Numerical Methods for Solving Systems of Ordinary Differential Equations

Simruy Hürol

Submitted to the
Institute of Graduate Studies and Research
in partial fulfillment of the requirements for the Degree of

Master of Science
in
Applied Mathematics and Computer Science

Eastern Mediterranean University
January 2013
Gazimağusa, North Cyprus

Approval of the Institute of Graduate Studies and Research

Prof. Dr. Elvan Yılmaz
Director

I certify that this thesis satisfies the requirements as a thesis for the degree of Master of Science in Applied Mathematics and Computer Science.

Prof. Dr. Nazım Mahmudov
Chair, Department of Mathematics

We certify that we have read this thesis and that in our opinion it is fully adequate in scope and quality as a thesis for the degree of Master of Science in Applied Mathematics and Computer Science.

Asst. Prof. Dr. Mehmet Bozer
Supervisor

Examining Committee

1. Assoc. Prof. Dr. Derviş Subaşı __________________________

2. Asst. Prof. Dr. Mehmet Bozer __________________________

3. Asst. Prof. Dr. Mustafa Rıza __________________________

ABSTRACT

This thesis concantrates on numerical methods for solving ordinary differential equations. Firstly, we discuss the concept of convergence, local-truncation error, global-truncation error, consistency, types of stability, zero-stability, and weak-stability. Afterwards, we inform some materials for Euler and Runge-Kutta method. The given ordinary differential equation is analyzed on Euler and Runge-Kutta method to find the approximated solution with the given initial conditions. Then, the stability of each method is examined briefly. We also focus on numerical methods for systems. Then, the reason of the stiff system is discussed. After investigating the numerical methods, we gave advantages and disadvantages of Euler method and Fourth Order Runge-Kutta method. Finally, numerical experiments is applied on Explicit Euler method and Explicit Fourth Order Runge-Kutta method. The approximated solutions with different step-size and analytical solutions of methods are computed in Matlab software. The computation of approximated solutions of methods are compared with analytical solutions. Then we discussed the accuracy of these methods when they are applied to the specified system in Chapter 7. Finally, we conclude that Explicit Fourth Order Runge-Kutta method is more accurate than the Explicit Euler method.

Keywords: Ordinary Differential Equations, Numerical solutions, Euler's method, Runge-Kutta method, Stiff System

ÖZ

Bu çalışmada adi diferansiyel denklemlerin çözümü için sayısal yöntemler irdelenmiştir. İlk olarak, yakınsama kavramı, yerel kesme hatası, küresel kesme hatası, tutarlılık, kararlılık türleri, sıfır kararlılık ve zayıf kararlılık kavramları incelenmiştir. Ayrıca, Euler ve Runge-Kutta metodları verilmiştir. Verilen bir diferansiyel denklemin, yaklaşık çözümünü bulmak için Euler ve Runge-Kutta yöntemi verilen başlangıç koşulları ile analiz edilmiş ve her bir yöntem için kararlılık kısaca ele alınmıştır. Daha sonra, verilen sistemler için sayısal yöntemlere odaklanılmış ve sert sistemin çıkış sebebi incelenmiştir. Euler yöntemin ve dördüncü dereceden Runge-Kutta yöntemin avantajları ve dezavantajları verilmiştir. Son olarak, sayısal deneyler üzerinde Açık Euler ve Açık dördüncü dereceden Runge-Kutta yöntemleri uygulanmıştır. Farklı adım büyüklüğü ele alınarak yaklaşılan çözümler ve yöntemlerin analitik çözümleri Matlab yazılımıkullanılarak hesaplanmıştır. Elde edilen yaklaşılır çözümler ile analitik çözümler karşılaştırılmıştır. Daha sonra yöntemler Bölüm 7'de belirtilen sistem üzerine uygulanıp yöntemlerin doğruluğu tartışılmıştır. Son olarak, Açık Runge-Kutta yöntemin yaklaştırılmış çözümünün Açık Euler methoduna göre daha az hatalı olduğu sonucuna varılmıştır.

Anahtar Kelimeler: Adi Diferensiyel Denklemler, Sayısal Çözümler, Euler Yöntemi, Runge-Kutta Yöntemi, Sert Sistemi

ACKNOWLEDGMENTS

First of all, I would like to express my gratidute to my supervisor, Dr. Mehmet Bozer for his recommendation, patient, and encouragement during this thesis. Special thanks are extended to Dr. Arif Akkeleş and Dr. Mustafa Rıza for assist on editing my dissertation. I would like to express my sincere thanks to my cousin Dr. Neyre Tekbıyık Ersoy for her guidance and persistent help on my dissertation. I am also thankful to my office mates for their help and moral support. Furthermore, I wish to thank my closest friend for unending spiritual support throught all the hard times. Last, my thanks to my family for their support during this studies.

TABLE OF CONTENTS

ABSTRACT iii

ÖZ iv

ACKNOWLEDGMENTS v

LIST OF TABLES ix

LIST OF FIGURES x

1 INTRODUCTION 1

2 CONCEPT OF CONVERGENCE, LOCAL-GLOBAL TRUNCATION ERROR, CONSISTENCY, STABILITY, WEAK STABILITY AND ZERO STABILITY 3

3 STIFF SYSTEMS AND PROBLEMS OF STABILITY FOR STIFF SYSTEM 19

3.1 Introduction 19

3.2 Definition of Stiffness 20

3.3 The Problem of Stability for Stiff System 22

4 EULER METHOD 26

4.1 Introduction 26

4.2 Definition of Euler's Method 26

4.3 Error Analysis of Euler's Method 31

4.4 Numerical Stability of Euler's Method 38

4.5 Rounding Error Accumulation of Euler's Method 44

4.6 Euler's Method for Systems 47

5 RUNGE-KUTTA METHOD 49

5.1 Introduction 49

5.2 Problem of Runge-Kutta Method 50

5.3 Explicit Runge Kutta Method 51

5.4 Implicit Runge-Kutta Method 59

5.5 Fourth Order Runge-Kutta Method 60

5.6 Stability of Runge-Kutta Method 64

5.6.1 Absolute Stability of Runge-Kutta Method 70

6 SYSTEM OF ORDINARY DIFFERENTIAL EQUATIONS 74

6.1 Introduction 74

6.2 Stability Theory for System 76

7 NUMERICAL EXPERIMENTS ON SIMPLE SYSTEMS 83

7.1 Introduction 83

7.2 Numerical Experiments on Stiff System 84

7.2.1 Numerical Experiments on Explicit Euler method 84

7.2.2 Numerical Experiments on Explicit Fourth Order Runge-Kutta method 86

8 COMPARISON OF EULER METHOD AND RUNGE-KUTTA METHOD WITH ADVANTAGES AND DISADVANTAGES 88

9 CONCLUSION 90

REFERENCES 91

APPENDICES 96

Appendix A. Explicit Euler method 97

Appendix B. Fourth Order Runge Kutta method 112

LIST OF TABLES

Table 1. Explicit Euler method with h=0.1 for n=10 98

Table 2. Explicit Euler method with h=0.05 for n=20 100

Table 3. Explicit Euler method with h=0.025 for n=40 102

Table 4. Explicit Euler method with h=0.0125 for n=80 104

Table 5. Explicit Euler method with h=0.00625 for n=160 106

Table 6. Explicit Euler method with h=0.003125 for n=320 108

Table 7. Explicit Euler method with h= 0.0015625 for n=640 110

Table 8. Fourth Order Runge Kutta method with h=0.1 for n=10 113

Table 9. Fourth Order Runge Kutta method with h=0.05 for n=20 . . . 115

Table 10. Fourth Order Runge Kutta method with h=0.025 for n=40 . . . 117

Table 11. Fourth Order Runge Kutta method with h=0.0125 for n=80 . . 119

Table 12. Fourth Order Runge Kutta method with h=0.00625 for n=160 . 121

LIST OF FIGURES

Figure 3.1 The region of $A-$Stability 23

Figure 3.2 The region of $A(\alpha)-$Stability 24

Figure 3.3 The region of Stiff Stability 25

Figure 4.1 The region of absolute stability of Forward Euler Method . . . 42

Figure 4.2 The region of absolute stability of Backward Euler Method . . 44

Figure 4.3 Rounding Error Curve for (4.5.7) 47

Figure 1. Graph of approximated solution and exact solution by using Explicit Euler method with $h = 0.1$ 99

