APPM 3050 Project #2

Solving the Wave Equation Using Finite Difference Approximations

Fiona Pigott, Dustin Martin, Christopher Miller

May 6, 2012

Abstract

The propagation of a wave through liquid is a mathematically challenging differential equation, which is difficult, if not impossible, to solve exactly for any generalized case. In the case of the motion of a wave in liquid (or any medium), the acceleration of the wave with respect to time can be given as a function of the acceleration across space scaled by some constant of the speed of propagation of the wave.

The goal of this project is to analyze the motion of a wave in liquid as it propagates across space and reflects off of physical boundaries, as well as to explore how depth affects the physical properties of a certain wave. It is possible to implement a general-case solution to demonstrate the motion of a wave under various conditions by numerically approximating the derivative of the wave function at each point and using those approximate derivatives to calculate the wave function, updating position and derivative values for a series of time steps to create a fluid movie of the wave over time.

1 Mathematics of Waves

1.1 1D Wave

In general, the equation of a time varying wave across space is given by:

$$\frac{1}{c^2}\frac{\partial^2 u}{\partial t^2} = \nabla^2 \tag{1}$$

which can be derived using Newton's 2^{nd} law, that $\Sigma F = ma$.

The one-dimensional wave equation is given by the function:

$$\frac{1}{c^2}\frac{\partial^2 u}{\partial t^2} = \frac{\partial^2 u}{\partial x^2}, \tag{2}$$

This equation is the result of the summation of all forces acting on any point on the wave at any time. In order to solve this second-order, differential equation, we applied the finite difference approximation to estimate values for $\frac{\partial^2 u}{\partial t^2}$ and $\frac{\partial^2 u}{\partial x^2}$. By using this approximation, it was not necessary to calculate the general solution for these differential equations. Instead, the partial derivatives can be rewritten as a function of the previous, current, and future positions of the 1-D wave in the form of:

$$\frac{\partial^2 u}{\partial x^2} = \frac{u_{i-1}^k - 2u_i^k + u_{i+1}^k}{\Delta x^2} \tag{3}$$

and similarly with time:

$$\frac{\partial^2 u}{\partial t^2} = \frac{u_i^{k+1} - 2u_i^k + u_i^{k-1}}{\Delta t^2} \tag{4}$$

We can substitute equations (**??**) and (**??**) into equation (**??**) to get:

$$(\, u_i^{k+1} - 2u_i^k + u_i^{k-1}\,)/\Delta t^2 = c^2(\, u_{i-1}^k - 2u_i^k + u_{i+1}^k\,)/\Delta x^2\,, \tag{5}$$

Since this equation relies on previous as well as future values of u with respect to x and t, two initial conditions and two boundary conditions must be known. For the case of the 1-D wave, we set the initial displacement of the wave to zero, and for the first few time steps, we change the displacement on the left hand side, causing a wave to propagate from left to right.

$$u^0(x, y) = 0\,, \tag{6}$$

On the left-hand side

$$\frac{\partial u}{\partial t} \neq 0$$

for some amount of time $0 \leq t \leq t_{initial}$, to allow the wave to form.

$$u^t(0, y) = f(y) \tag{7}$$

The boundary conditions for the right and left-hand sides must also be known in order to calculate how the wave reacts when it reaches a side. We manually input $f(x)$ which

varies over time, to dictate the position of the left-hand side of the wire, creating a wave simulation. A primitive way to approximate the movement of the right hand side is to set the derivitive equal to zero, but that doesn't account for any "lapping" motion that the wave would have against a boundrary, which can be mathematically modeled using the equation:

$$\alpha u(L,t) + \beta \frac{du(L,t)}{dx} = \gamma \tag{8}$$

And can be approximated using the finite difference approximation:

$$\alpha u^k + \beta \frac{u^{k+1} - u^{k-1}}{2\Delta x} = \gamma$$

$$\beta \frac{u^{k+1} - u^{k-1}}{2\Delta x} = \gamma - \alpha u^k$$

$$u^{k+1} - u^{k-1} = \frac{2\Delta x}{\beta}(\gamma - \alpha u^k)$$

$$u^{k+1} = \frac{2\Delta x}{\beta}(\gamma - \alpha u^k) + u^{k-1}$$

This equation calculates the future u value based on the current and initial u values. However, when the timestep of the wave reaches the right side boundary condition, the future value of u is unknown. As a result, we replace the value of u^{k+1} with a phantom variable, which will be assigned a value, even though the displayed wave will not display it.

