The Bright Side of the Dark Side of DSP: Audio Effects using GNU Radio

Ashish Chaudhari
GNU Radio Conference 2018
Henderson, NV, U.S.A.
Deconstructing the Title

The Bright Side of the Dark Side of DSP:
Audio Effects using GNU Radio

- Signal processing phenomena that RF and communications engineers try to avoid, or work around...
- ... that we embrace, in order to build...
- ... some cool guitar effects in GNU Radio.
Overview: Audio DSP

Filter Resonance
- Wah-Wah Guitar Effect
- Demo

Non Linearity
- Distortion Guitar Effect
- Demo

Takeaways
Audio Spectrum: Frequency

Frequency Range: 20 Hz to 20 kHz

Human Perception Bands
1. Sub-bass (20 - 60 Hz): Deep bass that is “felt”
2. Bass (60 - 250 Hz): Determines thickness of sound
3. Low Midrange (250 - 500 Hz): Low order harmonics of instruments
4. Midrange (500 Hz - 2 kHz): Vocals
5. Upper Midrange (2 - 4 kHz): Determines the timbre of the audio
6. Presence (4 - 6 kHz): Conveys the distance of sound
7. Brilliance (6 - 20 kHz): Conveys sparkle and air of a sound

Human perception of frequency is logarithmic

https://www.headphonezone.in/pages/what-are-the-different-components-of-the-audio-spectrum
Audio Spectrum: Amplitude

- The “amplitude” of sound is defined in terms of sound pressure level.
- dB SPL (decibels of sound pressure level) is the unit loudness. *dB SPL is often abbreviated as dB*.

- Human perception of loudness (volume) is logarithmic
- Human threshold of hearing is assumed to be 0 dB SPL (typically 20 uPa)
- Volume Ranges
  - Whisper: 15-25 dB
  - Background noise: about 35 dB
  - Normal home or office background: 40-60 dB
  - Normal speaking voice: 65-70 dB
  - Live Rock music: 120 dB+
  - Pain Threshold: 130 dB
  - Jet aircraft: 140-180 dB

https://en.wikipedia.org/wiki/Sound_pressure
https://www.lifewire.com/what-are-decibels-db-1846876
Is Audio DSP Really Different from RF DSP?

Fundamentally, **NO**. There are some differences though.

- **Sample rates** are much lower than typical RF applications
  - 44.1 kHz, 48 kHz, 96 kHz, 192 kHz (studio quality)
- **Human perception** drives filter design
- **Frequency axis on spectrum plots** is often **logarithmic**
  - So are cutoff frequencies on filters
- It’s **hard** for us to “hear” **phase** except in the stereo sense
  - IIR filters for the win!
  - It is typical to do audio DSP in the spectral domain
Resonance

Wah-Wah Effect
Resonance

Resonance is the tendency of a physical system to oscillate at a high amplitude at certain frequencies.

- The term actually originates from acoustics.
- Resonance is not inherently bad, however, it can cause filters and control systems to go unstable.
- A filter can “ring” indefinitely when stimulated with an input at the resonant frequency.
- Resonance is usually minimized (damped) when designing a filter or control loop.

https://pedals.thedelimagazine.com/production-advice-know-your-filters-low-pass-band-pass-high-pass-resonance/
The Wah–Wah Effect

**A guitar effect that alters the tone of the guitar signal to create a distinctive sound, mimicking the human voice saying the onomatopoeic name "wah-wah".**

The effect sweeps the cutoff frequency of a resonant digital State-Variable Filter (SVF) to create the sound, a spectral glide, know as the wah effect.

Recognizable as the quintessential Jimi Hendrix sound
State Variable Filter (SVF)

A filter comprised of multiple **series integrators**, each **feeding back** and **summing**.

**Example: 2-pole Butterworth LP filter**

https://ccrma.stanford.edu/~jos/svf/

**Hardware Implementation of an SVF**
Digitize the SVF using the **Forward and Backward Euler transform**

**Forward Euler**
\[
\tilde{s} \leftarrow \frac{z - 1}{T} = \frac{1 - z^{-1}}{T z^{-1}}
\]

**Backward Euler**
\[
\tilde{s} \leftarrow \frac{1 - z^{-1}}{T}
\]

\[
H_1(s) = \frac{1}{s^2 + \frac{1}{Q} \tilde{s} + 1}
\]

\[
Q = \frac{\omega_c}{2 \alpha} \quad \tilde{s} \triangleq \frac{s}{\omega_c}
\]

Damping

Cutoff Frequency

\[
y_h(n) = u(n) - y_l(n) - Qy_b(n)
\]
\[
y_b(n) = \omega_c Ty_h(n) + y_b(n)
\]
\[
y_l(n) = \omega_c Ty_b(n) + y_l(n)
\]
The Wah SVF

- Effect = A linear combination of the **bandpass** and **lowpass** output of the SVF
- Vary the **cutoff frequency** \((\omega_C)\) over time
Wah SVF: DEMO
Non-Linearity  Distortion
Do we care about non-linearity?

