THE TWO DIMENSIONAL COUPLED NONLINEAR SCHRÖDINGER EQUATION-NUMERICAL METHODS AND EXPERIMENTS

by

HARINI MEDIKONDURU

(Under the Direction of Thiab R. Taha)

ABSTRACT

The coupled nonlinear Schrödinger equation is of tremendous importance in both theory and applications. Coupled nonlinear Schrödinger equation (CNLS) is the vectorial version of the nonlinear Schrödinger equation (NLS). The NLS equation is the main governing equation in the area of optical solitons.

In this thesis, we perform numerical simulations of two dimensional CNLS equation using various numerical methods like the split-step Fourier methods and the finite difference methods. We implement parallel methods on the zcluster multiprocessor system. We then compare the numerical results time wise to see whether these methods have given a good speed up thus enhancing performance.

INDEX WORDS: Split-step method, CNLS, Parallelization, FFTW, Finite difference Method, MPI

THE TWO DIMENSIONAL COUPLED NONLINEAR SCHRÖDINGER EQUATION- NUMERICAL METHODS AND EXPERIMENTS

by

HARINI MEDIKONDURU

B.TECH, ACHARYA NAGARJUNA UNIVERSITY, INDIA, 2008

A Thesis submitted to the Graduate Faculty of The University of Georgia in Partial Fulfillment of the Requirements for the Degree

MASTER OF SCIENCE

ATHENS, GEORGIA

2012

THE TWO DIMENSIONAL COUPLED NONLINEAR SCHRÖDINGER EQUATION- NUMERICAL METHODS AND EXPERIMENTS

by

HARINI MEDIKONDURU

Major Professor: Thiab R. Taha

Committee: Hamid R. Arabnia
E. Rodney Canfield

Electronic Version Approved:

Maureen Grasso
Dean of the Graduate School
The University of Georgia
May 2012

DEDICATION

To the best parents in the whole world- Love you Amma and Nanna!

ACKNOWLEDGEMENTS

I would like to extend my many thanks to my major professor Dr. Thiab Taha for all the guidance and encouragement. I also would like to thank Dr. Hamid Arabnia and Dr. Rod Canfield for all the support. I am really indebted to all my friends who were like a second family to me and made Athens a home away from home. Last but the greatest, are my parents who have always been a source of inspiration and tremendous support, who believed in their little girl through and through when at times she herself did not.

TABLE OF CONTENTS

Page

LIST OF TABLES

Page

LIST OF FIGURES

Page

CHAPTER 1

INTRODUCTION

In the studies about optical fibres, the nonlinear Schrödinger typed equations are highly used and are of current importance [1]. Coupled nonlinear Schrödinger equation (CNLS) is the vectorial version of the nonlinear Schrödinger equation [15].

Here, we study the (2+1) dimensional system of CNLS equations, with x and y derivatives given by [1]

$$\begin{aligned} i\psi_t + \psi_{xx} + \psi_{yy} + \sigma\,(|\psi|^2 + \alpha\,|\phi|^2)\psi = 0 \\ i\phi_t + \phi_{xx} + \phi_{yy} + \sigma\,(|\phi|^2 + \alpha\,|\psi|^2)\phi = 0 \end{aligned} \qquad (1.1)$$

where ψ and ϕ are two dimensional complex functions.

There are many numerical methods to solve the CNLS equation like the split-step Fourier (SSF) method which was originally proposed by R.H. Hardin and F.D.Tappert [11]. There are different versions of the SSF method to solve the system of CNLS equations. Taha and Xu have developed SSF methods for CNLS as well as the NLS equations. They have also developed parallel implementations of these methods [14]. Ismail and Taha have also developed a finite difference approach to numerically simulate the coupled nonlinear Schrödinger equation [9].

This thesis includes implementations of split-step Fourier method in the first, second and fourth order schemes, over the two dimensional CNLS equation. Along with that, there are also implementations of finite difference method in two types- the explicit method and the implicit

method for the two dimensional CNLS equation. Finally, the parallel implementations for these five methods were developed. We used Fastest Fourier Transform in the West (FFTW) developed by M.Frigo and S.G. Johnson [7] for the serial and parallel implementations. Message Passing Interface (MPI) standard was used for the parallel implementations [8].

The thesis layout is as follows: The first chapter introduces the coupled nonlinear Schrödinger equation in two dimensions (2D CNLS) and touches upon the previous work done upon the CNLS. The second chapter briefly introduces the numerical methods we perform on the 2D CNLS. The third chapter contains the actual numerical methods with plots and results, which also contain the parallel implementation values. The fourth chapter provides with a summary and conclusion for this thesis.

CHAPTER 2

PRELIMINARIES

2.1 THE DISCRETE TWO DIMENSIONAL FOURIER TRANSFORM

If $\{f\ [m,n]\}$ is a sequence of size $M\ \times\ N$, obtained by taking samples of a continuous function f with equal intervals at the direction of m and n, respectively, then its Discrete Fourier Transform (DFT) is given by

$$F[k,l] = \frac{1}{\sqrt{MN}} \sum_{n=0}^{N-1} \sum_{m=0}^{M-1} f[m,n]\, e^{-i2\pi\left(\frac{mk}{M}+\frac{nl}{N}\right)}, 0 \le k < M, 0 \le l < N \tag{2.1.1}$$

where M and N are the numbers of samples in x and y directions in both spatial and frequency domains, respectively. And $F[k,l]$ is the two dimensional discrete spectrum of $f\ [m,n]$.

The inverse two dimensional DFT flips the sign of the exponent, which is defined as

$$f[m,n] = \frac{1}{\sqrt{MN}} \sum_{l=0}^{N-1} \sum_{k=0}^{M-1} F[k,l]\, e^{i2\pi\left(\frac{mk}{M}+\frac{nl}{N}\right)}, 0 \le m < M, 0 \le n < N \tag{2.1.2}$$

It is the "inverse" of the forward two dimensional DFT, in the sense that computing the inverse transfer after the forward transform of a given sequence would yield the original sequence.

Both $F[k,l]$ and $f\ [m,n]$ could be considered as elements of two $M\ \times\ N$ matrices f and F, respectively.

2.2 THE TWO DIMENSIONAL FOURIER TRANSFORM

The Fast Fourier transform (FFT) is an efficient algorithm to compute the discrete Fourier transform (DFT) and its inverse. A DFT decomposes a sequence of values into components of different frequencies. It is useful in many fields. However, calculating DFT directly from its definition is often too slow. Instead, FFT is a better way to calculate the same result more quickly.

Computing a DFT of N points requires $O(N^2)$. However, an FFT can calculate the same result in only $O(N \log_2 N)$ operations [17]. The difference in speed could be substantial, especially for large data sets where Nmay be very huge. In this case, FFTs are of great importance to a large number of applications, like, digital signal processing and solving partial differential equations.

A two dimensional FFT is achieved by first transforming each row, replacing each row with its one dimensional transform FFT and then transforming each column, replacing each column with its transform. A two dimensional FFT of size $M\ \times\ N$ requires $M\ +\ N$one dimensional FFT.

2.3 THE FASTEST FOURIER TRANSFORM IN THE WEST

The Fastest Fourier Transform in the West (FFTW) is a software library used to calculating DFTs, developed by M. Frigo and S. G. Johnson in MIT [7]. FFTW is a comprehensive collection of fast C routines for calculating the DFT in one or more dimensions,

of both real and complex data, and of arbitrary input size. “It has gained a wide acceptance in both academia and industry, because it provides excellent performance on a variety of machines (even competitive with or faster than equivalent libraries supplied by vendors)” [5].

FFTW automatically adapts the DFT algorithm to details of the underlying hardware (cache size, memory size, registers, etc.). The inner loop of FFTW is generated automatically by a special-purpose compiler [5]. The FFTW begins by generating codelets. A codelet is a fragment of C code that computes a Fourier transform of a fixed small size (e.g. 16 or 19). A composition of codelets is called a plan which depends on the size of the input and the underline hardware. At runtime, the FFTW’s planner finds the optimal decomposition for transforms of a specified size on your machine and produces a plan that contains this information. The resulting plan can be reused as many times as needed. This makes the FFTW’s relatively expensive initialization acceptable. FFTW also includes a shared-memory implementation on top of POSIX threads, and a distributed-memory implementation based on MPI (Message Passing Interface) [14].

2.4 THE SPLIT-STEP METHOD

The split-step Fourier method is a pseudo-spectral numerical method used to solve nonlinear partial differential equations. (Split-step method)

If we consider the following equations,

$$\psi_t = (L+N)\psi;$$
$$\psi(x,y,0) = \psi_0(x,y)$$
$$\phi_t = (L+N)\phi;$$
$$\phi(x,y,0) = \phi_0(x,y) \qquad (2.4.1)$$

in which L and N are linear and nonlinear operators, respectively. Generally, these operators are non commutative.

