Parallel Numerical Methods for Solving Nonlinear Evolution Equations

Thiab R. Taha
Department of Computer Science
University of Georgia
Athens, GA 30602
email: thiab@cs.uga.edu

MASCOT 2010

Abstract

Nonlinear evolution equations are of tremendous interest in both theory and applications. In this talk we introduce parallel algorithms for numerical simulations of CMKdV, NLS and and CNLS equations in 1+1 and 1+2 dimensions. The parallel methods are implemented on multiprocessor system. Numerical experiments have shown that these methods give accurate results and considerable speedup.

This talk is organized as follows:

- Introduction to the NLS and CNLS equations
- The split-step method and Fourier transform.
- Numerical methods for the NLS and CNLS equations.
- Numerical methods for the CMKdV equation.
- Numerical methods for (1+2) NLS.
- Parallel Numerical methods for the KdV-Like equations.

Nonlinear Schrödinger Equations

The nonlinear Schrödinger(NLS) and the coupled nonlinear Schrödinger (CNLS) equations are of tremendous interest in both theory and applications. Various regimes of pulse propagation in optical fibers are modeled by some form of the NLS type equation. The CNLS equation is the governing equation for the propagation of two orthogonally polarized pulses in a monomode birefringent fibers.

In this presentation, different numerical methods will be presented for numerical simulations of the above equations. More emphasis will be on the design and implementation of parallel split-step Fourier methods for these equations. These parallel methods are implemented on the Origin 2000 multiprocessor computer. Our numerical experiments have shown that these methods give accurate results and considerable speedup.

INTRODUCTION

A wide class of physical phenomena (e.g., modulation of deep water waves, propagation of pulses in optical fibers, self-trapping of a light beam in a color-dispersive system) is described by the NLS equation

$$iu_t - u_{xx} + q|u|^2 u = 0, \qquad (1)$$

where u is a complex-valued function, and q is a real number.

Since large-scale simulations of the NLS equation are required for many physical problems, S. Zoldi et al implemented a parallel split-step Fourier method for the numerical simulation of the NLS equation.

In this talk we employ the well known split-step Fourier method for the numerical simulation of the NLS and CNLS equations. We also present a parallelization of the split-step Fourier method using the *Fastest Fourier Transform in the West* (FFTW) developed by M. Frigo and S. G. Johnson.

The split-step Fourier (SSF) method proposed by R. H. Hardin and F. D. Tappert is one of the most popular numerical methods for solving the NLS equation. Various versions of the split-step method have been developed to solve the NLS equation. G. M. Muslu and H. A. Erbay introduced three different split-step schemes for the numerical simulation of the complex modified Korteweg-de Vries (CMKdV) equation

$$w_t + w_{xxx} + \alpha(|w|^2 w)_x = 0, \qquad (2)$$

where w is a complex-valued function of the spatial coordinate x and time t, and α is a real parameter.

PRELIMINARIES

The split-step method

Consider a general evolution equation of the form

$$\begin{aligned} u_t &= (\mathcal{L} + \mathcal{N})\, u, \\ u(x,0) &= u_0(x), \end{aligned} \qquad (3)$$

where $\mathcal{L}$ and $\mathcal{N}$ are linear and nonlinear operators, respectively. In general, the operators $\mathcal{L}$ and $\mathcal{N}$ do not commute with each other.

For example, the NLS equation

$$u_t = -iu_{xx} + iq|u|^2 u,$$

with q a real number, can be rewritten as

$$u_t = \mathcal{L}\, u + \mathcal{N}\, u,$$

where

$$\mathcal{L}\, u = -iu_{xx},\ \mathcal{N}\, u = iq|u|^2 u.$$

The solution of equation (3) may be advanced from one time-level to the next by means of the following formula

$$u(x, t + \Delta t) \doteq \exp[\Delta t(\mathcal{L} + \mathcal{N})]\, u(x,t), \qquad (4)$$

where Δt denotes the time step. It is first order accurate. However, it turns out to be exact if operators $\mathcal{L}$ and $\mathcal{N}$ are time-independent. In fact, by Taylor's theorem we have

$$u(x, t+\Delta t) = u(x,t) + u_t(x,t)\Delta t + \frac{1}{2!}u_{tt}(x,t)(\Delta t)^2 + \cdots,$$

and

$$\exp[\Delta t(\mathcal{L}+\mathcal{N})]u = u + \Delta t(\mathcal{L}+\mathcal{N})u + \frac{1}{2!}(\Delta t)^2(\mathcal{L}+\mathcal{N})^2 u$$

$$+ \cdots .$$

Hence, equation (3) implies that (4) is first order accurate.

The time-splitting procedure now consists of replacing the right-hand side of (4) by an appropriate combination of products of the exponential operators $\exp(\Delta t\mathcal{L})$ and $\exp(\Delta t\mathcal{N})$. An answer can be found by considering the Baker-Campbell-Hausdorf (BCH) formula for two operators A and B given by

$$\exp(\lambda A)\exp(\lambda B) = \exp(\sum_{n=1}^{\infty} \lambda^n Z_n), \qquad (5)$$

where

$$Z_1 = A + B,$$

and the remaining operators Z_n are commutators of A and B, commutators of commutators of A and B, etc. The expression for Z_n are actually rather complicated, e.g.

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

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

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

From this result, one can easily get the first-order approximation of the exponential operator in (4) as follows

$$\mathcal{A}_1(\Delta t) = \exp(\Delta t \mathcal{L}) \exp(\Delta t \mathcal{N}). \tag{6}$$

Note that this expression is exact whenever $\mathcal{L}$ and $\mathcal{N}$ commute.

It is convenient to view the scheme (6) as first solving the nonlinear equation

$$u_t = \mathcal{N}\, u,$$

then advancing the solution by solving the linear equation

$$u_t = \mathcal{L}\, u,$$

employing the solution of the former as the initial condition of the latter. That is, the advancement in time is carried out in two steps, the so called split-step method.

The second-order approximation of the exponential operator in (4) is given by

$$\mathcal{A}_2(\Delta t) = \exp(\frac{1}{2}\Delta t\mathcal{N})\exp(\Delta t\mathcal{L})\exp(\frac{1}{2}\Delta t\mathcal{N}). \qquad (7)$$

It is symmetric in the sense that $\mathcal{A}_2(\Delta t)\mathcal{A}_2(-\Delta t) = 1$.

The fourth-order approximation of the exponential operator in (4) which preserves the symmetry can also be constructed, e.g.

$$\mathcal{A}_4(\Delta t) = \mathcal{A}_2(\omega\Delta t)\,\mathcal{A}_2[(1-2\omega)\Delta t]\,\mathcal{A}_2(\omega\Delta t), \qquad (8)$$

where

$$\omega = \frac{2+\sqrt[3]{2}+\frac{1}{\sqrt[3]{2}}}{3}. \qquad (9)$$

Note that the operators $\mathcal{L}$ and $\mathcal{N}$ in (6)–(8) may be interchanged without affecting the order of the method.

The Fourier transform

Fourier transforms are used to decompose a signal into its constituent frequencies. It is a powerful tool in linear system analysis.

The continuous Fourier transform

For every real number p, $1 \leq p < +\infty$, let $L^p(\mathbb{R})$ denote the set of all complex-valued Lebesgue measurable functions f such that

$$\int_{-\infty}^{+\infty} |f(x)|^p \, dx < +\infty. \tag{10}$$

The Fourier transform of a one-dimensional function $f \in L^1(\mathbb{R})$ is defined as

$$\mathcal{F}(f)(\xi) = \widehat{f}(\xi) = \int_{-\infty}^{+\infty} f(x) e^{-i2\pi x\xi} \, dx, \tag{11}$$

where $i = \sqrt{-1}$. It follows that $\widehat{f}$ is continuous and $\widehat{f}(\xi)$ approaches 0 as $|\xi|$ tends to $+\infty$.

The inverse Fourier transform of a function $g \in L^1(\mathbb{R})$ is defined as

$$\mathcal{F}^{-1}(g)(x) = \breve{g}(x) = \int_{-\infty}^{+\infty} g(x) e^{i2\pi\xi x} \, d\xi. \tag{12}$$

The only difference between the forward and inverse Fourier transforms is in the sign of the exponent.

If $f \in L^1(\mathbb{R})$, $\widehat{f} \in L^1(\mathbb{R})$, and f is continuous, then we have

$$f(x) = \int_{-\infty}^{+\infty} \widehat{f}(\xi) e^{i2\pi\xi x}\, d\xi, \tag{13}$$

for every $x \in \mathbb{R}$. That is, $f = \mathcal{F}^{-1}(\widehat{f})$.

If $f \in L^1(\mathbb{R}) \cap L^2(\mathbb{R})$, then $\widehat{f} \in L^2(\mathbb{R})$ and satisfies the Parseval identity

$$\int_{-\infty}^{+\infty} |\widehat{f}(\xi)|^2\, d\xi = \int_{-\infty}^{+\infty} |f(x)|^2\, dx. \tag{14}$$

The discrete Fourier transform

If $\{f_j\}$ is a sequence of length N, obtained by taking samples of a continuous function f at equal intervals, then its discrete Fourier transform (DFT) is the sequence $\{F_k\}$ given by

$$F_k = \frac{1}{\sqrt{N}} \sum_{j=0}^{N-1} f_j \, \omega_N^{-jk}, \; 0 \le k < N, \tag{15}$$

where $\omega_N = e^{i\frac{2\pi}{N}}$ is a primitive N-th root of unity.

The inverse DFT flips the sign of the exponent of ω_N, and it is defined as

$$f_j = \frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} F_k \, \omega_N^{jk}, \; 0 \le j < N. \tag{16}$$

It is the "inverse" of the forward DFT, in the sense that computing the inverse transform after the forward transform of a given sequence yields the original sequence.

After the required values of the complex exponential have been stored in a table, the number of arithmetic (multiplication or addition) operations required to implement DFT as in (15) is about $2N^2$, and hence it is of order N^2. So is the inverse DFT.

The Fast Fourier Transform

As mentioned above, the DFT requires $O(N^2)$ operations to compute and makes the computation potentially burdensome. Fortunately, there exists an algorithm called fast Fourier transform (FFT) that reduces the required number of arithmetic operations to $O(N \log_2(N))$. This requires that N can be factored into a product of small integers. The most common case is $N = 2^q$ for an integer q.

Suppose N can be factored as $N = p_1 p_2$, then the indices j and k in (15) can be represented as

$$j = j_1 p_2 + j_0;\ j_1 = 0, \ldots, p_1 - 1,\ j_0 = 0, \ldots, p_2 - 1,$$

and

$$k = k_1 p_1 + k_0;\ k_1 = 0, \ldots, p_2 - 1,\ k_0 = 0, \ldots, p_1 - 1.$$

Substitute into the expression (15), we obtain

$$\begin{aligned} F_k &= \frac{1}{\sqrt{N}} \sum_{j_0=0}^{p_2-1} \sum_{j_1=0}^{p_1-1} f_{j_1p_2+j_0} \omega_N^{-(j_1p_2+j_0)k} \\ &= \frac{1}{\sqrt{N}} \sum_{j_0=0}^{p_2-1} \Big(\sum_{j_1=0}^{p_1-1} f_{j_1p_2+j_0} \omega_N^{-j_1k_0p_2} \Big) \omega_N^{-j_0k}. \end{aligned}$$

Note that we have used the fact that $\omega_N^{-j_1kp_2} = \omega_N^{-j_1k_0p_2}$, since $\omega_N^N = 1$. It follows that

$$F_k = \frac{1}{\sqrt{N}} \sum_{j_0=0}^{p_2-1} \tilde{F}_{j_0,k_0} \omega_N^{-j_0k}, \qquad (17)$$

where

$$\tilde{F}_{j_0,k_0} = \sum_{j_1=0}^{p_1-1} f_{j_1p_2+j_0} \omega_N^{-j_1k_0p_2}. \qquad (18)$$

Observe that the number of arithmetic operations has indeed been reduced by this procedure. Each of the N elements in (18), $\tilde{F}_{j_0,k_0}$, requires $2p_1$ arithmetic operations, for a total of $2Np_1$ operations. Each F_k in (17) requires additional $2p_2$ operations. Thus the number of arithmetic operations to obtain all the F_k is $N(p_1 + p_2)$.

