NUMERICAL SIMULATION OF NONLINEAR PROPAGATION OF TSUNAMIS

BY

SILAS R. BOWMAN
B.S. (Pacific Lutheran University 1997)
M.S. (University of Missouri in Rolla 2001)

THESIS

Submitted in partial fulfillment of the requirements
for the degree of Masters of Science in Mathematics
in the Graduate College of the
University of Illinois at Chicago, 2007

Chicago, Illinois

Copyright by

Silas R. Bowman

2007

Dedicated in Memory of My Soldiers from A Company, 54th Engineer Battalion, the "Outlaws", Specialist Joseph Micks (K.I.A. 2005), Specialist Jared Kubasak (K.I.A. 2004), and all the other service members who have given their lives in the line of duty.

May We Never Forget.

ACKNOWLEDGMENTS

The list of people I would like to thank could take up this entire thesis. My professors, fellow students, friends, and family have given me the mental strength to keep trying when the going got tough. Many people have contributed to where I am today, I will try to thank the most important few:

My advisor David Nicholls, for his tireless help over the last two years.

My wife Cindy for sticking with me through this my third degree and numerous deployments keeping us apart for more than three years of the ten years we have been married. My children Andrew, Sarah, and Nathan for giving me something to keep me busy when I was not studying.

My dad Gordon, and mom Kathy, and my brother and sister Obadiah and Elisabeth.

Ted, Carlo, Sarah, Ryan, Ed, Joe, and Xin who have been my friends at school for the last two years.

Kari Dueball and Steve Hurder for the help in getting me in to the program and keeping me headed in the right direction.

My fellow veterans at the Naperville Veterans of Foreign Wars for understanding me in a way no one else can.

My friends from Saint Timothy Lutheran Church for helping me in my spiritual walk.

God who always provides a window for me to climb up to, break, and crawl through when it seems the doors are all closed.

SRB

TABLE OF CONTENTS

CHAPTER		PAGE
1	**INTRODUCTION**	1
1.1	Introduction	1
1.2	Basics of the Water Wave Problem	1
1.2.1	Newtonian fluid mechanics	1
1.2.2	Euler Equations or "water wave problem"	2
1.2.3	Navier-Stokes as the "Bernoulli equation"	3
1.2.4	Dirichlet-Neumann Operator (DNO)	4
1.2.5	Derivation of the linear model equations	6
1.2.6	The nonlinear model equations	8
2	**EXACT SOLUTION TO LINEAR EQUATIONS**	9
2.1	Linear System of ODE's with Constant Coefficients	9
2.2	Real Vector Solutions	10
2.3	Final Exact Solution	11
3	**EULER APPROXIMATION**	12
3.1	Euler Approximation Method	12
3.2	Euler Results	13
4	**LEAP-FROG, RUNGE-KUTTA, AND EULER**	17
4.1	Leap-Frog	17
4.2	4th Order Runge-Kuta	17
4.3	Numerical Method Results	18
4.4	Order of Error Confirmation	20
4.4.1	Order of Error Solver	20
4.4.2	Results of Error Solver	20
5	**COMPUTING DERIVATIVES SPECTRALLY WITH THE FAST FOURIER TRANSFORM**	22
5.1	Fast Fourier Transform	22
5.2	Compare FFT to Known Derivative	23
6	**APPROXIMATION IN SPACE AND TIME**	24
6.1	Modified Water Wave Program	24
6.2	Wave as an Initial Condition	25
6.3	Linear Water Wave Problem	26
6.4	Approximations	26

TABLE OF CONTENTS (Continued)

CHAPTER **PAGE**

7 LINEAR TRAVELING WATER WAVE 29
7.1 Derivation of the Exact Solution to the Traveling Wave 29
7.1.1 Steady State Solution 29
7.1.2 Simplification 30
7.1.3 Solving the Linear System 31
7.1.4 Exact Solution to the Traveling Wave 32
7.2 Matlab Error Approximations 33

8 A NONLINEAR WATER WAVE MODEL 36
8.1 Nonlinear System Analysis 37

9 NONLINEAR EVOLUTION OF SURFACE WATER WAVES . 39
9.1 Gaussian Bell Curve 39

10 TSUNAMI EVOLUTION 41
10.1 Tsunami Source 41
10.2 Finite Depth 41
10.3 Tsunami Initial Wave Shape 42
10.4 Linear and Nonlinear Evolution of Tsunamis 45

CITED LITERATURE 50

LIST OF FIGURES

FIGURE		PAGE
1a	Euler a=1, b=0, k=1, $\Delta t = .01$	15
1b	Euler a=1, b=1, k=1, $\Delta t = .01$	15
1c	Euler a=1, b=1, k=2, $\Delta t = .01$	15
2a	Euler a=1, b=1, k=10, $\Delta t = .01$	16
2b	Euler a=1, b=1, k=100, $\Delta t = .01$	16
3	Euler, Leap-Frog, and RK4 with k = 10	19
4	Linear Wave Experiment I	27
5	Linear Wave Experiment II	28
6a	$\kappa = 2$, T=1	34
6b	$\kappa = 4$, T=1	34
7a	$\kappa = 2$, T=10	35
7b	$\kappa = 4$, T=10	35
8a	Non-Linear System 1, $\rho = 0.001$	38
8b	Non-Linear System 2, $\rho = 0.01$	38
8c	Non-Linear System Blow Up, $\rho = 0.01$	38
9	Gaussian with Linear vs. Nonlinear	40
10	Tensile Fault Tsunami Initial Condition	43
11	Strike Slip Fault Tsunami Initial Condition	44
12	Dip-Slip Fault Tsunami Initial Condition	44

LIST OF FIGURES (Continued)

FIGURE **PAGE**

13 Tensile Fault Tsunami T=150 46

14 Strike Slip Fault Tsunami T=150 47

15 Dip-Slip Fault Tsunami T=150 48

SUMMARY

In this thesis we study the nonlinear propagation of tsunami waves. In the introduction we explain the basics of the water wave equation. We take the system of ordinary differential equations describing wave behavior and find an exact solution for the time variable in the system. We use the Euler method to approximate the system and test it against the exact solution. We further use the "Leap-Frog" and 4th Order Runge-Kutta method to approximate the system and test those results against the exact solution. We confirm the order of error of each of these methods. We then start working with Fast Fourier Transforms to develop a spatial approximation method. Putting the methods together we can now approximate the wave in space and time. We test this approximation using a traveling wave because we can test it against an exact solution. In the second part of this thesis we introduce non-linearity into the program and add depth to the water which had been infinite before. We test both the linear and nonlinear model using a Gaussian function to illustrate the need for the nonlinear system. Finally we take three tsunami waves and approximate their behavior paying particular attention to the leading edge of the wave.

CHAPTER 1

INTRODUCTION

1.1 Introduction

In this thesis, we study the evolution of tsunami waves as they move toward the shore. We will use initial conditions on the surface of the ocean after a dip-slip fault, a strike-slip fault, and a tensile fault. Developing first exact solutions to the linear water wave equations we will test three different time approximation schemes against the exact solution. We then introduce a spatial approximation scheme. We put these schemes together and test them against "smooth" waves and then against a traveling wave of which we have an exact solution. We introduce depth and nonlinearity into the system and test the new system against the well known Gaussian curve. Finally we use the three tsunami waves as initial conditions and observe the propagation of these waves in space and time.

1.2 Basics of the Water Wave Problem

1.2.1 Newtonian fluid mechanics

The governing relations of (Newtonian) fluid mechanics are the Navier–Stokes equations (Ach90; Lam93). These equations can be used to model a wide array of physical phenomena including the motion of ocean waves which interest us in this research. However, for waves on the surface of an ocean there are several modeling assumptions we may make which result in

negligible disagreement with experimental observations and significantly simplify the governing equations. These assumptions are:

1. The fluid is incompressible.
2. The fluid is irrotational.
3. The fluid is inviscid.

1.2.2 Euler Equations or "water wave problem"

Such a fluid is typically termed an "ideal" or "perfect" fluid and the resulting equations are the Euler equations. As there are several systems which go by this name it is typical to refer to these equations as the "water wave problem." It is the purpose of this section to briefly introduce them and to motivate the model equations which we consider.

To begin, let the body of water (ocean) whose motion we wish to model occupy the domain

$$S_{h,\eta} := \{(x, y) \in R^{d-1} \times R \mid -h < y < \eta(x, t)\}.$$

Here η is the free air–fluid interface and h is the mean depth of the fluid (which may be infinite). Due to irrorationality, the velocity of particles in the fluid, u, may be expressed as the gradient of a potential function, φ:

$$u = \nabla\varphi$$

giving rise to the term "potential flow." The incompressibility of the fluid states that the divergence of the velocity is zero implying that

$$0 = \nabla \cdot [u] = \nabla \cdot [\nabla \varphi] = \Delta \varphi, \tag{1.1}$$

i.e. that φ satisfies Laplace's equation inside $S_{h,\eta}$.

Regarding boundary conditions, we demand that fluid particles have zero normal component of velocity at the bottom (though tangential velocities, or "slip," is allowed) so that

$$\partial_y \varphi = 0, \quad y = -h. \tag{1.2}$$

For a fluid of infinite depth this is generalized by

$$\partial_y \varphi \to 0, \quad y \to -\infty. \tag{1.3}$$

At the free surface we must specify both a boundary condition for φ (so that our problem is "well posed") and an evolution equation for η. The latter is given by the "Kinematic condition" which requires that particles which start at the free surface always stay there:

$$\partial_t \eta - \partial_y \varphi + \nabla_x \eta \cdot \nabla_x \varphi = 0. \tag{1.4}$$

1.2.3 Navier-Stokes as the "Bernoulli equation"

Finally, the Navier–Stokes equations make an appearance (after an integration (Ach90; Lam93)) as the "Bernoulli equation":

$$\partial_t \varphi + \frac{1}{2} |\nabla \varphi|^2 + g\eta = 0. \tag{1.5}$$

We note that (Equation 1.5) only takes into account gravitational forces and excludes surface tension; this can easily be included if desired.

