MUS420 Lecture
Elementary Digital Waveguide Models for Vibrating Strings

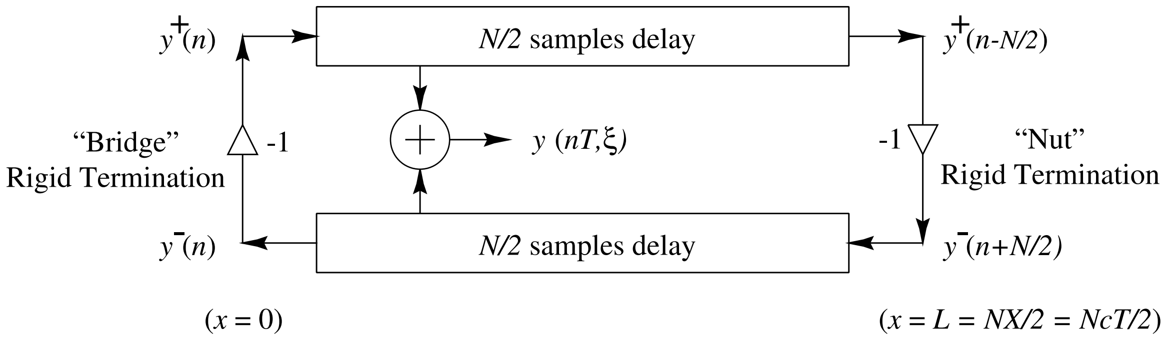
Julius O. Smith III (jos@ccrma.stanford.edu)
Center for Computer Research in Music and Acoustics (CCRMA)
Department of Music, Stanford University
Stanford, California 94305

June 27, 2020

Outline

- Terminated string
- Plucked and struck string
- Damping and dispersion
- String Loop Identification
- Nonlinear "overdrive" distortion

Rigidly Terminated Ideal String



- Reflection *inverts* for displacement, velocity, or acceleration waves (proof below)
- Reflection *non-inverting* for slope or force waves

Boundary conditions:

$$y(t,0) \equiv 0 \qquad y(t,L) \equiv 0 \qquad (L = \text{string length})$$

Expand into Traveling-Wave Components:

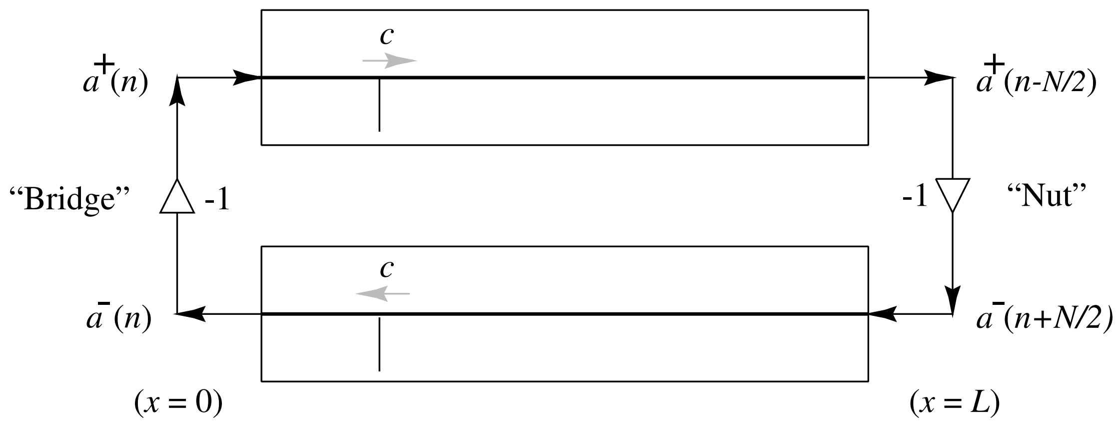
$$\begin{aligned} y(t,0) &= y_r(t) + y_l(t) = y^+(t/T) + y^-(t/T) \\ y(t,L) &= y_r(t - L/c) + y_l(t + L/c) \end{aligned}$$

Solving for outgoing waves gives

$$\begin{aligned} y^+(n) &= -y^-(n) \\ y^-(n + N/2) &= -y^+(n - N/2) \end{aligned}$$