Yes, *all the time!*

- **Power Amplifiers**
  - PAs are driven outside their linear range for higher efficiency
  - Sophisticated methods like digital predistortion (DPD) to make a non-linear PA linear
- **RF/Comms Design**
  - Calibration to remove physical nonlinear effects
  - We prefer FIR filters because phase is linear
- ...
Distortion Effect

*Distortion (a.k.a. overdrive) is an effect where the *gain* of an instrument is increased to push amplitudes into a *non-linear* region of operation to produce a “fuzzy” or “gritty” tone.*

**Stages of Distortion**
1. **Boost** the input signal (apply gain)
2. Apply a **non-linearity** to the signal, primarily to drive it into clipping

Typically used with an electric guitar

Prevalent in genres like rock, metal, blues
Clipping Functions

- **Soft Clip**
  - Function: Sine
  - Function: Quadratic

- **Hard Clip**
  - Function: Linear
  - Function: Exponential
  - Function: Inverse

Boost = 2
Clipping Function Evaluation

**Hard Clipping**

**Soft Clipping**
Oversampling + Clipping

Hard Clipping

Soft Clipping

Gain $\uparrow5$ Non Linearity $\downarrow5$
Distortion can be **post-filtered** using shelving equalization filters to add “character”

While high and low pass filters are useful for removing unwanted signal above or below a set frequency, **shelving filters** can be used to **reduce or increase signals** above or below a set frequency.
Distortion Block Diagram

Distortion Chain Components

1. **Interpolating IIR Filter**: Upsample and anti-imaging IIR filter
2. **Distortion**: Boost and apply non-linearity
3. **Decimating IIR Filter**: Anti-aliasing IIR filter and downsample
4. **Shelving Filter**: Attenuate frequencies higher than the upper midrange
Distortion: DEMO

Options:
- distortion_chain
- Example
- Generate Options: QT GUI

Parameter:
- ID: samp_rate
  - Num Tabs: 3
  - Label: Sample Rate
  - Value: 64,1k
  - Type: Float

Parameter:
- ID: param_isolated
  - Label: Boost
  - Value: 1
  - Type: Float

Parameter:
- ID: param_mix
  - Label: Wet/Dry Mix
  - Value: 500m
  - Type: Float

Parameter:
- ID: param_enabled
  - Label: Enabled
  - Value: False
  - Type: Float

Parameter:
- ID: param_skip
  - Label: Skip
  - Value: 0
  - Type: Float

Parameter:
- ID: param_dial
  - Label: Dial
  - Value: Linear
  - Type: String

Parameter:
- ID: param_cutoff
  - Label: Cutoff Frequency
  - Value: 4k
  - Type: Float

Variable:
- ID: f挹�x5
  - Value: [1.0, 0.2915602882]

Variable:
- ID: f挹�x5
  - Value: [1.0, 0.2915602882]

Wave File Source:
- File: samplesample_1.wav
  - Repeat: No

Interpolating IIR Filter:
- Interpolation: 5
  - Feedback Taps: fl_taps, fl

Multiply Const Constant: 1

Distortion Function: 1
- Boost: 3
- Wet/Dry Mix (gamma): 500m

Decimating IIR Filter:
- Feedback Taps: fl_taps, fl

Shaping Filter:
- Sample Rate: 64,1k
  - Filter Type: High-Shelf
  - PowerBand Gain (dB): 0
  - Cutoff Frequency (Hz): 4k

Audio Sink:
- Sample Rate: 44,1 kHz

QT GUI Time Sink:
- Name: Original Signal
  - Number of Points: 22,05k
  - Sample Rate: 44,1k
  - Autocor: No

QT GUI Waterfall Sink:
- Name: Original Signal
  - FFT Size: 1,024k
  - Center Frequency (Hz): 0
  - Bandwidth (Hz): 44,1k

QT GUI Frequency Sink:
- Name: Original Signal
  - FFT Size: 1,024k
  - Center Frequency (Hz): 0
  - Bandwidth (Hz): 44,1k

QT GUI Check Box:
- ID: ok
  - Label: Enabled
  - Value: False
  - Type: Float
Takeaways
Educational Value

- Easy to **perceive a DSP algorithm** when you hear it
  - That’s how I learned my DSP (my background is in computer architecture, not DSP or Comms)
- The effects I presented are **not complicated** but they cover several **fundamental** concepts
  - Easy to present in an educational setting
- Implementing them in GNU Radio was **extremely easy**
  - No need for additional hardware to “graduate” from simulation. Every computer has a sound card.
  - Started with Python, moved to C++ because Python is not real-time (even for 44.1 kHz)
- Focus on the **DSP**, not the plumbing or data movement.
Thank You!
gr-guitar
https://github.com/achaudhari/gr-guitar