1.2.4 Dirichlet-Neumann Operator (DNO)

Due to the simple nature of the governing equation in the fluid bulk (Laplace's equation), it is not surprising that the water wave problem Equation 1.1, Equation 1.2, Equation 1.4, and Equation 1.5 can be restated on the *surface* of the fluid domain. The surprising discovery of Zakharov (Zak68) is that if the proper surface variables are chosen then, in fact, the resulting system of evolution equations is Hamiltonian. Zakharov's variables are (η, ξ), where

$$\xi(x,t) := \varphi(x, \eta(x,t), t) \tag{1.6}$$

is the velocity potential at the surface of the fluid. Zakharov's formulation of the water wave problem in terms of (η, ξ) is somewhat implicit in nature and a clarifying contribution was made by Craig & Sulem (CS93) who introduced the Dirichlet–Neumann operator (DNO) to the problem. For the water wave problem this operator relates the Dirichlet data (values of φ at η) for Laplace's equation on the domain $S_{h,\eta}$ to Neumann data (values of $\partial_N \varphi$ at η). More precisely, given the problem:

$$\Delta\varphi = 0 \qquad \in S_{h,\eta} \tag{1.7a}$$

$$\varphi(x,\eta) = \xi(x) \tag{1.7b}$$

$$\partial_y \varphi(x,-h) = 0, \tag{1.7c}$$

the DNO, G, produces the normal derivative at the surface:

$$G(\eta)[\xi] := \left(\nabla\varphi|_{y=\eta}\right) \cdot N = \partial_y \varphi(x,\eta) - \nabla_x \eta \cdot \nabla_x \varphi(x,\eta), \tag{1.8}$$

where $N = (-\nabla_x \eta, 1)^T$ is an (unnormalized) normal to the fluid domain. In terms of this operator the Hamiltonian is given by

$$H(\eta,\xi) = \frac{1}{2}\int \xi G(\eta)[\xi] + g\eta^2 \, dx, \tag{1.9}$$

and the evolution equations for (η,ξ) are:

$$\partial_t \begin{pmatrix} \eta \\ \xi \end{pmatrix} = \begin{pmatrix} \delta_\xi H \\ -\delta_\eta H \end{pmatrix}, \tag{1.10}$$

where δ denotes functional variation.

1.2.5 Derivation of the linear model equations

While beautiful, equations (Equation 1.10) are rather unwieldy, especially for numerical simulation. Therefore it is desireable to derive model equations which include most of the central features of (Equation 1.10) while being easier to address numerically. A useful fact which makes such a simplification possible is the analyticity of the DNO with respect to boundary deformation. More precisely, if we set $\eta(x) = \varepsilon f(x)$, i.e. η is a *small* deformation, then we can formally expand

$$G(\eta)[\xi] = G(\varepsilon f)[\xi] = \sum_{n=0}^{\infty} G_n(f)[\xi]\varepsilon^n. \tag{1.11}$$

A large body of recent work (see (CM85; CSS97; CN00; NR01; HN05)) has been devoted to making this expansion rigorous and indeed, under certain smoothness assumptions on $f(x)$, (Equation 1.11) is true in a rather strong, functional analytic, sense. With this result in mind one can imagine approximating the full DNO by

$$G^N(\eta)[\xi] := \sum_{n=0}^{N} G_n(\eta)[\xi], \tag{1.12}$$

and, therefore, the full Hamiltonian by

$$H^N(\eta, \xi) := \frac{1}{2}\int \xi G^N(\eta)[\xi] + g\eta^2 \, dx. \tag{1.13}$$

From this, evolution equations for (η, ξ) are derived from a calculation like (Equation 1.10). In this work we will consider two model equations, those generated by H^0 and those generated by H^1.

To compute these model equations all we need are forms for G_0 and G_1. At this point we specialize to the two–dimensional problem ($d = 2$) and the classical periodic boundary conditions (with period L). It has been shown in several publications (see, e.g., (CS93; NR01)) that the first two terms in the expansion of the DNO have the form:

$$G_0[\xi] = \sum_{p=-\infty}^{\infty} |p| \tanh(h|p|)\hat{\xi}_p =: |D| \tanh(h|D|)[\xi] \tag{1.14a}$$

$$G_1(\eta)[\xi] = \partial_x \left[\eta \partial_x \xi\right] - G_0 \left[\eta G_0[\xi]\right]. \tag{1.14b}$$

While rather nonstandard, the operator G_0 is simply an order–one Fourier multiplier which behaves much like a first derivative. Furthermore, such operators fit quite naturally into a numerical scheme based upon Fourier series which is why we advocate such a technique later.

With the formulas (Equation 1.14) we can now state our two model equations. Using simply the zeroeth order approximation of the DNO $G^0 = G_0$ we have

$$\partial_t \eta = G_0[\xi] \tag{1.15a}$$

$$\partial_t \xi = -g\eta, \tag{1.15b}$$

which is linear and has Hamiltonian

$$H^0 = \frac{1}{2}\int \xi G_0[\xi] + g\eta^2 \, dx. \tag{1.16}$$

1.2.6 The nonlinear model equations

Next, using the first order approximation of the DNO, $G^1 = G_0 + G_1$ we find

$$\partial_t \eta = G_0[\xi] + G_1(\eta)[\xi] \tag{1.17a}$$

$$\partial_t \xi = -g\eta - \frac{1}{2}\left\{(\partial_x \xi)^2 - (G_0[\xi])^2\right\}, \tag{1.17b}$$

which is nonlinear (though only quadratic) and has Hamiltonian

$$H^1 = \frac{1}{2}\int \xi G_0[\xi] + \xi G_1(\eta)[\xi] + g\eta^2 \, dx. \tag{1.18}$$

CHAPTER 2

EXACT SOLUTION TO LINEAR EQUATIONS

2.1 Linear System of ODE's with Constant Coefficients

To start deriving the exact solution to the linear equation (Equation 1.15) in infinite depth I will let $u = \hat{\eta}(k)$, (the Fourier Transform of $\eta(x)$), $v = \hat{\xi}(k)$, and $G_0 = |k|$. Also throughout this thesis $\partial_t u = u_t$ or u'. Finally setting the initial values of $u(0)$ and $v(0)$ to a and b respectively gives us the following equations:

$$u' = |k|v \tag{2.1a}$$

$$v' = -gu. \tag{2.1b}$$

The next step is to get the two ODE's in the form $y' = Ay$ where A is a 2x2 matrix and y' and y are column vectors consisting of u and v. The equations are:

$$\begin{pmatrix} u \\ v \end{pmatrix}' = \begin{pmatrix} 0 & |k| \\ -g & 0 \end{pmatrix} \begin{pmatrix} u \\ v \end{pmatrix}. \tag{2.2}$$

We recall that to solve $y' = Ay$ it is sufficient to find the eigenvalues and eigenvectors of A (BD04). The eigenvalues of A are $\lambda = \pm i\sqrt{g|k|}$ and the corresponding eigenvectors are $\xi = (1, \pm i\sqrt{\frac{g}{|k|}})^T$.

2.2 Real Vector Solutions

The result is two linearly independent real vector solutions:

$$e^{\alpha t}\mathbf{b}\cos(\beta t) - e^{\alpha t}\mathbf{a}\sin(\beta t) \tag{2.3a}$$

$$e^{\alpha t}\mathbf{a}\sin(\beta t) + e^{\alpha t}\mathbf{b}\cos(\beta t), \tag{2.3b}$$

Where $\beta = \sqrt{g|k|}$.

Substituting in the values for $\alpha, \beta, \mathbf{a}, and\ \mathbf{b}$ gives

$$C_1 \left\{ \begin{bmatrix} 1 \\ \\ 0 \end{bmatrix} \cos\sqrt{g|k|}\ t - \begin{bmatrix} 0 \\ \\ \sqrt{\frac{g}{|k|}} \end{bmatrix} \sin\sqrt{g|k|}\ t \right\}$$
$$+ C_2 \left\{ \begin{bmatrix} 1 \\ \\ 0 \end{bmatrix} \sin\sqrt{g|k|}\ t + \begin{bmatrix} 0 \\ \\ \sqrt{\frac{g}{|k|}} \end{bmatrix} \cos\sqrt{g|k|}\ t \right\}.$$

Now we can split this back into two equations

$$u(t) = C_1 \cos\sqrt{g|k|}\ t + C_2 \sin\sqrt{g|k|}\ t \tag{2.4a}$$

$$v(t) = -C_1\sqrt{\frac{g}{|k|}} \sin\sqrt{g|k|}\ t + C_2\sqrt{\frac{g}{|k|}} \cos\sqrt{g|k|}\ t. \tag{2.4b}$$

Using our initial conditions we have

$u(0) = a = C_1 \quad \Rightarrow \quad C_1 = a$

$v(0) = b = C_2\sqrt{\frac{g}{|k|}} \quad \Rightarrow \quad C_2 = b\sqrt{\frac{|k|}{g}},$

So finally our exact solution to the linear system of ODEs is:

$$u(t) = a\ \cos\sqrt{g|k|}\ t + b\sqrt{\frac{|k|}{g}}\ \sin\sqrt{g|k|}\ t \tag{2.5a}$$

$$v(t) = -\ a\sqrt{\frac{g}{|k|}}\ \sin\sqrt{g|k|}\ t + b\ \cos\sqrt{g|k|}\ t. \tag{2.5b}$$

We set $\omega^2 = g|k|$ and so $\omega = \sqrt{g|k|}$, and then $\frac{\omega}{g} = \frac{\sqrt{g|k|}}{g} = \frac{\sqrt{|k|}}{g}$ and $\frac{\omega}{|k|} = \frac{\sqrt{g|k|}}{|k|} = \frac{\sqrt{g}}{|k|}$.

2.3 Final Exact Solution

So our exact solution can be written as

$$u(t) = a\ \cos\omega\ t + b\ \frac{\omega}{g}\ \sin\omega\ t \tag{2.6a}$$

$$v(t) = -\ a\ \frac{\omega}{|k|}\ \sin\omega\ t + b\ \cos\omega\ t. \tag{2.6b}$$

CHAPTER 3

EULER APPROXIMATION

We now want to compare our exact solution to a numerical approximation. We use the exact solutions (Equation 2.6):

$$u(t) = a\ \cos\omega\ t + b\ \frac{\omega}{g}\ \sin\omega\ t \tag{3.1a}$$

$$v(t) = -\ a\ \frac{\omega}{|k|}\ \sin\omega\ t + b\ \cos\omega\ t. \tag{3.1b}$$

3.1 Euler Approximation Method

We use the Euler method to approximate the solutions. The Euler method estimates the solution at the next time step by multiplying the "slope" or (derivative in time) at the current time by the time step and adding it back to the current value. For the ODE:

$$x' = f(x,t), \quad x(0) = \alpha \tag{3.2}$$

Euler's method reads:

$$x_{n+1} = x_n + (\Delta t) f(x_n, t_n), \quad n \geqq 0, \quad x(0) = \alpha \tag{3.3}$$

where $x_n \approx x(t_n)$ and Δt is the time step.