The two dimensional CNLS equation

$$i\psi_t + \psi_{xx} + \psi_{yy} + \sigma(|\psi|^2 + \alpha|\phi|^2)\psi = 0$$
$$i\phi_t + \phi_{xx} + \phi_{yy} + \sigma(|\phi|^2 + \alpha|\psi|^2)\phi = 0 \qquad (2.4.2)$$

can be rewritten for any real number k as,

$$\psi_t = L\psi + N\psi$$
$$\phi_t = L\phi + N\phi$$

where,

$$L\psi = i\psi_{xx} + i\psi_{yy}\,;\, N\psi = ik[\sigma(|\psi|^2 + \alpha|\phi|^2)]\psi$$
$$L\phi = i\phi_{xx} + i\phi_{yy}\,;\, N\phi = ik[\sigma(|\phi|^2 + \alpha|\psi|^2)]\phi$$

The solution of (1.1) can be advanced from one time-level to the next time-level by using the following formulae [14]

$$\psi(x,y,t+\Delta t) = \exp[\Delta t(L+N)]\,\psi(x,y,t)$$
$$\phi(x,y,t+\Delta t) = \exp[\Delta t(L+N)]\,\phi(x,y,t) \qquad (2.4.3)$$

where Δt denotes the time step. This method is first order approximate. It would be first order exact, if the operators L and N are time-independent [6].

The procedure for time splitting consists of replacing the right hand side of the above equations with an appropriate combination of products of the exponential operators $exp\ (\Delta tL)$ and $exp\ (\Delta tN)$. This produces a splitting error due to the non-commutability of L and N [18].

Thus the first, second and fourth order approximations of the splitting operator $exp\ [\Delta t(L + N)]$ are devised that reduce the splitting error.

The best scheme to minimize this error would be using Baker-Campbell-Hausdorf (BCH) formula [2] for the two operators A and B as follows

$$\exp(\lambda A)\exp(\lambda B) = \exp\left(\sum_{n=1}^{\infty} \lambda^n Z_n\right) \tag{2.4.4}$$

where λ is the coefficient of A and B, and

$$Z_1 = A + B$$

the remaining operators Z_n are commutators of A and B, commutators of commutators of A and B and so on. The expressions for Z_n are rather complicated:

$$Z_2 = \frac{1}{2}[A, B]$$

where $[A, B] = AB - BA$ is the commutator of A and B, and

$$Z_3 = \frac{1}{12}\left(\left[A, [A, B]\right] + \left[[A, B], B\right]\right)$$

From these results, we can obtain the first order approximation of the exponential operator in (2.4.3) as follows [14]

$$A_1(\Delta t) = \exp(\Delta t L)\exp(\Delta t N) \tag{2.4.5}$$

We usually first solve the nonlinear part

$$\psi_t = N\psi$$

$$\phi_t = N\phi$$

And using this solution, we advance to solving the linear part

$$\psi_t = L\psi$$

$$\phi_t = L\phi$$

This advancement is carried out in two steps, thus giving the procedure the name "split-step" method.

The second-order approximation of the exponential operator in (2.4.2) is given by

$$\mathrm{A}_2(\Delta \mathrm{t}) = \exp\left(\frac{1}{2}\Delta t N\right) \exp(\Delta t L) \exp\left(\frac{1}{2}\Delta t N\right) \tag{2.4.6}$$

The fourth-order approximation of the exponential operator in (2.4.3) which preserves the symmetry could also be constructed [10] [14],

$$\mathrm{A}_4(\Delta \mathrm{t}) = \mathrm{A}_2(\omega \Delta t)\, \mathrm{A}_2[(1-2\omega)\, \Delta t]\, \mathrm{A}_2(\omega \Delta t) \tag{2.4.7}$$

where

$$\omega = \frac{2 + \sqrt[3]{2} + \frac{1}{\sqrt[3]{2}}}{3} \tag{2.4.8}$$

The operators L and N in (2.4.4) – (2.4.6) may be interchanged without affecting the order of the method [3].

2.5 THE FINITE DIFFERENCE METHOD

Finite difference methods are numerical methods for approximating the solutions to differential equations using finite difference equations to approximate derivatives [4].

2.5.1 EXPLICIT METHOD

In explicit finite difference schemes, the value of a function at time $n + 1$ depends explicitly on its value at time n. This is also called forward difference method.

By using the explicit method, with central difference formula, the finite difference representation of (1.1) would be

$$i\left(\frac{\psi_{k,j}^{n+1}-\psi_{k,j}^{n}}{\Delta t}\right)+\frac{\psi_{k+1,j}^{n}-2\psi_{k,j}^{n}+\psi_{k-1,j}^{n}}{(\Delta x)^2}+\frac{\psi_{k,j+1}^{n}-2\psi_{k,j}^{n}+\psi_{k,j-1}^{n}}{(\Delta y)^2}$$
$$+\sigma\left(|\psi_{k,j}^{n}|^2+\alpha|\phi_{k,j}^{n}|^2\right)\psi_{k,j}^{n}=0$$
$$i\left(\frac{\phi_{k,j}^{n+1}-\phi_{k,j}^{n}}{\Delta t}\right)+\frac{\phi_{k+1,j}^{n}-2\phi_{k,j}^{n}+\phi_{k-1,j}^{n}}{(\Delta x)^2}+\frac{\phi_{k,j+1}^{n}-2\phi_{k,j}^{n}+\phi_{k,j-1}^{n}}{(\Delta y)^2}$$
$$+\sigma\left(|\phi_{k,j}^{n}|^2+\alpha|\psi_{k,j}^{n}|^2\right)\phi_{k,j}^{n}=0 \qquad (2.5.1)$$

where $0 \le k \le N_x$ and $0 \le j \le N_y$.

In order to obtain $\psi(x,y,t)$ and $\phi(x,y,t)$we start off by using the initial conditions $\psi(x,y,0)$and $\phi(x,y,0)$ in the above equations to compute $\psi(x,y,\Delta t)$ and $\phi(x,y,\Delta t)$. We thereby use $\psi(x,y,\Delta t)$ and $\phi(x,y,\Delta t)$ in order to compute the next step $\psi(x,y,2\Delta t)$ and $\phi(x,y,2\Delta t)$. This procedure is followed until we obtain the values of $\psi(x,y,t)$ and $\phi(x,y,t)$.

In other words, for a function *P* if we know P_{i+1}^{n}, P_{i}^{n} and P_{i-1}^{n}, we can compute P_{i+1}^{n+1} as it depends explicitly on the previous state of the function.

2.5.2 IMPLICIT METHOD

In the Implicit method, we find the solution to the two dimensional CNLS by solving an equation involving both the current and the later states of the system.

We use the alternating direction implicit (ADI) method [16] with central difference in time to solve our two dimensional CNLS. The main idea of the ADI method is to proceed in two stages, treating only one operator (one spatial discretization i.e. in this case either ψ_{xx} or ψ_{yy} and either ϕ_{xx} or ϕ_{yy}) implicitly at each stage. First a half step is taken in the direction of

ψ_{xx} implicitly and explicitly in the direction of ψ_{yy} . Next a half step is taken implicitly in the direction of ψ_{yy} and explicitly in the direction of ψ_{xx}. Similar procedure is done for ϕ_{xx} and ϕ_{yy} as well.

Now (2.4.2) will become

$$i\left(\frac{\psi_{k,j}^{n+1/2} - \psi_{k,j}^{n}}{\Delta t/2}\right) + \frac{\psi_{k+1,j}^{n+1/2} - 2\psi_{k,j}^{n+1/2} + \psi_{k-1,j}^{n+1/2}}{(\Delta x)^2} + \frac{\psi_{k,j+1}^{n} - 2\psi_{k,j}^{n} + \psi_{k,j-1}^{n}}{(\Delta y)^2}$$
$$+ \sigma\left(|\psi_{k,j}^{n}|^2 + \alpha|\phi_{k,j}^{n}|^2\right)\psi_{k,j}^{n} = 0$$

$$i\left(\frac{\phi_{k,j}^{n+1/2} - \phi_{k,j}^{n}}{\Delta t/2}\right) + \frac{\phi_{k+1,j}^{n+1/2} - 2\phi_{k,j}^{n+1/2} + \phi_{k-1,j}^{n+1/2}}{(\Delta x)^2} + \frac{\phi_{k,j+1}^{n} - 2\phi_{k,j}^{n} + \phi_{k,j-1}^{n}}{(\Delta y)^2}$$
$$+ \sigma\left(|\phi_{k,j}^{n}|^2 + \alpha|\psi_{k,j}^{n}|^2\right)\phi_{k,j}^{n} = 0 \qquad (2.5.2)$$

and

$$i\left(\frac{\psi_{k,j}^{n+1} - \psi_{k,j}^{n+1/2}}{\Delta t/2}\right) + \frac{\psi_{k+1,j}^{n+1/2} - 2\psi_{k,j}^{n+1/2} + \psi_{k-1,j}^{n+1/2}}{(\Delta x)^2} + \frac{\psi_{k,j+1}^{n+1} - 2\psi_{k,j}^{n+1} + \psi_{k,j-1}^{n+1}}{(\Delta y)^2}$$
$$+ \sigma\left(|\psi_{k,j}^{n+1/2}|^2 + \alpha|\phi_{k,j}^{n+1/2}|^2\right)\psi_{k,j}^{n+1/2} = 0$$

$$i\left(\frac{\phi_{k,j}^{n+1} - \phi_{k,j}^{n+1/2}}{\Delta t/2}\right) + \frac{\phi_{k+1,j}^{n+1/2} - 2\phi_{k,j}^{n+1/2} + \phi_{k-1,j}^{n+1/2}}{(\Delta x)^2} + \frac{\phi_{k,j+1}^{n+1} - 2\phi_{k,j}^{n+1} + \phi_{k,j-1}^{n+1}}{(\Delta y)^2}$$
$$+ \sigma\left(|\phi_{k,j}^{n}|^2 + \alpha|\psi_{k,j}^{n}|^2\right)\phi_{k,j}^{n+1/2} = 0 \qquad (2.5.3)$$

where $0 \le k \le N_x$ and $0 \le j \le N_y$.

