AN ABSTRACT OF THE THESIS OF

Peter W. White for the degree of Doctor of Philosophy in Mathematics presented on June 29, 1994.

Title: The Davey-Stewartson Equations: A Numerical Study

Redacted for Privacy

Abstract Approved: _____
Andre Weideman

In 1974 Davey and Stewartson used a multi-scale analysis to derive a coupled system of nonlinear partial differential equations which describes the evolution of a three dimensional wave packet in water of a finite depth. This system of equations is the closest integrable two dimensional analog of the well-known one dimensional nonlinear Schrödinger equation. The method of inverse scattering can thus be used to solve the Davey-Stewartson equations in theory, but in practice this method is not feasible with arbitrary initial conditions. In this thesis we present a numerical method for solving the Davey-Stewartson equations. It is an extension of the split-step Fourier method that proved to be so successful for the nonlinear Schrödinger equation. This method is tested on some known soliton and dromion solutions and then used to study modulational instability and solutions which become singular in finite time.

The Davey-Stewartson Equations: A Numerical Study

by

Peter W. White

A THESIS

submitted to

Oregon State University

in partial fulfillment of
the requirements for the
degree of

Doctor of Philosophy

Completed June 29, 1994
Commencement June 1995

APPROVED:

Redacted for Privacy

Professor of Mathematics in charge of major

Redacted for Privacy

Head of Department of Mathematics

Redacted for Privacy

Dean of Graduate School

Date thesis is presented June 29, 1994

Typed by Peter W. White (typeset by LaTeX) for Peter W. White

Table of Contents

List of Figures

Chapter 1

Introduction

1.1 Background

One of the major successes in the field of applied mathematics in the last decade or two is the method of inverse scattering, associated with the names of Ablowitz, Gardner, Greene, Kaup, Kruskal, Lax, Miura, Newell, Segur, Zakharov, and others; see [1] and [5] for historical reviews. Originally developed in the context of the Korteweg-de Vries equation

$$\text{KdV:} \qquad u_t + 6uu_{xx} + u_{xxx} = 0,$$

the method of inverse scattering allows one to construct solutions by linking the PDE to a certain linear scattering problem, thereby enabling one to solve the *nonlinear* PDE with essentially *linear* techniques.

After the initial success of the method of inverse scattering on the KdV, it was soon discovered that the method had wider applicability. For example, the nonlinear Schrödinger and Sine-Gordon equations were both shown to be solvable by the method of inverse scattering. These equations are respectively defined by

$$\text{NLS:} \qquad iu_t + u_{xx} + 2|u|^2u = 0,$$

and

$$\text{SG:} \qquad u_{tt} - u_{xx} + \sin u = 0.$$

In these equations the boundary conditions are usually taken to be periodic on a finite domain, or decay to zero at infinity on an infinite domain. The KdV and NLS are both models of nonlinear water waves in one dimension while the SG equation models a coupled pendulum chain. In two space dimensions the Kadomtsev-Petviashvili and the Davey-Stewartson I and II equations, given by

$$\text{KP:} \qquad (u_t + 6uu_x + u_{xxx})_x + 3\sigma^2 u_{yy} = 0,$$

and

$$\text{DSI and DSII:} \qquad \begin{aligned} iu_t + \frac{1}{2}\sigma^2(u_{xx} + \sigma^2 u_{yy}) &= -\alpha|u|^2 u + u\phi_x \\ \phi_{xx} - \sigma^2\phi_{yy} &= 2\alpha\left(|u|^2\right)_x, \end{aligned} \tag{1.1}$$

where these equations have been scaled so that $\sigma^2 = \pm 1$ and $\alpha = \pm 1$, are also known to be solvable by the method of inverse scattering. (The case $\sigma^2 = 1$ corresponds to the DSI system and $\sigma^2 = -1$ corresponds to the DSII system). Note that if the x-dependence is removed in the DS system, then the NLS equation is recovered.

These equations have several common properties which include an infinite number of conserved quantities and the occurrence of soliton solutions[1]. Perhaps the most fundamental common feature is that these equations may all be expressed as Hamiltonian systems of infinite dimension, which are moreover *integrable* thanks to the infinite number of conserved quantities. The method of inverse scattering was realized to be the infinite dimensional analogue of the action-angle transformation by which finite dimensional Hamiltonian systems are commonly solved [13].

Whereas the method of inverse scattering is very general and powerful, it is not easy to apply when arbitrary initial conditions are specified. This is especially

[1] A soliton is a traveling wave which becomes localized in finite time and then maintains its shape indefinitely, except for brief moments during nonlinear interaction.

true in the case of two space dimensions such as the DSI, DSII and KP equations. For this reason accurate numerical simulation remains an indispensable tool in the study of these equations. Whereas the one-dimensional problems such as NLS, KdV, and SG have received considerable attention in the numerical analysis literature, see for example [17], [27], and [31], the same cannot be said for the two-dimensional problems such as the KP and DS equations. The only references known to this author are [22] which involves a non-integrable case of the DS system

$$\begin{aligned} iu_t + u_{xx} + u_{yy} + |u|^2 u - \phi_x u = 0, \\ \phi_{xx} + \phi_{yy} = -\left(|u|^2\right)_x, \end{aligned} \tag{1.2}$$

and [29] where we have presented what we believe to be the first known simulations of soliton and dromion solutions of the DS system. This absence of reported results is in some part due to the computational complexity of these problems: not only are they nonlinear, but they are typically posed on a doubly infinite domain, and in the case of the DS equation there is the added complication of dealing with a coupled system of equations.

In this thesis we introduce and analyze a numerical method for the DS systems (1.1) and (1.2). This method is based on an extension of the well-known and successful split-step Fourier method originally proposed for the KdV equation by Tappert [27], and also used in [28] on the NLS equation. The extension of the split-step method from the NLS to the DS system is however not trivial. We must address the following problems: (i) inverting the second equation in (1.1) to solve for ϕ_x in both the elliptic and hyperbolic cases ($\sigma^2 = 1$ and $\sigma^2 = -1$ resp.), (ii) incorporating boundary conditions, (iii) truncating from an infinite to a finite domain, and (iv) optimizing the method to deal with the large number of computational grid points needed.

Testing the quality of schemes such as the ones introduced here can be achieved in one of two ways: first, by rigorous convergence proofs and error estimates, or

second, by ensuring that the relevant physical properties in the continuous equations are represented accurately by the numerical method. We will follow the second approach here. Classical convergence proofs and error estimates, which typically do not take the particular nonlinear dynamical features of these equations into account, have had limited success in predicting how good or bad a particular method works *in practice*. A reason for this is that such convergence proofs are designed to predict what happens in the limit as the mesh-size, both in space and time, shrinks to zero. A computer works with finite values of the mesh-size however, and nonlinear features such as blow-up and numerical chaos may be observed which cannot be accounted for by classical analysis. Perhaps the best example is presented in the paper by Ablowitz and Herbst [2], where the following two discretizations of the NLS are compared

$$i\frac{d}{dt}U_j + \frac{1}{h^2}\left(U_{j-1} - 2U_j + U_{j+1}\right) + 2|U_j|^2 U_j = 0, \tag{1.3}$$

and

$$i\frac{d}{dt}U_j + \frac{1}{h^2}\left(U_{j-1} - 2U_j + U_{j+1}\right) + |U_j|^2\left(U_{j-1} + U_{j+1}\right) = 0. \tag{1.4}$$

The numerical results of Ablowitz and Herbst show that when the nonlinear phenomenon of recurrence (defined below) is simulated, scheme (1.3) quickly collapses into chaos (see Figure 1), whereas scheme (1.4) maintains a nice and smooth periodic solution in a long-time integration (see Figure 2). The crucial difference between the two schemes is that (1.4) has been designed to be integrable, which is not the case for (1.3). Whereas both schemes are formally accurate to order $O(h^2)$, only one of them reflects the dynamical property of integrability and is therefore superior.

In the above simulations a standard Runge-Kutta method was employed. It should be pointed out that the use of a finer space grid in (1.3) will produce recurrence for a period of time, but at some later time numerical chaos will occur.

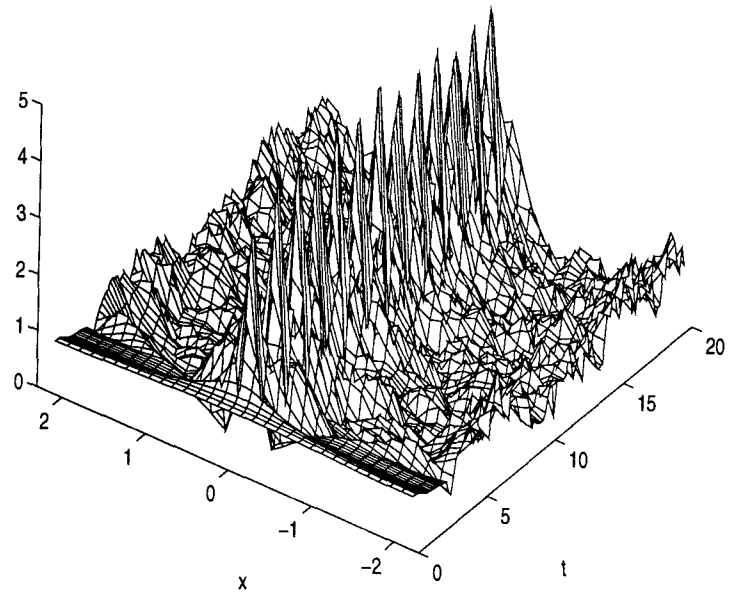

Figure 1. Discretization (1.3)

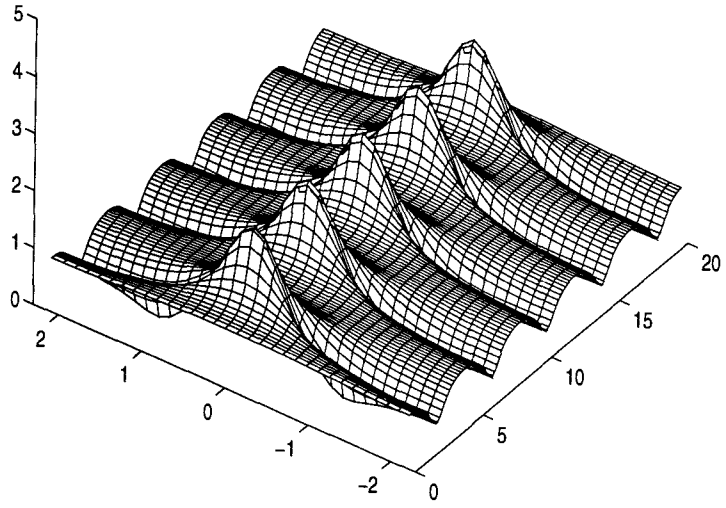

Figure 2. Discretization (1.4)

Instead of pursuing classical convergence proofs, we will therefore attempt to ensure that the relevant dynamical properties of the DS system are accurately reflected in the numerical approximations. To this end we will show that our semi-discrete approximation is a finite dimensional Hamiltonian system which approximates the infinite dimensional Hamiltonian of the continuous problem. We will also show that the split-step method that we use to integrate our system is a symplectic transformation, a feature that is increasingly recognized to be of paramount importance in the integration of Hamiltonian systems.

To set the background it is perhaps best to review the relevant issues as they pertain to the NLS equation.

1.2 Review of NLS

The NLS equation is given by

$$iu_t + u_{xx} + 2|u|^2 u = 0, \tag{1.5}$$

where $u(x,t) \in \mathbb{C}$ and $i = \sqrt{-1}$. Two of the conserved quantities of (1.5) are the L^2-norm

$$\frac{d}{dt} I = 0, \qquad \text{where} \qquad I = \int |u|^2 \, dx, \tag{1.6}$$

and the Hamiltonian

$$\frac{d}{dt} H = 0, \qquad \text{where} \qquad H = -i \int \left(|u_x|^2 - |u|^4 \right) \, dx. \tag{1.7}$$

Here the integration is over one period in the case of periodic boundary conditions

$$u(x,t) = u(x+P,t), \qquad \text{for all } t \geq 0,$$

or over all of $\mathbb{R}$ in the case of decaying boundary conditions

$$u(x,t) \to 0, \quad u_x(x,t) \to 0 \quad \text{as} \quad |x| \to \infty.$$

A proof of (1.6) and (1.7) will follow from the analysis in the next chapter, where we derive the analogous conserved quantities of the DS system. The reason for referring to H as the Hamiltonian is as follows. The NLS (1.5), and its conjugate

$$u_t = iu_{xx} + 2i|u|^2u, \quad \text{and} \quad u_t^* = -iu_{xx}^* - 2i|u|^2u^*,$$

may be expressed as

$$u_t = \frac{\delta H}{\delta u^*}, \qquad u_t^* = -\frac{\delta H}{\delta u}, \tag{1.8}$$

where H is written as

$$H(u, u^*) = -i \int \left(u_x u_x^* - u^2 (u^*)^2 \right) dx.$$

The variational derivatives on the right-hand side of equation (1.8) are defined so that

$$\lim_{\epsilon \to 0} \frac{H(u + \epsilon \tilde{u}, u^*) - H(u, u^*)}{\epsilon} = \int \frac{\delta H}{\delta u} \, \tilde{u} \, dx,$$

for all suitably smooth functions $\tilde{u}$ which decay to zero at infinity.

To see that H is the Hamiltonian, note that

$$\begin{aligned} \frac{1}{\epsilon}(H(u + \epsilon\tilde{u}, u^*) \; - \; & H(u, u^*)) \\ &= -\frac{i}{\epsilon} \int \left\{ (u + \epsilon\tilde{u})_x u_x^* - (u + \epsilon\tilde{u})^2 (u^*)^2 + |u_x|^2 - |u|^4 \right\} dx \\ &= -\frac{i}{\epsilon} \int \left\{ \epsilon \tilde{u}_x u_x^* - 2\epsilon\tilde{u} u (u^*)^2 - \epsilon^2 \tilde{u}^2 (u^*)^2 \right\} dx. \end{aligned}$$

Now use integration by parts on the first term in the integral and cancel a factor of ϵ to get

$$\frac{1}{\epsilon} \left(H(u + \epsilon\tilde{u}, u^*) - H(u, u^*) \right) = i \int \left\{ (u_{xx}^* + 2|u|^2 u^*) \tilde{u} + \epsilon \tilde{u}^2 (u^*)^2 \right\}.$$

So as $\epsilon \to 0$ we get

$$\frac{\delta H}{\delta u} = iu_{xx}^* + 2i|u|^2u^*,$$

and then

$$u_t^* = -iu_{xx}^* - 2i|u|^2u^* = -\frac{\delta H}{\delta u}.$$

The other equation in (1.8) follows in a similar manner.

Two nonlinear phenomena that will be of particular interest to this thesis are those of solitons and recurrence. As for solitons, these solutions may be obtained by the method of inverse scattering, see [1] for example. An example of a 2-soliton solution to the NLS equation can be seen in Figure 3. The two solitons collide and then emerge without a change of shape. In Figure 4 a contour plot corresponding to Figure 3 shows how the nonlinear interaction of the solitons causes a phase shift in the paths taken by the solitons during the collision.

The other interesting nonlinear phenomenon, that of recurrence, is related to the modulational instability described by the NLS. First note that

$$u(x,t) = ae^{2i|a|^2 t}$$

is a solution to (1.5). Consider a perturbation to this solution given by

$$u(x,t) = ae^{2i|a|^2 t}\left(1 + \varepsilon(x,t)\right),$$

where $\varepsilon(x,t) \in \mathbb{C}$ and $|\varepsilon(x,0)| \ll 1$. Substitute this perturbed solution into (1.5) and keep only first order terms in ε to get

$$\varepsilon_t = i\varepsilon_{xx} + 2i|a|^2\left(\varepsilon + \varepsilon^*\right). \tag{1.9}$$

Now assume that the perturbation, ε, is periodic on the interval $[-\frac{1}{2}P, \frac{1}{2}P]$ and write

$$\varepsilon = \sum_{m=-\infty}^{\infty} \widehat{\varepsilon}_m(t) e^{i\mu_m x},$$

where $\mu_m = \frac{2\pi m}{P}$. Substituting into (1.9) gives a system of ordinary differential equations

$$\frac{d}{dt}\begin{pmatrix} \widehat{\varepsilon}_m \\ \widehat{\varepsilon}^*_{-m} \end{pmatrix} = G_m \begin{pmatrix} \widehat{\varepsilon}_m \\ \widehat{\varepsilon}^*_{-m} \end{pmatrix}, \qquad m \neq 0$$

where

$$G_m = i\begin{pmatrix} 2|a|^2 - \mu_m^2 & 2|a|^2 \\ -2|a|^2 & -2|a|^2 + \mu_m^2 \end{pmatrix}.$$

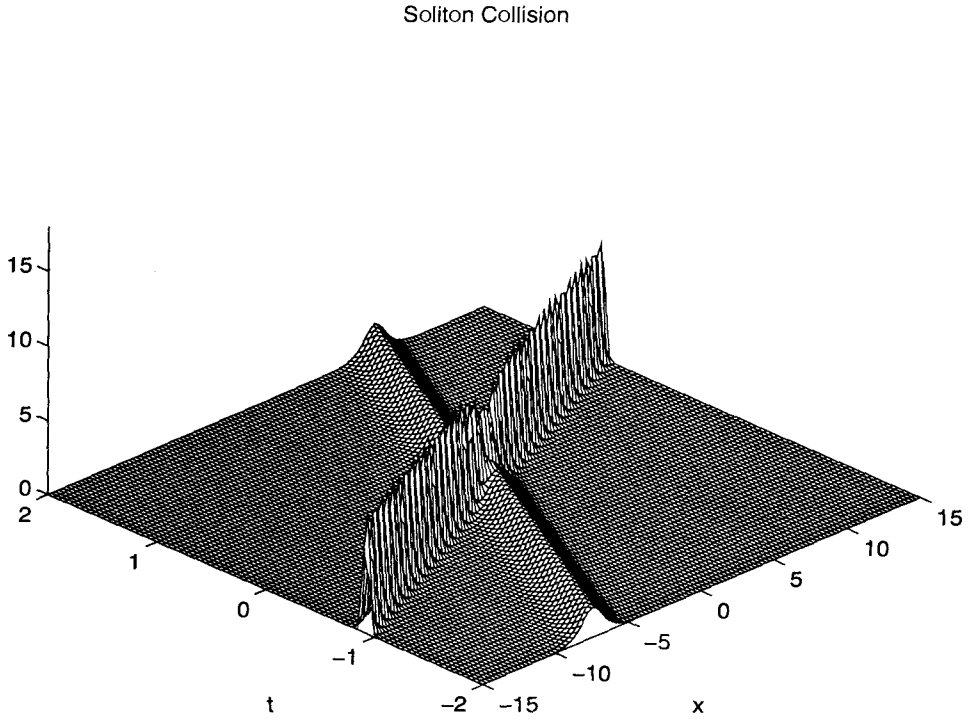

Figure 3. Soliton interaction in the NLS equation

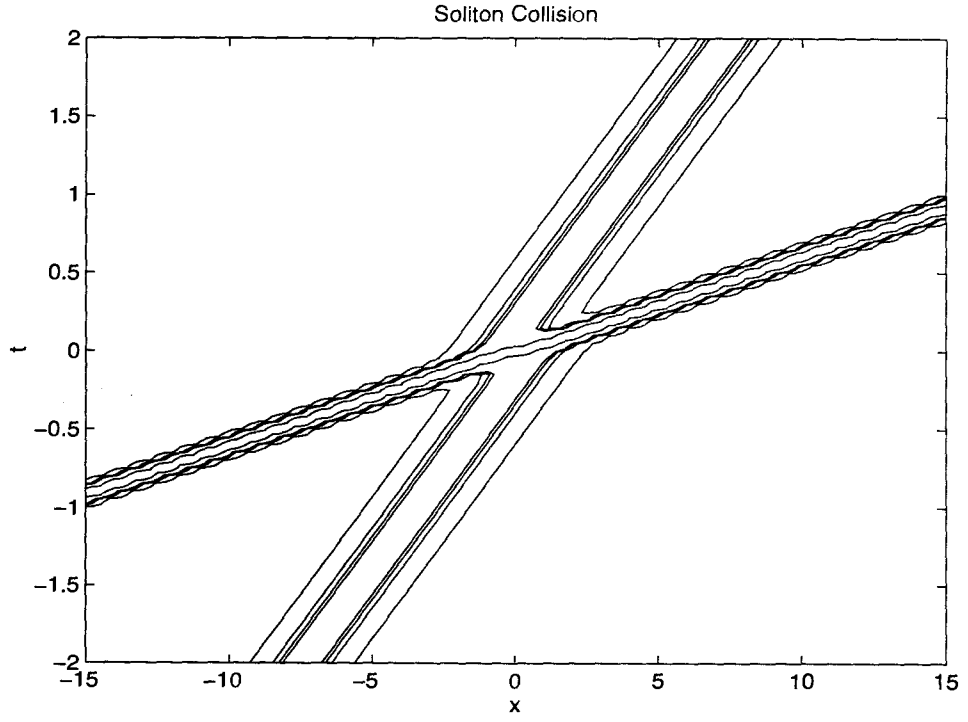

Figure 4. Contour plot of soliton interaction in the NLS equation

The eigenvalues of the growth matrix G_m are given by $\pm\mu_m\sqrt{4|a|^2-\mu_m^2}$. One of the eigenvalues is a positive real number if

$$0 < \mu_m^2 < 4|a|^2, \tag{1.10}$$

which is a region of instability associated with the names of Benjamin and Feir; see [8] and also the discussion in Section 5.1. Whenever μ_m is inside this region the mode $\left(\widehat{\varepsilon}_m, \widehat{\varepsilon}^*_{-m}\right)^T$ will grow exponentially. However, this exponential growth in the Fourier modes of the perturbation does not continue indefinitely. In [30] Yuen and Ferguson show that the long-term evolution displays the recurrence phenomenon: the unstable modes take turns in dominating the solution, with intermittent returns to the almost uniform state. Figure 5 shows an example where only one mode of the perturbation lies in the instability region (1.10). This corresponds to $P = \sqrt{2}\pi$, $a = 1$, and

$$u(x,0) = 1 + 0.1\cos(\sqrt{2}x). \tag{1.11}$$

Figure 6 shows the evolution of the primary and the unstable modes of the solution. An example where two modes dominate may be seen later in Figure 7.

These pictures were produced by the split-step Fourier scheme. In this thesis we will extend the split-step scheme to the DS system of equations. We will then use this scheme to simulate solitons and modulational instability in the DS system.

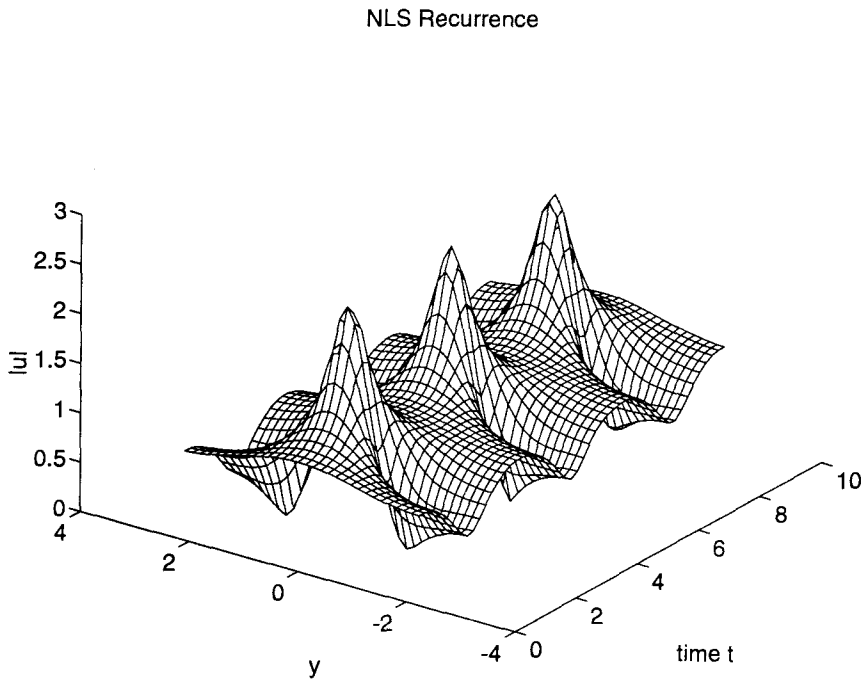

Figure 5. Recurrence in the NLS equation.

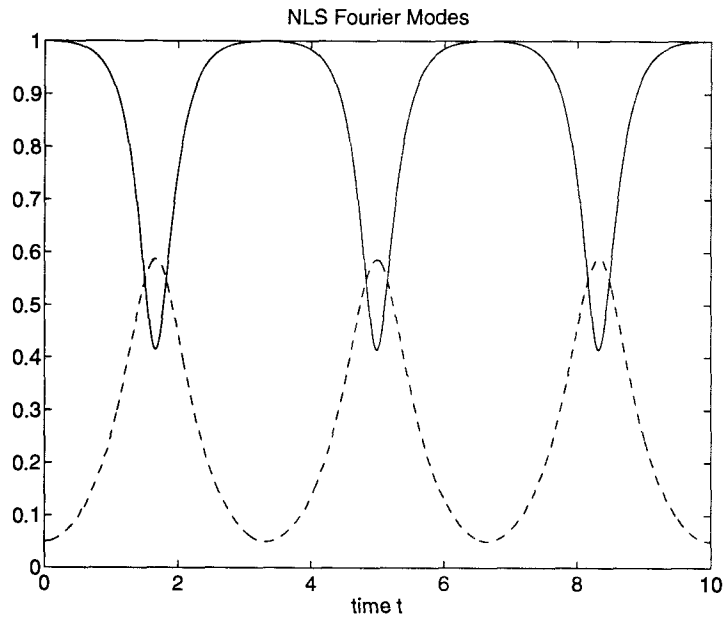

Figure 6. Primary mode: solid line and unstable mode: dashed line, in the NLS corresponding to one unstable mode.

Chapter 2

Analytic Aspects

In this chapter we review some of the literature related to the physical aspects of the DS system. The main sources for this review are [3], [12] and [15]. Although no original results will be produced, this chapter will establish the DS system in the context of water waves and introduce properties of the DS system which will play an important role in establishing the validity of the numerical scheme used to solve the DS system.

2.1 Derivation of DS System

In this section we reproduce some of the analysis found in [12] which results in a system of equations of the form

$$iu_t + \alpha_1 u_{xx} + \alpha_2 u_{yy} + \alpha_3 |u|^2 u + \alpha_4 u\phi_x = 0 \tag{2.1}$$

$$\phi_{xx} + \beta_1 \phi_{yy} = \beta_2 (|u|^2)_x \,, \tag{2.2}$$

where $\alpha_1, \ldots, \alpha_4$, β_1, $\beta_2 \in \mathbb{R}$, $u(x,y,t) \in \mathbb{C}$ and $\phi(x,y,t) \in \mathbb{R}$. We will refer to these partial differential equations as the DS system.

Davey and Stewartson considered a cartesian coordinate system with the xy-plane coinciding with the undisturbed free surface of a body of water [12]. The positive z-axis is directed so that the bed of the water is at $z = -h$. At $t = 0$

it is assumed that a progressive wave is established such that the free surface is raised to $z = \zeta$, where

$$g\zeta(x, y, 0) = i\varepsilon\omega\, a(\varepsilon x, \varepsilon y) \exp(ikx) + \text{c.c.}.$$

Here g is the acceleration due to gravity, k is the wave number and ω is the frequency of the progressive wave, a is a given function, ε is a small positive constant and c.c. indicates the complex conjugate of the preceding term. A progressive wave such as this corresponds to a wave of wavelength $2\pi/k$ traveling in the positive x direction with an amplitude slowly varying with position and on a scale inversely proportional to its height. We will assume the linear dispersion relation for water waves, $\omega = \sqrt{gk\sigma}$ where $\sigma = \tanh(kh)$, holds.

Let $\phi(x, y, z, t)$ denoted the velocity potential of an incompressible, irrotational, inviscid fluid. The incompressibility condition implies that ϕ satisfies the Poisson equation

$$\phi_{xx} + \phi_{yy} + \phi_{zz} = 0 \quad \text{in} \quad -h < z < \zeta\,. \tag{2.3}$$

The corresponding boundary conditions are an impermeable bottom, so

$$\phi_z = 0, \quad \text{when} \quad z = -h, \tag{2.4}$$

and two conditions at the free surface, the kinematic boundary condition

$$\phi_z = \zeta_t + \phi_x\zeta_x + \phi_y\zeta_y, \quad \text{when} \quad z = \zeta, \tag{2.5}$$

and

$$2g\zeta + 2\phi_t + \phi_x^2 + \phi_y^2 + \phi_z^2 = 0, \quad \text{when} \quad z = \zeta, \tag{2.6}$$

which is a statement of uniform pressure at the surface, which follows from Bernoulli's law. For the moment we have neglected the effects of surface tension.

Since the disturbance is a progressive wave, we can look for a solution of (2.3)–(2.6) of the form

$$\phi = \sum_{n=-\infty}^{\infty} \phi_n E^n, \quad \zeta = \sum_{n=-\infty}^{\infty} \zeta_n E^n, \tag{2.7}$$

where $E = \exp\{i(kx - \omega t)\}$, $\phi_{-n} = \phi_n^*$, $\zeta_{-n} = \zeta_n^*$ and * denotes the complex conjugate. Also write

$$\phi_n = \sum_{j=n}^{\infty} \varepsilon^j \phi_{nj}, \quad \zeta_n = \sum_{j=n}^{\infty} \varepsilon^j \zeta_{nj}, \quad (n \geq 0), \tag{2.8}$$

where ϕ_{nj} is a function of ξ, η, z, and τ only, ζ_{nj} is a function of ξ, η, and τ only and $\phi_{00} = \zeta_{00} = 0$. The variables ξ, η and τ are given by

$$\xi = \varepsilon(x - c_g t), \quad \eta = \varepsilon y, \quad \text{and} \quad \tau = \varepsilon^2 t,$$

where c_g is the group velocity of the primary progressive wave given by

$$c_g = \omega'(k) = \frac{g}{2\omega}\left\{\sigma + kh(1 - \sigma^2)\right\}.$$

Substitute (2.7) and (2.8) into (2.3) and use (2.4) to get

$$\phi_{11} = A\frac{\cosh k(z+h)}{\cosh kh},$$

$$\phi_{22} = F\frac{\cosh 2k(z+h)}{\cosh 2kh},$$

and

$$\phi_{12} = D\frac{\cosh k(z+h)}{\cosh kh} - iA_\xi \frac{(z+h)\sinh k(z+h) - h\sigma \cosh k(z+h)}{\cosh kh},$$

where A, D and F are unknown functions of ξ, η, and τ only. ϕ_{01} and ϕ_{02} are independent of z and

$$\frac{\partial \phi_{03}}{\partial z} = -(z+h)\left\{\frac{\partial^2 \phi_{01}}{\partial \xi^2} + \frac{\partial^2 \phi_{01}}{\partial \eta^2}\right\}.$$

Now substitute (2.7) and (2.8) into the boundary conditions (2.5) and (2.6). Equating coefficients of $\varepsilon^j E^n$ gives

$$\varepsilon E^0 : \qquad \zeta_{01} = 0, \tag{2.9}$$

$$\varepsilon E^1 : \qquad g\zeta_{11} = i\omega A, \tag{2.10}$$

$$\varepsilon^2 E^0 : \qquad g\zeta_{02} = c_g \tfrac{\partial \phi_{01}}{\partial \xi} - k^2(1-\sigma^2)|A|^2, \tag{2.11}$$

$$\varepsilon^2 E^1 : \qquad g\zeta_{12} = i\omega D + c_g \tfrac{\partial A}{\partial \xi}, \tag{2.12}$$

$$\varepsilon^2 E^2 : \quad g\zeta_{22} = k^2 A^2 \left(\tfrac{\sigma^3 - 3}{2\sigma^2}\right), \quad \omega F = 3ik^2 A^2 \left(\tfrac{1-\sigma^4}{4\sigma^2}\right). \tag{2.13}$$

The coefficient of $\varepsilon^3 E^0$ in (2.5) involves $\frac{\partial \zeta_{02}}{\partial \xi}$ from the chain rule when $\frac{\partial \zeta}{\partial t}$ is calculated, and a contribution from $\frac{\partial \phi_{03}}{\partial z}$. Using (2.11) to eliminate ζ_{02} gives

$$\left(gh - c_g^2\right) \frac{\partial^2 \phi_{01}}{\partial \xi^2} + gh \frac{\partial^2 \phi_{01}}{\partial \eta^2} = -k^2 \left(2c_p + c_g\left(1-\sigma^2\right)\right) \frac{\partial |A|^2}{\partial \xi}, \tag{2.14}$$

where $c_p = \omega/k$ denotes the phase speed of the primary wave. Equating the coefficients of $\varepsilon^3 E^1$ in (2.5) and (2.6) gives two algebraic equations for ϕ_{13} and ζ_{13}. These equations are compatible only if

$$2i\omega \tfrac{\partial A}{\partial \tau} - \left(c_g^2 - gh\left(1-\sigma^2\right)\left(1 - kh\sigma\right)\right) \tfrac{\partial^2 A}{\partial \xi^2} + c_p c_g \tfrac{\partial^2 A}{\partial \eta^2} =$$
$$\tfrac{1}{2}k^4 \left(9\sigma^{-2} - 12 + 13\sigma^2 - 2\sigma^4\right)|A|^2 A + k^2 \left(2c_p + c_g\left(1-\sigma^2\right)\right) A \tfrac{\partial \phi_{01}}{\partial \xi}. \tag{2.15}$$

Together (2.14) and (2.15) describe the evolution of the progressive wave up to first order in ε, where $A(\xi, \eta, 0) = a(\xi, \eta)$ is given. Note that the form of equations (2.14) and (2.15) is the same as that of (2.1) and (2.2). One set of physically reasonable boundary conditions is that the wave dies away sufficiently far from its center. Thus

$$|A| \to 0, \quad \nabla \phi_{01} \to \vec{0} \quad \text{as} \quad \xi^2 + \eta^2 \to \infty.$$

In the next section we will show that up to first order in ε the nonlinearity in (2.15) only effects the phase of the progressive wave.

