NEW APPROACHES FOR SOLVING NONLINEAR OSCILLATION PROBLEMS

A Thesis Submitted to
the Graduate School of Engineering and Sciences of
İzmir Institute of Technology
in Partial Fulfillment of the Requirements for the Degree of

DOCTOR OF PHILOSOPHY

in Mathematics

by
Sıla Övgü KORKUT UYSAL

June 2015
İZMİR

We approve the thesis of **Sıla Övgü KORKUT UYSAL**

Examining Committee Members:

Prof. Dr. Gamze TANOĞLU
Department of Mathematics, İzmir Institute of Technology

Prof. Dr. Emine MISIRLI
Department of Mathematics, Ege University

Prof. Dr. Siraj-ul-ISLAM
Department of Basic Sciences, N-W.F.P. University of Engineering & Technology

Assist. Prof. Dr. Fatih ERMAN
Department of Mathematics, İzmir Institute of Technology

Assist. Prof. Dr. Utku ERDOĞAN
Department of Mathematics, Uşak University

25 June 2015

Prof. Dr. Gamze TANOĞLU
Supervisor, Department of Mathematics
İzmir Institute of Technology

Prof. Dr. Oğuz YILMAZ
Head of the Department of
Mathematics

Prof. Dr. Bilge KARAÇALI
Dean of the Graduate School of
Engineering and Sciences

ACKNOWLEDGMENTS

There are several people I need to acknowledge because of their help during the completion of this thesis. First and foremost, I am indebted my deepest gratitude to my supervisor Prof. Dr. Gamze TANOĞLU whose valuable help, encouragement, patient guidance and contributive feedback enabled me to carry out this dissertation.

I also thank The Scientific and Technological Research Council of Turkey (TÜBİTAK) for its financial support during my Ph.D candidature.

I sincerely thank my friends, Nurcan GÜCÜYENEN, Neslişah İMAMOĞLU and Yeşim ÇİÇEK for all fruitful interactions, suggestions on my research and their friendly emotional supports. Many thanks also to my workmates, Tina BEŞERİ SEVİM, Oya YARKINOĞLU GÜCÜK and Selma EZBER KURTAR, for all the helpful advice. I would like to thank Barış ÇİÇEK and Berkant USTAOĞLU for their contribution on improving the technical design of this thesis.

The last but not least, I also extend my thanks to my husband, Umut UYSAL and to my family. This thesis would not have accomplished without their encouragement, patience and their endless supports.

ABSTRACT

NEW APPROACHES FOR SOLVING NONLINEAR OSCILLATION PROBLEMS

This thesis proposes two different numerical methods for solving nonlinear oscillation problems which appear in engineering and physics. Thus, the study is conducted in two parts. The first part introduces and analyzes the new iterative splitting method. In the construction of this method I utilize both the iterative splitting process and nonlinear Magnus expansion. Due to the fact that the iterative splitting procedure is employed, the constructed method can also be considered as a kind of operator splitting method. The second part presents a new linearization technique, based on the Newton-Raphson method and the Fréchet derivatives, for oscillation systems. Duffing oscillator and damped oscillator are used for testing the methods, respectively. Moreover, the proposed iterative splitting method and the proposed linearization technique are applied to both Van-der Pol equation and cubic nonlinear Schrödinger equation. Although the examples considered are a small sample of nonlinear oscillation equations, it is believed that the methods are easily adapted to solve such problems numerically.

ÖZET

LİNEER OLMAYAN TİTREŞİM PROBLEMLERİNİ ÇÖZMEK İÇİN YENİ YAKLAŞIMLAR

Bu tez, mühendislik ve fizik alanında karşılaşılan lineer olmayan titreşim problemlerinin çözümleri için iki farklı sayısal metot önermektir. Bu yüzden çalışma iki parça halinde yönetilmektedir. Birinci bölüm yeni iteratif ayırma metodunu tanıtmakta ve analiz etmektetir. Bu metodun oluşturulmasında hem iteratif ayırma sürecinden hem de lineer olmayan Magnus açılımından yararlanılmıştır. Metodun oluşturulmasında iteratif ayırma yönteminin kullanılmasından dolayı önerilen metod aynı zamanda operatör ayırma metodun bir çeşidi olarak da ele alınabilir. İkinci bölüm titreşim problemleri için, Newton-Raphson metodu ve Fréchet türevlerini baz alan, yeni lineerizasyon tekniği sunar. Metotları test edebilmek için sırasıyla Duffing osilatör ve sönümlü osilatör kullanılmıştır. Ayrıca, önerilen iteratif ayırma metodu ve önerilen lineerizasyon tekniği hem Van-der Pol denklemi hem de kübik lineer olmayan Schrödinger denklemine uygulanmıştır. Uygulamalarda lineer olmayan titreşim problemlerinin az sayıda örneklerinin düşünülmüş olmasına rağmen metodun bu şekildeki problemlere sayısal olarak kolayca adapte edilebileceğine inanılmaktadır.

TABLE OF CONTENTS

LIST OF FIGURES viii

LIST OF TABLES x

CHAPTER 1. INTRODUCTION 1
- 1.1. Introduction 1
- 1.2. Layout of Thesis 4

CHAPTER 2. THE PROPOSED ITERATIVE SPLITTING METHOD (PISM) 6
- 2.1. Derivation of PISM 6
  - 2.1.1. Traditional Operator Splitting Methods 6
  - 2.1.2. Construction of the Proposed Method 7
- 2.2. Convergence Analysis of PISM 10
  - 2.2.1. Analysis for Bounded Operators 11
  - 2.2.2. Analysis for Unbounded Operators: Abstract Analysis 16
  - 2.2.3. Convergence Results 25

CHAPTER 3. NUMERICAL TESTS AND SIMULATIONS: THE PROPOSED ITERATIVE SPLITTING METHOD 29
- 3.1. Duffing Equation 29
- 3.2. Van-der Pol Equation 32
- 3.3. Nonlinear Schrödinger Equation 35
  - 3.3.1. Equation in the form: $i\hbar\partial_t\Psi = \beta\partial_x^2\Psi + \alpha|\Psi|^2\Psi$ 36
  - 3.3.2. Equation in the form: $i\hbar\partial_t\Psi = \beta\partial_x^2\Psi + (G(x) + \alpha|\Psi|^2)\Psi$ 40

CHAPTER 4. AN ALTERNATIVE METHOD: THE PROPOSED LINEARIZED METHOD 44
- 4.1. Derivation of the Method 44
- 4.2. Application to the Three Oscillation Problems 46
  - 4.2.1. Damped oscillator 47
  - 4.2.2. Van-der Pol Equation 48
  - 4.2.3. Nonlinear Schrödinger Equation 50

CHAPTER 5. NUMERICAL TESTS AND SIMULATIONS: THE PROPOSED LINEARIZED METHOD 52
  5.1. Damped Oscillator 52
  5.2. Van-der Pol Equation.. 54
  5.3. Nonlinear Schrödinger Equation 56
    5.3.1. Equation in the form: $i\partial_t\Psi + \beta\partial_x^2\Psi + \alpha|\Psi|^2\Psi = 0$ 56
    5.3.2. Equation in the form: $i\partial_t\Psi + \beta\partial_x^2\Psi + (G(x) + \alpha|\Psi|^2)\Psi = 0$ 60

CHAPTER 6. CONCLUSION 65

REFERENCES 67

APPENDICES

APPENDIX A. MAGNUS EXPANSION 71
APPENDIX B. EXPONENTIAL INTEGRATORS 73
APPENDIX C. ANALYTICAL FRAMEWORK 75
APPENDIX D. MAT-LAB CODES FOR NEW ITERATIVE SPLITTING 77
APPENDIX E. MAT-LAB CODES FOR FRÉCHET TECHNIQUE 97

LIST OF FIGURES

Figure **Page**

Figure 2.1. Diagram for the proposed iterative splitting method. 10

Figure 3.1. The Hamiltonian of Duffing Equation (3.1) for $\varepsilon = 1$ on $[0, 10]$. 32

Figure 3.2. The Hamiltonian of Duffing Equation (3.1) for $\varepsilon = 3$ on $[0, 10]$. 32

Figure 3.3. The solution of Van-der Pol Equation (3.6) using PISM on the left, ODE23s on the right. 33

Figure 3.4. Trajectories of Van-der Pol Equation (3.6) for $\mu = 10$ PISM on the left and ODE23s on the right. 34

Figure 3.5. The Hamiltonian of Van-der Pol Equation (3.6) for $\mu = 1$ on $[0, 60]$. 34

Figure 3.6. On the left PISM, exact solution on the right. 38

Figure 3.7. Plots of the global energy error vs time and the absolute error vs. time are presented for **Case 1**. 39

Figure 3.8. The numerical solutions are obtained on $x \in [-20, 5]$ and $t \in [0, 3]$ where $N_x = N_t = 100$. 40

Figure 3.9. Global Energy Error is on the left and the comparisons of numerical solution and exact solution for fixed time on the right for **Case 2**. 41

Figure 3.10. Probability density of the particle in Equation (3.26) for PISM. 42

Figure 3.11. Probability density of the particle in Equation (3.26) for SMS. 42

Figure 4.1. Diagram for the proposed linearized method. 46

Figure 5.1. Exact solution and PLM for solving equation in 4.10. 53

Figure 5.2. PLM vs ODE45 where $\alpha = 1$. 54

Figure 5.3. PLM vs. ODE23s obtained for various values of μ. 54

Figure 5.4. Limit cycle obtained from the solutions from PLM and ODE23s. 55

Figure 5.5. The Hamiltonian of Van-der Pol Equation (3.6) for $\mu = 1$ on $[0, 60]$. 55

Figure 5.6. The numerical solutions are obtained for $\Delta x = 0.25$ and $\Delta t = 0.01$ where $N_x = N_t = 100$. 57

Figure 5.7. The numerical solutions are obtained for $t \in [0, 1]$ and $x \in [-15, 10]$ where $N_x = N_t = 100$. The exhibited figures belong to global energy error on the left and the absolute error on the right. 57

Figure 5.8. Comparison of the numerical solution and the exact solution of the Equation (5.2) with the initial condition given in (5.7) for various values of t_{end}. .. 62

Figure 5.9. Exact and the numerical solutions of Equation (5.2) at certain time. 63

Figure 5.10. Probability density of the particle in Equation (5.9) for PLM. 63

Figure 5.11. Probability density of the particle in Equation (5.9). PLM is given on the right whereas the RK2 method on the left. 64
Figure 5.12. The numerical solutions obtained from the RK2 method for $t \in [0, 5]$. 64

LIST OF TABLES

Table **Page**

Table 3.1. Comparison of errors for several h on [0, 10] interval with various methods where $\varepsilon = 10^{-4}$. The expected order is 2. 30

Table 3.2. Numerical errors are established for different Δt and different ε on $[0, 1]$. ... 31

Table 3.3. Estimated errors using L_∞, L_2 and L_1 norm for conservation of the mass with $N_x = N_t = 100$ where $\Delta x = 0.25$. 39

Table 3.4. Estimated errors using L_∞, L_2 and L_1 norm for conservation of the mass with $N_x = N_t = 100$ where $\Delta x = 0.25$. 40

Table 3.5. The CPU runtimes are measured in seconds for different N_t and N_x on $x \in [-20, 5]$ and $t \in [0, 3]$. 41

Table 3.6. Elapsed time for different N_t and N_x. 43

Table 5.1. Numerical errors for different Δt. Estimated errors using L_∞, L_2 and L_1 norm are obtained comparing with the analytical solution on $[0, 1]$. 53

Table 5.2. Estimated errors using L_∞, L_2 and L_1 norm for conservation of the mass with $N_x = N_t = 100$ where $\Delta x = 0.25$. 58

Table 5.3. Estimated errors using L_∞, L_2 and L_1 norm for conservation of the mass where $t \in [0, 3]$ and $x \in [-20, 5]$. 60

Table 5.4. Estimated errors using L_∞, L_2 and L_1 norm for conservation of the mass where $t \in [0, 20]$ and $x \in [-20, 5]$. 60

Table 5.5. The elapsed time is measured in seconds where $t \in [0, 3]$ and $x \in [-20, 5]$. . 60

CHAPTER 1

INTRODUCTION

Universally, many phenomena exhibit substantially nonlinear behavior. Thus, nonlinear differential equations appear naturally in engineering and science. Because of this, remarkable attention has been paid to the solution of nonlinear problems. The aim of this thesis is to propose two different approaches for numerical solutions of nonlinear oscillation problems.

1.1. Introduction

Nonlinear oscillations play important role in engineering since a great deal of engineering components arise from vibrating systems that may be modeled by oscillatory systems, see (Nayfeh and Mook, 1995), (Fidlin, 2006) and (Dimarogonas and Haddad, 1992). Solving the governing equation and the existence of the exact solution are problematic for vibration problems. Although the traditional perturbation theory may be used to provide a solution, if there is no small parameter, this theory does not work. To approximate the solution there are considerable studies in the literature such as homotopy analysis methods (Liao and Cheung,1998), the harmonic balance method (Mickens, 2001), the homotopy perturbation method (He, 1999), He's energy balanced method (Ganji et al., 2008) etc.

A great deal of effort have been also spent for developing numerical solutions of nonlinear oscillation problems. The differential transform method is used in (Tabatabaei and Günerhan, 2014) while the Taylor matrix method has been improved for solving Duffing Equation in (Bülbül and Sezer, 2013). A compact finite difference scheme has been applied to the nonlinear Schrödinger equation in (Dehghan and Taleei, 2010). Another technique, based on finding periodic solutions approximated by a trigonometric series, has been proposed for second and fourth order Duffing equation in (Groves, 1983). Operator splitting method can be seen in the study of *Lubich* in (Lubich, 2008).

From a numerical point of view, the operator splitting method is one of the most known techniques in the literature. Because the time derivative of the described equation depends on collecting the operators corresponding to the different processes, it is not always possible to derive an effective method for solving the sum of these operators. The idea behind the operator splitting procedures is instead of the sum, apply an effective numerical method to each of the operators separately on the condition of accurate solutions. The beauty of the splitting of the equation into sub-equations is that each sub-equation easy to comprehend. Moreover, these sub-equations often lead to major facilities in the analysis of the original equation, see the survey by (McLachlan and Quispel, 2002) and (Holden, Karlsen, Lie and Risebro, 2010).

The idea of sequential splitting, known as Lie-Trotter splitting, dates back to the 1950s. In (Bagrinovskii and Gudunov, 1957), this method was first applied to a partial differential equation. During 1960-1970, the first splitting methods were developed. They were connected with finite difference methods. A renovation of the methods was performed in the 1980s while using the methods for complex processes underlying partial differential methods (Crandall and Majda, 1980). The simplest type is sequential splitting, which in terms of the local splitting error is first order accurate in time. A second order and more popular method is Strang splitting in (Marchuk, 1968) and (Strang, 1968). The split-step method for the nonlinear Schrödinger equation by (Hardin, Tappert, 1973), which is the first example of a symplectic integrator for a partial differential equations(PDEs).

There is another sort of splitting method, iterative splitting, which is a recently popular technique of operator splitting methods. The idea of the iterative splitting method goes back at least to (Kelly, 1995) in 1995. In the study of *Kanney et al.*, (Kanney et al., 2003), the authors focused on the convergence of an iterative splitting procedure for nonlinear reactive transport problems. *Faragó* and *Geiser* suggest a new scheme which is based on the combination of splitting time interval and traditional iterative operator splitting in 2007, see (Faragó and Geiser, 2007). That technique is used in (Geiser, 2008); (Geiser, 2008). The iterative splitting method combining with the Newton's method was studied in (Geiser and Noack, 2011) for a nonlinear equation. We proposed a method for solving linear time-dependent differential equations in (Tanoğlu and Korkut, 2012). The question of whether the given method in (Tanoğlu and Korkut, 2012) can be extended to nonlinear problems motivates our work. The essential tools in the construction of this new scheme include the iterative splitting process and the Magnus expansion.

Thesis is concerned with non-linear evolutionary problems

$$\frac{d}{dt}u(t) = A(t,u)u(t), \quad 0 \leq t \leq t_{end} \tag{1.1}$$

$$u(0) = u_0 \in X \tag{1.2}$$

where the structure of the unbounded nonlinear operator $A(t,u) \in [0,t_{end}] \times X$, suggests a decomposition into two parts

$$A(t, u(t))u(t) = Tu(t) + V(t, u(t))u(t), \quad u(t) : \mathbb{D}(T) \cap \mathbb{D}(V)$$

with the unbounded operator $T : \mathbb{D}(T) \subset X \rightarrow X$ and bounded operator $V(t, u) : \mathbb{D}(V) \in [0, t_{end}] \times X \rightarrow X$. In the remainder of this thesis X is real or complex Banach space. Although we apply suitable difference approximation techniques in order to solve differential operator from the numerical point of view, to analyze the convergence of the given methods we follow two approaches:

- When T and $V(t)$ are bounded, we use Taylor series expansion.
- When T is unbounded and $V(t)$ is bounded, we use C_0 semigroup approaches combining with the exponential integrator.

Various nonlinear evolutionary equations were recently analyzed by using the splitting methods. Descombes and Thalhammer, (Descombes and Thalhammer, 2013), use Lie-Trotter splitting method was used to deduce the error analysis of non-linear Schödinger equation. In (Lubich, 2008), cubic nonlinear Schrödinger equation was analyzed for Strang splitting method. The analysis of Strang splitting method for Vlasov-type equations was given in (Einkemmer and Ostermann, 2013). The error analysis of higher order splitting method was carried out in (Koch et al., 2013).

Our analysis combines C_0 semigroup approaches with the exponential integrator. The exponential integrator is a tool to analyze the convergence property of the new iterative splitting method. A brief introduction to exponential integrators is given in Appendix B. Our general reference on the exponential integrator is the survey of *Hochbruck and Ostermann*, (Hochbruck and Ostermann, 2010).

The second part of this thesis consists of the alternative linearization technique based on the Fréchet derivative and Newton-Raphson method. The combination of these two techniques is nowadays popular for solving non-linear equations. It originated with Liu & Wu, (Liu and Wu, 2000), to solve ordinary differential equations of Duffing-type non-linearity. In that study, the method is combined with the generalized differential quadrature rule (GDQR). This method was also applied to Blasius and Onsager equations in (Liu and Wu, 2001).

The method also appears in (Fazel et al., 2013), which is our starting point in. The authors used the Fréchet derivative to convert the nonlinear differential equation into a linear one via an iterative process. Likewise, all nonlinear term are left in the old step of variables in an iteration procedure. Henceforth, Fréchet derivative is useful and efficient in solving nonlinear differential equations. The intention of this thesis is to solve oscillation problems as a system using Fréchet derivative combined with central difference method.

1.2. Layout of Thesis

This thesis presents two different approaches for solving nonlinear oscillation problems. The first approach is discussed in Chapter 2 and 3; Chapter 4 and 5 discuss the second approach.

Chapter 2 aims to construct and analyze a new iterative method for the numerical solutions of the oscillation equations in the classes of nonlinear science. The idea behind the construction is based on both the iterative splitting procedure and the Magnus expansion. A short overview of Magnus expansion is provided in Appendix A. Due to the iterative process this method can also be considered as a linearization technique. Furthermore, we provide an overview and comparison with other traditional splitting methods involving the Magnus expansion. For simplicity we restrict ourselves to second order splitting methods. Section 2.2 analyze of the developed method. We focus on consistency, stability and order. The method is analyzed for not only bounded operators but also for unbounded ones. To establish the convergence results we impose extra assumptions both in the bounded and the unbounded cases. At the end we utilize the "Lady windermere's fan" argument to complete the proof of the convergence.

Various numerical examples of the method are given in Chapter 3. The effectiveness is demonstrated by comparing other splitting methods and ODE23s code in MATLAB. We test the performance of our construction on the Duffing equation. We observe that the numerical solutions are in perfect agreement with the theoretical results. Although the traditional splitting methods have the reduction of order, our method achieves the expected order, namely 2. In addition to this, we test our method on different oscillation problems, such as: Van-der Pol equation and nonlinear Schrödinger equation. Numerical simulations and tests are illustrated in this Chapter 3.

An alternative method is proposed in Chapter 4 and is based on the Newton- Raphson and the Fréchet derivatives. Throughout the thesis, it is called as PLM. A brief overview of these concepts is provided in Appendix C. We introduce the method for nonlinear systems. In order to see how to apply the proposed method to any nonlinear differential equations we carry out various nonlinear oscillation equations.

Chapter 5 deals with the numerical tests and simulations for the method proposed in Chapter 4. The purpose of Chapter 5 is to test the suggested method on various oscillation problems. We use the damped oscillation equation, the Van-der Pol equation and the nonlinear

Schrödinger equation and its solitary waves. The testing parameters are the accuracy, the CPU time and the conservation of the qualitative properties of the equations.

Finally, Chapter 6, ends with several remarks and a brief conclusion.

CHAPTER 2

THE PROPOSED ITERATIVE SPLITTING METHOD (PISM)

This chapter develops and analyzes the proposed iterative splitting method (PISM). Section 2.1 is devoted to the construction of PISM after a brief introduction for the traditional splitting methods. The convergence analysis is investigated in Section 2.2 for not only bounded operators but also unbounded operators.

2.1. Derivation of PISM

The particular goal of this section is to develop a kind of the second operator splitting methods combining with some efficient tools. In order to accomplish this, we investigate the numerical scheme for solving the operators separately providing the efficiency. Therefore, recall the nonlinear evolutionary problems

$$\frac{\partial u}{\partial t} \quad = \quad A(t,u)u(t), \quad u(0) = u_0, \qquad 0 \leq t \leq t_{end}, \tag{2.1}$$

where $A(t,u) \in [0,t_{end}] \times X$. Assume that $A(t,u(t))u(t) = Tu(t) + V(t,u(t))u(t), \quad u(t) : \mathbb{D}(T) \cap \mathbb{D}(V) \subset X \to X$ with the linear operator $T : \mathbb{D}(T) \subset X \to X$ and nonlinear operator $V(t,u) : \mathbb{D}(V) \in [0,t_{end}] \times X \to X$. In Subsection 2.1.1, the frequently employed second order splitting methods are combined with the nonlinear Magnus expansion. Appendix A has an overview on the nonlinear Magnus expansion. Subsection 2.1.2 suggests a new iterative splitting method combining the iterative splitting process and the nonlinear Magnus expansion.

2.1.1. Traditional Operator Splitting Methods

Here our main objective is to produce formulas of the well-known second order splitting methods by means of the Magnus expansion. For that purpose, we focus on Equation

(2.1) and then separate it into two sub-equations as follows:

$$\frac{\partial w}{\partial t} = Tw(t), \quad w(0) = u_0 \tag{2.2}$$

$$\frac{\partial v}{\partial t} = V(t,v)v(t), \quad v(0) = w(t). \tag{2.3}$$

Note that the numerical solution at time $t = nh$, as $h \to 0$, is given by $u_{n+1} = \Phi_T^h \circ \Phi_{V(t,u)}^h u_n$, in which Φ_T^h and $\Phi_{V(t,u)}^h$ are numerical flow of Equation (2.2) and Equation (2.3), respectively.

Next consider the second order splitting methods. For convenience of the reader we give the numerical solutions on $[0,h]$.

- Starting with Strang splitting method, the solution is

$$U_{sp}(h) = e^{T\frac{h}{2}} e^{\Omega_V(h,u(h))} e^{T\frac{h}{2}} u_0. \tag{2.4}$$

  Due to the Magnus expansion in Appendix A for the numerical flow of $\Omega_V(h,u(h))$, we obtain

$$U_{sp}(h) = e^{T\frac{h}{2}} e^{\frac{h}{2}(V(0,u_0)+V(h,u(h)))} e^{T\frac{h}{2}} u_0. \tag{2.5}$$

- Alternatively, using symmetrically weighted splitting,

$$U_{sp}(h) = \frac{1}{2}(e^{Th} e^{\Omega_V(h,u(h))} + e^{\Omega_V(h,u(h))} e^{Th}) u_0. \tag{2.6}$$

  With the help of the Magnus expansion which is defined in Appendix A for the numerical flow of $\Omega_V(h,u(h))$. Thus, we have

$$U_{sp}(h) = \frac{1}{2}(e^{Th} e^{\frac{h}{2}(V(0,u_0)+V(h,u(h)))} + e^{\frac{h}{2}(V(0,u_0)+V(h,u(h)))} e^{Th}) u_0 \tag{2.7}$$

For more details, see (Bátkai, Csomós and Nickel, 2009). We will compare the above two approaches with PISM in Chapter 3. These two splitting methods are given because of this comparison.

2.1.2. Construction of the Proposed Method

This subsection a whole develops the new method. The keypoint of the new approach is to combine the iterative splitting process, which is a class of operator splitting methods, with the Magnus expansion. We start with the general initial value problem given in (2.1) on the time interval $[0, t_{end}]$ where $t_{end} \in \mathbb{R}^+$.

Our objective is to extend the method, which is constructed for non-autonomous linear problems in (Tanoğlu and Korkut, 2012), to nonlinear problems. For the sake of clarity, the second order iterative process is described below on each subinterval $[0, h]$,

$$\dot{u}_1 = Tu_1 + V(t, u_0)u_0 \quad u_1(0) = u_0, \tag{2.8}$$

$$\dot{u}_2 = Tu_1 + V(t, u_2)u_2 \quad u_2(0) = u_0. \tag{2.9}$$

In general, the formal solution of the sub-equations given in (2.8) and (2.9) on the time interval $[t, t+h]$ can be written as

$$u_i(t+h) = \Phi_i(t+h, t)\, U(t) + \int_t^{t+h} \Phi_i\,(t+h, s) F_i(s)\, ds, \quad i = 1, 2 \tag{2.10}$$

where $F_1 = V(t, U(t))U(t)$, $F_2 = Tu_1(t)$ and

$$\begin{aligned} \Phi_1(t+h, t) &= e^{hT} \\ \Phi_2(t+h, t) &= e^{\Omega_V(h, u(h))}. \end{aligned}$$

Due to the time-dependence and nonlinearity of $V(t, u)$, $\Phi_2(t+h, t)$ is expressed by means of the nonlinear Magnus expansion. Here $\Omega_V(h, u(h))$ is as follows:

$$\Omega_V(h) = \int_0^h V(s, e^{\Omega^{[1]}} u_0) ds. \tag{2.11}$$

We utilize from *trapezoidal rule* in order to get the second order Magnus expansion which is in accordance with the order of method. It leads to

$$\Omega_V(h) = \int_0^h V(s, e^{\Omega^{[1]}} u_0) ds = \frac{h}{2}\Big(V(0, u_0) + V(h, e^{\Omega^{[1]}} u_0)\Big) + O(h^3), \tag{2.12}$$

where $e^{\Omega^{[1]}}u_0$ corresponds the Equation (2.13). Detailed discussion for the nonlinear Magnus expansion can be found in (Casas and Iserles, 2006). Substituting in (2.14)

$$\begin{aligned} \Phi_1(t+h,t) &= e^{hT} \qquad (2.13) \\ \Phi_2(t+h,t) &= e^{\frac{h}{2}[V(t,u(t))+V(t+h,u_1(t+h))]}. \qquad (2.14) \end{aligned}$$

In regard to the derivation of the proposed method, we approximate the integral in (2.10) using the trapezoidal rule to obtain. Thus, we have

$$\int_t^{t+h} \Phi_i F_i \, ds = \frac{h}{2}[F_i(t+h) + \Phi_i(t+h,t)F_i(t)] + O(h^3). \qquad (2.15)$$

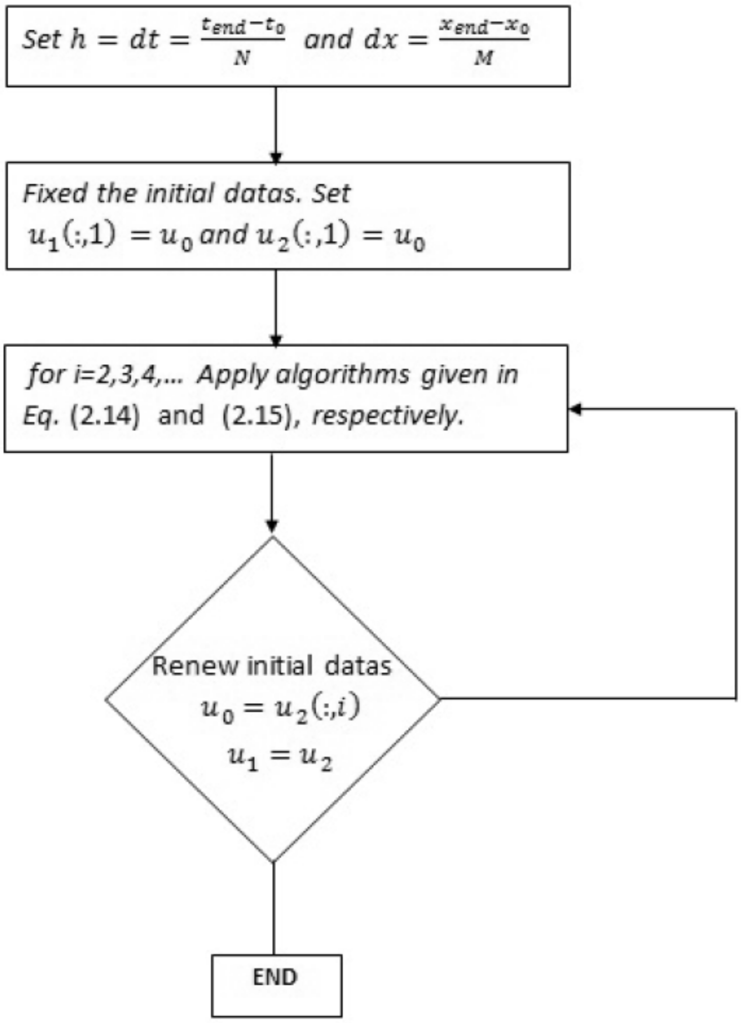
Note that $\Phi_i(t+h, t+h) = I$. Combining approximation (2.15) with the iterative schemes (2.8), (2.9) and rearranging expressions, we obtain the first order approximation

$$u_1^{n+1} = e^{Th}[u_1^n + \frac{h}{2}V(t_n, u_0(t_n))u_0^n] + \frac{h}{2}V(t_n+h, u_0^{n+1})u_0^{n+1}, \qquad (2.16)$$

and the second order approximation

$$u_2^{n+1} = e^{\frac{h}{2}[V(t_n,u^n)+V(t_n+h,u_1^{n+1})]}[U(t_n) + \frac{h}{2}Tu_1^n] + \frac{h}{2}Tu_1^{n+1}, \qquad (2.17)$$

where $U(t_n+h) = u_2(t_n+h)$ and $u_i^n = u_i(t_n)$. Repeat this procedure by taking $u_0(t_n) = u_2(t_n+h)$ for next interval until the desired time t_{end} is reached.



