GUITAR AMP BASIC - DOCUMENTATION

Martin Penberthy

Signal flow:

Gain (input) → ProcessorChain [LadderFilter → IIRFilter1 → Gain1 → Waveshaper1 → Gain2 → Waveshaper2 → Gain3 → Waveshaper3 → IIRFilter2 → IIRFilter3 → IIRFilter4] → Gain (output) → Convolver

Modules:

Gains: juce::dsp::Gain<float>

- InputGain: This control has a range of -96 - +48dB and is set to a default value of 0dB. The way this plugin is setup, even if the pregains of all the waveshapers are set to 0, they are not bypassed. Turning up the InputGain will increase the amount of distortion applied to the signal. This can be used to the user's advantage if more distortion is desired.
- Gain1(Pregain1): This control has a range of 0 - +48dB and is essentially a control for the amount of distortion that is applied via the first waveshaper.
- Gain2(Pregain2): Same as above but for the second waveshaper.
- Gain3(Pregain3): Same as above but for the third waveshaper.
- OutputGain: This control has a range of -96 - +48dB and is set to a default value of 0dB. This should be used as a final level control for the output signal.

LadderFilter:

This is the PreEQ control and is controllable by the user. The PreEQ knob value goes from 1.0 to 10.0 with a default value of 5.0 which is representative of the frequency of the low pass filter divided by 1000. For convenience the value shown to the user is 1.0-10.0 and not the actual frequency. This knob is extremely useful for achieving a good high gain tone. As distortion is applied, the high frequency content of the signal is augmented and can result in an undesirable sound, especially with large amounts of gain. Applying this low pass filter with a low resonance smooths out the tone significantly. For crunch or other lightly distorted tones, adding in more of the high frequencies will likely be more desirable. With that being said, use your ears. If it sounds good, it sounds good.

juce::dsp::LadderFilterMode::LPF12

Res: 0.2

Freq: (SliderVal 1.0-10.0) * 1000.0 Hz

IIRFilters: using IIRFilter = juce::dsp::IIR::Filter<float>
using IIRCoefs = juce::dsp::IIR::Coefficients<float
juce::dsp::ProcessorDuplicator<IIRFilter, IIRCoefs>

- IIRFilter1(Internal): This filter is an internal low shelf and is not controllable by the user. It is used to shave off some of the low end which results in a tighter final tone. Gain values for IIRFilters go from 0.0-2.0 with 1.0 being unity.
  juce::dsp::IIR::Coefficients<float>::makeLowShelf(sampleRate, 400.0f, 0.4f, 0.1f)
  Freq: 400Hz
  Q: 0.4
  Gain: 0.1

- IIRFilter2(Low): This is the peak filter LowEQ knob which is controllable by the user. The user parameter knob goes from the value 0.0 to 0.2 with a default value of 1.0 (unity). The value is retrieved from the parameter and stored in a float gainLow. One quirk of IIRFilter coefficients is that a gain value of 0.0 will cause silence so the knob value goes all the way to 0.0 but internally it is snapped to 0.1.
  juce::dsp::IIR::Coefficients<float>::makePeakFilter(getSampleRate(), 100.0f, 0.6f, gainLow)
  Freq: 100Hz
  Q: 0.6
  Gain: 0.0 (default slider value, otherwise is set to whatever the slider is set to)

- IIRFilter3(Mid): This is the peak filter MidEQ knob which is controllable by the user. The user parameter knob goes from the value 0.0 to 0.2 with a default value of 1.0 (unity). The value is retrieved from the parameter and stored in a float gainMid. One quirk of IIRFilter coefficients is that a gain value of 0.0 will cause silence so the knob value goes all the way to 0.0 but internally it is snapped to 0.1.
  juce::dsp::IIR::Coefficients<float>::makePeakFilter(getSampleRate(), 500.0f, 0.9f, gainMid)
  Freq: 500Hz
  Q: 0.9
  Gain: 0.0 (default slider value, otherwise is set to whatever the slider is set to)

- IIRFilter4(High): This is the peak filter HighEQ knob which is controllable by the user. The user parameter knob goes from the value 0.0 to 0.2 with a default value of 1.0 (unity). The value is

retrieved from the parameter and stored in a float gainHigh. One quirk of IIRFilter coefficients is that a gain value of 0.0 will cause silence so the knob value goes all the way to 0.0 but internally it is snapped to 0.1.

juce::dsp::IIR::Coefficients<float>::makePeakFilter(getSampleRate(), 5000.0f, 0.6f, gainHigh)

Freq: 5000Hz

Q: 0.6

Gain: 0.0 (default slider value, otherwise is set to whatever the slider is set to)

Waveshapers: juce::dsp::WaveShaper<float>

- Waveshaper1: This is the first waveshaper and whose waveshaping function can be selected via the "Dist Type" dropdown menu. The functions available to choose from are listed below. (x = sample value)
  - Tanh
    - std::tanh (x);
  - x/abs(x)+1
    - x / (std::abs(x) + 1)
  - Amp2
    - (x * (std::abs(x) + 0.9f)) * 1.5f / (x * x + (0.3f) * (0.1f / std::abs(x)) + 1.0f) * 0.6f;
  - Atan
    - std::atan(x);
  - HalfRect
    - if(x < 0.0f)
      return 0.0f
      else
      return x;
  - Amp1
    - ((x / (std::abs(x) + 0.9f) * 1.5f) / (x * x + (0.0f - 1.0f) * std::abs(x) + 1.0f)) * 0.7f;

- Waveshaper2: This is the second waveshaper whose waveshaping function is not changeable by the user. Future expanded versions of the plugin will provide control over functions for all waveshapers. The function used in Waveshaper2 is shown below.
  - x / (std::abs(x) + 1)

- Waveshaper3: This is the third waveshaper whose waveshaping function is not changeable by the user. The function used in Waveshaper3 is shown below. (Notice the 2 instead of the 1, after some testing, it sounded better.)
    - x / (std::abs(x) + 2)

Convolver: This convolution module is used for cabinet emulation. The user can load a .wav or .mp3 impulse response file to be used. If the project is closed and reopened, the plugin stores the file to be recalled later.

Resources:

A very helpful article I used as a starting point for my design and some of the waveshaping functions.
https://thmstudio.com/blog/how-i-made-a-guitar-amp-plugin/

Stream VOD that I used to implement loading and saving IR files. Also made implementing convolution much easier.
https://www.youtube.com/watch?v=gApXhjM-API

JUCE Documentation
https://docs.juce.com/master/structdsp_1_1IIR_1_1Coefficients.html
https://docs.juce.com/master/classdsp_1_1IIR_1_1Filter.html
https://docs.juce.com/master/classdsp_1_1LadderFilter.html
https://docs.juce.com/master/structdsp_1_1WaveShaper.html
https://docs.juce.com/master/classdsp_1_1Gain.html

Helpful for visualizing waveshaping function
https://www.desmos.com/calculator