ADDIS ABABA UNIVERSITY

COLLEGE OF NATURAL SCIENCE
DEPARTEMENT OF MATHEMATICS

Explicit Finite Difference Scheme for the Finite String

Prepared by: Solomon Amsalu

Advisor: Tadesse Abdi(Ph.D)

A Project Submitted to the Department of Mathematics, Addis Ababa University in Partial Fulfillment of the Requirements of the Master of Science Degree in Mathematics

August, 2014

Contents

Addis Ababa University
Department of Mathematics

The undersigned hereby certify that they have read and recommend to the School of Graduate Studies for acceptance of a project entitled **Explicit Finite Difference Scheme for the Finite String** by Solomon Amsalu in partial fulfillment of the requirements for the degree of Master of Science in Mathematics.

Dated: August, 2014

Advisor: ________________________
Dr.Tadesse Abdi

Examining committee: ________________________

____

August, 2014

Acknowledgement

First of all, I would like to express my deepest gratitude to God for giving me patience; I am also grateful to my advisor Dr. Tadesse Abdi for his helpful discussion, comments and providing the necessary materials in the preparation of this paper.

Secondly I would like to extend my thanks to my families and all who encouraged me to complete my project and their heart full help while writing the paper. Lastly I would like thanks to Department of mathematics, Addis Ababa University for giving the necessary materials throughout the preparation of this paper.

Abstract

This project provides a practical overview of numerical solutions to the wave equation of finite string using the finite difference method. The second order centered differences for time and space is applied to a simple problem involving the one-dimensional wave equation of finite string which lead to an explicit numerical scheme. It also allow the reader to experiment with the consistency, stability and convergency of explicit finite difference scheme for finite string and an example with working Matlab code for the scheme is presented.

Notations

$\mathbb{R}$	The set of all real numbers.
$U = U(x,t)$	The vertical displacement of the vibration of a string.
c	Velocity
Δt	The local distance between adjacent time steps.
Δx	The local distance between adjacent points in space.
$u^{(n)}$	The n^{th} derivative of u
$U_x(x_0)$	The derivative of U with respect to x evaluated at $x = x_0$.
$O(h^n)$	Discretization error or Truncation error.
$\|.\|$	Norm of a function or vector.

Introduction

The finite difference approximations for derivatives are one of the simplest and of the oldest methods to solve differential equations. It was already known by L. Euler (1707-1783), in one dimension of space and was probably extended to dimension two by C. Runge (1856-1927). The advent of finite difference techniques in numerical applications began in the early 1950s and their development was stimulated by the emergence of computers that offered a convenient framework for dealing with complex problems of science and technology. Theoretical results have been obtained during the last five decades regarding the accuracy, stability and convergence of the finite difference method for partial differential equations.

The finite difference method consists of replacing each derivative by a difference quotient in the classic formulation. It is simple to code and economic to compute. In a sense, a finite difference formulation offers a more direct approach to the numerical solution of partial differential equations than does a method based on other formulations. It consists in approximating the differential operator by replacing the derivatives in the equation using differential quotients. This project is organized into three chapters. In the first chapter, we consider basic notions of wave equation of finite string, finite difference method, deriving finite difference approximations and properties of the finite difference equation. The second chapter deals with how to solve wave equation of finite string using explicit finite difference method. The third chapter deals with consistency, stability and convergency of the scheme. And the fourth chapter deals with Matlab version of the explicit finite difference scheme of wave equation of finite string .

Chapter 1

Preliminaries

1.1 Wave Equation of Finite String

Consider a stretched string of length L with ends fastened on the x-axis at $x = 0$ and $x = L$. Suppose that the string is set to vibrate by displacing it from its equilibrium position and then releasing it. Assuming that the string vibrates only in a fixed plane, we let $U(x,t)$ denote the transverse displacement at time $t \geq 0$ of the point on the string at position x. In particular, $U(0,x)$ denotes the initial shape of the string. We wish to determine the subsequent motion of the string by finding $U(t,x)$ for $t > 0$ and $0 < x < L$. The one dimensional wave equation of finite string is:

$$U_{tt} = c^2 U_{xx} \quad 0 < x < L,\ t > 0 \tag{1.1}$$

To find U, we will solve this equation subject to the boundary conditions:

$$U(t,0) = 0 \ \ and \ \ U(t,L) = 0 \ \ for\ all \ \ t > 0 \tag{1.2}$$

and the initial conditions:

$$U(0,x) = f(x), \quad U_t(0,x) = g(x) \ \ for \ \ 0 < x < L \tag{1.3}$$

where, $U = U(t,x)$ is the vertical displacement of the vibration of a string and c is the velocity.

The boundary conditions state that the ends of the string are held fixed for all time, while the initial conditions give the initial shape of the string $f(x)$ and its initial velocity $g(x)$.

1.2 Finite Difference Method

The finite difference approach is one of the premier mathematical tools employed to solve partial differential equations. It is a means of obtaining numerical solutions to partial differential equations. The most common finite difference methods for the solution of partial differential equation are:

- Explicit method
- Implicit method
- Crank Nicolson Method

These are closely related but differ in stability, accuracy and execution speed. In the formulation of a partial differential equation problem, there are three components to be considered:

- The partial differential equation.
- The region of space-time on which the partial differential equation is required to be satisfied.
- The boundary and initial conditions to be met.

The application of finite difference method to a particular differential equation problem includes the following steps:

1. Construction of a discrete finite-difference model of the problem:

  - coverage of the computational domain by a space-time grid,
  - approximations to derivatives, functions, initial and/or boundary condition all at the grid point.
  - construction of a system of the finite-difference (i.e., algebraic) equations.

2. Analysis of the finite-difference model:

  - consistency and order of the approximation
  - stability
  - convergence

3. Numerical computations

1.2.1 Grid (Mesh)

The mesh (grid) is the set of locations where the discrete solution is computed. These points are called nodes, and if one were to draw lines between adjacent nodes in the domain the resulting image would resemble a net or mesh. Two key parameters of the mesh are Δx the local distance between adjacent points in space, and Δt, the local distance between adjacent time steps.

The domain is partitioned in space and in time and approximations of the solution are computed at the space or time points. In addition, there are some practically useful schemes that can fail to yield a solution for bad combinations of Δx and Δt.

Consider a computational domain in the two-dimensional space of variables (t, x). Cover this space by a grid of discrete points (t_i, x_j) given by

$$t_i = t_0 + i\Delta t$$
$$x_j = x_0 + j\Delta x,$$

$$\text{where } i = 0, 1, 2, ... \text{ and } j = 0, 1, 2, ...$$

Here, Δx is usually called grid spacing, and Δt is called time step since t usually represents time. At the grid points a function $u(t, x)$ is to be approximated by a grid function $U(t_i, x_j)$. A value of $u(t_i, x_j)$ can be denoted by $u_{i,j}$ while approximation to $u_{i,j}$ can be denoted by $U_{i,j}$.

A spatial grid that is the most appropriate for the problem under consideration should be chosen. In many applications, the regular (uniform) grid with the grid spacing Δx is a natural and reasonable choice.

