3 strongly boosts the low levels of each half-cycle and rounds off the peaks; **Fet Thresh** at 0.8 clips them, with the 3 dB **Fet drive** bringing the effective threshold down to about 0.56; **fet amp** at 0.5 halves the product. The whole thing follows the gain reduction: the harder the compressor works, the thicker the sound gets. **Fet sin x** at 2 has no effect, because **Fet sin *** has stayed at 0. Finally, the bias sets up a push-pull saturation that would grow with the gain reduction (height **Bias x gr**, limiter at 0 dBFS, inverse path at half), with its product stripped of its lows below about 257 Hz. But **bias amp** is at 0: the bias product is canceled before it even reaches its filters, and, with no input height, the bias no longer has any audible effect (chapter 21).

> **♪ In the studio: mixland_vacattack**
>
> Try it on a drum bus, a bass or a vocal, the uses the plugin recommends. Set the **Threshold** first: with the offset, it decides whether the compression stays supple or hardens. Listen to the density that builds up with the compression, and isolate it with **Monitor Fet**; if it gets too thick, lower **fet amp**.

> **To be confirmed by François**
>
> In **mixland_vacattack**, **bias amp** is 0: the bias product is canceled before its filters and, with no input height, the bias has no audible effect. The settings in the BIAS src, BIAS dst and F (bs) tabs therefore serve no purpose, other than doubling the CPU load (inverse path active). Is this intentional?

acqua_opal: Acustica Audio Opal Comp

The target. Opal Comp, by Acustica Audio, is presented as the digital "bigger sibling" of the Sonicomp 2 by John Oram (Oram Sonics). Its design is hybrid: the optical stage provides a soft, rounded knee at the threshold, and the solid-state stage handles the peaks. The plugin offers a feed-forward mode, a VU meter and dual-mono operation, and aims at final mixing and transparent peak control. Ratios and times are not documented.

The settings.

- Main row: **Ratio** 42 (infinite); **Attack** 65.8 ms; **Release** 453 ms.
- **INPUT** tab: **FF->FB** 1 (feedback).
- **DET (in)** tab: **rms secs** 3.2 ms; **rms db** 0 dB; **peak db** −24 dB; **Det Mix** avg (average).
- **ENV (cv)** tab: **Env->Mu** 1; **Mu Min Rel** 0.57.
- **A/R (cv)** tab: **Attack Shape** 0.58; **Att Ret Spd** 0.008; **Release Shape** 2.
- **OUTPUT** tab: **deq** (the second one) 0.

Analysis. The preset is a **pure feedback** "limiter." The plugin offers a feed-forward mode; the preset, for its part, works in feedback. But a feedback compressor can't reach an infinite ratio: its detector measures an output that has already been reduced (chapter 14). With this ratio of 42, the linear curve and the default threshold (−20 dB), you get at steady state an effective ratio of about 2:1 above the threshold: moderate compression, with the smoothness François attributes to feedback. The curves figure at the start of the chapter, which ignores feedback, shows a ceiling that doesn't exist here.

The detection is modified: the peak is excluded (−24 dB) and only the RMS stream remains, with a very short time constant of 3.2 ms. The **Knee** has stayed at 0: the rounding at the threshold comes from the feedback and from this averaged measurement, not from the curve. The times are slow, 66 ms of attack and 453 ms of release, and the Mu envelope (**Mu Min Rel** at 0.57) shortens the release when the measured level, here the output level, is low. **Attack Shape** at 0.58 grabs the beginning of the reduction quickly, then slows down; with **Att Ret Spd** almost at zero, the position on the attack curve hardly moves back during releases, so after the first attack, the following ones resume farther along the curve, where it is slower. **Release Shape** at 2 holds back the gain's return before letting it go. The second **deq** in the OUTPUT tab, at 0, has no effect: the DEQ 2 has no active band.

> **♪ In the studio: acqua_opal**
>
> Put it on a mix bus or an instrument group, where you're after transparent control rather than an effect. The **Threshold**, still at −20 dB, is the first setting to adjust: lower it until you get a few decibels of reduction. Listen to how the peaks are rounded off without the groove being affected.

acqua_eminence: Acustica Audio Eminence

The target. Eminence, by Acustica Audio, is a free channel strip released in December 2021. Its "Frankenstein" compressor brings together two solid-state dynamics processors from an unnamed American manufacturer: an optical compressor (591) and a FET limiter (612). The EQ is a 1960s British three-band design. The developer speaks of "fat, crunchy" compression; ratios and times are not documented.

The settings.

- Main row: **Threshold** −24 dB; **Ratio** 5; **Attack** 5 ms; **Release** 1.2 s.
- **A/R (cv)** tab: **Attack Mode** 2 (linear); **Release Mode** 2 (linear).
- **GR (cv)** tab: **Gr|Env Pow** −2.5; **Env Offset** −0.1; **Opto** −0.2; **Opto Env** 1; **Opto A/R mul** 421; **Opto severity** 2.66; **Opto precision** 0.



Figure 146 — The acqua_eminence preset, GR (cv) tab: the only preset in the family that activates the opto designer, with a negative Opto built into the envelope (Opto Env at 1) and Opto A/R mul at its maximum, 421. Gr|Env Pow and Env Offset also shape the envelope.

Analysis. This is the only preset in the family that uses the opto designer, and it does so in an unexpected way. **Opto Env** at 1 builds the opto into the envelope: there is no saturation product, so the F (gr) tab has nothing to process. **Opto A/R mul** is at the maximum of its range, 421: the opto's times are 421 times those of the main follower, which means about 2 s to go down and several minutes to come back up. **Opto severity** at 2.66 lets the reduction speed up the opto only moderately, and **Opto precision** at 0 removes the rounding. The optical follower thus becomes a **memory** of the reductions of the last few minutes. This brings to mind the documented memory effect of the T4 cell, whose time constant, according to Universal Audio, is measured in minutes.

But **Opto** is negative: the gap between the opto and the envelope is reversed, at 20%. As long as the memory shows more reduction than the envelope, that is, after a loud passage, the final gain applies a little less. In practice, the quiet passages that follow a loud one are slightly lifted, on the order of 1 dB in a simple simulation, and this lasts for several minutes; at the start of a song, before the memory has built up, the first loud passages are, on the contrary, compressed a little more. This is the opposite of the T4's memory, which prolongs the reduction after heavy compression: an adjustment found through analysis, on a unit (the 591) whose behavior isn't documented.

The envelope is shaped as well. **Gr|Env Pow** at −2.5 strongly softens small reductions and large ones much less: the 3 dB requested by the curve at −20 dBFS become 0.5 dB, and the 12 dB requested at 0 dBFS become 6.8 (after **Env Offset**, which attenuates by 0.9 dB at rest and deepens heavy reductions). Compression therefore starts very gradually above the threshold. The **linear modes** make the gain change at a constant rate: the 1.2 s release takes longer the farther it has to travel. No saturation, on the other hand: the "crunchy" side of the FET limiter is not imitated.

> **♪ In the studio: acqua_eminence**
>
> This preset is judged over time: play a whole song, on a bus or a vocal, and compare the beginning with the end. First set the **Threshold** to get a few decibels of reduction on the choruses; the opto's memory then discreetly lifts the verses that follow them. To hear what this memory does, compare with **Opto** at 0.

Transistor speed: the FETs

In 1966, Bill Putnam replaced the tubes in his Universal Audio 175 and 176 compressors with field-effect transistors (FETs): the **1176** came out in 1967. The FET acts as a voltage-controlled resistor at the bottom of a divider: the higher the control voltage, the lower its resistance, and the more the gain drops. The component is inherently nonlinear, and the whole art of the circuit lies in keeping it within its linear region; the "low noise" (LN) circuit of revision C, in 1970, helps with that. The 1176 is a feedback compressor: its sidechain measures the signal after gain reduction. It has no threshold: you set the amount of compression with the input, on four ratios (4, 8, 12 and 20:1) that also raise the threshold. Its times made its reputation: an attack of 20 to 800 µs and a release of 50 ms to 1.1 s, partly program-dependent. Pressing all four ratio buttons together ("all buttons" mode, or "British mode") shifts the bias points of the whole circuit and sends the distortion soaring. Next to the Fairchild, soft and warm, the 1176 brightens the tone and adds bite; on a bass, its slight FET distortion helps it cut through. Dave Berners, of Universal Audio, sums up the family this way: behavior close to that of the vari-mu, but with "solid-state" saturation and a tighter knee.

Seven of the nine presets in this family imitate plugins derived from the 1176: the Waves CLA-76, the Arturia Comp FET-76, the Softube FET Compressor, the DMGAudio TrackComp, the Analog Obsession FETish, the Acustica Audio Nickel and the Pulsar 1178, an emulation of the stereo version of the 1176. The other two depart from it: Chandler Limited's Germanium Compressor, a FET with a very different character, and Korneff Audio's Pawn Shop Comp 2.0, a tube-and-FET hybrid.

In Comprendre, these nine presets share several traits (see the preset map at the start of the chapter):

- **The curve.** All of them adjust the main row. Four switch the curve to decibels (**waves_76**, **dmg_trackcomp_76a**, **ao_fetish**, **acustica_nickel**), and three of those add a knee of 3.5 to 9 dB, a tight knee like the one Berners describes. Four presets use a ratio of 4, the 1176's first button.
- **The waveshaper.** Six presets out of nine set up the **WS** tab, and none touches **Fet gr** or **Fet env**: all keep the default dynamic saturation, which grows with the gain reduction (chapter 17). Yet the FET designer isn't mandatory: **arturia_fet**, **softube_fet** and **acustica_nickel** leave it alone, and the one in **korneff_comp2** isn't heard. The three presets that then filter the product in the **F (ws)** tab make the same choice: **fet src mix** at 0 and a single band, which means they keep only one region of the saturation's spectrum (the highs for **waves_76**, the extreme lows for **pulsar_1178**, the lows for **softube_germanium**).
- **The offset.** Three presets (**softube_fet**, **pulsar_1178**, **softube_germanium**) lower the envelope with **Env Offset**. This offset acts on everything that depends on the envelope: the waveshaper's saturation is no longer zero at rest, and the dynamic EQs are no longer at their "max" position when the compressor isn't working.
- **The dynamic EQs.** Four presets use them (**arturia_fet**, **softube_fet**, **ao_fetish**, **softube_germanium**), almost always in the same way: a correction set by **max**, and therefore present at rest, which fades away when the compressor works. This is the opposite of the use François presents as the most natural (min at 1, max at 0, chapter 18).

Three surprises stand out.

- **The attack.** No preset in the family shows an attack on the order of a hundred microseconds: the 1176 emulations range from 2.6 to 10 ms, and even 44.6 ms for **dmg_trackcomp_76a**.

Manufacturers don't all define their times the same way (chapter 12), but above all, the speed of the FET is often obtained somewhere other than the **Attack** knob: through an opto designer set negative, which speeds up the start of attacks (**waves_76**), through the shape and the return of the attack curve (**dmg_trackcomp_76a**), or through an **Attack Shape** slightly below 1 (**ao_fetish**, **acustica_nickel**). The attacks set below a millisecond are actually found in the Distressor family (later in this chapter).

- **Feedback.** The 1176 is documented as a feedback compressor, but only one preset, **waves_76**, uses **FF->FB**, and only halfway.
- **The opto designer.** The only preset in the family that engages it, **waves_76**, sets it negative, and the effect it gets has nothing to do with an optical cell: it makes the start of attacks snappier.

> **A word from the author**
>
> François designed the waveshaper as a saturation designer meant to simulate FET compressors. By default, the gain reduction drives the saturation, as on real hardware: the more it compresses, the more it saturates. The presets in this family all keep this behavior.

waves_76: Waves CLA-76

The target. The Waves CLA-76 (2009, CLA Classic Compressors series) models two of engineer Chris Lord-Alge's 1176s: a revision B "Bluey" and a revision D-LN "Blacky." We don't know which one François imitated. Waves documents ratios of 4, 8, 12, 20:1 and All, a threshold that rises with the ratio (−18 dBFS at 4:1, −12 dBFS at 20:1), an attack graduated from 1 (1 ms, the slowest) to 7 (less than 50 µs), a release from 1 (1 s) to 7 (50 ms), as well as even and odd harmonic distortion and the noise of the original preamp.

The settings.

- Main row: **Threshold** −18.8 dB; **Ratio** 4; **Knee** 9 dB; **Cv Lin/Db** 1 (decibel curve); **Attack** 2.6 ms; **Release** 543 ms.
- **INPUT** tab: **FF->FB** 0.5.
- **A/R (cv)** tab: **Release Mode** 2 (linear).
- **GR (cv)** tab: **Opto** −1; **Opto A/R mul** 2.46; **Opto severity** −2.97; **Opto precision** 0.
- **WS** tab: **Fet Thresh** 0.74; **Fet Knee** 240; **Fet *** 0; **Fet sin *** 1; **Fet sin x** 8; **Fet Precision** 2.68.
- **F (ws)** tab: **fet src mix** 0; **fet hgh herz** 5,566 Hz; **fet hgh zdf->svf** 0 (one-pole filter); **fet hgh mix** 1.18.

Analysis. The ratio of 4 and the threshold of −18.8 dB land almost exactly on the CLA-76's 4:1 setting, for which Waves puts the threshold at −18 dBFS. The curve is in decibels, with a 9 dB knee (from −23.3 to −14.3 dB): neither Waves nor the history of the 1176 gives the shape of the knee, so this is an adjustment found through analysis. **FF->FB** at 0.5 makes the detector measure an equal mix of the input signal and the output of the previous sample (chapter 14): this is the only preset in the family that makes room for feedback, the documented topology of the 1176.

The 2.6 ms attack is slower than the CLA-76's slowest. The opto designer, set in an unexpected way, makes up for a small part of that. With **Opto** at −1, the gap between the optical gain and the envelope is reversed (chapter 15); with an **Opto severity** of −2.97 and no rounding (**Opto precision** at 0), this gap exists only for the very smallest reductions, since from 3 dB on, the opto's time is divided by about sixty. The result: at the very beginning of each attack, where the opto would lag behind,

the gain drops a little faster than the envelope. A simulation of the formulas in the code, on a 12 dB reduction, gives the first decibel in 0.2 ms instead of 0.4 ms, then an almost unchanged attack. The 543 ms release, within the CLA-76's range, is **linear**: the gain comes back at a constant rate and stops dead at its target, with no tail. The recovery time therefore grows with the depth of the reduction: about 0.25 s after 3 dB, 0.64 s after 12 dB. This goes in the same direction as the original 1176, documented as releasing faster after a transient than after heavy, sustained compression.

The waveshaper is the most original part of the preset. The clipper has an outsized knee (**Fet Knee** 240, a width of 2): the rounding covers the entire range, and the curve becomes a gentle slope that lowers the level by 3 to 4 dB and rounds off the peaks (0.6 out for an input of 1). Since half of the knee lies beyond the threshold, this is the case flagged in chapter 17, where the curve makes a small jump at the zero crossing. **Fet *** at 0 and **Fet Precision** at 2.68 then activate the spreader mode: with **Fet sin x** at 8, the sine wave becomes a noise whose amplitude follows the level to the power of 2.68. Before being dosed by the gain reduction, this noise stays 10 dB below a −6 dBFS peak, 20 dB below a −12 dBFS peak and more than 30 dB below a −20 dBFS peak. Finally, the F (ws) tab keeps only the part of the product above about 5.6 kHz (6 dB per octave high-pass), slightly boosted. When the compressor is working, you therefore hear a noisy grain on the peaks, the high harmonics of the rounding and a slight softening of the highs (since the curve passes below the diagonal, the product contains an inverted copy of the signal). Waves documents even and odd distortion and preamp noise; the preset, for its part, creates the grain François was after with the spreader, a noise that appears only on loud signals (chapter 17).



Figure 147 — The waves_76 preset, WS tab: a clipper with the maximum knee and the spreader mode (Fet * at 0, Fet Precision 2.68), which confines the noise of the sine (Fet sin x at 8) to the peaks. The INPUT, A/R (cv), GR (cv) and F (ws) tabs, also modified, are in yellow.

> **♪ In the studio: waves_76**
>
> Try it on a bass, the instrument the 1176 is most often mentioned for, or on drums blended in parallel with **Dry/Wet**, in the manner of the parallel drum compression done with the 1176. Set the **Threshold** first, then listen to the grain on its own with **Monitor Fet** (WS tab) during a passage that compresses: it appears only on the peaks. If the linear release feels too mechanical on a vocal, shorten the **Release** before touching the mode.

arturia_fet: Arturia Comp FET-76

The target. Arturia's Comp FET-76 (2019, in the "3 Compressors You'll Actually Use" bundle) emulates the 1176; the targeted revision isn't specified. As on the original, it has no adjustable threshold: you set the amount of compression with the input. It offers fixed ratios of 4, 8, 12 and 20:1, a button that reproduces the "all buttons" trick, automatic level compensation, times as fast as 20 µs, an equalized sidechain and a compression pre-delay ("Time Warp").

The settings.

- Main row: **Threshold** −32 dB; **Ratio** 4; **Attack** 4.2 ms; **Release** 323 ms.
- **DEQ 2** tab: **Deq low herz** 65 Hz; **Deq low max** −0.75; **Deq low min** −0.2; **Deq hgh herz** 9,000 Hz; **Deq hgh max** −0.5; **Deq hgh min** −0.1; **Deq hgh zdf->svf** 0 (one-pole filter).

Analysis. The ratio of 4 corresponds to the 1176's first button. The −32 dB threshold has no equivalent on the plugin, which has no threshold: it probably reflects the input level of the excerpt analyzed, and it's the first setting to redo on your track, just as you would turn the 1176's input. The preset keeps the default linear curve: its ratio of 4 is therefore not a 4:1 in the usual sense, in decibels, and it compresses high levels noticeably less (chapter 10). The very low threshold partly makes up for this (curves figure at the start of the chapter).

Since the waveshaper isn't engaged, the DEQ 2 simply acts as a dynamic EQ on the target. At rest, it removes the extreme lows (about −12 dB below 65 Hz) and softens the highs (down to −6 dB above 9 kHz, with a gentle 6 dB per octave slope). When the compressor reduces the gain, these corrections fade: at 6 dB of reduction, they're down to about −5.6 dB and −3 dB, and at full reduction to −1.9 dB and −0.9 dB. Nothing in Arturia's documentation describes this behavior: it's a tonal correction found through analysis, which tightens the sound a little at both ends of the spectrum at rest and opens it up when the compression sets in. The 4.2 ms attack remains far from the advertised 20 µs, as in the rest of the family.

> **♪ In the studio: arturia_fet**
>
> On a vocal or an electric guitar, turn the **Threshold** as you would turn the input of an 1176, until you get the amount of compression you want. Then listen to the top and bottom of the spectrum: they tighten up in quiet passages and open up when the compression kicks in. If you prefer a constant tone, bring **Deq low max** and **Deq hgh max** to the value of their **min**.

softube_fet: Softube FET Compressor

The target. Softube's FET Compressor (2009) is "based on the most famous hardware FET compressor," without naming it; Sound On Sound identifies it as the 1176. Its Input knob serves as both gain and threshold, its ratio is continuous (marks at 1, 4, 8, 12, 20:1 and ALL), its attack ranges from about 20 to 800 µs and its release from 50 ms to 1.1 s, both program-dependent. Softube describes input-stage distortion that colors the sound even at 1:1, and adds a lookahead, a parallel blend ("Parallel Inject") and filters on the detector.

The settings.

- Main row: **Attack** 3.1 ms; **Release** 410 ms (the **Threshold** and **Ratio** stay at −20 dB and 2, on a linear curve).

- **GR (cv)** tab: **Env Offset** −0.2.
- **DEQ** tab: **Deq low herz** 100 Hz; **Deq low max** −0.5; **Deq hgh herz** 5,000 Hz; **Deq hgh max** 0.5.

Analysis. This is the most restrained preset in the family: on the main row, only the times change. The 410 ms release is within the documented range; the 3.1 ms attack is far from it. The depth of the compression owes a lot to the offset, François's favorite imitation tool (chapter 15). At −0.2, the signal is lowered by 1.9 dB at rest, and the reduction deepens during compression: at steady state, on a sine at 0 dBFS, a simulation gives about 9 dB of reduction, versus 5 dB without the offset. The curves figure at the start of the chapter doesn't show this effect, since it represents only the transfer curve.

The DEQ dynamic EQ, placed on the target, removes lows below 100 Hz and adds highs above 5 kHz when the compressor isn't working, then fades away with compression. Because of the offset, the envelope is only 0.8 at rest: the two corrections there amount to about −4.4 dB and +2.9 dB. You can see this as the coloration Softube attributes to the input stage, present even without compression; its fading with compression, on the other hand, is an adjustment found through analysis. Note that the plugin's filters act on its detector, whereas the preset's act on the sound.

> **♪ In the studio: softube_fet**
>
> Softube recommends it on a drum bus: try it there, then on a bass. Set the **Threshold** first, and compare level-matched by adding about 2 dB of **Makeup**, since the offset lowers the level even at rest. For deeper or shallower compression at the same threshold, play with **Env Offset** (GR (cv) tab) between 0 and −0.3.

dmg_trackcomp_76a: DMGAudio TrackComp (76A model)

The target. TrackComp, by DMGAudio, brings together several compressor models; the 76A is revision A of the 1176 (1967, serial numbers 101 to 125), whose post-FET preamp combines a FET and a bipolar transistor, before the arrival of the LN circuit. DMGAudio claims a real-time circuit simulation (SPICE), with zero latency, and stepped controls as on the hardware.

The settings.

- Main row: **Threshold** −14 dB; **Ratio** 40; **Cv Lin/Db** 1; **Attack** 44.6 ms; **Release** 100 ms.
- **DET (in)** tab: **Det Mix** avg.
- **ENV (cv)** tab: **EvMu Lin/Db** 1 (envelope in decibels); **EvMu Br/Cp** 1 (decoupled envelope); **Mu Min Rel** 1.32; **Mu Max Rel** 0.
- **A/R (cv)** tab: **Attack Auto** 0.5; **Attack Altitude** 0.004; **Attack Shape** 3.08; **Att Ret From** 1.19; **Release Altitude** 0.632; **Release Shape** 1.8.
- **GR (cv)** tab: **Gr|Env Pow** 1.
- **WS** tab: **Fet drive** +0.05 dB; **Fet even** 0.056; **Fet odd** 1.016; **Fet *** 0; **Fet sin *** 0.048; **Fet sin x** 1.88; **Fet Precision** 256.
- **DET (bs)** tab: **Det Mix** avg.

Analysis. This is the preset in the family that works hardest on the attack, and it nicely illustrates the point made in the introduction. On paper, the attack is slow: 44.6 ms, with an altitude of 0.004 (the set time becomes the time for 99.6% of the way) and an S-shape (**Attack Shape** 3.08). Taken on its own, this curve covers only 5% of the way in 10 ms, half by around 23 ms and 93% at 35 ms. But the envelope is decoupled, and in this mode, with **Att Ret Spd** at 1, the position on the attack curve

never returns to the start: it stays at **Att Ret From**, that is, 1.19 times the attack time (chapter 12). After the very first attack following loading, each attack therefore starts in the steep part of the S-curve. A simulation of the code, on a 12 dB reduction, then gives the first decibel in 0.15 ms, 6 dB in 1.2 ms and 11 dB in 4.2 ms, versus 11.6 ms, 22.7 ms and 34.4 ms for the first attack. The preset thus recovers the very fast attack of the 1176 with an **Attack** setting that gives no hint of it. **Attack Auto** at 0.5 shortens the time further on percussive passages, and lengthens it a little on sustained sounds.

The release is reshaped too. The decoupled envelope, in decibels, makes the gain wait a little after a long attack (one of the “two schools” of chapter 11). The altitude of 0.632 makes the **Release** time the time for just 36.8% of the way, and the shape of 1.8 holds back the return before letting it go: after a 12 dB reduction, the simulation still gives 6 dB of reduction after 32 ms, 3 dB after 61 ms and 1 dB after 103 ms. The ratio of 40, in decibels, far exceeds the 1176’s 20:1: at this setting, the preset is almost limiting. The waveshaper adds a discreet coloration, dosed by the gain reduction: a bit of second harmonic (**Fet even** 0.056) and a hint of odd (**Fet odd** 1.016). The spreader mode (**Fet *** at 0, **Fet Precision** 256) weights its sine by the level to the power of 256: it appears only in the immediate vicinity of full internal level, and remains practically inaudible.

Five settings are modified without changing anything in the sound (the DET (in), GR (cv) and DET (bs) tabs light up in yellow only because of them): **Det Mix** on avg in DET (in), since the peak is the only stream included and the average of a single stream is that stream itself; **Mu Min Rel** and **Mu Max Rel**, since the Mu envelope isn’t computed as long as **Env->Mu** is 0; **Gr|Env Pow** at 1, a neutral value just like 0; and **Det Mix** in DET (bs), since the bias isn’t active.



Figure 148 — The dmg_trackcomp_76a preset, A/R (cv) tab: a 44.6 ms S-shaped attack (Attack Shape 3.08, Attack Altitude 0.004) whose position is kept from one attack to the next (Att Ret From 1.19), half automatic, and a release reshaped by its altitude and shape.

> **♪ In the studio: dmg_trackcomp_76a**
>
> Try it on a snare or a rock vocal, where you expect an 1176 to grab the peaks instantly. Set the **Threshold** first: with a ratio of 40, the preset behaves almost like a limiter. For a gentler take, closer to the 1176’s 4:1 or 8:1 buttons, bring the **Ratio** down to somewhere between 4 and 8, and listen to whether the held-back release stays natural.

To be confirmed by François

The dmg_trackcomp_76a preset (decoupled envelope, Attack Shape 3.08, Att Ret From 1.19, Att Ret Spd at 1) owes its fast attack to the behavior flagged in chapter 12: in decoupled mode, the position on the attack curve never returns to the start. If this behavior is fixed, each attack will start again from the beginning of the S-curve (first decibel in 11.6 ms instead of 0.15 ms): will this preset need to be revised?

ao_fetish: Analog Obsession FETish

The target. FETish, by Analog Obsession, is a free compressor "inspired by classic 1176 behavior," not licensed by Universal Audio. It is presented as a compressor with fast, aggressive gain reduction, for snare, kick, room mics, parallel drum bus and vocals. The details of its controls could not be consulted.

The settings.

- Main row: **Threshold** −12 dB; **Ratio** 4.05; **Knee** 3.51 dB; **Cv Lin/Db** 1; **Release** 200 ms (the **Attack** stays at 10 ms).
- **ENV (cv)** tab: **EvMu Lin/Db** 1.
- **A/R (cv)** tab: **Attack Shape** 0.936; **Release Mode** 1.41.
- **DEQ** tab: **Deq low herz** 14 Hz; **Deq low min** and **Deq low max** −0.818; **Deq low zdf->svf** 0.776; **Deq mid herz** 30 Hz; **Deq mid min** and **Deq mid max** −0.055; **Deq hgh herz** 6,400 Hz; **Deq hgh min** and **Deq hgh max** −0.087; **Deq hgh reso** 0.319.
- **WS** tab: **Fet even** 0.008.
- **DEQ 2** tab: **Deq low herz** 10.5 Hz; **Deq low max** −0.631; **Deq low reso** 0.229; **Deq low zdf->svf** 0.804; **Deq mid herz** 30 Hz; **Deq hgh herz** 6,400 Hz.

Analysis. The curve suggests an 1176 set to 4:1: a ratio of 4.05, in decibels, with a tight 3.5 dB knee. Surprisingly, for a plugin presented as fast and aggressive, the **Attack** stays at its default value, 10 ms. Only an **Attack Shape** of 0.936, barely below 1, makes the attack start a little faster. The decibel envelope and the **Release Mode** at 1.41 blend the logarithmic release with the linear release, which, in decibels, comes back at a constant rate: 9.48 dB per **Release** time, or about 47 dB per second here. It's the same recipe as **acustica_nickel**, right down to the value of the **Attack Shape**.

The two dynamic EQs work at the edges of the audible band. The DEQ is static (**min** equal to **max**): it removes subsonics below 14 Hz (about −15 dB), barely dips 30 Hz (−0.3 dB) and tilts the highs very slightly above 6.4 kHz (−0.8 dB at most, with a low resonance that softens the slope). The DEQ 2 attenuates subsonics below 10 Hz at rest and gives them back with compression; its mid and high bands have a frequency but zero coefficients, so they have no effect. The waveshaper adds only a trace of second harmonic (**Fet even** 0.008), dosed by the gain reduction. These adjustments, found through analysis, remain very subtle: you can see them as the limited bandwidth of an analog circuit, which the plugin perhaps imitates.

> **♪ In the studio: ao_fetish**
>
> Analog Obsession intends it for drums: try it on a snare, a kick or room mics, blended in parallel with **Dry/Wet**. Set the **Threshold** first. If you're expecting the advertised aggressiveness, shorten the **Attack**, still at 10 ms, to somewhere between 1 and 3 ms, and listen to the snare trade its crack for density.

acustica_nickel: Acustica Audio Nickel

The target. Nickel, by Acustica Audio (2021), is a suite built around an emulation of the 1176: its compressor, Nickel Comp, switches between two revisions, H and LN, each sampled from a different unit with the developer's dynamic convolution engine. The suite also contains two EQs and a preamp. The compressor's controls are not documented in the sources consulted.

The settings.

- Main row: **Threshold** −11.2 dB; **Ratio** 6.15; **Knee** 8.04 dB; **Cv Lin/Db** 1; **Attack** 4.7 ms; **Release** 250 ms.
- **DET (in)** tab: **Det Mix** avg.
- **ENV (cv)** tab: **EvMu Lin/Db** 1.
- **A/R (cv)** tab: **Attack Shape** 0.936; **Release Mode** 1.7.

Analysis. This preset imitates only the dynamics: no waveshaper, no EQ. That's a surprise for a plugin whose very principle is to capture a unit's coloration by sampling. Its recipe is that of **ao_fetish**: a decibel curve with a moderate knee, a decibel envelope, an **Attack Shape** of 0.936, and a blended release, here 70% linear (the linear part, in decibels, comes back at a constant rate, about 38 dB per second for 250 ms). The ratio of 6.15 falls between the 1176's 4:1 and 8:1 buttons: it's a value found through analysis, at a setting of the plugin that we don't know. **Det Mix** on avg has no effect, since the peak is the only stream included in the mix.

> **♪ In the studio: acustica_nickel**
>
> Try it on a vocal or a bass: it's firm compression (ratio 6.15), but rounded off by its knee. Set the **Threshold** first. If you miss the color of real hardware, add it yourself with the waveshaper: **Fet odd** between 1.5 and 3, then listen with **Monitor Fet** (the workflow from chapter 17).

pulsar_1178: Pulsar Audio Pulsar 1178

The target. The Pulsar 1178 emulates the 1178, the stereo version of the 1176 released in 1980, with no input transformer or class A circuitry, and based on revision H. As on the 1176, the threshold is fixed and you set the amount of compression with the input; the attack runs from about 800 to 20 µs, the release from about 1.1 s to 50 ms. Pulsar documents a slight saturation in the original, and adds four saturations built into the gain reduction (Tape, Warm, Triode, Clip), a sidechain EQ and a lookahead.

The settings.

- Main row: **Ratio** 4; **Attack** 9.4 ms; **Release** 14.1 ms (the **Threshold** stays at −20 dB, on a linear curve).
- **GR (cv)** tab: **Env Offset** −0.4.

- **WS** tab: **Fet drive** +1 dB; **Fet even** 0.2.
- **F (ws)** tab: **fet src mix** 0; **fet low herz** 100 Hz; **fet low reso** 0.25; **fet low mix** 4.

Analysis. The ratio of 4 echoes the 1176's first button, on a linear curve. The 14 ms release is shorter than the 1176's fastest (50 ms): with peak detection, such a short release lets the gain follow the individual cycles of the low frequencies, a source of distortion (chapter 12). Softube, in fact, writes that its FET Compressor distorts most with the fastest release (and the ALL ratio). The −0.4 offset is the strongest in the family. It lowers the signal by 4.4 dB at rest and silences the target as soon as the envelope reaches 8 dB of reduction. At steady state, on a sine and with peak detection, a simulation gives −4.4 dB of gain up to −20 dBFS, −10 dB at −12 dBFS, −17.6 dB at −6 dBFS and −35 dB at 0 dBFS: above about −12 dBFS, the output level drops as the input rises. This preset is therefore very sensitive to input level, and the curves figure at the start of the chapter, which ignores the offset, doesn't show it.

The waveshaper is set to the second harmonic (**Fet even** 0.2, drive +1 dB). Its product also contains a DC component and a copy of the fundamental in inverted polarity (chapter 17). The F (ws) tab keeps only the extreme lows: the low band at 100 Hz, with a resonance of 0.25, is a very gentle low-pass, and its mix of 4 multiplies the product by 4 in the subsonic range, by about 2 at 50 Hz and by 1 at 100 Hz. What gets fed back in, then, is mainly the very low components of the squared signal, which follow its energy, and an inverted copy of the extreme lows, which slightly attenuates them, all the more as the compression gets heavier. Since the amount of saturation follows the envelope shifted by the offset, it never drops below 0.4: the effect exists even without compression. Pulsar describes nothing of the kind: this is an adjustment found through analysis.

> **♪ In the studio: pulsar_1178**
>
> Start with a moderate input level, for example a bass or a kick peaking around −18 dBFS, then adjust the **Threshold** while listening carefully: beyond a certain level, the compression crushes the signal instead of containing it. Listen to the product with **Monitor Fet**: a very low movement that follows the level. For shallower compression, bring **Env Offset** back toward −0.2.

softube_germanium: Softube Chandler Limited Germanium Compressor

The target. Softube emulates under license (2020) Chandler Limited's Germanium Compressor, a mono FET compressor designed by Wade Goeke. The word "germanium" refers to the diodes in the knee circuit (six curves, from R SOFT to ZENER HARD) and to the "Germanium Drive" output stage. The unit offers "Clean" or "Dirty" compression (the FET distortion is compensated or not) and a Feedback control on the output stage, which adds distortion, gain, lows and highs. Its times have never been quantified: the attack goes from "relatively fast" to "very slow." According to Softube, "it's never transparent."

The settings.

- Main row: **Attack** 188 ms; **Release** 568 ms (the **Threshold**, the **Ratio** and the curve stay at −20 dB, 2 and linear).
- **GR (cv)** tab: **Env Offset** −0.455.
- **DEQ** tab: **Deq low herz** 240 Hz; **Deq low max** 4.71; **Deq low zdf->svf** 0.
- **WS** tab: **Fet odd** 0.863.
- **F (ws)** tab: **fet src mix** 0; **fet low herz** 450 Hz; **fet low zdf->svf** 0; **fet low mix** 1.

- **DEQ 2** tab: **Deq low herz** 80 Hz; **Deq low max** −1; **Deq hgh herz** 4,800 Hz; **Deq hgh max** 0.5.
- **BIAS src** tab: **Bias input add** 1.
- **BIAS dst** tab: **Bias x base** 5.995.
- **F (bs)** tab: **bias amp** −0.5.
- **OUTPUT** tab: **deq** (the first one) 0.

Analysis. The 188 ms attack, the slowest in the family, corresponds to the "very slow" end of the hardware. The threshold and ratio have stayed at their default values: it's the −0.455 offset that does most of the compressing. It lowers the signal by 5.3 dB at rest; at steady state, on a sine, a simulation gives about 10 dB of reduction at −12 dBFS, 14 dB at −6 dBFS and 17 dB at 0 dBFS, versus 2 to 5 dB without the offset. Softube's "never transparent" can be taken literally here: even without compression, the preset is never neutral.

It's also the richest preset in the family, with nine tabs modified in addition to the main row, and three saturators. The DEQ strongly boosts the lows below 240 Hz, but the first **deq** in the OUTPUT tab is at 0: this low end isn't heard, it only feeds the waveshaper. This is the "saturate the lows without hearing them" approach from chapter 18. Because of the offset, the envelope is 0.545 at rest, and the boost there reaches about +11 dB; it decreases as the compression sets in. The waveshaper applies an odd curve below 1 (**Fet odd** 0.863), and therefore an expansive one (chapter 17): it slightly attenuates medium levels and leaves the peaks alone. Its amount follows the gain reduction and, again because of the offset, never drops below 0.455. The F (ws) tab keeps only the product below 450 Hz. The DEQ 2, placed afterward, removes lows below 80 Hz (about −6.8 dB at rest) and adds highs above 4.8 kHz (about +2.1 dB), two corrections that fade with compression. Finally, this is the only preset in the family that calls on the Vari-Mu designer. The input height (**Bias input add** 1) makes the normal path and the inverse path compress differently; the output height (**Bias x base** 5.995), with no limiter, acts only through the gain difference between these two paths (chapter 21); **bias amp** at −0.5 flips the bias saturation and halves it: for the part due to the input height, the DC component and even harmonics of the normal path are then reinforced instead of being canceled (**bias amp** card, chapter 21).

The hardware's colorations (Germanium Drive, Feedback, "Dirty" compression) aren't described precisely enough to be matched one by one with these settings: it was through analysis that François combined these three saturators. One last surprise: the noisy grain that François associates with germanium saturators, the spreader mode from chapter 17, isn't used here; you'll find it in **waves_76**.

> **♪ In the studio: softube_germanium**
>
> Softube recommends the Germanium on drums, vocals, bass, buses and the master; this preset, with its slow times, lends itself well to a bus or a bass. Compare it level-matched, adding about 5 dB of **Makeup**, since it lowers the level even at rest. Then set the **Threshold**, and listen separately to its two saturations, with **Monitor Fet** (WS tab) and **Mon bias sat** (BIAS dst tab).

korneff_comp2: Korneff Audio Pawn Shop Comp 2.0

The target. The name probably refers to Korneff Audio's Pawn Shop Comp 2.0 (2020), an original design: a hybrid tube-and-FET circuit whose tubes, transformers and resistors you can swap. It offers a threshold from 0 to −100 dB, a ratio from 2:1 to 50:1, an attack from 0.2 to 200 ms and a release from 10 ms to 1.2 s, as well as an EQ (Weight and Focus). Sound On Sound considers it ideal for squashing a vocal in parallel, and on drums.

The settings.

- Main row: **Ratio** 40; **Attack** 0; **Release** 0 (the **Threshold** stays at −20 dB, on a linear curve).
- **DET (in)** tab: **Det Mix** env; **peak db** −24 dB; **env db** 0 dB; **env pow** 0.907.
- **A/R (in)** tab: **Attack** 35 ms; **Release** 180 ms.
- **WS** tab: **Fet Precision** 25.
- **F (ws)** tab: **fet amp** −0.5.
- **OUTPUT** tab: **deq** (the first one) 0; **ws** 0; **deq** (the second one) 0.

