Outline

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

Acceleration-Wave Simulation

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

Recall:
\[
y'' = \frac{1}{c^2} y
\]

Acceleration waves are proportional to “curvature” waves.

Rigidly Terminated Ideal String

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

Boundary conditions:
\[ y(t, 0) \equiv 0 \quad y(t, L) \equiv 0 \quad (L = \text{string length}) \]

Expand into Traveling-Wave Components:
\[
y(t, 0) = y_+/(t/T) + y_-(t/T) \\
y(t, L) = y_+(t - L/c) + y_-(t + L/c)
\]

Solving for outgoing waves gives
\[
y_+(n) = -y_-(n) \\
y_-(n + N/2) = -y_+(n - N/2)
\]

\[ N \triangleq 2L/X = \text{round-trip propagation time in samples} \]

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:
\[ F_o(z) = z^{-N} \left\{ F_i(z) + z^{-2M} \left[ F_i(z) + z^{-N} H_i(z) F_o(z) \right] \right\} \]
\[ H(z) = \frac{F_o(z)}{F_i(z)} = z^{-N} \frac{1 + z^{-2M}}{1 - z^{-2M-2N} H_i(z)} \]
\[ = \left( 1 + z^{-2M} \right) \frac{z^{-N}}{1 - z^{-2M-2N} H_i(z)} \]
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.

### 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**

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

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

### Computational Savings

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

### 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 \( f = -K y' \) so that
  \[
y' = -\frac{1}{K} (f^+ + f^-) = -\frac{R}{K} (v^+ - v^-) = \frac{1}{c} (v^- - v^+)
  \]
  implying
  \[
v^+ = -c(v^+)' \quad v^- = c(v^-)'
  \]
- Karplus-Strong string is both “plucked” and “struck” by random amounts along entire length of string!
- A “splucked” string?

Extended Karplus-Strong (EKS) Algorithm

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

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)

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

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

- Linear phase \( \Rightarrow b_0 = b_2 \) (\( \Rightarrow \) delay = 1 sample)
- Unity dc gain \( \Rightarrow b_0 + b_1 + b_2 = 2b_0 + b_1 = 1 \) \( \Rightarrow \)
  \[ H_l(e^{j\omega T}) = e^{-j\omega T}[(1 - 2b_0) + 2b_0 \cos(\omega T)] \]
- 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

\[ H_l(z) = b_0 + b_1 z^{-1} + b_0 z^{-2} \] (linear phase)

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

\[ g_0 \Delta = e^{-0.91P/S} \]
\[ b_0 = g_0(1 - B)/4 = b_2 \]
\[ b_1 = g_0(1 + B)/2 \]

where

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

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

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

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 \( \Delta f_i \)
- Pitch estimate \( \hat{f}_0 \) most common spacing \( \Delta f_i \)
- Refine \( \hat{f}_0 \) with gradient search using harmonic comb:
  \[ \hat{f}_0 \Delta = \arg \max_{f_0} \sum_{k=1}^{K} \log |X(k\hat{f}_0)| \]
  \[ = \arg \max_{f_0} \prod_{k=1}^{K} |X(k\hat{f}_0)| \]
  where
  \[ K = \text{number of peaks, and} \]
  \[ k = \text{harmonic number of } k \text{th peak} \]
  (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**

\[
 f(x) = \begin{cases} 
 -\frac{2}{3}, & x \leq -1 \\ 
 x - \frac{x^3}{4}, & -1 \leq x \leq 1 \\ 
 \frac{2}{3}, & x \geq 1 
\end{cases}
\]

![Graph of Soft Clipper](image)