Figure 2. Graph of approximated solution and exact solution by using Explicit Euler method with $h = 0.05$ 101

Figure 3. Graph of approximated solution and exact solution by using Explicit Euler method with $h = 0.025$ 103

Figure 4. Graph of approximated solution and exact solution by using Explicit Euler method with $h = 0.0125$ 105

Figure 5. Graph of approximated solution and exact solution by using Explicit Euler method with $h = 0.00625$ 107

Figure 6. Graph of approximated solution and exact solution by using Explicit Euler method with $h = 0.003125$ 109

Figure 7. Graph of approximated solution and exact solution by using Explicit Euler method with $h = 0.0015625$ 111

Figure 8. Graph of approximated solution and exact solution by using Explicit RK4 method with $h = 0.1$ 114

Figure 9. Graph of approximated solution and exact solution by using Explicit RK4 method with $h = 0.05$ 116

Figure 10. Graph of approximated solution and exact solution by using Explicit RK4 method with $h = 0.025$ 118

Figure 11. Graph of approximated solution and exact solution by using Explicit RK4 method with $h = 0.0125$ 120

Figure 12. Graph of approximated solution and exact solution by using Explicit RK4 method with $h = 0.00625$ 122

Chapter 1

INTRODUCTION

Differential equations are among the most important mathematical tools used in producing models in the physical sciences, biological sciences, and engineering. In this text, we consider numerical methods for solving ordinary differential equations, that is, those differential equations that have only one independent variable. We consider the differential equations in the form of

$$y'(x) = f(x, y(x))$$

where $y(x)$ is an unknown function that is being sought. The given function $f(x,y)$ of two variables defines the differential equation. This equation is called a first order differential equation because it contains a first order derivative of unknown function. The numerical methods for a first-order equation can be extended to a system of first order equations ([12]). The brief of thesis is organized as follows: In the second chapter, the concept of convergence, local–global truncation error, consistency, zero-stability, weak-stability are investigated for ordinary differential equations. At this stage we focused on simple first order differential equations, and multistep method for the given necessary conditions in Chapter 2. The purpose of this study is to find the best approximated solution which converge to analytical solution. Chapter

3 consist of the problem of stability for stiff system which focuses on the model problem. It is the instructive form to determine the stiff numerical method. Then, in the Chapter 4, Euler's method is reviewed. The derivation of Euler's method is stated and using some basic definitions, error and the region of stability is determined. In the next chapter, we discuss the derivation of the families of Runge-Kutta method. After completing this, we present some necessary definitions of the stability, and examined the absolute stability of Fourth Order Runge-Kutta method. In chapter 6, we give the general form of system for first order equations. The aim of this study is to compute the approximated solutions of system of differential equations. Then, it follows the stability theory for systems. In the last chapter, some examples are given, and each numerical method is solved by using Matlab software. After that, the results of numerical methods are analyzed. Finally, we concluded by giving the advantages and disadvantages of the specified numerical methods in Chapter 4 and Chapter 5. Additionally, in this dissertation, equations and formulas is benefited from two books.([15]), ([25])

Chapter 2

CONCEPT OF CONVERGENCE, LOCAL-GLOBAL TRUNCATION ERROR, CONSISTENCY, STABILITY, WEAK STABILITY AND ZERO STABILITY

A simple first order differential equation can be defined as

$$\frac{\partial y}{\partial x} = y'(x) = f(x, y(x)) \tag{2.1}$$

where $\frac{\partial y}{\partial x}$ is describing the first derivative of y with respect to time and $f(x, y)$ is any function of time and y. We look for a solution to (2.1) for $x > x_0$ as

$$y(x_0) = b \tag{2.2}$$

where $x_0 < x < b$, x_0 and b are two real numbers. The first order differential equation (2.1) with the condition described in (2.2) is known as the **Initial Value Problem**. Analytical solution is defined by $y(x)$, and approximate solution is represented by y_n. Introduce a general multistep method for solving initial value problem. Let the several nodes given by $x_n = x_0 + nh$, $n = 0, 1, 2, ...$

The general conformation of multistep method can be shown as

$$y_{n+1} = \sum_{j=0}^{p} a_j y_{n-j} + h \sum_{j=-1}^{p} b_j f(x_{n-j}, y_{n-j}) \tag{2.3}$$

where $h > 0$ and $a_0, a_1, ..., a_p, b_{-1}, b_0, ..., b_p$ are given constants term with $|a_p| + |b_p| \neq 0$ as $p \geq 0$. This is named $(p+1)$ step method. In following chapters, we will see that some methods can be produced by using multistep method as Euler Method, The Midpoint Rule, Trapezoidal Rule, and Adams methods. When any numerical method applied to a system, errors comprise in two forms.

1. Truncation (or discretization) Error: It is caused when approximations are used to estimate some quantity. Truncation error are composed of two parts.

   (a) Local Truncation Error: It is defined by T_{n+p}, and introduced the local error at x_{n+p}. It is shown as

   $$T_{n+p} = y(x_{n+p}) - y(x_n) - h\Phi(x_n, y(x_n); h) \tag{2.4}$$

   Local Truncation error arises when a numerical method is used to solve initial value problem. The error occur after the first step and form in each step.

   $$T_{n+1} = y(x_{n+1}) - y_{n+1}$$

   To obtain the local truncation error, take difference between left hand side and right hand side of method, and expand by using Taylor series. The remaining term is called Local Truncation error. Finally, divide the result to step size h. It is shown as

$$\tau_n(y) = \frac{1}{h}T_n \tag{2.5}$$

If $y(x)$ is assumed to be sufficiently differentiable, then the local truncation error for both explicit and implicit method can be written in the form

$$\begin{aligned} y(x_{n+p}) - y_{n+p} &= C_{p+1}h^{p+1}y^{p+1}(x_n) + O(h^{p+2}) \\ T_{n+p} &= C_{p+1}h^{p+1}y^{p+1}(x_n) + O(h^{p+2}) \end{aligned}$$

where C_{p+1} states the error constant, p indicates the order of the method, and $C_{p+1}hy^{p+1}(x)$ is the principal local truncation error ([27]). Thus, if the local truncation error is $O(h^{p+1})$, we can say that p represent the order of the method. This means that $\left|T_{n+p}\right| < C_{p+1}h^{p+1}$. If p is larger, then the method is more accurate.

(b) Global (or Accumulated) Truncation Error: It is denoted by e_n which is expressed

$$e_n = y(x_n) - y_n \tag{2.7}$$

where $y(x_n)$ is exact solution and y_n is approximate solution. Global Truncation error is caused by the accumulation of the local error in all of the iterations.

2. Round-off Error: It originate due to the operations of computer that takes lim-

ited edition number of digit. After calculating the approximations of methods, the result is dropped in specific location. It is denoted by R_{n+k} which is also committed at the n th application of the methods.