Analysis. This preset has an architecture all its own. With **Attack** and **Release** at 0, the follower placed after the curve is disabled: all the smoothing is left to the detection envelope, placed before the curve, with 35 ms of attack and 180 ms of release. This is exactly configuration B from chapter 9, the one used by many commercial compressors. It has two consequences: the release time depends on the level reached (the farther the signal went past the threshold, the longer it takes the smoothed level to come back down), and a transient shorter than the attack may never cross the threshold. The times are within the plugin's ranges, and the ratio of 40, on a linear curve, pins the output almost to the threshold, within the advertised 50:1. The power of 0.907 makes the detection a little more sensitive and the compression a little softer (chapter 13).

The waveshaper is set up as a bit crusher (**Fet Precision** 25, i.e., steps of 0.04) with a flipped product reduced by half (**fet amp** −0.5). But the three target amounts in the OUTPUT tab are at 0, and **FX ws** stays at 0 in the INPUT tab: the waveshaper isn't computed, and these two settings have no audible effect (**Monitor Fet** stays silent). The "tube" side of the Pawn Shop Comp is therefore not represented: the preset imitates only its dynamics.

> **♪ In the studio: korneff_comp2**
>
> Do as Sound On Sound suggests: squash a vocal and blend it in parallel with **Dry/Wet**, or try it on drums. Set the **Threshold** first. To let more or fewer of the attacks through, play with the **Attack** in the A/R (in) tab: it's this one, not the one on the main row, that sets the speed of this preset.

> **To be confirmed by François**
>
> Two settings are surprising. In **pulsar_1178**, the −0.4 offset combined with the linear 4:1 curve at −20 dB makes the output level drop when the input exceeds about −12 dBFS at steady state (−35 dB of gain on a sine at 0 dBFS): is this intentional? In **korneff_comp2**, the **deq**, **ws** and **deq** amounts in the OUTPUT tab are at 0, so **Fet Precision** 25 and **fet amp** −0.5 have no audible effect: should they be removed, or was the waveshaper meant to be heard?

The breath of tubes: vari-mu and tube compressors

In a **vari-mu** compressor (variable-mu, or variable-bias), the element that reduces the gain is a tube: the control voltage shifts its grid bias, and the tube's gain drops. Remote-cutoff tubes lend themselves to this job. The documented traits of the genre follow from it: no fixed ratio, but compression that gets firmer as the level rises, a soft knee, times that are often slow or program-dependent, and the color of tubes and transformers. The Pulsar Audio article that inspired Comprendre's bias (chapter 19) adds two points: the loop is a feedback loop, with the output driving the gain reduction, and the harder you drive the stage, the more the compression turns into saturation, especially on transients.

The fifteen presets in this family cover the whole history of the genre: Rein Narma's Fairchild 670, through three plugins; the Altec 436C from the 1950s; the Gates Sta-Level from the broadcast world; the first vari-mu designs of the 1950s, reinterpreted by Klanghelm; two Ian Sefchick units inspired by the Fairchilds; the Manley Variable Mu, through Pulsar Audio's plugin; then more recent machines, the SPL Iron and the Rockruepel Comp.Two. Three Acustica Audio products, sampled from tube units, round out the list, along with a poorly identified T-RackS module.

Comprendre offers this family four tools tailored to it: the **Vari-Mu designer** (the bias, chapters 19 to 21), the **Mu** envelope follower, whose speed depends on level (chapter 11), **feedback** detection (**FF->FB**, chapter 14), and a very wide knee combined with an infinite ratio, which turns the entire curve into a knee (chapter 10). Here is what the presets do with them (see also the matrix at the start of the chapter):

- **7 of the 15 presets turn on the bias.** The three 670 imitations (**overloud_670**, **tr_670**, **acustica_ultramarine4**) use it the same way: a large input height (**Bias input add** from 2.5 to 42), computed from the output as smoothed by the bias detector. This is the control loop described in the Pulsar article. Their destination settings (output height, limiter) do almost nothing below 0 dBFS. **klang_mjuc_mk1** and **pulsar_mu** work on both sides, detector and audio; **rockruepel_comptwo_drums** takes the raw output, with no smoothing; and **arturia_tube-sta** only wakes the module up to use its limiter and the **F (bs)** tab.
- **8 presets do without it**: **tr_vintage**, **ddmf_mde**, **ddmf_mde2**, **blackrooster_omnitec436**, **spl_iron**, **acqua_magenta**, **acqua_titanium** and **acustica_cobalt**. Their vari-mu character then comes by other routes, some of which the bias presets also take: feedback (4 presets in all, counting **tr_670** and **acustica_ultramarine4**), the very wide knee with an infinite ratio (4 presets: **acustica_ultramarine4**, **blackrooster_omnitec436**, **arturia_tube-sta**, **klang_mjuc_mk1**), **Env Offset**, François's favorite imitation tool (3 presets), or **Gr Amp**.
- **The Mu follower is used three times**: in the **ENV (cv)** tab for **rockruepel_comptwo_drums** (in an extreme way) and **acqua_titanium** (almost imperceptibly), and in the **ENV (bs)** tab for **overloud_670**, where it stretches the bias release.
- **Color** comes from the waveshaper in 6 presets, most often through a touch of **Fet odd** that grows with the compression. Six presets apply EQ with **DEQ** or **DEQ 2**, and one trait keeps coming back: four of them (**overloud_670**, **tr_670**, **tr_vintage**, **pulsar_mu**) cut the extreme low end at rest, by 3 to 10 dB at 20 Hz. It's an adjustment found through analysis, and no documentation explains it.

- **The curve** is in decibels for 7 presets, linear for 7 others, and halfway between for **rockruepel_comptwo_drums**. Six presets set the ratio to 42.

A few surprises are worth pointing out. **pulsar_mu**, the imitation of the plugin whose publisher wrote the article the bias is built on, gets its transient saturation from the **opto designer**; its bias limiter, for its part, stays silent below 0 dBFS. **spl_iron**, which targets a "variable-bias limiter", doesn't use the bias and keeps the default times. **acustica_cobalt** imitates a feedback tube compressor but stays in feed-forward. Finally, none of the three 670 imitations touches **L/R to M/S** or **Channels Unlink**: they work in linked stereo, whereas the machine could run as two independent limiters or in lateral/vertical mode (sum and difference).

The curve figure at the start of the chapter shows only the main-row curve. In this family, feedback, the power settings, the offset and above all the bias take the actual compression far away from it: the sections that follow give estimates, based on calculations with the formulas in the code.

overloud_670: Overloud Gem Comp670

The target. Overloud's Gem Comp670, announced in October 2018, emulates the Fairchild 670; Overloud sampled three units, in London, Los Angeles and Milan. Designed by Rein Narma in the 1950s, the 670 is a vari-mu: the control voltage acts on the grid of the tubes, which themselves serve as the gain-reduction element. Its time-constant selector runs from 0.2 ms / 0.3 s (position 1) to automatic positions whose release depends on the program (0.2 s for an isolated peak, 10 s for repeated peaks, up to 25 s for a constant high level). Overloud mentions the original's feedback amplifier; the plugin adds an M/S mode, a soft or hard knee, and a "DC Threshold" control.

The settings.

- Main row: **Threshold** 0 dB; **Ratio** 42 (infinite); **Attack** 0; **Release** 0; **Cv Lin/Db** 1 (decibel curve).
- **DET (in)** tab: **peak db** −18 dB.
- **GR (cv)** tab: **Gr|Env Pow** 8.
- **WS** tab: **Fet odd** 1.025.
- **DEQ 2** tab: low band at 27 Hz, **Deq low max** −1, in ZDF (**Deq low zdf->svf** 0); high band at 6.75 kHz, **Deq hgh max** 0.125, **Deq hgh reso** 1.
- **BIAS src** tab: **Bias input add** 25.
- **DET (bs)** tab: **peak db** −24 dB (measurement excluded); **env db** 0 dB; **env pow** 0.5; **destination** *env* and **formula** *d=max(s,p)*.
- **ENV (bs)** tab: **Env->Mu** 1; **Mu Min Rel** 421; **Mu Max Rel** 0.6226; **Mu Pow Rel** 1.704.
- **A/R (bs)** tab: **Attack** 0.2 ms; **Release** 0.3 s.
- **BIAS dst** tab: **Bias on pre->out** 1 ("out" position); **Bias x env** −1; **Bias on limit** 1; **Bias limit knee** 6.92; **Bias on ratio** 1.3.

Analysis. The main compressor seems to be neutralized: threshold at 0 dB, detection lowered by 18 dB, zero attack and release. Without the bias, in fact, it would never compress a signal below 0 dBFS. Everything goes through the input height: **Bias input add** adds 25 times the bias value to the detection, a height much larger than the signal itself. This value is taken from the output (the default source), then smoothed by the bias detector, which is set to the envelope alone and to its square root (**env pow** 0.5). The compressor is therefore driven by its own output, through a detector

that has its own attack and release: this is exactly the loop described in the Pulsar article (chapter 19). Since the main-row times are zero, it's the times in the **A/R (bs)** tab that give the compressor its dynamics: 0.2 ms as the bias rises, 0.3 s as it falls, the values of the 670's position 1.

The Mu envelope in the **ENV (bs)** tab then stretches this release according to the output level. In this tab, the "Max" speed applies when the bias is zero, and the "Min" speed when it is at its maximum (chapter 11): the release is about 0.2 s in quiet passages, but the calculation gives about 1 s for an output around −30 dBFS, 5 s around −20 dBFS and 15 s around −14 dBFS. So the bias comes back down slowly as long as the music stays loud, and quickly as soon as it calms down: a program-dependent release reminiscent of the machine's automatic positions 5 and 6. Finally, **Gr|Env Pow** at 8 multiplies the reduction in decibels by eight, and it's what gives the loop its firmness. A simplified simulation of the code's formulas, on a sustained 1 kHz sine, puts the onset of compression around −20 dBFS, with a slope of about 4:1 above it: with 0 dBFS at the input, the output stays around −15 dBFS.

The rest does little. In the "out" position, an output height cancels out in the recombination of the two paths as long as the limiter isn't working (chapter 21); and this limiter, set at 0 dBFS with a narrow knee and a gentle ratio of 1.3, is only reached by output peaks close to 0 dBFS, which hardly ever happens with an output this compressed. **Bias x env** and the limiter thus act as a safety net more than as a color. The **DET (bs)** formula is selected, but its parameter stays at 0: no formula is applied. **Fet odd** at 1.025 leaves only a trace. At rest, the **DEQ 2** removes the extreme low end (a 6 dB-per-octave high-pass around 27 Hz, or about −4.5 dB at 20 Hz) and adds about 1 dB above 10 kHz; both corrections fade by half at 6 dB of gain reduction. These are adjustments found through analysis.



Figure 149 — The overloud_670 preset, A/R (bs) tab. On the main row, Attack and Release are at 0; it's the attack (0.2 ms) and release (0.3 s) of the bias detector, the values of the Fairchild 670's position 1, that set the dynamics of the compression. The tabs the preset modifies are in yellow.

> **♪ In the studio: overloud_670**
>
> Try it on a stereo bus or a vocal, and listen to how the compression stays settled in after a loud passage, then lets go when the music breathes: that's the bias Mu at work. To set the amount, start with the **Threshold**: since the detection roughly follows the square root of the output, each decibel you take off the threshold makes the compression start about two decibels lower. To change the speed, leave the main-row times alone and use the **Attack** and **Release** in the **A/R (bs)** tab. And to recreate the 670's lateral/vertical mode, turn on **L/R to M/S** (MAIN tab) and set **Channels Unlink** to 1: Mid and Side are then compressed separately (chapter 7).

tr_670: IK Multimedia T-RackS Vintage Tube Compressor/Limiter Model 670

The target. This T-RackS module, introduced with T-RackS 3 Deluxe in 2008, emulates the Fairchild 670, a vari-mu compressor/limiter whose entire signal path runs through tubes. IK promises an "extremely fast" attack, a set of release times, two independent limiters usable in L/R or vertical/lateral mode, and tube circuitry that "enriches the timbre" by adding presence and harmonics. For the Fairchild 660, the documentation gives 2:1 in compression and 30:1 in peak limiting, with intermediate settings. IK recommends it first and foremost for the stereo bus and mastering.

The settings.

- Main row: **Threshold** −23.6 dB; **Ratio** 42; **Attack** 5 ms; **Release** 0.3 s; **Cv Lin/Db** 1.
- **INPUT** tab: **FF->FB** 1 (feedback).
- **DET (in)** tab: **peak pow** 1.987.
- **A/R (cv)** tab: **Release Mode** 2 (linear).
- **GR (cv)** tab: **Gr|Env Pow** −0.6666.
- **DEQ** tab: low band at 104 Hz, **Deq low max** −1.342, in ZDF; mid band at 224 Hz, **Deq mid max** 0.1207, **Deq mid reso** 1.12.
- **WS** tab: **Fet odd** 1.706; **Fet even** 0.072.
- **DEQ 2** tab: low band at 20 Hz, **Deq low min** and **Deq low max** −0.5, in ZDF; high band at 16 kHz, **Deq hgh min** and **Deq hgh max** −0.25, in ZDF.
- **BIAS src** tab: **Bias input add** 2.534.
- **DET (bs)** tab: **peak db** −24 dB; **env db** 0 dB; **env pow** 0.821.
- **A/R (bs)** tab: **Attack** 20 ms; **Release** 1 s.
- **BIAS dst** tab: **Bias on limit** 1; **Bias x gr** 0.3333.

Analysis. The structure is that of **overloud_670**, with one difference: the main compressor works too, and in feedback (**FF->FB** at 1). It measures the output, as a vari-mu does according to the Pulsar article. In feedback, the detection sees a signal that has already been reduced, and the ratio actually obtained stays well below the displayed ratio (chapter 14); **peak pow** close to 2 doubles the decibels of the detected level and stiffens the compression (chapter 13). The 0.3 s release is the 670's position 1, and it's linear (**Release Mode** at 2): the gain comes back up at a constant rate. The 5 ms attack, on the other hand, is slower than the machine's 0.2 to 0.4 ms. **Gr|Env Pow** at −0.6666 mostly accentuates small reductions (chapter 15).

The bias adds a height of 2.5 times its value to the detection, taken from the smoothed output (20 ms attack, 1 s release). This height is at least two and a half times the output level: the detection is largely driven by the smoothed output, well beyond the amount François recommends. The effect is clear. According to a simplified simulation of the code's formulas, on a sustained 1 kHz sine, the compressor without the bias would only start working around −10 dBFS; with the bias, it starts around −25 dBFS, and its slope gets steeper with level, from about 2.5:1 at first to 4:1 around −8 dBFS, and more still near 0 dBFS. This is the behavior you'd expect from a vari-mu: a ratio that grows with level. On the destination side, **Bias x gr** and the limiter set at 0 dBFS only form the "push-pull" bend of chapter 19 for peaks very close to 0 dBFS; on a sustained sound, the simulation finds almost no effect from them.

The color comes from the waveshaper: **Fet odd** at 1.706 and a hint of **Fet even** produce harmonics, mostly odd, whose amount follows the gain reduction. This is a direct translation of the "presence and harmonics" IK promises. The dynamic EQs draw an almost fixed profile: about −10 dB at 20 Hz, −5.6 dB at 50 Hz, −2 dB at 100 Hz, a bump of a third of a decibel around 224 Hz, and −1 dB above 16 kHz. **Deq low max** goes beyond −1: the extreme low end is removed, then comes back with inverted polarity (chapter 18). This profile is an adjustment found through analysis.

> **♪ In the studio: tr_670**
>
> It's a natural candidate for the stereo bus: the compression comes in gently and firms up as the mix pushes harder. Listen to it on a chorus that steps up a notch: the gain should tighten without flattening the attacks. Adjust the **Threshold** first; if the low end feels too thinned out, check what the **DEQ** tab is removing by bringing **Deq low max** back toward 0. As with **overloud_670**, **L/R to M/S** and **Channels Unlink** at 1 bring the preset closer to the machine's lateral/vertical mode.

acustica_ultramarine4: Acustica Audio Ultramarine 4

The target. Released in May 2019, Ultramarine 4 is built around the A-70, a stereo and mid-side compressor that reproduces the Fairchild 670, sampled from the original unit. Acustica describes short attack times and long release times that hold the transients while giving the body of the track a "massage", and recalls the 670's historical use in mastering and broadcasting. On the machine, position 3 of the time-constant selector gives 0.4 ms of attack and 2 s of release.

The settings.

- Main row: **Threshold** −36 dB; **Ratio** 42; **Knee** 20 dB; **Attack** 0.61 ms; **Release** 2 s; **Cv Lin/Db** 1.
- **MAIN** tab: **Pre Clip Mode** and **Post Clip Mode** at a value outside the list of modes.
- **INPUT** tab: **FF->FB** 1.
- **DET (in)** tab: **peak pow** 2.03.
- **BIAS src** tab: **Bias input add** 42.
- **DET (bs)** tab: **peak db** −24 dB; **env db** 0 dB; **env pow** 4.
- **A/R (bs)** tab: **Release** 2 s; **Release Auto** 1; **Release Altitude** 0.8479; **Release Mode** 2 (linear).
- **BIAS dst** tab: **Bias x gr** 2.5.

Analysis. The times are those of the 670's position 3: 0.61 ms of attack versus 0.4 ms, and exactly 2 s of release. The curve combines an infinite ratio, a 20 dB knee, feedback detection and a detection power close to 2: it gets progressively firmer, with no break. The bias adds a second stage on top. Its value is the smoothed output raised to the power of 4 (**env pow**), multiplied by 42: it stays negligible at moderate levels, then grows very fast; the height only exceeds the signal once the output goes above about −11 dBFS. According to a simplified simulation on a sustained 1 kHz sine, the preset gives practically the same curve with or without the bias up to about −16 dBFS (less than 0.3 dB of difference); beyond that, the bias adds reduction, up to 1.7 dB at 0 dBFS, and the slope goes from about 3:1 to 4.4:1. So the bias firms up the compression on the loudest passages: once again, a ratio that grows with level.

The bias detector keeps the default attack (10 ms) and comes back down very slowly. Its release is linear, and the altitude of 0.8479 makes it cover only 15% of full scale in 2 s. **Release Auto** at 1 further adapts this time to the crest factor: it is roughly preserved on a sustained sound and shortened on a percussive one (chapter 12). The bias charges quickly and discharges slowly. **Bias x gr**, without

a limiter, can only be heard through the gain difference between the two paths (chapter 21): nothing measurable on a sustained sound, a slight asymmetry on transients. The modes of the input and output clippers have no effect, since those clippers stay disabled (thresholds at 0 dB). The preset doesn't touch Mid/Side, even though the A-70 is advertised as a mid-side compressor.

> **♪ In the studio: acustica_ultramarine4**
>
> On a master or a bus, with a few decibels of reduction, listen to the low end and the body of the track: the very short attack holds the peaks, and the 2 s release avoids pumping. Then push a loud passage to hear the bias firm up the compression. Set the **Threshold** first; on more rhythmic material, shorten the main-row **Release** before touching the bias release.

tr_vintage: IK Multimedia T-RackS (uncertain module)

The target. The name points to IK Multimedia T-RackS, but no T-RackS module is called just "Vintage". Only two modules contain the word: the 670 compressor already imitated by **tr_670**, and an equalizer. Two hypotheses remain: a second imitation of the 670 module, without the bias, or another module from the Vintage line (an emulation of the 1176, of the LA-2A, or an optical compressor). Nothing in the settings points to a FET or an opto.

The settings.

- Main row: **Threshold** −27 dB; **Attack** 59.9 ms; **Release** 255.6 ms (the **Ratio** stays at 2, the curve linear).
- **DEQ** tab: low band at 77.7 Hz, **Deq low max** −0.751; mid band at 2.07 kHz, **Deq mid max** −0.149, **Deq mid reso** 0.44; high band at 5.23 kHz, **Deq hgh max** −0.121.
- **DEQ 2** tab: low band at 300 Hz, **Deq low max** −0.25; mid band at 50 Hz, **Deq mid max** 0.15; high band at 5 kHz, **Deq hgh max** −0.05.

Analysis. The compression is simple: a ratio of 2 on a linear curve, gentler than a 2:1 in decibels on strong peaks, a slow 60 ms attack and a medium release. No bias, no feedback, no waveshaper: nothing specific to vari-mu designs. All the character lies in the two dynamic EQs, whose bands all have only a **max** setting. So they act at rest and fade out when the compressor works (chapter 18). At rest, the calculation gives about −10 dB at 20 Hz, −7 dB at 30 Hz, −2.7 dB at 50 Hz, half a decibel more between 300 and 500 Hz, then −1 to −1.5 dB above 8 kHz; at 6 dB of reduction, these deviations are roughly halved. The sound thickens in the extreme low end and opens up slightly in the top end as the compression settles in. This low-end profile is reminiscent of **tr_670**'s, which points toward a module from the same family, without proving it.

> **To be confirmed by François**
>
> No T-RackS module is called "Vintage", and the preset has no bias, no feedback and no waveshaper: only its dynamic EQs set it apart. Which module (and which setting of that module) does **tr_vintage** imitate?

> **♪ In the studio: tr_vintage**
>
> Try it on a vocal or a guitar bus, listening to the extreme low end: it hollows out in quiet passages and comes back when the compressor works. With 60 ms of attack, note attacks get through almost untouched. First set the **Threshold** for a few decibels of reduction, then, if the low end floats too much, bring **Deq low max** in the **DEQ** tab back toward 0.

ddmf_mde: DDMF MagicDeathEye

The target. MagicDeathEye is the mono version of a DDMF plugin that reproduces a unit by Ian Sefchick, a mastering engineer at Capitol Records, “loosely based on a Fairchild 660”: a variable-gain tube compressor. According to the manual, **Threshold** sets the level sent to the sidechain; a Time selector runs from 1 (fastest) to 4 (slowest), plus two dual-release positions; a Fast, Medium or Slow switch modifies all the times; and a 150 Hz filter is fitted to the sidechain. The publisher intends it for vocals, bass, keyboards and the master bus.

The settings.

- Main row: **Ratio** 2.19; **Attack** 2.29 ms; **Release** 675.5 ms (the **Threshold** stays at −20 dB, the curve linear).
- **GR (cv)** tab: **Env Offset** −0.39.

Analysis. The base is restrained: a moderate ratio, a fairly fast attack and a long release, in the spirit of a vari-mu’s medium-to-slow times. The standout feature is **Env Offset**, François’s favorite imitation tool. At rest, it takes off 4.3 dB permanently. During gain reduction, it weighs much more heavily, because it is added in amplitude to an envelope that is already low (chapter 15). The static calculation gives an output of −24.3 dBFS for a peak at −20 dBFS, −23.4 dBFS at −14 dBFS, −21.1 dBFS at −6 dBFS and −18.3 dBFS at 0 dBFS. Just above the threshold, the compression is therefore very firm (about 7:1 between −20 and −14 dBFS); it then softens (about 2:1 between −6 and 0 dBFS). That’s the opposite of the usual vari-mu tendency, and it was the analysis that dictated it. The envelope never drops below 0.51 for a signal below 0 dBFS, so the offset can’t make the target go silent.

> **♪ In the studio: ddmf_mde**
>
> On a vocal or a bass, this preset holds the level very early above the threshold, then lets the peaks breathe a little more. Since the offset lowers the volume by 4.3 dB even at rest, compensate with the **Makeup** before comparing with and without the plugin. Then adjust the **Threshold**: it decides where this first firm step sits.

ddmf_mde2: DDMF MagicDeathEyeStereo

The target. The name probably refers to MagicDeathEyeStereo, released in May 2020; no “MagicDeathEye 2” could be found. It reproduces the stereo model of Ian Sefchick’s unit: two mono compression circuits, with additions for mastering and bus compression. It offers an EQ built into the vari-mu gain structure, a low cut, an elliptical filter for vinyl cutting, adjustable harmonic distortion, a 200 Hz sidechain high-pass and a Punch mode. Its compression is described as much gentler than the mono version’s.

The settings.

- Main row: **Threshold** −23.36 dB; **Ratio** 1.83; **Knee** 20.41 dB; **Attack** 6.85 ms; **Release** 167.4 ms; **Cv Lin/Db** 1; **Channels Unlink** 1.
- **DEQ 2** tab: low band at 60.5 Hz, **Deq low max** −0.0545, **Deq low reso** 1.208; mid band set to 73 Hz, but with no amount.

Analysis. The preset faithfully translates the two documented traits. The compression is very gentle: a ratio below 2 and a 20 dB knee in decibels, which starts nearly 34 dB below 0 dBFS. **Channels Unlink** at 1 gives two independent compressors, like the machine's two mono circuits. The release is noticeably shorter than that of **ddmf_mde**. The dynamic EQ does almost nothing: about −0.5 dB below 30 Hz and +0.2 dB around 80 Hz at rest. The mid band has a frequency but neither **min** nor **max**: it is inactive. The machine's own EQ and distortion are not reproduced.

> **♪ In the studio: ddmf_mde2**
>
> This preset is made for the master or the bus: a few decibels of reduction, almost invisible. Since the channels are unlinked, a loud source on one side lowers only that side; listen to the stability of the stereo image, and bring **Channels Unlink** back toward 0 if it moves too much. The **Threshold** remains the first control to adjust.

blackrooster_omnitec436: Black Rooster Audio OmniTec-436C

The target. Black Rooster Audio's OmniTec-436C emulates the Altec 436C, a vari-mu tube compressor from the 1950s. The 436 series was used for telephone, broadcast and sound reinforcement: the 436A had no controls at all, the 436B (1958) an input gain, the 436C a threshold and a release time; EMI modified 436Bs to make Abbey Road's RS124. The plugin has no ratio knob: its vari-mu curve depends on the program. It advertises an attack of 10 to 100 ms, a release of 250 ms to 2.5 s, and a Comp mode (rounded knee) or Lim mode (a firmer, faster grab on peaks).

The settings.

- Main row: **Threshold** −14 dB; **Ratio** 42; **Knee** 31.11; **Attack** 0.77 ms; **Release** 119 ms (linear curve).
- **CURVE** tab: **Knee Auto** −14.53.
- **WS** tab: **Fet odd** 1.299.
- **F (ws)** tab: **fet amp** 0.5287.

Analysis. Since the target has no ratio, the preset takes an infinite ratio and leaves everything to the knee. On a linear curve, a **Knee** of 31.11 spreads around the threshold, from about −23 to −10 dBFS: within this zone, the ratio grows continuously up to infinity, which gives a curve with no fixed ratio. A negative **Knee Auto** narrows this knee as the smoothed reduction increases: it disappears at around 4 dB of reduction (chapter 10). On a loud, sustained passage, the curve therefore becomes a hard limiter at −14 dBFS; in quiet passages, it stays rounded. This automatic shift from a round knee to a firm grip is reminiscent of the plugin's two modes, Comp and Lim. The times, on the other hand, are much shorter than the advertised ranges (0.77 ms versus at least 10 ms, 119 ms versus at least 250 ms). These are the values that made the envelopes overlap in analysis mode: a displayed time isn't necessarily measured the same way from one compressor to another (chapter 12). The color is a touch of odd harmonics (**Fet odd** 1.299), which grows with the compression and which **fet amp** cuts in half.

> **♪ In the studio: blackrooster_omnitec436**
>
> The publisher suggests it for the mix bus, drums, bass and vocals. On drums, listen to how isolated hits get rounded off, then how a dense passage is held more firmly as the knee tightens. Adjust the **Threshold** first; for a slower character, closer to the plugin's ranges, lengthen the **Attack** and **Release**.

arturia_tube-sta: Arturia Comp TUBE-STA

The target. Arturia's Comp TUBE-STA, released in April 2019, emulates the Gates Sta-Level, a varimu-type tube broadcast compressor. The original shares its circuit with the GE Unilevel, uses a 6386 tube, and offers up to 40 dB of gain reduction and a single or double release. The plugin offers a variable Recovery control, an external sidechain with EQ, five detection modes including an M/S one, a dry/wet mix and automatic level compensation.

The settings.

- Main row: **Ratio** 42; **Knee** 38.29 dB; **Attack** 182.9 ms; **Release** 632.5 ms; **Cv Lin/Db** 1 (the **Threshold** stays at −20 dB).
- **GR (cv)** tab: **Env Offset** −0.2.
- **WS** tab: **Fet Thresh** 0.66; **Fet Knee** 149.23.
- **BIAS src** tab: **Bias input add** 0.0000038.
- **BIAS dst** tab: **Bias Fm On** 0.5; **Bias on limit** 0.054; **Bias limit knee** 25.
- **F (bs)** tab: **bias src mix** 0; low band at 80 Hz, **bias low mix** 1; mid band at 160 Hz, **bias mid mix** 0.5; high band at 8 kHz, **bias hgh mix** 0.2036, in ZDF.

Analysis. The curve is that of a broadcast leveler. An infinite ratio and a 38 dB knee centered on −20 dB cover almost the whole useful range: the reduction is 1.1 dB at −30 dBFS, 4.8 dB at −20 dBFS, 11 dB at −10 dBFS and 20 dB at 0 dBFS, with a ratio that keeps growing. The 183 ms attack and the 632 ms release level things out without grabbing the transients. **Env Offset** at −0.2 deepens the reduction further: 1.9 dB at rest, 8.5 dB instead of 4.8 at −20 dBFS, 22 dB instead of 11 at −10 dBFS. As soon as the envelope reaches 14 dB of reduction, it goes to zero (chapter 15): on a loud, sustained passage, the reduction becomes total. This matches the range of the machine, capable of 40 dB of reduction. The waveshaper is a clipper with a knee so wide that the rounding starts at low levels; since its amount follows the gain reduction, the saturation grows with the compression.

The bias is used in an unusual way. Its input height is minute: **Bias input add**, one of the five amount controls, only serves to wake up the module (chapter 19). The destination then becomes a filtered saturator. The signal is divided by two (**Bias Fm On** at 0.5, with no unbiased fusion to compensate for it), then goes through a limiter set around −25 dBFS; only three bands of the product are kept (**bias src mix** at 0). A simulation of this stage alone, without compression, gives about −7 dB at 30 Hz, −1.5 dB at 60 Hz, −1 dB around 160 Hz and above 8 kHz, as well as odd harmonics in the low end: the third harmonic of a 30 Hz sine sits between 22 and 37 dB below the fundamental, depending on level. It is relatively stronger on quiet sounds. The limiter's knee (0.21 in amplitude) is indeed more than twice its threshold (0.054): near zero, the limiter no longer returns 0, but a small square step of about ±0.006, opposite in sign to the signal.

To be confirmed by François

In **arturia_tube-sta**, **Bias input add** is 0.0000038 and seems to serve only to activate the module. **Bias limit knee** (25, or 0.21 in amplitude) is more than twice **Bias on limit** (0.054): the limiter returns about ±0.006 (−44 dBFS) for a weak signal, a case pointed out in chapter 21, which adds grain to the low end of quiet passages. Is this intended?

♪ In the studio: arturia_tube-sta

It's a leveler: on spoken voice, a bass or a bus that varies a lot, it slowly brings everything toward the same level. Compensate the volume with the **Makeup**, then listen to the quiet passages: that's where the grain the bias adds to the low end is easiest to hear, and **Mon bias sat** (**BIAS dst** tab) lets you hear it on its own. Set the **Threshold** first; since the knee covers almost the whole range, lowering it raises the ratio little by little.

klang_mjuc_mk1: Klanghelm MJUC (Mk1 model)

The target. MJUC is a Klanghelm software compressor with three models. The Mk1 "is based on the very early variable-mu designs" of the 1950s: four vari-mu stages, compression done directly in the audio path, a very high control voltage to avoid the "thumping" of fast vari-mu designs, and the most pronounced color of the three models. It offers a saturation control (Drive), a tone control, six timing positions (the first one fast in both attack and release), a compression/limiting switch, and mono, stereo and dual-mono modes.

The settings.

- Main row: **Threshold** 0 dB; **Ratio** 42; **Knee** 67.42 dB; **Attack** 7.8 ms; **Release** 35.9 ms; **Cv Lin/Db** 1; **Channels Unlink** 0.5.
- **MAIN** tab: **Headroom** +12 dB.
- **BIAS src** tab: **Bias g.e->out** 0 (source: gain reduction); **Bias g.e g->e** 1 (envelope); **Bias Feedback** 0.3162; **Bias input add** 1; **Bias Fm In** 0.25.
- **BIAS dst** tab: **Bias x base** −30.3; **Bias on limit** 8.

Analysis. The curve first. The **Headroom** sends the signal into the compressor 12 dB hotter, and **Bias Fm In** at 0.25 takes those 12 dB back off the detection: the curve sees roughly the input level, while the audio processed inside, including in the bias limiter, is 12 dB hotter. With an infinite ratio and a 67 dB knee centered on 0 dB, the whole useful range is a knee. The local ratio is about 1.2:1 at −24 dBFS, 1.5:1 at −12 dBFS and 2:1 at 0 dBFS, for 0.7, 3.5 and 8.4 dB of reduction. This is the simplest translation of the vari-mu trait: compression that gets firmer as the level rises. The short times are reminiscent of the first position of the timing selector.

The bias, in turn, reproduces the mechanism of the Pulsar article better than any other preset in the family. Its value is the smoothed gain reduction, attenuated by 10 dB by **Bias Feedback**: like the bias of a tube driven by the control voltage, it grows when the compressor is working (chapter 20). It acts on both sides. At the input, a small height makes the two paths asymmetrical. Above all at the output, **Bias x base** adds a negative height of 30 times the bias before the gain, and the limiter, set at 8 (about +6 dBFS once the Headroom is removed), clips both paths. After recombination, you get the "push-pull" bend of chapter 19, and this bend moves down as the reduction grows. According to a static calculation, peaks start to saturate around −3 dBFS at the input; at 0 dBFS, the bend starts

around −6 dBFS and the peak loses about 2.4 dB. The harder you drive it, the more the compression turns into saturation: this is exactly the principle described by Pulsar. Finally, **Channels Unlink** at 0.5 blends the linked and unlinked gains, between the plugin's stereo and dual-mono modes.

> **♪ In the studio: klang_mjuc_mk1**
>
> On a drum bus or a bass, raise the **Input Db** to hear the compression gradually turn into saturation on the peaks, then compare with **Mon bias sat** (**BIAS dst** tab), which lets you hear the product on its own. For more discreet compression, stay below −6 dBFS: the wide knee does the work, and the bias bend stays at rest. Set the **Threshold** (or the **Input Db**) first.

pulsar_mu: Pulsar Audio Mu (now VM-Comp)

The target. Pulsar Audio's Mu, announced in September 2019 and since renamed VM-Comp, emulates the Manley Variable Mu, though the publisher doesn't name the machine. On the market since 1994, the Manley re-biases its tubes using the "remote cut-off" technique; in COMPRESS mode it has a soft knee and a 1.5:1 ratio, in LIMIT mode a harder knee, from 4:1 to 20:1. Its attack runs from 25 to 70 ms and its release from 0.2 to 8 s in five positions; the plugin extends the releases toward faster values and models transformer saturation. Above all, it was Pulsar's article on variable-bias compressors that inspired the bias: a remote-cutoff tube, a detector that is "typically" a peak detector, a feedback loop, a push-pull circuit, and transient saturation that the article holds responsible for the "glue".

The settings.

- Main row: **Threshold** −6 dB; **Knee** 6 dB; **Attack** 1 ms; **Release** 150 ms; **Cv Lin/Db** 1 (the **Ratio** stays at 2).
- **DET (in)** tab: **rms secs** 0.00001 s.
- **ENV (cv)** tab: **Gamma R delai** 0.25.
- **GR (cv)** tab: **Opto** 1; **Opto A/R mul** 0.2247; **Opt src gn->env** 0.96; **Opto saturation** 0.9; **Opto sat limit** 5.979 dB; **Opto precision** 0.6788 dB.
- **F (gr)** tab: **opto amp** 0.5287.
- **DEQ** tab: low band at 50 Hz, **Deq low max** −0.5.
- **DEQ 2** tab: low band at 18 Hz, **Deq low max** −0.33; high band at 16 kHz, **Deq hgh max** −0.05, in ZDF.
- **BIAS src** tab: **Bias input add** 1; **Bias Fm In** 2; **Bias out bs->ub** 1.
- **A/R (bs)** tab: **Attack** 0.
- **BIAS dst** tab: **Bias x base** 1; **Bias Fm On** 2; **Bias <> bs->ub** 1; **Bias on limit** 4; **Bias on ratio** 1.5.

Analysis. The compression stays on the machine's COMPRESS side: the ratio is unchanged (2:1) and the 6 dB knee is soft. The 150 ms release is close to the machine's fastest position (0.2 s), which the plugin extends toward shorter values. The 1 ms attack, on the other hand, is much faster than the machine's range: the plugin setting being imitated was probably fast, but we don't know it.

The bias follows the article closely. Its source is the output, the default source, as in the feedback loop described by Pulsar. It is measured after unbiasing (**Bias out bs->ub** at 1), so the bias follows the true output level. The bias detector stays in pure peak mode, with no smoothing, like the article's "typical" detector: the bias value follows the output waveform, sample by sample. So the preset doesn't reproduce the demonstration in chapter 19, which smooths the bias with RMS. François

notes, in fact, that he didn't get the article's shape with the Pulsar Mu itself. At the input, the height equals the bias, and **Bias Fm In** doubles the detection signal: the normal path's detector sees twice the signal plus the output level, the inverse path's detector twice the signal minus that level. The two compressors don't reduce in quite the same way, hence the asymmetry that the "push-pull" recombination turns into saturation. This detection gain also places the effective threshold around −15 dBFS, not at −6 dBFS.

The destination applies the "compensated drive" recipe from chapter 21: **Bias Fm On** at 2 pushes the target toward the limiter, the unbiased fusion (**Bias <> bs->ub** at 1) divides it back down, and **Bias x base** adds a height equal to the bias. But the limiter is set at 4, that is, +12 dBFS. On the doubled and shifted target, it is only reached above about +2.5 dBFS at the input: below 0 dBFS, the drive and the fusion cancel each other out exactly, and the height cancels out between the two paths, apart from a small residue. The transient saturation comes from elsewhere, and that's the surprise of this preset: from the **opto designer**. The opto follows the envelope at 96%, with very short times (0.22 ms and 34 ms at rest, shorter still as the reduction grows). During each attack, when it lags behind the reduction the curve calls for, **Opto saturation** makes it jump toward that reduction, all the more as the gap is large, within the limit set by **Opto sat limit** (chapter 15). Kept apart as a saturation product, this effect is cut in half by **opto amp**. The result is an abrupt reaction on transients that grows with the depth of compression: the saturation described by the article, reached by another route.