1.2.2 Truncation Error

In the limit as the mesh spacing (Δx and Δt) or the step size goes to zero, the numerical solution obtained with any useful scheme will approach the true solution to the original differential equation. However, the rate at which the numerical solution approaches the true solution varies with the scheme. The error between the numerical solution and the exact solution is determined by the error that is committed by going from a differential operator to a difference operator. This error is called the discretization error or truncation error. The term truncation error reflects the fact that a finite part of a Taylor series is used in the approximation.

1.3 Deriving Finite Difference Approximations

The finite difference method works by replacing the region over which the independent variables in the partial differential equation are defined by the finite grid (mush) of points at which the dependent variable is approximated. The partial derivatives in the partial differential equation at each grid point are approximated from neighboring values by using Taylor series.

1.3.1 Taylor Series

The Taylor series expansion of the function $u(x)$ about the point $x = x_0$ is given by the formula

$$u(x) = \sum_{n=0}^{\infty} \frac{u^{(n)}(x_0)}{n!}(x - x_0)^n \tag{1.4}$$

where

- $u^{(n)}(x_0) = \frac{d^n u}{dx^n}$ at $x = x_0$
- $u^{(0)}(x_0) = u(x_0)$

If we let $x = x_0 + h$, then $x - x_0 = h$, and the series can be written as

$$u(x_0 + h) = \sum_{n=0}^{\infty} \frac{u^{(n)}(x_0)}{n!}.h^n \tag{1.5}$$

$$= u(x_0) + \frac{u^{'}(x_0)}{1!}.h + \frac{u^{''}(x_0)}{2!}.h^2 + O(h^3), \tag{1.6}$$

where the expression $O(h^3)$ represents the remaining terms of the series and indicates that the leading term is of order h^3. Because h is a small quantity, we can write $0 < h < 1$, and $h > h^2 > h^3 > h^4 > \cdots$. Therefore, the remaining of the series represented by $O(h^3)$ provides the order of the error incurred in neglecting this part of the series expansion when calculating $u(x_0 + h)$.

From the Taylor series expansion shown above we can obtain an expression for the derivative $u^{'}(x_0)$ as

$$u^{'}(x_0) = \frac{u(x_0 + h) - u(x_0)}{h} - \frac{u^{''}(x_0)}{2!}.h - O(h^2) \tag{1.7}$$

$$= \frac{u(x_0 + h) - u(x_0)}{h} - O(h) \tag{1.8}$$

In practical applications of finite differences, we will replace the first-order derivative $\frac{du}{dx}$ at $x = x_0$, with the expression $\frac{u(x_0+h)-u(x_0)}{h}$, selecting an appropriate value for h, and indicating that the error introduced in the calculation is of order h, i.e., $error = O(h)$.

1.3.2 Taylor Series Applied to the Finite Difference Method

Taylor series have been widely used to study the behavior of numerical approximation to differential equations. Let $U(x)$ has n continuous derivatives over the interval (a, b). Then for $a < x_0,\ x_0 + h < b$,

$$U(x_0+h) = U(x_0)+hU_x(x_0)+h^2\frac{U_{xx}(x_0)}{2!}+...+h^{n-1}\frac{U_{(n-1)}(x_0)}{(n-1)!}+O(h^n), \quad (1.9)$$

Where,

- $U_x = \frac{dU}{dx},\ U_{xx} = \frac{d^2U}{dx^2},\ ...,\ U_{(n-1)} = \frac{d^{n-1}U}{dx^{n-1}}$
- $U_x(x_0)$ is the derivative of U with respect to x evaluated at $x = x_0$.
- $O(h^n)$ is discretization error or truncation error.

The usual interpretation of Taylor series says that if we know the value of U and the values of its derivatives at point x_0 then we can write down the equation (1.9) for its value at the (nearby) point $x_0 + h$. This expression contains an unknown quantity which is written in as $O(h^n)$ and pronounced 'order h to the n'. If we discard the term $O(h^n)$ in equation (1.9) (i.e. truncate the right hand side of equation (1.9) we get an approximation is $O(h^n)$.

In the finite difference method we know the U values at the grid points and we want to replace partial derivatives in the partial differential equation we are solving by approximations at this grid points. We do this by interpreting equation (1.9) in another way. In the finite difference method both x_0 and $x_0 + h$ are grid points and $U(x_0)$ and $U(x_0 + h)$ are known. This allows us to rearrange equation (1.14).

1.3.3 Simple Finite Difference Approximation to a Derivative

First Order Forward Difference

Truncating (1.9) after the first derivative term gives,

$$U(x_0 + h) = U(x_0) + hU_x(x_0) + O(h^2) \quad (1.10)$$

Rearranging (1.9) gives,

$$\begin{aligned} U_x(x_0) &= \frac{U(x_0+h) - U(x_0)}{h} + \frac{O(h^2)}{h} \\ &= \frac{U(x_0+h) - U(x_0)}{h} + O(h) \end{aligned}$$

Neglecting the $O(h)$ term gives,

$$U_x(x_0) \approx \frac{U(x_0+h) - U(x_0)}{h} \tag{1.11}$$

Equation (1.11) is called a first order finite difference approximation to $U_x(x_0)$ since the approximation error (i.e.$O(h)$) which depends on the first power of h. This approximation is called a forward finite difference approximation since we start at x_0 and step forwards to the point $x_0 + h$, h is called the step size ($h > 0$).

First Order Backward Difference

An alternative first order finite difference formula is obtained if the Taylor series like that in equation (1.10) is written with $h = -h$. Using the discrete mesh variables in place of all the unknowns, one obtains

$$U(x_0 - h) = U(x_0) - hU_x(x_0) + O(h^2) \tag{1.12}$$

Notice the alternating signs of terms on the right hand side.

Rearranging (1.11) gives,

$$\begin{aligned} U_x(x_0) &= \frac{U(x_0) - U(x_0-h)}{h} + \frac{O(h^2)}{h} \\ &= \frac{U(x_0) - U(x_0-h)}{h} + O(h) \end{aligned}$$

Neglecting the $O(h)$ term gives,

$$U_x(x_0) \approx \frac{U(x_0) - U(x_0-h)}{h} \tag{1.13}$$

Equation (1.13) is called the backward difference formula because it involves the values of U at x_0 and $x_0 - h$. The order of magnitude of the truncation error for the backward difference approximation is the same as that of the forward difference approximation.