$$phantom_{right} = \frac{2\Delta x}{\beta}(\gamma - \alpha u^k) + u^{k-1}$$

The values of α, β, and γ can be changed by the user to simulate different levels of friction between the wave and the boundary in order to display different degrees of wave "lapping".

The final addition to the 1-D wave function was to incorporate the variable depth. The general function of a one dimensional wave can be rewritten as:

$$\frac{1}{c^2}\frac{\partial^2 u}{\partial t^2} = \frac{\partial}{\partial x}(h(x)\frac{\partial u}{\partial x}) \tag{9}$$

$$\frac{1}{c^2}\frac{\partial^2 u}{\partial t^2} = h(x)\frac{\partial^2 u}{\partial^2 x} + \frac{\partial h}{\partial x}\frac{\partial u}{\partial x}$$

$$\frac{1}{c^2}\frac{u_i^{k+1} - 2u_i^k + u_i^{k-1}}{\Delta t^2} = h(x)\frac{u_{i-1}^k - 2u_i^k + u_{i+1}^k}{\Delta x^2} + \frac{\partial h}{\partial x}\frac{u_{i+1}^k - u_{i-1}^k}{2\Delta x}$$

$$u_i^{k+1} - 2u_i^k + u_i^{k-1} = \frac{c^2\Delta t^2}{\Delta x^2}(h(x)(u_{i-1}^k - 2u_i^k + u_{i+1}^k) + \frac{\partial h}{\partial x}\Delta x(\frac{u_{i+1}^k - u_{i-1}^k}{2}))$$

$$u_i^{k+1} = \frac{c^2\Delta t^2}{\Delta x^2}(h(x)(u_{i-1}^k - 2u_i^k + u_{i+1}^k) + \frac{\partial h}{\partial x}\Delta x(\frac{u_{i+1}^k - u_{i-1}^k}{2})) + 2u_i^k - u_i^{k-1}$$

This function requires the input of $h(x)$, (the height of the water from the bottom of the ocean) and $\frac{\partial h}{\partial x}$. Both of these values come from a user-defined function depth which is called in the main program. The smaller the value of h becomes, the more the wave slows down and the larger the wave becomes. The opposite is true for increases depth.

1.1.1 Finite Difference Approximation for a 1D Wave

1.2 2D Wave

Many of the same approaches applied to 1-D waves can be repeated with 2-D waves by adding a dimension, y. The general equation of a two-dimensional wave is:

$$\frac{1}{c^2}\frac{\partial^2 u}{\partial t^2} = \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}, \tag{10}$$

By slightly modifying equations (**??**) and (**??**), and adding another equation for the y component, we see that:

$$\frac{\partial^2 u}{\partial x^2} = \frac{u^k_{i-1,j} - 2u^k_{i,j} + u^k_{i+1,j}}{\Delta x^2} \tag{11}$$

$$\frac{\partial^2 u}{\partial y^2} = \frac{u^k_{i,j-1} - 2u^k_{i,j} + u^k_{i,j+1}}{\Delta x^2} \tag{12}$$

$$\frac{\partial^2 u}{\partial t^2} = \frac{u^{k+1}_{i,j} - 2u^k_{i,j} + u^{k-1}_{i,j}}{\Delta t^2} \tag{13}$$

When we substitute these component equations into the general form for a 2-D wave, we get:

$$\frac{u^{k+1}_{i,j} - 2u^k_{i,j} + u^{k-1}_{i,j}}{\Delta t^2} = c^2\left(\frac{u^k_{i-1,j} - 2u^k_{i,j} + u^k_{i+1,j}}{\Delta x^2} + \frac{u^k_{i,j-1} - 2u^k_{i,j} + u^k_{i,j+1}}{\Delta y^2}\right)$$

In order to simplify our results, we did not incorporate the lapping boundary function into our 2-D model. Because of this, our boundary conditions on the top, bottom, and right side of the wave set the derivative equal to zero by setting the created phantom values at these boundaries equal to their equivalent point just inside the exterior.