If p_1 and p_2 are factorable then the procedure can be repeated. In fact, if

$$N = p_1 p_2 p_3 \dots p_m,$$

then the entire process applied recursively in this manner requires

$$2N(p_1 + p_2 + \cdots + p_m)$$

operations. For $p_1 = p_2 = \cdots = p_m = p$,

$$2pN \log_p N$$

operations are needed. In particular, for $p = 2$, which is the most common case, a total of $4N \log_2 N$ arithmetic operations are required to compute the DFT.

The Fastest Fourier Transform in the West

The Fastest Fourier Transform in the West (FFTW) is a library developed by M. Frigo and S. G. Johnson in MIT. FFTW is a comprehensive collection of fast C routines for computing the discrete Fourier transform 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)."

FFTW automatically adapts the DFT algorithm to details of the underlying hardware (cache size, memory size, registers, etc.). The inner loop of FFTW are generated automatically by a special-purpose compiler. 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 produce a plan that contains this information. The resulting plan can be reused as many times as needed. Many transforms of the same size are computed in a typical high-performance applications. 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). The FFTW's MPI routines are significantly different from the ordinary FFTW because the transformed data are distributed over multiple processes, so that each processes gets only a portion of the transform data.

Message Passing Interface

The Message Passing Interface (MPI) is a library specification for message-passing, proposed as a standard by a broadly based committee of vendors, implementors, and users. MPI was designed for high performance on both massively parallel machines and on workstation clusters. Message passing is a paradigm used widely on certain classes of parallel machines, especially those with distributed memory. Processes running on such machines communicates through messages.

NONLINEAR SCHRÖDINGER EQUATION

Consider the following NLS equation

$$iu_t = u_{xx} + 2|u|^2 u, \tag{19}$$

where u is a complex-valued function. The exact one-soliton solution of (19) on the infinite interval is

$$u(x,t) = 2\eta \exp\{-i[2\xi x - 4(\xi^2 - \eta^2)t + \phi_0 + \frac{\pi}{2}]\} \operatorname{sech}(2\eta x - 8\xi\eta t - x_0), \tag{20}$$

where x_0, η, ξ, ϕ_0 are constants.

Numerical method

We study the NLS equation (19) with the initial condition given by

$$u(x,0) = 2\eta \exp\{-i[2x + \frac{\pi}{2}]\} \operatorname{sech}(2\eta x), \qquad (21)$$

where $\eta = 3$. We assume that $u(x,t)$ satisfies periodic boundary condition with period $[-10, 10]$.

If the spatial period is normalized to $[0, 2\pi]$, then equation (19) becomes

$$iu_t = \frac{\pi^2}{P^2} u_{XX} + 2|u|^2 u, \qquad (22)$$

where $P = 10$, the half length of the period, and $X = \pi(x+P)/P$. We divide the interval $[0, 2\pi]$ into N equal subintervals with grid spacing $\Delta X = 2\pi/N$, and denote $X_j = j\Delta X, j = 0, 1, \cdots, N$ as the spatial grid points.

The solution of (19) may be advanced from time t to the next time-level $t + \Delta t$ by the following two steps.

(1) Advance the solution using only the nonlinear part:

$$iu_t = 2|u|^2 u, \tag{23}$$

through

$$\tilde{u}(X_j, t + \Delta t) = \text{exp}\{-2i|u(X_j, t)|^2 \Delta t\}\, u(X_j, t). \tag{24}$$

(2) Advance the solution according to the linear part:

$$iu_t = \frac{\pi^2}{P^2} u_{XX}, \tag{25}$$

by means of computing

$$\widehat{u}(X_k, t + \Delta t) = \mathtt{F}(\tilde{u}(X_j, t + \Delta t))_k, \tag{26}$$

followed by

$$\bar{u}(X_k, t + \Delta t) = \text{exp}\{ik^2 \Delta t \frac{\pi^2}{P^2}\} \widehat{u}(X_k, t + \Delta t), \tag{27}$$

and

$$u(X_j, t + \Delta t) = \mathtt{F}^{-1}(\bar{u}(X_k, t + \Delta t))_j, \tag{28}$$

where Δt denotes the time step, and $\mathtt{F}$ and $\mathtt{F}^{-1}$ are the discrete Fourier transform and its inverse respectively. This is the split-step Fourier method corresponding to the first-order splitting approximation (6).

Similarly, the advancement in time from t to $t+\Delta t$ by the split-step Fourier method using the second-order splitting approximation (7) can be carried out by the following three steps:

(1') Advance the solution using the nonlinear part (23) through the following scheme

$$\tilde{u}(X_j, t+\frac{1}{2}\Delta t) = \exp\{-2i|u(X_j,t)|^2\frac{1}{2}\Delta t\}\, u(X_j,t).$$

(2') Advance the solution according to the linear part (25) by means of the discrete Fourier transforms

$$\bar{u}(X_k, t+\frac{1}{2}\Delta t) = \mathtt{F}^{-1}\big(\exp\{ik^2\Delta t\frac{\pi^2}{P^2}\}\mathtt{F}(\tilde{u}(X_j, t+\frac{1}{2}\Delta t))\big).$$

(3') Advance the solution using the nonlinear part (23) through the following scheme

$$u(X_j, t+\Delta t) = \exp\{-2i|\bar{u}(X_j, t+\frac{1}{2}\Delta t)|^2\frac{1}{2}\Delta t\}\, \bar{u}(X_j, t+\frac{1}{2}\Delta t).$$

The split-step method based on the fourth-order splitting approximation scheme (8) is described as follows. First, we advance in time from t to $t+\omega\Delta t$ by the second-order split-step Fourier method described above with

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

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

Numerical experiments

In our numerical experiments, we calculated the L_∞ norm, L_2 norm at the terminating time $T = 1$. We also calculated the relative errors, i_1, i_2, of the following two conserved quantities

$$I_1 = \int_{-\infty}^{+\infty} |u|^2 \, dx, \tag{29}$$

and

$$I_2 = \int_{-\infty}^{+\infty} \left(|u|^4 - |\frac{\partial u}{\partial x}|^2\right) dx, \tag{30}$$

respectively. The two conserved quantities are calculated by means of the Simpson's rule, and the derivatives in (30) are calculated using Fourier method.

We let $N = 512$ be a fixed number to keep spatial accuracy high, and perform numerical experiments for various values of time step Δt to show the convergence rates in time for different split-step schemes. The results are shown in Tables(1-3). It is clear that the first-order split-step Fourier method converges linearly in time. The convergence rates in time for the second-order and fourth-order split-step Fourier method are second-order and fourth-order, respectively, although we cannot guarantee the second- and fourth-order convergence rate in time for these methods theoretically. Moreover, the computational cost of the second-order scheme is 1.2 times of the first-order scheme, whereas the computational cost of the fourth-order scheme is about 3 times of the second-order scheme (See Tables(1-3)).

In order to show the convergence rate in space for these schemes, we perform numerical experiments for different values of N and a fixed value of time step $\Delta t = 0.000125$ to keep the temporal errors small. Tables(4-6) show the results. We can see that all of the three split-step Fourier methods converge exponentially in space.

Table 1: Convergence rates in time for the first-order splitting method

($N = 512,\ -10 \leq x \leq 10,\ 0 \leq t \leq 1,\ T = 1$).

Δt	L_∞	L_2	i_1	i_2	cpu(s)
0.004	1.13E-01	1.05E-01	2.81E-08	5.10E-02	0.81
0.002	3.44E-02	5.07E-02	4.36E-08	1.28E-02	1.54
0.001	1.72E-02	2.50E-02	5.92E-09	3.22E-03	2.87
0.0005	8.63E-03	1.24E-02	5.15E-09	8.10E-04	5.40
0.00025	4.32E-03	6.20E-03	6.34E-09	2.04E-04	9.24
0.000125	2.16E-03	3.09E-03	6.46E-09	5.24E-05	16.64

Table 2: Convergence rates in time for the second-order splitting method

($N = 512,\ -10 \leq x \leq 10,\ 0 \leq t \leq 1,\ T = 1$).

Δt	L_∞	L_2	i_1	i_2	cpu(s)
0.004	7.45E-02	2.54E-02	2.23E-07	3.42E-03	0.97
0.002	1.92E-02	6.21E-03	5.88E-08	1.52E-04	1.81
0.001	4.81E-03	1.55E-03	3.49E-09	1.55E-03	3.34
0.0005	1.20E-03	3.86E-04	4.04E-09	1.96E-06	6.39
0.00025	3.01E-04	9.61E-05	5.72E-09	1.66E-06	11.92
0.000125	7.54E-05	2.40E-05	6.14E-09	1.68E-06	21.72

Table 3: Convergence rates in time for the fourth-order splitting method

($N = 512,\ -10 \leq x \leq 10,\ 0 \leq t \leq 1,\ T = 1$).

Δt	L_∞	L_2	i_1	i_2	cpu(s)
0.004	3.66E-02	1.51E-02	1.65E-06	7.17E-03	2.94
0.002	1.94E-03	9.63E-04	1.25E-07	3.80E-05	5.13
0.001	1.28E-04	5.43E-05	3.57E-09	7.74E-07	9.87
0.0005	9.51E-06	3.53E-06	5.68E-09	1.65E-06	19.18
0.00025	6.25E-07	2.22E-07	6.26E-09	1.70E-06	34.90
0.000125	3.99E-08	1.39E-08	6.28E-09	1.70E-06	65.23

Table 4: Convergence rates in space for the first-order splitting method

($\Delta t = 0.000125,\ -10 \leq x \leq 10,\ 0 \leq t \leq 1,\ T = 1$).

N	L_∞	L_2	i_1	i_2	CPU(sec)
128	5.72+00	2.69E+00	2.88E-02	5.53E-01	3.92
160	1.76E+00	6.78E-01	1.04E-02	2.96E-01	5.02
192	7.97E-02	3.20E-02	1.78E-03	7.09E-02	5.84
224	5.00E-03	4.44E-03	1.05E-03	5.55E-02	7.33
256	2.25E-03	3.10E-03	3.17E-04	2.18E-02	8.05
384	2.16E-03	3.09E-03	2.38E-06	3.14E-04	12.58
512	2.16E-03	3.09E-03	6.46E-09	5.24E-05	16.64

Table 5: Convergence rates in space for the second-order splitting method ($\Delta t = 0.000125,\ -10 \le x \le 10,\ 0 \le t \le 1,\ T = 1$).

N	L_∞	L_2	i_1	i_2	CPU(sec)
128	5.74E+00	2.69E+00	2.97E-02	5.71E-01	5.35
160	1.75E+00	6.76E-01	1.03E-02	2.94E-01	6.66
192	7.99E-02	3.17E-02	1.77E-03	7.03E-02	7.86
224	5.38E-03	3.12E-03	1.04E-03	5.56E-02	9.09
256	5.27E-04	3.06E-04	3.14E-04	2.17E-02	10.85
384	7.48E-05	2.40E-05	2.37E-06	3.65E-04	16.07
512	7.54E-05	2.40E-05	6.14E-09	1.68E-06	21.72

Table 6: Convergence rates in space for the fourth-order splitting method ($\Delta t = 0.000125,\ -10 \le x \le 10,\ 0 \le t \le 1,\ T = 1$).

N	L_∞	L_2	i_1	i_2	CPU(sec)
128	5.74E+00	2.69E+00	2.96E-02	5.69E-01	15.15
160	1.76E+00	6.76E-01	1.04E-02	2.95E-01	19.39
192	8.00E-02	3.17E-02	1.77E-03	7.04E-02	21.91
224	5.36E-03	3.12E-03	1.04E-03	5.56E-02	26.36
256	4.99E-04	3.08E-04	3.15E-04	2.18E-02	29.10
320	1.30E-05	6.85E-06	3.45E-05	3.68E-03	39.29
384	4.75E-07	1.98E-07	2.39E-06	3.66E-04	46.59
512	3.99E-08	1.39E-08	6.28E-09	1.70E-06	65.23

The numerical solutions of the NLS equation (19) at $t = 1$ with initial condition (21) using the above split-step Fourier methods with $\Delta t = 0.000125$ and $N = 512$ are shown in Figure 1.