First Order Central Difference

The Taylor series expansions for $U(x_0+h)$ and $U(x_0-h)$ are:

$$U(x_0+h) = U(x_0) + hU_x(x_0) + h^2\frac{U_{xx}(x_0)}{2!} + -h^3\frac{U_{xxx}(x_0)}{3!} + ... \quad (1.14)$$

$$U(x_0-h) = U(x_0) - hU_x(x_0) + h^2\frac{U_{xx}(x_0)}{2!} - h^3\frac{U_{xxx}(x_0)}{3!} + ... \quad (1.15)$$

Subtracting equation (1.19) from equation (1.20) yield

$$U(x_0+h) - U(x_0-h) = 2hU_x(x_0) + O(h^3)$$

Solving for $U(x_0)$ gives,

$$\begin{aligned} U_x(x_0) &= \frac{U(x_0+h) - U(x_0-h)}{2h} + \frac{O(h^3)}{h} \\ &= \frac{U(x_0+h) - U(x_0-h)}{2h} + O(h^2) \end{aligned}$$

Neglecting the $O(h^2)$ term gives,

$$U_x(x_0) \approx \frac{U(x_0+h) - U(x_0-h)}{2h} \quad (1.16)$$

This is the central difference approximation to $U_x(x_0)$. To get good approximations to the continuous problem small h is chosen. When $0 < h \ll 1$; the truncation error for the central difference approximation goes to zero much faster than the truncation error in equation (1.11) or equation (1.13). There is a complication with equation (1.16) because it does not include the value for $U(x_0)$. This may cause problems when the central difference approximation is included in an approximation to a differential equation.

Second Order Central Difference

Finite difference approximations to higher order derivatives can be obtained with the additional manipulations of the Taylor Series expansion about $U(x_0)$ Adding equations (1.19) and (1.20) yields:

$$U(x_0+h) + U(x_0-h) = 2U(x_0) + h^2U_{xx}(x_0) + 2h^4\frac{U_{(4)}(x_0)}{4!} + ... \quad (1.17)$$

Solving for $U_{xx}(x_0)$ gives,

$$\begin{aligned} U_{xx}(x_0) &= \frac{U(x_0+h) - 2U(x_0) + U(x_0-h)}{h^2} + h^2\frac{U_{(4)}(x_0)}{12} + ... \quad (1.18) \\ &= \frac{U(x_0+h) - 2U(x_0) + U(x_0-h)}{h^2} + O(h^2) \quad (1.19) \end{aligned}$$

This is also called the central difference approximation, but (obviously) it is the approximation to the second derivative, whereas equation (1.16) is the central difference approximation to the first derivative.

1.4 Properties of the Finite Difference Equation

The most important properties of the finite difference equation are consistency, stability and convergence. These notions cover different aspects of the relation between the partial differential equation and finite difference equation, and the exact and numerical solutions of the partial differential equation. The two fundamental sources of error are the truncation error in the stock price discretization and in the time discretization.

Consistency: A finite difference of a partial differential equation is consistent, if the difference between partial differential equation and finite differential equation vanishes as the space and time step size approach zero. Consistency deals with how well the finite difference equation approximates the partial differential equation and it is the necessary condition for convergence.

Stability: For a stable numerical scheme, the errors from any source will not grow unboundedly with time.

Convergence: It means that the solution to a finite difference equation approaches the true solution to the partial differential equation as both grid interval and time step sizes are reduced. The necessary and sufficient conditions for convergence are consistency and stability.

These three factors that characterize a numerical scheme are linked together by Lax equivalence theorem (Frankel and Du Fort, 1953) which states that given a well-posed linear initial value problem and a consistent finite difference scheme, stability is the necessary and sufficient condition for convergence.

In general, a problem is said to be well-posed if:

- A solution to the problem exists.
- The solution is unique when it exists.
- The solution depends continuously on the problem data.

Chapter 2

Explicit Finite Difference Scheme for Finite String

The finite difference approximations developed in the preceding section are now assembled into a discrete approximation to equation (1.1). Both the time and space derivatives are replaced by finite differences. Doing so requires specification both the time and spatial locations of the U values in the finite difference formulas. In the next section we will see that choosing the time step at which the spatial derivatives are evaluated will have a large impact on the performance and ease of implementation of the finite difference model.

2.1 Discretizing the Wave Equation

In order to effect a numerical approximation to the solution to this initial value problem, we begin by introducing a rectangular mesh consisting of points (t_i, x_j) with

$$0 = t_0 < t_1 < t_2 < ... \text{ and } 0 = x_0 < x_1 < ... < x_n = L$$

For simplicity, we maintain a uniform mesh spacing in both directions, with

$$\Delta t = t_{i+1} - t_i, \quad \Delta x = x_{j+1} - x_j = \tfrac{L}{n}$$

representing, the time step size and the spatial mesh size respectively. It will be essential that we do not a priori require the two to be the same. We shall use the notation

$$U_{i,j} = U(t_i, x_j) \qquad \text{where } t_i = i\Delta t \text{ and } x_j = j\Delta x$$

to denote the numerical approximation to the solution value at the indicated mesh point. Thus $U_{i,j}$ represents the value of U at the grid point (t_i, x_j).

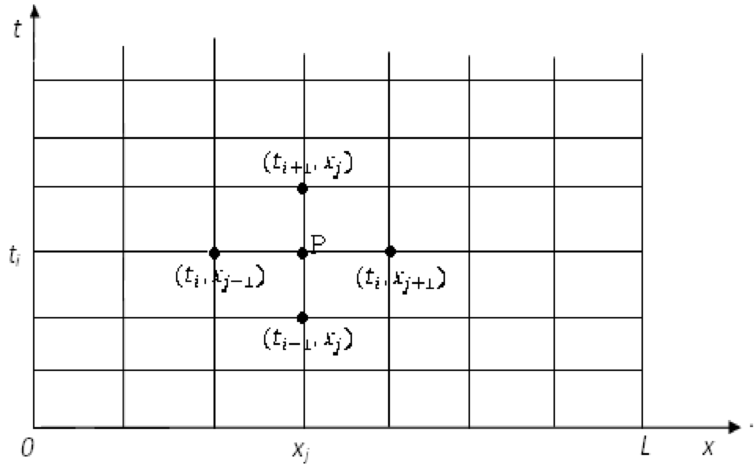

Figure1. Grid points in the xt-plane

Discretization is implemented by replacing the second order derivatives in the wave equation by their standard finite difference approximations (1.12)

$$U_{tt}(t_i, x_j) = \frac{U(t_{i+1}, x_j) - 2U(t_i, x_j) + U(t_{i-1}, x_j)}{(\Delta t)^2} + O((\Delta t)^2) \qquad (2.1)$$

$$U_{xx}(t_i, x_j) = \frac{U(t_i, x_{j+1}) - 2U(t_i, x_j) + U(t_i, x_{j-1})}{(\Delta x)^2} + O((\Delta x)^2) \qquad (2.2)$$

Since the errors are of orders of $(\Delta t)^2$ and $(\Delta x)^2$, it is expected to choose the space and time step sizes of comparable magnitude

$$\Delta t \approx \Delta x$$

Substituting the finite difference formula (2.1) and (2.2) into the partial differential equation (1.1) gives:

$$\frac{U_{i+1,j} - 2U_{i,j} + U_{i-1,j}}{(\Delta t)^2} - c^2 \frac{U_{i,j+1} - 2U_{i,j} + U_{i,j-1}}{(\Delta x)^2} = 0 \qquad (2.3)$$

