Musical Acoustics
Homework Laboratory 3: Modeling techniques

Marin Pasin Davide 10799610
De Bortoli Gian Marco 10805035

A.Y. 2021/2022

POLITECNICO
MILANO 1863

1 Modeling of a piano string

The objective of this study is to implement a finite difference method able to approximate the sound signal produced by the piano string corresponding to the note C2 when hit by the hammer.
The fundamental frequency is $f_0 = 65.4\ Hz$, the length of the string is $L = 1.92\ m$, the overall mass of the string is $M_S = 35 \times 10^{-3}\ kg$, the tension acting on it is $T_e = 750\ N$, its stiffness is $\kappa = 7.5 \times 10^{-6}\ kg/s^2$ and $\rho = M_S/L$ is the linear density.

1.1 Compute the simulation with correct CFL condition

The analytical wave equation for a stiff and lossy string with scalar impedance on both ends and excited by the hammer is:

$$\frac{\partial^2 y}{\partial t^2} = c^2 \frac{\partial^2 y}{\partial x^2} - \kappa^2 \frac{\partial^4 y}{\partial x^4} - 2b_1 \frac{\partial y}{\partial t} + 2b_2 \frac{\partial^3 y}{\partial x^2 \partial t} + \rho^{-1} f(x, x_0, t) \tag{1}$$

where $f(x, x_0, t) = f_H(t)g(x, x_0)$ is the distributed force applied by the hammer on the string.

$$f_H(t) = F_H(t) \bigg(\int_{x_0-\delta x}^{x_0+\delta x} g(x, x_0) dx \bigg)^{-1} \tag{2}$$

is the time history of the force $F_H(t)$ applied on x_0, as if the hammer was a point, while $g(x, x_0)$ is the spatial window, of length x and centered in x_0, that creates the distribution. Inside this study we consider a Hanning window of length $w = 0.2\ m$.
The hammer displacement with respect to the string equilibrium position is the solution of:

$$M_H \frac{d^2\eta}{dt^2} = -F_H(t) - b_H \frac{d\eta}{dt} \tag{3}$$

where $M_H = 4.9 \times 10^{-3}\ kg$ is the mass of the hammer and $b_H = 1 \times 10^{-4}\ s^{-1}$ is the fluid damping coefficient that takes into account the compression of the felt during the motion of the hammer.
The hammer can be modelled as a lumped mass attached to a non-linear mass and its force is given by:

$$F_H(t) = K\xi^p = K|\eta(t) - y(x_0, t)|^p \tag{4}$$

where $K = 4 \times 10^8\ kg/s^2$ is the stiffness of the hammer, ξ is the time dependent function describing the distance between the hammer and the string and $p = 2.3$ is the stiffness exponent responsible of representing how the stiffness changes with the force due to the felt compression.

The finite difference model is an algorithm based on time and space discretization of the problem: at each time step the displacement of a point $\hat{x}$ on the string is computed with a linear combination of the values of displacements of some points in a neighbourhood around $\hat{x}$ from previous time instants.
The Courant-Friedrichs-Lewy condition guarantees that the chosen time resolution is suitable for the chosen space resolution, or viceversa. This problem was addressed with a time resolution of:

$$f_s = 4 \times 44.1\ kHz \tag{5}$$

Based on this value, we can assess the maximum number of points for the space discretization as:

$$N_{max} = \left[\frac{-1 + (1 + 16\epsilon\gamma^2)^{\frac{1}{2}}}{8\epsilon} \right]^{\frac{1}{2}} \tag{6}$$

where $\gamma = \frac{f_s}{2f_0}$.
We implemented the algorithm inside the Matlab software, and defined some constants as follows:

$T = 1/f_s$	Time step
$X = L/N_{max}$	spacial step
$N = signal\,duration \times f_s$	total number of time samples
$M = \lfloor N_{max} \rfloor$	total number of space samples

The displacement of a point m on the string at a time instant n is given by:

$$y_m^{n+1} = a_1(y_{m+2}^n + y_{m-2}^n) + a_2(y_{m+1}^n + y_{m-1}^n) + a_3 y_m^n \tag{7}$$

$$+ a_4 y_m^{n-1} + a_5(y_{m+1}^{n-1} + y_{m-1}^{n-1}) + a_F F_m^n$$

where the coefficients a_i are:

$$a_1 = \frac{-\lambda^2\mu}{1 + b_1 T}$$
$$a_2 = \frac{\lambda^2 + 4\lambda^2\mu + \nu}{1 + b_1 T}$$
$$a_3 = \frac{2 - 2\lambda^2 - 6\lambda^2\mu - 2\nu}{1 + b_1 T}$$
$$a_4 = \frac{-1 + b_1 T + 2\nu}{1 + b_1 T}$$
$$a_5 = \frac{-\nu}{1 + b_1 T}$$
$$a_F = \frac{T^2/\rho}{1 + b_1 T}$$