To compute $\psi(x, y, t)$ and $\phi(x, y, t)$ we first use $\psi(x, y, 0)$ and $\phi(x, y, 0)$ as the initial conditions and use them to get $\psi(x, y, \Delta t/2)$ and $\phi(x, y, \Delta t/2)$.

After this, we use $\psi(x, y, \Delta t/2)$ and $\phi(x, y, \Delta t/2)$ in (2.5.3) to obtain $\psi(x, y, \Delta t)$ and $\phi(x, y, \Delta t)$.

2.6 MESSAGE PASSING INTERFACE

Message Passing Interface (MPI) is a library specification for message-passing, proposed as a standard by a broadly based committee of vendors, implementations, and users [8].

Message passing is a paradigm that has been widely used on certain classes of parallel machines, especially those with distributed memory. Processes running on such machines communicate through messages [14].

CHAPTER 3

TWO DIMENSIONAL COUPLED NONLINEAR SCHRÖDINGER EQUATION

The original two dimensional coupled nonlinear Schrödinger Equation (2D CNLS), is represented as follows:

$$\begin{aligned} i\psi_t + \psi_{xx} + \psi_{yy} + \sigma(|\psi|^2 + \alpha|\phi|^2)\psi = 0 \\ i\phi_t + \phi_{xx} + \phi_{yy} + \sigma(|\phi|^2 + \alpha|\psi|^2)\phi = 0 \end{aligned} \tag{3.0.1}$$

where ψ and ϕ are complex valued functions with respect to t, x, y which represent the amplitudes of two circularly polarized waves. $\sigma = \pm 1$ which is the parameter to distinguish between self focusing or self defocusing Kerr nonlinearity. The value of α varies over $\alpha \geq 2/3$ and $\alpha \leq 7$ [1]. We consider (3.0.1) with the Kerr-type electronic nonlinearity, when $\sigma = 1$ and $\alpha = 1$.

The exact one soliton solution of (3.0.l) is given by [1]

$$\begin{aligned} \psi &= \frac{\alpha_1}{2} e^{-\eta/2}\,\mathrm{Sech}\left[\mathrm{Re}(\theta_1) + \frac{\eta}{2}\right] e^{iIm(\theta_1)} \\ \phi &= \frac{\beta_1}{2} e^{-\eta/2}\,\mathrm{Sech}\left[\mathrm{Re}(\theta_1) + \frac{\eta}{2}\right] e^{iIm(\theta_1)} \end{aligned} \tag{3.0.2}$$

where

$$e^{\eta} = \frac{1}{2}\frac{|\alpha_1|^2 + |\beta_1|^2}{(k_1 + k_1^*)^2 + (l_1 + l_1^*)^2} \tag{3.0.3}$$

and $$\theta_1 = k_1 x + l_1 y + i(k_1^2 + l_1^2)t$$

where t is the time and$\alpha_1, \beta_1, k_1,$ and l_1 are arbitrary complex parameters whose values are $\alpha_1 = \beta_1 = 1$ and $k_1 = l_1 = 1 + i$. Also $k_1^* = l_1^* = 1 - i$ because they are the complex conjugates of k_1 and l_1 respectively.

3.1 NUMERICAL METHOD

We start off with studying the two dimensional CNLS equation (3.0.1) with the solution given in (3.0.2) and (3.0.3) with the values $\sigma = 1$, $\alpha = 1$, $\alpha_1 = \beta_1 = 1$, $k_1 = l_1 = 1 + i$ and $k_1^* = l_1^* = 1 - i$.

Here we assume that $\psi(x, y, t)$ and $\phi(x, y, t)$ to satisfy the boundary condition with period $[-P, P]$.

We normalize the spatial period to$[0, 2\pi]$, and we have

$$i\psi_t + (\psi_{xx} + \psi_{yy})\frac{\pi^2}{P^2} + \sigma(|\psi|^2 + \alpha|\phi|^2)\psi = 0$$

$$i\phi_t + (\phi_{xx} + \phi_{yy})\frac{\pi^2}{P^2} + \sigma(|\phi|^2 + \alpha|\psi|^2)\phi = 0 \qquad (3.1.1)$$

where $P = 10$, which is the half length of the period. $X = \pi(x + P)/P$ and $Y = \pi(y + P)/P$. Then, the interval $[0,2\pi]$ is divided into N_x sub intervals in the x-direction and N_y sub intervals in the y- direction respectively where $N_x = N_y$ with grid spacing $\Delta X = 2\pi/N_x$ and $\Delta Y = 2\pi/N_y$. It is also denoted as $X_j = j\Delta X$ where $j = 0,1, \dots, N_x$ and $Y_k = k\Delta Y$ where $k = 0,1, \dots, N_y$.

We now advance the solutions of (3.0.1) from time t to the next time-level $t + \Delta t$ in the following manner:

Firstly, we focus on the nonlinear part of the solution [13]:

$$i\psi_t = -\sigma(|\psi|^2 + \alpha|\phi|^2)\psi$$
$$i\phi_t = -\sigma(|\phi|^2 + \alpha|\psi|^2)\phi \qquad (3.1.2)$$

which can be solved exactly with

$$\hat{\psi}(X_j, Y_k, t+\Delta t) = \exp\left\{i\begin{pmatrix}\sigma|\psi(X_j,Y_k,t)|^2 + \\ \sigma\alpha|\phi(X_j,Y_k,t)|^2\end{pmatrix}\Delta t\right\}\psi(X_j,Y_k,t)$$

$$\hat{\phi}(X_j, Y_k, t+\Delta t) = \exp\left\{i\begin{pmatrix}\sigma|\phi(X_j,Y_k,t)|^2 + \\ \sigma\alpha|\psi(X_j,Y_k,t)|^2\end{pmatrix}\Delta t\right\}\phi(X_j,Y_k,t) \qquad (3.1.3)$$

Second, we take the linear part:

$$i\psi_t = -(\psi_{xx} + \psi_{yy})\frac{\pi^2}{P^2}$$
$$i\phi_t = -(\phi_{xx} + \phi_{yy})\frac{\pi^2}{P^2} \qquad (3.1.4)$$

For the two-dimensional DFT (Discrete Fourier Transform), we have

$$\hat{\psi}_{mn} = F_{m,n}(\psi_{jk}) = \frac{1}{\sqrt{N_x}\sqrt{N_y}}\sum_{j=0}^{N_x-1}\sum_{k=0}^{N_y-1}\psi_{jk}e^{-i(mX_j+nY_k)}$$

$$\frac{-N_x}{2} \le m \le \frac{N_x}{2}-1; \frac{-N_y}{2} \le n \le \frac{N_y}{2}-1$$

$$\hat{\phi}_{mn} = F_{m,n}(\phi_{jk}) = \frac{1}{\sqrt{N_x}\sqrt{N_y}}\sum_{j=0}^{N_x-1}\sum_{k=0}^{N_y-1}\phi_{jk}e^{-i(mX_j+nY_k)}$$

$$\frac{-N_x}{2} \le m \le \frac{N_x}{2}-1; \frac{-N_y}{2} \le n \le \frac{N_y}{2}-1 \qquad (3.1.5)$$

The inverse DFT is given by

$$\psi_{jk} = F_{j,k}^{-1}(\hat{\psi}_{mn}) = \sum_{m=\frac{-N_x}{2}}^{\frac{N_x}{2}-1} \sum_{n=\frac{-N_y}{2}}^{\frac{N_y}{2}-1} \hat{\psi}_{mn} e^{i(mX_j + nY_k)}$$

$$j = 0,1,2,\dots,N_x - 1, k = 0,1,2,\dots,N_y - 1$$

$$\phi_{jk} = F_{j,k}^{-1}(\hat{\phi}_{mn}) = \sum_{m=\frac{-N_x}{2}}^{\frac{N_x}{2}-1} \sum_{n=\frac{-N_y}{2}}^{\frac{N_y}{2}-1} \hat{\phi}_{mn} e^{i(mX_j + nY_k)} \tag{3.1.6}$$

$$j = 0,1,2,\dots,N_x - 1, k = 0,1,2,\dots,N_y - 1$$

Combining (3.1.4), (3.1.5) and (3.1.6), we have

$$\psi(X_j, Y_k, t + \Delta t) = F^{-1}\left(F\left(\hat{\psi}(X_j, Y_k, t + \Delta t)\right)_{m,n}\right)$$

$$\phi(X_j, Y_k, t + \Delta t) = F^{-1}\left(F\left(\hat{\phi}(X_{j,} Y_{k,} t + \Delta t)\right)_{m,n}\right)$$

which is nothing but

$$\psi(X_j, Y_k, t + \Delta t) = F^{-1}\left(\exp\left[-i\frac{\pi^2}{P^2}(m^2 + n^2)\Delta t\right] F\left(\hat{\psi}(X_{m,} Y_{n,} t)\right)\right)$$

$$\phi(X_j, Y_k, t + \Delta t) = F^{-1}\left(\exp\left[-i\frac{\pi^2}{P^2}(m^2 + n^2)\Delta t\right] F\left(\hat{\phi}(X_{m,} Y_{n,} t)\right)\right) \tag{3.1.7}$$

The split-step Fourier method for the first order approximation (2.4.5) is given by (3.1.7), where Δt is the time step, F and F^{-1} are the forward and inverse discrete Fourier transforms respectively.