$N \stackrel{\Delta}{=} 2L/X =$ *round-trip* propagation time in samples

Acceleration-Wave Simulation



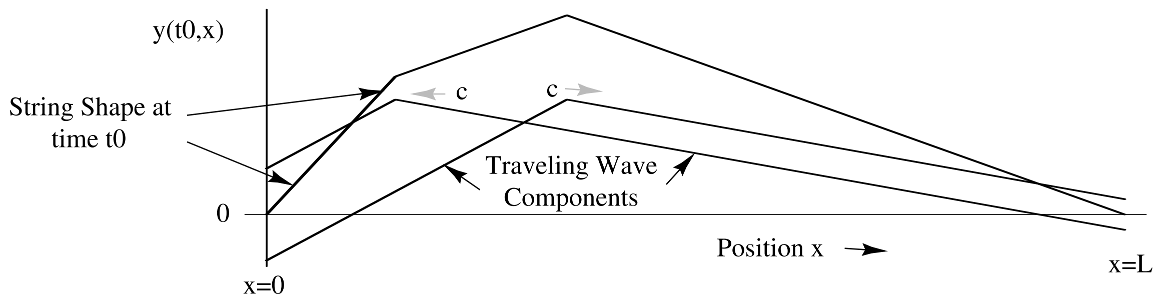
Initial conditions for the ideal plucked string: acceleration or curvature waves.

Recall:

$$\boxed{y'' = \frac{1}{c^2}\ddot{y}}$$

Acceleration waves are proportional to "curvature" waves.

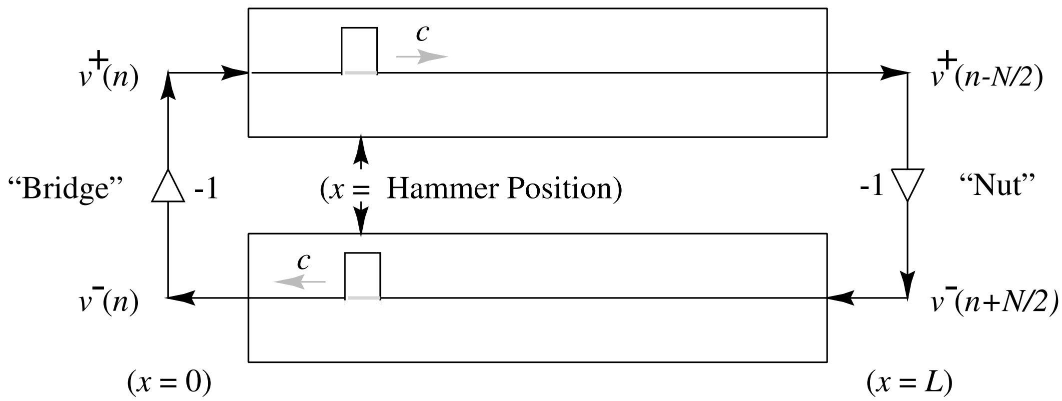
Doubly Terminated Ideal Plucked String



A doubly terminated string, "plucked" at 1/4 its length.

- Shown short time after pluck event.
- Traveling-wave components and physical string-shape shown.
- Note traveling-wave components sum to zero at terminations. (Use image method.)

Ideal Struck-String Velocity-Wave Simulation

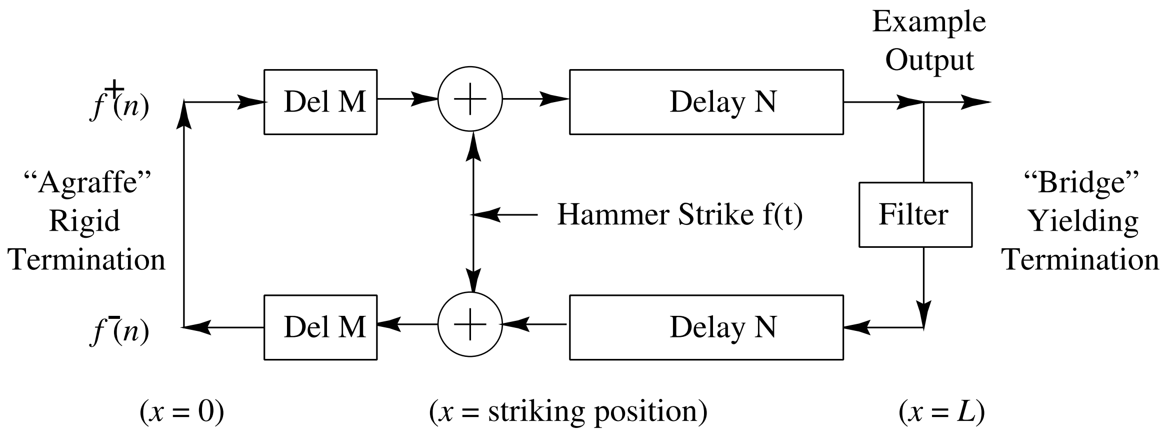


Initial conditions for the ideal struck string in a *velocity wave* simulation.