In our case given the initial conditions are $u(0) = a$, $v(0) = b$, our system (Equation 2.1) is approximated by:

$$u_{n+1} = u_n + \Delta t[|k|v_n] \tag{3.4a}$$

$$v_{n+1} = v_n + \Delta t[-gu_n]. \tag{3.4b}$$

3.2 Euler Results

Using MATLAB to generate graphs we can compare the approximate solution to the exact one. We compare the graphs for u and v using various values for the coefficients a (the initial value of u), b (the initial value of v), k (the wave number), but keeping $g = 1$. We will also vary T, and Δt the size of the time step.

The first set of graphs display results with a = 1, b = 0, g = 1, k = 1, T = 3, and Δt varied from 1 to .1 to .01. I found that with Δt at .1 we got a reasonable approximation, but, as expected, Δt at .01 was the best see, Figure 1a.

The next set of graphs I tried b = 1 and all other variables the same as before and found similar results with the best approximation being with $\Delta t = .01$, see Figure 1b.

The third set of graphs I kept everything the same but changed k = 2. Again, $\Delta t = .01$ was the best approximation, but unfortunately the approximation was not as good as when k = 1, see Figure 1c.

I then changed k to 10 and then to 100. With both of these as expected the approximation was worse than with k = 1. With $\Delta t = .01$ the error increase is apparent see Figure 2a for k = 10, and see Figure 2b for k = 100. With k = 100 I had to reduce Δt to .001 to get a good approximation.

The following table lists the error found for each graph which was calculated by taking the maximum difference between v actual and v approximated, $\max|v(t) - v_N(t)|$:

b	k	Δt	error
0	1	0.01	0.0987
1	1	0.01	0.1219
1	2	0.01	0.2171
1	10	0.01	1.024
1	100	0.01	13.76
1	100	0.001	0.9034

So Euler leaves something to be desired as we would like more flexibility in choosing our k values without the cost of computing time. In our next chapter we will introduce two new methods of approximation.

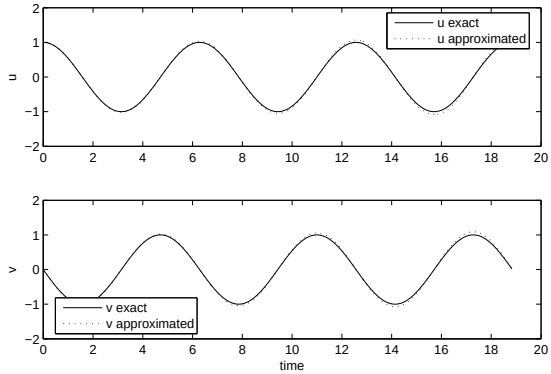

Figure 1a. Euler a=1, b=0, k=1, $\Delta t = .01$

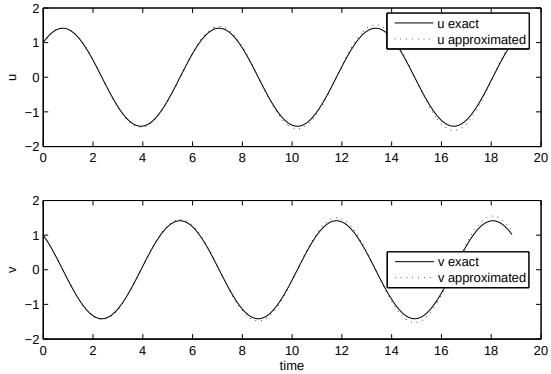

Figure 1b. Euler a=1, b=1, k=1, $\Delta t = .01$

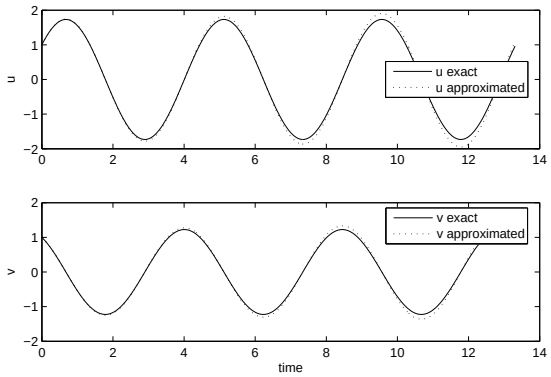

Figure 1c. Euler a=1, b=1, k=2, $\Delta t = .01$

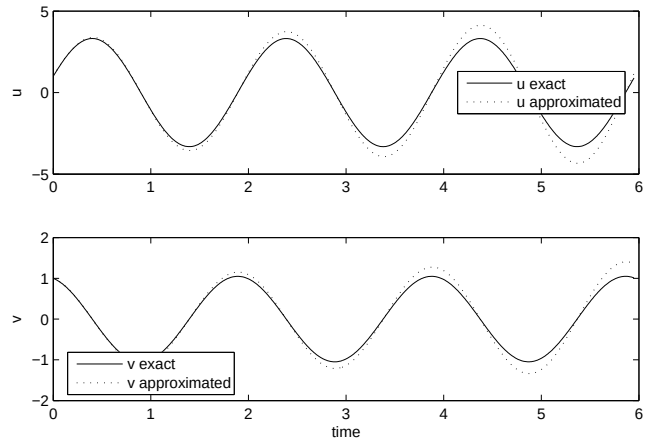

Figure 2a. Euler a=1, b=1, k=10, $\Delta t = .01$

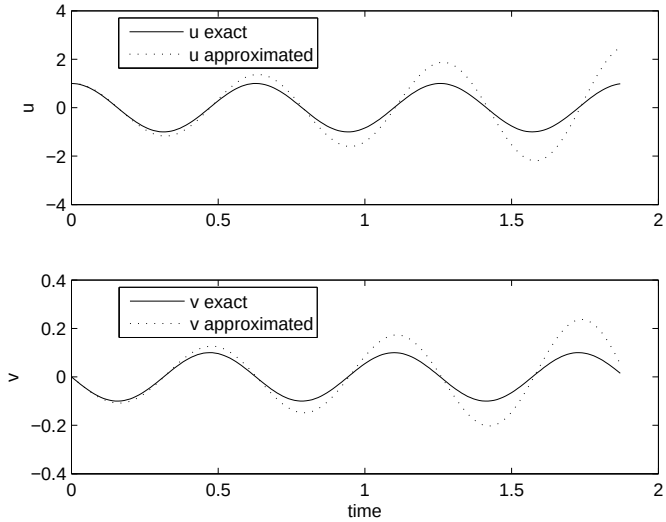

Figure 2b. Euler a=1, b=1, k=100, $\Delta t = .01$

CHAPTER 4

LEAP-FROG, RUNGE-KUTTA, AND EULER

We now want to compare our exact solution to a few more numerical approximations and then compare the approximations to each other.

4.1 Leap-Frog

Now we consider the "Leap-Frog" method which approximates $x(t)$ using the two previous time levels. We use our initial values and then approximate our first values using one step of the Euler Method. From then on we use:

$$x_n = x_{n-2} + 2(\Delta t) f(x_{n-1}, t_{n-1}) \tag{4.1}$$

to approximate our solution. This should give us error of $\mathrm{O}(\Delta t)^2$ (Str05).

4.2 4th Order Runge-Kuta

Finally for an even better approximation we use 4th Order Runge-Kutta to approximate the solution. The formula for this is:

$$x_{n+1} = x_n + \frac{1}{6}(k1 + 2 \cdot k2 + 2 \cdot k3 + k4); \tag{4.2a}$$

$$k1 = \Delta t f(x_n), \tag{4.2b}$$

$$k2 = \Delta t f(x_n + \frac{1}{2}k1), \tag{4.2c}$$

$$k3 = \Delta t f(x_n + \frac{1}{2}k2), \tag{4.2d}$$

$$k4 = \Delta t f(x_n + k3). \tag{4.2e}$$

Runge-Kutta should give us error $O(\Delta t)^4$ (GW04).

4.3 Numerical Method Results

Using MATLAB to generate graphs we can compare the exact solution to the numerical approximations and we can also graph the error for each approximation. The error graph is relative error. This error for the approximation (u_a, v_a) is determined by the equation:

$$err(u_a, v_a) = \frac{\sqrt{(u_e - u_a)^2 + (v_e - v_a)^2}}{\sqrt{u_e^2 + v_e^2}}, \tag{4.3}$$

where (u_e, v_e) is the exact solution.

We compare the graphs for u and v using various values for the coefficients a (the initial value of u), b (the initial value of v), and k (the wave number), and g will remain 1. We will also vary T, and Δt.

The solution with a = 1, b = 1, g = 1, k = 10, T = 3, and Δt = .01 is plotted below.

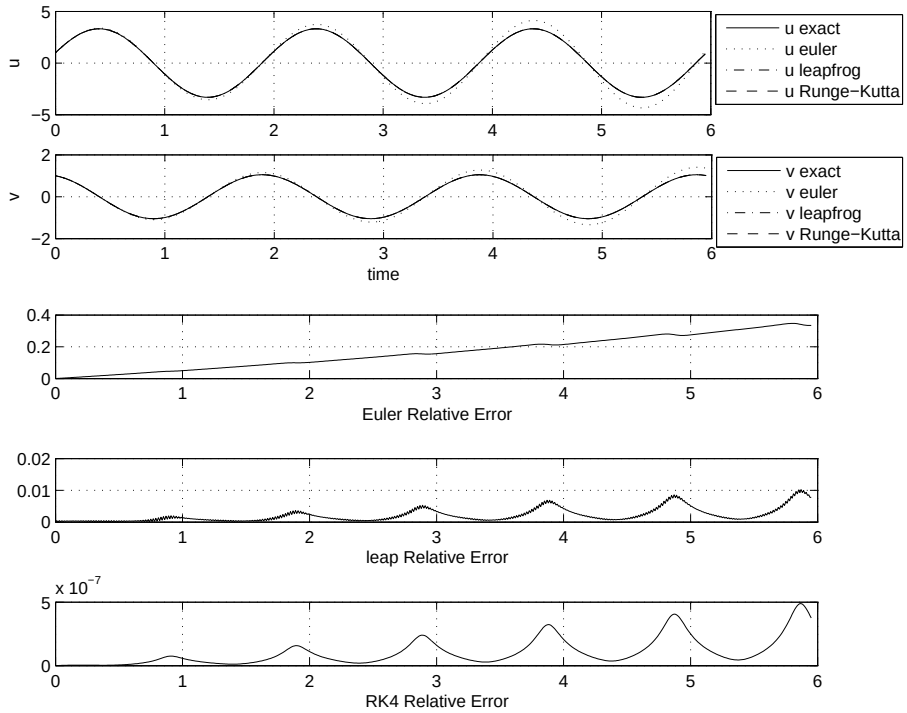

Figure 3. Euler, Leap-Frog, and RK4 with k = 10

All the approximations get better as Δt gets smaller, however, Leap-Frog is as expected order 2 better than Euler and Runge-Kutta is order 4 better than Euler.