The general multistep method of initial value problem satisfies

$$y(x_{n+1}) = \sum_{j=0}^{p} a_j y(x_{n-j}) + h \sum_{j=-1}^{p} b_j f(x_{n-j}, y(x_{n-j})) + T_{n+p} \tag{2.8}$$

where T_{n+p} is the Local Truncation error, and

$$y_{n+1} = \sum_{j=0}^{p} a_j y_{n-j} + h \sum_{j=-1}^{p} b_j f(x_{n-j}, y_{n-j}) + R_{n+p} \tag{2.9}$$

where $\{y_n\}$ is the solution of multistep method and R_{n+p} is the Round-off error. Subtracting (2.9) from (2.8) and considering the global error $e_n = y(x_n) - y_n$, we obtain

$$e_{n+1} = \sum_{j=0}^{p} a_j e_{n-j} + h \sum_{j=-1}^{p} b_j \left[f(x_{n-j}, y(x_{n-j}) - f(x_{n-j}, y_{n-j}) \right] + \emptyset_{n+p}$$

where $\emptyset$ is obtained by subtracting round-off error from local truncation error as $\emptyset = T_{n+p} - R_{n+p}$. For any differentiable function $y(x)$, define the truncation error

$$T_n(y) = y(x_{n+1}) - \left[\sum_{j=0}^{p} a_j y(x_{n-j}) + h \sum_{j=-1}^{p} b_j y^{'}(x_{n-j}) \right], \quad n \geq p \tag{2.10}$$

To state the error of numerical methods, use the Taylor expansion series for $y(x_{n+1})$. Then the error will be determined by

$$T_n(y) = O(h^{p+1})$$

Consistency is used to indicate the accuracy of the method. Consider the expression of (2.5) to follow the consistency condition that is satisfied by

$$\tau_n(y) \longrightarrow 0 \quad \text{as} \quad h \longrightarrow 0$$

Use the notation (2.5), and rewrite the multistep method. We obtain

$$y(x_{n+1}) = \sum_{j=0}^{p} a_j y(x_{n-j}) + h \sum_{j=-1}^{p} b_j y^{'}(x_{n-j}) + h\tau_n(y)$$

Then introduce

$$\tau(h) \equiv \max_{x_0 \le x_n \le b} |\tau_n(y)| \tag{2.11}$$

where node points $[x_0, b]$ are $x_0 \le x_1 \le .... \le x_N \le b$. Multistep method is consistent if $y(x)$ is continuously differentiable on $[x_0, b]$ and satisfied the following status

$$\tau(h) \longrightarrow 0 \;\; \text{as} \;\; h \longrightarrow 0 \tag{2.12}$$

Order of the methods have a major role for the speed of the convergence. If the condition (2.11) goes to zero rapidly, then approximate solution will be closer to the exact solution. Thus, the speed of the convergence of approximated solution $\{y_n\}$ to the exact solution $y(x)$ is related with the speed of the convergence of condition (2.11). When step size h will be chosen small, the intervals will increase. This provide to be closer to 0. This relation is defined as

$$\tau(h) = O(h^p) \tag{2.13}$$

Theorem 2.0.1 *([1])Let $m \geq 1$ be a given integer. For (2.11) to hold for all continuously differentiable functions $y(x)$, that is, for the method (2.3) to be consistent, it is necessary and sufficient that*

$$\sum_{j=0}^{p} a_j = 1, \tag{2.14}$$

$$\sum_{j=-1}^{p} b_j - \sum_{j=0}^{p} j a_j = 1 \tag{2.15}$$

Further, for $\tau(h) = O(h^m)$ to be valid for all functions $y(x)$ that are $(m+1)$ times continuously differentiable, it is necessary and sufficient that (2.14)-(2.15) hold and that

$$\sum_{j=0}^{p} (-j)^i a_j + i \sum_{j=-1}^{p} (-j)^{i-1} b_j = 1, i = 2, ..., m$$

Proof. ([2])Note that

$$T_n(\alpha y + \beta w) = \alpha T_n(y) + \beta T_n(w) \tag{2.17}$$

for all constant α,β and for all differentiable functions $y(x)$, W. To examine the consequences of (2.11)-(2.12) and (2.13), expand $y(x)$ about x_n using the Taylor's theorem to obtain

$$y(x) = \sum_{i=0}^{m} \frac{1}{i!} (x - x_n)^i y^{(i)}(x_n) + R_{m+1}(x), \tag{2.18}$$

$$\begin{aligned} R_{m+1}(x) &= \frac{1}{m!}\int_{x_n}^{x}(x-s)^m y^{(m+1)}(s)\ ds \\ &= \frac{(x-x_s)^{m+1}}{(m+1)!}y^{(m+1)}(\xi_n) \end{aligned} \tag{2.19}$$

with ξ_n between x and x_n. We are assuming that $y(x)$ is $m+1$ times continuously differentiable on the interval bounded by x and x_n. Substituting into (2.10) and using (2.17), we obtain

$$T_n(y) = \sum_{i=0}^{m}\frac{1}{i!}y^{(i)}(x_n)T_n\left((x-x_n)^i\right) + T_n\left(R_{m+1}\right)$$

It is necessary to calculate $T_n\left((x-x_n)^i\right)$ for $i \geq 0$

- For $i = 0$,

$$T_n(1) = c_0 \equiv 1 - \sum_{j=0}^{p}a_j$$

- For $i \geq 1$,

$$\begin{aligned} T_n\left((x-x_n)^i\right) &= (x_{n+1}-x_n)^i \\ &\quad - \left[\sum_{j=0}^{p}a_j\left(x_{n-j}-x_n\right)^i + h\sum_{j=-1}^{p}b_j i\left(x_{n-j}-x_n\right)^{i-1}\right] \\ &= c_i h^i \end{aligned}$$

$$c_i = 1 - \sum_{j=0}^{p}(-j)^i a_j + i\sum_{j=-1}^{p}(-j)^{i-1}b_j,\ \ i \geq 1 \tag{2.20}$$

This gives us

$$T_n(y) = \sum_{j=0}^{m} \frac{c_i}{i!} h^i y^{(i)}(x_n) + T_n(R_{m+1}) \tag{2.21}$$

From (2.19), it is straightforward that $T_n(R_{m+1}) = O\left(h^{m+1}\right)$. If y is $m+2$ times continuously differentiable, we may write the remainder $R_{m+1}(x)$ as

$$R_{m+1}(x) = \frac{1}{(m+1)!}(x - x_n)^{(m+1)} y^{(m+1)}(x_n) + ...,$$

and then

$$T_n(R_{m+1}) = \frac{c_{m+1}}{(m+1)!} h^{m+1} y^{(m+1)}(x_n) + O\left(h^{m+2}\right) \tag{2.22}$$

with

$$c_{m+1} = 1 - \left[\sum_{j=0}^{p} (-j)^{m+1} a_j + (m+1) \sum_{j=-1}^{p} (-j)^m b_j\right]$$

To obtain the consistency condition (2.11)−(2.12), assuming that y is an arbitrary twice continuously differentiable function, we need $\tau(h) = O(h)$ and this requires $T_n(y) = O\left(h^2\right)$. Using (2.21) with $m = 1$, we must have $c_0 = c_1 = 0$, which gives the set of equations (2.14)−(2.15). In some texts, these equations are referred to as the *consistency conditions.* It can be further shown that (2.14)−(2.15) are the necessary and sufficient condition for consistency (2.11)−(2.12), even when y is only assumed to be differentiable. To obtain (2.13) for some $m \geq 1$, we must have $T_n(y) = O(h^{m+1})$. From (2.21)−(2.20), this will be true if and only if $c_2 = c_3 = \ldots = c_m = 0$. This proves the conditions (2.16). ■

For the largest value of p, which (2.13) holds, is known as the *order or order of convergence* of method, stated in (2.3). Convergence occurs for multistep methods then approximated solution $\{y_n\}$ tends to analytical solution $\{y(x_n)\}$ as the step size h goes to zero. This describes that convergence in the limit as $h \to 0, n \to \infty, nh = x - x_0$ remaining fixed where x is between x_0 and b. It can be represented as

$$\lim_{h \longrightarrow 0} y_n = y(x_n)$$

To define the convergence of the initial value problem in (2.24) and the general multistep method in condition (2.3),

$$y'(x) = f(x, y(x)), \quad x \geq x_0 \tag{2.24}$$

$$y(x_0) = y_0$$

The following theorem will be examined.