Set $h = dt = \frac{t_{end}-t_0}{N}$ and $dx = \frac{x_{end}-x_0}{M}$
Fixed the initial datas. Set
$u_1(:,1) = u_0$ and $u_2(:,1) = u_0$
for i=2,3,4,... Apply algorithms given in Eq. (2.14) and (2.15), respectively.
Renew initial datas
$u_0 = u_2(:,i)$
$u_1 = u_2$
END

Figure 2.1. Diagram for the proposed iterative splitting method.

2.2. Convergence Analysis of PISM

This section is devoted to analyze the convergence of the method given in the previous section. To do so, we study separately two cases:

- When T and $V(t)$ are bounded, we use Taylor series expansion.
- When T is unbounded and $V(t)$ is bounded, we use C_0 semigroup combined with the exponential integrator.

For the convergence issue, we use the concepts of consistency, stability and order. In our analysis, we define an operator norm $||.||_{X \leftarrow X}$ on a (complex) Banach space X, $(X, ||.||_X)$. To simplify the notations, we write $||.||$ instead of $||.||_X$. Throughout the analyses $u(h)$ and $U(h)$ represent exact solution and numerical solution, respectively.

Lastly, we utilize the "Lady Windermere's fan" argument for the convergence results of both bounded and unbounded cases.

2.2.1. Analysis for Bounded Operators

Although T is unbounded operator, it is possible to accept to be bounded by means of any difference approximation. We also assume that $V(t,u)$ is bounded operator. In our proofs, we use the following auxiliary assumptions:

Assumption 2.1 *There are non-negative constant $\tilde{K}$, $\tilde{R}$ and $\tilde{M}$ with*

$$\sup_{0\le t\le t_{end}} ||V(t,u)|| \le \tilde{K},$$
$$||T|| \le \tilde{R} \quad \text{on} \quad 0 \le \mathrm{t} \le \mathrm{t_{end}},$$
$$||u(t)|| \le \tilde{M} \quad \text{on} \quad 0 \le \mathrm{t} \le \mathrm{t_{end}}.$$

Assumption 2.2 *In regard to the Assumption 2.1, there exists a non-negative constant such that* $\sup_{0\le t\le t_{end}} ||A(t,u)|| \le \tilde{S}$.

Assumption 2.3 *Let Assumption 2.1 hold. $V(t,u)$ is bounded linear operator on some Banach space X. Due to the equation (A.5) and the Assumption 2.1, we get*

$$e^{\Omega_V(t,u(t))} \quad \le \quad e^{t||V(t,u(t))||} \le e^{t\tilde{K}}, \tag{2.18}$$

where $\Omega_V(t,u) \approx \Omega^{[2]}(t)$. As the convergence of Magnus expansion, we refer to (Casas and Iserles, 2006) and the references therein.

Assumption 2.4 *For existence of Equation (1.1), we guaranteed that the Lipschitz condition is satisfied. That is,*

$$||A(t,u) - A(t,\check{u})|| \le L_A||u - \check{u}||.$$

Furthermore, it is valid for $V(t, u)$ operator as follows,

$$||V(t, u) - V(t, \breve{u})|| \leq L_V ||u - \breve{u}||.$$

The following remark is a direct consequence of the *Assumptions* 2.3 and 2.1.

Remark 2.1 *Due to the Assumption 2.1 and Assumption 2.3, there exist C_1 and C_2 such that*

$$||\Phi_1(t + h, s)|| \leq C_1, \quad 0 \leq s \leq t_{end},$$
$$||\Phi_2(t + h, t)|| \leq C_2, \quad 0 \leq s \leq t_{end}.$$

Under these conditions, the following consistency and stability analyses are obtained for the proposed iterative splitting method.

Proposition 2.1 *The proposed iterative splitting is first order for only one iteration given in (2.8) with error bound*

$$||u(h) - U(h)|| \quad \leq \quad \beta h^2. \tag{2.19}$$

Here β only depends on L_V, C_1, $\tilde{S}$ and $\tilde{\mathrm{M}}$.

Proof

We denote the local error by $e_j = U(t_j) - u(t_j)$, $j = 0, 1, 2, \ldots, n.$ For simplicity we only consider the time interval [0,h]. The exact solution of Equation (1.1) is

$$u(h) \quad = \quad e^{Th} u_0 + \int_0^h e^{T(h-s)} V(s, u(s)) u(s) ds. \tag{2.20}$$

A primary tool for the derivation of local error representation is variation-of-constant formula. The first order numerical solution of (2.1) is

$$U(h) \quad = \quad e^{Th} u_0 + \int_0^h e^{T(h-s)} V(s, u_0) u_0 ds. \tag{2.21}$$

Subtracting (2.45) from (2.46) leads to

$$||u(h) - U(h)|| \quad = \quad || \int_0^h e^{T(h-s)} V(s, u(s)) - V(s, u_0) ds||.$$

Assumption 2.4 and *Remark 2.1* give

$$||u(h) - U(h)|| \quad \leq \quad hL_V C_1 ||u(s) - u_0||. \tag{2.22}$$

To obtain the error bound for $||u(h) - u_0||$ we use *Assumption 2.1* and *Assumption 2.2*,

$$\begin{aligned} u(h) &= u_0 + \int_0^h A(s, u(s))u(s)ds \\ ||u(h) - u_0|| &= h||A(s, u(s))u(s)|| \leq h\tilde{S}||u|| \\ ||u(h) - u_0|| &\leq h\tilde{S}\tilde{M}. \end{aligned} \tag{2.23}$$

By substituting (2.23) in Equation (2.22), we get

$$||u(h) - U(h)|| \quad \leq \quad h^2 L_V C_1 \tilde{S}\tilde{M} = \beta h^2, \tag{2.24}$$

where $\beta = L_V C_1 \tilde{S}\tilde{M}$. □

Proposition 2.2 *The new iterative splitting is the second order for two iterations given in Equation (2.8) and Equation (2.9) with error bound*

$$||u(h) - U(h)|| \quad \leq \quad \gamma h^3. \tag{2.25}$$

Here γ only depends on C_2, $\tilde{R}$, and *β.*

Proof The proof follows the lines of previous one. We start with writing the exact solution as follows:

$$u(h) \quad = \quad \Phi_2(t + h, t)u_0 + \int_0^h \Phi_2(h, s)Tu(s)ds, \tag{2.26}$$

where the numerical solution is

$$U(h) \;\; = \;\; \Phi_2(t+h,t)u_0 + \int_0^h \Phi_2(h,s)Tu_1 ds. \tag{2.27}$$

To estimate the error bound subtract Equation (2.27) from Equation (2.26), we have

$$u(h) - U(h) \;\; = \;\; \int_0^h \Phi_2(h,s)T[u(h-s) - u_1]ds.$$

After taking norm, *Assumption* 2.1 and *Remark* 2.1 imply

$$||u(h) - U(h)|| \;\; \leq \;\; h\, C_2\, \tilde{R}\, ||u(h-s) - u_1||, \tag{2.28}$$

where u_1 is the solution of Equation (2.8). Using the bound given in Equation (2.24), we have

$$||u(h) - U(h)|| \;\; \leq \;\; h\, C_2\, \tilde{R} \beta\, h^2 = h^3 \gamma \tag{2.29}$$

where the constant γ depends on C_2, β and $\tilde{\mathrm{R}}$. Note that the constant C_2 depends on $\tilde{K}$ because of *Remark* 2.1. □

Proposition 2.3 *The new first order iterative splitting scheme is stable on* $[0, t_{end}]$ *if* $\Phi_1(h,s) \leq 1$ *for* $0 \leq s \leq h$.

Proof To prove the stability of the new first order method, we employ standard techniques. Start by rewriting the formulation of the method in the following form

$$U^1 = U(h) \;\; = \;\; \Phi_1(h,s) + G_1, \quad U^0 = u_0, \tag{2.30}$$

where

$$G_1 = \int_0^h e^{T(h-s)} V(s,u_0) u_0,$$

which is bounded by $||G_1|| \leq hC_1\tilde{K}||u_0||$ where C_1 is the bound for $\Phi_1(t+h,s)$ defined in *Remark* 2.1. By rearranging Equation (2.30),

$$\begin{aligned} ||U^1|| &= ||e^{Th}U^0 + G_1|| \leq ||e^{Th}U^0|| + ||G_1|| \\ ||U^1|| &\leq C_1\,||u_0|| + hC_1\,\tilde{K}\,||u_0|| = (C_1\,(1+h\tilde{K}))\,||u_0||. \end{aligned} \tag{2.31}$$

Recursively we get the stability polynomial for iterative scheme at first order,

$$||U^n|| \leq {C_1}^n(1+h\tilde{K})^n||u_0|| = \zeta\,||u_0||, \tag{2.32}$$

where $\zeta = {C_1}^n e^{t_{end}\tilde{K}}$. As a result, if

$$||\Phi_1(h,s)|| \leq C_1 \leq 1,$$

then the proof follows. □

Proposition 2.4 *Similarly, the new second order iterative splitting scheme is stable on* $[0, t_{end}]$ *if* $\Phi_2(h,s) \leq 1$ *for* $0 \leq s \leq h$.

Proof

In order to obtain the stability bound for second order method, we employ the same line with the proof of Proposition 2.3. Then Equation (2.9) is rewritten as in the following form

$$U^1 = U(h) = \Phi_2(t+h,t)U^0 + G_2, \quad U^0 = u_0, \tag{2.33}$$

where

$$G_2 = \int_0^h \Phi_2(h,s)Tu_1,$$

which is bounded by $||G_2|| \leq hC_2\tilde{R}||u_1||$. As we mentioned in *Remark* 2.1, C_2 is the bound for the fundamental set of solution of (2.9), namely $\Phi_2(h, s)$. By rearranging the equation (2.33),

$$\begin{aligned} ||U^1|| &= ||C_2U^0 + G_2|| \leq C_2||U^0|| + ||G_2|| \\ ||U^1|| &\leq C_2||u_0|| + hC_2\tilde{R}||u_1|| \\ ||U^1|| &\leq C_2\,||u_0|| + hC_2\,\tilde{R}\,(C_1(1 + h\tilde{K}))||u_0|| = (1 + hJ)\,C_2\,||u_0||. \end{aligned} \tag{2.34}$$

Here $J = \tilde{R}\,(C_1(1 + h\,\tilde{K}))$. Recursively we get the stability polynomial for iterative scheme at second order,

$$||U^n|| \leq C_2{}^n(1 + hJ)^n||u_0|| = C_2{}^n e^{t_{end}J}||u_0||. \tag{2.35}$$

As a consequence,the statement holds provided that

$$||\Phi_2(h, s)|| \leq C_2 \leq 1.$$

□

2.2.2. Analysis for Unbounded Operators: Abstract Analysis

This subsection focuses on the abstract analysis of the proposed iterative splitting method which introduced in Subsection 2.1.2. For the convergence, we will investigate consistency, stability and order issues. We assume that T is unbounded whereas $V(t, u)$ is bounded operator. In our proofs, we will use the following auxiliary assumptions:

Assumption 2.5 *Let X be the Banach space with the norm $||.||$. We assume T is a linear operator on X and that T is the infinitesimal generates C_0 semigroup e^{tT} on X. Particularly, this assumption implies that there exist constants such that C and ω such that*

$$||e^{tT}|| \leq Ce^{\omega t} \leq 1, \quad t \geq 0. \tag{2.36}$$

Assumption 2.6 *In order to simplify the calculations, we denote $V(t, u(.))u(.)$ as $g(u(.))$ through-*

out this section. Then, there exists a constant $\alpha_k, \; k = 0, 1, 2, \ldots$ such that

$$||g^{(k)}(u(t))|| \;\; \leq \;\; \alpha_k \tag{2.37}$$

Assumption 2.7 *Let $V(t, u) \in \mathbb{D}(T) \cap H^1$. Assume there exist a constant $\tilde{S}_k$ such that*

$$||Tg^{(k)}(u(t))|| \;\; \leq \;\; \tilde{S}_k u_0 \tag{2.38}$$

Assumption 2.8 *Let $u(.)$ be the solution of Equation (2.1). Then, there exists a constant $M_k, \; k = 0, 1, 2, \ldots$ such that*

$$||Tu^{(k)}|| \;\; \leq \;\; M_k u_0 \tag{2.39}$$

The proposed iterative splitting method is analyzed with the help of the $\varphi-$functions, namely the exponential integrators. A short overview for exponential integrators is given in Appendix B, see *Hochbruck and Ostermann*, (Hochbruck and Ostermann, 2010) and further reference therein for details.

Remark 2.2 *Observe that $[T, \varphi_k(sT)] = 0$, $k = 1, 2, 3, \ldots$ where $[T, e^{sT}] = 0$.*

Proof This proof follows the line of induction. Consider $k = 1$. Using the definition of the $\varphi - function$ and the identity $[T, e^{sT}] = 0$, we have

$$\begin{aligned} [T, \varphi_1(sT)] &= T\varphi_1(sT) - \varphi_1(sT)T \\ &= T\frac{e^{sT} - I}{sT} - \frac{e^{sT} - I}{sT}T \\ &= \frac{Te^{sT} - T}{sT} - \frac{e^{sT}T - T}{sT} \\ &= \frac{Te^{sT} - e^{sT}T}{sT} \\ &= \frac{[T, e^{sT}]}{sT} = 0. \end{aligned} \tag{2.40}$$

Assume that $[T, \varphi_k(sT)] = 0$. In order to show $[T, \varphi_{k+1}(sT)] = 0$, note that

$$\varphi_{k+1}(sT) \;\; = \;\; \frac{\varphi_k(sT) - \varphi_k(0)}{sT}, \quad \varphi_0(sT) = e^{sT} \tag{2.41}$$

holds. Thus

$$\begin{aligned}
[T, \varphi_{k+1}(sT)] &= T\varphi_{k+1}(sT) - \varphi_{k+1}(sT)T \\
&= T\frac{\varphi_k(sT) - \varphi_k(0)}{sT} - \frac{\varphi_k(sT) - \varphi_k(0)}{sT}T \\
&= \frac{T\varphi_k(sT) - T\varphi_k(0)}{sT} - \frac{\varphi_k(sT)T - \varphi_k(0)T}{sT} \\
&= \frac{T\varphi_k(sT) - T\varphi_k(0) - (\varphi_k(sT)T - \varphi_k(0)T)}{sT} \\
&= \frac{T\varphi_k(sT) - \varphi_k(sT)T - (T\varphi_k(0) - \varphi_k(0)T)}{sT} \\
&= \frac{[T, \varphi_k(sT)] - [T, \varphi_k(0)]}{sT}. \qquad (2.42)
\end{aligned}$$

Due to the assumption $[T, \varphi_k(sT)] = 0$ and the definition of $\varphi_k(0)$, we have

$$\begin{aligned}
[T, \varphi_{k+1}(sT)] &= \frac{[T, \varphi_k(sT)] - [T, \varphi_k(0)]}{sT} \\
&= 0. \qquad (2.43)
\end{aligned}$$

□

Under these assumptions and remark, the consistency results of the proposed iterative splitting method are achieved with the expected order.

Proposition 2.5 *The proposed iterative splitting is first order if we consider (2.8) with the error bound*

$$||u(h) - U(h)|| \leq \beta h^2. \qquad (2.44)$$

Here β only depends on α_1.

Proof Denote the local error by $e_j = U(t_j) - u(t_j)$, $j = 0, 1, 2, \ldots, n$. For simplicity we only consider the time interval $[0, h]$. The exact solution of Equation (2.1) is,

$$u(h) = e^{Th}u_0 + \int_0^h e^{T(h-s)}g(u(s))ds. \qquad (2.45)$$

To establish the local error, we employ the variation-of-constant formula. The first order numerical solution of (2.1) is

$$U(h) = e^{Th}u_0 + \int_0^h e^{T(h-s)}g(u_0)ds. \tag{2.46}$$

Subtracting (2.45) from (2.46) leads to

$$\begin{aligned} ||u(h) - U(h)|| &= ||\int_0^h e^{T(h-s)}g(u(s)) - g(u_0)ds||. \\ &\leq ||\int_0^h e^{T(h-s)}(\int_0^s \partial g(u(\rho))d\rho)ds||. \end{aligned} \tag{2.47}$$

By *Assumption* 2.5 and *Assumption* 2.6, we have

$$||u(h) - U(h)|| \leq \beta h^2.$$

□

Proposition 2.6 *Let Assumption 2.5 and Assumption 2.7 hold. Then, the proposed method has second order approximation if we consider Equation (2.17) with the error bound*

$$||u(h) - U(h)|| \leq Gh^3. \tag{2.48}$$

Here G depends on E_1, $\tilde{R}_2$ and $\tilde{\mathrm{S}}_1$ *where* E_1 *is the bound for* $||\Phi_2(h,s)||$ *and* $\tilde{R}_2$ *is the bound for* $\varphi_2(hT)$.

Proof By employing the variation-of-constant formula for obtaining the solutions of Equation (2.8) and Equation (2.9), we get

$$u_1(h) = e^{Th}u_0 + \int_0^h e^{T(h-s)}V(u_0)u_0 ds, \tag{2.49}$$

$$u_2(h) = \Phi_2(h,0)u_0 + \int_0^h \Phi_2(h,s)Tu_1(s)ds, \tag{2.50}$$

respectively. Here, $\Phi_2(h,0)$ is the numerical flow of Equation (2.9). We construct this flow by using Magnus expansion due to the nonlinearity of the operator.

On the other hand, the exact solution can be written as

$$u(h) \;=\; e^{Th}u_0 + \int_0^h e^{T(h-s)}V(u(s))u(s)ds, \tag{2.51}$$

or

$$u(h) \;=\; \Phi_2(h,0)u_0 + \int_0^h \Phi_2(h,s)Tu(s)ds. \tag{2.52}$$

Moreover,

$$u(h) \;=\; u_0 + \int_0^h F(u(s))ds, \tag{2.53}$$

where $F(u(s)) = Tu(s) + V(s,u(s))u(s)$.

By substituting Equation (2.49) into Equation (2.50), we have

$$\begin{aligned} u_2(h) &= \Phi_2(h,0)u_0 + \int_0^h \Phi_2(h,s)Te^{Th}u_0 \\ &+ \int_0^h \Phi_2(h,s)T(\int_0^s e^{T(s-\rho)}g(u_0)d\rho)ds. \end{aligned} \tag{2.54}$$

Further, substituting Equation (2.51) into Equation (2.52) leads to

$$\begin{aligned} u(h) &= \Phi_2(h,0)u_0 + \int_0^h \Phi_2(h,s)Te^{Th}u_0 \\ &+ \int_0^h \Phi_2(h,s)T(\int_0^s e^{T(s-\rho)}g(u(\rho))d\rho)ds. \end{aligned} \tag{2.55}$$

Since T is unbounded, to prove the consistency of the second order method, we restrict ourselves to the $\varphi - function$ representation. Then, Equation (2.54) and Equation (2.55) are represented by $\varphi - function$ as follows:

$$\begin{aligned} u_2(h) &= \Phi_2(h,0)u_0 + \int_0^h \Phi_2(h,s)Te^{Th}u_0 \\ &+ \int_0^h \Phi_2(h,s)T\varphi_1(sT)sg(u_0)ds, \end{aligned} \tag{2.56}$$

whereas the rearranged exact one is

$$
\begin{aligned}
u(h) &= \Phi_2(h,0)u_0 + \int_0^h \Phi_2(h,s)Te^{Th}u_0 \\
&+ \int_0^h \Phi_2(h,s)Ts\Sigma_{k=1}^{p}\varphi_k(sT)s^{k-1}g^{(k-1)}(u_0)ds \\
&+ \int_0^h \Phi_2(h,s)\int_0^\tau \frac{(\tau-\xi)^{p-1}}{(p-1)!}Tg^{(p)}(\xi)d\xi ds. \qquad (2.57)
\end{aligned}
$$

To estimate the error bound subtract (2.56) from (2.57). The remaining term is

$$
\begin{aligned}
u(h) - u_2(h) &= \int_0^h \Phi_2(h,s)Ts\Sigma_{k=2}^{p}\varphi_k(sT)s^{k-1}g^{(k-1)}(u_0)ds \\
&+ \int_0^h \Phi_2(h,s)\int_0^\tau \frac{(\tau-\xi)^{p-1}}{(p-1)!}Tg^{(p)}(\xi)d\xi ds. \qquad (2.58)
\end{aligned}
$$

From *Remark* 2.2, we have

$$
\begin{aligned}
u(h) - u_2(h) &= \int_0^h \Phi_2(h,s)s\,\Sigma_{k=2}^{p}\varphi_k(sT)s^{k-1}Tg^{(k-1)}(u_0)ds \\
&+ \int_0^h \Phi_2(h,s)\int_0^\tau \frac{(\tau-\xi)^{p-1}}{(p-1)!}Tg^{(p)}(\xi)d\xi ds, \qquad (2.59)
\end{aligned}
$$

taking norm on both sides, we have

$$
\begin{aligned}
\|u(h) - u_2(h)\| &= \|\int_0^h \Phi_2(h,s)\Sigma_{k=2}^{p}\varphi_k(sT)s^{k}Tg^{(k-1)}(u_0)ds \\
&+ \int_0^h \Phi_2(h,s)\int_0^\tau \frac{(\tau-\xi)^{p-1}}{(p-1)!}Tg^{(p)}(\xi)d\xi ds\|. \qquad (2.60)
\end{aligned}
$$

The triangle inequality implies

$$
\begin{aligned}
\|u(h) - u_2(h)\| &\leq \int_0^h \|\Phi_2(h,s)s^2\varphi_2(sT)Tg'(u_0)\|ds + \delta_3^p \\
&\leq E_1\int_0^h s^2\|\varphi_2(sT)Tg'(u_0)\|ds + O(h^4) \qquad (2.61)
\end{aligned}
$$

Since $V(t,u)$ is bounded in the above, E_1 denotes the bound for $||\Phi_2(h,s)||$. Moreover,

$$\delta_3^p = \int_0^h \Phi_2(h,s) \int_0^\tau \frac{(\tau-\xi)^{p-1}}{(p-1)!} T g^{(p)}(\xi) d\xi ds \tag{2.62}$$

where $||\delta_3^p||$ is bounded by $O(h^4)$, see (Hochbruck and Ostermann, 2010) for details.

Due to the *Assumption* 2.5 and using the definition of $\varphi - functions$ given in Equation (B.5) in Appendix B, we can deduce that

$$\begin{aligned} ||\varphi_k(hT)|| &\leq \int_0^1 ||\frac{\theta^{k-1}}{(k-1)!}|| d\theta. \\ ||\varphi_k(hT)|| &\leq \tilde{R}_k \end{aligned} \tag{2.63}$$

By using the Equation (2.63) and *Assumption* 2.7, Equation (2.61) becomes

$$||u(h) - u_2(h)|| \leq E_1 \tilde{R}_2 \tilde{S}_1 h^3 + O(h^4) \tag{2.64}$$

Here $\tilde{R}_2$ is the bound for $\varphi_2(hT)$, which proves that the second order proposed iterative splitting method is also consistent. □

Proposition 2.7 *The new first order iterative splitting scheme is stable on* $[0, t_{end}]$ *with the bound*

$$||U^n|| \leq K_1 ||u_0|| \tag{2.65}$$

where the constant K_1 *depends on* t_{end} *and* α_0.

Proof To prove the stability we start with rewriting the formulation of the method as in the following form

$$U^1 = U(h) = e^{Th} u_0 + G_1, \ U^0 = u_0, \tag{2.66}$$

where

$$G_1 = \int_0^h e^{T(h-s)} V(s, u_0) u_0 \, ds,$$

which is bounded by $||G_1|| \leq h\alpha_0 ||u_0||$. Rearranging Equation (2.66),

$$\begin{aligned} ||U^1|| &= ||e^{Th} U^0 + G_1|| \leq ||e^{Th} U^0|| + ||G_1|| \\ ||U^1|| &\leq ||u_0|| + h\alpha_0 ||u_0|| = ((1 + h\alpha_0))||u_0||. \end{aligned} \tag{2.67}$$

Recursively we get the stability polynomial for iterative scheme at first order,

$$||U^n|| \leq (1 + h\alpha_0)^n ||u_0|| \leq e^{t_{end}\alpha_0} ||u_0||, \tag{2.68}$$

which concludes the proof. □

Proposition 2.8 *The new second order iterative splitting scheme is stable if* $||\Phi(t, s)|| \leq 1$ *on* $[0, t_{end}]$ *with the bound*

$$||U^n|| \leq K_2 u_0$$

where K_2 *depends on* E_1, t_{end}, $\tilde{R}_1$, $\tilde{S}_0$ *and* M_0. *The constants* E_1 *and* R_1 *are the bounds for* $\Phi_2(h, s)$ *and* φ_1, *respectively.*

Proof The proof follows the previous one. To obtain the stability bound for the second order proposed iterative splitting method, Equation (2.9) is rewritten as

$$U^1 = U(h) = \Phi_2(t + h, t) U^0 + \int_0^h \Phi_2(h, s) T u_1, \tag{2.69}$$

where $U^0 = u(0)$. Substituting u_1 given in Equation (2.49) into Equation (2.69) and using the linearity of T leads to

$$\begin{aligned} U^1 = U(h) &= \Phi_2(h,0)u_0 + \int_0^h \Phi_2(h,s)Te^{Th}u_0 \\ &+ \int_0^h \Phi_2(h,s)T(\int_0^s e^{T(s-\rho)}g(u_0)d\rho)ds. \end{aligned} \tag{2.70}$$

Rearranging the above equation using the *φ−functions*, we have

$$\begin{aligned} U^1 = U(h) &= \Phi_2(h,0)u_0 + \int_0^h \Phi_2(h,s)Te^{Th}u_0 \\ &+ \int_0^h \Phi_2(h,s)Ts\varphi_1(sT)g(u_0)ds. \end{aligned} \tag{2.71}$$

Taking norm and employing *Assumption* 2.7 and *Assumption* 2.8 leads to

$$\begin{aligned} ||U^1|| &\leq ||\Phi_2(h,0)u_0|| + ||\int_0^h \Phi_2(h,s)M_0|| \\ &+ ||\int_0^h \Phi_2(h,s)Ts\varphi_1(sT)g(u_0)ds||. \end{aligned}$$

Since $V(t,u)$ is bounded, we get

$$||U||^1 \leq E_1u_0 + \int_0^h E_1M_0u_0ds + \int_0^h E_1s\varphi_1(sT)||Tg(u_0)||ds,$$

where E_1 is the bound for $\Phi_2(h,s), \quad 0 \leq s \leq h$. Finally, by the *Assumption* 2.7, we have

$$\begin{aligned} ||U||^1 &\leq E_1u_0 + h\,E_1\,M_0u_0 + \frac{h^2}{2}\,E_1\,\tilde{R}_1\,\tilde{S}_0U_0 \\ &\leq E_1(1 + hM_0 + h^2\frac{\tilde{R}_1\,\tilde{S}_0}{2})U_0, \end{aligned}$$

where $\tilde{R}_1$ is the bound for $\varphi_1(sT)$ as shown in Equation (2.63). Recursively,

$$||U||^n \leq E_1^n e^{t_{end}\gamma}U_0$$

where γ depends on $\tilde{R}_1$, $\tilde{S}_0$ and M_0. As a consequence, the second order scheme is stable if $||\Phi_2|| \leq 1$. □

2.2.3. Convergence Results

Next we investigate convergence of PISM. To that end, we utilize the "Lady Windermere's fan" argument. As mentioned before u_n and U_n represents the exact and numerical solutions at $t = t_n$, respectively.