4.4 Order of Error Confirmation

We now want to confirm the orders of error of each numerical method. Euler should gives us error O(Δt), Leap-Frog should give us error of O$(\Delta t)^2$,and Runge-Kutta should give us error O$(\Delta t)^4$.

4.4.1 Order of Error Solver

I used my MATLAB program "solver" to generate estimates, the exact solution, and the maximum relative error. I then created a program "checkerror" which takes an array of different Δt and gets the maximum error of the numerical method and then uses MATLAB's built in least squares method "polyfit" to come up with coefficients.

The equation for the error is $error = C(\Delta t)^P$, so we take the log of both sides of the equation giving $\log(error) = \log(C(\Delta t)^P)$, simplifying gives $\log(error) = \log(C) + \log((\Delta t)^P)$, simplifying further gives $\log(error) = \log(C) + P\log(\Delta t)$.

Now "polyfit" takes an equation of the form $y = mx + b$ where the user provides an array of x and y and the program gives m and b. In our equation $y = \log(error)$, $x = \log(\Delta t)$, $m = P$, and $b = \log(C)$. Now we have P from our equation and only need to get C by $C = e^b$.

4.4.2 Results of Error Solver

Entering dt as $[1/10, .., 1/80]$ and $a = 1, b = 1, g = 1, k = 1, T = 3$, gave the following results:

(1.) RK4 $P = 3.9967$, and $C = 0.1548$

(2.) Leapfrog $P = 2.0020$, and $C = 3.3205$

(3.) Euler $P = 1.2043$, and $C = 23.4096$

These numbers are close to what was expected, however, Δt must be chosen sufficiently small for a given k.

CHAPTER 5

COMPUTING DERIVATIVES SPECTRALLY WITH THE FAST FOURIER TRANSFORM

We now want to add spatial dependence to our system of equations. The challenge is to numerically compute derivatives which we will do using the Fast Fourier Transform.

5.1 Fast Fourier Transform

We consider $f(x)$ periodic with period $L = 2\pi$ and want to get an approximation of $\partial_x f(x)$ using the FFT. Fourier analysis tells us $f(x) = \sum_k \hat{f}_k e^{ikx}$. Differentiating with respect to x gives $\partial_x f(x) = \sum_k (ik)\hat{f}_k e^{ikx}$. The spectral method for approximating derivatives achieves this by approximating the $\hat{f}_k$, doing this multiplication and then inverting the Fourier Transform. If we are given $f(x)$ and we know $\partial_x f(x)$ we can test this spectral differentiation by letting it approximate $\partial_x f(x)$ and compare it to the known derivative.

We do this in a few steps:

Step 1) Choose an N which will be the number of wavenumbers k in the FFT, e.g. 2, 4, 8, 16, or 32.

Step 2) Set up Δx where $\Delta x = L/N$.

Step 3) Define $x_j = 0 + (\Delta x)j = (\frac{j}{N})L$ where we will sample the function $f(x)$.

Step 4) Evaluate $f(x)$ at each x_j which gives us an array of values for $f(x)$ with N components giving us $\overrightarrow{f}$. We can also get $\overrightarrow{f_x}$ by evaluating the known $\partial_x f(x)$ at the points.

Step 5) We can now get $\overrightarrow{\hat{f}}$ by using Matlab's fft on our $\overrightarrow{f}$.

Step 6) We now multiply $\overrightarrow{\hat{f}}$ by ik to get $\overrightarrow{ik\hat{f}}$.

Step 7) Now we use Matlab's Inverse Fast Fourier Transform (ifft) to get the approximation of f_x. If we compare it to the known f_x at these points we can see the error of our method.

5.2 Compare FFT to Known Derivative

Consider the function $f(x) = e^{\cos x}$ and its derivative $-\sin x e^{\cos x}$ each of which is 2π periodic. As the table below shows the error gets very small for reasonable N:

N	error
4	1.1
8	2.7×10^{-2}
16	1.1×10^{-6}
32	2.7×10^{-14}

CHAPTER 6

APPROXIMATION IN SPACE AND TIME

Now that we know how to approximate spatial derivatives and that 4th Order Runge-Kutta is a good time stepping approximation, we want to approximate the wave in space and time and look at it.

6.1 Modified Water Wave Program

With the time stepping approximation we put in a and b which were constants representing two points in space. We then approximated what would happen to those two points as time went on. This was done using RK4 on equation (Equation 2.1). We now extend this to our linear water wave model for water of infinite depth:

$$\partial_t u = |D|v \tag{6.1a}$$

$$\partial_t v = -gu, \tag{6.1b}$$

where $|D|$ is the zeroth order approximation of the DNO:

$$|D|f(x) = \sum_{k=-\infty}^{\infty} |k|\hat{f}_k e^{ikx}. \tag{6.2}$$

6.2 Wave as an Initial Condition

We now consider arbitrary initial conditions $a(x)$ and $b(x)$ meant to represent the initial wave shape and velocity potential:

$$u(x,0) = a(x) \tag{6.3a}$$

$$v(x,0) = b(x). \tag{6.3b}$$

Now to approximate these initial data we use Fourier series:

$$a(x) = \sum_{k=-\infty}^{\infty} \hat{a}_k e^{ikx} \tag{6.4a}$$

$$b(x) = \sum_{k=-\infty}^{\infty} \hat{b}_k e^{ikx}. \tag{6.4b}$$

Where $\hat{a}_k = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} e^{-ikx} a(x) dx$ (BD04). Thus, our initial conditions transform to:

$$\hat{u_k}(0) = \hat{a_k} \tag{6.5a}$$

$$\hat{v_k}(0) = \hat{v_k}. \tag{6.5b}$$

Using the same Fourier analysis our linear water wave model becomes:

$$\partial_t u = |D|v \quad \Longrightarrow \quad \partial_t \hat{u_k}(t) = |k|\hat{v_k}(t) \tag{6.6a}$$

$$\partial_t v = -gu \quad \Longrightarrow \quad \partial_t \hat{v_k}(t) = -g\hat{u_k}(t). \tag{6.6b}$$

6.3 Linear Water Wave Problem

Using our RK4 solver we evolve solutions to (Equation 6.5) and (Equation 6.6). To recover the functions u and v we use the inverse Fourier Transform:

$$u(x,t) = \sum_{k=-\infty}^{\infty} \hat{u_k}(t)e^{ikx} \tag{6.7a}$$

$$v(x,t) = \sum_{k=-\infty}^{\infty} \hat{v_k}(t)e^{ikx}. \tag{6.7b}$$

6.4 Approximations

So finally we have an approximation for our wave in space and time. For the matrix of values of u and v at time values t^n and spatial values x_m we used Matlab's meshz function to give a 3-D graph with a reference plane drawn below it for u and v.

In Figure 4 we set $a(x) = e^{\sin x}$ and $b(x) = e^{\cos x}$, which are 2π periodic, and followed the evolution with $g = 1$, $L = 1$, $N = 64$, $T = 1$, and $\Delta t = 1/10$.

In Figure 5 we show a similar experiment with $a(x) = \sin x$ and $b(x) = \cos x$.