Theorem 2.0.2 *([3])Assume that the derivative function $f(t,y)$ is continuous and satisfies the Lipschitz Condition*

$$|f(x,y_1) - f(x,y_2)| \leq K \ |y_1 - y_2| \tag{2.25}$$

for all $-\infty < y_1$, $y_2 < \infty$, $x_0 \leq x \leq b$, and for some constant $K > 0$. Let the initial *errors satisfy*

$$\eta(h) \equiv \max_{0 \leq i \leq p} |y(x_i) - y_i| \longrightarrow 0 \quad as\, h \longrightarrow 0 \tag{2.26}$$

Assume that the solution $y(x)$ is continuously differentiable and method is consistent, that is, that it satisfies (2.12). Finally, assume that the coefficients a_j are all nonnegative

$$a_j \geq 0, \quad j = 0, 1, ..., p \tag{2.27}$$

then the multistep method (2.3) is convergent and

$$\max_{x_0 \leq x_n \leq b} |y(x_n) - y_n| \leq c_1 \eta(h) + c_2 \tau(h) \tag{2.28}$$

for suitable constants c_1, c_2. If the solution $y(x)$ is $(m+1)$ times continuously differentiable, the method (2.3) is of order m, and the initial errors satisfy $\eta(h)$ = $O(h^m)$, then the order of convergence of the method is m; that is, the error is of size $O(h^m)$.

Proof. ([4])Rewrite (2.10) and use $y^{'}(x) = f(x, y(x))$ to get

$$y(x_{n+1}) = \left[\sum_{j=0}^{p} a_j y(x_{n-j}) + h \sum_{j=-1}^{p} b_j f(x_{n-j}, y(x_{n-j})) + h\tau_n(y) \right]$$

Subtracting (2.3) from this equality and using the notation $e_n = y(x_n) - y_n$, we obtain

$$e_{n+1} = \sum_{j=0}^{p} a_j e_{n-j} + h \sum_{j=-1}^{p} b_j \left[f(x_{n-j}, y(x_{n-j}) - f(x_{n-j}, y_{n-j}) \right] + h\tau(h)$$

Apply the Lipschitz condition and the assumption (2.27) to obtain

$$|e_{n+1}| \le \sum_{j=0}^{p} a_j \left|e_{n-j}\right| + hK \sum_{j=-1}^{p} \left|b_j\right| \ \left|e_{n-j}\right| + h\tau(h)$$

where

$$\tau(h) \equiv \max |\tau_n(y)|$$

Introduce the maximum of error

$$f_n = \max_{i=0,\ldots n} |e_i|, n = 0, 1, \ldots, N(h).$$

Using this function, we have

$$|e_{n+1}| \le \sum_{j=0}^{p} a_j f_n + hK \sum_{j=-1}^{p} \left|b_j\right| f_{n+1} + h\tau(h)$$

and applying (2.14), we obtain

$$|e_{n+1}| \le f_n + hcf_{n+1} + h\tau(h), \qquad n \ge p$$

$$c = K \sum_{j=-1}^{p} \left|b_j\right|$$

The right hand side is trivially a bound for f_n and thus,

$$f_{n+1} \le f_n + hcf_{n+1} + h\tau(h)$$

For $hc \leq \frac{1}{2}$, which is true as $h \longrightarrow 0$, we obtain

$$\begin{aligned} f_{n+1} &\leq \frac{f_n}{1-hc} + \frac{h}{1-hc}\tau(h) \\ &\leq (1+2hc)f_n + 2h\tau(h) \end{aligned}$$

Noting that $f_p = \eta(h)$, then

$$f_n \leq e^{2c(b-x_0)}\eta(h) + \left[\frac{e^{2c(b-x_0)}-1}{hc}\right]\tau(h), \; x_0 \leq x_n \leq b \tag{2.29}$$

■

The convergence of methods depends on consistency condition and stability. The stability and convergence of numerical solution is related with the roots of the polynomial

$$p(r) = r^{p+1} - \sum_{j=0}^{p} a_j r^{p-j} \tag{2.30}$$

We can see that $p(1) = 0$ by considering the consistency condition $\sum_{j=0}^{p} a_j = 1$. Let $r_0 = 1$ and $r_1, r_2, ..., r_p$ be the roots of polynomial $p(r)$. The method (2.3) satisfies the **root condition** if

$$\left|r_j\right| \leq 1, \qquad j = 0, ..., p \tag{2.31}$$

$$\left|r_j\right| = 1 \Longrightarrow \rho^{'}(r_j) = \sigma(1) \neq 0 \tag{2.32}$$

These says that all roots of $\rho(r)$, which is known first characteristic polynomial, belong to the closed unit circle $\{z : |z| \leq 1\}$ in complex plane, and some simple roots of

$p(r)$ lie on the boundary of the circle. It associated a characteristic polynomial as

$$\pi(r, h\lambda) = \rho(r) - h\lambda\sigma(r) \tag{2.33}$$

The general solution of (2.26) can be defined if the roots $r_j(h\lambda)$ are all distinct

$$y_n = \sum_{j=0}^{p} \gamma_j \left[r_j(h\lambda)\right]^n \tag{2.34}$$

Since $h\lambda = 0$, we get from equation (2.33) that $\rho(r) = 0$. So, roots of equation become $r_j(0) = r_j$, $j = 0, 1, ..., p$. When $r_0 = 1$, then the root of (2.33) is $r_0(0) = 1$.

Definition 2.0.3 *([5])Assume the consistency conditions (2.14)-(2.15). Then the multistep method (2.3) is* ***stable*** *if and only if the root condition is satisfied.*

Stability determines how well the method will perform in finding the approximated solution in terms of satisfactoriness before actually performing the method. This means that numerical errors generated during the solution, and these errors should not be magnified.If Initial condition y_0 contains numerical errors, then this will effect the further approximation y_n.