Hammer strike = *momentum transfer* = velocity step:

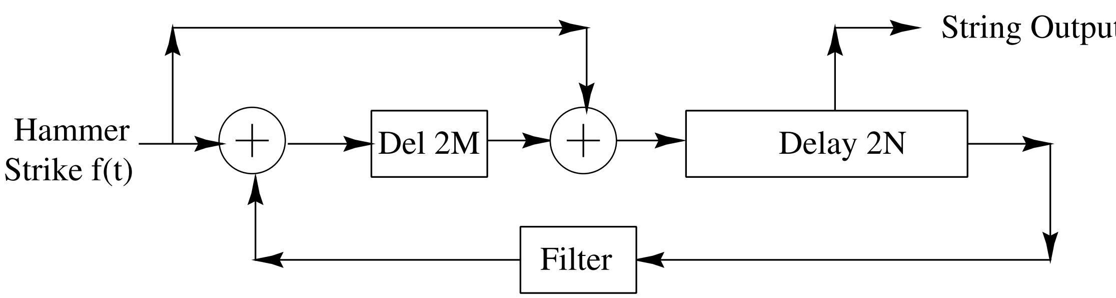
$$m_h v_h(0-) = (m_h + m_s) v_s(0+)$$

External String Excitation at a Point



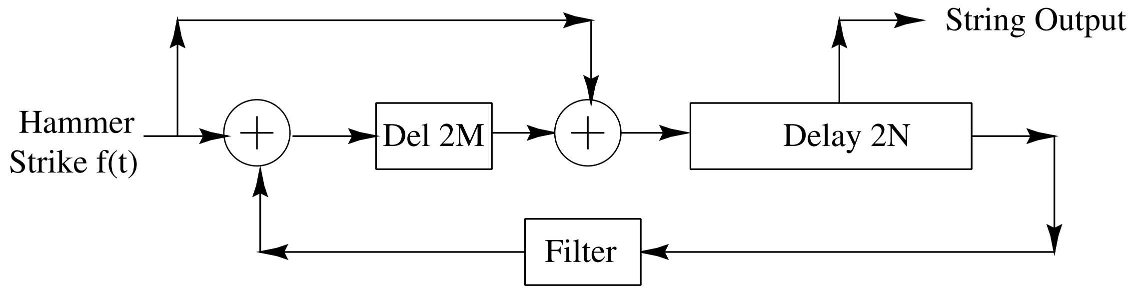
"Waveguide Canonical Form"

Equivalent System: Delay Consolidation

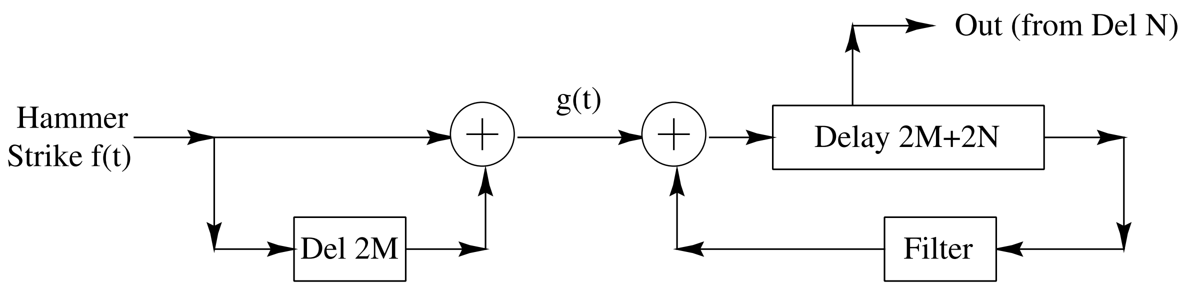


Finally, we "pull out" the comb-filter component:

Delay Consolidated System (Repeated):