The above analysis leading to the DS system is formal and no rigorous justification of it appears to have been given in the literature. In the NLS case however, such justification has been presented by Craig, Sulem and Sulem [11].

In this derivation the coefficient of $(\phi_{01})_{\xi\xi}$ in (2.14) is positive for all wave numbers k and all mean water depths h. The coefficient of $A_{\xi\xi}$ in (2.15) is negative. So the analysis of Davey and Stewartson leads to what is now commonly called the DSII system (after rescaling). In this analysis it was assumed that there was no surface tension. When the effects of surface tension are included, the linearized dispersion relation becomes

$$\omega^2 = (g\kappa + \kappa^3 T)\tanh \kappa h, \qquad \kappa = \sqrt{k^2 + l^2},$$

where $\kappa = (k, l)$ is the horizontal wavenumber characteristic of the disturbance ($l \ll k$) and T is the ratio of the surface tension coefficient to the fluid density. The boundary condition (2.6) now becomes

$$g\zeta + \phi_t + \frac{1}{2}(\phi_x^2 + \phi_y^2 + \phi_z^2) = T\frac{\zeta_{xx}(1+\zeta_y^2) + \zeta_{yy}(1+\zeta_x^2) - 2\zeta_{xy}\zeta_x\zeta_y}{(1+\zeta_x^2+\zeta_y^2)^{\frac{3}{2}}}.$$

The method of multiple scales now leads to the DS equation

$$iA_\tau + \lambda A_{\xi\xi} + \mu A_{\eta\eta} = \chi|A|^2 A + \chi_1 A\Phi_\xi,$$

$$\alpha\Phi_{\xi\xi} + \Phi_{\eta\eta} = -\beta(|A|^2)_\xi,$$

where Φ is the mean motion of the wave packet and

$$\sigma = \tanh kh, \quad \tilde{T} = \frac{k^2 T}{g}, \quad \omega^2 = gk\sigma(1+\tilde{T}) \geq 0,$$

$$\omega_0^2 = g\kappa, \quad \lambda = \frac{\kappa^2\omega_{\kappa\kappa}}{2\omega_0}, \quad \mu = \frac{\kappa^2\omega_{ll}}{2\omega_0} = \frac{kc_g^2}{2\omega_0} \geq 0, \quad c_g = \frac{d\omega}{d\kappa},$$

$$\chi = \frac{\omega_0}{4\omega}\left\{\frac{(1-\sigma^2)(9-\sigma^2) + \tilde{T}(2-\sigma^2)(7-\sigma^2)}{\sigma^2 - \tilde{T}(3-\sigma^2)} + 8\sigma^2\right.$$
$$\left. -2(1-\sigma^2)^2(1+\tilde{T}) - \frac{3\sigma^2\tilde{T}}{1+\tilde{T}}\right\},$$

$$\chi_1 = 1 + \frac{\kappa c_g}{2\omega}(1-\sigma^2)(1+\tilde{T}) \geq 0,$$

$$\alpha = \frac{gh - c_g^2}{gh},$$

$$\beta = \frac{\omega}{\omega_0 kh}\left\{\frac{\kappa c_g}{\omega}(1-\sigma^2) + \frac{2}{1+\tilde{T}}\right\} \geq 0,$$

and where all of the above functions are evaluated at $l = 0$ (flow in nearly one direction).

The signs of λ and α depend on k, h and T and can assume the values $(+,-)$, $(-,+)$ or $(+,+)$ giving the DSI, DSII and equation (1.2) respectively. For small values of $\tilde{T}$ compared to kh we get the DSII system. When $\tilde{T}$ is large compared to kh then we get the DSI system. The third type of DS system, equation (1.2), is achieved for values of the parameters between those of the DSI and DSII systems. See [15, Figure 2] or [4, Figure 1] for graphs of the relationship between the type of DS equation and the parameters $\tilde{T}$ and kh.

2.1.1 Stokes Wave Formulation

In this section we will show that a position independent solution for A in (2.15) leads to traveling wave profile or Stokes wave in ζ. In Chapter 5 we will analyze the modulational stability of this Stokes wave.

Substituting (2.7) into (2.8) and using (2.9), (2.10) and the fact that $\zeta_{00} = 0$, we see that the free surface satisfies

$$\zeta = \frac{i\omega}{g} A \exp\{i(kx - \omega t)\} + \text{c.c.} \tag{2.16}$$

up to first order in ε. If A is independent of ξ and η then (2.14) becomes

$$\left(gh - c_g^2\right)\frac{\partial^2 \phi_{01}}{\partial \xi^2} + gh\frac{\partial^2 \phi_{01}}{\partial \eta^2} = 0.$$

Now assume that $gh > c_g^2$. Then (2.15) becomes

$$\frac{\partial A}{\partial \tau} + i\omega_2 |A|^2 A = 0, \tag{2.17}$$

where

$$\omega_2 = \frac{k^4\left(9\sigma^{-2} - 12 + 13\sigma^2 - 2\sigma^4\right)}{4\omega}.$$

The solution to (2.17) is easily found to be

$$A = a_0 \exp\left(-i\omega_2 a_0^2 \tau\right) \tag{2.18}$$

with a_0 real. Substituting back into (2.16) shows that up to $\mathcal{O}(\varepsilon^2)$, nonlinearity changes the wave phase only. That is, the wave profile is given by

$$\zeta = \frac{i\omega a_0}{g} \exp\left\{i(kx - \tilde{\omega}t)\right\} + \text{c.c.}, \tag{2.19}$$

up to leading order, where $\tilde{\omega} = \omega + \omega_2 a_0^2 \varepsilon^2$.

The modulational stability of (2.19) depends only on the stability of the position independent solution (2.18) of the DS system. Modulational stability refers to the possible effects on the progressive wave from the presence of waves which have side-band frequencies and wave-numbers adjacent to ω_2 and k interacting with the fundamental progressive wave. In section 5.1 we will analyze the stability of (2.19) using a linear stability analysis on the solution (2.18) of the DS system.

2.2 Existence, Uniqueness and Boundary Conditions

In this section we will state some of the results of [15] concerning existence and uniqueness of solutions of the DS system. We will then discuss appropriate boundary conditions for the DS system [3].

Ghidaglia and Saut [15] discuss the existence and uniqueness of solutions to the general form of the DS system

$$iu_t + \alpha_1 u_{xx} + \alpha_2 u_{yy} + \alpha_3 |u|^2 u + \alpha_4 u\phi_x = 0$$

$$\phi_{xx} + \beta\phi_{yy} = (|u|^2)_x. \tag{2.20}$$

Under certain choices of the coefficients, (2.20) can be solved (at least in theory) by the method of inverse scattering. The theory behind the method of inverse scattering has existence, uniqueness and continuous dependence on initial conditions results built into it, see [1]. However, Ghidaglia and Saut develop results for the DS system using more general theories of nonlinear partial differential equations. Their analysis also cover cases where the method of inverse scattering is not applicable.

In [15] the coefficients of (2.20) have been normalized so that $\alpha_2 = 1$ and $|\alpha_1| = |\beta| = 1$. These systems of equations can be classified as elliptic-elliptic, elliptic-hyperbolic or DSI system, hyperbolic-elliptic or DSII system and hyperbolic-hyperbolic depending on the signs of (α_1, β): $(+,+)$, $(+,-)$, $(-,+)$ and $(-,-)$ respectively. The last case, h/h, does not seem to occur in the context of water waves and no existence theorems are given for this case.

Ghidaglia and Saut prove that the DS system have several constants of motion

$$\int\int |u|^2\,dx\,dy, \quad \int\int \alpha_1|u_x|^2 + |u_y|^2 + \frac{1}{2}(\alpha_3|u|^4 + \alpha_4(\phi_x^2 + \beta\phi_y^2))\,dx\,dy,$$

which we will verify in section 2.3, and

$$\int\int (uu_x^* - u^*u_x)\,dx\,dy, \quad \int\int (uu_y^* - u^*u_y)\,dx\,dy.$$

They use these constants of motion to help prove existence results in all but the h/h case of the DS system. Ghidaglia and Saut have shown three results for the e/e and h/e systems.

The e/e and h/e systems:

Existence and Uniqueness: (i) Let $u_0 \in L^2(\mathbb{R}^2)$. There exists a unique solution (u, ϕ) of (2.20) (with $\beta = 1$) on $[0, T)$, $T > 0$ such that

$$u \in C\left([0,T), L^2(\mathbb{R}^2)\right) \cap L^4\left((0,t) \times \mathbb{R}^2\right)$$

$$\nabla\phi \in L^2\left((0,t) \times \mathbb{R}^2\right)$$

$$u(0) = u_0, \qquad 0 \le t < T.$$

(ii)If u_0 is sufficiently small in $L^2(\mathbb{R}^2)$, then $T = \infty$: the solution is global.

Regularity: (i) If $u_0 \in H^1(\mathbb{R}^2)$, the previous solution satisfies

$$u \in C\left([0,T), H^1(\mathbb{R}^2)\right) \cap C^1\left([0,T), H^{-1}(\mathbb{R}^2)\right)$$

$$\nabla u \in L^4\left((0,t) \times \mathbb{R}^2\right), \qquad \nabla\phi \in C\left([0,T), L^p(\mathbb{R}^2)\right)$$

$$\nabla^2\phi \in L^4\left((0,t), L^q(\mathbb{R}^2)\right)$$

for every $t \in [0,T)$, $p \in [2,\infty)$ and $q \in [2,4]$.

(ii) If furthermore $u_0 \in H^2(\mathbb{R}^2)$, then

$$u \in C\left([0,T), H^2(\mathbb{R}^2)\right) \cap C^1\left([0,T), L^2(\mathbb{R}^2)\right)$$

$$\nabla\phi \in C\left([0,t), H^2(\mathbb{R}^2)\right).$$

Continuous dependence: The map $u \to (u, \nabla\phi)$ is continuous from $H^1(\mathbb{R}^2)$ into $C\left(I, H^1(\mathbb{R}^2)\right) \times C\left(I, L^p(\mathbb{R}^2)\right)$, where $I = [0,\tilde{T}]$ and $p > 2$ in the following sense. Let $u \in C\left(I, H^1(\mathbb{R}^2)\right)$, $\nabla\phi \in C\left(I, L^p(\mathbb{R}^2)\right)$ be a solution of (2.20) and let $u_{0n} \to u(0)$ in $H^1(\mathbb{R}^2)$ as $n \to \infty$. Then the solution (u_n, ϕ_n) with $u_n(0) = u_{0n}$ exists on I provided n is sufficiently large and $(u_n, \nabla\phi_n) \to (u, \phi)$ in $C\left(I, H^1(\mathbb{R}^2)\right) \times C\left(I, L^p(\mathbb{R}^2)\right)$, $p > 2$.

To prove these results they start by expressing ϕ in terms of u by inverting the second equation in (2.20). We will follow this method of inverting the elliptic operator in section 4.1. Solving for ϕ_x gives

$$\phi_x = E(|u|^2), \tag{2.21}$$

where E, defined in Fourier variables by

$$\widehat{E(f)}(\xi_1,\xi_2) = \frac{\xi_1^2}{\xi_1^2 + \beta\xi_2^2}\hat{f}(\xi_1,\xi_2), \tag{2.22}$$

is a bounded operator in $L^p(\mathbb{R}^2)$ and there exists a constant $c = c_p > 0$ so that

$$\|\phi_x\|_{L^p(\mathbf{R}^2)} \le c_p\|u\|_{L^{2p}(\mathbf{R}^2)} \quad 1 < p < \infty. \tag{2.23}$$

Then (2.20) becomes a nonlinear, non-local Schrödinger equation

$$iu_t + \alpha_1 u_{xx} + u_{yy} = -\alpha_3 |u|^2 u - \alpha_4 u E(|u|^2), \quad \alpha_1 = \pm 1,$$

which is given the initial value

$$u(x, y, 0) = u_0(x, y).$$

This is a particular case of the more general form

$$iu_t + Lu = F(u),$$

where the operator L is given by

$$Lu = \sum_{j=1}^{n} \sum_{k=1}^{n} a_{jk} \frac{\partial^2 u}{\partial x_j \partial x_k} \quad \text{in } \mathbb{R}^n,$$

and the real symmetric matrix (a_{jk}) is invertible. They then use classical arguments to prove the above assertions.

The e/h system:

In the e/h case the DS system is given by

$$\begin{aligned} iu_t + u_{xx} + u_{yy} + \alpha_3 |u|^2 u + \alpha_4 u \phi_x &= 0, \\ \phi_{xx} - c^2 \phi_{yy} &= (|u|^2)_x, \\ u(x, y, 0) &= u_0(x, y), \end{aligned} \tag{2.24}$$

where $\beta < 0$ and $c = \sqrt{-\beta}$. Ghidaglia and Saut found that for every $u_0 \in H^1(\mathbb{R}^2)$ satisfying

$$\left[\frac{2|\alpha_4|}{c} + \max(\alpha_3, 0) \right] \int_{\mathbf{R}^2} |u_0|^2 \, dx \, dy < 1$$

there exist u and ϕ with

$$u \in L^\infty \left(\mathbb{R}_+, H^1(\mathbb{R}^2) \right) \cap C \left(\mathbb{R}_+, H^1(\mathbb{R}^2) \right)$$

$$\phi \in L^\infty \left(\mathbb{R}_+, C_b(\mathbb{R}^2) \right), \qquad \nabla \phi \in L^\infty \left(\mathbb{R}_+, L^q_{\text{loc}}(\mathbb{R}^2) \right) \qquad 1 \leq q \leq 2$$

which satisfies $u(0) = u_0$ and (2.24) in the sense of distributions.

To prove this they first use characteristic coordinates, $\xi = cx - y$ and $\eta = cx + y$, and solved for ϕ in

$$\phi_{xx} - c^2\phi_{yy} = f,$$

where $f \in L^1(\mathbb{R}^2)$ and ϕ satisfies

$$\lim_{\xi\to\infty} \phi = \lim_{\eta\to\infty} \phi = 0.$$

For ever $f \in L^1(\mathbb{R}^2)$, $\phi = K(f)$ where the integral operator, K, is given by

$$K(f) = \int_{\mathbb{R}^2} k(x, y; x_1, y_1) f(x_1, y_1)\, dx_1\, dy_1,$$

$$k(x, y; x_1, y_1) = \frac{1}{2} h(c(x_1 - x) + y - y_1) h(c(x_1 - x) + y_1 - y)$$

and h is the usual Heaviside function. The DS system now becomes

$$iu_t + \Delta u = -\alpha_3 |u|^2 u - \alpha_4 u \left(K\left(\left(|u|^2\right)_x\right)\right)_x$$

$$u(x, y, 0) = u_0(x, y). \tag{2.25}$$

Since the operator K has no regularizing effects, the same classical arguments do not apply. Thus, Ghidaglia and Saut find solutions to (2.25) as a limit of solutions, u^ε, to a regularised equation

$$iu_t^\varepsilon + i\varepsilon\Delta^2 u_t^\varepsilon + \Delta u^\varepsilon = -\alpha_3 |u^\varepsilon|^2 u^\varepsilon - \alpha_4 u^\varepsilon \left(K\left(\left(|u^\varepsilon|^2\right)_x\right)\right)_x$$

$$u^\varepsilon(x, y, 0) = u_0^\varepsilon(x, y).$$

Boundary conditions:

In [3], Ablowitz, et. al., investigate the question of the proper boundary condition for applying the method of inverse scattering to the e/h or DSI system and the h/e or DSII system of equations. They embed the DS system within the KP evolution equation, maintaining well posedness in time. They then sought physically acceptable boundary conditions on the DS system when decaying boundary

conditions are assumed on the KP equation. They found that in both the DSI and DSII systems $|u| \to 0$ as $x^2 + y^2 \to \infty$. For the DSII system ϕ_x vanishes at infinity. However in the DSI system $\phi_x \to 0$ either as $x \to \infty$ or $x \to -\infty$, but not both, are necessary to apply the method of inverse scattering. In section 4.2 we will see that soliton like solutions can be driven by nonzero boundary conditions on ϕ_x in the DSI system [14].

In this thesis we will use $|u| \to 0$ and $\phi_x \to 0$ as $x^2 + y^2 \to \infty$ for the h/e and e/e DS systems when considering solitons in section 4.1 and solutions which become singular in Chapter 6. In Chapter 5 periodic perturbations of (uniform) Stokes wave solutions to the h/e and e/e DS systems will be analyzed.

2.3 Invariants of DS System

In this section we show that the solutions to the DS system must satisfy several conservation laws. In particular, we shall show that

$$\frac{d}{dt}I = 0, \qquad \text{where} \qquad I = \int\int |u|^2 \, dx \, dy,$$

and

$$\frac{d}{dt}H = 0, \text{ where } H = i\int\int \left\{-\alpha_1|u_x|^2 - \alpha_2|u_y|^2 + \frac{1}{2}|u|^2\left(\alpha_3|u|^2 + \alpha_4\phi_x\right)\right\} dx\, dy.$$

These two quantities should be compared with (1.6) and (1.7) in the NLS case. We follow the approach used in [15], where conservation laws are derived for a more general form of the DS system which includes as special cases (1.1).

Consider the system of partial differential equations

$$iu_t + \alpha_1 u_{xx} + \alpha_2 u_{yy} + \alpha_3|u|^2u + \alpha_4 u\phi_x = 0, \tag{2.26}$$

$$\phi_{xx} + \beta\phi_{yy} = (|u|^2)_x, \tag{2.27}$$

where $u \in \mathbb{C}$, $\phi \in \mathbb{R}$ and $\alpha_1, \ldots, \alpha_4$, β are real constants. The complex conjugate of (2.26) is

$$-iu_t^* + \alpha_1 u_{xx}^* + \alpha_2 u_{yy}^* + \alpha_3|u|^2u^* + \alpha_4 u^*\phi_x = 0, \tag{2.28}$$

where u^* denotes the complex conjugate of u. To derive the Hamiltonian of the DS system as a conserved quantity, multiply (2.26) by u_t^* and (2.28) by u_t and add to get

$$\begin{aligned} &\alpha_1 \left(u_{xx}u_t^* + u_{xx}^* u_t\right) + \alpha_2 \left(u_{yy}u_t^* + u_{yy}^* u_t\right) \\ &+ \alpha_3 |u|^2 \left(uu_t^* + u^* u_t\right) + \alpha_4 \phi_x \left(uu_t^* + u^* u_t\right) \;=\; 0. \end{aligned} \tag{2.29}$$

Thus

$$\begin{aligned} \int\int \Big\{ &\alpha_1 \left(u_{xx}u_t^* + u_{xx}^* u_t\right) + \alpha_2 \left(u_{yy}u_t^* + u_{yy}^* u_t\right) \\ &+ \alpha_3 |u|^2 \left(|u|^2\right)_t + \alpha_4 \phi_x \left(|u|^2\right)_t \Big\} \, dx\, dy \;=\; 0. \end{aligned} \tag{2.30}$$

If either periodic or decaying boundary conditions are assumed on u, then integration by parts on the first two terms yields

$$\int\int \left\{ \left(-\alpha_1 |u_x|^2 - \alpha_2 |u_y|^2 + \frac{1}{2}\alpha_3 \left(|u|^4\right) \right)_t + \alpha_4 \phi_x \left(|u|^2\right)_t \right\} dx\, dy = 0. \tag{2.31}$$

To write the last term as a time derivative, differentiate (2.27) by t and multiply the result by ϕ to get

$$\phi\phi_{xxt} + \beta\phi\phi_{yyt} = \phi \left(|u|^2\right)_{xt}$$

or

$$\left(\phi\phi_{xt}\right)_x + \beta \left(\phi\phi_{yt}\right)_y - \phi_x\phi_{xt} - \beta\phi_y\phi_{yt} = \left[\left(|u|^2\right)_t \phi \right]_x - \left(|u|^2\right)_t \phi_x.$$

Now integrate and use the assumption of periodic or decaying boundary conditions to get

$$\begin{aligned} \int\int \phi_x \left(|u|^2\right)_t dx\, dy \;&=\; \int\int \left\{ \phi_x\phi_{xt} + \beta\phi_y\phi_{yt} \right\} dx\, dy \\ &=\; \frac{1}{2}\frac{d}{dt} \int\int \left\{ (\phi_x)^2 + \beta \left(\phi_y\right)^2 \right\} dx\, dy. \end{aligned} \tag{2.32}$$

Integration by parts of the right hand side of this equation yields

$$\int\int \phi_x \left(|u|^2\right)_t dx\, dy = -\frac{1}{2}\frac{d}{dt} \int\int \left(\phi_{xx} + \beta\phi_{yy}\right) \phi \, dx\, dy.$$

Then the use of (2.27) gives

$$\iint \phi_x \left(|u|^2\right)_t \, dx\, dy = -\frac{1}{2}\frac{d}{dt}\iint \left(|u|^2\right)_x \phi \, dx\, dy,$$

and integration by parts gives

$$\iint \phi_x \left(|u|^2\right)_t \, dx\, dy = \frac{1}{2}\frac{d}{dt}\iint |u|^2 \phi_x \, dx\, dy.$$

Therefore, (2.31) becomes

$$\frac{d}{dt}\iint \left\{-\alpha_1|u_x|^2 - \alpha_2|u_y|^2 + \frac{1}{2}|u|^2\left(\alpha_3|u|^2 + \alpha_4\phi_x\right)\right\} dx\, dy = 0.$$

Thus, we see that

$$H = i\iint \left\{-\alpha_1|u_x|^2 - \alpha_2|u_y|^2 + \frac{1}{2}|u|^2\left(\alpha_3|u|^2 + \alpha_4\phi_x\right)\right\} dx\, dy, \tag{2.33}$$

is conserved. Note that if there is no x-dependence, then (2.33) becomes the Hamiltonian for the NLS equation (1.7).

To see that H is the Hamiltonian for the DS system note that (2.26) and (2.28) form a system of equations in the variables u and u^*

$$u_t = i\left(\alpha_1 u_{xx} + \alpha_2 u_{yy} + \alpha_3|u|^2 u + \alpha_4 u\phi_x\right),$$

$$u_t^* = -i\left(\alpha_1 u_{xx}^* + \alpha_2 u_{yy}^* + \alpha_3|u|^2 u^* + \alpha_4 u^*\phi_x\right),$$

$$\phi_{xx} + \beta\phi_{yy} = (|u|^2)_x.$$

The Hamiltonian formulation of this system is

$$\frac{\partial u}{\partial t} = \frac{\delta H}{\delta u^*}, \quad \frac{\partial u^*}{\partial t} = -\frac{\delta H}{\delta u} \tag{2.34}$$

where H can be written as

$$H = -i\int\int \left\{\alpha_1 u_x u_x^* + \alpha_2 u_y u_y^* - \frac{1}{2}\alpha_3 u^2 (u^*)^2 - \frac{1}{2}\alpha_4 u u^* \phi_x\right\}$$

and the variational derivative used on the right hand sides of (2.34) satisfy

$$\lim_{\epsilon\to 0}\frac{H(u+\epsilon\tilde{u}, u^*) - H(u, u^*)}{\epsilon} = \int \frac{\delta H}{\delta u}\,\tilde{u}\, dx,$$

for all $\widetilde{u}$ which are suitably smooth and decay to zero at the boundary. To see this, first note that if $v = \phi_x$, then

$$v_{xx} + \beta v_{yy} = (|u|^2)_{xx},$$

which has solution

$$v = \int\int G(x-x', y-y') \left[u(x',y')u^*(x',y')\right]_{x'x'} \, dx' \, dy',$$

where the Green's function is given by (see [1, page 255])

$$G(x,y) = \frac{i}{4\pi\sqrt{\beta}} \ln\left(y^2 + \beta x^2\right).$$

Next note that the variational derivatives of all but the last term in the Hamiltonian are derived in the same manner as for the NLS case. Thus, it remains to be shown that if

$$J(u,u^*) = -i \int\int u u^* v \, dx \, dy,$$

then

$$\frac{\delta J}{\delta u} = -2iu^*v.$$

Let $q = u(x,y)$ be the generalized coordinates and $r = u^*(x,y)$ be the generalized momenta and denote $q' = q(x',y')$ and $r' = r(x',y')$. Then

$$\begin{aligned}
&\frac{J(q+\epsilon\widetilde{q}, r) - J(q,r)}{\epsilon} \\
= \; & -\frac{i}{\epsilon}\int\int (q+\epsilon\widetilde{q})r \int\int G(x-x', y-y') \left[(q'+\epsilon\widetilde{q}')r'\right]_{x'x'} \, dx' \, dy' \, dx \, dy \\
& +\frac{i}{\epsilon}\int\int qr \int\int G(x-x', y-y') \left[q'r'\right]_{x'x'} \, dx' \, dy' \, dx \, dy \\
= \; & -i\int\int qr \int\int G(x-x', y-y') \left[\widetilde{q}'r'\right]_{x'x'} \, dx' \, dy' \, dx \, dy \\
& -i\int\int \widetilde{q}r \int\int G(x-x', y-y') \left[q'r'\right]_{x'x'} \, dx' \, dy' \, dx \, dy \\
& -i\int\int \epsilon\widetilde{q}r \int\int G(x-x', y-y') \left[\widetilde{q}'r'\right]_{x'x'} \, dx' \, dy' \, dx \, dy \, ,
\end{aligned}$$

where we have multiplied out the terms in the first integral and then canceled like terms. Now let $\epsilon \to 0$ to get

$$\lim_{\epsilon\to 0} \frac{1}{\epsilon}\Big(J(q+\epsilon\widetilde{q}, r) \;\; - \;\; J(q,r)\Big)$$

$$
\begin{aligned}
&= -i \int\int qr \int\int G(x-x', y-y') \left[\tilde{q}' r'\right]_{x'x'} \, dx' \, dy' \, dx \, dy \\
&\quad -i \int\int \tilde{q} r v \, dx \, dy \\
&= -i \int\int \left[\tilde{q}' r'\right]_{x'x'} \int\int G(x'-x, y'-y) qr \, dx \, dy \; dx' \, dy' \\
&\quad -i \int\int \tilde{q} r v \, dx \, dy \,, \qquad (2.35)
\end{aligned}
$$

where we have changed the order of integration and used the symmetry of G. But

$$\int\int G(x'-x, y'-y) qr \, dx \, dy$$

is a function of x' and y'. Define this function to be $w(x', y')$. Then w satisfies

$$w_{xx} + \beta w_{yy} = |u|^2 \; (= qr).$$

So, integration by parts twice in (2.35) gives

$$
\begin{aligned}
\lim_{\epsilon \to 0} \frac{1}{\epsilon} \Big(J(q + \epsilon \tilde{q}, r) \; &- \; J(q, r) \Big) \\
&= -i \int\int \left[\tilde{q}' r'\right] w'_{x'x'} \, dx' \, dy' - i \int\int \tilde{q} r v \, dx \, dy \,,
\end{aligned}
$$

where we have used the notation $w' = w(x', y')$. Now note that w_{xx} satisfies the same equation as v, namely

$$\left(w_{xx}\right)_{xx} + \beta \left(w_{xx}\right)_{yy} = \left(|u|^2\right)_{xx}.$$

Since the solution to this equation is unique we may set $v = w_{xx}$ provided we assume the same boundary conditions on v and w_{xx}. Then the desired result is attained.

Note that with an additional integration by parts

$$H = i \iint \left\{ \left(\alpha_1 u_{xx} + \alpha_2 u_{yy}\right) u^* + \frac{1}{2} \left(\alpha_3 |u|^2 + \alpha_4 \phi_x\right) |u|^2 \right\} dx \, dy. \qquad (2.36)$$

For computational purposes (2.36) will be used.

To see that the L^2-norm of u is conserved, multiply (2.26) by u^* and (2.28) by u and subtract to get

$$i \left(u^* u_t + u u_t^*\right) + \alpha_1 \left(u^* u_{xx} - u u_{xx}^*\right) + \alpha_2 \left(u^* u_{yy} - u u_{yy}^*\right) = 0.$$

Thus

$$\frac{d}{dt}\iint |u|^2\,dx\,dy = i\iint \left\{\alpha_1\left(uu^*_{xx} - u^*u_{xx}\right) + \alpha_2\left(uu^*_{yy} - u^*u_{yy}\right)\right\}\,dx\,dy.$$

Integrate by parts and assume either periodic or decaying boundary conditions on u to arrive at

$$\frac{d}{dt}\iint |u|^2\,dx\,dy = 0. \tag{2.37}$$

Note that this is the two dimensional analog of the NLS case (1.6).

Chapter 3
Numerical Method

In this chapter a numerical method for solving the DS system is introduced. A method for approximating the space derivatives is discussed, and then a semi-discrete system of ordinary differential equations is introduced as an approximation to the DS system of partial differential equations. This semi-discrete system is shown to have a conserved Hamiltonian and l_2-norm which are discrete approximations to the corresponding continuous quantities derived in Chapter 2.

The split-step integration technique is then introduced as a numerical scheme for solving the semi-discrete system. The accuracy of the split-step method and its computational complexity are discussed. The integration technique is then shown to be a symplectic method.

3.1 Fourier Differentiation

To approximate the DS equations it is necessary to have discrete approximations to $\frac{\partial}{\partial x}$, $\frac{\partial^2}{\partial x^2}$ and $\frac{\partial^2}{\partial y^2}$. If u is periodic and sufficiently smooth, then u can be written as a Fourier series. If u decays to zero at infinity and is sufficiently smooth, then u can be approximated on a truncated domain by the Fourier series:

$$u(x,y,t) = \sum_{m=-\infty}^{\infty} \sum_{n=-\infty}^{\infty} a_{mn}(t) e^{i\mu_m x + i\nu_n y}, \tag{3.1}$$

where $\mu_m = \frac{2\pi m}{P}$, $\nu_n = \frac{2\pi n}{P}$ and P is either the period of u or chosen sufficiently large so that the derivatives of u are negligible outside of the region $[-\frac{1}{2}P, \frac{1}{2}P] \times$

$[-\frac{1}{2}P, \frac{1}{2}P]$. Equation (3.1) may be approximated by the finite series

$$u(x,y,t) = \sum_{m=-N}^{N-1} \sum_{n=-N}^{N-1} a_{mn}(t) e^{i\mu_m x + i\nu_n y},$$

provided a_{mn} is negligible for $|m| > N$ or $|n| > N$. Then the derivatives of u are given by

$$u_x = \sum_{m=-N}^{N-1} \sum_{n=-N}^{N-1} i\mu_m a_{mn} e^{i\mu_m x + i\nu_n y},$$

$$u_{xx} = \sum_{m=-N}^{N-1} \sum_{n=-N}^{N-1} -\mu_m^2 a_{mn} e^{i\mu_m x + i\nu_n y},$$

and

$$u_{yy} = \sum_{m=-N}^{N-1} \sum_{n=-N}^{N-1} -\nu_n^2 a_{mn} e^{i\mu_m x + i\nu_n y}.$$

Now we can discretize the space variables, x and y. Let $\Delta x = \Delta y = \frac{P}{2N}$, $x_j = -\frac{1}{2}P + j\Delta x$ and $y_k = -\frac{1}{2}P + k\Delta y$. If U is a $2N$ by $2N$ matrix such that $U_{jk}(t)$ approximates $u(x_j, y_k, t)$, then the approximation to u_x is given by

$$L_x U = \mathcal{F}^H E_x \mathcal{F} U,$$

where L_x is a matrix approximation of $\frac{\partial}{\partial x}$, $\mathcal{F}$ is the discrete Fourier Transform matrix given by

$$\mathcal{F}_{jk} = \frac{1}{\sqrt{2N}} \exp\left(\frac{-2\pi i(j-1)k}{2N}\right)$$

and $E_x = \text{diag}\,\{(i\mu_m) : m = -N, \ldots, N-1\}$ is the diagonal matrix of eigenvalues for the differentiation matrix. It is easily verified that $\mathcal{F}$ is unitary, i.e., $\mathcal{F}^H = \mathcal{F}^{-1}$. An approximation to u_{xx} is then given by

$$L_{xx} U = \mathcal{F}^H E_{xx} \mathcal{F} U, \tag{3.2}$$

where $E_{xx} = E_x^2$. A similar approach is used to approximate u_{yy}.