Solving the initial value problem numerically shows that there is a solution $y(x)$ on a given finite interval $x_0 \leq x \leq b$. Accordingly, it is needed to analyze the stability for methods. We perturb the initial value problem (2.1) with $y_0 + \epsilon$. The result is

$$y_{\epsilon}^{'}(x) = f(x, y_{\epsilon}(x)) \tag{2.35}$$

$$y_{\epsilon}(x) = y_0 + \epsilon$$

This notations shows that $y(x)$ and $y_\epsilon(x)$ are exist on the given interval for small values of ε and besides,

$$||y_\epsilon - y||_\infty \equiv \max_{x_0 \leq\, x \,\leq\, b} |y_\epsilon(x) - y(x)| \leq c\epsilon \;\text{ for }\; c > 0 \tag{2.36}$$

Similar procedure is analyzed for stability of multistep method. Let state the solution of (2.4) with initial values $y_0, y_1, ..., y_p$ for some differential equation $y^{'}(x) = f(x, y(x))$ which perturb to the new values $z_0, z_1, ..., z_p$. Then,

$$\max_{0 \leq n \leq p} |y_n - z_n| \leq \epsilon \tag{2.37}$$

These initial values are valid for depending on h. In contrast, the numerical solution $\{y_n : \; 0 \leq n \leq N(h)\}$ is stable if it is independent of h, and there is a constant c with for all sufficiently small ϵ which is

$$\max_{0 \leq n \leq N(h)} |y_n - z_n| \leq c\epsilon, \;\; 0 \leq h \leq h_0$$

and $N(h)$ is accepted the largest subscript N which $x_N \leq b$. In addition, it is satisfying the Lipschtz condition. Thus, we can say that numerical solution of multistep method is **stable** if all approximate solution $\{y_n\}$ is stable. If the maximum error is less than the beginning error which is ϵ, then the I.V.P is called *well-conditioned.* Otherwise, it is *ill-conditioned.* Consequently, the small changes in the initial value y_0 will affect the solution of $y(x)$ of the initial value problem for small ϵ.

Theorem 2.0.4 *(As described in [32])A linear multistep method is* ***zero stable*** *for* *any ordinary differential equation of the form (2.1) where f satisfies Lipschitz condition if and only if its first characteristic polynomial has zeros inside the closed unit disc with any which lie on the unit circle being simple.*

We say that if first characteristic polynomial's all roots belong to the closed unit disc, **zero stability** holds for a linear multistep method. This means that modulus of first characteristics polynomial $p(r)$ is less than or equal to 1, and satisfies the root condition. There are two fundamental condition for convergence which is specified in the following theorem;

Theorem 2.0.5 *([33])To be consistent and zero-stable are the most basic condition for convergence of linear multistep method.*

Definition 2.0.6 *([6])The linear multi-step method (2.3) is said to be* ***relatively stable*** *for a given $h\lambda$ if, for a given $h\lambda$, the root r_j satisfy $\left|r_j(h\lambda)\right| \leq r_0(h\lambda)$, $j = 1, 2, 3, ..., p$ for sufficiently small values of h. If a linear method is stable but not relatively stable, then it is called* ***weakly stable.***

It is clear that a method is *relatively stable* if the characteristic roots $r_j(h\lambda)$ satisfies the given definition for nonzero values of $|h\lambda|$.

Definition 2.0.7 *([18])A numerical method is said to be* ***absolutely stable*** *in a region $\mathfrak{R}$ of complex plane if, for all $h\lambda \epsilon \mathfrak{R}$, all roots of the stability polynomial $\pi(r, h\lambda)$*

associated with the method, satisfy

$$\left|r_j\right| < 1, \quad j = 1,2,\ldots,p$$

Stability also includes some concept of absolute stability. Two parameters are considering for absolute stability: ”λ” which is the eigenvalues of Jacobian matrix and ”h” which is the step size. Absolute stability depends on value of the product of these two parameters, is called $h\lambda$. Analyzing of the stability at two parameters separately is insufficient. The region of the stability is considered in a complex plane. Because λ can be either negative real part or complex. When numerical method is applied to model equation $y^{'}(x) = \lambda y(x)$ for performing the region of stability, then the modulus of the *nth* step iteration should be less than or equal to 1. In system of differential equations, first , the Jacobian matrix form is created by writing the coefficients of the matrix variables. Then, the eigenvalues of Jacobian matrix is denoted by λ_t where $t = 1,2,...$ are achieved by taking determinant of the Jacobian matrix which is equal to zero. Finally, solving the system, we obtain the λ_t. If all eigenvalues of the matrix are real, then the matrix is symmetric, and can have complex eigenvalues. Otherwise, if eigenvalues of matrix are complex, then the parameter $h\lambda$ will be also complex. Consequently, the numerical method is absolutely stable if the parameter $h\lambda$ is in the region of absolute stability.

Chapter 3

STIFF SYSTEMS AND PROBLEMS OF STABILITY FOR STIFF SYSTEM

3.1. Introduction

Stiff systems arises applications, such as chemical kinetics, mechanical system, control theory, electronics, and mathematical biology, and numerical solution of partial differential equations. Stiffness is related with the concept of stability of numerical methods. There are several definitions of stiff equation.

Let A be the coefficient of matrix of mxm system. Let λ_μ and λ_υ be two eigenvalues of the coefficient matrix. The given problem is stiff if $\left|\mathrm{Re}(\lambda_\mu)\right| \gg |\mathrm{Re}(\lambda_\upsilon)|$. It is required to integrate numerically over a long range, using the hugely small step length to the interval.

Let A be the coefficient matrix of linear system. For system of differential equation, consider the eigenvalues λ_t of $f_y(x, y(x))$. If eigenvalues λ_t have negative real part

$$\mathrm{Re}(\lambda_t) < 0 \tag{3.1.1}$$

with the very large magnitude

$$\max_{1<j<n} |\mathrm{Re}(\lambda_t)| \gg \min_{1<j<n} |\mathrm{Re}(\lambda_t)| \tag{3.1.2}$$

then it make difficult to solve the system. We can say that the given problem is stiff. By this definition, a stiff problem has a stable point with greatly different magnitudes of eigenvalues.

To see the large magnitude between the eigenvalues, it is sufficient to look at the ratio of

$$\mathfrak{R} = \frac{\max|\mathrm{Re}(\lambda_t)|}{\min|\mathrm{Re}(\lambda_t)|} \tag{3.1.3}$$

which provides a measure of stiffness. It is clear that stiffness ratio is calculated by dividing highest eigenvalue by lowest eigenvalue.

3.2. Definition of Stiffness

Given the *m*x*m* linear systems

$$\mathbf{y}^{'} = A\mathbf{y} + \mathbf{\Phi}(x) \tag{3.2.1}$$

where A is the coefficient of matrix that has different eigenvalues λ_t and corresponding eigenvectors $c_t, t = 1, 2, .., m$ has general solution in the form of

$$\mathbf{y}(x) = \sum_{t=1}^{m} k_t e^{\lambda_t x}\mathbf{c}_t + \mathbf{\Psi}(x) \tag{3.2.2}$$

where k_t is a constant term. Suppose that $\mathrm{Re}(\lambda_t) < 0$, then the first term

$$\sum_{t=1}^{m} k_t e^{\lambda_t x}\mathbf{c}_t \longrightarrow 0 \;\text{ as }\; x \longrightarrow \infty \tag{3.2.3}$$

which includes exponential terms, is called "transient part" and the remaining term $\Psi(x)$ is called "steady state part". In this system, to reach the accurate approximation, the step size h is taken extremely small in our choice. It follows that the transient part approach to zero as $x \longrightarrow \infty$ and λ is real and negative. Steady state part will attain to exact solution. Thus, the aim of numerical solution is to find the approximate solution in steady state part and to ignore the transient part that includes the slowest decaying exponential e^{λ_t}. If the eigenvalues lie outside the region of stability, then step length should be chosen exceedingly small to satisfy $h\lambda$. Hence $h\lambda$ will lie in region of the absolute stability. In order to recognize the necessary condition, step size is to be excessively small for stability and it has negative real eigenvalues or negative real part of complex eigenvalues. Under this consideration system of differential equation is referred as stiff system.

In general, stiffness is affected from stiffness ratio which resulted in enormous numeral. It arises when the huge difference observed in the modulus of real part of eigenvalues.