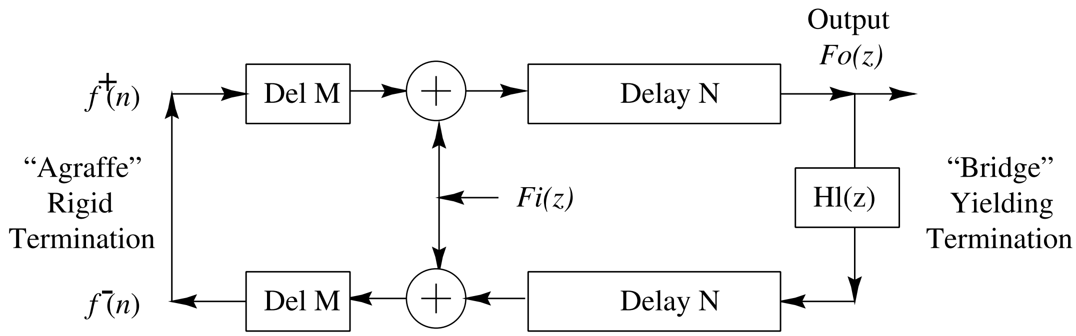


Equivalent System: FFCF Factored Out:



- Extra memory needed.
- Output "tap" can be moved to delay-line output.

Algebraic Derivation

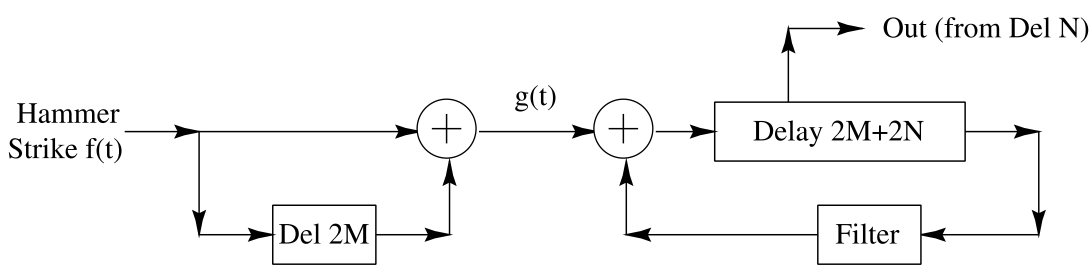


By inspection:

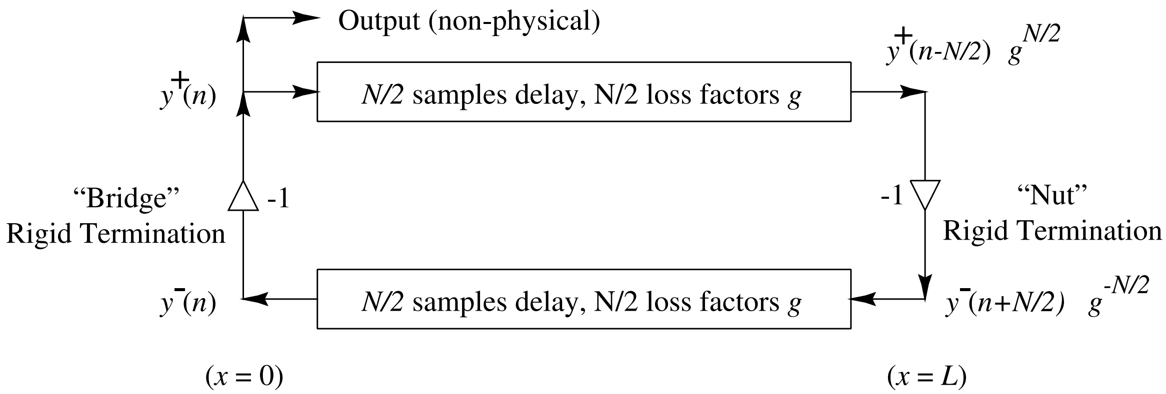
$$F_o(z) = z^{-N}\left\{F_i(z) + z^{-2M}\left[F_i(z) + z^{-N}H_l(z)F_o(z)\right]\right\}$$

$$\Rightarrow \quad H(z) \stackrel{\Delta}{=} \frac{F_o(z)}{F_i(z)} = z^{-N}\frac{1+z^{-2M}}{1-z^{-(2M+2N)}H_l(z)}$$