At rest, the dynamic EQs remove the extreme low end (about −3.3 dB at 20 Hz), add a small bump of about 1 dB around 80 Hz and take off a touch of top end above 16 kHz; all of this fades by half at 6 dB of reduction. These are adjustments found through analysis. Three settings have no effect. **rms secs** makes the plugin compute an RMS value that **rms db**, left at −24 dB, excludes from the detection. **Gamma R delai** doesn't count, since the Gamma envelope isn't used. Finally, the **Attack** in the **A/R (bs)** tab is inactive: the bias detector's envelope isn't in use.



Figure 150 — The pulsar_mu preset, BIAS dst tab. Bias Fm On at 2 and the unbiased fusion (Bias <> bs->ub at 1) form a compensated drive; Bias x base at 1 adds a height equal to the bias; the limiter (Bias on limit at 4, Bias on ratio at 1.5) is only reached above about +2.5 dBFS at the input. The modified tabs are in yellow: note GR (cv) and F (gr), the opto designer's tabs.

> **♪ In the studio: pulsar_mu**
>
> The publisher pitches the Mu as "glue" for the mix: try it on a stereo bus or a drum group, with a few decibels of reduction. Listen to the snare attacks and the consonants: that's where the opto designer adds its abrupt reaction. **Monitor opto** lets you hear that reaction on its own, **Mon bias sat** the bias's share. Set the **Threshold** first; to hear the bias limiter, you'd need a signal hotter than 0 dBFS, or a lower **Bias on limit**.

spl_iron: SPL Iron (Plugin Alliance)

The target. The SPL IRON plugin, developed by Brainworx and released in January 2019, emulates the SPL Iron, a tube mastering compressor released in 2015. SPL describes it as a "variable-bias limiter/compressor". Each channel has two tubes in parallel, a remote-cutoff 12AU7 and a sharp-cutoff 12AX7, whose grids receive the control voltage; an auxiliary optical limiter (Vactrol) catches extreme peaks in the control path. There is no ratio, and the intensity of the compression depends on the input level. A three-position Tube Bias control gives more reduction the higher it is set. The times vary with the chosen rectifier and are program-dependent. The unit, fitted with Lundahl transformers, is specified at 0.002% distortion at +10 dBu, and its output EQ offers AirBass and Tape Roll-Off.

The settings.

- **CURVE** tab: **Gr Amp** 1.66.
- **WS** tab: **Fet odd** 1.269.
- **DEQ 2** tab: high band at 5 kHz, **Deq hgh max** −0.125.

Everything else keeps its default values: **Threshold** −20 dB, **Ratio** 2 on a linear curve, **Attack** and **Release** at 10 ms.

Analysis. The preset leaves the main row alone and doesn't use the bias. Its only compression lever is **Gr Amp**, which multiplies the reduction at the moment it is applied (chapter 10): 2.4 dB instead of 1.4 dB for a peak at −17 dBFS, 7.3 dB instead of 3.6 at −10 dBFS, 11.9 dB instead of 5.2 at 0 dBFS. The compression is therefore much firmer just above the threshold (about 4:1) than on the loudest peaks (less than 2:1). Multiplying the reduction is reminiscent of the Iron's Tube Bias control, but we don't know which position was being imitated. The gain stays positive: the envelope doesn't drop below 0.55 for a signal below 0 dBFS, far from the polarity-inversion threshold (0.4). The default times are very short for a mastering compressor. With peak detection, a 10 ms release lets the gain follow the individual cycles of the low end, which creates distortion (chapter 12), and **Gr Amp** amplifies this modulation. The touch of **Fet odd**, proportional to the reduction, adds odd harmonics that the machine, known for being very clean, doesn't claim: it's an adjustment found by analyzing the plugin. The **DEQ 2** takes off about 1 dB above 12 kHz at rest, half as much at 6 dB of reduction.

> **♪ In the studio: spl_iron**
>
> On a master, start by lengthening the **Release** (100 to 300 ms): this keeps the low end from starting to buzz under compression. Then adjust the **Threshold** for a few decibels of reduction, and listen to the density gained on loud passages. The odd-harmonic grain and the slight softening of the top end only show up when the compressor is working.

rockruepel_comptwo_drums: Process.Audio Rockruepel COMP.TWO

The target. Process.Audio's COMP.TWO plugin, released in 2023, emulates the Rockruepel Comp.Two, an all-tube vari-mu designed and hand-built in Germany by Oliver Gregor. The gain reduction is done by a 6N3 input tube. It has no ratio, but a soft knee. The attack runs from 3.5 to 70 ms (with a measured minimum around 0.6 ms), the release from 200 ms to 4 s, and a fast release

also speeds up the attack. The press speaks of a “meaty” saturation. Among other uses, the publisher recommends parallel drum compression; “drums” probably refers to this use, not to a factory preset.

The settings.

- Main row: **Threshold** −26.8 dB; **Attack** 11.9 ms; **Release** 35.9 ms; **Cv Lin/Db** 0.5 (the **Ratio** stays at 2).
- **ENV (cv)** tab: **Env->Mu** 1; **Mu Min Rel** 0.0719; **Mu Max Rel** 421.
- **A/R (cv)** tab: **Attack Altitude** 0.284; **Attack Shape** 0.616; **Att Ret From** 0.
- **GR (cv)** tab: **Env Offset** −0.5; **Opto** −1; **Opto severity** −7.96; **Opto precision** 0; **Opto saturation** 1; **Opto sat limit** 0.000001 dB.
- **WS** tab: **Fet *** 0; **Fet Precision** 7.804; **Fet sin *** 1; **Fet sin x** 4.
- **F (ws)** tab: **fet amp** 42; **fet src mix** 0; low band at 1.1 kHz, **fet low mix** −1, in ZDF.
- **BIAS src** tab: **Bias input add** 7.63.

Analysis. This is the boldest use of the Mu follower in the family. The release goes from 0.07 times the **Release** in silence to 421 times when the detected level reaches 0 dBFS. The calculation gives about 3 ms in quiet passages, 0.19 s for a detected level of −20 dBFS, 0.67 s at −14 dBFS, 1.6 s at −10 dBFS and 3.9 s at −6 dBFS. On its own, the Mu thus reproduces the machine’s 200 ms to 4 s range, making it depend on level (chapter 11). The 12 ms attack falls within the machine’s range. With a shape below 1, a low altitude and **Att Ret From** at 0, each hit is grabbed right away, then the reduction finishes its course more gently (chapter 12).

With no smoothing at all, the bias adds 7.6 times the output level of the previous sample to the detection: the detector mostly reads the output, like a feedback compressor, and the two paths see each cycle differently. This height also raises the detection by about ten decibels, and therefore the level that drives the Mu: the long releases kick in sooner than you’d expect. Next come two radical settings in the **GR (cv)** tab. **Env Offset** at −0.5 takes off 6 dB at rest and zeroes the envelope as soon as it reaches 6 dB of reduction: on strong hits, once the attack is past, the target can go silent. The opto, for its part, follows the envelope almost instantly (**Opto severity** at −7.96), but **Opto saturation** at 1, with a near-zero limit, makes it jump all the way to the reduction the curve calls for at the start of each strong hit. With **Opto** at −1, this gap is flipped: instead of a stronger reduction, the gain rises up to twice the envelope, and the start of the hit pops out. So the preset behaves like an aggressive transient shaper: the front of the hit is pushed forward, the body is crushed, and the release depends on level.

Finally, the waveshaper adds a grain confined to the peaks. In “spreader” mode (**Fet *** at 0 and **Fet Precision** above 0), the very fast sine of **Fet sin x** at 4 becomes a noise whose amplitude only affects high levels (chapter 17), which François likens to certain germanium saturators. **fet amp** at 42 amplifies it enormously; the **F (ws)** tab keeps only the part below 1.1 kHz, with inverted polarity (**fet src mix** at 0, low band at −1). Since its amount follows the gain reduction, this dull grain only appears during compression. It’s an adjustment found through analysis; the press, for its part, speaks of a “meaty” saturation.

> **♪ In the studio: rockruepel_comptwo_drums**
>
> Use it the way the publisher recommends for drums: in parallel. Set the main-row **Dry/Wet** between 0.2 and 0.5, and listen to the kick and snare gain body and grain without losing their attack. Adjust the **Threshold** first: it decides how many hits cross the step where the body disappears. To measure what the variable release brings, compare with **Env->Mu** at 0.

acqua_magenta: Acustica Audio Magenta

The target. Magenta (version 4 in 2018, then 5) is an Acustica Audio line sampled from tube units by an American manufacturer that "rhymes with 'Stanley'", with no official license; Manley comes to mind, but no source confirms it. Magenta 4 includes a vari-mu-style compressor/limiter, an optical compressor and a FET compressor. Magenta 5 comes in several models, including a full vari-mu described as having a soft knee. The version and model being imitated are unknown.

The settings.

- Main row: **Ratio** 2.87; **Attack** 50 ms; **Release** 1 s (the **Threshold** stays at −20 dB, the curve linear).
- **INPUT** tab: **FF->FB** 1.
- **ENV (cv)** tab: **EvMu Br/Cp** 1 (decoupled envelope).

Analysis. Five settings, but all consistent with a vari-mu. The feedback detection follows the topology described in the Pulsar article. The 50 ms attack is slow, and it falls within the attack range of the Manley Variable Mu (25 to 70 ms), if that is indeed the unit that was sampled. The one-second release is long. The decoupled envelope adds a natural hold after attacks and lengthens the apparent release even more (chapter 11). In feedback, the displayed ratio overstates the actual compression: a static calculation on peaks gives a slope of about 1.4:1 above −20 dBFS, with 2.3 dB of reduction at −14 dBFS and 6.3 dB at 0 dBFS. This matches the gentleness advertised for Magenta's vari-mu.

> **♪ In the studio: acqua_magenta**
>
> A starting point for a bus or a laid-back vocal: the compression comes in slowly, holds for a long time and is barely audible as an effect. Listen to the ends of phrases and the breaths, which the one-second release lets come back up gently. Since feedback softens everything, lower the **Threshold** further than you would in feed-forward to get the same reduction.

acqua_titanium: Acustica Audio Titanium

The target. Titanium (Titanium3, March 2019) is an Acustica Audio suite sampled from "some incredible tube-based units released over the last 25 years", without the hardware being named; the term vari-mu isn't used. The suite includes a three-band compressor and a single-band compressor, an all-tube passive EQ, a preamp and a channel strip. No ratio or time values could be found.

The settings.

- Main row: **Threshold** −40.25 dB; **Ratio** 12; **Attack** 72.4 ms; **Release** 35.2 ms (linear curve).
- **MAIN** tab: **Headroom** −15 dB.
- **INPUT** tab: **FF->FB** 1.
- **ENV (cv)** tab: **EvMu Br/Cp** 1; **Env->Mu** 1; **Mu Min Rel** 2.

- **GR (cv)** tab: **Env Pow Unpow** 2.
- **MOD & LA** tab: **la delay** +0.032 s; **la det pk** 1.

Analysis. The −15 dB **Headroom** sends the signal into the compressor at a lower level: the effective threshold is around −25 dBFS. In feedback and on a linear curve, the ratio of 12 softens a lot. The static calculation gives an output of −22.5 dBFS for a peak at −20 dBFS, −17.1 dBFS at −10 dBFS and −11.4 dBFS at 0 dBFS, a slope of about 1.8:1. The envelope is decoupled: with a 72 ms attack, longer than the release, the start of each release is noticeably slowed down (chapter 11). The Mu follower is set the opposite way from the vari-mu recipe: **Mu Min Rel** at 2 doubles the release in quiet passages, and only a detected level close to 0 dBFS brings it back to 35 ms. Since the Headroom lowers the level the detector sees, the release in practice stays around 66 to 70 ms. **Env Pow Unpow** at 2 slows the attack further and speeds up the release, without changing the reduction on a sustained sound (chapter 15). Watch out for **la delay**: it is positive. It's not a lookahead but a "reverse": with **la det pk** at 1, the compression arrives 32 ms later than usual and lasts that much longer (chapter 22). Added to the 72 ms attack, this delay lets the transients through almost entirely.

> **♪ In the studio: acqua_titanium**
>
> On drums or a group bus, the preset lets the attacks crack through, then tamps down the body of the sound with a slight delay, which can give a very particular kind of breathing. Listen to the snare: its crack gets through, its ring is held in check. Set the **Threshold** first; for a faster grab, bring **la delay** back to 0 before touching the **Attack**.

acustica_cobalt: Acustica Audio Cobalt

The target. Cobalt (June 2017, followed by a Cobalt2) reproduces the LM 9804, a tube compressor handmade by Luca Martegani (Xelius). The project started out as a reproduction of a feedback compressor in the spirit of the Altec Lansing units; the gain-control element went from a 12AT7 to a 12AU7, then to a 6BZ7, with a push-pull output and a custom transformer. Acustica says it shines on sustained sounds (vocals, guitars, bass, pads, strings) and that on drums it can go "from pumping to extreme squashing".

The settings.

- Main row: **Ratio** 2.4; **Attack** 30 ms; **Release** 250 ms (the **Threshold** stays at −20 dB, the curve linear).
- **ENV (cv)** tab: **Hold branch** 12.3 ms.
- **A/R (cv)** tab: **Attack Shape** 0.84; **Release Mode** 1.5; **Release Shape** 1.8.

Analysis. Surprise: the machine is a feedback compressor, but the preset stays in feed-forward and doesn't use the bias. All the character is entrusted to the envelope. The 12.3 ms hold, a little longer than half a period of a 41 Hz bass note, freezes the gain between two low-end peaks and removes its ripples, without slowing down the attack (chapter 11). The release is held back at first, then let go (**Release Shape** at 1.8), with a share of constant step (**Release Mode** at 1.5) that makes it more energetic. The attack grabs a little faster at the start (**Attack Shape** at 0.84). On a sustained sound, this envelope stays discreet; on drums, the held-then-released release can produce the pumping Acustica talks about. These are adjustments found through analysis.

> **♪ In the studio: acustica_cobalt**
>
> Start with what the publisher prefers, a vocal, a bass or a pad: the hold keeps the low end clean and the release follows the phrases. Then try it on a drum loop, lowering the **Threshold**: you'll hear the compression start to breathe with the tempo. The **Release** is the second control to adjust, depending on the tempo of the song.

The precision of the VCA: bus and console compressors

The VCA (voltage-controlled amplifier) is an integrated amplifier whose gain follows a control voltage. David Blackmer developed it in 1971, and the control law of dbx VCAs is "decilinear": the voltage directly sets the gain in decibels. Very linear and very flexible, the VCA brings none of the "good accidental properties" of tubes, optical cells or FETs: according to Dave Berners, all the character of a VCA compressor comes from the design of its timing and its curve (chapter 2). That makes it the ideal playground for Comprendre, where this design is made entirely of settings.

Two machines dominate the family. The **dbx 160** (1976) pairs the VCA with an RMS detector and a feed-forward topology. The **Solid State Logic console bus compressor**, present according to SSL as early as the 4000 B of 1976 and associated above all with the G series (1987), is the benchmark for the "glue" of a mix. Its values are switched: attack of 0.1, 0.3, 1, 3, 10 or 30 ms; release of 0.1, 0.3, 0.6 or 1.2 s, or **Auto**; ratio of 2, 4 or 10:1. Its automatic release has two time constants, 100 ms and 12 s according to SSL: loud elements are released quickly, and the release gets longer when the compression goes on. Its topology is hybrid: according to SSL, the sidechain has a feedback topology, but the signal that feeds it is taken feed-forward. A "slave" VCA, placed in the sidechain, reproduces the gain reduction of the audio VCA there. Nine of the family's thirteen presets target this compressor, either directly (the five ssl presets) or through an emulation (The Glue, BUSTER, Sand3, the T-RackS Bus Compressor); bx_townhouse targets a compressor built with SSL parts. The last three target the dbx 160 (ao_dbcomp), a Dutch console (acqua_tulip) and a modern VCA compressor (pa_bettermaker).

What these presets have in common jumps out on the tab map: they all modify the main row, and almost everything else happens in the timing tabs. **A/R (cv)** is modified by 8 of the 13 presets, **ENV (cv)** by 7, **BIAS src** by 6, **GR (cv)** by 3. The other tabs show up only once or twice, and no preset touches **BIAS dst**, the lookahead or the **OUTPUT** tab. A few settings keep coming back:

- **Cv Lin/Db** at 1 (decibel curve) in 10 of the 13 presets, and **EvMu Lin/Db** at 1 (decibel envelope) in 6: this is consistent with a gain element controlled in decibels;
- the most classic SSL bus compressor setting, **4:1, 30 ms, 0.1 s**, appears as is in 5 presets (ssl_1_f, ssl_2_b_, cytomic_glue, tr_bus, acustica_sand3); ssl_2_b also lands on SSL positions (4:1, 10 ms, 0.1 s), and the 30 ms attack appears in 8 presets;
- the automatic times, Comprendre's answer to the "Auto" release: **Attack Auto** in 6 presets, **Release Auto** in 4.

The presets range from the most stripped-down (cytomic_glue, 5 settings) to the most elaborate (ao_dbcomp, 21 settings). Several surprises are worth pointing out:

- the **bias**, Comprendre's "vari-mu designer", appears in 6 presets, but only through **Bias input mul**, which multiplies the detection. None of them sets an output height: here, the bias doesn't saturate, it changes what the compressor measures;
- the slight coloration of the VCA, often cited among the ingredients of the SSL "glue", is reproduced by none of the five ssl presets. Only tr_bus (barely) and ao_dbcomp (more clearly) use the waveshaper;
- the **opto designer** appears on one SSL preset (ssl_2_b_), and in negative;
- ao_dbcomp, which targets a compressor famous for its RMS detection, keeps the default peak detection;
- only one preset (ssl_1_f) switches to true feedback, despite the SSL's hybrid topology;
- two presets (ssl_1_b and ao_dbcomp), the only two of the 80, load with the **Analyse SC** mode active; together with kazrog_true-iron (later, saturator family), they are also the only three that invert the output polarity.

On the curve figure at the start of the chapter, most of the family's curves are 4:1 in decibels. For the six presets that multiply the detection, keep in mind that this curve applies to the **multiplied** level: compression doesn't start at the displayed threshold, as the next paragraph explains.

The detection multiplier. François added **Bias input mul** to the **BIAS src** tab to better imitate the SSL compressor (chapter 20). With the bias's default source, the output, the detection becomes the product of the signal and the output level, multiplied by the value of the setting. In concrete terms, in decibels: the measured level equals the input level, plus the output level, plus a fixed gain (37.5 dB for a setting of 75, 18 dB for 8). Two consequences:

- as long as the compressor isn't reducing the gain, the output follows the input, and the measured level rises twice as fast as the signal. So compression doesn't start at the **Threshold**, but lower or higher depending on the value of the setting (box below); above that point, it is firmer than with ordinary detection at the same ratio;
- as soon as the gain drops, the output drops, and the detection with it: this is an element of feedback, which tempers the compression.

The six presets concerned all leave the **Threshold** at −20 dB: in practice, it's **Bias input mul** that sets the level where compression starts. Since the multiplication is the same on the signal and on its inverse, it creates no asymmetry: without an output height, it produces no saturation product. Four of these presets also cut the inverse path (**Bias +inverse** at 0), which roughly halves the processing load; the bias then works as a feedback mechanism rather than as a push-pull.



Figure 151 — The ssl_1_f preset, BIAS src tab: Bias input mul at 6.5 and Bias +inverse at 0. The modified settings and the tabs that contain them (INPUT, CURVE, ENV (cv), BIAS src) are in yellow; the main row shows the classic SSL bus compressor setting: 4:1, 30 ms, 0.1 s.

> **A word from the author**
>
> Studying the schematic of the SSL compressors, François found that their feedback isn't really feedback: you don't make a copy of the signal, you take the output and multiply. It was to reproduce this mechanism that he added this setting, which multiplies the detector's input. In his view, you then get a very particular kind of compression, one that takes the sound somewhere else entirely.

> **Going further: where compression begins**
>
> With **Bias mul %** at 1 (default), the detection signal d becomes d × β × k, where k is **Bias input mul** and β the bias value. With the "out" source and the bias detector at its default settings, β is the absolute value of the output at the previous sample, capped at 1 (chapter 20). In decibels, the measured level is therefore L_in + L_out + 20 $\log_{10}$ k. With no gain reduction, L_out = L_in: the measured level reaches the threshold T when the input reaches (T − 20 $\log_{10}$ k) / 2. With T = −20 dB, that gives about −29 dB for k = 75 (ssl_1_b), −26 dB for 38.57 (ssl_2_b), −25 dB for 30.52 (ssl_2_b_) and −19 dB for 8 (cytomic_glue); for ssl_1_f, which measures the output, about −18 dB. These values apply to the peaks of a sustained sound: the smoothing of the envelope rounds off this onset of compression. In practice, on these presets, lowering the **Threshold** by 6 dB only lowers the onset of compression by about 3 dB; doubling **Bias input mul** lowers it by as much.

The five SSL presets. The names ssl_1, ssl_1_b, ssl_1_f, ssl_2_b and ssl_2_b_ say neither which plug-in served as the model, nor with which settings. The table compares them. All five switch the curve to decibels (**Cv Lin/Db** at 1).

Preset	Attack / Release	Ratio	Envelope	What the detector measures	Bias input mul (Bias +inverse)	Other settings
ssl_1	20 ms / 10 ms (default)	4	dB, decoupled, auto times 0.6	the input	—	Threshold −23.6 dB; Release Shape 0.333
ssl_1_b	6.9 ms / 27 ms	2.11	linear	input × output	75 (0)	Analyse SC, Phase inverted; Env Pow Unpow 1.173
ssl_1_f	30 ms / 100 ms	4	dB	output × output (FF->FB 1)	6.5 (0)	Gr Amp 2.789

Preset	Attack / Release	Ratio	Envelope	What the detector measures	Bias input mul (Bias +inverse)	Other settings
ssl_2_b	10 ms (default) / 100 ms	4	linear	input × smoothed output	38.57 (1)	bias detector on envelope
ssl_2_b_	30 ms / 100 ms	4	dB	input × output	30.52 (0)	Opto −1, A/R mul 8; shapes 0.936 and 1.128

The table reveals two groups. **ssl_1** stands apart: it's the only one without bias, the only one with a decoupled envelope, the only one to use the automatic times and to set its Threshold. The other four multiply the detection. The "1" series is mixed: a multiplier of 75 or 6.5, very different times, a feedback variant. The "2" series is consistent: 4:1, a 100 ms release, a multiplier between 30 and 39; ssl_2_b_ adds to ssl_2_b a 30 ms attack, the decibel envelope, slight shape tweaks and the opto designer, cuts the inverse path and returns the bias detector to its default settings. ssl_1_f, ssl_2_b and ssl_2_b_ land exactly on positions of the machine; ssl_1 only through its ratio, and ssl_1_b through none (its ratio is close to 2:1).

This suggests some hypotheses about the names. "b" would stand for the detection multiplier, common to the three presets that carry this letter; "f" for feedback, since ssl_1_f is the only one that sets **FF->FB**. For "1" and "2", two readings are possible: two versions of SSL's plugin (the first, derived from the 2011 Duende Native Bus Compressor, then SSL Native Bus Compressor 2), or two rounds of trials by François. The 20 ms attack of ssl_1 weakens the first reading: it doesn't exist on the machine, and among SSL's plugins, only the Bus Compressor 2 guide lists it. But it may also have been adjusted through analysis, like the times of ssl_1_b.

> **To be confirmed by François**
>
> Which plugin (SSL Native Bus Compressor 2, its first version, another emulation) and which settings of that plugin were used for each of the five ssl presets? What do "1", "2", "b", "f" and the final "_" mean? The text assumes that "b" stands for Bias input mul and "f" for feedback.

ssl_1: SSL bus compressor

The target. The SSL console bus compressor, described in the introduction, imitated through one of Solid State Logic's plugins: SSL Native Bus Compressor 2, or its first version. The name doesn't say which. For what follows, keep in mind the machine's **Auto** release with its two time constants.

The settings.

- Main row: **Threshold** −23.6 dB; **Ratio** 4; **Attack** 20 ms; **Cv Lin/Db** 1 (decibel curve);
- **ENV (cv)** tab: **EvMu Lin/Db** 1 (decibel envelope); **EvMu Br/Cp** 1 (decoupled envelope);
- **A/R (cv)** tab: **Attack Auto** 0.6; **Release Auto** 0.6; **Release Shape** 0.333.

Analysis. The ratio of 4 is one of the machine's positions. The 20 ms attack isn't: on the machine, 10 and 30 ms bracket it, but the Bus Compressor 2 guide lists this value. The curve and the envelope are both in decibels, which matches a VCA's control law.

The heart of the preset is its release. **Release** was left at 10 ms, its default value, but the envelope is decoupled and **Release Auto** is 0.6. In this case, Comprendre subtracts the automatic attack time from the automatic release time (chapter 12). With an attack twice as long as the release, the result is zero on all sustained sounds (as long as the square of the crest factor stays below 3), and it never exceeds 4 ms: the first stage of the decoupled envelope recovers instantly, or nearly so. It's then the second stage, set by the attack time, that sets the speed of the recovery (chapter 11). So the gain goes down and comes back up at roughly the same pace: about 20 ms on a sustained sound, less on drums, since **Attack Auto** shortens the attack as the crest factor grows (8 ms, plus 24 ms divided by the square of the crest factor). **Release Shape** only acts when the automatic release time isn't zero, that is, on the most percussive sounds.

The machine gets a program-dependent release from its two time constants. Comprendre goes about it differently, through the crest factor, but in the same direction: short times on transients, longer ones on steady sounds. The resulting recovery is still faster than any of the machine's positions, the shortest of which is 100 ms: it's an adjustment found through analysis, on a plugin setting we don't know.

> **♪ In the studio: ssl_1**
>
> Try it on a drum bus or a group of rhythm guitars, with 2 to 4 dB of reduction: the gain comes back between hits almost as fast as it goes down, which gives a nervous compression that "breathes" with the tempo. If the low end starts to buzz because the gain is following the cycles of the bass, raise **Release** first: around 100 ms, the automatic time becomes positive again and the recovery gets noticeably longer. Then set the **Threshold**.

ssl_1_b: SSL bus compressor

The target. The same as ssl_1. The "b" probably stands for the detection multiplier, **Bias input mul**, which this preset pushes further than any other.

The settings.

- Main row: **Ratio** 2.11; **Attack** 6.855 ms; **Release** 27.33 ms; **Cv Lin/Db** 1 (decibel curve);
- **MAIN** tab: **Analyse SC** checked; **Phase** unchecked (output polarity inverted);
- **GR (cv)** tab: **Gr|Env Pow** 1; **Env Pow Unpow** 1.173;
- **BIAS src** tab: **Bias input mul** 75; **Bias +inverse** 0 (inverse path cut).

Analysis. None of the times is a position of the machine, and the ratio of 2.11 is only close to the 2:1 position: these values were probably found by overlaying the gain curves, on a plugin setting we don't know. The multiplier is the strongest in the family: +37.5 dB when the output is at 0 dBFS. With the **Threshold** left at −20 dB, compression therefore starts around −29 dB of input (technical box in the introduction). The inverse path is cut: the bias works as feedback, and the processing is roughly halved.

The envelope was left linear. **Env Pow Unpow** at 1.173 tweaks it slightly: the attack becomes a little slower and the release a little faster, without changing the reduction obtained on a steady sound (chapter 15). **Gr|Env Pow** is set to 1, a neutral value: it shows up in yellow, but has no effect.

The two settings in the MAIN tab are those of the analysis setup in chapter 24. **Analyse SC** has no effect on the sound: it draws the golden sidechain curve, which only makes sense if another compressor is plugged into it. **Phase** unchecked, on the other hand, inverts the polarity of the whole output.

On a solo track, you can't hear it; but as soon as the signal is mixed with a copy of itself (parallel compression, drums recorded with several mics, comparison with the original track), the two partly cancel out.

> **To be confirmed by François**
>
> ssl_1_b and ao_dbcomp are saved with **Analyse SC** checked and **Phase** unchecked: when they load, the output is polarity-inverted. Is this intended, or should these two presets be saved again with Phase checked and Analyse SC unchecked?

> **♪ In the studio: ssl_1_b**
>
> Start by checking **Phase**. On a mix bus or a vocal group, this preset gives a compression that is gentle in its ratio but quick in its times: 7 ms of attack, 27 ms of release. Listen to how it follows words or accents rather than phrases. To set the amount of compression, use **Bias input mul** rather than the Threshold: halving it delays the onset of compression by about 3 dB.

ssl_1_f: SSL bus compressor

The target. The same as ssl_1. The "f" probably stands for feedback: it's the only SSL preset that makes the detector measure the output. In 2022, SSL in fact added an F/B mode to its The Bus+ hardware compressor, which takes the sidechain signal after the VCA, for a "more relaxed" compression.

The settings.

- Main row: **Ratio** 4; **Attack** 30 ms; **Release** 100 ms; **Cv Lin/Db** 1 (decibel curve);
- **INPUT** tab: **FF->FB** 1 (pure feedback);
- **CURVE** tab: **Gr Amp** 2.789;
- **ENV (cv)** tab: **EvMu Lin/Db** 1 (decibel envelope);
- **BIAS src** tab: **Bias input mul** 6.5; **Bias +inverse** 0 (inverse path cut).

Analysis. The three main values are positions of the machine: 4:1, 30 ms, 0.1 s. As for the topology, ssl_1_f takes the feedback idea all the way. With **FF->FB** at 1, the detector reads the output of the previous sample; the multiplier, the weakest in the family (+16.3 dB), multiplies it once more by the output level. In decibels, the compressor therefore measures twice the output level plus 16.3 dB, and starts acting around −18 dB of input.

Feedback alone softens the compression: the detector sees a signal that has already been reduced (chapter 14). **Gr Amp** compensates: it multiplies the linear reduction by 2.8 at the moment it is applied to the signal. An envelope at −1 dB thus gives a reduction of about 3 dB, an envelope at −2 dB a reduction of 7.4 dB, an envelope at −3 dB a reduction of 14.6 dB. In feedback, this factor adds to the loop gain: a small change in the measured level produces a large change in gain, and therefore in the output the detector reads. The loop thus reacts faster and more firmly than the 30 ms attack and the ratio of 4 would suggest. Measurements published in 2026 on a 1991 unit of the G384 rack compressor attribute effective attack times much faster than the displayed values precisely to the machine's loop gain. The preset brings a comparable mechanism into play, by another route.

A word of caution: with **Gr Amp** above 1, the applied gain becomes negative, meaning the polarity flips, as soon as the envelope drops below 1 − 1/Gr Amp, here around −3.9 dB (chapter 10). In feedback, the loop generally keeps the envelope above this point, since the output it measures drops very quickly. Keep an eye on it if you feed in a very hot signal, for example by raising the **Headroom**.

> ♪ **In the studio: ssl_1_f**
>
> This tight loop suits drum buses and dense mixes, when you want to hold the peaks without a very short attack setting. Listen to the way the hits are caught, then released in 100 ms. Adjust the **Threshold** first; here it only has half its usual effect. If the compression gets too dense, lower **Gr Amp** toward 2; at 1, you'll hear the "bare" feedback, much gentler.

ssl_2_b: SSL bus compressor

The target. The SSL bus compressor, as for ssl_1. If "2" refers to the plugin version, this would be SSL Native Bus Compressor 2, which adds a parallel mix, a sidechain high-pass filter and oversampling to the machine.

The settings.

- Main row: **Ratio** 4; **Release** 100 ms; **Cv Lin/Db** 1 (decibel curve);
- **BIAS src** tab: **Bias input mul** 38.57;
- **DET (bs)** tab: **peak db** −24 dB (bias peak excluded); **env db** 0 dB (bias detection envelope included).

Analysis. This is the most restrained of the SSL presets with bias: six settings. **Attack** was left at 10 ms, so the three main values land on positions of the machine: 4:1, 10 ms, 0.1 s. Compression starts around −26 dB of input.

What sets it apart lies in the bias detector. The peak is excluded there and the detection envelope included: the bias value no longer follows the output sample by sample, but is smoothed by the envelope of the **ENV (bs)** and **A/R (bs)** tabs, with their default times of 10 ms (chapter 20). The detection becomes the product of the signal and a smoothed output level, a factor that varies much more slowly than the waveform, like a control voltage. Of the five presets, it's the one where the multiplication most resembles the SSL's slave VCA, which multiplies the input by a gain.

Bias +inverse was left at 1: the inverse path is computed, but since the multiplication is the same on both paths, it changes nothing in the sound (chapter 20). Setting it to 0 would give the same result with half the processing.

> ♪ **In the studio: ssl_2_b**
>
> This is the most classic "working" setting of the bus compressor: 4:1, 10 ms, 100 ms. Put it on a mix bus or on subgroups, and aim for 1 to 3 dB of reduction using **Bias input mul** or the **Threshold**. Compare it with ssl_2_b_ on the tails of drum hits: ssl_2_b lets them come back more calmly.

ssl_2_b_: SSL bus compressor

The target. The same as ssl_2_b, of which this preset is a variant. The SSL bus compressor has no optical element: so here the opto designer isn't imitating a trait of the machine.

The settings.

- Main row: **Ratio** 4; **Attack** 30 ms; **Release** 100 ms; **Cv Lin/Db** 1 (decibel curve);
- **ENV (cv)** tab: **EvMu Lin/Db** 1 (decibel envelope);
- **A/R (cv)** tab: **Attack Shape** 0.936; **Release Shape** 1.128;
- **GR (cv)** tab: **Opto** −1; **Opto A/R mul** 8; **Opto precision** 0;
- **BIAS src** tab: **Bias input mul** 30.52; **Bias +inverse** 0 (inverse path cut).

Analysis. The main setting is the classic 4:1, 30 ms, 0.1 s, with a multiplier that makes compression start around −25 dB of input. The two shapes, very close to 1, are small tweaks found through analysis: the attack grabs a little faster at the start, and the release holds back a little before coming back up.

The opto designer is used in reverse, as a tool for reshaping the envelope. With **Opto** at −1, the applied gain equals the envelope minus the gap between the opto and the envelope (chapter 15). Now, with **Opto A/R mul** at 8, the opto is eight times slower than the envelope when there's no reduction: 240 ms of attack and 800 ms of release, before the reduction speeds it up. So it lags behind, and this lag is flipped:

- at the start of an attack, the opto reduces less than the envelope, so the applied gain reduces **more**: the attack dives faster and deeper;
- during the release, the opto is still reducing, so the applied gain comes back up **faster** than the envelope; at the end of a release, it can even briefly exceed unity, and the signal can come out, for an instant, louder than it went in.

With **Opto precision** at 0, the opto's speed varies continuously, without steps. The 2026 measurements cited for ssl_1_f describe an accelerated attack on the machine and a trajectory that isn't an exponential: the negative opto reshapes the envelope in the same direction.

> **♪ In the studio: ssl_2_b_**
>
> On a snare, a drum bus or a fingerstyle bass, compare it with ssl_2_b: the attack bites harder, and the end of the releases can produce a slight level rebound that can give the groove a lift. If this rebound bothers you, bring **Opto** closer to 0 (−0.5, for example) while watching the light-blue line in the history. Then set the **Threshold** for 2 to 4 dB of reduction.

cytomic_glue: Cytomic The Glue

The target. Cytomic's The Glue (2009), presented as a cross between the bus compressors of the SSL 4000 E and G consoles. The plugin adds an attack that goes down to 0.01 ms, a **Range** control that caps the gain reduction, a soft clipper (**Peak Clip**), a gentle 6 dB-per-octave filter in the detector and an external sidechain with high-pass. According to its publisher, the attack is slower on small signals.

The settings.

- Main row: **Ratio** 4; **Attack** 30 ms; **Release** 100 ms;
- **BIAS src** tab: **Bias input mul** 8; **Bias +inverse** 0 (inverse path cut).

Analysis. This is the most stripped-down preset in the family: the classic SSL bus compressor setting, plus the detection multiplier. The **Threshold** stays at −20 dB; with a multiplier of 8 (+18 dB), compression starts around −19 dB of input.

The preset keeps the default **linear** curve: it's one of the two presets in the family in this case. On a linear curve, a ratio of 4 compresses high levels less than in decibels: with a −20 dB threshold, a signal at 0 dBFS comes out at −9.8 dB instead of −15 dB (chapter 10). The multiplied detection, which rises up to twice as fast as the signal, partly makes up for this gentleness. The traits specific to The Glue (Range, Peak Clip, detector filter, slower attack on small signals) aren't reproduced by dedicated modules: the preset keeps its core, the SSL bus compressor at its most common setting.

> **♪ In the studio: cytomic_glue**
>
> This is the easiest "glue" preset to get to grips with: on a mix bus or subgroups, aim for 1 to 3 dB of reduction. Compare it with tr_bus, set the same way but in decibels and without a multiplier: listen to how each one handles the loudest peaks. Set the **Threshold** first, then **Bias input mul** if you want to move the onset of compression without touching anything else.

bx_townhouse: Brainworx bx_townhouse Buss Compressor

The target. Brainworx's bx_townhouse Buss Compressor (Plugin Alliance, 2018). It emulates a compressor built in 1978 by the engineers of the Townhouse studios in London, with console components supplied by SSL; it's probably a VCA, by lineage, but Brainworx doesn't say. The plugin adds an internal sidechain and a parallel mix, and is aimed at the master bus, subgroups and stems. The values of its controls aren't published.

The settings.

- Main row: **Threshold** −26.08 dB; **Ratio** 3.02; **Attack** 127 ms; **Release** 410 ms; **Cv Lin/Db** 1 (decibel curve);
- **DET (in)** tab: **auto secs** 0.4 s;
- **CURVE** tab: **Knee Auto** 0.2; **Knee Auto Secs** 0.1274 s;
- **ENV (cv)** tab: **Env->Mu** 1 (Mu envelope); **Mu Min Rel** 0.6853;
- **A/R (cv)** tab: **Attack Auto** 1; **Release Auto** 1; **Release Shape** 0.776; **Rel Ret From** 0.776;
- **DEQ** tab: **Deq low herz** 100 Hz; **Deq low max** −0.33.

Analysis. This is the most "program-dependent" preset in the family: almost nothing in it is fixed. With **Attack Auto** and **Release Auto** at 1, the displayed times are only references: Comprendre uses twice the set time divided by the square of the crest factor (chapter 12). On a sustained sound, the attack is therefore 127 ms and the release 410 ms; on drums, they become much shorter (25 ms and 82 ms for a squared crest factor of 10); on a very steady signal, they can double. **auto secs** at 0.4 s, twice the default, holds on to the peak of each hit longer: the times shorten even more on percussive signals. The nominal values go far beyond the SSL compressor's positions (30 ms of attack at most), but it's the automatic times that count: on drums, the attack falls back into the SSL's range.