Definition 3.2.1 *([19])The linear system $y^{'} = Ay + \Phi(x)$ is said to be* ***stiff*** *if*

1. $\mathrm{Re}(\lambda_t) < 0$, $t = 1, 2, \ldots, m$ *and*

2. $\max\limits_{t=1,2,\ldots,m} |\mathrm{Re}(\lambda_t)| \gg \min\limits_{t=1,2,\ldots,m} |\mathrm{Re}(\lambda_t)|$, *where* λ_t,$t = 1, 2, \ldots, m$ *are the eigenvalues of* A. *The ratio*

$$\left[\max_{t=1,2,\ldots,m} |\mathrm{Re}(\lambda_t)|\right] : \left[\min_{t=1,2,\ldots,m} |\mathrm{Re}(\lambda_t)|\right] \tag{3.2.4}$$

is called the stiffness ratio.

3.3. The Problem of Stability for Stiff System

We have seen the difficulties of the numerical solution of stiff system. It is essential to consider the absolute stability of methods. There are various definitions of stability for stiff systems that is proposed. These are applicable to any numerical method which involves discretization with associated step length *h*.

Definition 3.3.1 *([20])(Dahlquist*[37]*) A numerical method is said to be A-stable if its region of absolute stability contains the whole of the left-hand half-plane* $\mathrm{Re}(h\lambda) < 0$.

It is shown in figure

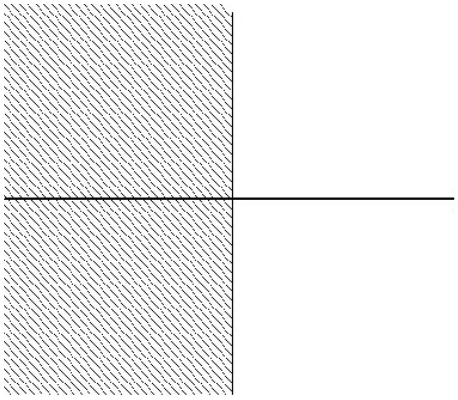

Figure 3.1. The region of $A-$Stability

When $A-$stable is considered on stiff system that $\mathrm{Re}(\lambda_j) < 0$, then step size can be taken without any restriction, and the magnitude of real part of eigenvalues, which represent $\max\limits_{t=1,2,\dots,m} |\mathrm{Re}(\lambda_t)|$, disregard.

Definition 3.3.2 *([21])([35])(Widlund* [181]*) A numerical method is said to be* $\mathbf{A}(\alpha)$*–**stable**,* $\alpha\epsilon$ $(0,\pi/2)$, *if its region of absolute stability contains the infinite wedge*

$$W_\alpha = \{h\lambda| \ -\alpha < \pi - \arg h\lambda < \alpha\}$$

it is said to be $\boldsymbol{A}(0)$ *–**stable** if it is* $A(\alpha)$ *–stable for some (sufficiently small)* $\alpha\epsilon(0,\pi/2)$*. A numerical method is* A_0*–stable if its region of absolute stability includes the negative real axis in the complex plane.*

If for a given λ, $\mathrm{Re}(\lambda) < 0$, then the point $h\lambda$ either will lie inside or outside the wedge W_α for all $h > 0$. Eventually, eigenvalues of a stiff system lie in a some wedge W_β; so, $A(\beta)-$ stable can be considered on the numerical solution of initial value problem without any restriction on step size. Especially, $A(0)-$stable can be used for real and nonnegative eigenvalues of matrix A. It is shown as

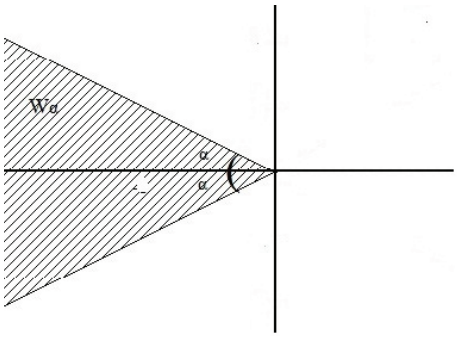

Figure 3.2. The region of $A\,(\alpha)$ –Stability

Definition 3.3.3 *([22])(Gear*[52,53]*) A numerical method is said to be **stiffly stable** iff*

1. *its region of absolute stability contains* $\Re 1$ *and* $\Re 2$,

2. *it is accurate for all* $h\epsilon\Re_2$ *when applied to the scalar test equation* $\mathbf{y}' = \lambda\mathbf{y}$, λ *complex constant with* $\mathrm{Re}(\lambda) < 0$, *where* $\Re_1 = \{h\lambda | \mathrm{Re}(h\lambda) < -a\}$, *and* $\Re_2 = \{h\lambda | -a \leq \mathrm{Re}(h\lambda) < b, -c \leq \mathrm{Im}(h\lambda) < c\}$ *and a, c are positive real numbers.*

It is shown as ([30])

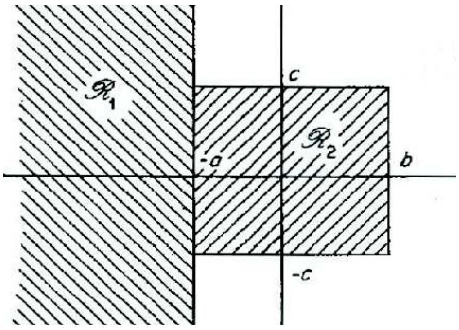

Figure 3.3. The region of Stiff Stability

Since λ is real and negative, the region R_1 has no restriction on h. Eigenvalues decays rapidly in the transient part. The value $h\lambda$ in $\Re_1$ will lie in stability region by eliminating the step size. Nevertheless, in the region $\Re_2$, the step size h should be chosen excessively small to satisfy $h\lambda$. Hereby, the value $h\lambda$ will lie in a stable region that is shown in the above figure.

Definition 3.3.4 *([23])A one step numerical method is said to be* ***L-Stable*** *if it is A-stable and, in addition, when applied to the scalar test equation* $y' = \lambda y$, λ *a complex constant with* $\mathrm{Re}(\lambda) < 0$, *it yields* $y_{n+1} = \Re(h\lambda)y_n$, *where* $\left|\Re(h\lambda)\right| \longrightarrow 0$ *as* $\mathrm{Re}(h\lambda) \longrightarrow -\infty$.

It is clear that $A-$stable is not enough to get accurate approximation, $L-$ stability is also required which can also be called as left stability. Definition of $L-$stability needed, the $\Re(h\lambda)$ tends to zero along with the real part of λ tends to negative infinity. Obviously, $A-$stability$\Longrightarrow$stiff-stability$\Longrightarrow A(\alpha)-$stability$\Longrightarrow A(0)$-stability$\Longrightarrow A_0-$stability ([34]).

Chapter 4

EULER METHOD

4.1. Introduction

Many differential equations in engineering are so intricate. It is inapplicable to have solution. Numerical methods provide ease for solving the differential equations. The simplest method to solve initial value problem is **Euler method** which have one step process for each equation before move on next step. This method is not an adequate method to get the certain approximation. Differential equation can be solved simply even though it is rather rough and least accurate. It is restricted to utilize because each successive step during the process accumulates large errors. It has slow of convergence which means a method of order 1. So the error is $O(h)$. In contrast, the remainder term and error analysis in Euler method provide convenience to state the difference between the approximate and exact solutions.