Approximating the space derivatives is therefore accomplished by calculating the Fourier coefficients $\widehat{U} = \mathcal{F}U$, multiplying the Fourier coefficient matrix by the diagonal matrix of eigenvalues for the derivative matrix and then applying an

inverse Fourier Transform matrix. If U is a n by n matrix, then a Fast Fourier Transform algorithm can be used instead of the standard matrix multiplication to calculate $\widehat{U}$. This is particularly efficient if n is a power of 2. The number of multiplications needed in approximating the space derivatives L_x, L_{xx} and L_{yy} is $\mathcal{O}((n \log n)^2)$ multiplications for the Fourier and inverse Fourier Transforms and n^2 multiplications for the multiplication by a diagonal matrix.

3.2 Invariants for the Semi-Discrete DS System

In this section we introduce a system of ordinary differential equations in the variable t which will be used as an approximation to the DS equations. This semi-discrete DS system will be shown to have conserved quantities analogous to those found for the DS system (2.36) and (2.37). The DS equation (2.26) can be written in operator form as

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

where

$$\mathcal{L}u = \alpha_1 u_{xx} + \alpha_2 u_{yy},$$

$$\mathcal{N}(u) = \alpha_3 |u|^2 + \alpha_4 \phi_x,$$

with ϕ satisfying:

$$\phi_{xx} + \beta \phi_{yy} = \left(|u|^2\right)_x.$$

Assume a periodic solution of period P and let U and Φ be the matrix approximation to the continuous variables u and ϕ. That is, discretize the space domain so that $x_j = -\frac{1}{2}P + j\Delta x$ and $y_k = -\frac{1}{2}P + k\Delta y$, with $j, k = 1, \ldots, n$ and $\Delta x = \Delta y = \frac{P}{n}$. We denote the approximation of $u(x_j, y_k)$ by U_{jk}, and $\phi(x_j, y_k)$ by Φ_{jk}.

The semi-discrete DS equations are the system of ordinary differential equations

$$\frac{d}{dt}U = iLU + iN(U) \circ U, \tag{3.3}$$

where

$$\mathcal{L} \approx L = \alpha_1 L_{xx} + \alpha_2 L_{yy}, \tag{3.4}$$

$$\mathcal{N}(u) \approx N(U) = \alpha_3 U \circ U^* + \alpha_4 L_x \Phi, \tag{3.5}$$

$$L_{xx}\Phi + \beta L_{yy}\Phi = L_x(U \circ U^*), \tag{3.6}$$

and $(U^*)_{jk} = (U_{jk})^*$ is the complex conjugate of U_{jk} (Note: the conjugate transpose of a matrix is denoted by $(U^H)_{jk} = U^*_{kj}$). L_x, L_{xx} and L_{yy} are matrices which are the discrete approximations of $\frac{\partial}{\partial x}$, $\frac{\partial^2}{\partial x^2}$ and $\frac{\partial^2}{\partial y^2}$ defined in the previous section. $(M \circ N)_{jk} = M_{jk}N_{jk}$ is the scalar or Hadamard product of matrices.

We will now show that this discrete system of ODE's has the conserved quantities

$$\frac{d}{dt}H = 0, \quad \text{where} \quad H = i\sum_{j=1}^{n}\sum_{k=1}^{n}\left[U^*_{jk}(LU)_{jk} + \frac{1}{2}N(U)_{jk}|U_{jk}|^2\right],$$

and

$$\frac{d}{dt}I = 0, \quad \text{where} \quad I = \sum_{j=1}^{n}\sum_{k=1}^{n}|U_{jk}|^2.$$

These should be compared with the analogous continuous quantities (2.36) and (2.37). First consider the jk-th equation in (3.3)

$$\frac{d}{dt}U_{jk} = i\left[(LU)_{jk} + N(U)_{jk}U_{jk}\right], \tag{3.7}$$

and its complex conjugate

$$\frac{d}{dt}U^*_{jk} = -i\left[(LU)^*_{jk} + N(U)_{jk}U^*_{jk}\right]. \tag{3.8}$$

Multiply equation (3.7) by $\frac{d}{dt}U^*_{jk}$ and (3.8) by $\frac{d}{dt}U_{jk}$ and subtract to get

$$\begin{aligned} 0 \;=\; & i\Big\{\alpha_1\left[(L_{xx}U)_{jk}\frac{d}{dt}U^*_{jk} + (L_{xx}U)^*_{jk}\frac{d}{dt}U_{jk}\right] \\ & + \alpha_2\left[(L_{yy}U)_{jk}\frac{d}{dt}U^*_{jk} + (L_{yy}U)^*_{jk}\frac{d}{dt}U_{jk}\right] \\ & + N(U)_{jk}\left[U_{jk}\frac{d}{dt}U^*_{jk} + U^*_{jk}\frac{d}{dt}U_{jk}\right]\Big\} \end{aligned}$$

or

$$\begin{aligned} 0 &= i\Big\{\alpha_1\left[(L_{xx}U)_{jk}\frac{d}{dt}U^*_{jk}+(L_{xx}U)^*_{jk}\frac{d}{dt}U_{jk}\right] \\ &\quad +\alpha_2\left[(L_{yy}U)_{jk}\frac{d}{dt}U^*_{jk}+(L_{yy}U)^*_{jk}\frac{d}{dt}U_{jk}\right] \\ &\quad +\alpha_3|U_{jk}|^2\frac{d}{dt}\left(|U_{jk}|^2\right)+\alpha_4(L_x\Phi)_{jk}\frac{d}{dt}\left(|U_{jk}|^2\right)\Big\}. \end{aligned} \tag{3.9}$$

Now we can sum equation (3.9) over all j and k to get

$$\begin{aligned} 0 &= i\sum_{j=1}^{n}\sum_{k=1}^{n}\Big\{\alpha_1\left[(L_{xx}U)_{jk}\frac{d}{dt}U^*_{jk}+(L_{xx}U)^*_{jk}\frac{d}{dt}U_{jk}\right] \\ &\quad +\alpha_2\left[(L_{yy}U)_{jk}\frac{d}{dt}U^*_{jk}+(L_{yy}U)^*_{jk}\frac{d}{dt}U_{jk}\right] \\ &\quad +\frac{d}{dt}\left[\frac{1}{2}\alpha_3\left(|U_{jk}|^4\right)\right]+\alpha_4(L_x\Phi)_{jk}\frac{d}{dt}\left(|U_{jk}|^2\right)\Big\}. \end{aligned} \tag{3.10}$$

We will now show that the terms in the first two square brackets can be written as a time derivative. Since $\mathcal{F}^{-1}=\mathcal{F}^H$ and

$$L_x^H=\left(\mathcal{F}^HE_x\mathcal{F}\right)^H=\mathcal{F}^HE_x^H\mathcal{F}=-\mathcal{F}^HE_x\mathcal{F}=-L_x,$$

we can write L_{xx} as

$$L_{xx}=\mathcal{F}^HE_x^2\mathcal{F}=\mathcal{F}^HE_x\mathcal{F}\mathcal{F}^HE_x\mathcal{F}=-L_x^HL_x.$$

This shows that L_{xx} is self adjoint

$$L_{xx}^H=\left(-L_x^HL_x\right)^H=-L_x^HL_x=L_{xx}.$$

If $\vec{w}$ is any time-dependent column vector, then L_{xx} being self adjoint gives

$$\begin{aligned} \sum_{l=1}^{n}\left[(L_{xx}\vec{w})_l\frac{d}{dt}w_l^*+(L_{xx}\vec{w})_l^*\frac{d}{dt}w_l\right] &= \left(\frac{d}{dt}\vec{w}^H\right)(L_{xx}\vec{w})+(L_{xx}\vec{w})^H\left(\frac{d}{dt}\vec{w}\right) \\ &= \left(\frac{d}{dt}\vec{w}^H\right)(L_{xx}\vec{w})+\vec{w}^H\left(\frac{d}{dt}L_{xx}\vec{w}\right) \\ &= \frac{d}{dt}(\vec{w}^HL_{xx}\vec{w}) \\ &= \frac{d}{dt}\sum_{l=1}^{n}w_l^*(L_{xx}\vec{w})_l. \end{aligned} \tag{3.11}$$

Similarly

$$\sum_{l=1}^{n}\left[(L_{yy}\vec{w})_l\frac{d}{dt}w_l^* + (L_{yy}\vec{w})_l^*\frac{d}{dt}w_l\right] = \frac{d}{dt}\sum_{l=1}^{n} w_l^*(L_{yy}\vec{w})_l. \tag{3.12}$$

Thus, substituting (3.11) and (3.12) into (3.10) gives

$$\begin{aligned} 0 &= i\frac{d}{dt}\sum_{j=1}^{n}\sum_{k=1}^{n}\left[\alpha_1 U_{jk}^*(L_{xx}U)_{jk} + \alpha_2 U_{jk}^*(L_{yy}U)_{jk} + \frac{1}{2}\alpha_3|U_{jk}|^4\right] \\ &\quad + \sum_{j=1}^{n}\sum_{k=1}^{n}\alpha_4(L_x\Phi)_{jk}\frac{d}{dt}(|U_{jk}|^2). \end{aligned} \tag{3.13}$$

To write the second sum in (3.13) as a time derivative, first look at

$$\begin{aligned} \sum_{j=1}^{n}\sum_{k=1}^{n}\left(|U_{jk}|^2\right)(L_x\Phi)_{jk} &= \sum_{j=1}^{n}\sum_{k=1}^{n}\sum_{l=1}^{n}\left(|U_{jk}|^2\right)(L_x)_{jl}\Phi_{lk} \\ &= \sum_{j=1}^{n}\sum_{k=1}^{n}\sum_{l=1}^{n}(L_x)_{lj}^*(U\circ U^*)_{jk}\Phi_{lk} \\ &= \sum_{k=1}^{n}\sum_{l=1}^{n}(L_x(U\circ U^*))_{lk}^*\,\Phi_{lk} \end{aligned} \tag{3.14}$$

where we have used $(L_x)_{jk} = (L_x)_{kj}^*$. Now since $U\circ U^*$ is real, $(L_x(U\circ U^*))_{lk}^* = (L_x(U\circ U^*))_{lk}$ so (3.6) implies

$$\sum_{j=1}^{n}\sum_{k=1}^{n}\left(|U_{jk}|^2\right)(L_x\Phi)_{jk} = \sum_{j=1}^{n}\sum_{k=1}^{n}\left(L_{xx}\Phi + \beta L_{yy}\Phi\right)_{jk}\Phi_{jk}. \tag{3.15}$$

Next, let $\vec{w}$ be a real column vector and look at

$$\begin{aligned} \frac{1}{2}\frac{d}{dt}\sum_{l=1}^{n}(L_{xx}\vec{w})_l w_l &= \frac{1}{2}\frac{d}{dt}\left[\vec{w}^H L_{xx}\vec{w}\right] \\ &= \frac{1}{2}\left[\left(\frac{d}{dt}\vec{w}^H\right)(L_{xx}\vec{w}) + \vec{w}^H\frac{d}{dt}(L_{xx}\vec{w})\right] \\ &= \frac{1}{2}\left[\left(\frac{d}{dt}\vec{w}\right)^H(L_{xx}\vec{w}) + (L_{xx}\vec{w})^H\left(\frac{d}{dt}\vec{w}\right)\right] \\ &= \sum_{l=1}^{n} w_l\left(L_{xx}\left(\frac{d}{dt}w\right)\right)_l, \end{aligned} \tag{3.16}$$

where we have again used the self adjoint property of L_{xx} in the third line. So

$$\frac{1}{2}\frac{d}{dt}\sum_{j=1}^{n}\sum_{k=1}^{n}(L_{xx}\Phi)_{jk}\Phi_{jk} = \sum_{j=1}^{n}\sum_{k=1}^{n}\left(L_{xx}\frac{d}{dt}\Phi\right)_{jk}\Phi_{jk}, \tag{3.17}$$

and similarly

$$\frac{1}{2}\frac{d}{dt}\sum_{j=1}^{n}\sum_{k=1}^{n}\left(L_{yy}\Phi\right)_{jk}\Phi_{jk}=\sum_{j=1}^{n}\sum_{k=1}^{n}\left(L_{yy}\frac{d}{dt}\Phi\right)_{jk}\Phi_{jk}. \tag{3.18}$$

Now differentiate (3.15) with respect to t and use (3.17) and (3.18) to get

$$\frac{1}{2}\frac{d}{dt}\sum_{j=1}^{n}\sum_{k=1}^{n}\left(|U_{jk}|^2\right)\left(L_x\Phi\right)_{jk}=\sum_{j=1}^{n}\sum_{k=1}^{n}\left[\left(L_{xx}\frac{d}{dt}\Phi\right)_{jk}+\left(L_{yy}\frac{d}{dt}\Phi\right)_{jk}\right]\Phi_{jk}. \tag{3.19}$$

Differentiating (3.6) with respect to t and substituting into (3.19) yields

$$\frac{1}{2}\frac{d}{dt}\sum_{j=1}^{n}\sum_{k=1}^{n}\left(|U_{jk}|^2\right)\left(L_x\Phi\right)_{jk}=\sum_{j=1}^{n}\sum_{k=1}^{n}\left(L_x\frac{d}{dt}(U\circ U^*)\right)_{jk}\Phi_{jk},$$

and using the same procedure as in (3.14), this equation becomes

$$\frac{1}{2}\frac{d}{dt}\sum_{j=1}^{n}\sum_{k=1}^{n}\left(|U_{jk}|^2\right)\left(L_x\Phi\right)_{jk}=\sum_{j=1}^{n}\sum_{k=1}^{n}\frac{d}{dt}\left(|U_{jk}|^2\right)\left(L_x\Phi\right)_{jk}. \tag{3.20}$$

Now (3.13) becomes

$$\begin{aligned}0 \;=\; & i\frac{d}{dt}\Bigg\{\sum_{j=1}^{n}\sum_{k=1}^{n}\left[\alpha_1U^*_{jk}(L_{xx}U)_{jk}+\alpha_2U^*_{jk}(L_{yy}U)_{jk}+\frac{1}{2}\alpha_3|U_{jk}|^4\right]\\ & +\sum_{j=1}^{n}\sum_{k=1}^{n}\frac{1}{2}\alpha_4\left(|U_{jk}|^2\right)\left(L_x\Phi\right)_{jk}\Bigg\}.\end{aligned}$$

So

$$H=i\sum_{j=1}^{n}\sum_{k=1}^{n}\left[U^*_{jk}(LU)_{jk}+\frac{1}{2}N(U)_{jk}|U_{jk}|^2\right], \tag{3.21}$$

is conserved by the semi-discrete DS system.

We will now show that (3.21) is the Hamiltonian for the semi-discrete DS system

$$\frac{d}{dt}U_{jk}=i(LU)_{jk}+\alpha_3i|U_{jk}|^2U_{jk}+\alpha_4iU_{jk}(L_x\Phi)_{jk},$$

$$L_{xx}\Phi+\beta L_{yy}\Phi=L_x(U\circ U^*),$$

and its conjugate

$$\frac{d}{dt}U^*_{jk}=-i(LU^*)_{jk}-\alpha_3i|U_{jk}|^2U^*_{jk}-\alpha_4iU^*_{jk}(L_x\Phi)_{jk}.$$

To see this, first expand H

$$H = i\sum_{j=1}^{n}\sum_{k=1}^{n}\left[U^*_{jk}(LU)_{jk} + \frac{1}{2}\alpha_3 U^2_{jk}\left(U^*_{jk}\right)^2 + \frac{1}{2}\alpha_4 U_{jk}U^*_{jk}(L_x\Phi)_{jk}\right],$$

and note that Φ depends on U and U^*. Note also that in this Hamiltonian formulation U and U^* play the roles of the generalized coordinates and generalized momenta of the system. Then

$$\frac{d}{dt}U_{jk} = \frac{\partial H}{\partial U^*_{jk}}, \quad \frac{d}{dt}U^*_{jk} = -\frac{\partial H}{\partial U_{jk}},$$

is the Hamiltonian formulation for the equations of motion.

To show that H is the Hamiltonian, differentiation of the terms involving $L_x\Phi$ is the only nontrivial operation; thus we will expand that term and show that

$$\frac{\partial}{\partial U_{pq}}\sum_{j,k}(L_x\Phi)_{jk}\,|U_{jk}|^2 = 2\,(L_x\Phi)_{pq}\,U^*_{pq}.$$

In the case $\beta = 1$ we can write

$$\begin{aligned}(L_x\Phi)_{jk} &= \left(\mathcal{F}^H B \circ \mathcal{F}\left(U \circ U^*\right)\right)_{jk}\\ &= \sum_{l=1}^{n}\left(\mathcal{F}^H\right)_{jl} b_{lk}\sum_{m=1}^{n}\mathcal{F}_{lm}|U_{mk}|^2\\ &= \sum_{l=1}^{n}\mathcal{F}^*_{lj}b_{lk}\sum_{m=1}^{n}\mathcal{F}_{lm}|U_{mk}|^2,\end{aligned}$$

where $\mathcal{F}$ is the Fourier Transform matrix and B is a constant matrix which we will formally derive in section 4.1. Thus

$$\sum_{j,k}(L_x\Phi)_{jk}\,|U_{jk}|^2 = \sum_{j,k,l,m}\mathcal{F}^*_{lj}b_{lk}\mathcal{F}_{lm}|U_{mk}|^2|U_{jk}|^2.$$

Now differentiate with respect to U_{pq}

$$\begin{aligned}\frac{\partial}{\partial U_{pq}}\sum_{j,k}(L_x\Phi)_{jk}\,|U_{jk}|^2 &= \sum_{l,j\neq p}\mathcal{F}^*_{lj}b_{lq}\mathcal{F}_{lp}U^*_{pq}|U_{jq}|^2\\ &\quad + \sum_{l,m\neq p}\mathcal{F}^*_{lp}b_{lq}\mathcal{F}_{lm}|U_{mq}|^2U^*_{pq}\\ &\quad + \sum_{l}\mathcal{F}^*_{lp}b_{lq}\mathcal{F}_{lp}\left(2U_{pq}\left(U^*_{pq}\right)^2\right)\\ &= 2\sum_{l,m}\mathcal{F}^*_{lm}b_{lq}\mathcal{F}_{lp}|U_{mq}|^2U^*_{pq}\\ &= 2\,(L_x\Phi)^*_{pq}\,U^*_{pq}.\end{aligned}$$

But $L_x\Phi$ is real so we get the desired result. In the case $\beta = -1$ the linear transformation which describes $L_x\Phi$ is more complicated but the same result holds.

Finally, we will show that the l_2-norm of the semi-discrete DS system is conserved. Multiply (3.7) by U_{jk}^* and (3.8) by U_{jk} and add to get

$$U_{jk}^* \frac{d}{dt} U_{jk} + U_{jk} \frac{d}{dt} U_{jk}^* = i \left[U_{jk}^* (LU)_{jk} - U_{jk} (LU)_{jk}^* \right].$$

Summing over j and k yields

$$\frac{d}{dt} \sum_{j=1}^{n} \sum_{k=1}^{n} |U_{jk}|^2 = i \sum_{j=1}^{n} \sum_{k=1}^{n} \left[U_{jk}^* (LU)_{jk} - U_{jk} (LU)_{jk}^* \right].$$

If $\vec{w}$ is a column vector and L_{xx} is self adjoint as in (3.11), then

$$\begin{aligned} \sum_{l=1}^{n} \left[w_l^* (L_{xx}\vec{w})_l - w_l (L_{xx}\vec{w})_l^* \right] &= \vec{w}^H L_{xx} \vec{w} - (L_{xx}\vec{w})^H \vec{w} \\ &= 0. \end{aligned}$$

Similarly

$$\sum_{l=1}^{n} \left[w_l^* (L_{yy}\vec{w})_l - w_l (L_{yy}\vec{w})_l^* \right] = 0.$$

Thus

$$\frac{d}{dt} \sum_{j=1}^{n} \sum_{k=1}^{n} |U_{jk}|^2 = 0. \tag{3.22}$$

3.3 Split-Step Method

In this section we introduce the split-step Fourier method. The split-step method is shown to be second order accurate in time when applied to the DS system in the case where $\beta = 1$ in (3.6). It is shown to be only a first order method in the case $\beta = -1$. We then discuss the computational complexity of the scheme and show that the split-step method conserves the l_2-norm of the solution.

Consider the system of differential equations,

$$\frac{d}{dt} U = (L + N(U))\, U, \tag{3.23}$$

where the linear and nonlinear operators, L and N, operate on the complex vector valued functions $U(t)$. Discretize the time variable and denote

$$U^m = U(t_m),$$

where $U(t_0)$ is a given initial condition, $t_m = t_0 + m\tau$ and τ is an increment of time.

The time integration method that will be considered is a split-step method which has been discussed in [25] and [26]. See also the discussion and other references in [28]. The split-step method takes the form of

$$\begin{aligned} W^m &= \exp\left(\frac{1}{2}\tau N(U^m)\right) U^m, \\ \widetilde{W}^m &= \exp(\tau L)\, W^m, \\ U^{m+1} &= \exp\left(\frac{1}{2}\tau N(\widetilde{W}^m)\right) \widetilde{W}^m. \end{aligned} \tag{3.24}$$

In the case of the semi-discrete DS system the linear and nonlinear terms, L and N, are given by (3.4) and (3.5). If Fourier differentiation is used for the linear term then, as in (3.2),

$$L_{xx} = \mathcal{F}^H E_{xx} \mathcal{F}.$$

Thus

$$\exp(\tau L_{xx}) = \mathcal{F}^H \exp(\tau E_{xx}) \mathcal{F}, \tag{3.25}$$

where

$$(\exp(\tau E_{xx}))_{jk} = \exp(\tau (E_{xx})_{jk}).$$

The nonlinear step is calculated as follows

$$\left[\exp\left(\frac{1}{2}\tau N(U^m)\right) U\right]_{jk} = \exp\left(\frac{1}{2}\tau N(U^m)_{jk}\right) U_{jk}.$$

We will discuss in detail the inversion of (3.6) to solve for Φ in section 4.1 for the h/e and e/e cases and in section 4.2 for the e/h case. The above linear step corresponds to solving $\frac{d}{dt}U = LU$ approximately. In section 4.1 we will show that

the above nonlinear step corresponds to solving $\frac{d}{dt}U = N(U) \circ U$ exactly in the case where $\beta = 1$ in (3.6) (the h/e and e/e DS systems). In section 4.2 we show that the nonlinear step corresponds to solving $\frac{d}{dt}U = N(U) \circ U$ approximately in the case $\beta = -1$ (the e/h DS system).

To calculate the formal order of accuracy of the method when applied to the DS system, first expand U about t_m and use (3.23),

$$\begin{aligned}
U(t_m+\tau) &= U(t_m) + \tau U_t(t_m) + \frac{1}{2}\tau^2 U_{tt}(t_m) + \mathcal{O}(\tau^3) \\
&= U(t_m) + \tau\left[L + N(U(t_m))\right]U(t_m) + \frac{1}{2}\tau^2\frac{d}{dt}\left[(L+N(U))\,U\right](t_m) \\
&\quad +\mathcal{O}(\tau^3) \\
&= U(t_m) + \tau\left[L + N(U(t_m))\right]U(t_m) \\
&\quad +\frac{1}{2}\tau^2\left[\left(L + N(U(t_m))\right)U_t(t_m) + \frac{dN}{dt}(U(t_m))U(t_m)\right] \\
&\quad +\mathcal{O}(\tau^3) \\
&= U(t_m) + \tau\left[L + N(U(t_m))\right]U(t_m) \\
&\quad +\frac{1}{2}\tau^2\left[\left(L + N(U(t_m))\right)\left(L + N(U(t_m))\right) + \frac{dN}{dt}(U(t_m))\right]U(t_m) \\
&\quad +\mathcal{O}(\tau^3)
\end{aligned}$$

$$\begin{aligned}
U(t_m+\tau) &= U(t_m) + \tau\left[L + N(U(t_m))\right]U(t_m) + \frac{1}{2}\tau^2\left[\frac{dN}{dt}(U(t_m))\right. \\
&\quad \left. +L^2 + LN(U(t_m)) + N(U(t_m))L + N(U(t_m))^2\right]U(t_m) \\
&\quad +\mathcal{O}(\tau^3)
\end{aligned} \tag{3.26}$$

Note that, in general, L and N do not commute.

Next expand (3.24)

$$W^m = U^m + \frac{1}{2}\tau N(U^m)U^m + \frac{1}{8}\tau^2 N(U^m)^2 U^m + \mathcal{O}(\tau^3).$$

Thus

$$\widetilde{W}^m = W^m + \tau L W^m + \frac{1}{2}\tau^2 L^2 W^m + \mathcal{O}(\tau^3)$$

$$= U^m + \tau\left[L + \frac{1}{2}N(U^m)\right]U^m + \frac{1}{2}\tau^2\left[L^2 + LN(U^n) + \frac{1}{4}N(U^m)^2\right]U^m + \mathcal{O}(\tau^3), \tag{3.27}$$

which leads to

$$\begin{aligned} U^{m+1} &= \left[\mathcal{I} + \frac{1}{2}\tau N(\widetilde{W}^m) + \frac{1}{8}\tau^2 N(\widetilde{W}^m)^2\right]\widetilde{W}^m + \mathcal{O}(\tau^3) \\ &= U^m + \tau\left[L + \frac{1}{2}N(U^m) + \frac{1}{2}N(\widetilde{W}^m)\right]U^m + \frac{1}{2}\tau^2\Big[L^2 + LN(U^m) \\ &\quad + \frac{1}{4}N(U^m)^2 + N(\widetilde{W}^m)\left(L + \frac{1}{2}N(U^m)\right) + \frac{1}{4}N(\widetilde{W}^m)^2\Big]U^m \\ &\quad + \mathcal{O}(\tau^3). \end{aligned} \tag{3.28}$$

At this point it is necessary to know the form of $N(U)$, namely $N(U) = i\alpha_3 U \circ U^* + i\alpha_4 L_x \Phi$, where Φ is a linear transformation of $U \circ U^*$. Thus, $N(U)$ is a linear transformation of the nonlinear quantity $U \circ U^*$. Let Ψ denote this transformation matrix, i.e. $N(U) = \Psi(U \circ U^*)$. Then with the use of (3.23) we get

$$\begin{aligned} \frac{dN}{dt}(U) &= \Psi(U_t \circ U^* + U \circ U_t^*) \\ &= \Psi(LU \circ U^* + U \circ LU^* + 2N(U)U \circ U^*). \end{aligned} \tag{3.29}$$

From (3.27)

$$\begin{aligned} N(\widetilde{W}^m) &= \Psi\left(\widetilde{W}^m \circ (\widetilde{W}^m)^*\right) \\ &= \Psi\Big(U^m \circ (U^m)^* + \frac{1}{2}\tau\{LU^m \circ (U^m)^* + U^m \circ L(U^m)^* \\ &\quad + 2N(U^m)U^m \circ (U^m)^*\} + \mathcal{O}(\tau^2)\Big) \\ &= N(U^m) + \frac{1}{2}\tau\frac{dN}{dt}(U^m) + \mathcal{O}(\tau^2). \end{aligned} \tag{3.30}$$

Then

$$U^{m+1} = U^m + \tau\left[L + N(U^m)\right]U^m + \frac{1}{2}\tau^2\Big[\frac{dN}{dt}(U^m) + L^2 + LN(U^m) + N(U^m)L + N(U^m)^2\Big]U^m + \mathcal{O}(\tau^3). \tag{3.31}$$

Comparison of (3.26) and (3.31) indicates that $U(t_m + \tau) = U^{m+1} + \mathcal{O}(\tau^3)$, or that the scheme is accurate to second order. In the case where $\beta = -1$, $N(U)$ is solved only up to first order in τ, so the method is also first order accurate in this case. Similarly

$$\begin{aligned} W^m &= \exp\left(\frac{1}{2}\tau L\right) U^m, \\ \widetilde{W}^m &= \exp\left(\tau N(W^m)\right) W^m, \\ U^{m+1} &= \exp\left(\frac{1}{2}\tau L\right) \widetilde{W}^m, \end{aligned} \tag{3.32}$$

is accurate to second order when $\beta = 1$ and first order for $\beta = -1$.

Computationally, the scheme (3.24) requires calculating $N(U)$ and exponentiating $2j$ times per j iterations. If Fourier derivatives are used, then $2j$ two dimensional Fast Fourier Transforms (FFT2) per j iterations are also needed for the linear step. Note that if $j > 1$ iterations are to be performed between outputs, then the second method (3.32) reduces to

$$\begin{aligned} &W^n = \exp\left(\tfrac{1}{2}\tau L\right) U^n, \\ &\widetilde{W}^n = \exp\left(\tau N(W^n)\right) W^n, \\ &\text{for } k = 1, j-1 \\ &\quad W^{n+k} = \exp\left(\tau L\right) \widetilde{W}^{n+k-1}, \\ &\quad \widetilde{W}^{n+k} = \exp\left(\tau N(W^{n+k})\right) W^{n+k}, \\ &\quad \text{end} \\ &U^{n+j} = \exp\left(\tfrac{1}{2}\tau L\right) \widetilde{W}^{n+j-1}, \end{aligned}$$

which requires calculating $N(U)$ and exponentiating j times and $2j + 2$ FFT2's. This concatenation of the linear terms can cause a significant savings in computation time in the case of the DS system where, as we will see in the next chapter, calculating $N(U)$ requires two or more FFT2's.

This amalgamation is due to McLachlan [20], [21], and was not known to this author until after the simulations of solitons and dromions had been conducted.

The simulations in Chapter 4 where conducted using (3.24) and later it was verified that (3.32) produced the same results in two-thirds the time.

To see that the split-step method conserves the l_2-norm of the solution we proceed as follows. First recall the form of the semi-discrete DS system

$$\frac{d}{dt}U = iLU + iN(U) \circ U,$$

$$L = \alpha_1 L_{xx} + \alpha_2 L_{yy} = \mathcal{F}^{-1}(E\circ)\mathcal{F}, \quad \text{and} \quad N(U) = \alpha_3 U \circ U^* + \alpha_4 L_x \Phi,$$

where

$$L_{xx}\Phi + \beta L_{yy}\Phi = L_x(U \circ U^*),$$

and E is a real, constant matrix. Then the split-step method becomes

$$\begin{aligned} W_{jk}^m &= \exp\left(\frac{1}{2}\tau i N(U^m)_{jk}\right) U_{jk}^m \\ \widetilde{W}^m &= \mathcal{F}^H \exp(\tau i E) \circ \mathcal{F} W^m \\ U_{jk}^{m+1} &= \exp\left(\frac{1}{2}\tau i N(\widetilde{W})_{jk}\right) \widetilde{W}_{jk}^m. \end{aligned}$$

In the first step, since $N(U)$ is real, we get

$$\begin{aligned} W_{jk}^m \left(W_{jk}^m\right)^* &= \exp\left(\frac{1}{2}\tau i N(U^m)_{jk}\right) U_{jk}^m \exp\left(-\frac{1}{2}\tau i N(U^m)_{jk}\right) \left(U_{jk}^m\right)^* \\ &= U_{jk}^m \left(U_{jk}^m\right)^*. \end{aligned}$$

Thus $\|W^m\|_{l_2} = \|U^m\|_{l_2}$. Similarly, $\|U^{m+1}\|_{l_2} = \|\widetilde{W}^m\|_{l_2}$. In the second step we use the fact that $\mathcal{F}$ is unitary and E is real to get $\|\widetilde{W}^m\|_{l_2} = \|W^m\|_{l_2}$. Therefore, the split-step method conserves the l_2-norm of the numerical solution of the semi-discrete DS system.