The release is further modulated by the **Mu** envelope, whose time varies between 0.69 times the Release in quiet passages and 1 times near 0 dBFS (chapter 11): the compression lets go a little faster when the music calms down. **Release Shape** at 0.776 makes the gain come back quickly at first,

then linger; **Rel Ret From**, at the same value, caps the point from which this curve restarts at each new release. The automatic knee is discreet: in decibels, it only widens by 0.1 dB per decibel of reduction, which means a 0.6 dB knee for 6 dB of reduction.

The dynamic EQ acts on the signal, not on the detection. At rest, its coefficient of −0.33 removes about 3 dB at 20 Hz and 2 dB at 50 Hz, and adds 0.6 dB around 150 Hz; when the compressor reduces the gain, this correction fades (by half at 6 dB of reduction). So the extreme low end comes back as the compressor works. No documented trait of the plugin corresponds to it: it's an adjustment found through analysis, perhaps the trace of a frequency response specific to the plugin.

> **♪ In the studio: bx_townhouse**
>
> Put it on a mix bus or stems, the use the plugin is aimed at. Listen to how it lets the drum attacks through, then settles onto the sustained parts, without you having to choose any times. Set the **Threshold** first. If the automatic behavior seems too pronounced, bring **Attack Auto** and **Release Auto** back toward 0.5: the set times then regain half their weight.

tr_bus: IK Multimedia T-RackS Bus Compressor

The target. The T-RackS Bus Compressor (IK Multimedia, 2013), which emulates the VCA bus compressor of a British large-format console from the 1980s, later sold as a rack unit. IK doesn't name the console, but the description matches the SSL G-series compressor. The plugin reproduces the soft knee measured on the original, and adds a 1.5:1 ratio, a sidechain high-pass and a **Grit** switch, which reproduces the slightly stronger second harmonic of a miscalibrated unit.

The settings.

- Main row: **Ratio** 4; **Attack** 30 ms; **Release** 100 ms; **Cv Lin/Db** 1 (decibel curve);
- **ENV (cv)** tab: **EvMu Lin/Db** 1 (decibel envelope);
- **A/R (cv)** tab: **Attack Shape** 0.808; **Attack Auto** 0.12; **Release Auto** 0.12; **Release Shape** 1.288; **Rel Ret From** 0.52;
- **DEQ** tab: **Deq hgh herz** 203.93 Hz; **Deq hgh max** −0.00446; **Deq hgh min** 0.02931;
- **WS** tab: **Fet odd** 1.05; **Fet even** −0.008.

Analysis. The base is the classic 4:1, 30 ms, 0.1 s setting, in decibels, without a multiplier. The work goes into shaping the times. The automatic times, at 0.12, only add a touch of nuance to the set times. The attack (**Attack Shape** 0.808) grabs quickly at first, then finishes gently; the release (**Release Shape** 1.288) holds back at first, then comes back faster and faster. With **Rel Ret From** at 0.52, each release restarts at most from the middle of its curve, closer to the start, where the curve is flat: the hold-back is better maintained when attacks follow one another (chapter 12).

The plugin's soft knee isn't reproduced by the curve, which keeps a hard knee. The grain of the Grit switch, on the other hand, is sketched by the waveshaper, whose product is scaled by the gain reduction: **Fet even** at −0.008 creates a hint of second harmonic, with inverted polarity; **Fet odd** at 1.05 adds a very slight odd saturation. Both are tiny: we can't say whether Grit was engaged during the analysis. The dynamic EQ is almost neutral too: above 300 Hz, it only moves from −0.04 dB at rest to +0.25 dB at full reduction.

> **♪ In the studio: tr_bus**
>
> IK's demos are titled "Pumping Electro Mix", "Slamming Drums" and "Mix Comp": that makes a good test program. On an electronic mix, push the reduction toward 4 to 6 dB and listen to the held-back release, which makes the mix pump with the tempo. Adjust the **Threshold** first, then **Release**.

ao_buster: Analog Obsession BUSTER

The target. Analog Obsession's BUSTER, or its free version BUSTERse (2020), "Analog Obsession's take on the familiar SSL Bus compressor", a VCA design. It adds an external sidechain with high-pass and transient controls, a **Turbo** mode (compression across the whole spectrum, otherwise centered on the mids) and an **Xformer** mode, which emulates a transformer.

The settings.

- Main row: **Ratio** 7.06; **Attack** 30 ms; **Release** 100 ms; **Cv Lin/Db** 1 (decibel curve);
- **DET (in)** tab: **rms secs** 0.3526 s; **rms db** −10.95 dB; **rms pow** 0.822; **peak db** +1.92 dB; **source** rms; **parameter** 1;
- **BIAS src** tab: **Bias input mul** 13.25; **Bias mul %** 0.308; **Bias Fm In** 1.16.

Analysis. The times are the classic 30 ms and 0.1 s, but the ratio of 7.06, between positions 4 and 10, isn't an SSL value. Above all, ao_buster is the only preset in the family that reworks the detection in depth.

In the **DET (in)** tab, the formula was left on "d-=sp" and the destination on peak, their default values; with **source** on rms and **parameter** at 1, the formula becomes active (chapter 13). The peak, raised by 1.9 dB, has a slow RMS (350 ms) subtracted from it, lowered by 11 dB and then raised to the power of 0.822, which brings it up a little at low levels. The *max* blend then keeps the larger of the two. On a sustained sound, the RMS digs into the peak; on a transient that appears suddenly, the slow RMS hasn't risen yet, and the peak gets through almost whole. So the compressor reacts more to what sticks out above the average level than to the level itself. You could see this as an answer to the transient controls on the BUSTER's sidechain, without knowing whether they were engaged during the analysis.

In the **BIAS src** tab, the multiplication is dosed at only 30.8% (**Bias mul %**): the detection blends the ordinary measurement and the multiplied measurement, then **Bias Fm In** raises it by 1.3 dB. When the output is low, the detection is roughly ordinary (−1.9 dB); at −6 dB of output, it is already boosted by 10 dB. So the SSL character sets in with level. The inverse path is computed (**Bias +inverse** stays at 1), with no effect on the sound. The preset doesn't filter the detection, which would correspond to the Turbo mode rather than to the mid-focused mode; we don't know which one was engaged.

> **♪ In the studio: ao_buster**
>
> Try it on a drum bus or a percussive loop: with its ratio of 7 and a detection that watches for transients, it holds the hits while leaving the sustained parts freer. Set the **Threshold** first. For more even compression, lower **parameter**: the formula fades out, and the compressor moves back toward more conventional detection.

acustica_sand3: Acustica Audio Sand3

The target. Acustica Audio's Sand3 (2019), a suite of six plugins that emulates British consoles, SSLs according to the press, including a three-mode compressor. The current Sand page describes a channel compressor mode, a bus compressor mode ("the quintessential bus compressor") and that same mode with automatic release; whether these are Sand3's three modes isn't confirmed. Like the whole Acustica range, it relies on sampling the hardware, played back by a dynamic convolution engine.

The settings.

- Main row: **Ratio** 4; **Attack** 30 ms; **Release** 100 ms; **Cv Lin/Db** 1 (decibel curve);
- **ENV (cv)** tab: **EvMu Lin/Db** 1 (decibel envelope);
- **A/R (cv)** tab: **Attack Auto** 0.24;
- **GR (cv)** tab: **Gr|Env Pow** −0.89; **Env Offset** +0.173.

Analysis. The three main values are, once again, the classic 4:1, 30 ms, 0.1 s. Like tr_bus, this preset imitates the SSL compressor without a detection multiplier: the family's emulations are split between the two approaches. **Attack Auto** at 0.24 makes the attack slightly program-dependent; the release, for its part, stays fixed.

The work is done in the **GR (cv)** tab, with François's favorite tool for imitation, the offset. **Gr|Env Pow** at −0.89 first slightly accentuates small reductions: 1 dB becomes 1.3 dB, 6 dB becomes 6.7 dB. **Env Offset** at +0.173 then raises the whole envelope. At rest, it produces a permanent gain of +1.4 dB; during compression, it reduces the reduction: 6 dB called for by the curve become only 4, and 12 dB only 8 (chapter 15). It's an adjustment found through analysis, which gives lighter compression than the 4:1 would suggest, and a preset a little louder than the original signal.

> **♪ In the studio: acustica_sand3**
>
> On a mix bus, it adds density without cutting too much into the peaks. For a fair comparison, first make up for the 1.4 dB of permanent gain with the **Makeup** or an auto-volume (chapter 8). Then set the **Threshold**; for firmer compression, bring **Env Offset** closer to 0.

acqua_tulip: Acustica Audio Tulip

The target. Acustica Audio's Tulip (2022), the third plugin in its "Boutique Collection". It emulates the rare consoles of a Dutch company that had been building professional audio equipment for its own label since the 1950s; the press speaks of a tribute to Philips. It's a VCA compressor-limiter, with a COMP mode (the original model), an ULTRACOMP mode specific to Acustica, two brickwall limiters and an emulation of the preamp. Its ratios and times aren't published.

The settings.

- Main row: **Threshold** −25.28 dB; **Attack** 30 ms; **Release** 222 ms; **Cv Lin/Db** 1 (decibel curve); **Channels Unlink** 1 (unlinked channels);
- **A/R (cv)** tab: **Attack Auto** 0.5; **Release Auto** 1.

Analysis. The ratio was left at 2, its default value: it's one of the gentlest compressions in the family. The times are made program-dependent: the release fully (222 ms on a sustained sound, much less on drums), the attack halfway. François says he usually prefers a **Release Auto** of 0.5, since to his

ears the value 1 gives a slightly too square result (chapter 12): yet to imitate Tulip, he chose 1. The channels are unlinked: each channel has its own compressor, like two console channel strips side by side. It's the only preset in the family set up this way. The grain of Tulip's preamp isn't reproduced: the preset uses neither the waveshaper nor the bias.

> **♪ In the studio: acqua_tulip**
>
> With its 2:1, it suits vocals, acoustic guitars or keyboards, where you want to control the dynamics without it being heard. On a stereo bus, listen to the image: the unlinked channels compress left and right separately, which can make the image shift when a loud instrument is panned to one side. Set the **Threshold** first; if the image bothers you, bring **Channels Unlink** back toward 0.

ao_dbcomp: Analog Obsession dBComp

The target. Analog Obsession's dBComp, a VCA compressor "in the style of a classic vintage DBX 160", popular on drums, bass and acoustic guitars. The dbx 160 pairs an RMS detector with a feed-forward topology; the original model has a hard knee, as the soft **Over Easy** knee only arrived with the dbx 165 in 1978. The plugin, for its part, claims soft-knee compression of the Over Easy type, a high-pass in the detector and a mix control.

The settings.

- Main row: **Threshold** −19.28 dB; **Ratio** 11; **Attack** 20 ms; **Release** 125 ms; **Knee** 25;
- **MAIN** tab: **Analyse SC** checked; **Phase** unchecked (output polarity inverted);
- **ENV (cv)** tab: **EvMu Lin/Db** 1 (decibel envelope);
- **A/R (cv)** tab: **Attack Auto** 0.5; **Release Mode** 2 (linear);
- **WS** tab: **Fet odd** 1.24; **Fet even** 0.088;
- **F (ws)** tab: **fet hgh herz** 1,000 Hz; **fet hgh mix** −1; **fet hgh zdf->svf** 0 (one-pole filter);
- **DEQ 2** tab: **Deq low herz** 6.86 Hz; **Deq low min** −0.2; **Deq low max** −4; **Deq hgh herz** 8,000 Hz; **Deq hgh reso** 0.1263; **Deq hgh max** −1.

Analysis. First observation: the detection isn't RMS. The **DET (in)** tab is untouched, and the compressor measures the instantaneous peak. So the imitation doesn't go through the measurement, but through the curve, the times and the color.

The curve carries the plugin's "Over Easy" knee, with one peculiarity: it was left **linear** (**Cv Lin/Db** at 0). On a linear curve, a Knee of 25 is an amplitude width; with this threshold, the knee starts around −47 dB and ends around −13 dB (chapter 10). The value is just below the limit (26) beyond which the knee would start at silence and begin to "shave" low levels. The gain reduction sets in very gradually: 0.5 dB at −30 dB, 2 dB at the threshold, 5 dB at −13 dB, 14 dB at 0 dBFS. The ratio of 11, on a linear curve, doesn't compress like an 11:1 in decibels: between −10 and 0 dBFS, the output still rises by 3.5 dB for 10 dB of input, or about 3:1.

The release is in linear mode within a decibel envelope: the gain comes back up at a constant rate, about 76 dB per second with a Release of 125 ms (chapter 12), like a release expressed in decibels per second. **Attack Auto** at 0.5 shortens the attack on transients.

The preset also adds more color than the others in the family. The waveshaper adds odd harmonics (**Fet odd** 1.24) and a hint of second harmonic (**Fet even** 0.088), in an amount proportional to the gain reduction. The **F (ws)** filter then removes the top end of this product: with a one-pole filter and

a mix of −1, only its low-pass at 1 kHz remains, at 6 dB per octave (−3 dB at 1 kHz, −12.5 dB at 4 kHz). So the saturation stays in the low mids. These settings, like those of the DEQ 2, are adjustments found through analysis: no saturation is documented for the plugin.

The **DEQ 2** acts at both ends of the spectrum, and mostly at rest. Its low band, set at 6.86 Hz with a coefficient of −4, adds about 3 dB at 20 Hz, 1.6 dB at 30 Hz and 0.6 dB at 50 Hz; below 7 Hz, it sends the infrasonic content back with inverted polarity and amplified up to three times, which you can't hear but which affects the DC component. Its high band, at 8 kHz with a very low resonance and a coefficient of −1, only attenuates the extreme top end: a few tenths of a decibel at 16 kHz, about 2 dB at 20 kHz (at 48 kHz). Both corrections fade when the compressor reduces the gain.

Finally, like ssl_1_b, the preset loads with **Analyse SC** checked and **Phase** unchecked: the output is polarity-inverted (see ssl_1_b and the "To be confirmed" box that goes with it).



Figure 152 — The ao_dbcomp preset, MAIN tab: Analyse SC is checked and Phase unchecked, as during an analysis session. In the main row, the Knee of 25 applies to a curve left linear (Cv Lin/Db at 0).

> **♪ In the studio: ao_dbcomp**
>
> Check **Phase** first. The dbx 160 is a drum classic: try this preset on a kick, a snare or a bass. Listen to the very rounded onset of compression, and to the saturation that thickens the low mids when the compressor works. Then set the **Threshold**; for firmer compression, reduce the **Knee**.

pa_bettermaker: Bettermaker Bus Compressor

The target. Most likely Bettermaker's Bus Compressor from Plugin Alliance (2023); the brand is certain, the product less so, since Plugin Alliance offers two other Bettermaker compressors. The hardware, from the Polish company Bettermaker, is described as "a new approach to a well-known VCA design". It is given as feed-forward, with ratios of 1.5, 2, 4, 6 and 10:1, a 20 dB knee, an attack of 0.1 to 60 ms and a release of 10 to 2,000 ms. The plugin offers peak or RMS detection, an automatic release and adjustable "VCA THD" saturation.

The settings.

- Main row: **Threshold** −26 dB; **Attack** 30 ms; **Release** 250 ms; **Cv Lin/Db** 0.3;
- **A/R (cv)** tab: **Attack Mode** 2 (linear); **Release Mode** 2 (linear).

Analysis. The ratio was left at 2, a position of the machine, and the times fall within its ranges. The 20 dB knee isn't reproduced by the **Knee**: instead, the preset uses a blend of curves, 70% linear and 30% in decibels. A linear curve, read in decibels, compresses the most just above the threshold, then

less and less. Here the reduction rises to 2.6 dB at −20 dB, then only gains a decibel or two at each step: 4.6 dB at −14 dB, 6.4 dB at −6 dB, 7.2 dB at 0 dBFS. The compression levels off by itself, which suits a bus well.

Both times are in linear mode, within an envelope left linear: the gain goes down and comes back up at a constant rate, in a straight line, and reaches its target in a finite time, without the tail of an exponential. A 6 dB reduction is reached in about 24 ms, and the return to 0 dB takes about 200 ms (chapter 12). The plugin's automatic release and saturation aren't used.

> **♪ In the studio: pa_bettermaker**
>
> On a mix, vocal or guitar bus, it's a compression that makes itself forgotten: it grabs the first few decibels quickly, then levels off. Listen to the straight-line recovery after a loud passage, with no tail. Set the **Threshold** first for 2 to 3 dB of reduction, then **Release** to suit the tempo.

The chameleon: the Distressor family

The Empirical Labs Distressor EL8, designed by Dave Derr, an Eventide alumnus, went on sale in 1995. It belongs to none of the gain-element families: it's an analog VCA, with an analog audio path and largely digital control. Its originality lies elsewhere: bringing together in a single box the character of several units, starting with its designer's 1176 and LA-2A. Each Distressor "ratio" sets the threshold, the slope and the shape of the knee all at once. The 2:1 and 3:1 have parabolic knees (+15 dB for the 2:1), the 4:1 and 6:1 sharper knees; the 10:1 "Opto" calls on circuits that emulate an optical compressor, with a release that can reach 20 s; the 20:1 and the "Nuke" limit, the Nuke with a logarithmic release, fast then slower, which according to the manual accounts for a large part of the unit's sound. The attack ranges from 50 µs to 30 ms, the release from 0.05 to 3.5 s. Two distortions come on top of the compression: **Dist 2**, a "class A" warmth made mostly of the 2nd harmonic, and **Dist 3**, which adds the 3rd harmonic and gradually flattens the top and bottom of the waveform, like magnetic tape. The detector has a high-pass and a boost around 6 kHz; the EL8-X version adds a "British Mode" derived from a 1176 setting. A single unit that imitates several families: it is probably the hardware closest to Comprendre's philosophy.

The family's three presets target three software takes on the Distressor: Slate Digital's FG-Stress, an official recreation; Kiive Audio's Xtressor, an emulation; and Empirical Labs' Arousor, an evolution of the Distressor by its own manufacturer.

What they have in common in Comprendre (see the preset map at the start of the chapter):

- **The curve.** All three switch the curve to decibels, with ratios from 5 to 8. None of them uses a Distressor ratio, and that makes sense: a Distressor ratio also sets the threshold and the shape of the knee, so a Distressor 6:1 isn't an ordinary 6:1. Only **eli_arouser** has a knee.
- **The attack.** The three attacks (0.5 to 0.84 ms) are among the shortest in the collection, three to twenty times shorter than those of the 1176 emulations in the FET family (2.6 to 10 ms). In Comprendre, the speed of the transistor is found here instead.
- **The waveshaper.** All three presets engage it, the natural counterpart of Dist 2 and Dist 3, and all keep the default dynamic saturation, which grows with the gain reduction. All three use **Fet odd** (from 1.18 to 1.5, odd harmonics); the second harmonic of **Fet even** is clear only in **kiive_xstressor**, and is just a trace in **eli_arouser**.

- **What isn't used.** No preset engages the opto designer, even though the Distressor has its Opto mode, nor feedback, nor the detection filters. Two presets out of three add a DEQ 2 band; only one adjusts the detection (**slate_fgstress**), and only one the shape of the attack (**eli_arouser**).

slate_fgstress: Slate Digital FG-Stress

The target. Slate Digital's FG-Stress, a Virtual Mix Rack module announced in 2016, is presented as an official recreation of the Distressor EL8. It takes up the unit's eight curves, Dist 2 and Dist 3 (mutually exclusive) and the Opto shortcut, and adds a parallel blend; its detection filters are an 86 Hz high-pass and a 6 kHz bell, and it offers an 80 Hz audio high-pass. Slate recommends it on drum buses, room mics and aggressive vocals.

The settings.

- Main row: **Threshold** −22.4 dB; **Ratio** 8; **Cv Lin/Db** 1; **Attack** 0.59 ms; **Release** 36.8 ms.
- **DET (in)** tab: **rms secs** 69.7 ms; **rms db** +1.92 dB; **rms pow** 0.976; **peak db** −1.72 dB.
- **WS** tab: **Fet drive** +6 dB; **Fet odd** 1.299.
- **DEQ 2** tab: **Deq low herz** 2.17 Hz; **Deq low max** −1.238; **Deq low zdf->svf** 0 (one-pole filter).

Analysis. The hard-knee ratio of 8 falls between the unit's 6:1 and 10:1, on a machine whose ratios from 4:1 up have sharp knees. The 0.59 ms attack is within the Distressor's range; the 36.8 ms release is a little shorter than its documented minimum (50 ms), which may come from the way each one defines its times (chapter 12). This is the only preset in the family that adjusts the detection: a 70 ms RMS, raised by 1.92 dB, and a peak lowered by 1.72 dB, combined by the default *max* mix. On a sustained sound, the RMS leads (on a sine, it then exceeds the peak by about 1 dB); on transients whose peak exceeds the RMS by more than about 3.6 dB, the peak takes over (chapter 13). The Distressor manual mentions special detection circuits for the 2:1, the 10:1 and the Nuke, without describing the one used for the other ratios: this blend is an adjustment found through analysis.

The waveshaper answers Dist 3 with its odd part, but without the second harmonic that Dist 3 also contains. With a **Fet drive** of +6 dB, **Fet odd** shapes only levels below 0.5 (−6 dBFS internal): low levels are raised by about 2 dB, the curve flattens out as it approaches 0.5, then rejoins the diagonal with a kink (chapter 17). Dosed by the gain reduction, this product thickens the signal when the compressor works and adds odd harmonics. Finally, the DEQ 2 low band, at 2.17 Hz, removes the DC component and subsonics at rest, and fades away with compression: its effect is practically inaudible. It has nothing to do with the FG-Stress's 80 Hz audio high-pass.

> **♪ In the studio: slate_fgstress**
>
> Follow Slate's advice: a drum bus, room mics, a vocal that needs to stay aggressive. Set the **Threshold** first, then listen to the odd product on its own with **Monitor Fet** (WS tab) during a passage that compresses. If the note endings pump too much on a vocal, lengthen the **Release**: at 37 ms, it's very short.

> **To be confirmed by François**
>
> The slate_fgstress preset relies on **Fet odd** 1.3 with a **Fet drive** of +6 dB, the case flagged in chapter 17 where the curve shapes only levels below 0.5 and then rejoins the diagonal with a kink. If this behavior is changed, the preset's character will change: should this be planned for?

kiive_xstressor: Kiive Audio Xtressor

The target. Kiive Audio's Xtressor (2022) emulates the Distressor. The developer insists on one point: each ratio button acts as a compressor in its own right, "8 compressors in one plugin." The plugin offers a British mode that can be applied to every ratio, a TYPE mode that produces 2nd- and 3rd-order harmonics, and, in sidechain mode, a detection high-pass and a bump at 6 kHz.

The settings.

- Main row: **Threshold** −19 dB; **Ratio** 8; **Cv Lin/Db** 1; **Attack** 0.5 ms; **Release** 220 ms.
- **WS** tab: **Fet drive** −5.38 dB; **Fet even** 0.168; **Fet odd** 1.5; **Fet Thresh** 0.4; **Fet Knee** 50.
- **F (ws)** tab: **fet amp** 0.0625.
- **DEQ 2** tab: **Deq hgh herz** 7,000 Hz; **Deq hgh max** −0.33; **Deq hgh zdf->svf** 0 (one-pole filter).

Analysis. The compression is that of **slate_fgstress**, with the same hard-knee ratio of 8, an even shorter attack (0.5 ms) and a release six times longer (220 ms). The waveshaper, on the other hand, is the most complete in the family, in keeping with the plugin's TYPE mode: **Fet even** for harmonic 2, **Fet odd** for the odd harmonics, and a clipper with a knee. The negative **Fet drive** pushes the shaping toward high levels (chapter 17): the clipper rounds off the peaks and caps the curve's output around −2.6 dBFS, low levels are raised by about 2 dB, and the second harmonic makes the curve slightly asymmetrical. But **fet amp** divides this product by sixteen: you hear only a sixteenth of the difference between this curve and the diagonal, that is, at full level, a level change on the order of 0.15 dB and harmonics far in the background. A rich curve, dosed homeopathically: the saturation colors the sound without being heard as distortion.

The DEQ 2 high band softens the highs at rest (down to −3.5 dB above 7 kHz, with a gentle 6 dB per octave slope) and gives them back as the compressor works: compressed passages gain brightness relative to quiet ones. This isn't the plugin's 6 kHz bump, which acts on its detection: it's a tonal correction found through analysis.



Figure 153 — The kiive_xstressor preset, WS tab: second harmonic (Fet even 0.168), odd harmonics (Fet odd 1.5) and a clipper with a knee (Fet Thresh 0.4, Fet Knee 50), with a negative drive. The product is then divided by sixteen in the F (ws) tab.

> **♪ In the studio: kiive_xstressor**
>
> Try it on drums or a vocal: the 0.5 ms attack grabs the peaks, and the 220 ms release lets things breathe. Set the **Threshold** first. To hear what the waveshaper's curve really has in it, raise **fet amp** (F (ws) tab) toward 0.25 or 0.5: the color becomes much more present, from the grain of the second harmonic to the rounding of the peaks.

eli_arouser: Empirical Labs Arousor

The target. The Arousor is Empirical Labs' own plugin, "more of an evolution of the Distressor than it is an emulation," built on the EL8. It takes up all of the unit's ratios, and adds a "Rivet" ratio (the equivalent of the Nuke) and 6.5:1 and 7:1 ratios in ALT mode, a soft clipper that combines the Distressor's two distortions, an Expert panel (harmonic 2, compressor before or after the clipper) and two OPTO modes. Its attack ranges from 50 µs to 40 ms, and its AtMod control changes the shape of the attack's rise. Reviewer Barry Rudolph finds its distortion rather even at low levels.

The settings.

- Main row: **Threshold** −22.25 dB; **Ratio** 5; **Knee** 12 dB; **Cv Lin/Db** 1; **Attack** 0.84 ms; **Release** 142 ms.
- **A/R (cv)** tab: **Attack Shape** 0.648; **Att Ret From** 3.46; **Release Mode** 1.5.
- **WS** tab: **Fet even** 0.008; **Fet odd** 1.182.

Analysis. This is the only preset in the family with a knee: 12 dB, in decibels, from −28.25 to −16.25 dB, comparable to the parabolic knees of the Distressor's low ratios (+15 dB for the 2:1), here with a ratio of 5. Above all, it's the only one that touches the shape of the attack, and the correspondence with the target is direct: where the Arousor offers AtMod to change the shape of the attack's rise, Comprendre offers **Attack Shape**. At 0.648, below 1, the attack grabs a large part of the reduction right away, then finishes more gently (chapter 12). **Att Ret From** at 3.46 lets the position move far along this curve when attacks come in close succession; since **Att Ret Spd** stays at 1 and the attack lasts only 0.84 ms, the position quickly returns to the start as soon as a release goes on for a while: each isolated hit gets back the initial bite. The **Release Mode** at 1.5 blends the logarithmic release and the linear release in equal parts. The exponential part, proportional to the gap, weighs mostly at the start of large recoveries; the linear part, which is constant, wins out near the target: the gain reaches it decisively, without a long tail.

The waveshaper does the job of the Arousor's soft clipper: **Fet odd** at 1.182 gives a soft saturation that raises low levels by about 1.4 dB and creates odd harmonics, dosed by the gain reduction. **Fet even** at 0.008 is no more than a trace: the preset therefore leans toward odd harmonics, whereas Barry Rudolph describes a rather even distortion at low levels. The Arousor's OPTO modes are not used.

> **♪ In the studio: eli_arouser**
>
> Try it on a vocal or a bass: the 12 dB knee and the ratio of 5 give firm but rounded compression. Set the **Threshold** first, then, as you would with AtMod, play with the **Attack Shape**: toward 1 or above, more of the attacks get through; lower, they're caught earlier. To lean toward the second harmonic described by Barry Rudolph, raise **Fet even** toward 0.1.

British grain: the diode bridges

In a diode-bridge compressor, the gain-reduction element is a small diode bridge whose resistance varies with the control current. Rupert Neve discovered that two diodes arranged in a bridge clearly reduced distortion; the bridge nevertheless remains very "lossy": the signal is attenuated by about 40 dB before it, then brought back up by an amplifier. According to Dave Berners (Universal

Audio), the curves and times of such a compressor are designed independently of the element, the isolation between control and audio is good, but distortion can reach 10% under certain conditions: hence a distinctive sound, readily described as creamy, warm, round or silky. Very fast attacks produce audible transient distortion in these units.

The reference units come from Neve. The **2254**, designed in 1969 by Rupert Neve and David Rees, offers ratios from 1.5:1 to 6:1, a fairly soft knee (a transition over about ten decibels), a fixed 5 ms attack on the early versions, recovery times from 100 ms to 1.5 s depending on the version, and an **Auto** mode whose attack and recovery adjust themselves: fast on isolated peaks, slow on sustained high levels, so that the program "rides on a slowly moving platform of gain." The **33609** is its stereo rack version, renowned for "gluing" a mix together; its current spec sheet gives ratios from 1.5:1 to 6:1, a fast (3 ms) or slow attack (6 ms, with a 100 Hz high-pass in the detection), a manual recovery from 100 to 1,500 ms and two automatic modes.

Three presets make up the family: **arturia_diode** (Comp DIODE-609, an emulation of the 33609), **kush_silka** (SILIKA, built on a 2254 circuit) and **ao_britpressor** (a "Neve style" compressor whose exact model isn't known). What they have in common:

- **all three use Env Offset** (−0.1, −0.30, −0.2), so **GR (cv)** is the only tab, after the main row, that all three modify: this is the only family in the chapter in which every preset uses the offset;
- all three keep the default **Threshold** (−20 dB), the linear curve and a hard knee; their ratios (2, 4 and 3) stay within the range of the Neve units, and their releases (100, 100 and 400 ms) within that of the 33609's recovery, of which 100 ms is the shortest value;
- **none of them uses automatic release**: the Auto mode, the defining trait of the 2254, is imitated by none of them;
- only one uses the waveshaper, the natural tool for diode color: **arturia_diode**, and it uses it less as a saturation than as an EQ;
- **kush_silka** doesn't color the sound at all, and **ao_britpressor** uses… the opto designer, with a negative **Opto**;
- two presets touch the low band of the **DEQ 2** at 100 Hz, but in opposite directions: **arturia_diode** boosts the lows at rest, **ao_britpressor** attenuates them.

The negative offset has a particular effect here, because the curve is linear. With a linear curve, the gain never drops below 1/ratio; a negative offset deepens every reduction, and if its absolute value exceeds 1/ratio, the output starts to **decrease** above the threshold: that's the case with **kush_silka** (0.30 versus 0.25 for a ratio of 4). The curves figure at the start of the chapter, which ignores the offset and the Headroom, doesn't show this.

arturia_diode: Arturia Comp DIODE-609

The target. Arturia's Comp DIODE-609 presents itself as a "re-engineered" emulation of the Neve 33609, modeled on version C, an earlier, fully discrete design. It offers two compression panels (slow or fast attack) and a variable attack, a limiter with a deliberately non-transparent character, mid-side, a sidechain EQ, a calibration control that adds harmonics, and a dry/wet. Arturia recommends it on drums and the stereo bus.

The settings.

- Main row: **Release** 100 ms.
- **GR (cv)** tab: **Env Offset** −0.1.

- **WS** tab: **Fet odd** 1.56; **Fet env** 1.
- **F (ws)** tab: **fet src mix** 0; **fet low herz** 100 Hz; **fet low mix** −0.97.
- **DEQ 2** tab: **Deq low herz** 100 Hz; **Deq low max** 0.2.

Analysis. The compression is pared down to the essentials: only the **Release** has moved, to 100 ms, the 33609's fastest manual recovery. The ratio (2), the threshold (−20 dB) and the attack (10 ms) have stayed at their defaults, with the linear curve. **Env Offset** at −0.1 attenuates by 0.9 dB at rest and deepens the reduction a little: at steady state, 3.5 dB instead of 3.1 dB for a detected level of −12 dBFS, and 6 dB instead of 5.2 dB at 0 dBFS. The compression remains gentle, on the order of 2:1 just above the threshold and less and less beyond, which suits the 33609's role as "glue."

The waveshaper is used in an unexpected way. **Fet env** at 1, with **Fet gr** at 1 (default), makes the saturation constant: it no longer depends on the compression (chapter 17). **Fet odd** at 1.56 gives a soft curve that boosts low levels. But the product keeps only its low band, below 100 Hz, and **subtracts** it (**fet low mix** −0.97, **fet src mix** 0). Since the product of a quiet signal is mostly an amplified copy of that signal, the result is first of all an **attenuation of the lows** below 100 Hz: about 6 dB for a quiet signal, less and less as it approaches full internal level (about 1 dB at 0 dBFS). The low end therefore grows faster than the rest as the level rises, with a few harmonics of the extreme lows as a bonus. This is an adjustment found through analysis, closer to a level-dependent EQ than to a saturation.

The low band of the **DEQ 2**, at 100 Hz, does the opposite and adds 0.2 at rest (about +1.6 dB at the center of the band), a boost that fades away when the compressor works: the low end is then reduced a little more than the rest. The 100 Hz frequency is reminiscent of the high-pass that the 33609's slow position places in its detection; but here, both filters act on the signal you hear, not on the detection: the link, if there is one, is not direct.

> **♪ In the studio: arturia_diode**
>
> Try it on a drum bus or a stereo bus, the uses Arturia recommends. Set the **Threshold** first, still at −20 dB, to get 2 to 4 dB of reduction; raise the **Makeup** by about 1 dB to make up for the offset. Listen to the kick and the bass: on soft passages, they recede a little; on loud passages, they come back into place.

> **To be confirmed by François**
>
> In **arturia_diode**, **Fet env** at 1 (with **Fet gr** at 1) makes the saturation constant, and the product keeps only its lows below 100 Hz, subtracted (**fet low mix** −0.97, **fet src mix** 0). The waveshaper then acts mainly as an attenuation of the lows of quiet signals (about 6 dB), more than as a saturation. Is this the intended effect?

kush_silka: Kush Audio SILIKA

The target. SILIKA, by Kush Audio (the preset's name is spelled "silka"), is a "dual-diode compressor" reviewed by Tape Op in late 2020. It's an original design that combines two references: distortions modeled on the transformers of a 1973 ADR Compex limiter, and a compressor built on a Neve 2254 circuit. Its input offers a Zener diode model (brighter) or a germanium model (darker, fattening up the low end). Ratios of 2, 4, 12 or 20:1, attack from 30 to 0.1 ms, release from 5,000 to 15 ms, and a Fury knob that multiplies the distortion and makes the times ten times faster.

The settings.

- Main row: **Ratio** 4; **Release** 100 ms.
- **MAIN** tab: **Headroom** −7 dB.
- **GR (cv)** tab: **Env Offset** −0.30.

Analysis. Four settings. The **Ratio** of 4 is one of the SILIKA's positions, and the 100 ms release, like the default attack (10 ms), is within its ranges. The **Headroom** of −7 dB brings the signal into the compressor lower, which amounts to raising the threshold to −13 dBFS. The whole character rests on **Env Offset** at −0.30, one of the strongest among the factory presets. At rest, it attenuates the signal by 3.1 dB. With the linear curve, a ratio of 4 can't reduce the gain below 0.25; the offset, which subtracts 0.30, therefore makes the **output drop** as soon as the threshold is crossed. At steady state and relative to the resting level, the reduction is 1.1 dB for a signal at −12 dBFS, 5.5 dB at −8 dBFS, 11.6 dB at −3 dBFS and 15.7 dB at 0 dBFS: the louder the signal, the lower it comes out. This is over-compression, found through analysis, which crushes the peaks to the point of pushing them below the less loud passages.

On the other hand, **no saturation module** is engaged: no waveshaper, no bias, no opto. The diode grain, the transformer color and the Fury knob, which made the SILIKA's reputation, are not imitated; the preset reproduces the dynamic behavior, at a setting of the plugin that we don't know.

> **♪ In the studio: kush_silka**
>
> Start by raising the **Makeup** by about 3 dB to compare level-matched. Then, on drum room mics or a distorted guitar, set the **Threshold** (or the **Headroom**): past the threshold, the loudest hits are pushed below the others, and the sound gets denser, all the way to a crushed squash. If you want diode color back, add a little **Fet odd** yourself in the **WS** tab (chapter 17).

> **To be confirmed by François**
>
> The SILIKA is known for its distortions (Zener or germanium diodes, transformers, Fury knob), but **kush_silka** engages no saturation module. Does the imitation deliberately cover only the compression, leaving the color to the user?

ao_britpressor: Analog Obsession Britpressor

The target. Analog Obsession's Britpressor is a free plugin (November 2020) from the BritBundle, presented as inspired by vintage solid-state compressors, "Neve style." The exact Neve model isn't known, and neither is the gain-reduction element: linking it to the diode bridge is a hypothesis. It has a compressor and a limiter, each with its own threshold and recovery, a ratio, a gain, a mix, and a sidechain filtered by a high-pass and by boosts around 10 kHz and 1.6 kHz.

The settings.

- Main row: **Ratio** 3; **Attack** 1 ms; **Release** 400 ms.
- **GR (cv)** tab: **Env Offset** −0.2; **Opto** −1; **Opto Env** 1; **Opto precision** 0; **Opto severity** 0.
- **DEQ** tab: **Deq low herz** 220 Hz; **Deq low max** −0.2; **Deq mid herz** 440 Hz; **Deq mid max** −0.125.
- **DEQ 2** tab: **Deq low herz** 100 Hz; **Deq low max** −0.40.



Figure 154 — The ao_britpressor preset, GR (cv) tab: the Env Offset shared by all three presets in the family, and an opto designer set negative (Opto at −1), built into the envelope. The two dynamic EQs, DEQ and DEQ 2, are also modified (tabs in yellow).

Analysis. The main row is consistent with a 2254-type compressor: a ratio of 3, a 1 ms attack (the 2254/R ranges from 0.1 to 2 ms) and a 400 ms release, one of the recovery values cited for that unit. **Env Offset** at −0.2 attenuates by 1.9 dB at rest and deepens the reductions: at steady state, 6 dB instead of 4.5 dB for a detected level of −12 dBFS, and 12 dB instead of 8 dB at 0 dBFS.