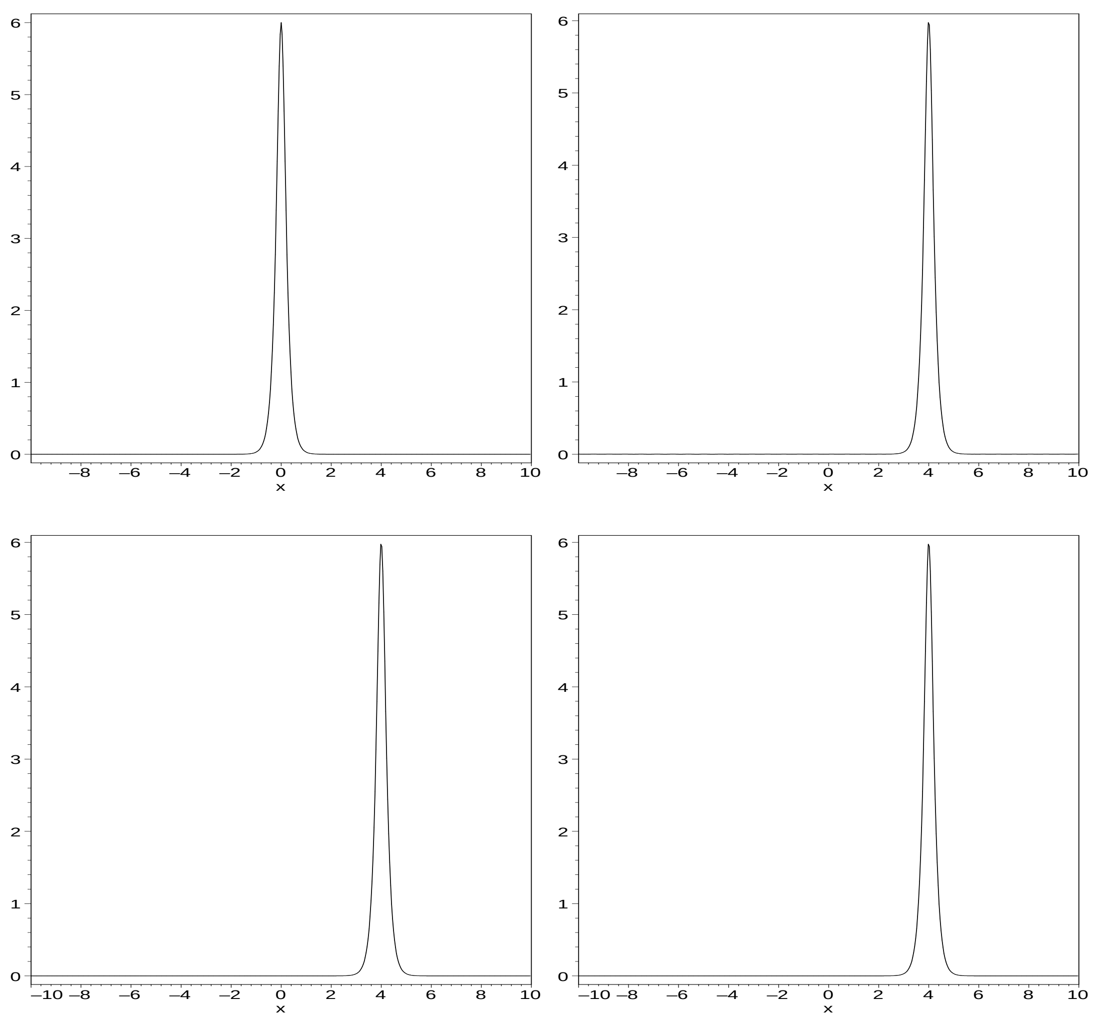

Fig.1: The modulus of the numerical solutions to equation (19). Top left, the initial condition at $t = 0$. Top right, the numerical solution at $t = 1$ using the first-order SSF. Bottom left, the numerical solution at $t = 1$ using the second-order SSF. Bottom right, the numerical solution at $t = 1$ using the fourth-order SSF.

Parallel Implementation

For first-order split-step Fourier method, we parallelize each of the four computational steps arise in (24) and (26)–(28).

Let A, of size N, be the approximate solution to u at time t. Suppose there are p processors in a distributed-memory parallel computer. Parallelizing (24) and (27) are straightforward. We distribute the array A among p processors. Processor n, $0 \leq n \leq p-1$, contains array elements $A[nN/p]$ to $A[(n+1)N/p-1]$. Each of the p processor works on its own subarrays independently without communicating with others. We employ FFTW's MPI routines to implement parallel discrete Fourier transforms to parallelize the computations in stages (26) and (28).

The parallel algorithms for the second-order and fourth-order split-step Fourier methods can be developed in a straightforward manner.

Parallel algorithms of the split-step Fourier methods are implemented on the Origin 2000 multiprocessor computer. All the codes are optimized at the same optimization level. All timings are the total wall-clock time for execution of the code. The results are shown on Tables(7 -9). The speedup S_p is defined by

$$S_p = \frac{\text{Time spent to run the MPI code on one processor}}{\text{Time spent to run the MPI code on } p \text{ processors}}.$$

From the results, it is clear that the speedup increases as the problem size N becomes larger for a fixed number of processors p. For small problem sizes the computation/communication ratio is small, thus speedup is small. For fixed p, we can also see that the fourth-order scheme has a better speedup than the second-order scheme, whereas the second-order scheme has a slightly better speedup than the first-order scheme. This is due to the fact that the fourth-order scheme is more computational intensive than the second-order scheme, whereas the second-order scheme is more computational intensive than the first-order scheme. For large N, the speedups achieved on the multiprocessor computer running the parallel codes are considerable.

Table 7: Results for parallel implementation of first-order split-step Fourier method (Δt = 0.0005). N indicates array size, NS the number of steps, t_p the time on p processors, S_p the speedup on p processors.

	N=2^{12} NS=2000	N=2^{14} NS=500	N=2^{16} NS=125	N=2^{18} NS = 32
t_1(sec)	11.4	11.4	12.9	23.8
t_2(sec)	9.7	9.5	9.5	16.9
t_4(sec)	7.6	5.6	6.2	9.6
t_8(sec)	5.7	3.7	3.6	5.3
S_2=t_1/t_2	1.2	1.2	1.4	1.4
S_4=t_1/t_4	1.5	2.1	2.1	2.5
S_8=t_1/t_8	2.0	3.1	3.6	4.5

Table 8: Results for parallel implementation of second-order split-step Fourier method (Δt = 0.0005). N indicates array size, NS the number of steps, t_p the time on p processors, S_p the speedup on p processors.

	N=2^{12} NS=2000	N=2^{14} NS=500	N=2^{16} NS=125	N=2^{18} NS = 32
t_1(sec)	12.3	12.5	13.8	25.4
t_2(sec)	10.2	9.6	10.2	16.4
t_4(sec)	7.3	6.1	6.4	9.3
t_8(sec)	5.9	3.9	3.7	5.2
S_2=t_1/t_2	1.2	1.3	1.4	1.5
S_4=t_1/t_4	1.7	2.1	2.2	2.7
S_8=t_1/t_8	2.1	3.2	3.7	4.9

Table 9: Results for parallel implementation of fourth-order split-step Fourier method ($\Delta t = 0.0005$). N indicates array size, NS the number of steps, t_p the time on p processors, S_p the speedup on p processors.

	N=2^{12} NS=2000	N=2^{14} NS=500	N=2^{16} NS=125	N=2^{18} NS = 32
t_1(sec)	45.5	45.0	48.1	75.8
t_2(sec)	34.1	34.0	35.3	48.2
t_4(sec)	24.5	20.2	21.8	26.6
t_8(sec)	17.8	13.4	13.0	15.2
S_2=t_1/t_2	1.3	1.3	1.4	1.6
S_4=t_1/t_4	1.9	2.2	2.2	2.8
S_8=t_1/t_8	2.6	3.4	3.7	5.0

More numerical experiments

In this section, we examine the NLS equation

$$iv_t = \frac{1}{2}\frac{\partial^2 v}{\partial x^2} + |v|^2 v, \tag{31}$$

with the initial condition

$$v(x, 0) = A\text{sech}(x). \tag{32}$$

It is known that solitary solutions are obtained whenever A is an integer. In fact, the explicit solutions in the case of $A = 1$ and $A = 2$ are given by

$$v(x,t) = \exp(-it/2)\text{sech}(x),$$

and

$$v(x,t) = 4\exp(-it/2)\,\frac{\cosh(3x) + 3\exp(-4it)\cosh(x)}{\cosh(4x) + 4\cosh(2x) + 3\cos(4t)},$$

respectively. When $A = 1$, the solution preserves its initial shape during the course of the time of simulation. For higher integer A, however, the solutions have periods $t = \pi/2$.

Numerical solutions of the NLS equation (31) with initial condition (32) using split-step Fourier method are shown in Figures 2- 3 for $A = 2, 3$ and 4. These solutions agree well with the exact solutions.

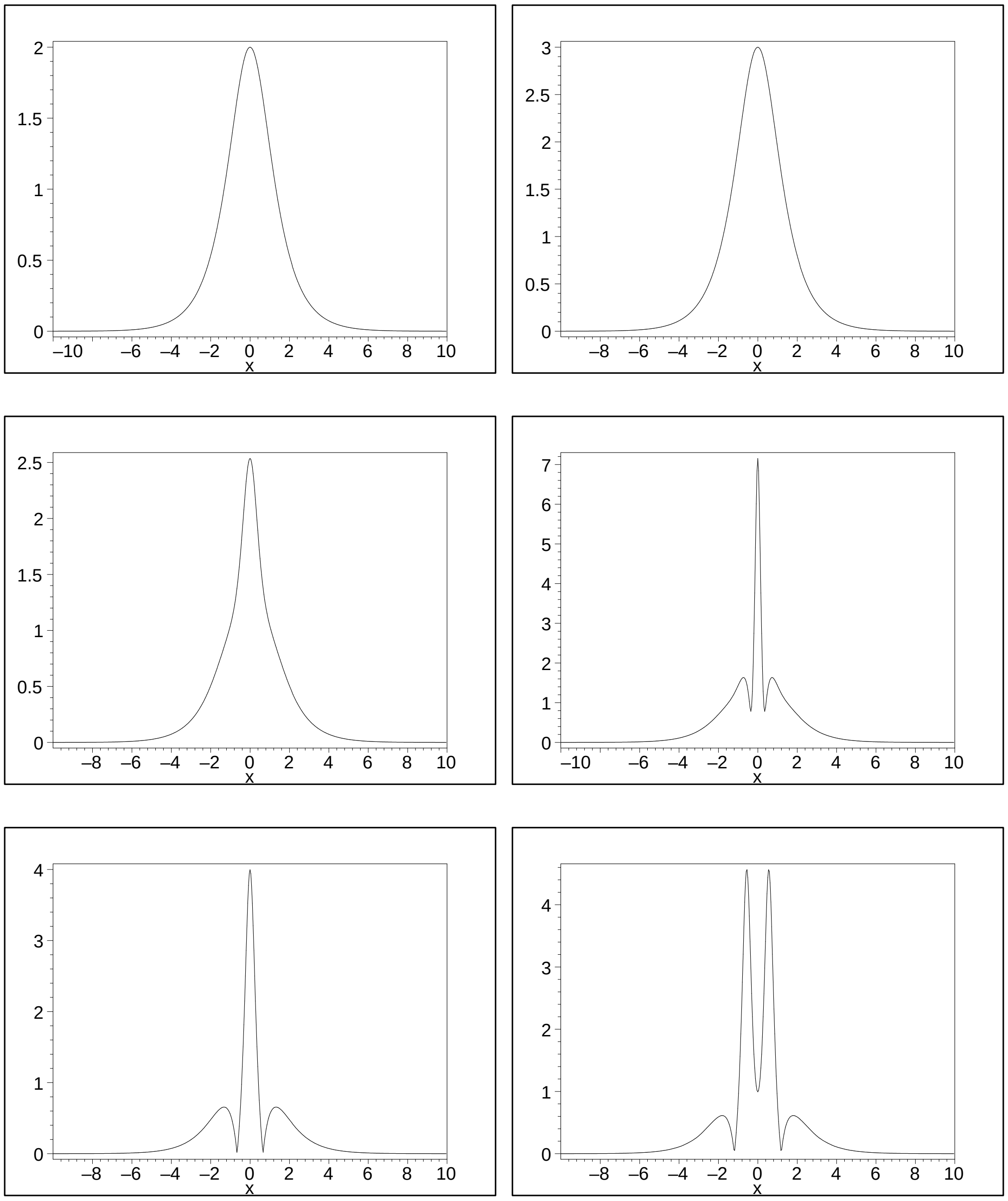

Fig.2: The modulus of the numerical solutions to equation (31). Left, the $A = 2$ soliton at $t = 0, \pi/8$, and $\pi/4$. Right, the $A = 3$ soliton at $t = 0, \pi/8$, and $\pi/4$.