$$= \left(1+z^{-2M}\right)\frac{z^{-N}}{1-z^{-(2M+2N)}H_l(z)}$$



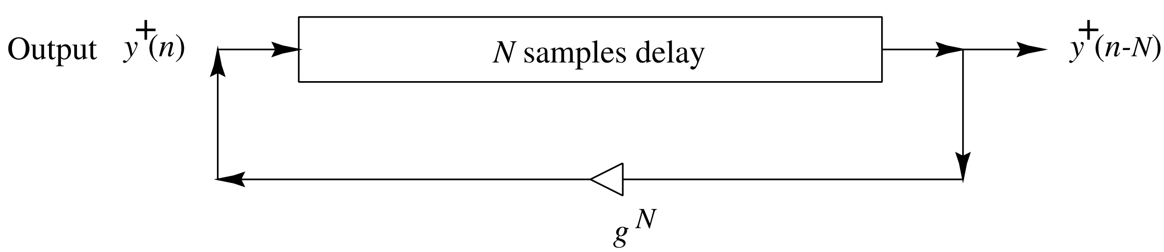
Damped Plucked String



Rigidly terminated string with distributed resistive losses.

- N loss factors g are embedded between the delay-line elements.

Equivalent System: Gain Elements Commuted



All N loss factors g have been "pushed" through delay elements and combined at a *single* point.

Computational Savings

- $f_s = 50$kHz, $f_1 = 100Hz \Rightarrow$ delay $= 500$
- Multiplies reduced by two orders of magnitude
- Input-output transfer function unchanged
- Round-off errors reduced

Frequency-Dependent Damping

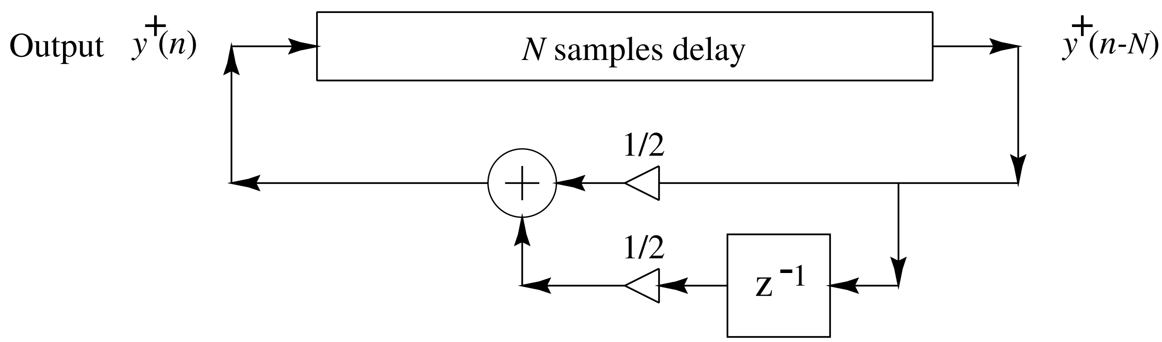
- Loss factors g should really be digital filters
- Gains in nature typically decrease with frequency
- Loop gain may not exceed 1 (for stability)
- Gain filters commute with delay elements (LTI)
- Typically only *one* gain filter used per loop

Simplest Frequency-Dependent Loop Filter

$$\boxed{H_l(z) = b_0 + b_1 z^{-1}}$$

- Linear phase $\Rightarrow b_0 = b_1$ ($\Rightarrow$ delay = 1/2 sample)
- Zero damping at dc $\Rightarrow b_0 + b_1 = 1$
  $\Rightarrow b_0 = b_1 = 1/2$
  $\Rightarrow$
  $$\boxed{H_l(e^{j\omega T}) = \cos\left(\omega T/2\right), \quad |\omega| \leq \pi f_s}$$

Karplus-Strong Algorithm



- To play a note, the delay line is initialized with random numbers ("white noise")

KS Physical Interpretation

- Rigidly terminated ideal string with the simplest damping filter
- Damping consolidated at one point and replaced by a one-zero filter approximation
- String *shape* = *sum* of upper and lower delay lines
- The *difference* of upper and lower delay lines corresponds to a nonzero initial string *velocity*. To show this, recall that $f \triangleq -Ky'$ so that

  $$y' = -\frac{1}{K}(f^{+}+f^{-}) = -\frac{R}{K}(v^{+}-v^{-}) = \frac{1}{c}(v^{-}-v^{+})$$