To advance in time from t to $t+\Delta t$ by the split-step Fourier method with the second order splitting approximation (2.4.6), we should take the following steps [13]:

(1) Apply (3.1.3) to advance the solution using the nonlinear part

$$\hat{\psi}\left(X_j, Y_k, t+\tfrac{1}{2}\Delta t\right) = \exp\left\{ i \begin{pmatrix} \sigma\left|\psi\left(X_j, Y_k, t\right)\right|^2 + \\ \sigma\alpha\left|\phi\left(X_j, Y_k, t\right)\right|^2 \end{pmatrix} \tfrac{1}{2}\Delta t \right\} \psi\left(X_j, Y_k, t\right)$$

$$\hat{\phi}\left(X_j, Y_k, t+\tfrac{1}{2}\Delta t\right) = \exp\left\{ i \begin{pmatrix} \sigma\left|\phi\left(X_j, Y_k, t\right)\right|^2 + \\ \sigma\alpha\left|\psi\left(X_j, Y_k, t\right)\right|^2 \end{pmatrix} \tfrac{1}{2}\Delta t \right\} \phi\left(X_j, Y_k, t\right)$$

(2) Apply (3.1.7) to advance the solution using the linear part

$$\tilde{\psi}\left(X_j, Y_k, t+\tfrac{1}{2}\Delta t\right) =$$

$$F^{-1}\left(\exp\left[-i\frac{\pi^2}{P^2}(m^2+n^2)\Delta t\right] F\left(\hat{\psi}\left(X_m, Y_n, t+\frac{1}{2}\Delta t\right)\right)\right)$$

$$\tilde{\phi}\left(X_j, Y_k, t+\tfrac{1}{2}\Delta t\right) =$$

$$F^{-1}\left(\exp\left[-i\frac{\pi^2}{P^2}(m^2+n^2)\Delta t\right] F\left(\hat{\phi}\left(X_m, Y_n, t+\frac{1}{2}\Delta t\right)\right)\right)$$

(3) Apply (3.1.3) to advance the solution using the nonlinear part

$$\psi\left(X_j, Y_k, t+\Delta t\right) =$$

$$\exp\left\{ i \begin{pmatrix} \sigma\left|\tilde{\psi}\left(X_j, Y_k, t+\tfrac{1}{2}\Delta t\right)\right|^2 + \\ \sigma\,\alpha\left|\tilde{\phi}\left(X_j, Y_k, t+\tfrac{1}{2}\Delta t\right)\right|^2 \end{pmatrix} \tfrac{1}{2}\Delta t \right\} \tilde{\psi}\left(X_j, Y_k, t+\tfrac{1}{2}\Delta t\right)$$

$$\phi\left(X_j, Y_k, t+\Delta t\right) =$$

$$\exp\left\{ i \begin{pmatrix} \sigma\left|\tilde{\phi}\left(X_j, Y_k, t+\tfrac{1}{2}\Delta t\right)\right|^2 + \\ \sigma\,\alpha\left|\tilde{\psi}\left(X_j, Y_k, t+\tfrac{1}{2}\Delta t\right)\right|^2 \end{pmatrix} \tfrac{1}{2}\Delta t \right\} \tilde{\phi}\left(X_j, Y_k, t+\tfrac{1}{2}\Delta t\right)$$

Advancement in time from t to $t + \Delta t$ by the split-step Fourier method with the fourth order splitting approximation (2.4.7) could be obtained with following steps [14]:

First, advance in time from t to $t + \omega\Delta t$ using the second-order split-step Fourier method, where

$$\omega = \frac{2 + \sqrt[3]{2} + \frac{1}{\sqrt[3]{2}}}{3}$$

Second, advance in time from $t + \omega\Delta t$ to $t + (1 - \omega)\Delta t$ using the second-order split-step Fourier method.

Finally, advance in time from $t + (1 - \omega)\Delta t$ to $t + \Delta t$ using the second-order split-step Fourier method and we obtain approximations to $\psi(x, y, t + \Delta t)$ and $\phi(x, y, t + \Delta t)$.

3.2 NUMERICAL EXPERIMENTS

In order for us to test the numerical methods that were implemented, we compute the L_∞ and L_2 norms at the terminating time $T = 0.1$. L_∞ or the infinity norm is the maximum deviation between the numerical and exact solutions. L_2 or the Euclidean norm is the most common norm, calculated by summing the squares of all the differences between numerical and analytical solution and taking the square root.

Shown below are the tables which bear the values of the norms along with the conditions used. We have used $N = 256$ for different values of Δt time steps to calculate the convergence rates in time and $\Delta t = 0.0000625$ or $\Delta t = 0.000125$ for different values of N to calculate the convergence rates in space for the 2D CNLSE, throughout this thesis.

Here we only show the results for the complex function ψ as the same results reflect for the function ϕ. This is due to the fact that ψ and ϕ are two similar functions involved in the 2D

CNLS equations with different amplitudes which are also same if the defined parameters' values are considered. The absolute value of the numerical solution of ψ i.e. $|\psi|$ is plotted for every method implemented in this thesis.

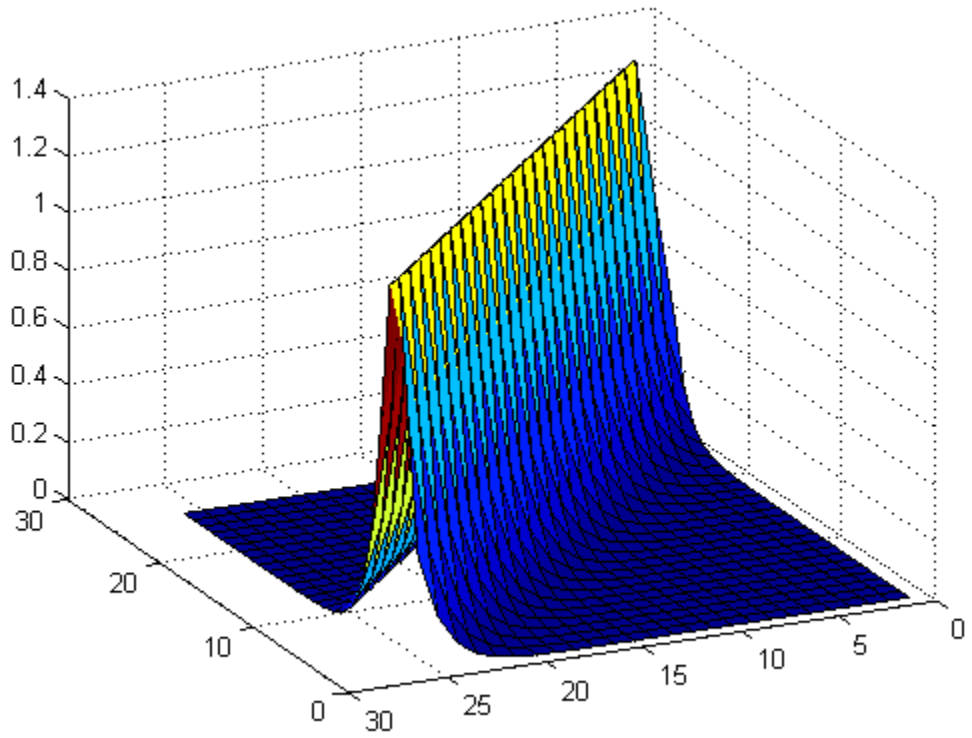

Figure 3.1: The exact solution

Table 3.1: Convergence rates in space for the first order SSF method

($\Delta t = 0.0000625, -10 \leq x \leq 10, -10 \leq y \leq 10, 0 < t < 1, T = 0.1$)

N	L_∞	L_2	cpu(s)
32	2.578502429E-01	6.116488661E-01	11.00
64	2.794540009E-01	6.159568066E-01	54.00
128	2.798998695E-01	6.178194602E-01	237.00

256	2.812919086E-01	6.187103923E-01	1383.00
512	2.954437911E-01	6.193919210E-01	2940.22

Table 3.2: Convergence rates in time for the first order SSF method

($N = 256, -10 \leq x \leq 10, -10 \leq y \leq 10, 0 < t < 1, T = 0.1$)

Δt	L_∞	L_2	cpu(s)
0.0000625	2.812919086E-01	6.187103923E-01	1383.00
0.000125	2.812920366E-01	6.187103799E-01	652.00
0.00025	2.812922928E-01	6.187103551E-01	332.00
0.0005	2.812928053E-01	6.187103055E-01	167.00
0.001	2.812938308E-01	6.187102064E-01	84.00

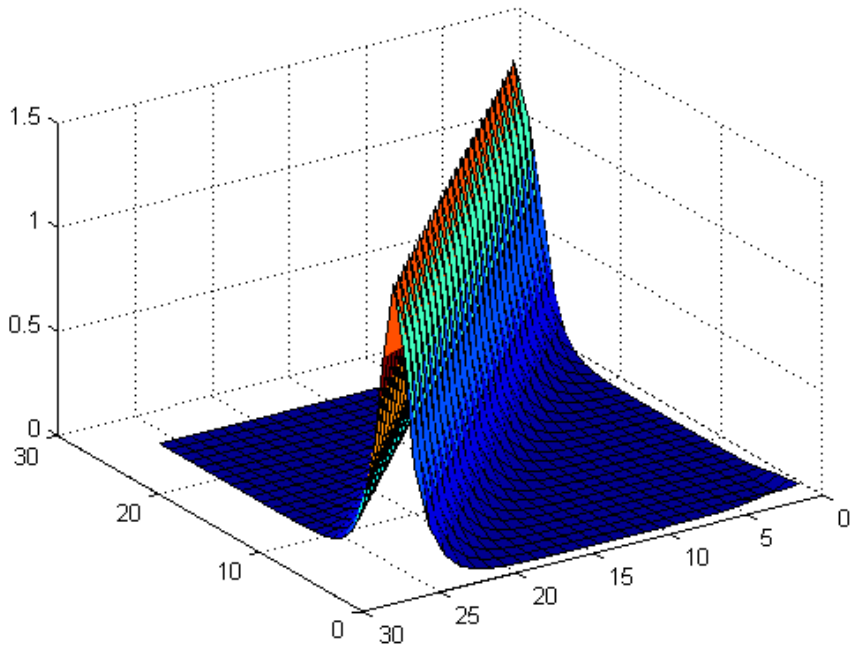

Figure 3.2: Two dimensional CNLSE using SSF in first order.

