Modelling a drum membrane using FEM

Aduamoah M. Louca N. Torkington D.

What is Numerical Analysis?
27 March 2018

Outline

Outline

Geometry of the drum

Geometry of the drum

- $\mathcal{C}$ - wooden body of drum

- $\mathcal{C}$ - wooden body of drum
- Σ - surface of the membrane

- $\mathcal{C}$ - wooden body of drum
- Σ - surface of the membrane
- a - membrane radius

Outline

What affects the sound of a drum?

What affects the sound of a drum?

- Materials used

What affects the sound of a drum?

- Materials used
    - Absorption of sound

What affects the sound of a drum?

- Materials used
  - Absorption of sound
  - Damping terms

What affects the sound of a drum?

- Materials used
  - Absorption of sound
  - Damping terms
  - Losses

What affects the sound of a drum?

- Materials used
  - Absorption of sound
  - Damping terms
  - Losses
- Volume and shape

Mallet behaviour

Mallet behaviour

- $u(t)$ - position of the center of gravity of mallet

Mallet behaviour

- $u(t)$ - position of the center of gravity of mallet
- $\delta := u(0) = 0.025m$

Mallet behaviour

- $u(t)$ - position of the center of gravity of mallet
- $\delta := u(0) = 0.025m$
- $v_0 := -\frac{du}{dt}(0) = 1.4ms^{-1}$

Mallet behaviour

- $u(t)$ - position of the center of gravity of mallet
- $\delta := u(0) = 0.025m$
- $v_0 := -\frac{du}{dt}(0) = 1.4ms^{-1}$
- $F(t)$ - interaction force between mallet and membrane

Newton's 2nd Law

$$m\frac{d^2u}{dt^2} = F(t)$$

Force

Force

- $K = 1.6 \times 10^8 Nm^{\alpha}$ - coefficient of mallet stiffness

Force

- $K = 1.6 \times 10^8 Nm^{\alpha}$ - coefficient of mallet stiffness
- $W(t)$ - mean displacement of membrane's area in contact with mallet

Force

- $K = 1.6 \times 10^8 Nm^{\alpha}$ - coefficient of mallet stiffness
- $W(t)$ - mean displacement of membrane's area in contact with mallet
  - $w(x, y, t)$ - transverse displacement of the membrane
  - $g(x, y)$ - spatial window
  - $W(t) = \int_{\Sigma} w(x, y, t) g(x, y)\, dx\, dy$

Force

- $K = 1.6 \times 10^8 Nm^{\alpha}$ - coefficient of mallet stiffness
- $W(t)$ - mean displacement of membrane's area in contact with mallet
  - $w(x, y, t)$ - transverse displacement of the membrane
  - $g(x, y)$ - spatial window
  - $W(t) = \int_{\Sigma} w(x, y, t) g(x, y)\, dx\, dy$

Force term

$$F(t) = K[(\delta - u(t) + W(t))^{+}]^{\alpha}$$

Membrane equation

Membrane equation

- $f(t)$ - force density

Membrane equation

- $f(t)$ - force density
- σ - area density of the membrane

Membrane equation

- $f(t)$ - force density
- σ - area density of the membrane
- T - membrane tension

Membrane equation

- $f(t)$ - force density
- σ - area density of the membrane
- T - membrane tension
- η - viscoelastic damping coefficient

Membrane equation

- $f(t)$ - force density
- σ - area density of the membrane
- T - membrane tension
- η - viscoelastic damping coefficient

Membrane equation

$$\underbrace{\sigma\frac{\partial^2 w}{\partial t^2}}_{\text{inertial force}} = \underbrace{\operatorname{div}\left(T\nabla\left(w+\eta\frac{\partial w}{\partial t}\right)\right)}_{\text{restoring force due to tension}} - \underbrace{f(t)g}_{\text{impact force}}$$

Membrane equation

$$\underbrace{\sigma\frac{\partial^2 w}{\partial t^2}}_{\text{inertial force}} = \underbrace{\operatorname{div}\left(T\nabla\left(w+\eta\frac{\partial w}{\partial t}\right)\right)}_{\text{restoring force due to tension}} - \underbrace{f(t)g}_{\text{impact force}}$$

- Inertial force (mass density $\sigma \cdot$ acceleration) balanced by:-
  - restoring force due to tension T – an ‘equilibrium-seeking’ force; and
  - impact force from mallet – *actively* displaces membrance, hence minus sign

Couple more comments on Membrane Equation...