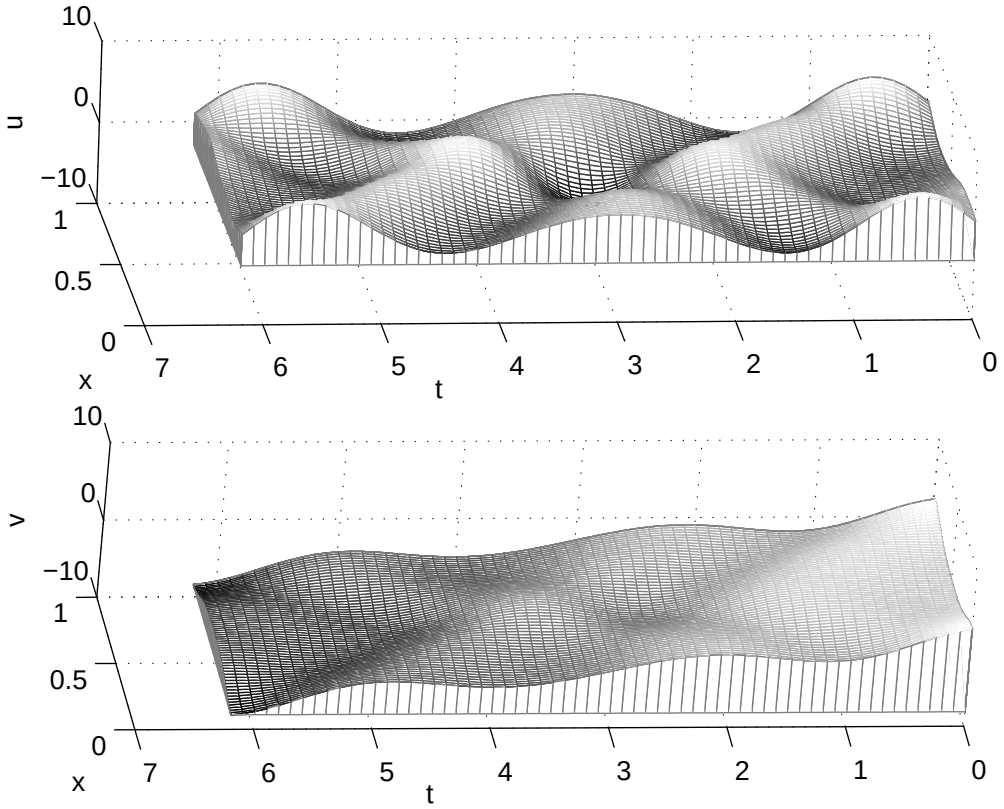

Figure 4. Linear Wave Experiment I

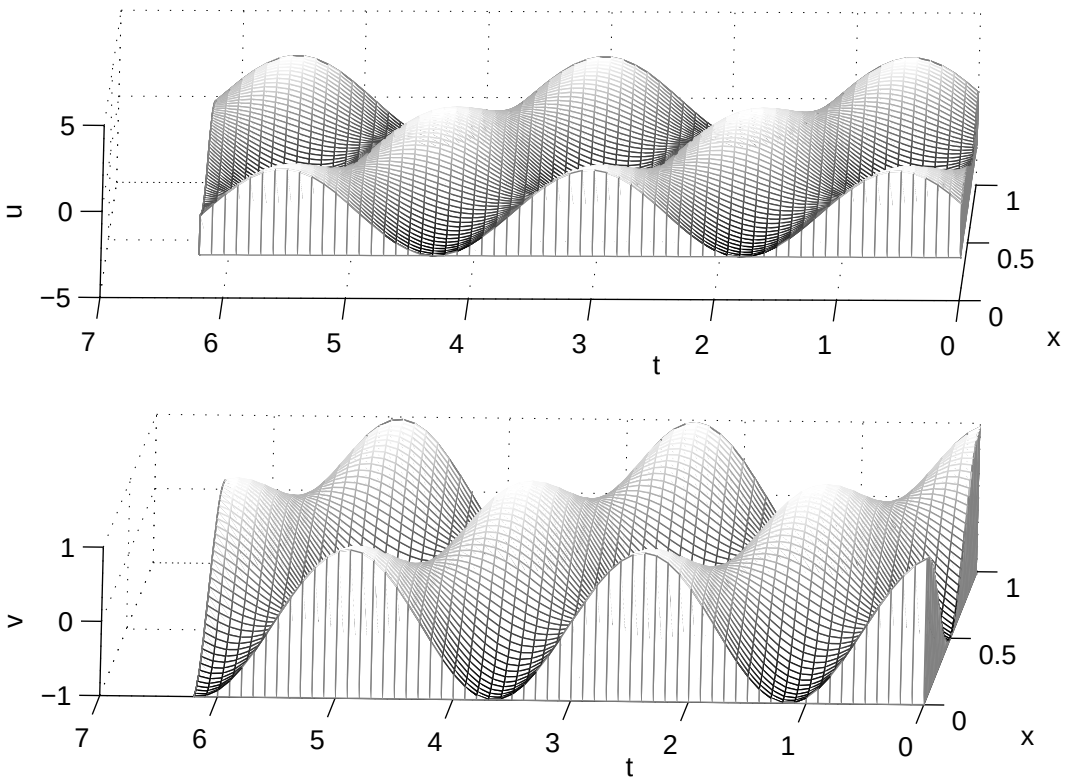

Figure 5. Linear Wave Experiment II

CHAPTER 7

LINEAR TRAVELING WATER WAVE

Now that we are able to approximate the wave in space and time we will check our approximation against an exact traveling wave solution. We are hoping to see a pattern after every period letting us know that our approximation is close to a traveling wave.

7.1 Derivation of the Exact Solution to the Traveling Wave

To get the equation for the traveling wave we start with (Equation 6.1) and (Equation 6.2) and change to a reference frame moving uniformly with speed c. In this frame (Equation 6.1) becomes:

$$\partial_t u - c\partial_x u = |D|v \tag{7.1a}$$

$$\partial_t v - c\partial_x v = -gu, \tag{7.1b}$$

7.1.1 Steady State Solution

The steady state solution where u and v are independent of time satisfy:

$$-c\partial_x u = |D|v \tag{7.2a}$$

$$-c\partial_x v = -gu. \tag{7.2b}$$

We assume that our solutions are 2π periodic and knowing that:

$$u(x) = \sum_{k=-\infty}^{\infty} \hat{u_k} e^{ikx} \tag{7.3a}$$

$$v(x) = \sum_{k=-\infty}^{\infty} \hat{v_k} e^{ikx}. \tag{7.3b}$$

Plugging u and v into our steady state equation we get:

$$\sum_{k=-\infty}^{\infty} -c(ik)\hat{u}_k e^{ikx} = \sum_{k=-\infty}^{\infty} |k|\hat{v}_k e^{ikx} \tag{7.4a}$$

$$\sum_{k=-\infty}^{\infty} -c(ik)\hat{v}_k e^{ikx} = \sum_{k=-\infty}^{\infty} -g\hat{u}_k e^{ikx}. \tag{7.4b}$$

7.1.2 Simplification

Factoring out like terms and removing the common summation then plugging into a system of ODEs gives the following for all k:

$$M_k(c)\begin{pmatrix}\hat{u}_k\\ \hat{v}_k\end{pmatrix}=\begin{pmatrix}-ick & -|k|\\ g & -ick\end{pmatrix}\begin{pmatrix}\hat{u}_k\\ \hat{v}_k\end{pmatrix}=\begin{pmatrix}0\\ 0\end{pmatrix}.$$

We will get a non-trivial (non-zero) solution $\iff detM_k = 0$. We set $detM_k = \Lambda(c,k) = -(ck)^2 + g|k|$. To find a traveling wave solution we pick an integer κ and then find a speed c_0 such that $\Lambda(c_0,\kappa) = 0$. This implies that $c_0\kappa = \pm\sqrt{g|\kappa|}$ and reducing further gives $c_0 = \pm\frac{\sqrt{g|\kappa|}}{\kappa}$. Finally we get $c_0 = \pm\sqrt{\frac{g}{\kappa}}$ where $\kappa > 0$.

7.1.3 Solving the Linear System

Now we want to solve our linear system to find our traveling wave. For $k \neq \kappa, detM_k(c_0) \neq 0$ so $\hat{u}_k = \hat{v}_k = 0$. If $k = \kappa$ then $detM_{\pm\kappa}(c_0) = 0$ and the augmented system we must solve is:

$$\begin{pmatrix}-ic_0\kappa & |\kappa| & | & 0\\ -g & -ic_0\kappa & | & 0\end{pmatrix},$$

this is row equivalent to:

$$\begin{pmatrix}-ic_0\kappa & |\kappa| & | & 0\\ 0 & -ic_0\kappa - \frac{g|\kappa|}{ic_0\kappa} & | & 0\end{pmatrix}. \tag{7.5}$$

The solutions of this linear system are:

$$\hat{u}_{+\kappa} = (|\kappa|)\frac{\alpha}{2} \tag{7.6a}$$

$$\hat{v}_{+\kappa} = (ic_0\kappa)\frac{\alpha}{2} \tag{7.6b}$$

$$\hat{u}_{-\kappa} = (|\kappa|)\frac{\beta}{2} \tag{7.6c}$$

$$\hat{v}_{-\kappa} = (-ic_0\kappa)\frac{\beta}{2}, \tag{7.6d}$$

where $\alpha \in \mathbb{C}$ so $\alpha = \rho e^{i\theta}$ where ρ, θ are real. To find real solutions require that $\alpha = \bar{\beta}$ so that

$$\begin{aligned} u(x) &= \hat{u}_\kappa e^{i\kappa x} + \hat{u}_{-\kappa} e^{-i\kappa x} \\ &= \frac{\rho}{2} e^{i\theta}|\kappa| e^{i\kappa x} + \frac{\rho}{2} e^{-i\theta}|\kappa| e^{-i\kappa x} \\ &= \rho|\kappa| \cos(\kappa x + \theta). \end{aligned}$$

In a similar fashion

$$v(x) = \rho c_0 \kappa \sin(\kappa x + \theta).$$

7.1.4 Exact Solution to the Traveling Wave

Moving back to our original coordinates:

$$u(x,t) = \rho|\kappa|\cos(|\kappa|(x - c_0 t) + \phi) \tag{7.7a}$$

$$v(x,t) = \rho c_0 \kappa \sin(|\kappa|(x - c_0 t) + \phi) \tag{7.7b}$$

where $c_0 = \frac{\sqrt{g|\kappa|}}{\kappa}$

7.2 Matlab Error Approximations

We can now use these to test our linear code. Using (Equation 7.7) for u and v and plugging them into my wave function with $\rho = 1, \kappa = 2, \phi = 0, N_x = 32, N_t = 64$ and $g = 1$ gives Figure 6a showing error for u and v. With $\rho = 1, \kappa = 4, \phi = 0, N_x = 32, N_t = 64$ and $g = 1$ we produce Figure 6b showing error for u and v. It is apparent that as κ increases the error gets worse.

With $\rho = 1, \kappa = 2, \phi = 0, T = 10, N_x = 32, N_t = 628$ and $g = 1$ we realize Figure 7a showing error for u and v. With $\rho = 1, \kappa = 4, \phi = 0, T = 10, N_x = 32, N_t = 628$ and $g = 1$ we find Figure 7b showing error for u and v. It is apparent that as time increases the error also gets worse. For larger values of κ one must choose Δx and Δt much smaller to see reliable results.