The surprise is the **opto designer**, in a diode-bridge preset, and set backward. **Opto Env** at 1 builds it into the envelope; **Opto severity** at 0 makes the opto slow when the reduction is small and fast when it is large; **Opto precision** at 0 removes the rounding. **Opto** at −1 reverses the gap (chapter 15): where the opto lags behind, the gain goes the other way. With a 1 ms attack, the opto hardly lags at all during attacks. During the 400 ms releases, on the other hand, it trails behind, especially toward the end, when the reduction becomes small: the final gain then comes back **faster** than the envelope, one or two decibels ahead in a simple simulation. The release thus loses part of its tail. This is an adjustment found through analysis; it doesn't reproduce the 2254's Auto mode, which adapts its times to the program.

The dynamic EQs work on the low end. At rest, the **DEQ** removes about 1.9 dB below 220 Hz and a little less than 1 dB around 440 Hz, and the **DEQ 2** removes roughly another 4.5 dB below 100 Hz. These corrections fade as the compressor reduces the gain (chapter 18): when it works, the low end comes back, so it is reduced less than the rest of the spectrum. The Britpressor filters its sidechain; the preset, for its part, leaves the detection alone and acts on the signal you hear.

> **♪ In the studio: ao_britpressor**
>
> Try it on a bass, an electric guitar or a drum bus. Raise the **Makeup** by about 2 dB, then set the **Threshold** (still at −20 dB) for a few decibels of reduction. Listen to the bottom of the spectrum: a little lighter at rest, it regains body when the compressor clamps down, and the note endings release quickly.

The mastering room

Mastering asks something of a compressor that mixing rarely does: to stay discreet. What you look for here are moderate ratios, releases that adapt to the music, detection that follows loudness rather than every peak, and no side effect that isn't intended. The tools in this room are often digital processors or meticulously built discrete circuits: the **Weiss DS1**, which Softube ported line by line into a plugin; the **elysia alpha compressor**, fully discrete, running in class A at all times, switchable be-

tween feedback and feed-forward; or George Massenburg's **MDWDRC2**, heir to the GML 8900 dynamics controller. The family also includes two mastering plugins from Acustica Audio and TC Electronic (Ivory, DYN 3000), the second of which recreates the dynamics of Midas consoles rather than a mastering unit in the strict sense.

> **The Weiss DS1, a digital standard**
>
> Weiss's Gambit series, which includes the DS1, went into production in 1998. "DS" stands for de-essing, but most engineers use it to compress or expand. The DS1 measures both the peak **and** the RMS of its control signal, and picks a fast or slow release based on their ratio; at 44.1 and 48 kHz, the MK2 version runs internally at double the sample rate to reduce aliasing. Softube ported the original code line by line, and Daniel Weiss says he can't hear a difference from the original.

The first thing that stands out in the eight presets of this family is what they do **not** touch. None uses the opto designer, none makes saturation audible, none engages the clippers or the linear-phase FIR filters, and only one uses lookahead (**massenbug_drc2_piano**). The only waveshaper that is set, in **pa_elysia**, is not heard: it serves as a modulation source. Color is not the point; everything happens in measurement and timing.

The recurring settings confirm it (see the matrix of modified tabs at the beginning of the chapter):

- **6 of the 8 presets modify the GR (cv) tab**, which reshapes the envelope after the curve: **Env Offset** in four presets (positive for **tc_dyn3000** and **pa_elysia**, strongly negative for **massenbug_drc2_mix** and **massenbug_drc2_groove**), **Env Pow Unpow** in two (**tc_dyn3000**, **acustica_ivory**), **Gr|Env Pow** in one (**softube_ds1mk3**). According to François, the offset is the setting he uses most to imitate a compressor;
- **4 of the 8 use automatic release** (**Release Auto**), based on the signal's crest factor: three on the follower after the curve, one (**massenbug_drc2_mix**) on the detection envelope. François says he generally prefers an amount of 0.5, which he finds more natural than 1; two presets in the family are nevertheless at 1: imitation won out over preference;
- **4 of the 8 modify the DET (in) tab**: the two Weiss presets and two of the three Massenburg presets; the third, **massenbug_drc2_piano**, transforms its detection differently, with lookahead and a dynamic EQ;
- **3 of the 8 switch the curve to decibels** (**Cv Lin/Db** at 1); the other five keep the linear curve, which compresses less and less as the level rises (chapter 10), a kind of natural gentleness at high levels;
- the two Weiss presets are the only ones to put attack and release in **linear mode** (**Attack Mode** and **Release Mode** at 2).

Two surprises are worth pointing out. Two of the three Massenburg presets put most of their time constants in the **detection** rather than in the follower after the curve: their **Attack** is zero on the main row. And the constant-speed release in decibels, which corresponds to a time expressed in "decibels per second" (chapter 12), shows up in **softube_weiss**, not in the Massenburg presets, even though the MDWDRC2 expresses its times precisely in dB per second.

The curves figure at the beginning of the chapter shows the variety of the family's static curves (**pa_elysia** is not included: its automatic knee has no fixed shape). These curves don't account for the adjustments made in the GR (cv) tab, so for this family they tell only part of the story.



Figure 155 — The massenbug_drc2_groove preset, DET (in) tab: the RMS (rms secs 19 ms, rms db +3 dB) and the detection envelope (env db 0 dB) run in parallel, the peak is excluded (peak db −24 dB), and Det Mix keeps the louder of the two. Modified labels and the tabs the preset touches are shown in yellow.

softube_ds1mk3: Softube Weiss DS1-MK3

The target. Softube presents its Weiss DS1-MK3 as a line-by-line port of the hardware's code: a fully digital dynamics processor (de-esser, compressor, limiter) used in mastering studios for nearly thirty years. Detection is peak-based by default, supplemented by an RMS measurement whose integration time is set by **Average**. Its best-known feature is its **dual release**: a fast release for momentary peaks and a slow one for changes in average level, chosen according to the ratio between peak and RMS. The soft knee is adjustable, and the attack goes down to 20 µs.

The settings.

- Main row: **Ratio** 4:1; **Release** 115 ms.
- **DET (in)**: **peak pow** 1.10.
- **A/R (cv)**: **Attack Mode** 2 (linear); **Release Mode** 2 (linear); **Release Auto** 0.5.
- **GR (cv)**: **Gr|Env Pow** −0.8.

Analysis. The most telling setting is **Release Auto**. Comprendre computes the crest factor of the detection signal, that is, the ratio between its peak and its RMS, and shortens the release when this ratio is high (percussive signal) and lengthens it when it is low (sustained signal). That is exactly the principle behind the Weiss's dual release, which picks its time constant from the peak/RMS ratio. At 0.5, the time used is the average of the set time (115 ms) and the automatic time: the release stays anchored around its nominal value, but it breathes with the music.

Switching attack and release to **linear mode** has no documented equivalent at Weiss: it is an adjustment found through analysis. With the envelope computed linearly (the default, kept here), the gain then moves at a constant speed, without slowing down as it approaches its target: gain reduction sets in and lets go in a single motion, without the exponential's tail.

The other two settings reshape the curve. **peak pow** at 1.10 multiplies the detected level, expressed in decibels, by 1.1: the effective threshold rises slightly and compression gets a little steeper above it. **Gr|Env Pow** at −0.8 applies a power of 0.8 to the linear reduction, which mostly strengthens small reductions: 2 dB becomes nearly 3, while 12 dB only becomes 13.7 dB. The Weiss offers an adjustable soft knee; the preset doesn't use Comprendre's **Knee**, and it is the linear curve, rounded when seen in decibels, together with these two powers, that shapes the transition.

> **♪ In the studio: softube_ds1mk3**
>
> Try it on a full mix or a stem bus, setting the **Threshold** (left at −20 dB, its default) for 2 to 3 dB of gain reduction on the densest passages. Listen to what follows each snare hit: the automatic release should give the gain back quickly while still holding pads without pumping. If the mix breathes too much, raise **Release** before touching the ratio.

softube_weiss: Softube Weiss Compressor/Limiter

The target. The name only identifies the brand. Softube sells several plugins derived from the DS1-MK3 (the DS1-MK3 itself, which has its own preset, a maximizer, a de-esser, a multiband compressor); the most likely is the **Weiss Compressor/Limiter**, which carries over the DS1-MK3's compression and limiting algorithms as ported code. It keeps the peak/RMS comparison, with **Average** setting the integration time, three release controls (fast, slow, Average) linked by an **Auto Release Offset**, Mid/Side processing, and a linked or unlinked sidechain.

The settings.

- Main row: **Ratio** 4:1; **Attack** 28.6 ms; **Release** 494 ms; **Cv Lin/Db** 1 (decibel curve); **Channels Unlink** 1 (unlinked channels).
- **DET (in)**: **auto secs** 100 ms (200 ms by default).
- **ENV (cv)**: **EvMu Lin/Db** 1 (decibel envelope).
- **A/R (cv)**: **Attack Mode** 2 (linear); **Release Mode** 2 (linear); **Attack Auto** 0.5; **Release Auto** 1.

Analysis. This preset takes the imitation of the Weiss detection further than the previous one. **auto secs** is not a detection time: it is the time constant of the crest-factor measurement that drives the automatic times (chapter 13). Shortening it to 100 ms amounts to setting the Weiss's **Average**, the integration time of the RMS that is compared with the peak. With **Release Auto** at 1, the release is left entirely to this peak/RMS ratio, like the hardware's dual release; **Attack Auto** at 0.5 also adapts the attack, halfway.

The envelope and the curve are both in decibels, the configuration of a "conventional" compressor. Combined with **linear mode**, the decibel envelope moves in constant steps **in decibels**: 9.48 dB per set time, or about 19 dB per second on release and 330 dB per second on attack, before the automatic times come into play. That is a release expressed in decibels per second, like the MDWDRC2's Timing control; nothing in the Weiss documentation describes it, so it is a behavior found through analysis. **Channels Unlink** at 1 gives each channel its own compressor: it is the counterpart of the plugin's unlinked sidechain.

> **To be confirmed by François**
>
> Does the **softube_weiss** preset really imitate Softube's Weiss Compressor/Limiter, and not another Softube Weiss plugin (the MM-1 maximizer, for example)? The preset name only specifies the brand.

> **♪ In the studio: softube_weiss**
>
> On a mix bus or a master, set the **Threshold** for 2 to 4 dB of gain reduction and listen to the constant speed of the recovery: the gain comes back up steadily, with no final acceleration. Since the channels are unlinked, check the stereo image on a mix with a loud instrument panned to one side; if the image shifts, bring **Channels Unlink** back toward 0. For calmer, steadier compression, lower **Release Auto** toward 0.5.

tc_dyn3000: TC Electronic DYN 3000

The target. TC Electronic's DYN 3000, announced in 2020 and designed with Midas, is a dynamics plugin (compressor and gate) that the company presents as a recreation of the dynamics processing of Midas consoles, the Heritage 3000 according to TC (other sources cite the XL8 and PRO consoles). It is not a mastering unit in the strict sense: it is offered for individual sources, buses and the master. Its compressor offers four types (adaptive, corrective, creative, vintage), whose differences one review judges marginal; the gain element, detection and timing are not documented.

The settings.

- Main row: **Cv Lin/Db** 1 (decibel curve); **Release** 100 ms.
- **GR (cv)**: **Env Offset** +0.05; **Env Pow Unpow** 1.414.

Analysis. This is the most restrained preset in the family: four settings. The ratio (2:1), threshold (−20 dB) and attack (10 ms) are left at Comprendre's defaults; we don't know which DYN 3000 setting François imitated, nor which of the four types. The decibel curve gives the behavior of a conventional console compressor.

The two settings in the GR (cv) tab are adjustments found through analysis. **Env Offset** at +0.05 adds a permanent gain of about 0.4 dB at rest and lightens each reduction a little (6 dB requested becomes 5.2). **Env Pow Unpow** at 1.414 (the square root of 2) doesn't change the reduction obtained on a steady sound, only its dynamics: with the default linear envelope, the attack is slowed a little and the release sped up a little.

> **♪ In the studio: tc_dyn3000**
>
> It is a good "console" starting point for a vocal, a bass or a bus: first set the **Threshold**, then the **Ratio**, both of which the preset leaves at their defaults. Keep the offset's slight permanent gain in mind when you compare with and without the plugin: the **Out->In** auto-volume (chapter 8) lets you judge level-matched.

massenbug_drc2_mix: Massenburg DesignWorks MDWDRC2 (mix)

The target. Massenburg DesignWorks' MDWDRC2 ("massenbug" is a typo) is a digital dynamics controller designed by George Massenburg; it builds on the technology of his GML 8900 and can even emulate it. Its signature is the **Variable Exponent Averager**, a patented detector that moves continuously from average to RMS to peak; two detectors run in parallel, **MAIN** (exponent 2 by default, a true RMS) and **PEAK** (exponent 3), with the PEAK threshold set 0 to 18 dB above the MAIN threshold. A single **Timing** control, in dB per second, sets both attack and release; a **Release Override**

speeds up the recovery when the level drops. The ratio ranges from 1.2:1 to 100:1 (4:1 by default). The "mix" suffix matches none of the DRC2's factory presets: it most likely refers to a setting François chose for a mix bus.

The settings.

- Main row: **Attack** 0; **Release** 0; **Threshold** −24.1 dB.
- **DET (in)**: **rms secs** 50 ms; **rms db** −11 dB; **rms pow** 0.571; **peak db** −24 dB (peak excluded); **env db** 0 dB.
- **A/R (in)**: **Release** 254 ms; **Release Auto** 0.5.
- **GR (cv)**: **Env Offset** −0.25.

Analysis. This preset does all the work **in the detector**, which fits the DRC2's design, where the time ranges depend on the detector and its exponent. The main-row attack and release are zero: the gain instantly follows what the curve asks for, and it is the two detection measurements that carry the time constants. You'll find the two parallel detectors here: a 50 ms RMS, which corresponds to the MAIN detector at its default setting (Comprendre's RMS is a true RMS, exponent 2), and a detection envelope (default 10 ms attack, 254 ms release) that plays the role of the PEAK detector. **Det Mix**, left on *max*, keeps the louder of the two at every instant.

The levels and the power set the balance between these two detectors. The RMS is lowered by 11 dB, but **rms pow** at 0.571 multiplies its level in decibels by 0.571, which pulls it toward 0 dBFS and "softens" its contribution (chapter 13). In practice, the detection envelope leads most of the time, and the RMS only takes over on quiet or very dense passages. This power is not the DRC2's exponent, which chooses the kind of averaging: it is another way, found through analysis, to set the sensitivity of one of the two detectors. The detection envelope's **Release Auto** at 0.5 shortens its release on transients, in the spirit of the Release Override.

That leaves **Env Offset** at −0.25, François's favorite adjustment, pushed hard here. At rest, it attenuates the signal by 2.5 dB permanently; during gain reduction, it weighs more and more heavily. With the linear curve at 2:1, the gain the curve asks for never drops below 0.5 (6 dB); after the offset, the reduction can approach 12 dB. The offset thus turns a very gentle curve into noticeably firmer compression, without touching the ratio.

> ♪ **In the studio: massenbug_drc2_mix**
>
> Put it on a mix bus and set the **Threshold** first: on an already loud mix, the preset can exceed 8 dB of gain reduction. Make up the 2.5 dB permanently lost with the **Makeup** before comparing. Above all, listen to how the quiet passages are held without the drum attacks being squashed: that's the interplay of the two detectors.

massenbug_drc2_groove: Massenburg DesignWorks MDWDRC2 (groove)

The target. Same plugin as the previous preset (see **massenbug_drc2_mix**). The "groove" suffix doesn't match a factory preset either; the DRC2 guide talks about adapting the detector exponents to the "groove" of a track. It is probably François's own setting for a rhythm track or a drum bus.

The settings.

- Main row: **Ratio** 3.68:1; **Threshold** −14.7 dB; **Attack** 0; **Release** 40 ms.
- **DET (in)**: **rms secs** 19 ms; **rms db** +3 dB; **peak db** −24 dB (peak excluded); **env db** 0 dB.

- **A/R (in)**: **Attack** 20 µs; **Release** 400 ms; **Release Shape** 1.35; **Rel Ret From** 0.13.
- **A/R (cv)**: **Release Shape** 2.09; **Rel Ret From** 0.
- **GR (cv)**: **Env Offset** −0.3.

Analysis. The structure is that of the **mix** preset, only snappier, and it is the one shown in the family's figure. The 19 ms RMS plays the MAIN detector; its +3 dB level is the value chapter 13 recommends to bring a sine wave's RMS up to its peak level, which puts the two detectors on an equal footing. The detection envelope, with its 20 µs attack, catches virtually every peak, then holds it and releases it over 0.4 s: that's the PEAK detector. Its **Release Shape** of 1.35 holds back the start of the descent a little before letting it go.

The follower after the curve has zero attack and a short release (40 ms), but with a very pronounced S shape: **Release Shape** at 2.09 with **Rel Ret From** at 0, meaning that each release restarts from the beginning of its curve, where it is flat. When hits come in quick succession, the gain recovery is strongly held back, then released all at once when the playing opens up (chapter 12). Nothing like this is documented for the DRC2, whose timing is a single control in dB per second: this is the shape the analysis revealed.

The **Env Offset** of −0.3 is the strongest in the family: −3.1 dB at rest, and a reduction that deepens quickly. With detection at 0 dBFS, the curve alone would ask for 7.8 dB; with the offset, you get 19.5 dB. It is this offset, far more than the ratio, that gives the preset its firmness.

> **♪ In the studio: massenbug_drc2_groove**
>
> Try it on a drum bus or a percussion loop. First set the **Threshold** so the gain reduction stays reasonable, then listen to the gain recovery between hits: it "glues" to the strong beats and lets go when the playing breathes. For a more open groove, raise **Release** slightly or move **Rel Ret From** (A/R (cv) tab) closer to 1.

massenbug_drc2_piano: Massenburg DesignWorks MDWDRC2 (piano)

The target. The MDWDRC2 again. "Piano" is neither a mode nor a factory preset; the official preset set covers piano with an engineer's setting ("JohnJ_Apno"), and Massenburg's video tutorials cover piano, among other topics. The DRC2 offers a look-ahead of a few dozen samples (up to 63) and a BS. 1770 filter in its sidechain.

The settings.

- Main row: **Ratio** 42 (infinite); **Threshold** −18.1 dB; **Knee** 8.05; **Attack** 1.2 ms; **Release** 303 ms; **Channels Unlink** 1.
- **INPUT**: second **FX deq** 1 (DEQ 2 acts on the detection).
- **OUTPUT**: second **deq** 0 (DEQ 2 is not heard).
- **A/R (cv)**: **Attack Shape** 3.4; **Release Auto** 1.
- **MOD & LA**: **la delay** −44.4 ms; **det->la.det** 1; **la det pk** 1; **env->la.env** 0.
- **A/R (la)**: **Attack** 13.1 ms.
- **DEQ 2**: **Deq low herz** 281 Hz; **Deq low max** 8; **Deq low reso** 0.35.

Analysis. This preset no longer imitates a compressor but a **lookahead limiter**, and it's a free interpretation rather than a copy. The ratio is infinite; the curve is left linear, so the **Knee** of 8.05 is not a width in decibels but in amplitude (8.05/120, about 0.067): the knee spans roughly −20.8 to −16 dBFS around the threshold.

The lookahead follows the "detector" school (chapter 22): the present chain is delayed by 44 ms, and it is the future detection signal, held at its maximum over the whole window (**la det pk** at 1), that enters the compressor. The detected level therefore rises 44 ms before each attack and only falls back when it would have without lookahead; the 1.2 ms attack, S-shaped thanks to **Attack Shape** at 3.4, has plenty of time to put the reduction in place before the note arrives. That's far more than the DRC2's look-ahead, which is counted in samples. The 303 ms release is fully automatic (**Release Auto** at 1): shorter after a percussive chord, longer on a sustained resonance.

DEQ 2, sent to the detection and not to the sound, turns the measurement into a **low-end** detector: at rest, the low band adds about 18 dB below 60 Hz, 12 dB at 200 Hz and 3 dB at 500 Hz, with a low resonance (0.35) that makes the slope start well below 281 Hz. So the limiter responds first to the left hand; high notes have to be much louder to trigger it. Because the band's coefficient follows the compressor's envelope (**Deq low min** is left at 0), this emphasis decreases as the limiter works. The DRC2 offers a BS.1770 filter in its sidechain; the preset doesn't try to reproduce it.

One setting has no effect: the 13 ms **Attack** in the A/R (la) tab sets the dedicated follower of the "envelope" school, which isn't even computed when **env->la.env** is 0.

> **To be confirmed by François**
>
> In **massenbug_drc2_piano**, the **Attack** in the A/R (la) tab is 13 ms, but **env->la.env** is at 0: the lookahead's dedicated follower isn't computed and this setting has no effect. Is it a leftover from exploration, or was some of the "envelope" school meant to be kept?

> **♪ In the studio: massenbug_drc2_piano**
>
> On a piano recorded in stereo, set the **Threshold** so that only the loudest left-hand passages are limited. Listen to the attack of the low chords: thanks to the lookahead, it should stay intact while the level is held. If the stereo image seems to wander, bring **Channels Unlink** back toward 0. François points out that it is often good to clip a little before limiting: a touch of **Pre Clip** (chapter 7) can complement this preset.

pa_elysia: elysia alpha compressor (Plugin Alliance)

The target. The name points to elysia, distributed by Plugin Alliance; the most likely product is the **alpha compressor**, the brand's first unit, introduced in 2006 as a fully discrete mastering compressor (the mpressor is the other candidate). Its knee is soft, and its actual ratio increases with gain reduction: the scale is calibrated for 6 dB of reduction, and at 17 dB, the actual ratio is 4.5:1 for a displayed 2.5:1. It switches between feedback, where the ratio tops out at 2.5:1, and feed-forward, which goes all the way to limiting and negative ratios; an "Auto Fast" function shortens the attack on strong impulses, and a Soft Clip rounds off the peaks.

The settings.

- Main row: **Threshold** −27.0 dB; **Ratio** 0.7; **Attack** 3.2 ms; **Release** 184 ms; **Cv Lin/Db** 1 (decibel curve).

- **CURVE**: **Knee Auto** 24; **mod ratio** +1.75.
- **ENV (cv)**: **EvMu Lin/Db** 1 (decibel envelope).
- **GR (cv)**: **Env Offset** +0.173.
- **MOD & LA**: **ws mod** −1; **mod att secs** 3.0 ms; **mod rel secs** 142 ms.
- **WS**: **Fet Thresh** 0.11; **Fet Knee** 32.8.
- **OUTPUT**: **ws** 0 (the waveshaper is not heard).

Analysis. This is the most ingenious preset in the family, and it matches the alpha's documented behavior point for point. The displayed ratio, 0.7, never acts on its own: with **GR Min** at 0 dB, a ratio below 1 has no effect (chapter 10). It serves as the anchor point for ratio modulation (chapter 23). At rest, the ratio is 0.7 + 1.75 = **2.45:1**, very close to the alpha's 2.5:1 in feedback mode, which argues for this model. Then the slope varies linearly with the modulation value: the further it rises above 1, the higher the ratio climbs, up to infinity when it reaches 1.4.

The source of this modulation is the **detection waveshaper**, set up as a simple clipper (threshold 0.11, about −19 dBFS, with a wide knee). Its amount follows the gain reduction: as soon as the compressor works, the clipper shaves the detection, **ws mod** at −1 turns this loss of amplitude into a modulation value above 1, and the ratio rises. The more the compressor reduces the gain and the louder the signal, the higher the actual ratio: that's exactly the alpha's "ratio that rises with gain reduction." The smoothing (3 ms rising, 142 ms falling) turns this modulation into a peak follower. The waveshaper is never heard, since its **ws** amount is at 0 in the OUTPUT tab.

The **Knee Auto** of 24, on a decibel curve, widens the knee by 12 dB per decibel of smoothed reduction: the moment the compressor starts working, the knee becomes very wide and the compression very gradual. This is the answer to the alpha's soft knee. Finally, the **Env Offset** of +0.173 gives a permanent gain of about 1.4 dB at rest and markedly lightens each reduction (6 dB becomes 3.4); the modulation compensates by raising the ratio. With this offset, the envelope slightly exceeds 1 at rest: the waveshaper's saturation amount then becomes slightly negative (chapter 18), which lowers the ratio a little in passages that barely compress. Neither the "Auto Fast," the Soft Clip, nor the gain element's harmonics are carried over.

> **♪ In the studio: pa_elysia**
>
> On a master or a stem bus, start from 1 to 2 dB of gain reduction by setting the **Threshold**, then push it: the compression should get firmer as it deepens, without ever hardening suddenly. To feel what the modulation does, temporarily set **mod amp** (MOD & LA tab) to 0: the ratio stays frozen at 2.45:1. Compensate for the offset's permanent gain so you can compare level-matched.

acustica_ivory: Acustica Audio Ivory

The target. Ivory is a mastering suite from Acustica Audio (Ivory3, then Ivory 5); we don't know which version or which module François imitated. Acustica doesn't name the sampled hardware; Ivory3 contains a single-band compressor-expander with a clipper (MAC-2) and its multiband version, and the Ivory 5 limiter is presented as an interpretation of a famous British mastering processor. Like the company's other plugins, Ivory relies on dynamic convolution (Volterra kernels); the gain element, ratios and times are not documented.

The settings.

- Main row: **Threshold** −24 dB; **Ratio** 3:1; **Attack** 30 ms; **Release** 800 ms.

- **GR (cv)**: **Env Pow Unpow** 7.

Analysis. On the main row, nothing but classic mastering fare: a slow attack that lets transients through, a long release, a moderate ratio. The curve is left linear: seen in decibels, it compresses less and less as the level rises, to the point where the local ratio is only about 1.1:1 at 0 dBFS. That's a natural gentleness at high levels, without any knee setting.

All the character lies in **Env Pow Unpow** at 7, a very high value found through analysis. This setting doesn't change the reduction obtained on a steady sound, but it does change its dynamics (chapter 15): the follower smooths a reduction stretched seven times over, then scales it back. The attack becomes much more gradual, almost a straight line in decibels: out of a 6 dB reduction, only 1.2 dB is done by the end of the 30 ms attack (3.3 dB without this setting). The release, on the other hand, is sped up considerably: within 80 ms, more than half the reduction has already been given back, whereas the default setting would have given back only 0.8 dB; only the last decibel takes the set time. A gentle attack and a fast, then lingering, recovery: on paper, a way to thicken a mix without holding it down for long.

> **♪ In the studio: acustica_ivory**
>
> Try it on a full mix, with 2 to 3 dB of gain reduction. Listen to the drum attacks, which should come through intact, then to the way the mix recovers its level between phrases. If the recovery seems too fast, lower **Env Pow Unpow** toward 3 or 4 rather than lengthening **Release**: you'll get back a more classic envelope.

Digital inventions

The plugins in this family imitate no hardware. No optical cell, no transistor, no tube: their gain-reduction element is a multiplication, and all their character comes down to computational choices. Their developers often document these choices precisely, and they are exactly the "schools" that Comprendre makes adjustable. TDR Kotelnikov combines peak and RMS detection, each with its own release; u-he Presswerk offers three topologies, feed-forward, feedback and an "INT" mode that moves from one to the other depending on the crest factor; Voxengo Polysquasher runs an array of more than ten compressors in feedback; sonible smart:comp 2 sets its parameters by learning; Playfair Dynamic Grading replaces threshold and ratio with a histogram of levels. To read these presets, keep chapter 13 (detection), chapter 14 (feed-forward and feedback), chapter 10 (the curve) and chapters 11 and 12 (the envelope, attack and release) close at hand.

What the thirteen presets have in common can be read on the preset map. All of them adjust the main row, and all of them change the **Release**. **Ten out of thirteen** switch the transfer curve to decibels (**Cv Lin/Db** at 1), compared with barely more than half across the whole collection: no surprise for compressors born in the digital world, which think in decibels. Above all, this is the family of **detection**: five presets modify the DET (in) tab. Three measure an RMS (**tdr_kotelnikov**, **uhe_presswerk**, **voxengo_polysquasher3**); three call on the detection envelope, placed before the curve (**tdr_kotelnikov**, **melda_mturbo**, **tonelib_multicomp**), and the last two even hand it all the smoothing, with **Attack** and **Release** at 0: that's the other school from chapter 9, the one that smooths the level rather than the gain. Next come the attack and release modes (4 presets), the knee

(4), **Env Offset** (4, always negative), automatic times (3) and feedback (2: **waves_rvox** fully, **caelumçsmmov** halfway). The curves figure at the beginning of the chapter shows twelve of these curves; **toneproject_unisum**, with its automatic knee, is not included.

> **A word from the author**
>
> In the world of compressors, François explains, there are often two schools: for the envelope follower's law, branching or decoupled, as well as for its placement, before or after the transfer curve. He offers all of them, including the ones he likes less, because many compressors, some of them from major players, use them. For his part, he prefers smoothing placed after the curve.

There's no shortage of surprises. No preset in the family uses the waveshaper, the opto designer, lookahead or the bias, even though Presswerk, Marquis, Manic, MTurboComp, smart:comp 2 and Unisum all document a coloration stage (saturation, triodes, Drive, "Style," the "HYGGE" circuit): these presets imitate their targets' dynamics, not their color. **boz_manic** is the shortest of the 80 presets, with two settings; **toneproject_unisum** alone touches the main row and six tabs. **voxengo_polysquasher3** departs from its target's documentation on two points (topology and detection), and its **Env Offset** produces a startling overcompression. Finally, three presets contain settings that have no effect in version 1.0.3: **Mu Min Rel** without a Mu envelope (**caelumçsmmov**), **Gr|Env Pow** at its neutral value (**melda_mturbo**), and bias settings while the bias is off (**ircam_purecomp**).

waves_rvox: Waves Renaissance Vox

The target. Waves' Renaissance Vox (or R-Vox) existed by 2003 at the latest. It copies no hardware: it brings together, in three controls, the technology of the Renaissance Compressor, itself derived from the C1 and the L1, with an in-house automatic release (ARC). A single fader, Compression, sets both the threshold and the ratio of a soft-knee curve; a gentle expander acts before the compression, and a peak limiter at the output follows the Gain setting. No time is adjustable: according to Waves, compression, gating, limiting and maximizing are all fully automatic. The plugin is intended for vocals, voice-overs and podcasts.

The settings.

- Main row: **Threshold** −16 dB; **Ratio** 42 (infinite); **Knee** 88 dB; **Release** 100 ms; **Cv Lin/Db** 1 (decibel curve);
- INPUT tab: **FF->FB** 1 (pure feedback);
- ENV (cv) tab: **EvMu Lin/Db** 1 (decibel envelope);
- A/R (cv) tab: **Attack Auto** 1 and **Release Auto** 1 (fully automatic times); **Attack Mode** 0.94 and **Release Mode** 0.94.

Analysis. The soft knee documented by Waves is pushed to the extreme here. Centered on −16 dB, an 88 dB **Knee** starts at −60 dBFS, exactly the floor below which the decibel curve no longer distinguishes levels (chapter 10), and would only end at +28 dBFS. Across the whole useful range, the curve is therefore nothing but knee: the slope moves gradually from 1:1 toward infinity without ever reaching it, and the ratio of 42 only serves to set the curvature. Pure feedback softens the whole thing further, since the detector measures an already compressed signal (chapter 14). In steady state, on a constant level, the math gives about 2 dB of gain reduction at −40 dBFS, 6 dB at −20 dBFS and 13 dB at 0 dBFS: compression that starts very low and grows steadily, with an effec-

tive ratio that stays between 1.3:1 and 1.5:1. With a knee this wide, the **Threshold** moves the whole parabola: lowering it increases both the gain reduction and the local ratio, a bit like R-Vox's single fader combines threshold and ratio.

The lack of adjustable times translates into **Attack Auto** and **Release Auto** at 1: the times become 2 × T / C², where C² is the square of the signal's crest factor (chapter 12). The attack (10 ms by default) and the release (100 ms) get shorter when the voice turns percussive and longer when it settles. This is Comprendre's academic method, not Waves' ARC, which is documented only for the Renaissance Compressor. The modes at 0.94 pull the envelope very slightly toward the extremes (chapter 12), and the decibel envelope attacks large reductions a little more decisively (chapter 11): two fine adjustments found through analysis. R-Vox's expander and output limiter are not reproduced: the gate stays inactive (**Ratio Gate** at 1) and no clipper is engaged.

> **♪ In the studio: waves_rvox**
>
> Try it on spoken voice (podcast, voice-over) or on a sung vocal at mixdown. Listen to how even the level is from one syllable to the next, and to what happens between phrases: since gain reduction starts as low as −60 dBFS, breaths and room noise may come up once the **Makeup** is set. Set the **Threshold** first; it controls both the amount and the firmness of the compression. If the breaths get in the way, add a gentle gate (**Ratio Gate** 2, **Tresh Gate** just above the noise, chapter 10): that's the job of the expander R-Vox places before its compression.

tdr_kotelnikov: TDR Kotelnikov

The target. Tokyo Dawn Records' Kotelnikov came out in December 2014, as a free version and a paid "Gentleman's Edition"; the compressor's core is shared by both. Its manual puts it bluntly: it does not try to emulate any device; it is a proud digital processor. It descends from the TDR Feedback Compressor, from which it inherits its control scheme and separate releases for peak and RMS. Its detection is twofold: a peak path follows the waveform, and an RMS path responds to energy and ignores transients. Kotelnikov's threshold is the RMS threshold, and a "Peak Crest" control places the peak threshold above it, with a documented sweet spot between 3 and 8 dB. The developer recommends a peak release of 25 to 100 ms and an RMS release of more than 150 ms; the ratio ranges from 1.1:1 to 7:1, with an adjustable soft knee. Intended use: the stereo bus.

The settings.

- Main row: **Knee** 27.16 dB; **Release** 100 ms; **Cv Lin/Db** 1 (decibel curve);
- DET (in) tab: **rms secs** 21.2 ms; **rms db** −1 dB; **peak db** −1 dB; **env db** −6 dB;
- A/R (in) tab: **Release** 200 ms.

Analysis. Of the whole family, this is the most faithful transcription of a documented design. Three detectors work in parallel, and **Det Mix**, left on *max*, keeps the loudest at every instant (chapter 13):

- the **peak**, at −1 dB, follows the waveform; the gain it asks for is smoothed by the follower after the curve, with the default attack (10 ms) and the 100 ms **Release**: this is the fast path;
- the **detection envelope**, at −6 dB, rises with the peaks in 10 ms (the A/R (in) tab's default attack) and lets go of them with a 200 ms release: it sustains a slow level, set 5 dB below the peak;
- a 21 ms **RMS**, at −1 dB, sits between the two and smooths out the ripple of the low end.

Both releases fall squarely within the ranges the developer recommends: 100 ms for the fast path (25 to 100 ms advised for the peak), 200 ms for the slow path (more than 150 ms for the RMS). The 5 dB gap between the peak and the slow envelope recalls the role of “Peak Crest,” which sets the gap between the two thresholds. It makes sense for the slow path to be an envelope rather than an RMS: Comprendre’s RMS has the same speed rising and falling, whereas the detection envelope rises quickly and releases slowly, with its own time (chapter 13).

The soft knee becomes a 27.16 dB **Knee** on the decibel curve, centered on −20 dB: it spans −33.6 to −6.4 dBFS, a wide bus-compressor knee. The ratio stays at 2:1, its default value, within Kotelnikov’s range. Finally, the preset stays in feed-forward: the “Feedback Compressor” lineage says nothing about Kotelnikov’s topology, and **FF->FB** is left untouched.



Figure 156 — The tdr_kotelnikov preset, DET (in) tab: the peak and the 21 ms RMS at −1 dB, and the detection envelope at −6 dB, combined by Det Mix on max. As in the plugin, modified settings and the tabs involved (DET (in), A/R (in)) are shown in yellow.

> **♪ In the studio: tdr_kotelnikov**
>
> Its natural home is the stereo bus: a mix, a drum bus, a group of backing vocals. Listen to how the transients get through (the peak only steps in when it exceeds the other detectors) while the body of the sound is held by the slow path. Set the **Threshold** first. To rein in the transients more, raise **peak db**; for more or less glue, adjust the **Release** in the A/R (in) tab, between 150 and 400 ms. With **Gfx Detects** at 1 (GRAPHS tab), you can see which detector is calling the shots.

uhe_presswerk: u-he Presswerk

The target. Presswerk, from u-he, came out in December 2014. It’s an original design: u-he assembled its own dynamics modules, with an M/S matrix and a saturation stage, then reworked the core after studying many hardware compressors. Its detection is an RMS with a window from 0.1 to 30 ms: at the minimum, it approaches peak detection; the developer advises 1 to 5 ms, and more than 10 ms to imitate the lag of an optical compressor. It offers three topologies: FF, “predictable” and direct, the only mode where the ratios are exact; FB, more “musical,” whose ratio tops out around 2.5:1; and INT, which blends the two according to the crest factor. Its “Adapt” control widens the release around its nominal value: percussive signals shorten it, soft passages lengthen it.

The settings.

- Main row: **Release** 250 ms; **Cv Lin/Db** 1 (decibel curve);
- MAIN tab: **Headroom** 3 dB;
- DET (in) tab: **rms secs** 10 ms; **rms db** 0 dB; **peak db** −24 dB (peak excluded);
- ENV (cv) tab: **Env->Mu** 1; **Mu Min Att** 0.5; **Mu Max Att** 0; **Mu Min Rel** 0.5; **Mu Max Rel** 0;

- A/R (cv) tab: **Attack Auto** 0.5; **Release Auto** 0.5;
- GR (cv) tab: **Gr|Env Pow** 0.75; **Env Offset** −0.1.

Analysis. Detection is a pure 10 ms RMS, with the peak excluded: that's Presswerk's principle, with a duration inside its documented range. Comprendre's **rms secs**, however, is the time constant of an exponential average, not a window (chapter 13): we're comparing orders of magnitude, not two identical definitions. The preset stays in feed-forward (**FF->FB** at 0): it imitates Presswerk's FF mode, neither FB nor INT.

The heart of the preset is how it renders "Adapt." It combines two mechanisms:

- **Attack Auto** and **Release Auto** at 0.5, the amount François considers most natural (chapter 12): the times get shorter on a percussive signal, with a high crest factor, and longer on a steady signal, exactly in the direction documented for Adapt;
- the **Mu envelope** (**Env->Mu** at 1), whose bounds go from 0.5 in silence to 0 at 0 dBFS: the louder the detected level, the shorter the times (chapter 11). With the 250 ms **Release**, and leaving the automatic times aside, the release lasts about 125 ms on a very quiet signal, 100 ms for a detected level of −20 dB and around thirty milliseconds at −6 dB; the attack (10 ms by default) goes from 5 ms to almost nothing. As long as the music stays loud, the compression releases quickly between hits; when it calms down, the release gets longer.

