MUS420 Supplement
Modal Synthesis of a Piano String

Stefan Bilbao and 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

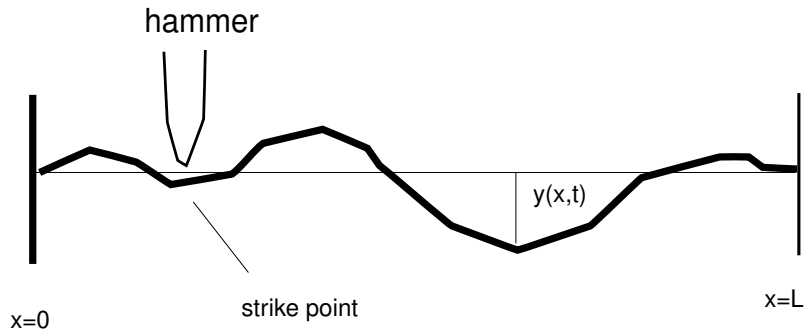
February 5, 2019

Outline:

- Piano String Modeling
- Finite Difference Scheme for the Piano String
- Listening to the String
- State Space Formulation
- Diagonalizing to Modal Coordinates
- Adding the Nonlinear Hammer
- Features of Modal State-Space Coordinates

Piano String Modeling

Physical picture:



Chaigne and Askenfelt (JASA, Feb. 1995) suggested the following partial differential equation (PDE) to model the piano string:

$$\frac{\partial^2 y}{\partial t^2} = c^2 \frac{\partial^2 y}{\partial x^2} - \epsilon c^2 L^2 \frac{\partial^4 y}{\partial x^4} - 2b_1 \frac{\partial y}{\partial t} + 2b_3 \frac{\partial^3 y}{\partial t^3} + f(x, y, t)$$

In addition, we must specify

- *boundary conditions*
- *initial conditions* (usually zero position and velocity).

Piano String Model, Continued

$$\frac{\partial^2 y}{\partial t^2} = c^2 \frac{\partial^2 y}{\partial x^2} - \epsilon c^2 L^2 \frac{\partial^4 y}{\partial x^4} - 2b_1 \frac{\partial y}{\partial t} + 2b_3 \frac{\partial^3 y}{\partial t^3} + f(x,y,t)$$

- $y(t,x)$ represents *transverse displacement* along the string at time t seconds and position m meters
- L is the string *length* in meters
- The PDE takes into account
  - *dispersion* (fourth space derivative)
  - frequency-dependent *loss* from 3rd time derivative
  - *hammer excitation* denoted by $f(x,y,t)$
- A derivation of the wave equation for stiff strings is given in Morse 1948 and in many acoustics texts. (Cremer 1984 is especially good.) (See the Course Bibliography[1] for full citations.)

[1] http://ccrma.stanford.edu/~jos/refs420/

Second-Order Finite Difference Schemes

The simplest (and traditional) way of discretizing the 1D wave equation is by replacing first derivatives by first-order differences

$$\left.\frac{\partial y}{\partial t}\right|_{x=mX,t=nT} \simeq \frac{y_m^n - y_m^{n-1}}{T}$$
$$\left.\frac{\partial y}{\partial x}\right|_{x=mX,t=nT} \simeq \frac{y_m^n - y_m^{n-1}}{X}$$

and second derivatives by second-order differences

$$\left.\frac{\partial^2 y}{\partial t^2}\right|_{x=mX,t=nT} \simeq \frac{y_m^{n-1} - 2y_m^n + y_m^{n+1}}{T^2}$$
$$\left.\frac{\partial^2 y}{\partial x^2}\right|_{x=mX,t=nT} \simeq \frac{y_{m-1}^n - 2y_m^n + y_{m+1}^n)}{X^2}$$

and so on, where $y_m^n \stackrel{\Delta}{=} y(nT, mX)$ are the *grid variables*. Note that we have *uniformly sampled* the time-space plane, with timestep T (the *temporal sampling interval*) and space step X (the *spatial sampling interval*).

FDS for the Ideal String

Consider first the *ideal* (lossless, dispersionless, unforced) string wave equation:

$$\frac{\partial^2 y}{\partial t^2} = c^2 \frac{\partial^2 y}{\partial x^2}$$

Replacing the second-derivatives by their finite-difference approximations gives

$$y_m^{n-1} - 2y_m^n + y_m^{n+1} = \frac{c^2 T^2}{X^2}(y_{m-1}^n - 2y_m^n + y_{m+1}^n)$$

