CLASSICAL TIME SPLITTING APPROACHES AND THEIR ERROR ANALYSES FOR NONLINEAR DIFFERENTIAL EQUATIONS

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

MASTER OF SCIENCE

in Mathematics

by
Elif HACISALİHOĞLU

June 2018
İzmir

We approve the thesis of **Elif HACISALİHOĞLU**

Examining Committee Members:

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

Asst. Prof. Sevin GÜMGÜM
Department of Mathematics, İzmir University of Economics

Prof. Dr. İsmail ASLAN
Department of Mathematics, İzmir Institute of Technology

26 June 2018

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

Prof. Dr. Engin BÜYÜKAŞIK
Head of the Department of
Mathematics

Prof. Dr. Aysun SOFUOĞLU
Dean of the Graduate School of
Engineering and Sciences

ACKNOWLEDGMENTS

I am indebted to all the people who have contributed to the successful completion of this thesis both directly and indirectly.

First and foremost, I would like to thank my supervisor Prof. Dr. Gamze TANOĞLU for her guidance, great support and understanding during my studies to prepare this thesis. I am extremely lucky to have such a supportive and thoughtful supervisor. I owe the same amount of thanks to Prof. Dr. İsmail ASLAN and Asst. Prof. Sevin GÜMGÜM for being members of my thesis committee.

I am sincerely grateful to Dr. Barış ÇİÇEK and Assist. Neslişah İMAMOĞLU KARABAŞ for sharing their knowledge and experiences about the Matlab implementation with me.

Last but not the least; I am especially grateful to my mother Ayşe HACISALİHOĞLU, my father Fahrettin Kerim HACISALİHOĞLU and my brother M. Yasin HACISALİHOĞLU for unconditionally supporting me and believing in me all this time. Indeed, without their emotionally and financially support, this accomplishment would not have been made possible.

ABSTRACT

CLASSICAL TIME SPLITTING APPROACHES AND THEIR ERROR ANALYSES FOR NONLINEAR DIFFERENTIAL EQUATIONS

In this thesis, Lie - Trotter splitting, Strang - Marchuk splitting and symmetrically weighted sequential (SWS) splitting methods which are known as classical operator splitting methods are considered to find the numerical solution of the various ordinary differential equations (ODEs) and partial differential equations (PDEs). We also presented their error analyses in order to show advantages and disadvantages of these methods.

Firstly, we considered simple linear and nonlinear ODE examples to motivate for the classical operator splitting methods. Then, two numerical examples which consist of a kinetic model of phage infection and the Newell - Whitehead - Segel (NWS) equation are studied.

All these examples show that the operator splitting methods are a powerful technique with respect to the accuracy and robustness.

ÖZET

LİNEER OLMAYAN DİFERANSİYEL DENKLEMLER İÇİN KLASİK ZAMAN AYIRMA YAKLAŞIMLARI VE HATA ANALİZLERİ

Bu tezde klasik operatör ayırma metodları olarak bilinen Lie - Trotter ayırma, Strang - Marchuk ayırma ve symmetrically weighted sequential (SWS) ayırma metodları çeşitli adi diferansiyel denklemlerin ve kısmi diferansiyel denklemlerin sayısal çözümünü bulmak için ele alınmıştır. Ayrıca bu yöntemlerin avantajlarını ve dezavantajlarını göstermek için hata analizlerini sunduk.

İlk olarak, klasik operatör ayırma metodlarına motive olmak için basit lineer ve lineer olmayan ODE örneklerini düşündük. Daha sonra, bir faj enfeksiyonun kinetic bir modelinden ve Newell - Whitehead - Segel (NWS) denkleminden oluşan iki sayısal örnek üzerinde çalışılmıştır.

Bütün bu örnekler operatör ayırma metodlarının doğruluk ve sağlamlık açısından güçlü bir teknik olduğunu göstermektedir.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

INTRODUCTION

Nonlinear differential equations can be used to describe many complex phenomena in sciences, such as fluid and plasma mechanics, biomathematics, convection, diffusion and chemical reactions. These equations are generally difficult to solve and their exact solutions are difficult to obtain. Therefore, some various approximate methods have been developed to solve nonlinear differential equations.

In this thesis, we concentrate on operator splitting methods which is one of the approximate methods. Especially, we'll deal with Lie-Trotter, Strang-Marchuk and symmetrically weighted sequential splitting methods also known as classical operator splitting methods. The main idea of these methods is to get simpler problems that can be analyzed separately by applying the operator splitting method to a complex problem.

Operator splitting methods separate the original equation into two parts contains over a time step, separately computes the solution to each part, and then combines the two separate solutions to form a solution to the original equation [1].

The idea of operator splitting, which was the Lie-Trotter splitting, dates back to the 1950s. It was probably in 1957 that this method was first used in the solution of partial differential equations. The first splitting methods were developed in the 1960s or 1970s and were based on fundamental results of finite difference methods [2].

There are many major benefits of the operator splitting methods, including dimension reduction, problem simplification, preservation of any order accuracy in time, and computational speed-up for some complex problems [3].

The aim of this thesis is to obtain approximated solutions of nonlinear differential equations numerically by using the classical operator splitting method. Within the scope of the thesis, the equation is divided into two parts as the linear and nonlinear by using this method.

The general idea behind splitting is breaking down a complicated problem into smaller parts for the sake of time stepping, such that the different parts can be solved efficiently with suitable integration formula [4]. In all cases, the computational advantage is that it is faster to compute the solution of the split terms separately, than to compute the solution directly when they are treated together. However, this comes at the cost of an error introduced by the splitting. So, strategies have been devised to control this error [1].

The consistency of different splitting schemes has mostly been studied for bounded operators by means of the traditional Taylor series expansion. In this thesis, first order of consistency is proved for Lie-Trotter splitting, second order of consistency is proved for the symmetrically weighted sequential splitting and the Strang-Marchkuk splitting [5].

The outline of this thesis as follows: In Chapter 2, we introduce Lie-Trotter splitting, Strang-Marchuk splitting and symmetrically weighted sequential splitting methods which are known as classical operator splitting methods on an abstract Cauchy problem. We also prove their local splitting errors by using Taylor series expansion. In Chapter 3, we deal with linear and nonlinear ODE problems in order to motivate the algorithms and error analyses of the classical operator splitting methods. In Chapter 4, we give some numerical examples to confirm our theoretical results and to demonstrate the effectiveness of our suggested method. Finally, in Chapter 5, we make a brief discussion for the results of our study.

CHAPTER 2

CLASSICAL OPERATOR SPLITTING METHODS

In this chapter, we give a short overview of the classical operator splitting methods which are known as Lie-Trotter splitting, Strang-Marchuk splitting and symmetrically weighted sequential splitting methods. In addition to this, we show their errors by using local splitting error. In order to introduce these methods, we consider the following abstract Cauchy problem, also called initial value problem (IVP) :

$$\begin{cases} \dfrac{du(t)}{dt} = (A+B)u(t), \quad t \in [0,T] \\ u(0) = u_0, \end{cases} \tag{2.1}$$

where A and B are assumed to be linear operators in Banach space X with $A, B : X \longrightarrow X$ and $u_0 \in X$ is initial condition. When A and B are bounded operators, the exact solution is given by

$$u(t^{n+1}) \quad = \quad e^{\Delta t(A+B)} u(t^n), \tag{2.2}$$

where time step is $\Delta t = t^{n+1} - t^n$ and $u(t^n)$ is a solution at time $t = t^n$.

Let us concentrate on the following classical operator splitting methods.

2.1. Lie - Trotter Splitting

Firstly, we describe the first order splitting method which is called Lie-Trotter splitting. It has a very simple process which separates the Cauchy problem (2.1) into two subproblems. The first subproblem is solved with operator A and the original initial condition. The second one is solved with operator B whose initial condition is derived from the solution of the first subproblem.

For Lie-Trotter splitting, this process can be formulated as follows:

$$\begin{cases} \dfrac{du_1(t)}{dt} = Au_1(t), \quad t \in [t^n, t^{n+1}] \\ u_1(t^n) = u_{sp}^n, \end{cases} \tag{2.3}$$

$$\begin{cases} \dfrac{du_2(t)}{dt} = Bu_2(t), \quad t \in [t^n, t^{n+1}] \\ u_2(t^n) = u_1(t^{n+1}), \end{cases} \tag{2.4}$$

where split condition at $t = 0$ is given by $u_{sp}^0 = u_0$ in (2.1) and approximated split solution at time $t = t^{n+1}$ is defined as

$$u_{sp}^{n+1} = u_2(t^{n+1}), \tag{2.5}$$

where $t^{n+1} = t^n + \Delta t$, Δt is time step, and $n = 0, 1, ..., N - 1$.

The above-mentioned process can be shown as in the following figure :

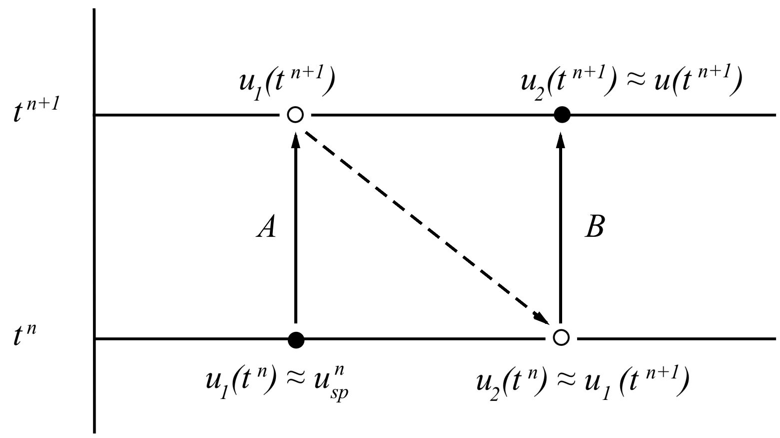


Figure 2.1. Systematic schema of Lie - Trotter splitting.

To analyze the error of the Lie-Trotter splitting, we compare the exact solution in (2.2) with the following solution

$$u_{sp}(t^{n+1}) = e^{\Delta t A} e^{\Delta t B} u(t^n) \tag{2.6}$$

which is obtained by Lie-Trotter splitting. By Taylor expansion of $u(\Delta t)$ and $u_{sp}(\Delta t)$ we get, respectively

$$u(\Delta t) = \left(I + \Delta t(A+B) + \frac{\Delta t^2}{2!}(A+B)^2 + O(\Delta t^3)\right)u_0, \tag{2.7}$$

$$u_{sp}(\Delta t) = \left(I + \Delta t(A+B) + \frac{\Delta t^2}{2!}(A^2 + 2BA + B^2) + O(\Delta t^3)\right)u_0. \tag{2.8}$$

Substracting (2.8) from (2.7) gives the following expression for the local splitting error of the Lie-Trotter splitting:

$$err_{lie} = \frac{\Delta t^2}{2!}[A, B]u_0 + O(\Delta t^3), \tag{2.9}$$

where $[A, B] = AB - BA$ is the commutator of A and B. Consequently, Lie-Trotter splitting method is first order consistent if the operators A and B do not commute. When the operators commute, then the method is exact.

2.2. Strang - Marchuk Splitting

Another classical operator splitting method is second order splitting which is called the Strang-Marchuk splitting. This splitting method divides the split time subinterval into two parts. Then, as in the Lie-Trotter splitting process, successively solves the problems on the first half interval with operator A, on the whole interval with operator B and on the second half interval again with operator A. The first subproblem uses the original initial condition and the others use the solutions of the previous problems as initial conditions.

For Strang-Marchuk splitting, this process can be formulated as follows:

$$\begin{cases} \dfrac{du_1(t)}{dt} = Au_1(t), \quad t \in [t^n, t^{n+1/2}] \\ u_1(t^n) = u^n_{sp}, \end{cases} \tag{2.10}$$

$$\begin{cases} \dfrac{du_2(t)}{dt} = Bu_2(t), \quad t \in [t^n, t^{n+1}] \\ u_2(t^n) = u_1(t^{n+1/2}), \end{cases} \tag{2.11}$$

$$\begin{cases} \dfrac{du_3(t)}{dt} = Au_3(t), \quad t \in [t^n, t^{n+1/2}] \\ u_3(t^{n+1/2}) = u_2(t^{n+1}), \end{cases} \tag{2.12}$$

where split condition at $t = 0$ is given by $u_{sp}^0 = u_0$ in (2.1) and approximated split solution at time $t = t^{n+1}$ is defined as

$$u_{sp}^{n+1} \quad = \quad u_3(t^{n+1}) \tag{2.13}$$

where $t^{n+1} = t^n + \Delta t$, Δt is time step, and $n = 0, 1, ..., N-1$.

This process can be illustrated as below figure :

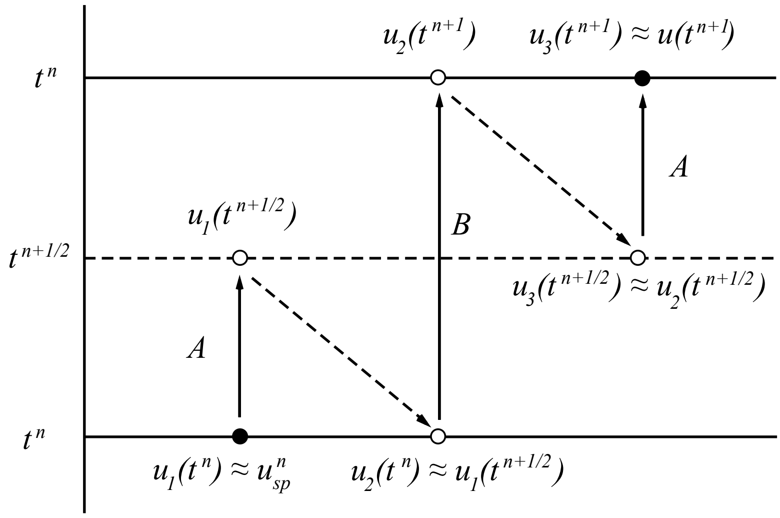


Figure 2.2. Systematic schema of Strang - Marchuk splitting.

In order to obtain the local splitting error of the Strang-Marchuk splitting, we compare the exact solution (2.2) with the following solution

$$u_{sp}(t^{n+1}) \quad = \quad e^{(\Delta t/2)A} e^{\Delta t B} e^{(\Delta t/2)A} u(t^n) \tag{2.14}$$

which is obtained by Strang-Marchuk splitting. By Taylor expansion of $u_{sp}(\Delta t)$, we get

$$u_{sp}(\Delta t) = \left(I + \Delta t(A + B) + \frac{\Delta t^2}{2!}(A^2 + BA + AB + B^2) + O(\Delta t^3)\right) u_0. \qquad (2.15)$$

Substracting (2.15) from (2.7) gives the following expression

$$err_{strang} = \frac{1}{24}\Delta t^2 \left(2[B,[B,A]] - [A,[A,B]]\right) u_0 + O(\Delta t^3), \qquad (2.16)$$

for the local splitting error of the Strang-Marchuk splitting and it is seen that this splitting gives second order accuracy.

2.3. Symmetrically Weighted Sequential Splitting

The third classical operator splitting method which is called symmetrically weighted sequential splitting is a combination of two Lie-Trotter splitting in different ordering. It makes that the splitting is symmetric and the accuracy of the splitting is second order. The Cauchy problem (2.1) with operators A and B processed in different ordering 'AB' and 'BA' respectively, and at the end of the time steps the obtained solutions are taken a weighted average.

For SWS splitting, this process can be formulated as follows:

We begin with 'AB' recombination

$$\begin{cases} \dfrac{du_1(t)}{dt} = Au_1(t), \quad t \in [t^n, t^{n+1}] \\ u_1(t^n) = u^n_{sp}, \end{cases} \qquad (2.17)$$

$$\begin{cases} \dfrac{du_2(t)}{dt} = Bu_2(t), \quad t \in [t^n, t^{n+1}] \\ u_2(t^n) = u_1(t^{n+1}), \end{cases} \qquad (2.18)$$

and similary 'BA' recombination

$$\begin{cases} \dfrac{dv_1(t)}{dt} = Bv_1(t), \quad t \in [t^n, t^{n+1}] \\ v_1(t^n) = v_{sp}^n, \end{cases} \tag{2.19}$$

$$\begin{cases} \dfrac{dv_2(t)}{dt} = Av_2(t), \quad t \in [t^n, t^{n+1}] \\ v_2(t^n) = v_1(t^{n+1}). \end{cases} \tag{2.20}$$

Then the split solution at the mesh points is defined as:

$$u_{sp}^{n+1} = \frac{u_2(t^{n+1}) + v_2(t^{n+1})}{2}. \tag{2.21}$$

Similarly, in order to show the local splitting error of the SWS splitting, we compare the exact solution (2.2) with the following solution

$$u_{sp}(t^{n+1}) = \frac{e^{A\Delta t}e^{B\Delta t} + e^{B\Delta t}e^{A\Delta t}}{2} u(t^n) \tag{2.22}$$

which is obtained by SWS splitting. By Taylor expansion of $u_{sp}(\Delta t)$, we get

$$u_{sp}(\Delta t) = \left(I + \Delta t(A + B) + \frac{\Delta t^2}{2!}(A^2 + 2BA + B^2) + O(\Delta t^3)\right) u_0. \tag{2.23}$$

Substracting (2.23) from (2.7) gives the following expression

$$err_{symm} = O(\Delta t^3) \tag{2.24}$$

for the local splitting error of the SWS splitting and we can easily see that SWS splitting is a second order of accuracy.

CHAPTER 3

MOTIVATION FOR THE OPERATOR SPLITTING METHODS

In this chapter, simple examples are illustrated to demonstrate the performance of the classical operator splitting methods. For this purpose, several linear and nonlinear ODEs we have chosen are studied. We also verified the theoretical analysis given in the previous chapter with numerical simulations.

3.1. Linear Demonstration

In order to solve a linear first order differential equation, we start with a differential equation in the normal form

$$u' \quad = \quad p(t)u + q(t), \tag{3.1}$$

where both $p(t)$ and $q(t)$ are continuous functions and t on a certain interval.

We begin with the following example :

Example 3.1 We consider the following IVP

$$u' \quad = \quad -u + 3e^{-2t}, \quad u(0) = 1. \tag{3.2}$$

We can solve (3.2) by finding an integrating factor $\mu(t)$. If we choose $\mu(t)$ to be

$$\mu \quad = \quad e^{\int 1dt} = e^{t},$$

and multiply both sides of the equation (3.2) by μ, we can rewrite it as

$$\frac{d}{dt}(e^{t}u(t)) \quad = \quad 3e^{-t}.$$

Integrating with respect to t, we obtain

$$\begin{aligned} e^t u(t) &= \int 3e^{-t} dt + c \\ &= -3e^{-t} + c. \end{aligned}$$

Dividing through by e^t, we calculate that the the general form of the solution of equation (3.2) is

$$u(t) = -3e^{-2t} + ce^{-t}.$$

Applying the initial condition $u(0) = 1$, we obtain the exact solution

$$u(t) = -3e^{-2t} + 4e^{-t}. \tag{3.3}$$

3.1.1. Lie - Trotter Splitting for Linear ODE

We shall consider the Lie-Trotter splitting of (3.2) into split equation 'A',

$$u_1' = -u_1, \quad u_1(0) = 1$$

and the split equation 'B',

$$u_2' = 3e^{-2t}, \quad u_2(0) = u_1(t)$$

and recombine their solutions in sequential scheme designed to preserve a certain level of accuracy in time.

For example, the 'AB' recombination scheme results in the piecewise solution,

$$u_{AB}(t) = \begin{cases} e^{-t}, & 0 \le t \le t^n \\ \frac{1}{2}\left(-3e^{-2t} + 2e^{-t} + 3\right), & 0 \le t \le t^n \end{cases} \tag{3.4}$$

while reordering the split equations to the 'BA' recombination scheme results in

$$u_{BA}(t) = \begin{cases} \frac{1}{2}\left(-3e^{-2t} + 5\right), & 0 \leq t \leq t^n \\ \frac{1}{2}\left(-3e^{-3t} + 5e^{-t}\right), & 0 \leq t \leq t^n \end{cases} \tag{3.5}$$

Although is these piecewise solutions (3.4) and (3.5) are not differentiable, they are continuous. It obvious that the two recombination schemes (3.4) and (3.5) are different, however, they both preserve a first order approximation in time to the exact solution of the equation of (3.2).

3.1.2. Accuracy of Lie - Trotter Splitting for Linear ODE

To demonstrate the Lie - Trotter splitting's accuracy for Example 3.1, we solve the equation over small steps $t_f = \Delta t$. The accuracy of the error due to splitting is determined by the order, under Taylor expansion, to which the solutions agree.

The first order splitting accuracy in time for the 'AB' recombination is shown through the following Taylor expansion,

$$\begin{aligned}
\left|u(\Delta t) - u_{AB}(\Delta t)\right| &= \left|\left(-3e^{-2\Delta t} + 4e^{-\Delta t}\right) - \left(-\frac{3}{2}e^{-2\Delta t} + e^{-\Delta t} + \frac{3}{2}\right)\right| \\
&= \Bigg|\left[-3\left(1 - 2\Delta t + 2\Delta t^2 + O(\Delta t^3)\right) + 4\left(1 - \Delta t + \frac{1}{2!}\Delta t^2 + O(\Delta t^3)\right)\right] \\
&\quad - \left[-\frac{3}{2}\left(1 - 2\Delta t + 2\Delta t^2 + O(\Delta t^3)\right) + \left(1 - \Delta t + \frac{1}{2!}\Delta t^2 + O(\Delta t^3)\right) + \frac{3}{2}\right]\Bigg| \\
&= \left|\left(1 + 2\Delta t - 4\Delta t^2 + O(\Delta t^3)\right) - \left(1 + 2\Delta t - \frac{5}{2}\Delta t^2 + O(\Delta t^3)\right)\right| \\
&= \left|-\frac{3}{2}\Delta t^2 + O(\Delta t^3)\right| \\
&= O(\Delta t^2).
\end{aligned}$$

Since the error between the exact and 'AB' split solution is $O(\Delta t^2)$, the solutions agree up to order $O(\Delta t)$, which is thus the splitting accuracy of the 'AB' recombination scheme.

The first order splitting accuracy in time of the 'BA' recombination scheme is similary shown as

$$\begin{aligned}
\left|u(\Delta t) - u_{BA}(\Delta t)\right| &= \left|\left(-3e^{-2\Delta t} + 4e^{-\Delta t}\right) - \left(-\frac{3}{2}e^{-3\Delta t} + \frac{5}{2}e^{-\Delta t}\right)\right| \\
&= \left|\left[-3\left(1 - 2\Delta t + 2\Delta t^2 + O(\Delta t^3)\right) + 4\left(1 - \Delta t + \frac{1}{2!}\Delta t^2 + O(\Delta t^3)\right)\right]\right. \\
&\quad \left. - \left[-\frac{3}{2}\left(1 - 3\Delta t + \frac{9}{2}\Delta t^2 + O(\Delta t^3)\right) + \frac{5}{2}\left(1 - \Delta t + \frac{1}{2!}\Delta t^2 + O(\Delta t^3)\right)\right]\right| \\
&= \left|\left(1 + 2\Delta t - 4\Delta t^2 + O(\Delta t^3)\right) - \left(1 + 2\Delta t - \frac{11}{2}\Delta t^2 + O(\Delta t^3)\right)\right| \\
&= \left|\frac{3}{2}\Delta t^2 + O(\Delta t^3)\right| \\
&= O(\Delta t^2).
\end{aligned}$$

Obivously, the splitting error for the two orderings of the first order recombination scheme is not the same but it is of the same order.

3.1.3. Strang - Marchuk Splitting for Linear ODE

We will consider the Strang - Marchuk splitting of (3.2) into split equation 'A',

$$u_1' = -\frac{u_1}{2}, \quad u_1(0) = 1$$

and the split equation 'B',

$$u_2' = 3e^{-2t}, \quad u_2(0) = u_1(t)$$

and again the split equation 'A',

$$u_3' = -\frac{u_3}{2}, \quad u_3(0) = u_2(t).$$

Then, recombine their solutions in sequential scheme designed to preserve a certain level of accuracy in time.