Table 3.3: Convergence rates in space for second order SSF method

($\Delta t = 0.0000625, -10 \leq x \leq 10, -10 \leq y \leq 10, 0 < t < 1, T = 0.1$)

N	L_∞	L_2	cpu(s)
32	2.5785029E-01	6.1164887E-01	16.09
64	2.7945395E-01	6.1595681E-01	73.38
128	2.7989978E-01	6.1781942E-01	313.89
256	2.8129179E-01	6.1871023E-01	1531.45
512	2.9544379E-01	6.1939192E-01	3930.20

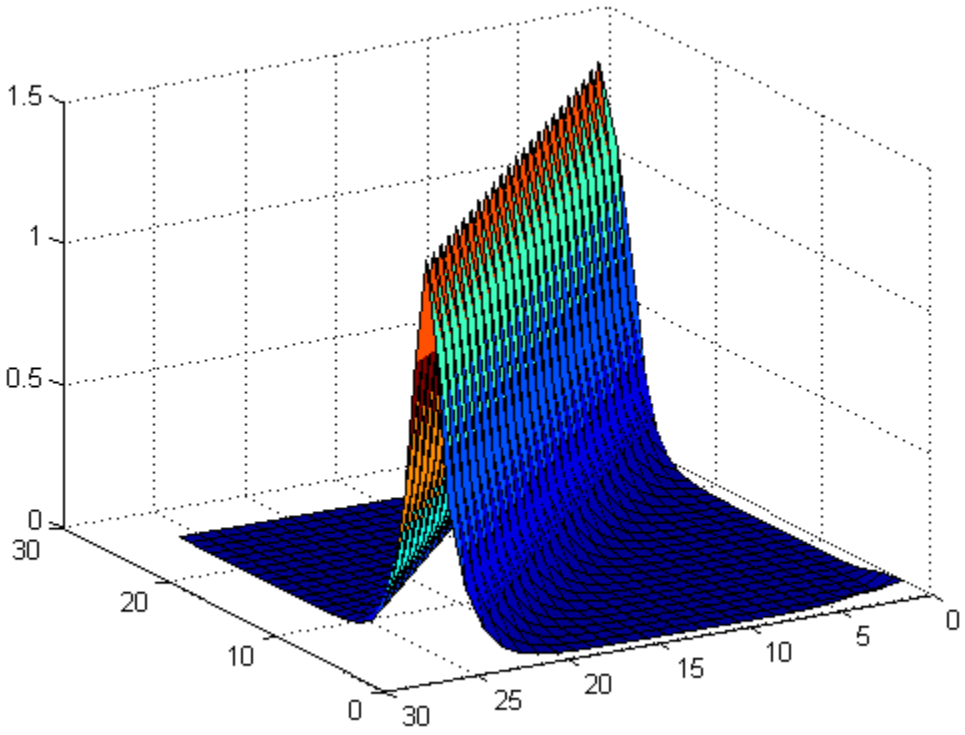

Figure 3.3: Two dimensional CNLSE using SSF in second order.

Table 3.4: Convergence rates in space for fourth order SSF method

($\Delta t = 0.0000625, -10 \leq x \leq 10, -10 \leq y \leq 10, 0 < t < 1, T = 0.1$)

N	L_∞	L_2	cpu(s)
32	2.5792688E-03	6.1166786E-03	42.45
64	2.7937306E-03	6.1597640E-03	177.89
128	2.7977111E-03	6.1783926E-03	384.23
256	2.8148341E-03	6.1873047E-03	3974.33
512	2.9547988E-03	6.1941251E-03	10285.42

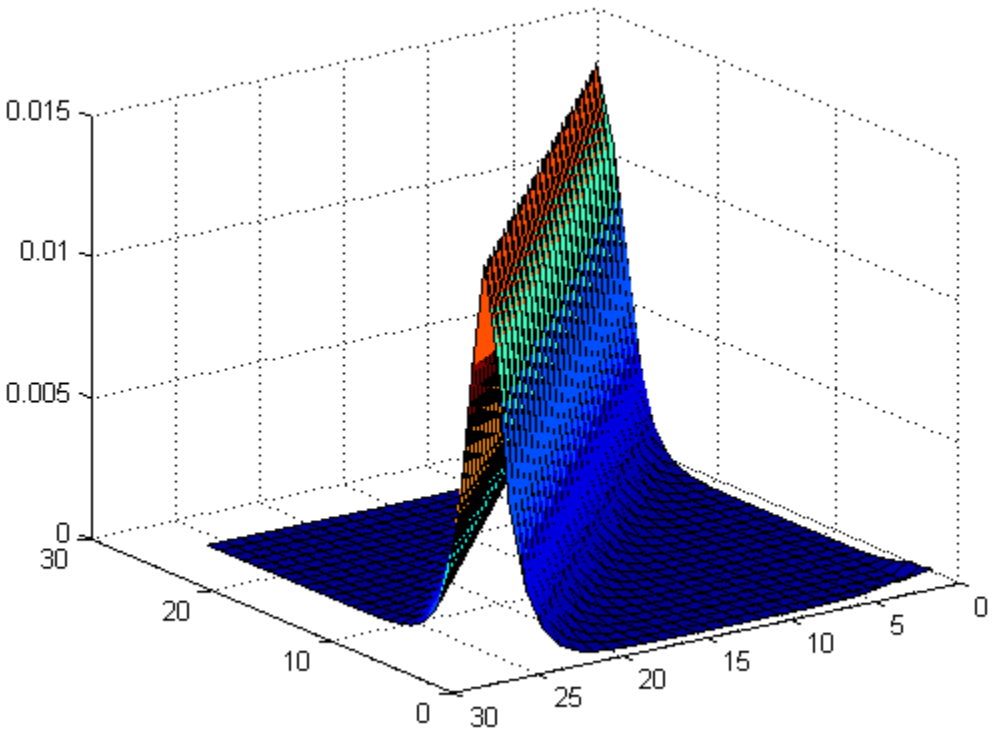

Figure 3.4: Two dimensional CNLSE using SSF in fourth order.

3.3 FINITE DIFFERENCE METHODS AND EXPERIMENTS

We implement the finite difference methods [4] i.e. the Explicit Finite Difference method and the Implicit Finite Difference method (Alternating Directions Implicit method) on the two dimensional CNLS from (3.0.1) as given below.

$$i\psi_t + \psi_{xx} + \psi_{yy} + \sigma(|\psi|^2 + \alpha|\phi|^2)\psi = 0$$

$$i\phi_t + \phi_{xx} + \phi_{yy} + \sigma(|\phi|^2 + \alpha|\psi|^2)\phi = 0$$

3.3.1 EXPLICIT METHOD

In explicit finite difference schemes, the value of a function at time $n+1$ depends explicitly on its value at time n. This is also called forward difference method.

Using the explicit forward difference method, at time t_{n+1} and a certain point(X, Y), where $X = k * x$ and $Y = j * y$, we have

$$\psi_{k,j}^{n+1} = \psi_{k,j}^{n} + i\left(\frac{\psi_{k+1,j}^{n} - 2\psi_{k,j}^{n} + \psi_{k-1,j}^{n}}{(\Delta x)^2} + \frac{\psi_{k,j+1}^{n} - 2\psi_{k,j}^{n} + \psi_{k,j-1}^{n}}{(\Delta y)^2} + \sigma\left(|\psi_{k,j}^{n}|^2 + \alpha|\phi_{k,j}^{n}|^2\right)\psi_{k,j}^{n}\right)\Delta t$$

$$\phi_{k,j}^{n+1} = \phi_{k,j}^{n} + i\left(\frac{\phi_{k+1,j}^{n} - 2\phi_{k,j}^{n} + \phi_{k-1,j}^{n}}{(\Delta x)^2} + \frac{\phi_{k,j+1}^{n} - 2\phi_{k,j}^{n} + \phi_{k,j-1}^{n}}{(\Delta y)^2} + \sigma\left(|\phi_{k,j}^{n}|^2 + \alpha|\psi_{k,j}^{n}|^2\right)\phi_{k,j}^{n}\right)\Delta t \quad (3.3.1)$$

where $0 \le k \le N_x$ and $0 \le j \le N_y$.

We devise a method to compute $\psi(x, y, t)$ and $\phi(x, y, t)$. At the very beginning, we use array to u_1 and u_2 to denote the initial conditions, where $u_1 = \psi(x, y, 0)$ and $u_2 = \phi(x, y, 0)$. Next, we use array u_3 to represent $\psi(x, y, t)$, at$t = \Delta t$ i.e. at the very first time step after $t = 0$. We use u_4 for $\phi(x, y, t)$, at $t = \Delta t$.