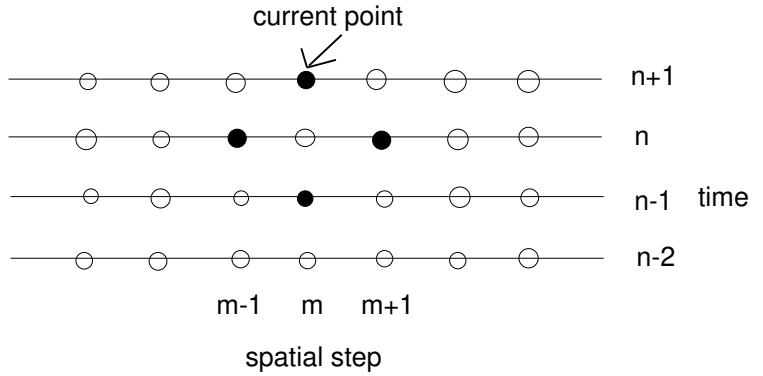
If we choose $X = cT$ (which is most natural physically), the equation reduces further to

$$y_m^{n+1} = y_{m-1}^n + y_{m+1}^n - y_m^{n-1}$$

Let's examine this recursion on the time-space grid, assuming for the moment no boundary conditions:

Time-Space Grid for the Second-Order Ideal String FDS

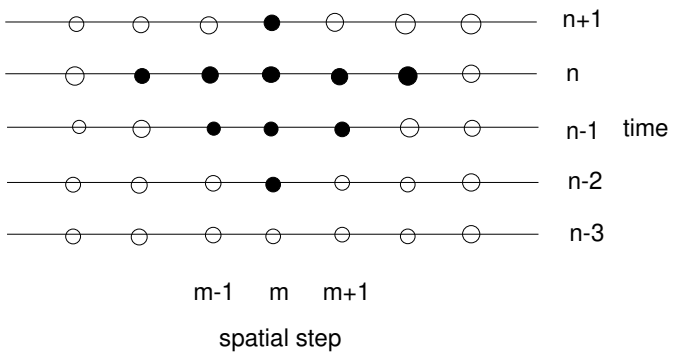
$$y_m^{n+1} = y_{m-1}^n + y_{m+1}^n - y_m^{n-1}$$



- Grid variable at "current" point depends on value at *two* previous time steps (a *second order* scheme in time). Our initial conditions must therefore specify $y(n, m)$ for all m at times $n = 0$ and $n = 1$.
- Grid variable at "current" point depends on values at adjacent locations on the string (at previous time).
- Difference scheme is *explicit* (thus *parallelizable*); that is, each grid variable at time $n + 1$ depends only on grid variables at previous time instants.

Piano String Finite Difference Scheme

The Chaigne and Askenfelt PDE is discretized in space and time to yield a recursive difference scheme with the following space-time "stencil":



- 3rd order in time, 4th order in space
- *Explicit*, enabling recursive time updates

The FDS can be written in matrix form as

$$\mathbf{y}_n = \mathbf{A_1 y_{n-1}} + \mathbf{A_2 y_{n-2}} + \mathbf{A_3 y_{n-3}} + \mathbf{g} f_n(\mathbf{y}_{n-1}[i_h], y^h_{n-1})$$

where

- The column vector $\mathbf{y}_n$ contains the displacements of the N nodes at time step n, i.e., $\mathbf{y}_n[i] = y^n_i$
- $\mathbf{A_1 A_2}$ and $\mathbf{A_3}$ are N by N matrices

Piano String FDS, Continued

We have

$$\mathbf{y}_n = \mathbf{A_1 y_{n-1}} + \mathbf{A_2 y_{n-2}} + \mathbf{A_3 y_{n-3}} + \mathbf{g} f_n(\mathbf{y}_{n-1}[i_h], y^h_{n-1})$$

The string displacement is updated in time by three matrix-vector products, plus a *forcing term* $\mathbf{g} f_n(\mathbf{y}_{n-1}[i_h], y^h(n-1))$:

- The length $N \times 1$ matrix $\mathbf{g}$ represents the *shape of the hammer excitation* over the length of the string. The nonzero elements correspond to the width of the hammer relative to that of the string. For greatest accuracy, it should be time varying, but it may be approximated by a constant shape.
- The element $\mathbf{y}_n[i_h]$ is the string displacement sample closest to the *hammer position*.
- The scalar function y^h_n denotes *hammer position* at time n.
- The scalar function $f_n(\mathbf{y}_n[i_h], y^h_n)$ sets the amplitude of the hammer force distribution across position at time n.

- The force exerted on the string by the hammer is a *nonlinear function of the hammer-string separation* $y_n^h - \mathbf{y}_n[i_h]$.
- The time evolution of the hammer must be computed numerically by a separate finite difference scheme. Typical models include a classical point mass, a *nonlinear spring* (which gets stiffer when compressed), and a very small amount of damping.

Note that the FDS involves 3 steps of "lookback." It turns out this model goes *unstable* in the limit as the sampling rate goes to infinity!

- There is no problem at any normal audio sampling rate—the sampling rate must be on the order of hundreds of megahertz to trigger the instability.
- The model can be stabilized at all sampling rates by using instead a *two-time-step* scheme.
- Watch for forthcoming publications by Stefan Bilbao at CCRMA.
- We'll talk about stability of finite difference schemes later.

Listening to the String

- Chaigne and Askenfelt compute string displacement at every node on the string, at each time step. This is useful for visualizations, etc.
- For sound synthesis, we need only "listen" to the string at the *"bridge"*. The *bridge* is where most of the string vibrational energy couples to the piano soundboard.
- Let's define the output as

$$y_{out} = \mathbf{C_1}\mathbf{y}_n$$

  where $\mathbf{C_1}$ is some constant row vector. A typical choice of $\mathbf{C_1}$ might be $\mathbf{C_1} = [0, \ldots, 0, 1]$, which picks out the last sample of string dispacement nearest the bridge.
- More generally, we could take any linear combination of string displacement samples as the output.