For instance, the 'ABA' recombination scheme results in the piecewise solution,

$$u_{ABA}(t) = \begin{cases} e^{-t/2}, & 0 \leq t \leq t^{n+1/2} \\ \frac{1}{2}\left(-3e^{-2t} + 2e^{-t/2} + 3\right), & 0 \leq t \leq t^{n} \\ \frac{1}{2}\left(-3e^{-5t/2} + 2e^{-t} + 3e^{-t/2}\right), & t^{n+1/2} \leq t \leq t^{n} \end{cases} \tag{3.6}$$

while reordering the split equations to the 'BAB ' recombination scheme results in

$$u_{BAB}(t) = \begin{cases} \frac{1}{4}\left(-3e^{-2t} + 7\right), & 0 \leq t \leq t^{n+1/2} \\ \frac{1}{4}\left(-3e^{-3t} + 7e^{-t}\right), & 0 \leq t \leq t^{n} \\ \frac{1}{4}\left(-3e^{-3t} - 3e^{-2t} + 7e^{-t} + 3\right), & t^{n+1/2} \leq t \leq t^{n} \end{cases} \tag{3.7}$$

Although these piecewise solutions (3.6) and (3.7) are not differentiable, they are continuous. Also, though the two recombination schemes (3.6) and (3.7) are different, they both preserve a second order approximation in time to the exact solution of the equation of (3.2).

3.1.4. Accuracy of Strang - Marchuk Splitting for Linear ODE

To demonstrate the Strang-Marchuk splitting's accuracy Example 3.1, we solve the equation over small steps $t_f = \Delta t$. The accuracy of the error due to splitting is determined by the order, under Taylor expansion, to which the solutions agree.

The second order splitting accuracy in time for the 'ABA' recombination is shown through the following Taylor expansions,

$$\begin{aligned} \left|u(\Delta t) - u_{ABA}(\Delta t)\right| &= \left|\left(-3e^{-2\Delta t} + 4e^{-\Delta t}\right) - \left(-\frac{3}{2}e^{-5\Delta t/2} + e^{-\Delta t} + \frac{3}{2}e^{-\Delta t/2}\right)\right| \\ &= \left|\left[-3\left(1 - 2\Delta t + 2\Delta t^2 + O(\Delta t^3)\right) + 4\left(1 - \Delta t + \frac{1}{2!}\Delta t^2 + O(\Delta t^3)\right)\right]\right. \\ &\quad -\left[-\frac{3}{2}\left(1 - \frac{5}{2}\Delta t + \frac{25}{8}\Delta t^2 + O(\Delta t^3)\right) + \left(1 - \Delta t + \frac{1}{2!}\Delta t^2 + O(\Delta t^3)\right)\right. \\ &\quad \left.\left. + \frac{3}{2}\left(1 - \frac{1}{2}\Delta t + \frac{1}{8}\Delta t^2 + O(\Delta t^3)\right)\right]\right| \\ &= \left|\left(1 + 2\Delta t - 4\Delta t^2 + O(\Delta t^3)\right) - \left(1 + 2\Delta t - 4\Delta t^2 + O(\Delta t^3\right)\right| \\ &= O(\Delta t^3). \end{aligned}$$

Since the error between the exact and '*ABA*' split solution is $O(\Delta t^3)$, the solutions agree up to order $O(\Delta t^2)$, which is thus the splitting accuracy of the '*ABA*' recombination scheme.

The second order splitting accuracy in time of the '*BAB*' recombination scheme is similary shown as

$$\begin{aligned}
\left|u(\Delta t) - u_{BAB}(\Delta t)\right| &= \left|\left(-3e^{-2\Delta t} + 4e^{-\Delta t}\right) - \left(-\frac{3}{4}e^{-3\Delta t} - \frac{3}{4}e^{-2\Delta t} + \frac{7}{4}e^{-\Delta t} + \frac{3}{4}\right)\right| \\
&= \Bigg|\left[-3\left(1 - 2\Delta t + 2\Delta t^2 + O(\Delta t^3)\right) + 4\left(1 - \Delta t + \frac{1}{2!}\Delta t^2 + O(\Delta t^3)\right)\right] \\
&\quad -\left[-\frac{3}{4}\left(1 - 3\Delta t + \frac{9}{2}\Delta t^2 + O(\Delta t^3)\right) - \frac{3}{4}\left(1 - 2\Delta t + 2\Delta t^2 + O(\Delta t^3)\right)\right. \\
&\quad \left. + \frac{7}{4}\left(1 - \Delta t + \frac{1}{2!}\Delta t^2 + O(\Delta t^3)\right) + \frac{3}{4}\right]\Bigg| \\
&= \left|\left(1 + 2\Delta t - 4\Delta t^2 + O(\Delta t^3)\right) - \left(1 + 2\Delta t - 4\Delta t^2 + O(\Delta t^3)\right)\right| \\
&= O(\Delta t^3).
\end{aligned}$$

It is clear now that the splitting error for the two orderings of the second order recombination scheme is not the same but it is of the same order.

3.1.5. Symmetrically Weighted Sequential Splitting for Linear ODE

We shall consider a second order splitting of (3.2) into split equation '*A*',

$$u_1' = -u_1, \quad u_1(0) = 1$$

and the split equation '*B*',

$$u_2' = 3e^{-2t}, \quad u_2(0) = u_1(t)$$

and recombine their solutions in sequential scheme designed to preserve a certain level of accuracy in time.

For example, the result of '*AB*' recombination scheme (3.4),

$$u_{AB}(t) = \frac{1}{2}\left(-3e^{-2t} + 2e^{-t} + 3\right)$$

and the result of 'BA' recombination scheme (3.5),

$$u_{BA}(t) = \frac{1}{2}\left(-3e^{-3t} + 5e^{-t}\right).$$

Then the split solution at the mesh points is defined as:

$$u_{sp}^{n+1} = \frac{u_{AB}(t^{n+1}) + u_{BA}(t^{n+1})}{2}. \tag{3.8}$$

Thus, we can obtain the approximate solution as

$$u_{symm}(t) = -\frac{3}{4}e^{-3t} - \frac{3}{4}e^{-2t} + \frac{7}{4}e^{-t} + \frac{3}{4}. \tag{3.9}$$

3.1.6. Accuracy of Linear ODE for Symmetrically Weighted Sequential Splitting

To show the SWS splitting's accuracy for Example 3.1, we solve the equation over small steps $t_f = \Delta t$. The accuracy of the error due to splitting is determined by the order, under Taylor expansion, to which the solutions agree.

The second order splitting accuracy in time for the SWS splitting is shown through the following Taylor expansion,

$$\begin{aligned}
\left|u(\Delta t) - u_{symm}(\Delta t)\right| &= \left|\left(-3e^{-2\Delta t} + 4e^{-\Delta t}\right) - \left(-\frac{3}{4}e^{-3\Delta t} - \frac{3}{4}e^{-2\Delta t} + \frac{7}{4}e^{-\Delta t} + \frac{3}{4}\right)\right| \\
&= \Bigg|\left[-3\left(1 - 2\Delta t + 2\Delta t^2 + O(\Delta t^3)\right) + 4\left(1 - \Delta t + \frac{1}{2!}\Delta t^2 + O(\Delta t^3)\right)\right] \\
&\quad -\left[-\frac{3}{4}\left(1 - 3\Delta t + \frac{9}{2}\Delta t^2 + O(\Delta t^3)\right) - \frac{3}{4}\left(1 - 2\Delta t + 2\Delta t^2 + O(\Delta t^3)\right)\right. \\
&\quad \left.+\frac{7}{4}\left(1 - \Delta t + \frac{1}{2!}\Delta t^2 + O(\Delta t^3)\right) + \frac{3}{4}\right]\Bigg| \\
&= \left|\left(1 + 2\Delta t - 4\Delta t^2 + O(\Delta t^3)\right) - \left(1 + 2\Delta t - 4\Delta t^2 + O(\Delta t^3)\right)\right| \\
&= O(\Delta t^3).
\end{aligned}$$

It can be easily observed that SWS splitting preserve second order accuracy.

3.1.7. Numerical Results for Linear ODE

In this section, we demonstrate the numerical solutions of the Example 3.1 by appliying the classical operator splitting methods.

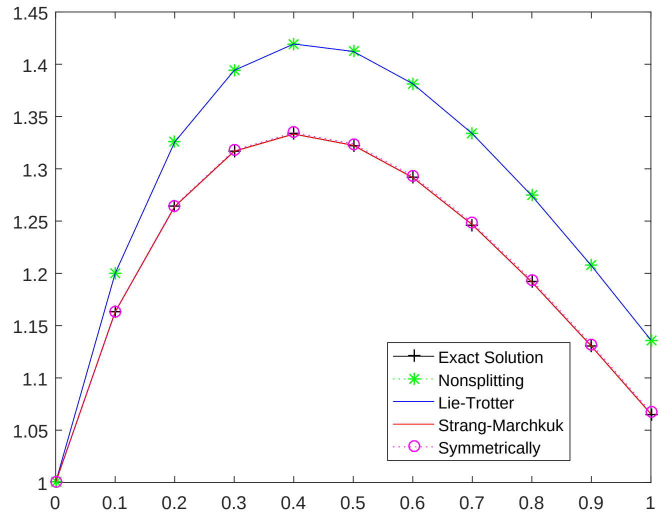


Figure 3.1. Comparison of the approximate solutions and the exact solution (3.3) of Example 3.1 for $\Delta t = 0.1$ on the time interval $t \in [0, 1]$.

Figure 3.1 shows the comparison of the different splitting methods, nonsplitting and exact solution (3.3) for Example 3.1 with $\Delta t = 0.1$ on the time interval $t \in [0, 1]$. We deduce that Strang-Marchuk and SWS splitting are quite close to the exact solution. Moreover, we see the overlap of the Lie-Trotter and nonsplitting.

Figure 3.2 shows the comparison of the different splitting methods, nonsplitting and exact solution (3.3) for Example 3.1 with $\Delta t = 0.01$ on the time interval $t \in [0, 1]$. We deduce that all of different splitting methods and nonsplitting are close to each other. Also, we can conclude that when time step size decreases, they are getting closer to the exact solution.

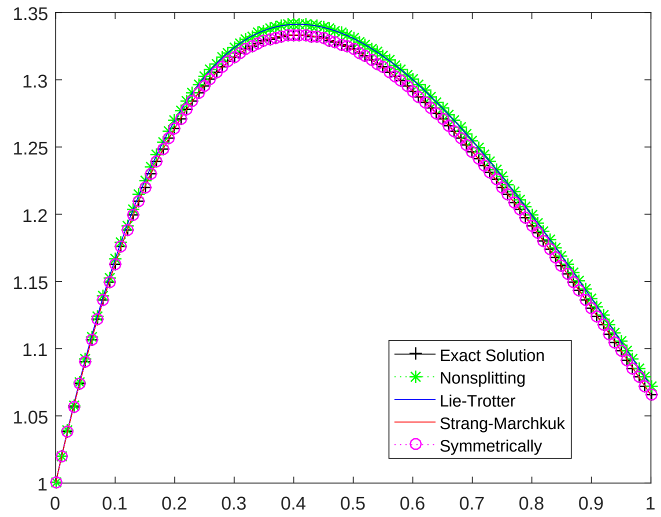


Figure 3.2. Comparison of the approximate solutions and the exact solution (3.3) of Example 3.1 for $\Delta t = 0.01$ on the time interval $t \in [0, 1]$.

	Error l^1	Error l^2	Error l^∞
Nonsplitting	0.6711	0.0221	$8.3052e-04$
Lie-Trotter Splitting	0.6711	0.0221	$8.3052e-04$
Strang-Marchuk Splitting	$6.3085e-06$	$2.3031e-07$	$1.0981e-08$
Symmetrically Weighted Sequential Splitting	$1.2636e-04$	$4.3461e-06$	$1.8721e-07$

Table 3.1. The errors of different splitting methods and nonsplitting for Example 3.1 with $\Delta t = 0.001$.

Table 3.1 shows comparison the local splitting errors of the different splitting methods and nonsplitting via l^1, l^2 and l^∞ norm. From the table, we deduce that the error of Lie-Trotter splitting is the same as that of nonsplitting. In addition, the error results revealed that the SWS splitting gave smaller error for Example 3.1.

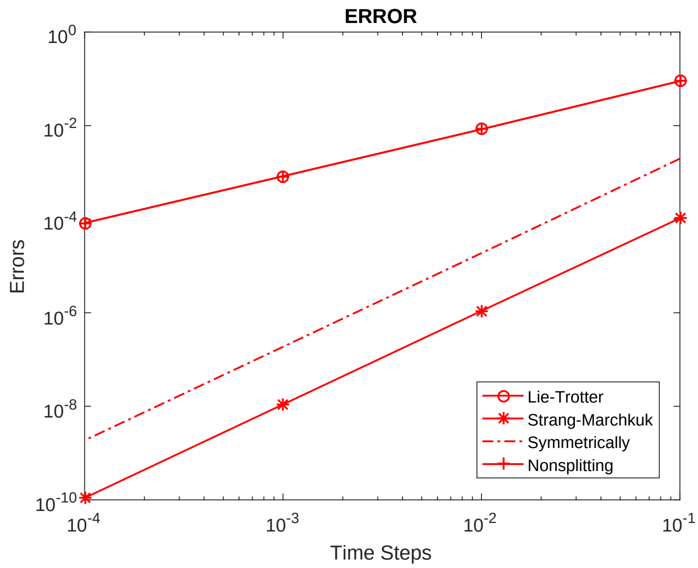


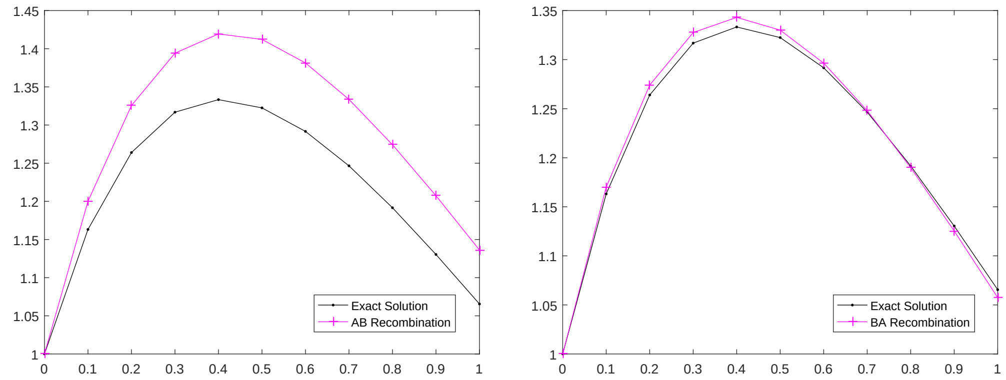
Figure 3.3. Comparison of the maximum errors of the approximate solutions for Example 3.1 for different Δt values.

Figure 3.3 represents comparison between the local splitting error of the different splitting methods and nonsplitting for various Δt values by using l^∞ norm. We deduce that Lie-Trotter splitting and nonsplitting are coincident.

Lie-Trotter Splitting	AB Recombination	BA Recombination
Error l^1	0.6711	0.0756
Error l^2	0.0221	0.0028
Error l^∞	$8.3052e-04$	$1.3161e-04$

Table 3.2. The errors of different splitting recombinations of the Lie-Trotter splitting for Example 3.1 with $\Delta t = 0.001$.

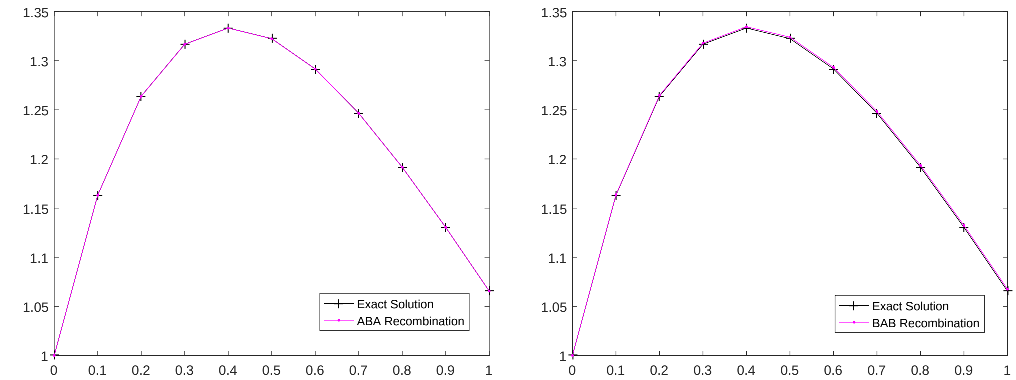
Table 3.2 shows comparison local splitting error of the 'AB' and 'BA' splitting recombinations by using l^1, l^2 and l^∞ norm. From Table 3.2, we deduce that the splitting error of 'BA' splitting recombination is less than the 'AB' splitting recombination.



(a) 'AB' Splitting - Exact Solution (b) 'BA' Splitting - Exact Solution

Figure 3.4. Comparison of the 'AB' and 'BA' splitting recombinations solutions and the exact solution (3.3) of Example 3.1 for $\Delta t = 0.1$.

Figure 3.4 shows comparison between the 'AB' and 'BA' splitting recombinations by Lie-Trotter splitting and the exact solution (3.3) for $\Delta t = 0.1$ on the time interval $t \in [0, 1]$. It follows that the 'BA' splitting recombination is closer to the exact solution than 'AB' splitting recombination.



(a) 'ABA' Splitting - Exact Solution (b) 'BAB' Splitting - Exact Solution

Figure 3.5. Comparison of the 'ABA' and 'BAB' splitting recombinations solutions and the exact solution (3.3) of Example 3.1 for $\Delta t = 0.1$.

Figure 3.5 shows the comparison between the 'ABA' and 'BAB' splitting recombinations by Strang-Marchuk splitting and the exact solution (3.3) for $\Delta t = 0.1$ on the time interval $t \in [0, 1]$. It follows that both the 'ABA' splitting recombination and 'BAB' splitting recombination are quite close to the exact solution.

Strang-Marchuk Splitting	ABA Recombination	BAB Recombination
Error l^1	$6.3085e-06$	$1.2636e-04$
Error l^2	$2.3031e-07$	$4.3461e-06$
Error l^∞	$1.0981e-08$	$1.8721e-07$

Table 3.3. The errors of different splitting recombinations of the Strang-Marchuk splitting for Example 3.1 with $\Delta t = 0.001$.

Table 3.3 shows the comparison local splitting error of the 'ABA' and 'BAB' splitting recombinations by using l^1, l^2 and l^∞ norm. From Table 3.3, we deduce that the splitting error of 'ABA' splitting recombination is less than the 'BAB' splitting recombination.

Time Step Size	Maximum Error	Order of Accuracy
$\Delta t = 0.1$	0.0898	
$\Delta t = 0.01$	0.0084	1.0306
$\Delta t = 0.001$	$8.3052e-04$	1.0031
$\Delta t = 0.0001$	$8.2993e-05$	1.0003

Table 3.4. Maximum Error of Lie-Trotter Splitting for Example 3.1 with Different Δt Values.

Table 3.4 shows the maximum error of Lie-Trotter splitting for Example 3.1. We conclude that the order of the Lie-Trotter splitting converges to 1.

Time Step Size	Maximum Error	Order of Accuracy
$\Delta t = 0.1$	$1.0477e-04$	
$\Delta t = 0.01$	$1.0934e-06$	1.9814
$\Delta t = 0.001$	$1.0981e-08$	1.9981
$\Delta t = 0.0001$	$1.0987e-10$	1.9998

Table 3.5. Maximum Error of Strang-Marchuk Splitting for Example 3.1 with Different Δt Values.

Table 3.5 shows maximum error of Strang-Marchuk splitting for Example 3.1. We can observe that the order of the Strang-Marchuk splitting converges to almost 2.

Time Step Size	Maximum Error	Order of Accuracy
$\Delta t = 0.1$	0.0020	
$\Delta t = 0.01$	$1.8809e - 05$	2.0199
$\Delta t = 0.001$	$1.8721e - 07$	2.0020
$\Delta t = 0.0001$	$1.8713e - 09$	2.0002

Table 3.6. Maximum Error of SWS Splitting for Example 3.1 with Different Δt Values.

Table 3.6 shows maximum error of SWS splitting for Example 3.1. We can conclude that the order of the SWS splitting converges to 2.

From Table 3.4, Table 3.5 and Table 3.6 we conclude the approximate solution converges to the exact solution, as the time step size decreases.

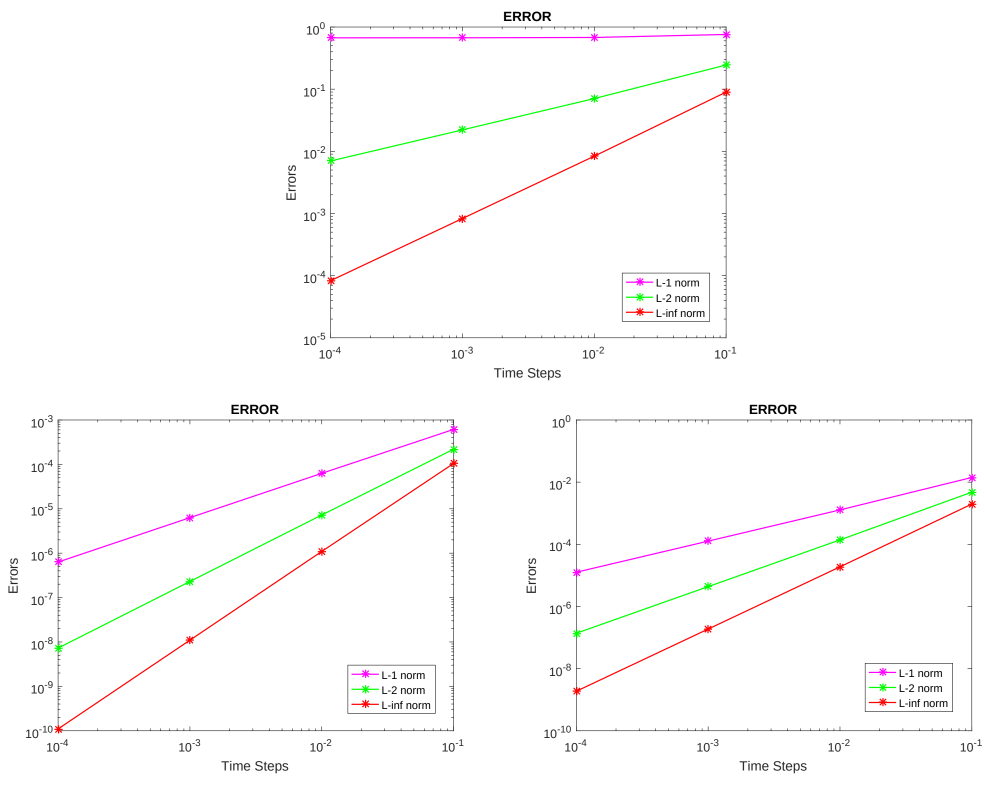


Figure 3.6. Comparison of the local splitting errors of the classical operator splitting methods for Example 3.1 for different Δt values.

Figure 3.6 illustrates the local splitting errors of the Lie-Trotter, Strang-Marchuk and SWS splittings, respectively, for the relatively large and small splitting time steps by using l^1, l^2 and l^∞ norm.

3.2. Nonlinear Demonstration

We shall consider the following normal form of a nonlinear ODE

$$u' = p(t)u + q(t)u^n, \qquad (3.10)$$

where both $p(t)$ and $q(t)$ are continuous functions and n is positive real number. Equation (3.10) is called a Bernoulli differential.