And rearranging terms led to the iterative system

$$U_{i+1,j} = \sigma^2(U_{i,j+1} + U_{i,j-1}) + 2(1 - \sigma^2)U_{i,j} - U_{i-1,j}, \qquad (2.4)$$

or

$$U_{i+1,j} = 2U_{i,j} - U_{i-1,j} + \sigma^2(U_{i,j+1} - 2U_{i,j} + U_{i,j-1}) \tag{2.5}$$

i = 1,2,..., and j = 1,2,...,

$$where \quad \sigma = c\frac{\Delta t}{\Delta x} > 0 \tag{2.6}$$

And the explicit finite difference scheme (2.5) is expressed in the tridiagonal $N \times N$ matrix form

$$\mathbf{U}^{i+1} = K\mathbf{U}^i - \mathbf{U}^{i-1} + b. \tag{2.7}$$

Where

$$K = \begin{pmatrix} 2(1-\sigma^2) & \sigma^2 & & & & \\ \sigma^2 & 2(1-\sigma^2) & \sigma^2 & & & \\ & \sigma^2 & 2(1-\sigma^2) & \sigma^2 & & \\ & & \sigma^2 & \ddots & \ddots & \\ & & & \ddots & \ddots & \sigma^2 \\ & & & & \sigma^2 & 2(1-\sigma^2) \end{pmatrix}$$

$$b = \begin{pmatrix} \sigma^2 U_{i,0} \\ 0 \\ \vdots \\ 0 \\ \sigma^2 U_{i,n} \end{pmatrix} \text{ and } \mathbf{U}^i = \begin{pmatrix} U_{i,1} \\ U_{i,2} \\ \vdots \\ U_{i,n} \end{pmatrix}$$

This is the finite difference approximation to the wave equation. The value at the $(i+1)^{th}$ time step is computed from the values at the i^{th} and the $(i-1)^{th}$ time steps (see in figure 2).

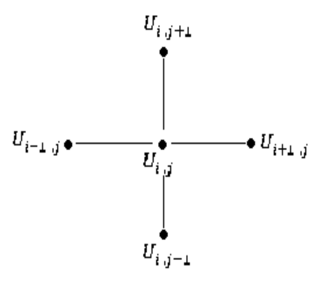

Figure 2. Stepping forward in time for the wave equation.

The main question is how to get the method started.

2.2 Discretizing the Boundary and Initial Conditions

Let's first deal with the boundary conditions. From (1.2) we get

$$U_{i,0} = U(i\Delta t, 0) = 0, \quad i > 0$$

and

$$U_{i,n} = U(i\Delta t, n\Delta x) = U(i\Delta t, L) = 0, \quad i > 0$$

And the initial condition yields

$$U(0, x) = f(x), \quad U_t(0, x) = g(x) \quad 0 < x < L$$

Since $U_{0,j} = f(x_j)$ is determined by the initial position, then we need to find $U_{1,j}$ with entries $U_{1,j} = U(\Delta t, x_j)$ at time $t_1 = \Delta t$ in order to launch the iteration and compute $U_{2,j}, U_{3,j}, U_{4,j}, ...$ but the initial velocity $U_t(0, x) = g(x)$ prescribes the derivatives $U_t(0, x_j) = g(x_j) = g_j$ at time $t_0 = 0$ instead. One way to resolve this difficulty would be to utilize the first order forward difference approximation.

$$U_t(t, x) = \tfrac{U(t+\Delta t, x) - U(t, x)}{\Delta t}$$

and

$$g_j = U_t(0, x_j) = \frac{U(\Delta t, x_j) - U(0, x_j)}{\Delta t} = \frac{U_{1,j} - f_j}{\Delta t} \tag{2.8}$$

To compute the required values

$$U_{1,j} = f_j + g_j \Delta t$$

However, the approximation (2.8) is only accurate to order Δt, whereas the rest of the scheme has errors proportional to $(\Delta t)^2$. The effect would be to introduce an unacceptably large error at the initial step, and the resulting solution would fail to conform to the desired order of accuracy.

$$\tfrac{U(\Delta t, x_j) - U(0, x_j)}{\Delta t} = U_t(0, x_j) + \tfrac{1}{2} U_{tt}(0, x_j)\Delta t + O((\Delta t)^2)$$

From equation (1.1) we have

$$U_{tt}(0, x_j) = c^2 U_{xx}(0, x_j),$$

then

$$\tfrac{U(\Delta t, x_j) - U(0, x_j)}{\Delta t} = U_t(0, x_j) + \tfrac{c^2}{2} U_{xx}(0, x_j)\Delta t + O((\Delta t)^2)$$

Since $U(t,x)$ solves the wave equation. Therefore,

$$\begin{aligned}
U_{1,j} &= U(\Delta t, x_j) \\
&= U(0,x_j) + U_t(0,x_j)\Delta t + \frac{c^2}{2}U_{xx}(0,x_j)(\Delta t)^2 \\
&= f(x_j) + g(x_j)\Delta t + \frac{c^2}{2}f''(x_j)(\Delta t)^2 \\
&\approx f(x_j) + g(x_j)\Delta t + \frac{c^2(f(x_{j+1}) - 2f(x_j) + f(x_{j-1}))(\Delta t)^2}{(2(\Delta x)^2)}
\end{aligned}$$

Where we can employ the finite difference approximations (3.1) and (3.2) to the second derivative of $f(x)$ if the explicit formula is either not known or too complicated to evaluate directly. Therefore, we initiate the scheme by setting

$$U_{1,j} = f(x_j) + g(x_j)\Delta t + \frac{\sigma^2}{2}[f(x_{j+1}) - 2f(x_j) + f(x_{j-1})] \qquad (2.9)$$

This is the second piece of initial data that we need in order to be able to start iterating formula (2.5).

Chapter 3

Consistency, Stability and Convergence

Three fundamental properties that every finite difference approximation of a partial differential equation should possess are consistency, stability and convergence. Consistency implies that the finite difference equation is a good approximation of the partial differential equation; stability implies that the solution of the difference equation is not too sensitive to small perturbations in the initial data and convergence implies that the solution of the difference equation approaches the solution of the partial differential equation as the computational mesh is refined. These properties are often difficult to verify for realistic problems, but they can be explained and illustrated quite easily using difference schemes for some simple model problems, such as for the wave equation (1.1).

3.1 Consistency

A numerical scheme is consistent if the discrete numerical equation tends to the exact differential equation as the mesh size (represented by Δx and Δt) tends to zero.