The curve keeps its default threshold (−20 dB) and ratio (2:1), computed in decibels, and the 3 dB **Headroom** makes the compressor see a signal 3 dB louder: the threshold behaves like a −23 dB threshold (chapter 7). Two adjustments in the GR (cv) tab, with no documented equivalent, modify the amount of gain reduction (chapter 15). **Gr|Env Pow** at 0.75 scales it down to three quarters, so the 2:1 behaves like a ratio of about 1.6:1. **Env Offset** at −0.1 adds a permanent 0.9 dB attenuation and weighs a little more as the reduction deepens. All told, in steady state, the gain reduction is about 2 dB at −20 dBFS, 6.5 dB at −10 dBFS and 11.4 dB at 0 dBFS. Presswerk's saturation stage is not carried over.



Figure 157 — The uhe_presswerk preset, ENV (cv) tab: the Mu envelope, whose attack and release times get shorter as the detected level rises (Min bounds at 0.5, Max bounds at 0). Modified settings and the tabs involved are shown in yellow.

> **♪ In the studio: uhe_presswerk**
>
> u-he designed Presswerk for drum groups, bass, guitars and vocals: all good ground for this preset. Listen to the release speed, which changes with the level: fast between hits in a loud passage, slower when the music calms down. The brown curve in the history display shows this relationship live. Set the **Threshold** first. To get the gentleness of Presswerk's FB mode, try **FF->FB** between 0.5 and 1, then lower the threshold to get back the same amount of compression (chapter 14).

voxengo_marquis: Voxengo Marquis Compressor

The target. Voxengo's Marquis Compressor has been around since 2005 (version 2 in 2016). It's a "universal" compressor, built around a cascade of amplifiers modeled on the triode; one of its three algorithms, T3, is inspired by a classic analog compressor that Voxengo doesn't name. It offers an in-house "Round" detector and a "Normal" detector, runs in feed-forward or, with the "Feedbk" switch, in "classic feedback (opto)" mode, and its Drive ranges from gentle coloration to brutal tube saturation.

The settings.

- Main row: **Ratio** 4; **Attack** 50 ms; **Release** 200 ms; **Cv Lin/Db** 1 (decibel curve);
- CURVE tab: **Gr Amp** 1.3;
- ENV (cv) tab: **EvMu Lin/Db** 1 (decibel envelope).

Analysis. The basis is that of a conventional compressor: curve and envelope in decibels, default threshold (−20 dB), 4:1 ratio, a moderate attack that lets transients through, a 200 ms release. The preset stays in feed-forward, Marquis's default position, and keeps Comprendre's peak detection.

The unusual setting is **Gr Amp** at 1.3. It multiplies the linear reduction at the moment the gain is applied (chapter 10): 1 dB of envelope reduction becomes 1.3 dB, 3 dB becomes 4.2 dB, 6 dB becomes 9.1 dB and 10 dB becomes 19 dB. So the compression digs in faster and faster the deeper it goes. Voxengo documents nothing of the kind: it's an adjustment found through analysis. It has a limit worth knowing about: beyond 12.7 dB of envelope reduction, the applied gain drops to zero and then turns negative, which inverts the polarity of the target (see the "To be confirmed" box on Gr Amp in chapter 10). With this ratio and threshold, that would take a detected level staying above roughly −3 dBFS. The triode stage and the Drive are not reproduced.

> **♪ In the studio: voxengo_marquis**
>
> Voxengo recommends it for vocals, drum buses and individual tracks. With a 50 ms attack, transients get through before the compression sets in; then listen to how it firms up as the gain reduction increases. Set the **Threshold** first while watching the white envelope in the history display: keep the gain reduction under ten decibels or so, well away from the zone where the gain drops to zero.

voxengo_polysquasher3: Voxengo Polysquasher 3

The target. Voxengo's Polysquasher is now at version 3, released in February 2017 with a redesigned compression algorithm. It emulates no hardware: Voxengo describes it as a feedback compression algorithm that uses an array ("poly") of compressors, usually more than ten. It uses peak detection. It has no release time: an "Inertia" parameter sets how slow the release is, which depends on the program material and the selected mode, and switches select the attack and release shapes. Its goal is transparency: Voxengo places it at the end of the mastering chain, just before the limiter.

The settings.

- Main row: **Ratio** 4; **Attack** 97 ms; **Release** 114 ms (linear curve, the default);
- DET (in) tab: **rms secs** 6.2 ms; **rms db** 0 dB; **peak db** −24 dB (peak excluded);
- ENV (cv) tab: **EvMu Br/Cp** 1 (decoupled envelope);
- A/R (cv) tab: **Release Auto** 1;

- GR (cv) tab: **Env Offset** −0.303.

Analysis. Two documented traits are indeed present. The program-dependent release becomes **Release Auto** at 1: the time follows the crest factor entirely, minus, in decoupled mode, the automatic attack time (chapter 12). The “inertia” that holds the compression back finds an echo in the decoupled envelope: after an attack, the release waits a little before starting again, all the longer as the attack is long (chapter 11), and here it is long, nearly 100 ms.

Two other traits run counter to the documentation. Polysquasher is a feedback compressor with peak detection; the preset stays in feed-forward, with a short RMS detection (6.2 ms) from which the peak is excluded. For the smoothness that feedback brings, it relies on other means: the RMS and the decoupled envelope.

The most surprising setting is **Env Offset** at −0.303, the largest offset in the family. At rest, it permanently attenuates the signal by 3.1 dB. During compression, it weighs much more heavily (chapter 15): in steady state, with a constant detected level, the output **drops** as the level rises above the threshold, from about −23 dBFS at the threshold to −33 dBFS for a detected level of 0 dBFS. This is an overcompression effect, comparable to that of a negative ratio. If the envelope dropped below 0.303 (10.4 dB of reduction), the target would even fall silent; with this threshold and ratio, that would take a detected level of about +3 dBFS, which only happens with a high **Headroom** or sidechain level.

> **To be confirmed by François**
>
> With **Env Offset** at −0.303 on a linear curve (threshold −20 dB, ratio 4), the voxengo_polysquasher3 preset attenuates the signal by 3.1 dB at rest, and its steady-state output decreases when the detected level exceeds the threshold (about −23 dBFS at the threshold, −33 dBFS at 0 dBFS). Does this overcompression match what you observed on Polysquasher, or should the offset value be revisited?

> **♪ In the studio: voxengo_polysquasher3**
>
> As is, this preset is nothing like a transparent finishing compressor: it crushes loud passages. Try it as an effect, on a drum bus or room mics, as parallel compression with the main-row **Dry/Wet**, and listen to the density it adds. Set the **Threshold** first: the higher it is, the less the signal enters the zone where the output drops. To get closer to the transparency Voxengo is after, bring **Env Offset** back toward 0 and compensate with the **Makeup**.

sonible_smartcomp2: sonible smart:comp 2

The target. sonible’s smart:comp 2 came out in July 2022: it’s the major update of the company’s “intelligent” compressor. During a learning phase, with profiles for instruments, buses and mixes, it sets threshold, ratio, attack and release by itself, then engages spectral compression that analyzes the signal across more than 2,000 bands. With spectral compression at 0, it goes back to being an ordinary broadband compressor. Its knee ranges from soft to hard; an automatic release and a Style control, from “clean” to “dirty,” round out the package.

The settings.

- Main row: **Attack** 25 ms; **Release** 150 ms; **Cv Lin/Db** 1 (decibel curve);
- ENV (cv) tab: **EvMu Lin/Db** 1 (decibel envelope).

Analysis. This is the most classic preset in the collection: curve and envelope in decibels, threshold (−20 dB) and ratio (2:1) left at their defaults. You'll recognize the "conventional compressor" from chapter 10, with times of 25 and 150 ms. It imitates only the broadband part of the plugin: spectral compression over thousands of bands has no equivalent in a broadband compressor, and Comprendre's dynamic EQs could only approximate a few of those bands. Since smart:comp 2 chooses its parameters by learning, the preset freezes one of its results, obtained on a source we don't know. The plugin's automatic release is not carried over (**Release Auto** stays at 0), and neither is the Style control.

> ♪ **In the studio: sonible_smartcomp2**
>
> A neutral starting point for a vocal, a guitar or a bus: nothing colors the sound, and everything is set from the main row. Set the **Threshold** first, then the **Ratio**. To get back some of the plugin's automatic adaptation, raise **Release Auto** to 0.5 (chapter 12) and listen to the release follow the character of the passage.

ircam_purecomp: FLUX:: Pure Compressor (likely)

The target. The name probably refers to Pure Compressor from FLUX:: (a brand now part of Harman): no IRCAM-branded "PureComp" plugin was found, and the prefix may be explained by the close ties between FLUX:: and IRCAM, whose tools FLUX:: distributes. Pure Compressor is an original digital design that promises transparent, discreet compression, with the option of classic "fat" compression. It offers eight detection modes, including "Feed Backward" modes inspired by vintage hardware; attack, hold and release, with manual, automatic or advanced release; zero attack thanks to a delay line; and a "Dynamics Ratio" that adjusts the ratio in real time according to the signal's dynamics.

The settings.

- Main row: **Ratio** 9.35; **Attack** 23 ms; **Release** 530 ms; **Cv Lin/Db** 1 (decibel curve);
- MAIN tab: **Headroom** 2.5 dB;
- CURVE tab: **GR Min** −6 dB;
- A/R (cv) tab: **Attack Mode** 2 and **Release Mode** 2 (linear modes); **Release Shape** 0.8; **Rel Ret Spd** 2;
- BIAS src tab: **Bias out unlink** 0;
- DET (bs) tab: **Det Mix** avg; **peak db** −24 dB; **env db** 0 dB.

Analysis. The key to this preset is **GR Min** at −6 dB. This setting caps the curve's gain at −6 dB: whatever the level, even below the threshold and even in silence, the compressor takes off at least 6 dB (chapter 10). It's a permanent gain reduction, which the history display shows as the dark red GR Min line. With a 9.35:1 ratio, a −20 dB threshold and a 2.5 dB **Headroom** (the compressor sees the signal 2.5 dB louder), the curve only asks for 6 dB from an input level of about −16 dBFS upward. Below that, the compressor behaves like a fixed 6 dB attenuator; above it, the 9.35:1 compression takes over from that floor. The first six decibels of compression are thus replaced by a constant attenuation, and the gain only really moves on loud passages; the release, for its part, never comes back up above −6 dB. FLUX:: documents nothing like this; it's an adjustment found through analysis. Remember to make up these 6 dB, with the **Makeup** or the **Output Db**, to compare level-matched.

The times are handled in linear mode (chapter 12): the envelope falls and rises in constant steps, giving an attack that doesn't slow down as it approaches its target and a straight-line release. **Release Shape** at 0.8 makes the release start a little faster and finish more slowly, and **Rel Ret Spd** at 2 makes the position on this curve return twice as fast during attacks. These are shape adjustments found through analysis; Pure Compressor documents several release modes without publishing their curves.

The four settings in the BIAS src and DET (bs) tabs have no effect: none of the five amounts that wake up the bias is set (chapter 19). They would only take effect if you turned the bias on. Finally, the preset keeps peak detection and feed-forward: Pure Compressor's "Feed Backward" modes are not imitated.

> ♪ **In the studio: ircam_purecomp**
>
> With its high ratio and long release, this preset keeps a firm grip on a vocal, a bass or a bus that needs to stay in place. Start by making up the 6 dB of permanent reduction, then set the **Threshold**: it decides the level at which the compression goes beyond this fixed step. To compare with a compressor that leaves quiet passages untouched, briefly set **GR Min** back to 0 dB.

boz_manic: Boz Digital Labs Manic Compressor

The target. Boz Digital Labs' Manic Compressor (release date not found; first customer review in November 2015) is an original design that brings together six unnamed compressor models. It is designed for parallel compression, with wet, dry and master faders. A "Beef" switch (Thin, Neutral, Thick) changes its weight; "Loud Relief" eases off the compression on transients when the gain reduction gets heavy; a Drive adds saturation, and two EQs act, one on the sidechain, the other on the tone.

The settings.

- Main row: **Ratio** 4; **Release** 60 ms.

Analysis. Just two settings: this is the shortest of the 80 presets. Everything else is default Comprendre: linear curve, −20 dB threshold, 10 ms attack, peak detection, linear branching envelope. The preset's character therefore comes from the linear curve, which compresses less and less as the level rises above the threshold (chapter 10): with a ratio of 4, it takes off about 4 dB at −14 dBFS, 6 dB at −10 dBFS and 9.8 dB at 0 dBFS, where a decibel curve would take off 15. This static behavior is not "Loud Relief," which acts over time, on transients, but it goes in the same direction: the loudest peaks get squashed less. The fairly fast 60 ms release lets the gain come back up between hits. Nothing else from Manic is carried over: no parallel compression, no "Beef," no Drive, no EQs, and we don't know which of the six models served as the reference.

> ♪ **In the studio: boz_manic**
>
> Manic is built for parallel compression: on drums, try the main-row **Dry/Wet** between 0.3 and 0.5, and listen to the body the compression adds beneath the hits, which stay intact. Set the **Threshold** first. To capture the spirit of the Drive, the waveshaper (chapter 17) is at your disposal.

caelumçsmmov: Caelum Audio Smoov (likely)

The target. The file name is garbled. The researchers' hypothesis is Smoov, from Caelum Audio, released in February 2021: "ç" would have been typed instead of "_" (keys 9 and 8 are neighbors on an AZERTY keyboard), and "smmov" instead of "smoov"; it's the only Caelum Audio product close to this name. Smoov is an original design: "vintage-style" program-dependent compression, whose attack and release are determined automatically by peak detection. It aims to be gentle and transparent. The plugin also offers expansion and gate ratios, a knee, auto gain, stereo linking and sidechain filters.

The settings.

- Main row: **Ratio** 4; **Attack** 6.8 ms; **Release** 1.5 s; **Cv Lin/Db** 1 (decibel curve);
- MAIN tab: **Headroom** 3 dB;
- INPUT tab: **FF->FB** 0.5;
- ENV (cv) tab: **Mu Min Rel** 6.907;
- A/R (cv) tab: **Attack Mode** 2 and **Release Mode** 2 (linear modes).

Analysis. The "vintage style" comes from the topology: **FF->FB** at 0.5 makes the detector measure an equal mix of the incoming signal and the previous output (chapter 14). The compressor sits halfway between feed-forward and feedback, gentler than pure feed-forward. The 3 dB **Headroom** makes the threshold behave like a −23 dB threshold.

The program-dependent times are approximated by a different means than Smoov's: the linear modes (chapter 12). With constant steps, the duration of an attack or release becomes proportional to the distance to be covered. With a **Release** of 1.5 s, the gain rises by 63.2% of full scale in 1.5 s: it recovers from 1 dB of reduction in about a quarter of a second, from 3 dB in 0.7 s, from 6 dB in 1.2 s. Small reductions let go quickly, large ones slowly: the release depends on what the music asked for. Likewise, the 6.8 ms attack reaches 6 dB of reduction in just over 5 ms.

Mu Min Rel at 6.907, on the other hand, has no effect: **Env->Mu** stays at 0, so the Mu envelope isn't computed (chapter 11).

> **To be confirmed by François**
>
> Does the caelumçsmmov preset really target Caelum Audio Smoov ("ç" typed instead of "_", "smmov" for "smoov")? Also, it sets **Mu Min Rel** to 6.907 but leaves **Env->Mu** at 0: since the Mu envelope isn't computed, this setting has no effect in version 1.0.3. Should **Env->Mu** have been turned on?

> **♪ In the studio: caelumçsmmov**
>
> Smoov bills itself as gentle, transparent compression: try this preset on a vocal, an acoustic guitar or a mix bus. Listen to the recovery after a loud passage: slow when the gain reduction was deep, almost immediate when it was light. Set the **Threshold** first. For a livelier character, shorten the **Release**: in linear mode, all release durations shrink in the same proportion.

melda_mturbo: MeldaProduction MTurboComp

The target. MeldaProduction's MTurboComp (2017) imitates not one machine but fourteen: its generic engine (four level followers, four processors, a dynamic EQ, two saturators) reproduces classic compressors and limiters learned through machine learning, plus a few in-house devices. The first model it learned was the 1176, and the manual only details the "Meldei 1176" device. In Edit mode, each level follower can be linear, squared, logarithmic or direct. The device imitated by this preset cannot be identified.

The settings.

- Main row: **Ratio** 4; **Attack** 0; **Release** 0; **Cv Lin/Db** 1 (decibel curve);
- DET (in) tab: **peak db** −24 dB (peak excluded); **env db** 0 dB;
- A/R (in) tab: **Release** 100 ms;
- GR (cv) tab: **Gr|Env Pow** 1; **Env Offset** −0.045.

Analysis. With **Attack** and **Release** at 0, the follower after the curve does no smoothing: the gain follows the curve moment by moment. All the smoothing is handed to the detection envelope, alone in the mix, with a 10 ms attack (the A/R (in) tab's default) and a 100 ms release. This is exactly the "Smoothing the level before the curve" workflow from chapter 13, and the other school from chapter 9: the gain follows the shape of the decibel curve applied to the smoothed level. The release then rises at a constant speed in decibels, stops dead as soon as the level falls back below the threshold, and its duration depends on how far the threshold was exceeded. This choice fits MTurboComp's architecture, built around "level followers." The 4:1 ratio is compatible with the "Meldei 1176" device, since the 1176 offers a 4:1 ratio, but there's no way to be sure.

In the GR (cv) tab, **Gr|Env Pow** at 1 is a neutral value with no effect (chapter 15). **Env Offset** at −0.045 adds a permanent 0.4 dB attenuation and very slightly deepens the reductions; it's what keeps the follower after the curve in service, with zero times, hence without smoothing.

> **♪ In the studio: melda_mturbo**
>
> Melda recommends its "Meldei 1176" for vocals, guitars, synths and drums: good testing grounds. Listen to the release, which lasts longer the further the signal went over the threshold, then stops dead. Set the **Threshold** first, then the **Release** in the A/R (in) tab, which alone governs the recovery. To compare the two schools, put a little **Release** back on the main row: the smoothing is then shared between the two followers.

tonelib_multicomp: ToneLib TL MultiComp

The target. ToneLib's TL MultiComp was announced on July 2, 2024. It combines two units, only one of which runs at a time: "Studio," with manual controls, and "EasyComp," where a single sensitivity knob chooses the attack, ratio, release and knee. The Studio unit offers a threshold from −60 to +10 dB, a knee from 0 (hard) to 72 dB, an attack from 0.01 to 250 ms and a release from 10 ms to 2.5 s. No hardware is cited. The product is recent, but the identification is consistent with the preset itself: it's one of the fifteen presets that contain all the settings of version 1.0.3, so it was saved with the latest version of the plugin.

The settings.

- Main row: **Threshold** −14 dB; **Ratio** 4; **Attack** 0; **Release** 0; **Cv Lin/Db** 1 (decibel curve);
- DET (in) tab: **Det Mix** env (the detection envelope alone); **peak db** −24 dB; **env db** 0 dB; **env pow** 0.707;
- ENV (in) tab: **EvMu Lin/Db** 1;
- A/R (in) tab: **Release** 250 ms;
- GR (cv) tab: **Env Offset** −0.075.

Analysis. This preset follows the same school as **melda_mturbo**, even more strictly: **Det Mix** keeps only the detection envelope, and **Attack** and **Release** at 0 let the gain follow the curve with no further smoothing. The level is smoothed before the curve, with a 10 ms attack (the A/R (in) tab's default) and a 250 ms release, values within the Studio unit's ranges. This smoothing is computed in decibels (**EvMu Lin/Db** in the ENV (in) tab); in this follower, the decibel computation applies to the complement of the level (chapter 11).

The **env pow** power of 0.707 changes how the threshold and ratio should be read. It multiplies the level in decibels by 0.707 before the curve (chapter 13): according to that chapter's formulas, compression starts around −19.8 dBFS rather than at −14, and the 4:1 ratio behaves like a ratio of about 2.1:1. So the displayed pair, −14 dB and 4:1, is in practice a compressor of about −20 dB and 2.1:1, with a hard knee. **Env Offset** at −0.075 adds a permanent 0.7 dB attenuation.

> **♪ In the studio: tonelib_multicomp**
>
> ToneLib intends TL MultiComp for mixing as well as mastering. Try this preset on a bus or a mix: the compression, smoothed before the curve, follows the level without jolts, and the 250 ms release lets passages breathe. Set the **Threshold** first, keeping in mind that compression starts about 6 dB lower than the displayed value. For softer or firmer compression, adjust **env pow** (DET (in) tab): lower, the compression starts earlier and becomes more gradual.

toneproject_unisum: Tone Projects Unisum

The target. Tone Projects' Unisum came out in April 2020; it's the company's only compressor. Tone Projects presents it as a mastering compressor based on a "novel topology": compression happens in a single band, but detection is split into three bands, each with an RMS detector and a peak detector in parallel. The knee is variable, with "Linearity" and "GR Limit" controls; the release curve goes continuously from linear to logarithmic; a "Memory Effect" makes attack and release program-dependent. The sidechain has its own high-pass (5 to 1,000 Hz), low-pass and "emphasis/dip" filters. A "HYGGE" circuit adds transformer- and tube-style warmth. Uses: mastering and mix bus.

The settings.

- Main row: **Knee** 36 (on a linear curve: an amplitude width of 0.3); **Attack** 25 ms; **Release** 350 ms;
- INPUT tab: **FX High Cut** 85 Hz; second **FX deq** 1 (DEQ 2 in the detection);
- CURVE tab: **Knee Auto** −72;
- A/R (cv) tab: **Release Mode** 1.5;
- GR (cv) tab: **Gr|Env Pow** −0.6888;
- DEQ 2 tab: **Deq mid herz** 3,502 Hz; **Deq mid max** 5.771;

• OUTPUT tab: second **deq** 0 (DEQ 2 inaudible on the target).

Analysis. This is the richest preset in the family, and it works first on what the compressor hears. **FX High Cut** at 85 Hz removes the low end from the detection (chapter 14), like Unisum's sidechain high-pass. DEQ 2 acts only in the detection: its mid band, centered on 3.5 kHz, adds about 14 dB at rest to the measured signal at the center of the band (coefficient 5.771, default resonance of 0.707), and this emphasis decreases as compression increases, down to about 10 dB for 6 dB of gain reduction (chapter 18). So the detector hears the presence region much louder than the rest, as with a sidechain "emphasis" filter, but with an emphasis that itself depends on the compression. The preset doesn't reproduce Unisum's three detection bands; it makes the detection frequency-selective.

Release Mode at 1.5 blends logarithmic and linear release in equal parts (chapter 12): this is the most direct match for Unisum's release curve, adjustable "from linear to logarithmic." **Gr|Env Pow** at −0.6888 mostly strengthens small reductions (chapter 15): 1 dB becomes 2.1 dB, 6 dB becomes 8.4 dB. Compression thus sets in more decisively just above the threshold. Unisum shapes its curve with its "Linearity" and "GR Limit" controls; the preset gets its own shape through this adjustment found through analysis.

The knee deserves special attention. On a linear curve (the preset keeps **Cv Lin/Db** at 0), **Knee** 36 is an amplitude width, and **Knee Auto** at −72 **narrows** it as the smoothed reduction increases: from about 0.8 dB on, the knee is fully closed and the curve returns to a hard knee. So the knee is hard in loud passages and wide in quiet ones, when the compressor isn't working. Now, with a −20 dB threshold, the linear knee starts at silence as soon as **Knee** exceeds 24: this preset is directly affected by the knee "shaving" described in chapter 10, and by the "To be confirmed" box that goes with it. As long as the knee stays open, the curve pushes detected levels below about −53 dBFS down toward the **GR Max** floor, and already takes off 3 to 5 dB between −40 and −20 dBFS. With peak detection, the detected value passes near zero on every half-cycle, which can accentuate the effect. In quiet passages, the preset therefore behaves partly like a downward expander.

> **♪ In the studio: toneproject_unisum**
>
> Unisum is a mastering and mix-bus compressor: try this preset on a full mix. Listen to how the compression reacts to the presence range (vocals, snare, guitars, around 3.5 kHz) rather than to the low end, and keep an eye on quiet passages, intros and song endings, which can be noticeably pulled down. Set the **Threshold** first. If quiet passages collapse, try a **Knee** below 24, or switch **Cv Lin/Db** to 1 for a knee in decibels, readjusting the Knee accordingly (chapter 10).

playfair_grading: Playfair Audio Dynamic Grading

The target. Playfair Audio's Dynamic Grading (Christian Luther, in Hanover) came out in May 2022, and version 2.0 was announced in February 2025. It's an original design: instead of threshold, ratio and attack, the plugin displays a histogram of loudness levels, on which you place three zones, "punch," "body" and "floor," to compress or expand them. The measurement goes through a tilt filter ("Spectrum"), to measure "the way we hear," and relies on a short-term power average rather than on the peak; an automatic look-ahead, equal to the delay of this average, replaces the attack, except in "Live" mode. The release is designed to compress and expand at the same time. Uses: tracks and buses in mixing, film dialogue, podcasts, mastering.

The settings.

- Main row: **Threshold** −15 dB; **Ratio** 1.28; **Knee** 2.16 dB; **Attack** 14 ms; **Release** 7.3 ms; **Cv Lin/Db** 1 (decibel curve);
- CURVE tab: **Ratio Gate** 42; **Knee Gate** 8.05 dB.

Analysis. Comprendre has no histogram, but its transfer curve is the product of two curves: the compressor above its threshold and the gate below its own (chapter 10). The preset uses the two as two zones. Above −15 dB, very gentle compression (1.28:1, with a small 2.16 dB knee): the "body" zone, barely tightened. Below the gate threshold, left at its default of −20 dB, a slope of 42: the "floor" zone, expanded downward all the way to closing. The gate's knee, 8.05 dB centered on −20 dB, starts at −16 dBFS, right where the compressor's knee begins: the two zones touch. On a decibel curve, the gate quickly reaches the −60 dB floor, which then means complete silence (chapter 10): in steady state, the gain reduction is about 10 dB at −18 dBFS and 40 dB at −20 dBFS, and the target is silent below roughly −21 dBFS.

The times are unusual: the attack (14 ms) is twice as long as the release (7.3 ms). For the gate, this makes sense: it closes at the attack speed and reopens at the release speed (chapter 10), so it opens quickly and closes a little more slowly. As in Dynamic Grading, a single pair of times serves both compression and expansion. On the other hand, the level measurement remains Comprendre's default, the instantaneous peak: no power average, no tilt filter, no look-ahead. With peak detection, the detected value passes near zero on every half-cycle, and the gate then asks to close; the envelope smooths these requests, but on low frequencies, a little RMS (**rms secs** of 10 to 30 ms, chapter 13) would bring the measurement closer to the plugin's.

> **♪ In the studio: playfair_grading**
>
> Dynamic Grading is aimed in particular at dialogue and podcasts: this preset cleans up what happens between phrases and barely tightens the voice. Before anything else, set the **Threshold**, and above all the **Tresh Gate** (CURVE tab), for your source: as is, the preset mutes everything detected below about −21 dBFS, which on many recordings includes the voice itself. Set the gate threshold just above the noise floor, then listen to the ends of words: if they get cut off, soften the gate with a lower **Ratio Gate** or a wider **Knee Gate**.

Color without compression: saturators, transformers, clippers

Comprendre is not just a compressor. It contains three saturators (the opto designer, the waveshaper, the bias), two dynamic EQs and three clippers, and most of these modules keep working when the compressor itself is doing nothing. The seven presets in this family imitate plugins that are not, or not only, compressors: a tube processor that adds harmonics, a line amp, a transformer saturator, a preamp module, a sound "fattener," a tape and console emulation, a clipper. They show how François turned Comprendre into a color machine.

The clearest move is **neutralizing the compression**: **Ratio** at 1 and **Threshold** at 0 dB. With a ratio of 1, the transfer curve isn't even computed and its gain is 1 (chapter 10). The compressor never reduces the gain, the envelope stays at 1, and everything that depends on gain reduction shuts off. What remains active is decisive, and it's what each preset exploits:

- the **dynamic EQs** take their **max** coefficient, the one at rest: they become fixed EQs (chapter 18);

- the **waveshaper** falls silent at its default setting, because its amount follows the gain reduction; it only saturates permanently if **Fet env** is engaged (static waveshaping, chapter 17);
- the **bias** stays active as soon as one of its five amounts is anything other than 0: its fixed height (**Bias x base**), **Bias Fm On** and its limiter work without compression, while **Bias x gr** and the "gain reduction" source stay at zero (chapters 19 to 21);
- the **opto designer**, which follows the envelope, has nothing left to follow.

Four of the seven presets neutralize the compression this way: **pa_hg2**, **pulsarmodular_p42**, **kazrog_true-iron** and **dafalife_sausage**. In the curves figure at the beginning of the chapter, they merge with the diagonal: their color doesn't go through the transfer curve. The following table sums up what acts in each of the seven.

Preset	Compression	What acts all the time	What waits for gain reduction
pa_hg2	neutral	static waveshaper, DEQ 2 (treble)	nothing
pulsarmodular_p42	neutral	bias limiter, Bias Fm On	nothing
kazrog_true-iron	neutral	DEQ and DEQ 2 (fixed EQs)	nothing
dafalife_sausage	neutral	DEQ and DEQ 2 (fixed EQs)	nothing
kiive_filk2_saturation	2:1 above −23 dB, detection shifted by the bias	bias limiter (ceiling at −12 dBFS)	waveshaper
acqua_taupe	2:1 in dB above −25.3 dB	(no saturator)	(no saturator)
acqua_ash-ultra	infinite and instantaneous: a clipper	(the clipping itself)	bias, whose source is the gain reduction

Counting the modules turns up several surprises. Two presets target saturators (**kazrog_true-iron**, **dafalife_sausage**) yet engage **no** saturator at all: they keep only the plugin's frequency fingerprint. **acqua_taupe**, filed here for its magnetic tape, contains no color module: it's a compressor. **acqua_ash-ultra** imitates a clipper without using any of Comprendre's three clippers: it turns the compressor itself into a clipper, then reshapes it with the bias. Only **kiive_filk2_saturation** ties the saturation to the compression, as the waveshaper does by default. Finally, no preset in the family uses the opto designer; the waveshaper is used in two presets, the bias in three, the dynamic EQs in three.

> **A word from the author**
>
> In his video on bias fusion, François notes that just by playing with the bias settings alone, without even using the limiter-compressor at its destination, which is yet another kind of saturator, you already get a wide variety of saturations. In his French walkthrough, he distinguishes about ten types.

Two tips apply to the whole family. First, in the presets where compression is neutral, the **Headroom** (MAIN tab, chapter 7) becomes a real **drive knob**: it drives the signal harder into the waveshaper, the dynamic EQs and the bias limiter, then takes that gain back off afterward, without

changing anything else. Second, like all the factory presets, these are saved without oversampling (1X). For overt saturation, try 2X or 4X: oversampling reduces the aliasing produced by nonlinear stages (chapter 7).



Figure 158 — The pa_hg2 preset, WS tab: Fet env at 1 makes the waveshaping static (saturation no longer depends on compression), and Fet odd at 0.907 and the clipper with its knee (Fet Thresh 0.22, Fet Knee 27.92) shape the curve. Ratio 1 and Threshold 0 dB neutralize the compression. Modified labels and the tabs the preset touches are shown in yellow.

pa_hg2: Black Box Analog Design HG-2

The target. Black Box Analog Design's HG-2, emulated by Brainworx for Plugin Alliance, is a stereo tube processor that is **not a compressor**: it's a harmonics and saturation generator. It has four emulated tube stages, a 6U8A pentode and a triode in series between modeled transformers, plus a parallel saturation circuit based on a 12AX7. According to the manual, the pentode produces even harmonics and the triode the 3rd harmonic, which is "grittier." It offers a saturation frequency selector, an **Air** control above 10 kHz, Dark/Norm/Bright calibration, a Density control and a mix control; the manual speaks of "saturation-based compression" for mix buses, vocals, piano and full mixes.

The settings.

- Main row: **Ratio** 1; **Threshold** 0 dB.
- **WS**: **Fet env** 1; **Fet odd** 0.91; **Fet Thresh** 0.22; **Fet Knee** 27.9.
- **DEQ 2**: **Deq hgh herz** 3,036 Hz; **Deq hgh max** 0.12; **Deq hgh zdf->svf** 0 (one-pole filter).

Analysis. The compression is neutral, and yet the waveshaper works all the time. **Fet gr** is left at 1; with **Fet env** at 1, the saturation amount is 1 × (1 − e) + 1 × e = 1, whatever the envelope: this is **static waveshaping**. François made dynamic waveshaping, which follows the gain reduction, the default behavior, because a real compressor saturates more when it compresses; he also provided the other option, and this preset uses it to make the waveshaper a saturator in its own right.

The curve chains two stages. **Fet odd** at 0.91, a little below 1, is a slightly expansive shape: low levels lose about 0.8 dB. Then comes the clipper: threshold at 0.22 and a wide knee (27.9/120, or about 0.23 in amplitude). The rounding starts around 0.10 (−20 dBFS), and the ceiling, 0.22 (about −13 dBFS), is reached around 0.34. In practice, a signal that stays below −20 dBFS passes almost intact; above that, its peaks get rounded off, and **no peak comes out above roughly −13 dBFS**. It is indeed "saturation-based compression," but at a level that assumes a moderate input. All the engaged stages are symmetrical: the preset produces only **odd** harmonics, the triode side of the HG-2; the pentode's even harmonics, which **Fet even** could provide, are not carried over.

DEQ 2, placed after the waveshaper, is a fixed EQ (without compression, its coefficient stays at **max**): a one-pole high shelf that adds about 0.5 dB at 3 kHz and approaches 1 dB above 10 kHz. It gives a little brightness back to the saturated signal, much like the HG-2's **Air** control or Bright calibration, though we don't know which plugin setting François imitated.

> **♪ In the studio: pa_hg2**
>
> Try it on a mix bus, a vocal or a piano. The main move is the **Headroom** in the MAIN tab: at −6 or −12 dB, the signal hits the waveshaper less hard and the saturation softens; at 0 dB, a mix at a typical level gets heavily shaved. Bring the level back up with the **Makeup**, then listen to the saturation alone with **Monitor Fet** to judge the grain.

pulsarmodular_p42: Pulsar Modular P42 Climax

The target. Pulsar Modular's P42 Climax (a different company from Pulsar Audio), released in 2021, is **not a compressor** but a line amp and saturator. The company presents it as a tribute to the first DI box, the Wolfbox and its Triad transformer, "a Motown-flavored line amp" able to evoke several line amps (Neve 1080, Studer A812, SSL Fusion) without copying any of them. Its saturation, at the input or output and adjustable from 0 to 150%, produces "odd and even harmonics, mostly odd"; a class A stage adds even harmonics, and an input control drives the transformer into saturation. A brickwall limiter was added later.

The settings.

- Main row: **Ratio** 1; **Threshold** 0 dB.
- **BIAS dst**: **Bias x base** 0.000019; **Bias on pre->out** 1 ("out"); **Bias Fm On** 1.53; **Bias on limit** 0.097; **Bias on ratio** 3.91.
- **F (bs)**: **bias amp** 0.475.

Analysis. This preset is a little gem of repurposing: it uses the bias without compression, and almost without bias. The value of **Bias x base**, 0.000019, is so small that the height it adds stays below −90 dBFS. Its real job is to **switch the module on**: as soon as one of the five amounts is anything other than 0, the whole bias wakes up, inverse path, **Bias Fm On** and limiter included (chapter 19).

What does the work then is the destination. **Bias Fm On** at 1.53 boosts the signal by about 3.7 dB just before the bias limiter: it's a drive, the equivalent of the P42's input control. The limiter, set to 0.097 (about −20 dBFS) with a ratio of 3.91 and no attack or release, acts instantly on the waveform: it's really a saturation curve with a sharp bend. The "out" position places it after the compressor's gain, which does nothing here. Since the height is practically zero, the normal and inverse paths are symmetrical: the recombination keeps only **odd** harmonics, which matches the P42's "mostly odd." Finally, **bias amp** at 0.475 sets the saturation product to 47.5%, like an amount knob.

The result is easy to calculate. Below about −24 dBFS, the signal comes out intact, but 1.9 dB louder: direct fusion (the default) keeps the gain of **Bias Fm On**, which **bias amp** lets through only in part. Above that, peaks are progressively held in check: a peak at 0 dBFS comes out around −2.6 dBFS. The signal gains 4 to 5 dB of density, with an odd-harmonic grain. Neither the class A stage with its even harmonics nor the P42's brickwall limiter is carried over.

> **♪ In the studio: pulsarmodular_p42**
>
> On a bass track, a vocal or a bus, compare level-matched: the preset raises quiet passages by nearly 2 dB, and the ear easily mistakes that gain for color. Then set the amount with **bias amp** (F (bs) tab) and the drive with **Bias Fm On** or the **Headroom**. **Mon bias sat** (BIAS dst tab) lets you hear the grain on its own.

kazrog_true-iron: Kazrog True Iron

The target. Kazrog's True Iron (2018) is **not a compressor** but a transformer saturator. It originally modeled two line transformers (UTC 108X and Malotki 4001B), then six transformers taken from Ampex, Urei, Neve, Rohde & Schwarz, Siemens, Neumann and Telefunken units. Its controls set saturation (Crush), nonlinearity (Strength), input impedance, even harmonics (Morph) and the mix; depending on the transformer chosen, the harmonic profile ranges from mostly even to a 3rd harmonic present right from the start.

The settings.

- Main row: **Ratio** 1; **Threshold** 0 dB.
- **MAIN**: **Phase** unchecked (inverted polarity).
- **DEQ**: **Deq low herz** 50 Hz; **Deq low max** −0.25; **Deq hgh herz** 6,888 Hz; **Deq hgh max** 0.27; **Deq hgh zdf->svf** 0 (one-pole filter).
- **DEQ 2**: **Deq low herz** 60 Hz; **Deq low max** −0.5; **Deq low zdf->svf** 0; **Deq hgh herz** 9,000 Hz; **Deq hgh max** −1; **Deq hgh zdf->svf** 0.

Analysis. First surprise: a preset that imitates a saturator engages **no** saturator at all. The waveshaper and the bias are at neutral; only the two dynamic EQs act, made fixed by the absence of compression. Both are needed, because each has only one low band and one high band, and the preset stacks four corrections. Only one is a little subtle: DEQ 2's high band, at −1 with a one-pole filter, removes the band **exactly** (chapter 18); the signal minus its one-pole high-pass is its one-pole low-pass at 9 kHz.