Finally, for $t = \Delta t$ to $t = numsteps \times \Delta t$, i.e. we start at Δt and increment t by Δt at every step. The same above procedure is followed simultaneously for obtaining $\phi(x, y, t)$which would be $u_4(j\Delta x, k\Delta y)$.

The following steps are performed. Here every pair of steps i.e. 1 & 2 and 3 & 4 are the same operations on the two functions ψ and ϕ respectively:

1. $u_3(j\Delta x, k\Delta y) = \frac{u_1\big((j+1)\Delta x,(k)\Delta y\big)-2u_1(j\Delta x,k\Delta y)+ u_1\,((j-1)\Delta x,k\Delta y)}{(\Delta x)^2}$
2. $u_4(j\Delta x, k\Delta y) = \frac{u_2\big((j+1)\Delta x,(k)\Delta y\big)-2u_2(j\Delta x,k\Delta y)+ u_2\,((j-1)\Delta x,k\Delta y)}{(\Delta x)^2}$
3. $u_3(j\Delta x, k\Delta y) = u_3(j\Delta x, k\Delta y) + \frac{u_1(j\Delta x,(k+1)\Delta y)-2u_1(j\Delta x,k\Delta y)+ u_1\,(j\Delta x,(k-1)\Delta y)}{(\Delta y)^2}$
4. $u_4(j\Delta x, k\Delta y) = u_4(j\Delta x, k\Delta y) + \frac{u_2(j\Delta x,(k+1)\Delta y)-2u_2(j\Delta x,k\Delta y)+ u_2\,(j\Delta x,(k-1)\Delta y)}{(\Delta y)^2}$
5. $u_3(j\Delta x, k\Delta y) = u_3(j\Delta x, k\Delta y) + \sigma[|u_1(j\Delta x, k\Delta y)|^2 + \alpha|u_2(j\Delta x, k\Delta y)|^2]u_1(j\Delta x, k\Delta y)$
6. $u_4(j\Delta x, k\Delta y) = u_4(j\Delta x, k\Delta y) + \sigma[|u_2(j\Delta x, k\Delta y)|^2 + \alpha|u_1(j\Delta x, k\Delta y)|^2]u_2(j\Delta x, k\Delta y)$
7. $u_3(j\Delta x, k\Delta y) = u_3(j\Delta x, k\Delta y) * \Delta t$
8. $u_4(j\Delta x, k\Delta y) = u_4(j\Delta x, k\Delta y) * \Delta t$
9. $u_3(j\Delta x, k\Delta y) = u_3(j\Delta x, k\Delta y) + u_1(j\Delta x, k\Delta y)$
10. $u_4(j\Delta x, k\Delta y) = u_4(j\Delta x, k\Delta y) + u_2(j\Delta x, k\Delta y)$

At the end of this procedure, we finally obtain $\psi(x, y, t)$ and $\phi(x, y, t)$ which are our numerical solutions.

Below are the values for the numerical experiments:

Table 3.5: Convergence rates in space for 2D CNLSE using explicit finite difference method.

$(\Delta t = 0.0000625, -10 \leq x \leq 10, -10 \leq y \leq 10, 0 < t < 1, T = 0.1)$

N	L_∞	L_2	cpu(s)
32	1.127249E-01	5.323813E-01	4.54
64	3.409962E-01	7.500453E-01	19.00
128	3.240042E-01	8.010477E-01	79.72
256	3.031719E-01	8.100455E-01	333.84

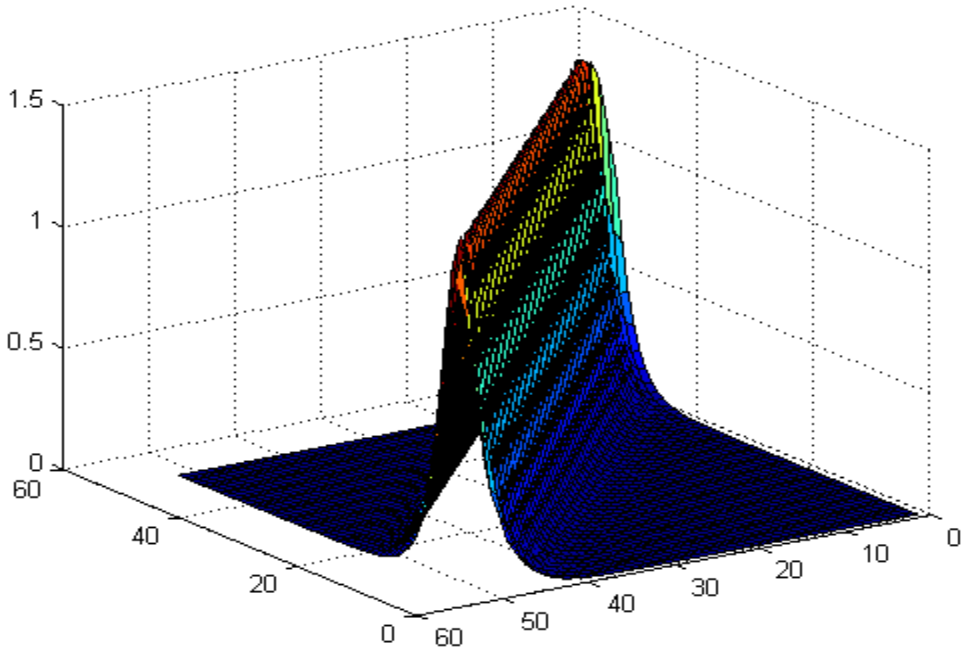

Figure 3.5: Two dimensional CNLSE using explicit finite difference method.

The explicit method performs better at $N = 512$ and Δt =0.00000008 with

L_∞ =5.969983E-02 and L_2 =2.475833E-02

3.3.2 IMPLICIT METHOD:

In the Implicit method, we find the solution to the two dimensional CNLS by solving an equation involving both the current and the later states of the system.

We use the alternating direction implicit (ADI) method [16] with central difference in time to solve our two dimensional CNLS. This ADI method is used because it is second order accurate in time.

Now (3.0.1) will become

$$\psi_{k,j}^{n+1/2} - \frac{i\Delta t}{2}\left(\frac{\psi_{k+1,j}^{n+1/2} - 2\psi_{k,j}^{n+1/2} + \psi_{k-1,j}^{n+1/2}}{(\Delta x)^2}\right) = \psi_{k,j}^{n} + \frac{i\Delta t}{2}\left(\frac{\psi_{k,j+1}^{n} - 2\psi_{k,j}^{n} + \psi_{k,j-1}^{n}}{(\Delta y)^2} + \sigma\left(|\psi_{k,j}^{n}|^2 + \alpha|\phi_{k,j}^{n}|^2\right)\psi_{k,j}^{n}\right)$$

$$\phi_{k,j}^{n+1/2} - \frac{i\Delta t}{2}\left(\frac{\phi_{k+1,j}^{n+1/2} - 2\phi_{k,j}^{n+1/2} + \phi_{k-1,j}^{n+1/2}}{(\Delta x)^2}\right) = \phi_{k,j}^{n} + \frac{i\Delta t}{2}\left(\frac{\phi_{k,j+1}^{n} - 2\phi_{k,j}^{n} + \phi_{k,j-1}^{n}}{(\Delta y)^2} + \sigma\left(|\phi_{k,j}^{n}|^2 + \alpha|\psi_{k,j}^{n}|^2\right)\phi_{k,j}^{n}\right) \tag{3.3.2}$$

and

$$\psi_{k,j}^{n+1} - \frac{i\Delta t}{2}\left(\frac{\psi_{k,j+1}^{n+1} - 2\psi_{k,j}^{n+1} + \psi_{k,j-1}^{n+1}}{(\Delta y)^2}\right) = \psi_{k,j}^{n+1/2} + \frac{i\Delta t}{2}\left(\frac{\psi_{k+1,j}^{n+1/2} - 2\psi_{k,j}^{n+1/2} + \psi_{k-1,j}^{n+1/2}}{(\Delta x)^2} + \sigma\left(|\psi_{k,j}^{n+1/2}|^2 + \alpha|\phi_{k,j}^{n+1/2}|^2\right)\psi_{k,j}^{n+1/2}\right)$$

$$\phi_{k,j}^{n+1} - \frac{i\Delta t}{2}\left(\frac{\phi_{k,j+1}^{n+1} - 2\phi_{k,j}^{n+1} + \phi_{k,j-1}^{n+1}}{(\Delta y)^2}\right)$$

$$= \phi_{k,j}^{n+1/2}$$

$$+ \frac{i\Delta t}{2}\left(\frac{\phi_{k+1,j}^{n+1/2} - 2\phi_{k,j}^{n+1/2} + \phi_{k-1,j}^{n+1/2}}{(\Delta x)^2}\right.$$

$$\left. + \sigma\left(|\phi_{k,j}^{n+1/2}|^2 + \alpha|\phi_{k,j}^{n+1/2}|^2\right)\psi_{k,j}^{n+1/2}\right) \tag{3.3.3}$$

where$0 \leq k \leq N_x$ and $0 \leq j \leq N_y$.