Definition 3.1.1. *Consider a differential equation $Pu = 0$ and a finite difference approximation of it $P_\Delta U_{i,j} = 0$. Let $v(t, x)$ be any smooth function, then the local discretization or local truncation error is*

$$\tau_{i,j} = Pv(i\Delta t, j\Delta x) - P_\Delta v(i\Delta t, j\Delta x) \tag{3.1}$$

Remark 3.1.1. *P is a differential operator, $Pv \equiv v_{tt} - c^2 v_{xx}$ is for the wave equation (1.1). Similarly, P_Δ is a finite difference operator; thus, the finite difference operator for the wave operator is*

$$P_\Delta v \equiv \frac{v_{i+1,j}-2v_{i,j}+v_{i-1,j}}{(\Delta t)^2} - c^2\frac{v_{i,j+1}-2v_{i,j}+v_{i,j-1}}{(\Delta x)^2}$$

Remark 3.1.2. *The function v is often regarded as the exact solution u of the partial differential equation. This is convenient but not necessary; thus, v may be any smooth function. This gives us a way of defining the local discretization error even when the solution of the partial differential equation has singularities.*

The local discritization error of the finite difference approximation for the wave equation is

$$\tau_{i,j} = (v_{tt} - c^2 v_{xx})_{i,j} - [\frac{v_{i+1,j}-2v_{i,j}+v_{i-1,j}}{(\Delta t)^2} - c^2\frac{v_{i,j+1}-2v_{i,j}+v_{i,j-1}}{(\Delta x)^2}]$$

Or

$$\tau_{i,j} = (v_{tt})_{i,j} - \frac{v_{i+1,j}-2v_{i,j}+v_{i-1,j}}{(\Delta t)^2} - c^2[(v_{xx})_{i,j} - \frac{v_{i,j+1}-2v_{i,j}+v_{i,j-1}}{(\Delta x)^2}] \quad (3.2)$$

Using the Taylor series

$$\tau_{i,j} = \frac{(\Delta t)^2}{12}(v_{tttt})_{i,j} + c^2\frac{(\Delta x)^2}{12}(v_{xxxx})_{i,j} \quad (3.3)$$

The local error is often used in place of the local discretization error.

Definition 3.1.2. *The local error is the difference between solutions of the partial differential equation and its finite difference approximation (i.e. $u_{i+1,j} - U_{i+1,j}$) assuming that no errors were committed prior to time level $i+1$.*

The finite difference approximation of the wave equation is

$$\frac{U_{i+1,j}-2U_{i,j}+U_{i-1,j}}{(\Delta t)^2} - c^2\frac{U_{i,j+1}-2U_{i,j}+U_{i,j-1}}{(\Delta x)^2} = 0$$

According to Definition (3.1) we commit no errors prior to time level $i+1$; thus, $U_{i,j} = u_{i,j}$ and

$$\frac{U_{i+1,j}-2u_{i,j}+u_{i-1,j}}{(\Delta t)^2} - c^2\frac{u_{i,j+1}-2u_{i,j}+u_{i,j-1}}{(\Delta x)^2} = 0$$

Using (3.2) with v replaced by u and the wave equation reveals

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

Combining the previous two relations, we obtain

$$u_{i+1,j} - U_{i+1,j} = -(\Delta t)^2\tau_{i,j} \quad (3.4)$$

Thus, the local error is the negative of the product of the local discretization error and the time step. This relationship between local and local discretization errors generally holds for explicit finite difference schemes.

Definition 3.1.3. *A finite difference scheme $P_\Delta U_{i,j} = 0$ is consistent with a partial differential equation $Pu = 0$ if the local discretization error tends to zero as $\Delta x \longrightarrow 0$ and $\Delta t \longrightarrow 0$.*

Equation (3.3) is the local discretization error for wave equation (1.1) i.e.

$$\tau_{i,j} = \frac{(\Delta t)^2}{12}(v_{tttt})_{i,j} + c^2\frac{(\Delta x)^2}{12}(v_{xxxx})_{i,j}$$

The left-hand side of this equation will tends to zero as Δt and Δx tend to zero. This means that the numerical scheme (2.3) tends to the exact equation at point x_j and time level t_i and therefore this approximation is consistent.

3.2 Stability

The solution u of the partial differential equation can be regarded as being an element of a function space. There is a positive scalar, called a norm, that is associated with many function spaces and which is used to measure the "size" of member functions and the "distances" between them.

Definition 3.2.1. *The norm of a function u denoted by $\|u\|$ is a scalar that satisfies:*

1. $\|u\| \geq 0$ *and* $\|u\| = 0$ *if and only if* $u = 0$,
2. $\|\alpha u\| = |\alpha|\|u\|$ *for any constant* α, *and*
3. $\|u + v\| \leq \|u\| + \|v\|$.

Condition 2 is called the condition of homogeneity and Condition 3 is the triangular inequality.

The two norms that are most suited to our purposes are the maximum norm

$$\|u\|_\infty \equiv max_{x\in\Omega}|u(t,x)| \tag{3.5}$$

And the Euclidean or L^2 norm

$$\|u\|_2 \equiv [\int_\Omega (u(t,x))^2 dx]^{\frac{1}{2}} \tag{3.6}$$

In both instances Ω is the spatial domain of the partial differential equation. Norms often give us useful estimates of the growth or decay of solutions with time.

While solutions u of the partial differential equation are regarded as elements of a function space, we may think of finite difference solutions as elements of a linear vector space. To this end, collect all of the unknowns at a given time level i into a vector U_i. The unknowns at the time level i for the finite difference scheme (2.2) for the wave problem (1.1) are $U_{i,1}, U_{i,2}, ..., U_{i,n-1}$; hence, define the vector

$$U_i = [U_{i,1}, U_{i,2}, ..., U_{i,n}]^T \tag{3.7}$$

Definition 3.2.2. *$\boldsymbol{V}$ is a linear vector space if*

1. *$U, V \in \boldsymbol{V}$ then $U + V \in \boldsymbol{V}$ and*
2. *α is a scalar constant and $U \in \boldsymbol{V}$ then $\alpha U \in \boldsymbol{V}$*

The properties of a norm on a vector space are identical to those for a function space.

Again, the two norms that most suit our purposes are the maximum norm

$$\|U_i\|_\infty \equiv \max_j \|U_{i,j}\|$$

And the Euclidean or L^2 norm

$$\|U_i\|_2 \equiv [\textstyle\sum_j (U_{i,j})^2]^{\frac{1}{2}}$$

The range on j is over all components of the vector U^n

Next we investigate the behavior of the discretization error as n increases, for fixed Δx and Δt.

Definition 3.2.3. *A finite difference scheme (2.5) for a homogeneous initial value problem is stable if there exists a positive constant C_T, such that*

$$\|U_i\| = C_T \|U_0\|,$$

Where $i \longrightarrow \infty, \quad \Delta x \longrightarrow 0, \quad \Delta t \longrightarrow 0, \quad i\Delta t \le T$ and the constant C_T may depend on T but not on the mesh spacing and initial data.

Remark 3.2.1. *The notion of stability expressed by Definition 3.4 implies a bound on the growth of the solution and, as such, is similar to the concept of a well-posedness partial differential equation.*

Here well-posedness of *initial value problem* means that:

$$\|u(t,x)\|_2 \le C_T \|u(0,x)\|_2, \quad t \in [0,T].$$

There are at least three ways in which the stability of a scheme can be tested. These are:

1. the direct method,
2. the energy method and
3. von Neumann's method.

A very powerful tool for testing the stability of linear partial difference equations with constant coefficients is von Neumann's method.