We begin with the following example for this type of nonlinear ODE:

Example 3.2 We consider the following first order nonlinear ODE

$$u' = -\frac{1}{3}u + e^t u^2, \quad u(0) = 1 \qquad (3.11)$$

This is a Bernoulli differential equation; we use the substitution

$$v = \frac{1}{u},$$

to get the first order linear equation

$$v' - \frac{1}{3}v = -e^t.$$

Multiplying both sides of the reduced equation by the integrating factor

$$\mu = e^{\int(-1/3)dt} = e^{-t/3},$$

leads to

$$\frac{d}{dt}(e^{-t/3}v(t)) = -e^{2t/3}.$$

Integrating the above equation with respect to t, we obtain

$$e^{-t/3}v(t) = -\int e^{2t/3}dt + c,$$
$$= -\frac{3}{2}e^{2t/3} + c.$$

Dividing the last equation through by $e^{-t/3}$, we obtain

$$v(t) = -\frac{3}{2}e^t + ce^{t/3}.$$

Since $v = \dfrac{1}{u}$, we get the general solution

$$u(t) = \frac{1}{-\frac{3}{2}e^t + ce^{t/3}}.$$

Applying the initial condition $u(0) = 1$ gives $c = \dfrac{5}{2}$, and thus the solution becomes

$$u(t) = -\frac{2}{3e^t - 5e^{t/3}}. \tag{3.12}$$

3.2.1. Lie-Trotter Splitting For Nonlinear ODE

We will consider the Lie-Trotter splitting of (3.11) into split equation 'A',

$$u_1' = -\frac{1}{3}u_1, \quad u_1(0) = 1$$

and the split equation 'B',

$$u_2' = e^t u_2^2, \quad u_2(0) = u_1(t)$$

and recombine their solutions in sequential scheme designed to preserve a certain level of accuracy in time.

For example, the 'AB 'recombination scheme results in the piecewise solution.

$$u_{AB}(t) = \begin{cases} e^{-t/3}, & 0 \le t \le t^n \\ -\frac{1}{e^t - e^{t/3} - 1}, & 0 \le t \le t^n \end{cases} \tag{3.13}$$

while reordering the split equations to the 'BA 'recombination scheme results

$$u_{BA}(t) = \begin{cases} -\frac{1}{e^t - 2}, & 0 \le t \le t^n \\ -\frac{1}{e^{4t/3} - 2e^{t/3}}, & 0 \le t \le t^n \end{cases} \tag{3.14}$$

Although these piecewise functions (3.13) and (3.14) are not differentiable, they are continuous. It is obvious that the two recombination schemes (3.13) and (3.14) are different, however, they both preserve a first order approximation in time to the exact solution of the equation of (3.11).

3.2.2. Accuracy of Nonlinear ODE for Lie-Trotter Splitting

To demonstrate the Lie-Trotter splitting's accuracy for Example 3.2, we solve the equation over small steps $t_f = \Delta t$. The accuracy of the error due to splitting is determined by the order, under Taylor expansion, to which the solutions agree.

The first order splitting accuracy in time for the '*AB*' recombination is shown through the following Taylor expansion,

$$
\begin{aligned}
\left|u(\Delta t) - u_{AB}(\Delta t)\right| &= \left|\left(-\frac{2}{3e^{\Delta t} - 5e^{\Delta t/3}}\right) - \left(-\frac{1}{e^{\Delta t} - e^{\Delta t/3} - 1}\right)\right| \\
&= \left|\left[-\frac{2}{3\left(1 + \Delta t + \frac{1}{2!}\Delta t^2 + O(\Delta t^3)\right) - 5\left(1 + \frac{1}{3}\Delta t + \frac{1}{18}\Delta t^2 + O(\Delta t^3)\right)}\right]\right. \\
&\quad \left. - \left[-\frac{1}{\left(1 + \Delta t + \frac{1}{2!}\Delta t^2 + O(\Delta t^3)\right) - \left(1 + \frac{1}{3}\Delta t + \frac{1}{18}\Delta t^2 + O(\Delta t^3)\right) - 1}\right]\right| \\
&= \left|\left(1 + \frac{2}{3}\Delta t + \frac{19}{18}\Delta t^2 + O(\Delta t^3)\right) - \left(1 + \frac{2}{3}\Delta t + \frac{8}{9}\Delta t^2 + O(\Delta t^3)\right)\right| \\
&= \left|\frac{1}{6}\Delta t^2 + O(\Delta t^3)\right| \\
&= O(\Delta t^2).
\end{aligned}
$$

Since the error between the exact and '*AB*' split solution is $O(\Delta t^2)$, the solutions agree up to order $O(\Delta t)$, which is thus the splitting accuracy of the '*AB*' recombination scheme.

The first order splitting accuracy in time of the '*BA*' recombination scheme is similary shown as

$$
\begin{aligned}
\left|u(\Delta t) - u_{BA}(\Delta t)\right| &= \left|\left(-\frac{2}{3e^{\Delta t} - 5e^{\Delta t/3}}\right) - \left(-\frac{1}{e^{4\Delta t/3} - 2e^{\Delta t/3}}\right)\right| \\
&= \left|\left[-\frac{2}{3\left(1 + \Delta t + \frac{1}{2!}\Delta t^2 + O(\Delta t^3)\right) - 5\left(1 + \frac{1}{3}\Delta t + \frac{1}{18}\Delta t^2 + O(\Delta t^3)\right)}\right]\right. \\
&\quad \left. - \left[-\frac{1}{\left(1 + \frac{4}{3}\Delta t + \frac{8}{9}\Delta t^2 + O(\Delta t^3)\right) - 2\left(1 + \frac{1}{3}\Delta t + \frac{1}{18}\Delta t^2 + O(\Delta t^3)\right)}\right]\right| \\
&= \left|\left(1 + \frac{2}{3}\Delta t + \frac{19}{18}\Delta t^2 + O(\Delta t^3)\right) - \left(1 + \frac{2}{3}\Delta t + \frac{11}{9}\Delta t^2 + O(\Delta t^3)\right)\right| \\
&= \left|-\frac{1}{6}\Delta t^2 + O(\Delta t^3)\right| \\
&= O(\Delta t^2).
\end{aligned}
$$

It is clear now that the splitting error for the two orderings of the first order recombination scheme is not the same but it is of the same order.

3.2.3. Strang-Marchuk Splitting For Nonlinear ODE

We will consider the Strang-Marchuk splitting of (3.11) into split equation 'A',

$$
u_1{}' = -\frac{u_1}{6}, \quad u_1(0) = 1
$$

and the split equation 'B',

$$
u_2{}' = e^t u_2^2, \quad u_2(0) = u_1(t)
$$

and again the split equation 'A',

$$
u_3{}' = -\frac{u_3}{6}, \quad u_3(0) = u_2(t)
$$

Then, recombine their solutions in sequential scheme designed to preserve a certain level of accuracy in time.

For instance, the direct 'ABA' recombination scheme results in the piecewise solution

$$u_{ABA}(t) = \begin{cases} e^{-t/6}, & 0 \leq t \leq t^{n+1/2} \\ -\frac{1}{e^t - e^{t/6} - 1}, & 0 \leq t \leq t^n \\ -\frac{1}{e^{7t/6} - e^{t/3} - e^{t/6}}, & t^{n+1/2} \leq t \leq t^n \end{cases} \tag{3.15}$$

while reordering the split equations to the 'BAB ' recombination scheme results

$$u_{BAB}(t) = \begin{cases} -\frac{2}{e^t - 3}, & 0 \leq t \leq t^{n+1/2} \\ -\frac{2}{e^{4t/3} - 3e^{t/3}}, & 0 \leq t \leq t^n \\ -\frac{2}{e^{4t/3} + e^t - 3e^{t/3} - 1}, & t^{n+1/2} \leq t \leq t^n \end{cases} \tag{3.16}$$

3.2.4. Accuracy of Nonlinear ODE for Strang-Marchuk Splitting

To demonstrate the Strang-Marchuk splitting's accuracy, we solve the equation over small steps $t_f = \Delta t$. The accuracy of the error due to splitting is determined by the order, under Taylor expansion, to which the solutions agree.

The second order accuracy in time for the 'ABA' recombination is shown through the following Taylor expansion,

$$\begin{aligned} \left|u(\Delta t) - u_{ABA}(\Delta t)\right| &= \left|\left(-\frac{2}{3e^{\Delta t} - 5e^{\Delta t/3}}\right) - \left(-\frac{1}{e^{7\Delta t/6} - e^{\Delta t/3} - e^{\Delta t/6}}\right)\right| \\ &= \left|\left[-\frac{2}{3\left(1 + \Delta t + \frac{1}{2!}\Delta t^2 + O(\Delta t^3)\right) - 5\left(1 + \frac{1}{3}\Delta t + \frac{1}{18}\Delta t^2 + O(\Delta t^3)\right)}\right]\right. \\ &\quad \left. - \left[-\frac{1}{\left(1 + \frac{7}{6}\Delta t + \frac{49}{72}\Delta t^2 + O(\Delta t^3)\right) - \left(1 + \frac{1}{3}\Delta t + \frac{1}{18}\Delta t^2 + O(\Delta t^3)\right) - \left(1 + \frac{1}{6}\Delta t + \frac{1}{72}\Delta t^2 + O(\Delta t^3)\right)}\right]\right| \\ &= \left|\left(1 + \frac{2}{3}\Delta t + \frac{19}{18}\Delta t^2 + O(\Delta t^3)\right) - \left(1 + \frac{2}{3}\Delta t + \frac{19}{18}\Delta t^2 + O(\Delta t^3)\right)\right| \\ &= O(\Delta t^3). \end{aligned}$$

Since the error between the exact and 'ABA' split solution is $O(\Delta t^2)$, the solutions agree up to order $O(\Delta t)$, which is thus the splitting accuracy of the 'ABA' recombination scheme.

The second order splitting accuracy in time of the 'BAB' recombination scheme is similary shown as

$$\begin{aligned}
\left|u(\Delta t) - u_{BAB}(\Delta t)\right| &= \left|\left(-\frac{2}{3e^{\Delta t} - 5e^{\Delta t/3}}\right) - \left(-\frac{2}{e^{4\Delta t/3} + e^{\Delta t} - 3e^{\Delta t/3} - 1}\right)\right| \\
&= \left|\left[-\frac{2}{3\left(1 + \Delta t + \frac{1}{2!}\Delta t^2 + O(\Delta t^3)\right) - 5\left(1 + \frac{1}{3}\Delta t + \frac{1}{18}\Delta t^2 + O(\Delta t^3)\right)}\right]\right. \\
&\quad \left. - \left[-\frac{2}{\left(1 + \frac{4}{3}\Delta t + \frac{8}{9}\Delta t^2 + O(\Delta t^3)\right) + \left(1 + \Delta t + \frac{1}{2!}\Delta t^2 + O(\Delta t^3)\right) - 3\left(1 + \frac{1}{3}\Delta t + \frac{1}{18}\Delta t^2 + O(\Delta t^3)\right) - 1}\right]\right| \\
&= \left|\left(1 + \frac{2}{3}\Delta t + \frac{19}{18}\Delta t^2 + O(\Delta t^3)\right) - \left(1 + \frac{2}{3}\Delta t + \frac{19}{18}\Delta t^2 + O(\Delta t^3)\right)\right| \\
&= O(\Delta t^3).
\end{aligned}$$

It is clear now that the splitting error for the two orderings of the second order recombination scheme is not the same but it is of the same order.

3.2.5. Symmetrically Weighted Sequential Splitting for Nonlinear ODE

We will consider a first order splitting of (3.11) into split equation 'A',

$$u_1' = -\frac{1}{3}u_1, \quad u_1(0) = 1$$

and the split equation 'B',

$$u_2' = e^t u_2^2, \quad u_2(0) = u_1(t)$$

and recombine their solutions in sequential scheme designed to preserve a certain level of accuracy in time.

For example, the result of 'AB' recombination scheme,

$$u_{AB}(t) = -\frac{1}{e^t - e^{t/3} - 1}$$

and the result of 'BA' recombination scheme is

$$u_{BA}(t) = -\frac{1}{e^{4t/3} - 2e^{t/3}}.$$

Then the split solution at the mesh points is defined as:

$$u_{sp}^{n+1} = \frac{u_{AB}(t^{n+1}) + u_{BA}(t^{n+1})}{2} \tag{3.17}$$

Thus, we can obtain the approximate solution as

$$u_{symm}(t) = -\frac{1}{2e^{t} - 2e^{t/3} - 2} - \frac{1}{2e^{4t/3} - 4e^{t/3}}. \tag{3.18}$$

3.2.6. Accuracy of Nonlinear ODE for Symmetrically Weighted Sequential Splitting

To show the SWS splitting's accuracy for Example 3.2, we solve the equation over small steps $t_f = \Delta t$. The accuracy of the error due to splitting is determined by the order, under Taylor expansion, to which the solutions agree.

The second order splitting accuracy in time for the SWS splitting is shown through the following Taylor expansion,

$$\begin{aligned}
\left|u(\Delta t) - u_{symm}(\Delta t)\right| &= \left|\left(-\frac{2}{3e^{\Delta t} - 5e^{\Delta t/3}}\right) - \left(-\frac{1}{2e^{\Delta t} - 2e^{\Delta t/3} - 2} - \frac{1}{2e^{4\Delta t/3} - 4e^{\Delta t/3}}\right)\right| \\
&= \Bigg|\left[-\frac{2}{3\left(1 + \Delta t + \frac{1}{2!}\Delta t^2 + O(\Delta t^3)\right) - 5\left(1 + \frac{1}{3}\Delta t + \frac{1}{18}\Delta t^2 + O(\Delta t^3)\right)}\right] \\
&\quad - \left[-\frac{1}{2\left(1 + \Delta t + \frac{1}{2!}\Delta t^2 + O(\Delta t^3)\right) - 2\left(1 + \frac{1}{3}\Delta t + \frac{1}{18}\Delta t^2 + O(\Delta t^3)\right) - 2}\right. \\
&\quad \left. - \frac{1}{2\left(1 + \frac{4}{3}\Delta t + \frac{8}{9}\Delta t^2 + O(\Delta t^3)\right) - 4\left(1 + \frac{1}{3}\Delta t + \frac{1}{18}\Delta t^2 + O(\Delta t^3)\right)}\right]\Bigg| \\
&= \left|\left(1 + \frac{2}{3}\Delta t + \frac{19}{18}\Delta t^2 + O(\Delta t^3)\right) - \left(1 + \frac{2}{3}\Delta t + \frac{19}{18}\Delta t^2 + O(\Delta t^3)\right)\right| \\
&= O(\Delta t^3).
\end{aligned}$$

Now, it is clear that SWS splitting is second order as well.

3.2.7. Numerical Results for Nonlinear ODE

In this part, we demonstrate the numerical solutions of the Example 3.2 by appliying the classical operator splitting methods.

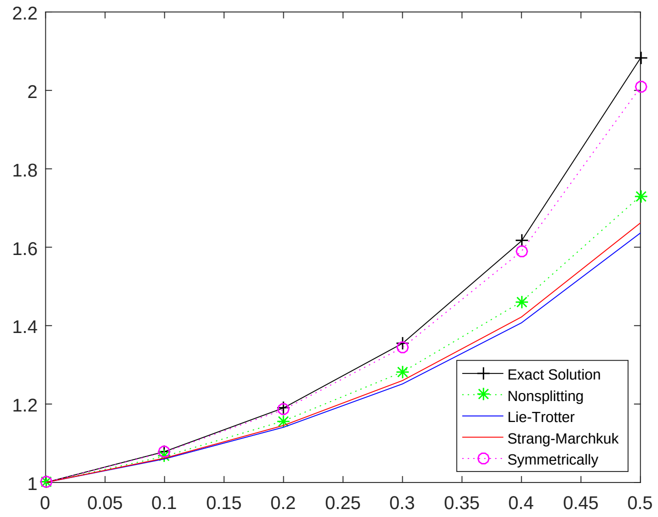


Figure 3.7. Comparison of the approximate solutions and the exact solution (3.12) of Example 3.2 for $\Delta t = 0.1$ on the time interval $t \in [0, 1/2]$.

Figure 3.7 represents the comparison of the different splitting methods, nonsplitting and exact solution (3.12) for Example 3.2 with $\Delta t = 0.1$ on the time interval $t \in [0, 1/2]$. We deduce that SWS splitting are quite close to the exact solution.

Figure 3.8 represents the comparison of the different splitting methods, nonsplitting and exact solution (3.12) for Example 3.2 with $\Delta t = 0.01$ on the time interval $t \in [0, 1/2]$. We deduce that all of different splitting methods and nonsplitting are close to each other.

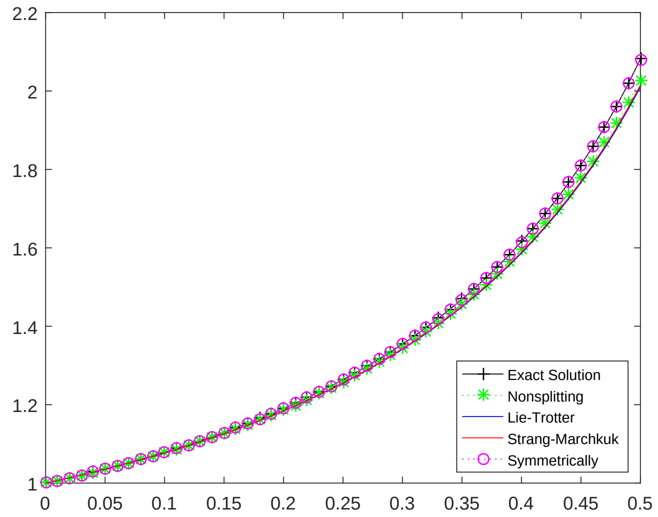


Figure 3.8. Comparison of the approximate solutions and the exact solution (3.12) of Example 3.2 for $\Delta t = 0.01$ on the time interval $t \in [0, 1/2]$.

	Error l^1	Error l^2	Error l^∞
Nonsplitting	0.6069	0.0412	0.0057
Lie-Trotter Splitting	0.8593	0.0571	0.0077
Strang-Marchuk Splitting	0.0011	$7.7368e-05$	$1.1736e-05$
Symmetrically Weighted Sequential Splitting	0.0010	$7.6222e-05$	$1.1562e-05$

Table 3.7. The errors of different splitting methods and nonsplitting for Example 3.2 with $\Delta t = 0.001$.

Table 3.7 shows comparison the local splitting errors of the different splitting methods and nonsplitting via l^1, l^2 and l^∞ norm. From the table, we deduce that the errors of Strang-Marchuk splitting and SWS splitting are very close to each other. However, the error results revealed that the SWS splitting gave smaller error for Example 3.2.

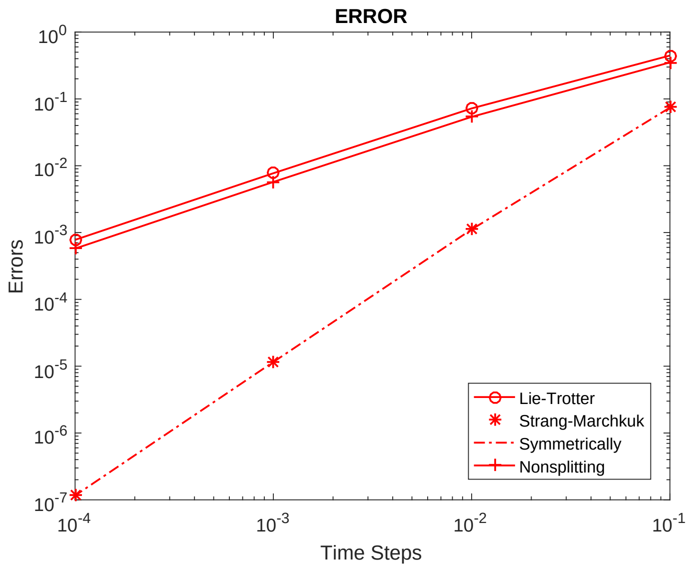


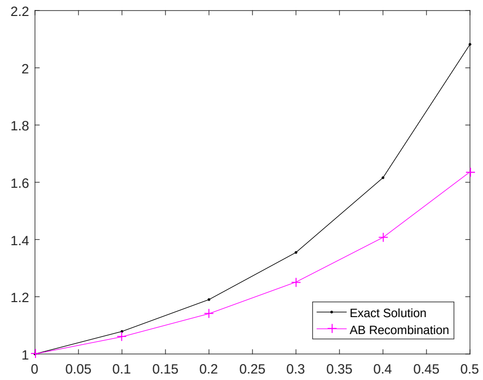
Figure 3.9. Comparison of the maximum errors of the approximate solutions for Example 3.2 for differen Δt values.

Figure 3.9 shows comparison between the local splitting error of the different splitting methods and nonsplitting with various Δt values by using l^∞ norm. It seems like Strang-Marchuk and SWS splitting are coincident.

Lie-Trotter Splitting	AB Recombination	BA Recombination
Error l^1	0.8593	0.7331
Error l^2	0.0571	0.0492
Error l^∞	0.0077	0.0067

Table 3.8. The errors of different splitting recombinations of the Lie-Trotter splitting for Example 3.2 with $\Delta t = 0.001$.

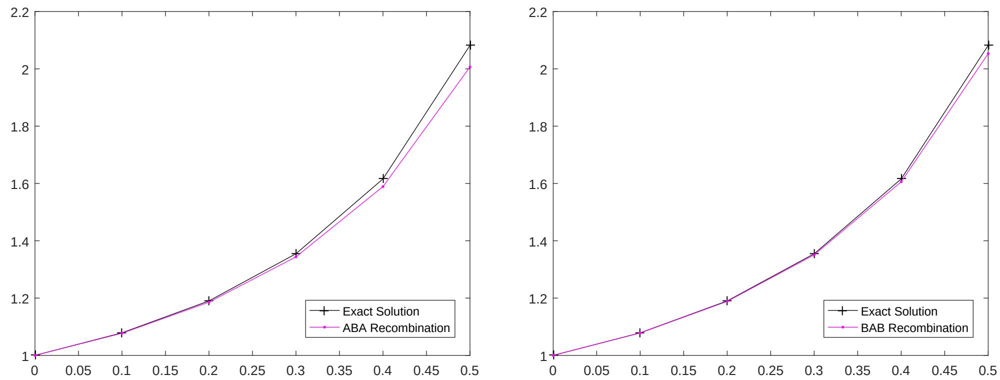
Table 3.8 shows comparison local splitting error of the '*AB*' and '*BA*' splitting recombinations by using l^1, l^2 and l^∞ norm. From Table 3.8, we deduce that the splitting error of '*BA*' splitting recombination is less than the '*AB*' splitting recombination.



(a) 'AB' Splitting - Exact Solution (b) 'BA' Splitting - Exact Solution

Figure 3.10. Comparison of the 'AB' and 'BA' splitting recombinations solutions and the exact solution (3.12) of Example 3.2 for $\Delta t = 0.1$.

Figure 3.10 shows comparison between the 'AB' and 'BA' splitting recombinations by Lie-Trotter splitting and the exact solution (3.12) for $\Delta t = 0.1$ on the time interval $t \in [0, 1/2]$. It follows that the 'BA' splitting recombination is closer to the exact solution than 'AB' splitting recombination.



(a) ABA Splitting - Exact Solution (b) BAB Splitting - Exact Solution

Figure 3.11. Comparison of the 'ABA' and 'BAB' splitting recombinations solutions and the exact solution (3.12) of Example 3.2 for $\Delta t = 0.1$.

Figure 3.11 shows the comparison between the 'ABA' and 'BAB' splitting recombinations by Strang-Marchuk splitting and the exact solution (3.12) for $\Delta t = 0.1$ on the time interval $t \in [0, 1/2]$. It follows that the 'BAB' splitting recombination is closer to the exact solution than 'ABA' splitting recombination.

Strang-Marchuk Splitting	ABA Recombination	BAB Recombination
Error l^1	0.0011	$3.3929e-04$
Error l^2	$7.7368e-05$	$2.4337e-05$
Error l^∞	$1.1736e-05$	$3.6344e-06$

Table 3.9. The errors of different splitting recombinations of the Strang-Marchuk splitting for Example 3.2 with $\Delta t = 0.001$.

Table 3.9 represents the comparison local splitting error of the '*ABA*' and '*BAB*' splitting recombinations by using l^1, l^2 and l^∞ norm. From Table 3.9, we deduce that the splitting error of '*BAB*' splitting recombination is less than the '*ABA*' splitting recombination.