Computed at 48 kHz, the overall response is that of a bandwidth limited at both ends: −0.9 dB at 80 Hz, −2.4 dB at 50 Hz and −5 dB at 30 Hz in the low end; a region flat within ±0.3 dB from 80 Hz to 5 kHz; then, despite the slight lift from the DEQ, −1.5 dB at 9 kHz, −3.4 dB at 12 kHz and −7 dB at 16 kHz. This is the frequency fingerprint the analysis captured from the plugin. True Iron's harmonics (Crush, Morph) are not reproduced; the preset's only nonlinear element is the dynamic EQs' safety clipping, at an internal 0 dBFS, which only affects peaks that reach it.

That leaves **Phase** unchecked: the output is inverted. Chapter 7 gives it one main use: the analysis method of chapter 24, where you invert Comprendre and sum it with the other plugin to hear their difference. On its own, a polarity inversion usually goes unnoticed; it matters as soon as the processed signal is mixed with the original outside Comprendre.

> **To be confirmed by François**
>
> The **kazrog_true-iron** preset is saved with **Phase** unchecked, that is, with inverted polarity, like **ao_dbcomp** and **ssl_1_b**. Is this intentional (the imitated plugin's polarity) or a leftover from the analysis session, where this button is used to hear the difference?

> **♪ In the studio: kazrog_true-iron**
>
> Use it as a subtle touch of "iron" on a bass, a kick or a bus: it tightens the extreme lows and softens the extreme highs. If you use it on a parallel bus in your DAW, check **Phase** again; otherwise the blend with the original track will sound hollow. To add some transformer grain, try a little static waveshaping (**Fet odd** around 1.2 with **Fet gr** and **Fet env** at 1, chapter 17).

kiive_filk2_saturation: Kiive Audio FilkChannel Strip MK2

The target. The name probably refers to Kiive Audio's FilkChannel Strip MK2, a channel strip "based on some of the best Danner Cassette Modules from the 70's." It has seven reorderable modules: two preamps (Pre 1, with a smooth sound or aggressive harmonics; Pre 2, "German radio engineering," with two saturation modes), three EQs and two compressors, one opto-style, the other VCA-style. The "saturation" suffix suggests François imitated one of the preamps rather than a compressor.

The settings.

- Main row: **Threshold** −23 dB; **Attack** 25 ms; **Release** 50 ms (the **Ratio** stays at 2:1, on a linear curve).
- **WS**: **Fet drive** +4.5 dB; **Fet even** −1; **Fet odd** 8; **Fet Thresh** 0.6.
- **BIAS src**: **Bias input add** 5.
- **BIAS dst**: **Bias on limit** 0.25.

Analysis. This is the only preset in the family where saturation depends on compression: **Fet gr** is left at 1 and **Fet env** at 0, so the waveshaper only saturates in proportion to the gain reduction. Here the compressor acts as a saturation pedal. The curve is extreme: **Fet odd** at 8 gives a very steep saturation (low levels are amplified eight times before being flattened), **Fet even** at −1 doubles the fundamental and adds a 2nd harmonic of opposite polarity, and the +4.5 dB drive concentrates the shaping on low levels and brings the final clipping down from 0.6 to about 0.36. This is "aggressive harmonics" territory.

The bias does two things. **Bias input add** at 5 adds to the detection a height on the order of five times the output level (the bias source is the output, the default): it far exceeds the signal itself, and the compression is then driven mostly by the output, as in a feedback loop (chapter 20). So the compressor works well below the displayed −23 dB threshold, and the waveshaper is almost always engaged. The bias limiter, for its part, is a hard clip at 0.25, placed after the waveshaper and before the compressor's gain: it acts all the time, with or without compression, and **no peak comes out above −12 dBFS**.

Finally, there's a less obvious consequence. Since the bias is active with its inverse path (the default), the output is the "push-pull" recombination of the two paths, which cancels whatever is identical on both sides (chapter 19). The even part of the waveshaper's curve, 2nd harmonic included, is therefore largely removed; what remains of **Fet even** is mostly its effect on the shape of the odd curve.

To be confirmed by François

In **kiive_filk2_saturation**, the bias's inverse path (active by default) recombines the two paths by their difference: the even part of the waveshaper, including the 2nd harmonic set by **Fet even** at −1, largely cancels out, except insofar as the two paths compress differently. Was this even harmonic meant to be heard (in which case **Bias +inverse** at 0 would bring it back)?

♪ In the studio: kiive_filk2_saturation

Save it for sounds that can take an overt grain: drums, bass, guitar, synth. Since the compressor drives the saturation, the **Threshold** sets both the compression and the amount of grain; raise it for a cleaner sound. To hit the bias limiter less hard, lower the **Headroom**, then lower the **Threshold** by the same amount to keep the same compression. Finally, bring the level back up with the **Makeup**.

dafalife_sausage: Dada Life Sausage Fattener

The target. The Sausage Fattener (2011) by the duo Dada Life, programmed by Tailored Noise, is a plugin version of the processing chain the duo used in Ableton Live. It is classified as a saturator and has only three controls: **Fatness** (saturation, up to +30 dB), **Color** (a brighter tone) and **Gain**. The processing is not disclosed; one reviewer suspects a multiband compressor, saturation and a limiter or clipper, and describes Color as a resonant distortion EQ.

The settings.

- Main row: **Ratio** 1; **Threshold** 0 dB.
- **DEQ**: **Deq low herz** 80 Hz; **Deq low max** −0.9; **Deq low zdf->svf** 0 (one-pole filter); **Deq mid herz** 70 Hz; **Deq mid max** 0.025.
- **DEQ 2**: **Deq low herz** 80 Hz; **Deq low max** −0.45; **Deq mid herz** 70 Hz.

Analysis. Like **kazrog_true-iron**, this preset engages no saturator: only the dynamic EQs act, fixed for lack of compression. And what they do is surprising for a "fattener": they **thin out the extreme low end**. The DEQ's low band, at −0.9 with a one-pole filter, removes 90% of the low end below 80 Hz; DEQ 2's low band, at −0.45 with the default two-pole filter, takes off up to another 5 dB in the extreme low end. Together they give −1.2 dB at 100 Hz, −2.4 dB at 80 Hz, −6.7 dB at 50 Hz and −16 dB at 20 Hz, with no change above 200 Hz.

This is the frequency fingerprint the analysis captured from the plugin, at a Fatness and Color setting we don't know; neither the Sausage Fattener's saturation nor its compression is reproduced. Two settings have no audible effect: the DEQ's mid band, with a coefficient of 0.025 and the default resonance, only raises 70 Hz by about 0.15 dB, and DEQ 2's mid band has a frequency but no coefficient, which leaves it inactive. These are probably traces of exploration.

♪ In the studio: dafalife_sausage

On a synth bass, an electronic kick or an electronic music bus, it cleans up the sub-bass that eats up headroom without adding body. To get closer to the plugin's idea, add some density: static saturation with the waveshaper, or the **pulsarmodular_p42** recipe (bias limiter and **Bias Fm On**). Compare level-matched: a lighter low end often sounds quieter.

acqua_taupe: Acustica Audio Taupe

The target. Acustica Audio's Taupe (reviewed in 2019) brings together two worlds. On one side, sampled tape machines (Studer, MCI, Otari, Ampex, Revox), with tape compression that goes up to 6 dB of "instant limiting." On the other, a dynamics module "derived from a top-quality Swiss analog console manufactured in 1982," which the company doesn't name: compressor and limiter, an "SH Mod" control that changes the slope of the compression onset, and two switchable compressors, recommended for bass, drums, piano and spoken voice. The preset name doesn't say which of the two François imitated.

The settings.

- Main row: **Threshold** −25.3 dB; **Attack** 29 ms; **Release** 99 ms; **Cv Lin/Db** 1 (decibel curve).
- **A/R (cv)**: **Attack Auto** 1.
- **MOD & LA**: **la delay** −4 ms.

Analysis. The family's second surprise: no color module. It's a compressor in its own right, at 2:1 (the default ratio) on a decibel curve, which tips the identification toward the dynamics module rather than the tape saturation.

Its attack is fully automatic: with **Attack Auto** at 1, it equals twice the set **Attack** divided by the square of the crest factor (chapter 12). On a percussive signal, it becomes very short; on a sustained sound, it stretches to about 58 ms. The 4 ms lookahead, in the default "envelope" school, makes each reduction start 4 ms earlier without bringing the release forward (chapter 22). Together, these two settings catch sharp transients early and let sustained sounds breathe, which recalls the tape's "instant limiting"; yet nothing tells us whether François imitated the tape or the compressor's "SH Mod" control. It's an adjustment found through analysis.

> **♪ In the studio: acqua_taupe**
>
> Try it on the sources the company recommends: bass, drums, piano, spoken voice. Set the **Threshold** for a few decibels of gain reduction and listen to the note attacks: the sharpest ones get caught earlier than the others. The lookahead adds 4 ms of latency, which your DAW normally compensates for.

acqua_ash-ultra: Acustica Audio Ash Ultra

The target. Ash Ultra is a module from Acustica Audio's Ash suite, which is **not a compressor** but a clipper: "a unique collection of classic clipping circuits" taken from unnamed high-end converters. The suite offers five families of clipping circuits and a brickwall limiter; its Flat variant emphasizes aliasing rejection.

The settings.

- Main row: **Threshold** 0 dB; **Ratio** 42 (infinite); **Attack** 1 µs; **Release** 1 µs; **Channels Unlink** 1.
- **MAIN**: **Headroom** +12 dB.
- **BIAS src**: **Bias input add** 0.41; **Bias g.e->out** 0 (the bias follows the gain reduction).
- **BIAS dst**: **Bias x base** −0.044; **Bias x gr** −4.68; **Bias Delay Secs** 14.5 ms; **Bias Delay FB** 0.19.

Analysis. Comprendre has three clippers, and this preset uses none of them: it builds the clipper out of the compressor itself. An infinite ratio pins the output to the threshold; attack and release times of 1 µs, shorter than one sample period, let the gain follow every sample; detection, still on in-

stantaneous peak, measures each sample as is. The gain is then practically the threshold divided by the level as soon as the signal exceeds the threshold: that's **hard clipping**. The +12 dB **Headroom** drives the signal in 12 dB hotter: clipping starts at −12 dBFS of the input signal, and since that gain is removed afterward, output peaks stay around −12 dBFS. **Channels Unlink** at 1 clips each channel on its own, like the clippers' *Unlink* mode.

Why go through the compressor? Because the bias can then reshape the clipping. Its source is the gain reduction: as long as the signal stays below the threshold, the bias is zero and the preset is transparent. As soon as it clips, the input height (0.41 times the bias) shifts what the normal and inverse paths measure in opposite directions, so they no longer clip their half-cycles the same way; the output height, negative and larger the heavier the gain reduction (**Bias x gr** at −4.68), shifts the signal of both paths further still. The recombination keeps a symmetrical shape, but it's no longer quite hard clipping: the detail of the bend changes, and that detail is what distinguishes one clipping circuit from another. These values are adjustments found through analysis, for one of Ash Ultra's circuits, we don't know which. Finally, what the bias adds goes through a 14.5 ms delay with low feedback: a light comb filter, with teeth spaced about 69 Hz apart and notches of less than 3 dB, which colors only this product (chapter 16).

> **♪ In the studio: acqua_ash-ultra**
>
> Put it at the end of the chain on a drum bus or a master, and make up the 12 dB with the **Makeup** (or lower the **Headroom** to clip less). Raise the **Oversampling** to 2X or 4X: instantaneous clipping produces a lot of aliasing. To hear the bias's touch, compare with its amounts at 0, or listen to **Mon bias sat** with gain; and to compare with a classic clipper, try the **Pre Clip** in *Adaa 1* mode (chapter 7).

CHAPTER 26

Starting Points

More than 360 settings is an immense freedom, and sometimes a question: where do you start? This last chapter suggests ways in, based on what you do and on the source you are working on. Each one starts from a preset in chapter 25 or from an approach described earlier in this manual, and points to the two or three settings to adjust first. These are not recipes to apply as is: compression is always set by ear, on your own sound. They are directions to head off in.

When a preset is suggested for a source, it is generally because the machine it imitates is known for that use (chapter 2). Remember that a preset imitates a plugin set up a certain way (chapter 25): it will put you on the trail of a character, and it is up to you to make it fit.

Three habits before you start

Start from a fresh Comprendre. Many presets don't contain every setting and inherit those of the previous preset (chapter 25). Keep a "default" preset (chapter 5) and load it before trying anything else.

Compare level-matched. Compression that raises the level always sounds better. Compensate for the level with **Makeup**, or let the auto-volumes do it (chapter 8), before you judge, and compare with Comprendre bypassed in your host.

Listen to what is being removed. Monitor Delta (MAIN tab, chapter 7) lets you hear the difference between the output and the input: what the compressor takes away or adds. It is the best way to find out whether you are compressing the right things. The monitors of the saturation modules (**Monitor Fet**, **Monitor opto**, the bias monitor) do the same for color (chapters 15 to 21).

By role

You compose, produce or make beats

Think of the presets as a box of colors. On a given track, try one preset from each family in chapter 25 and listen to what changes: speed, thickness, grain. The following table suggests where to begin.

You are looking for…	Start with the family…	For example
a steady, smooth vocal that stays up front	optical (chapter 25)	waves_la2a, slate_fg2a, softube_cl1b

You are looking for…	Start with the family…	For example
bite, snappy attacks, a bass that cuts through	FET	waves_76, arturia_fet, pulsar_1178
thickness, a “vintage” density	vari-mu and tubes	overloud_670, pulsar_mu, klang_mjuc_mk1
cohesion on a group of instruments	VCA and bus	ssl_1, cytomic_glue, bx_townhouse
a strong character, ready for anything	Distressor	slate_fgstress, kiive_xstressor
a warm, round grain	diode bridge	arturia_diode, kush_silka
color without compression	saturators	pa_hg2, kazrog_true-iron, dafalife_sausage

Once you have found the character, set the amount with **Threshold**, then the blend with **Dry/Wet**: a compressor pushed hard but mixed in low under the original signal keeps the source natural while making it denser. That is parallel compression, made famous by New York engineers (chapter 7).

You have a home studio and want to learn

Comprendre is a terrific teacher, as long as you change only one thing at a time.

1. Load your default preset and set **Threshold** to get 3 to 6 dB of reduction on a drum loop. Watch the graphs (chapter 4): the level, the curve, the white gain reduction envelope.
2. Sweep **Attack** from 0.1 ms to 50 ms: you will hear the snare lose its crack and then get it back, and you will see the envelope arrive earlier or later.
3. Sweep **Release**, then try **Release Auto** (chapter 12).
4. Move **Cv Lin/Db** from 0 to 1: two families of compressors, two ways of computing the curve (chapter 10).
5. Do the same with **FF->FB** (chapter 14), **EvMu Br/Cp** (chapter 11), then the peak/RMS blend of the detection (chapter 13).

To see exactly what a setting changes, use François’s method (chapter 24): two instances of Comprendre, the first one in the sidechain of the second, with **Analyse SC** mode enabled. That way you compare Comprendre with itself, setting by setting.

And read the presets: load one, look at which tabs turn yellow, and find out in chapter 25 why François made those choices. Every preset is a little lesson in compression.

You mix

Vocals. Start from an optical preset (waves_la2a, softube_cl1b) or, for a more up-front vocal, from a FET. For a vocal whose level varies a lot, try the automatic knee, which François finds very handy for compressing vocals: **Knee Auto** in the CURVE tab, with the curve in decibels, as he advises for working on the knee (chapter 10). For the release, his starting point is the automatic release with a 200 ms constant, blended at 50% (chapter 12). If the sibilants jump out, add a second Comprendre as a de-esser (chapter 14, “A de-esser”).

Drums. An attack of a few tens of milliseconds lets the transients through before compressing (chapter 12, “Letting transients through”). To make the kit breathe, a dynamic EQ can boost a band when it hits: that is François’s example, a bump around 300 Hz (chapter 18). For grain, a FET preset, or the waveshaper on its own (chapter 17). For a crushed room sound, the Distressor family: its **Nuke** mode, a brickwall limiter (chapter 2), lends itself well to crushing drum room mics.

Bass. The 1176 tops producer polls for bass, and its slight distortion helps the bass cut through: try the FET presets. To make the bass give way to the kick, send the kick to the sidechain input (chapter 14, “Make the bass give way to the kick”).

A bus or a group. The recipe for the SSL bus compressor is well known: a ratio of 2:1 to 4:1, an attack of 10 to 30 ms, automatic release, and only a few decibels of reduction. The ssl_*, cytomic_glue, bx_townhouse and tr_bus presets are variations on it (chapter 25). If the low end makes the whole thing pump, filter it out of the detection (chapter 14, “Keep the low end from causing pumping”).

Electric guitar. Neve diode-bridge compressors are renowned on guitars: try arturia_diode or kush_silka.

Piano. François built a dedicated preset, massenbug_drc2_piano, based on the Massenburg DesignWorks MDWDRC2 (chapter 25).

You master

Mastering calls for subtle moves and very precise listening. Comprendre helps you in several ways.

- **Compare honestly.** The auto-volumes (chapter 8) make up for the loss of level so you can judge level-matched; **Snap Out->In** transfers the compensation into **Output Db** once you have found your setting.
- **Respect the phase.** If you compress only one band, replace the Linkwitz-Riley filters with linear-phase FIR filters (MAIN tab, **X Filters Taps** at 257 or more, chapter 14), at the cost of some latency.
- **Avoid aliasing.** As soon as the compression is fast or a module saturates, turn up **Oversampling** (chapter 7).
- **Work in Mid/Side. L/R to M/S** (chapter 7) or the Mid/Side filters of the detection and the target (chapter 14) let you hold the center without touching the sides, or the other way around.
- **Limit cleanly. Ratio** at 42, zero attack, a lookahead with peak hold (chapter 22, “A limiter-style lookahead”), and the pre-clipper just before it: François points out that it is often a good idea to clip a little before limiting (chapter 7).
- **Start from a preset in the mastering family**: softube_ds1mk3, softube_weiss, pa_elysia, the three massenbug_drc2 presets, or, on the tube side, pulsar_mu (the Manley Variable Mu is a standard in mastering studios) and spl_iron.
- **Make the compression responsive to the spectrum.** Send a dynamic EQ into the detection and have it modulate the release (chapters 18 and 23). It is a delicate setting, but François sees it as a way to achieve an overall coherence that borders on mastering.

> **Latency**
>
> FIR filters and the lookahead delay the signal. Your host in principle compensates for this latency, but it matters if you play or record through Comprendre. With FIR filters, also avoid the **In->Out** compensation, which does not apply to the copy of the signal that feeds them (chapter 8).

You do sound design

Explore what no hardware compressor can do:

- **negative ratios**: above the threshold, the louder the sound, the more it is attenuated (chapter 10);
- the **inverse lookahead**, which makes the compressor act *after* the sound, and the lookahead on the envelope, whose shape you draw yourself (chapter 22);
- the **waveshaper** as a bit crusher, or as noise grain reserved for the peaks (chapter 17, setting ideas at the end of the chapter);
- the **bias**, its "atom" fusion and its dozen or so saturation types (chapters 19 to 21);
- the **modulations**, which vary the ratio, the gate, the attack or the release according to what the dynamic EQs and the waveshaper are doing (chapter 23);
- the **delay** on the saturation products, which gives the grain resonance (chapter 16).

Turn up your listening level, use the monitors to isolate each saturation product, and keep a copy of the plugin's state before you start "tinkering": as François says, some settings are extremely sensitive.

You teach audio

Every major concept in compression can be demonstrated on a single plugin, by ear and on screen:

Concept	Demonstration
threshold, ratio, knee	displayed transfer curve, **Cv Lin/Db**, **Knee** (chapters 4 and 10)
attack and release	white envelope in the graphs, a drum loop (chapter 12)
peak and RMS	**Det Mix**, peak and rms levels in the DET (in) tab (chapter 13)
feed-forward and feedback	**FF->FB** from 0 to 1 (chapter 14)
branching and decoupled detectors	**EvMu Br/Cp** (chapter 11)
optical, FET and vari-mu compression	opto designer, waveshaper, bias, and the presets of each family (chapters 15 to 21 and 25)
detection and target	filters in the INPUT tab and the OUTPUT tab (chapter 14)
imitating a compressor	**Analyse SC** mode and the method in chapter 24

The yellow highlighting of modified settings lets students see instantly what sets one setting apart from another.

By source

Source	Presets to try	Settings to adjust first
Spoken or sung vocals	waves_la2a, native_vc2a, slate_fg2a, softube_cl1b, kiive_kc1	**Threshold**, **Knee Auto**, **Release Auto**

Source	Presets to try	Settings to adjust first
Rock vocals, very up front	waves_76, slate_fgstress	**Threshold**, **Attack**, **Dry/Wet**
Drums (bus)	ao_dbcomp, rockruepel_comptwo_drums, ssl_1, kiive_xstressor	**Attack**, **Release**, **Dry/Wet**
Kick, snare	arturia_fet, softube_fet	**Attack** (let the transient through), **Release**
Bass	waves_76, arturia_fet, ao_fetish	**Threshold**, **Release**; kick in the sidechain
Electric guitar	arturia_diode, kush_silka	**Threshold**, **Release**
Piano	massenbug_drc2_piano	**Threshold**, **Release**
Mix bus	ssl_1, cytomic_glue, bx_townhouse, tr_bus, ao_buster	**Threshold** for 1 to 3 dB of reduction, **Release Auto**
Master	softube_ds1mk3, softube_weiss, pa_elysia, pulsar_mu, spl_iron	**Threshold**, **Oversampling**, FIR filters when working on a band
Color without compression	pa_hg2, pulsarmodular_p42, kazrog_true-iron	settings of the color modules (chapter 25)

Now it’s your turn

The finest compliment you can pay Comprendre is to make it your own. Imitate the compressor in your DAW, the one on your console, or the one you heard on a record, from an excerpt before and after processing (chapter 24). Mix the schools. Invent behaviors that exist on no machine. And save your settings: every preset you create will join a story that began in the 1930s, with the first radio limiters.

APPENDIX 1

Settings Reference

Every setting of Comprendre, in interface order, with its ID (the name your host shows for automation), its range and its default value, as declared in the plugin's code. The last column gives the page of the setting's card; for the ENV and A/R tabs of the detection (in), the lookahead (la) and the bias (bs), it points to the card of the equivalent setting of the follower after the curve (cv).

Main row

Setting	ID	Range / choices	Default	Page
Threshold	`seuil`	−60 to 0	−20	92
Ratio	`ratio`	0.25 to 42	2	92
Attack	`attaque`	0 to 2	0.01	132
Release	`relachement`	0 to 2	0.01	132
Knee	`genou`	0 to 240	0	93
Cv Lin/Db	`gain`	0 to 1	0	89
Gfx Lin/Db	`decibels`	0 to 1	0	28
Gfx Secs	`samples`	0 to 1	0.016667	32
Out->In Secs	`compensation`	0 to 42	42	65
Pre Clip	`sol`	−24 to 0	0	54
Makeup	`maquillage`	−24 to 24	0	58
Post Clip	`plafond`	−24 to 0	0	58
Channels Unlink	`rupture`	0 to 1	0	59
Dry/Wet	`drywet`	0 to 1	1	60

MAIN

Setting	ID	Range / choices	Default	Page
Monitor Delta	`delta`	0 to 1	0	61
Input Db	`entree_premier`	−24 to 24	0	51
Ch.0 Db	`lrms_a`	−24 to 24	0	52

Setting	ID	Range / choices	Default	Page
Ch.1 Db	`lrms_z`	−24 to 24	0	52
Sidechain	`externe`	0 to 1	0	177
Analyse SC	`analyse`	0 to 1	0	373
Oversampling	`oversampling`	1X · 2X IIR · 4X IIR · 8X IIR · 16X IIR · 2X FIR · 4X FIR · 8X FIR · 16X FIR	1X	50
X Filters ...	`fir_taps`	0 · 17 · 33 · 49 · 65 · 97 · 129 · 193 · 257 · 385 · 513 · 769 · 1025	0	193
beta	`fir_beta`	0 to 12	0	196
Phase	`phase`	0 to 1	1	61
Pre Clip Mode	`sol_adaa`	Linked · Unlink · Adaa 1 · Adaa 2	Linked	54
Headroom	`chambre`	−24 to 24	0	55
Post Clip Mode	`plafond_adaa`	Linked · Unlink · Adaa 1 · Adaa 2	Linked	58
L/R to M/S	`lrms`	−1 to 1	0	51
Output Db	`sortie_dernier`	−24 to 24	0	59
(FIR window)	`fir_fenetre`	blackman · blackmanHarris · flatTop · hamming · hann · kaiser · rectangular · triangular	blackman	195

INPUT

Setting	ID	Range / choices	Default	Page
Monitor Input	`fuite`	0 to 1	0	62
In->Sc Secs	`int_a_ext`	0 to 42	42	68
In->Out Secs	`int_a_out`	0 to 42	42	67
FF->FB	`retour`	0 to 1	0	174
FB cmp->all	`cible_retour`	0 to 1	0	175
FX High Cut	`ciseau_bas`	0 to 20000	0	180
FX Low Cut	`ciseau_top`	0 to 20000	20000	180
FX fb cmp	`retour_moymax_fbs`	−16 to 16	0	175
FX Mid/Side	`cotes`	0 to 1	0.5	182
FX deq	`retour_moymax_deq`	−1 to 1	0	183
FX deq->ws	`ideqfet`	0 to 1	0	183
FX ws	`retour_moymax_fet`	−1 to 1	0	184
FX deq	`retour_moymax_dep`	−1 to 1	0	184
Channels Mix	`sss`	Avg · 0 · 1 · Min · Max	Avg	185

DET (in)

Setting	ID	Range / choices	Default	Page
auto secs	`rms`	0 to 2	0.2	166
rms secs	`det`	0 to 2	0	153
flow	`pik_nik`	(empty) · rms=>env · env=>rms	(empty)	155
peak db	`pik_mix`	−24 to 24	0	157
rms db	`det_mix`	−24 to 24	−24	158
env db	`var_mix`	−24 to 24	−24	158
peak pow	`pik_pow`	0.125 to 8	1	159
rms pow	`det_pow`	0.125 to 8	1	159
env pow	`var_pow`	0.125 to 8	1	159
destination	`op_dst`	peak · rms · env	peak	162
formula	`op_sig`	d-=sp · d*=sp · d=dp/s · d+=sp · d=1-d^sp · d=d^sp · d=s<p?s:0 · d=s>p?s:O · d+=p · d-=p · d=max(s,p) · d=min(s,p) · d=ssp+d(1-s)(1-p) · d=min(s,p)s+(1-m)d · d=max(s,p)s+(1-m)d · d=1-sp	d-=sp	163
source	`op_src`	peak · rms · env	peak	163
parameter	`op_par`	0 to 1	0	163
Det Mix	`mmm`	max · avg · min · peak · rms · env	max	160

ENV (in)

Setting	ID	Range / choices	Default	Page
EvMu Lin/Db	`var_db`	0 to 1	0	111
EvMu Br/Cp	`var_indirect`	0 to 1	0	114
Hold branch	`var_pause`	0 to 2	0	115
Env->Mu	`var_mu`	0 to 1	0	118
Mu Min Att	`var_mu_min_a`	0 to 421	1	118
Mu Max Att	`var_mu_max_a`	0 to 421	1	118
Mu Pow Att	`var_mu_a`	0 to 8	1	119
Mu Min Rel	`var_mu_min`	0 to 421	1	119
Mu Max Rel	`var_mu_max`	0 to 421	1	119
Mu Pow Rel	`var_mu_r`	0 to 8	1	119
Mu Speed	`var_mu_sqrt`	0.000001 to 1	1	120
Env/Mu->Gamma	`var_gamma`	0 to 1	0	122
Gamma A delai	`var_gamma_attaque_delai`	0 to 0.25	0	123
Gamma R delai	`var_gamma_relachement_delai`	0 to 0.25	0	123

A/R (in)

Setting	ID	Range / choices	Default	Page
Attack	`var_attaque`	0 to 2	0.01	132
Attack Auto	`var_auto_attaque`	0 to 1	0	136
Attack Altitude	`var_altitude_attaque`	0.000001 to 1	0.367879	133
Attack Shape	`var_attaque_forme`	0 to 8	1	138
Att Ret From	`var_attaque_retour_depart`	0 to 8	1	141
Att Ret Spd	`var_attaque_retour_vitesse`	0 to 8	1	141
Attack Mode	`var_attaque_douceur`	−2 to 2	1	145
Release	`var_relachement`	0 to 2	0.01	132
Release Auto	`var_auto_relachement`	0 to 1	0	136
Release Altitude	`var_altitude_relachement`	0.000001 to 1	0.367879	133
Release Shape	`var_relachement_forme`	0 to 8	1	138
Rel Ret From	`var_relachement_retour_depart`	0 to 8	1	141
Rel Ret Spd	`var_relachement_retour_vitesse`	0 to 8	1	141
Release Mode	`var_relachement_douceur`	−2 to 2	1	145

CURVE

Setting	ID	Range / choices	Default	Page
Ratio Sens	`ratio_sens`	−1 to 1	1	101
mod ratio	`freq_ratio`	−42 to 42	0	365
Knee Auto	`auto_genou`	−240 to 240	0	97
Knee Auto Secs	`plie`	0 to 2	0.2	97
GR Max	`gr_max`	−60 to 60	−60	99
mod gate	`freq_porte`	−42 to 42	0	367
Tresh Gate	`seuil_porte`	−60 to 0	−20	103
Ratio Gate	`ratio_porte`	0.25 to 42	1	104
Knee Gate	`genou_porte`	0 to 240	0	104
Knee Gate Auto	`auto_genou_porte`	−240 to 240	0	104
KG Auto Secs	`plie_porte`	0 to 2	0.2	105
GR Min	`gr_min`	−60 to 60	0	100
Gr Amp	`env_amp`	−8 to 8	1	106

ENV (cv)

Setting	ID	Range / choices	Default	Page
EvMu Lin/Db	`db`	0 to 1	0	111
EvMu Br/Cp	`indirect`	0 to 1	0	114
Hold branch	`pause`	0 to 2	0	115
Env->Mu	`mu`	0 to 1	0	118
Mu Min Att	`mu_min_a`	0 to 421	1	118
Mu Max Att	`mu_max_a`	0 to 421	1	118
Mu Pow Att	`mu_a`	0 to 8	1	119
Mu Min Rel	`mu_min`	0 to 421	1	119
Mu Max Rel	`mu_max`	0 to 421	1	119
Mu Pow Rel	`mu_r`	0 to 8	1	119
Mu Speed	`mu_sqrt`	0.000001 to 1	1	120
Env/Mu->Gamma	`gamma`	0 to 1	0	122
Gamma A delai	`gamma_attaque_delai`	0 to 0.25	0	123
Gamma R delai	`gamma_relachement_delai`	0 to 0.25	0	123

A/R (cv)

Setting	ID	Range / choices	Default	Page
mod attack	`freq_attaque`	−2 to 2	0	368
Attack Auto	`auto_attaque`	0 to 1	0	136
Attack Altitude	`altitude_attaque`	0.000001 to 1	0.367879	133
Attack Shape	`attaque_forme`	0 to 8	1	138
Att Ret From	`attaque_retour_depart`	0 to 8	1	141
Att Ret Spd	`attaque_retour_vitesse`	0 to 8	1	141
Attack Mode	`attaque_douceur`	−2 to 2	1	145
mod release	`freq_relachement`	−2 to 2	0	368
Release Auto	`auto_relachement`	0 to 1	0	136
Release Altitude	`altitude_relachement`	0.000001 to 1	0.367879	133
Release Shape	`relachement_forme`	0 to 8	1	138
Rel Ret From	`relachement_retour_depart`	0 to 8	1	141
Rel Ret Spd	`relachement_retour_vitesse`	0 to 8	1	141
Release Mode	`relachement_douceur`	−2 to 2	1	145

GR (cv)

Setting	ID	Range / choices	Default	Page
Env Pow Unpow	`env_puissance`	0.000001 to 64	1	204
Gr|Env Pow	`env_diode`	−8 to 8	0	205
Env Offset	`env_offset`	−1 to 1	0	206
Monitor opto	`delta_opto`	0 to 1	0	215
Opto	`opto`	−1 to 1	0	211
Opto Env	`opto_onenv`	−8 to 8	0	215
Opt src gn->env	`opto_source`	0 to 1	1	211
Opto A/R mul	`opto_ar`	0 to 421	1	212
Opto severity	`opto_severite`	−8 to 8	1	212
Opto precision	`opto_precision`	0 to 60	0.5	213
Opto saturation	`opto_saturation`	0 to 1	0	213
Opto sat limit	`opto_limite`	0.000001 to 480	24	214
Opto Delai Secs	`opto_delai_nb`	0 to 2	0	217
Opto Delai FB	`opto_delai_fb`	−64 to 64	0	217

F (gr)

Setting	ID	Range / choices	Default	Page
opto amp	`opto_amp`	−42 to 42	1	218
opto src mix	`opto_src_mix`	0 to 1	1	218
opto low herz	`opto_bas_coupe`	0 to 16000	0	218
opto low reso	`opto_bas_resonance`	0.000001 to 16	0.707107	219
opto low mix	`opto_bas_mix`	−8 to 8	0	219
opto low zdf->svf	`opto_bas_type`	0 to 1	1	219
opto mid herz	`opto_mid_coupe`	0 to 16000	0	219
opto mid reso	`opto_mid_resonance`	0.000001 to 16	0.707107	219
opto mid mix	`opto_mid_mix`	−8 to 8	0	219
opto hgh zdf->svf	`opto_top_type`	0 to 1	1	220
opto hgh herz	`opto_top_coupe`	0 to 16000	0	220
opto hgh reso	`opto_top_resonance`	0.000001 to 16	0.707107	220
opto hgh mix	`opto_top_mix`	−8 to 8	0	220
opto dst limit	`opto_dst_limit`	0 to 42	42	220

MOD & LA

Setting	ID	Range / choices	Default	Page
deq mod	`freq_deq`	−1 to 1	0	359
ws mod	`freq_fet`	−1 to 1	0	359
deq mod	`freq_dep`	−1 to 1	0	360
mod amp	`freq_amp`	−421 to 421	1	361
mod pow	`freq_puissance`	0.125 to 8	1	362
mod att secs	`freq_vitesse_a`	0 to 2	0	362
mod rel secs	`freq_vitesse_r`	0 to 2	0	363
la delay	`la_delay`	−1 to 1	0	340
la det pk	`la_det_pk`	0 to 1	0	347
la det min max	`la_det_mm`	−1 to 1	1	347
det->la.det	`la_det2det`	0 to 1	0	342
la env pk	`la_env_pk`	0 to 1	1	345
la env min max	`la_env_mm`	−1 to 1	−1	346
env->la.env	`la_env2env`	0 to 1	1	341

ENV (la)

Setting	ID	Range / choices	Default	Page
EvMu Lin/Db	`la_db`	0 to 1	0	111
EvMu Br/Cp	`la_indirect`	0 to 1	0	114
Hold branch	`la_pause`	0 to 2	0	115
Env->Mu	`la_mu`	0 to 1	0	118
Mu Min Att	`la_mu_min_a`	0 to 421	1	118
Mu Max Att	`la_mu_max_a`	0 to 421	1	118
Mu Pow Att	`la_mu_a`	0 to 8	1	119
Mu Min Rel	`la_mu_min`	0 to 421	1	119
Mu Max Rel	`la_mu_max`	0 to 421	1	119
Mu Pow Rel	`la_mu_r`	0 to 8	1	119
Mu Speed	`la_mu_sqrt`	0.000001 to 1	1	120
Env/Mu->Gamma	`la_gamma`	0 to 1	0	122
Gamma A delai	`la_gamma_attaque_delai`	0 to 0.25	0	123
Gamma R delai	`la_gamma_relachement_delai`	0 to 0.25	0	123

A/R (la)

Setting	ID	Range / choices	Default	Page
Attack	`la_attaque`	0 to 2	0	132
Attack Auto	`la_auto_attaque`	0 to 1	0	136
Attack Altitude	`la_altitude_attaque`	0.000001 to 1	0.367879	133
Attack Shape	`la_attaque_forme`	0 to 8	1	138
Att Ret From	`la_attaque_retour_depart`	0 to 8	1	141
Att Ret Spd	`la_attaque_retour_vitesse`	0 to 8	1	141
Attack Mode	`la_attaque_douceur`	−2 to 2	1	145
Release	`la_relachement`	0 to 2	0	132
Release Auto	`la_auto_relachement`	0 to 1	0	136
Release Altitude	`la_altitude_relachement`	0.000001 to 1	0.367879	133
Release Shape	`la_relachement_forme`	0 to 8	1	138
Rel Ret From	`la_relachement_retour_depart`	0 to 8	1	141
Rel Ret Spd	`la_relachement_retour_vitesse`	0 to 8	1	141
Release Mode	`la_relachement_douceur`	−2 to 2	1	145

DEQ

Setting	ID	Range / choices	Default	Page
Deq low herz	`deq_bas_coupe`	0 to 16000	0	269
Deq low reso	`deq_bas_resonance`	0.000001 to 16	0.707107	270
Deq low min	`deq_bas_min`	−8 to 8	0	270
Deq low max	`deq_bas_max`	−8 to 8	0	270
Deq low zdf->svf	`deq_bas_type`	0 to 1	1	271
Deq mid herz	`deq_mid_coupe`	0 to 16000	0	271
Deq mid reso	`deq_mid_resonance`	0.000001 to 16	0.707107	271
Deq mid min	`deq_mid_min`	−8 to 8	0	271
Deq mid max	`deq_mid_max`	−8 to 8	0	272
Deq hgh zdf->svf	`deq_top_type`	0 to 1	1	273
Deq hgh herz	`deq_top_coupe`	0 to 16000	0	272
Deq hgh reso	`deq_top_resonance`	0.000001 to 16	0.707107	272
Deq hgh min	`deq_top_min`	−8 to 8	0	272
Deq hgh max	`deq_top_max`	−8 to 8	0	272

WS

Setting	ID	Range / choices	Default	Page
Monitor Fet	`delta_fet`	0 to 1	0	259
Fet drive	`fet_input`	−24 to 24	0	246
Fet gr	`fet_gr`	0 to 1	1	244
Fet env	`fet_env`	0 to 1	0	245
Fet even	`fet_pair`	−1 to 1	0	247
Fet odd	`fet_impair`	0.125 to 8	1	248
Fet Thresh	`fet_seuil`	0 to 1	1	249
Fet Knee	`fet_genou`	0 to 240	0	250
Fet *	`fet_rebond_min`	−1 to 1	1	251
Fet sin *	`fet_rebond_max`	−1 to 1	0	252
Fet sin x	`fet_rebond_ratio`	0 to 8	0	252
Fet Precision	`fet_precision`	0 to 256	0	254
Fet Delai Secs	`fet_delai_nb`	0 to 2	0	258
Fet Delai FB	`fet_delai_fb`	−64 to 64	0	258