In general, the local error bound is defined as

$$d_n = D(h_{n-1})u(t_{n-1}) \;=\; (\Psi(h_{n-1}) - E(h_{n-1}))u(t_{n-1}) \tag{2.72}$$

with $t_0 < t_1 < t_2 < \ldots < t_N = t_{end}$. Here $h_{n-1} = t_n - t_{n-1},\;\; 1 \leq n \leq N$, $\Psi(h_{n-1})$ denotes numerical flow and $E(h_{n-1})$ denotes the exact solution.

Theorem 2.1 ***Lady Windermere's Fan.*** *To establish the global and local error, we employ the telescopic identity,*

$$U_N - u_N \;=\; \prod_{j=0}^{N-1} \Psi(h_j)(U_0 - u_0) + \sum_{n=1}^{N} \prod_{j=n}^{N-1} \Psi(h_j) d_n \tag{2.73}$$

Proof The validity of relation Equation (2.73) is verified by a short calculation as follows

$$\begin{aligned}
& \prod_{j=0}^{N-1} \Psi(h_j)(U_0 - u_0) + \sum_{n=1}^{N} \prod_{j=n}^{N-1} \Psi(h_j) d_n \\
= \;& \prod_{j=0}^{N-1} \Psi(h_j)(U_0 - u_0) + \sum_{n=1}^{N} \prod_{j=n}^{N-1} \Psi(h_j)(\Psi(h_{n-1}) - E(h_{n-1}))u_{n-1} \\
= \;& \prod_{j=0}^{N-1} \Psi(h_j) U_0 - \prod_{j=0}^{N-1} \Psi(h_j) u_0 \\
+ \;& \sum_{n=1}^{N} \prod_{j=n-1}^{N-1} \Psi(h_j) u_{n-1} - \sum_{n=1}^{N} \prod_{j=n}^{N-1} \Psi(h_j) u_n \\
= \;& U_N - \prod_{j=0}^{N-1} \Psi(h_j) u_0 + \sum_{n=0}^{N-1} \prod_{j=n}^{N-1} \Psi(h_j) u_n - \sum_{n=1}^{N} \prod_{j=n}^{N-1} \Psi(h_j) u_n
\end{aligned}$$

$$
\begin{aligned}
&= U_N - \prod_{j=0}^{N-1}\Psi(h_j)u_0 + \prod_{j=0}^{N-1}\Psi(h_j)u_0 - u_N \\
&= U_N - u_N.
\end{aligned}
$$

□

Rearranging the Equation (2.73) leads to

$$
\begin{aligned}
U_N - u_N &= \sum_{j=0}^{N_1}\Psi^{(N-j-1)h}(\Psi^h u(t_j) - u(t_{j+1})) \\
&= \sum_{j=0}^{N_1}\Psi^{(N-j-1)h}(\Psi^h - E^h))u(t_j), \qquad (2.74)
\end{aligned}
$$

where $t^j = jh$ and h is uniform mesh. To achieve the convergence result, we use appropriate norm, $||.||$ and we have

$$
\begin{aligned}
||U_N - u(t_N)|| &= ||\sum_{j=0}^{N_1}\Psi^{(N-j-1)h}(\Psi^h - E^h))u(t_j)|| \\
&\leq \sum_{j=0}^{N_1}||\Psi^{(N-j-1)h}||\,||(\Psi^h - E^h))||\,||u(t_j)||. \qquad (2.75)
\end{aligned}
$$

We note that $||\Psi^{(N-j-1)h}||$ and $||(\Psi^h - E^h))||$ correspond to the stability and the consistency results of the proposed iterative splitting method, respectively. The well-posedness of the initial value problem given in Equation (2.1) implies the boundedness of $||u(t_j)||$ for all t_j, $j = 1, 2, 3, \ldots, N$.

Therefore, the following assertion is an immediate consequence of the fact that the consistency and the stability results of the proposed iterative splitting method for both bounded and unbounded case.

Theorem 2.2 ***(Bounded Case)*** *Suppose assumptions 2.1-2.4 hold. The proposed iterative splitting method is convergent if* $\Phi_2(t, s) \leq 1$. *Then, the second order global error*

$$
||U^n(h) - u^n(h)|| \leq F\,h^2.
$$

The constant F *in general depends on* t_{end}, *but not on* h.

Proof The proof is given when T and $V(t, u)$ are bounded operators. Therefore, we need *Assumptions* 2.1-2.4. With the help of *Proposition* 2.2 and *Proposition* 2.4, we have

$$\begin{aligned}
||U_N - u(t_N)|| &= ||\sum_{j=0}^{N_1} \Psi^{(N-j-1)h}(\Psi^h - E^h))u(t_j)|| \\
&\leq \sum_{j=0}^{N_1} ||\Psi^{(N-j-1)h}||||(\Psi^h - E^h))||||u(t_j)|| \\
&\leq \sum_{j=0}^{N_1} \zeta||u_0||\gamma h^3||u(t_j)|| \\
&\leq NF_1h^3 \\
&\leq t_{end}F_1h^2 = Fh^2 \qquad (2.76)
\end{aligned}$$

where ζ and γ are defined in Equation (2.32) and Equation (2.25), respectively. Thus, convergence is achieved. □

Theorem 2.3 ***(Unbounded Case)*** *Suppose assumptions 2.5-2.8 hold. The proposed iterative splitting method is convergent with the second order global error*

$$||U^n(h) - u^n(h)|| \leq \hat{F}h^2.$$

The constant $\hat{F}$ is independent from h.

Proof The proof follows is similar to the preceding one. We consider the case when T is unbounded but $V(t, u)$ is bounded operator. We assume that assumptions 2.5-2.8 hold. Using *Proposition* 2.6 and *Proposition* 2.8 we have

$$\begin{aligned}
||U_N - u(t_N)|| &= ||\sum_{j=0}^{N_1} \Psi^{(N-j-1)h}(\Psi^h - E^h))u(t_j)|| \\
&\leq \sum_{j=0}^{N_1} ||\Psi^{(N-j-1)h}||||(\Psi^h - E^h))||||u(t_j)|| \\
&\leq \sum_{j=0}^{N_1} K_2||u_0||Gh^3||u(t_j)|| \\
&\leq t_{end}\hat{F}_1h^2 = \hat{F}h^2. \qquad (2.77)
\end{aligned}$$

Here, $\hat{F}$ depends on the G given in Equation (2.48) and K_2 can be found in the proof of *Proposition* 2.8. Thus, the proposed iterative splitting method is convergent for not only the bounded case but also the unbounded case. $\square$

CHAPTER 3

NUMERICAL TESTS AND SIMULATIONS: THE PROPOSED ITERATIVE SPLITTING METHOD

Having established the theoretical convergence rates for the proposed method in Chapter 2, we now provide supporting numerical tests and simulations. We choose three oscillation problems. Numerical solutions are derived with MATLAB. The codes are given in Appendix D.

In the numerical examples, we consider the efficiency of the methods in terms of the errors, the convergence rates for Δt and the CPU runtimes. In order to get the errors we use the L_1, L_2 and L_∞ norms

$$\begin{aligned}
||.||_{L_1} &= \sum_{n=0}^{n=N} |u(nh) - U(nh)|, \\
||.||_{L_2} &= (\sum_{n=0}^{n=N} |u(nh) - U(nh)|^2)^{1/2} \\
||.||_{L_\infty} &= \max_{0\leq n\leq N} |u(nh) - U(nh)|.
\end{aligned}$$

Throughout this section, the proposed iterative splitting method is labeled as PISM.

3.1. Duffing Equation

The one dimensional unforced Duffing oscillator is

$$q'' + \alpha q + \varepsilon q^3 = 0. \tag{3.1}$$

Here α controls the size of the stiffness and ε controls the amount of nonlinearity in the restoring force. First, we redefine the variables as $q(t) = q_1(t)$ and $\dot{q}(t) = q_2(t)$, and $u(t) =$

$(q_1(t), q_2(t))^T$. Thus, it yields

$$\begin{aligned}
\begin{pmatrix} \dot{q}_1 \\ \dot{q}_2 \end{pmatrix} &= \begin{pmatrix} 0 & 1 \\ -1-\varepsilon q_1^2 & 0 \end{pmatrix}\begin{pmatrix} q_1 \\ q_2 \end{pmatrix} \\
&= \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix}\begin{pmatrix} q_1 \\ q_2 \end{pmatrix} + \begin{pmatrix} 0 & 0 \\ -\varepsilon q_1^2 & 0 \end{pmatrix}\begin{pmatrix} q_1 \\ q_2 \end{pmatrix}. \qquad (3.2)
\end{aligned}$$

To compare the size of errors for different splitting methods described in Chapter 2, we solve Equation (3.1) for $t_{end} = 10$ with $\varepsilon = 10^{-4}$ for various values Δt. We take the initial conditions as $q_1(0) = 1$ and $q_2(0) = 0$, and the analytic solution is given in (Bender and Orszag, 1999) as

$$q(t) = \cos t + \varepsilon[\frac{1}{32}\cos 3t - \frac{1}{32}\cos t - \frac{3}{8}t \sin t], \quad \varepsilon \to 0^+. \qquad (3.3)$$

In Table 3.1, the convergence orders are presented numerically in the discrete L_∞-norm. The obtained numerical results are in a agreement with theoretical results given in Section 2.2. Table 3.1 reveals that the new second order iterative splitting scheme is more efficient than not only Strang splitting method (SMS) but also symmetrically weighted splitting (SWS).

	PISM		SMS		SWS	
Δt	Error	Order	Error	Order	Error	Order
1/8	3.4097e-006	-	7.2936e-006	-	7.2704e-006	-
1/16	8.4748e-007	2.0336	3.6175e-006	1.0116	3.6150e-006	1.0080
1/32	2.08912e-007	2.0203	1.7779e-006	1.0248	1.7774e-006	1.0242

Table 3.1. Comparison of errors for several h on [0, 10] interval with various methods where $\varepsilon = 10^{-4}$. The expected order is 2.

Although, there is a reduction of order of the convergence of the SMS, PISM achieves second order accuracy.

From Table 3.2, we observe that PISM is more accurate than the other splitting methods.

Method	ε	Δt	L_∞	L_2	L_1
PISM	0.01	2^{-3}	3.5838e-005	5.8357e-005	1.3204e-004
SMS	0.01	2^{-3}	5.5170e-005	7.5197e-005	1.4932e-004
SWS	0.01	2^{-3}	5.7048e-005	7.9755e-005	1.6290e-004
PISM	0.01	2^{-4}	4.7238e-006	1.3086-005	4.4547e-005
SMS	0.01	2^{-4}	2.6018e-005	4.7427e-005	1.3360e-004
SWS	0.01	2^{-4}	2.6453e-005	4.9006e-005	1.3999e-004
PISM	0.01	2^{-5}	4.2273e-006	8.0718e-006	2.6895e-005
SMS	0.01	2^{-5}	9.9518e-006	2.5903e-005	1.0542e-004
SWS	0.01	2^{-5}	1.0057e-005	2.6468e-005	1.0853e-004
PISM	0.005	2^{-3}	1.9641e-005	3.1267e-005	7.0055e-005
SMS	0.005	2^{-3}	2.9255e-005	3.9719e-005	7.8598e-005
SWS	0.005	2^{-3}	3.0161e-005	4.1933e-005	8.5252e-005
PISM	0.005	2^{-4}	3.8650e-006	9.1589e-006	2.9788e-005
SMS	0.005	2^{-4}	1.4741e-005	2.6540e-005	7.4137e-005
SWS	0.005	2^{-4}	1.4950e-005	2.7302e-005	7.7258e-005
PISM	0.005	2^{-5}	3.3566e-007	1.2292e-006	5.8964e-006
SMS	0.005	2^{-5}	6.7342e-006	1.6797e-005	6.6791e-005
SWS	0.005	2^{-5}	6.7847e-006	1.7066e-005	6.8307e-005

Table 3.2. Numerical errors are established for different Δt and different ε on $[0, 1]$.

Moreover, Equation (3.1) is a Hamiltonian system. The Hamiltonian of the Duffing equation is expressed as follows:

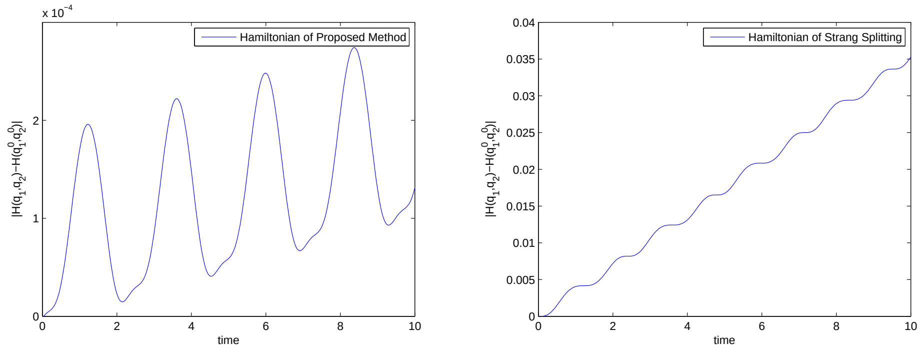
$$\begin{aligned} \dot{q}_1 &= -H_{q_2} \\ \dot{q}_2 &= H_{q_1} \end{aligned} \tag{3.4}$$

where $H_{q_2} = -q_2$ and $H_{q_1} = -q_1 - \varepsilon {q_1}^3$. By integrating Equation (3.4), one can obtain

$$H(q_1, q_2) = -\frac{{q_1}^2}{2} - \frac{{q_2}^2}{2} - \varepsilon\frac{{q_1}^4}{4}, \tag{3.5}$$

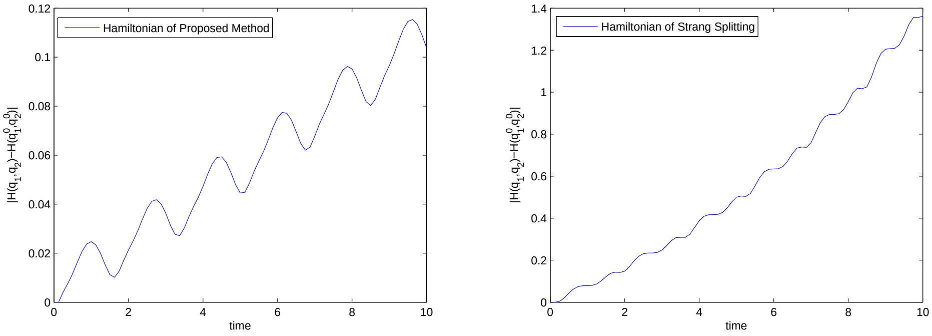
where $H(q_1(0), q_2(0)) = -\frac{1}{2} - \frac{\varepsilon}{4}$.

The conservation of the Hamiltonian of Equation (3.1) is exhibited the in Figure 3.1 and Figure 3.2 for $\varepsilon = 1$ and $\varepsilon = 3$ on $[0, 10]$, respectively.



(a) PISM (b) SMS

Figure 3.1. The Hamiltonian of Duffing Equation (3.1) for $\varepsilon = 1$ on $[0, 10]$.



(a) PISM (b) SMS

Figure 3.2. The Hamiltonian of Duffing Equation (3.1) for $\varepsilon = 3$ on $[0, 10]$.

To see the Hamiltonian conservation examine $|H(q_1, q_2) - H(q_1^0, q_2^0)|$ where $q_i^0 = q_i(0),\ \ i = 1, 2$. From Figure 3.1 and Figure 3.2 suggest that the difference is smaller for PISM than Strang splitting method. Thus, PISM is more accurate than Strang splitting with regards to energy preservation.

3.2. Van-der Pol Equation

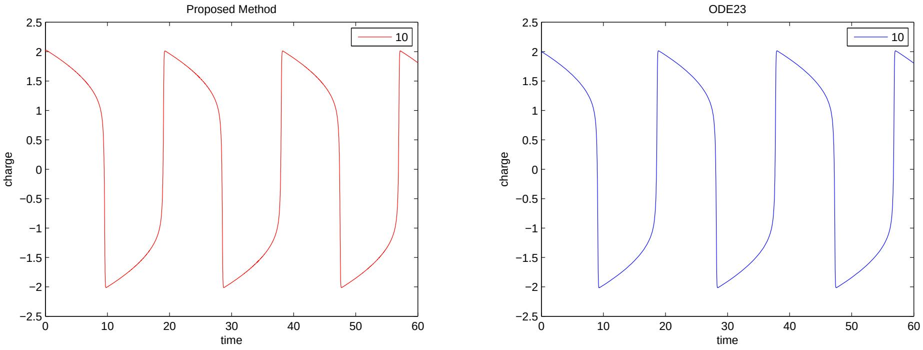
Our second example is the Van-der Pol equation which is one of the widely studied systems in nonlinear dynamics. Here, we concentrate on the second order autonomous Van-der Pol equation

$$\ddot{x} + \mu(x^2 - 1)\dot{x} + x \;=\; 0, \quad x(0) = 2, \quad \dot{x}(0) = 0. \tag{3.6}$$

The parameter μ is a positive scalar denoting the nonlinearity and the strength of the damping. By redefining Equation (3.6) with $\dot{x}_1 = x_2$, $\dot{x}_2 = -\mu({x_1}^2 - 1)x_2 - x_1$ and by splitting the operators as follows:

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & -\mu({x_1}^2 - 1) \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}. \tag{3.7}$$

To illustrate the effects of the nonlinearity, in our implementation, we set $\mu = 10$. Then PISM is in agreement with the solution ODE23s-code in MATLAB.



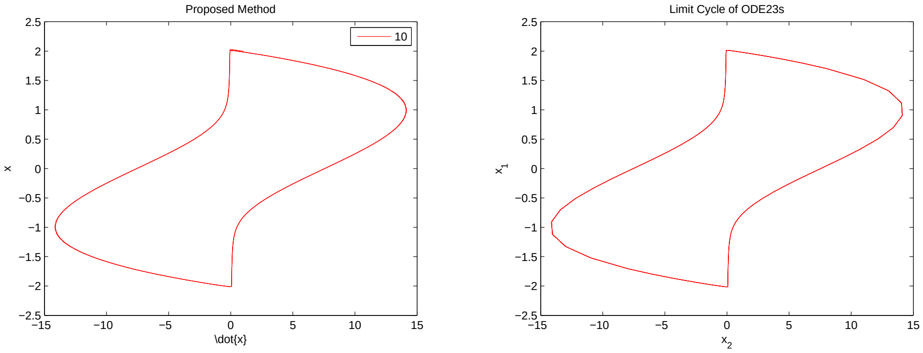
(a) PISM (b) ODE23s in MATLAB

Figure 3.3. The solution of Van-der Pol Equation (3.6) using PISM on the left, ODE23s on the right.

We deal with the phase diagram of the solution for different methods in Figure 3.4. Although PISM is explicit, it preserves the limit cycle of Equation (3.6).

Equation (3.6) is also a Hamiltonian system with Hamiltonian

$$\begin{aligned} \dot{x}_1 &= -H_{x_2} \\ \dot{x}_2 &= H_{x_1} \end{aligned} \tag{3.8}$$



(a) PISM (b) ODE23s in MATLAB

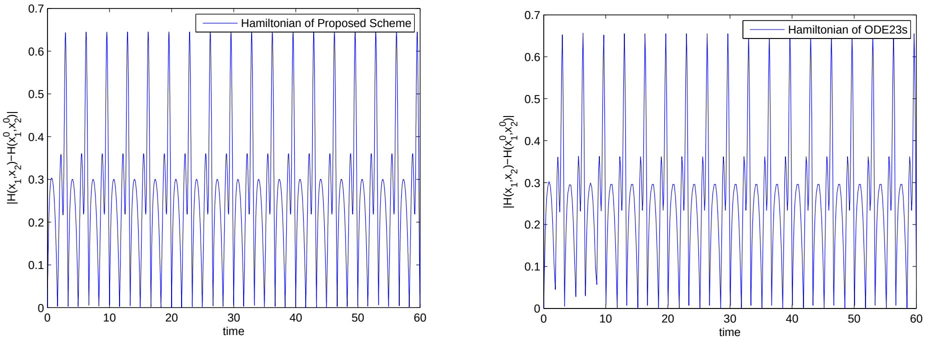
Figure 3.4. Trajectories of Van-der Pol Equation (3.6) for $\mu = 10$ PISM on the left and ODE23s on the right.

where $H_{x_2} = -x_2$ and $H_{x_1} = -\mu({x_1}^2 - 1)x_2 - x_1$. By integrating Equation (3.8), we have

$$H(x_1, x_2) \quad = \quad -\frac{{x_1}^2}{2} - \frac{{x_2}^2}{2} - \mu(\frac{{x_1}^3}{3} - x_1)x_2, \tag{3.9}$$

where $H(x_1(0), x_2(0)) = -2$.

To see the conservation of the Hamiltonian considering PISM and ODE23s on Equation (3.6), the Figure 3.5 is presented for $\mu = 1$ on $[0, 60]$.



(a) PISM (b) ODE23s in MATLAB

Figure 3.5. The Hamiltonian of Van-der Pol Equation (3.6) for $\mu = 1$ on $[0, 60]$.

Note that the ODE23s code is an implicit solver in MATLAB. Although PISM is an explicit method, it achieves the same results with the ODE23s both for the solutions and the Hamiltonian.

3.3. Nonlinear Schrödinger Equation

After showing the efficiency of PISM on ordinary differential equations, we now look at the effectiveness of PISM on partial differential equations. For that purpose, we focus on the cubic nonlinear Schrödinger (NLS) equation

$$\begin{aligned} i\hbar\partial_t\psi &= \hat{H}\psi(x,t) && (3.10)\\ i\hbar\partial_t\psi &= \beta\partial_x^2\psi + (G(x) + \alpha|\psi|^2)\psi && (3.11)\end{aligned}$$

where $\psi(x,t)$ denotes the probability amplitude of the particle to be found at position x at time t, and $\hbar$ is the Planck constant. Here, ∂_x^2 denotes the spatial derivative. To approximate it, we use central difference. We consider the Dirichlet boundary conditions

$$\psi(x_L,t) = \psi(x_R,t) = 0, \quad t\in[0,t_{end}], \tag{3.12}$$

with the initial condition

$$\psi(x,0) = \psi_0, \quad x\in(x_L,x_R). \tag{3.13}$$

To split the equation, we first change the variable ψ to $\eta + i\xi$ in Equation (3.11). Then

$$\begin{aligned} i\hbar(\eta_t + i\xi_t) &= \beta(\eta_{xx} + i\xi_{xx}) + (G(x) + \alpha(|\eta|^2 + |\xi|^2))(\eta + i\xi)\\ i\eta_t - \xi_t &= \beta\eta_{xx} + i\beta\xi_{xx} + (G(x) + \alpha(|\eta|^2 + |\xi|^2))\eta\\ &+ i(G(x) + \alpha(|\eta|^2 + |\xi|^2))\xi. && (3.14)\end{aligned}$$

Rearranging Equation (3.14), we obtain

$$\begin{aligned}\hbar\eta_t &= \beta\xi_{xx} + (G(x) + \alpha(|\eta|^2 + |\xi|^2))\xi \\ \hbar\xi_t &= -\beta\eta_{xx} - (G(x) + \alpha(|\eta|^2 + |\xi|^2))\eta\end{aligned}$$

which corresponds the system of equations:

$$\hbar\begin{bmatrix}\dot{\eta} \\ \dot{\xi}\end{bmatrix} = \begin{bmatrix}\mathbf{0} & \beta\partial_x{}^2 \\ -\beta\partial_x{}^2 & \mathbf{0}\end{bmatrix}\begin{bmatrix}\eta \\ \xi\end{bmatrix} + \begin{bmatrix}\mathbf{0} & (G(x) + \alpha(|\eta|^2 + |\xi|^2)) \\ -(G(x) + \alpha(|\eta|^2 + |\xi|^2)) & \mathbf{0}\end{bmatrix}\begin{bmatrix}\eta \\ \xi\end{bmatrix} \tag{3.15}$$

We split the system as $T + V(x, \psi)$ where

$$T = \begin{bmatrix}\mathbf{0} & \beta\partial_x{}^2 \\ -\beta\partial_x{}^2 & \mathbf{0}\end{bmatrix}, \quad V(x,\psi) = \begin{bmatrix}\mathbf{0} & (G(x) + \alpha(|\eta|^2 + |\xi|^2)) \\ -(G(x) + \alpha(|\eta|^2 + |\xi|^2)) & \mathbf{0}\end{bmatrix}$$

We note that **0** denotes the zero matrix whose size equals to the size of $\partial_x{}^2$ and that the matrices in Equation (3.15) are skew-symmetric. Therefore, the eigenvalues are purely imaginary, which guarantees that the solutions are bounded.

3.3.1. Equation in the form: $i\hbar\partial_t\Psi = \beta\partial_x^2\Psi + \alpha|\Psi|^2\Psi$

We start with the NLS equation given in Equation (3.11) without external force, namely $G(x) = 0$. Inspired by Polyanin & Zaitsev, (Polyanin and Zaitsev, 2003), we take $\alpha = \beta = -1$ and $\hbar = 1$, to get

$$\begin{aligned} i\frac{\partial\psi}{\partial t} + \frac{\partial^2\psi}{\partial x^2} + |\psi|^2\psi &= 0, \qquad (3.16) \\ \psi(x_L, t) = \psi(x_R, t) &= 0, \end{aligned}$$

where the analytic solution for the above equation is

$$\psi(x,t) = A\sqrt{2}sech(Ax - 2ABxt + C_2)e^{i(Bx+(A^2-B^2)t+C_1)}. \tag{3.17}$$

Here A, B, C_2 and C_1 are arbitrary real constants, see (Polyanin and Zaitsev, 2003) and references therein.

The NLS equation above conserves many densities. In this study, we only focus on the mass and energy (or Hamiltonian) conservation. For the purpose of comparative analysis, we consider the study by Aydın and Karasözen, (Aydın and Karasözen, 2011), for the case of $\nu = 0$, $\alpha_1 = 1$, $\beta = \gamma = \Gamma = 0$ in their work. The mass and the energy conservations are as follows:

$$M(t) := \int_{x_L}^{x_R} |\psi|^2 dx := M(0), \tag{3.18}$$

$$E(t) := \int_{x_L}^{x_R} \left(-|\psi_x|^2 + \frac{|\psi|^4}{2}\right) dx := E(0). \tag{3.19}$$

In discrete space, mass and energy conservations are expressed as follows:

$$AE = \Delta x \sum_{k=1}^{N_x} |M_k^n - M_k^0|. \tag{3.20}$$

$$GE = \Delta x \sum_{k=1}^{N_x} (E_k^n - E_k^0). \tag{3.21}$$

Here, E^0 and M^0 denote the initial energy and mass, respectively. Moreover, AE and GE denote the absolute error and the global energy error, respectively. Then, from Equations (5.5) and (5.6), we have

$$E_i^n = -\frac{1}{4}((\eta_i^n)^2 + (\xi_i^n)^2)^2 + \frac{1}{2}((a_i^n)^2 + (b_i^n)^2) \tag{3.22}$$

$$M_i^n = (\eta_i^n)^2 + (\xi_i^n)^2, \tag{3.23}$$

where $a = \eta_x$, $b = \xi_x$ and E_i^n is the energy at $t = n\Delta t$. To check the validity of the solutions, we compute the absolute errors using Equation (3.20).

For the numerical results, we consider two different initial conditions:

Case 1 The initial condition is $\psi(x, 0) = \frac{2e^{ix}}{\cosh(\sqrt{2}x)}$, then the analytic solution is

$$\psi(x, t) = \frac{\sqrt{2}e^{i(x+t)}}{\cosh(\sqrt{2}x - 2\sqrt{2}t)}. \tag{3.24}$$

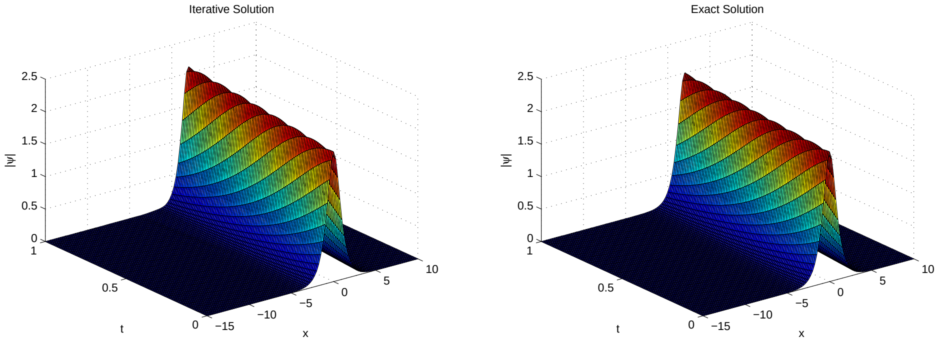
where $A = \sqrt{2}$, $B = 1$, and $C_2 = C_1 = 0$.

