Numerical Solution for 2-d Wave Equation Using Finite Difference Method

Introduction

Based on the given situation, we have a snare drum where we want to model the waves of vibration right after it gets struck. The diameter of the drum is 30cm, and when it gets hit on the center, it has a downward displacement of 3mm. If we assume that the surface of the drum is a plane, then the vibrations cased by the struck travels over time through this plane. If we consider another plane perpendicular to the drum surface, then the vertical vibration caused by the struck happens through this plane. Figure 1 shows a how I picture the problem.

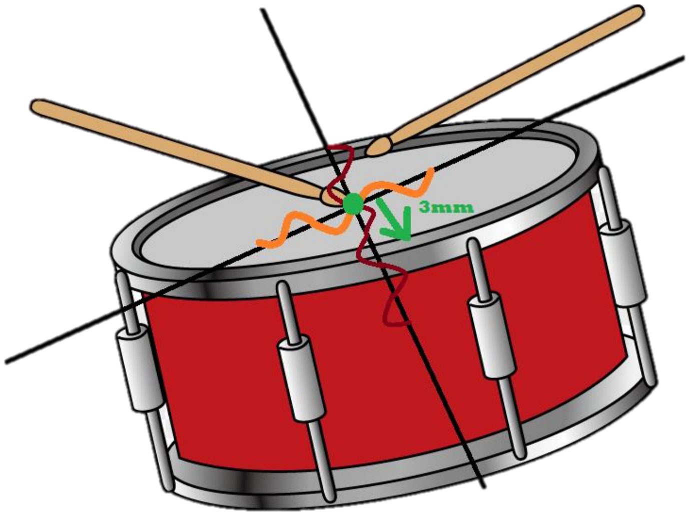

Figure 1

https://www.clipart1001.com/snare-drum-clipart-instruments-kisspng-musical-percussion/

Since we have sinusoidal waves going on the both planes, it is easiest to picture this problem in a polar coordinates, where the waves have are in terms of r, sinθ , and cosθ .

Let us start by the 2d wave equation:

$$\frac{\partial^2 u}{\partial t^2} = c^2\left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}\right) = c^2(Uxx + Uyy)$$

To transform the equation to polar, let

$$x(r,\phi) = r\cos\phi \qquad r(x,y) = \sqrt{x^2+y^2}$$

$$y(r,\phi) = r\sin\phi \qquad \phi(x,y) = \arctan\left(\frac{y}{x}\right).$$

Now we have to find the partial derivatives

$$\partial_x = \frac{\partial r}{\partial x}\partial_r + \frac{\partial \phi}{\partial x}\partial_\phi = \cos\phi\,\partial_r - \frac{\sin\phi}{r}\partial_\phi$$

$$\partial_y = \frac{\partial r}{\partial y}\partial_r + \frac{\partial \phi}{\partial y}\partial_\phi = \sin\phi\,\partial_r + \frac{\cos\phi}{r}\partial_\phi$$

On Assignment 6, we already have shown how to transform Laplacian from Cartesian to polar, so we are going to use the results.

$$\Delta = \mathrm{div}(\mathrm{grad}) = \partial_x\overrightarrow{\nabla_x} + \partial_y\overrightarrow{\nabla_y} = \frac{1}{r}\partial_\phi\overrightarrow{\nabla_\phi} + \frac{1}{r}\overrightarrow{\nabla_r} + \partial_r\overrightarrow{\nabla_r}$$

$$= \frac{1}{r}\partial_\phi\left(\frac{1}{r}\partial_\phi\right) + \frac{1}{r}\partial_r + \partial_r(\partial_r) = \frac{1}{r^2}\partial_\phi^2 + \frac{1}{r}\partial_r + \partial_r^2.$$

Therefore $\Delta u(x,y,t) = \frac{1}{c^2}\partial_t^2 u(x,y,t)$ becomes:

$$\Delta u(r,\phi,t) = \left[\frac{1}{r^2}\partial_\phi^2 + \frac{1}{r}\partial_r + \partial_r^2\right]u(r,\phi,t) = \frac{1}{r^2}\partial_\phi^2 u + \frac{1}{r}\partial_r u + \partial_r^2 u = \frac{1}{r^2}\partial_\phi^2 u + \frac{1}{r}\partial_r(r\,\partial_r u)$$
$$= \frac{1}{c^2}\partial_t^2 u.$$

Which is simply is $U_{tt} = c^2(U_{rr} + 1/rU_r + 1/r^2U_{\emptyset\emptyset})$

We are given a boundary and an initial condition on the question. First one is the diameter of the drum which means, when the waves travel that far, they hit the end. This can be translated in to u(15,t)=0 . Initial condition is when the drum is struck, caused a downward motion at t= 0:u (0, 0)=0.3.