  implying

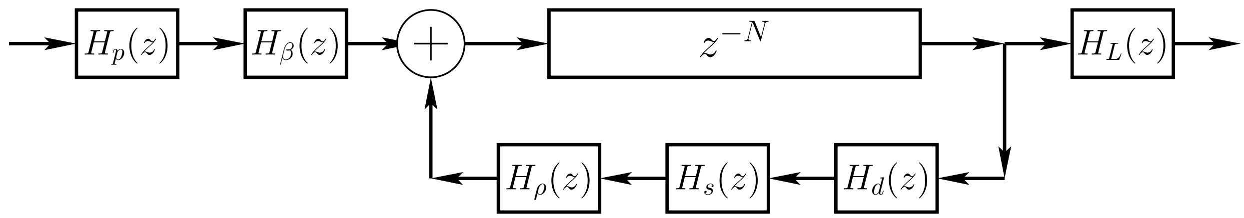
  $$\boxed{v^{+} = -c(y^{+})' \qquad v^{-} = c(y^{-})'}$$

- Karplus-Strong string is both "plucked" and "struck" by random amounts along entire length of string!
- A "splucked" string?

KS Sound Examples

- “Vintage” 8-bit sound examples:
  - Original Plucked String: (AIFF) (MP3)
  - Drum: (AIFF) (MP3)
  - Stretched Drum: (AIFF) (MP3)
- STK Plucked String: (WAV) (MP3)
- Extended KS (EKS) Scale: (WAV) (MP3)

Extended Karplus-Strong (EKS) Algorithm



$$
\begin{aligned}
N &= \text{pitch period } (2\times \text{ string length) in samples} \\
H_p(z) &= \frac{1-p}{1-p\,z^{-1}} = \text{pick-direction lowpass filter} \\
H_\beta(z) &= 1 - z^{-\beta N} = \text{pick-position comb filter, } \beta \in (0,1) \\
H_d(z) &= \text{string-damping filter (one/two poles/zeros typical)} \\
H_s(z) &= \text{string-stiffness allpass filter (several poles and zeros)} \\
H_\rho(z) &= \frac{\rho(N) - z^{-1}}{1-\rho(N)\,z^{-1}} = \text{first-order string-tuning allpass filter} \\
H_L(z) &= \frac{1-R_L}{1-R_L\,z^{-1}} = \text{dynamic-level lowpass filter}
\end{aligned}
$$

EKS Sound Example

Bach A-Minor Concerto—Orchestra Part: (WAV) (MP3)

- Executes in real time on one Motorola DSP56001 (20 MHz clock, 128K SRAM)
- Developed for the NeXT Computer introduction at Davies Symphony Hall, San Francisco, 1989
- Solo violin part was played live by Dan Kobialka of the San Francisco Symphony

Simplest Frequency-Dependent Loss

Recall that the two-point average used in the Karplus-Strong algorithm can be interpreted as the simplest possible frequency-dependent loss filter for the otherwise ideal vibrating string:

$$H_l(z) = \frac{1+z^{-1}}{2}$$

Next Simplest Case: Length 3 FIR Loop Filter

$$\boxed{H_l(z) = b_0 + b_1 z^{-1} + b_2 z^{-2}}$$

- Linear phase $\Rightarrow b_0 = b_2$ ($\Rightarrow$ delay $=$ 1 sample)
- Unity dc gain $\Rightarrow b_0 + b_1 + b_2 = 2b_0 + b_1 = 1 \Rightarrow$

$$H_l(e^{j\omega T}) = e^{-j\omega T}\left[(1-2b_0) + 2b_0\cos(\omega T)\right]$$

- Remaining degree of freedom $=$ *damping control*

Length 3 FIR Loop Filter with Variable DC Gain

Relaxing the unity-dc-gain restriction, but keeping linear phase, we have

$$H_l(z) = b_0 + b_1 z^{-1} + b_0 z^{-2} \qquad \text{(linear phase)}$$

We can use the remaining two degrees of freedom for *brightness* B & *sustain* S:

$$\begin{aligned} g_0 &\stackrel{\Delta}{=} e^{-6.91P/S} \\ b_0 &= g_0(1-B)/4 = b_2 \\ b_1 &= g_0(1+B)/2 \end{aligned}$$

where

$$\begin{aligned} P &= \text{period in seconds (total loop delay)} \\ S &= \text{desired sustain time in seconds} \\ B &= \text{brightness parameter in the interval } [0,1] \end{aligned}$$

Sustain time S is defined here as the time to decay 60 dB (or 6.91 time-constants) when brightness B is maximum ($B=1$). At minimum brightness ($B=0$), we have

$$|H_l(e^{j\omega T})| = g_0 \frac{1+\cos(\omega T)}{2} = g_0 \cos^2(\omega T)$$

Loop Filter Identification

For loop-filter design, we wish to minimize the error in

- partial decay time (set by amplitude response)
- partial overtone tuning (set by phase response)