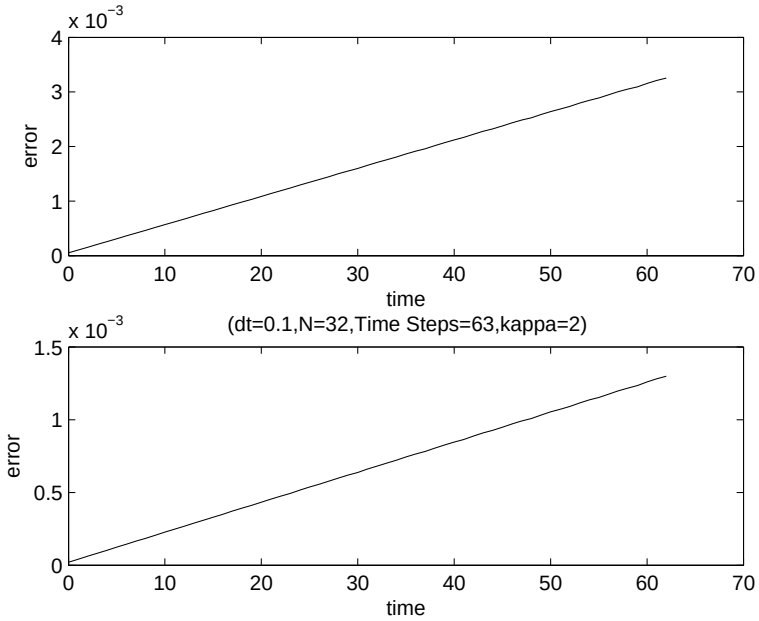

Figure 6a. $\kappa = 2$, T=1

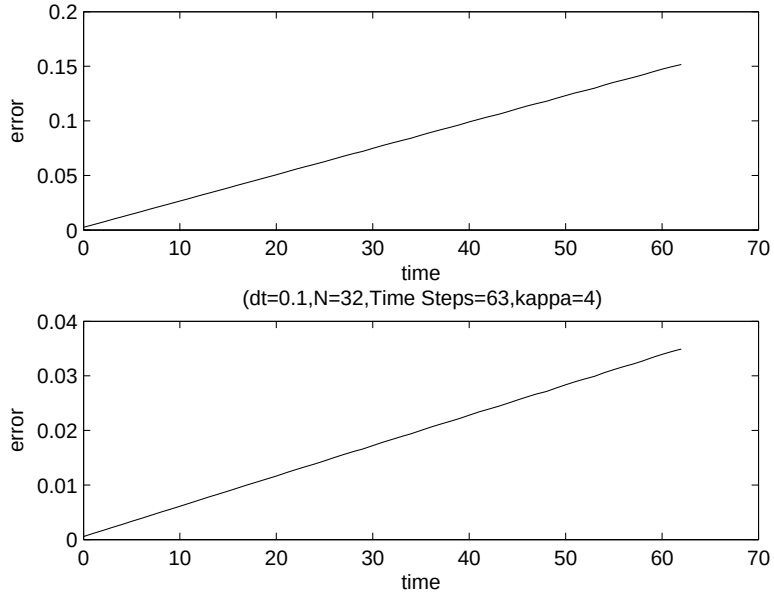

Figure 6b. $\kappa = 4$, T=1

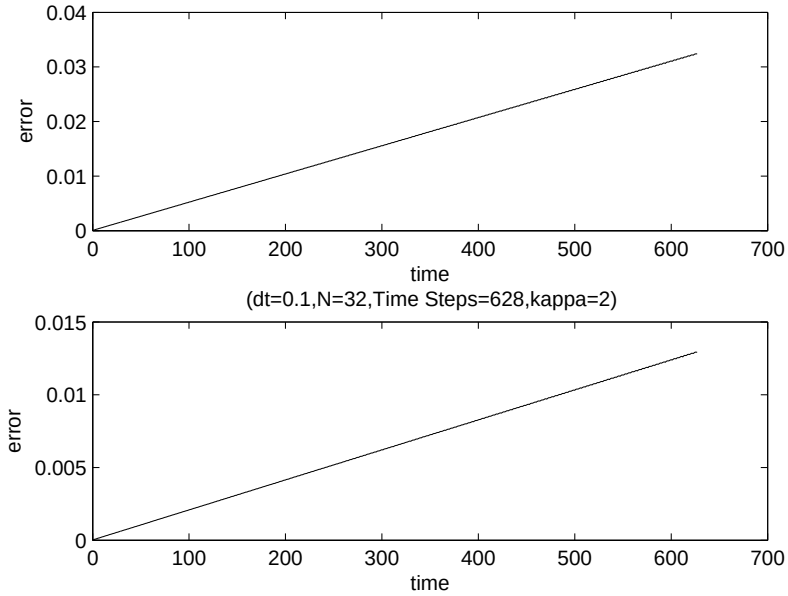

Figure 7a. $\kappa = 2$, T=10

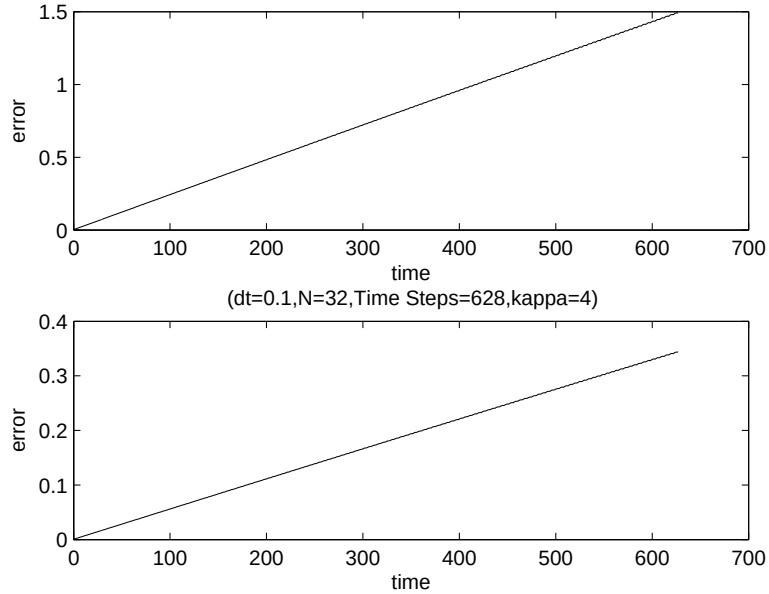

Figure 7b. $\kappa = 4$, T=10

CHAPTER 8

A NONLINEAR WATER WAVE MODEL

We are now in a position to consider our nonlinear water wave model:

$$\partial_t \eta = |D|[\xi] - \partial_x[\eta \partial_x \xi] - |D|[\eta(|D|[\xi])] \tag{8.1a}$$

$$\partial_t \xi = -g\eta - \frac{1}{2}\{(\partial_x \xi)^2 - (|D|[\xi])^2\}. \tag{8.1b}$$

We will once again consider Euler, Leap-Frog, and RK4 time-stepping methods combined with a spectral method to compute derivatives. To deal with the nonlinear terms we make use of the fact that on the physical side multiplication of two functions is straight forward as it is simply pointwise multiplication.

For example to compute $\partial_x[\eta \partial_x \xi]$ at the points x_j we follow the steps:

1. Define $\overrightarrow{a} \in \mathbb{R}^N$ by $a_j = \xi(x_j)$
2. Use the FFT to find $\overrightarrow{b} = \text{fft}(\overrightarrow{a})$; the b_k approximate $\hat{\xi}_k$
3. Define $\overrightarrow{c}$ by $c_k = (ik)b_k$
4. Use the iFFT to define $\overrightarrow{d} = \text{ifft}(\overrightarrow{c})$; the d_j approximate $(\partial_x \xi)(x_j)$
5. Define $\overrightarrow{e} \in \mathbb{R}^N$ by $e_j = \eta(x_j)$
6. Compute $\overrightarrow{f}$ by $f_j = e_j d_j$
7. Use the FFT to find $\overrightarrow{g} = \text{fft}(\overrightarrow{f})$.

8. Define $\overrightarrow{h}$ by $h_k = (ik)gk$

9. Use the iFFT to define $\overrightarrow{m} = \text{ifft}(\overrightarrow{h})$. These m_j approximate $\partial_x[\eta\partial_x\xi](x_j)$.

Using similar steps to the above, one can compute each of the new nonlinear terms.

8.1 Nonlinear System Analysis

We now use the traveling wave solution (Equation 7.7) of the linear water wave problem at an initial condition. We found it necessary to choose ρ rather small to avoid blow up. In Figure 8a we display results with $\rho = 0.001$ for a moderate time, however, we see in Figure 8b and Figure 8c when ρ is larger that instability eventually sets in.

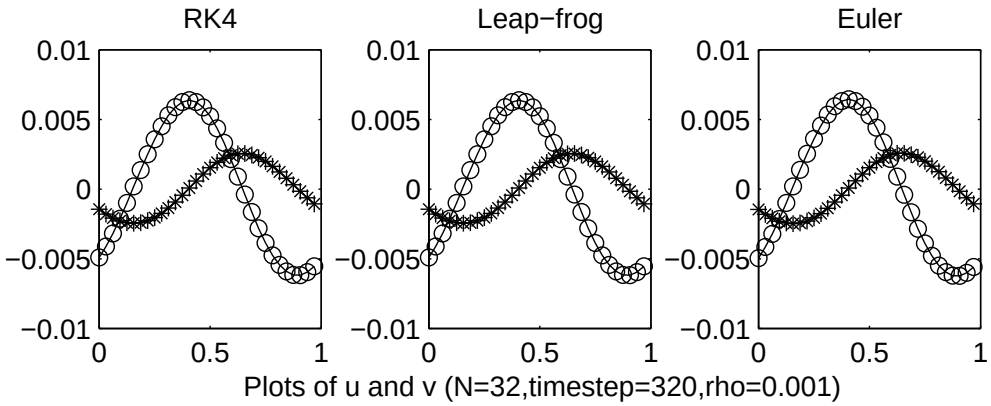

Figure 8a. Non-Linear System 1, $\rho = 0.001$

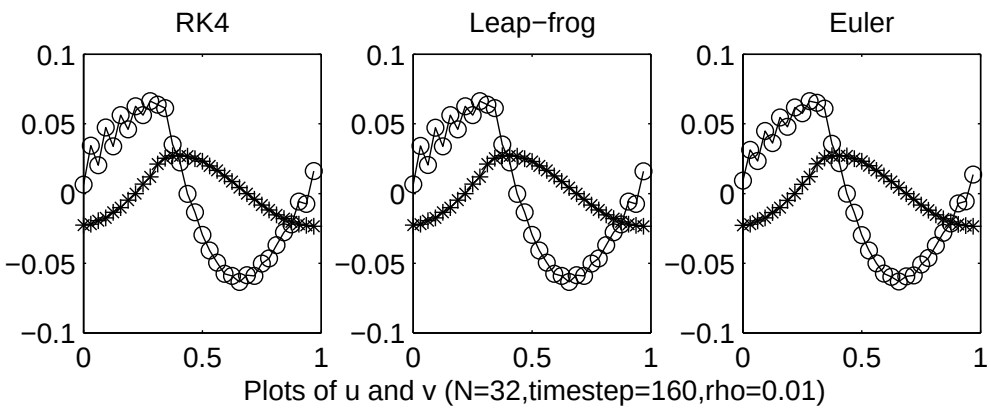

Figure 8b. Non-Linear System 2, $\rho = 0.01$

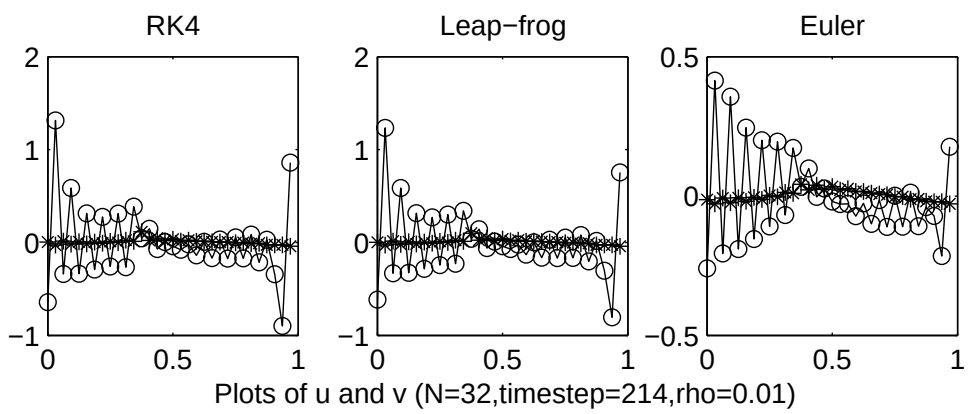

Figure 8c. Non-Linear System Blow Up, $\rho = 0.01$

CHAPTER 9

NONLINEAR EVOLUTION OF SURFACE WATER WAVES

Our goal in this section is to demonstrate that for certain initial conditions a fully nonlinear model is necessary to correctly model the propagation of water waves.