Case 2 The initial condition is $\psi(x,0) = \sqrt{2}sech(x+10)e^{ix/4}$, then the analytic solution is

$$\psi(x,t) = \sqrt{2}sech(x - \frac{t}{2} + 10)e^{i(\frac{x}{4}+\frac{15}{16}t)}, \tag{3.25}$$

where $A = 1$, $B = 1/4$, $C_2 = 10$, and $C_1 = 0$.

Figure 3.6 and Figure 3.7 present the numerical results of **Case 1** on the space interval $[-15, 10]$ divided into 100 uniform grid points with $\Delta x = 0.25$. We integrate the system using PISM with the time step size $\Delta t = 0.01$ up to final time $t = 1$. We obtain the solutions which are in a perfect agreement with the analytic one, see Figure 3.6.



(a) Numerical Solution (b) Exact Solution

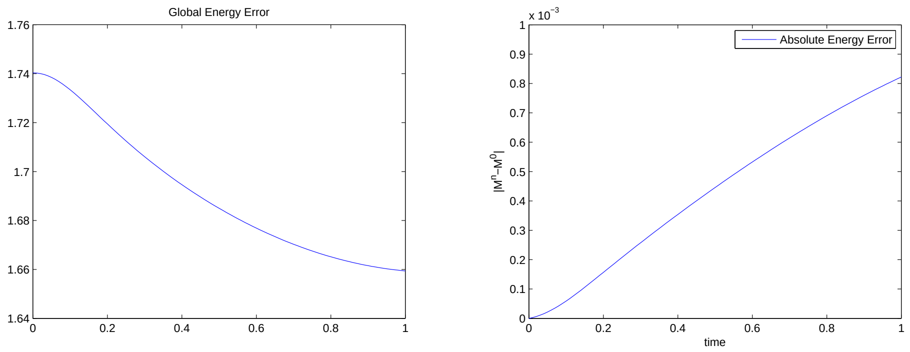
Figure 3.6. On the left PISM, exact solution on the right.

The subsequent Figure 3.7 illustrates conservation of the energy and the mass. Furthermore, the conserved density from the analytic solution is

$$\int_{-15}^{10} |\psi|^2 dx = 5.6569, \quad \int_{-15}^{10} (-|\psi_x|^2 + \frac{|\psi|^4}{2})dx = 3.1884$$

while from the numerical solution at $t = 1.0$ we have

$$\int_{-15}^{10} |\psi|^2 dx = 5.6560, \quad \int_{-15}^{10} (-|\psi_x|^2 + \frac{|\psi|^4}{2})dx = 3.2704.$$



(a) Global Energy Error (b) Absolute Error

Figure 3.7. Plots of the global energy error vs time and the absolute error vs. time are presented for **Case 1**.

The absolute error with different norms and various final time is presented in Table 3.3 where $N_x = N_t = 100$.

t_{end}	Δt	L_∞	L_2	L_1
0.3	0.003	1.6650e-005	8.9497e-005	7.2631e-004
0.5	0.005	8.7306e-005	4.9791e-004	0.0042
0.8	0.008	3.9789e-004	0.0024	0.0203
1	0.01	8.2197e-004	0.0050	0.0432

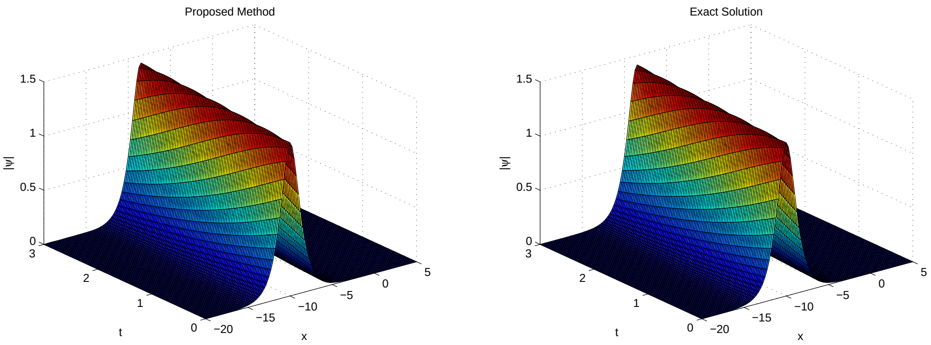
Table 3.3. Estimated errors using L_∞, L_2 and L_1 norm for conservation of the mass with $N_x = N_t = 100$ where $\Delta x = 0.25$.

For **Case 2**, we study the space interval $[-20, 5]$ divided into 100 uniform grid points with $\Delta x = 0.25$. We integrate the system using PISM with time-step size $\Delta t = 0.03$ up to a final time $t = 3$. Although PISM is an explicit scheme, in all solutions it preserves the qualitative properties.

Moreover, Figure 3.9 presents the accuracy of PISM with respect to the global energy error. In addition to this, due to the solitary wave, it can be seen the solutions of Equation (3.11) with the initial condition given in **Case 2** at some fixed time on the right.

Additionally, the conserved density from the analytic solution is

$$\int_{-20}^{5} |\psi|^2 dx = 4.0000, \quad \int_{-20}^{5} (-|\psi_x|^2 + \frac{|\psi|^4}{2}) dx = 1.1749,$$



(a) PISM (b) Exact Solution

Figure 3.8. The numerical solutions are obtained on $x \in [-20, 5]$ and $t \in [0, 3]$ where $N_x = N_t = 100$.

while we have from the numerical solution at $t = 3.0$.

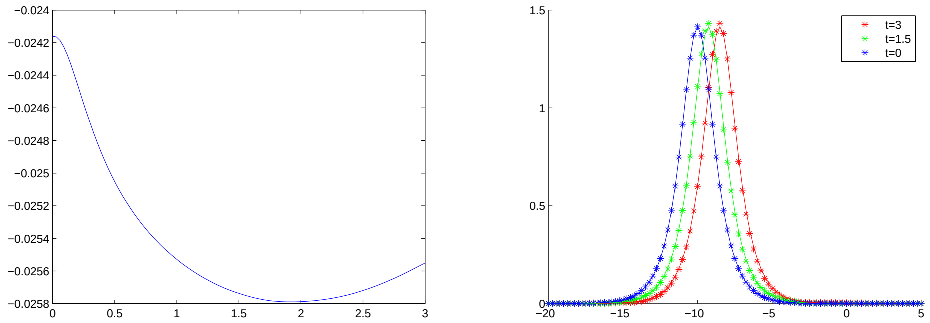
$$\int_{-20}^{5} |\psi|^2 dx = 3.9991, \quad \int_{-20}^{5} \left(-|\psi_x|^2 + \frac{|\psi|^4}{2} \right) dx = 1.2167,$$

where $N_x = N_t = 100$ in our calculations.

The absolute errors computed using different norms are in Table 3.4. The elapsed time are in Table 3.5, from which we deduce that the constructed method is more efficient than the Strang splitting method with regards to CPU runtimes.

time	L_∞	L_2	L_1
0.5	2.0479e-006	1.1582e-005	9.6575e-005
1	1.9412e-005	1.1707e-004	0.0010
2	2.1047e-004	0.0013	0.0113
3	9.2325e-004	0.0056	0.0494

Table 3.4. Estimated errors using L_∞, L_2 and L_1 norm for conservation of the mass with $N_x = N_t = 100$ where $\Delta x = 0.25$.



(a) Global Energy Error (b) The Solutions

Figure 3.9. Global Energy Error is on the left and the comparisons of numerical solution and exact solution for fixed time on the right for **Case 2**.

N_x	N_t	Strang Splitting	PISM
50	100	1.5156	1.1250
100	100	7.2031	5.7344
100	150	9.6875	6.5313
150	150	29.7344	26.3125

Table 3.5. The CPU runtimes are measured in seconds for different N_t and N_x on $x \in [-20, 5]$ and $t \in [0, 3]$.

3.3.2. Equation in the form: $i\hbar\partial_t\Psi = \beta\partial_x^2\Psi + (G(x) + \alpha|\Psi|^2)\Psi$

We turn our attention to the cubic nonlinear Schrödinger Equation (3.11) with real-valued external force. We consider $\hbar = 1, \beta = -\frac{1}{2}$ and $G(x) = \frac{1}{1+\sin^2 x}$. That is,

$$\text{ß}\partial_t \psi = -\frac{1}{2}\partial_x{}^2\psi + (\frac{1}{1+\sin^2 x} + \alpha * |\psi|^2)\psi \tag{3.26}$$

where $\alpha = 30$. We set the initial condition as $\psi(x, 0) = \gamma e^{\sin 2x}$.

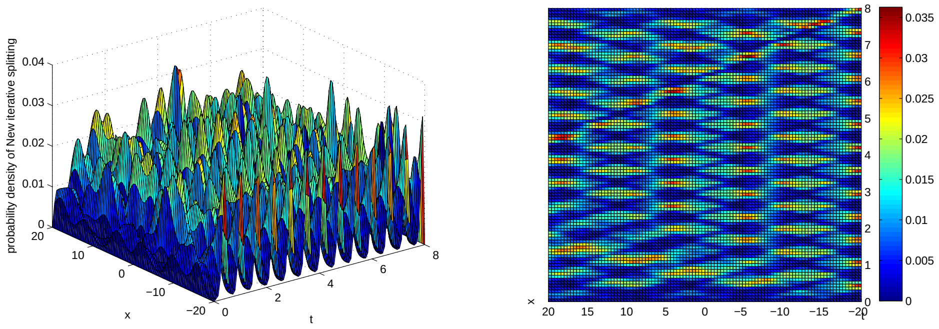
Physically, the probability density of any particle in Equation (3.26) is

$$\int_{x_L}^{x_R} |\psi|^2 dx \leq 1.$$

To see the effects of PISM on the numerical solutions, we need to start with $||\psi(x, 0)|| \leq 1$. We set γ as $\dfrac{1}{||\psi_0||}$ since

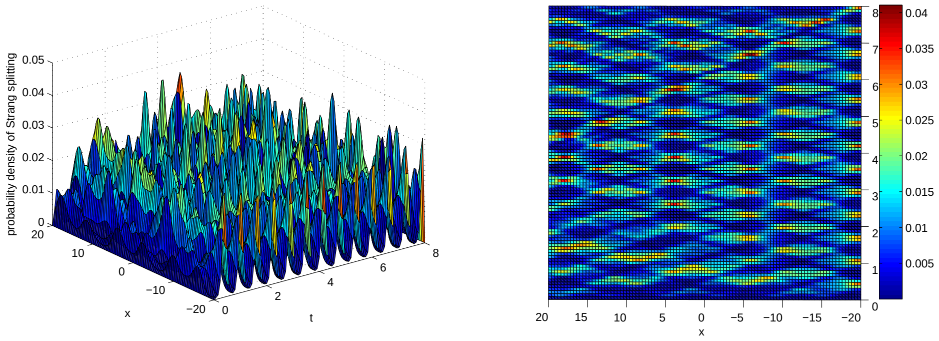
$$\int_{x_L}^{x_R} |\psi_0|^2 dx = 1.$$

To solve Equation (3.26), we follow the splitting process given in Equation (3.15). The numerical solutions and the contour plots are demonstrated for PISM and SMS in Figure 3.10 and Figure 3.11, respectively.



(a) Numerical Solution of Equation (3.26) (b) Contour Plot

Figure 3.10. Probability density of the particle in Equation (3.26) for PISM.



(a) Numerical Solution of Equation (3.26) (b) Contour Plot

Figure 3.11. Probability density of the particle in Equation (3.26) for SMS.

For the exhibited figures, we suppose that the system is defined in the interval $x \in [-20, 20]$, which is split into $N_x = 100$ parts. We integrate the system using PISM with time-step size $\Delta t = 0.08$ up to a final time $t = 8$. We observed that our method also preserves the probability density of particle in (3.26).

Additionally, we check the conservation of the mass for the initial condition

$$\int_{-20}^{20} |\psi_0|^2 dx = 0.4000$$

and, for $t = 8$, namely at the end point, we obtain

$$\int_{-20}^{20} |\psi_{PISM}|^2 dx = 0.3788, \quad \int_{-20}^{20} |\psi_{SMS}|^2 dx = 0.3780,$$

for the numerical solutions of PISM and SMS method. We deduce that there is little difference between the initial mass and the final mass for both PISM and SMS.

Although both PISM and SMS work well, Table 3.6 reveals that PISM is faster than SMS in the sense of CPU runtimes.

N_t	N_x	Δt	Δx	PISM	Strang Splitting
100	100	0.08	0.4	5.2188	6.1094
200	100	0.04	0.4	9.4844	13.0625
160	200	0.05	0.2	49.5938	75.1875
200	200	0.04	0.2	56.7813	100.2969

Table 3.6. Elapsed time for different N_t and N_x.

CHAPTER 4

AN ALTERNATIVE METHOD: THE PROPOSED LINEARIZED METHOD

Now, we present an alternative method for solving oscillation problems. The essential idea of the proposed method is Newton-Raphson iterative process combining with the Fréchet derivative. It can be also considered as a linearization technique. Thus, the new method is called the proposed linearized method.

A brief introduction is given in Appendix C for the Newton-Raphson method and for the Fréchet derivative.

4.1. Derivation of the Method

We first consider the general nonlinear differential equation as follows:

$$L(U) = 0 \tag{4.1}$$

where L is a differential operator. The solution of (4.1) is

$$U^{n+1} = U^n + \theta^n. \tag{4.2}$$

Here n corresponds to the iteration number and θ^n corresponds to the refinement variable for the correcting function U^n. To solve for the refinement variable, we deal with the following differential equation

$$\theta^n L'(U^n) + L(U^n) = 0. \tag{4.3}$$

Using the Fréchet derivative, the term $\theta^n L'(U^n)$ is

$$\theta^n L'(U^n) \;=\; \frac{\partial}{\partial\varepsilon} L(U^n + \varepsilon\theta^n)\Big|_{\varepsilon} = 0. \tag{4.4}$$

Instead of the simple formulation, we employ a more general one. For any system of equations, the formulation becomes

$$\begin{aligned} L_1(U, V) &= 0, \\ L_2(U, V) &= 0. \end{aligned} \tag{4.5}$$

Solving Equation (4.5) using the Newton-Raphson method results in

$$\begin{aligned} U^{n+1} &= U^n + {\theta_1}^n, \\ V^{n+1} &= V^n + {\theta_2}^n. \end{aligned} \tag{4.6}$$

As mentioned before n is the iteration number and ${\theta_1}^n$ and ${\theta_2}^n$ are the refinements obtained from the following differential equations.

$$\begin{aligned} {\theta_1}^n {L_1}'(U^n, V^n) + {\theta_2}^n {L_1}'(U^n, V^n) + L_1(U^n, V^n) &= 0, \\ {\theta_1}^n {L_2}'(U^n, V^n) + {\theta_2}^n {L_2}'(U^n, V^n) + L_2(U^n, V^n) &= 0. \end{aligned} \tag{4.7}$$

In order to get ${\theta_i}^n {L_j}'(U^n, V^n), \quad i, j = 1, 2$ we use Fréchet derivatives. Then,

$$\begin{aligned} {\theta_1}^n {L_i}'(U^n, V^n) &= \frac{\partial}{\partial\varepsilon} L_i(U^n + \varepsilon{\theta_1}^n, V^n)\Big|_{\varepsilon=0}, \\ {\theta_2}^n {L_i}'(U^n, V^n) &= \frac{\partial}{\partial\varepsilon} L_i(U^n, V^n + \varepsilon{\theta_2}^n)\Big|_{\varepsilon=0}. \end{aligned} \tag{4.8}$$

The following figure outlines the proposed linearized method.

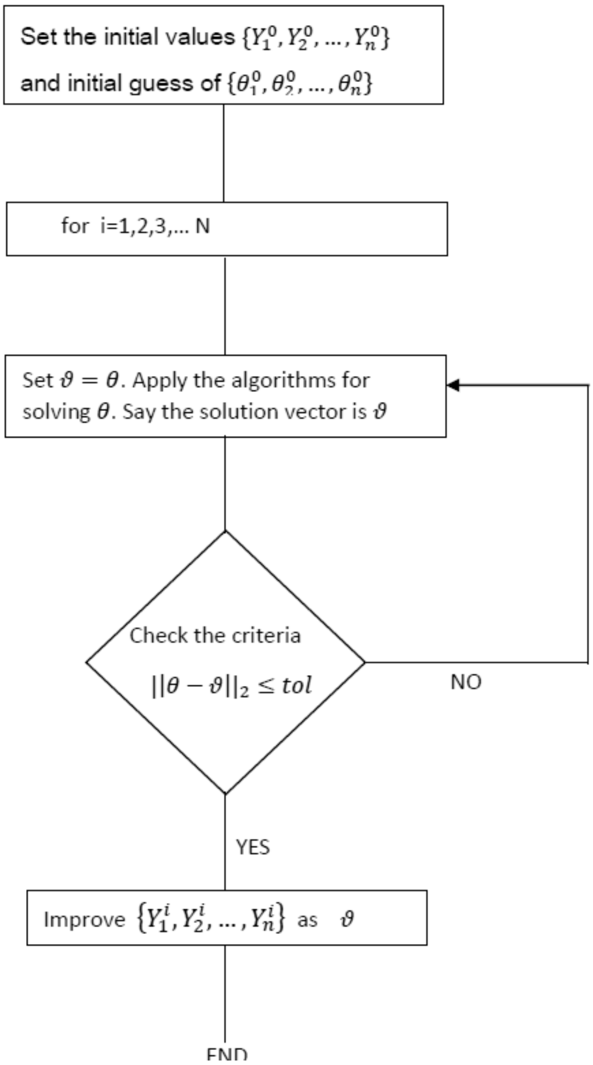

Set the initial values $\{Y_1^0, Y_2^0, \ldots, Y_n^0\}$
and initial guess of $\{\theta_1^0, \theta_2^0, \ldots, \theta_n^0\}$
for i=1,2,3,... N
Set $\vartheta = \theta$. Apply the algorithms for solving θ. Say the solution vector is ϑ
Check the criteria
$||\theta - \vartheta||_2 \leq tol$
NO
YES
Improve $\{Y_1^i, Y_2^i, \ldots, Y_n^i\}$ as ϑ
END

Figure 4.1. Diagram for the proposed linearized method.

Here ϑ is used to denote the temporary variable. The convergence criterion is controlled by the Euclidean norm,

$$||\theta - \vartheta||_2 \quad \leq \quad 10^{-6}. \tag{4.9}$$

4.2. Application to the Three Oscillation Problems

This section applies the proposed linearized method to the nonlinear oscillation problems. In the following subsections we present the application of the proposed linearized method on damped oscillator, Van-der Pol equation and cubic Schrödinger equation.

4.2.1. Damped oscillator

Consider the one dimensional damped oscillator

$$q'' + q + \alpha(q')^3 = 0. \tag{4.10}$$

Redefine $q(t) = q_1(t)$ and $\dot{q}(t) = q_2(t)$ leading to

$$\begin{aligned} q_1{}' - q_2 &= 0, & (4.11)\\ q_2{}' + q_1 + \alpha q_2{}^3 &= 0. & (4.12) \end{aligned}$$

Thus, we have system of two equations

$$\begin{aligned} L_1(q_1, q_2) &= q_1{}' - q_2, & (4.13)\\ L_2(q_1, q_2) &= q_2{}' + q_1 + \alpha q_2{}^3. & (4.14) \end{aligned}$$

For simplicity, we deal with the time interval $[0, h]$. Using the methods given in Equation (4.7) and Equation (4.8), we obtain

$$\begin{aligned} \frac{\partial}{\partial\varepsilon}[q_1{}' + \varepsilon\theta_1{}' - q_2]\Big|_{\varepsilon=0} &+ \frac{\partial}{\partial\varepsilon}[q_1{}' - q_2 - \varepsilon\theta_2]\Big|_{\varepsilon=0} + L_1(q_1, q_2),\\ \frac{\partial}{\partial\varepsilon}[q_2{}' + (q_1 + \varepsilon\theta_1) + \alpha(q_2)^3]\Big|_{\varepsilon=0} &+ \frac{\partial}{\partial\varepsilon}[q_2{}' + \varepsilon\theta_2{}' + q_1 + \alpha(q_2 + \varepsilon\theta_2)^3]\Big|_{\varepsilon=0} + L_2(q_1, q_2). \end{aligned}$$

Consequently,

$$
\begin{aligned}
\theta_1{}' - \theta_2 + q_1{}' - q_2 &= 0, && (4.15)\\
\theta_2{}' + \theta_1 + 3\alpha {q_2}^2\theta_2 + q_2{}' + q_1 + \alpha {q_2}^3 &= 0. && (4.16)
\end{aligned}
$$

In matrix form

$$
\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}\begin{bmatrix} \theta_1 \\ \theta_2 \end{bmatrix}_t + \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix}\begin{bmatrix} \theta_1 \\ \theta_2 \end{bmatrix} + \begin{bmatrix} 0 \\ 3\alpha {q_2}^2\theta_2 \end{bmatrix} = -\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}\begin{bmatrix} q_1 \\ q_2 \end{bmatrix}_t - \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix}\begin{bmatrix} q_1 \\ q_2 \end{bmatrix} - \begin{bmatrix} 0 \\ \alpha {q_2}^3 \end{bmatrix}, \tag{4.17}
$$

or

$$
I_2\Theta_t + A\Theta + D(\Theta, Q) = -I_2Q_t - AQ - C(Q), \tag{4.18}
$$

where $\Theta = (\theta_1, \theta_2)^T$ and $Q = (q_1, q_2)^T$. Here, A is a matrix, $D(\Theta, Q)$ and $C(Q)$ are vectors defined in Equation (4.17). I_2 is the identity matrix. In what follows, we solve the above system for Θ applying Crank-Nicolson scheme, leading to

$$
I_2\frac{\Theta^{n+1} - \Theta_n}{\Delta t} + A\frac{\Theta^{n+1} + \Theta^n}{2} + D(\Theta_n^{n+1}, Q_n^{n+1}) = -I_2\frac{Q^{n+1} - Q_n}{\Delta t} - A\frac{Q^{n+1} + Q^n}{2} - C(Q_n^{n+1}). \tag{4.19}
$$

After solving the system for Θ, we need to check the convergence condition given in Equation (4.9) where $\theta = \Theta^n$ and $\vartheta = \Theta^{n+1}$. If the condition holds, then

$$
Q^{n+1} = Q^n + \Theta^{n+1}. \tag{4.20}
$$

The numerical results are given in Chapter 5.

4.2.2. Van-der Pol Equation

For our second application, we consider the second order autonomous Van-der Pol equation which is one of the most generally studied systems in non-linear dynamics. That is,

$$\ddot{x} + \mu(x^2 - 1)\dot{x} + x = 0, \tag{4.21}$$

The parameter μ is a positive scalar denoting the non-linearity and the strength of the damping. We rewrite the equation into a system form

$$\begin{aligned} \dot{x}_1 &= x_2 \\ \dot{x}_2 &= -\mu({x_1}^2 - 1)x_2 - x_1 \end{aligned}$$

corresponding to

$$L_1(x_1, x_2) = {x_1}' - x_2 \tag{4.22}$$

$$L_2(x_1, x_2) = {x_2}' + \mu({x_1}^2 - 1)x_2 + x_1. \tag{4.23}$$

To establish the linearized system, we apply the method in Section 4.1. Following some tedious calculations,

$$\theta_1' - \theta_2 = -{x_1}' + x_2 \tag{4.24}$$

$$\theta_2' + \theta_1 + \mu\theta_2({x_1}^2 - 1) + 2\mu x_1 x_2 \theta_1 = -{x_2}' - \mu({x_1}^2 - 1)x_2 - x_1. \tag{4.25}$$

That is,

$$\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \theta_1 \\ \theta_2 \end{bmatrix}_t + \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} \theta_1 \\ \theta_2 \end{bmatrix} + \begin{bmatrix} 2\mu x_1 x_2 \theta_1 \\ \mu({x_1}^2 - 1)\theta_2 \end{bmatrix} = -\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} q_1 \\ q_2 \end{bmatrix}_t - \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} q_1 \\ q_2 \end{bmatrix} - \begin{bmatrix} 0 \\ \mu({x_1}^2 - 1)x_2 \end{bmatrix}. \tag{4.26}$$

Equation (4.26) in a compact form is

$$I_2\Theta_t + A\Theta + D(\Theta, X) = -I_2X_t - AX - C(X), \tag{4.27}$$

where $\Theta = (\theta_1, \theta_2)^T$ and $X = (x_1, x_2)^T$. Here, A is a matrix, $D(\Theta, X)$ and $C(X)$ are vectors defined in Equation 4.26. I_2 is identity matrix. We solve the above system for Θ applying Crank-Nicolson scheme, leading to

$$I_2\frac{\Theta^{n+1} - \Theta_n}{\Delta t} + A\frac{\Theta^{n+1} + \Theta^n}{2} + D(\Theta_n^{n+1}, X_n^{n+1}) = -I_2\frac{X^{n+1} - X_n}{\Delta t} - A\frac{X^{n+1} + X^n}{2} - C(X_n^{n+1}) \tag{4.28}$$

After solving the system for Θ, we need to check the convergence condition given in Equation (4.9) where $\theta = \Theta^n$ and $\vartheta = \Theta^{n+1}$. If the condition holds, then

$$Q^{n+1} = Q^n + \Theta^{n+1}. \tag{4.29}$$

The numerical results are given in Chapter 5.

4.2.3. Nonlinear Schrödinger Equation

Finally, to see the efficiency of the method on partial differential equations, we handle the cubic nonlinear Schrödinger equation

$$\begin{aligned} i\hbar\psi_t &= \hat{H}\psi(x,t) && (4.30)\\ i\hbar\psi_t + \beta\psi_{xx} + (G(x) + \alpha|\psi|^2)\psi &= 0 && (4.31)\end{aligned}$$

where $\psi(x,t)$ denotes the probability amplitude of the particle to be found at position x at time t, $\hbar$ is the Planck constant and $G(x)$ is the external potential. We consider the Dirichlet boundary conditions

$$\psi(x_L, t) = \psi(x_R, t) = 0, \quad t \in [0, t_{end}] \tag{4.32}$$

with the initial condition

$$\psi(x,0) \;=\; \psi_0, \quad x \in (x_L, x_R). \tag{4.33}$$

We rewrite the equation into the system by applying change of variables, $\psi = U + iV$, we get the following two differential equation

$$\begin{aligned} U_t + \beta V_{xx} + (G(x) + \alpha(U^2 + V^2))V &= 0, && (4.34)\\ V_t - \beta U_{xx} - (G(x) + \alpha(U^2 + V^2))U &= 0. && (4.35)\end{aligned}$$

Here $L_1(U, V)$ and $L_2(U, V)$ represent Equation (4.34) and Equation (4.35), respectively. By repeating the procedure given in Section 4.1, we obtain

$$\Theta_t + A\Theta + B\Theta + C(U, V)\Theta = -\Psi_t - A\Psi - B\Psi - D(|\Psi|)\Psi, \tag{4.36}$$

where $\Theta = (\theta_1, \theta_2)^T$ and $\Psi = (U, V)^T$. In addition,

$$A = \begin{bmatrix} \mathbf{0} & \beta\partial_x^2 \\ -\beta\partial_x^2 & \mathbf{0} \end{bmatrix}, \qquad B = \begin{bmatrix} \mathbf{0} & G(x) \\ -G(x) & \mathbf{0} \end{bmatrix}$$

$$C(U, V) = \begin{bmatrix} 2\alpha UV & 3\alpha V^2 + \alpha U^2 \\ -3\alpha U^2 - \alpha V^2 & -2\alpha UV \end{bmatrix}, \qquad D(|\Psi|) = \begin{bmatrix} \mathbf{0} & \alpha(U^2 + V^2) \\ -\alpha(U^2 + V^2) & \mathbf{0} \end{bmatrix}.$$

Here, the operator ∂_x^2 is also corresponding to a matrix due to the central difference approximation and $\mathbf{0}$ denotes the zero matrix. In order to solve the system given in Equation (4.36), we apply Crank-Nicolson method as in Subsection 4.2.1. Then, we solve the system to find Θ.