3.2.1 Von Neumann (or Fourier) Analysis of Stability

The stability of finite difference schemes for hyperbolic partial differential equations can be analyzed by the von Neumann or Fourier method. The idea behind the method is the following. The analytical solutions of the model wave equation $u_{tt} - c^2 u_{xx} = 0$ can be found in the form

$$u(x,t) = \sum_{m=-\infty}^{\infty} e^{\beta_m t} e^{Ik_m x}$$

If$\beta_m + c^2 k_m^2 = 0$. This solution involves a Fourier series in space and an exponential decay in time since $\beta_m \leq 0$ for $c^2 > 0$. Here we have included the complex version of the Fourier series, $e^{Ik_m x} = \cos k_m x + I \sin k_m x$ with $I = \sqrt{-1}$, because this simplifies considerably later algebraic manipulations. To analyze the growth of different Fourier modes as they evolve under the numerical scheme we can consider each frequency separately, namely

$$u(x,t) = e^{\beta_m t} e^{Ik_m x}$$

A discrete version of this equation is

$$\begin{aligned} u_{i,j} &= u(x_i, t_n) \\ &= e^{\beta_m t_i} e^{Ik_m x_j} \\ &= e^{\beta_m i \Delta t} e^{Ik_m j \Delta x} \\ &= (e^{\beta_m \Delta t})^i e^{Ik_m j \Delta x} \end{aligned}$$

The term $e^{Ik_m x} = \cos k_m x + I \sin k_m x$ is bounded and, therefore, any growth in the numerical solution will arise from the term $G = e^{\beta_m \Delta t}$, known as the amplification factor. Therefore the numerical method will be stable, or the numerical solution $u_{i,j}$ bounded as $n \longrightarrow \infty$, if $|G| < 1$ for solutions of the form

$$u_{i,j} = G^i e^{Ik_m j\Delta x}$$

Let us now analyze the stability of scheme (2.4) by plugging in the separated solution $u_{i,j} = G^i e^{Ik_m j\Delta x}$. The stability condition is $|G| < 1$, since otherwise the scheme would lead to exponentially growing solutions.

$$G^{i+1}e^{Ik_m j\Delta x} =$$
$$\sigma^2(G^i e^{Ik_m(j+1)\Delta x} + G^i e^{Ik_m(j-1)\Delta x}) + 2(1-\sigma^2)G^i e^{Ik_m j\Delta x} - G^{i-1}e^{Ik_m j\Delta x}$$

Dividing both sides of the above identity by $G^i e^{Ik_m j\Delta x}$ and collecting the remaining G terms on the left, we get

$$G + \frac{1}{G} = \sigma^2(e^{Ik_m\Delta x} + e^{-Ik_m\Delta x}) + 2(1-\sigma^2)$$
$$= 2 + 2\sigma^2(\cos k_m\Delta x - 1)$$

Denoting the right hand side by $p = 2 + 2\sigma^2(\cos k_m\Delta x - 1)$, we observe that $p \leq 2$ since $\cos k_m\Delta x - 1 \leq 0$. Then we have the equation

$$G + \frac{1}{G} = p$$
$$\Rightarrow G^2 - pG + 1 = 0$$

The roots of the last quadratic equation are

$$G_\pm = \tfrac{p\pm\sqrt{p^2-4}}{2}$$

If the discriminant is positive, i.e. $p^2 - 4 > 0$, which implies that $p < -2$, since p cannot be larger than 2, then the quadratic equation will have two distinct real roots. But one of these roots is

$$G_- = \tfrac{p-\sqrt{p^2-4}}{2} < \tfrac{p}{2} < \tfrac{-2}{2} = -1$$

which would lead to instability.

On the other hand, if the discriminant is non-positive, i.e. $p^2 - 4 \leq 0$, which implies $-2 \leq p \leq 2$, then the roots of the quadratic equation are complex,

$$G_\pm = \tfrac{p}{2} \pm I\tfrac{\sqrt{4-p^2}}{2}$$

With norm

$$|G_\pm| = \sqrt{\tfrac{p^2}{4} + \tfrac{4-p^2}{4}} = \sqrt{1} = 1$$

This is consistent with our intuition about solutions of the wave equation, since the time component of the separated solution will be

$$G^i = (\cos\theta + I\sin\theta)^i = \cos i\theta + I\sin i\theta$$

So the stability condition is equivalent to the requirement that the discriminant is non positive, $p^2 - 4 \leq 0$ or $-2 \leq p \leq 2$. Substituting the expressions of p in terms of σ^2, we have

$$-2 \leq 2 + 2\sigma^2(\cos k\Delta x - 1) \leq 2.$$

The inequality on the right is always true, while the worst case for the left inequality is when $\cos k\Delta x \approx -1$, which leads to the inequality $-2 \leq 2-4\sigma^2$, so the stability condition

$$\sigma^2 = c^2\frac{(\Delta t)^2}{(\Delta x)^2} \leq 1. \tag{3.8}$$

Notice that if we define the speed of the numerical scheme to be $\frac{\Delta x}{\Delta t}$, then the stability condition implies that

$$\text{speed of the scheme} > c,$$

with c being the speed of the exact solution. Thus, the necessary condition for stability of the numerical scheme is that the speed of the scheme must be at least as large as the speed of the exact equation.

3.3 Convergence

The most basic property that a scheme must have in order to be useful is that its solutions approximate the solution of the corresponding partial differential equation and that the approximation improves as the grid spacing, h and k, tend to zero. We call such a scheme a convergent scheme.

Definition 3.3.1. *A finite difference approximation converges to the solution of a partial differential equation on $0 \leq t \leq T$ in a particular vector norm if*

$$\|u_i - U_i\| \longrightarrow 0 \tag{3.9}$$

$$\textit{Where } i \longrightarrow \infty, \quad \Delta x \longrightarrow 0, \quad \Delta t \longrightarrow 0 \textit{ and } n\Delta t \leq T$$

Remark 3.3.1. *The vector u_i is the restriction of the continuous solution $u(x, t_i)$ to the mesh.*

Remark 3.3.2. *When applying Definition 3.3.1 visualize a sequence of computations performed on* $0 \leq t \leq T$ *using finer and finer meshes. Convergence implies that the discrete and continuous solutions approach each other for* $t \in (0, T]$ *in a particular vector norm as the mesh spacing decreases.*

This is the fundamental property to be sought from a numerical scheme but it is difficult to verify directly. On the other hand, consistency and stability are easily checked as shown in the previous sections. The main result that permits the assessment of the convergence of a scheme from the requirements of consistency and stability is the equivalence theorem of Lax.