Time Step Size	Maximum Error	Order of Accuracy
$\Delta t = 0.1$	0.4457	
$\Delta t = 0.01$	0.0720	0.7915
$\Delta t = 0.001$	0.0077	0.9699
$\Delta t = 0.0001$	$7.7767e-04$	0.9968

Table 3.10. Maximum Error of Lie-Trotter Splitting of Noninear ODE with Different Δt Values.

Table 3.10 shows maximum error of Lie-Trotter splitting for Example 3.2. We can see that the order of the Lie-Trotter splitting converges to almost 1.

Time Step Size	Maximum Error	Order of Accuracy
$\Delta t = 0.1$	0.0754	
$\Delta t = 0.01$	0.0011	1.8238
$\Delta t = 0.001$	$1.1736e-05$	1.9839
$\Delta t = 0.0001$	$1.1779e-07$	1.9984

Table 3.11. Maximum Error of Strang-Marchuk Splitting of Nonlinear ODE with Different Δt Values.

Table 3.11 shows maximum error of Strang-Marchuk splitting for Example 3.2. We can observe that the order of the Strang-Marchkuk splitting converges to almost 2.

Time Step Size	Maximum Error	Order of Accuracy
$\Delta t = 0.1$	0.0741	
$\Delta t = 0.01$	0.0011	1.8231
$\Delta t = 0.001$	$1.1562e - 05$	1.9837
$\Delta t = 0.0001$	$1.1605e - 07$	1.9984

Table 3.12. Maximum Error of Symmetrically Weighted Sequential Splitting of Nonlinear ODE with Different Δt Values.

Table 3.12 shows maximum error of SWS splitting for Example 3.2. We can conclude that the order of the SWS splitting converges to 2.

From Table 3.10, Table 3.11 and Table 3.12, we conclude the approximate solution converges to the exact solution, as the time step size decreases.

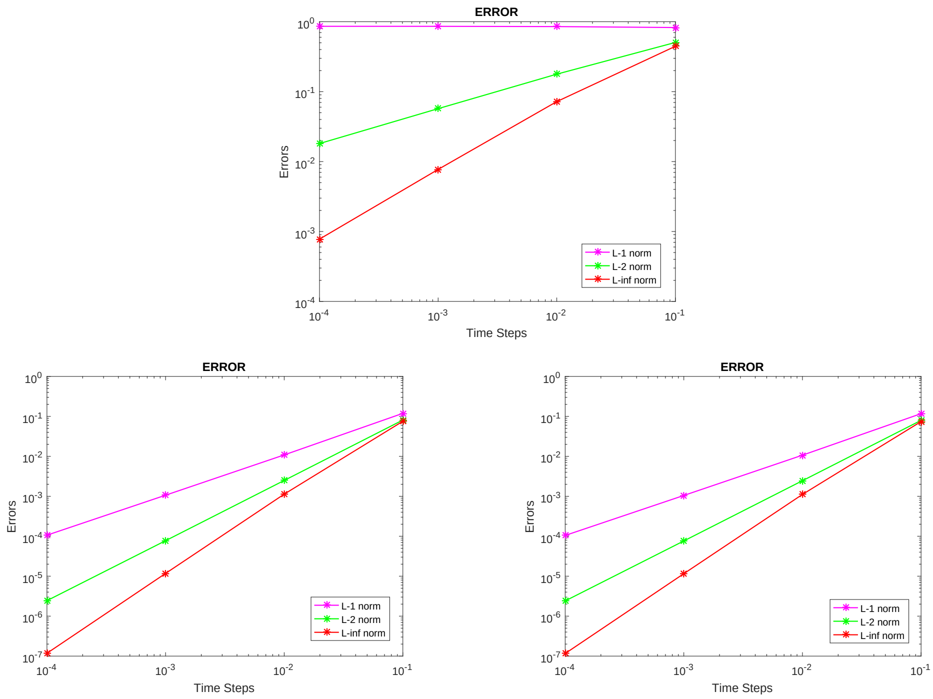


Figure 3.12. Comparison of the local splitting errors of the classical operator splitting methods for Example 3.2 for different Δt values.

Figure 3.12 illustrate the local splitting errors of the Lie-Trotter, Strang-Marchuk and SWS splittings, respectively, for the relatively large and small splitting time steps by using l^1, l^2 and l^∞ norm.

CHAPTER 4

NUMERICAL CONSIDERATIONS

In this chapter, we consider the classical operator splitting methods for nonlinear differential equations with linear and nonlinear operators. For this purpose, firstly, we prove accuracy of these methods for nonlinear differential equations by using local splitting error. Next, we confirm two numerical examples which is consist of a kinetic model of phage infection and the Newell - Whitehead - Segel equation to show the effectiveness of the classical operator splitting methods.

4.1. Accuracy of the Operator Splitting Methods for Nonlinear Differential Equations

In this section, we derive the local splitting error of the operator splitting methods for nonlinear differential equations.

Given a nonlinear differential equation which can be nontrivially written as

$$\frac{du}{dt} = Au + B(u) \tag{4.1}$$

where A and B are linear and nonlinear operators upon u, respectively.

Suppose we have a nonlinear differential equation which can be seperated into linear and nonlinear operators and has been discretized in space to the form

$$\frac{du^n}{dt} = Au^n + B(u^n),$$

where A is a $m \times m$ matrix discretization of operator A and $B(u^n)$ is a m dimensional vector discretization of operator $B(u)$.

Using the Taylor expansion at time step $t^{n+1} = t^n + \Delta t$, the solution to the full problem, $u_F^{n+1} = u_F(t^{n+1})$ becomes

$$\begin{aligned}
u_F^{n+1} &= u^n + \Delta t\frac{du^n}{dt} + \frac{\Delta t^2}{2!}\frac{d^2u^n}{dt^2} + O(\Delta t^3) \\
&= u^n + \Delta t(Au^n + B(u^n)) + \frac{\Delta t^2}{2!}\frac{d(Au^n + B(u^n))}{dt} + O(\Delta t^3) \\
&= u^n + \Delta t(Au^n + B(u^n)) + \frac{\Delta t^2}{2!}\left(A\frac{du^n}{dt} + \frac{dB(u^n)}{du^n}\frac{du^n}{dt}\right) + O(\Delta t^3) \\
&= u^n + \Delta t(Au^n + B(u^n)) \\
&\quad + \frac{\Delta t^2}{2!}\left(A^2u^n + AB(u^n) + \frac{dB(u^n)}{du^n}Au^n + \frac{dB(u^n)}{du^n}B(u^n)\right) + O(\Delta t^3).
\end{aligned}$$

4.1.1. Accuracy of the Lie - Trotter Splitting

To show that the local splitting error of the Lie-Trotter splitting, we split the general form (4.1) as follows

$$\frac{du}{dt} = Au, \quad u(0) = u_0 \quad t \in [t^n, t^{n+1}] \tag{4.2}$$

$$\frac{dv}{dt} = B(v), \quad v(0) = u(t^n) \quad t \in [t^n, t^{n+1}] \tag{4.3}$$

where u_0 is an original initial condition.

The 'AB' recombination of the split solution gives the following approximation by Taylor expansion

$$\begin{aligned}
v^{n+1} &= u^{n+1} + \Delta t\frac{du^{n+1}}{dt} + \frac{\Delta t^2}{2!}\frac{d^2u^{n+1}}{dt^2} + O(\Delta t^3) \\
&= u^{n+1} + \Delta t B(u^{n+1}) + \frac{\Delta t^2}{2!}\frac{dB(u^{n+1})}{dt} + O(\Delta t^3) \\
&= e^{A\Delta t}u^n + \Delta t B(e^{A\Delta t}u^n) + \frac{\Delta t^2}{2!}\frac{dB(e^{A\Delta t}u^n)}{dt} + O(\Delta t^3) \\
&= \left(u^n + \Delta t Au^n + \frac{\Delta t^2}{2!}A^2u^n + O(\Delta t^3)\right) + \Delta t B(u^n + \Delta t Au^n + O(\Delta t^2)) \\
&\quad + \frac{\Delta t^2}{2!}\frac{dB((u^n + O(\Delta t))}{dt} + O(\Delta t^3) \\
&= u^n + \Delta t Au^n + \frac{\Delta t^2}{2!}A^2u^n + \Delta t\left(B(u^n) + \Delta t\frac{dB(u^n)}{du^n}Au^n\right) + \frac{\Delta t^2}{2!}\frac{dB(u^n)}{dt} + O(\Delta t^3) \\
&= u^n + \Delta t(Au^n + B(u^n)) + \frac{\Delta t^2}{2!}\left(A^2u^n + 2\frac{dB(u^n)}{du^n}Au^n + \frac{dB(u^n)}{du^n}B(u^n)\right) + O(\Delta t^3).
\end{aligned}$$

Thus,

$$\begin{aligned}
\left|u_F^{n+1} - v^{n+1}\right| &= \Bigg\|\Bigg[u^n + \Delta t(Au^n + B(u^n)) \\
&\quad + \frac{\Delta t^2}{2!}\left(A^2u^n + AB(u^n) + \frac{dB(u^n)}{du^n}Au^n + \frac{dB(u^n)}{du^n}B(u^n)\right) + O(\Delta t^3)\Bigg] \\
&\quad - \Bigg[u^n + \Delta t(Au^n + B(u^n)) \\
&\quad + \frac{\Delta t^2}{2!}(A^2u^n + 2\frac{dB(u^n)}{du^n}Au^n + \frac{dB(u^n)}{du^n}B(u^n)) + O(\Delta t^3)\Bigg]\Bigg\| \\
&= \left|\frac{\Delta t^2}{2!}\left(AB(u^n) - \frac{dB(u^n)}{du^n}Au^n\right) + O(\Delta t^3)\right| \\
&= O(\Delta t^2).
\end{aligned}$$

The 'BA' recombination of the split solution gives the following approximation by Taylor expansion

$$\begin{aligned}
v^{n+1} &= e^{\Delta tA}u^{n+1} \\
&= \left(I + \Delta tA + \frac{\Delta t^2}{2!}A^2 + O(\Delta t^3)\right)\left(u^n + \Delta t\frac{du^n}{dt} + \frac{\Delta t^2}{2!}\frac{d^2u^n}{dt^2} + O(\Delta t^3)\right) \\
&= \left(I + \Delta tA + \frac{\Delta t^2}{2!}A^2 + O(\Delta t^3)\right)\left(u^n + \Delta tB(u^n) + \frac{\Delta t^2}{2!}\frac{dB(u^n)}{dt} + O(\Delta t^3)\right) \\
&= u^n + \Delta t(Au^n + B(u^n)) + \frac{\Delta t^2}{2!}\left(A^2u^n + 2AB(u^n) + \frac{dB(u^n)}{du^n}B(u^n)\right) + O(\Delta t^3).
\end{aligned}$$

Thus,

$$\begin{aligned}
\left|u_F^{n+1} - v^{n+1}\right| &= \Bigg\|\Bigg[u^n + \Delta t(Au^n + B(u^n)) \\
&\quad + \frac{\Delta t^2}{2!}(A^2u^n + AB(u^n) + \frac{dB(u^n)}{du^n}Au^n + \frac{dB(u^n)}{du^n}B(u^n)) + O(\Delta t^3)] \\
&\quad - \Bigg[u^n + \Delta t(Au^n + B(u^n)) \\
&\quad + \frac{\Delta t^2}{2}(A^2u^n + 2AB(u^n) + \frac{dB(u^n)}{du^n}B(u^n)) + O(\Delta t^3)\Bigg]\Bigg\| \\
&= \left|\frac{\Delta t^2}{2}\left(-AB(u^n) + \frac{dB(u^n)}{du^n}Au^n\right) + O(\Delta t^3)\right| \\
&= O(\Delta t^2).
\end{aligned}$$

Hence, the computations for both split orderings gives first order accuracy in time.

4.1.2. Accuracy of the Strang-Marchuk Splitting

To show that the local splitting error of the Lie-Trotter splitting, we split the general form (4.1) as follows

$$\frac{du}{dt} = Au, \quad u(0) = u_0 \quad t \in [t^n, t^{n+1/2}] \tag{4.4}$$

$$\frac{dv}{dt} = B(v), \quad v(0) = u(t^n) \quad t \in [t^n, t^{n+1}] \tag{4.5}$$

$$\frac{dw}{dt} = Aw, \quad w(0) = v(t^n) \quad t \in [t^{n+1/2}, t^{n+1}] \tag{4.6}$$

where u_0 is an original initial condition.

The 'ABA' recombination of the split solution gives the following approximation by Taylor expansions,

$$\begin{aligned}
w^{n+1} &= e^{\Delta tA} w^{n+1/2} \\
&= \left(I + \Delta tA + \frac{\Delta t^2}{2!}A^2 + O(\Delta t^3)\right)\left(w^{n+1/2} + \frac{\Delta t}{2}\frac{dw^{n+1/2}}{dt} + \frac{\Delta t^2}{8}\frac{d^2w^{n+1/2}}{dt^2} + O(\Delta t^3)\right) \\
&= \left(I + \Delta tA + \frac{\Delta t^2}{2!}A^2 + O(\Delta t^3)\right)\left(v^{n+1} + \frac{\Delta t}{2}B(v^{n+1}) + \frac{\Delta t^2}{8}\frac{dB(v^{n+1})}{dt} + O(\Delta t^3)\right) \\
&= v^{n+1} + \frac{\Delta t}{2}B(v^{n+1}) + \frac{\Delta t^2}{8}\frac{dB(v^{n+1})}{dt} + \Delta tAv^{n+1} + \frac{\Delta t^2}{2}AB(v^{n+1}) + \frac{\Delta t^2}{2!}A^2v^{n+1} + O(\Delta t^3) \\
&= e^{\Delta tA}u^{n+1/2} + \frac{\Delta t}{2}B(e^{\Delta tA}u^{n+1/2}) + \frac{\Delta t^2}{8}\frac{dB(e^{\Delta tA}u^{n+1/2})}{dt} + \Delta tA\left(e^{\Delta tA}u^{n+1/2}\right) \\
&\quad + \frac{\Delta t^2}{2}AB(e^{\Delta tA}u^{n+1/2}) + \frac{\Delta t^2}{2!}A^2(e^{\Delta tA}u^{n+1/2}) + O(\Delta t^3) \\
&= \left(I + \Delta tA + \frac{\Delta t^2}{2!}A^2 + O(\Delta t^3)\right)\left(u^n + \frac{\Delta t}{2}\frac{du^n}{dt} + \frac{\Delta t^2}{8}\frac{d^2u^n}{dt^2} + O(\Delta t^3)\right) \\
&\quad + \frac{\Delta t}{2}B\left(\left(I + \Delta tA + O(\Delta t^2)\right)\left(u^n + \frac{\Delta t}{2}\frac{du^n}{dt} + O(\Delta t^2)\right)\right) + \frac{\Delta t^2}{8}\frac{dB\left((I + O(\Delta t))\,(u^n + O(\Delta t))\right)}{dt} \\
&\quad + \Delta tA\left((I + \Delta tA + O(\Delta t^2))\left(u^n + \frac{\Delta t}{2}\frac{du^n}{dt} + O(\Delta t^2)\right)\right) + \frac{\Delta t^2}{2}AB\left((I + O(\Delta t))\,(u^n + O(\Delta t))\right) \\
&\quad + \frac{\Delta t^2}{2}A^2\left((I + O(\Delta t))\,(u^n + O(\Delta t)) + O(\Delta t^3)\right) \\
&= u^n + \frac{\Delta t}{2}B(u^n) + \frac{\Delta t^2}{8}\frac{dB(u^n)}{dt} + \Delta tAu^n + \frac{\Delta t^2}{2}AB(u^n) + \frac{\Delta t^2}{2!}A^2u^n + \frac{\Delta t^2}{2}AB(u^n) + \frac{\Delta t^2}{2}Au^n \\
&\quad + \frac{\Delta t}{2}\left(B(u^n) + \frac{\Delta t}{2}\frac{dB(u^n)}{dt} + \Delta tAB(u^n)\right) + \frac{\Delta t^2}{8}\frac{dB(u^n)}{dt} + \Delta tA\left(u^n + \frac{\Delta t}{2}B(u^n) + \Delta tAu^n\right) + O(\Delta t^3) \\
&= u^n + \Delta t(Au^n + B(u^n)) + \frac{\Delta t^2}{2}\left(A^2u^n + AB(u^n) + \frac{dB(u^n)}{du^n}Au^n + \frac{dB(u^n)}{du^n}B(u^n)\right) + O(\Delta t^3)
\end{aligned}$$

Thus,

$$\begin{aligned}
\left|u_F^{n+1} - w^{n+1}\right| &= \Bigg\| \Bigg[u^n + \Delta t\,(Au^n + B(u^n)) \\
&\quad + \frac{\Delta t^2}{2!}\left(A^2u^n + AB(u^n) + \frac{dB(u^n)}{du^n}Au^n + \frac{dB(u^n)}{du^n}B(u^n)\right) + O(\Delta t^3)\Bigg] \\
&\quad - \Bigg[u^n + \Delta t\,(Au^n + B(u^n)) \\
&\quad + \frac{\Delta t^2}{2}\left(A^2u^n + AB(u^n) + \frac{dB(u^n)}{du^n}Au^n + \frac{dB(u^n)}{du^n}B(u^n)\right) + O(\Delta t^3)\Bigg] \Bigg\| \\
&= O(\Delta t^3).
\end{aligned}$$

On the other hand, the 'BAB' recombination of the split solution gives the following approximation by Taylor expansion

$$\begin{aligned}
w^{n+1} &= w^{n+1/2} + \frac{\Delta t}{2}\frac{dw^{n+1/2}}{dt} + \frac{\Delta t^2}{8}\frac{d^2w^{n+1/2}}{dt^2} + O(\Delta t^3) \\
&= v^{n+1} + \frac{\Delta t}{2}B(v^{n+1}) + \frac{\Delta t^2}{8}\frac{dB(v^{n+1})}{dt} + O(\Delta t^3) \\
&= e^{\Delta tA}u^{n+1/2} + \frac{\Delta t}{2}B(e^{\Delta tA}u^{n+1/2}) + \frac{\Delta t^2}{8}\frac{dB(e^{\Delta tA}u^{n+1/2})}{dt} + O(\Delta t^3) \\
&= \left(I + \Delta tA + \frac{\Delta t^2}{2!}A^2 + O(\Delta t^3)\right)u^{n+1/2} + \frac{\Delta t}{2}B\left(\left(I + \Delta tA + O(\Delta t^2)\right)u^{n+1/2}\right) \\
&\quad + \frac{\Delta t^2}{8}\frac{dB\left((I + O(\Delta t))\,u^{n+1/2}\right)}{dt} + O(\Delta t^3) \\
&= \left(I + \Delta tA + \frac{\Delta t^2}{2!}A^2 + O(\Delta t^3)\right)\left(u^n + \frac{\Delta t}{2}\frac{du^n}{dt} + \frac{\Delta t^2}{8}\frac{d^2u^n}{dt^2} + O(\Delta t^3)\right) \\
&\quad + \frac{\Delta t}{2}\left(B(u^{n+1/2}) + \Delta t\frac{dB(u^{n+1/2})}{du^{n+1/2}}Au^{n+1/2} + O(\Delta t^2)\right) + \frac{\Delta t^2}{8}\frac{dB(u^{n+1/2})}{dt} + O(\Delta t^3) \\
&= \left(I + \Delta tA + \frac{\Delta t^2}{2!}A^2 + O(\Delta t^3)\right)\left(u^n + \frac{\Delta t}{2}\frac{du^n}{dt} + \frac{\Delta t^2}{8}\frac{d^2u^n}{dt^2} + O(\Delta t^3)\right) \\
&\quad + \frac{\Delta t}{2}\left(B\left(u^n + \frac{\Delta t}{2}\frac{du^n}{dt} + O(\Delta t^2)\right) + \Delta t\frac{dB(u^n + O(\Delta t))}{du^n + O(\Delta t)}A\,(u^n + O(\Delta t)) + O(\Delta t^2)\right) \\
&= \left(I + \Delta tA + \frac{\Delta t^2}{2!}A^2 + O(\Delta t^3)\right)\left(u^n + \frac{\Delta t}{2}B(u^n) + \frac{\Delta t^2}{8}\frac{dB(u^n)}{dt} + O(\Delta t^3)\right) \\
&\quad + \frac{\Delta t}{2}\left(B(u^n) + \frac{\Delta t}{2}\frac{dB(u^n)}{dt} + \Delta t\frac{dB(u^n)}{du^n}Au^n + O(\Delta t^2)\right) + \frac{\Delta t^2}{8}\frac{dB(u^n)}{dt} + O(\Delta t^3) \\
&= u^n + \Delta t(Au^n + B(u^n)) + \frac{\Delta t^2}{2}\left(A^2u^n + AB(u^n) + \frac{dB(u^n)}{du^n}Au^n + \frac{dB(u^n)}{du^n}B(u^n)\right) + O(\Delta t^3)
\end{aligned}$$

Thus, we have

$$
\begin{aligned}
\left|u_F^{n+1} - w^{n+1}\right| &= \Bigg|\Big[u^n + \Delta t\,(Au^n + B(u^n)) \\
&\quad + \frac{\Delta t^2}{2!}\left(A^2u^n + AB(u^n) + \frac{dB(u^n)}{du^n}Au^n + \frac{dB(u^n)}{du^n}B(u^n)\right) + O(\Delta t^3)\Big] \\
&\quad - \Big[u^n + \Delta t\,(Au^n + B(u^n)) \\
&\quad + \frac{\Delta t^2}{2}\left(A^2u^n + AB(u^n) + \frac{dB(u^n)}{du^n}Au^n + \frac{dB(u^n)}{du^n}B(u^n)\right) + O(\Delta t^3)\Big]\Bigg| \\
&= O(\Delta t^3).
\end{aligned}
$$

Hence, the computations for both split orderings gives second order accuracy in time.

4.1.3. Accuracy of the Symmetrically Weighted Sequential Splitting

The 'AB' recombination of the split solution given in Section 4.1.1. is

$$
v^{n+1} = u^n + \Delta t(Au^n + B(u^n)) + \frac{\Delta t^2}{2!}\left(A^2u^n + 2\frac{dB(u^n)}{du^n}Au^n + \frac{dB(u^n)}{du^n}B(u^n)\right) + O(\Delta t^3),
$$

and the 'BA' recombination of the split solution is

$$
v^{n+1} = u^n + \Delta t(Au^n + B(u^n)) + \frac{\Delta t^2}{2!}\left(A^2u^n + 2AB(u^n) + \frac{dB(u^n)}{du^n}B(u^n)\right) + O(\Delta t^3).
$$

Then, taking average 'AB' and 'BA' splitting recombination, we get

$$
v^{n+1} = u^n + \Delta t(Au^n + B(u^n)) + \frac{\Delta t^2}{2!}\left(A^2u^n + 2AB(u^n) + \frac{dB(u^n)}{du^n}B(u^n)\right) + O(\Delta t^3).
$$

Thus, it follows that

$$
\begin{aligned}
\left|u_F^{n+1} - v^{n+1}\right| &= \Bigg|\Big[u^n + \Delta t(Au^n + B(u^n)) \\
&\quad + \frac{\Delta t^2}{2!}(A^2u^n + AB(u^n) + \frac{dB(u^n)}{du^n}Au^n + \frac{dB(u^n)}{du^n}B(u^n)) + O(\Delta t^3)\Big] \\
&\quad - \Big[u^n + \Delta t(Au^n + B(u^n)) \\
&\quad + \frac{\Delta t^2}{2!}(A^2u^n + 2AB(u^n) + \frac{dB(u^n)}{du^n}B(u^n)) + O(\Delta t^3)\Big]\Bigg| \\
&= \left|\frac{\Delta t^2}{2!}\left(-AB(u^n) + \frac{dB(u^n)}{du^n}Au^n\right) + O(\Delta t^3)\right| \\
&= O(\Delta t^3).
\end{aligned}
$$

Hence, we can obtain second order of accuracy as well.