9.1 Gaussian Bell Curve

Consider the Gaussian function:

$$y = \frac{1}{\sqrt{2\pi}} \quad e^{\frac{-x^2}{2\sigma}}. \tag{9.1}$$

Depending on the value of σ we can vary the curve from a gradual bell shape to a sharp bell shape. We can also multiply the equation by a constant which will increase the amplitude of the bell without affecting the rest of the initial condition. Using these variables we can observe differences in the solutions and see where our nonlinear model is more appropriate.

In Figure 9 we display the evolution of a Gaussian with amplitude 56, $\sigma = 1.25 \times 10^{-4}$ and let time go to 100. Notice the difference between the linear and non-linear is up to 1.4 kilometers. These measurements are both unrealistic and nonphysical, however the linear code makes an error of $[1.4/20] \times 100 = 7\%$, which can be very significant for open ocean structures.

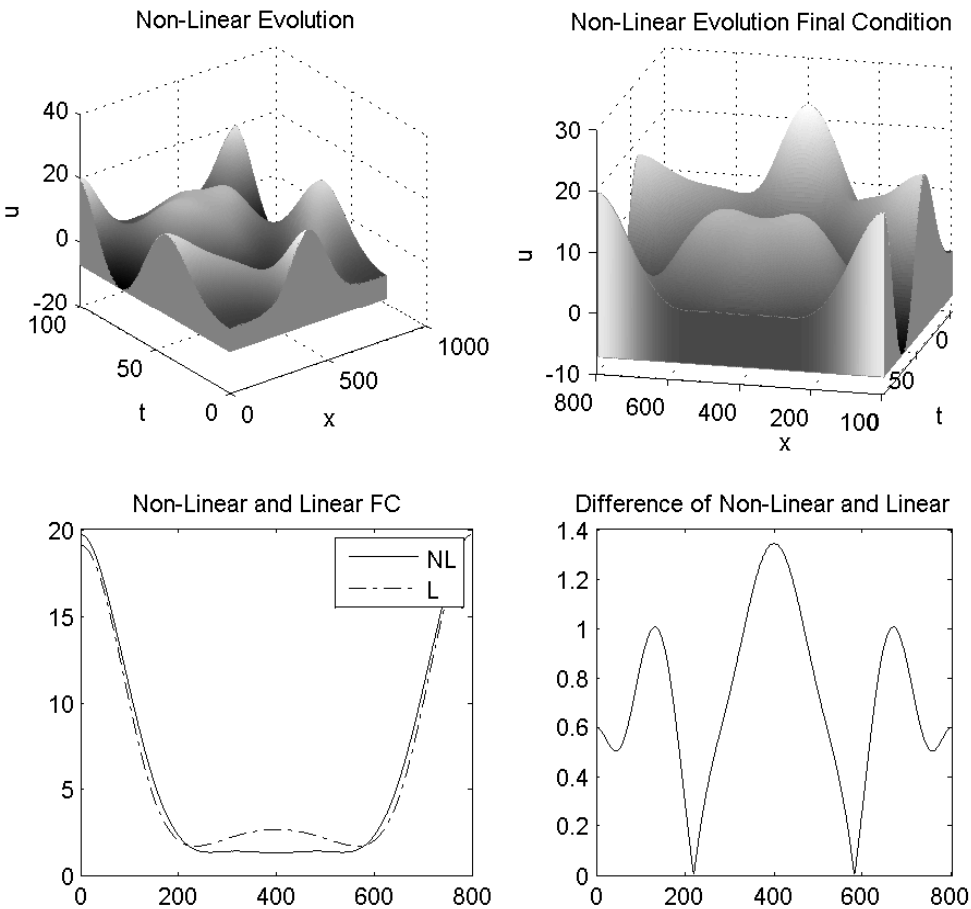

Figure 9. Gaussian with Linear vs. Nonlinear

CHAPTER 10

TSUNAMI EVOLUTION

10.1 Tsunami Source

Our goal is to simulate tsunami propagation and, given the models we've built thus far, all we need now is appropriate initial conditions. Considerable work on this topic has been done by Dias and Dutykh (see, e.g., (DD07)). Generally the life of a tsunami is divided into three phases: the generation (tsunami source), the propagation and the inundation. Dias and Dutykh give a comprehensive treatment of the mechanisms and modeling of tsunami generation and we seek to understand the propagation of surfaces shortly after the earthquake event.

This paper deals with tsunamis created by different underwater earthquakes. We consider three different initial waves proposed by Dias and Dutykh caused by underwater earthquakes. These dislocations of the ocean floor are defined as strike-slip, dip-slip, and tensile. The dislocation can be defined by three angles: the dip angle δ of the fault, the slip angle θ, and the angle ϕ between the fault plane and the Burger's vector D.

10.2 Finite Depth

Up until now we have been dealing with an ocean of infinite depth but from now on all the codes will incorporate depth of the water. To do this we simply use the operator G_0 in its finite depth form:

$$G_0[\xi] = \sum_{p=-\infty}^{\infty} |p| \tanh(h|p|)\hat{\xi}_p =: |D| \tanh(h|D|)[\xi],$$

which gives

$$G_1(\eta)[\xi] = \partial_x \left[\eta \partial_x \xi\right] - G_0 \left[\eta G_0[\xi]\right].$$

Now we simply replace $G_0[\xi]$ with $|D| \tanh(h|D|)[\xi]$ in our non-linear or linear equations. For example the non-linear equation is now:

$$\partial_t \eta = |D| \tanh(h|D|)[\xi] - \partial_x[\eta \partial_x \xi] - |D| \tanh(h|D|)[\xi][\eta(|D| \tanh(h|D|)[\xi])] \tag{10.1a}$$

$$\partial_t \xi = -g\eta - \frac{1}{2}\{(\partial_x \xi)^2 - (|D| \tanh(h|D|)[\xi])^2\}. \tag{10.1b}$$

10.3 Tsunami Initial Wave Shape

Using the three dimensional formulas from Dias and Dutykh's paper we extracted a two dimensional "slice" of the data and evolved this in our linear and non-linear codes. The first tsunami initial condition is a result of a tensile fault and is shown in Figure 10. The second tsunami is a result of a strike-slip fault and is depicted in Figure 11. The final and most interesting tsunami is a result of a dip-slip fault; this is shown in Figure 12. In these graphs the vertical component representing the wave amplitude is divided by 30.

We can now input these tsunamis into our models and follow the linear and nonlinear evolution.

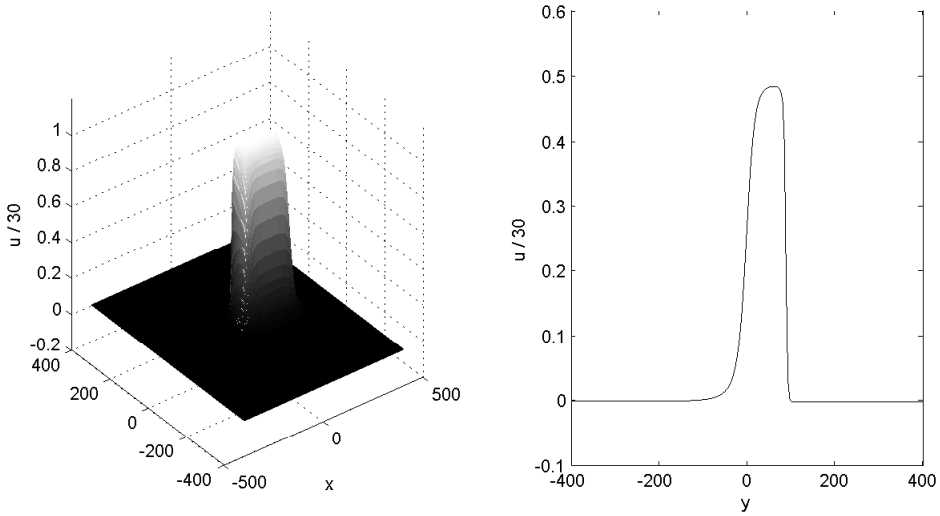

Figure 10. Tensile Fault Tsunami Initial Condition

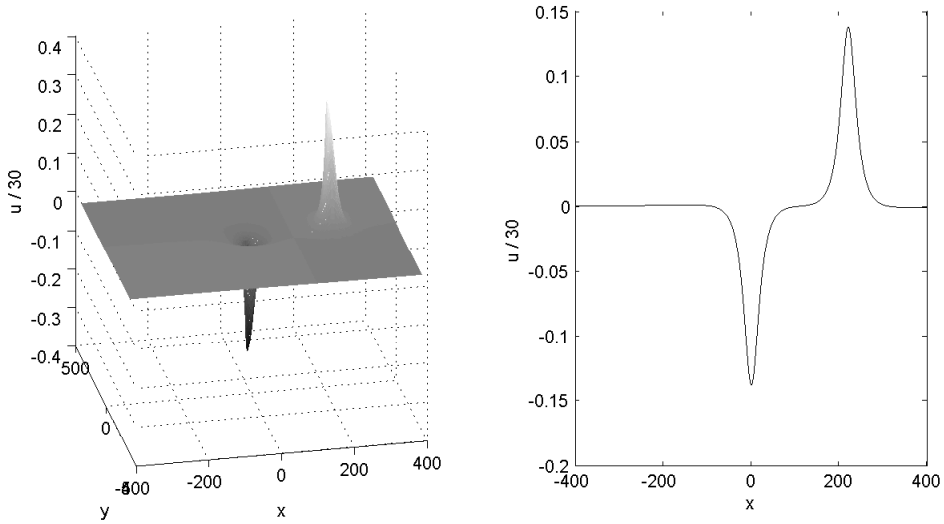

Figure 11. Strike Slip Fault Tsunami Initial Condition

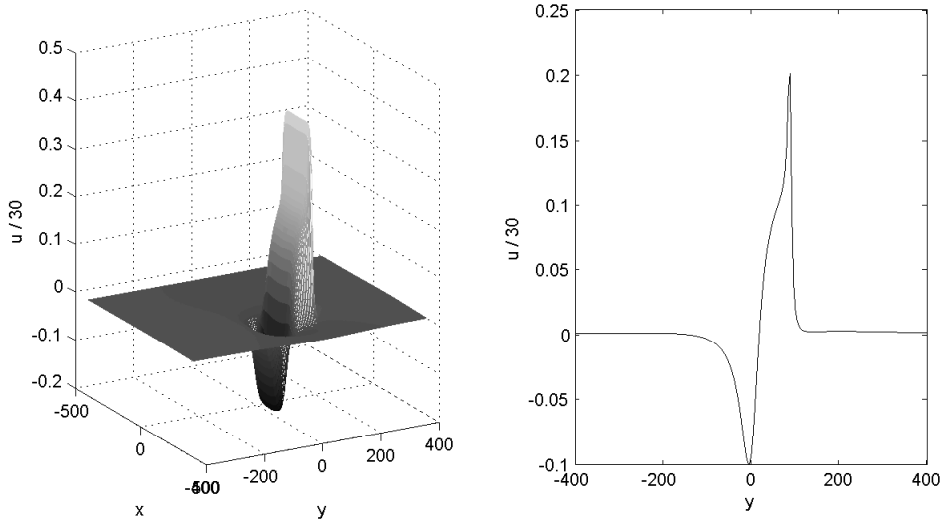

Figure 12. Dip-Slip Fault Tsunami Initial Condition

10.4 Linear and Nonlinear Evolution of Tsunamis

The results of our simulation are given in Figure 13 (tensile fault), Figure 14 (strike-slip fault), and Figure 15 (dip-slip fault). Each is plotted over a test area of 2000 km with depth 7 km. In each the full non-linear evolution is shown from both the initial view and the final view, while we also show the final waveforms and the difference at the final time between the linear and nonlinear propagation. In these graphs the vertical component representing the wave amplitude is divided by 30.