Proceeding (3.3.3) further, we have

$$\psi_{k,j}^{n+1} = \psi_{k,j}^{n+1/2} + i\left(\frac{\psi_{k+1,j}^{n+1/2} - 2\psi_{k,j}^{n+1/2} + \psi_{k-1,j}^{n+1/2}}{(\Delta x)^2} + \frac{\psi_{k,j+1}^{n+1} - 2\psi_{k,j}^{n+1} + \psi_{k,j-1}^{n+1}}{(\Delta y)^2}\right.$$

$$\left. + \sigma\left(|\psi_{k,j}^{n+1/2}|^2 + \alpha|\phi_{k,j}^{n+1/2}|^2\right)\psi_{k,j}^{n+1/2}\right)\Delta t/2$$

$$\phi_{k,j}^{n+1} = \phi_{k,j}^{n+1/2} + i\left(\frac{\phi_{k+1,j}^{n+1/2} - 2\phi_{k,j}^{n+1/2} + \phi_{k-1,j}^{n+1/2}}{(\Delta x)^2} + \frac{\phi_{k,j+1}^{n+1} - 2\phi_{k,j}^{n+1} + \phi_{k,j-1}^{n+1}}{(\Delta y)^2}\right.$$

$$\left. + \sigma\left(|\phi_{k,j}^{n+1/2}|^2 + \alpha|\psi_{k,j}^{n+1/2}|^2\right)\phi_{k,j}^{n+1/2}\right)\Delta t/2 \tag{3.3.4}$$

where$0 \leq k \leq N_x$ and $0 \leq j \leq N_y$.

The ADI method follows a pattern such that the above system of equations is reduced into a tridiagonal matrix and is then solved in two steps to obtain $\psi(x, y, t)$ and $\phi(x, y, t)$.

From (3.3.4) , we have

$$
\begin{aligned}
&(i+2r)\psi_{k,j}^{n+1} - r\psi_{k+1,j}^{n+1} - r\psi_{k-1,j}^{n+1} \\
&\qquad = (i-2r)\psi_{k,j}^{n+1/2} + r\psi_{k,j+1}^{n+1/2} + r\psi_{k,j-1}^{n+1/2} \\
&\qquad -\frac{\Delta t}{2}\sigma\left(|\psi_{k,j}^{n+1/2}|^2 + \alpha|\phi_{k,j}^{n+1/2}|^2\right)\psi_{k,j}^{n} \\
&(i+2r)\phi_{k,j}^{n+1} - r\phi_{k+1,j}^{n+1} - r\phi_{k-1,j}^{n+1} \\
&\qquad = (i-2r)\phi_{k,j}^{n+1/2} + r\phi_{k,j+1}^{n+1/2} + r\phi_{k,j-1}^{n+1/2} \\
&\qquad -\frac{\Delta t}{2}\sigma\left(|\phi_{k,j}^{n+1/2}|^2 + \alpha|\psi_{k,j}^{n+1/2}|^2\right)\phi_{k,j}^{n}
\end{aligned}
\tag{3.3.5}
$$

Now for $0 \leq k \leq N_x$ and $0 \leq j \leq N_y$, we put the coefficients of the left side of (3.3.5) in to the matrix A.

This leads to a tridiagonal system of matrices represented by:

$$
A = \begin{bmatrix}
I+2r & -r & 0 & 0 & 0 & 0 & 0 \\
-r & I+2r & -r & 0 & 0 & 0 & 0 \\
0 & -r & I+2r & 0 & 0 & 0 & 0 \\
. & . & . & . & . & . & . \\
. & . & . & . & . & . & . \\
. & . & . & . & . & . & . \\
. & . & . & . & . & . & . \\
. & . & . & . & . & . & . \\
0 & 0 & 0 & -r & I+2r & -r & 0 \\
0 & 0 & 0 & 0 & -r & I+2r & -r \\
0 & 0 & 0 & 0 & 0 & -r & I+2r
\end{bmatrix}
$$

where $r = \Delta t/(2 * (\Delta x)^2)$

To compute $\psi(x, y, t)$ and $\phi(x, y, t)$ we first use $\psi(x, y, 0)$ and $\phi(x, y, 0)$ as the initial conditions and use them in (3.3.2) and (3.3.3) to get $\psi(x, y, \Delta t/2)$ and $\phi(x, y, \Delta t/2)$.

After this, we use $\psi(x, y, \Delta t/2)$ and $\phi(x, y, \Delta t/2)$ in (3.3.4) to obtain $\psi(x, y, \Delta t)$ and $\phi(x, y, \Delta t)$.

First step of the ADI method is implemented in a loop over the y-direction:

$for\ j = \ 1{:}\ N_y$

$for\ k = 1{:}\ N_x$

$g_1(k) = (I - 2r)\psi_{k,j} + r\psi_{k,j+1} + r\psi_{k,j-1} - \Delta t/2\ \sigma\left(|\psi_{k,j}|^2\ + \alpha|\phi_{k,j}|^2\right)\psi_{k,j}$

$g_2(k) = (I - 2r)\phi_{k,j} + r\phi_{k,j+1} + r\phi_{k,j-1} - \Delta t/2\sigma\left(|\phi_{k,j}|^2\ + \alpha|\psi_{k,j}|^2\right)\phi_{k,j}$

end

$solve\ A\psi^{new}(:,j) = \ g_1$

$solve\ A\phi^{new}(:,j) = \ g_2$

end

where,

ψ^{new} and ϕ^{new} are intermediate stages.

For the next step, we follow the sequence of methods in the following page in which A can be obtained in the same procedure as above and here $r\ = \ \Delta t/(2 * (\Delta y)^2)$.

Next, we implement the ADI method in a loop over the x-direction:

$for\ k = \ 1{:}\ N_x$

$for\ j = 1{:}\ N_y$

$g_1(j) = (I - 2r)\psi_{k,j} + r\psi_{k+1,j} + r\psi_{k-1,j} - \Delta t/2\ \sigma\left(|\psi_{k,j}|^2\ + \alpha|\phi_{k,j}|^2\right)\psi_{k,j}$

$g_2(j) = (I - 2r)\phi_{k,j} + r\phi_{k+1,j} + r\phi_{k-1,j} - \Delta t/2\sigma\left(|\phi_{k,j}|^2\ + \alpha|\psi_{k,j}|^2\right)\phi_{k,j}$

end

$solve\ A\psi^{new}(k,:) = \ g_1$

$solve\ A\phi^{new}(k,:) = \ g_2$

end

By the end of this second process, our final $\psi(x,y,t)$ and $\phi(x,y,t)$ are obtained

The directions for the first and second stages of the ADI method are flexibly interchangeable.

For the numerical experiments, we have followed the same pattern as we did throughout this thesis: $N = 256$ for convergence rates in time and $\Delta t = 0.000125$ for convergence rates in space, respectively.

Table 3.6: Convergence rates in time for 2D CNLSE using implicit finite difference method.

($N = 256, -20 \leq x \leq 20, -20 \leq y \leq 20, 0 < t < 1, T = 0.1$)

Δt	L_∞	L_2	cpu(s)
0.0000625	4.511875E-02	2.876414E+00	1360.32
0.000125	4.635814E-02	2.876781E+00	674.58
0.00025	4.883871E-02	2.877511E+00	341.39
0.0005	5.380334E-02	2.878956E+00	169.25
0.001	6.371754E-02	2.881789E+00	93.44

Table 3.7: Convergence rates in space for 2D CNLSE using implicit finite difference method.

($\Delta t = 0.000125, -20 \leq x \leq 20, -20 \leq y \leq 20, 0 < t < 1, T = 0.1$)

N	L_∞	L_2	cpu(s)
32	6.477549E-02	2.612447E+00	6.67
64	8.886952E-02	2.820218E+00	26.93
128	2.193703E-02	2.830029E+00	122.82
256	4.635814E-02	2.876781E+00	674.58

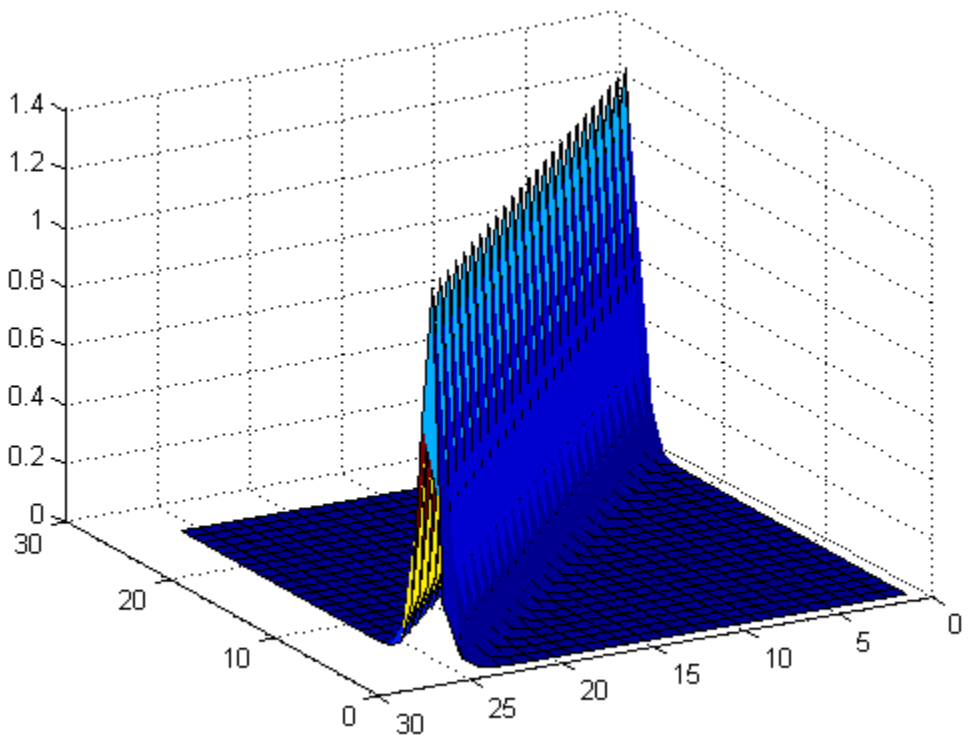

Figure 3.6: Two dimensional CNLSE using implicit finite difference method

3.4 PARALLEL IMPLEMENTATION AND EXPERIMENTS

Large-scale numerical simulations of the 2D CNLS are required for many problems in fiber optics. Such a simulation is computationally intensive and time consuming using sequential methods described earlier [14]. The parallelization methods for SSF as well as the finite difference methods are presented here in this section.