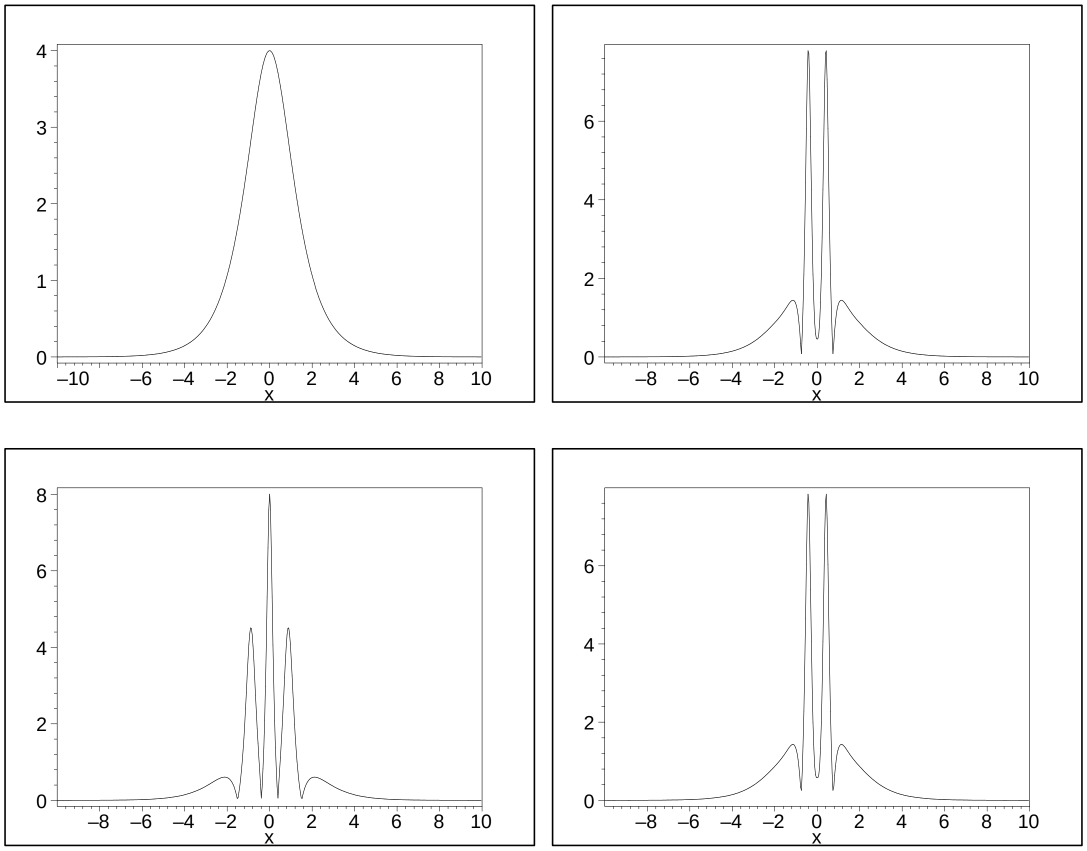

Fig.3: The modulus of the numerical solutions to (31) for $A = 4$. Above, $t = 0$ and $\pi/8$; below, $t = \pi/4$ and $3\pi/8$.

Perturbed nonlinear Schrödinger equation

In this section, we examine the perturbed NLS equation of the form:

$$iw_t + \frac{1}{2}D(t)\frac{\partial^2 w}{\partial x^2} + g(t)|w|^2 w = 0, \tag{33}$$

where $D(t)$ represents dispersion, which is given by the following periodic function

$$D(t) = \begin{cases} D_1, & 0 \le t < \theta t_m, \\ D_2, & \theta t_m \le t < t_m, \end{cases} \qquad (0 \le \theta < 1)$$

and $g(t)$ relates to effective nonlinearity, which is given by the periodic function

$$g(t) = \Big(\frac{2\Gamma}{1 - e^{-2\Gamma t_a}}\Big)e^{-2\Gamma t}, \text{ for } 0 \le t \le t_a.$$

In our numerical experiments, we have chosen $D_1 = 1$, $D_2 = -1$, $\theta = 0.8$, the map period $t_m = 0.1$, the damping coefficient $\Gamma = 4$, and the amplifier spacing $t_a = 0.1$.

We study equation (33) with periodic boundary condition of period $[-20, 20]$, and initial condition of the form

$$w(t,x) = A\,\text{sech}[A(x-\Omega t-x_0)]\,\text{exp}\{i[\Omega x-\frac{1}{2}(\Omega^2-A^2)t+\varphi]\}, \tag{34}$$

with $t = 0$, amplitude $A = 1$, velocity $\Omega = 2$, initial position $x_0 = 0$, and phase $\varphi = 0$.

Numerical solutions of the perturbed NLS equation (33) with initial condition (34) using split-step Fourier method are shown in Figure 4.

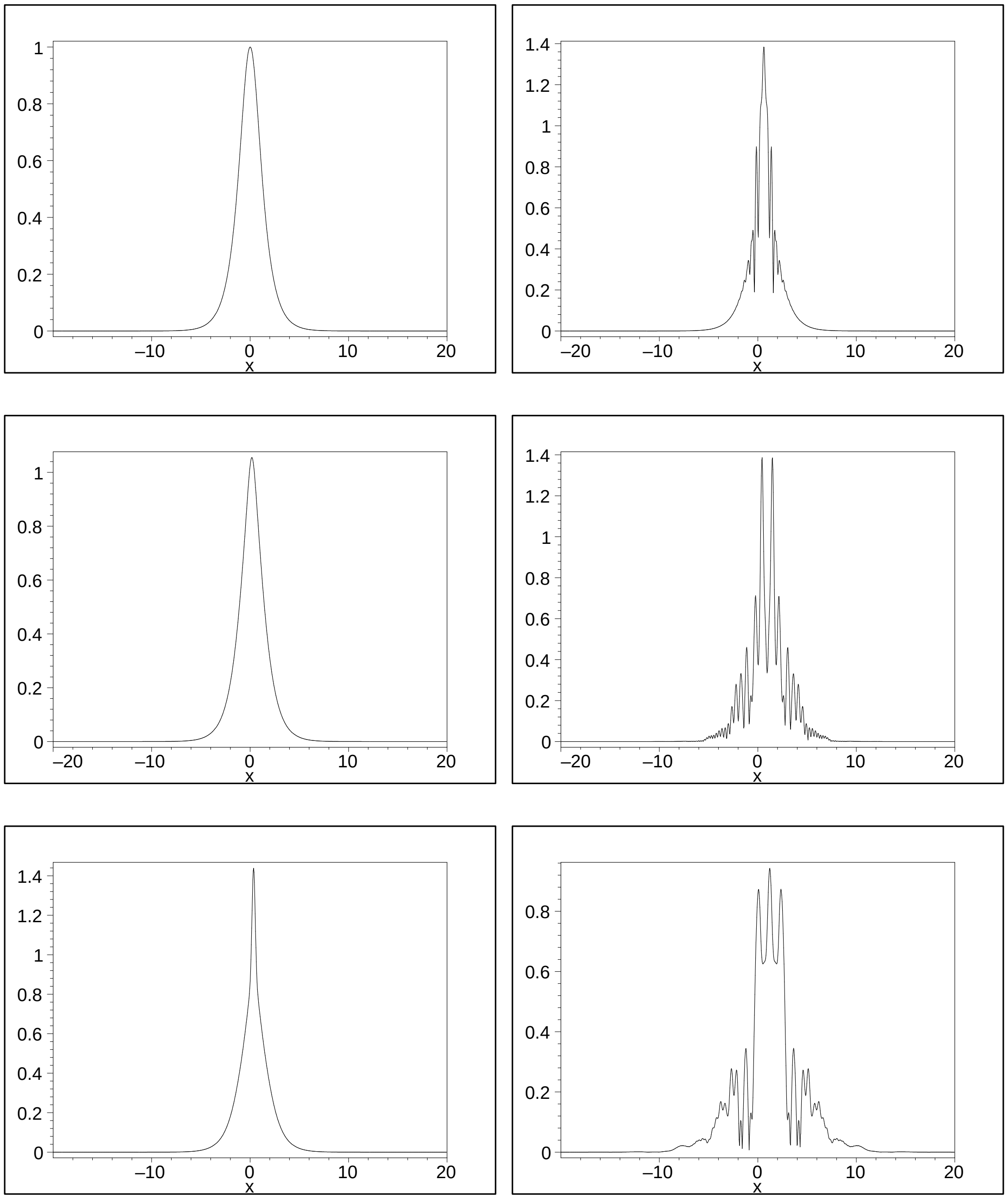

Fig.4: The modulus of the numerical solutions to (33). Left, from top to bottom, the solution at $t = 0$, 0.125, 0.25. Right, from top to bottom, the solution at $t =$ 0.5, 0.75, 1.0.

Modified nonlinear Schrödinger equation

The pulse propagation in a dispersion exponentially decreasing fiber can be described by the modified NLS equation

$$iU_t - \frac{1}{2}\beta_2(t)\frac{\partial^2 U}{\partial x^2} - i\frac{1}{6}\beta_3\frac{\partial^3 U}{\partial x^3} + \gamma|U|^2U = -i\frac{1}{2}\alpha U, \qquad (35)$$

where U is the normalized field envelope; $\beta_2(t)$ is the second-order dispersion; β_3 is the third-order dispersion; $\gamma = n_2\omega/cA_{\text{eff}}$, where n_2 is the Kerr coefficient, ω is the carrier frequency, c is the velocity of the light in vacuum, and A_{eff} is the effective fiber cross section; α is the fiber loss.

In our numerical experiments, we let $\beta_2(t)$ be a periodic function

$$\beta_2(t) = e^{-\alpha t}\beta_2(0), \text{ for } 0 \le t < 0.01,$$

where $\beta_2(0) = -0.5$. Other parameters are taken to be $\beta_3 = 0.14, \gamma = 3.2/1.55$, and $\alpha = 0.2$.

We study equation (35) with periodic boundary condition of period $[-10, 10]$, and the initial condition

$$U(x, 0) = \text{sech}(x)\, e^{ix}. \tag{36}$$

Numerical solutions of the modified NLS equation (35) with initial condition (36) using split-step Fourier method are shown in Figure 5. From our numerical experiments, we found that energy of the soliton, $\int_{-10}^{10} |U|^2\, dx$, which is equal to 2.0 at $t = 0$, decreases to 1.0976 at $t = 3$.

If we choose $\alpha = 0$, then the soliton energy is well conserved. It remains as a constant 2.0 for $t = 0$ to $t = 3$. Figure 6. shows the numerical solution of the modified NLS equation (35) with initial condition (36).