In the next section we show that the split-step method is symplectic when applied to the semi-discrete DS system.

3.4 Symplectic Transformations

3.4.1 Symplecticness of the Split-Step Method

In this section we will show that the split-step scheme is symplectic when applied to the DS system. A transformation, $W = MV$ where M is a $2n \times 2n$ matrix is called symplectic if

$$M^H J M = J \quad \text{where} \quad J = \begin{pmatrix} 0 & -I \\ I & 0 \end{pmatrix}.$$

First consider the easier one dimensional case of applying the split-step method to the NLS equation

$$u_t = iu_{xx} + 2i|u|^2 u, \tag{3.33}$$

where $u = u(x,t)$ is complex valued. Let $u = p + iq$, where $p = p(x,t)$ and $q = q(x,t)$ are real valued, then the NLS can be written as a system

$$\begin{aligned} p_t &= -q_{xx} - 2(p^2+q^2)q \\ q_t &= p_{xx} + 2(p^2+q^2)p. \end{aligned} \tag{3.34}$$

This system has a linear and a nonlinear part. Consider solving the linear part of this system:

$$\begin{aligned} p_t &= -q_{xx} \\ q_t &= p_{xx}. \end{aligned} \tag{3.35}$$

This can be discretized as

$$\frac{d\vec{v}}{dt} = \begin{pmatrix} 0 & -L_{xx} \\ L_{xx} & 0 \end{pmatrix} \vec{v},$$

where

$$\vec{v} = (p_1, \ldots, p_n, q_1, \ldots, q_n)^T \qquad \text{and} \qquad L_{xx} = \mathcal{F}^H E_{xx} \mathcal{F}.$$

Here L_{xx} is the $n \times n$ second derivative matrix. Advancing the linear part via the split-step method becomes

$$\vec{v}^{\,l+1} = \exp(\tau D)\vec{v}^{\,l}, \tag{3.36}$$

where

$$\begin{aligned} D &= \begin{pmatrix} 0 & -L_{xx} \\ L_{xx} & 0 \end{pmatrix} \\ &= \begin{pmatrix} \mathcal{F}^H & 0 \\ 0 & \mathcal{F}^H \end{pmatrix} \begin{pmatrix} 0 & -E_{xx} \\ E_{xx} & 0 \end{pmatrix} \begin{pmatrix} \mathcal{F} & 0 \\ 0 & \mathcal{F} \end{pmatrix}. \end{aligned} \tag{3.37}$$

Thus, the linear transformation matrix for advancing the linear part is

$$\exp(\tau D) = F^H \mathcal{E} F$$

where

$$F = \begin{pmatrix} \mathcal{F} & 0 \\ 0 & \mathcal{F} \end{pmatrix} \quad \text{and} \quad \mathcal{E} = \exp \begin{pmatrix} 0 & -E_{xx} \\ E_{xx} & 0 \end{pmatrix}.$$

If A is any real diagonal $n \times n$ matrix, then basic calculations give

$$\exp \begin{pmatrix} 0 & -A \\ A & 0 \end{pmatrix} = \begin{pmatrix} \cos(A) & -\sin(A) \\ \sin(A) & \cos(A) \end{pmatrix}$$

and

$$\exp \begin{pmatrix} 0 & -A \\ A & 0 \end{pmatrix}^H J \exp \begin{pmatrix} 0 & -A \\ A & 0 \end{pmatrix} = J.$$

Thus

$$\exp \begin{pmatrix} 0 & -A \\ A & 0 \end{pmatrix} \tag{3.38}$$

is symplectic for A any real diagonal $n \times n$ matrix. Likewise, a direct calculation gives

$$F^H J F = J.$$

Similarly $F J F^H = J$.

To show that advancing the linear part in the split-step method is symplectic, look at

$$\begin{aligned} \exp(\tau D)^H J \exp(\tau D) &= \left(F^H \mathcal{E} F\right)^H J F^H \mathcal{E} F \\ &= J. \end{aligned} \tag{3.39}$$

Now consider the nonlinear part of the NLS equation

$$u_t = 2i|u|^2 u.$$

Let $u = p + iq$ as before to get

$$\begin{aligned} p_t &= -2(p^2+q^2)q \\ q_t &= 2(p^2+q^2)p. \end{aligned} \tag{3.40}$$

Note that

$$pp_t + qq_t = 0,$$

which implies

$$\frac{d}{dt}(p^2+q^2) = 0 \quad \text{or} \quad p^2+q^2 = c,$$

where c is independent of t. Thus (3.40) becomes

$$\begin{aligned} p_t &= -2cq \\ q_t &= 2cp, \end{aligned} \tag{3.41}$$

which can be discretized as

$$\frac{d\vec{v}}{dt} = 2c \begin{pmatrix} 0 & -I \\ I & 0 \end{pmatrix} \vec{v},$$

where $\vec{v}$ is as above. Now advancing the nonlinear part becomes

$$\vec{v}^{l+1} = \exp \begin{pmatrix} 0 & -2c\tau I \\ 2c\tau I & 0 \end{pmatrix} \vec{v}^{l}.$$

In this case the transformation matrix

$$\exp \begin{pmatrix} 0 & -2c\tau I \\ 2c\tau I & 0 \end{pmatrix}$$

is of the form in (3.38) and is therefore symplectic.

Since the composition of symplectic maps is symplectic, the split-step scheme is a symplectic integrator when applied to the one dimensional NLS equations.

To apply the split-step method to the DS system, consider the linear and nonlinear parts

$$u_t = i\alpha_1 u_{xx} + i\alpha_2 u_{yy}$$

and

$$\begin{aligned} u_t &= i\alpha_3 |u|^2 u + i\alpha_4 \phi_x u \\ \phi_{xx} &+ \beta\phi_{yy} = (|u|^2)_x. \end{aligned} \tag{3.42}$$

Advancing the linear part of the DS system and showing that the map which advances the linear part is symplectic follows from two dimensional arguments analogous to the above one dimensional arguments. In the nonlinear part write $u = p + iq$ where p and q are real functions of x, y and t. Then the nonlinear part can be written as

$$\begin{aligned} p_t &= -\left(\alpha_3(p^2+q^2) + \alpha_4\phi_x\right) q \\ q_t &= \left(\alpha_3(p^2+q^2) + \alpha_4\phi_x\right) p. \end{aligned} \tag{3.43}$$

Multiply the first equation by p and the second by q and add to get

$$\frac{d}{dt}(p^2+q^2) = 0, \quad \text{or} \quad p^2+q^2 = c,$$

where c is independent of t. Since $|u|^2$ is independent of t, the second equation in (3.42) implies that ϕ_x is independent of t provided that the boundary conditions on ϕ are time independent. In the h/e and e/e DS systems we assume that the boundary conditions are independent of t, so (3.43) becomes

$$\begin{aligned} p_t &= -\tilde{c}q \\ q_t &= \tilde{c}p, \end{aligned} \tag{3.44}$$

where $\tilde{c}$ is independent of time. Thus, the map which advances the nonlinear part of the DS system and its symplecticness follows in a similar manner to the one dimensional NLS.

In the e/h DS system the boundary conditions can depend on time. It is not clear how this effects the symplecticness of the split-step method. However, in section 4.2 we will see that the split-step method accurately simulates known dromion solutions of the e/h DS system.

3.4.2 Why Use a Symplectic Integrator?

Why is it preferable to use a symplectic integration method over a non-symplectic one? To answer this question, consider the one dimensional NLS equation instead of the more computationally complicated DS system. The NLS equation (3.33) can be approximated by the system of ordinary differential equations

$$\frac{d}{dt}u_j = i\left((L_{xx}\vec{u})_j + 2|u_j|^2 u_j\right), \tag{3.45}$$

where the computational domain, $[-\frac{1}{2}P, \frac{1}{2}P]$, is discretized by $\Delta x = \frac{P}{N}$, $x_j = -\frac{1}{2}P + j\Delta x$ and $u_j(t) \approx u(x_j, t)$ the approximation of the solution to the NLS equation, which is assumed to be periodic. This is basically the same as (1.3) except that now we are using Fourier transforms rather than finite differences to approximate the derivatives. With this discretization we showed that a standard Runge-Kutta method does not do a good job of simulating recurrence. We now give an example of how the split-step method does simulate recurrence with this discretization.

The software package MATLAB was used to solve the system (3.45) for an example corresponding to the recurrence phenomenon discussed in section 1.2 with two unstable modes in the region (1.10). The parameters used were $P = 4\sqrt{2}\pi$, $N = 32$, $\tau = 0.01$ and an initial condition $u(x, 0) = 0.5\,(1 + 0.05\cos(2\pi x/P))$.

Both the second order split-step method and MATLAB'S ode23 routine were used to approximate the solution to the NLS equations with the above parameters. MATLAB'S ode23 routine uses second and third order Runge-Kutta integrators with an adaptive step-size to solve initial value problems. Figure 7 shows the

numerical solutions generated by the symplectic split-step method and the non-symplectic Runge-Kutta method. Figure 8 shows contour plots of Figure 7. Note that the Runge-Kutta method has lost the symmetry of the solution by $t = 80$. The symplectic split-step method, by contrast, maintained the recurrent behavior and the symmetry of the solution throughout the simulation.

In this case the split-step method proved superior to the Runge-Kutta method because it preserves the symplectic structure of the solution. The split-step method does this by solving the linear and nonlinear parts of the NLS separately and in the case of the NLS, the nonlinear part is solved exactly. The Runge-Kutta method is not designed to solve the nonlinear part accurately with this discretization and thus it fails to simulate the nonlinear effects as well as the split-step method. In this example, it appears that it is not necessary to have an integrable discretization of the NLS to avoid numerical chaos. The non-integrable discretization of the NLS does exhibit the nonlinear phenomenon of recurrence in a long-time integration provided that the numerical method applied is symplectic. For this reason, we will use the split-step method to integrate the semi-discrete DS system (3.3)–(3.6).

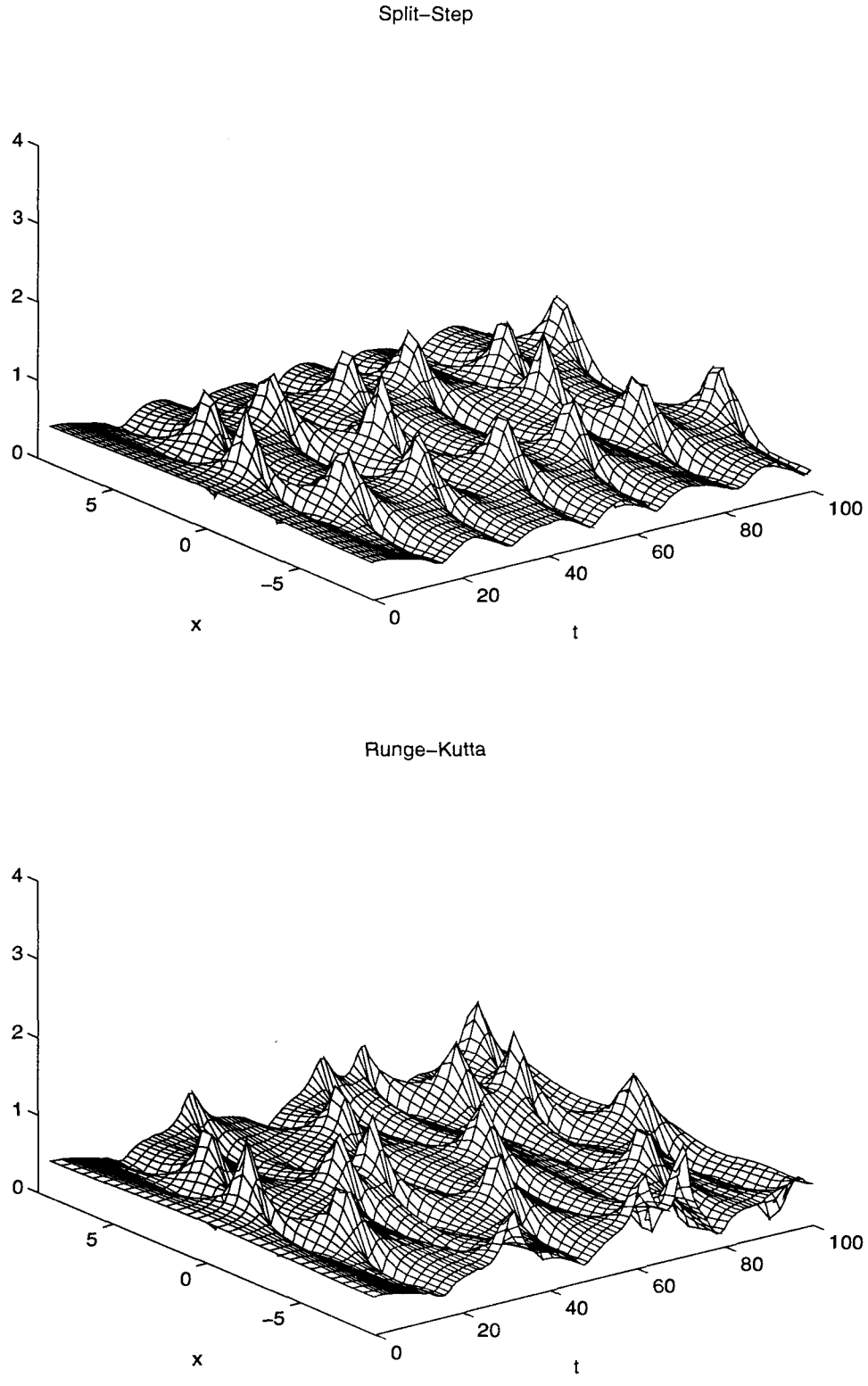

Figure 7. Solution of NLS equation: $P = 4\sqrt{2}\pi$, $N = 32$, $q = 2$, $\tau = 0.1$ in split-step method, $u(x, 0) = 0.5\,(1 + 0.05\cos(2\pi x/P))$. $|u|$ is shown for $0 \leq t \leq 100$.

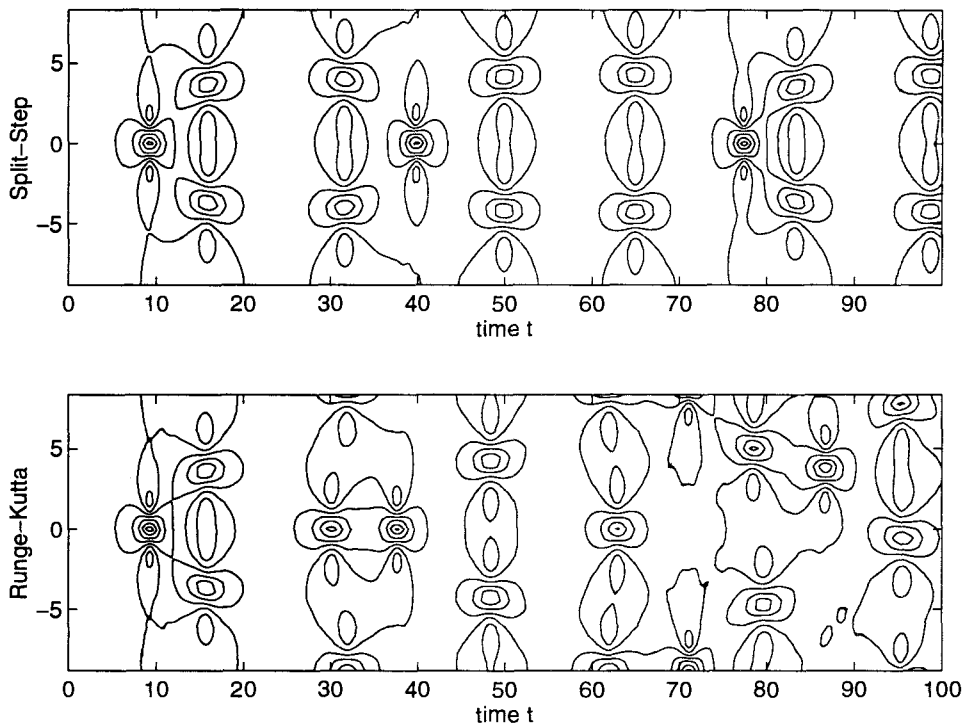

Figure 8. Contour plots of solution of NLS equation.

Chapter 4

Solitons

In this chapter soliton and soliton-like solutions of the DS system are discussed. We begin by showing how the split-step method can be applied to the DSII system. The DSII system is first split into linear and nonlinear parts. The inversion of the second equation in the DSII system is then formally derived and the split-step method is shown to be equivalent to solving a linear and a nonlinear problem separately. Finally the method is used to simulate a rational soliton solution of the DSII system.

Next, the DSI system is studied. The DSI system is transformed into characteristic coordinates so that the second equation in the DSI system can be inverted. The solution to the nonlinear part of the DSI system is then discussed and examples of dromion simulations are presented.

These simulations represent what we believe to be the first numerically generated soliton and dromion solutions to the DS system and will appear in [29].

4.1 DSII System

4.1.1 Applying the Split-Step Method

Recall the DSII equations (1.1), with $\sigma^2 = -1$,

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

where

$$\mathcal{L}u = \frac{i}{2}\left(u_{xx} - u_{yy}\right), \qquad \mathcal{N}(u) = i\left(\alpha|u|^2 - \phi_x\right),$$

and ϕ satisfies

$$\phi_{xx} + \phi_{yy} = 2\alpha\left(|u|^2\right)_x. \tag{4.1}$$

In order to use the split-step method to calculate an approximation to the solution of the DSII equation, we must solve for ϕ_x at each time step. Thus, it is necessary to invert the operator $\frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2}$.

In this case a spectral method can be used to calculate ϕ up to a constant. Write ϕ as

$$\phi(x,y,t) = \sum_{j=-\infty}^{\infty}\sum_{k=-\infty}^{\infty} \widehat{\phi}_{jk}(t)e^{i(\mu_j x+\nu_k y)},$$

where $\mu_j = \frac{2\pi j}{P}$, $\nu_k = \frac{2\pi k}{P}$ and x and y are restricted to intervals of length P. If $|u|^2$ is also written as a Fourier expansion,

$$|u|^2 = \sum_{j=-\infty}^{\infty}\sum_{k=-\infty}^{\infty} \widehat{u}_{jk}(t)e^{i(\mu_j x+\nu_k y)},$$

then (4.1) becomes

$$\sum_{j=-\infty}^{\infty}\sum_{k=-\infty}^{\infty}\left(-\mu_j^2 - \nu_k^2\right)\widehat{\phi}_{jk}e^{i(\mu_j x+\nu_k y)} = \sum_{j=-\infty}^{\infty}\sum_{k=-\infty}^{\infty} i\mu_j\widehat{u}_{jk}e^{i(\mu_j x+\nu_k y)}.$$

Thus, for j and k such that $\mu_j^2 + \nu_k^2 \neq 0$, $\widehat{\phi}_{jk}$ is given by

$$\widehat{\phi}_{jk} = \frac{-i\mu_j}{\mu_j^2 + \nu_k^2}\widehat{u}_{jk}.$$

This determines ϕ up to an additive constant. Thus,

$$\phi_x(x,y,t) = \sum_{j=-\infty}^{\infty}\sum_{k=-\infty}^{\infty} b_{jk}(t)e^{i(\mu_j x+\nu_k y)}$$

where

$$b_{jk} = \begin{cases} 0, & \text{if } j = 0 \text{ and } k = 0 \\ \frac{\mu_j^2}{\mu_j^2+\nu_k^2}\widehat{u}_{jk}, & \text{otherwise.} \end{cases}$$

Now applying the split-step method to the DSII system corresponds to solving

$$iu_t + \frac{1}{2}(u_{yy} - u_{xx}) = 0, \tag{4.2}$$

and

$$\begin{aligned} iu_t + \alpha|u|^2 u - u\phi_x &= 0 \\ \phi_{xx} + \phi_{yy} - 2\alpha\left(|u|^2\right)_x &= 0, \end{aligned} \tag{4.3}$$

the linear and nonlinear parts of the DSII system. Thus, (3.24) corresponds to time integrations of (4.3) over a half time step, then integration of (4.2) over a full time step and finally, integration of (4.3) over a half time step. To see this, first consider (4.2). Since we assume that $u \to 0$ and $\phi \to 0$ as $x^2 + y^2 \to \infty$, the computational domain can be restricted to $(x, y) \in [-\frac{1}{2}P, \frac{1}{2}P] \times [-\frac{1}{2}P, \frac{1}{2}P]$, where P is sufficiently large so that u, ϕ and their derivatives may be accurately approximated by finite Fourier series. Let

$$U_{jk} = \sum_m \sum_n \hat{U}_{mn} e^{i(\mu_m x_j + \nu_n y_k)}, \tag{4.4}$$

where $U_{jk} \approx u(x_j, y_k)$,

$$x_j = -\frac{1}{2}P + j\Delta x, \quad y_k = -\frac{1}{2}P + k\Delta y, \quad \Delta x = \Delta y = \frac{P}{2N},$$

and

$$\mu_m = m\Delta\mu, \quad \nu_n = n\Delta\nu, \quad \Delta\mu = \Delta\nu = \frac{2\pi}{P}.$$

The indices m, n, j, k range over $-N$ to $N-1$. (Note that it is possible to use a rectangular rather than a square grid). The equivalent equation in Fourier space to (4.2) is then

$$i\frac{d}{dt}\hat{U}_{mn} + \frac{1}{2}\left(\mu_m^2 - \nu_n^2\right)\hat{U}_{mn} = 0,$$

which has solution

$$\hat{U}_{mn}(t_l + \tau) = \exp\left(\frac{1}{2}i\left(\mu_m^2 - \nu_n^2\right)\Delta t\right)\hat{U}_{mn}(t_l). \tag{4.5}$$

Thus, the linear part of the split-step method is computed by transforming $U_{jk}(t_l)$ into Fourier space thereby giving the values of $\hat{U}_{mn}$ in (4.4), then advancing the solution in Fourier space via (4.5) and finally, the solution $U_{jk}(t_l + \tau)$ is given by an inverse Fourier transform.

To solve the nonlinear problem, the second equation in (4.3) is solved for ϕ_x. To solve the first equation in (4.3), we use a method which is similar to the approach used in [24] for finding the strong coupling limit of the DS equations. Let $u = r\exp(i\theta)$, where r and θ are real functions of x, y and t. Then (4.3) yields

$$r_t = 0, \quad \text{and} \quad \theta_t = \alpha r^2 - \phi_x.$$

The first equation implies that $r = r_0(x, y)$ is independent of time. The second equation in (4.3) now becomes $\phi_{xx} + \phi_{yy} - 2\alpha(r_0^2)_x = 0$, so ϕ is also time-independent. Thus,

$$\theta = (\alpha r_0^2 - \phi_x)t + \theta_0$$

and the solution of (4.3) over one time-step becomes

$$u(x, y, t_l + \tau) = u(x, y, t_l) \exp\left(i\left(\alpha|u(x, y, t_l)|^2 - \phi_x(x, y, t_l)\right)\tau\right).$$

Notice that in the above, (4.3) is solved exactly and (4.2) is solved up to the number of Fourier modes included.

4.1.2 Numerical Test for DSII

As a test of the split-step method applied to the DSII system, consider the rational one-soliton solutions derived in [7] and also reproduced in [1]

$$u(x, y, t) = \frac{2\nu \exp\{\lambda(x + iy) - \lambda^*(x - iy) - i(\lambda^2 + (\lambda^*)^2)t\}}{|(x + iy) + \mu - 2i\lambda t|^2 + |\nu|^2},$$

where λ, μ and ν are arbitrary (complex) constants. Setting these constants all to 1 gives a solution to the DSII system of

$$u(x, y, t) = \frac{2\exp(2i(y - t))}{1 + (x + 1)^2 + (y - 2t)^2}. \tag{4.6}$$

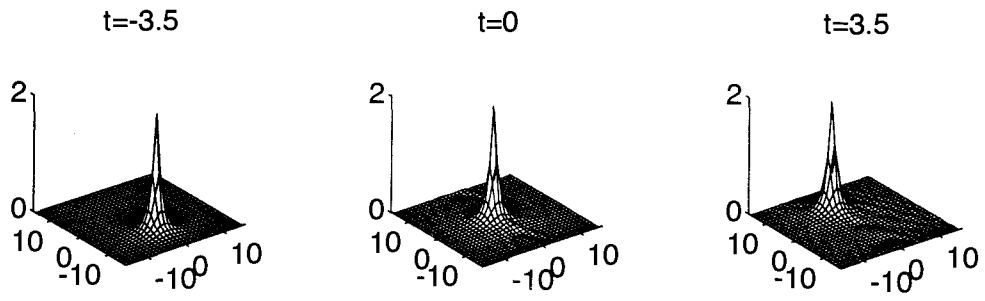

Figure 9. Numerical simulation of the rational 1-soliton of the DSII system, as computed by the split-step method. Here $N = 64$, $\Delta x = \Delta y = 0.5$, $x, y \in [-16, 16]$, $\tau = 0.01$.

Using $t = -3.5$ in (4.6) as an initial condition for the split step method, Figure 9 shows $|u|$, as a function of x and y, which is a single hump traveling with speed 2 along the line $x = -1$. For this simulation the computational domain was $[-16, 16] \times [-16, 16]$, 64 grid points were used in both the x and y directions, i.e., $\Delta x = \Delta y = 0.5$, and the time step was $\tau = 0.01$. Even on this coarse grid, the computed solution simulated the true solution to an acceptable degree. Figure 10 shows contour maps of the simulated and the true solution. Note that the speed of the soliton is also approximated accurately.

Rational solitons of this form are notoriously hard to simulate numerically, even in the case of one space dimension. As an example, such solitons arise in the Benjamin-Ono equation; see [18]. The difficulty in simulating rational solitons arises because of the slow rate of decay as $|x| \to \infty$. A much larger computational domain is required than in the case of exponentially decaying soliton solutions. This causes a lack of resolution. However, the scheme used here did not have any difficulty in simulating the true solution accurately.

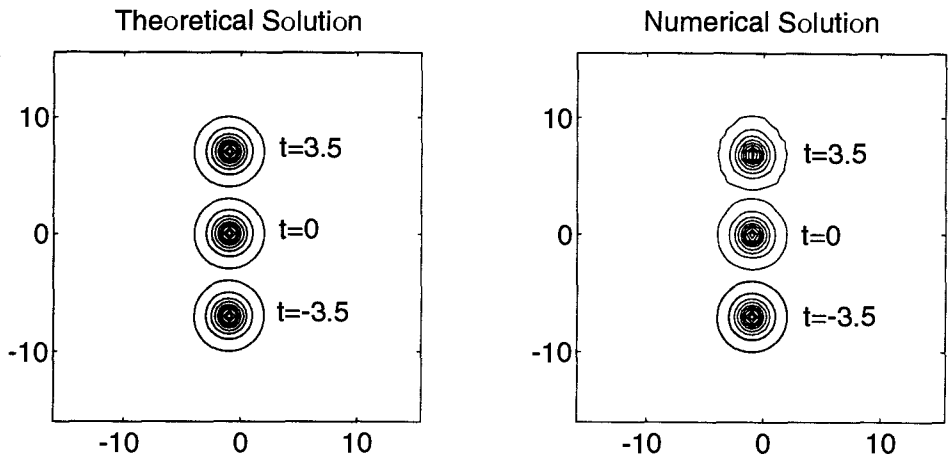

Figure 10. Comparison of the theoretical and numerical solutions of the rational soliton. Level curves of $|u|$ are shown.

As another check on the split-step scheme, the Hamiltonian (3.21) and the l_2-norm for the numerical solution were calculated. The split-step scheme preserved the l_2-norm exactly (up to the accuracy of the computer). Thus, for the numerical simulation of this rational soliton, (3.22) is satisfied. The split-step scheme does not conserve the Hamiltonian exactly, but in this simulation (3.21) was conserved to at least six significant digits over $t \in [-3.5, 3.5]$.

4.2 DS I System

4.2.1 Applying the Split-Step Method

Following the convention in [14] and [23], and changing the notation of the space variables for later convenience, the DS I system is given by

$$iu_t + \frac{1}{2}(u_{\eta\eta} + u_{\xi\xi}) + \alpha|u|^2 u - u\phi_\xi = 0 \tag{4.7}$$

$$\phi_{\xi\xi} - \phi_{\eta\eta} - 2\alpha \left(|u|^2\right)_{\xi} = 0 \tag{4.8}$$

where it is assumed that $u \to 0$ as $\xi^2 + \eta^2 \to \infty$ and $\alpha = \pm 1$. To apply the split-step method to this system, the second operator (4.8) must first be inverted. To this end, change the space variables to characteristic coordinates: $x = \xi + \eta$ and $y = \xi - \eta$. So (4.7) becomes

$$iu_t + u_{xx} + u_{yy} + u\left(\alpha|u|^2 - \phi_x - \phi_y\right) = 0, \tag{4.9}$$

and (4.8) becomes

$$4\phi_{xy} - 2\alpha \left(|u|^2\right)_x - 2\alpha \left(|u|^2\right)_y = 0.$$

This in turn implies that

$$\phi_x = \frac{1}{2}\alpha \int_{-\infty}^{y} \left(|u|^2\right)_x dy + \frac{1}{2}\alpha|u|^2 + \phi_x(x, -\infty, t),$$

and

$$\phi_y = \frac{1}{2}\alpha \int_{-\infty}^{x} \left(|u|^2\right)_y dx + \frac{1}{2}\alpha|u|^2 + \phi_y(-\infty, y, t),$$

where the last term on the right of each equation represents boundary conditions which must be specified. Now (4.9) can be written in the form

$$iu_t + u_{xx} + u_{yy} - uV = 0,$$

where

$$V = \frac{1}{2}\alpha \int_{-\infty}^{y} \left(|u|^2\right)_x dy + \frac{1}{2}\alpha \int_{-\infty}^{x} \left(|u|^2\right)_y dx + \phi_x(x, -\infty, t) + \phi_y(-\infty, y, t).$$

The DS I system can therefore be written in the form

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

where

$$\mathcal{L}u = u_{xx} + u_{yy} \qquad \text{and} \qquad \mathcal{N}(u) = V.$$

In this case the nonlinear problem, $iu_t - uV = 0$, can be solved as follows: let $u = r\exp(i\theta)$ with r and θ real to get

$$r = r_0(x,y), \qquad \theta = -\int_0^t V\,dt + \theta_0.$$

Over one time step, θ may be approximated by

$$\begin{aligned}\theta(t_{l+1}) &= -\int_{t_l}^{t_{l+1}} V\,dt + \theta_0 \\ &= -V(t_l)\tau + \mathcal{O}(\tau^2) + \theta_0.\end{aligned}$$

So

$$u(x,y,t_l+\tau) = u(x,y,t_l)\exp\left(-iV(x,y,t_l)\tau\right),$$

approximately.

To calculate V, the integrals

$$\int_{-\infty}^{x} \left(|u|^2\right)_y dx \quad \text{and} \quad \int_{-\infty}^{y} \left(|u|^2\right)_x dy$$

must be approximated. Since it is assumed that $u \to 0$ as $x^2+y^2 \to \infty$, a spectral method may be used to approximate these integrals. That is, consider $[-\frac{1}{2}P, \frac{1}{2}P] \times [-\frac{1}{2}P, \frac{1}{2}P]$ to be an approximation to $\mathbb{R} \times \mathbb{R}$, where P is chosen so that u and its derivatives are sufficiently small for $|x| > \frac{1}{2}P$ and $|y| > \frac{1}{2}P$ so that u can be approximated by a finite Fourier series

$$u(x,y,t) = \sum_{m=-N}^{N-1}\sum_{n=-N}^{N-1} a_{mn}(t)e^{i\mu_m x}e^{i\nu_n y},$$

where $\mu_m = \frac{2\pi m}{P}$, $\nu_n = \frac{2\pi n}{P}$ and N is a positive integer. Then

$$\begin{aligned}\int_{-\frac{P}{2}}^{x_j} \left(|u|^2\right)_y dx &= \int_{-\frac{P}{2}}^{x_j} \left(\sum_{m=-N}^{N-1}\sum_{n=-N}^{N-1} i\nu_n a_{mn}(t)e^{i\mu_m x}e^{i\nu_n y}\right) dx \\ &= \sum_{m\neq 0}\sum_{n=-N}^{N-1} \left(i\nu_n a_{mn}e^{i\nu_n y}\int_{-\frac{P}{2}}^{x_j} e^{i\mu_m x}\,dx\right)\end{aligned}$$

$$
\begin{aligned}
&+(x_j+\frac{P}{2})\sum_{n=-N}^{N-1} i\nu_n a_{0,n}e^{i\nu_n y} \\
&= \sum_{m\neq 0}\sum_{n=-N}^{N-1}\left(\frac{\nu_n}{\mu_m}a_{mn}e^{i\nu_n y}\left[e^{i\mu_m x_j}-e^{-i\mu_m\frac{P}{2}}\right]\right) \\
&+(x_j+\frac{P}{2})\sum_{n=-N}^{N-1} i\nu_n a_{0,n}e^{i\nu_n y}\,. \qquad (4.10)
\end{aligned}
$$

Similarly

$$
\begin{aligned}
\int_{-\frac{P}{2}}^{y_k}\left(|u|^2\right)_x dy &= \sum_{m=-N}^{N-1}\sum_{n\neq 0}\left(\frac{\mu_m}{\nu_n}a_{mn}e^{i\mu_m x}\left[e^{i\nu_n y_k}-e^{-i\nu_n\frac{P}{2}}\right]\right) \\
&+(y_k+\frac{P}{2})\sum_{m=-N}^{N-1} i\mu_m a_{m,0}e^{i\mu_m x}\,. \qquad (4.11)
\end{aligned}
$$

With the boundary conditions, $\phi_x(x,-\infty,t)$ and $\phi_y(-\infty,y,t)$, now assumed to be given at $y=-\frac{1}{2}P$ and $x=-\frac{1}{2}P$ respectively, V can be calculated with three two-dimensional Fast Fourier Transforms (FFT2's) and two one-dimensional Fast Fourier Transforms (FFT1's).