4.2. A Kinetic Model of Phage Infection

Bacteriophages, more commonly known as phages, are viruses that kill bacteria. They are used to treat food or animals infected with bacteria [6].

We define a kinetic model of phage infection for a generalized phage-bacterium system. Let $x(t)$ represent the number of uninfected bacteria, $y(t)$ the infected bacteria, and $v(t)$ the free phage. Then, we can write this model as follows:

$$\begin{aligned} \frac{dx}{dt} &= ax - bvx, \\ \frac{dy}{dt} &= ay + bvx - ky, \\ \frac{dv}{dt} &= kLy - bvx - mv, \end{aligned} \tag{4.7}$$

where a is the replication coefficient of the bacteria, b is the transmission coefficient, k the lysis rate, L the burst size, and m the decay rate of free phages [7].

This model investigate with the following typical parameter values (time units: hours): $a = 0.3$, $b = 10^{-6}$, $k = 0.706$, $L = 15$, $m = 34.8$ with the initial conditions

$$\begin{pmatrix} x_0 \\ y_0 \\ v_0 \end{pmatrix} = \begin{pmatrix} 19000 \\ 5400 \\ 72000 \end{pmatrix}. \tag{4.8}$$

Then, we write the matrix representation of (4.7) as

$$\underbrace{\begin{pmatrix} x' \\ y' \\ v' \end{pmatrix}}_{u'} = \underbrace{\begin{pmatrix} a & 0 & 0 \\ 0 & a-k & 0 \\ 0 & kL & -m \end{pmatrix}}_{A} \underbrace{\begin{pmatrix} x \\ y \\ v \end{pmatrix}}_{u} + \underbrace{\begin{pmatrix} -bvx \\ bvx \\ -bvx \end{pmatrix}}_{B(u)} \tag{4.9}$$

with the initial condition

$$u_0 = \begin{pmatrix} x_0 \\ y_0 \\ v_0 \end{pmatrix} = u(0). \tag{4.10}$$

To introduce the method, we first rewrite (4.9) in operator form

$$\frac{du}{dt} = Au + B(u), \tag{4.11}$$

where A and B are linear and nonlinear operators, respectively.

We will use (4.11) in order to solve given model in (4.7) numerically.

4.2.1. Lie-Trotter Splitting For A Kinetic Model of Phage Infection

To apply Lie-Trotter splitting, we split the problem (4.11) into the two subproblems as following:

$$\frac{du_1}{dt} = Au_1, \quad u_1(0) = u_0 \tag{4.12}$$

$$\frac{du_2}{dt} = B(u_2), \quad u_2(0) = u_1(t) \tag{4.13}$$

where A is a linear operator and B is a nonlinear operator. Then, we solved these subproblems sequentially for small time step Δt. The first subproblem (4.12) is solved exactly. The subproblem (4.13) is solved by using forward Euler's method.

4.2.2. Strang-Marchuk Splitting For A Kinetic Model of Phage Infection

To apply the Strang-Marchuk splitting method to (4.11), we divide it into three subproblems as

$$\frac{du_1}{dt} = Au_1, \quad u_1(0) = u_0, \tag{4.14}$$

and

$$\frac{du_2}{dt} = B(u_2), \quad u_2(0) = u_1(t), \tag{4.15}$$

and again

$$\frac{du_3}{dt} = Au_3, \quad u_3(0) = u_2(t), \tag{4.16}$$

where A is a linear operator and B is a nonlinear operator. Then, we solved these subproblems sequentially for small time step Δt. The first subproblem (4.14) is solved exactly, the second subproblem (4.15) is solved by using 2^{nd} order Runge-Kutta method and the last subproblem (4.16) is solved again exactly.

4.2.3. Symmetrically Weighted Sequential Splitting For A Kinetic Model of Phage Infection

To apply the SWS splitting method to (4.11), we divide it into 'AB' splitting solution as

$$\frac{du_1}{dt} = Au_1, \quad u_1(0) = u_0 \tag{4.17}$$

$$\frac{du_2}{dt} = B(u_2), \quad u_2(0) = u_1(t) \tag{4.18}$$

and 'BA' splitting solution as

$$\frac{du_1}{dt} = B(u_1), \quad u_1(0) = u_0 \tag{4.19}$$

$$\frac{du_2}{dt} = Au_2, \quad u_2(0) = u_1(t) \tag{4.20}$$

where A is a linear operator and B is a nonlinear operator. Hence, SWS splitting can be obtained by symmetrizing the Lie-Trotter. Then, the splitting solution is obtained by averaging the corresponding results. We solved subproblems (4.17) and (4.20) exactly and the other subproblems (4.18) and (4.19) by using 2^{nd} order Runge-Kutta method.

4.2.4. Numerical Results For A Kinetic Model of Phage Infection

Since we do not have an exact solution for the problem (4.7), we need a reference solution for our study. So, we solved this problem with the 4^{th} order Runge-Kutta method as a reference solution.

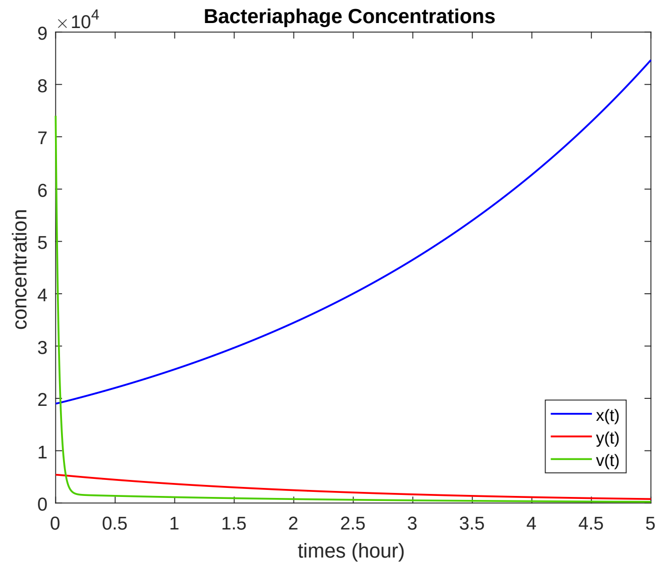


Figure 4.1. Reference solution of the problem (4.7).

Figure 4.1 exhibits reference solution of the problem (4.7) with time step $\Delta t = 10^{-6}$ by using 4^{th} order Runge-Kutta method on time interval $t \in [0, 5]$.

	x	y	v
Lie-Trotter	0.3147	0.0085	0.0039
Strang-Marchuk	$9.0332e-05$	$2.7844e-06$	$4.7724e-07$
Symmetrically W. S.	$1.7880e-04$	$5.2350e-06$	$8.7069e-07$

Table 4.1. Maximum errors of different splitting methods of Problem 3 for $\Delta t = 0.0001$ on time interval $t \in [0, 5]$.

Table 4.1 shows comparison the maximum errors of the classical operator splitting methods.From the table, we deduce that the errors of Strang-Marchuk splitting gave smaller error for (4.7).

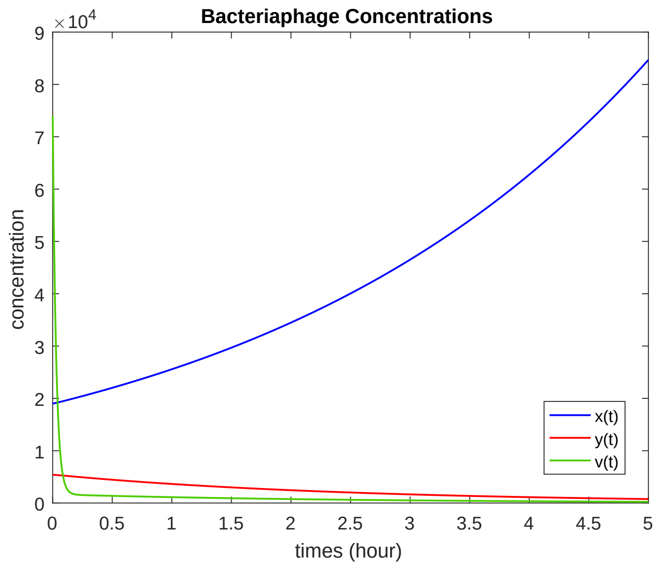


Figure 4.2. Numerical solution of the Lie-Trotter splitting for the problem (4.7).

Figure 4.2 shows numerical solution of the x, y and v components of the problem (4.7) by applying Lie-Trotter splitting for $\Delta t = 0.01$ on time interval $t \in [0, 5]$.

	x		y		v	
Time Step Size	Error	Order	Error	Order	Error	Order
$\Delta t = 0.1$	164.4330		4.0713		3.1779	
$\Delta t = 0.01$	29.7225	0.7429	0.7994	0.7070	0.3831	0.9188
$\Delta t = 0.001$	3.1315	0.9773	0.0846	0.9753	0.0391	0.9914
$\Delta t = 0.0001$	0.3147	0.9978	0.0085	0.9976	0.0039	0.9991

Table 4.2. Maximum error and order of the Lie-Trotter splitting for x, y and v components.

Table 4.2 shows maximum error and order of accuracy of Lie-Trotter splitting for x, y and v components with different Δt values by using l^∞ norm. It can be seen that the maximum errors decrease for x, y and v components when the time decreases. Also, we can see that the expected order is confirmed, and thus the Lie-Trotter splitting converges as first order.

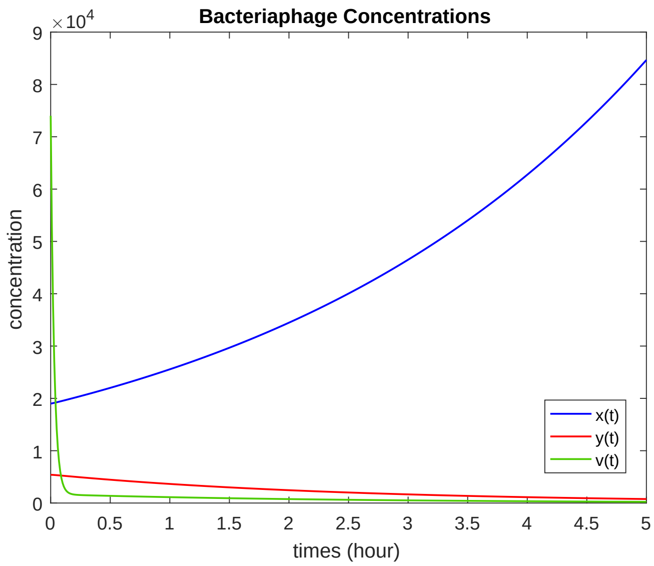


Figure 4.3. Numerical solution of the Strang-Marchuk splitting for the problem (4.7).

Figure 4.3 shows numerical solution of the x, y and v components of the problem (4.7) by applying Strang-Marchuk splitting for $\Delta t = 0.01$ on time interval $t \in [0, 5]$.

	x		y		v	
Time Step Size	Error	Order	Error	Order	Error	Order
$\Delta t = 0.1$	66.3106		2.0771		0.3616	
$\Delta t = 0.01$	0.9010	1.8669	0.0278	1.8736	0.0048	1.8799
$\Delta t = 0.001$	0.0090	1.9985	$2.7881e-04$	1.9986	$4.7836e-05$	1.9986
$\Delta t = 0.0001$	$9.0332e-05$	2.0004	$2.7844e-06$	2.0006	$4.7724e-07$	2.0010

Table 4.3. Maximum error and order of the Strang-Marchuk splitting for x, y and v components.

Table 4.3 shows the maximum error and the order of accuracy of the Strang-Marchuk splitting for x, y and v components with different Δt values by using l^∞ norm. We conclude that the maximum errors decrease for x, y and v components when the time decreases. Also, we can see that the expected order is confirmed, and thus the Strang-Marchuk splitting converges second order.

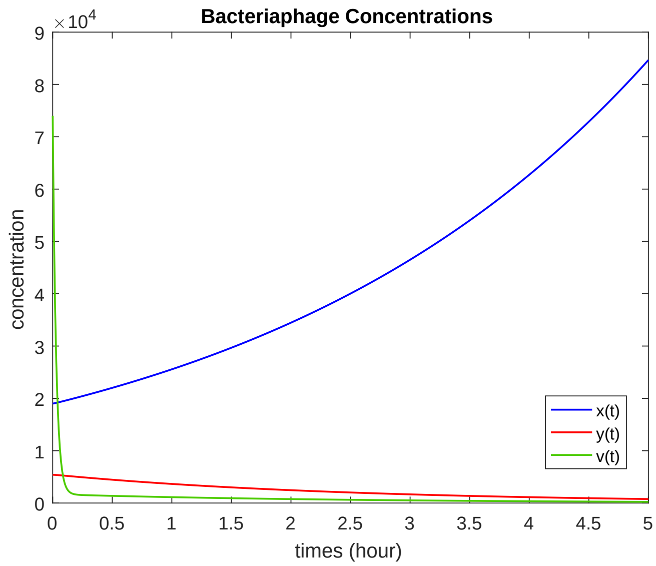


Figure 4.4. Numerical solution of the SWS splitting for the problem (4.7).

Figure 4.4 exhibits numerical solution of the x, y and v components of the problem (4.7) by applying SWS splitting for $\Delta t = 0.01$ on time interval $t \in [0, 5]$.

	x		y		v	
Time Step Size	Error	Order	Error	Order	Error	Order
$\Delta t = 0.1$	150.2231		4.4266		0.7387	
$\Delta t = 0.01$	1.7824	1.9257	0.0522	1.9282	0.0087	1.9298
$\Delta t = 0.001$	0.0179	1.9991	$5.2329e-04$	1.9992	$8.7004e-05$	1.9992
$\Delta t = 0.0001$	$1.7880e-04$	1.9995	$5.2350e-06$	1.9998	$8.7069e-07$	1.9997

Table 4.4. Maximum error and order of the SWS splitting for x, y and v components.

Table 4.4 shows the maximum error and the order of accuracy of the SWS splitting for x, y and v components with different Δt values by using l^∞ norm. We conclude that the maximum errors decrease for x, y and v components when the time decreases. Also, we can see that the expected order is confirmed, and thus the SWS splitting converges second order.

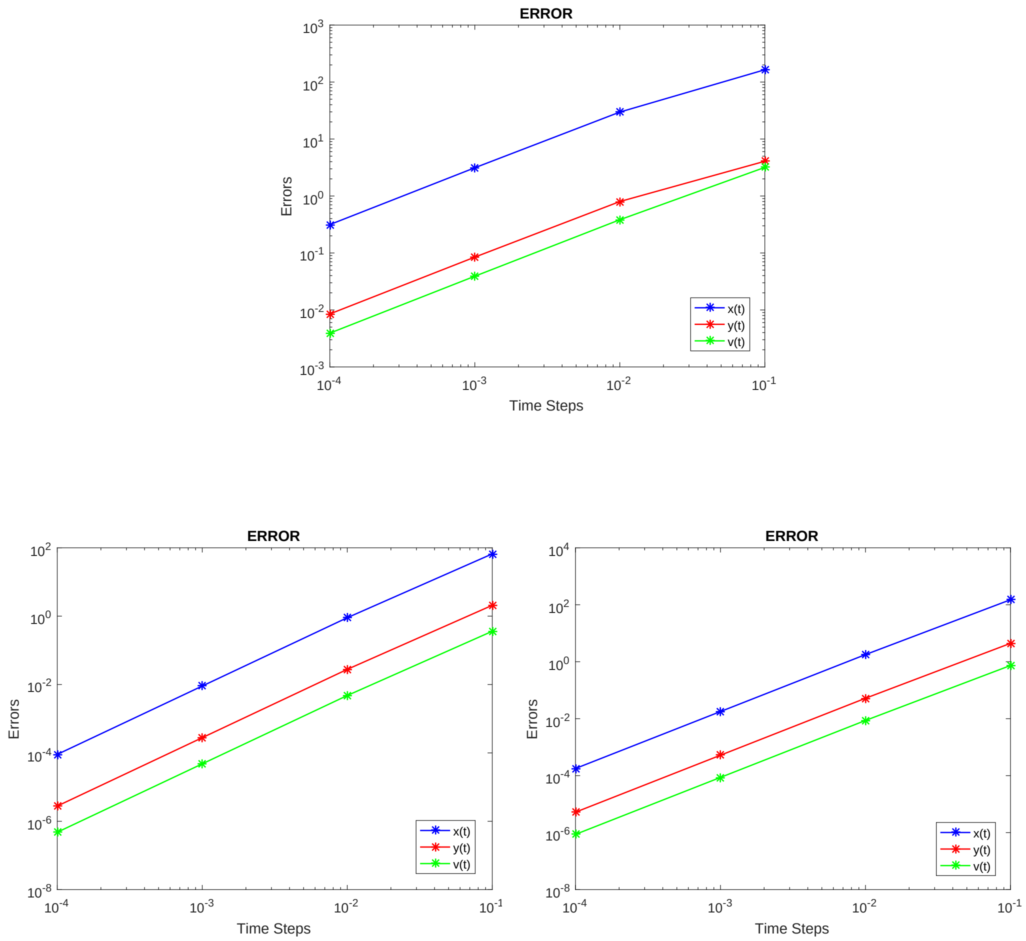


Figure 4.5. The maximum errors of Lie-Trotter, Strang-Marchuk and SWS splitting for (4.7).

Figure 4.5 represents local splitting error of Lie-Trotter, Strang-Marchuk and SWS splitting, respectively, for x, y and v components by using l^{∞} norm. We deduce that the numerical solutions close to each other when applying SWS splitting for (4.7).

Figure 4.6 exhibits the maximum errors of the x, y and v components, respectively, for the classical operator splitting methods by using l^1, l^2 and l^{∞} norm.

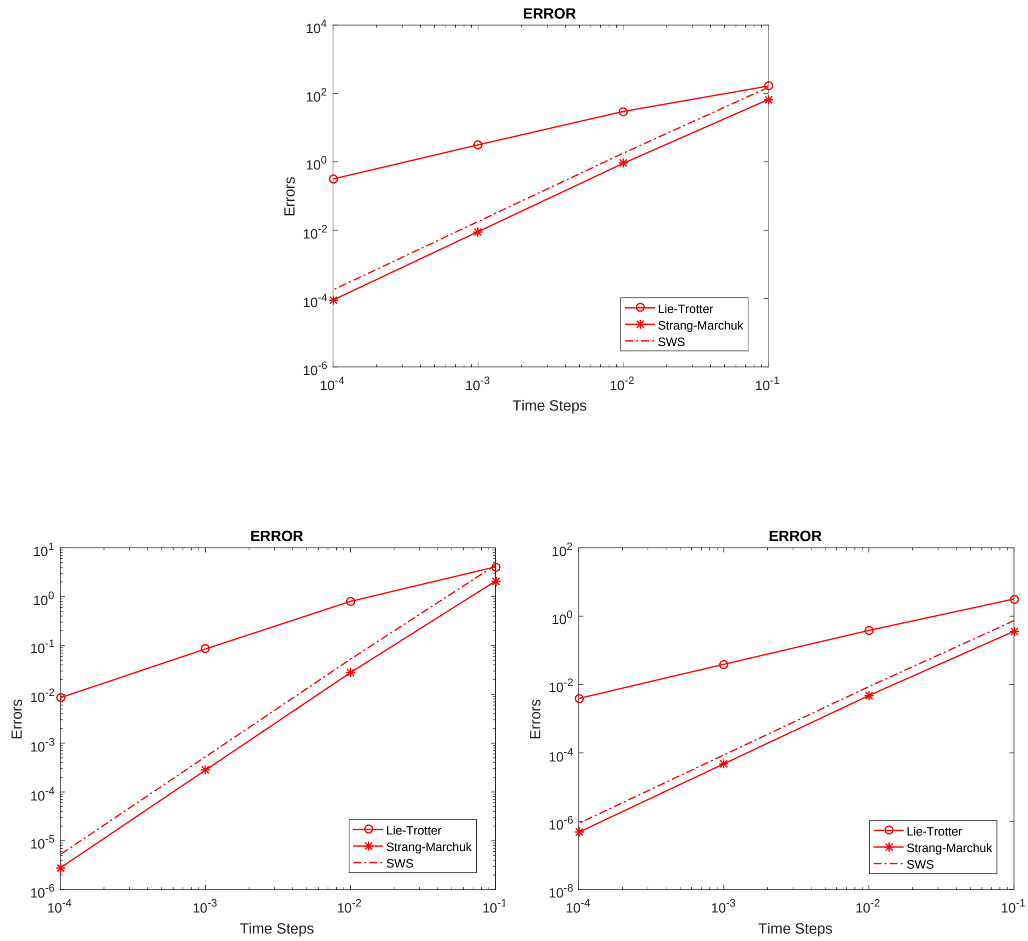


Figure 4.6. The maximum errors of Lie-Trotter, Strang-Marchuk and SWS splitting for (4.7).

4.3. The Newell-Whitehead-Segel Equation

The Newell-Whitehead-Segel (NWS) equation models the interaction of the effect of the diffusion term with the nonlinear effect of the reaction term. This equation can be viewed as a generalization of the NWS equation which appeared in the investigation of fluid mechanics [8].

The NWS equation is a reaction-diffusion equation written of the form

$$u_t = ku_{xx} + au - bu^q, \tag{4.21}$$

where a, b and k are real numbers with $k > 0$ which is the coefficient of diffusion, and q is a positive integer [8].

In Eq. (4.21), if we take $a = 1, b = 1$, and $q = 3$, then the NWS equation becomes the Allen-Cahn equation [9]

$$u_t = ku_{xx} + u - u^3, \quad x \in [0, 2\pi], \quad 0 \leq t \leq 1 \tag{4.22}$$

where u_{xx} is a linear diffusion term and $u - u^3$ is a nonlinear reaction term. We will take k=0.01 with the initial condition

$$u(x, 0) = 0.05 \sin(x) \tag{4.23}$$

and periodic boundary conditions

$$u(0, t) = u(2\pi, t) = 0. \tag{4.24}$$

We performed a spacial discretization with lenght parameter $\Delta x = \pi/64$ that is we divided $x \in [0, 2\pi]$ into $Nx = 128$ parts of equal lenght. The spatial derivative is approximated with the finite difference scheme:

$$\left.\frac{\partial u_m(t)}{\partial t}\right|_{(x_m,t)} = \frac{u_{m+1}(t) - 2u_m(t) + u_{m-1}(t)}{\Delta x^2}, \tag{4.25}$$

where Δx is the space step size and $m = 1, ..., N_x + 1$.

Thus, we obtain the following semi-discrete differential equation

$$u_t = k\frac{1}{\Delta x^2}Au(t) + u(t) - u^3(t) \tag{4.26}$$

where $u(t)$ in equation (4.26) is in the form of $u(t) = (u(x_1, t), u(x_2, t), ..., u(x_{N-1}, t))^T$ and A is $(N - 1) \times (N - 1)$ tridiagonal matrix. $u(0) = (u(x_1, 0), u(x_2, 0), ..., u(x_{N-1}, 0))^T$ is the initial condition and the boundary conditions $u(x_0, 0)$ and $u(x_N, 0)$ are embedded into the matrix.

$$\begin{bmatrix} u'_1 \\ u'_2 \\ \vdots \\ \\ u'_{N-1} \end{bmatrix} = \frac{1}{\Delta x^2} \begin{bmatrix} -2 & 1 & & & & \\ 1 & -2 & 1 & & & \\ & 1 & -2 & 1 & & \\ & & \ddots & \ddots & \ddots & \\ & & & & -2 & 1 \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ \vdots \\ \\ u_{N-1} \end{bmatrix}.$$

We will use (4.26) in order to solve NWS equation numerically.

4.3.1. Lie-Trotter Splitting For The Newell-Whitehead-Segel Equation

To apply Lie - Trotter splitting to (4.26), we construct the two subproblems

$$U_t = AU = kU_{xx} \tag{4.27}$$

$$V_t = f(V) = V - V^3 \tag{4.28}$$

which are solved subsequently for small time steps Δt. We will use the centered finite difference method for the second derivative of u for (4.27) and forward Euler method for (4.28) which is time evolution.