$\lambda = cT/X$ is the Courant number and $\mu = \kappa^2/c^2X^2$ and $\nu = 2b_2T/X^2$ are convenient variables all dependent from both time and space resolution. Inside λ and μ, the considered speed of the wave is given by $c = \sqrt{T_e/\rho}$, while $b_1 = 0.5$ is the air damping coefficient and $b_2 = 6.25 \times 10^{-9}$ is the string internal friction coefficient.
Inside equation (7) the term $F_m^n = F_H(n)g(m, m_0)$ represents the force of the hammer acting on the string.
The indexes m and n should range from 1 to M and from 1 to N respectively (the index $n = 1$ corresponds to the initial time instant $t = 0$ and the index $n = 2$ corresponds to $t = T$, the same reasoning is valid for the index m), but, as shown from subscripts and superscripts in equation (7), the computation of y_m^n cannot take place for values of $n < 3$ and $m < 3$ or $m > M - 2$. This means that for the first time instants and for the two ends of the string we need to establish other relations.
First of all let's impose the conditions for the extremities of the string:

- $m = 2$:

$$y_m^{n+1} = a_1(y_{m+2}^n - y_m^n + 2y_{m-1}^n) + a_2(y_{m+1}^n + y_{m-1}^n) \tag{8}$$

$$+ a_3 y_m^n + a_4 y_m^{n-1} + a_5(y_{m+1}^{n-1} + y_{m-1}^{n-1}) + a_F F_m^n$$

- $m = 1$:

$$y_m^{n+1} = b_{L_1} y_m^n + b_{L_2} y_{m+1}^n + b_{L_3} y_{m+2}^n + b_{L_4} y_m^{n-1} + b_{L_F} F_m^n \tag{9}$$

- $m = M - 1$

$$y_m^{n+1} = a_1(2y_{m+1}^n - y_m^n + y_{m-2}^n) + a_2(y_{m+1}^n + y_{m-1}^n) \tag{10}$$

$$+ a_3 y_m^n + a_4 y_m^{n-1} + a_5(y_{m+1}^{n-1} + y_{m-1}^{n-1}) + a_F F_m^n$$

- $m = M$

$$y_m^{n+1} = b_{R_1} y_m^n + b_{R_2} y_{m-1}^n + b_{R_3} y_{m-2}^n + b_{R_4} y_m^{n-1} + b_{L_F} F_m^n \tag{11}$$

These equations take into account the boundary conditions: the string is hinged at both ends, so its displacement at the extremities is zero. On the left end the normalized impedance is $\zeta_l = 1 \times 10^{20}\ \Omega/kgm^{-2}s^{-1}$, while on the right end, in correspondence of the bridge, the normalized impedance is $\zeta_2 = 1000\ \Omega/kgm^{-2}s^{-1}$. The coefficients b_{R_i} are grouped into the following table:

$$b_{R_1} = \frac{2 - 2\lambda^2\mu + 2\lambda^2}{1 + b_1 T + \zeta_b\lambda}$$
$$b_{R_2} = \frac{4\lambda^2\mu + 2\lambda^2}{1 + b_1 T + \zeta_b\lambda}$$
$$b_{R_3} = \frac{-2\lambda^2\mu}{1 + b_1 T + \zeta_b\lambda}$$
$$b_{R_4} = \frac{-1 + b_1 T + \zeta_b\lambda}{1 + b_1 T + \zeta_b\lambda}$$
$$b_{R_F} = \frac{T^2/\rho}{1 + b_1 T + \zeta_b\lambda}$$