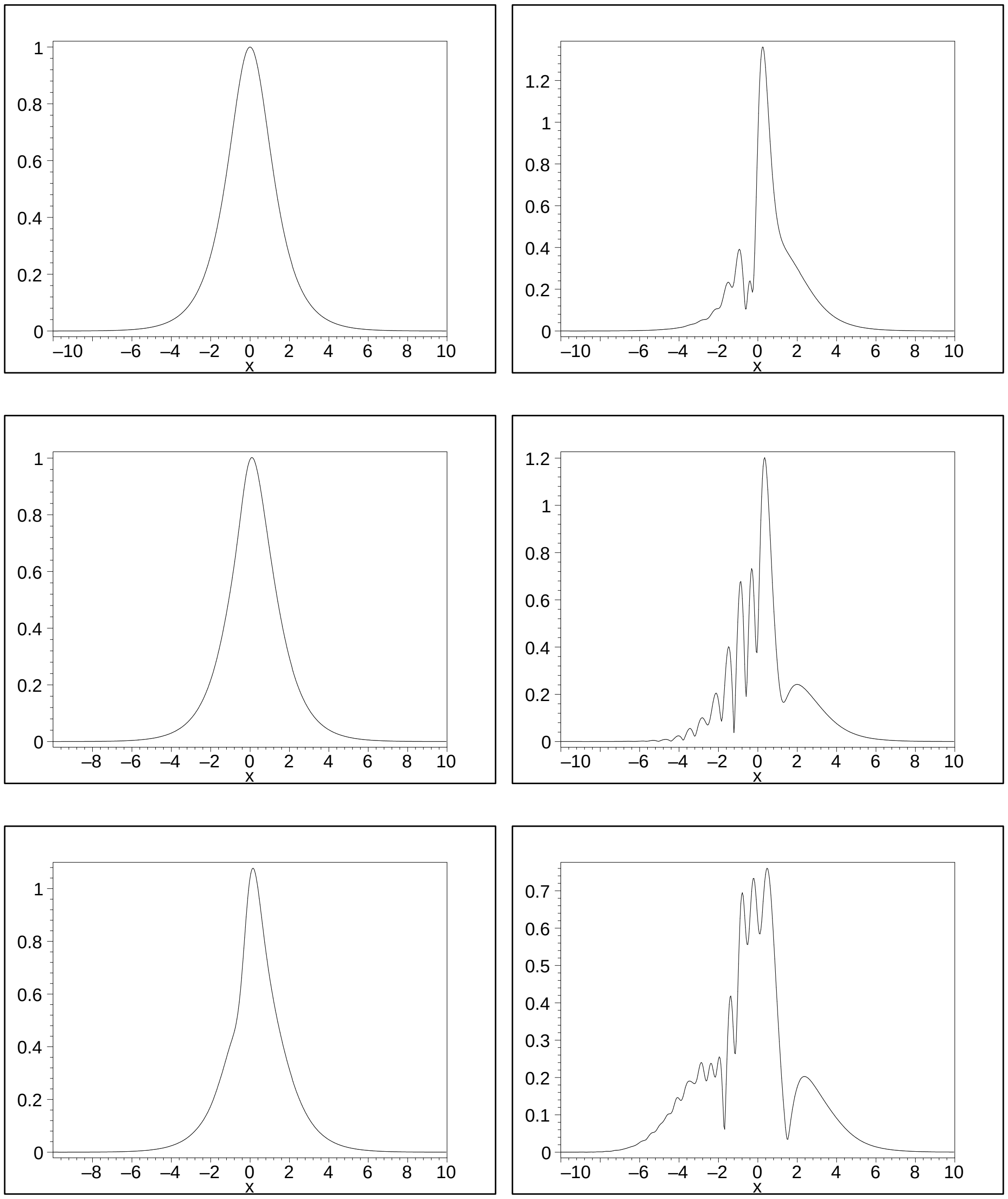

Fig.5: The modulus of the numerical solutions to (35) for $\alpha = 0.2$. From top to bottom, the solution at $t = 0$, 0.375, 0.75. Right, from top to bottom, the solution at $t = 1.5$, 2.25, 3.0.

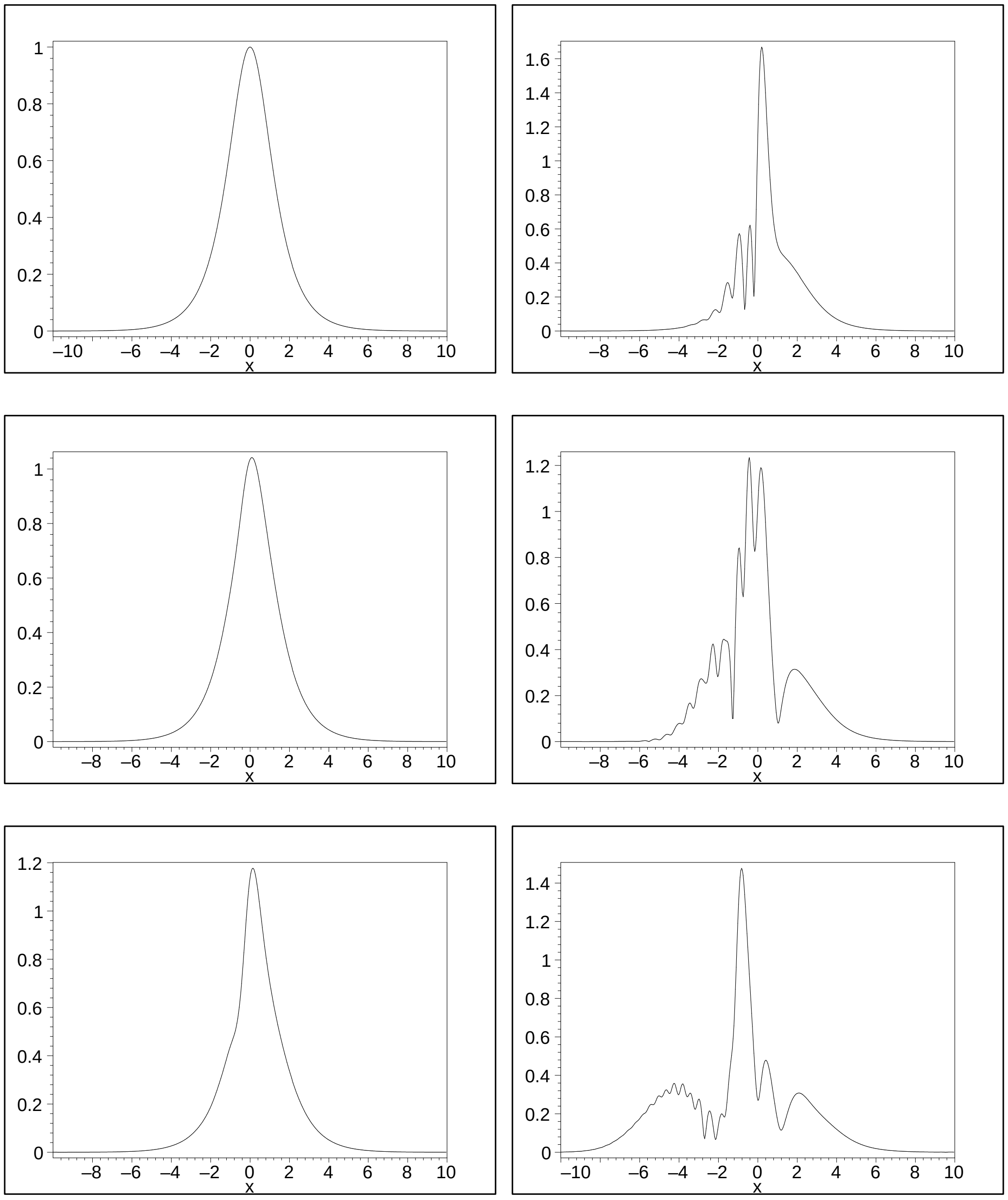

Fig.6: The modulus of the numerical solutions to (35) for $\alpha = 0$. Left, from top to bottom, the solution at t = 0, 0.375, 0.75. Right, from top to bottom, the solution at t = 1.5, 2.25, 3.0.

NUMERICAL METHODS FOR THE CNLS EQUATION

The CNLS equation is of tremendous interest in both theory and applications. The governing equation for the propagation of two orthogonally polarized pulses in a monomode birefringent fibers is given by a CNLS equation.

We consider a CNLS equation of the form:

$$\begin{aligned} i\Big(\frac{\partial\psi_1}{\partial t}+\delta\,\frac{\partial\psi_1}{\partial x}\Big)+\frac{1}{2}\frac{\partial^2\psi_1}{\partial x^2}+(|\psi_1|^2+\mu|\psi_2|^2)\,\psi_1=0,\\ i\Big(\frac{\partial\psi_2}{\partial t}-\delta\,\frac{\partial\psi_2}{\partial x}\Big)+\frac{1}{2}\frac{\partial^2\psi_2}{\partial x^2}+(\mu|\psi_1|^2+|\psi_2|^2)\,\psi_2=0, \end{aligned} \tag{37}$$

where ψ_1 and ψ_2 are the two polarized waves, μ is a real parameter, $i=\sqrt{-1}$, and δ is the normalized strength of the linear birefringence.

In general, the CNLS equation with arbitrary coefficients is not integrable. For $\mu = 1$, equation (37) reduces to the Manakov equation which is integrable.

The explicit form of soliton solution of the equation (37) is given by

$$\psi_1 = \sqrt{\frac{2\alpha}{1+\mu}}\,\mathrm{sech}[\sqrt{2\alpha}(x - vt)]\ \exp\{i(v-\delta)x - i[\frac{1}{2}(v^2-\delta^2) - \alpha]\,t\},$$
$$\psi_2 = \pm\sqrt{\frac{2\alpha}{1+\mu}}\,\mathrm{sech}[\sqrt{2\alpha}(x - vt)]\ \exp\{i(v+\delta)x - i[\frac{1}{2}(v^2-\delta^2) - \alpha]\,t\}, \tag{38}$$

The CNLS equation has the following two conserved quantities

$$E_1 = \int_{-\infty}^{+\infty} |\psi_1|^2 \, dx, \tag{39}$$

and

$$E_2 = \int_{-\infty}^{+\infty} |\psi_2|^2 \, dx, \tag{40}$$

that remain constant in time. Note that they represent the energy of the system. From the exact solution (38), it is easy to show that

$$E_1 = E_2 = \frac{2}{1+\mu}\sqrt{2\alpha}. \tag{41}$$

Recently, M. S. Ismail and T. R. Taha introduced a finite difference method for a numerical simulation of the CNLSE.

In this talk we employ the well known split-step Fourier method for the numerical simulation of the CNLS equations. We also present a parallelization of the split-step Fourier method using the *Fastest Fourier Transform in the West* (FFTW) developed by M. Frigo and S. G. Johnson.

Space discretization

Although the CNLS equation (37) is defined over the real line, we need to impose conditions at a finite interval $[x_l, x_r]$ for numerical computation. For the numerical experiments considered, we assume that the solution of equation (37) is negligibly small outside the interval $[x_l, x_r]$. The boundaries are far apart enough so that they do not affect the propagation of solitary waves.

In the following, we study the coupled nonlinear Schrödinger equation

$$\begin{aligned} i(\frac{\partial \psi_1}{\partial t} + \delta \frac{\partial \psi_1}{\partial x}) + \frac{1}{2}\frac{\partial^2 \psi_1}{\partial x^2} + (|\psi_1|^2 + \mu|\psi_2|^2)\,\psi_1 = 0, \\ i(\frac{\partial \psi_2}{\partial t} - \delta \frac{\partial \psi_2}{\partial x}) + \frac{1}{2}\frac{\partial^2 \psi_2}{\partial x^2} + (\mu|\psi_1|^2 + |\psi_2|^2)\,\psi_2 = 0, \end{aligned} \tag{42}$$

and assume that ψ_1 and ψ_2 satisfy the initial conditions

$$\psi_1(x,0) = g_1(x),\ \psi_2(x,0) = g_2(x),\ x \in [x_l, x_r], \tag{43}$$

and periodic boundary conditions

$$\begin{aligned} \psi_1(x_l,t) = \psi_1(x_r,t),\ t \in [0,T], \\ \psi_2(x_l,t) = \psi_2(x_r,t),\ t \in [0,T]. \end{aligned} \tag{44}$$

The space discretization is accomplished by a Fourier method. For convenience, the finite interval $[x_l, x_r]$ is normalized to $[0, 2\pi]$ by the linear transform $X = (x - x_l)\pi/P$, where P is the half length of the interval, i.e. $P = (x_r - x_l)/2$. Equations (42)–(44) may be rewritten as

$$
\begin{aligned}
i(\frac{\partial \psi_1}{\partial t} + \frac{\delta\pi}{P}\frac{\partial \psi_1}{\partial X}) + \frac{1}{2}\frac{\pi^2}{P^2}\frac{\partial^2 \psi_1}{\partial X^2} + (|\psi_1|^2 + \mu|\psi_2|^2)\,\psi_1 = 0,\\
i(\frac{\partial \psi_2}{\partial t} - \frac{\delta\pi}{P}\frac{\partial \psi_2}{\partial X}) + \frac{1}{2}\frac{\pi^2}{P^2}\frac{\partial^2 \psi_2}{\partial X^2} + (\mu|\psi_1|^2 + |\psi_2|^2)\,\psi_2 = 0,
\end{aligned}
\tag{45}
$$

with initial conditions

$$
\psi_1(X, 0) = \tilde{g}_1(X),\ \psi_2(X, 0) = \tilde{g}_2(X),\ X \in [0, 2\pi], \tag{46}
$$

and periodic boundary conditions

$$
\begin{aligned}
\psi_1(0, t) = \psi_1(2\pi, t),\ t \in [0, T],\\
\psi_2(0, t) = \psi_2(2\pi, t),\ t \in [0, T].
\end{aligned}
\tag{47}
$$

The interval $[0, 2\pi]$ is divided into N equal subintervals with grid spacing $\Delta X = 2\pi/N$. The spatial grid points are denoted by $X_j = j\Delta X$, $j = 0, 1, \ldots, N$. Let $\Psi_1^j(t)$ and $\Psi_2^j(t)$ be the numerical approximation to $\psi_1(X_j, t)$ and

$\psi_2(X_j,t)$ at time t, respectively. The discrete Fourier transform for the sequences $\{\Psi_m^j\}$ is defined as

$$\widehat{\Psi}_m^k = \mathtt{F}(\Psi_m)_k = \frac{1}{\sqrt{N}} \sum_{j=0}^{N-1} \Psi_m^j e^{-ikX_j},\ -\frac{N}{2} \le k \le \frac{N}{2}-1,\ m=1,2, \tag{48}$$

The inverse discrete Fourier transform is given by

$$\widehat{\Psi}_m^j = \mathtt{F}^{-1}(\widehat{\Psi}_m)_j = \frac{1}{\sqrt{N}} \sum_{j=-\frac{N}{2}}^{\frac{N}{2}-1} \widehat{\Psi}_m^k e^{ijX_k},\ 0 \le j \le N-1,\ m=1,2. \tag{49}$$

These transforms can be implemented very efficiently by a fast Fourier transform algorithm, e.g. the Fastest Fourier Transform in the West (FFTW).