Simple and effective method (MUS421 style):

- Estimate pitch (elaborated next page)
- Set Hamming FFT-window length to four periods
- Compute the short-time Fourier transform (STFT)
- Detect peaks in each spectral frame
- Connect peaks through time (amplitude envelopes)
- Amplitude envelopes must decay *exponentially*
- On a dB scale, exponential decay is a *straight line*
- Slope of straight line determines decay time-constant
- Can use 1st-order `polyfit` in Matlab or Octave
- For beating decay, connect amplitude envelope peaks
- Decay rates determine ideal amplitude response
- Partial tuning determines ideal phase response

Plucked/Struck String Pitch Estimation

- Take FFT of middle third of plucked string tone
- Detect spectral peaks
- Form histogram of peak spacing Δf_i
- Pitch estimate $\hat{f}_0 \triangleq$ most common spacing Δf_i
- Refine $\hat{f}_0$ with gradient search using harmonic comb:

$$\begin{aligned}\hat{f}_0 &\triangleq \arg\max_{\hat{f}_0} \sum_{k=1}^{K} \log\left|X(k\hat{f}_0)\right| \\ &= \arg\max_{\hat{f}_0} \prod_{k=1}^{K} \left|X(k\hat{f}_0)\right|\end{aligned}$$

where

K = number of peaks, and
k = harmonic number of kth peak
(valid method for non-stiff strings)

Must skip over any missing harmonics,
i.e., omit k whenever $|X(k\hat{f}_0)| \approx 0$.

The text provides further details and pointers to recent papers on pitch estimation.

Nonlinear "Overdrive"

A popular type of distortion, used in *electric guitars*, is *clipping* of the guitar waveform.