4.3.2. Strang-Marchuk Splitting For The Newell-Whitehead-Segel Equation

To apply Strang-Marchuk splitting to (4.26), we construct the three subproblems

$$U_t = AU = kU_{xx} \tag{4.29}$$

$$V_t = f(V) = V - V^3 \tag{4.30}$$

$$U_t = AU = kU_{xx} \tag{4.31}$$

which are solved subsequently for small time steps Δt. We will use the centered finite difference method for the second derivative of u in (4.29) and (4.31) and 2^{nd} order Runge-Kutta method in (4.30) which is time evolution.

4.3.3. Symmetrically Weighted Sequential Splitting For The Newell-Whitehead-Segel Equation

To apply SWS splitting method to (4.26), we divide it into 'AB' splitting solution as

$$U_t = AU = kU_{xx} \tag{4.32}$$

$$V_t = f(V) = V - V^3 \tag{4.33}$$

and 'BA' splitting solution as

$$U_t = f(U) = U - U^3 \tag{4.34}$$

$$V_t = AV = kV_{xx} \tag{4.35}$$

Thus, SWS splitting can be obtained by symmetrizing the Lie-Trotter. Then, the splitting solution is obtained by averaging the corresponding results. We solved subproblems (4.32) and (4.35) exactly, the other subproblems (4.33) and (4.34) by using 2^{nd} order Runge-Kutta method.

4.3.4. Numerical Results For The Newell-Whitehead-Segel Equation

Since there is no exact solution for (4.22), we solved the full problem (4.22) − (4.24) with the 4^{th} order Runge-Kutta method for our study.

Figure 4.7 represents the reference solution of the problem (4.22) generated by 4^{th} order Runge-Kutta method with time step $\Delta t = 10^{-4}$ for $x \in [0, 2\pi]$ on time interval $t \in [0, 1]$.

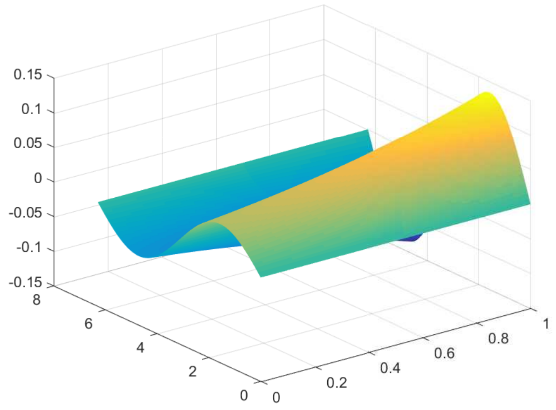
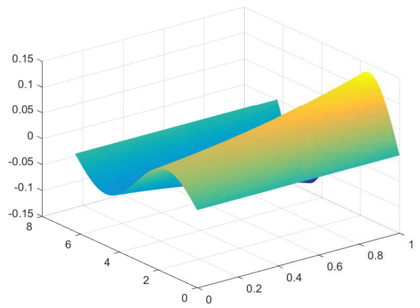
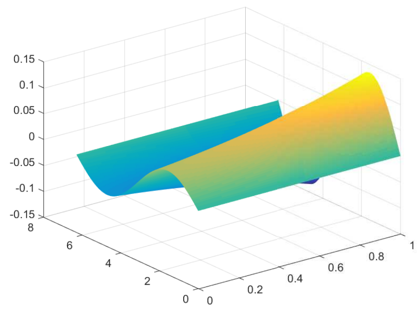


Figure 4.7. Reference solution of the problem (4.22).



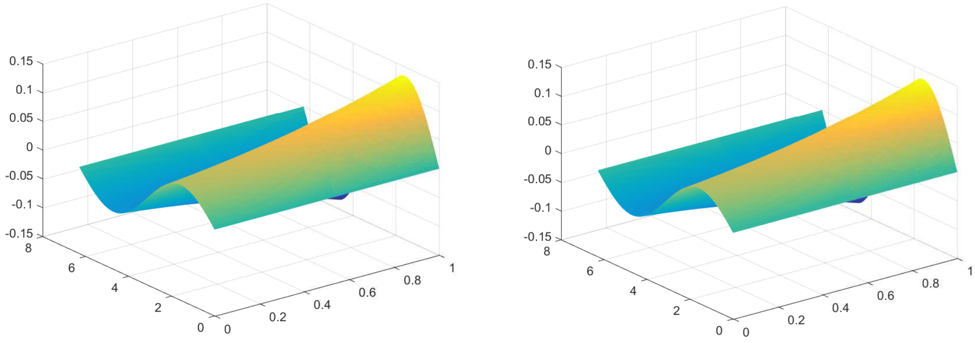
Approximate Solution



Reference Solution

Figure 4.8. Comparison of the approximate and reference solution for the NWS Equation (4.22) by using Lie-Trotter splitting for $\Delta t = 0.001$ with $x \in [0, 2\pi]$ on time interval $t \in [0, 1]$.

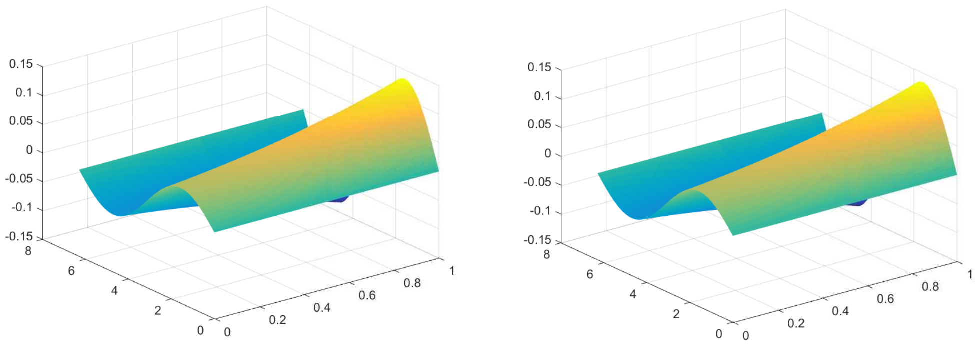
Figure 4.8 shows numerical solution of the Lie-Trotter splitting and reference solution for (4.22) for $\Delta t = 0.001$ with $x \in [0, 2\pi]$ on time interval $t \in [0, 1]$.



Approximate Solution Reference Solution

Figure 4.9. Comparison of the approximate and reference solution for the NWS Equation (4.22) by using Strang-Marchuk splitting for $\Delta t = 0.001$ with $x \in [0, 2\pi]$ on time interval $t \in [0, 1]$.

Figure 4.9 shows numerical solution of the Strang-Marchuk splitting and reference solution for (4.22) for $\Delta t = 0.001$ with $x \in [0, 2\pi]$ on time interval $t \in [0, 1]$.



Approximate Solution Reference Solution

Figure 4.10. Comparison of the approximate and reference solution for the NWS Equation (4.22) by using Lie-Trotter splitting for $\Delta t = 0.001$ with $x \in [0, 2\pi]$ on time interval $t \in [0, 1]$.

Figure 4.10 shows numerical solution of the SWS splitting and reference solution for (4.22) for $\Delta t = 0.001$ with $x \in [0, 2\pi]$ on time interval $t \in [0, 1]$.

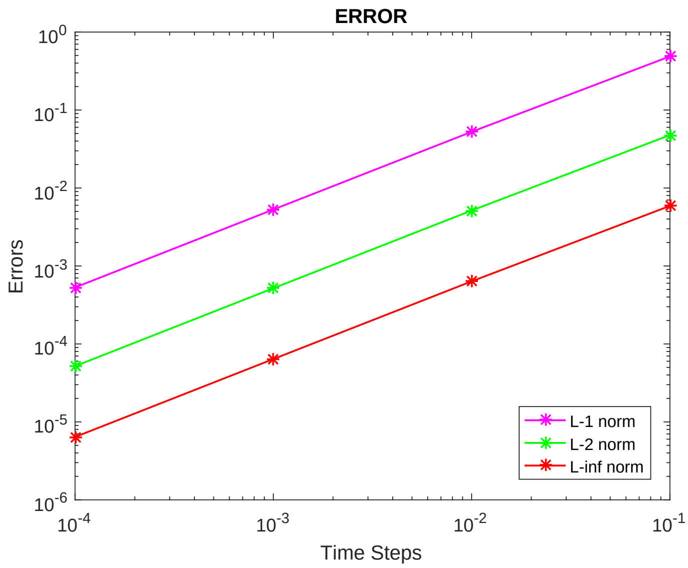


Figure 4.11. Error of Lie-Trotter splitting for (4.22).

Figure 4.11 represents the local splitting error of the Lie-Trotter splitting for (4.22) with different Δt values by using l^1, l^2 and l^∞ norms. We can see that as the time step decreases, the errors also decrease.

Time Step Size	Maximum Error	Order of Accuracy
$\Delta t = 0.1$	0.0059	
$\Delta t = 0.01$	$6.3496e-04$	0.9679
$\Delta t = 0.001$	$6.3987e-05$	0.9967
$\Delta t = 0.0001$	$6.4036e-06$	0.9997

Table 4.5. Maximum Error of Lie-Trotter Splitting for the problem (4.22).

Table 4.5 shows the maximum error of Lie-Trotter splitting for (4.22) with different Δt values. We can clearly see how the errors changes when the time steps are getting smaller. Moreover, we conclude that the expected order is confirmed, that is, Lie-Trotter splitting converges to 1.

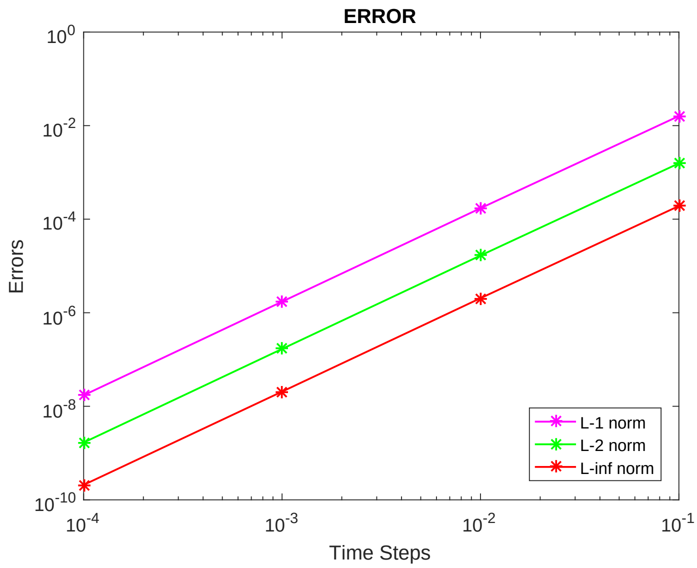


Figure 4.12. Error of Strang-Marchuk splitting for (4.22).

Figure 4.12 represents the local splitting error of Strang-Marchuk splitting for (4.22) with different Δt values by using l^1, l^2 and l^∞ norms. We can see that as the time step decreases, the errors also decrease.

Time Step Size	Maximum Error	Order of Accuracy
$\Delta t = 0.1$	$1.9157e-04$	
$\Delta t = 0.01$	$2.0453e-06$	1.9716
$\Delta t = 0.001$	$2.0587e-08$	1.9972
$\Delta t = 0.0001$	$2.0869e-10$	1.9941

Table 4.6. Maximum Error of Strang-Marchuk Splitting of Nonlinear PDE with Different Δt Values.

Table 4.6 shows the maximum error of the Strang-Marchuk splitting for (4.22) with different Δt values. Moreover, we conclude that the expected order is confirmed, that is, Strang-Marchuk splitting converges to 2.

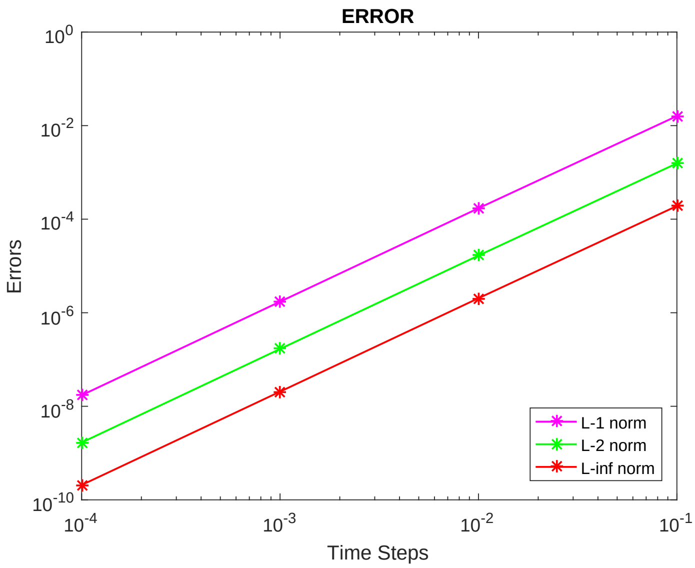


Figure 4.13. Error of SWS splitting for (4.22).

Figure 4.13 represents the local splitting error of SWS splitting for (4.22) with different Δt values by using l^1, l^2 and l^∞ norms. We can see that as the time step decreases, the errors also decrease.

Time Step Size	Maximum Error	Order of Accuracy
$\Delta t = 0.1$	$1.9158e-04$	
$\Delta t = 0.01$	$2.0453e-06$	1.9716
$\Delta t = 0.001$	$2.0587e-08$	1.9972
$\Delta t = 0.0001$	$2.0767e-10$	1.9962

Table 4.7. Maximum Error of SWS Splitting of Nonlinear PDE with Different Δt Values.

Table 4.7 shows the maximum error of the SWS splitting for (4.22). Moreover, we conclude that the expected order is confirmed, that is, SWS splitting converges to 2.

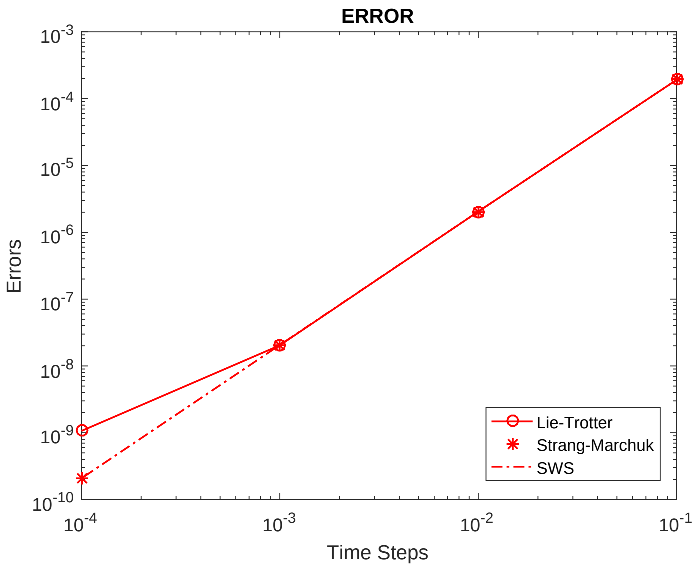


Figure 4.14. Comparison of Errors of Different Splitting Methods of the problem (4.22).

Figure 4.14 represents a comparison between the local splitting error of the different splitting methods with various Δt values by using l^{∞} norm. From Figure 4.14, we can see that Strang-Marchuk splitting and SWS splitting are coincident.

	Error l^1	Error l^2	Error l^{∞}
Lie-Trotter Spliting	0.0053	$5.1813e-04$	$6.3987e-05$
Strang-Marchuk Splitting	$1.7252e-06$	$1.6825e-07$	$2.0587e-08$
Symmetrically Weighted Sequential Splitting	$1.7252e-06$	$1.6826e-07$	$2.0587e-08$

Table 4.8. Comparison of Errors of Different Splitting Methods of the problem (4.22) for $\Delta t = 0.001$.

Table 4.8 shows a comparison of the local splitting errors of the different splitting methods and nonsplitting via l^1, l^2 and l^{∞} norm. From the table, we deduce that the error of Strang-Marchuk splitting is the same as that of SWS splitting.

CHAPTER 5

SUMMARY AND CONCLUSIONS

In this thesis, Lie-Trotter splitting, Strang-Marchuk splitting and symmetrically weighted sequential splitting methods which are known as classical operator splitting methods have been successfully applied to solve variety ODE and PDE problems.

First, simple examples are considered in order to demonstrate the effectiveness of the operator splitting methods. We also studied the accuracy of the operator splitting methods for each problem. Theoretically, we prove that Lie-Trotter splitting has an order 1, Strang-Marchuk and symmetrically weighted sequential splitting have an order 2. Our theoretical results are also confirmed by the numerical computations. Many figures and tables are presented to show agreements of theoretical and numerical computations.

Second, we consider the classical operator splitting methods for nonlinear differential equations with linear and nonlinear operators. Moreover, we prove order of accuracy of these methods by using local splitting error.

Next, we considered real-life problem; a kinetic model of phage infection. The classical operator splitting methods are successfully applied to find the approximate solution of the system of ODE. Reference solution was used as an exact solution.

Finally, the Newell-Whitehead-Segel equation as a PDE problem is considered. Since we do not know the exact solution of these problems for some parameters, we obtain the approximate solution of this PDE by using the classical operator splitting methods. We also confirmed the teoretical results with numerical computations.

In the light of our studies, we recommend the operator splitting method to obtain the numerical solution of various nonlinear ODE and PDE problems. They are very efficient and robustness methods.

REFERENCES

[1] MacNamara, S.; Strang, G., Operator splitting. In *Splitting Methods in Communication, Imaging, Science, and Engineering*, Springer: 2016; pp 95-114.

[2] Yazıcı, Y., Operator splitting methods for differential equations. M.Sc. Thesis, *İzmir Institute of Technology*, 2010.

[3] Harwood, R. C., Operator splitting method and applications for semilinear parabolic partial differential equations. Ph.D. Dissertation, Washington State University, May 2011.

[4] Hundsdorfer, W.; Verwer, J. G., *Numerical solution of time-dependent advection-diffusion-reaction equations*. Springer Science & Business Media: 2013; Vol. 33.

[5] Faragó, I.; Havasi, Á., *Consistency analysis of operator splitting methods for C_0-semigroups*, Semigroup Forum, Springer: 2007; pp 125-139.

[6] Vidurupola, S. W. Mathematical models for bacteriophage dynamics applicable to phage therapy. Ph.D. Dissertation, Texas Tech University, August 2013.

[7] Payne, R. J.; Jansen, V. A., Understanding bacteriophage therapy as a density-dependent kinetic process. *Journal of Theoretical Biology* 2001, 208 (1), 37-48.

[8] Akçağıl, Ş.; Aydemir, T., New exact solutions for the Khokhlov-Zabolotskaya-Kuznetsov, the Newell-Whitehead-Segel and the Rabinovich wave equations by using a new modification of the tanh-coth method. *Cogent Mathematics* 2016, 3 (1), 1193104.

[9] Yang, J.; Du, Q.; Zhang, W., Uniform L^p-Bound of the Allen-Cahn Equation And Its Numerical Discretization. *International Journal of Numerical Analysis & Modeling* 2018, 15(1-2), 213-227.

[10] Céline C., Bacteria-Phage mathematical model applied to the cheese industry. *Bacteriophage Ecology group*, Biomathematics Project, Massey University, 2003.

[11] Faragó, I.; Havasi, Á.; Zlatev, Z., *Advanced numerical methods for complex environmental models: needs and availability*. Bentham Science Publishers: 2013.

[12] Geiser, J., *Decomposition methods for differential equations: theory and applications*. CRC Press: 2009.

[13] Faragó, I.; Gnandt, B.; Havasi, Á., Additive and iterative operator splitting methods and their numerical investigation. *Computers & Mathematics with Applications* 2008, 55 (10), 2266-2279.

[14] Ostermann, A.; Schratz, K., Stability of exponential operator splitting methods for noncontractive semigroups. *SIAM Journal on Numerical Analysis* 2013, 51 (1), 191-203.

[15] Zhao, S.; Ovadia, J.; Liu, X.; Zhang, Y.-T.; Nie, Q., Operator splitting implicit integration factor methods for stiff reaction–diffusion–advection systems. *Journal of computational physics* 2011, 230 (15), 5996-6009.

[16] Bátkai, A.; Csomós, P.; Farkas, B., Operator splitting for nonautonomous delay equations. *Computers & Mathematics with Applications* 2013, 65 (3), 315-324.

[17] Geiser, J., Consistency of iterative operator-splitting methods: Theory and applications. *Numerical Methods for Partial Differential Equations: An International Journal* 2010, 26 (1), 135-158.

[18] Dai, X.; Sun, J.; Cheng, X., Error estimates for an operator-splitting method for Navier–Stokes equations: Second-order schemes. *Journal of computational and applied mathematics* 2009, 231 (2), 696-704.

[19] Faragó, I. *New operator splitting methods and their analysis*, International Conference on Numerical Methods and Applications, Springer: 2006; pp 443-450.

[20] Rouhi, A.; Wright, J., Spectral implementation of a new operator splitting method for solving partial differential equations. *Computers in Physics* 1995, 9 (5), 554-563.

[21] Faragó, I.; Havasi, Á.; Horvath, R., On the order of operator splitting methods for time-dependent linear systems of differential equations. *International Journal of Numerical Analysis and Modeling* 2011, 2 (2-3), 142-154.

[22] Descombes, S.; Duarte, M.; Massot, M., Operator splitting methods with error estimator and adaptive time-stepping. Application to the simulation of combustion phenomena. In *Splitting Methods in Communication, Imaging, Science, and Engineering*, Springer: 2016; pp 627-641.

[23] Holden, H.; Karlsen, K. H.; Lie, K.-A., Risebro N.H., *Splitting methods for partial differential equations with rough solutions: Analysis and MATLAB programs.* European Mathematical Society: 2010; Vol. 11.

[24] Payne, R. J.; Jansen, V. A., Phage therapy: the peculiar kinetics of self-replicating pharmaceuticals. *Clinical Pharmacology & Therapeutics* 2000, 68 (3), 225-230.

[25] Faragó, I., A modified iterated operator splitting method. *Applied mathematical modelling* 2008, 32 (8), 1542-1551.

[26] Bjϕrhus, M., Operator splitting for abstract Cauchy problems. *IMA journal of Numerical Analysis* 1998, 18 (3), 419-443.

[27] Csomós, P. Theoretical and numerical analysis of operator splitting procedures. Ph.D. Thesis, Eötvös Lorónd University, Budapest, 2007.

[28] Yano, M.; Penn, J. D.; Konidaris, G.; Patera, A. T., Math, Numerics, & Programming (for Mechanical Engineers). 2012.

[29] Csomós, P.; Faragó, I.; Havasi, Á., Weighted sequential splittings and their analysis. *Computers & Mathematics with Applications* 2005, 50 (7), 1017-1031.

[30] Gücüyenen,, N., Operator splitting method for parabolic partial differential equations: Analyses and applications. Ph.D. Thesis, *İzmir Institute of Technology*, 2013.

[31] Zürnacı, F. Convergence analysis and numerical solutions of the Fisher's and Benjamin-Bono-Mahony equations by operator splitting method. M.Sc. Thesis, *İzmir Institute of Technology*, 2014.