F (ws)

Setting	ID	Range / choices	Default	Page
fet amp	`fet_amp`	−42 to 42	1	227
fet src mix	`fet_src_mix`	0 to 1	1	235
fet low herz	`fet_bas_coupe`	0 to 16000	0	232
fet low reso	`fet_bas_resonance`	0.000001 to 16	0.707107	232
fet low mix	`fet_bas_mix`	−8 to 8	0	235
fet low zdf->svf	`fet_bas_type`	0 to 1	1	232
fet mid herz	`fet_mid_coupe`	0 to 16000	0	233
fet mid reso	`fet_mid_resonance`	0.000001 to 16	0.707107	233
fet mid mix	`fet_mid_mix`	−8 to 8	0	235
fet hgh zdf->svf	`fet_top_type`	0 to 1	1	233
fet hgh herz	`fet_top_coupe`	0 to 16000	0	233
fet hgh reso	`fet_top_resonance`	0.000001 to 16	0.707107	233
fet hgh mix	`fet_top_mix`	−8 to 8	0	236
fet dst limit	`fet_dst_limit`	0 to 42	42	237

DEQ 2

Setting	ID	Range / choices	Default	Page
Deq low herz	`dep_bas_coupe`	0 to 16000	0	274
Deq low reso	`dep_bas_resonance`	0.000001 to 16	0.707107	274
Deq low min	`dep_bas_min`	−8 to 8	0	274
Deq low max	`dep_bas_max`	−8 to 8	0	274
Deq low zdf->svf	`dep_bas_type`	0 to 1	1	275
Deq mid herz	`dep_mid_coupe`	0 to 16000	0	275
Deq mid reso	`dep_mid_resonance`	0.000001 to 16	0.707107	275
Deq mid min	`dep_mid_min`	−8 to 8	0	275
Deq mid max	`dep_mid_max`	−8 to 8	0	275
Deq hgh zdf->svf	`dep_top_type`	0 to 1	1	276
Deq hgh herz	`dep_top_coupe`	0 to 16000	0	275
Deq hgh reso	`dep_top_resonance`	0.000001 to 16	0.707107	276
Deq hgh min	`dep_top_min`	−8 to 8	0	276
Deq hgh max	`dep_top_max`	−8 to 8	0	276

BIAS src

Setting	ID	Range / choices	Default	Page
Bias +inverse	`bias_inverse`	0 to 1	1	294
Bias g.e g->e	`bias_source`	0 to 1	0	295
Bias out atom	`bias_source_brute`	0 to 1	0	296
Bias out unlink	`bias_rupture`	0 to 1	1	298
Bias out bs->ub	`bias_sortie_pp`	0 to 1	0	297
Bias g.e->out	`bias_sortie`	0 to 1	1	295
Bias Feedback	`bias_feedback`	−42 to 42	1	299
Bias thresh	`bias_seuil`	−60 to 0	0	307
Bias ratio	`bias_ratio`	1 to 42	1	307
Bias knee	`bias_genou`	0 to 240	0	308
Bias input add	`bias_input`	−42 to 42	0	309
Bias Fm In	`bias_fm_in`	0.000001 to 8	1	310
Bias input mul	`bias_input_ssl`	−168 to 168	0	310
Bias mul %	`bias_ssl`	0 to 1	1	311

DET (bs)

Setting	ID	Range / choices	Default	Page
auto secs	`bias_rms`	0 to 2	0.2	301
rms secs	`bias_det`	0 to 2	0	301
flow	`bias_pik_nik`	(empty) · rms=>env · env=>rms	(empty)	301
peak db	`bias_pik_mix`	−24 to 24	0	301
rms db	`bias_det_mix`	−24 to 24	−24	302
env db	`bias_var_mix`	−24 to 24	−24	302
peak pow	`bias_pik_pow`	0.125 to 8	1	302
rms pow	`bias_det_pow`	0.125 to 8	1	302
env pow	`bias_var_pow`	0.125 to 8	1	302
destination	`bias_op_dst`	peak · rms · env	peak	303
formula	`bias_op_sig`	d-=s*p · d*=s*p · d=(d*p)/s · d+=s*p · d=1-d^(s*p) · d=d^(s*p) · d=s<p?s:0 · d=s>p?s:O · d+=p · d-=p · d=max(s,p) · d=min(s,p) · d=ssp+d(1-s)(1-p) · d=min(s,p)s+(1-m)d · d=max(s,p)s+(1-m)d · d=1-sp	d-=s*p	303
source	`bias_op_src`	peak · rms · env	peak	303
parameter	`bias_op_par`	0 to 1	0	303
Det Mix	`bias_mmm`	max · avg · min · peak · rms · env	max	304

ENV (bs)

Setting	ID	Range / choices	Default	Page
EvMu Lin/Db	`bias_db`	0 to 1	0	111
EvMu Br/Cp	`bias_indirect`	0 to 1	0	114
Hold branch	`bias_pause`	0 to 2	0	115
Env->Mu	`bias_mu`	0 to 1	0	118
Mu Min Att	`bias_mu_min_a`	0 to 421	1	118
Mu Max Att	`bias_mu_max_a`	0 to 421	1	118
Mu Pow Att	`bias_mu_a`	0 to 8	1	119
Mu Min Rel	`bias_mu_min`	0 to 421	1	119
Mu Max Rel	`bias_mu_max`	0 to 421	1	119
Mu Pow Rel	`bias_mu_r`	0 to 8	1	119
Mu Speed	`bias_mu_sqrt`	0.000001 to 1	1	120
Env/Mu->Gamma	`bias_gamma`	0 to 1	0	122
Gamma A delai	`bias_gamma_attaque_delai`	0 to 0.25	0	123
Gamma R delai	`bias_gamma_relachement_delai`	0 to 0.25	0	123

A/R (bs)

Setting	ID	Range / choices	Default	Page
Attack	`bias_attaque`	0 to 2	0.01	132
Attack Auto	`bias_auto_attaque`	0 to 1	0	136
Attack Altitude	`bias_altitude_attaque`	0.000001 to 1	0.367879	133
Attack Shape	`bias_attaque_forme`	0 to 8	1	138
Att Ret From	`bias_attaque_retour_depart`	0 to 8	1	141
Att Ret Spd	`bias_attaque_retour_vitesse`	0 to 8	1	141
Attack Mode	`bias_attaque_douceur`	−2 to 2	1	145
Release	`bias_relachement`	0 to 2	0.01	132
Release Auto	`bias_auto_relachement`	0 to 1	0	136
Release Altitude	`bias_altitude_relachement`	0.000001 to 1	0.367879	133
Release Shape	`bias_relachement_forme`	0 to 8	1	138
Rel Ret From	`bias_relachement_retour_depart`	0 to 8	1	141
Rel Ret Spd	`bias_relachement_retour_vitesse`	0 to 8	1	141
Release Mode	`bias_relachement_douceur`	−2 to 2	1	145

BIAS dst

Setting	ID	Range / choices	Default	Page
Mon bias sat	`delta_bias`	0 to 256	0	318
Bias x base	`bias_output_base`	−42 to 42	0	319
Bias x gr	`bias_output_gr`	−42 to 42	0	320
Bias x env	`bias_output_env`	−42 to 42	0	320
Bias Fm On	`bias_fm_out`	0.000001 to 8	1	321
Bias on pre->out	`bias_cible`	0 to 1	0	321
Bias on limit	`bias_limite`	0 to 42	0	323
Bias limit knee	`bias_limite_genou`	0 to 240	0	323
Bias on ratio	`bias_limite_ratio`	1 to 42	42	324
Bias on attack	`bias_limite_attaque`	0 to 2	0	324
Bias on release	`bias_limite_relachement`	0 to 2	0	324
Bias <> bs->ub	`bias_fusion`	0 to 1	0	326
Bias Delay Secs	`bias_delai_nb`	0 to 2	0	330
Bias Delay FB	`bias_delai_fb`	−64 to 64	0	330

F (bs)

Setting	ID	Range / choices	Default	Page
bias amp	`bias_amp`	−42 to 42	1	331
bias src mix	`bias_src_mix`	0 to 1	1	331
bias low herz	`bias_bas_coupe`	0 to 16000	0	332
bias low reso	`bias_bas_resonance`	0.000001 to 16	0.707107	332
bias low mix	`bias_bas_mix`	−8 to 8	0	332
bias low zdf->svf	`bias_bas_type`	0 to 1	1	332
bias mid herz	`bias_mid_coupe`	0 to 16000	0	332
bias mid reso	`bias_mid_resonance`	0.000001 to 16	0.707107	332
bias mid mix	`bias_mid_mix`	−8 to 8	0	333
bias hgh zdf->svf	`bias_top_type`	0 to 1	1	333
bias hgh herz	`bias_top_coupe`	0 to 16000	0	333
bias hgh reso	`bias_top_resonance`	0.000001 to 16	0.707107	333
bias hgh mix	`bias_top_mix`	−8 to 8	0	333
bias dst limit	`bias_dst_limit`	0 to 42	42	334

OUTPUT

Setting	ID	Range / choices	Default	Page
Monitor Output	`sortie_cible`	0 to 1	0	62
FX High Cut	`ciseau__bas`	0 to 20000	0	186
FX Low Cut	`ciseau__top`	0 to 20000	20000	187
FX Mid/Side	`cotes_`	0 to 1	0.5	187
deq	`retour_cible_deq`	0 to 1	1	188
deq->ws	`odeqfet`	0 to 1	1	188
ws	`retour_cible_fet`	0 to 1	1	189
deq	`retour_cible_dep`	0 to 1	1	189
Monitor Comp	`delta_cible`	0 to 1	0	62
Comp Clip	`fauxplafond`	−24 to 0	0	56
Comp Clip Mode	`fauxplafond_adaa`	Unlink · Adaa 1 · Adaa 2	Unlink	56
Comp Db	`sortie_mix`	−24 to 24	0	57
dW | Dw	`sortie_heros`	−1 to 1	0	57
Out->Sc Secs	`out_a_ext`	0 to 42	42	68

GRAPHS

Setting	ID	Range / choices	Default	Page
Gfx Cum Amp	`graph_cumamp`	1 to 16	4	30
Gfx Cum Tmp	`graph_cumtmp`	1 to 3720	600	30
Gfx Detects	`graph_detects`	−2 to 2	0	33
Gfx Lines	`graph_lines`	0 to 2	2	31
Gfx Floor Lin	`graph_degrade_sol_lin`	0 to 199	150	34
Gfx Ceil Lin	`graph_degrade_plafond_lin`	0 to 199	50	34
Gfx DB	`graph_db`	1 to 60	60	28
Gfx Photo	`graph_obturateur`	0 to 1	0	35
Gfx Floor Db	`graph_degrade_sol_db`	0 to 199	0	35
Gfx Ceil Db	`graph_degrade_plafond_db`	0 to 199	0	35
Gfx Make Up	`graph_make`	0 to 1	1	34
Gfx Detects Bias	`graph_detects_bias`	−2 to 2	0	33
Gfx Histos	`graph_gains`	____ · ___g · __o_ · __og · _u__ · _u_g · _uo_ · _uog · a___ · a__g · a_o_ · a_og · au__ · au_g · auo_ · auog	auog	34
Gfx Windows	`graph_fenetre`	0 to 1	1	36

APPENDIX 2

Glossary

ADAA (antiderivative anti-aliasing). A technique for computing a nonlinearity (here, clipping) that uses the antiderivative of the function to reduce aliasing. The *Adaa 1* and *Adaa 2* modes of the clippers (chapter 7).

Altitude. In Comprendre, the fraction of the distance an envelope still has to cover after the set attack or release time. The default value, 1/e ≈ 0.368, corresponds to the classic definition of the time constant (63% of the way covered). See chapter 12.

Attack. The speed at which the gain decreases when the level rises. In Comprendre, each envelope follower has its own.

Bias. In Comprendre, a "height" added to the signal in the Vari-Mu designer, which causes asymmetric saturation on a normal path and an inverse path that are then recombined (chapters 19 to 21).

Branching / **decoupled**. Two ways of building an envelope follower, described by Giannoulis, Massberg and Reiss: the *branching* follower chooses at every moment between attack and release; the *decoupled* follower chains a permanent release stage and an attack stage (chapter 11).

Clipper. A stage that clamps the amplitude of the signal at a threshold. Comprendre has three: Pre Clip, Comp Clip and Post Clip.

Crest factor. The ratio between the peak value and the RMS value of a signal. It drives the automatic attack and release.

dBFS. Decibels relative to the maximum digital level (0 dBFS).

Detection. The level measurement that drives the compressor: peak, RMS, detection envelope, their blends and formulas (chapter 13).

Dry/Wet. The mix between the unprocessed (dry) signal and the processed (wet) signal.

Envelope. A slow curve that follows the level or the gain of a signal. The *envelope follower* computes it from a target and from attack and release times.

Factory (*usine*). The name used in Comprendre's code for the common engine that chains detection, the transfer curve and the envelope follower (chapter 9).

Feed-forward / **feedback**. In feed-forward, the compressor measures its input; in feedback, it measures its own output. Comprendre lets you move gradually from one to the other (chapter 14).

FIR (finite impulse response). A filter whose impulse response is finite. Comprendre's FIR band filters are linear-phase, at the cost of latency (chapter 14).

Gamma. An envelope follower algorithm by Aleksey Vaneev (Voxengo), more abrupt than a classic follower, available in all the ENV tabs (chapter 11).

Gate. A downward expander: it reduces the level of signals **below** its threshold. Set in the CURVE tab.

Headroom. Gain applied before a process and removed after it, to make that process work harder without changing the output level. Comprendre has an internal headroom (Headroom) and an external headroom per channel (Ch.0 Db, Ch.1 Db).

Hold. Holding the gain for a given duration before the release begins.

Knee. A zone of gradual transition around the threshold, between no compression and the full ratio.

Latency. The delay introduced by the plugin and reported to the host, which compensates for it.

Linkwitz-Riley. A family of filters whose low-pass and high-pass outputs sum to a flat amplitude response. Comprendre's default band filters.

Lookahead. Anticipation: the compressor sees the signal a little before hearing it, at the cost of latency (chapter 22).

Makeup. Compensation gain applied after compression to recover the lost level.

Mid/Side. A representation of a stereo signal by its sum (Mid) and its difference (Side).

Monitor. In Comprendre, a setting that lets you hear an internal signal instead of the output (detection, target, difference, saturation product…).

Mu. In Comprendre, an envelope follower whose speed varies with level, inspired by variable-mu tube compressors ("vari-mu") (chapter 11).

Oversampling. Processing at a sample rate that is a multiple of the project's, to reduce aliasing in nonlinear processes.

Peak. The instantaneous absolute value of the signal. Comprendre's default detector.

Ratio. The compression ratio above the threshold: a 4:1 ratio divides level changes beyond the threshold by 4 (in decibels for a dB curve, in amplitude for a linear curve).

Release. The speed at which the gain comes back up when the level drops.

RMS. The root mean square of the signal, a measurement closer to perceived intensity than the peak. In Comprendre, it is computed with an exponential average.

Saturation product. The difference between a saturated signal and the original signal. Comprendre isolates it so it can delay it, filter it, set its amount and limit it before feeding it back in (chapter 16).

Sidechain. The plugin's auxiliary input, which can drive the detection or serve as a reference for the auto-volumes or for the analysis of another compressor.

SVF (state variable filter). A two-pole state variable filter with adjustable resonance.

Target. The part of the signal to which the compressor's gain is applied, selected by the filters and the Mid/Side of the OUTPUT tab. The rest of the signal passes through uncompressed (chapters 6 and 14).

Threshold. The level at which compression begins.

Transfer curve. The relationship between the detected level and the gain to apply, set by Threshold, Ratio, Knee and the CURVE tab (chapter 10).

Waveshaper. A nonlinear transfer function applied to the signal to saturate it (chapter 17).

ZDF (zero delay feedback). A digital filter structure with no delay in the feedback loop; in Comprendre, one-pole filters without resonance.

WORKING DOCUMENT

Points to Confirm

This list gathers the open questions raised while writing the manual. It will be removed from the published edition.

1. On the linear curve, when Knee exceeds 240 times the threshold in amplitude (24 for a threshold of −20 dB, 2.4 for −40 dB), the knee formula gives a negative output at the lowest levels, and the gain drops to the GR Max floor: weak signals are then heavily attenuated, as if by a gate. Is this intended, and is this the “shaving” mentioned in the video? (page 94)
2. In the video, the auto knee display seems incorrect when the channels are unlinked. The code displays the average of the two channels’ knees only if Channels Unlink is exactly 1 (for an intermediate value, it displays the linked knee); moreover, on the dB curve, the per-channel auto knees start from an improperly initialized state after each plugin reset, which gives an almost hard knee for a few time constants. Is this the cause of the defect observed? (page 98)
3. In the videos, a positive GR Min is presented as necessary for negative ratios. In the code, a negative ratio (Ratio Sens at −1) never produces a gain above 1 and works with GR Min at 0; it’s ratios below 1 (and Ratio Gate below 1) that need a positive GR Min. Were these the ratios being referred to? (page 100)
4. With Ratio Sens at −1 and mod ratio active, the code computes the modulation from the positive slope 1/(Ratio + mod ratio): when the modulation is at its resting level, the slope becomes positive again (for example, Ratio 2 and mod ratio +2 give a slope of +1/4 instead of −1/4). Is this intended? It could explain the “weird” behavior noticed in the video. (page 102)
5. With Gr Amp above 1, the applied gain becomes negative as soon as the envelope drops below 1 − 1/Gr Amp (6 dB of reduction for Gr Amp = 2), which inverts the target’s polarity; the code even carries the comment “sécurisation courant < 0 à faire” (a to-do note about guarding against a negative value). Is this intended, or should the gain be clamped at zero? (page 106)
6. In the videos, a power of 2 is presented as making the variation “linear.” In the code, the interpolated speed is proportional to the square root of the time, so no value of Mu Pow makes the time proportional to the level when Mu Min is not zero; it is rather a power of 0.5 that comes close when Mu Min is small. The manual describes the code: is this really the intent? (page 120)
7. The Mu Speed compensation (× 44,100 / host sample rate) does not make the setting independent of the sample rate: the controller’s actual time constant varies as the square root of the host sample rate divided by the square root of the oversampling factor (about 1.5 times slower at 96 kHz, twice as fast at ×4). Is this intended? (page 120)

8. The Gamma envelopes are initialized to 0 (full reduction) when the host prepares the plugin, whereas the other envelopes start at 1; the code carries the note “BUG.” With Env/Mu->Gamma above 0, the very beginning of playback therefore gets a fade-in lasting as long as Gamma’s release. Should they be initialized to 1? (page 123)
9. With lookahead active (la delay other than 0), the Gamma envelopes of the “future” chain only receive their times, delays and direction when the plugin is prepared: a change to Attack, Release, altitude or a Gamma delay during playback does not reach them (unless Auto or Shape is active). Is this intended? (page 123)
10. In the walkthrough, zero mode is presented as also working with Gamma, and the decoupled linear mode as acting only on the attack. In the code, Gamma completely ignores Attack Mode and Release Mode, and in decoupled mode it is Release Mode that acts (linear and zero included) while Attack Mode has no effect (its version is commented out). The manual describes the code: is this the intended behavior? (page 125)
11. The direction of Mu Min and Mu Max is opposite between ENV (cv) (“Max” = loud detected level) and the other three tabs (“Max” = silence for ENV (in), gain with no reduction for ENV (la), zero bias for ENV (bs)). The code carries the comment “contre détecteur” (counter-detector): is this inversion intended? (page 126)
12. In ENV (in) and ENV (bs), decibel computation smooths the logarithm of (1 − level), not the level in decibels. Is that the intent, or was decibel computation supposed to apply to the level itself? (page 126)
13. In the walkthrough, it is said that for the envelope placed before the curve “the attack is the release.” In the current code, the inversion is forced, so the Attack of A/R (in) does set the rise of the detected level (and thus the onset of compression). The manual describes this behavior: is this really the intended current behavior? (page 126)
14. For ENV (la), the Gamma envelope is set up as if the signal being followed were a level (“non-inverted” direction), whereas it is a gain: for the Gamma part of this tab only, the attack (gain going down) takes the longer of the two A/R (la) times and the release the shorter, the opposite of the classic and Mu envelopes in the same tab. Is this intended? (page 127)
15. With a Shape other than 1, the time passed to the Gamma envelope is recalculated at each sample without the −1/ln(altitude) factor applied in the normal case: with an altitude other than 1/e, the Gamma part therefore changes time as soon as Shape moves away from 1. Is this intended? (page 134)
16. With Shape = 0, the envelope makes a single jump of (1 − altitude) at the start of each phase, then stays frozen as long as the position has not returned to the start; the shape graph shows a step. Should this case be presented as a usable effect, or discouraged? (page 138)
17. In the decoupled envelope and in Gamma, the curve for both directions is advanced at each sample, before the return of the inactive direction: the effective return speed there is Ret Spd − 1, so at the default setting (1) the position never comes down (it stays capped at Ret From), whereas in branching mode it returns to the start. In decoupled mode, the attack position also keeps advancing during releases before being capped. Is this intended? (page 142)
18. With Channels Unlink above 0, the calculation of the displayed position adds each channel’s contribution twice: at 1, the vertical line sits at double the actual position (and always at double

for A/R (bs)). For A/R (bs), whether the line is displayed depends on the bias master switch, not on whether this follower is active. Is this intended? (page 147)

19. In the walkthrough, you explain that for the envelope placed before the curve, attack and release are “reversed.” In the current code, the internal inversion is compensated for: Attack in A/R (in) sets the rise of the detected level and Release its fall, as described above. Is that the intended behavior? (page 155)
20. The detection envelope is clamped to 0 dBFS (before its volume), whereas the peak and the RMS are not. With a high input gain or Headroom, or a hot sidechain, it can therefore no longer follow the level above 0 dBFS. Is this intended? (page 155)
21. With Det Mix set to min, the detection starts from 1: it never exceeds 0 dBFS, and if all three streams are at −24 dB, it stays at 1 permanently, which forces constant compression, as for a signal at 0 dBFS. Is this intended? (page 161)
22. The labels of four formulas don’t match the computation: “d=sp?s:O” (with the letter O) keep the destination d, not the source s; “d=1-d^sp” and “d=d^sp” use the exponent $s \cdot (p^2 + 1)^8$, not $s \cdot p$. The manual describes the actual computation. Should the labels be fixed, or the computation? (page 165)
23. In FIR mode (X Filters Taps other than 0), as soon as a detection filter is engaged, the code feeds the filter with the internal signal alone: FF->FB then no longer has any effect on the detection, while FX fb cmp stays active. Is this intended? (page 175)
24. With an active lookahead (negative la delay) and Sidechain above 0, the “future” detection reads the sidechain ahead of time only if a detection process is engaged (filter, Mid/Side, FX fb cmp, send to the DEQ/WS); otherwise, it reads the sidechain without any advance. The sidechain’s behavior therefore changes as soon as you engage a detection filter. Is this intended? (page 177)
25. In Min mode, the code starts from the value 1 before looking for the minimum of the channels: if all the channels exceed 1 (0 dBFS in the Headroom domain, which happens with a positive Headroom or a very hot signal), the detection stays stuck at 1. Is this intended? (page 185)
26. The code applies the target’s DEQ, WS and DEQ 2 before multiplying the target by the compressor’s gain. In video 24, you said you thought they were “normally after.” Is the order in the code the one you want? (page 189)
27. JUCE interprets the value of X Filters Taps as an order: the filter has taps + 1 coefficients and a delay of taps/2 samples, whereas the plugin compensates for (taps − 1)/2. According to the calculation, this half-sample offset limits the attenuation of the filters obtained by subtraction (with FX High Cut, at 44.1 kHz: about −43 dB at 100 Hz, and only −17 dB at 2 kHz for a 5 kHz cutoff), and the center coefficient, placed next to the actual center, creates a floor that rises with the cutoff frequency (about −46 dB for a 5 kHz low-pass). Should taps − 1 be passed to JUCE? (page 194)
28. In FIR mode, a filter is only recomputed when “its” frequency changes (or the taps, the window, the beta, the oversampling). According to the code, setting FX Low Cut before FX High Cut leaves the second filter of the band uncomputed, and setting FX High Cut back to 0 while FX Low Cut is active leaves the filter on the old FX High Cut frequency; touching the other frequency afterward, or X Filters Taps, fixes the situation. Can you check this by ear? (page 194)
29. With the flatTop window, JUCE’s window is not normalized (its center value is 4.6): according to the calculation, the low-pass has a gain of up to +13 dB in its passband, and the filters obtained

by subtraction (FX High Cut, bands) boost instead of cutting. Should this window be removed from the menu? (page 196)

30. The code truncates beta to an integer: only the values 0, 1, 2… 12 have an effect, whereas the knob moves in steps of 0.01. Is this intended? (page 196)
31. Env Pow Unpow also raises to the power p the target gain as it is displayed (the purple line in the history and the graph's curve during operation), and as it is received by the opto designer ("curve" source) and by the bias. Is this intended, or was only the input of the followers meant to be modified? (page 204)
32. The opto's time is divided by e^severity + R. When the envelope goes above 0 dB (positive Env Offset, positive GR Min…), R becomes negative; if e^severity + R becomes negative, the time becomes negative and the optical follower diverges. In a faithful simulation of the code (severity 1, envelope at +3.5 dB, Opto 1), the target goes silent in less than 0.1 s. Was this case anticipated? (page 215)
33. The delay memory holds 16 seconds at the host's sample rate, but the delay is counted at the oversampled rate. With ×8 oversampling, a 2 s delay comes down to a delay of a single sample; with ×16, any delay longer than 1 s is shortened. Is this intended? (page 230)
34. Fet even and the DC bias (Fet sin x at 0) add a DC component to the saturation product, and no filter in the chain seems to remove it afterward. Since this product is dosed by the gain reduction, this DC varies with the compression. Is this intended, or should users be advised to remove it with the low band of F (ws)? (page 248)
35. With a positive Fet drive, Fet odd distorts only levels below 1/drive: beyond that, the curve returns to the diagonal, with a break in slope visible on the graph (for example above 0.5 with +6 dB of drive). Is this the intended behavior? (page 249)
36. When half the knee width exceeds the threshold (Fet Knee / 240 > Fet Thresh, for example a threshold of 0.2 with a knee of 120), the knee zone includes zero and the curve shows a small jump at the zero crossing. Is this intended? (page 250)
37. In spreader mode, the amplitude of the sine is multiplied by $|x \times \text{drive}|$ raised to the power Fet Precision. With a positive drive and a high precision (for example +24 dB and 40), this factor exceeds the range of floating-point arithmetic and the result becomes infinite. Should this computation be clamped? (page 255)
38. As soon as a band of a dynamic EQ is active, the signal is clipped at ±1 (internal 0 dBFS), before the compressor's gain. With a positive Headroom or a band that rises a lot, this clipping is audible. Is it intended as a safety measure? (page 264)
39. The gain envelope can exceed 1 (for example with a positive Env Offset or a GR Min above 0 dB). The dynamic EQs' coefficients are then extrapolated beyond max, and the waveshaper's Fet gr amount becomes negative (the saturation product is inverted). Is this an intended use case? (page 267)
40. There are two small differences in the code between the normal and inverse paths. The Fet gr modulation of the waveshaper sent to the detection receives, on the inverse path, the envelope instead of the gain reduction (the target side is correct). And in linked stereo, with an active modulation (MOD & LA tab, chapter 23) and the Channels Mix modes "channel 0," "channel 1" or "max," the inverse path's modulation value is accumulated instead of being copied. These two

discrepancies can create a saturation product even without an input height; are they intended? (page 283)

41. The videos don't give all the values used in the demonstration (threshold, knee, RMS time of the bias detector, attack and release, sine level). Could you provide the exact settings from the introduction video, so they can be turned into an example preset? And could you confirm that the article cited is indeed "Variable-bias compression explained" from the Pulsar Audio blog, and give its link? (page 291)
42. Intermediate values of Bias +inverse produce a level bump of up to +1.4 dB, and the label "+inverse" suggests a switch. Are the intermediate values meant to be used, or should users be advised to stay at 0 or 1? (page 294)
43. In linear mode (Cv Lin/Db less than 1), with a ratio other than 1, when the bias value is exactly 0 and the bend extends down to 0 (for example a knee above 120 with a threshold of −6 dB), the code divides by 0. The bias value becomes undefined and can spread to the detection and the sound. A zero value is common (silence, "g.e" source with no compression). Is a safeguard needed? (page 308)
44. Bias Fm In changes the detection as soon as one of the five amounts is active, including when only an output height (Bias x base, x gr, x env) is used. A control in the BIAS src tab then changes the compression with no connection to the input height. Is that intended? (page 310)
45. Bias input mul goes from −168 to +168, but only positive values multiply the detection: a negative value activates the bias (inverse path included) with no other effect. Is that intended, or should the absolute value be used? (page 311)
46. For Monitor Input, the height removed is that of the last channel, applied to all channels, and the multiplication by Bias input mul isn't undone. In stereo, with a per-channel bias (Bias out unlink at 1), the monitor therefore doesn't play exactly the detection without the bias. Is that acceptable? (page 312)
47. In the rule "the orange curve shouldn't be stronger than the red one," is it indeed the input height (final bias value, in orange) compared with the final detection (in red)? And should the graph also show the bias when it is used only at the destination (Bias input add at 0), where it currently stays at zero? (page 312)
48. Bias Fm On goes down to 0.000001. With unbiased fusion, the signal is divided by this value: whatever is left of the height after unbiasing (for example with Bias +inverse at 0, or with different gains on the two paths) can then be amplified enormously, and bias dst limit doesn't limit anything by default. Is a safeguard needed? (page 322)
49. When half the knee width exceeds the threshold (Bias limit knee / 240 greater than Bias on limit, for example Bias on limit at 0 and Bias limit knee nonzero), the limiter no longer returns 0 for a zero signal: its output jumps abruptly at the zero crossing (±0.0625 for a threshold of 0 and a knee of 60). This produces a buzz or noise. Is this intended? Is it the source of the "new kind of white noise" in the fusion video? (page 324)
50. In the "pre" position, unbiasing removes the height B, whereas the output contains B × e (the height has been multiplied by the compressor gain). With Bias +inverse at 0, or with different gains on the two paths, an offset of −B × (1 − e) / Fm On therefore remains, and it follows the gain reduction. Is this intended, as part of the saturation product, or should B × e be removed instead? (page 326)

51. The “table of 32 combinations” announced in the video isn’t in the plugin. Could you provide a link to it or its content, and specify the five binary choices that make it up? The tables above show, for example, that with Bias +inverse at 1 and Bias Fm On at 1, bs or ub fusion changes nothing: that would explain why only about ten of the 32 combinations are useful. (page 330)
52. With Channels Unlink greater than 0, the per-channel detector hold is applied to the unrectified signal: the maximum only keeps the positive peaks and the minimum the negative peaks, unlike in linked mode. Moreover, in reverse with a nonzero hold window, the memory for the per-channel detector hold isn’t created, and the per-channel anticipated detector is 0. Is this intended? (page 347)
53. The delay line holds sixteen times the host sample rate, which is only one second at ×16 oversampling. With ×16 and a la delay close to ±1 s (plus the FIR filter delay), the requested delay exceeds the line, and the delay actually read is wrong. Should la delay be limited in this case? (page 351)
54. Changing la delay, la det pk or la env pk rebuilds the hold buffers while the audio is being processed. Could this cause clicks, or even a crash, during playback? Should we advise setting these three parameters with playback stopped? (page 351)
55. With the “envelope” school, only the final gain is replaced by the future gain. The opto designer’s envelope (GR (cv) tab, chapter 15) is still computed by the present chain, relative to the present gain, and is therefore not anticipated. Is this intended? (page 352)
56. The two DEQ sources carry the same label, “deq mod,” in the interface and in the host’s automation. Would a distinct label for the third one (for example “deq2 mod”) be preferable? (page 360)
57. If the modulation value is negative (high amplitude) and mod pow is other than 1, the power calculation doesn’t produce a number (NaN), and this value then stays trapped in the smoothing. Should the modulation value be clamped to 0 before the power? (page 362)
58. With Ratio Sens at −1, the code computes the ratio modulation even when mod ratio is 0: the added amount is then 2/Ratio × m, so that at rest (m = 1) the slope becomes positive again (+1/Ratio) and, with no source active, it depends on a modulation value that isn’t computed. Shouldn’t the negative ratio obtained with Ratio Sens −1 be independent of the modulations? (page 366)
59. With no source active, mod ratio, mod gate (and mod attack or mod release if an Auto control in the A/R (cv) tab is active) use a modulation value that isn’t computed: in linked mode, it’s undefined; in unlinked mode, it stays frozen at its last value. Should these targets be neutralized when no source is active? (page 368)
60. These 21 presets contain settings from earlier versions (for example mu_reverse, var, fet_ciseau, courbe_mix) at a non-neutral value, which version 1.0.3 ignores: do they still sound as intended? In addition, 65 presets are incomplete and therefore inherit values from whatever was set before: would you like to re-save them with version 1.0.3, so that each one contains every setting? (page 379)
61. slate_fg2a carries the strongest Env Offset of the 80 presets (−0.55), while waves_la2a has none: is the LA-2A you talk about in the French video the FG-2A? With the default threshold, ratio and curve, this offset attenuates the signal by 6.9 dB at rest and makes the target go silent when the detected level reaches 0 dBFS: is that really the intended effect? (page 389)

62. In mixland_vacattack, bias amp is 0: the bias product is canceled before its filters and, with no input height, the bias has no audible effect. The settings in the BIAS src, BIAS dst and F (bs) tabs therefore serve no purpose, other than doubling the CPU load (inverse path active). Is this intentional? (page 392)
63. The dmg_trackcomp_76a preset (decoupled envelope, Attack Shape 3.08, Att Ret From 1.19, Att Ret Spd at 1) owes its fast attack to the behavior flagged in chapter 12: in decoupled mode, the position on the attack curve never returns to the start. If this behavior is fixed, each attack will start again from the beginning of the S-curve (first decibel in 11.6 ms instead of 0.15 ms): will this preset need to be revised? (page 401)
64. Two settings are surprising. In pulsar_1178, the −0.4 offset combined with the linear 4:1 curve at −20 dB makes the output level drop when the input exceeds about −12 dBFS at steady state (−35 dB of gain on a sine at 0 dBFS): is this intentional? In korneff_comp2, the deq, ws and deq amounts in the OUTPUT tab are at 0, so Fet Precision 25 and fet amp −0.5 have no audible effect: should they be removed, or was the waveshaper meant to be heard? (page 405)
65. No T-RackS module is called “Vintage”, and the preset has no bias, no feedback and no waveshaper: only its dynamic EQs set it apart. Which module (and which setting of that module) does tr_vintage imitate? (page 411)
66. In arturia_tube-sta, Bias input add is 0.0000038 and seems to serve only to activate the module. Bias limit knee (25, or 0.21 in amplitude) is more than twice Bias on limit (0.054): the limiter returns about ±0.006 (−44 dBFS) for a weak signal, a case pointed out in chapter 21, which adds grain to the low end of quiet passages. Is this intended? (page 415)
67. Which plugin (SSL Native Bus Compressor 2, its first version, another emulation) and which settings of that plugin were used for each of the five ssl presets? What do “1”, “2”, “b”, “f” and the final “_” mean? The text assumes that “b” stands for Bias input mul and “f” for feedback. (page 425)
68. ssl_1_b and ao_dbcomp are saved with Analyse SC checked and Phase unchecked: when they load, the output is polarity-inverted. Is this intended, or should these two presets be saved again with Phase checked and Analyse SC unchecked? (page 427)
69. The slate_fgstress preset relies on Fet odd 1.3 with a Fet drive of +6 dB, the case flagged in chapter 17 where the curve shapes only levels below 0.5 and then rejoins the diagonal with a kink. If this behavior is changed, the preset’s character will change: should this be planned for? (page 437)
70. In arturia_diode, Fet env at 1 (with Fet gr at 1) makes the saturation constant, and the product keeps only its lows below 100 Hz, subtracted (fet low mix −0.97, fet src mix 0). The waveshaper then acts mainly as an attenuation of the lows of quiet signals (about 6 dB), more than as a saturation. Is this the intended effect? (page 441)
71. The SILIKA is known for its distortions (Zener or germanium diodes, transformers, Fury knob), but kush_silka engages no saturation module. Does the imitation deliberately cover only the compression, leaving the color to the user? (page 442)
72. Does the softube_weiss preset really imitate Softube’s Weiss Compressor/Limiter, and not another Softube Weiss plugin (the MM-1 maximizer, for example)? The preset name only specifies the brand. (page 446)

73. In massenbug_drc2_piano, the Attack in the A/R (la) tab is 13 ms, but env->la.env is at 0: the lookahead's dedicated follower isn't computed and this setting has no effect. Is it a leftover from exploration, or was some of the "envelope" school meant to be kept? (page 450)
74. With Env Offset at −0.303 on a linear curve (threshold −20 dB, ratio 4), the voxengo_polysquasher3 preset attenuates the signal by 3.1 dB at rest, and its steady-state output decreases when the detected level exceeds the threshold (about −23 dBFS at the threshold, −33 dBFS at 0 dBFS). Does this overcompression match what you observed on Polysquasher, or should the offset value be revisited? (page 458)
75. Does the caelumçsmmov preset really target Caelum Audio Smoov ("ç" typed instead of "_", "smmov" for "smoov")? Also, it sets Mu Min Rel to 6.907 but leaves Env->Mu at 0: since the Mu envelope isn't computed, this setting has no effect in version 1.0.3. Should Env->Mu have been turned on? (page 461)
76. The kazrog_true-iron preset is saved with Phase unchecked, that is, with inverted polarity, like ao_dbcomp and ssl_1_b. Is this intentional (the imitated plugin's polarity) or a leftover from the analysis session, where this button is used to hear the difference? (page 469)
77. In kiive_filk2_saturation, the bias's inverse path (active by default) recombines the two paths by their difference: the even part of the waveshaper, including the 2nd harmonic set by Fet even at −1, largely cancels out, except insofar as the two paths compress differently. Was this even harmonic meant to be heard (in which case Bias +inverse at 0 would bring it back)? (page 471)