We notice that the difference between the linear and nonlinear results are negligible indicating that a simple linear analysis suffices to describe the initial evaluation of tsunamis.

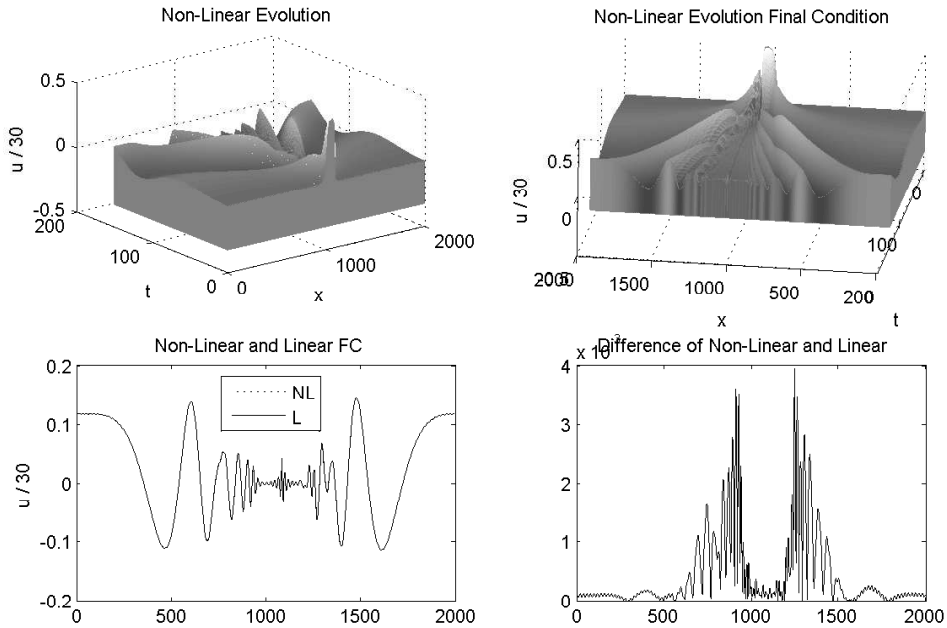

Figure 13. Tensile Fault Tsunami T=150

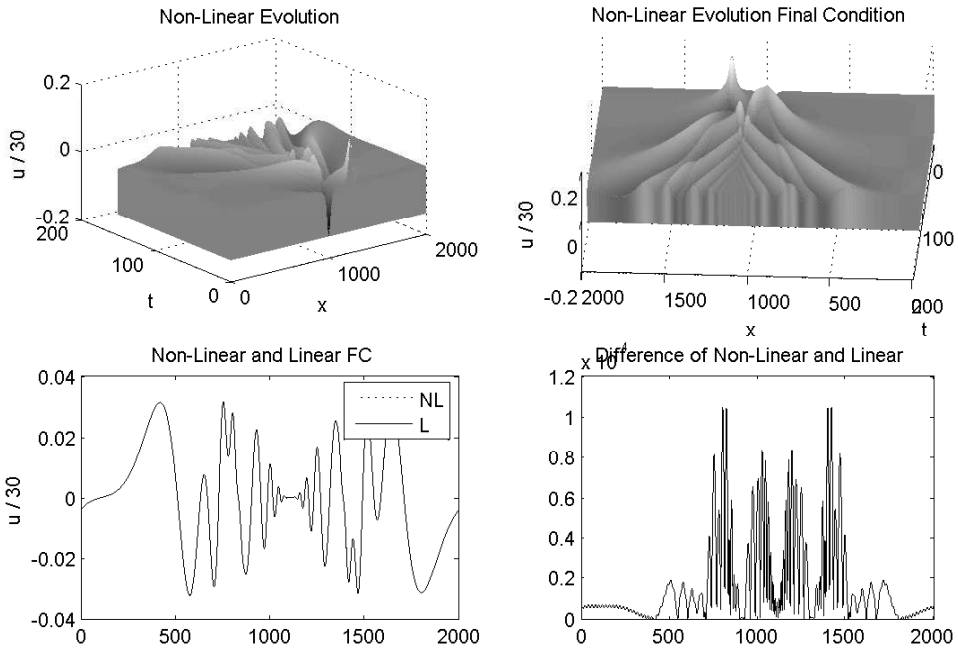

Figure 14. Strike Slip Fault Tsunami T=150

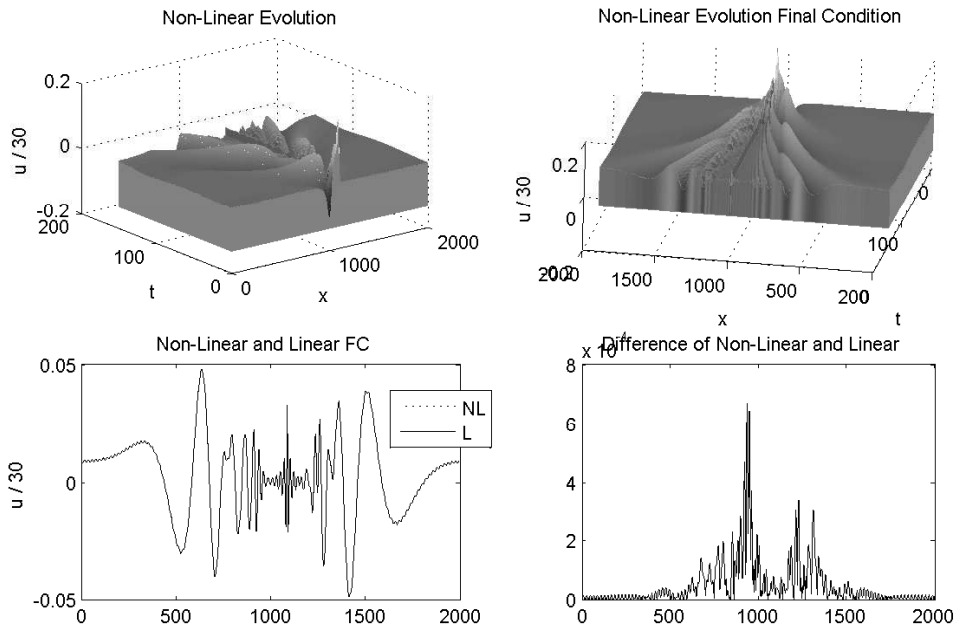

Figure 15. Dip-Slip Fault Tsunami T=150

SUMMARY

Our goal in this thesis was to use nonlinear models to evolve tsunamis. We started by explaining the basics of the water wave equation. We then solved an exact solution in time. We tested the Euler Method, the Leap-Frog Method, and the 4th Order Runge-Kutta Method against the exact solution. Using Fast Fourier Transforms we developed a spatial approximation and together with our time approximation methods were able to model the wave in space and time. We tested our model on smooth waves and on traveling waves to test the error. We then developed a nonlinear model and tested it on the Gaussian function. Finally we developed tsunami initial conditions and observed their evolution. We found that the linear and nonlinear evolution were virtually identical. Thus in our initial evaluation of the propagation of tsunamis it may be sufficient to use the linear model.

CITED LITERATURE

D. J. Acheson. *Elementary fluid dynamics*. The Clarendon Press Oxford University Press, New York, 1990.

William E Boyce and Richard C. DiPrima. *Elementary Differential Equations and Boundary Value Problems*. SIAM, Philadelphia, eighth edition, 2004.

R. Coifman and Y. Meyer. Nonlinear harmonic analysis and analytic dependence. In *Pseudodifferential operators and applications (Notre Dame, Ind., 1984)*, pages 71–78. Amer. Math. Soc., 1985.

Walter Craig and David P. Nicholls. Traveling two and three dimensional capillary gravity water waves. *SIAM J. Math. Anal.*, 32(2):323–359, 2000.

Walter Craig and Catherine Sulem. Numerical simulation of gravity waves. *Journal of Computational Physics*, 108:73–83, 1993.

Walter Craig, Ulrich Schanz, and Catherine Sulem. The modulation regime of three-dimensional water waves and the Davey-Stewartson system. *Ann. Inst. Henri Poincaré*, 14:615–667, 1997.

Frédéric Dias and Denys Dutykh. Dynamics of tsunami waves. *Extreme Man-Made and Natural Hazards in Dynamics of Structures*, pages 1–25, 2007.

Curtis F. Gerald and Patrick O. Wheatley. *Applied Numerical Analysis*. Pearson, New York, seventh edition, 2004.

Bei Hu and David P. Nicholls. Analyticity of Dirichlet–Neumann operators on Hölder and Lipschitz domains. *SIAM J. Math. Anal.*, 37(1):302–320, 2005.

Horace Lamb. *Hydrodynamics*. Cambridge University Press, Cambridge, sixth edition, 1993.

David P. Nicholls and Fernando Reitich. A new approach to analyticity of Dirichlet-Neumann operators. *Proc. Roy. Soc. Edinburgh Sect. A*, 131(6):1411–1433, 2001.

John C Strikwerda. *Finite Difference Schemes and Partial Differential Equations*. Wiley, New Jersey, second edition, 2005.

Vladimir Zakharov. Stability of periodic waves of finite amplitude on the surface of a deep fluid. *Journal of Applied Mechanics and Technical Physics*, 9:190–194, 1968.