4.2.2 Simulation of Dromions

The initial-boundary value problem for the DSI system can be solved using the method of inverse scattering. In [14] Fokas and Santini show that in the case of non-zero boundary conditions the DSI system can have solutions which are localized, exponential decaying structures which travel in the plane. These dromion solutions are given by the following algebraic systems, as given in [14] and [23]. Let $u_1(y,t)$ be given by

$$
u_1(y,t) = -2\frac{\partial}{\partial y}\sum_{j=1}^{L} l_j^* \exp\left[-\lambda_j^*(y+i\lambda_j^* t)\right] Y_j(y,t), \qquad (4.12)
$$

where the Y_j's solve the system

$$
Y_j + \sum_{k=1}^{L}(C^y)_{jk}Y_k = l_j \exp\left[-\lambda_j(y-i\lambda_j t)\right],
$$

and the matrix C^y is given by

$$(C^y)_{jk} = \frac{l_j l_k^*}{\lambda_j + \lambda_k^*} \exp\left[-(\lambda_j + \lambda_k^*)\left(y - i(\lambda_j - \lambda_k^*)t\right)\right].$$

Similarly $u_2(x,t)$ is given by

$$u_2(x,t) = -2\frac{\partial}{\partial x}\sum_{j=1}^{M} m_j^* \exp\left[-\mu_j^*(x + i\mu_j^* t)\right] X_j(x,t),$$

$$X_j + \sum_{k=1}^{M}(C^x)_{jk} X_k = m_j \exp\left[-\mu_j(x - i\mu_j^* t)\right],$$

$$(C^x)_{jk} = \frac{m_j m_k^*}{\mu_j + \mu_k^*} \exp\left[-(\mu_j + \mu_k^*)\left(x - i(\mu_j - \mu_k^*)t\right)\right]. \tag{4.13}$$

In (4.12) through (4.13) above l_j, m_j, λ_j, $\mu_j \in \mathbb{C}$ and $\mathrm{Re}(\lambda_j)$, $\mathrm{Re}(\mu_j) \in \mathbb{R}^+$. Then the solution to the DSI system is given by

$$u(x,y,t) = 2\sum_{j=1}^{M}\sum_{k=1}^{L} X_j(x,t) Y_k(y,t) Z_{jk}(x,y,t),$$

where Z_{jk} satisfies

$$Z_{jk} - \epsilon \sum_{r=1}^{M} A_{jr} Z_{rk} = \rho_{jk},$$

and A_{jk} is given by

$$A = \rho(I + C^y)^{-1}\left[(I + C^x)^{-1}\rho^*\right]^T,$$

where the superscript T denotes the transpose of a matrix, and the matrix ρ depends on the initial data. The above solution is called a (M,L)-dromion.

Using a combination of the software packages Mathematica and MATLAB, this author was able to solve the above system. The split-step method was applied to the DSI system using initial and boundary conditions corresponding to $\lambda_1 = 2-2i$, $\lambda_2 = 4 - 0.5i$, $l_1 = 2 + i$, $l_2 = 1 + 2i$, $\mu_1 = 1 - 1.5i$, $\mu_2 = 3 - 0.5i$, $m_1 = 1 + i$, $m_2 = 2 + 3i$, $\rho_{11} = 1 + i$, $\rho_{12} = \rho_{21} = 0$ and $\rho_{22} = 2 + 3i$ to simulate the solution represented in Figures 4–6 in [14]. With a grid size of 256 in both the x and y directions, a computational domain of $[-20,20] \times [-20,20]$ and $\tau = 0.01$, Figure

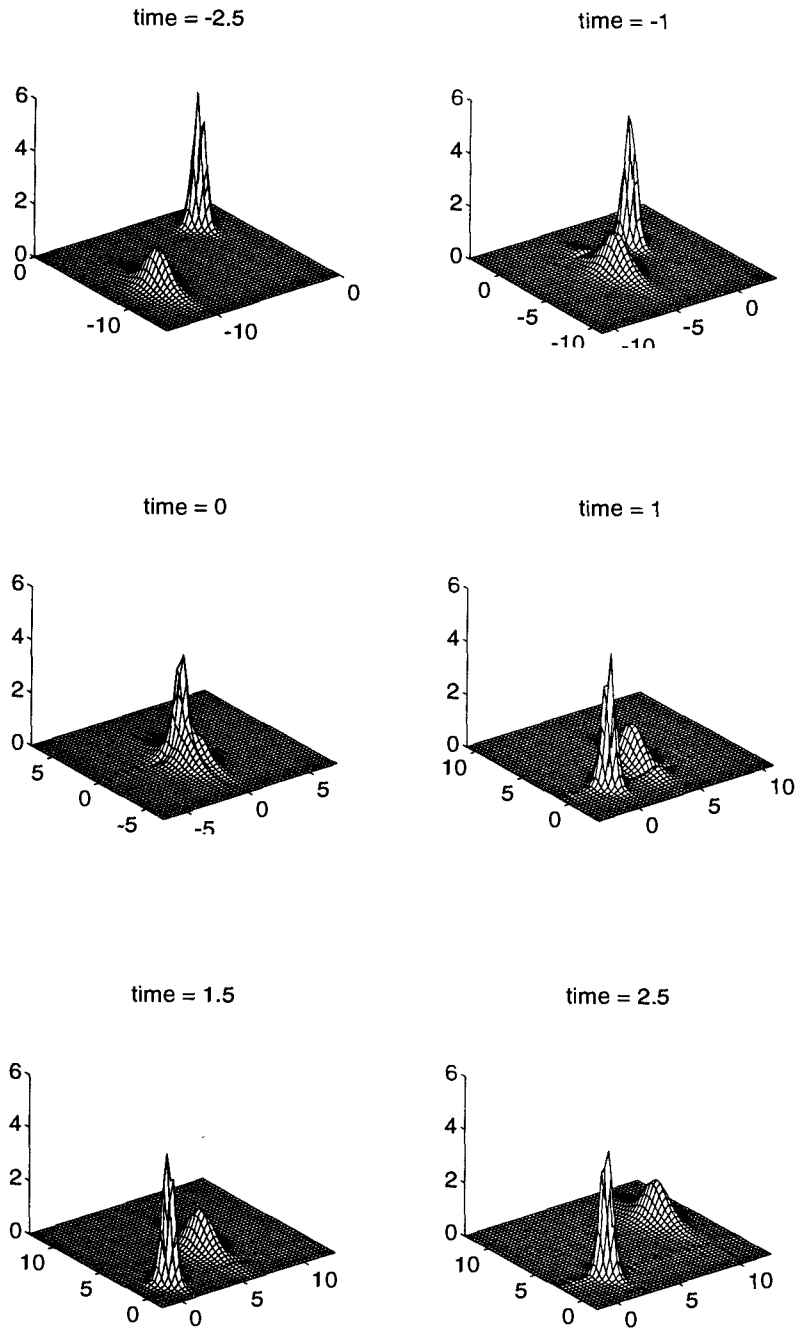

Figure 11. Numerical simulation of a (2,2)-dromion solution of the DSI system with symmetric ρ for $t = -2.5, \ldots, 2.5$.

11 shows two dromions passing through one another. As noted in [14], when $\rho_{12} = \rho_{21}$ there is no exchange of energy in the collision.

In Figure 12 contour plots of the true and simulated solutions corresponding to times $t = 0$ and $t = 2.5$ of Figure 11 are shown. From these contour plots it is apparent that the split-step method accurately simulated the overall shape and speed of these dromions.

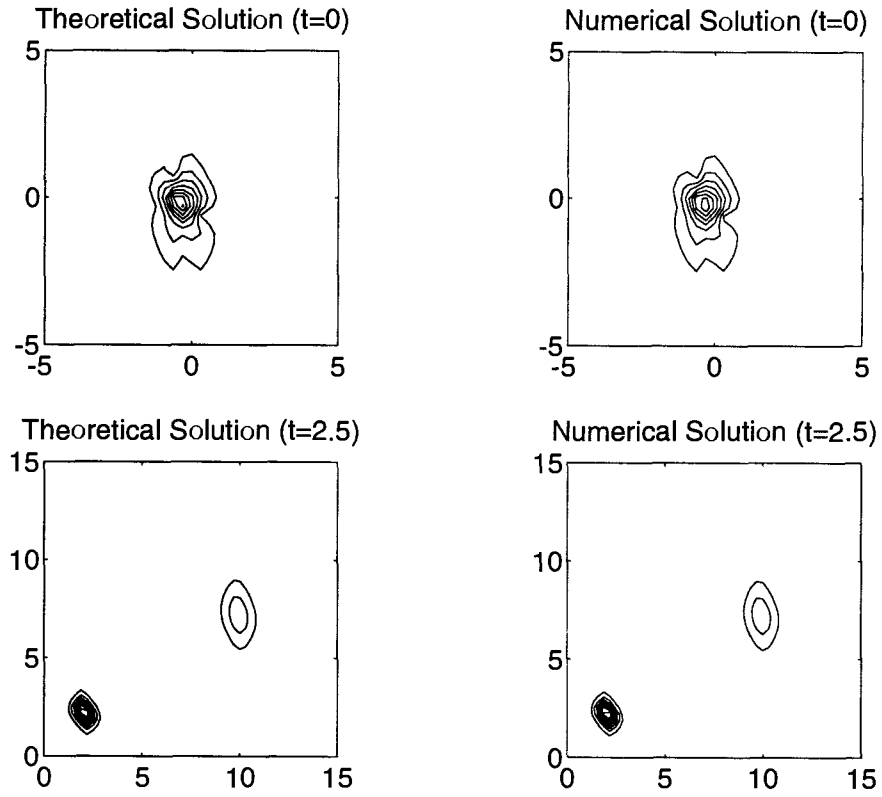

Figure 12. Comparison of the theoretical and numerical solutions of a (2,2)-dromion solution of the DSI system with symmetric ρ. Level curves of $|u|$ are shown.

As a more stringent test of the split-step method, a (2,2)-dromion with non-symmetric ρ was also simulated. With $\rho_{12} = 3 + 2i$, $\rho_{21} = 4 + i$ and the other constants the same as above, the split-step method was applied with a grid of 256 points in the x and y directions, a computational domain of $[-22, 22] \times [-22, 22]$ and $\tau = 0.01$. Figures 13 through 15 show the result of the simulation. The theoretical solution was used to generate the initial condition for the simulation which is shown in Figure 13.

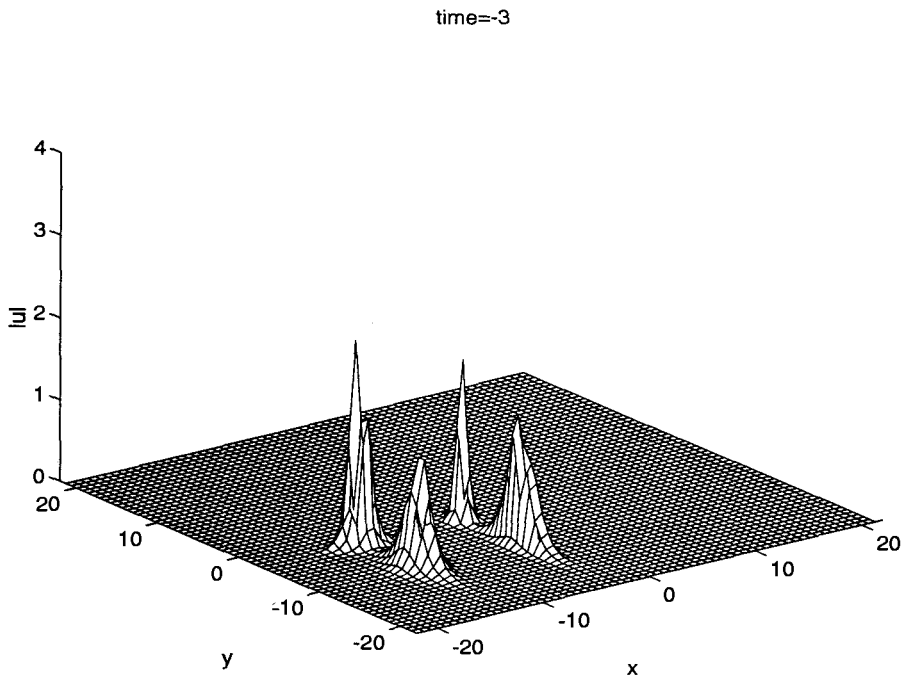

Figure 13. Numerical simulation of a (2,2)-dromion with non-symmetric ρ. $\lambda_1 = 2 - 2i$, $\lambda_2 = 4 - 0.5i$, $l_1 = 2 + i$, $l_2 = 1 + 2i$, $\mu_1 = 1 - 1.5i$, $\mu_2 = 3 - 0.5i$, $m_1 = 1 + i$, $m_2 = 2 + 3i$, $\rho_{11} = 1 + i$, $\rho_{12} = 3 + 2i$, $\rho_{21} = 4 + i$, $\rho_{22} = 2 + 3i$, at $t = -3$.

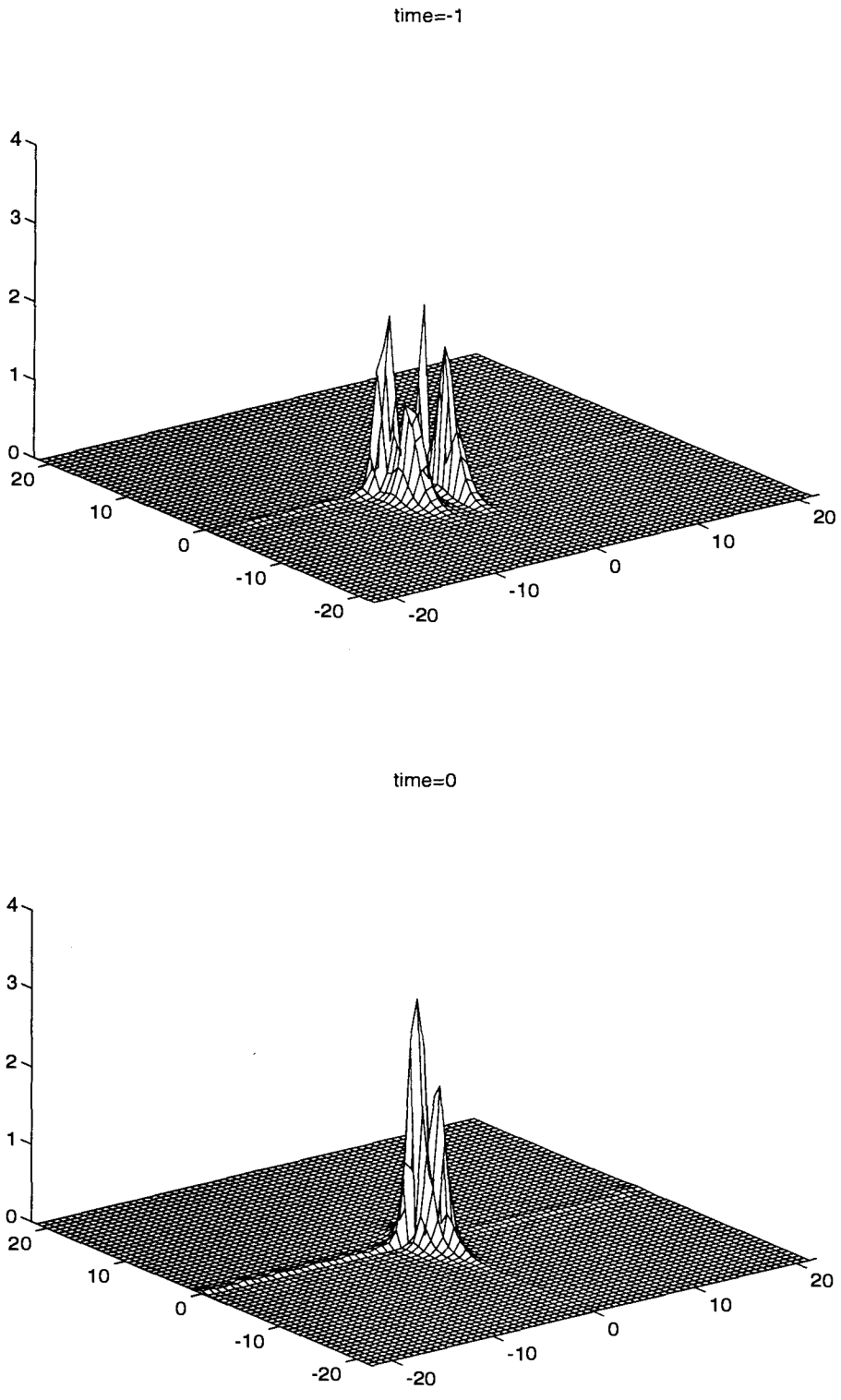

Figure 14. Continuation of Figure 13. $t = -1$ and $t = 0$.

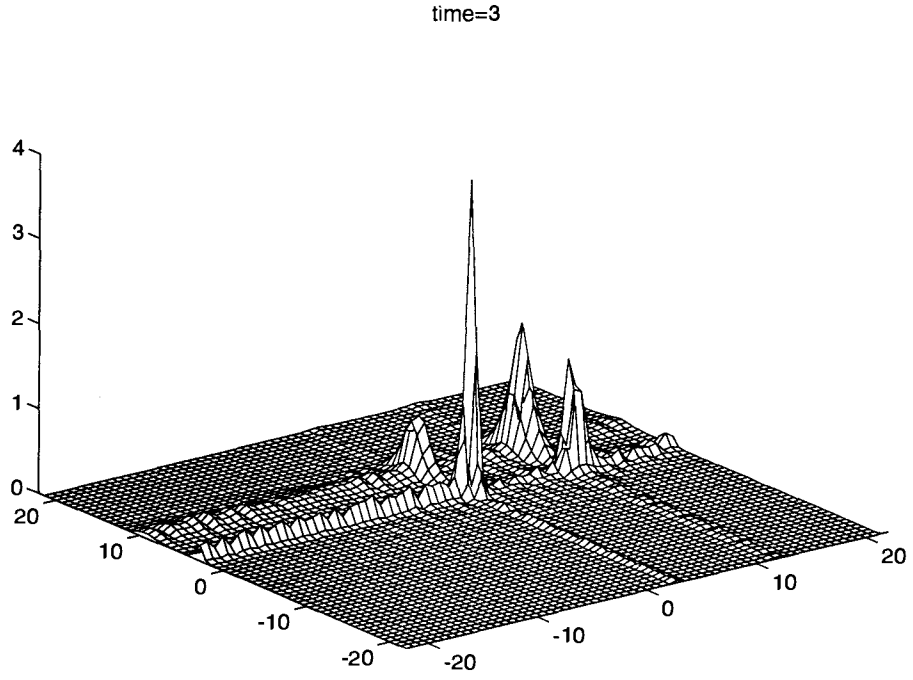

Figure 15. Continuation of Figure 14. $t = 3$.

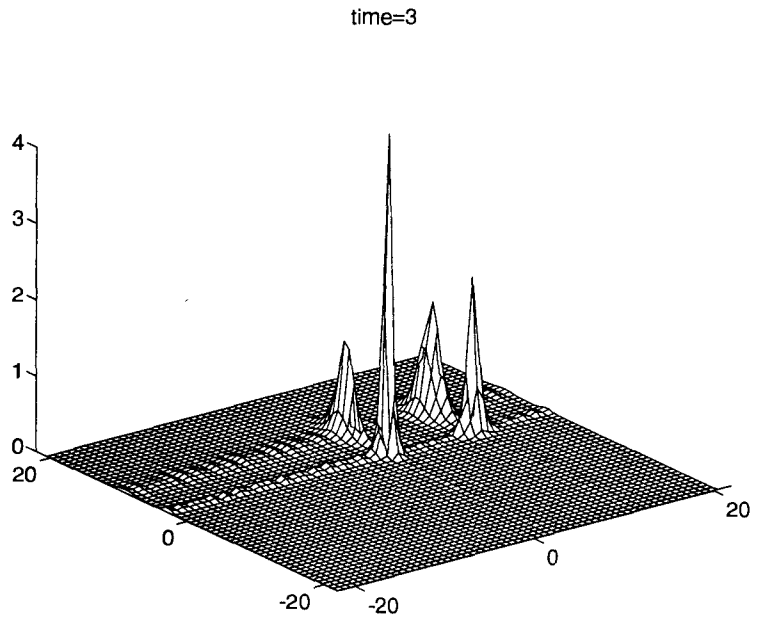

Figure 16. Numerical solution of (2,2)-dromion with $N = 512$ and $\tau = 0.0025$, at $t = 3$.

Figures 13–15 correspond to $|u|$ which is initially composed of four exponentially decaying humps, localized in the third quadrant of the (x, y) plane. They move towards the origin, where they collide at approximately $t = 0$. After the collision four localized coherent structures continue to move into the first quadrant. Note that during the collision energy was exchanged. This differs from the interaction of solitons which do not exchange energy.

As a comparison to the theoretical solution, Figure 17 shows level curves of $|u|$ for the theoretical and numerical solutions. Again the overall shape and speed of the localized structures are accurately simulated using the split-step method with a relatively course grid and large time step. The deviation from the theoretical solution which can be seen as low amplitude bands in Figure 15 can be reduced by

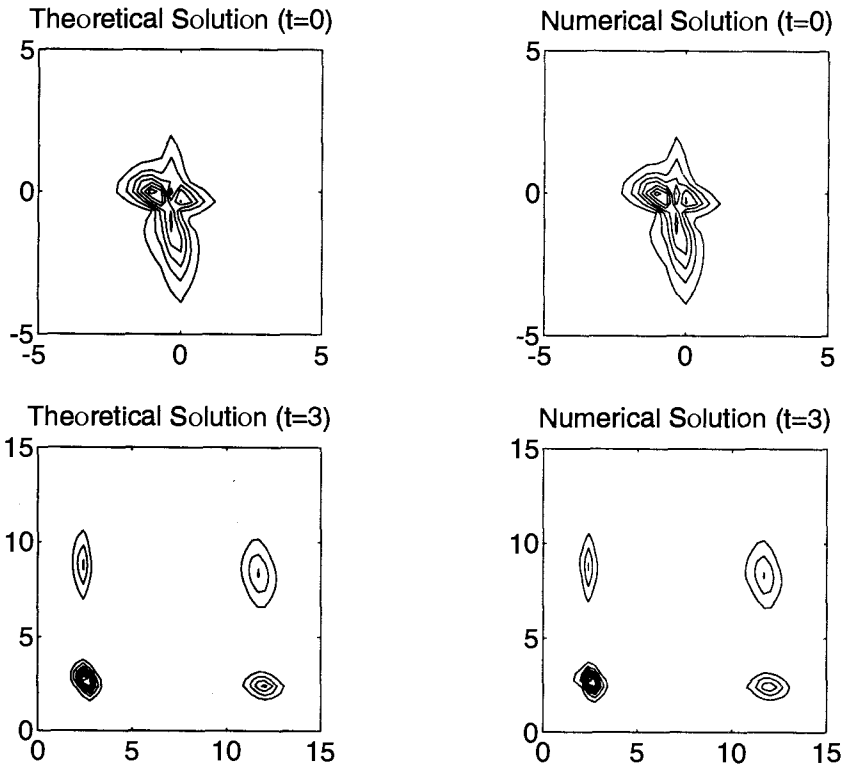

Figure 17. Comparison of the theoretical and numerical solutions of a (2,2)-dromion with non-symmetric ρ. Level curves of $|u|$ are shown.

choosing the computational parameters so that they more accurately portray the physical situation. With a finer grid of 512 points in both the x and y directions on a computational domain of $[-23, 23] \times [-23, 23]$ and a step size of $\tau = 0.0025$ this "noise" is reduced as seen in Figure 16.

Chapter 5

Modulational Instability

In this chapter the modulational stability of the Stokes wave profile (2.19) is analyzed through the stability of position independent solutions of the DS system. This stability analysis is an extension to two dimensions of the analysis carried out on the NLS equation in Chapter 1. In [8] it is shown that energy is transferred from the primary mode of a Stokes wave to the side bands and that the rate of growth in the side bands can increase exponentially. In [30] the instability of the uniform Stokes wave is studied for the NLS equation. In this setting the instability leads to an exponential growth in the unstable modes for a period of time and then a return to a nearly uniform state. In the NLS equations this process of growth and decay recurs periodically in time. When using the NLS to analyze modulational instability in the Stokes wave, the modulations are in one space direction only. In nature, modulation in any direction is possible. In [9] and [19] side-band disturbances which propagate obliquely to the primary Stokes waves were studied. However, these studies did not make use of the DS system. In this chapter the DS equations are considered as extensions of the one dimensional NLS equation and a linear stability analysis of the uniform solution is studied and compared to that of the NLS. Instability regions are given and simulations are conducted to show that the split-step method accurately simulates these analytic instabilities. Theoretical and simulated growth rates for the instabilities are compared.

5.1 Linear Stability Analysis

The Davey-Stewartson (DS) equations are given by

$$iu_t + \alpha_1 u_{xx} + \alpha_2 u_{yy} + \alpha_3 |u|^2 u + \alpha_4 uv = 0,$$

$$v_{xx} + \beta v_{yy} = (|u|^2)_{xx}, \tag{5.1}$$

where $u = u(x,y,t) \in \mathbb{C}, v = \phi_x(x,y,t) \in \mathbb{R}$, $\alpha_2 > 0$ and $\beta > 0$. Then $\tilde{u} = a \exp(i\alpha_3 |a|^2 t)$, $\tilde{v} = 0$ is a position independent solution of (5.1). To analyze the modular stability of the Stokes wave, we must analyze the stability of this solution of the DS system. To that end, let $u = (1+\varepsilon)\tilde{u}$, $v = \delta$ be a small perturbation of the above solution where $\varepsilon = \varepsilon(x,y,t) \in \mathbb{C}$, $\delta = \delta(x,y,t) \in \mathbb{R}$, $|\varepsilon(x,y,0)| \ll 1$ and $|\delta(x,y,0)| \ll 1$. Substituting this perturbed solution into (5.1) and keeping only first order terms yields:

$$i\varepsilon_t + \alpha_1 \varepsilon_{xx} + \alpha_2 \varepsilon_{yy} + \alpha_3 |a|^2 (\varepsilon + \varepsilon^*) + \alpha_4 \delta = 0,$$

$$\delta_{xx} + \beta \delta_{yy} = |a|^2 (\varepsilon_{xx} + \varepsilon^*_{xx}), \tag{5.2}$$

where ε^* is the complex conjugate of ε.

If ε and δ are assumed to be periodic in the space variables, with period P_x in the x-direction and period P_y in the y-direction, then ε and δ can be expanded formally into Fourier Series:

$$\varepsilon(x,y,t) = \sum_{m=-\infty}^{\infty} \sum_{n=-\infty}^{\infty} \hat{\varepsilon}_{m,n}(t) \exp(i(\mu_m x + \nu_n y))$$

$$\delta(x,y,t) = \sum_{m=-\infty}^{\infty} \sum_{n=-\infty}^{\infty} \hat{\delta}_{m,n}(t) \exp(i(\mu_m x + \nu_n y))$$

$$\mu_m = \frac{2\pi m}{P_x}, \quad \nu_n = \frac{2\pi n}{P_y}.$$

Substituting into (5.2) yields:

$$\sum_{m=-\infty}^{\infty} \sum_{n=-\infty}^{\infty} \left(i\frac{d}{dt}\hat{\varepsilon}_{m,n} - \alpha_1 \mu_m^2 \hat{\varepsilon}_{m,n} - \alpha_2 \nu_n^2 \hat{\varepsilon}_{m,n} \right) E_{m,n}$$
$$+\alpha_3 |a|^2 \left(\hat{\varepsilon}_{m,n} E_{m,n} + \hat{\varepsilon}^*_{m,n} E_{-m,-n} \right) + \alpha_4 \hat{\delta}_{m,n} E_{m,n} = 0,$$

$$\sum_{m=-\infty}^{\infty}\sum_{n=-\infty}^{\infty}\left(-\mu_m^2\hat{\delta}_{m,n}-\beta\nu_n^2\hat{\delta}_{m,n}\right)E_{m,n}=$$
$$\sum_{m=-\infty}^{\infty}\sum_{n=-\infty}^{\infty}|a|^2\left(-\mu_m^2\hat{\varepsilon}_{m,n}-\mu_m^2\hat{\varepsilon}^*_{-m,-n}\right)E_{m,n}, \tag{5.3}$$

where $E_{m,n}=\exp(i(\mu_m x+\nu_n y))$ and we note that $\mu^2_{-m}=\mu^2_m$. Since the functions $E_{m,n}$ form an orthogonal set we have

$$i\frac{d}{dt}\hat{\varepsilon}_{m,n}-\alpha_1\mu_m^2\hat{\varepsilon}_{m,n}-\alpha_2\nu_n^2\hat{\varepsilon}_{m,n}+\alpha_3|a|^2(\hat{\varepsilon}_{m,n}+\hat{\varepsilon}^*_{-m,-n})+\alpha_4\hat{\delta}_{m,n}=0,$$
$$-\mu_m^2\hat{\delta}_{m,n}-\beta\nu_n^2\hat{\delta}_{m,n}=|a|^2(-\mu_m^2\hat{\varepsilon}_{m,n}-\mu_m^2\hat{\varepsilon}^*_{-m,-n}). \tag{5.4}$$

From (5.4) we find that

$$\hat{\delta}_{m,n}=\frac{|a|^2\mu_m^2}{\mu_m^2+\beta\nu_n^2}(\hat{\varepsilon}_{m,n}+\hat{\varepsilon}^*_{-m,-n}),\qquad m^2+n^2\neq 0.$$

Then equation (5.1) can be written in the form:

$$\frac{d}{dt}\hat{\varepsilon}_{m,n}=i\left(-\alpha_1\mu_m^2-\alpha_2\nu_n^2+\alpha_3|a|^2+\frac{\alpha_4|a|^2\mu_m^2}{\mu_m^2+\beta\nu_n^2}\right)\hat{\varepsilon}_{m,n}$$
$$+i\left(\alpha_3|a|^2+\frac{\alpha_4|a|^2\mu_m^2}{\mu_m^2+\beta\nu_n^2}\right)\hat{\varepsilon}^*_{-m,-n} \tag{5.5}$$

or

$$\frac{d}{dt}\begin{pmatrix}\hat{\varepsilon}_{m,n}\\ \hat{\varepsilon}^*_{-m,-n}\end{pmatrix}=G_{m,n}\begin{pmatrix}\hat{\varepsilon}_{m,n}\\ \hat{\varepsilon}^*_{-m,-n}\end{pmatrix}\qquad m^2+n^2\neq 0, \tag{5.6}$$

where the matrix $G_{m,n}$ is given by

$$G_{m,n}=i\begin{pmatrix}B_{m,n}-A_{m,n} & B_{m,n}\\ -B_{m,n} & A_{m,n}-B_{m,n}\end{pmatrix},$$

$$A_{m,n}=\alpha_1\mu_m^2+\alpha_2\nu_n^2,\quad B_{m,n}=\alpha_3|a|^2+\frac{\alpha_4|a|^2\mu_m^2}{\mu_m^2+\beta\nu_n^2}. \tag{5.7}$$

The eigenvalues of $G_{m,n}$ are given by:

$$\lambda^2_{m,n}=A_{m,n}(2B_{m,n}-A_{m,n})$$

or

$$\lambda^2_{m,n}=\frac{\alpha_1\mu_m^2+\alpha_2\nu_n^2}{\mu_m^2+\beta\nu_n^2}\Big[2|a|^2(\alpha_3+\alpha_4)\mu_m^2+2\alpha_3\beta|a|^2\nu_n^2$$
$$-(\alpha_1\mu_m^2+\alpha_2\nu_n^2)(\mu_m^2+\beta\nu_n^2)\Big]. \tag{5.8}$$

Eq. Type	α_1	α_4	fig.
e/e	1	1	18
h/e	−1	1	19
e/e	1	−1	20
h/e	−1	−1	21

Table 1. Coefficients of DS equations for instability plots.