Similarly to the general equation of a 1-D wave with variable depth, the 2-D equation of a wave with variable depth is:

$$\frac{1}{c^2}\frac{\partial^2 u}{\partial t^2} = \frac{\partial}{\partial x}\left(h(x)\frac{\partial u}{\partial x}\right) + \frac{\partial}{\partial y}\left(h(x)\frac{\partial u}{\partial y}\right) \tag{14}$$

By using the same steps as in the 1-D wave, this equation can be simplified to the following equation to find the future value of u at any location (i, j).

$$u^{k+1}_i = \frac{c^2\Delta t^2}{\Delta x^2}\left(h(x)(u^k_{i-1,j} - 2u^k_{i,j} + u^k_{i+1,j}) + \frac{\partial h}{\partial x}\Delta x\left(\frac{u^{k+1}_{i,j} - u^{k-1}}{2}\right)\right)$$

$$+\frac{c^2\Delta t^2}{\Delta y^2}\left(h(x)(u^k_{i,j-1} - 2u^k_{i,j} + u^k_{i,j+1}) + \frac{\partial h}{\partial y}\Delta y\left(\frac{u^{k+1}_{i,j} - u^{k-1}_{i,j}}{2}\right)\right) + 2u^k_{i,j} - u^{k-1}_{i,j} \tag{15}$$

2 Computational Model

In order to successfully implement a suitably general wave simulation, the computer model must accomplish two tasks: it must accept a range of different functions for the depth and for the initial displacement, and it must calculate the finite difference approximations of the wave's derivative at several hundred points. Furthermore, the model is generating a movie in real time, so the code implementation of the model should run quickly enough to show a smooth, fluid image.

2.1 Handling (Literal) Edge-Cases

In terms of calculating reasonable finite-difference approximations, "phantom" values are needed to approximate derivatives on the edge of the space. In this model there is no friction between the waves and the walls of the space, so the spatial derivative (∇) of the wave function at the walls is going to be zero.

$$\frac{\partial u}{\partial x} = 0 \implies \frac{u_{x-1} - u_{x+1}}{2\Delta x} \implies u_{x-1} - u_{x+1} \implies u_{x-1} = u_{x+1} \tag{16}$$

Therefore, at an edge of the physical space, where either u_{x+1} or u_{x-1} is not within the space, it can be said that $u_{x+1} = u_{x-1}$ for points where $x = x_{max}$ or $u_{x-1} = u_{x+1}$ for points where $x = x_{min}$.

2.2 Method: 1D Wave

A wave in one dimension is the displacement of the top of the wave calculated over a series of discrete spacial steps of size Δx in the x-direction. Displacement is then calculated for a series of time steps Δt and string together to form a movie.

Each time step of the wave is calculated based on previous displacement values and forms the next vector of displacement values. The initial displacement (the wave) happens at $x = 0$ and propagates to the right. At the right hand side, the wave reflects of the right boundary and also interacts frictionally with the right boundary, following At the left hand side (the direction from which the original wave comes) the wave simply floats out into the ocean. This is achieved by setting each u_i^{k+1} value to the u_{i+1}^k (the more interior value from the previous time step). The displacement values only begin being set this way after the original wave has passed–an if statement controls the number of time steps that have passed by controlling the k value at which the boundary begins being evaluated in this new way.

2.3 Method: 2D Wave

The wave is essentially displacement calculated over a space in two dimensions, y and x. The displacement of any point on the wave at any given time is a function of its y and x position over the space.