Membrane equation

$$\underbrace{\sigma \frac{\partial^2 w}{\partial t^2}}_{\text{inertial force}} = \underbrace{\operatorname{div}\left(T\nabla \left(w + \eta \frac{\partial w}{\partial t} \right) \right)}_{\text{restoring force due to tension}} - \underbrace{f(t)g}_{\text{impact force}}$$

- Membrane's internal damping modelled by **relaxation** term with coefficient η
- Here, we investigate *homogeneous* membranes $\implies \sigma, T, \eta$ *constant*

Membrane is clamped at its periphery: Dirichlet BCs

$$w(x, y, t) = 0 \quad \forall (x, y) \in \partial\Sigma \quad \forall t > 0$$

Membrane assumed to be at equilibrium & at rest at $t = 0$

$$w(x, y, 0) = \frac{\partial w}{\partial t}(x, y, 0) = 0 \quad \forall (x, y) \in \Sigma$$

Equation

- The membrane equation

$$\begin{aligned} \sigma\frac{\partial^2 w}{\partial t^2} &= \textbf{div}\left(T\nabla(w+\eta\frac{\partial w}{\partial t})\right) - f(t)g \\ w(x,y,t) &= 0 \text{ on } \partial\Sigma \\ w(x,y,0) &= 0. \end{aligned}$$

- Simplify

$$\sigma\frac{\partial^2 w}{\partial t^2} = T\Delta w + \eta T\frac{\partial}{\partial t}\Delta w - f(t)g.$$

Weak Form

- Multiply by test function w^* and integrate over Σ

$$\sigma \frac{\partial^2}{\partial t^2} \int_\Sigma w w^* ds - \int_\Sigma T \Delta w w^* ds - \frac{\partial}{\partial t} \int_\Sigma \eta T \Delta w w^* ds + f \int_\Sigma g w^* ds = 0$$

Weak Form

- Multiply by test function w^* and integrate over Σ

$$\sigma \frac{\partial^2}{\partial t^2} \int_\Sigma w w^* ds - \int_\Sigma T \Delta w w^* ds - \frac{\partial}{\partial t} \int_\Sigma \eta T \Delta w w^* ds + f \int_\Sigma g w^* ds = 0$$

- Simplify

$$\sigma \frac{\partial^2}{\partial t^2} \int_\Sigma w w^* ds + \int_\Sigma T \nabla w \nabla w^* ds + \frac{\partial}{\partial t} \int_\Sigma \eta T \nabla w \nabla w^* ds$$
$$+ f \int_\Sigma g w^* ds + \text{boundary conditions} = 0 \quad \forall w^*$$

Discretization

- **In space**: We use piecewise linear functions ($P1$) from space $H_h \subseteq H_0^1(\Sigma)$ on a triangular mesh. The approximated solution is denoted as w_h.

Discretization

- **In space**: We use piecewise linear functions ($P1$) from space $H_h \subseteq H_0^1(\Sigma)$ on a triangular mesh. The approximated solution is denoted as w_h.
- **In time**: The time derivatives are approximated using finite difference central difference formula.

$$\begin{aligned} \frac{\partial^2}{\partial t^2} w_h &= \frac{w_h^{n+1} - 2w_h^n + w_h^{n-1}}{\Delta t^2} \\ \frac{\partial}{\partial t} w_h &= \frac{w_h^{n+1} - w_h^{n-1}}{2\Delta t} \end{aligned}$$

Newmark Scheme

In order to better approximate the Laplacian, we use the Newmark scheme that takes an average

Newmark Method

$$\Delta w_h^n = \frac{1}{2}\Delta w_h^{n+1} + \frac{1}{2}\Delta w_h^{n-1}$$

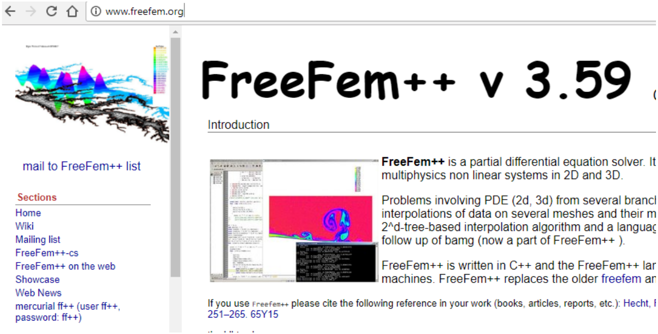
Discretized Equation

$$\int_\Sigma \sigma \frac{(w_h^{n+1} - 2w_h^n + w_h^{n-1})}{\Delta t^2} w_h^* ds + \int_\Sigma T \frac{(\nabla w_h^{n+1} + \nabla w_h^{n-1})}{2} \nabla w_h^* ds$$
$$+ \int_\Sigma \eta T \frac{(w_h^{n+1} - w_h^{n-1})}{2\Delta t} w_h^* ds + \int_\Sigma f g w_h^* ds + \text{boundary conditions} = 0.$$