Theorem 3.3.1. The Lax- Richtmyer Equivalence Theorem. *A consistent finite difference scheme for a partial differential equation for which the initial value problem is well posed is convergent if and only if it is stable.*

Here well-posedness of initial value problem means that:

$$\|u(t,x)\|_2 \leq c_T \|u(0,x)\|_2, t \in [0,T].$$

Proof. We will prove that a consistent and stable difference scheme is convergent. The proof that unstable schemes do not converge is not included here. A general formula for the evolution of the finite difference solution is the following:

$$U^{i+1} = AU^i + b^i \tag{3.10}$$

where A is the evolution matrix, and b is a vector containing forcing terms and the effects of boundary conditions. The vector U^{i+1} holds the vector of solution values at time $i+1$. The truncation error at a specific time level can be obtained by applying the above matrix operation to the vector of exact solution values:

$$u^{i+1} = Au^i + b^i + z^i \Delta t \tag{3.11}$$

where z is the vector of truncation error at time level i. Subtracting equation 3.11 from 3.10, we get an evolution equation for the error, namely:

$$e^{i+1} = Ae^i + z^i \Delta t. \tag{3.12}$$

where $e^{i+1} = u^{i+1} - U^{i+1}$ is the total error at time $t_{i+1} = (i+1)\Delta t$. Equation 3.12 shows that the error at time level $i+1$ is made up of two parts. The first one is the evolution of the error inherited from the previous time level, the first term on the right hand side of equation 3.12, and the second part is the truncation error committed at the present time level. Since, this expression applies to a generic time level, the same expression holds for e^i:

$$e^i = Ae^{i-1} + z^{i-1} \Delta t. \tag{3.13}$$

where we have assumed that the matrix A does not change with time to simplify the discussion (this is tantamount to assuming constant coefficients for the partial differential equation). By repeated application of this argument we get:

$$\begin{aligned} e^{i+1} &= A^2 e^{i-1} + (Az^{i-1} + z^i)\Delta t \\ &= A^3 e^{i-2} + (A^2 z^{i-2} + Az^{i-1} + z^i)\Delta t \\ &\vdots \\ &= A^{i+1} e^0 + (A^{i+1} z^0 + Aiz^1 + \cdots + Az^{i-1} + z^i)\Delta t \end{aligned}$$

This shows that the error growth depends on the truncation error at all time levels, and on the discretization through the matrix A. We can use the triangle inequality to get a bound on the norm of the error. Thus,

$$\|e^{i+1}\| \leq \|A^{i+1}\|\|e^0\| + (\|A^{i+1}\|\|z^0\| + \|Ai\|\|z^1\| + \cdots + \|A\|\|z^{i-1}\| + \|z^i\|)\Delta t \tag{3.14}$$

In order to make further progress we assume that the norm of the truncation error at any time is bounded by a constant ϵ such that

$$\epsilon = \max_{0 \leq m \leq i+1} (\|z^m\|) \tag{3.15}$$

The right hand side of inequality 3.14 can be bounded by

$$\|e^i\| \leq \|A^i\|\|e^0\| + (\Sigma_{m=0}^{i} \|A^m\|$$

The initial errors and the subsequent truncation errors are thus modulated by the evolution matrices A^m. In order to prevent the unbounded growth of the error norm as $i \to \infty$, we need to put a limit on the norm of the these matrices. This is in effect the stability property needed for convergence:

$$\|A^m\| \leq C = \max_{0 \leq m \leq i} (\|A^m\|)$$

where C is a constant independent of i, Δt and Δx. The sum in bracket can be bounded by the factor iC; the final expression becomes:

$$\|e^i\| \leq C(\|e^0\| + t_i \epsilon)$$

where $t_i = i\Delta t$ is the final integration time. When $\Delta x \to 0$, the initial error $\|e^i\|$ can be made as small as desired. Furthermore, by consistency, the truncation error $\epsilon \to 0$ when $\Delta t, \Delta x \to 0$. The global error is hence guaranteed to go to zero as the computational grid is refined, and the scheme is convergent. □

Remark 3.3.3. *The Lax Equivalence Theorem expresses a relationship between consistency, stability and convergency.*

Remark 3.3.4. *In practice convergence is difficult to establish while consistency and stability are less so. Consistency merely requires the use of Taylor's series expansions and stability can be proven, at least, in simplified situations using von Neumann's approach. Thus, Lax's theorem provides very useful information.*

Since scheme (2.5) is consistent and stable, then by Theorem 3.1 (The Lax-Richtmyer Equivalence Theorem). it is convergent.

Chapter 4

Implementation

Matlab version of the explicit finite difference scheme of wave equation of finite string is presented and demonstrated in this section.

4.1 Test Problem

Consider the initial and boundary problem of (1.1)

$$\begin{cases} U_{tt} = c^2 U_{xx} \quad 0 < x < L,\ t > 0 \\ U(t,0) = U(t,L) = 0 \\ U(0,x) = f(x), \quad U_t(0,x) = g(x) \quad 0 < x < L. \end{cases} \tag{4.1}$$

Using method of separation of variables the exact analytical solution of the above wave equation is:

$$u(t,x) = \sum_{n=1}^{\infty} \sin \frac{n\pi}{L} x (b_n \cos \lambda_n t + b_n^* \sin \lambda_n t) \tag{4.2}$$

where

$$b_n = \frac{2}{L} \int_0^L f(x) \sin \frac{n\pi}{L} x dx$$
$$b_n^* = \frac{2}{cn\pi} \int_0^L g(x) \sin \frac{n\pi}{L} x dx \quad and$$
$$\lambda_n = c\frac{n\pi}{L}, \quad n = 1, 2, \ldots$$

Use the explicit scheme (2.4) to simulate the following example.

Example 4.1.1. *Test the result of the explicit finite difference scheme (2.4) using the following data:*

$$c = L = 1, f(x) = \sin\frac{2\pi x}{L} \ \ and \ g(x) = 0$$

Using the above data and by (4.2) the exact solution of the wave equation is:

$$u(x,t) = \sin 2\pi x \cos 2\pi t \tag{4.3}$$

4.2 Implementation

The explicit finite difference scheme (2.4) is implemented in the Matlab function *wave* in Listing 1.

```
function [xo,to,Uo] = wave (nt,nx,alpha,L,tmax)
% wave Solve 1D wave equation with the finite difference scheme scheme
%
% Synopsis: wave
% wave(nt)
% wave(nt,nx)
% wave(nt,nx,alpha)
% wave(nt,nx,alpha,L)
% wave(nt,nx,alpha,L,tmax)
% Input: nt = number of steps.
% nx = number of mesh points in x direction.
% alpha = speed.
% L = length of the domain.
% tmax = maximum time for the simulation.
% Output: x = location of finite difference nodes
% t = values of time at which solution is obtained (time nodes)
% U = matrix of solutions: U(:,j) is U(x) at t = t(j)
if nargin<1, nt = 101; end
if nargin<2, nx =101; end
if nargin<3, alpha = 1; end
if nargin<4, L = 1; end
if nargin<5, tmax = 0.5; end
% --- Compute mesh spacing and time step
dx = L/(nx-1);
dt = tmax/(nt-1);
r = alpha^2*dt^2/dx^2; r2 = 2 - 2*r;
% --- Create arrays to save data for export
x = linspace(0,L,nx)';
t = linspace(0,tmax,nt);
U = zeros(nx,nt);
% --- Set IC and BC
U(:,1) = sin(2*pi*x/L);
U(:,2) = sin(2*pi*x/L) - 2*(pi/L)^2*dt^2*sin(2*pi*x/L); % implies u0 = 0; uL = 0;
u0 = 0; uL = 0; % needed to apply BC inside time step loop
% --- Loop over time steps
for m=3:nt
   for i=2:nx-1
U(i,m) = r2*U(i,m-1) + r*U(i+1,m-1) + r*U(i-1,m-1) - U(i,m-2);
   end
end
% --- Compare with exact solution at end of simulation
ue = sin(2*pi*x/L)*cos(2*alpha*pi*t/L);

plot(x,ue(:,15),'r--',x,U(:,15),'.','LineWidth',1);
 title('Wave on string','FontSize',14);
 xlabel('x','FontSize',14);
 ylabel('U(x,t)','FontSize',14);
 legend('exact','appro');
```

Listing 1: Code for Matlab implementation of the explicit finite difference scheme for solution to the wave equation.