State Space Formulation of the Piano String FDS

- We can now write the entire system, ignoring the nonlinear hammer for the moment, as

$$\begin{aligned} \mathbf{y}_n &= \mathbf{A_1 y_{n-1}} + \mathbf{A_2 y_{n-2}} + \mathbf{A_3 y_{n-3}} \\ y_{out} &= \mathbf{C_1 y_n} \end{aligned}$$

- Collecting the displacements into a *state vector*, we can rewrite this as

$$\begin{bmatrix} \mathbf{y}_n \\ \mathbf{y}_{n-1} \\ \mathbf{y}_{n-2} \end{bmatrix} = \begin{bmatrix} \mathbf{A_1} & \mathbf{A_2} & \mathbf{A_3} \\ \mathbf{I_N} & - & - \\ - & \mathbf{I_N} & - \end{bmatrix} \begin{bmatrix} \mathbf{y}_{n-1} \\ \mathbf{y}_{n-2} \\ \mathbf{y}_{n-3} \end{bmatrix}$$

$$y_{out} = \begin{bmatrix} \mathbf{C_1} & \mathbf{0} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{y}_{n-1} \\ \mathbf{y}_{n-2} \\ \mathbf{y}_{n-3} \end{bmatrix}$$

or

$$\begin{aligned} \mathbf{x_n} &= \mathbf{A x_{n-1}} \\ y_{out} &= \mathbf{C x_n} \end{aligned}$$

Changing to Modal Coordinates

Now the system is in state space form, suppose we change to diagonal coordinates:

$$\begin{aligned} \mathbf{z_n} &= \mathbf{D z_{n-1}} \\ y_{out} &= \mathbf{C' z_n} \end{aligned}$$

where

$$\mathbf{D} = \begin{bmatrix} \begin{bmatrix} \lambda_1 & & & & \\ & \lambda_1^* & & & \\ & & \ddots & & \\ & & & \lambda_N & \\ & & & & \lambda_N^* \end{bmatrix} & & \\ & \begin{bmatrix} \epsilon_1 & & \\ & \ddots & \\ & & \epsilon_N \end{bmatrix} & \\ & & \ddots \end{bmatrix}$$

The N complex-conjugate-pair eigenvalues λ_i represent the *modal frequencies* and *dampings*

$$\lambda_i = R_i e^{j\omega_i T}, \quad 0 < R_i < 1, \quad -\pi < \omega_i T < \pi$$

- The main (desired) effect of the third order partial derivative with respect to time is to make the damping factors R_i be smaller for higher-frequency modes.

- A secondary effect is that the modal frequencies ω_i are slightly shifted.
- Another secondary effect of the third-order time derivative is the introduction of N real rapidly decaying modes, characterized by the ϵ_i. These eigenvalues are typically *negligible*.

We can reduce the size of the system to $2N$, and further change coordinates so that the matrix is made up of N 2 by 2 real "modal" blocks as follows:

$$\mathbf{D}'' = \begin{bmatrix} \begin{bmatrix} 0 & 1 \\ -|\lambda_1|^2 & 2Re(\lambda_1) \end{bmatrix} & & & \\ & \begin{bmatrix} 0 & 1 \\ -|\lambda_2|^2 & 2Re(\lambda_2) \end{bmatrix} & & \\ & & \ddots & \\ & & & \begin{bmatrix} 0 & 1 \\ -|\lambda_N|^2 & 2Re(\lambda_N) \end{bmatrix} \end{bmatrix}$$

Features of Modal State-Space Coordinates

- Input and output locations must be known and fixed
- The ith mode of the system is simulated *in isolation*

$$\begin{aligned} \mathbf{x}_n[i] &= \lambda_i \mathbf{x}_{n-1}[i] + b'_i u_n \\ y_n &= \mathbf{C}' \mathbf{x}_n \end{aligned}$$

  Notes:
  - the input gain b'_i specifies *how the input signal u_n excites mode i*
  - the output state-gain-vector $\mathbf{C}'$ specifies *how all the modes add up (weighted) to form the output*
  - The mode-input gain vector $\mathbf{B}' = [b'_1, \ldots, b'_N]$ *changes if the physical input is moved*
  - $\mathbf{C}'$ *changes if the physical output is moved*
  - The system modes *do not change when the input or output are moved*
- Dispersion modeling (mode tuning) is much easier than in a finite difference approximation
- Frequency-dependent damping is also much easier
- Extends to higher order models (time and/or space)

- Extends simply to spatially varying media (not Fourier based)
- Can also be used for *implicit* finite difference schemes
- Relatively efficient when modes are *inharmonic* (bells, gongs, mallet percussion)
- When mode frequencies are nearly *harmonic* (strings, woodwinds, brasses) *digital waveguide models are more efficient* (we'll take them up later).