CHAPTER 5

NUMERICAL TESTS AND SIMULATIONS: THE PROPOSED LINEARIZED METHOD

This Chapter presents numerical results for the method introduced in Chapter 4. As it mentioned before, the focus is essentially on the oscillation problems. Since proposed linearized method (PLM) is explicit, it is necessary to test not only its accuracy but also its conservative structure. To that end in addition to damped oscillator, Van-der Pol equation and nonlinear Schrödinger equation are studied. The application of PLM to the equations are given in Section 4.2. The errors and the CPU runtimes are included to judge the efficiency of the method. Accuracy of the results are computed by means of L_1, L_2 and L_∞ norms given by

$$\begin{aligned} ||.||_{L_1} &= \sum_{n=0}^{n=N} |u(nh) - U(nh)|, \\ ||.||_{L_2} &= (\sum_{n=0}^{n=N} |u(nh) - U(nh)|^2)^{1/2} \\ ||.||_{L_\infty} &= \max_{0 \le n \le N} |u(nh) - U(nh)|. \end{aligned}$$

5.1. Damped Oscillator

To see the performance of the method, firstly we consider the damped oscillator. The algorithm given in Figure 4.1 is applied to the linearized Equation (4.17)

We solve the problem given in Equation (4.10) with $q(t) = 1$ and $q'(t) = 0$. The analytic solution of the method is

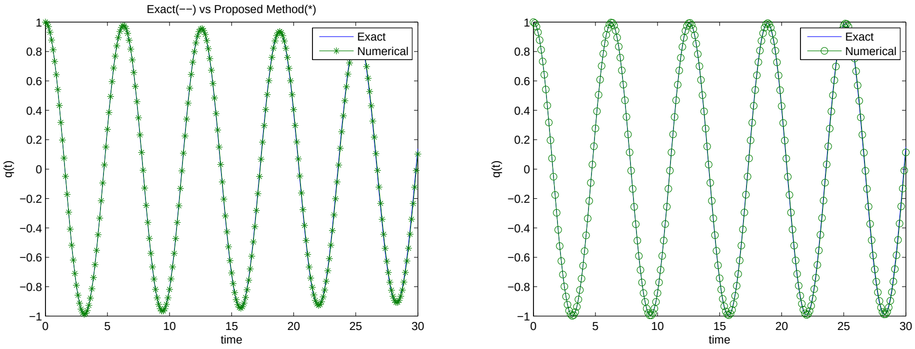
$$y(t) = \frac{\cos t}{\sqrt{1 + \frac{3\alpha t}{4}}}, \quad \text{where} \quad \alpha \to 0^+, \tag{5.1}$$

see (Bender and Orszag, 1999) for details.

In Table 5.1, the errors are presented in L_1, L_2 and L_∞ norms. We compare the numerical solution with the analytical one on $[0, 1]$ for various α. Table 5.1 reveals that PLM is more efficient than the second order Runge-Kutta method (RK2) Figure 5.1 illustrates that there is no difference between the numerical solutions and the analytical solutions for $\alpha = 0.1$ and $\alpha = 0.001$ on $[0, 10]$.

Method	α	Δt	L_∞	L_2	L_1
PLM	0.0001	2^{-3}	0.0011	0.0017	0.0038
RK2		2^{-3}	0.0020	0.0030	0.0065
PLM	0.0001	2^{-4}	2.9783e-004	6.1720e-004	0.0020
RK2		2^{-4}	5.1439e-004	0.0010	0.0030
PLM	0.0001	2^{-5}	9.2573e-005	2.8095e-004	0.0013
RK2		2^{-5}	1.1844e-004	2.9516e-004	0.0012

Table 5.1. Numerical errors for different Δt. Estimated errors using L_∞, L_2 and L_1 norm are obtained comparing with the analytical solution on $[0, 1]$.



(a) $\alpha = 0.01$

(b) $\alpha = 0.001$

Figure 5.1. Exact solution and PLM for solving equation in 4.10.

To see the effects of nonlinearity on the method, we choose as $\alpha = 1$. To the best our knowledge there is no exact solution to Equation (4.10). We compare our result with the ODE45 in MATLAB; $\Delta t = 1/8$ on $[0, 30]$.

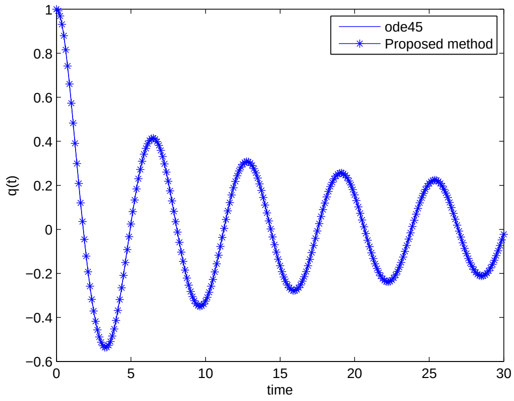


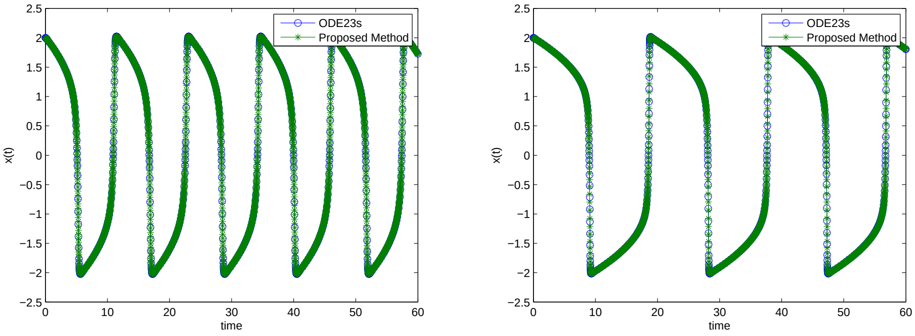
Figure 5.2. PLM vs ODE45 where $\alpha = 1$.

It can be seen from the Figure 5.2, the method introduced in Chapter 4 is in a perfect agreement with the ODE45 in MATLAB.

5.2. Van-der Pol Equation

As our second example, we consider the Van-der Pol equation given in Equation (4.21). In this section, we focus on the case of $\mu \geq 1$, i.e., the relaxation oscillations.

In Figure 5.3, we use $\Delta t = 0.01$ on $[0, 60]$ where $\mu = 5$ and $\mu = 10$, respectively. We compare our solutions with the ODE23s code in MATLAB since the exact solution of the problem for these parameters is unknown.



(a) $\mu = 5$

(b) $\mu = 10$

Figure 5.3. PLM vs. ODE23s obtained for various values of μ.

As a consequence of linearization, PLM is explicit. To the best our knowledge explicit methods do not preserve various qualitative property. Thus, we also check the limit cycle of the Van-der Pol equation for PLM. From the Figure 5.4, we deduce that PLM preserves the limit cycle of the equation even though it is explicit. Here, $\Delta t = 0.01$ on $[0, 60]$ with $\mu = 10$.

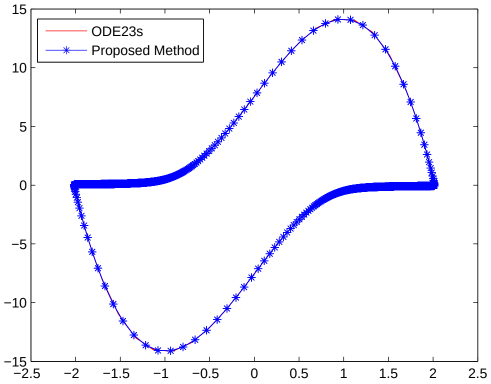
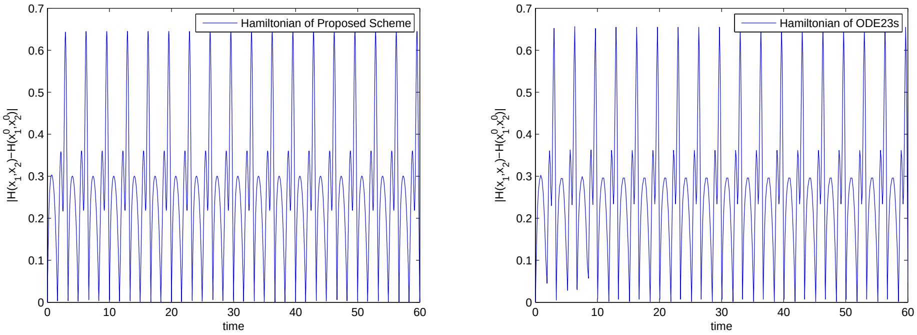


Figure 5.4. Limit cycle obtained from the solutions from PLM and ODE23s.

For the parameters above, we also observe that the elapsed time by considering same number of steps for PLM is 0.2031 whereas for the software of MATLAB, ODE23s, is 1.4063 Therefore, we deduce that PLM is faster than ODE23s in CPU runtime.

As mentioned in Chapter 3, Van-der Pol equation is a Hamiltonian system as given in Equation (3.9). Figure 5.5 demonstrates the Hamiltonian of Equation (4.21) with PLM and ODE23s.



(a) PLM (b) ODE23s in MATLAB

Figure 5.5. The Hamiltonian of Van-der Pol Equation (3.6) for $\mu = 1$ on $[0, 60]$.

For the exhibited figure, Figure 5.5, we study with $\mu = 1$ and $\Delta t = 0.01$ on $[0, 60]$. Therefore, we also deduce that there is no difference between PLM and ODE23s. We note that ODE23s is an implicit method whereas PLM is explicit.

5.3. Nonlinear Schrödinger Equation

Our final objective is to apply the method constructed in Chapter 4 to Equation (4.31). As in Chapter 3, we delve into the two cases:

- without external potential, namely $G(x) = 0$.
- with real-valued external potential.

5.3.1. Equation in the form: $i\partial_t\Psi + \beta\partial_x^2\Psi + \alpha|\Psi|^2\Psi = 0$

Start with the case $G(x) = 0, \beta = \alpha = 1$ and $\hbar = 1$. That is,

$$\begin{aligned} i\psi_t + \psi_{xx} + |\psi|^2\psi &= 0, \\ \psi(x_L, t) = \psi(x_R, t) &= 0, \end{aligned} \tag{5.2}$$

We deal with two different initial conditions in our numerical implementations. In the first case the initial condition is

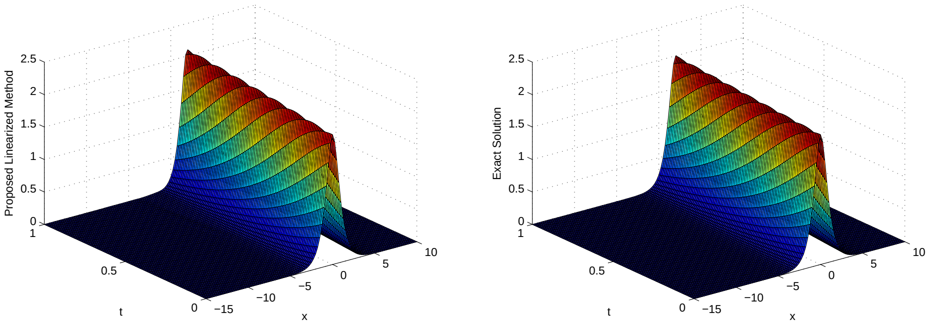
$$\psi(x, 0) = 2e^{ix} sech(\sqrt{2}x). \tag{5.3}$$

Here, the exact solution is

$$\psi(x, t) = \frac{\sqrt{2}e^{i(x+t)}}{\cosh(\sqrt{2}x - 2\sqrt{2}t)}. \tag{5.4}$$

For more details, we refer to the study of *Polyanin & Zaitsev*, (Polyanin and Zaitsev, 2003).

Figure 5.6 and Figure 5.7 are exhibited on the space interval $[-15, 10]$ divided into 100 uniform grid points with $\Delta x = 0.25$. We integrate the system using PLM with time step size $\Delta t = 0.01$ up to final time $t = 1$. The numerical solutions are in a perfect agreement with the analytic one, see Figure 5.6.

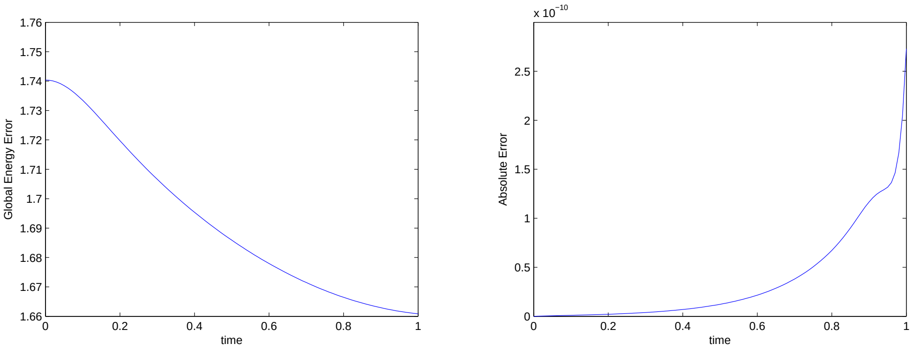


(a) PLM of Equation (5.2) and (5.3)

(b) Exact Solution

Figure 5.6. The numerical solutions are obtained for $\Delta x = 0.25$ and $\Delta t = 0.01$ where $N_x = N_t = 100$.

Figure 5.7 illustrates the global energy error and the absolute error.



(a) Global Energy Error for equations (5.2) and (5.3)

(b) Absolute Error for equations (5.2) and (5.3)

Figure 5.7. The numerical solutions are obtained for $t \in [0, 1]$ and $x \in [-15, 10]$ where $N_x = N_t = 100$. The exhibited figures belong to global energy error on the left and the absolute error on the right.

We investigate the energy and the mass conservation

$$M(t) := \int_{x_L}^{x_R} |\psi|^2 dx := M(0), \tag{5.5}$$

$$E(t) := \int_{x_L}^{x_R} \left(- |\psi_x|^2 + \frac{|\psi|^4}{2} \right) dx := E(0). \tag{5.6}$$

Note that for conservation of the mass the absolute error is evaluated by

$$AE = \Delta x \sum_{k=1}^{N_x} |M_k^n - M_k^0|$$

in discrete space. Here, $M_i^n = (U_i^n)^2 + (V_i^n)^2$, for $\Psi = \mathrm{U} + \mathrm{iV}$ and M^0 denotes the initial mass. The global energy in discrete space is

$$GE = \Delta x \sum_{k=1}^{N_x} |E_k^n - E_k^0|$$

where E^0 denotes the initial energy.

Table 5.2 is presented to show the efficiency of the method. It can be seen from Table 5.2 that the mass is preserved for all L_∞, L_2 and L_1 norms. In our implementation $N_x = N_t = 100$ grid points are taken.

t_{end}	Δt	L_∞	L_2	L_1
0.3	0.003	1.2150e-012	6.2350e-012	5.2767e-011
0.5	0.005	6.2839e-012	2.6592e-011	2.0517e-010
0.8	0.008	5.4293e-011	1.8396e-010	1.2022e-009
1	0.01	2.7304e-010	6.2927e-010	3.6911e-009

Table 5.2. Estimated errors using L_∞, L_2 and L_1 norm for conservation of the mass with $N_x = N_t = 100$ where $\Delta x = 0.25$.

The conserved density from the exact solution is

$$\int_{-15}^{10} |\psi|^2 dx = 5.6569, \quad \int_{-15}^{10} (-|\psi_x|^2 + \frac{|\psi|^4}{2}) dx = 3.1884$$

while from the numerical solution at $t = 1.0$ we have

$$\int_{-15}^{10} |\psi|^2 dx = 5.6569, \quad \int_{-15}^{10} (-|\psi_x|^2 + \frac{|\psi|^4}{2}) dx = 3.2694.$$

Secondly, we take the initial condition for Equation (5.2)

$$\psi(x,0) = \sqrt{2}sech(x+10)e^{ix/4}. \tag{5.7}$$

The exact solution is given in (Polyanin and Zaitsev, 2003) as

$$\psi(x,t) = \sqrt{2}sech(x-\frac{t}{2}+10)e^{i(\frac{x}{4}+\frac{15}{16}t)}. \tag{5.8}$$

Figure 5.8 is pointed out that PLM works well for the soliton solution. Although it is a linearization technique, it is in a perfect agreement with the exact solution for a long-time. We set $N_t = N_x = 100$ on which $x \in [-20,5]$ for Equation (5.2) and (5.7).

One of the most important properties of solitary waves is that they can travel enormous distances without changing their whole structure or energy. Figure 5.9 suggests that the PLM does not cause any alteration in the structure of Equation (5.2). The exact and the numerical solutions of Equation (5.2) are exhibited for $t = 5$, $t = 10$, $t = 15$ and $t = 20$, respectively. Here, $N_t = N_x = 100$ on $x \in [-20,5]$ and $t \in [0,20]$.

Due the definition of the solitary wave, the objective of this section is followed up by checking mass and energy conservation of the method. It is necessary for testing the validity of the given method. Thus,

$$\int_{-20}^{5} |\psi|^2 dx = 4.0000, \quad \int_{-20}^{5} (-|\psi_x|^2 + \frac{|\psi|^4}{2}) dx = 1.1749$$

while we have from the numerical solution at $t = 3$

$$\int_{-20}^{5} |\psi|^2 dx = 4.0000, \quad \int_{-20}^{5} (-|\psi_x|^2 + \frac{|\psi|^4}{2}) dx = 1.2162.$$

Moreover, we have the following results numerical results at $t = 20$

$$\int_{-20}^{5} |\psi|^2 dx = 3.9998, \quad \int_{-20}^{5} (-|\psi_x|^2 + \frac{|\psi|^4}{2}) dx = 1.1850,$$

which guarantee that PLM has the long-time behavior for the conservative quantities.

To comment on the efficacy of PLM, the mass conservation is investigated. In Table

5.3 the absolute error is indicated in different norm for different N_t and N_x.

N_t	N_x	L_∞	L_2	L_1
100	100	2.8572e-006	8.6262e-006	4.4454e-005
100	150	7.6629e-007	2.4568e-006	1.3471e-005
150	100	2.1974e-006	8.1635e-006	5.1972e-005
150	200	2.3080e-007	9.4481e-007	6.8182e-006

Table 5.3. Estimated errors using L_∞, L_2 and L_1 norm for conservation of the mass where $t \in [0, 3]$ and $x \in [-20, 5]$.

Table 5.4 emphasizes that PLM also preserves the mass on $t \in [0, 20]$.

N_t	N_x	L_∞	L_2	L_1
100	100	2.4327e-004	4.9666e-004	0.0028
150	100	2.8689e-004	8.3550e-004	0.0069
100	150	1.0418e-004	1.8285e-004	6.5519e-004
150	200	6.8457e-005	1.4257e-004	5.8034e-004

Table 5.4. Estimated errors using L_∞, L_2 and L_1 norm for conservation of the mass where $t \in [0, 20]$ and $x \in [-20, 5]$.

At last, we test the CPU runtime of PLM for various Δx and Δt in Table 5.5.

Δx	Δt	Elapsed Time
100	100	5.7344
100	150	8.2031
150	150	19.7969

Table 5.5. The elapsed time is measured in seconds where $t \in [0, 3]$ and $x \in [-20, 5]$.

5.3.2. Equation in the form: $i\partial_t\Psi + \beta\partial_x^2\Psi + (G(x) + \alpha|\Psi|^2)\Psi = 0$

We now turn our attention to the cubic Nonlinear Schrödinger equation with the external potential. As in Section 3.3.2 $G(x) = -\frac{1}{1+\sin^2(x)}$ where $\hbar = 1, \beta = \frac{1}{2}$ and $\alpha = -30$. Thus, we have

$$i\psi_t + \Big(\frac{1}{2}\frac{\partial^2}{\partial x^2} - (\frac{1}{1+\sin^2 x} + 30 * |\psi|^2)\Big)\psi \quad = \quad 0. \tag{5.9}$$

Here the boundary conditions are given as $\psi(x_L, t) = \psi(x_R, t) = 0$ and the initial condition is taken as $\psi_0(x) = \gamma \exp(\sin 2x)$.

The probability density of any particle in Equation (5.9) is defined as

$$\int_{x_L}^{x_R} ||\psi|| dx \leq 1.$$

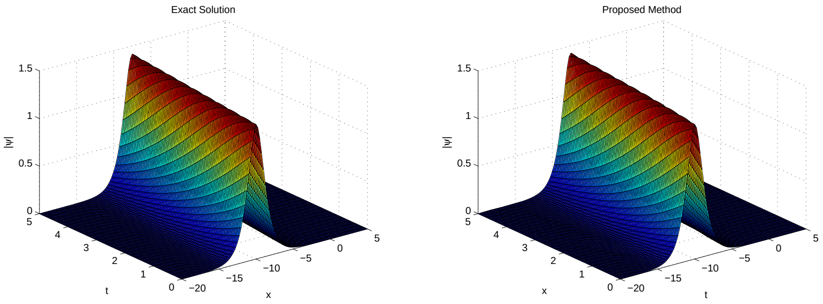
To verify the solutions obtained from PLM, we need to set γ as $\frac{1}{||\psi_0||}$ due to

$$\int_{x_L}^{x_R} ||\psi_0|| dx = 1.$$

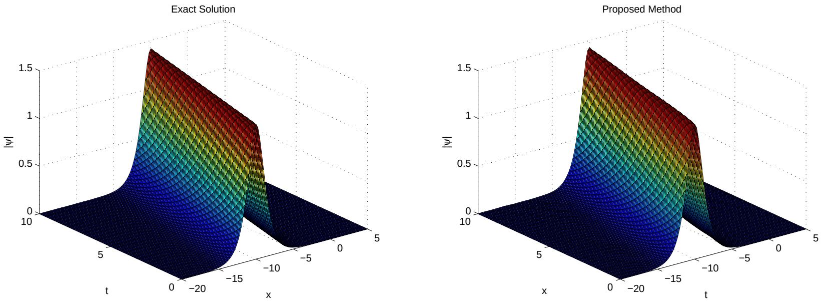
For both Figure 5.10(a) and Figure 5.10(b) we assume that the system is defined on the interval $x \in [-20, 20]$, which is split into $N_x = 100$ parts. We integrate the system with the time-step size $\Delta t = 0.08$ up to final time $t = 8$. Figure 5.10 demonstrates that our method preserves the probability density of particle in (5.9) for the given potential.

Figure 5.2 reveals that PLM preserves the probability of the particle in Equation (5.9).

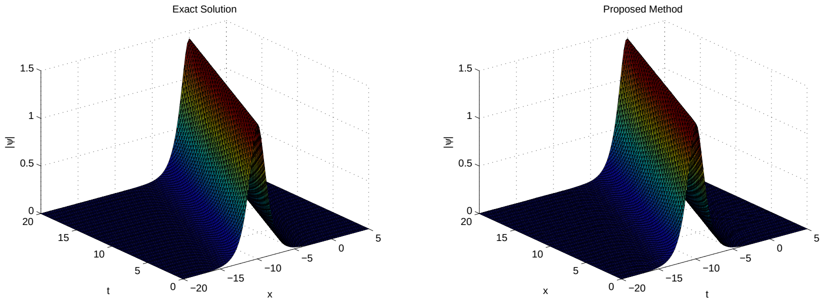
To compare PLM with the second order Runge-Kutta(RK2) method, we also study for $x \in [-20, 20]$ where $N_x = N_t = 100$. The system is integrated to the final time $t = 4$. Figure 5.11 illustrates that the numerical solutions from PLM and RK2, respectively. In Figure 5.11, the numerical solutions are similar to each other. However, when $t_{end} = 5$ the RK2 does not preserve the probability density. The Figure 5.12 shows that the RK2 method does not preserve the probability density of the particle in Equation (4.31) whereas PLM does, see Figure 5.10.



(a) Exact Solution for $t_{end} = 5$ (b) PLM for $t_{end} = 5$



(c) Exact Solution for $t_{end} = 10$ (d) PLM for $t_{end} = 10$



(e) Exact Solution for $t_{end} = 20$ (f) PLM for $t_{end} = 20$

Figure 5.8. Comparison of the numerical solution and the exact solution of the Equation (5.2) with the initial condition given in (5.7) for various values of t_{end}.

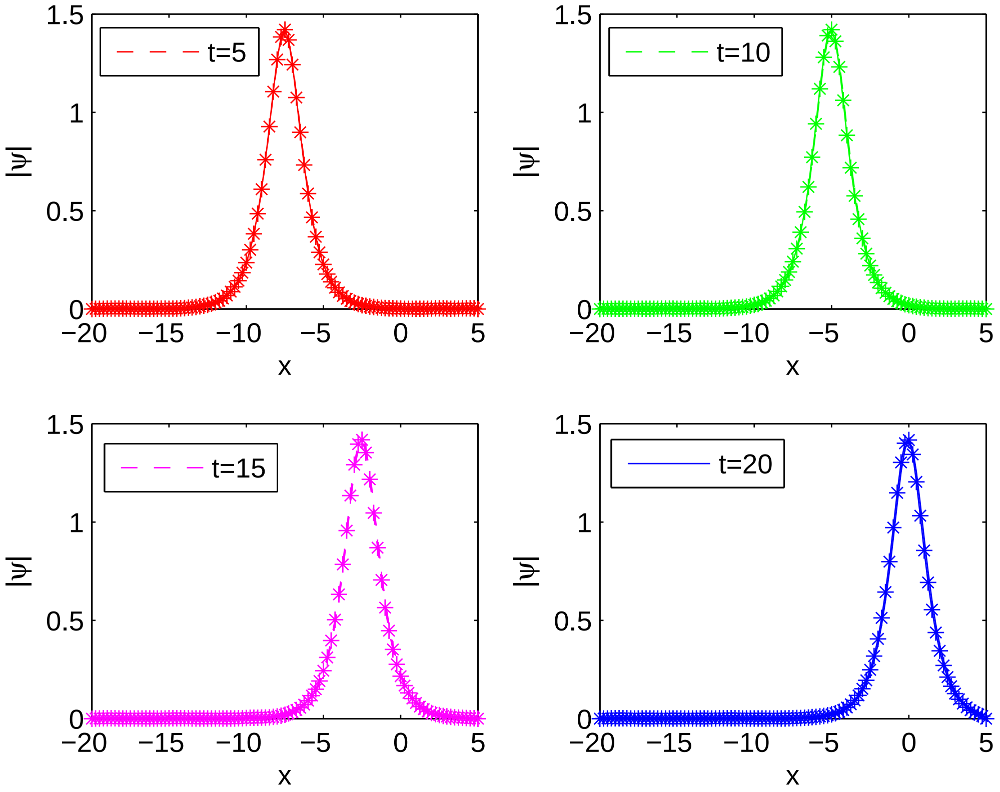
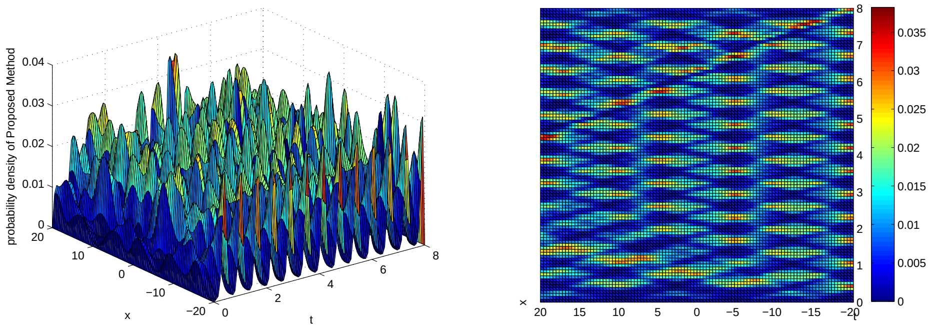


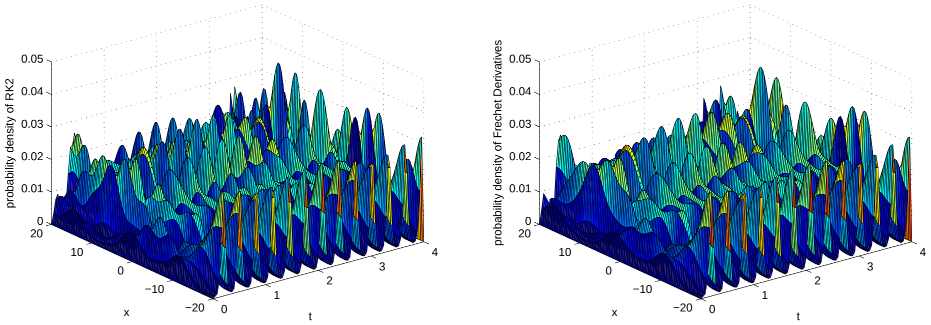
Figure 5.9. Exact and the numerical solutions of Equation (5.2) at certain time.



(a) Numerical Solution

(b) Contour Plot

Figure 5.10. Probability density of the particle in Equation (5.9) for PLM.



(a) The RK2

(b) PLM

Figure 5.11. Probability density of the particle in Equation (5.9). PLM is given on the right whereas the RK2 method on the left.