Each of the steps from (3.1.5) to (3.1.8) are parallelized for the first order SSF method. Similar procedures are followed for second order and fourth order SSF methods as well for those respective steps.

For the finite difference methods, the second order differentials ψ_{xx},ψ_{yy}, ϕ_{xx} and ϕ_{yy} are parallelized.

Let an array P of size $N_x \times N_y$ be the array that includes the approximate solution to ψ or ϕ at time t. We assume that there are p processors in a distributed memory parallel computer. We distribute this array P among p processors, where processor n has the limits $0 \leq n \leq p-1$ [14]. In other words, n denotes the id of the pth processor and it starts from 0 and goes up to $p-1$. Each process contains array elements $P[nN_x/p][N_y]$ to $P[(n+1)N_x/p]$ $[N_y]$.

All the parallel algorithms including the ones for the split-step Fourier methods are implemented on zcluster of UGA which is a linux cluster. Optimizations are done on the same level.

The speedup factor S_p is defined as

$$S_p = \frac{time\ spent\ on\ single\ processor\ (t_1)}{time\ spent\ on\ n\ processors\quad (t_n)}$$

As the number of physical processes (or the number of processors) available increases, the speedup S_p increases proportionally. In other words, as the problem size $N_x \times N_y$ increases, $S_p$obtained on the multiprocessor computer running the parallel codes in an MPI environment is considerably good given that $N_x \times N_y$ is large.

For all the numerical experiments, we varied N sizes at $\Delta t = 0.000125$.

Table 3.8: Parallel implementation results for first order SSF method

	$N = 128$	$N = 256$	$N = 512$
$t1$ (sec)	93.74	415.85	1201.23
$t2$ (sec)	60.77	209.16	764.67
$t4$ (sec)	36.21	119.30	465.23
$t8$ (sec)	29.40	88.87	278.54
$S_2 = t1/t2$	1.54	1.98	1.57
$S_4 = t1/t4$	2.58	3.48	2.58
$S_8 = t1/t8$	3.19	4.67	4.31

Table 3.9: Parallel implementation results for second order SSF method

	$N = 128$	$N = 256$	$N = 512$
$t1$ (sec)	103.63	426.25	1999.38
$t2$ (sec)	65.18	278.16	972.00
$t4$ (sec)	39.70	152.83	791.80
$t8$ (sec)	49.32	112.79	583.42
$S_2 = t1/t2$	1.59	1.53	2.05
$S_4 = t1/t4$	2.61	2.78	2.56
$S_8 = t1/t8$	2.08	3.78	3.43

Table 3.10: Parallel implementation results for fourth order SSF method

	$N = 128$	$N = 256$	$N = 512$
$t1$ (sec)	306.33	1247.14	4790.67
$t2$ (sec)	195.67	817.64	3881.87
$t4$ (sec)	123.34	466.72	1814.51
$t8$ (sec)	163.27	533.48	1157.55
$S_2 = t1/t2$	1.56	1.51	1.23
$S_4 = t1/t4$	2.48	2.69	2.64
$S_8 = t1/t8$	1.87	2.33	4.13

Table 3.11: Parallel implementation results for explicit method

	$N = 128$	$N = 256$	$N = 512$
$t1$ (sec)	26.78	104.92	419.98
$t2$ (sec)	12.96	53.63	218.90
$t4$ (sec)	10.17	26.57	133.62
$t8$ (sec)	6.53	19.93	122.47
$S_2 = t1/t2$	2.06	1.95	1.91
$S_4 = t1/t4$	2.63	3.94	3.14
$S_8 = t1/t8$	3.95	5.26	3.42

Table 3.12: Parallel implementation results for implicit method

	$N = 128$	$N = 256$	$N = 512$
$t1$ (sec)	85.37	478.53	2113.60
$t2$ (sec)	41.76	251.55	978.27
$t4$ (sec)	31.18	132.51	488.83
$t8$ (sec)	16.56	73.12	247.94
$S_2 = t1/t2$	2.04	1.90	2.16
$S_4 = t1/t4$	2.73	3.61	4.32
$S_8 = t1/t8$	5.15	6.54	8.52

CHAPTER 4

CONCLUSION AND FUTURE WORK

In this thesis, we have introduced the one soliton solution of the 2D CNLS developed by Hai-Qiang Zhang et al. [1] from the University of Beijing. We have described five different methods to solve the 2D CNLS. We have also parallelized the methods to compute their speed ups.

The five serial methods comprise of two finite difference methods namely implicit and explicit method and three split-step fourier methods in first, second and fourth orders. Also computed for them are the infinity and the Euclidean norms. Based on these values, a "ranking" was given to the accuracy of these methods.

It is as follows:

1. Two dimensional split-step fourier method in fourth order
2. Two dimensional implicit method
3. Two dimensional split-step fourier method in second order.
4. Two dimensional split-step fourier method in first order.
5. Two dimensional explicit method.

As per the time consumed, the higher order SSF methods take more time than their lower oder counterparts. The explicit method takes the least amount of time (computed in cpu seconds) followed by the implicit method.

The conclusion we have arrived upon using the parallel methods is that, for a given problem, the speed up increases as the size of the problem increases when running on a multiprocessor machine.

There is prospective future work in this area for aspirants. Hai-Qiang Zhang et al. [1] have also defined the two soliton solutions for a 2D CNLS on which these numerical methods can be applied and tested. Literature is also available for coupled nonlinear Schrödinger equation in 3D. The application of these numerical methods on such problems definitely provides scope for good and challenging future work.

REFERENCES

1. Hai-Qiang Zhang, X.-H. M.-L. (2007). *Interactions of bright solitions for the (2+1) dimensional coupled nonlinear Schrödinger equations from optical fibres with symbolic computation.*PHYSICA SCRIPTA, vol. 75, pp. 537-542.
2. J. M. Sanz -Serna, M. P. Calvo (1994). *Numerical Hamiltonian Problems.* Chapman & Hall, London.
3. G. M. Erbay (2003). *A split-step Fourier method for the complex modified Korteweg-de Vries equation.*Computers & Mathematics with Applications,vol 45, pp. 503-514.
4. Wikipedia. *Finite difference method*.
   Cited: Available at http://en.wikipedia.org/wiki/Finite_difference_method
5. M. Frigo (1999). A fast Fourier transform compiler, preprint.
6. J. A. C. Weideman, B. M. Herbst (1986). *Split-step methods for the solution of the nonlinear Schrödinger equation.*SIAM Journal on Numerical Analysis, vol. 23, Issue 3, pp. 485-507.
7. M. Frigo, S. G. Johnson (1997). *The Fastest Fourier Transform in the West.* Technical report, MIT- LCS - TR - 728, MIT Laboratory for Computer Science.
8. M. Snir, S. Otto, S. H. Lederman, D. Walker, and J. Dongarra (1996). *MPI: The Complete Reference.*MIT Press, London.
9. M.S. Ismail, T. R.Taha (2001). *Numerical simulation of coupled nonlinear Schrödinger equation,* Mathematics and Computers in Simulation, Special Issue on 'Optical Solitons'.

10. R. McLachlan (1994). *Symplectic integration of Hamiltonian wave equations.* Numerical Mathematics, vol. 66, pp. 465-492
11. R.H. Hardin, F. D. Tappert (1973). *Applications of the split-step Fourier method to the numerical solution of nonlinear and variable coefficient wave equations.*SIAM Review Chronicle, vol. 15, pp. 423.
12. Wikipedia. *Split-step method*.
    Cited: Available at http://en.wikipedia.org/wiki/Split-step_method
13. T. R. Taha, M. J. Ablowitz (1984). *Analytical and Numerical Aspects of Certain Nonlinear Evolution Equations. II. Numerical Nonlinear Schrödinger Equation.*Journal of Computational Physics, vol. 55,(No. 2), pp. 203-230.
14. THIAB R. TAHA, Xiangming Xu (2005). *Parallel Split-Step Fourier Methods for the Coupled Nonlinear Schrödinger Type Equations.*The Journal of Supercomputing, vol.32, pp.5-23.
15. Cited: Available at
    http://optilux.sourceforge.net/Documentation/optilux_doc/Coupled_NLSE_CNLSE.html
16. Cited: http://www.mth.pdx.edu/~daescu/mth410_510s/notes_week8.pdf
17. J. W. Cooley, J. W. Tukey (1965). *An algorithm for the machine computation of complex Fourier series*. Mathematics of Computation, vol. 19, pp. 297 - 301.
18. G.M. Muslu, H.A.Erbay (2005). *Higher-order split-step Fourier schemes for the generalized nonlinear Schrödinger equation,* Mathematics and Computers in Simulation, 581-595.