Right hand side

$$g(x,y) = \frac{\exp[-10^7((x-x_0)^4 + (y-y_0)^4]}{\int_\Sigma \exp[-10^7((x-x_0)^4 + (y-y_0)^4]}$$
$$f(t) = \sin^2(100t)\exp(-50t)$$

FreeFem++


www.freefem.org
FreeFem++ v 3.59
Introduction
mail to FreeFem++ list
Sections
Home
Wiki
Mailing list
FreeFem++-cs
FreeFem++ on the web
Showcase
Web News
mercurial ff++ (user ff++, password: ff++)
FreeFem++ is a partial differential equation solver. It
multiphysics non linear systems in 2D and 3D.
Problems involving PDE (2d, 3d) from several branc
interpolations of data on several meshes and their m
2^d-tree-based interpolation algorithm and a languag
follow up of bamg (now a part of FreeFem++).
FreeFem++ is written in C++ and the FreeFem++ lar
machines. FreeFem++ replaces the older freefem ar
If you use Freefem++ please cite the following reference in your work (books, articles, reports, etc.): Hecht, F
251–265. 65Y15

Advantages of Freefem++

- Creates mesh and automatically produce the mass and rigid matrices

Advantages of Freefem++

- Creates mesh and automatically produce the mass and rigid matrices
- Uses ffglut for graphical output, but results can also be visualised with programs like Medit, Matlab, Visit and Mathematica

Visualisation

Visit

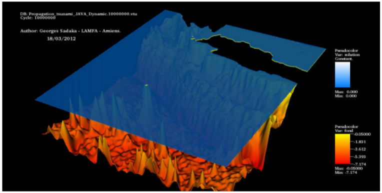


Figure 5: Visualising of the solution using visit

Mathematica

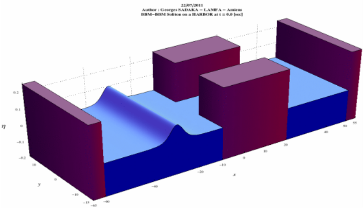


Figure 4: Visualising of the solution using Mathematica

Freefem++ Code

- Preliminaries: defining domain, creating mesh and defining function space

Code

```
border a(t=0,2*pi){ x=(1/(.5*cos(t)^2 +sin(t)^2)^.5)
        *cos(t); y=(1/(.5*cos(t)^2
        +sin(t)^2)^.5)*sin(t);label=1;}
mesh Th = buildmesh(a(50));
fespace Vh(Th,P1);
Vh uh,vh,uh0=0,uh1=uh0;
```

Freefem++ Code

- Preliminaries: Declaring constants and right hand side functions

Code

```
real sigma=0.262,eta=0.0000006, T = 3325, x0=0.21,
    y0=0. ,k=1.6e8 , alpha = 2.54,delta = 0.025,
    m=0.028,dt=.001,Tf=.1;

func g=exp(-(10^7) *((x-x0)^4+(y-y0)^4 ))
    /int2d(Th)(exp(-(10^7)*((x-x0)^4+(y-y0)^4 )));
```

Freefem++ Code

- Defining the problem : Time stepping algorithm

Code

```
for (real t=0.;t<Tf;t+=dt) {
func f = sin(100*t)^2*exp(-50*t);

solve membrane(uh,vh) = int2d(Th)(sigma*uh*vh)
 +int2d(Th)(Grad(uh)'*Grad(vh)*T*(dt)^2*.5 )
 + int2d(Th)(Grad(uh0)'*Grad(vh)*T*(dt)^2*.5 )
+ int2d(Th)(Grad(uh)'*Grad(vh)*dt*eta *T*.5)
+ int2d(Th)( f * g *vh *(dt)^2 )
 - int2d(Th)(Grad(uh0)'*Grad(vh)*dt*eta*T*.5 )
- int2d(Th)(sigma*(2.*uh1*vh - uh0*vh))
+ on(1,2,3,4,uh=0);
uh0 = uh1;
uh1 = uh; }
```

Freefem++ Code

- Postprocessing: Plotting and outputting results

Code

```
plot(uh,cmm="t="+t,fill=true,value=true,wait=0,dim=2,
    boundary=false);

ofstream file("mem" + kk + ".val");
file << "2 1 1 "<< uh[].n << " 2 \n";
for (int j=0;j<uh[].n ; j++){
file << uh[][j] << endl;
}
```