APPENDIX A

MATLAB CODES FOR THE APPLICATIONS

CLASSICAL OPERATOR SPLITTING METHODS MATLAB CODES FOR LINEAR ODE

```
clear all
close all
clc

%INPUT
dt = 0.01;
stime = 0;
ftime = 1;
initial_c = 1;

%DISCRETIZATION
Nt = (ftime - stime)/dt;
t = stime:dt:ftime;

%EXACT SOLUTION
u_exact = -3.*exp(-2.*t)+4.*exp(-t);

%ASSIGNING INITIAL DATAS
t(1) = stime;
u_appr(1) = initial_c;
u1_AB=initial_c;
u_AB(1)=u1_AB;
u1_ABA=initial_c;
u_ABA(1)=u1_ABA;
```

```
u1_AB_sym = initial_c;
u_AB_sym(1) = u1_AB_sym;
u1_BA_sym = initial_c;
u_BA_sym(1) = u1_BA_sym;

%LIE-TROTTER SPLITTING
for i=1:Nt
u1_AB = u1_AB+dt*RHS_func_A(t(i),u1_AB);
u2_AB = u1_AB;
u2_AB = u2_AB+dt*RHS_func_B(t(i),u2_AB);
u_AB(i+1) = u2_AB;
u1_AB = u2_AB;
end

%STRANG-MARCHKUK SPLITTING
for i = 1:Nt
k11= (dt/2).*RHS_func_A(t(i),u1_ABA);
k12= (dt/2).*RHS_func_A(t(i)+(dt./2),u1_ABA+(k11/2));
u1_ABA = u1_ABA+k12;
u2_ABA = u1_ABA;
k21=dt.*RHS_func_B(t(i),u1_ABA);
k22=dt.*RHS_func_B(t(i)+(dt./2),u1_ABA+(k21/2));
u2_ABA = u2_ABA+k22;
u3_ABA = u2_ABA;
k31= (dt./2).*RHS_func_A(t(i),u3_ABA);
k32= (dt./2).*RHS_func_A(t(i)+(dt/2),u3_ABA+(k31/2));
u3_ABA = u3_ABA+k32;
u_ABA(i+1) = u3_ABA;
u1_ABA = u3_ABA;
end
```

```
%SYMMETRICALLY WEIGHTED SEQUENTIAL SPLITTING
for i = 1:Nt
k11_sym = dt.*RHS_func_A(t(i),u1_AB_sym);
k12_sym = dt.*RHS_func_A(t(i)+(dt./2),u1_AB_sym+(k11_sym/2));
u1_AB_sym = u1_AB_sym+k12_sym;
u2_AB_sym = u1_AB_sym;
k21_sym = dt.*RHS_func_B(t(i),u1_AB_sym);
k22_sym = dt.*RHS_func_B(t(i)+(dt./2),u1_AB_sym+(k21_sym/2));
u2_AB_sym = u2_AB_sym+k22_sym;
u_AB_sym(i+1) = u2_AB_sym;
u1_AB_sym = u2_AB_sym;
k31_sym = dt.*RHS_func_B(t(i),u1_BA_sym);
k32_sym = dt.*RHS_func_B(t(i)+(dt./2),u1_BA_sym+(k31_sym/2));
u1_BA_sym = u1_BA_sym+k32_sym;
u2_BA_sym = u1_BA_sym;
k41_sym = dt.*RHS_func_A(t(i),u1_BA_sym);
k42_sym = dt.*RHS_func_A(t(i)+(dt./2),u1_BA_sym+(k41_sym/2));
u2_BA_sym = u2_BA_sym+k42_sym;
u_BA_sym(i+1) = u2_BA_sym;
u1_BA_sym = u2_BA_sym;
end

for i=1:Nt+1
u_sym(i)=(u_AB_sym(i)+u_BA_sym(i))./2;
end

%NONSPLITTING
for i=1:Nt
u_appr(i+1)=u_appr(i)+dt*RHS_func(t(i),u_appr(i));
end
```

```
%PLOT
plot(t,u_exact,'color',[0 0.6 0.1])
plot(t,u_exact,'k+-')
hold on
plot(t,u_appr,'g*:')
hold on
plot(t,u_AB,'b')
hold on
plot(t,u_ABA,'r')
hold on
plot(t,u_sym,'mo:')
grid off
legend('Exact Solution','Nonsplitting','Lie-Trotter',...
  'Strang-Marchkuk','Symmetrically')

clear all
close all
clc

%INPUT
step = 4;
stime = 0;
ftime = 1;
initial_c = 1;

for j=1:step
dt(j)=10^(-j);

%DISCRETIZATION
Nt = (ftime - stime)/dt(j);
t = stime:dt(j):ftime;
```

```
%EXACT SOLUTION
u_exact = -3.*exp(-2.*t)+4.*exp(-t);

%ASSIGNING INITIAL DATAS
t(1) = stime;
u_appr(1) = initial_c;
u1_AB = initial_c;
u_AB(1) = u1_AB;
u1_ABA = initial_c;
u_ABA(1) = u1_ABA;

u1_AB_symm = initial_c;
u_AB_symm(1) = u1_AB_symm;
u1_BA_symm = initial_c;
u_BA_symm(1) = u1_BA_symm;

%NONSPLITTING
for i = 1:Nt
u_appr(i+1) = u_appr(i)+dt(j).*RHS_func(t(i),u_appr(i));
end

%LIE-TROTTER SPLITTING
for i = 1:Nt
u1_AB = u1_AB+dt(j).*RHS_func_A(t(i),u1_AB);
u2_AB = u1_AB;
u2_AB = u2_AB+dt(j).*RHS_func_B(t(i),u2_AB);
u_AB(i+1) = u2_AB;
u1_AB = u2_AB;
end
```

```
%STRANG-MARCHKUK SPLITTING
for i = 1:Nt
k11_ABA= (dt(j)./2).*RHS_func_A(t(i),u1_ABA);
k12_ABA= (dt(j)./2).*RHS_func_A(t(i)+(dt(j)./2),u1_ABA+(k11_ABA/2));
u1_ABA = u1_ABA+k12_ABA;
u2_ABA = u1_ABA;
k21_ABA=dt(j).*RHS_func_B(t(i),u1_ABA);
k22_ABA=dt(j).*RHS_func_B(t(i)+(dt(j)./2),u1_ABA+(k21_ABA/2));
u2_ABA = u2_ABA+k22_ABA;
u3_ABA = u2_ABA;
k31_ABA= (dt(j)./2).*RHS_func_A(t(i),u3_ABA);
k32_ABA= (dt(j)./2).*RHS_func_A(t(i)+(dt(j)/2),u3_ABA+(k31_ABA/2));
u3_ABA = u3_ABA+k32_ABA;
u_ABA(i+1) = u3_ABA;
u1_ABA = u3_ABA;
end

%LIE-TROTTER SPLITTING (AB SPLITTING)
for i = 1:Nt
k11_AB= dt(j).*RHS_func_A(t(i),u1_AB_symm);
k12_AB= dt(j).*RHS_func_A(t(i)+(dt(j)./2),u1_AB_symm+(k11_AB/2));
u1_AB_symm = u1_AB_symm+k12_AB;
u2_AB_symm= u1_AB_symm;
k21_AB=dt(j).*RHS_func_B(t(i),u1_AB_symm);
k22_AB=dt(j).*RHS_func_B(t(i)+(dt(j)./2),u1_AB_symm+(k21_AB/2));
u2_AB_symm = u2_AB_symm+k22_AB;
u_AB_symm(i+1) = u2_AB_symm;
u1_AB_symm = u2_AB_symm;
end
```

```
%LIE-TROTTER SPLITTING (BA SPLITTING)
for i = 1:Nt
k31_BA= dt(j).*RHS_func_B(t(i),u1_BA_symm);
k32_BA= dt(j).*RHS_func_B(t(i)+(dt(j)./2),u1_BA_symm+(k31_BA/2));
u1_BA_symm = u1_BA_symm+k32_BA;
u2_BA_symm = u1_BA_symm;
k41_BA=dt(j).*RHS_func_A(t(i),u1_BA_symm);
k42_BA=dt(j).*RHS_func_A(t(i)+(dt(j)./2),u1_BA_symm+(k41_BA/2));
u2_BA_symm = u2_BA_symm+k42_BA;
u_BA_symm(i+1) = u2_BA_symm;
u1_BA_symm = u2_BA_symm;
end

%SYMMETRICALLY WEIGHTED SEQUENTIAL SPLITTING
for i=1:Nt+1
u_symm(:,i)=(u_AB_symm(:,i)+u_BA_symm(:,i))./2;
end

%ERROR
E_lie(j) = norm(abs(u_exact-u_AB),inf);
E_strang(j)=norm(abs(u_exact-u_ABA),inf);
E_symm(j)=norm(abs(u_exact-u_symm),inf);
E_non(j)=norm(abs(u_exact-u_appr),inf);
end

%ORDER
for j=1:step-1
order_lie(j)=log(E_lie(j+1)/E_lie(j))/log(dt(j+1)/dt(j));
order_strang(j)=log(E_strang(j+1)/E_strang(j))/log(dt(j+1)/dt(j));
order_symm(j)=log(E_symm(j+1)/E_symm(j))/log(dt(j+1)/dt(j));
order_non(j)=log(E_non(j+1)/E_non(j))/log(dt(j+1)/dt(j));
end
```

```
%PLOT
plot(order_lie,'-*b','LineWidth',1)
xlabel('$j$','Interpreter','Latex');
ylabel('$\frac{\log(E(j+1)/E(j))}{\log(dt(j+1)/dt(j))}$',...
   'Interpreter','LaTex')
title('ORDER')

hold on

figure
s = linspace(1,step,step);
loglog(10.^(-s),abs(E_lie),'r-o',10.^(-s),abs(E_strang),'r-*',...
10.^(-s),abs(E_symm),'r-.',10.^(-s),abs(E_non),'r-+','LineWidth',1)
xlabel('Time Steps')
ylabel('Errors')
title('ERROR')
legend('Lie-Trotter','Strang-Marchkuk','Symmetrically','Nonsplitting')




function [ z ] = RHS_func(t,u)
z = -u+3.*exp(-2.*t);
end

function [ z ] = RHS_func_A(t,u)
z = -u;
end

function [ z ] = RHS_func_B(t,u)
z = 3.*exp(-2.*t);
end
```

CLASSICAL OPERATOR SPLITTING METHODS MATLAB CODES FOR NONLINEAR ODE

```
clear all
close all
clc

%INPUT
dt = 0.01;
stime = 0;
ftime = 1/2;
initial_c = 1;

%DISCRETIZATION
Nt = (ftime - stime)/dt;
t = stime:dt:ftime;

%EXACT SOLUTION
u_exact = -2./(3.*exp(t)-5*exp(t./3));

%ASSIGNING INITIAL DATAS
t(1) = stime;
u_appr(1) = initial_c;
u1_AB=initial_c;
u_AB(1)=u1_AB;
u1_ABA=initial_c;
u_ABA(1)=u1_ABA;
```

```
u1_AB_sym = initial_c;
u_AB_sym(1) = u1_AB_sym;
u1_BA_sym = initial_c;
u_BA_sym(1) = u1_BA_sym;

%LIE-TROTTER SPLITTING
for i=1:Nt
u1_AB=u1_AB+dt*RHS_func_A(t(i),u1_AB);
u2_AB=u1_AB;
u2_AB=u2_AB+dt*RHS_func_B(t(i),u2_AB);
u_AB(i+1)=u2_AB;
u1_AB=u2_AB;
end

%STRANG-MARCHKUK SPLITTING
for i=1:Nt
u1_ABA=u1_ABA+(dt/2)*RHS_func_A(t(i),u1_ABA);
u2_ABA=u1_ABA;
u2_ABA=u2_ABA+dt*RHS_func_B(t(i),u2_ABA);
u3_ABA=u2_ABA;
u3_ABA=u3_ABA+(dt/2)*RHS_func_A(t(i),u3_ABA);
u_ABA(i+1)=u3_ABA;
u1_ABA=u3_ABA;
end


%SYMMETRICALLY WEIGHTED SEQUENTIAL SPLITTING
for i = 1:Nt
k11_sym = dt.*RHS_func_A(t(i),u1_AB_sym);
k12_sym = dt.*RHS_func_A(t(i)+(dt./2),u1_AB_sym+(k11_sym/2));
u1_AB_sym = u1_AB_sym+k12_sym;
u2_AB_sym = u1_AB_sym;
```

```
k21_sym = dt.*RHS_func_B(t(i),u1_AB_sym);
k22_sym = dt.*RHS_func_B(t(i)+(dt./2),u1_AB_sym+(k21_sym/2));
u2_AB_sym = u2_AB_sym+k22_sym;
u_AB_sym(i+1) = u2_AB_sym;
u1_AB_sym = u2_AB_sym;
k31_sym = dt.*RHS_func_B(t(i),u1_BA_sym);
k32_sym = dt.*RHS_func_B(t(i)+(dt./2),u1_BA_sym+(k31_sym/2));
u1_BA_sym = u1_BA_sym+k32_sym;
u2_BA_sym = u1_BA_sym;
k41_sym = dt.*RHS_func_A(t(i),u1_BA_sym);
k42_sym = dt.*RHS_func_A(t(i)+(dt./2),u1_BA_sym+(k41_sym/2));
u2_BA_sym = u2_BA_sym+k42_sym;
u_BA_sym(i+1) = u2_BA_sym;
u1_BA_sym = u2_BA_sym;
end

for i=1:Nt+1
u_sym(i)=(u_AB_sym(i)+u_BA_sym(i))./2;
end

%NONSPLITTING
for i=1:Nt
u_appr(i+1)=u_appr(i)+dt*RHS_func(t(i),u_appr(i));
end

%PLOT
%plot(t,u_exact,'color',[0 0.6 0.1])
plot(t,u_exact,'k+-')
hold on
plot(t,u_appr,'g*:')
hold on
```

```
plot(t,u_AB,'b')
hold on
plot(t,u_ABA,'r')
hold on
plot(t,u_sym,'mo:')
grid off
legend('Exact Solution','Nonsplitting','Lie-Trotter',...
'Strang-Marchkuk','Symmetrically')


clear all
close all
clc

%INPUT
step = 4;
stime = 0;
ftime = 1/2;
initial_c = 1;

for j=1:step
dt(j)=10^(-j);

%DISCRETIZATION
Nt = (ftime - stime)/dt(j);
t = stime:dt(j):ftime;

%EXACT SOLUTION
u_exact = -2./(3.*exp(t)-5*exp(t./3));
```

```
%ASSIGNING INITIAL DATAS
t(1) = stime;
u_appr(1) = initial_c;
u1_AB = initial_c;
u_AB(1) = u1_AB;
u1_ABA = initial_c;
u_ABA(1) = u1_ABA;

u1_AB_symm = initial_c;
u_AB_symm(1) = u1_AB_symm;
u1_BA_symm = initial_c;
u_BA_symm(1) = u1_BA_symm;


%NONSPLITTING
for i = 1:Nt
u_appr(i+1) = u_appr(i)+dt(j).*RHS_func(t(i),u_appr(i));
end

%LIE-TROTTER SPLITTING
for i = 1:Nt
u1_AB = u1_AB+dt(j).*RHS_func_A(t(i),u1_AB);
u2_AB = u1_AB;
u2_AB = u2_AB+dt(j).*RHS_func_B(t(i),u2_AB);
u_AB(i+1) = u2_AB;
u1_AB = u2_AB;
end

%STRANG-MARCHKUK SPLITTING
for i = 1:Nt
k11_ABA= (dt(j)./2).*RHS_func_A(t(i),u1_ABA);
k12_ABA= (dt(j)./2).*RHS_func_A(t(i)+(dt(j)./2),u1_ABA+(k11_ABA/2));
```

```
u1_ABA = u1_ABA+k12_ABA;
u2_ABA = u1_ABA;
k21_ABA=dt(j).*RHS_func_B(t(i),u1_ABA);
k22_ABA=dt(j).*RHS_func_B(t(i)+(dt(j)./2),u1_ABA+(k21_ABA/2));
u2_ABA = u2_ABA+k22_ABA;
u3_ABA = u2_ABA;
k31_ABA= (dt(j)./2).*RHS_func_A(t(i),u3_ABA);
k32_ABA= (dt(j)./2).*RHS_func_A(t(i)+(dt(j)/2),u3_ABA+(k31_ABA/2));
u3_ABA = u3_ABA+k32_ABA;
u_ABA(i+1) = u3_ABA;
u1_ABA = u3_ABA;
end

%LIE-TROTTER SPLITTING (AB SPLITTING)
for i = 1:Nt
k11_AB= dt(j).*RHS_func_A(t(i),u1_AB_symm);
k12_AB= dt(j).*RHS_func_A(t(i)+(dt(j)./2),u1_AB_symm+(k11_AB/2));
u1_AB_symm = u1_AB_symm+k12_AB;
u2_AB_symm= u1_AB_symm;
k21_AB=dt(j).*RHS_func_B(t(i),u1_AB_symm);
k22_AB=dt(j).*RHS_func_B(t(i)+(dt(j)./2),u1_AB_symm+(k21_AB/2));
u2_AB_symm = u2_AB_symm+k22_AB;
u_AB_symm(i+1) = u2_AB_symm;
u1_AB_symm = u2_AB_symm;
end


%LIE-TROTTER SPLITTING (BA SPLITTING)
for i = 1:Nt
k31_BA= dt(j).*RHS_func_B(t(i),u1_BA_symm);
k32_BA= dt(j).*RHS_func_B(t(i)+(dt(j)./2),u1_BA_symm+(k31_BA/2));
```

```
u1_BA_symm = u1_BA_symm+k32_BA;
u2_BA_symm = u1_BA_symm;
k41_BA=dt(j).*RHS_func_A(t(i),u1_BA_symm);
k42_BA=dt(j).*RHS_func_A(t(i)+(dt(j)./2),u1_BA_symm+(k41_BA/2));
u2_BA_symm = u2_BA_symm+k42_BA;
u_BA_symm(i+1) = u2_BA_symm;
u1_BA_symm = u2_BA_symm;
end

%SYMMETRICALLY WEIGHTED SEQUENTIAL SPLITTING
for i=1:Nt+1
u_symm(:,i)=(u_AB_symm(:,i)+u_BA_symm(:,i))./2;
end

%ERROR
E_lie(j) = norm(abs(u_exact-u_AB),inf);
E_strang(j)=norm(abs(u_exact-u_ABA),inf);
E_symm(j)=norm(abs(u_exact-u_symm),inf);
E_non(j)=norm(abs(u_exact-u_appr),inf);
end

%ORDER
for j=1:step-1
order_lie(j)=log(E_lie(j+1)/E_lie(j))/log(dt(j+1)/dt(j));
order_strang(j)=log(E_strang(j+1)/E_strang(j))/log(dt(j+1)/dt(j));
order_symm(j)=log(E_symm(j+1)/E_symm(j))/log(dt(j+1)/dt(j));
order_non(j)=log(E_non(j+1)/E_non(j))/log(dt(j+1)/dt(j));
end
```

```
%PLOT
plot(order_lie,'-*b','LineWidth',1)
xlabel('$j$','Interpreter','Latex');
ylabel('$\frac{\log(E(j+1)/E(j))}{\log(dt(j+1)/dt(j))}$',...
  'Interpreter','LaTex')
title('ORDER')

hold on

figure
s = linspace(1,step,step);
loglog(10.^(-s),abs(E_lie),'r-o',10.^(-s),abs(E_strang),'r*',...
10.^(-s),abs(E_symm),'r-.',10.^(-s),abs(E_non),'r-+','LineWidth',1)
xlabel('Time Steps')
ylabel('Errors')
title('ERROR')
legend('Lie-Trotter','Strang-Marchkuk','Symmetrically','Nonsplitting')



function [ z ] = RHS_func(t,u)
z = -(1/3).*u+exp(t).*u.^2;
end

function [ z ] = RHS_func_A(t,u)
z = -u./3;
end

function [ z ] = RHS_func_B(t,u)
z = exp(t).*(u.^2);
end
```