When $\lambda_{m,n}^2$ is positive, we can expect an eigenvalue of $G_{m,n}$ with positive real part and hence exponential growth, at least locally in $\hat{\varepsilon}_{m,n}$. Note that if the perturbation is only dependent on y then

$$\lambda_n^2 = \nu_n^2 \left(2\alpha_3|a|^2 - \nu_n^2\right),$$

where we have set $\alpha_2 = 1$. This gives the same region of instability found in the NLS equation and by Benjamin and Feir in [8] (recall equation (1.10)) when $\alpha_3 = 2$.

Figures 18–21 show contour plots of λ^2 as a function of μ and ν, with λ^2 defined by

$$\lambda^2 = \frac{\alpha_1\mu^2 + \alpha_2\nu^2}{\mu^2 + \beta\nu^2}\left(2|a|^2(\alpha_3+\alpha_4)\mu^2 + 2\alpha_3\beta|a|^2\nu^2 - (\alpha_1\mu^2 + \alpha_2\nu^2)(\mu^2 + \beta\nu^2)\right).$$

In Figures 18–21 $\alpha_2 = 1$, $\alpha_3 = 2$, $\beta = 1$ and the other coefficients are given in Table 1, where "Eq. Type" refers to the equation type, i.e., e/e refers to an elliptic/elliptic type equation, and h/e refers to an hyperbolic/elliptic type equation. In Figures 19 and 21 the instability regions are of a shape found in [19, Figure 4.3].

In Figures 18–21 the shaded region indicates values of λ where exponential growth can occur. In Figure 18, darker regions indicate higher values of λ and hence faster growth rates for modes in the darker regions.

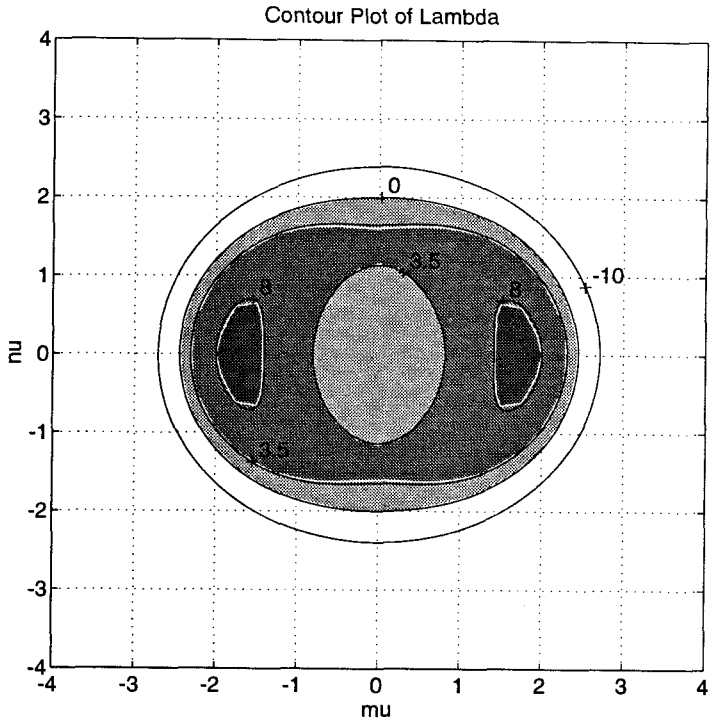

Figure 18. Instability region for elliptic/elliptic.

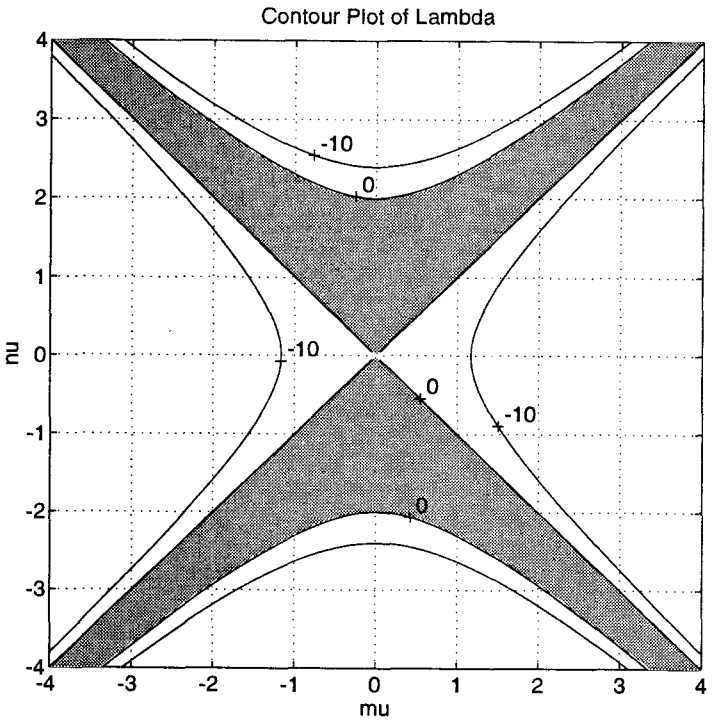

Figure 19. Instability region for hyperbolic/elliptic.

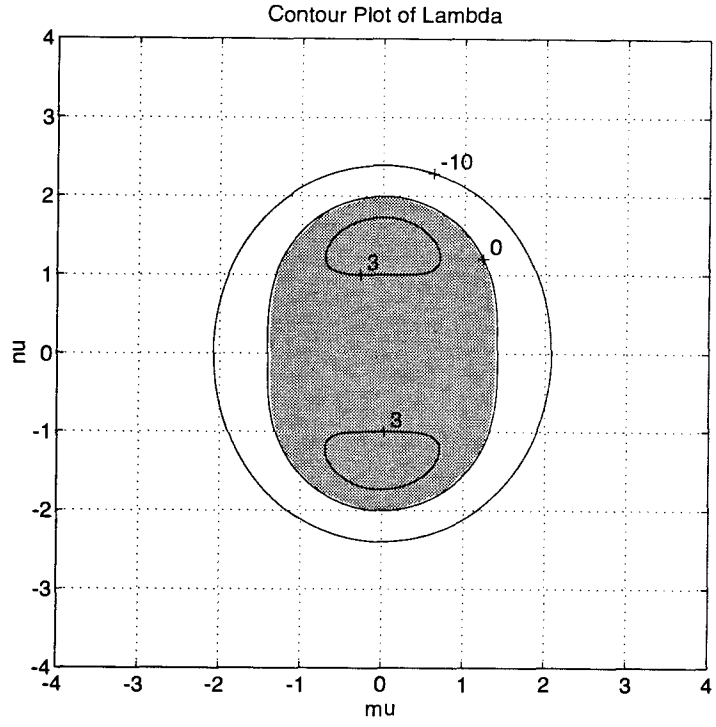

Figure 20. Instability region for elliptic/elliptic.

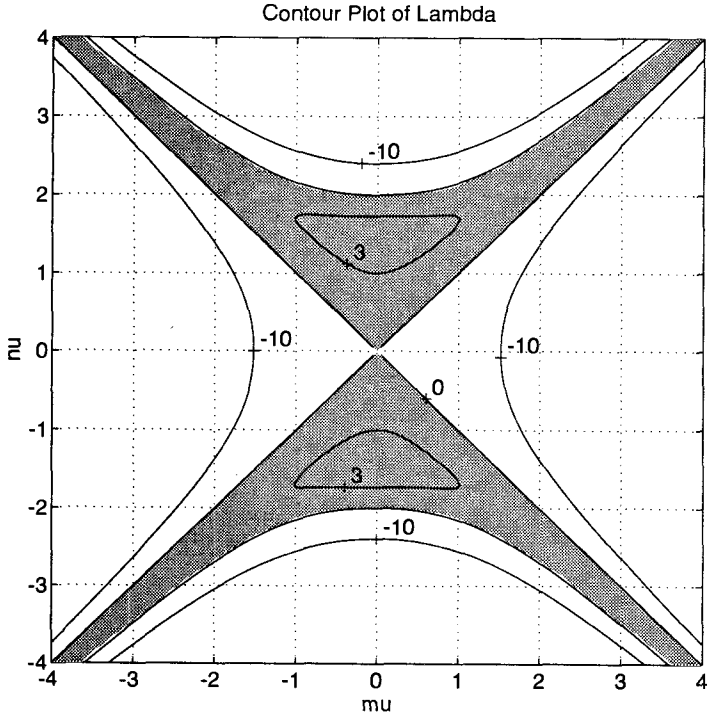

Figure 21. Instability region for hyperbolic/elliptic.

5.2 Numerical Results

5.2.1 The DSII System

We extend the example presented in Chapter 1, Figure 5, to two dimensions. We look at the DSII equations,

$$iu_t + u_{yy} - u_{xx} + 2|u|^2u + uv = 0,$$

$$v_{xx} + v_{yy} = (|u|^2)_{xx}, \tag{5.9}$$

where $v = \phi_x$. Let $P = \sqrt{2}\pi$ and

$$u(x, y, 0) = 1 + 0.1\cos(\sqrt{2}y) + 0.0001\cos\left(\sqrt{2}(x + y)\right). \tag{5.10}$$

This system is essentially the same as that stated in Chapter 1 (1.11), with the roles of x and y interchanged and the introduction of a small x-dependence in the initial condition. The corresponding instability region is given by Figure 22 where the region of instability for these parameters corresponds to Figure 21. The solid line from $(0, -2)$ to $(0, 2)$ corresponds to the instability region for the NLS equation, the circles indicate where the Fourier modes (μ_j, ν_k) are located for this value of P, the circled *'s correspond to the modes associated with $\cos(\sqrt{2}y)$ and the circled +'s correspond to the modes associated with $\cos\left(\sqrt{2}(x + y)\right)$. The evolution of the Fourier modes, $\hat{u}_{0,0}$, $\hat{u}_{1,0}$ and $\hat{u}_{1,1}$, are shown in Figure 23 where $\hat{u}_{0,0}$ is indicated by the solid line, $\hat{u}_{1,0}$ by the dashed line and $\hat{u}_{1,1}$ by the dotted line.

Figures 25–26 are surface plots of the magnitude of the simulated solution to (5.9) with a grid size of 128 by 128 and $\tau = .00025$, using the split-step method. Figures 27–28 show the magnitude of the Fourier Transform of the simulated solution. In this case the recurrence phenomenon observed in Figure 6 of the NLS equations is short lived. Energy is transferred to the higher modes along the diagonals of the stability region. This causes a solution which is highly irregular as can be seen in the last frame of Figure 26.

It should be pointed out that although the solution becomes irregular the split-step method conserved the l_2-norm to machine accuracy and the Hamiltonian to five significant digits (see Figure 24).

The question arises whether the behavior observed in Figure 23, $t \geq 5$, is chaos, or just complicated behavior. If it is chaos, is it present in the continuous problem, or is it a numerical artifact? It is unlikely that the continuous equations are chaotic, since the DSII system is known to be integrable, at least in the infinite domain case and this is very likely also true in the periodic case. To check whether the irregular behavior might be numerical chaos, as in the discretization (1.3) in the introductory chapter, we investigated whether the evolution of the Fourier

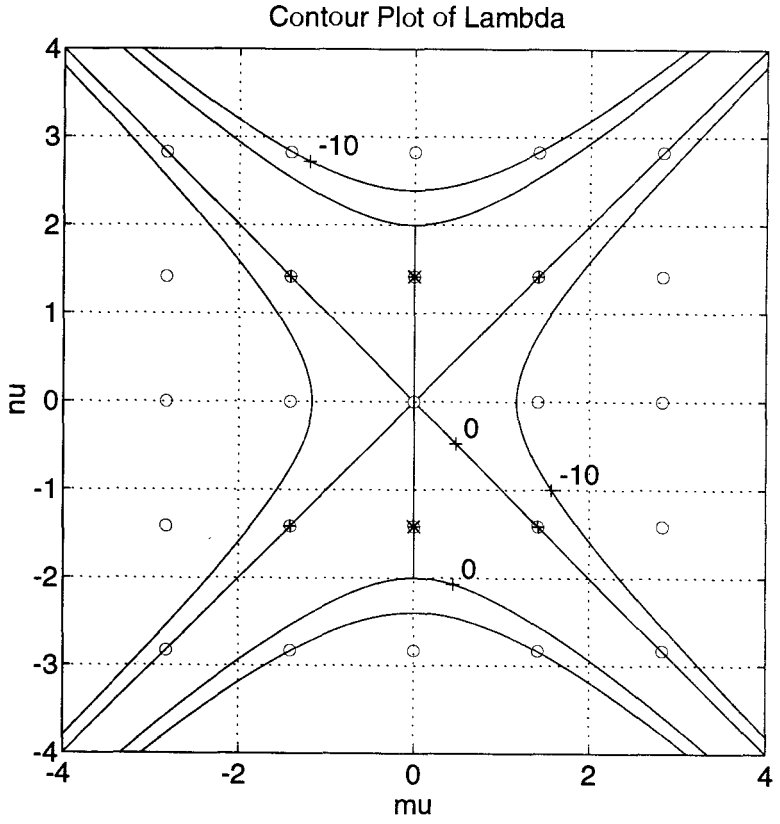

Figure 22. Instability region and Fourier mode locations. Level curves at $\lambda^2 = -10$ and 0 are shown.

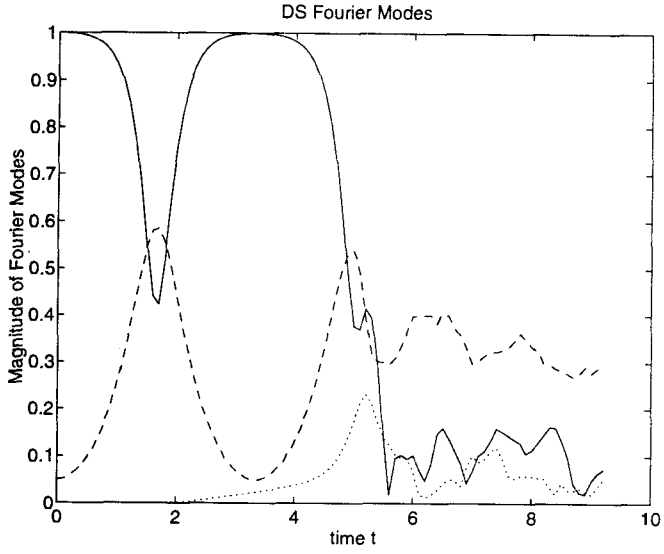

Figure 23. Evolution of the Fourier modes in the DSII system. $\widehat{u}_{0,0}$: solid line, $\widehat{u}_{1,0}$: dashed line and $\widehat{u}_{1,1}$: dotted line.

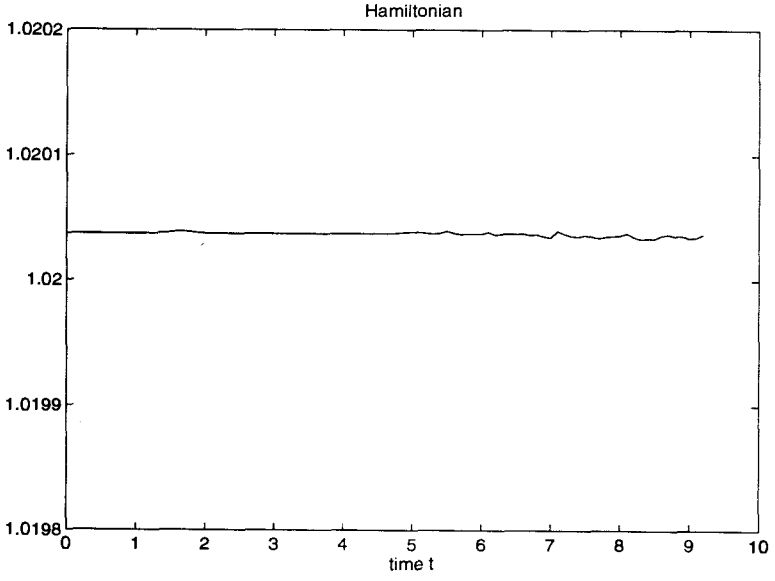

Figure 24. Hamiltonian of simulation.

modes was affected by the mesh size. Simulations with $N = 64$, 128 and 256 all resulted in exactly the same transfer of energy to the modes on the diagonal of the instability region and hence the same irregular behavior.

To investigate whether the solution was indeed chaotic we tested for sensitivity to the initial conditions. We used the following perturbations to the initial condition (5.10):

$$u(x,y,0) = 1 + 0.10001\cos(\sqrt{2}y) + 0.0001\cos\left(\sqrt{2}(x+y)\right),$$

$$u(x,y,0) = 1 + 0.1\cos(\sqrt{2}y) + 0.0001\cos\left(\sqrt{2}(x+y)\right) + 0.0001\cos\left(2\sqrt{2}(x+y)\right),$$

$$u(x,y,0) = 1 + (0.1 + 0.00001i)\cos(\sqrt{2}y) + 0.0001\cos\left(\sqrt{2}(x+y)\right)$$

and

$$u(x,y,0) = 1 + 0.1\cos(\sqrt{2}y) + 0.0001\cos\left(\sqrt{2}(x+y)\right) + 0.00001i\cos(\sqrt{2}x).$$

None of these initial conditions produced any significant changes from Figure 23. Thus, we conjecture that the behavior observed in Figure 23 is not chaos, but complicated behavior. Computer resources precluded us from doing more extensive tests involving a finer grid over a longer period of time.

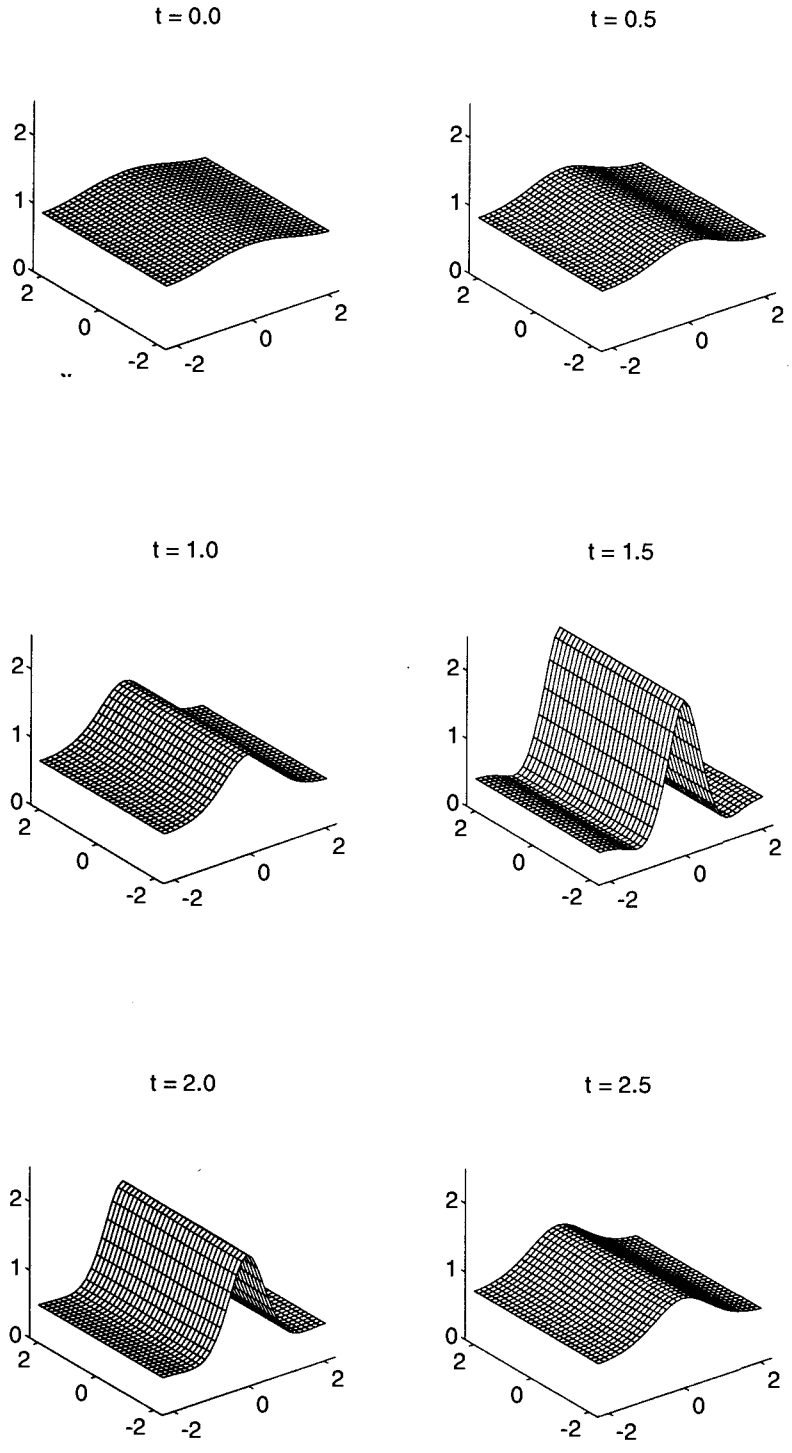

Figure 25. Magnitude of solution for numerical simulation. Time $t = 0$–2.5.

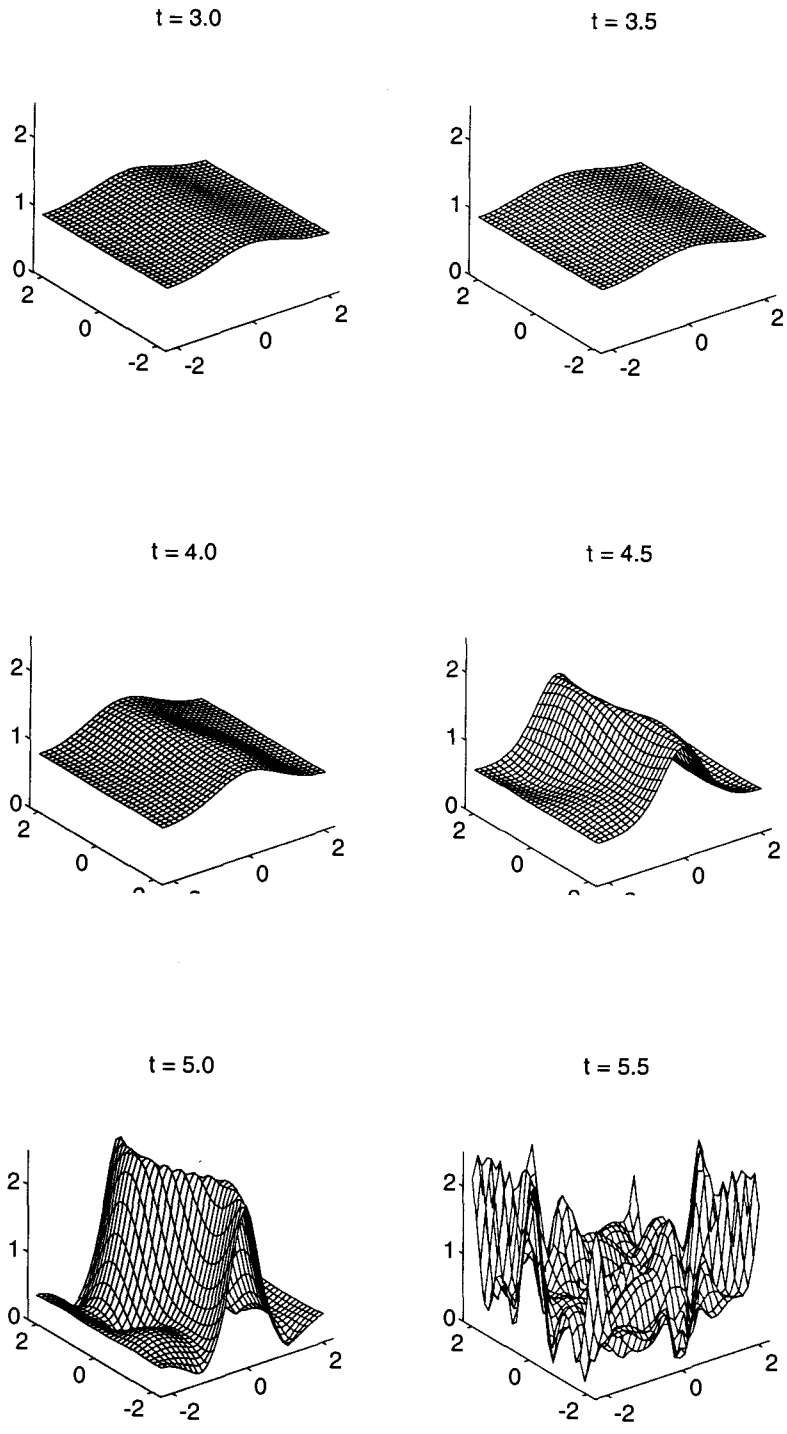

Figure 26. Continuation of Figure 25. Time $t = 3.0$–5.5.

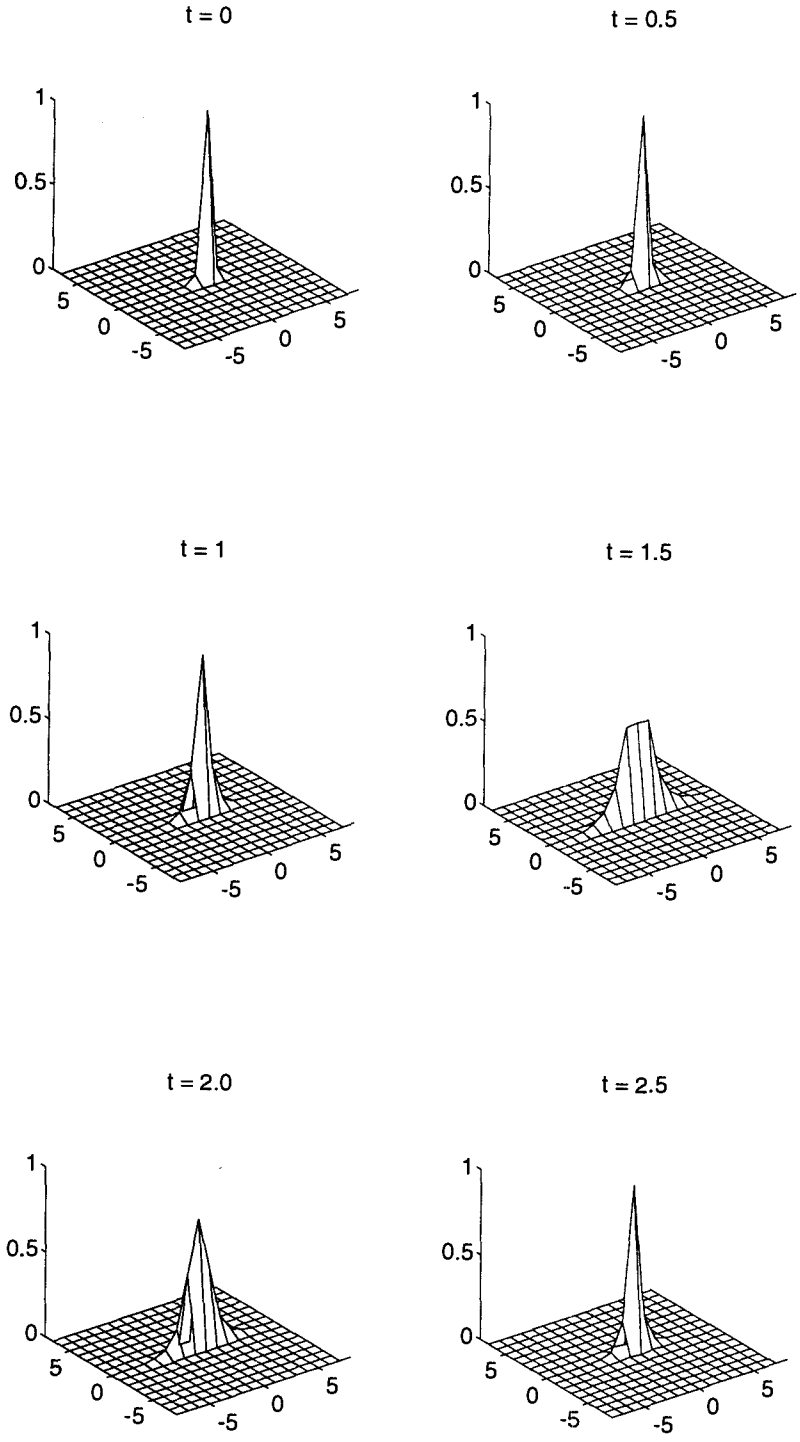

Figure 27. Magnitude of Fourier modes for numerical simulation. Time $t = 0$–2.5.

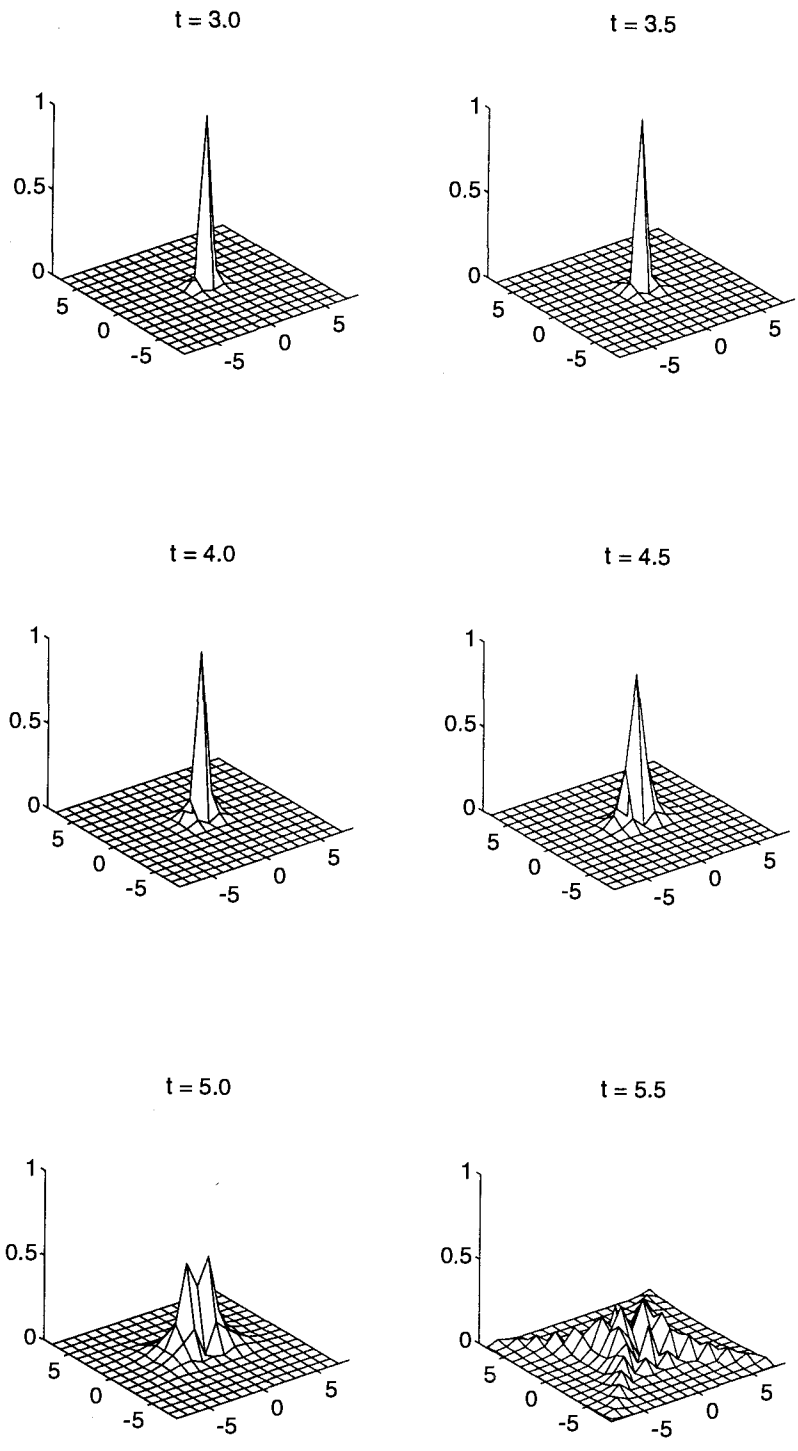

Figure 28. Continuation of Figure 27. Time $t = 3.0$–5.5.