4.2. Definition of Euler's Method

Assume that initial value problem (2.1)−(2.2) is applied to numerical method on the specified interval: $[x_0, b]$. We can create nodes with equi-spaced subinterval for simplicity as

$$x_0 < x_1 < ... < x_N < b$$

If nodes is taken to be evenly spaced, numerical methods will generate an approximate solution y_n. It is written simply which is called **mesh point** as follows

$$x_n = x_0 + nh, \quad n = 0, 1, ..., N$$

where

$$h = \frac{(b - x_0)}{N}$$

and h is defined as **step-size** or (**step of integration**), a positive real number N. For each n, the numerical approximation y_n at a mesh points x_n can be smoothly obtained. The initial condition is known as

$$y(x_0) = y_0$$

Assume that we have already calculated y_n up to some n. This represent

$$y_{n+1} = y_n + hf(x_n, y_n) \tag{4.2.1}$$

It is known as **Euler method** with the initial condition. To attain Euler's method, consider a forward difference approximation to the derivative

$$y^{'}(x) \approx \frac{1}{h}[y(x+h)-y(x)]$$

Equalize this to the initial value problem (2.1−2.2) at x_n. We obtain

$$\frac{1}{h}[y(x_n+h)-y(x_n)] = f(x_n,y(x_n))$$

$$y(x_{n+1}) = y(x_n)+hf(x_n,y(x_n))$$

Euler's method can be represented by considering the approximated values

$$y_{n+1} = y_n+hf(x_n,y_n), \quad 0 \le n \le N-1$$

The other way to derive the Euler's method is to integrate the differential equation (2.1) between two consecutive mesh points x_n and x_{n+1}. We conclude that

$$\int_{x_n}^{x_{n+1}} y^{'}(x)dx = \int_{x_n}^{x_{n+1}} f(x,y(x))dx$$

$$y(x_{n+1})-y(x_n) = \int_{x_n}^{x_{n+1}} f(x,y(x))dx, \quad n = 0,1,...,N-1$$

$$y(x_{n+1}) = y(x_n)+\int_{x_n}^{x_{n+1}} f(x,y(x))dx \tag{4.2.2}$$

Here, we cannot integrate $f(x,y(x))$ without knowing $y(x)$. Hence we must approximate the integral. Apply the numerical integration rule

$$\int_{x_n}^{x_{n+1}} g(x)dx \approx hg(x)$$

which is knowns as the **rectangle rule** with $g(x) = f(x,y(x))$. This means the simplest approach is to approximate the area under function $f(x,y(x))$ by the rectangle with base $[x_n,x_{n+1}]$ and height $f(x,y(x))$. We can define the rectangle rule

$$\int_{x_n}^{x_{n+1}} g(x)dx \approx h\left[(1-\Theta)\,g(x_n)+\Theta g(x_{n+1})\right] \tag{4.2.3}$$

with $\Theta\epsilon\,[0,1]$. Then, substitute (4.2.3) into (4.2.2) by considering $g(x_n) = f(x_n,y(x_n)$ to obtain

$$\begin{aligned}
y(x_{n+1}) &= y(x_n)+h\left[(1-\Theta)\,g(x_n)+\Theta g(x_{n+1})\right] \\
y(x_{n+1}) &\approx y(x_n)+h\left[(1-\Theta)\,f(x_n,y\,(x_n))+\Theta f(x_{n+1},y\,(x_{n+1}))\right] \\
y(x_0) &= y_0
\end{aligned}$$

Then, supply the initial conditions to the one parameter method that is mentioned above. This gives us the following method where $\Theta\epsilon\,[0,1]$

$$y_{n+1} = y_n+h\left[(1-\Theta)\,f(x_n,y_n)+\Theta f\,(x_{n+1},y_{n+1})\right]$$

This definition motivates $\Theta-$**method** by considering approximate values. The $\Theta-$method referred Euler's method as $\Theta = 0$ which y_{n+1} must be found merely left hand side.

This definition that give y_{n+1} directly is called **explicit methods.** For $\Theta = 1$ we recover the **Implicit (backward) Euler Method.**

$$y_{n+1} = y_n + hf(x_{n+1}, y_{n+1}), \;\; n = 0, .., N-1 \tag{4.2.4}$$

In order to identify y_{n+1} (4.2.4) need the solution of an implicit equation. Euler's method is also referred as Explicit Euler Method in order to pick out the difference. This scheme gives a result for the value of $\Theta = \frac{1}{2}$ which is denominated **Trapezium Rule Method.** It is shown as

$$y_{n+1} = y_n + \frac{1}{2}h[f(x_n, y_n) + f(x_{n+1}, y_{n+1})], \;\; n = 0, .., N-1$$

The other way to obtain numerical method is using multistep method. In general form of multistep method, it is easier to achieve any numerical methods. Consider the given form of (2.3) to obtain Euler method. As $p = 0$, we get that

$$y_{n+1} = a_0 y_n + h\left[b_{-1} f(x_{n+1}, y_{n+1}) + b_0 f(x_n, y_n)\right]$$

If we give values instead of a_0, b_{-1}, b_0 kind of

$$a_0 = 1, b_{-1} = 0, b_0 = 1$$

gives Euler method formulae. In contrast, if the values are taken as

$$a_0 = 1, b_{-1} = 1, b_0 = 0$$

, we get Implicit Euler method

$$y_{n+1} = y_n + hf(x_{n+1}, y_{n+1})$$

which y_{n+1} will occur in both sides.

4.3. Error Analysis of Euler's Method

The purpose of examining error of Euler's method is to see how the approximated solution works. In Euler's method the slope of function affects the accuracy of methods. In order to minimize the error that occur, the step size should be chosen very small. In other words, number of point need to be taken enormous between the given interval. Thus, when step size is chosen excessively small, error is minimized and approximated solution will be more better.

For error analysis, we consider the differential equation $y(x)$ that satisfies

$$y^{'}(x) = f(x, y(x))$$

Assume that the solution of initial value problem is unique on $x_0 \leq x \leq b$, and this solution has bounded second derivative $y^{''}(x)$ over given interval. To state the error of Euler method, we begin by applying Taylor expansion series for $y(x_{n+1})$.

$$y(x_{n+1}) = y(x_n) + hy^{'}(x_n) + \frac{h^2}{2}y^{''}(\xi_n) + O(h^3)$$

for some $x_n \le \xi_n \le x_{n+1}$. Taylor approximation becomes

$$y(x_{n+1}) = y(x_n) + hf(x_n, y(x_n)) + \frac{h^2}{2} y''(\xi_n) + O(h^3) \tag{4.3.1}$$

If we compare the eguations (4.2.1) and (4.3.1), it yields

$$y_{n+1} = y_n + hf(x_n, y_n) + O(h^2)$$

It is shown that local truncation error of forward Euler method is $O\left(h^2\right)$. The other way to find truncation error; for any differentiable function $y(x)$, we can define the truncation error as follows

$$T_n(y) = y(x_{n+1}) - y(x_n) - hf(x_n, y(x_n)) \tag{4.3.2}$$

and the term

$$T_n = \frac{h^2}{2} y''(\xi_n) + O\left(h^3\right) \tag{4.3.3}$$

is called **truncation error** for Euler method. By considering the 2.5 in Chapter 2, we have the local truncation error of Euler method

$$\tau_n = \frac{1}{h} T_n$$
$$\tau_n = \frac{1}{h}[y(x_{n+1}) - y(x_n) - hf(x_n, y(x_n))],\ as\ y'(x_n) = f(x_n, y(x_n))$$

Then substitute the Taylor expansion about $y(x_{n+1})$ to get

$$\tau_n = \frac{1}{h}\left[y(x_n) + hy^{'}(x_n) + O(h^2) - y(x_n) - hf(x_n, y(x_n))\right]$$

$$\tau_n = \frac{1}{h}\left[O(h^2)\right]$$

$$\tau_n = O(h^1)$$

From the above explanation, It is clear that the truncation error is defined $T_n = O(h^2)$ for Euler's method and in general if $T_n = O(h^{p+1})$ which p indicates the order of method, then method is the pth order. That is; Euler method is the first order method. It can be explained that exact solution is equal to the approximate solution addedly to local truncation error: $y(x_n) = y_n + T_n$. As the order of method is 1, the method is too small. For this reason, it is not efficient method to obtain accuracy. To comprehend the error in Euler method, subtract

$$y_{n+1} = y_n + hf(x_n, y_n)$$

from (4.3.1) getting

$$y(x_{n+1}) - y_{n+1} = y(x_n) - y_n + hf(x_n, y(x_n)) - hf(x_n, y_n) + \frac{h^2}{2}y^{''}(\xi_n) \tag{4.3.4}$$