CLASSICAL OPERATOR SPLITTING METHODS MATLAB CODES FOR A KINETIC MODEL OF PHAGE INFECTION

```
clear all
clc
close all

%INPUT
dt = 0.1;
stime = 0;
ftime = 5;

x_initial_c = 1.9*10^4;  %the concentration of uninfected bacteria
y_initial_c = 5.4*10^3;  %the lytic bacteria
v_initial_c = 7.4*10^4;  %free phage

%DISCRETIZATION
Nt = (ftime - stime)/dt;
t = stime:dt:ftime;

%DETERMINE COEFFICIENT VALUES
a = 0.3;       %replication coefficient of bacteria
b = 10^(-6);   %the transmission coefficient
k = 0.706;     %the lysis rate coefficient
L = 15;        %the burst size
m = 34.8;      %the decay rate of free phage

%MATRIX
A = [a 0 0; 0 a-k 0; 0 k*L -m];
```

```
%ASSIGNING INITIAL CONDITIONS
u_AB_initial = [x_initial_c; y_initial_c; v_initial_c];
u1_AB = u_AB_initial;
u_AB(:,1) = u1_AB;


%LIE-TROTTER SPLITTING (AB SPLITTING)
for i=1:Nt
u1_AB = expm(A.*dt)*u1_AB;
u2_AB = u1_AB;
u2_AB = u2_AB+dt*B(u2_AB(1),u2_AB(3),b);
u_AB(:,i+1) = u2_AB;
u1_AB = u2_AB;
end

x_appr_AB = u_AB(1,:)' ;
y_appr_AB = u_AB(2,:)' ;
v_appr_AB = u_AB(3,:)' ;

%PLOT
plot(t,x_appr_AB,'b','LineWidth',1)
hold on
plot(t,y_appr_AB,'r','LineWidth',1)
hold on
plot(t,v_appr_AB,'color',[0.3 0.8 0],'LineWidth',1)
legend('x-appr','y-appr','v-appr')
title('Bacteriaphage Concentrations');
xlabel('times (hour)');
ylabel('concentration');
```

```
%ASSIGNING INITIAL CONDITIONS
u_ABA_initial = [x_initial_c; y_initial_c; v_initial_c];
u1_ABA(:,1) = u_ABA_initial;
u_ABA(:,1) = u1_ABA;

%STRANG-MARCHKUK SPLITTING (ABA SPLITTING)
for i=1:Nt
u1_ABA = expm(A.*(dt/2))*u1_ABA;
u2_ABA = u1_ABA;
k1 = dt.*B(u2_ABA(1),u2_ABA(3),b);
k2 = dt.*B(u2_ABA(1)+k1(1)./2,u2_ABA(3)+k1(3)./2,b);
u2_ABA = u2_ABA+k2;
u3_ABA = u2_ABA;
u3_ABA = expm(A.*(dt/2))*u2_ABA;
u_ABA(:,i+1) = u3_ABA;
u1_ABA = u3_ABA;
end

x_appr_ABA = u_ABA(1,:)' ;
y_appr_ABA= u_ABA(2,:)' ;
v_appr_ABA = u_ABA(3,:)' ;

%PLOT
plot(t,x_appr_ABA,'b','LineWidth',1)
hold on
plot(t,y_appr_ABA,'r','LineWidth',1)
hold on
plot(t,v_appr_ABA,'color',[0.3 0.8 0],'LineWidth',1)
legend('x-appr','y-appr','v-appr')
title('Bacteriaphage Concentrations');
xlabel('times (hour)');
ylabel('concentration');
```

```
%ASSIGNING INITIAL CONDITIONS
u_AB_initial = [x_initial_c; y_initial_c; v_initial_c];

u1_AB = u_AB_initial;
u_AB(:,1) = u1_AB;

u1_BA = u_AB_initial;
u_BA(:,1)= u1_BA;

%LIE-TROTTER SPLITTING (AB SPLITTING)
for i=1:Nt
u1_AB=expm(A.*dt)*u1_AB;
u2_AB=u1_AB;
k11=dt.*B(u2_AB(1,:),u2_AB(3,:),b);
k12=dt.*B(u2_AB(1,:)+dt./2,u2_AB(3,:)+(k11(1,:)./2),b);
u2_AB=u2_AB+k12;
u_AB(:,i+1)=u2_AB;
u1_AB=u2_AB;
end

%LIE-TROTTER SPLITTING (BA SPLITTING)
for i=1:Nt
k21=dt.*B(u1_BA(1,:),u1_BA(3,:),b);
k22=dt.*B(u1_BA(1,:)+dt./2,u1_BA(3,:)+(k21(1,:)./2),b);
u1_BA=u1_BA+k22;
u2_BA=u1_BA;
u2_BA=expm(A.*dt)*u2_BA;
u_BA(:,i+1)=u2_BA;
u1_BA=u2_BA;
end
```

```
%SYMMETRICALLY WEIGHTED SEQUENTIAL SPLITTING
for i=1:Nt+1
u_symm(:,i)=(u_AB(:,i)+u_BA(:,i))./2;
end

x_appr_symm = u_symm(1,:)' ;
y_appr_symm = u_symm(2,:)' ;
v_appr_symm = u_symm(3,:)' ;



%PLOT
plot(t,x_appr_symm,'b','LineWidth',1)
hold on
plot(t,y_appr_symm,'r','LineWidth',1)
hold on
plot(t,v_appr_symm,'color',[0.3 0.8 0],'LineWidth',1)
legend('x-appr','y-appr','v-appr')
title('Bacteriaphage Concentrations');
xlabel('times (hour)');
ylabel('concentration');

clear all
clc
close all

%INPUT
step = 4;

%TIME
dt_full = 10^(-6);
stime = 0;
ftime = 5;
```

```
Nt_full =  (ftime - stime)/dt_full;
t_full = stime:dt_full:ftime;

%INITIAL CONDITIONS
x_initial_c = 1.9*10^4;  %the concentration of uninfected bacteria
y_initial_c = 5.4*10^3;  %the lytic bacteria
v_initial_c = 7.4*10^4;  %free phage

for e=1:step
%DISCRETIZATION
dt_split(e) = 10^(-e);

Nt_split = (ftime - stime)/dt_split(e);
t_split = stime:dt_split(e):ftime;

x = zeros(1,Nt_full);
y = zeros(1,Nt_full);
v = zeros(1,Nt_full);
x(1) = x_initial_c;
y(1) = y_initial_c;
v(1) = v_initial_c;


%DETERMINE COEFFICIENT VALUES
a = 0.3;       %replication coefficient of bacteria
b = 10^(-6);   %the transmission coefficient
k = 0.706;     %the lysis rate coefficient
L = 15;        %the burst size
m = 34.8;      %the decay rate of free phage
```

```
%ASSIGNING INITIAL CONDITIONS
u1_initial=[x_initial_c; y_initial_c; v_initial_c];
u(:,1) = u1_initial;

%MATRIX
A = [a 0 0; 0 a-k 0; 0 k*L -m];

u1_initial_AB=[x_initial_c; y_initial_c; v_initial_c];
u_AB(:,1) = u1_initial_AB;

%NONSPLITTING
for i=1:Nt_full
k1=F(x(i),y(i),v(i),a,b,k,L,m);
k2=F(x(i)+(dt_full/2)*k1(1),y(i)+(dt_full/2)*k1(2),v(i)+...
   (dt_full/2)*k1(3),a,b,k,L,m);
k3=F(x(i)+(dt_full/2)*k2(1),y(i)+(dt_full/2)*k2(2),v(i)+...
   (dt_full/2)*k2(3),a,b,k,L,m);
k4=F(x(i)+dt_full*k3(1),y(i)+dt_full*k3(2),v(i)+dt_full*k3(3),a,b,k,L,m);
u(:,i+1)=u(:,i)+(dt_full/6)*(k1+2.*k2+2.*k3+k4);
x(i+1)=u(1,i+1); y(i+1)=u(2,i+1); v(i+1)=u(3,i+1);
end

%LIE-TROTTER SPLITTING
for i=1:Nt_split
u1_AB=expm(A.*dt_split(e))*u1_initial_AB;
u2_AB=u1_AB;
u2_AB=u2_AB+dt_split(e)*B(u2_AB(1),u2_AB(3),b);
u_AB(:,i+1)=u2_AB;
u1_initial_AB=u2_AB;
end
```

```
x_appr=u_AB(1,:)' ;
y_appr=u_AB(2,:)' ;
v_appr=u_AB(3,:)' ;

errx(e)=norm(abs(x(end)'-x_appr(end)),inf);
erry(e)=norm(abs(y(end)'-y_appr(end)),inf);
errv(e)=norm(abs(v(end)'-v_appr(end)),inf);
end

for j=1:step-1
orderx(j)=log(errx(j+1)/errx(j))/log(dt_split(j+1)/dt_split(j));
ordery(j)=log(erry(j+1)/erry(j))/log(dt_split(j+1)/dt_split(j));
orderv(j)=log(errv(j+1)/errv(j))/log(dt_split(j+1)/dt_split(j));
end

 %PLOT
 plot(order1,'-*b','LineWidth',1)
 xlabel('$j$','Interpreter','Latex');
 ylabel('$\frac{\log(E(j+1)/E(j))}{\log(dt(j+1)/dt(j))}$',...
   'Interpreter','LaTex')
 title('ORDER')

 hold on

 figure
 s = linspace(1,step,step);
 loglog(10.^(-s),abs(E1),'r-*','LineWidth',1)
 xlabel('Time Steps')
 ylabel('Errors')
 title('ERROR')
```

```
figure
s = linspace(1,step,step);
loglog(10.^(-s),abs(errx),'b*-',10.^(-s),abs(erry),'r*-',...
   10.^(-s),abs(errv),'g*-','LineWidth',1)
xlabel('Time Steps')
ylabel('Errors')
title('ERROR')
legend('x(t)','y(t)','v(t)')

clear all
clc
close all

%INPUT
step = 4;

%TIME
dt_full = 10^(-6);
stime = 0;
ftime = 5;
Nt_full =  (ftime - stime)/dt_full;
t_full = stime:dt_full:ftime;

%INITIAL CONDITIONS
x_initial_c = 1.9*10^4;  %the concentration of uninfected bacteria
y_initial_c = 5.4*10^3;  %the lytic bacteria
v_initial_c = 7.4*10^4;  %free phage

for e=1:step
%DISCRETIZATION
dt_split(e) = 10^(-e);
```

```
Nt_split = (ftime - stime)/dt_split(e);
t_split = stime:dt_split(e):ftime;

x = zeros(1,Nt_full);
y = zeros(1,Nt_full);
v = zeros(1,Nt_full);
x(1) = x_initial_c;
y(1) = y_initial_c;
v(1) = v_initial_c;


%DETERMINE COEFFICIENT VALUES
a = 0.3;      %replication coefficient of bacteria
b = 10^(-6);  %the transmission coefficient
k = 0.706;    %the lysis rate coefficient
L = 15;       %the burst size
m = 34.8;     %the decay rate of free phage

%ASSIGNING INITIAL CONDITIONS
u1_initial=[x_initial_c; y_initial_c; v_initial_c];
u(:,1) = u1_initial;

%MATRIX
A = [a 0 0; 0 a-k 0; 0 k*L -m];

%ASSIGNING INITIAL CONDITIONS
u_ABA_initial = [x_initial_c; y_initial_c; v_initial_c];
u1_ABA(:,1) = u_ABA_initial;
u_ABA(:,1) =  u_ABA_initial;
```

```
%NONSPLITTING
for i=1:Nt_full
k1=F(x(i),y(i),v(i),a,b,k,L,m);
k2=F(x(i)+(dt_full/2)*k1(1),y(i)+(dt_full/2)*k1(2),v(i)+...
   (dt_full/2)*k1(3),a,b,k,L,m);
k3=F(x(i)+(dt_full/2)*k2(1),y(i)+(dt_full/2)*k2(2),v(i)+...
   (dt_full/2)*k2(3),a,b,k,L,m);
k4=F(x(i)+dt_full*k3(1),y(i)+dt_full*k3(2),v(i)+dt_full*k3(3),a,b,k,L,m);
u(:,i+1)=u(:,i)+(dt_full/6)*(k1+2.*k2+2.*k3+k4);
x(i+1)=u(1,i+1); y(i+1)=u(2,i+1); v(i+1)=u(3,i+1);
end

%STRANG-MARCHKUK SPLITTING (ABA SPLITTING)
for i=1:Nt_split
u1_ABA = expm(A.*(dt_split(e)/2))*u1_ABA;
u2_ABA = u1_ABA;
k1 = dt_split(e).*B(u2_ABA(1),u2_ABA(3),b);
k2 = dt_split(e).*B(u2_ABA(1)+k1(1)./2,u2_ABA(3)+k1(3)./2,b);
u2_ABA = u2_ABA+k2;
u3_ABA = u2_ABA;
u3_ABA = expm(A.*(dt_split(e)/2))*u2_ABA;
u_ABA(:,i+1) = u3_ABA;
u1_ABA = u3_ABA;
end

x_appr_ABA = u_ABA(1,:)’ ;
y_appr_ABA= u_ABA(2,:)’ ;
v_appr_ABA = u_ABA(3,:)’ ;
```

```
errx(e)=norm(abs(x(end)'-x_appr_ABA(end)),inf);
erry(e)=norm(abs(y(end)'-y_appr_ABA(end)),inf);
errv(e)=norm(abs(v(end)'-v_appr_ABA(end)),inf);
end

for j=1:step-1
orderx(j)=log(errx(j+1)/errx(j))/log(dt_split(j+1)/dt_split(j));
ordery(j)=log(erry(j+1)/erry(j))/log(dt_split(j+1)/dt_split(j));
orderv(j)=log(errv(j+1)/errv(j))/log(dt_split(j+1)/dt_split(j));
end

 %PLOT
 plot(order1,'-*b','LineWidth',1)
 xlabel('$j$','Interpreter','Latex');
 ylabel('$\frac{\log(E(j+1)/E(j))}{\log(dt(j+1)/dt(j))}$',...
 'Interpreter','LaTex')
 title('ORDER')

 hold on

figure
s = linspace(1,step,step);
loglog(10.^(-s),abs(errx),'b*-',10.^(-s),abs(erry),'r*-',...
10.^(-s),abs(errv),'g*-','LineWidth',1)
xlabel('Time Steps')
ylabel('Errors')
title('ERROR')
legend('x(t)','y(t)','v(t)')
```

```
clear all
clc
close all

%INPUT
step = 4;

%TIME
dt_full = 10^(-6);
stime = 0;
ftime = 5;
Nt_full =  (ftime - stime)/dt_full;
t_full = stime:dt_full:ftime;

%INITIAL CONDITIONS
x_initial_c = 1.9*10^4;  %the concentration of uninfected bacteria
y_initial_c = 5.4*10^3;  %the lytic bacteria
v_initial_c = 7.4*10^4;  %free phage


for e=1:step
dt_split(e) = 10^(-e);

%DISCRETIZATION
Nt_split = (ftime - stime)/dt_split(e);
t_split = stime:dt_split(e):ftime;

x = zeros(1,Nt_full);
y = zeros(1,Nt_full);
v = zeros(1,Nt_full);
```

```
x(1) = x_initial_c;
y(1) = y_initial_c;
v(1) = v_initial_c;

%DETERMINE COEFFICIENT VALUES
a = 0.3;      %replication coefficient of bacteria
b = 10^(-6);  %the transmission coefficient
k = 0.706;    %the lysis rate coefficient
L = 15;       %the burst size
m = 34.8;     %the decay rate of free phage

%ASSIGNING INITIAL CONDITIONS
u1_initial=[x_initial_c; y_initial_c; v_initial_c];
u(:,1) = u1_initial;

%MATRIX
A = [a 0 0; 0 a-k 0; 0 k*L -m];

%ASSIGNING INITIAL CONDITIONS
u_initial_AB = [x_initial_c; y_initial_c; v_initial_c];
u_initial_BA= [x_initial_c; y_initial_c; v_initial_c];

u1_AB(:,1) = u_initial_AB;
u_AB(:,1) =  u_initial_AB;
u1_BA(:,1)= u_initial_BA;
u_BA(:,1)= u_initial_BA;

%NONSPLITTING
for i=1:Nt_full
k1=F(x(i),y(i),v(i),a,b,k,L,m);
k2=F(x(i)+(dt_full/2)*k1(1),y(i)+(dt_full/2)*k1(2),v(i)+...
   (dt_full/2)*k1(3),a,b,k,L,m);
```

```
k3=F(x(i)+(dt_full/2)*k2(1),y(i)+(dt_full/2)*k2(2),v(i)+...
   (dt_full/2)*k2(3),a,b,k,L,m);
k4=F(x(i)+dt_full*k3(1),y(i)+dt_full*k3(2),v(i)+dt_full*k3(3),a,b,k,L,m);
u(:,i+1)=u(:,i)+(dt_full/6)*(k1+2.*k2+2.*k3+k4);
x(i+1)=u(1,i+1); y(i+1)=u(2,i+1); v(i+1)=u(3,i+1);
end


%LIE-TROTTER SPLITTING (AB SPLITTING)
for i=1:Nt_split
u1_AB=expm(A.*dt_split(e))*u1_AB;
u2_AB=u1_AB;
k11=dt_split(e).*B(u2_AB(1),u2_AB(3),b);
k12=dt_split(e).*B(u2_AB(1)+k11(1)./2,u2_AB(3,:)+k11(3)./2,b);
u2_AB=u2_AB+k12;
u_AB(:,i+1)=u2_AB;
u1_AB=u2_AB;
end


%LIE-TROTTER SPLITTING (BA SPLITTING)
for i=1:Nt_split
k21=dt_split(e).*B(u1_BA(1),u1_BA(3),b);
k22=dt_split(e).*B(u1_BA(1)+k21(1)./2,u1_BA(3)+k21(3)./2,b);
u1_BA=u1_BA+k22;
u2_BA=u1_BA;
u2_BA=expm(A.*dt_split(e))*u2_BA;
u_BA(:,i+1)=u2_BA;
u1_BA=u2_BA;
end
```

```
%SYMMETRICALLY WEIGHTED SEQUENTIAL SPLITTING
for i=1:Nt_split+1
u_symm(:,i)=(u_AB(:,i)+u_BA(:,i))./2;
end

x_appr_symm = u_symm(1,:)' ;
y_appr_symm = u_symm(2,:)' ;
v_appr_symm = u_symm(3,:)' ;



errx(e)=norm(abs(x(end)'-x_appr_symm(end)),inf);
erry(e)=norm(abs(y(end)'-y_appr_symm(end)),inf);
errv(e)=norm(abs(v(end)'-v_appr_symm(end)),inf);
end

for j=1:step-1
orderx(j)=log(errx(j+1)/errx(j))/log(dt_split(j+1)/dt_split(j));
ordery(j)=log(erry(j+1)/erry(j))/log(dt_split(j+1)/dt_split(j));
orderv(j)=log(errv(j+1)/errv(j))/log(dt_split(j+1)/dt_split(j));
end

%PLOT
plot(order1,'-*b','LineWidth',1)
xlabel('$j$','Interpreter','Latex');
 ylabel('$\frac{\log(E(j+1)/E(j))}{\log(dt(j+1)/dt(j))}$',...
     'Interpreter','LaTex')
title('ORDER')

hold on
```

```
figure
s = linspace(1,step,step);
loglog(10.^(-s),abs(errx),'b*-',10.^(-s),abs(erry),'r*-',...
    10.^(-s),abs(errv),'g*-','LineWidth',1)
xlabel('Time Steps')
ylabel('Errors')
title('ERROR')
legend('x(t)','y(t)','v(t)')



function [z]= B(x,v,b)
z = [-b*v*x; b*v*x; -b*v*x];
end

function [z]= F(x,y,v,a,b,k,L,m)
z = [a.*x-b.*v.*x; a.*y+b.*v.*x-k.*y; k.*L.*y-b.*v.*x-m.*v];
end
```

CLASSICAL OPERATOR SPLITTING METHODS MATLAB CODES FOR THE NEWELL-WHITEHEAD-SEGEL EQUATION

```
clear all
close all
clc

%INPUT
eps=0.1;
epsnew=eps.^2;
```

```
%SPACE DISCRETIZATION
x0 = 0; xf = 2*pi;
dx=pi/64;
Nx = (xf - x0)./dx;
x = x0:dx:xf;

%TIME DISCRETIZATION
stime = 0; ftime = 1;
dt = 0.001;
Nt = (ftime-stime)/dt;
t = stime:dt:ftime;

lambda = 1./dx^2;
A = fin(Nx);
AA = epsnew.*lambda.*A;

%ASSIGNING INITIAL CONDITIONS
u_initial(:,1)=(0.05).*sin(x);
u1_AB(:,1) = u_initial(2:Nx,1);
u1_ABA(:,1) = u_initial(2:Nx,1);

u1_AB_symm(:,1) = u_initial(2:Nx,1);
u1_BA_symm(:,1)=u_initial(2:Nx,1);



%LIE-TROTTER SPLITTING (AB SPLITTING)
for i=1:Nt
u1_AB(:,i+1)=expm(AA.*dt)*u1_AB(:,i);
u2_AB(:,i)=u1_AB(:,i+1);
u2_AB(:,i+1)=u2_AB(:,i)+dt.*f(u2_AB(:,i));
u1_AB(:,i+1)=u2_AB(:,i+1);
end
```

```
BCs=zeros(1,Nt+1);
u_appr=[BCs;u1_AB;BCs];

%STRANG-MARCHUK SPLITTING
for i=1:Nt
u1_ABA(:,i+1)=expm(AA.*(dt/2))*u1_ABA(:,i);
u2_ABA(:,i)=u1_ABA(:,i+1);
k11=dt.*f(u2_ABA(:,i));
k12=dt.*f(u2_ABA(:,i)+k11/2);
u2_ABA(:,i+1)=u2_ABA(:,i)+k12;
u3_ABA(:,i)=u2_ABA(:,i+1);
u3_ABA(:,i+1)=expm(AA.*(dt/2))*u3_ABA(:,i);
u1_ABA(:,i+1)=u3_ABA(:,i+1);
end

u1_ABA(:,Nt+1);

BCs=zeros(1,Nt+1);
u_appr=[BCs;u1_ABA;BCs];


%LIE-TROTTER SPLITTING(AB SPLITTING)
for i=1:Nt
u1_AB_symm(:,i+1)=expm(AA.*dt)*u1_AB_symm(:,i);
u2_AB_symm(:,i)=u1_AB_symm(:,i+1);
k11_symm=dt.*f(u2_AB_symm(:,i));
k12_symm=dt.*f(u2_AB_symm(:,i)+k11_symm/2);
u2_AB_symm(:,i+1)=u2_AB_symm(:,i)+k12_symm;
u1_AB_symm(:,i+1)=u2_AB_symm(:,i+1);
end
```

```
%LIE-TROTTER SPLITTING(BA SPLITTING)
for i=1:Nt
k21_symm=dt.*f(u1_BA_symm(:,i));
k22_symm=dt.*f(u1_BA_symm(:,i)+k21_symm/2);
u1_BA_symm(:,i+1)=u1_BA_symm(:,i)+k22_symm;
u2_BA_symm(:,i)=u1_BA_symm(:,i+1);
u2_BA_symm(:,i+1)=expm(AA.*dt)*u2_BA_symm(:,i);
u1_BA_symm(:,i+1)=u2_BA_symm(:,i+1);
end

for i=1:Nt+1
u_symm(:,i)=(u1_AB_symm(:,i)+u1_BA_symm(:,i))./2;
end

u_symm(:,Nt+1);

BCs=zeros(1,Nt+1);
u_appr=[BCs;u_symm;BCs];



%PLOT
mesh(t,x,u_appr)
title('Approximate Solution')

clear all
close all
clc

%INPUT
k=0.01;
```

```
%SPACE DISCRETIZATION
x0 = 0; xf = 2*pi;
dx=pi/64;
Nx = (xf - x0)./dx;
x = x0:dx:xf;

%TIME DISCRETIZATION
stime = 0; ftime = 1;

for e=1:4

dt(e) = 10^(-e);
Nt = (ftime-stime)/dt(e);
t = stime:dt(e):ftime;

dtnonsplit=10^(-4);
Ntnonsplit=(ftime-stime)/dtnonsplit;
tnonsplit=stime:dtnonsplit:ftime;

lambda = 1./dx^2;
A = fin(Nx);
AA = k.*lambda.*A;
C = lambda*[0;zeros(Nx-3,1);0];

%ASSIGNING INITIAL CONDITIONS
u_initial(:,1)=(0.05).*sin(x);
u1_AB(:,1) = u_initial(2:Nx,1);
u1_ABA(:,1) = u_initial(2:Nx,1);
u1_BA(:,1)=u_initial(2:Nx,1);
ex(:,1)=u_initial(2:Nx,1);
```

```
%NONSPLITTING
for i=1:Ntnonsplit
k1 = f2(AA,C,ex(:,i));
k2 = f2(AA,C,ex(:,i)+0.5*dtnonsplit*k1);
k3 = f2(AA,C,ex(:,i)+0.5*dtnonsplit*k2);
k4 = f2(AA,C,ex(:,i)+k3*dtnonsplit);
ex(:,i+1) = ex(:,i) + (1/6)*(k1+2*k2+2*k3+k4)*dtnonsplit;
end

%LIE-TROTTER SPLITTING
for i=1:Nt
u1_AB(:,i+1)=expm(AA.*dt(e))*u1_AB(:,i);
u2_AB(:,i)=u1_AB(:,i+1);
u2_AB(:,i+1)=u2_AB(:,i)+dt(e).*f(u2_AB(:,i));
u1_AB(:,i+1)=u2_AB(:,i+1);
end

%STRANG-MARCHUK SPLITTING
for i=1:Nt
u1_ABA(:,i+1)=expm(AA.*(dt(e)/2))*u1_ABA(:,i);
u2_ABA(:,i)=u1_ABA(:,i+1);
k111=dt(e).*f(u2_ABA(:,i));
k121=dt(e).*f(u2_ABA(:,i)+k111/2);
u2_ABA(:,i+1)=u2_ABA(:,i)+k121;
u3_ABA(:,i)=u2_ABA(:,i+1);
u3_ABA(:,i+1)=expm(AA.*(dt(e)/2))*u3_ABA(:,i);
u1_ABA(:,i+1)=u3_ABA(:,i+1);
end
```

```
%LIE-TROTTER SPLITTING(AB SPLITTING)
for i=1:Nt
u1_AB(:,i+1)=expm(AA.*dt(e))*u1_AB(:,i);
u2_AB(:,i)=u1_AB(:,i+1);
k11=dt(e).*f(u2_AB(:,i));
k12=dt(e).*f(u2_AB(:,i)+k11/2);
u2_AB(:,i+1)=u2_AB(:,i)+k12;
u1_AB(:,i+1)=u2_AB(:,i+1);
end

%LIE-TROTTER SPLITTING(BA SPLITTING)
for i=1:Nt
k21=dt(e).*f(u1_BA(:,i));
k22=dt(e).*f(u1_BA(:,i)+k21/2);
u1_BA(:,i+1)=u1_BA(:,i)+k22;
u2_BA(:,i)=u1_BA(:,i+1);
u2_BA(:,i+1)=expm(AA.*dt(e))*u2_BA(:,i);
u1_BA(:,i+1)=u2_BA(:,i+1);
end

for i=1:Nt+1
u_symm(:,i)=(u1_AB(:,i)+u1_BA(:,i))./2;
end

u_exact=[zeros(1,Ntnonsplit+1);ex;zeros(1,Ntnonsplit+1)];

u1_AB(:,Nt+1);

BCs=zeros(1,Nt+1);
u_appr_lie=[BCs;u1_AB;BCs];
u_appr_strang=[BCs;u1_ABA;BCs];
u_appr_symm=[BCs;u_symm;BCs];
```

```
E1(e)=norm(abs(u_appr_lie(:,end)-u_exact(:,end)),inf);
E2(e)=norm(abs(u_appr_strang(:,end)-u_exact(:,end)),inf);
E3(e)=norm(abs(u_appr_symm(:,end)-u_exact(:,end)),inf);
end

for e=1:3
order1(e)=log(E1(e+1)/E1(e))/log(dt(e+1)/dt(e));
order2(e)=log(E2(e+1)/E2(e))/log(dt(e+1)/dt(e));
order3(e)=log(E3(e+1)/E3(e))/log(dt(e+1)/dt(e));
end

figure
s = linspace(1,4,4);
loglog(10.^(-s),abs(E1),'ro-',10.^(-s),abs(E2),'r*',...
10.^(-s),abs(E3),'r-.','LineWidth',1)
xlabel('Time Steps')
ylabel('Errors')
title('ERROR')
legend('Lie-Trotter','Strang-Marchuk','SWS')


function [ z ] = f(u)
z = u-u.^3;
end

function  z  = f2(AA,C,u)
p=diag(u);
z = AA*u+C+p*(1-u.^2);
end
```

```
function A=fin(N)
%A=zeros(N-1,N-1);
for i=1:N-1
for j=1:N-1
if i==j
A(i,j)=-2;
end
if (i-j)==1
A(i,j)=1;
end
if (i-j)==-1
A(i,j)=1;
end
end
end
A;
```