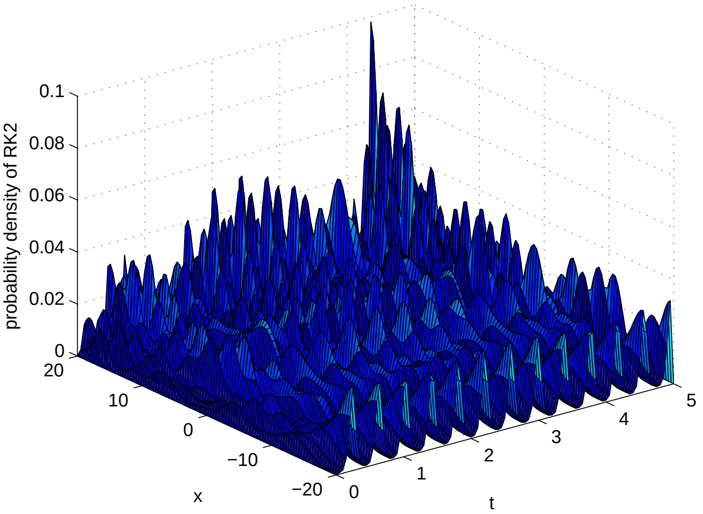


Figure 5.12. The numerical solutions obtained from the RK2 method for $t \in [0, 5]$.

CHAPTER 6

CONCLUSION

Nonlinear oscillations related with the vibrating systems have important role in engineering. Thus, the solving nonlinear oscillation problems is crucial. The main purpose of this thesis is to propose two alternative numerical methods for solving nonlinear oscillation equations.

First of all, we construct a method which is based on the Magnus expansion and the iterative splitting procedure. Afterwards, the consistency and stability properties are analyzed bounded and unbounded operators. For that purpose, we utilize from the Taylor expansion and C_0 semigroup approaches combined with the exponential integrator, respectively. We get the convergence of the proposed method using "Lady windermere's fan" argument.

Since the proposed method is a kind of operator splitting method, we compare it with the traditional splitting methods. We test the proposed method on the Duffing equation in order to see the efficiency. From Table 3.1, we deduce that the proposed method achieves second order accuracy. The well-known second order splitting methods, namely Strang splitting and symmetrically weighted splitting, have reduction of order. Since Strang splitting and symmetrically weighted splitting have similar behavior and order, we compare the proposed method with only Strang splitting method. Strang splitting involves three subequations, while the proposed method involves two. Thus, from a computational point of view, the proposed method is more efficient than Strang splitting. The method is also applied to Van-der Pol oscillator in nonlinear dynamics. Although the proposed method is explicit Figure 3.4 and 3.3 demonstrate that it is in a perfect agreement with the ODE23s code in MATLAB which is an implicit method. Then, the cubic nonlinear Schrödinger equation(NLSE) in nonlinear optics is considered for two cases: without and with external force. In the first case, we present the numerical solutions of the proposed method for the exact soliton solution. Additionally, some of the conserved densities, namely mass and energy, are given for the correctness of the solutions. Finally, the probability density of the particle for NLSE with the external force is illustrated in Figure 3.10 and Figure 3.11 for the proposed method and Strang splitting, respectively.

Due to the stability problem of the proposed iterative splitting method, in the second part of this thesis, we propose an alternative method based on the "Newton-Raphson" and the "Fréchet derivatives." To the best our knowledge this is the first application of the Fréchet derivatives combining with the central difference approximation. In the procedure of the proposed linearized method, we use the Crank-Nicolson method. We do not analyze of the proposed linearized method (PLM) in this thesis. Thus, this part of the dissertation is mainly devoted to application and implementation of PLM.

We solve damped oscillator, Van-der Pol equation and the cubic nonlinear Schrödinger equation using PLM. We test the proposed linearized method on the damped oscillator. Since the analytical solution is given for $\alpha \to 0^+$, to see the efficiency of the method, we present the errors for $\alpha = 0.0001$ comparing then with 2nd order Runge-Kutta method in different norms, namely L_∞, L_2 and L_1. We also compare PLM with ODE23s from MATLAB. This code is known as an implicit solver. Figure 5.3 and Figure 5.5 reveals that PLM is also in a perfect agreement with the ODE23s. Moreover, we see that PLM preserves the limit cycle of the Van-der Pol equation, cf. Figure (5.4). As a result of the unconditionally stability for the nonlinear Schrödinger equation (NLSE) without external force, we see from the Figure 5.8 that the proposed linearized method and the exact solution have similar behavior for a long time. Furthermore, PLM preserves the probability density of NLSE for a long time, see Figure 5.10, whereas the second order Runge-Kutta method does not, see Figure 5.12.

Overall, we suggested two different methods for numerical solution of nonlinear oscillation problems. As it can be deduced from the results and simulations, both PISM and PLM are in a perfect agreement with the exact solution and the well-known solvers in MATLAB. Although the examples considered are a small sample of nonlinear oscillation equations, they suggest that the methods are easily adaptable to solve similar problems.

REFERENCES

Aydın A. and Karasözen B., 2011: Lobatto IIIA-IIIB discretization of the strongly coupled nonlinear Schrödinger equation. *Journal of Computational and Applied Mathematics,* 235, 4770-4779.

Bagrinovskii K. A. and Gudunov S. K., 1957: Difference schemes for multidimensional problems. *Dokl. Akad. Nauk SSSR(NS),* 115, 431-433.

Bátkai A., Csomós P and Nickel G., 2009: Operators and spatial approximations for evolution equations. *J. Evol. Equ.9,* 613-636.

Bender C.M. and Orszag S.A., 1999: Advanced Mathematical Methods for Scientists and Engineers. *Springer-Verlag, New York.*

Bert C. W. and Malik M., 1996: Differential quadrature method in computational mechanics: A review, *Applied Mechanics Review, 49,* 1-27.

Bülbül B. and Sezer M., 2013: Numerical Solution of Duffing Equation by Using an Improved Taylor Matrix Method, *Journal of Applied Mathematics,* http://dx.doi.org/10.1155/2013/691614.

Casas F. and Iserles A., 2006: Explicit Magnus expansions for nonlinear equations. *J. Phys. A: Math. Gen.* 39, 5445-5461.

Certaine J., 1960: The solution of ordinary differential equations with large time constants. *In Mathematical methods for digital computers Wiley, New York,* 128û132.

Cox S. M. and Matthews P. C., 2001: Exponential time differencing for stiff systems. *J. Comput. Phys.,* 176(2), 430-455.

Crandall M.G. and Majda A., 1980: The method of fractional steps for conservation laws. *Math. Comp.,* 34, 285-314.

Dehghan M. and Taleei A., 2010: A compact split-step finite difference method for solving the nonlinear Schrödinger equations with constant and variable coefficients, *Computer Physics Communications,* 181(1), 43-51.

Descombes S. and Thalhammer M., 2013: The Lie Trotter splitting method for nonlinear evolutionary problems involving critical parameters. An exact local error representation and application to nonlinear Schrödinger equations in the semi-classical regime. *IMA Journal of Numerical Analysis, Oxford University Press (OUP): Policy A - Oxford Open Option A*, 33, 2, 722-745.

Dimarogonas A. D. and Haddad S., 1992: Vibration for Engineers, *Prentice-Hall, Englewood Cliffs, New Jersey.*

Einkemmer L. and Ostermann A., 2013: Convergence Analysis of Strang Splitting for Vlasov-type Equations.

Engel K.J. and Nagel R., 2006: A short course on operator semigroups. *Universitext. Springer, New York.*

Faragó I. and Geiser J., 2007: Iterative operator splitting methods for linear problems. *International Journal of Computational Science and Engineering,* 3(4), 255-263.

Fazel M. R., Moghaddam M. M. and Poshtan J. 2013: Application of GDQ method in nonlinear manipulator undergoing large deformation, *J. Mech. Eng. Science,* 227(12), 2671-2685.

Fidlin A., 2006: Nonlinear Oscillations in Mechanical Engineering, *Springer-Verlag, Berlin Heidelberg.*

Ganji D.D., Karimpour S. and Ganji S.S., 2008: Approximate Analytical Solutions to Nonlinear Oscillations of Non-natural Systems Using He's Energy Balance Method.*Progress In Electromagnetics Research M,* 5, 43-54.

Geiser J., 2008: Iterative Operator-Splitting Methods with higher order Time- Integration Methods and Applications for Parabolic Partial Differential Equations.*Journal of Computational and Applied Mathematics, Elsevier,* 217, 227-242.

Geiser J., 2008: Decomposition methods for differential equations : Theory and application. *CRC Press, Taylor and Francis Group.*

Geiser J., Noack L., 2011: Iterative operator-splitting methods for nonlinear differential equations and applications of deposition processes. *Numerical Methods for Partial Differential Equations,* 27(5), 1026-1054.

Groves F. R., 1983: Numerical solution of nonlinear differential equation using computer algebra, *International Journal of Computer Mathematics, 13*, 301-309.

Hardin R.H. and Tappert F.D., 1973: Applications of the split-step Fourier method to the numerical solution of the nonlinear and variable coefficient wave equations. *SIAM Review 15*, 423.

He J.H., 1999: Homotopy perturbation technique, *Computer Methods in Applied Mechanics and Engineering, 178*, 257û262.

Hochbruck M. and Ostermann A., 2010: Exponential Integrator, *Acta Numerica, Cambridge University Press*, 209û286.

Holden H., Karlsen K. H., Lie K-A. and Risebro N. H., 2010 : Splitting Methods for Partial Differential Equations with Rough Solutions. Analysis and Matlab programs. *European Mathematical Society.*

Jahnke T. and Altıntan D., 2004: Efficient Simulation of discrete stochastic reaction system with a splitting method. *BIT Numerical Mathematics,* 50, Number 4, 797-822.

Iserles A. and Nørsett S.P., 1999: On the solution of linear differential equations in Lie groups. *Philos. Trans. Royal Soc. A,* 357, 983-1019.

Kanney J., Miller C. and Kelley C, 2003: Convergence of iterative-split operator approaches for approximating nonlinear reactive transport problems. *Advances in Water Resources,* 26, 247-261.

Kelly C., 1995: Iterative Methods for linear and nonlinear Equations. *Frontiers in Applied Math.,SIAM.*

Koch O., Neuhauser C., Thalhammer M., 2013: Error analysis of high-order splitting methods for nonlinear evolutionary Schrödinger equations and application to the MCT-DHF equations in electron dynamics. *ESAIM: Mathematical Modelling and Numerical Analysis,* 47, 5, 1265-1286

Krogstad S., 2005: Generalized integrating factor methods for stiff PDEs. *J. of Comp. Phys.*, 203, 1, 72û88.

Liao S.J. and Cheung A.T., 1998: Application of homotopy analysis method in nonlinear oscillations, *ASME Journal of Applied Mechanics,* 65, 914û922.

Liu G.R. and Wu T.Y., 2000: Numerical solution for Differential Equations of Duffing-Type non-linearity using the Generalized differential Quadrature Rule, *Journal of Sound and Vibration,* 237(5), 805-817.

Liu G. R. and Wu T. Y., 2001: An application of the generalized differential quadrature rule in Blasius and Onsager equations *International Journal for Numerical Methods in Engineering,* 52(9), 1013û1027.

Lubich C., 2008 On splitting methods for Schrödinger-Poisson and cubic nonlinear Schrödinger equations, *Math. Comp.*, 77, 2141û2153.

Magnus W., 1954: On the exponential solution of differential equations for a linear operator. *Commun. Pure Appl. Math.,* 7, 649-673.

Marchuk G.I., 1968: Some application of splitting-up methods to the solution of mathematical physics problems. *Aplik. Mat. ,* 13, No. 2, 103-132.

Mathews J.H. and Fink K.D. 2004: Numerical Methods usin MATLAB. *Pearson; 4 edition ,* ISBN-13:978-0130652485

McLachlan R.I. and Quispel G.R.W., 2002: Splitting Methods. *Cambridge University Press*, 341-434.

Mickens R.E.,2001: Mathematical and numerical study of the Duffing-harmonic oscillator, *Journal of Sound and Vibration,* 244, 563û567.

Munkres J., 1997: Analysis on Manifolds, *Westview Press.*

Nayfeh A. H. and Mook D. T.,1995: Nonlinear Oscillations, *John Wiley and Sons, Newyork-Chichester-Brisbane-Toronto-Singapore.*

Pechukas P. and Light J. C., 1966: On the exponential form of time-displacement operators in quantum mechanics. *J. Chem. Phys.,* 3897-3912.

Polyanin A. D. and Zaitsev V. F., 2003: Handbook of Nonlinear Partial Differential Equations. *Chapman and Hall/CRC.*

Robinson D.W., 1963: Multiple Coulomb excitations of deformed nuclei. *Helv. Phys. Acta,* 36, 54-140.

Rudin W., 1976: Principles of Mathematical Analysis, *McGraw-Hill Science/Engineering/Math; 3rd edition.*

Smith D.R., 1998: Variational Methods in Optimization, *Dover.*

Spivak M., 1971: Calculus on Manifolds, *Westview Press.*

Strang G., 1968: On the construction and comparison of difference schemes. *SIAM J. Numer. Anal.,* 5, No. 3, 506-517.

Tabatabaei K. and Günerhan E., 2014: Numerical Solution of Duffing Equation by the Differential Transform Method, *Appl. Math. Inf. Sci. Lett. 2,* No. 1, 1-6.

Tanoğlu G. and Korkut S., 2012: The convergence of a new symmetric iterative splitting method for non-autonomous systems, *International Journal of Computer Mathematics, 89,* 1837-1846, DOI:10.1080/00207160.2012.687447.

Trefethen L. N., 2000: Spectral Methods in MATLAB, *SIAM, Philadelphia.*

APPENDIX A

MAGNUS EXPANSION

Magnus integrators are an interesting class of numerical methods for Hamiltonian problems. The problem which Magnus expansion (ME) solves has a history dating back at least to the studies by *Peano*, at the end of 19th century, and *Baker* at the beginning of the 20th. They combine the theory of differential equations with the algebraic formulation. The so called Baker-Campbell-Hausdorff formula is related to these results. (Magnus, 1954) is considered the father of the Magnus expansion. The convergence of the problem than the ambiguous considerations in Magnus paper were studied in (Pechukas and Light, 2008). The study of *Robinson*, (Robinson, 1963), is the first application of the Magnus expansion to a physical problem. Between 1971 and 1990, Magnus expansion was successfully applied to a wide class of problems in Physics and Chemistry. In the last decade of the 20th century Magnus expansion has been adapted for specific types of equations such as stochastic differential equations. In numerical analysis using Magnus expansion as a geometric integrator was pioneered by *Iserles and Nørsett*, (Iserles and Nørsett, 1999). For a comprehensive overview for the nonlinear Magnus expansion, see (Casas and Iserles, 2006) and the references therein.

In this appendix, we briefly recall the Magnus expansion for a nonlinear equation. Magnus expansion establishes an common tool to find an approximate solutions of nonlinear operators such as

$$\frac{du}{dt} = A(t, u)u(t) , \tag{A.1}$$

with solution

$$u(t) = \exp(\Omega(t))u(0). \tag{A.2}$$

As in the linear case, Ω can be obtained by Picard's iteration,

$$\begin{aligned}
\Omega^{[0]} &\equiv 0, \\
\Omega^{[1]} &= \int_0^t V(s, v_0) ds, \\
\Omega^{[m]} &= \sum_{k=0}^{m-2} \frac{B_k}{k!} \int_0^t ad^k_{\Omega^{[m-1]}} V(s, e^{\Omega^{[m-1]}} v_0) ds. \quad m \geq 2
\end{aligned} \tag{A.3}$$

and taking the approximation $\Omega(t) \approx \Omega^{[m]}(t)$. Here $\{B_k\}_{k \in Z^+}$ denotes the Bernoulli numbers and ad^k represents the iterated commutator.

$$ad^0_\Omega A = A, ad^{k+1}_\Omega A = [\Omega, ad^k_\Omega A].$$

Using *Euler method*, we have the first order Magnus operator as

$$\Omega^{[1]} = \int_0^t V(s, v_0) ds = hV(0, v_0) + O(h^2). \tag{A.4}$$

With the aid of the *trapezoidal rule*, the second order one is

$$\Omega^{[2]} = \int_0^h V(s, e^{\Omega^{[1]}} v_0) ds = \frac{h}{2}\Big(V(0, v_0) + V(h, e^{\Omega^{[1]}} v_0)\Big) + O(h^3). \tag{A.5}$$

APPENDIX B

EXPONENTIAL INTEGRATORS

As their names emphasizes these integrators use the exponential function (and related functions) of the Jacobian or to approximate to the exponential function. The first study on Exponential integrators was published in 1960 by *Certaine* (Certaine, 1960). There are various studies in the literature on this subject, see Cox and Matthews (2014), (Krogstad, 2005), etc. Results presented here are available in the monograph of *Hochbruck and Ostermann*, (Hochbruck and Ostermann, 2010).

Without loss of generality, consider

$$u'(t) = A(t,u) = Tu + g(t,u) \tag{B.1}$$

$$u(0) = u_0. \tag{B.2}$$

By means of the *variation-of constant formula*, the exact solution of Equation (B.1) with the initial value (B.2) can be expressed as

$$u(t) = e^{Th}u_0 + \int_0^t e^{T(h-s)}g(u(s))ds \tag{B.3}$$

The main idea of the construction of exponential integrators is the linearization of a semi-linear or a nonlinear evolution equation. Linearizing Equation (B.1), we have

$$u'(t) = A(t_0,u_0) + \frac{\partial A}{\partial u}(t_0,u_0)(u-u_0) \tag{B.4}$$

where $\frac{\partial A}{\partial u}$ is the Jacobian of $A(t,u)$. Then the exact solution of the linearized equation (B.4) is

$$\begin{aligned} u(t) &= u_0 + h\varphi_1(h\frac{\partial A}{\partial u}(t_0,u_0))A(u_0), \\ u(t) &= e^{Th}u_0 + h\varphi_1(hT)g(t_0,u_0). \end{aligned}$$

Here the function φ_1 is defined as

$$\varphi_1(z) = \int_0^1 e^{(1-\tau)z}\, d\tau = \frac{e^z - 1}{z}.$$

More precisely,

$$\varphi_k(z) = \int_0^1 e^{(1-\tau)z} \frac{\tau^{k-1}}{(k-1)!} d\tau, \quad k \geq 1, \tag{B.5}$$

where $\varphi - functions$ satisfy $\varphi_k(0) = 1/k!$ and the recurrence relation

$$\varphi_{k+1}(z) \;=\; \frac{\varphi_k(z) - \varphi_k(0)}{z}, \quad \varphi_0(z) = e^z. \tag{B.6}$$

For the exponential integrators within the framework of semigroups refer to (Engel and Nagel, 2006).

Assumption B.1 *((Hochbruck and Ostermann, 2010), Assumption 2.2) Let X be a Banach space with norm* $||.||$*. Assume that T is a linear operator on X and that T is the infinitesimal generator of a strongly continuous semigroup* e^{tT} *on X. Then, there exist constants C and* ω *such that*

$$||e^{tT}||_{X \leftarrow X} \leq Ce^{\omega t}.$$

If the assumption above is satisfied, then we deduce that $\varphi_k(hT),\ k = 1, 2, 3, \ldots$ are bounded.

APPENDIX C

ANALYTICAL FRAMEWORK

The purpose of this appendix is to collect the definitions that are used throughout this report.

C.1. Newton-Raphson Method

Form a numerical point of view, the Newton-Raphson method is one of the most convenient and best known algorithms. The method was developed in the second half of 17th century for finding root of real-valued functions.

Theorem C.1 *Assume that* $f \in C^2[a,b]$ *and there exists a number* $p \in [a,b]$, *where* $f(p) = 0$. *If* $f'(p) \neq 0$, *then there exists a* $\delta > 0$ *such that the sequence* $\{p_k\}_{k=0}^{\infty}$ *define by the iteration*

$$p_k = p_{k-1} - \frac{f(p_{k-1})}{f'(p_{k-1})} \quad for \quad k = 1, 2, ...$$

will convergence to p for any initial approximation $p_0 \in [p - \delta, p + \delta]$.

For more details, we refer to (Mathews and Fink, 2004).

C.2. Derivatives in Banach Space

The main objective of this section is to describe the Gateaux and Fréchet derivatives. In arbitrary vector space, a generalized directional derivative is called Gateaux differential and a generalized gradient is called Fréchet derivative, see (Smith, 1998) for infinite- dimensional vector spaces and (Spivak, 1971), (Munkres, 1997) or (Rudin, 1976) for finite-dimensional case.

The idea of Gateaux derivative is based on the generalization of the directional derivative. The following definition states the Gateaux differential.

Definition C.1 *(Gateaux differential)*

Let $f : V \rightarrow U$ be a function and let $h \neq 0$ and x be vectors in V. The Gateaux differential $D_h f$ is defined

$$D_h f = \lim_{\epsilon \to 0} \frac{f(x + \epsilon h) - f(x)}{\epsilon}$$

The Fréchet derivative generalizes the idea of gradient.

Definition C.2 *Fréchet derivative Df of $f : V \rightarrow U$ is defined implicitly by*

$$f(x + k) = f(x) + (Df)k + o(\|k\|).$$

In order to see the association with the Gateaux differential, take $k = \epsilon h$, it leads to

$$f(x + \epsilon h) = f(x) + \epsilon(Df)h + ho(\epsilon).$$

Here, notice that

- there is not a single Gateaux differential at each point. In one dimension, there are two Gateaux differential for x due to the forward and backward direction. In higher dimensions, there are infinitely many Gateaux differentials at each point;
- the Fréchet derivative exists at $x = a$ if Gateaux differentials are continuous functions of x at $x = a$. If it exists for a function f at a point x, it is also unique, see (Munkres, 1997) or (Spivak, 1971);
- if f is Fréchet differentiable at x, then it is also Gateaux differentiable at x. The converse is not true.

APPENDIX D

MAT-LAB CODES FOR NEW ITERATIVE SPLITTING

D.1. Codes for Duffing equation

THE SECOND ORDER SPLITTING METHODS

```
%%%Duffing Equation y''+y+ep*y^3=0,y(0)=1,y'(0)=0%%%
clear all; close all; clc;
%% Problem definition
q10=1; q20=0; tson=10; U0=[q10;q20];
h=1/64; %%%%stepsize
t=0:h:tson; %%%interval
n=tson/h; %%%%%number of steps
P=U0;  Q=U0;
EP=1e-4; %%%\varepsilon
EX=zeros(1,n+1);
%% exact solution from (Bender and Orszag 1999)
for i=1:n+1 EX(i)=cos(t(i))+EP*(cos(3*t(i))/32-...
...cos(t(i))/32-(3*t(i)*sin(t(i)))/8);
end
%%%%%%%%%% Numerical Solutions%%%%%%%%%%%%%%%%%
%%%%%%%%%%  PROPOSED METHOD
%%% initialization
IT2=zeros(2,n+1); ITS=zeros(2,n+1); Tr2=zeros(1,n+1);
T=[0,1;-1,0]; %V(t,u) is given in function "nonlin"
IT2(:,1)=U0;   Tr2(1)=q10;  ITS(:,1)=U0;
for i=2:n+1
new=expm(h*nonlin(EP,IT2(1,i-1)))*IT2(:,i-1);
IT2(:,i)=(expm((h/2)*(nonlin(EP,IT2(1,i-1))+nonlin(EP,new(1,1))))*...
...(IT2(:,i-1)+((h/2)*T*U0)))+((h/2)*T*U0);
ITS(:,i)=(expm(h*T)*(ITS(:,i-1)+((h/2)*nonlin(EP,ITS(1,i-1))*...
...IT2(:,i-1))))+((h/2)*nonlin(EP,IT2(1,i))*IT2(:,i));
U0=ITS(:,i); IT2=ITS; Tr2(i)=ITS(1,i);
end
```

```
%%%%%%%HAMILTONIAN of PROPOSED SPLITTING
%%%H(q_1,q_2)=-\frac{q_2^2}{2}-\frac{q_1^2}{2}-\frac{epsilon q_1^4}{4}
%%% initialization
H=zeros(length(t),1); EH=zeros(length(t),1);
H(1)=-0.5-(EP*(1/4)); %%%% initial for Hamiltonian
for i=1:length(t)-1
H(i+1)=-(((IT2(2,i))^2)/2)-(((IT2(1,i))^2)/2)-...
...(EP*((((IT2(1,i))^4)/4))); EH(i+1)=abs(H(i+1)-H(1));
end
%%%%%%%%%%% STRANG SPLITTING
SM=zeros(2,n+1); Smss=zeros(1,n+1);
SM(:,1)=P;  Smss(1)=q10; %%%%initialization
b=expm(T*h/2);
%Main Loop
for i=2:n+1
    a1=b*SM(:,i-1);
    a2=expm((h/2)*(nonlin(EP,SM(1,i-1))+nonlin(EP,a1(1,1))));
    SM(:,i)=b*a2*a1;
    Smss(i)=SM(1,i);
end

%%%%%%%HAMILTONIAN of STRANG SPLITTING
HS=zeros(length(t),1); EHS=zeros(length(t),1);
HS(1)=-0.5-(EP*(1/4)); for i=1:length(t)-1
HS(i+1)=-(((SM(2,i))^2)/2)-(((SM(1,i))^2)/2)-...
...(EP*((((SM(1,i))^4)/4))); EHS(i+1)=abs(HS(i+1)-HS(1));
end

%%%%%%%%%%% SYMMETRICALLY WEIGHTED SPLITTING
SWS=zeros(2,n+1); swsss=zeros(1,n+1);
SWS(:,1)=Q;  swsss(1)=q10;
c=expm(T*h);
for i=2:n+1 b3=c*SWS(:,i-1);
       ara=expm((h/2)*(nonlin(EP,SWS(1,i-1))+nonlin(EP,b3(1,1))));
       symet1=ara*b3;
    a3=expm(h*nonlin(EP,SWS(1,i-1)))*SWS(:,i-1);
    symet2=c*a3;
```

```
     SWS(:,i)=0.5*(symet1+symet2);
    swsss(i)=SWS(1,i);
end

%%%%%%%HAMILTONIAN of STRANG SPLITTING
HSW=zeros(length(t),1); EHSW=zeros(length(t),1);
HSW(1)=-0.5-(EP*(1/4));

for i=1:length(t)-1
HSW(i+1)=-(((SWS(2,i))^2)/2)-(((SWS(1,i))^2)/2)-...
...(EP*((((SWS(1,i))^4)/4))); EHSW(i+1)=abs(HSW(i+1)-HSW(1));
end

%%%%%%%  ERRORS Comparison with the Analtic Solution
%%%Here it must choosen as \varepsilon is small
%%% because of the analytic one.
%%%%for choosen $h$
 ERIT=max(abs(Tr2-EX))
 ERIT2=norm((Tr2-EX),2)
 ERIT1=norm((Tr2-EX),1)
 ERST=max(abs(Smss-EX))
 ERST2=norm((Smss-EX),2)
 ERST1=norm((Smss-EX),1)
 ERSWS=max(abs(swsss-EX))
 ERSWS2=norm((swsss-EX),2)
 ERSWS1=norm((swsss-EX),1)
%%%%%%FIGURES

figure(1)
plot(t,EH)
legend('Hamiltonian of Proposed Scheme')
 xlabel('time')
 ylabel('|H(q_1,q_2)-H(q_1^0,q_2^0)|')

figure(2)
plot(t,EHS) legend('Hamiltonian of Strang Splitting')
 xlabel('time') ylabel('|H(q_1,q_2)-H(q_1^0,q_2^0)|')
```

THE PROGRAMME WHICH IS USED IN SECOND ORDER METHODS

```
%%% nonlinear part of Duffing Equation
%%% for Proposed and Strang Splitting methods
 function  y=nonlin(EP,yi)
K=zeros(2);
K(2,1)=-EP*(yi^2);
y=K;
```

D.2. Codes for Van-der Pol Equation

THE SECOND ORDER PROPOSED METHOD vs. ODE23s

```
%%%%% Van-der Pol eq: To see conservation of limit cycle
%%%%% Equation:
close all; clear all; clc;
% nu=5:5:25; %%%% for sketching the solution with different nu(s)
% cc = jet(length(nu));
%  for s=1:length(nu)
dt=0.01; tf=60; t0=0;

t=t0:dt:tf; nx=length(t); solution=zeros(2,nx); ce=zeros(2,1);
%initial conditions
coeff=10;   %%%% uncomment nu, then write coeff=nu(s)
ce(1)=2; ce(2)=0; solution(:,1)=ce;
%%%% Initialization
U1=zeros(2,nx); U2=zeros(2,nx);
%%% Programme
A=[0,1;-1,0];

U1(:,1)=ce; U2(:,1)=ce;

for k=1:length(t)-1
U1(:,k+1)=expm(A*dt)*...
...(U1(:,k)+((dt/2)*nonlinear(coeff,U1(1,k),U1(:,k))))+...
...((dt/2)*nonlinear(coeff,U1(1,k),U1(:,k)));
newvar=nonlinear(coeff,U2(1,k),1)+nonlinear(coeff,U1(1,k+1),1);
```

```
U2(:,k+1)=expm(newvar*(dt/2))*(U2(:,k)+((dt/2)*A*U2(:,k)))+...
...((dt/2)*A*U1(:,k+1)); U1=U2; solution(:,k+1)=U2(:,k+1);
end

%%%%%%%% Hamiltonian
H=zeros(length(t),1); EH=zeros(length(t),1); H(1)=-2;

for i=1:length(t)-1
H(i+1)=-(((solution(2,i))^2)/2)-(((solution(1,i))^2)/2)-...
...(coeff*((((solution(1,i))^3)/3)-solution(1,i))*solution(2,i));
EH(i+1)=abs(H(i+1)-H(1));
end

%%%%%% Solution for ODE23s
[T,Y] = ode23s(@(t,y)ypvdpol(coeff,t,y),[t0 tf],[2 0]);

%%%%%Hamiltonian of ODE23s
H1(1)=-2; for i=1:length(T)-1
H1(i+1)=-(((Y(i,2))^2)/2)-(((Y(i,1))^2)/2)-...
...(coeff*((((Y(i,1))^3)/3)-Y(i,1))*Y(i,2));
EH1(i+1)=abs(H1(i+1)-H1(1));
end

%%%% Figures
figure(1) plot(t,EH)
legend('Hamiltonian of Proposed Scheme')
xlabel('time') ylabel('|H(x_1,x_2)-H(x_1^0,x_2^0)|')

figure(2) plot(solution(2,:),solution(1,:),'r')

title('Limit Cycle of Proposed Method') xlabel('x_2') ylabel('x_1')
figure(3)
plot(T,EH1)
legend('Hamiltonian of ode23s')
xlabel('time')
ylabel('|H(x_1,x_2)-H(x_1^0,x_2^0)|')
```

```
figure(4) plot(Y(:,2),Y(:,1),'r')
title('Limit Cycle of ode23s')
xlabel('x_2') ylabel('x_1')

figure(5) plot(t,solution(1,:),'b',T,Y(:,1),'r')
legend('proposed','ode23s')
```