Physically, the space over which the wave propagates is broken up into small rectangles of dimension Δy and Δx and one value of displacement $u(y, x)$ is calculated for each spatial

rectangle $\Delta_{y,x}$. Displacement values are represented as values in a matrix whose individual elements $u_{i,j}$ have indexes (i, j) which reference their physical position, with the (y, x) coordinates being $(i\Delta y, j\Delta x)$.

In calculating each future time step of the wave, each next displacement value is dependent only on the displacement values in its immediate vicinity in the previous time step. That is to say that no displacement value $u_{i,j}$ is dependent at all on the $u_{i,j}$ values around it in the present, but instead dependent on the nearby $u_{i,j}^{k-1}$ values (in the previous time step). In terms of code, no value in any one time step matrix u^k (where k refers to the time index) is dependent on any other value in u^k, meaning that the entire u^k matrix can be calculated at once, with only information about the boundary conditions and u^{k-1}.

2.3.1 Implementation

In order to handle the aforementioned edge cases computationally without wasting time evaluation "if" or "case" statements and without declaring each element of the matrix separately, our implementation of a two-dimensional wave model uses an extra row or column outside of the physical space for each u matrix, and then calculates the entire u^{k+1} matrix at once. These outer "phantom" values are then set to equal their corresponding inner values.

The entire matrix of values for any given $u_{i,j}$ is given by shifting the indexes of the piece of the larger $u_{y_{max+1},x_{max+1}}$ matrix.

A brief description of the method of assigning u^{k+1} after the initial disturbance, where h(y,x) defines the height from the bottom of the ocean.

u is a matrix y_{dim} by x_{dim}

$i = 2 : y_{dim} - 1$, The vector that defines the indexes of the physical y limits for u

$j = 2 : x_{dim} - 1$, The vector that defines the indexes of the physical x limits for u

for Δt : end **do**

$$u_{i,j}^{k+1} = \frac{c^2\Delta t}{\Delta y}(h)(u_{i-1,j} - 2u_{i,j} + u_{i+1,j}) + \frac{\partial h}{\partial y}(u_{i-1,j} - u_{i+1,j})\frac{1}{2}$$
$$+\frac{c^2\Delta t}{\Delta x}(h)(u_{i,j-1} - 2u_{i,j} + u_{i,j+1}) + \frac{\partial h}{\partial x}(u_{i,j-1} - u_{i,j+1})\frac{1}{2} + 2u_{i,j} - u_{i,j}^{k-1}$$

set phantom values:

$u_{top} = u_{i_{top}-1}$

$u_{bottom} = u_{i_{bottom}+1}$

$u_{right} = u_{j_{top}-1}$

$u_{left} = u_{j_{top}+1}$

end for

The method of assigning the entire matrix works out to be about 20% faster than using a for loop to define every element of the matrix with if statements to determine phantom

values. The difference is dramatic and important because of the size of the matrices that are being calculated (whereas in the one dimensional implementation, fewer values are being calculated and the computing time advantage is minimal).

3 Conclusion

By using a finite difference approximation, a difficult modeling problem using a complex, second-order differential equation became relatively simple to model. We modified the finite difference approach in order to calculate future values of the wave, generated by the past values of the wave and the boundary equations. These equations were used to generate numerical approximations of a wave in both 1-D and 2-D. By using this method, it was possible to approximate the motion of a wave as it propagates through a liquid, and reflects off of any boundary it comes up against. The speed of a wave is determined by the propagation constant c. A function of the depth over which the wave is propagating affects the speed of the water, ultimately affecting the height of the waves passing over the area. A depth equation was created and used in this numerical approximation of a wave, and the change in propagation speed was noticeable in our model using our depth function. A somewhat crude way to approximate the wave's behavior against the boundaries involves setting the slope of the wave to zero against said surfaces, but a more realistic approach is to try to emulate the "lapping" behavior that real waves do by using equation (8), something that we incorporated into our 1-D model, but not our 2-D model.