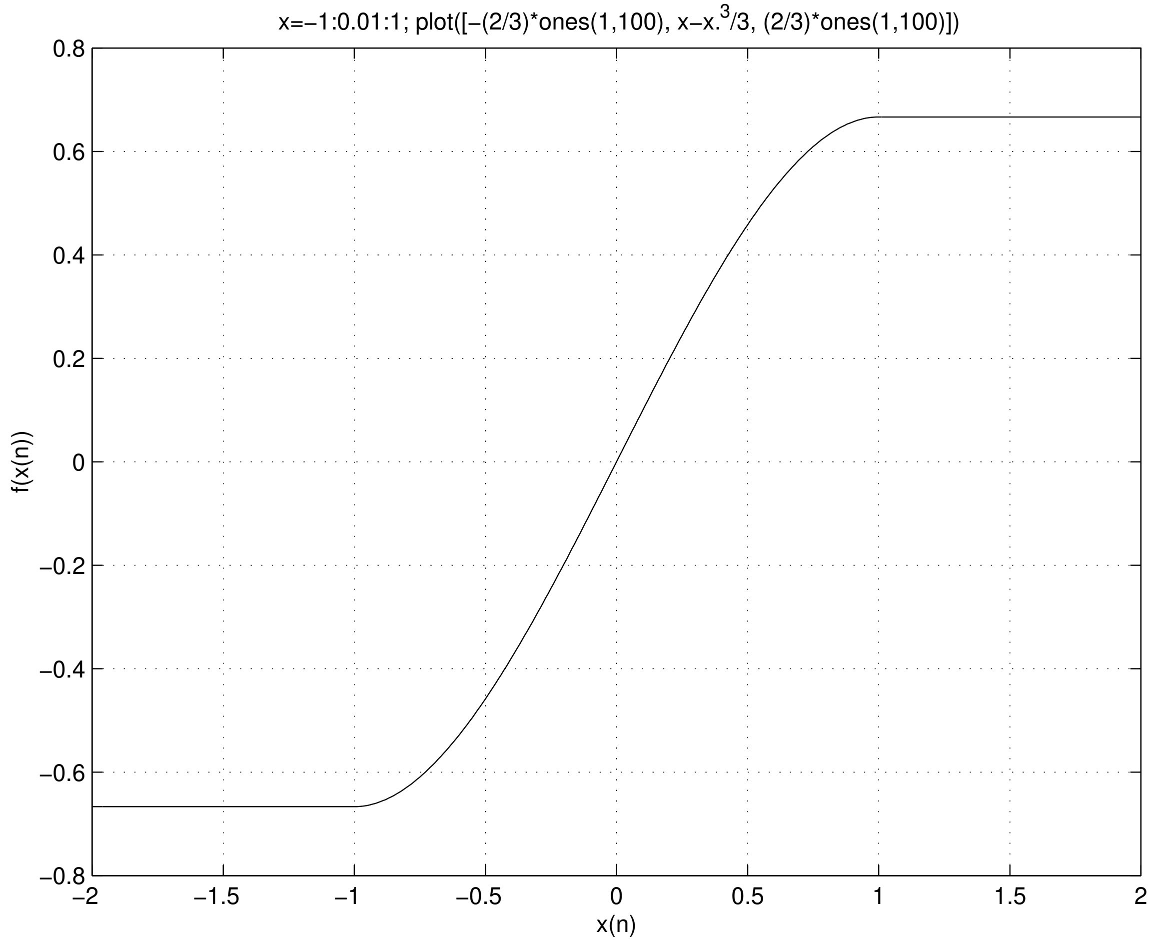
Hard Clipper

$$f(x) = \begin{cases} -1, & x \leq -1 \\ x, & -1 \leq x \leq 1 \\ 1, & x \geq 1 \end{cases}$$

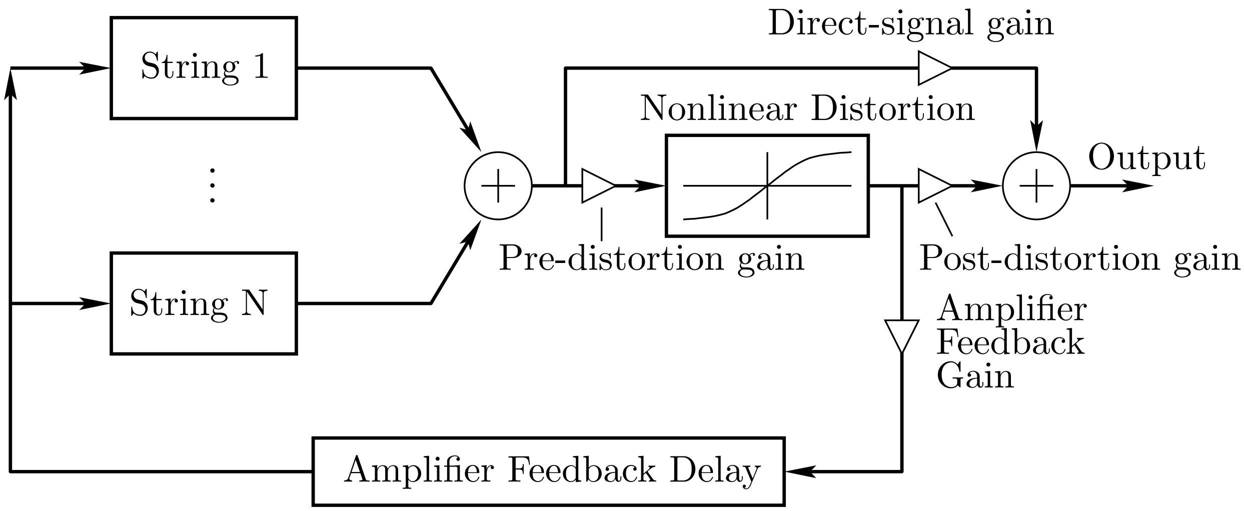
where x denotes the current input sample $x(n)$, and $f(x)$ denotes the output of the nonlinearity.

Soft Clipper

$$f(x) = \begin{cases} -\frac{2}{3}, & x \leq -1 \\ x - \frac{x^3}{3}, & -1 \leq x \leq 1 \\ \frac{2}{3}, & x \geq 1 \end{cases}$$



Amplifier Distortion + Amplifier Feedback



Distorted Electric Guitar with Amplifier Feedback

- Distortion can be preceded and followed by *EQ*
  E.g., integrator "pre" and differentiator "post"
- Distortion output signal often further filtered by an *amplifier cabinet filter*, representing speaker cabinet, driver responses, etc.
- In Class A tube amplifiers, there should be *duty-cycle modulation* as a function of signal level[1]
  - 50% at low levels (no duty-cycle modulation)
  - 55-65% duty cycle observed at high levels
    $\Rightarrow$ even harmonics come in
  - Example: Distortion input can *offset by a constant* (e.g., input RMS level times some scaling)

[1] http://www.trueaudio.com/at_eetjlm.htm