THE PROGRAMME WHICH IS USED IN THE PROGRAMME ABOVE

```
 %nonlinear part of van Der Pol
function y=nonlinear(coeff,a,b) K=zeros(2,2);
K(2,2)=-coeff*((a^2)-1); K=K*b; y=K;

%% full problem for van-der Pol---- ODE23s
function ypvdpol=ypvdpol(nu,t,y)

ypvdpol(1) = y(2);
ypvdpol(2)=nu*(1-y(1)^2)*y(2)-y(1);
ypvdpol=[ypvdpol(1) ypvdpol(2)]';
```

D.3. Codes for Nonlinear Schrödinger Solutions

D.3.1. Equation in the form: $i\partial_t\Psi = \beta\partial_x^2\Psi + \alpha|\Psi|^2\Psi$

MAIN PROGRAMME for CASE 1

```
%%%% Soliton Solution $iu_t+u_{xx}+|u|^2u=0$
%%%% with initial condition $u(x,0)=2sech(x)e^{ix}$
%%%%%%%%%%Problem definition
clear all; close all; clc;

t0=0; tson=1; N=100; h=(tson-t0)/N; x0=-15.0; xson=10;
dx=(xson-x0)/N; cfl=h/dx
t=t0:h:tson ; x=x0:dx:xson;

%% Construction initial condition
```

```
C0=zeros(2*N+2,1);

for i=1:N+1
    C0(i)=2*sech(x(i)*sqrt(2))*cos(x(i));
    C0(N+1+i)=2*sech(x(i)*sqrt(2))*sin(x(i));
end
%%%%%%%%%%%%%%%%%%% Initial Condition %%%%%%%%%%%%%%%%%
yini=C0;
%%%%%%%%%% Boundary Conditions: Dirichlet %%%%%%%%%%%%%
yini(1)=0; yini(end)=0; yini(N+1)=0; yini(N+2)=0;

yini1=yini; %Initial for Strang
 yini2=yini;

%%%%%%% Exact solution from (Polyanin and Zaitsev, 2004)
exact=zeros(2*N+2,N+1);
for q=1:N+1 %%timecounter for exact
    for r=1:N+1 %%xcounter for exact
       exact(r,q)=2*cos(x(r)+t(q))*sech(sqrt(2)*x(r)-(2*sqrt(2)*t(q)));
       exact(N+1+r,q)=2*sin(x(r)+t(q)).*sech(sqrt(2)*x(r)-...
       ...(2*sqrt(2)*t(q)));
    end
end

%%%%%%%% Numerical Solutions
%%%%%% Linear operator

%%%%% central Difference
n=N; UU=toeplitz([-2 1 0 0    zeros(1,n-3)],[-2 1 0 0
zeros(1,n-3)]); US=(1/((dx)^2))*full(UU); BB=US;

T=[zeros(N+1) -BB;BB zeros(N+1)];
%%%%%%%%% PROPOSED METHOD
%%%% Construction of nonlinear matrix
 KT=zeros(N+1,N+1);   KT1=zeros(N+1,N+1);
 GG=expm(T*h);
 IT1(:,1)=yini1;
 ITS(:,1)=yini1;
for j=2:N+1 %%time counter
 for i=1:N+1 %%% x counter
   KT(i,i)=nonl(yini(i),yini(N+1+i));
 end
 KTT=[zeros(N+1) KT;-KT zeros(N+1)];
 IT1(:,j)=GG*(IT1(:,j-1)+((h/2)*KTT* IT1(:,j-1)))+...
... ((h/2)*KTT* IT1(:,j-1));
 next=IT1(:,j);
for i=1:N+1
  KT1(i,i)=nonl(next(i),next(N+1+i));
end
 KT11=[zeros(N+1) KT1;-KT1 zeros(N+1)];
 JJ=expm((h/2)*((KTT+KT11)));
  ITS(:,j)=JJ*(ITS(:,j-1)+((h/2)*T*IT1(:,j-1)))+((h/2)*T*(IT1(:,j)));
 %%%Boundary conditions
  ITS(1,j)=0; ITS(N+1,j)=0; ITS(N+2,j)=0;  ITS(end,j)=0;
%%%Boundary conditions
  IT1=ITS;
  yini=ITS(:,j);  %%%%% ITS(x,t) form
end

 %%%%% Global Energy for PROPOSED METHOD
 %%%%% Inspired by (Aydin and Karsozen,2011)

for j=1:N+1 %%t counter
top=0;
for i=2:N %%x counter
     der=((((ITS(i+1,j)-ITS(i-1,j))/(2*dx)).^2)+...
     ...(((ITS(N+1+i+1,j)-ITS(N+1+i-1,j))/(2*dx)).^2));
     energy0=((((-1/4)*((2*sech(x(i)*sqrt(2))).^4))+...
     ...((1/2)*(((2*sqrt(2)*tanh(sqrt(2)*x(i))*...
     ...sech(sqrt(2)*x(i))).^2)+(sech(sqrt(2)*x(i)).^2))));
    ham=((-(1/4)*((((ITS(i,j).^2))+((ITS(N+1+i,j)).^2))).^2)+...
   ...(0.5*der)-energy0);
    top=top+ham;
end
```

```
 GE(j)=dx*top;
end

%%%%%%% Absolute Error conservation of the mass
M0=((ITS(1:N+1,1).^2)+(ITS(N+2:end,1).^2)); top2m0=0; for i=1:N+1
top2m0=top2m0+M0(i); end
for j=1:N+1 %%t counter
top1=0;
for i=1:N+1 %%x counter
    pot=(ITS(i,j).^2)+(ITS(N+1+i,j).^2);
    top1=top1+pot;
end
 AET(j)=dx*top1;
 AE(j)=abs(AET(j)-(dx*top2m0));
end

%%%%% ERROS OF ABSOLUTE ENERGY
aberr=norm(AE,inf)
aberr2=norm(AE,2)
aberr1=norm(AE,1)

%%%%%%%%%%%% Strang Splitting
MT=zeros(N+1,N+1);   MT1=zeros(N+1,N+1);
 GGG=expm(T*h/2);
 ST(:,1)=yini2;
 newini=yini2;
 for j=2:N+1
     for k=1:N+1
   MT1(k,k)=nonl2(yini2(k),yini2(N+1+k));
 end
 MT11=[zeros(N+1) MT1;-MT1 zeros(N+1)];
      newini=GGG*ST(:,j-1);
 for i=1:N+1
   MT(i,i)=nonl(newini(i),newini(N+1+i));
 end
 MTT=[zeros(N+1) MT;-MT zeros(N+1)];
 ST(:,j)=GGG*expm((MTT+MT11)*(h/2))*GGG*ST(:,j-1);
```

```
  ST(1,j)=0; ST(N+1,j)=0; ST(N+2,j)=0; ST(end,j)=0;
%%%Boundary conditions
 newini=ST(:,j);
 end

%%%%% GLOBAL ENERGY of STRANG SPLITTING
 for j=1:N+1 %%t counter
top=0;
for i=2:N %%x counter
     der=((((ST(i+1,j)-ST(i-1,j))/(2*dx)).^2)+...
     ...(((ST(N+1+i+1,j)-ST(N+1+i-1,j))/(2*dx)).^2));
     energy0=((((-1/4)*((2*sech(x(i)*sqrt(2))).^4))+...
     ...((1/2)*(((2*sqrt(2)*tanh(sqrt(2)*x(i))*sech(sqrt(2)*x(i))).^2)+
     ...(sech(sqrt(2)*x(i)).^2))));
    ham=((-(1/4)*((((ST(i,j).^2))+((ST(N+1+i,j)).^2))).^2)+...
   ...(0.5*der)-energy0);
    top=top+ham;
end
 GEST(j)=dx*top;
end

for i=1:N+1 %%% ENEIT(t,x)
   ENEIT(:,i)= sqrt(((ITS(i,:).^2 +ITS(N+1+i,:).^2)));
end

for i=1:N+1  ENEEXA(t,x)
   ENEEXA(:,i)= sqrt(((exact(i,:).^2 +exact(N+1+i,:).^2)));
end
ENEEXA(:,1)=0; ENEEXA(:,N+1)=0;  %%due to the Boundary condtions

for i=1:N+1
   ENEST(:,i)= sqrt((ST(i,:).^2 +ST(N+1+i,:).^2));
end



%%%% ENERGY CONSERVATION
 energybasit=dx*(norm(ENEIT(1,:),2)^2)
 energysonit=dx*(norm(ENEIT(end,:),2)^2)
 energysonst=dx*(norm(ENEST(end,:),2)^2)
 energybasexa=dx*(norm(ENEEXA(1,:),2)^2 )
  energysonexa=dx*(norm(ENEEXA(end,:),2)^2)
 figure(1)
 [X,Y] = meshgrid(x0:dx:xson,t0:h:tson);
 Z = ENEIT;
 surf(X,Y,Z)
 xlabel('x')
 ylabel('t')
 zlabel('|\psi|')
 title('Iterative Solution')
 axis([x0 xson t0 tson 0 2.5])

 figure(2)
 [X,Y] = meshgrid(x0:dx:xson,t0:h:tson );
 Z = ENEEXA;
 surf(X,Y,Z)
 xlabel('x')
 ylabel('t')
 zlabel('|\psi|')
 axis([x0 xson t0 tson 0 2.5])
 title('Exact Solution')

 figure(3)
 [X,Y] = meshgrid(x0:dx:xson,t0:h:tson );
 Z = ENEST;
 surf(X,Y,Z)
 xlabel('x')
 ylabel('t')
 zlabel('|\psi|')
 axis([x0 xson t0 tson 0 2.5])
 title('Exact Solution')

figure(4) plot(t,GE) title('Global Energy Error')

figure(5) plot(t,AE)
figure(6) plot(t,GEST)
```

THE PROGRAMME WHICH IS USED IN MAIN PROGRAMME for CASE 1

```
%nonlinear for 1soliton solution
function y=nonl(u,v)
     K=-((u^2)+(v^2));
 y=K;
```

MAIN PROGRAMME for CASE 2

```
%%%% Soliton Solution iu_t+u_{xx}+$|u|^2$u=0
%%%% with initial condition u(x,0)=sqrt(2)sech(x+10)e^{i0.25*x}
%%%% Corresponding to CASE 2
%%%%%%%%%Problem definition
clear all; close all; clc;

t0=0; tson=3; N=100; h=(tson-t0)/N;
x0=-15.0; xson=5;
dx=(xson-x0)/N;
t=t0:h:tson ; x=x0:dx:xson;

%% Construction initial condition for CASE 2
C0=zeros(2*N+2,1);

for i=1:N+1
    C0(i)=sqrt(2)*sech(x(i)+10)*cos((0.25)*x(i));
    C0(N+1+i)=sqrt(2)*sech(x(i)+10)*sin((0.25)*x(i));
end

%%%%%%%%%%%%%%%%%%%%%%%%%%Initial Condition%%%%%%%%%%%%%%%%%%%%%%%%%%
yini=C0;
%%%%%%%%%%%%%%%%%%%%% Boundary Conditions%%%%%%%%%%%%%%%%%%%%%%%%%
 yini(1)=0; yini(end)=0; yini(N+1)=0; yini(N+2)=0;
yini1=yini; %%%% using for Strang

%% Exact solution from (Polyanin and Zaitsev,2004)

exact2=zeros(2*N+2,N+1);
```

```
for q=1:N+1 %%timecounter for exact
    for r=1:N+1 %%xcounter for exact
       exact2(r,q)=sqrt(2)*cos((0.25*x(r))+((1-(1/16))*t(q)))*...
       ...sech(x(r)-(0.5*t(q))+10);
       exact2(N+1+r,q)=sqrt(2)*sin((0.25*x(r))+((1-(1/16))*t(q)))*...
       ...sech(x(r)-(0.5*t(q))+10);
    end
end
%% Constant part %%% different space discretization
%%%%% central difference
n=N; UU=toeplitz([-2 1 0 0 zeros(1,n-3)],[-2 1 0 0 zeros(1,n-3)]);
US=(1/((dx)^2))*full(UU);
BB=US;
T=[zeros(N+1) -BB;BB zeros(N+1)];
%%%%%%%%%%%%%%%%%% PROPOSED METHOD %%%%%%%
 KT=zeros(N+1,N+1);
 KT1=zeros(N+1,N+1);
 GG=expm(T*h);
 IT1(:,1)=yini1;
 ITS(:,1)=yini1;
for j=2:N+1
for i=1:N+1
   KT(i,i)=nonl2(yini(i),yini(N+1+i));
end
KTT=[zeros(N+1) KT;-KT zeros(N+1)];
IT1(:,j)=GG*(IT1(:,j-1)+((h/2)*KTT* IT1(:,j-1))) +...
...((h/2)*KTT* IT1(:,j-1));
next=IT1(:,j);
%%%% construction for the second order proposed meth.
for i=1:N+1
  KT1(i,i)=nonl2(next(i),next(N+1+i));
end
 KT11=[zeros(N+1) KT1;-KT1 zeros(N+1)];
 JJ=expm((h/2)*((KTT+KT11)));
ITS(:,j)=JJ*(ITS(:,j-1)+((h/2)*T*IT1(:,j-1)))+((h/2)*T*(IT1(:,j)));
%%%Boundary conditions
ITS(1,j)=0; ITS(N+1,j)=0; ITS(N+2,j)=0; ITS(end,j)=0;
```

```
IT1=ITS; yini=ITS(:,j);
end
%%%%%% Global Energy Error only Proposed Method
for j=1:N+1 %%t counter
top=0;
for i=2:N %%x counter
     der=((((ITS(i+1,j)-ITS(i-1,j))/(2*dx)).^2)+...
     ...(((ITS(N+1+i+1,j)-ITS(N+1+i-1,j))/(2*dx)).^2));
     energy0=((((-1/4)*(sqrt(2)*sech(x(i)+10)).^4)))+...
     ...((1/2)*(((sqrt(2)*tanh(x(i)+10)*sech(x(i)+10)).^2)+...
     ...(((sqrt(2)/4)*sech(x(i)+10)).^2)));
    ham=((-(1/4)*((((ITS(i,j).^2))+((ITS(N+1+i,j)).^2))).^2)+...
   ...(0.5*der)-energy0);
    top=top+ham;
end
 GE(j)=dx*top;
end

%%%%%% Absolute Error conservation of the mass only for Proposed Method
M0=((ITS(1:N+1,1).^2)+(ITS(N+2:end,1).^2)); top2m0=0; for i=1:N+1
top2m0=top2m0+M0(i); end
for j=1:N+1 %%t counter
top1=0;
 for i=1:N+1 %%x counter
    pot=(ITS(i,j).^2)+(ITS(N+1+i,j).^2);
    top1=top1+pot;
 end
 AET(j)=dx*top1;
 AE(j)=abs(AET(j)-(dx*top2m0));
end

%%%%%%%STRANG SPLITTING
MT=zeros(N+1,N+1); MT1=zeros(N+1,N+1); GGG=expm(T*h/2);
 ST(:,1)=yini1;
 newini=yini1;
 for j=2:N+1
  for k=1:N+1
```

```
   MT1(k,k)=nonl2(yini2(k),yini2(N+1+k));
  end
  MT11=[zeros(N+1) MT1;-MT1 zeros(N+1)];
  newini=GG*ST(:,j-1);
 for i=1:N+1
   MT(i,i)=nonl2(newini(i),newini(N+1+i));
 end
 MTT=[zeros(N+1) MT;-MT zeros(N+1)];
 ST(:,j)=GGG*expm(MTT*h)*GGG*ST(:,j-1);
   ST(1,j)=0; ST(N+1,j)=0;
   ST(N+2,j)=0; ST(end,j)=0; %%%Boundary conditions
 newini=ST(:,j);
 end

%%%%%%%% Solutions |u|

for i=1:N+1 %%%
   ENEIT(:,i)= sqrt(((ITS(i,:).^2 +ITS(N+1+i,:).^2)));   %%%%proposed
end

 for i=1:N+1
   ENEST(:,i)=sqrt((ST(i,:).^2 +ST(N+1+i,:).^2));      %%%%%strang
 end

for i=1:N+1
   ENEEXA(:,i)= sqrt(((exact2(i,:).^2 +exact2(N+1+i,:).^2))); %% exact
end

%%%% CONSERVATION of ENERGY

energybasit=dx*(norm(ENEIT(1,:),2)^2)
energysonit=dx*(norm(ENEIT(end,:),2)^2)
energysonst=dx*(norm(ENEST(end,:),2)^2)
energybasexa=dx*(norm(ENEEXA(1,:),2)^2)
energysonexa=dx*(norm(ENEEXA(end,:),2)^2)

%%%% ERRORS for Absolute Energy
```

```
errorAE=norm(AE,inf)
errorAE2=norm(AE,2)
errorAE1=norm(AE,1)

%%% FIGURES

figure(1)%% due to the soliton solution, we plot x,t
[X,Y] = meshgrid(x0:dx:xson,t0:h:tson);
Z=ENEIT; surf(X,Y,Z)
xlabel(’x’)
ylabel(’t’)
zlabel(’|\psi|’)
title(’Proposed Method’)

figure(2)
[X,Y] = meshgrid(x0:dx:xson,t0:h:tson );
Z = ENEEXA;
surf(X,Y,Z)
 xlabel(’x’)
 ylabel(’t’)
 zlabel(’|\psi|’)
title(’Exact Solution’)

figure(3)
[X,Y] = meshgrid(x0:dx:xson,t0:h:tson);
Z = ENEST;
surf(X,Y,Z)
xlabel(’x’)
ylabel(’t’)
zlabel(’|\psi|’)

title(’Strang Splitting’)

figure(4) plot(t,AE)

title(’Absolute Energy’)
```

```
figure(5) plot(t,GE)

title('Global Energy of Proposed Method')
```

THE PROGRAMME WHICH IS USED IN MAIN PROGRAMME for CASE 2

```
%%%%%%nonlinear for 2soliton solution
function y=nonl2(u,v)
     K=-((u^2)+(v^2));
 y=K;
```

D.3.2. Equation in the form: $i\partial_t\Psi = \beta\partial_x^2\Psi + (V(x) + \alpha|\Psi|^2)\Psi$

MAIN PROGRAMME

```
%%%%%Schrodinger equation with non-linear potential
%%%% iu_t=-0.5u_xx+(V(x)+\lambda |u|^2)u
%%Problem definition
clear all1; close all; clc;

t0=0; tson=1; N=100; h=(tson-t0)/N;
x0=-20; xson=20; dx=(xson-x0)/N;
t=t0:h:tson ; x=x0:dx:xson;
%% Construction initial condition
C0=zeros(N+1,1); for i=1:N+1
    C0(i)=exp(sin(2*x(i)));
end
%%%%%%%%%%%%%%Initial Condition%%%%%%%%%%%%
yini=[C0;zeros(N+1,1)];

yini=yini/(norm(yini,2));  %%%% normalization

%%% Boundary Conditions
yini(1)=0; yini(end)=0; yini(N+1)=0; yini(N+2)=0; yini1=yini;
%% Constant part
%%%%%%% Central Difference
n=N;
UU=toeplitz([-2 1 0 0    zeros(1,n-3)],[-2 1 0 0
zeros(1,n-3)]); US=(1/((dx)^2))*full(UU);
BB=US;

T=[zeros(N+1) -0.5*BB;0.5*BB zeros(N+1)];

%%%%% Proposed Method
 KT=zeros(N+1,N+1);   KT1=zeros(N+1,N+1);
 GG=expm(T*h);
 IT1(:,1)=yini1;
 ITS(:,1)=yini1;
 starttimeit=cputime;
for j=2:N+1
 for i=1:N+1
   KT(i,i)=nl(x(i),yini(i),yini(N+1+i));
 end
 KTT=[zeros(N+1) KT;-KT zeros(N+1)];
 IT1(:,j)=GG*(IT1(:,j-1)+((h/2)*KTT* IT1(:,j-1))) +...
...((h/2)*KTT* IT1(:,j-1));
 next=IT1(:,j);
for i=1:N+1
  KT1(i,i)=nl(x(i),next(i),next(N+1+i));
end
 KT11=[zeros(N+1) KT1;-KT1 zeros(N+1)];
 JJ=expm((h/2)*((KTT+KT11)));
  ITS(:,j)=JJ*(ITS(:,j-1)+((h/2)*T*IT1(:,j-1)))+((h/2)*T*(IT1(:,j)));
  %%%Boundary conditions
  ITS(1,j)=0; ITS(N+1,j)=0; ITS(N+2,j)=0; ITS(end,j)=0;
 IT1=ITS;
 yini=ITS(:,j);
end
 ITS;
elapsedit=cputime-starttimeit

for i=1:N+1 %%%ENEIT(t,x)
   ENEIT(:,i)= ((ITS(i,:).^2 +ITS(N+1+i,:).^2));
   SOLIT(:,i)= sqrt((ITS(i,:).^2 +ITS(N+1+i,:).^2));
end
```

```
%% Strang Splitting
 MT=zeros(N+1,N+1);   MT1=zeros(N+1,N+1);
 GGG=expm(T*h/2);
 ST(:,1)=yini1;
 newini=yini1;

 starttimest=cputime; %%% for elapsed time of PISM

 for j=2:N+1
       for k=1:N+1
   MT1(k,k)=nl(x(k),yini2(k),yini2(N+1+k));
 end
 MT11=[zeros(N+1) MT1;-MT1 zeros(N+1)];
 newini=GG*ST(:,j-1);
 for i=1:N+1
   MT(i,i)=nl(x(i),newini(i),newini(N+1+i));
 end
 MTT=[zeros(N+1) MT;-MT zeros(N+1)];
 ST(:,j)=GGG*expm((MTT+MT11)*(h/2))*GGG*ST(:,j-1);
 ST(1,j)=0; ST(N+1,j)=0; ST(N+2,j)=0; ST(end,j)=0;
 yini2=ST(:,j);
 end
 ST; elapsedstr=cputime-starttimest %%% to see the elapsed time

for i=1:N+1 %%ENEST(t,x)
   ENEST(:,i)= ((ST(i,:).^2 +ST(N+1+i,:).^2));
   STSOL(:,i)=sqrt((ST(i,:).^2 +ST(N+1+i,:).^2));
end
energybasit=(dx)*(norm(SOLIT(1,:),2)^2)
energysonit=(dx)*(norm(SOLIT(end,:),2)^2)
energybasst=(dx)*(norm(STSOL(1,:),2)^2)
energysonst=(dx)*(norm(STSOL(end,:),2)^2)
figure(1)
[X,Y] = meshgrid(t0:h:tson,x0:dx:xson);
Z = ENEIT;
surf(X,Y,Z)
```

```
xlabel('t')
ylabel('x')
zlabel('probability density of the PISM'
figure(2)
[X,Y] = meshgrid(t0:h:tson,x0:dx:xson);
Z = ENEST;
surf(X,Y,Z)
xlabel('t')
ylabel('x')
zlabel('probability density of Strang splitting')
```

APPENDIX E

MAT-LAB CODES FOR FRÉCHET TECHNIQUE

E.1. Codes for Damped oscillator

MAIN PROGRAMME

```
%%%% Frechet Derivatives  for Damped oscilatory
clear all; close all; clc;
dt=0.004; tson=0.4; t=0:dt:tson;
nx=length(t);
%%%% initialiing
EX=zeros(nx,1); temp=zeros(2,1); teta=zeros(2,1);
solution=zeros(2,nx); ce=zeros(2,1);

itermax=1000;
alpha=1e-4;  %%%% for using analytical solution
%  alpha=1; %%% for ode45

%Initial conditions
ce(1)=1; ce(2)=0; solution(:,1)=ce; cy=[1;0]; P=ce;
temp(1)=0;temp(2)=0; teta=ce;

%% Exact solution
  exact=cos(t)./(sqrt(1+(3*alpha*t/4)));
  EX=exact';

 %% 2nd order Runge-kutta
starttimerk=cputime;
rk(:,1)=P;
for i=1:nx-1
    k1=nonlindampedrunge(alpha,rk(:,i));
    k2=nonlindampedrunge(alpha,rk(:,i)+(0.5*dt*k1));
   rk(:,i+1)=rk(:,i)+dt*k2; %%2nd order
end
```

```
rk; elapsedtimerk = cputime-starttimerk

%%%% Proposed Method
B=[0,-1;1,0];
starttimeit=cputime;

for k=1:length(t)-1
for iter=1:itermax
    A=[0,0; 0, 3*alpha*((((ce(2)+cy(2))/2)^2))];
    C=[0;alpha*(((cy(2)+ce(2))/2).^3)];
    amat=((1/dt)*eye(size(B)))+(B/2)+(0.5*A);
    bmat1=((-(1/dt)*eye(size(B)))-((0.5)*B))*cy;
    bmat2=((1/dt)*eye(size(B))-((0.5)*B))*ce;
    bmat=bmat1+bmat2-C;
    teta=amat\bmat;
    temp=cy+teta;
    err=norm(temp-cy,2);
    cy=temp;
    if err<1e-6
        break
    end
end ce=cy; solution(:,k+1)=ce;
end

solution; elapsedtimeit = cputime-starttimeit

%%%%%% ERRORS
 %%%%%%%%%Proposed Method
 errFr=norm((exact-solution(1,:)),inf)
 errFr22=norm((exact-solution(1,:)),2)
 errFr21=norm((exact-solution(1,:)),1)
 %%%%%% Runge Kutta
  errRK=norm((exact-rk(1,:)),inf)
  errRK22=norm((exact-rk(1,:)),2)
  errRK21=norm((exact-rk(1,:)),1)

%%%% For non-exact solution
```

```
[T,Y] = ode45(@(t,y)damped(alpha,t,y),[0 tson],[1 0]);

%%%%% FIGURES
figure(1)
 plot(t,EX,’-’,t,solution(1,:),’-o’)
 legend(’Exact’,’Numerical’)
 % title(’Exact(--) vs Proposed Method(*)’)
 xlabel(’time’)
 ylabel(’displacement’)
figure(2)
plot(T,Y(:,1),’ro’,t,solution(1,:),’-*’,t,rk(1,:),’-o’)
legend(’ode45’,’Proposed method’,’runge kutta’)
```

THE PROGRAMME WHICH IS USED IN ALGORITHM ABOVE

```
function  y=nonlindampedrunge(nu,v)
 K=zeros(2);
 K=[0, -1;1, nu*(v(2,1)^2)]*v;
 y=K;
```