Time integration

We use split-step Fourier method for the coupled nonlinear Schrödinger equation (45). The basic idea is to split the exponential operator $\exp[\Delta t(\mathcal{L}+\mathcal{N})]$ using the Baker-Campbell-Hausdorf formula as discussed in section . For instance, the first-order version of the split-step method (6) is carried out as the following two steps for the advancement in time from t to $t+\Delta t$.

(1) Advance the solution using only the nonlinear part:

$$\begin{aligned} i\frac{\partial\psi_1}{\partial t} + (|\psi_1|^2 + \mu|\psi_2|^2)\,\psi_1 = 0, \\ i\frac{\partial\psi_2}{\partial t} + (\mu|\psi_1|^2 + |\psi_2|^2)\,\psi_2 = 0, \end{aligned} \tag{50}$$

through the following scheme

$$\begin{aligned} \tilde{\psi}_1(X_j, t+\Delta t) = \exp\{i(|\psi_1(X_j,t)|^2 + \mu|\psi_2(X_j,t)|^2)\Delta t\}\,\psi_1(X_j,t), \\ \tilde{\psi}_2(X_j, t+\Delta t) = \exp\{i(\mu|\psi_1(X_j,t)|^2 + |\psi_2(X_j,t)|^2)\Delta t\}\,\psi_2(X_j,t). \end{aligned} \tag{51}$$

(2) Advance the solution according to the linear part:

$$\begin{aligned} i\left(\frac{\partial\psi_1}{\partial t} + \frac{\delta\pi}{P}\frac{\partial\psi_1}{\partial X}\right) + \frac{1}{2}\frac{\pi^2}{P^2}\frac{\partial^2\psi_1}{\partial X^2} = 0, \\ i\left(\frac{\partial\psi_2}{\partial t} - \frac{\delta\pi}{P}\frac{\partial\psi_2}{\partial X}\right) + \frac{1}{2}\frac{\pi^2}{P^2}\frac{\partial^2\psi_2}{\partial X^2} = 0, \end{aligned} \tag{52}$$

by means of computing

$$\begin{aligned}\widehat{\psi}_1(X_k, t+\Delta t) &= \mathtt{F}(\tilde{\psi}_1(X_j, t+\Delta t))_k,\\ \widehat{\psi}_2(X_k, t+\Delta t) &= \mathtt{F}(\tilde{\psi}_2(X_j, t+\Delta t))_k,\end{aligned} \tag{53}$$

followed by

$$\begin{aligned}\breve{\psi}_1(X_k, t+\Delta t) &= \exp\{i(-\frac{1}{2}\frac{\pi^2}{P^2}k^2 - \frac{\pi}{P}\delta k)\Delta t\}\widehat{\psi}_1(X_k, t+\Delta t),\\ \breve{\psi}_2(X_k, t+\Delta t) &= \exp\{i(-\frac{1}{2}\frac{\pi^2}{P^2}k^2 + \frac{\pi}{P}\delta k)\Delta t\}\widehat{\psi}_2(X_k, t+\Delta t),\end{aligned} \tag{54}$$

and

$$\begin{aligned}\psi_1(X_j, t+\Delta t) &= \mathtt{F}^{-1}(\breve{\psi}_1(X_k, t+\Delta t))_j,\\ \psi_2(X_j, t+\Delta t) &= \mathtt{F}^{-1}(\breve{\psi}_2(X_k, t+\Delta t))_j,\end{aligned} \tag{55}$$

where the transform $\mathtt{F}$ and its inverse $\mathtt{F}^{-1}$ are given by (48) and (49), respectively.

Similarly, the advancement in time from t to $t+\Delta t$ by the split-step Fourier method using the second order splitting approximation (7) is described in the following three steps.

(1') Advance the solution using the nonlinear part (50) through the following scheme

$$\tilde{\psi}_1(X_j, t+\frac{1}{2}\Delta t) = \text{exp}\{i(|\psi_1(X_j,t)|^2 + \mu|\psi_2(X_j,t)|^2)\frac{1}{2}\Delta t\}\,\psi_1(X_j,t),$$
$$\tilde{\psi}_2(X_j, t+\frac{1}{2}\Delta t) = \text{exp}\{i(\mu|\psi_1(X_j,t)|^2 + |\psi_2(X_j,t)|^2)\frac{1}{2}\Delta t\}\,\psi_2(X_j,t).$$

(2') Advance the solution according to the linear part

(52) by means of the discrete Fourier transforms

$$\bar{\psi}_1(X_j, t+\frac{1}{2}\Delta t) = \mathtt{F}^{-1}\big(\text{exp}\{i(-\frac{1}{2}\frac{\pi^2}{P^2}k^2 - \frac{\pi}{P}\delta k)\Delta t\}\mathtt{F}(\tilde{\psi}_1(X_j, t+\frac{1}{2}\Delta t))\big),$$
$$\bar{\psi}_2(X_j, t+\frac{1}{2}\Delta t) = \mathtt{F}^{-1}\big(\text{exp}\{i(-\frac{1}{2}\frac{\pi^2}{P^2}k^2 + \frac{\pi}{P}\delta k)\Delta t\}\mathtt{F}(\tilde{\psi}_2(X_j, t+\frac{1}{2}\Delta t))\big).$$

(3') Advance the solution using the nonlinear part (50)

through the following scheme

$$\psi_1(X_j, t+\Delta t) = \text{exp}\{i(|\bar{\psi}_1(X_j, t+\frac{1}{2}\Delta t)|^2 + \mu|\bar{\psi}_2(X_j, t+\frac{1}{2}\Delta t)|^2)\frac{1}{2}\Delta t\}\bar{\psi}_1($$
$$\psi_2(X_j, t+\Delta t) = \text{exp}\{i(\mu|\bar{\psi}_1(X_j, t+\frac{1}{2}\Delta t)|^2 + |\bar{\psi}_2(X_j, t+\frac{1}{2}\Delta t)|^2)\frac{1}{2}\Delta t\}\bar{\psi}_2($$

The split-step method based on the fourth-order splitting approximation scheme (8) can be developed in a similar fashion. First, we advance in time from t to $t + \omega\Delta t$ by the second-order split-step Fourier method described above with

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

Then we advance in time from $t + \omega\Delta t$ to $t + (1 - \omega)\Delta t$ by the second-order split-step Fourier method. Finally, we advance in time from $t + (1 - \omega)\Delta t$ to $t + \Delta t$ by the second-order split-step Fourier method, and obtain approximations to $\psi_1(x, t + \Delta t)$ and $\psi_2(x, t + \Delta t)$.

Parallel algorithm

Large-scale numerical simulations of coupled nonlinear Schrödinger equation are required for many problems in fiber optics. Such a simulation is computationally intensive and time consuming using the sequential split-step Fourier (SSF) method described in section . In this section, we will discuss a parallel algorithm for the SSF method.

For first-order split-step Fourier method, we parallelize each of the four computational steps arise in (51) and (53)–(55).

Let A, of size N, be the array that includes the approximate solution to the ψ_1 or ψ_2 at time t. Suppose that there are p processors in a distributed-memory parallel computer. Parallelizing (51) and (54) are straightforward. We distribute the array A among p processors. Processor n, $0 \leq n \leq p-1$, contains array elements

$A[nN/p]$ to $A[(n+1)N/p-1]$. Each of the p processor works on its own subarrays independently without involving interprocessor communication.

However, the computation stages (53) and (55) cannot be parallelized in a straightforward manner, because all of the elements in the array A are used to compute each element of $\mathtt{F}(A)_k$ and $\mathtt{F}^{-1}(A)_j$, an element after forward DFT and backward DFT of A, respectively. We employ FFTW's MPI routines to implement parallel discrete Fourier transforms. The basic idea is as follows.

Suppose N can be factored as $N = p_1 p_2$, then the indices j and k can be represented as

$$j = j_1 p_2 + j_0;\ j_1 = 0, \ldots, p_1 - 1,\ j_0 = 0, \ldots, p_2 - 1,$$

and

$$k = k_1 p_1 + k_0;\ k_1 = 0, \ldots, p_2 - 1,\ k_0 = 0, \ldots, p_1 - 1.$$

We rewrite the discrete Fourier transform

$$\widehat{A}_k = \frac{1}{\sqrt{N}} \sum_{j=0}^{N-1} A_j\, \omega_N^{-jk},\ 0 \le k < N, \tag{56}$$

as the form

$$\widehat{A}_k = \frac{1}{\sqrt{N}} \sum_{j_0=0}^{p_2-1} \Big[\big(\sum_{j_1=0}^{p_1-1} A_{j_1 p_2 + j_0} \omega_N^{-j_1 k_0 p_2} \big) \omega_N^{-j_0 k_0} \Big] \omega_N^{-j_0 k_1 p_1}. \qquad (57)$$

where $\omega_N = e^{i\frac{2\pi}{N}}$ is a primitive N-th root of unity. It follows that

$$\widehat{A}_{k_1 p_1 + k_0} = \frac{1}{\sqrt{p_2}} \sum_{j_0=0}^{p_2-1} \big(\tilde{A}_{j_0,k_0} \omega_N^{-j_0 k_0} \big) \omega_{p_2}^{-j_0 k_1}, \qquad (58)$$

where

$$\tilde{A}_{j_0,k_0} = \frac{1}{\sqrt{p_1}} \sum_{j_1=0}^{p_1-1} A_{j_1 p_2 + j_0} \omega_{p_1}^{-j_1 k_0}. \qquad (59)$$

The algorithm computes p_2 independent DFTs of size p_1 according to (59). Then it multiplies the results by the so-called twiddle factors $\omega_N^{-j_0 k_0}$, and finally performs p_1 independent DFTs of size p_2 according to (58). It is necessary to communicate data between processors.

Similarly, we can develop parallel version of the second-order and fourth-order split-step Fourier methods.

NUMERICAL EXPERIMENTS

In this talk, we investigate the performance of the proposed split-step Fourier methods by performing some numerical experiments.

Sequential split-step Fourier methods One solitary wave solution We restrict ourselves to problems with known analytical solution, so that we are able to investigate the performance of the proposed split-step Fourier methods.

We consider the CNLS equation (37) with the initial conditions

$$\begin{aligned} \psi_1(x,0) &= \sqrt{\frac{2\alpha}{1+\mu}}\,\mathrm{sech}[\sqrt{2\alpha}\,x]\;\exp\{i(v-\delta)x\}, \\ \psi_2(x,0) &= \sqrt{\frac{2\alpha}{1+\mu}}\,\mathrm{sech}[\sqrt{2\alpha}\,x]\;\exp\{i(v+\delta)x\}, \end{aligned} \tag{60}$$

where α, μ and v are constants. The problem has known solitary wave solution given by (38).

We choose $\alpha = 0.5$, $\delta = 0.5$, $\mu = 2/3$ and $v = 1.0$. For this problem the exact values of conserved quantities are

$$E_1 = 1.2,\; E_2 = 1.2.$$

The problem is solved in the interval $-20 \leq x \leq 80$ for $0 \leq t \leq 60$ using first-order, second-order and fourth-order split-step schemes.