To verify at least the onset of the instability, we compare the initial growth rate of the unstable mode in the simulation corresponding to Figure 23 with the rate predicted by the linear stability analysis. From equation (5.7) of the linear stability analysis, if $P = \sqrt{2}\pi$, then $\nu_1 = \sqrt{2}$ and

$$A_{0,1} = 2, \text{ and } B_{0,1} = 2.$$

So

$$G_{0,1} = i \begin{pmatrix} 0 & 2 \\ -2 & 0 \end{pmatrix},$$

and equation (5.6) then implies that

$$\begin{aligned} \frac{d}{dt}\widehat{\varepsilon}_{0,1} &= 2i\widehat{\varepsilon}^*_{0,-1} \\ \frac{d}{dt}\widehat{\varepsilon}^*_{0,-1} &= -2i\widehat{\varepsilon}_{0,1}. \end{aligned} \tag{5.11}$$

Solving for $\widehat{\varepsilon}_{0,1}$ gives,

$$\widehat{\varepsilon}_{0,1} = C_1 e^{2t} + C_2 e^{-2t}. \tag{5.12}$$

With an initial condition of $u(x, y, 0) = 1 + 0.1\cos(\sqrt{2}y) + 0.0001\cos\left(\sqrt{2}(x+y)\right)$, $\widehat{\varepsilon}_{0,1}(0) = 0.05$ and $\widehat{\varepsilon}^*_{0,-1}(0) = 0.05$, so $C_1 = C_2^* = 0.025 + 0.025i$.

Figure 29 shows a comparison of the growth rate of $\widehat{u}_{0,1}$ of the numerical simulation with the growth rate predicted from the linear stability analysis. The circles correspond to equation (5.12) at time intervals of 0.1 and the solid line is the evolution of $\widehat{u}_{0,1}$.

As can be seen in Figure 29 the split-step method accurately simulates the initial rate of growth predicted by the linear stability analysis. After about $t = 1$ the assumption of $|\varepsilon| \ll 1$ is no longer valid and the nonlinear properties of the DS system, such as conservation laws, determine the long-time behavior of the solution.

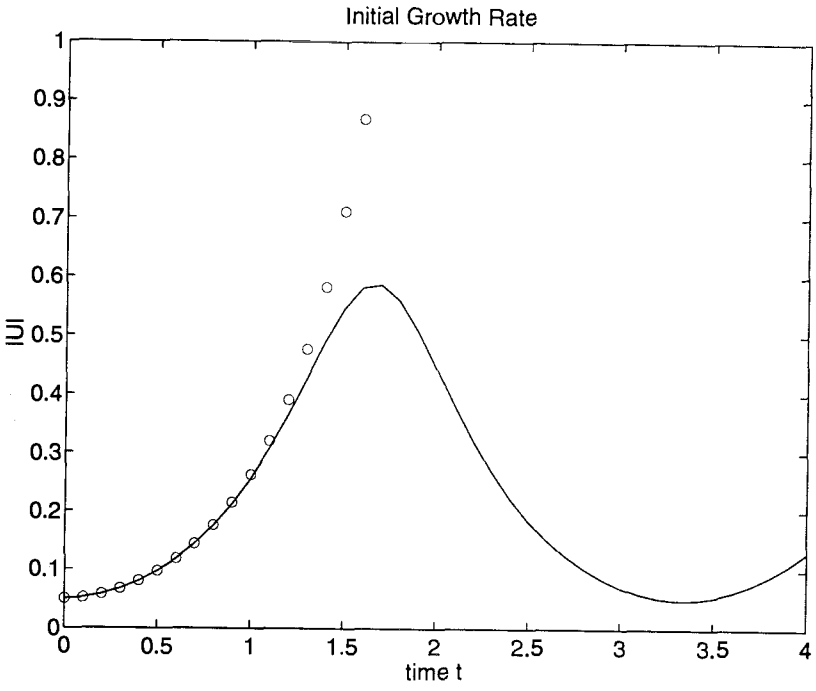

Figure 29. Growth rate of $\widehat{u}_{0,1}$. $\widehat{u}_{0,1}$: solid line, $C_1 \exp(2t) + C_1^* \exp(-2t)$: circles.

5.2.2 The Other Cases of the DS System

In the previous sections the h/e case of the DS system with $\alpha_4 = 1$ was used as an extension to the NLS equation. The h/e case of the DS system with $\alpha_4 = -1$ has a similar instability region and the behavior of both systems was found to be similar as well. In the case of the e/e DS systems, the instability regions are bounded ellipses and the behavior of the numerical solution differs from the h/e cases dramatically.

As an example consider the e/e case

$$iu_t + u_{xx} + u_{yy} + 2|u|^2 u + u\phi_x = 0$$

$$\phi_{xx} + \phi_{yy} = \left(|u|^2\right)_x,$$

with the initial condition $u(x, y, 0) = .85 + .1\cos(\pi y/2) + .0001\cos(\pi(x + y)/2)$ where $P = 4$, $N = 128$ and $\tau = .00025$. In this case the region of instability

given by Figure 30 has energy input into the two unstable modes indicated by the circled x's. The evolution of the three Fourier modes which were initially nonzero is shown in Figure 31, where $\widehat{u}_{0,0}$ is indicated by the solid line, $\widehat{u}_{0,1}$ by the dashed line and $\widehat{u}_{1,1}$ by the dotted line. Some of the other Fourier modes are shown in Figure 32. In this case the nonlinear effects of the DS equation quickly spread energy to modes outside the region of instability as shown by Figure 33 and the solution begins to show signs of approaching a singular solution, Figure 34, which we will discuss in the next chapter. We remind the reader that the e/e case is not integrable, which is consistent with the unlimited spreading of energy. However, the initial growth in the unstable mode does behave as predicted by the linear stability analysis as can be seen in Figure 35.

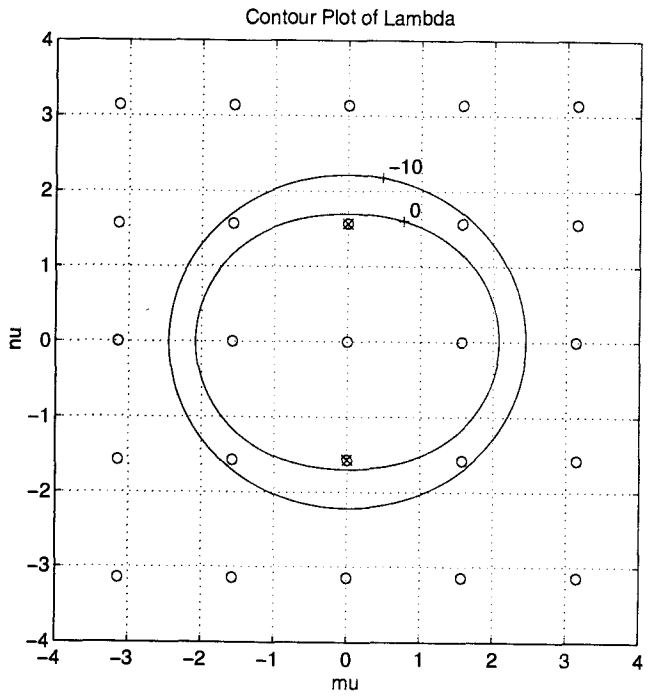

Figure 30. Region of instability for e/e case and $a = .85$.

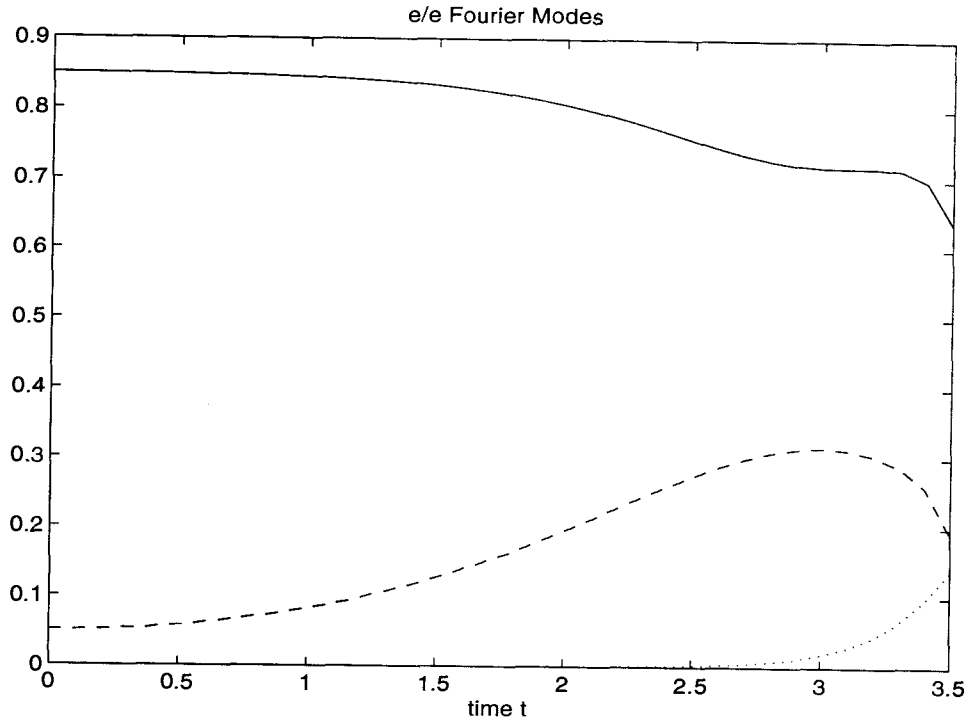

Figure 31. Fourier modes in e/e case. $\widehat{u}_{0,0}$: solid, $\widehat{u}_{0,1}$: dashed, $\widehat{u}_{1,1}$: dotted.

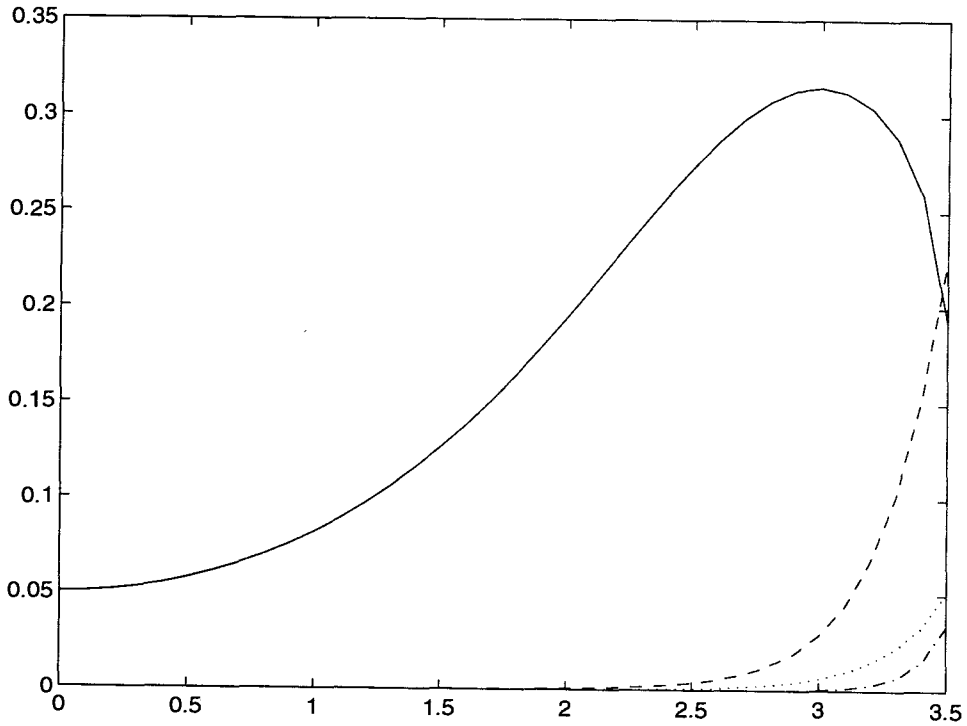

Figure 32. Fourier modes in e/e case. $\widehat{u}_{0,1}$: solid, $\widehat{u}_{1,0}$: dashed, $\widehat{u}_{1,2}$: dotted, $\widehat{u}_{2,2}$: dash-dotted.

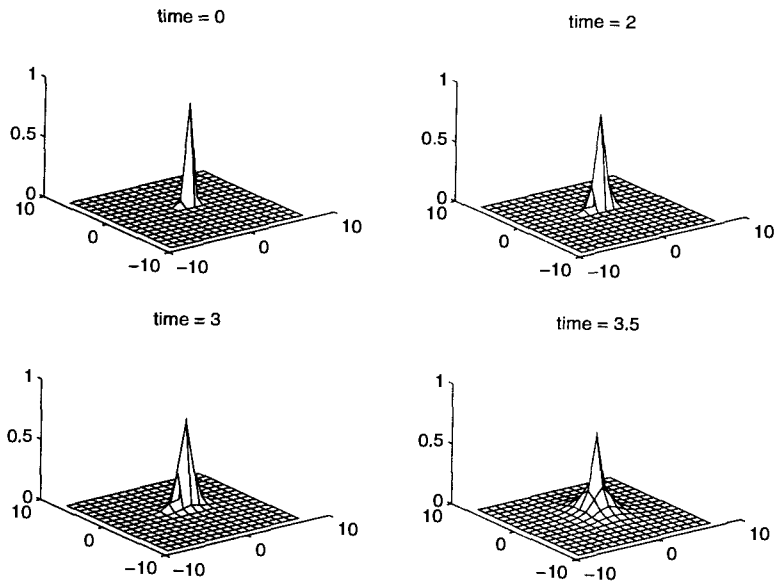

Figure 33. Mesh plot of Fourier modes for e/e case at $t = 3.5$.

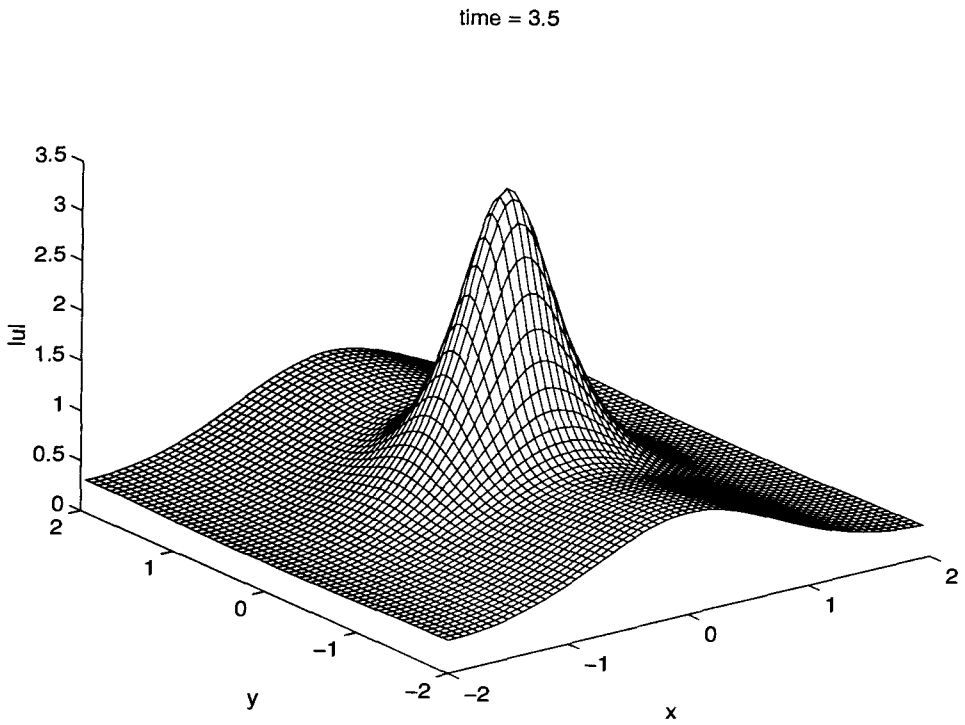

Figure 34. Mesh plot of solution for e/e case at $t = 3.5$

In conclusion, we observe that the uniform solution of the DS system, which leads to a traveling wave in the free surface of the Davey and Stewartson derivation, is unstable to side-band interaction of modes which lie within the regions given. The growth in the unstable side-band modes is initially exponential as was also seen in the NLS equation. However, in the tests conducted here, no long-term recurrence was observed. The effects of the nonlinearity in the DS system differed from that in the NLS in that energy was quickly transferred to modes which were not unstable. It remains to be seen what the effect of changing the coefficients of the DS system and hence the geometry of the unstable regions has on the above observations. One question which remains open is whether or not there exists values of the coefficients which produce recurrent instabilities.

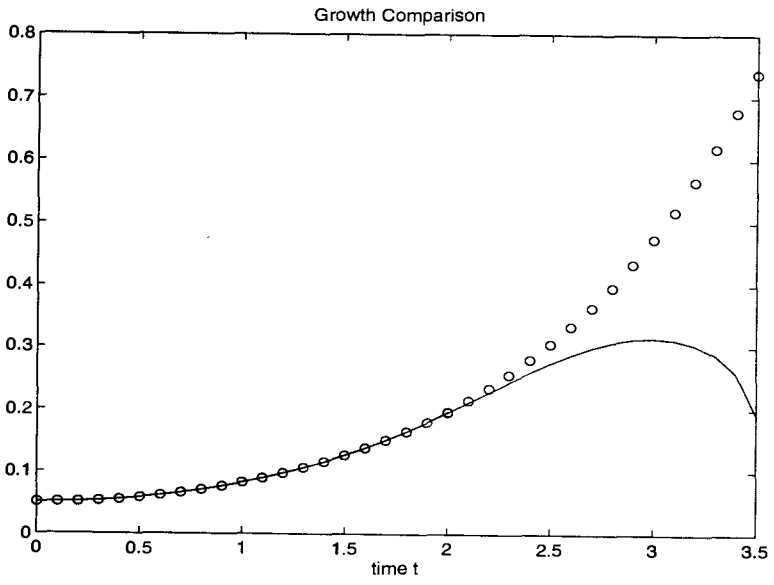

Figure 35. Growth comparison in unstable Fourier mode. $\widehat{u}_{0,0}$: solid line, theoretical growth curve $y = .033e^{.886t} + .017e^{-2.185t}$: circles.

Chapter 6

Blow-up in the DS System

In [22] solutions of the elliptic/elliptic Davey Stewartson system which develop singularities in finite time are discussed. A condition for when singular solutions arise and a rate of blow-up are derived. In this chapter the split-step method is applied to the Davey Stewartson system to simulate the singularity numerically. The rate of blow-up derived in [22] is compared with the numerical solution produced by the split-step method.

6.1 Applying the Split-Step Method

In the case of the elliptic/elliptic Davey Stewartson (DSEE) system

$$iu_t + u_{xx} + u_{yy} + |u|^2 u - \phi_x u = 0,$$

$$\phi_{xx} + \phi_{yy} = -\left(|u|^2\right)_x, \tag{6.1}$$

the split step method is applied in the same manner as in the DS II system. The only difference is the introduction of an adaptive grid size.

As the singularity time, t_*, is approached both $|u|$ and $|\nabla u|$ become large for some (x, y). In order to accurately approximate u and its derivatives with Fourier Transforms, a finer grid is needed as the singularity time is approached. Thus, methods for detecting when the number of grid points must be increased and a technique for interpolating u at the new grid points are needed.

When approximating u by a finite Fourier series

$$u(x,y,t) = \sum_{j=-N}^{N-1} \sum_{k=-N}^{N-1} \widehat{u}_{mn}(t) e^{i\mu_j x + i\nu_k y},$$

it is assumed that $\widehat{u}_{mn}$ is negligible when $|m| \geq N$ or $|n| \geq N$. As the singularity time is approached, N must be increased to satisfy this assumption. An indication that more Fourier modes are needed is when any of the $\widehat{u}_{mn}$ become significant in size for $|m|$ or $|n|$ near N. When the split step scheme is applied to the DSEE system the discrete quantity

$$\sum_{j=1}^{N} \sum_{k=1}^{N} \left[U_{jk}^{*} (LU)_{jk} + \frac{1}{2} N(U)_{jk} |U_{jk}|^2 \right], \tag{6.2}$$

of the numerical solution, which corresponds to the conserved Hamiltonian of the semi-discrete DS system (2.36), is not conserved exactly by the split-step method. However, from numerical experiments, the variation in (6.2) is related to the size of the time step used in the split step method and to the relative size of the higher Fourier modes used in the approximation of u. When the higher Fourier modes become significant in size, the variation in (6.2) increases. Thus, the variation in (6.2) can be used as an indicator for when more grid points are needed to accurately model the continuous DS system.

Next, a technique for interpolating the numerical solution over a finer grid is needed for when the grid size is increased. Consider a computational domain of $[-\frac{1}{2}P, \frac{1}{2}P] \times [-\frac{1}{2}P, \frac{1}{2}P]$ with N grid point in both the x and the y directions. Then $\Delta x = \Delta y = \frac{P}{N}$. If Fast Fourier Transforms (FFT's) are to be used in the approximations to the derivatives of u, then N should be a power of 2 for optimum efficiency. Thus, the approximation of u on the grid is given by

$$U_{jk} \approx u(x_j, y_k) = \sum_{m} \sum_{n} \widehat{u}_{mn} e^{i\mu_m x_j + i\nu_n y_k},$$

where $x_j = -\frac{1}{2}P + j\Delta x$, $y_k = -\frac{1}{2}P + k\Delta y$ and the indices, j and k range over $1, \ldots, N$ and m and n range over the values $\left\{-\frac{1}{2}N, \ldots, \frac{1}{2}N - 1\right\}$. If the number

of grid points in both the x and the y directions is doubled whenever a finer grid is needed, then FFT's can be used for the interpolation. The interpolation then becomes

$$\begin{aligned} u(x_j + \frac{\Delta x}{2}, y_k) &= \sum_m \sum_n \widehat{u}_{mn} e^{i\mu_m (x_j + \frac{\Delta x}{2}) + i\nu_n y_k} \\ &= \sum_m \sum_n \widehat{u}_{mn} e^{i\mu_m \frac{\Delta x}{2}} e^{i\mu_m x_j + i\nu_n y_k}, \end{aligned} \tag{6.3}$$

and similarly

$$u(x_j, y_k + \frac{\Delta y}{2}) = \sum_m \sum_n \widehat{u}_{mn} e^{i\nu_n \frac{\Delta y}{2}} e^{i\mu_m x_j + i\nu_n y_k},$$

$$u(x_j + \frac{\Delta x}{2}, y_k + \frac{\Delta y}{2}) = \sum_m \sum_n \widehat{u}_{mn} e^{i\mu_m \frac{\Delta x}{2} + i\nu_n \frac{\Delta y}{2}} e^{i\mu_m x_j + i\nu_n y_k}.$$

This corresponds to one FFT2 to find the $\widehat{u}_{mn}$, and three series of multiplications of the $\widehat{u}_{mn}$ by constants and FFT2's to get the interpolated values of u.

In addition to increasing the grid size, the time step should also be reduced as the singularity time is approached. If the time step were too large, then the split-step method could attempt to step past the singularity time causing numerical inaccuracy.

6.2 Numerical Simulation

In [22] it has been shown that the DSEE system produces solutions which become singular in a finite period of time provided the initial condition satisfies a minimum amplitude requirement. We applied the split-step method to the DSEE system (6.1) using an initial condition $u(x, y, 0) = 4 \exp(-(x^2 + y^2)/4)$. The grid size initially was 128×128 with a computational domain of $[-8, 8] \times [-8, 8]$ and $\tau = 10^{-4}$. The grid size was modified based on a tolerance of 5×10^{-6} in the variation of (6.2). Each time a grid modification was indicated, the grid size was doubled in both the x and y directions and the τ was reduced by a factor of 4.

Figure 36 shows the profile of the numerical solution as the solution approaches the singularity time. Figure 37 shows the amplitude of the numerical solution

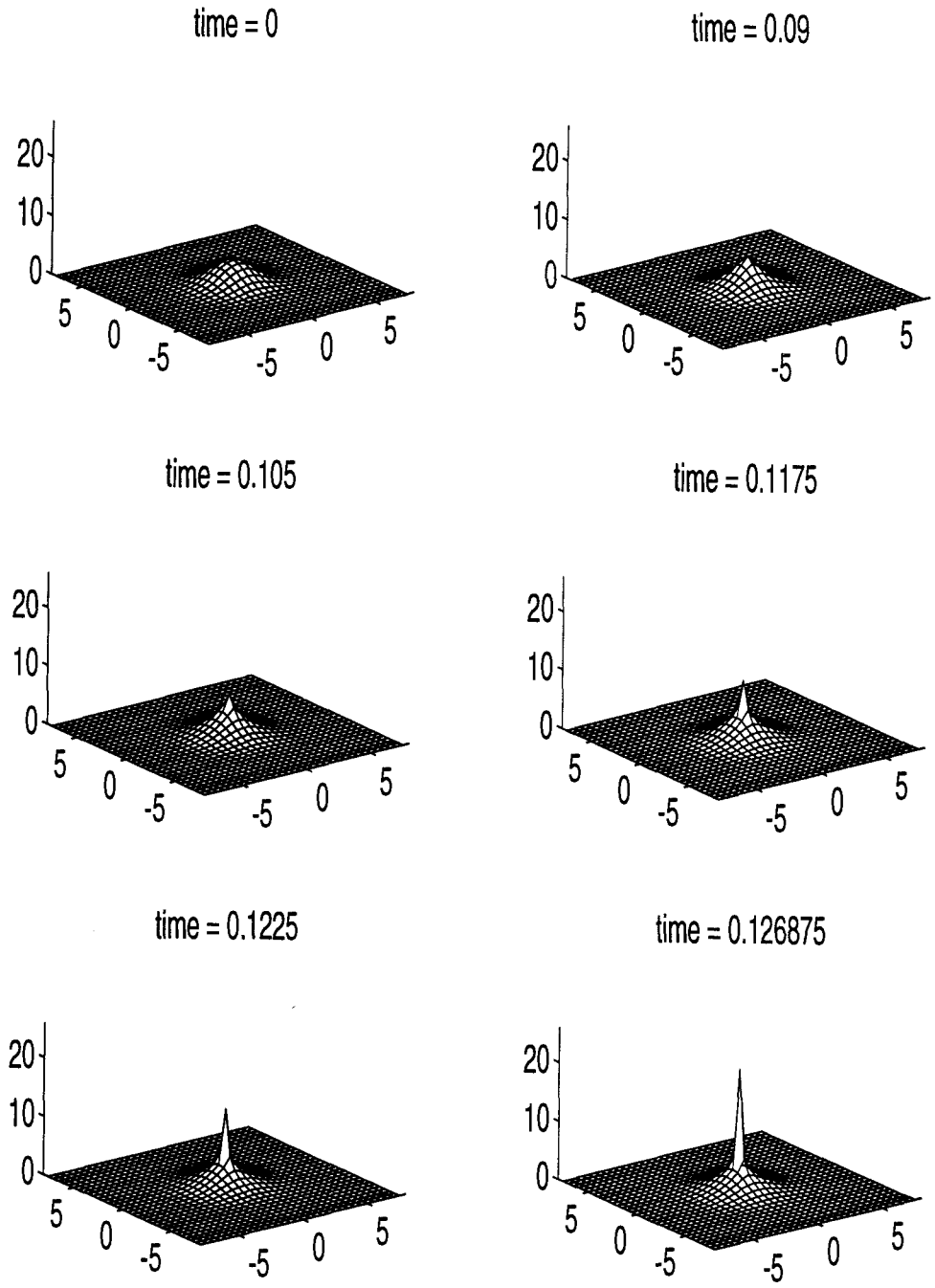

Figure 36. Evolution of the numerical solutions of the DSEE system.

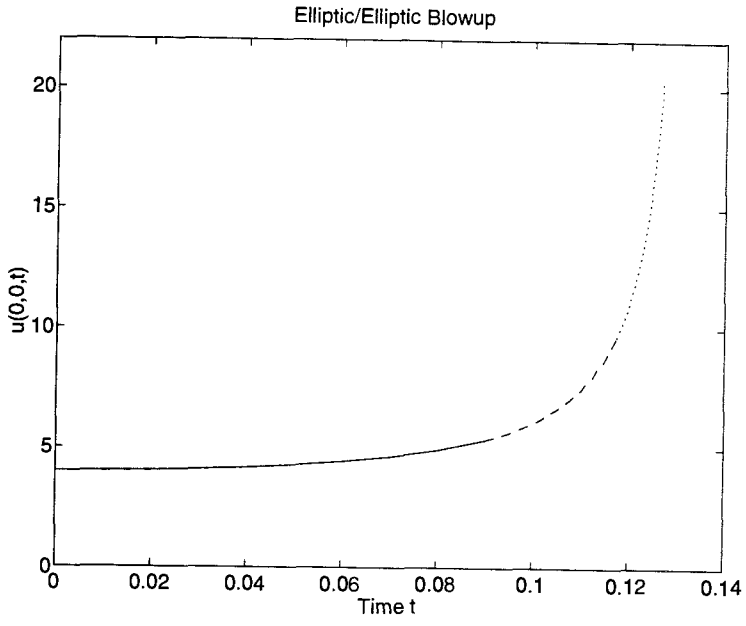

Figure 37. Amplitude of the numerical solution. The change in grid size is indicated by the change in line style. 128×128 grid: solid, 256×256 grid: dashed, 512×512 grid: dotted.

and indicates the grid size during the simulation. In Figure 38 MATLAB was used to compare the growth rate of the numerical solution to the asymptotic estimate derived in [22]. Theoretically, the growth rate is

$$A(t) = \mathcal{O}\left(\left(\frac{\ln\ln\left(\frac{1}{t_*-t}\right)}{(t_*-t)}\right)^{\frac{1}{2}}\right),$$

as $t \to t_*$. In Figure 38 the circles represent the amplitude of the numerical solution and the solid line represents the function

$$y = C\left(\left(\frac{\ln\ln\left(\frac{1}{t_*-t}\right)}{(t_*-t)}\right)^{\frac{1}{2}}\right),$$

where we used the MATLAB routine FMINS in calculating the values of $C = 0.907221$ and $t_* = 0.130305$ which give the best least squares fit.

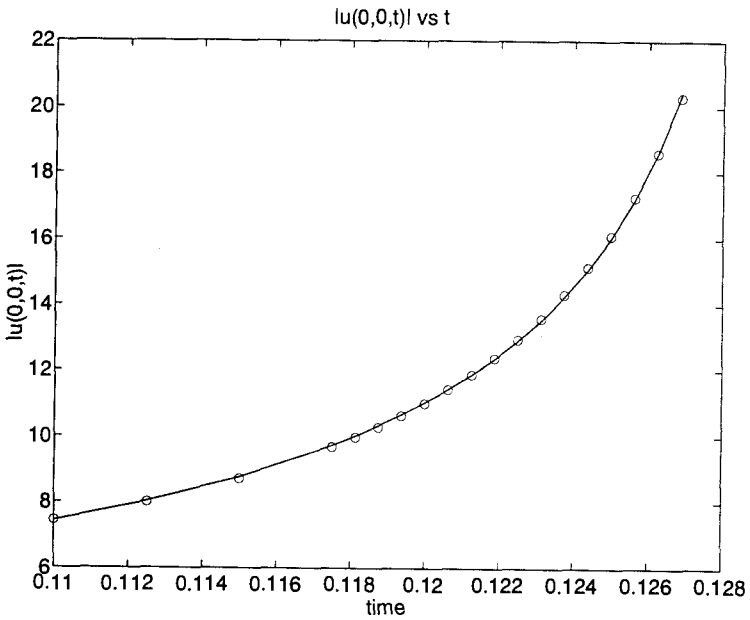

Figure 38. Comparison of the growth in amplitude with the theoretical estimate.

Conclusions

The split-step Fourier method has successfully been applied to the NLS equation to simulate soliton solutions and to investigate modulational stability in Stokes waves. In this thesis we have extended the application of the split-step Fourier method from the one dimensional NLS equation to the DS system. We have tested our method on known soliton and dromion solutions [29], and found good agreement between the theoretical and numerical results. We have studied the stability of the Stokes wave through the modulational stability of the h/e and e/e DS system using a linear stability analysis and found that position independent solutions of the DS system are susceptible to side-band instability. Regions of instability were found for these cases and we showed that the split-step method accurately simulated the initial growth in the side-band modes as predicted by the instability analysis. We then modified the split-step method in order to simulate solutions that blow-up in finite time. By introducing an adaptive grid scheme to the split-step method, we were able to efficiently simulate solutions to the e/e DS system for times which approach a singularity. We then compared the rate of blow-up in the simulation to known theoretical results and found that the numerical solution had the same asymptotic growth rate.