The coefficients b_{L_i} are obtained by substituting into the previous ones ζ_b with ζ_l.
Now we want to compute the term $F_m^n = F_H(n)g(m, m_0)$.
Since the window is limited in space, in order to compute the algorithm inside Matlab we defined a vector G with dimension M and we positioned the window $g(m, m_0)$ inside that vector centered at the index m_0 (which is found knowing that the hammer is placed with its center at the position $x_0 = a/L$ with $a = 0.12$). In this way, at every given time instant n, F_m^n has the same spacial dimension of the string and can be directly summed into the equation (7). The module of the force is, from theory, obtained by equation (4). Inside the script this equation is translated into:

$$F_H(n) = K|\eta^n - y_{m_0}^n|^p \tag{12}$$

where we can see that the displacement of the hammer $\eta(n)$ at the same time instant is required. But the computation of the displacement of the hammer at time instant n, on the other hand, would require the value of the force of the hammer $F_H(n)$ at the same time index. In order to solve this problem, it is suitable to consider a delay of $\Delta t = T$ between the force and the displacement, so that the force of the hammer $F_H(n)$ is used for the computation of the displacement $\eta(n+1)$. Equation (12) is therefore used for the computation of the force of the hammer at time instant n and its value is inserted, for the computation of the displacement at instant $n+1$, inside the following:

$$\eta^{n+1} = d_1\eta^n + d_2\eta^{n-1} + d_F F_H(n) \tag{13}$$

where the coefficients d_i are given by:

$$d_1 = \frac{2}{1 + b_H T/2M_H}$$
$$d_2 = \frac{-1 + b_H T/2M_H}{1 + b_H T/2M_H}$$
$$d_F = \frac{-T^2/M_H}{1 + b_H T/2M_H}$$

The last things that need to be set are the conditions for the first time instants:
At time index $n = 1$:

- the string is in its equilibrium position, so $y_m^1 = 0\ \forall\ m = 1, 2, \ldots, M$;
- the hammer displacement with respect to the string equilibrium position is $\eta(1) = 0$, so that they are in contact at the start of the computation;
- the hammer velocity is different from 0 and set to $V_{H_0} = 2.5\ m/s$;

At time index $n = 2$:

- the string is still in its equilibrium position;
- the hammer displacement is obtained with $\eta(2) = V_{H_0}T$;
- the hammer force can be computed as $F_H(2) = K|\eta(2) - y(m_0, 2)|^p$ where $y(m_0, 2) = 0$ as written above.

Knowing these initial conditions, the script is built with the following order:

- initialize vectors and matrices at zero;
- compute $\eta(2)$ and $F_H(2)$;
- begin the loop over n from index $n = 2$ computing at first the force distribution F_m^n to be given to the displacement of the string, then, one after the other, the future events at instant $n+1$ for the string displacement, the displacement of the hammer, the force of the hammer.

There is one condition missing, which is the control over the fact the after a certain time instant the string leaves the hammer and from that point on the hammer is not acting on the string anymore, so its force should be set to zero. This condition is placed inside the previously explained loop for the computation of the force of the hammer at the time instant $n+1$ and it is formulated as:

$$\eta(n+1) < y(m_0, n+1) \implies F_H(n+1) = 0 \tag{14}$$

The script, as described above, gives the following results:

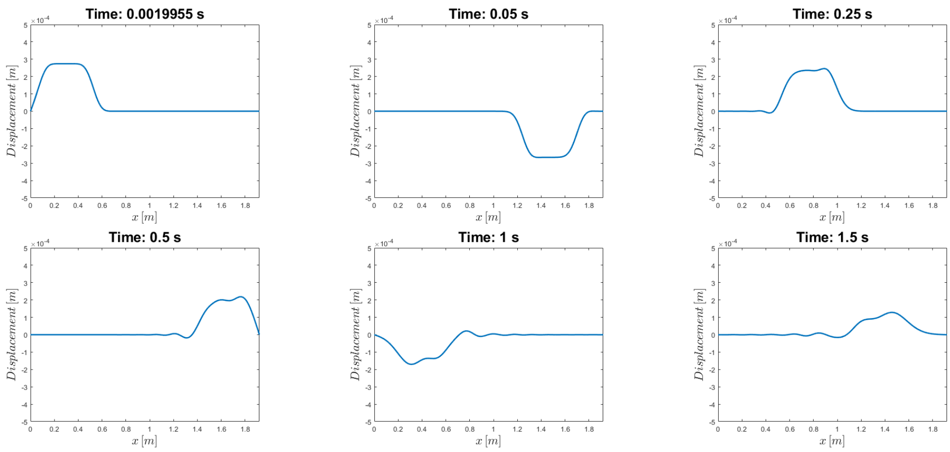


Figure 1: Configurations of the string displacement at different time instants

After the computation of the finite difference method, we considered a point on the string specular with respect to the hammer striking position, we averaged the displacement of that point over a neighbourhood of 12 samples around it and we plotted the obtained evolution on time in figure 2. That signal gives us the representation of the sound of the modelled string. As a final step of the study, we played the sound of that signal inside Matlab and save it as a *.wave* file with the functions:

$$sound(signal, f_s)$$

$$audiowrite(filename, signal, f_s)$$

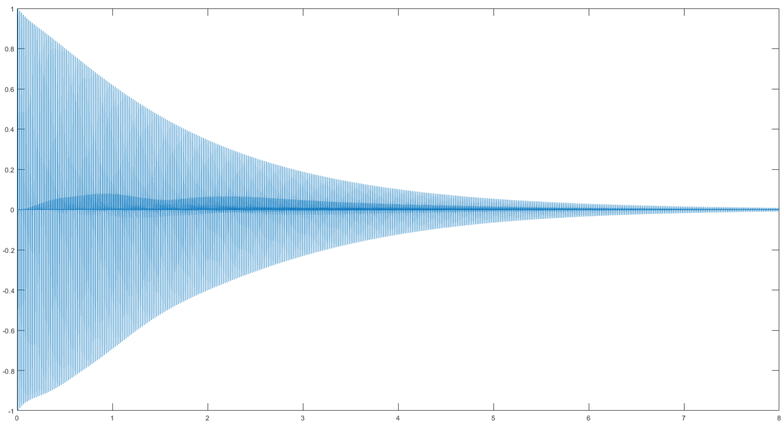


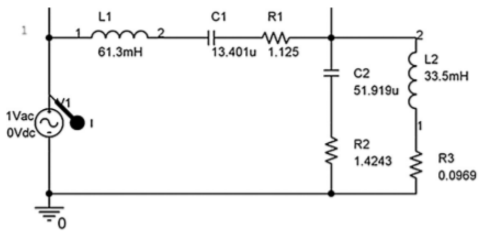
Figure 2: Evolution over time of the displacement of a point on the string

2 Modeling of a guitar

Our objective is now to model and simulate the sound of a guitar, using Matlab and Simscape to build an equivalent electric circuit.
At first we will only focus on the resonances of the top plate of the instrument, approximating the vibrating string with a damped square wave generator, and successively we will substitute this generator with a more accurate model that makes use of a transmission line.
The simulations can be run through the Matlab script named 'guitar_script', with which it is also possible to visualize the waveform of the resulting sound and its magnitude and phase, as well as playing and saving the sound on disk.

2.1 Approximating the resonances of the top plate

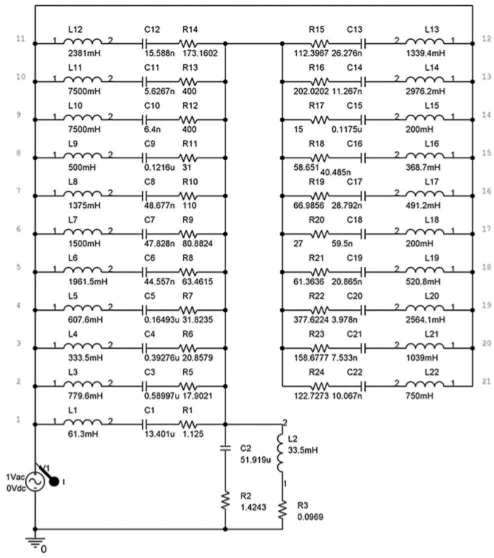
We start from the model of the guitar as a vibrating plate and an air cavity.



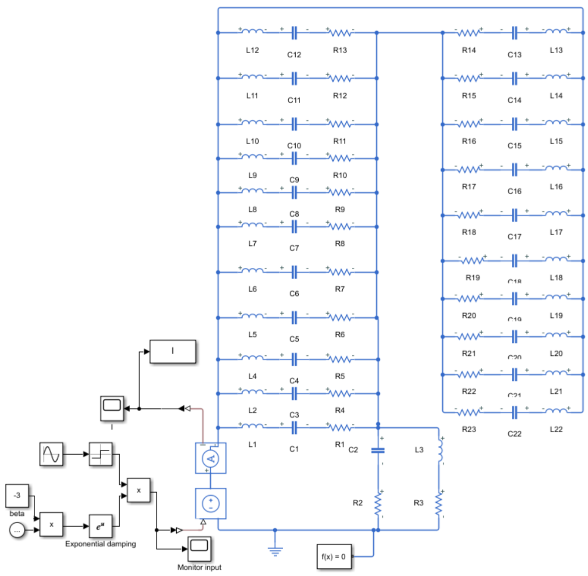
The branch with C_2 and R_2 represents the air volume, the one with L_2 and R_3 represents the sound hole, while the one with L_1, C_1 and R_1 models the plate.
The voltage generator approximates the vibration of the string, modeled with a damped square wave:

$$V_{in} = sgn(\sin 2\pi f_0 t)e^{-\beta t} \tag{15}$$

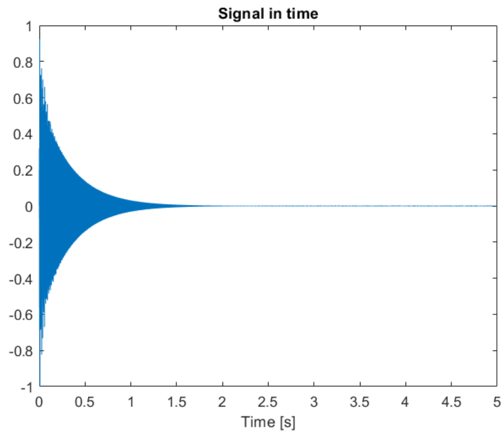
with $f_0 = 300Hz$ and $\beta = 3$. We now want to update the model increasing the number of resonances of the plate up to 20. To do so, we add, in parallel with the branch representing the plate, other branches containing an inductor, a capacitor and a resistor, each branch increasing the number of resonances by one.
the resulting schematics, where the values of the components are visible, is the following:



While the following picture shows our implementation in Simulink.

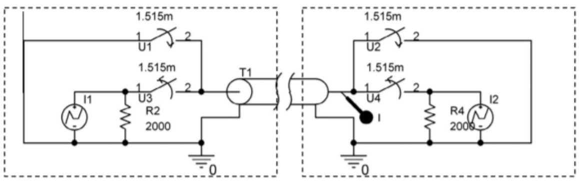


The sound resulting from this simulation is modeled by the current measured near the voltage generator, which needs to be resampled in order to be used, because it contains non constant time intervals between samples. It is then converted to sound wave inside the Matlab script.
The resulting sound wave is the following:

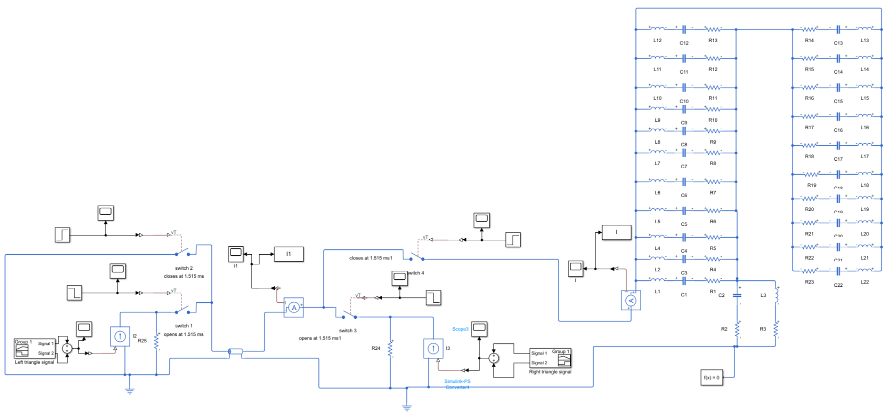


2.2 Adding the string model with transmission line

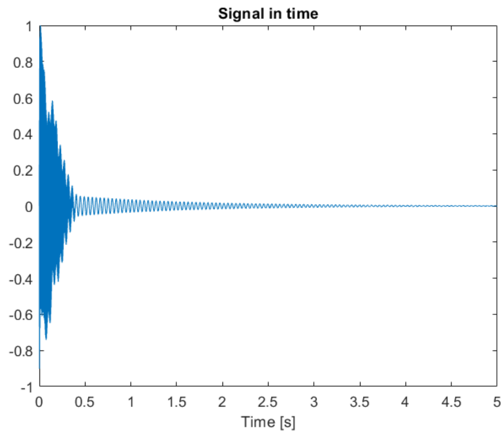
Listening to the sound resulting from the previous simulation, we can hear that we are still pretty far from the sound of an actual guitar. To improve on this, we want now to substitute the generator modeling the plucked string with a more accurate model, making use of a transmission line.



The transmission line has a delay of $1.515ms$ and an impedance of 2000Ω, while the current generators implement an asymmetric triangular wave $1.515ms$ long, with turning point at one fifth of its length and peak of $1mA$.
The complete model we implemented becomes the one below:



We again measure the current, which goes through the same processing described in the previous section, giving the following waveform:



Listening to the produced sound, we can hear that it is a pretty good simulation of the sound produced plucking the string of a classical guitar, although it presents an undesired tail given by one frequency damped too slowly, clearly visible also in the wave plot.