Methods

Since our model is in polar coordinates, the finite difference method is the most appropriate for this complex domain. Let us write the finite difference formulae for this model:

$$d^2u/dt^2 \approx \Big(\big(u(t+\delta t,r,\emptyset) + u(t-\delta t,r,\emptyset) - 2u(t,r,\emptyset)\big)\Big)/(\delta t^2)$$

$$d^2u/dr^2 \approx \Big(\big(u(t,r+\ \delta r,\emptyset) + u(t,r-\ \delta\,r,\emptyset) - 2u(t,r,\emptyset)\big)\Big)/(\delta r^2)$$

$$d^2u/d\emptyset^2 \approx \Big(\big(u(t,r,\emptyset+\ \delta\emptyset) + u(t,r,\emptyset-\ \delta\emptyset) - 2u(t,r,\emptyset)\big)\Big)/(\ \delta\emptyset^2)$$

$$du/dr \approx \frac{u(t, r + \delta\mathrm{r}, \emptyset) + \mathrm{u}(\mathrm{t}, \mathrm{r} - \delta\mathrm{r}, \emptyset)}{\delta\mathrm{r}}$$

Since the Excel modelling is rather lengthy, I have tried to write the Matlab code (Enroth, 2017). The codes are described line by line. Also, there are comments below, explaining how I used the parameters.

```
function Two_D_drum_wave
clc; clear all; close all;
```

- This function solves the 2-D WAVE EQUATION $U_{tt} = c^2(U_{rr} + 1/rU_r + 1/r^2U_{\emptyset\emptyset})$) ,Using second-order finite difference explicit approximation for the partial derivatives

```
ni=10;
nj=10;
```

- These are small units which we are going to use for approximation. If we increase ni and nj you should change dt in order to comply with Courant-Friedrich Stability Condition. Figure 2 is a visual description of this approximation.

```
dx=1/ni;
dy=1/nj;
```

`x = 0.1:dx:1` will be the radial coordinate

`y = 0.1:dy:1;` will be transformed to phi coordinate

`c =1;` Wave velocity

`sigma = 1/sqrt(2); gamma = 1/sqrt(2);` Courant-Friedrich Stability Condition

```
dt = sigma*(dx/c);
```

`dt=0.001;` time step

```
t = 0:dt:1;
```

`u = zeros(length(x),length(y),length(t));` Matrix representation of the problem

- Initial condition is written below.

```
for j=1:length(y)-1
u(1,j,1)=-0.3;
end
```

- Below are the loops for finite difference method listed previously. The only difference is we used the small units introduced above instead of dx and dy. Just a reminder that dx=dr and dy=$d\emptyset$.
- Since there are going to be zeros in denominator when r=0 and $\emptyset = 0$ (B. Holman, 2015), I have asked Dr. Miguel Villegas, a Mathworks expert, to give me a possible solution. It was recommended to add more similar iterations to this code, where the singularities are introduced to Matlab.
- Below is when n=1

```
for i=2:length(x)-2
    for j=2:length(y)-2
        u(i,j,2)=u(i,j,1)+(1/2)*((c^2*dt^2/dx^2)*(u(i+1,j,1)-2*u(i,j,1)+u(i-1,j,1))...
                +((c^2*dt^2)/((x(i))^2*dy^2))*(u(i,j+1,1)-2*u(i,j,1)+u(i,j-1,1))...
            +(c^2*dt^2/(2*x(i)*dx))*(u(i+1,j,1)-u(i-1,j,1)));
    end
end
```

- Below is the If loop for the singularity.

```
if i==length(x)-1
    if j==length(y)-1
        u(i,j,2)=u(i,j,1)+(1/2)*((c^2*dt^2/dx^2)*(0-2*u(i,j,1)+u(i-1,j,1))...
                +((c^2*dt^2)/((x(i))^2*dy^2))*(0-2*u(i,j,1)+u(i,j-1,1))...
            +(c^2*dt^2/(2*x(i)*dx))*(0-u(i-1,j,1)));
    end
```

```
end
```

- Below is for every n >2

```
for n=2:length(t)-1
    for i=2:length(x)-2
        for j=2:length(y)-2
            u(i,j,n+1)=2*u(i,j,n)-u(i,j,n-1)+(c^2*dt^2/dx^2)*(u(i+1,j,n)-
2*u(i,j,n)+u(i-1,j,n))...
                +((c^2*dt^2)/((x(i))^2*dy^2))*(u(i,j+1,n)-2*u(i,j,n)+u(i,j-1,n))...
            +(c^2*dt^2/(2*x(i)*dx))*(u(i+1,j,n)-u(i-1,j,n));
        end
    end
```

- Below is the If loop for the singularities again

```
    if i==length(x)-1
        if j==length(y)-1
            u(i,j,n+1)=2*u(i,j,n)-u(i,j,n-1)+(c^2*dt^2/dx^2)*(0-2*u(i,j,n)+u(i-
1,j,n))...
                +((c^2*dt^2)/((x(i))^2*dy^2))*(0-2*u(i,j,n)+u(i,j-1,n))...
            +(c^2*dt^2/(2*x(i)*dx))*(0-u(i-1,j,n));
        end
    end
end
```