Running *wave* with the above parameters gives

Case 1. For $r = \sigma^2 = c^2 \frac{(\Delta t)^2}{(\Delta x)^2} < 1$

Let $r = \sigma^2 = \frac{1}{4}$, then the exact and approximate solutions are shown below

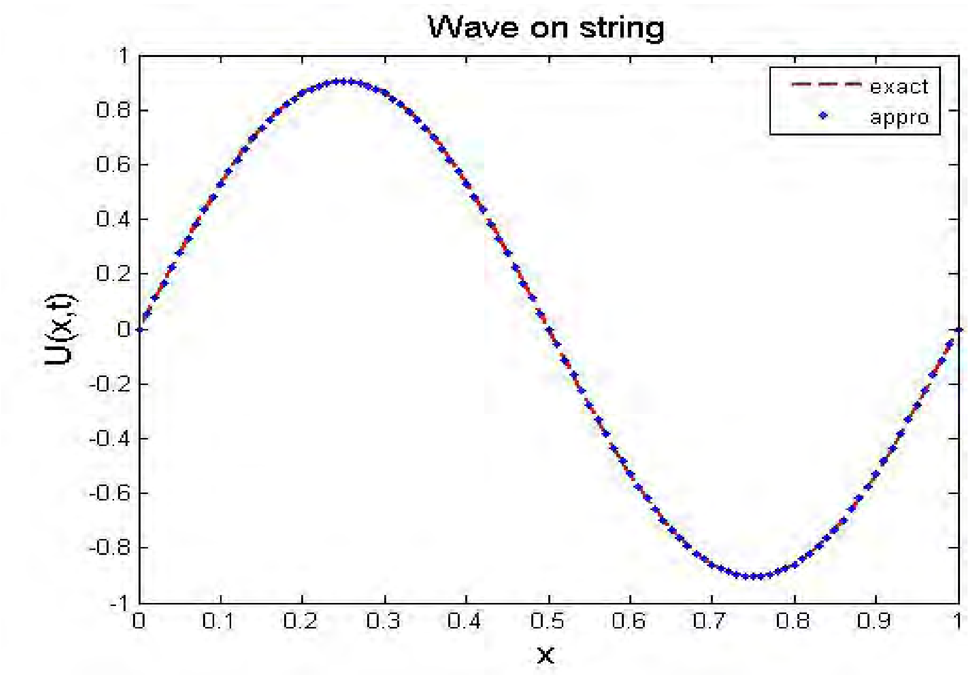

Figure 1: The explicit finite difference scheme solution to the wave equation at the 15^{th} time level obtained with $\sigma = \frac{1}{2}$. The solution is stable.

Case 2. For $r = \sigma^2 = c^2 \frac{(\Delta t)^2}{(\Delta x)^2} > 1$

Let $r = \sigma^2 = 4$, then the exact and approximate solutions are shown below

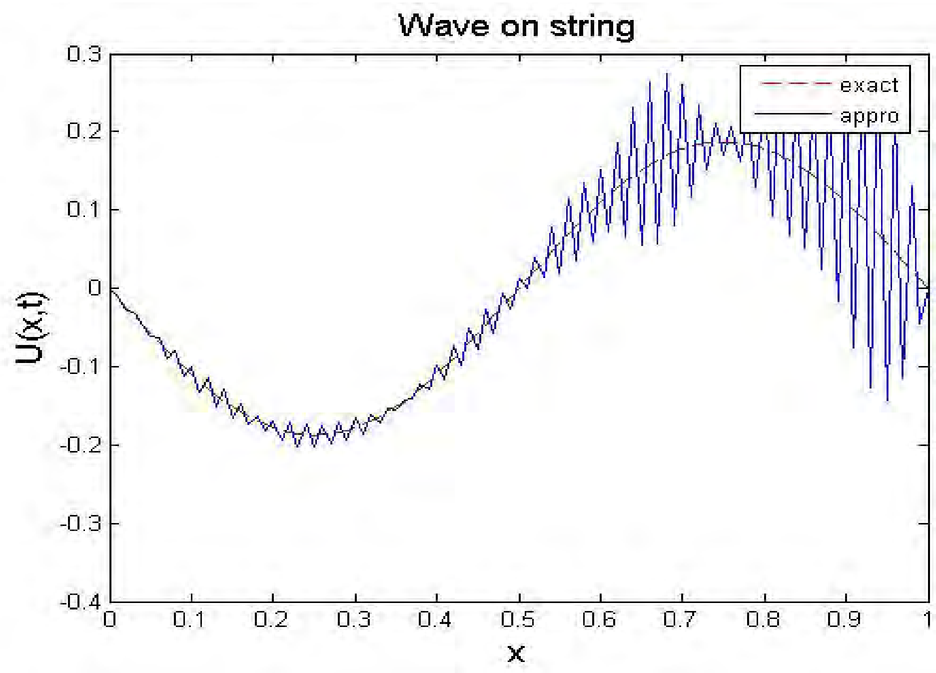

Figure 2: The explicit finite difference scheme solution to the wave equation at the 15^{th} obtained with $\sigma = 2$. The solution is unstable.

We can see from the Figures 1 and 2 that the numerical solution is stable when $\sigma^2 < 1$ and is unstable when $\sigma^2 > 1$. The explicit scheme (2.4) is conditional stable.

Summary

Using the centered differences for the second order time and space derivatives we derived a difference equation equivalent to the wave equation up to an error of order $O(\Delta x)^2$, which lead to an explicit numerical scheme. Analyzing the stability of this scheme we arrived at the condition $c\Delta t \leq \Delta x$, where c is the wave speed. This condition means that the wave speed of the numeric scheme must be at least as large as the wave speed of the exact equation. We also observed that the derived scheme uses two previous time steps to compute the values of the numerical solution at a particular grid point, thus one needs the values of two initial times steps to run the scheme. These we were able to find from the initial conditions by centered difference approximation, which does not add smaller order errors to the numerical scheme.

Bibliography

[1] G. D. Smith., *Numerical Solution of Partial Differential Equations: Finite Difference Methods Third edition.*, Oxford University Press; UK, 1985.

[2] J. Peiro and S. Sherwin, *Handbook of Materials Modeling. Volume I: Methods and Models,*, Springer,Netherlands, 2005.

[3] J.W. Thomas *Numerical Partial Differential Equations: Finite Difference Methods*, Springer, Verlag, Inc; New York, 1995.

[4] K. W. Morton/D. F. Mayers *Numerical Solution of Partial Differential Equations Second Edition*, Cambridge University Press, New York,2005.

[5] Nakhle H. Asmar *Partial Differential Equations with Fourier Serious and Boundary Value Problems Second Edition*, Pearson Education, Inc; New Jersey, 2005.

[6] Walter A. Strauss, *Partial Differential Equations Second Edition*, John Wiley and Sons, Inc.,New York, 2008.