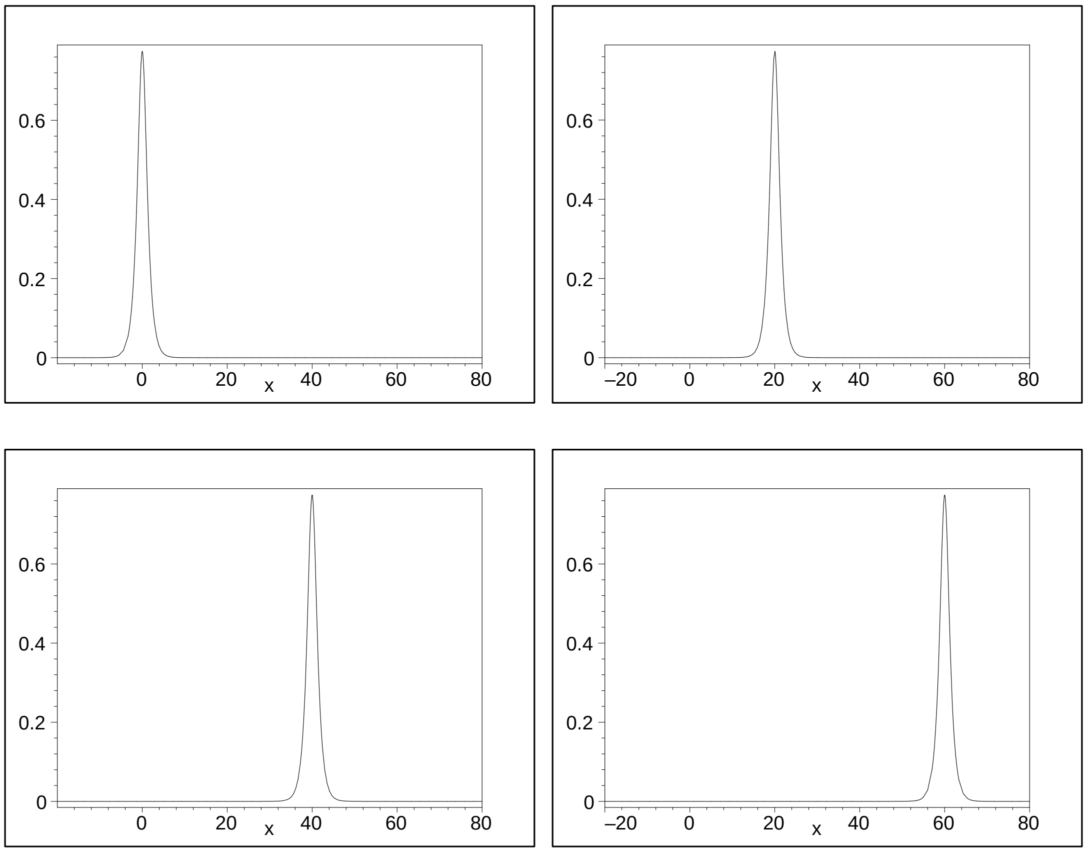

Fig.7: The modulus of a single soliton moving to the right with velocity $v = 1$, at time $t = 0, 20, 40$ and 60, respectively. Here $\alpha = 0.5$, $\delta = 0.5$, $\mu = 2/3$. Only the values of $|\psi_1|$ are shown above.

In our numerical experiments, we present L_∞-errors, L_2-errors, and relative errors of two conserved quantities at the terminating time $T = 60$. They are defined by

$$\max_{m=1,2}\{ \max_{0\le j\le N-1} \big|\, |\tilde{\psi}_m(x_j,T)| - |\psi_m(x_j,T)| \,\big| \},$$

$$\max_{m=1,2}\Big\{ \Big[\sum_{j=0}^{N-1} (\, |\tilde{\psi}_m(x_j,T)| - |\psi_m(x_j,T)| \,)^2 \Delta x \Big]^{1/2} \Big\},$$

and

$$e_1 = \frac{|\tilde{E}_1 - E_1|}{E_1},\ e_2 = \frac{|\tilde{E}_2 - E_2|}{E_2},$$

respectively, where $\tilde{\psi}_m(x_j,T)$ denotes the numerical approximation of $\psi_m(x_j,T)$, and $\tilde{E}_m$ denotes the numerical approximation of E_m for $m = 1,2$. The two conserved quantities are computed by the well-known Simpson's rule.

In order to show the convergence rates in time for different split-step schemes, we let $N = 512$ to keep spatial accuracy high and perform numerical experiments for various values of time step Δt. Tables (10 - 12) show the results. The results show that each of these schemes preserves the two conserved quantities very well. The first-order split-step Fourier method converges linearly in time. The convergence rates in time for the second-order and fourth-order split-step Fourier methods are second-order and fourth-order, respectively, although we cannot guarantee the second- and fourth-order convergence rate in time for these methods in theory. Furthermore, the computational cost of the second-order scheme is 1.25 times of the first-order scheme, whereas the computational cost of the fourth-order scheme is about 3 times of the second-order scheme.

To show the convergence rate in space for these schemes, we perform numerical experiments for different values of N and a fixed value of time step $\Delta t = 0.0094$ to keep the temporal errors small. The results are shown in Tables (13-15). From the results, it is clear that the fourth-order split-step Fourier method converges exponentially in space. Similar claims are also valid for the first-order and second-order schemes.

Table 10: Convergence rates in time for the first-order splitting method ($N = 512,\ -20 \le x \le 80,\ 0 \le t \le 60,\ T = 60$).

Δt	L_∞	L_2	e_1	e_2	CPU(sec)
0.3000	1.48E-02	1.56E-02	2.06E-07	3.38E-07	1.25
0.1500	4.89E-03	7.46E-03	3.77E-09	9.45E-09	2.44
0.0750	2.47E-03	3.69E-03	4.11E-11	4.34E-10	4.80
0.0375	1.24E-03	1.84E-03	1.01E-11	3.27E-12	9.22
0.0187	6.19E-04	9.17E-04	9.28E-11	9.10E-11	16.03
0.0094	3.10E-04	4.58E-04	1.04E-10	1.06E-10	28.84

Table 11: Convergence rates in time for the second-order splitting method ($N = 512,\ -20 \le x \le 80,\ 0 \le t \le 60,\ T = 60$).

Δt	L_∞	L_2	e_1	e_2	CPU(sec)
0.3000	1.11E-02	4.43E-03	1.89E-07	2.98E-07	1.61
0.1500	2.86E-03	9.68E-04	4.94E-10	7.05E-09	3.09
0.0750	7.27E-04	2.43E-04	7.74E-11	5.29E-10	5.78
0.0375	1.82E-04	6.04E-05	3.15E-11	4.04E-12	11.06
0.0187	4.56E-05	1.50E-05	8.50E-11	8.39E-11	20.51
0.0094	1.14E-05	3.75E-06	9.96E-11	1.02E-10	37.70

Table 12: Convergence rates in time for the fourth-order splitting method

($N = 512,\ -20 \le x \le 80,\ 0 \le t \le 60,\ T = 60$).

Δt	L_∞	L_2	e_1	e_2	CPU(sec)
0.3000	8.21E-03	4.78E-03	2.63E-06	2.77E-06	4.73
0.1500	2.52E-04	1.61E-04	6.87E-09	3.64E-08	8.85
0.0750	2.02E-05	8.97E-06	3.85E-10	1.20E-09	17.00
0.0375	1.53E-06	5.86E-07	4.74E-11	3.25E-11	33.17
0.0187	1.01E-07	3.69E-08	1.02E-10	1.05E-10	61.21
0.0094	6.31E-09	2.38E-09	1.04E-10	1.08E-10	114.85

Table 13: Convergence rates in space for the first-order splitting method

($\Delta t = 0.0094,\ -20 \le x \le 80,\ 0 \le t \le 60,\ T = 60$).

N	L_∞	L_2	e_1	e_2	CPU(sec)
128	7.17E-01	3.64E-01	2.73E-02	4.70E-03	6.20
160	9.42E-02	3.90E-02	2.61E-03	4.63E-03	8.20
192	3.46E-03	1.94E-03	4.93E-04	5.91E-04	9.19
224	4.21E-04	5.41E-04	2.04E-04	1.98E-04	11.42
256	3.28E-04	4.59E-04	4.72E-05	4.66E-05	12.52
384	3.10E-04	4.58E-04	1.21E-07	1.22E-07	19.91
512	3.10E-04	4.58E-04	1.04E-10	1.06E-10	28.84

Table 14: Convergence rates in space for the second-order splitting method

($\Delta t = 0.0094,\ -20 \leq x \leq 80,\ 0 \leq t \leq 60,\ T = 60$).

N	L_∞	L_2	e_1	e_2	CPU(sec)
128	7.17E-01	3.64E-01	2.73E-02	4.73E-03	8.62
160	9.42E-02	3.90E-02	2.62E-03	4.62E-03	10.88
192	3.60E-03	1.88E-03	4.89E-04	5.84E-04	12.41
224	2.45E-04	2.84E-04	2.04E-04	1.99E-04	14.96
256	2.98E-05	3.94E-05	4.67E-05	4.66E-05	16.82
384	1.15E-05	3.75E-06	1.21E-07	1.22E-07	26.68
512	1.14E-05	3.75E-06	9.96E-11	1.02E-10	37.70

Table 15: Convergence rates in space for the fourth-order splitting method

($\Delta t = 0.0094,\ -20 \leq x \leq 80,\ 0 \leq t \leq 60,\ T = 60$).

N	L_∞	L_2	e_1	e_2	CPU(sec)
128	7.17E-01	3.64E-01	2.74E-02	4.77E-03	24.91
160	9.42E-02	3.90E-02	2.61E-03	4.62E-03	31.80
192	3.60E-03	1.89E-03	4.88E-04	5.85E-04	36.27
224	2.40E-04	2.85E-04	2.05E-04	2.00E-04	44.00
256	2.54E-05	3.93E-05	4.68E-05	4.67E-05	50.95
320	3.06E-07	3.34E-07	3.21E-06	3.07E-06	62.84
384	1.00E-08	4.52E-09	1.23E-07	1.24E-07	78.92
512	6.31E-09	2.38E-09	1.04E-10	1.08E-10	114.85

Interaction of two solitary waves

Here we study the coupled nonlinear Schrödinger equation (37) with the initial condition

$$\psi_1(x,0) = \sum_{k=1}^{2} \sqrt{\frac{2\alpha_k}{1+\mu}} \operatorname{sech}[\sqrt{2\alpha_k}\, x_k] \exp\{i(v_k - \delta)x_k\},$$
$$\psi_2(x,0) = \sum_{k=1}^{2} \sqrt{\frac{2\alpha_k}{1+\mu}} \operatorname{sech}[\sqrt{2\alpha_k}\, x_k] \exp\{i(v_k + \delta)x_k\},$$
(61)

where $k = 1, 2$, $x_1 = x$, $x_2 = x - 25$, $\alpha_1 = 1.0$, $\alpha_2 = 0.5$, $v_1 = 1.0$, $v_2 = 0.1$ and $\mu = 2/3$. The initial condition represents two solitary waves separated by a distance of 25 units. At $t = 0$, the faster wave is located at $x = 0$ moving to the right with speed 1 and the slower one located at $x = 25$ moving to the right with speed 0.1.

The problem is solved in the interval $-20 \le x \le 80$ for $0 \le t \le 50$ using first-order split-step Fourier methods. For this problem the exact values of the conserved quantities are

$$E_1 = E_2 = \frac{2}{1+\mu} \sum_{k=1}^{2} \sqrt{2\alpha_k}.$$

Since an analytical solution is not available for the problem, we cannot preset the L_∞-errors and L^2-errors. We present in Table 16 the two conserved quantities. These results are obtained for $N = 512$ and $\Delta t = 0.025$. It is clear that both quantities remain constant with respect to time t. It provides a valuable check on the correctness of the numerical results.

Table 16: Conserved quantities for two solitary wave interaction.

Time	$\delta = 0.2$		$\delta = 0.5$	
	E_1	E_2	E_1	E_2
0	2.897056	2.897056	2.897056	2.897056
10	2.897056	2.897056	2.897056	2.897056
20	2.897057	2.897057	2.897057	2.897057
30	2.897056	2.897056	2.897057	2.897057
40	2.897055	2.897056	2.897056	2.897056
50	2.897056	2.897056	2.897056	2.897056

Figures 8 and 9 show the interactions of two solitary waves for $\delta = 0.2$ and $\delta = 0.5$, respectively. We can see, from Figure 8, that the interaction are inelastic when $\delta = 0.2$. Since the taller one moves faster than the shorter one, it catches up and collides with the shorter one, then moves away. Note that after the collision, the amplitude of the taller one becomes larger while the amplitude of the shorter one becomes smaller. For the case of $\delta = 0.5$ as shown in Figure 9, the interaction is elastic. The two solitary waves undergo elastic collision and then restore their original shape. These results are the same as the results reported by Ismail and Taha.