- Below are some pre-written codes so our graph would run smoothly. There is no mathematics behind it.

```
 [T,R] = meshgrid(linspace(0.1,2*pi,10),linspace(0.1,1,10));
X = R.*cos(T);
Y = R.*sin(T);
figure
for j=1:length(t)
       colormap(cool);
       p1 = surf(X,Y,u(:,:,j));  view(15, 35)
       title(sprintf('Two-D wave equation polar coordinates  at t = %1.2f
',t(j)),'Fontsize',11, 'interpreter', 'latex');
       xlabel('r','Fontsize',11); ylabel('\theta','Fontsize',11);
       zlabel(sprintf('u(r,\theta,t = %1.2f)',t(j)),'Fontsize',11);
       axis ([-1 1 0 2*pi -0.0025 0.0025]);
       pause(0.001);
       hold on;
       if(j~=length(t))
       delete(p1);
       end  end
```

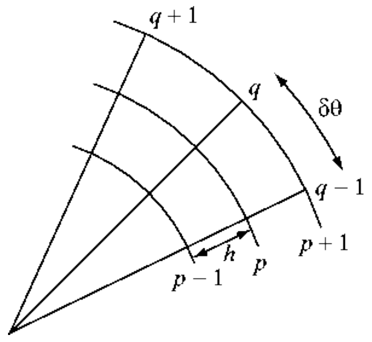

Figure 2 (Harris, 2003)

Results

The numerical results are listed in Matlab. There are all 10x10 arrays, an example of such results is in Table1. We can see that results are both are positive and negative, showing that waves move in different directions. It is important to note that ion initial iterations, most of the array is zero. This is due to the fact that it takes time for the struck to produce, and transfer waves to its surrounding. Figure 3 is a snapshot of the finalized model. The full result is not explainable by photos or matrix, because it has so many steps. A short video of the results will be attached to the assignment. A full list of values for every variable is available upon the request.

```
val(:,:,982) =

  1.0e-03 *

         0         0         0         0         0         0         0         0         0         0
         0    0.0734    0.1668    0.1827    0.1608    0.1827    0.1668    0.0734         0         0
         0   -0.0556   -0.0986   -0.1211   -0.1267   -0.1211   -0.0986   -0.0556         0         0
         0    0.0425    0.0775    0.1000    0.1076    0.1000    0.0775    0.0425         0         0
         0   -0.0356   -0.0655   -0.0852   -0.0921   -0.0852   -0.0655   -0.0356         0         0
         0    0.0229    0.0422    0.0550    0.0595    0.0550    0.0422    0.0229         0         0
         0    0.0046    0.0086    0.0114    0.0124    0.0114    0.0086    0.0046         0         0
         0   -0.0316   -0.0584   -0.0764   -0.0827   -0.0764   -0.0584   -0.0316         0         0
         0         0         0         0         0         0         0         0         0         0
         0         0         0         0         0         0         0         0         0         0
```

Table 1

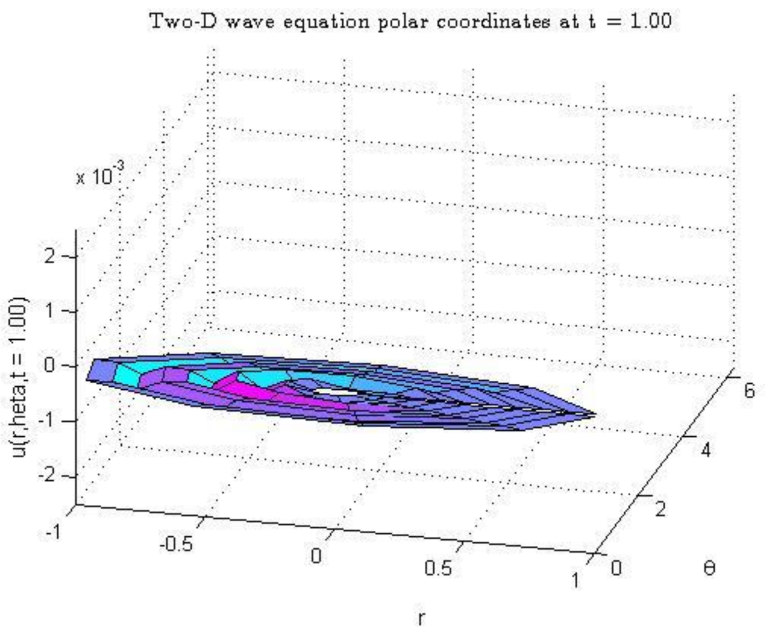

Figure 3

References

B. Holman, L. K. (2015). A Second-order Finite Difference Scheme For The Wave Equation on a Reduced Polar Grid. Arizona, USA. Retrieved from https://www.math.arizona.edu/~leonk/papers/polarFD7.pdf

Enroth, V. (2017, March 27). Retrieved from Mathworks: https://www.mathworks.com/matlabcentral/fileexchange/62204-2d-wave-equation-simulation

Harris, S. (2003). Retrieved from Finite differences in polar coordinates: https://homepages.see.leeds.ac.uk/~amt6xw/Distance%20Learning/CFD5030/node10.html#fd_fig_4_5