Some of the questions which have arisen in this study are as follows: (i) How can the split-step method be applied to the DS system in the case that u does not vanish at infinity? Such solutions were found by Anker and Freeman [6]. (ii) What is the effect of changing the coefficients of the DS system in the linear stability analysis? The coefficients can be chosen so that the regions of instability for the h/e case are bounded lines other than the forty-five degree lines in Figures 19 and 21. Thus, modes do not lie on the boundary of the region of instability. What is the physical significance and evolution of unstable modes in this case? (iii) In the case of solutions which become singular, can a dynamic rescaling algorithm, as in [22], be developed which is symplectic? In [22] it is not

clear whether the algorithm preserves the symplectic structure of the DS system. Finally, no convergence proof has yet been given for the split-step method.

To date, the material in this thesis has generated one paper on the computation of soliton and dromion solutions [29]. It is also believed that another paper can be produced from a more comprehensive study of the linear stability analysis in the DS system.

BIBLIOGRAPHY

[1] M. J. Ablowitz and P. A. Clarkson, *Solitons, Nonlinear Evolution Equations, and Inverse Scattering*, (Cambridge University Press, Cambridge, 1991).

[2] M. J. Ablowitz and B. M. Herbst, On Homoclinic Structure and Numerically Induced Chaos for the Nonlinear Schrödinger Equation, *SIAM J. Appl. Math.* 50 no. 2 (1990) 339–351.

[3] M. J. Ablowitz, S. V. Manakov and C. L. Schultz, On the Boundary Conditions of the Davey-Stewartson Equations, *Phys. Lett. A* 148 (1990) 50–52.

[4] M. J. Ablowitz and H. Segur, On the Evolution of Packets of Water Waves, *J. Fluid Mech.* 92 (1979) 691–715.

[5] M. J. Ablowitz and H. Segur, Solitons and the Inverse Scattering Transform, *SIAM*, Philadelphia (1981).

[6] D. Anker and N. C. Freeman, On the Soliton Solution of the Davey-Stewartson Equation for Long Waves, *Proc. Roy. Soc. London A.* 360 (1978) 529–540.

[7] V. A. Arkadiev, A. K. Pogrebkov and M. C. Plivanov, Inverse Scattering Transform Method and Soliton Solutions for Davey-Stewartson II Equation, *Physica D* 36 (1989) 189–197.

[8] T. B. Benjamin and J. E. Feir, The Disintegration of Wave Trains on Deep Water, *J. Fluid Mech., vol. 27, part 3* (1967) 417–430.

[9] D. J. Benney and G. J. Roskes, Wave Instabilities, *Studies Appl. Math* 48 (1969) 377–385.

[10] M. Boiti, J. J. Leon, L. Martina and F. Pempinelli, Scattering of Localized Solitons in the Plane, *Phys. Lett. A* 132 (1988) 432–439.

[11] W. Craig, S. Sulem and P. L. Sulem, Nonlinear Modulation of Gravity Waves: A Rigorous Approach, *Nonlinearity* 5 (1992) 497–522.

[12] A. Davey and K. Stewartson, On Three-dimensional Packets of Surface Waves, *Proc. Roy. Soc. London A* 338 (1974) 101–110.

[13] L. D. Faddeev and L. A. Takhtajan, *Hamiltonian Methods in the Theory of Solitons*, Springer-Verlag, Berlin-Heidelberg-New York (1987).

[14] A. S. Fokas and P. M. Santini, Dromions and a Boundary Value Problem for the Davey-Stewartson 1 Equations, *Physica D* 44 (1990) 99–130.

[15] J. M. Ghidaglia and J. C. Saut, On the Initial Value Problem for the Davey-Stewartson System, *Nonlinearity* 3 (1990) 475–506.

[16] H. Hasimoto and H. Ono, Nonlinear Modulation of Gravity Waves, *J. Phys. Soc. Japan* 33 (1972) 805–811.

[17] B. M. Herbst and M. J. Ablowitz, Numerically Induced Chaos in the Nonlinear Schrödinger Equation, *Phys. Rev. Lett.*, 62 (1989a) 2065–2068.

[18] R. L. James and J. A. C. Weideman, Pseudospectral Methods for the Benjamin-Ono Equations, in: R. Vichnevestky, K. Doyle, and G. Richter (eds.), *Advances in Computer Methods for Partial Differential Equations VII* (IMACS, 1992.)

[19] C. C. Mei, *The Applied Dynamics of Ocean Surface Waves*, (World Scientific 1989).

[20] R. McLachlan, On the Numerical Integration of Ordinary Differential Equations by Symmetric Composition Methods, *preprint.*

[21] R. McLachlan, Symplectic Integration of Hamiltonian Wave Equations, *Numerische Mathematik*, 66 (1994) 465–492.

[22] G. C. Papanicolaou, C. Sulem, P. L. Sulem and X. P. Wang, Focusing Singularities of Davey-Stewartson Equations for Gravity-Capillary Waves, *preprint.*

[23] P. M. Santini, Energy Exchange of Interacting Coherent Structures in Multidimensions, *Physica D* 41 (1990) 26–54.

[24] C. L. Schultz and M. J. Ablowitz, Strong Coupling Limit of Certain Multidimensional Nonlinear Wave Equations, *Stud. Appl. Math.* 80 (1989) 229–238.

[25] G. Strang, On the Construction and Comparison of Difference Schemes, *SIAM J. Numer. Anal.* 5 (1969) 506–517.

[26] T. R. Taha and M. J. Ablowitz, Analytical and Numerical Aspects of Certain Nonlinear Evolution Equations. II. Numerical, Nonlinear Schrödinger equation. *J. Comp. Phys.* 55 (1984) 203–230.

[27] F. D. Tappert, Numerical Solutions of the Korteweg-de Vries Equation and its Generalizations by the Split-Step Fourier Method, *Lect. Appl. Math. Am. Math. Soc.* 15 (1974) 215–216.

[28] J. A. C. Weideman and B. M. Herbst, Split-Step Methods for the Solution of the Nonlinear Schrödinger Equation, *SIAM J. Numer. Anal.* 23 (1986) 485–507.

[29] P. W. White and J. A. C. Weideman, Numerical Simulations of Solitons and Dromions in the Davey-Stewartson System, to appear in: *Journal of Mathematics and Computers in Simulation (Special Issue on Nonlinear Wavelike Equations).*

[30] H. C. Yuen and W. E. Ferguson, Relationship between Benjamin-Feir Instability and Recurrence in the Nonlinear Schrödinger Equation, *Phys. Fluids*, 21 (1978) 1275–1278.

[31] F. Zhang and Luis-Vázquez, Two Energy Conserving Numerical Schemes for the Sine-Gordon Equation, *Applied Mathematics and Computation* 45 (1991) 17–30.

Appendix

A Programs

In this appendix the matlab code used in the simulations of the NLS equation and Fortran code for the simulations of the DS system are given. In the following Matlab program the initial condition and some of the parameters must be specified.

```
%*****  Second order split step method for NLS  *****
%
%            u_t = iu_x + iq|u|^2u
%
%  Variables which must be specified:
%    dt - time step size.
%    M1 - number of outputs (columns of U).
%    M2 - number of iterations between outputs.
%    N - number of grid points.
%    q - coefficient in NLS equation.
%    u0 - initial condition.
%
U = [u0];
u = u0;
tcount=0;
n = [-N/2:1:N/2-1]';
mu = fftshift(i*2*pi*n/L);
e = exp(-4*n.*n*pi*pi*dt*i/L/L);  % Multipliers for linear step.
e = fftshift(e);
du = ifft(mu.*fft(u));
H = -sum(abs(du).^2) + (q/2)*sum(abs(u).^4); % Hamiltonian
t=tcount;

for m = 1:1:M1  % M1 = number of outputs.
for m2=1:1:M2   % M2 = number of iterations between outputs.

% Advance according to nonlinear term with 1/2 a time step.
v = exp(0.5*dt*i*q*(abs(u).*abs(u))).*u;

% Advance according to linear term.
w = ifft(e.*fft(v));

% Advance according to nonlinear term with 1/2 a time step.
u = exp(0.5*dt*i*q*(abs(w).*abs(w))).*w;

tcount=tcount+dt;
end

t=[t tcount];  % t = vector with times corresponding to outputs.
U = [U,u];     % U = solution matrix.  Each column a different
               %     time.

du = ifft(mu.*fft(u));
h = -sum(abs(du).^2) + (q/2)*sum(abs(u).^4);
```

```
H = [H,h];

end
%*****  End of Program *****
```

The code for solving the DS system is broken into four parts. The first part contains the main line code in which the size of the computational grid and the coefficients in the DS equations are specified. The following code is for solving the DSII and the elliptic/elliptic DS equations.

```
      program ds
c*****************************************************************
c
c  This program finds a numerical solution to the DS equations.
c
c       u_t = i(a1*u_xx + a2*u_yy + a2*u*|u|^2 + a4*u*v)
c       v_xx + b1*v_yy = b2*(|u|^2)_xx
c
c  The values in the parameter statement below and the initialize
c  subroutine may need modification for particular physical and
c  initial conditions.
c
c  Note: this solution technique assumes a periodic initial
c  condition and periodic solution.
c
c  Variables:
c    d2 - constant multipliers used in calculation of ham.
c    dt - time step size.
c    du - working space variable, derivative of u.
c    e - constant multipliers used in linear step.
c    el2 - 2 norm of u.
c    f - array used to calculate FFT.
c    g - constant multipliers used in calculating v.
c    ham - Hamiltonian of u.
c    hamlast - used in stopping criteria.
c    htol - used in stopping criteria.
c    hx, hy - period lengths in x and y directions.
c    inct - number of iterations between outputs.
c    nxo, nyo - used in output routine to control amount of
c               output.
c    lastu - holds previous u value for output upon termination.
c    nd - maximum number of grid points in x or y direction.
c    nmax - maximum of nx and ny, used in FFT routine.
c    nt - number of iterations (time steps) to perform.
c    nx, ny - number of grid points in x and y directions.
c    tf - keeps track of time of iteration of last output.
c    ti - initial time of simulation.
c    tol - used in output to control number of digits of output.
c    u, v - solution to DS equations.
c    uf - Fourier Transform of u.
c    w - working space variable.
```

```
c*****************************************************************
      implicit real*8(a-h,o-z), integer*4(i-n)

      parameter(nd=256)
      parameter(a1=-1.0d0, a2=1.0d0, a3=2.0d0, a4=1.0d0)
      parameter(b1=1.0d0, b2=1.0d0)

      complex*16 i
      parameter(i=dcmplx(0.0d0,1.0d0))
      complex*16 u(nd,nd), v(nd,nd), w(nd,nd)
      complex*16 uf(nd,nd), lastu(nd,nd)
      complex*16 e(nd,nd), du(nd,nd)
      dimension f(2*nd,nd), g(nd,nd), d2(nd,nd)

      open(10,file='u.dat',status='unknown')
      open(11,file='v.dat',status='unknown')
      open(12,file='uf.dat',status='unknown')
      open(13,file='t.dat',status='unknown')
      open(15,file='l2.dat',status='unknown')
      open(16,file='ham.dat',status='unknown')
      open(17,file='stopstat.dat',status='unknown')

      nx = nd
      ny = nd
      nmax = nd

      call initialize(u,nd,nx,ny,nxo,nyo,hx,hy,nt,inct,dt,ti,tol,
     , htol)

9000 format(i4)
9001 format(i6)
9002 format(f8.4)
9003 format(f12.8)
9004 format(f18.15)

      do 9010 j=10,12
         write(j,9000) nx
         write(j,9000) ny
         write(j,9000) nxo
         write(j,9000) nyo
         write(j,9002) hx
         write(j,9002) hy
         write(j,9001) nt
         write(j,9004) dt
         write(j,9001) inct
         write(j,9004) ti
9010 continue

      call getmult(d2,e,g,nd,nx,ny,hx,hy,a1,a2,b1,b2,dt)

      call calcv(u,v,g,f,uf,w,nd,nx,ny,nmax)
      call output(u,v,nd,nx,ny,nmax,nxo,nyo,uf,w,f,tol)
      write(13,9004) ti
```

```
      call calc_h(u,v,uf,w,f,du,d2,a3,a4,nd,nx,ny,nmax,ham,el2)
      write(15,9003) el2
      write(16,9003) ham

      hamlast = ham

      tf = ti
      call backup(u,lastu,nd,nx,ny)
      tflast = tf
c*****************************************************************
c    Start of main loop
c*****************************************************************
      do 9020 loop=1,nt
         call calc_u(u,v,uf,w,f,e,g,nd,nx,ny,nmax,a3,a4,dt)

         if (mod(loop,inct).eq.0) then
         call calc_h(u,v,uf,w,f,du,d2,a3,a4,nd,nx,ny,nmax,ham,el2)
            if ((dabs(hamlast-ham)).lt.htol) then
               write(15,9003) el2
               write(16,9003) ham
            call output(u,v,nd,nx,ny,nmax,nxo,nyo,uf,w,f,tol)
               call backup(u,lastu,nd,nx,ny)
               tflast = tf + dt
               write(13,9004) tflast
               open(20,file='tflast.dat',status='unknown')
               write(20,9004) tflast
               close(20)
               else
                  write(17,*) 'htol exceded'
                  goto 9021
            end if
         end if

         tf = tf + dt
9020  continue

9021  open(14,file='ldata.dat',status='unknown',
     , form='unformatted')

      write(14) nx
      write(14) ny
      write(14) nxo
      write(14) nyo
      write(14) hx
      write(14) hy
      write(14) nt
      write(14) dt
      write(14) inct
      write(14) tflast
      do 9030 k=1,ny
         do 9030 j=1,nx
            write(14) lastu(j,k)
9030     continue

      close(10)
```

```
      close(11)
      close(12)
      close(13)
      close(14)
      close(15)
      close(16)
      close(17)
      end
c*****  End of Main Line Code  *****
```

The second part of the code is the initialization section which either reads the initial condition from disk or generates the initial condition from equations which must be modified in the routine to fit the simulation.

```
      subroutine initialize(u,ndim,nx,ny,nxo,nyo,hx,hy,nt,inct,
     , dt,ti,tol,htol)
c*****************************************************************
c
c  Subroutine initialize initializes the matrix of function
c  values.  It sets the initial value of the IBV  problem.  The
c  initial condition should be periodic (if changed)
c
c  Inputs: nx,ny  -- the dimension of the data matrix.
c          file 17 -- file 'stopstat.dat'.
c
c  Output: hx,hy     -- the length of the sides of the solution
c                       domain.
c          nt        -- number of time steps to perform.
c          inct      -- number of time steps between outputs.
c          dt        -- time step size.
c          ti        -- initial time.
c          u(nx,ny)  -- the initial values.
c          tol       -- used to control the number of digits of
c                       output.
c          htol      -- used for stopping criteria.
c
c  Other:  fileinput -- true if initialization from disk.
c                       false if initialization from equations
c                       in this routine.
c
c*****************************************************************
      implicit real*8(a-h,o-z), integer*4(i-n)
      complex*16 u(ndim,ny), i
      logical fileinput
      parameter (fileinput = .true.)
      parameter (i = dcmplx(0.0d0,1.0d0))

      tol = 9.999d-5
      htol = 5.0d-5

      if (fileinput) then
```

```
c-----------------------------------------------------------------
c    This section reads the initializing data from the file
c  lastdata in the current directory.  lastdata stores the
c  variables at the end of the program run so that the solution
c  can be continued by runing this program with fileinput set
c  to .true. .
c-----------------------------------------------------------------
      open(9,file='ldata.dat',status='old',form='unformatted')
            read(9) nnnx
            read(9) nnny
            if (.not.((nx.eq.nnnx).and.(ny.eq.nnny))) then
               close(9)
               write(17,*) 'lastdata does not match'
               stop
               endif
            read(9) nxo
            read(9) nyo
            read(9) hx
            read(9) hy
            read(9) nt
            read(9) dt
            read(9) inct
            read(9) tf_lastrun
            do 305 k=1,ny
               do 305 j=1,nx
                  read(9) u(j,k)
305               continue
            close(9)
            ti = tf_lastrun
         else
c-----------------------------------------------------------------
c    This section assigns the initializing data for a first run.
c  The values should be modified accordingly.
c-----------------------------------------------------------------
            pi = 2.0d0*dasin(1.0d0)

            sqrt2 = dsqrt(2.0d0)

            nxo = 64
            nyo = 64
            hx = sqrt2*pi
            hy = hx
            nt = 40000
            inct = 400
            dt = 0.00025d0
            ti = 0.0d0

            dx = hx/nx
            dy = hy/ny
            x = -hx/2
            do 300 j=1,nx
               y = -hy/2
               do 310 k=1,ny
      s=1.0d0+0.1d0*dcos(sqrt2*y)+1.0d-4*cos(sqrt2*(x+y))
         u(j,k)=dcmplx(s,0.0d0)
```

```
                  y = y + dy
 310              continue
                x = x + dx
 300            continue
          endif
       return
       end
c*****  End of Section 2  *****
```

The third section contains the subroutines for applying the split-step method and outputting the solution.

```
      subroutine calcv(u,v,g,f,uf,w,ndim,nx,ny,nm)
c*****************************************************************
c
c  This routine calculates v, where v satisfies
c
c          v_xx + b1*v_yy = b2*(|u|^2)_xx
c
c  and b1>0.
c
c*****************************************************************

      implicit complex*16(a-h,o-z), integer*4(i-n)
      complex*16 u(ndim,ny), v(ndim,ny), uf(ndim,ny), w(ndim,ny)
      real*8 f(2*ndim,ndim), g(ndim,ny)

      do 400 k=1,ny
         do 400 j=1,nx
            v(j,k) = cdabs(u(j,k))**2
 400     continue

      call four2(v,uf,w,f,-1,ndim,nx,ny,nm)

      do 410 k=1,ny
         do 410 j=1,nx
            uf(j,k) = g(j,k)*uf(j,k)
 410     continue

      call four2(uf,v,w,f,1,ndim,nx,ny,nm)

      return
      end
c---------------------------------------------------------------

      subroutine getmult(d2,e,g,ndim,nx,ny,hx,hy,a1,a2,b1,b2,dt)
c*****************************************************************
c
c  This subroutine calculates the multipliers used in finding v
c  and in advancing the linear term.
c
c*****************************************************************
```

```
      implicit real*8(a-h,o-z), integer*4(i-n)
      dimension g(ndim,ny), d2(ndim,ny)
      complex*16 i, const, e(ndim,ny)

      i = dcmplx(0.0d0,1.0d0)
      pi = 2.0d0*dasin(1.0d0)

      const = -4.0d0*pi*pi

      do 520 j=1,nx
         m = -nx/2 + j - 1
         cm = m*m/(hx*hx)
         do 510 k=1,ny
            n = -ny/2 + k - 1
            cn = n*n/(hy*hy)
            d2(j,k) = const*(a1*cm + a2*cn)
            e(j,k) = cdexp(d2(j,k)*dt*i)
            if ((m.eq.0).and.(n.eq.0)) then
               g(j,k) = 0.0d0
            else
                 g(j,k) = b2*cm/(cm + b1*cn)
              end if
 510       continue
 520    continue
      call rf2shift(d2,ndim,nx,ny)
      call cf2shift(e,ndim,nx,ny)
      call rf2shift(g,ndim,nx,ny)
      return
      end

      subroutine output(u,v,ndim,nx,ny,nmax,nxo,nyo,uf,w,f,tol)
c*****************************************************************
c
c  This subroutine outputs u, v and uf
c    Inputs:
c      u, v  - the solution (nx by ny matrices)
c      tol   - number of decimal digits to output
c    Outputs:
c      u, v, uf are written to files
c    Workspace:
c      w, f (nmax is used by routine four2)
c
c*****************************************************************
      implicit real*8(a-h,o-z), integer*4(i-n)
      complex*16 u(ndim,ny), v(ndim,ny), uf(ndim,ny), w(ndim,ny)
      dimension f(2*ndim,ndim)

 601  format(f7.4)
 602  format(f8.4)
 603  format(f9.4)
 604  format(f12.4)
 605  format(i1)

      call four2(u,uf,w,f,-1,ndim,nx,ny,nmax)
```

```
      call cf2shift(uf,ndim,nx,ny)
      do 610 k=1,ny
         do 610 j=1,nx
            temp = cdabs(uf(j,k))
            if (temp.lt.tol) then
               write(12,605) 0
            else if (temp.lt.100) then
               write(12,601) temp
               else
                  write(12,604) temp
            end if
 610     continue

      nxinc=nx/nxo
      nyinc=ny/nyo

      do 620 k=1,ny,nyinc
         do 620 j=1,nx,nxinc
            temp = cdabs(u(j,k))
            if (temp.lt.100) then
               write(10,601) temp
            else
               write(10,602) temp
            end if
 620     continue

      do 630 k=1,ny,nyinc
         do 630 j=1,nx,nxinc
            temp = dreal(v(j,k))
            if (dabs(temp).lt.100) then
               write(11,602) temp
            else
               write(11,603) temp
            end if
 630     continue

      return
      end

      subroutine calc_h(u,v,uf,w,f,du,d2,a3,a4,ndim,nx,ny,nm,ham,
     , el2)
c*****************************************************************
c
c  This routine calculates the Hamiltonian and l^2 norm of the
c  solution.
c    Inputs:
c      u, v  - the solution (nx by ny matrices)
c      d2    - coefficients used to calculate Lu, where L is the
c              linear part of the DS equations
c    Outputs:
c      ham   - Hamiltonian of solution
c      el2   - l^2 norm of solution
c    Workspace:
c      uf,w,du,duf (nm is used by routine four2)
```

```
c
c*****************************************************************
      implicit real*8(a-h,o-z), integer*4(i-n)
      complex*16 u(ndim,ny), v(ndim,ny), uf(ndim,ny), w(ndim,ny)
      complex*16 h, du(ndim,ny)
      dimension d2(ndim,ny), f(2*ndim,ndim)

      call four2(u,uf,w,f,-1,ndim,nx,ny,nmax)

      do 800 k=1,ny
         do 800 j=1,nx
            uf(j,k) = d2(j,k)*uf(j,k)
 800     continue
      call four2(uf,du,w,f,1,ndim,nx,ny,nmax)

      do 810 k=1,ny
         do 810 j=1,nx
            w(j,k) = cdabs(u(j,k))**2
 810     continue

      h = 0.0d0
      el2 = 0.0d0
      do 820 k=1,ny
         do 820 j=1,nx
            h = h + dconjg(u(j,k))*du(j,k)
            h = h + w(j,k)*(a3*w(j,k) + a4*v(j,k))/2.0d0
            el2=el2+w(j,k)
 820     continue
      ham = cdabs(h)/(nx*ny)
      el2 = dsqrt(el2/(nx*ny))

      return
      end


      subroutine calc_u(u,v,uf,w,f,e,g,ndim,nx,ny,nmax,a3,a4,dt)
c*****************************************************************
c  This routine does one time step of the split-step method on
c  the DS equations
c    Inputs:
c      u, v - the solution at the previous time step (nx by ny
c             matrices)
c      e, g - nx by ny matices of constants
c    Outputs:
c      u, v - the solution at the current time step
c    Workspace:
c      c, uf - used as intermediate values
c      w, f  - used by the subroutine four2 (which also needs
c              nmax)
c*****************************************************************
      implicit real*8(a-h,o-z), integer*4(i-n)
      complex*16 i,c1,c2
      parameter(i=dcmplx(0.0d0,1.0d0))
      complex*16 u(ndim,ny), v(ndim,ny), uf(ndim,ny)
```

```
      complex*16 w(ndim,ny), e(ndim,ny)
      dimension f(2*ndim,ndim), g(ndim,ny)

      c1 = dt*a3*i/2.0d0
      c2 = dt*i*a4/2.0d0

      do 900 k=1,ny
         do 900 j=1,nx
      u(j,k) = cdexp(c1*u(j,k)*dconjg(u(j,k))+c2*v(j,k))*u(j,k)
 900     continue

      call four2(u,uf,w,f,-1,ndim,nx,ny,nmax)

      do 910 k=1,ny
         do 910 j=1,nx
            uf(j,k) = e(j,k)*uf(j,k)
 910     continue

      call four2(uf,u,w,f,1,ndim,nx,ny,nmax)
      call calcv(u,v,g,f,uf,w,ndim,nx,ny,nmax)

      do 920 k=1,ny
         do 920 j=1,nx
      u(j,k) = cdexp(c1*u(j,k)*dconjg(u(j,k))+c2*v(j,k))*u(j,k)
 920     continue

      call calcv(u,v,g,f,uf,w,ndim,nx,ny,nmax)

      return
      end


      subroutine backup(u,lastu,ndim,nx,ny)
c*****************************************************************
c
c  This routine copies u into lastu.
c
c*****************************************************************
      implicit real*8(a-h,o-z), integer*4(i-n)
      complex*16 u(ndim,ny), lastu(ndim,ny)

      do 1000 k=1,ny
         do 1000 j=1,nx
            lastu(j,k) = u(j,k)
 1000       continue

      return
      end
c*****  End of Section 3  *****
```

The fourth section contains two routine used to calculate the fast Fourier transforms. These routines can be found in most numerical methods manuals. The

routines have been modified so that the code can easily be run using a vector processor.

```
      subroutine four2(u,v,w,f,isgn,ndim,nx,ny,nm)
c*****************************************************************
c
c  Subroutine four2 computes the 2-dimensional Fourier transform
c
c    Input:  u(nx,ny) -- data
c            isgn     -- sign of "i" in transform, i.e., isgn =
c                        +1 for transform and -1 for inverse.
c            nx, ny   -- number of points in x and y directions
c                        respectively.  They must each be a power
c                        of 2 (the subroutine does not check for
c                        this).
c            nm       -- maximum of nx and ny
c
c    Output: v(nx,ny) -- transform if isgn = -1, inverse if isgn =
c                        +1.
c
c    Workspace:  w(nx,ny),f(2*nm)
c
c*****************************************************************
      implicit complex*16(a-h,o-z), integer*4(i-n)
      real*8 f(2*ndim,ndim)
      complex*16 u(ndim,ny), v(ndim,ny), w(ndim,ny)

      do 100 k = 1,ny
         do 100 j = 1,nx
            f(2*j-1,k) = dreal(u(j,k))
            f(2*j,k)   = dimag(u(j,k))
100      continue
      call fft(f,ndim,nx,ny,isgn,nm)
      do 110 k=1,ny
         do 110 j = 1,nx
            w(j,k) = dcmplx(f(2*j-1,k),f(2*j,k))
110      continue
      do 130 j = 1,nx
         do 130 k = 1,ny
            f(2*k-1,j) = dreal(w(j,k))
            f(2*k,j)   = dimag(w(j,k))
130      continue
      call fft(f,ndim,ny,nx,isgn,nm)
      do 140 j=1,nx
         do 140 k = 1,ny
            v(j,k) = dcmplx(f(2*k-1,j),f(2*k,j))
140      continue
      if(isgn.eq.-1) then
          do 160 j = 1,nx
             do 160 k = 1,ny
                v(j,k) = v(j,k)/nx/ny
160          continue
      endif
      return
```

```
      end

      subroutine fft(data,ndim,nn,nl,isgn,nm)
c*****************************************************************
c
c  The subroutine FFT computes the Fourier-transform of the data,
c  by the algorithm in Numerical Recipes, p. 394.
c
c    Input:
c      data(nn,nl) -- data is nl complex arrays of nn elements or
c                     real arrays of 2*nn elements that are to be
c                     transformed.
c      nn, nl      -- dimension of data (if complex). nn must be
c                     a power of 2 (the subroutine does not check
c                     for this).
c      isgn        -- if isgn = +1 then replace data with its
c                     discrete transform, if isgn = -1 then
c                     replace data with nn  times its inverse
c                     discrete transform.
c
c    Output:
c      data(nn,nl) -- discrete transform if isgn = -1,
c                     inverse discrete transform if isgn = +1.
c*****************************************************************
      implicit real*8(a-h,o-z), integer*4(i-n)
      dimension data(2*ndim,nl)

      pi = 2.0d0*dasin(1.0d0)
      n = 2*nn
      j = 1
      do 180 i = 1,n,2
         if(j.gt.i) then
           do 165 k=1,nl
              tempr = data(j,k)
              tempi = data(j+1,k)
              data(j,k) = data(i,k)
              data(j+1,k) = data(i+1,k)
              data(i,k) = tempr
              data(i+1,k) = tempi
165         continue
         endif
         m = n/2
170      if((m.ge.2).and.(j.gt.m)) then
            j = j-m
            m = m/2
            go to 170
         endif
         j = j+m
180   continue
      mmax = 2
190   if(n.gt.mmax) then
         istep = 2*mmax
         theta = 2.0d0*pi/(isgn*mmax)
         wpr = -2.0d0*dsin(0.5d0*theta)**2.0d0
```

```
            wpi = dsin(theta)
            wr = 1.0d0
            wi = 0.0d0
            do 210 m = 1,mmax,2
               do 200 i = m,n,istep
                  j = i + mmax
                  do 195 k=1,nl
                     tempr = wr*data(j,k)-wi*data(j+1,k)
                     tempi = wr*data(j+1,k)+wi*data(j,k)
                     data(j,k) = data(i,k) - tempr
                     data(j+1,k) = data(i+1,k) - tempi
                     data(i,k) = data(i,k) + tempr
                     data(i+1,k) = data(i+1,k) + tempi
 195              continue
 200           continue
               wtemp = wr
               wr = wr*wpr-wi*wpi+wr
               wi = wi*wpr+wtemp*wpi+wi
 210        continue
            mmax = istep
            go to 190
            endif
         return
         end


         subroutine fftshift(data,n)
c*****************************************************************
c
c  Subroutine fft shift swaps the left and right hand sides of the
c  vector data.
c
c  Inputs:
c    data(n) -- data to be shifted (complex array of n elements)
c    n       -- number of elements in u, n should be even.
c
c  Output:
c    data(n) -- the data with the left and right sides swapped.
c
c*****************************************************************
         integer j,k,n
         complex*16 data(n), temp

         j = n/2
         do 220 k=1,j
            temp = data(k)
            data(k) = data(j+k)
            data(j+k) = temp
 220        continue
         return
         end


         subroutine cf2shift(u,ndim,nx,ny)
c*****************************************************************
```

```
c
c  Subroutine four2 shift swaps the diagonal quadrants of a
c  matrix of data.  Used to "unshift" the output of the 2-d
c  fourier transform.
c
c  Inputs:
c    u(nx,ny) -- data to be shifted (complex values matrix)
c    nx, ny   -- dimensions in the x and y directions. nx
c                        and ny should be even.
c
c  Output:
c    u(nx,ny) -- the data  with the top-left quadrant swapped
c                with the bottom-right and the top-right
c                swapped with the bottom-left.
c
c*****************************************************************
       implicit complex*16 (a-h,o-z), integer*4 (i-n)
       complex*16 u(ndim,ny)


       j = nx/2
       k = ny/2
       do 240 m=1,j
          do 240 n=1,k
             temp = u(m,n)
             u(m,n) = u(m+j,n+k)
             u(m+j,n+k) = temp
             temp = u(m,n+k)
             u(m,n+k) = u(m+j,n)
             u(m+j,n) = temp
 240         continue
             return
       end

       subroutine rf2shift(u,ndim,nx,ny)
c*****************************************************************
c
c  Subroutine four2 shift swaps the diagonal quadrants of a
c  matrix of data.  Used to "unshift" the output of the 2-d
c  fourier transform.
c
c  Inputs:
c    u(nx,ny) -- data to be shifted (real valued matrix)
c    nx, ny   -- dimensions in the x and y directions. nx and ny
c                should be even.
c
c  Output:
c    u(nx,ny) -- the data  with the top-left quadrant swapped
c                with the bottom-right and the top-right
c                swapped with the bottom-left.
c
c*****************************************************************
       implicit real*8 (a-h,o-z), integer*4 (i-n)
       dimension u(ndim,ny)
```

```
      j = nx/2
      k = ny/2
      do 250 m=1,j
         do 250 n=1,k
            temp = u(m,n)
            u(m,n) = u(m+j,n+k)
            u(m+j,n+k) = temp
            temp = u(m,n+k)
            u(m,n+k) = u(m+j,n)
            u(m+j,n) = temp
250         continue
      return
      end
c*****  End of Section 4  ****
```

To apply the split-step method to the DSI system the routines which initialize the multipliers and calculate v must be modified.