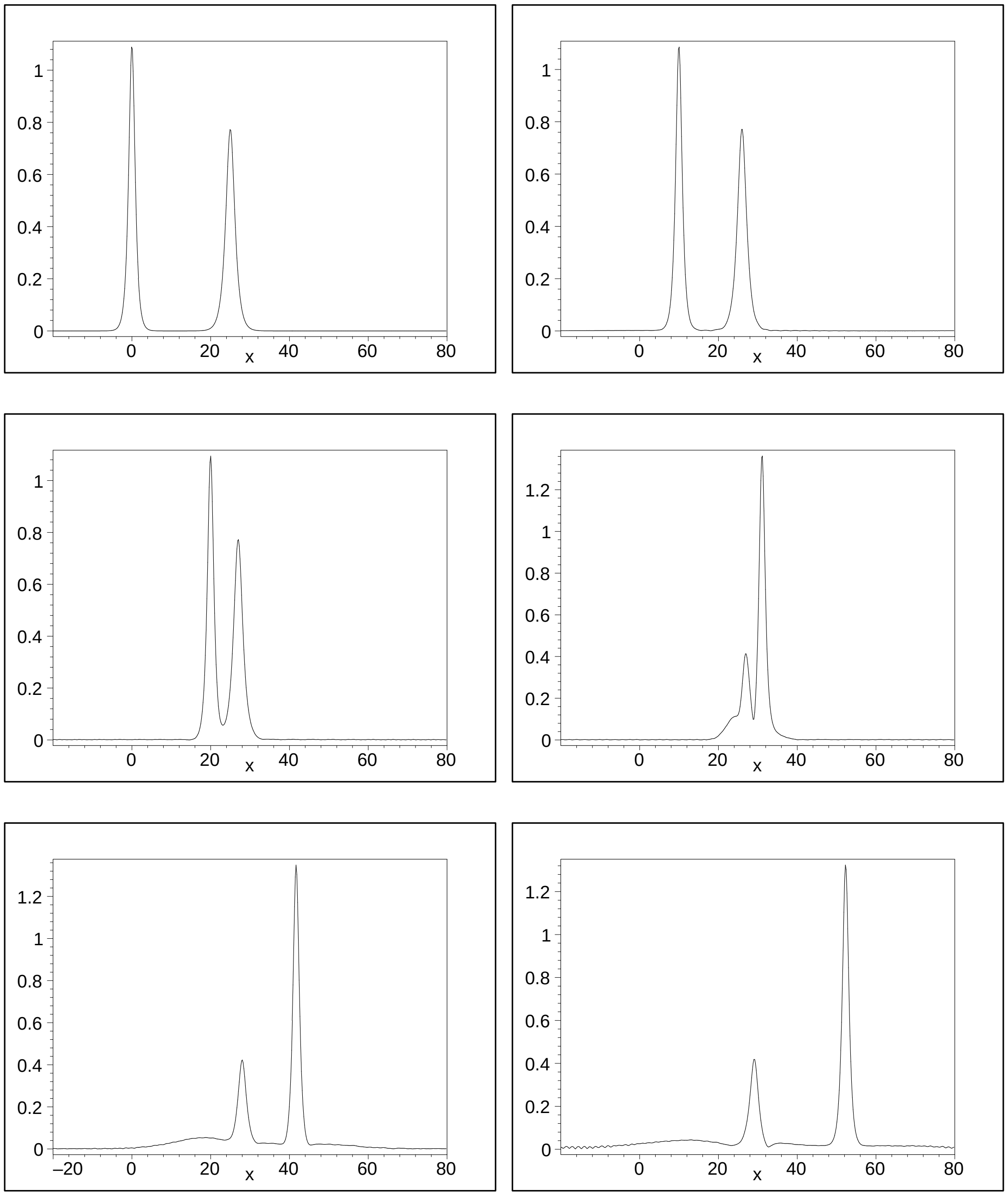

Fig.8: Interaction of two solitary waves for time t from 0 to 50 ($\delta = 0.2$). The taller one moves to the right with speed 1, the shorter one moves to the right with speed 0.1. Only the values of $|\psi_1|$ are shown above.

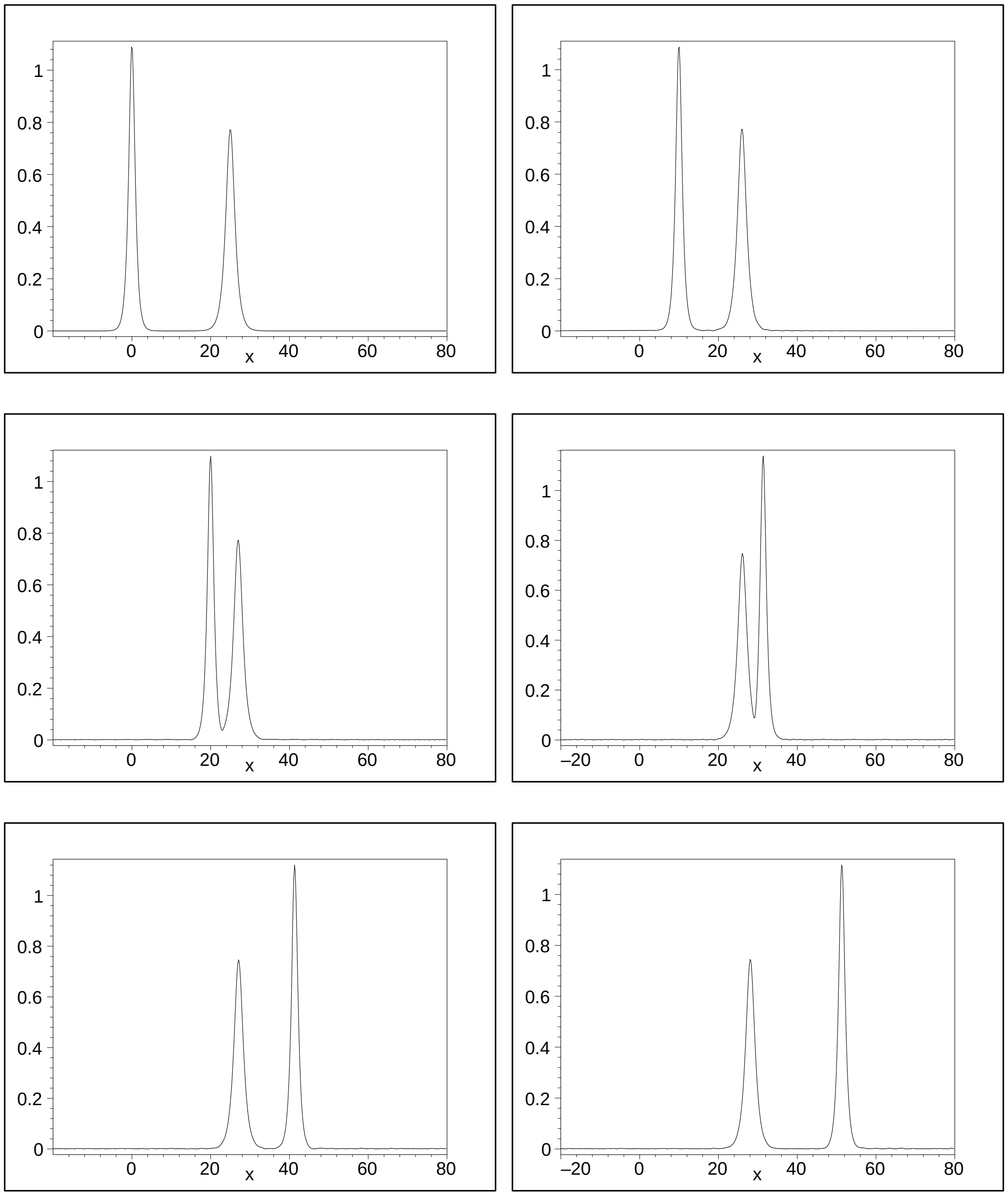

Fig.9: Interaction of two solitary wave for time t from 0 to 50 ($\delta = 0.5$). The taller one moves to the right with speed 1, the shorter one moves to the right with speed 0.1. Only the values of $|\psi_1|$ are shown above.

Parallel split-step Fourier methods

Parallel algorithms of the split-step Fourier methods are implemented on the Origin 2000 multiprocessor computer. The system consists 24×300 MHz MIPS R12000 processors with 4MB cache memory and 8 GB of system memory. The same problem in Section is solved using the parallel algorithms. All timings are the total wall-clock time for execution of the code. All the codes are optimized at the same optimization level. The results are shown on Tables(17-19). The speedup S_p is defined by

$$S_p = \frac{\text{Time spent to run the MPI code on one processor}}{\text{Time spent to run the MPI code on } p \text{ processors}}.$$

From the results, it is clear that the speedup increases as the problem size N becomes larger for a fixed processor number p. For small problem sizes the computation/communication ratio is small, thus speedup is small. For fixed p, we can also see that the fourth-order scheme has a better speedup than the second-order scheme, whereas the second-order scheme has a slightly better speedup than the first-order scheme. This is due to the fact that the fourth-order scheme is more computational intensive than the second-order scheme, whereas the second-order scheme is more computational intensive than the first-order scheme.

Table 17: Results for parallel implementation of first-order split-step Fourier method (Δt = 0.015). N indicates array size, NS the number of steps, t_p the time on p processors, S_p the speedup on p processors.

	N=2^{11} NS=4000	N=2^{13} NS=1000	N=2^{15} NS=250	N=2^{17} NS = 62
t_1(sec)	20.0	23.2	23.7	31.7
t_2(sec)	18.9	19.8	18.5	23.0
t_4(sec)	14.1	12.3	11.7	14.3
t_8(sec)	10.5	8.9	6.9	8.4
S_2=t_1/t_2	1.1	1.2	1.3	1.4
S_4=t_1/t_4	1.4	1.9	2.0	2.2
S_8=t_1/t_8	1.9	2.6	3.4	3.6

Table 18: Results for parallel implementation of second-order split-step Fourier method (Δt = 0.015). N indicates array size, NS the number of steps, t_p the time on p processors, S_p the speedup on p processors.

	N=2^{11} NS=4000	N=2^{13} NS=1000	N=2^{15} NS=250	N=2^{17} NS = 62
t_1(sec)	21.3	24.8	25.4	34.8
t_2(sec)	19.5	20.1	19.8	25.5
t_4(sec)	14.4	12.7	12.3	15.3
t_8(sec)	10.7	8.7	6.8	9.2
S_2=t_1/t_2	1.1	1.2	1.3	1.4
S_4=t_1/t_4	1.5	2.0	2.1	2.3
S_8=t_1/t_8	2.0	2.9	3.5	3.8

Table 19: Results for parallel implementation of fourth-order split-step Fourier method (Δt = 0.015). N indicates array size, NS the number of steps, t_p the time on p processors, S_p the speedup on p processors.

	N=2^{11}
	NS=4000
		N=2^{13}
		NS=1000
			N=2^{15}
			NS=250
				N=2^{17}
				NS = 62

t_1(sec)	79.9	88.5	90.5	113.3
t_2(sec)	68.7	67.1	67.2	78.0
t_4(sec)	47.9	41.7	41.6	48.5
t_8(sec)	33.3	28.8	24.3	27.1
S_2=t_1/t_2	1.2	1.3	1.3	1.5
S_4=t_1/t_4	1.7	2.1	2.2	2.3
S_8=t_1/t_8	2.4	3.1	3.7	4.2

Conclusions

In this study, we apply the well-known split-step Fourier method for solving nonlinear Schrödinger equation to the coupled nonlinear Schrödinger equation. We present three split-step schemes for solving the coupled nonlinear Schrödinger equation. The numerical solutions obtained using these schemes agree with the exact solutions for one solitary wave case. All of the three schemes converge exponentially in space and converge at least linearly in time. The numerical results show that the convergence rate in time of the fourth-order split-step Fourier method is fourth-order, although it is not guaranteed to be true in theory. However, the higher-order split-step scheme needs more computational time than the lower-order one. Moreover, the collision of two solitary waves with different amplitudes is investigated numerically. The pictures of such interaction are displayed.

For the parallel implementation of each of the three algorithms with fixed p, the speedup increases as the problem size N becomes larger, where N is the total number of spatial mesh points. For large N, the speedups achieved on the multiprocessor computer running the parallel codes are considerable.