E.2. Codes for Van-der Pol Equation

MAIN PROGRAMME

```
close all
clear all;
clc;
% nu=5:5:25; %uncommet this line then change coeff=nu(s)
nu=5;
% cc = jet(length(nu));
% for s=1:length(nu) %%%% uncomment when nu=nu(s)
dt=0.01;
tf=60;
t=0:dt:tf;
nx=length(t); %%% time steps

%%%%%% Initialization
temp=zeros(2,1);
```

```
teta=zeros(2,1);
solution=zeros(2,nx); ce=zeros(2,1);
itermax=1000;

%%%%% Initial Condititons
coeff=nu; %nu(s)
ce(1)=2; ce(2)=0;
cy=[0;0];
solution(:,1)=ce; temp(1)=0;temp(2)=0;
teta(1)=ce(1); teta(2)=ce(2);

%%%%%%% PROPOSED METHOD
B=[0,-1;1,0];

starttimefre=cputime;
for k=1:length(t)-1
for iter=1:itermax
    A=[0,0; ...
    ...2*coeff*((ce(1)+cy(1))/2)*((cy(2)+ce(2))/2),...
    ...coeff*((((ce(1)+cy(1))/2)^2)-1)];
    RS1=[0,0;0, coeff*((cy(1)^2)-1)]*cy;
    RS2=[0,0;0, coeff*((ce(1)^2)-1)]*ce;
    C=(RS1+RS2)/2;
    amat=(((1/dt)*eye(size(B)))+((0.5)*B)+((0.5)*A));
    bmat1=(((-1/dt)*eye(size(B)))-((0.5)*B))*cy;
    bmat2=(((1/dt)*eye(size(B)))-((0.5)*B))*ce;
    bmat=bmat1+bmat2-C;
     teta=amat\bmat;
    temp=cy+teta;
    err=norm(temp-cy,2);
    cy=temp;
    if err<1e-6
        break
    end
end

ce=cy; solution(:,k+1)=ce;
```

```
end
 %%%% elapsed time for proposed method
elapsedtimefre=cputime-starttimefre

starttime23=cputime;

[T,Y] = ode23s(@(t,y)ypvdpol(coeff,t,y),[0 tf],[2 0]);
%%%%% elapsed time for obtaining solution from the ODE23s
elapsedtime23=cputime-starttime23

%%%%%% FIGURES
figure(2)
plot(T,Y(:,1),'-o',t,solution(1,:),'-*')
legend('ODE23s','Proposed Method')
%   figure(s)
%  plot(t,solution(1,:),'color',rand(1,3))
%  legend(num2str(nu(s)))
xlabel('time')
ylabel('x(t)')

figure(3)
plot(Y(:,1),Y(:,2),'r') hold on
plot(solution(1,:),solution(2,:),'-*')

legend('ODE23s','Proposed Method')
```

E.3. Codes for Schrödinger Equation

E.3.1. Equation in the form: $i\partial_t\Psi + \beta\partial_x^2\Psi + \alpha|\Psi|^2\Psi = 0$

MAIN PROGRAMME for CASE 1

```
%%%%%Schrodinger equation with non-linear potential for Case1
%%Problem definition
```

```
clear all;
close all;
clc;
t0=0; tson=1; N=100; h=(tson-t0)/N; %t=0.1
x0=-15.0; xson=10; dx=(xson-x0)/N;
cfl=h/dx
t=t0:h:tson ;
x=x0:dx:xson;
%% Construction initial condition
C0=zeros(2*N+2,1);
for i=1:N+1
    C0(i)=2*sech(x(i)*sqrt(2))*cos(x(i));
    C0(N+1+i)=2*sech(x(i)*sqrt(2))*sin(x(i));
end
%%%%%%%%%%%%%%%%%%%Initial Condition%%%%%%%%%%%%%%%%
yini=C0;
%%%% boundary conditions
yini(1)=0;  yini(end)=0;  yini(N+1)=0; yini(N+2)=0;
yini1=yini;
yini2=yini;
%% Exact solution from polyanin
exact=zeros(2*N+2,N+1);
for q=1:N+1 %%timecounter for exact
    for r=1:N+1 %%xcounter for exact
       exact(r,q)=2*cos(x(r)+t(q))*sech(sqrt(2)*x(r)-(2*sqrt(2)*t(q)));
       exact(N+1+r,q)=2*sin(x(r)+t(q)).*...
       ...sech(sqrt(2)*x(r)-(2*sqrt(2)*t(q)));
    end
end
 %% Constant part
n=N;
UU=toeplitz([-2 1 0 0    zeros(1,n-3)],[-2 1 0 0  zeros(1,n-3)]);
US=(1/((dx)^2))*full(UU);
BB=US;
T=[zeros(N+1) -BB;BB zeros(N+1)];

%%%% given boundary conditions
```

```
exact(1,:)=0; exact(M+1,:)=0; exact(M+2,:)=0; exact(end,:)=0;

%%%%%%%%% Numerical Solution
A=toeplitz([-2,1,0,0, zeros(1,M-3)],[-2,1,0,0, zeros(1,M-3)]);
b=(1/(dx^2))*full(A);
BA=[zeros(M+1,M+1), b; -b, zeros(M+1,M+1)];

%% Main loop
sol=zeros(2*M+2,N+1);
sol(:,1)=ce;
 for j=1:N
     ce(1)=0;ce(2*M+2)=0;
    ce(M+1)=0; ce(M+2)=0;
    cy(1)=0; cy(2*M+2)=0;
    cy(M+1)=0; cy(M+2)=0;
    for k=1:itermax
    % left side
    cvec11=2*(((ce(1:M+1)+cy(1:M+1))/2).*((ce(M+2:end)+...
    ...cy(M+2:end))/2)); %%2lambdau1u2
    cvec12=((((ce(1:M+1)+cy(1:M+1))/2).^2)+(3*(((ce(M+2:end)+...
    ...cy(M+2:end))/2).^2))); %%lambda(u1^2+3*u2^2)
    cvec21=((3*(((ce(1:M+1)+cy(1:M+1))/2).^2))+((((ce(M+2:end)+...
    ...cy(M+2:end))/2).^2))); %%lambda(3u1^2+u2^2)
    ADM=[diag(cvec11), diag(cvec12); -diag(cvec21), -diag(cvec11)];
    DMAT=(((1/dt)*eye(size(BA)))+(BA*0.5)+(ADM*0.5));
        % rigth side
   rvec1=((((0.5)*(ce(1:M+1)+cy(1:M+1))).^2)+(((0.5)*(ce(M+2:end)+...
   ...cy(M+2:end))).^2)).*((0.5)*(ce(M+2:end)+cy(M+2:end)));
   rvec2=-((((0.5)*(ce(1:M+1)+cy(1:M+1))).^2)+(((0.5)*(ce(M+2:end)+...
   ...cy(M+2:end))).^2)).*((0.5)*(ce(1:M+1)+cy(1:M+1)));
   DVEC=vertcat(rvec1,rvec2);
   rmat1=((-(1/dt)*eye(size(BA)))-(BA*0.5))*cy;
   rmat2=(((1/dt)*eye(size(BA)))-(BA*0.5))*ce;
   RS=rmat1+rmat2-DVEC;
   ters=inv(DMAT);
   teta=ters*RS;
    temp=cy+teta;
    err=norm((temp-cy),2);
    cy=temp;
    if err<=tol
        break
    end
    end
    ce=cy; %%%sol(x,t)
    sol(:,j+1)=ce;
    sol(1,j+1)=0;
    sol(M+1,j+1)=0;
    sol(M+2,j+1)=0;
    sol(end,j+1)=0;
end

%% Global energy
top66=0;
for p=1:M
        part1= (((sol(p+1,2)-sol(p,2))^2) +...
        ...((sol(M+1+p+1,2)-sol(M+1+p,2))^2))/(dx^2);
        part2= (((sol(p+1,1)-sol(p,1))^2) +...
        ...((sol(M+1+p+1,1)-sol(M+1+p,1))^2))/(dx^2);
        part3= (sol(p,2)^2+sol(M+1+p,2)^2)*(sol(p,1)^2+...
        ...sol(M+1+p,1)^2);
        bak=part1+part2-part3;
        top66=top66+bak;
end
    energysecond0=0.5*dx*top66;
for k=1:N%time counter
    top5=0;
    for s=1:M %xcounter
        part1= (((sol(s+1,k+1)-sol(s,k+1))^2) +...
        ...((sol(M+1+s+1,k+1)-sol(M+1+s,k+1))^2))/(dx^2);
        part2= (((sol(s+1,k)-sol(s,k))^2) +...
        ...((sol(M+1+s+1,k)-sol(M+1+s,k))^2))/(dx^2);
        part3= (sol(s,k+1)^2+sol(M+1+s,k+1)^2)*(sol(s,k)^2+sol(M+1+s,k)^2);
        total=part1+part2-part3;
        top5=top5+total;
    end
     energysecond=(0.5*dx*top5)-energysecond0;
     GEE(k)=energysecond;
end



%% Absolute Error conservation of the mass
M0=((sol(1:M+1,1).^2)+(sol(M+2:end,1).^2));
top2m0=0;
for i=1:M+1
top2m0=top2m0+M0(i);
end

for j=1:N+1 %%t counter
top1=0;
for i=1:M+1 %%x counter
    pot=(sol(i,j).^2)+(sol(M+1+i,j).^2);
    top1=top1+pot;
end
 AET(j)=dx*top1;
 AE(j)=abs(AET(j)-(dx*top2m0));
end

%   ENEEXA=zeros(N+1,M+1)
for i=1:M+1
   ENEEXA(:,i)= sqrt(((exact(i,:).^2 +exact(M+1+i,:).^2)));
end
%   PROB=zeros(N+1,M+1);
for i=1:M+1 %%PROB(t,x)
    PROB(:,i)=sqrt(((sol(i,:).^2)+(sol(M+1+i,:).^2)));
end

%% ERRORS
% TESTING INITIAL ENERGY AND THE LAST ENERGY
  energybasfre=dx*(norm(PROB(1,:),2)^2)
  energysonfre=dx*(norm(PROB(end,:),2)^2)
  energybasexa=dx*(norm(ENEEXA(1,:),2)^2)
  energysonexa=dx*(norm(ENEEXA(end,:),2)^2)

% Error for different norms
errorAE=norm(AE,inf)
errorAE2=norm(AE,2)
errorAE1=norm(AE,1)

%% FIGURES
figure(1) [X,Y] = meshgrid( x0:dx:xe,t0:dt:te);
Z = ENEEXA;
surf(X,Y,Z)
xlabel('x') ylabel('t') zlabel('|\psi|')

title('Exact Solution')

figure(2) [X,Y] = meshgrid(x0:dx:xe,t0:dt:te);
Z = PROB;
surf(X,Y,Z)
xlabel('t')
ylabel('x')
zlabel('|\psi|')

title('Proposed Method')

figure(3) plot(t,AE) xlabel('time') ylabel('Absolute energy')

figure(4) plot(t(1:end-1),GEE)

figure(5)
% plot(x,ENEEXA(26,:),'r-',x,ENEEXA(51,:),'g-',...
%...x,ENEEXA(76,:),'m-',x,ENEEXA(end,:),'b-')
% hold all
plot(x,PROB(26,:),'r-*',x,PROB(51,:),'g-*',...
...x,PROB(76,:),'m-*',x,PROB(end,:),'b-*')
legend('t=5','t=10','t=15','t=20') xlabel('x') ylabel('|\psi|')

% figure(6)
% subplot(2,2,1)
% plot(x,ENEEXA(26,:),’r--’,x,PROB(26,:),’r-*’)
% legend(’exact’,’proposed’)
% title(’t=5’)
% xlabel(’x’)
% ylabel(’|\psi|’)
% subplot(2,2,2)
% plot(x,ENEEXA(51,:),’g--’,x,PROB(51,:),’g-*’)
% legend(’exact’,’proposed’)
% title(’t=10’)
% xlabel(’x’)
% ylabel(’|\psi|’)
% subplot(2,2,3)
% plot(x,ENEEXA(76,:),’m--’,x,PROB(76,:),’m-*’)
% legend(’exact’,’proposed’)
% title(’t=15’)
% xlabel(’x’)
% ylabel(’|\psi|’)
% subplot(2,2,4)
% plot(x,ENEEXA(end,:),’r--’,x,PROB(end,:),’r-*’)
% legend(’exact’,’proposed’)
% title(’t=20’)
% xlabel(’x’)
% ylabel(’$|\psi|$’)
```

MAIN PROGRAMME for CASE 2

```
%%%% Soliton Solution iu_t+u_{xx}+|u|^2u=0
%%%% with initial condition u(x,0)=sqrt(2)sech(x+10)e^{i0.25*x}
%%%% Corresponding to CASE 2
%%%%%%%%%%Problem definition
 clear all; close all; clc

% for space
x0=-15; xe=5; M=150; dx=(xe-x0)/M; x=x0:dx:xe;
% for time
N=100; t0=0; te=20; dt=(te-t0)/N; cfl=dt/dx; t=t0:dt:te;

itermax=100; tol=1e-6;
```

```
%%%%%%%%%%% initial condition & initial guess
C0=zeros(2*M+2,1);
for i=1:M+1 %xcounter
    C0(i)=sqrt(2)*sech(x(i)+10)*cos((0.25)*x(i));
    C0(M+1+i)=sqrt(2)*sech(x(i)+10)*sin((0.25)*x(i));
end

yini=C0; ce=yini; cy=zeros(2*M+2,1); temp=zeros(2*M+2,1);
teta=zeros(2*M+2,1);
%%%%%%%%%%%% Boundary Conditions %%%%%%%%%%%
ce(1)=0;ce(2*M+2)=0;  ce(M+1)=0; ce(M+2)=0;

cy(1)=0;cy(2*M+2)=0; cy(M+1)=0; cy(M+2)=0;

%% Exact solution from polyanin
exact=zeros(2*M+2,N+1);
for q=1:N+1 %%timecounter for exact
    for r=1:M+1 %%xcounter for exact
       exact(r,q)=sqrt(2)*cos((0.25*x(r))+((1-(1/16))*t(q)))*...
       ...sech(x(r)-(0.5*t(q))+10);
       exact(M+1+r,q)=sqrt(2)*sin((0.25*x(r))+((1-(1/16))*t(q)))*...
       ...sech(x(r)-(0.5*t(q))+10);
    end
end

%%%% given boundary conditions
exact(1,:)=0; exact(M+1,:)=0; exact(M+2,:)=0; exact(end,:)=0;

%%%%%%%%% Numerical Solution
A=toeplitz([-2,1,0,0, zeros(1,M-3)],[-2,1,0,0, zeros(1,M-3)]);
b=(1/(dx^2))*full(A);
BA=[zeros(M+1,M+1), b; -b, zeros(M+1,M+1)];

%% Main loop
sol=zeros(2*M+2,N+1);
sol(:,1)=ce;
```

```
for j=1:N
    ce(1)=0;ce(2*M+2)=0;
   ce(M+1)=0; ce(M+2)=0;
   cy(1)=0; cy(2*M+2)=0;
   cy(M+1)=0; cy(M+2)=0;
   for k=1:itermax
   % left side
   cvec11=2*(((ce(1:M+1)+cy(1:M+1))/2).*((ce(M+2:end)+...
   ...cy(M+2:end))/2)); %%2lambdau1u2
   cvec12=((((ce(1:M+1)+cy(1:M+1))/2).^2)+(3*(((ce(M+2:end)+...
   ...cy(M+2:end))/2).^2))); %%lambda(u1^2+3*u2^2)
   cvec21=((3*(((ce(1:M+1)+cy(1:M+1))/2).^2))+((((ce(M+2:end)+...
   ...cy(M+2:end))/2).^2))); %%lambda(3u1^2+u2^2)
   ADM=[diag(cvec11), diag(cvec12); -diag(cvec21), -diag(cvec11)];
   DMAT=(((1/dt)*eye(size(BA)))+(BA*0.5)+(ADM*0.5));
       % rigth side
  rvec1=((((0.5)*(ce(1:M+1)+cy(1:M+1))).^2)+(((0.5)*(ce(M+2:end)+...
  ...cy(M+2:end))).^2)).*((0.5)*(ce(M+2:end)+cy(M+2:end)));
  rvec2=-((((0.5)*(ce(1:M+1)+cy(1:M+1))).^2)+(((0.5)*(ce(M+2:end)+...
  ...cy(M+2:end))).^2)).*((0.5)*(ce(1:M+1)+cy(1:M+1)));
  DVEC=vertcat(rvec1,rvec2);
  rmat1=((-(1/dt)*eye(size(BA)))-(BA*0.5))*cy;
  rmat2=(((1/dt)*eye(size(BA)))-(BA*0.5))*ce;
  RS=rmat1+rmat2-DVEC;
  ters=inv(DMAT);
  teta=ters*RS;
   temp=cy+teta;
   err=norm((temp-cy),2);
   cy=temp;
   if err<=tol
       break
   end
   end
   ce=cy; %%%sol(x,t)
   sol(:,j+1)=ce;
   sol(1,j+1)=0;
   sol(M+1,j+1)=0;
```

```
    sol(M+2,j+1)=0;
    sol(end,j+1)=0;
end

%% Global energy
top66=0;
for p=1:M
        part1= (((sol(p+1,2)-sol(p,2))^2) +...
        ...((sol(M+1+p+1,2)-sol(M+1+p,2))^2))/(dx^2);
        part2= (((sol(p+1,1)-sol(p,1))^2) +...
        ...((sol(M+1+p+1,1)-sol(M+1+p,1))^2))/(dx^2);
        part3= (sol(p,2)^2+sol(M+1+p,2)^2)*(sol(p,1)^2+sol(M+1+p,1)^2);
        bak=part1+part2-part3;
        top66=top66+bak;
end
    energysecond0=0.5*dx*top66;
for k=1:N%time counter
    top5=0;
    for s=1:M %xcounter
        part1= (((sol(s+1,k+1)-sol(s,k+1))^2) +...
        ...((sol(M+1+s+1,k+1)-sol(M+1+s,k+1))^2))/(dx^2);
        part2= (((sol(s+1,k)-sol(s,k))^2) +...
        ...((sol(M+1+s+1,k)-sol(M+1+s,k))^2))/(dx^2);
        part3= (sol(s,k+1)^2+sol(M+1+s,k+1)^2)*(sol(s,k)^2+...
        ...sol(M+1+s,k)^2);
        total=part1+part2-part3;
        top5=top5+total;
    end
     energysecond=(0.5*dx*top5)-energysecond0;
     GEE(k)=energysecond;
end



%% Absolute Error conservation of the mass
M0=((sol(1:M+1,1).^2)+(sol(M+2:end,1).^2));
top2m0=0;
for i=1:M+1
```

```
top2m0=top2m0+M0(i);
end

for j=1:N+1 %%t counter
top1=0;
for i=1:M+1 %%x counter
    pot=(sol(i,j).^2)+(sol(M+1+i,j).^2);
    top1=top1+pot;
end
 AET(j)=dx*top1;
 AE(j)=abs(AET(j)-(dx*top2m0));
end

%   ENEEXA=zeros(N+1,M+1)
for i=1:M+1
   ENEEXA(:,i)= sqrt(((exact(i,:).^2 +exact(M+1+i,:).^2)));
end
%   PROB=zeros(N+1,M+1);
for i=1:M+1 %%PROB(t,x)
    PROB(:,i)=sqrt(((sol(i,:).^2)+(sol(M+1+i,:).^2)));
end

%% ERRORS
% TESTING INITIAL ENERGY AND THE LAST ENERGY
  energybasfre=dx*(norm(PROB(1,:),2)^2)
  energysonfre=dx*(norm(PROB(end,:),2)^2)
  energybasexa=dx*(norm(ENEEXA(1,:),2)^2)
  energysonexa=dx*(norm(ENEEXA(end,:),2)^2)

% Error for different norms
errorAE=norm(AE,inf)
errorAE2=norm(AE,2)
errorAE1=norm(AE,1)

%% FIGURES
figure(1) [X,Y] = meshgrid( x0:dx:xe,t0:dt:te);
Z = ENEEXA;
```

```
surf(X,Y,Z)
xlabel('x') ylabel('t') zlabel('|\psi|')

title('Exact Solution')

figure(2) [X,Y] = meshgrid(x0:dx:xe,t0:dt:te);
Z = PROB;
surf(X,Y,Z)
xlabel('t')
ylabel('x')
zlabel('|\psi|')

title('Proposed Method')

figure(3) plot(t,AE) xlabel('time') ylabel('Absolute energy')

figure(4) plot(t(1:end-1),GEE)

figure(5)
% plot(x,ENEEXA(26,:),'r-',x,ENEEXA(51,:),'g-',...
%...x,ENEEXA(76,:),'m-',x,ENEEXA(end,:),'b-')
% hold all
plot(x,PROB(26,:),'r-*',x,PROB(51,:),'g-*',...
...x,PROB(76,:),'m-*',x,PROB(end,:),'b-*')
legend('t=5','t=10','t=15','t=20') xlabel('x') ylabel('|\psi|')

% figure(6)
% subplot(2,2,1)
% plot(x,ENEEXA(26,:),'r--',x,PROB(26,:),'r-*')
% legend('exact','proposed')
% title('t=5')
% xlabel('x')
% ylabel('|\psi|')
% subplot(2,2,2)
% plot(x,ENEEXA(51,:),'g--',x,PROB(51,:),'g-*')
% legend('exact','proposed')
% title('t=10')
% xlabel('x')
% ylabel('|\psi|')
% subplot(2,2,3)
% plot(x,ENEEXA(76,:),'m--',x,PROB(76,:),'m-*')
% legend('exact','proposed')
% title('t=15')
% xlabel('x')
% ylabel('|\psi|')
% subplot(2,2,4)
% plot(x,ENEEXA(end,:),'r--',x,PROB(end,:),'r-*')
% legend('exact','proposed')
% title('t=20')
% xlabel('x')
% ylabel('$|\psi|$')
```

E.3.2. Equation in the form: $i\partial_t\Psi + \beta\partial_x^2\Psi + (G(x) + \alpha|\Psi|^2)\Psi = 0$

MAIN PROGRAMME

```
%%%%%Schrodinger equation with non-linear potential
%%%% iu_t=-0.5u_xx+(V(x)+lambda $|u|^2$)u
clear all; close all; clc

%%%%%%%% problem definition
% for space
x0=-20; xe=20; N=100; dx=(xe-x0)/N; x=x0:dx:xe;
% for time
t0=0; te=8; dt=(te-t0)/N; cfl=dt/dx; t=t0:dt:te;

lambda=30; itermax=100; tol=1e-8;
%%%%%%%% initial condition & initial guess
U1=exp(sin(2*x)); U1=U1/(norm(U1,2)); U1=U1'; norm(U1,2)
U2=zeros(N+1,1);
ce=vertcat(U1,U2); % initial condition
cy=ones(2*N+2,1);  % initial guess
```

```
temp=zeros(2*N+2,1); teta=zeros(2*N+2,1);

%%%%%% Boundary conditions
ce(1)=0;ce(2*N+2)=0;  ce(N+1)=0; ce(N+2)=0;

cy(1)=0;cy(2*N+2)=0; temp(1)=0;

temp(2*N+2)=0; teta(1)=0; teta(2*N+2)=0;

%%%%%% PROPOSED METHOD
A=toeplitz([-2,1,0,0, zeros(1,N-3)],[-2,1,0,0, zeros(1,N-3)]);
b=(1/(dx^2))*full(A);
% [D,x]=chebab(N,x0,xe);
% D2=D^2;
% b=D2;
%  [a,b]= LDQ10(N+1,dx);
for l=1:N+1 vector(l)=1/(1+((sin(x(l))^2))); end BA=[zeros(N+1,N+1),
(0.5)*b; -(0.5)*b, zeros(N+1,N+1)]; BB=[zeros(N+1,N+1),
-diag(vector); diag(vector), zeros(N+1,N+1)];
%%%%%%%% Main loop
for j=1:N %%time counter
     ce(1)=0;ce(2*N+2)=0;
    ce(N+1)=0; ce(N+2)=0;
    cy(1)=0;cy(2*N+2)=0;
    cy(N+1)=0; cy(N+2)=0;
    for k=1:itermax
        % left side
     cvec11=2*lambda*(((ce(1:N+1)+cy(1:N+1))/2).*...
     ...((ce(N+2:end)+cy(N+2:end))/2)); %%2lambdau1u2
    cvec12=lambda*((((ce(1:N+1)+cy(1:N+1))/2).^2)+...
    ...(3*(((ce(N+2:end)+cy(N+2:end))/2).^2))); %%lambda(u1^2+3*u2^2)
    cvec21=lambda*((3*(((ce(1:N+1)+cy(1:N+1))/2).^2))+...
    ...((((ce(N+2:end)+cy(N+2:end))/2).^2))); %%lambda(3u1^2+u2^2)
    ADM=[-diag(cvec11),- diag(cvec12); diag(cvec21), diag(cvec11)];
    DMAT=(((1/dt)*eye(size(BA)))+(BA*0.5)+(BB*0.5)+(ADM*0.5));
        % right side
   rvec1=-lambda*((((0.5)*(ce(1:N+1)+cy(1:N+1))).^2)+...
   ...(((0.5)*(ce(N+2:end)+cy(N+2:end))).^2)).*((0.5)*...
   ...(ce(N+2:end)+cy(N+2:end)));
   rvec2=lambda*((((0.5)*(ce(1:N+1)+cy(1:N+1))).^2)+...
   ...(((0.5)*(ce(N+2:end)+cy(N+2:end))).^2)).*((0.5)*...
   ...(ce(1:N+1)+cy(1:N+1)));
   DVEC=vertcat(rvec1,rvec2);
   rmat1=((-(1/dt)*eye(size(BA)))-(BA*0.5)-(BB*0.5))*cy;
   rmat2=(((1/dt)*eye(size(BA)))-(BA*0.5)-(BB*0.5))*ce;
   RS=rmat1+rmat2-DVEC;
   ters=inv(DMAT);
   teta=DMAT(2:2*N+1,2:2*N+1)\RS(2:2*N+1);
   temp(2:2*N+1)=cy(2:2*N+1)+teta;
    err=norm((temp-cy),2);
    cy=temp;
    if err<=tol
        break
    end
    end
    ce=cy;
    sol(:,j+1)=ce;
    sol(1,j+1)=0;
    sol(N+1,j+1)=0;
    sol(N+2,j+1)=0;
    sol(end,j+1)=0;
end
%   PROB=zeros(N+1,N+1);
for i=1:N+1 %%PROB(t,x)
    PROB(:,i)=(sol(i,:).^2)+(sol(N+1+i,:).^2);
     SOLIT(:,i)=sqrt((sol(i,:).^2)+(sol(N+1+i,:).^2));
end
%enerbasfre= dx*(norm(SOLIT(1,:),2)^2)
%enersonfre= dx*(norm(SOLIT(end,:),2)^2)
[X,Y] = meshgrid(t0:dt:te ,x0:dx:xe);
Z = PROB;
surf(X,Y,Z)
xlabel('t') ylabel('x')
zlabel('probability density of Frechet Derivatives')
```

VITA

Date and Place of Birth: 19.08.1985, Çorum-TURKEY

EDUCATION

2012 - 2015 Doctor of Philosophy in Mathematics
Graduate School of Engineering and Sciences, İzmir Institute of Technology,
İzmir -Turkey
Thesis Title: NEW APPROACHES FOR SOLVING NONLINEAR OSCILLATION PROBLEMS
Supervisor: Prof. Dr. Gamze Tanoğlu

2010 - 2012 Master of Science in Mathematics
Graduate School of Engineering and Sciences, İzmir Institute of Technology
İzmir -Turkey
Thesis Title: OPERATOR SPLITTING METHODS FOR NON-AUTONOMOUS DIFFERENTIAL EQUATIONS
Supervisor: Assoc. Prof. Dr. Gamze Tanoğlu

2008 - 2009 Master of Science in Educational Science(Non-thesis)
Department of Mathematics Education, Faculty of Education, Ege University
İzmir -Turkey

2003 - 2008 Bachelor of Mathematics
Department of Mathematics, Faculty of Science, Ege University
İzmir - Turkey

PROFESSIONAL EXPERIENCE

2009 - 2013 Research and Teaching Assistant
Department of Mathematics, İzmir Institute of Technology,
İzmir -Turkey

2013 - Present Instructor
Department of Engineering Science, Faculty of Engineering and Architecture
İzmir Kâtip Çelebi University, İzmir -Turkey

PUBLICATIONS

Korkut Uysal S. Ö.and Tanoğlu G., 2015: "A New Linearized Method for Solving Nonlinear Schrödinger Equation." Proceedings of the 15th International Conference. on Computational and Mathematical Methods in Science and Engineering, CMMSE 2015, ISBN: 978-84-617-2230-3, Pp. 679-689

Tanoglu G. and Korkut S. O., 2014: "A New Operator Splitting Method for Non-Linear Systems and Its Abstract Analysis." Abstract Book of ICRAPAM 2014, International Conference on Recent Advances in Pure and Applied Mathematics, ISBN: 978-975-00211-1-4, Pp. 226

Tanoglu G. and Korkut S. O., 2012: " The convergence of a new symmetric iterative splitting method for non-autonomous systems.", International Journal of Computer Mathematics Volume 89, Issue 13-14, 2012 , Pp. 1837-1846.
DOI: http://www.tandfonline.com/doi/abs/10.1080/00207160.2012.687447

Tanoglu G. and Korkut S., 2011: "Symmetric Iterative Splitting Method for Non-Autonomous Systems." Proceedings of the 11th International Conference on Computational and Mathematical Methods in Science and Engineering, CMMSE 2011, Volume III ISBN: 978-84-614-6167-7, Pp. 1104-1112.

Tanoglu G. and Korkut S., 2011: "Iterative Splitting Method for Schrödinger Equation with time dependent potential." Proceedings of the 2nd International Symposium on Computing in Science and Engineering, ISCSE 2011, (700/58) Pp: 1219-1224.

Tanoglu G., Korkut S. and Gucuyenen N., 2011: "Error Analysis of Splitting Methods for Non-Autonomous Systems." Proceeding Book of International Conference on Applied Analysis and Algebra - Abstracts, ICAAA2011, Pp.127.