From the above explanation, the propagated error arises as following

$$y(x_n) - y_n + h\left[f(x_n, y(x_n)) - f(x_n, y_n)\right] \tag{4.3.5}$$

Let $e_n = y(x_n) - y_n$ to rewrite the (4.3.4)

$$e_{n+1} = e_n\,[1 + hK] + \frac{h^2}{2} y^{''}(\xi_n)$$

These results give a general error of Euler Methods. Now, we investigate the convergence of Euler's method for solving the general initial value problem on a given interval $[x_0, b]$.

$$y^{'}(x) = f(x, y(x)), \; x_0 \le x \le b \tag{4.3.6}$$

$$y(x_0) = y_0$$

For complete the error analysis the following Lemma and Theorem will be considered.

Lemma 4.3.1 *([9])For any real* t,

$$1 + t \le e^t, \tag{4.3.7}$$

and for any $t \ge 1$, *any* $m \ge 0$,

$$0 \le (1+t)^m \le e^{mt} \tag{4.3.8}$$

Theorem 4.3.2 *([7])Let* $f(x, y)$ *be a continuous function for* $x_0 \le x \le b$ *and* $-\infty < y < \infty$, *and further assume that* $f(x, y)$ *satisfies the Lipschitz condition. Assume that the solution* $y(x)$ *of the general solution of initial value problem has a continuous second* *derivative on* $[x_0, b]$. *Then the solution* $\{y(x_n)|\ x_0 \le x \le b\}$ *obtained by Euler's method satisfies*

$$\max_{x_0 \le x_n \le b} |y(x_n) - y_n| \le e^{(b-x_0)K} |e_0| + \left[\frac{e^{(b-x_0)K} - 1}{K} \right] \tau(h), \tag{4.3.9}$$

where

$$\tau(h) = \frac{1}{2} h \left\| y^{''} \right\|_{\infty} = \frac{1}{2} h \max_{x_0 \le x_n \le b} \left| y^{''}(x) \right| \tag{4.3.10}$$

and $e_0 = y(x_0) - y_0$. *If, in addition, we have*

$$|y(x_0) - y_0| \le c_1 h \quad as\ h \longrightarrow 0 \tag{4.3.11}$$

for some $c_1 \ge 0$ *(e.g., if* $y(x_0) = y_0$ *for all h, then* $c_1 = 0$*), then there is a constant* $B \ge 0$ *for which*

$$\max_{x_0 \le x_n \le b} |y(x_n) - y_n| \le Bh \tag{4.3.12}$$

Proof. ([8])Let $e_n = y(x_n) - y_n, n \ge 0$. Let $N \equiv N(h)$ be integer for which

$$x_N \le b,\ \ x_{N+1} > b.$$

Define

$$\tau_n = \frac{1}{2} h y^{''}(\xi_n),\ 0 \le n \le N(h) - 1,$$

based on the truncation error. Easily, we obtain

$$\max_{0 \le n \le N-1} |\tau_n| \le \tau(h)$$

using (4.3.10). Recalling (4.3.5), we have

$$e_{n+1} = e_n + h\left[f(x_n, y(x_n)) - f(x_n, y_n)\right] + h\tau_n \tag{4.3.13}$$

Taking bounds using Lipschtz condition, we obtain

$$|e_{n+1}| \le |e_n| + hK\,|y(x_n) - y_n| + h\,|\tau_n|$$

$$|e_{n+1}| \le (1 + hK)\,|e_n| + h\tau(h),\, 0 \le n \le N(h) - 1 \tag{4.3.14}$$

Apply this recursively to obtain

$$\Big|e_n| \le (1 + hK)^n \Big|e_0| + \Big[1 + (1 + hK) + \ldots + (1 + hK)^{n-1}\Big] h\tau(h)$$

Using the formula for the sum of a finite geometric series,

$$1 + r + r^2 + ... + r^{n-1} = \frac{r^n - 1}{r - 1},\ r \ne 1 \tag{4.3.15}$$

we obtain

$$\Big|e_n| \le (1 + hK)^n \Big|e_0| + \left[\frac{(1 + hK)^n - 1}{K}\right]\tau(h) \tag{4.3.16}$$

Using the Lemma 1, we obtain

$$(1+hK)^n \leq e^{nhK} = e^{(x_n-x_0)K} \leq e^{(b-x_0)K}$$

and this with (4.3.16) implies the main result (4.3.9).

The remaining result (4.3.12) is a trivial corollary of (4.3.9) with the constant *B* given by

$$B = c_1 e^{(b-x_0)K} + \frac{1}{2}\left[\frac{e^{(b-x_0)K}-1}{K}\right]\left\|y''\right\|_\infty$$

■

We can say that the result (4.3.12) is consistent. When step size is halved, *Bh* is also halved. Euler's method is convergent with order 1, because that is the power of *h* that occurs in the error bound. In general, if we have

$$|y(x_n) - y_n| \leq ch^p,\ x_0 \leq x_n \leq b$$

for some constant $p \geq 0$, then numerical methods is convergent with order *p* ([14]). Let us consider the consistency on the **examples**:

1. For Euler methods: The numerical solution is

$$y_{n+1} = y_n + hf(x_{n+1}, y_{n+1})$$

$$
\begin{aligned}
T_n(y) &= y(x_{n+1}) - y(x_n) - hy'(x_n) + O(h^2) \\
\tau_n(y) &= \frac{1}{h}\left[y(x_{n+1}) - y(x_n) - hy'(x_n)\right]
\end{aligned}
$$

This says that

$$\frac{y(x_{n+1}) - y(x_n)}{h} \approx y^{'}(x_n)$$

and the order of the method is 1, $O(h^1)$.

2. ([31])For midpoint method:

$$y_{n+1} = y_{n-1} + 2hf(x_n, y_n)$$

Here

$$
\begin{aligned}
T_n(y) &= y(x_{n+1}) - y(x_{n-1}) - 2hy^{'}(x_n) \\
\tau_n(y) &= 2\left[\frac{y(x_{n+1}) - y(x_n)}{2h} - y^{'}(x_n)\right]
\end{aligned}
$$

Thusly,

$$\frac{y(x_{n+1}) - y(x_n)}{2h} \approx y^{'}(x_n)$$

4.4. Numerical Stability of Euler's Method

Recollect the definition of stability for initial value problem. Small changes in initial value y_0 will perturbed the solution $y(x)$ by ϵ. Similar analysis is valid for Euler's method. Introduce a new numerical solution $\{z_n\}$ by

$$z_{n+1} = z_n + hf(x_n, z_n), \quad n = 0, 1, ..., N-1$$

with initial values $z_0 = y_0 + \epsilon$. It is perturbed to the initial value problem. In order to inspect the stability, compare two numerical solution $\{y_n\}$ in (2.1) and $\{z_n\}$ by subtracting. We get

$$z_{n+1} - y_{n+1} = z_n - y_n + h[f(x_n, z_n) - f(x_n, y_n)]$$

with initial value $z_0 - y_0 = \epsilon$. Introduce $e_n = z_n - y_n$ to obtain

$$e_{n+1} = e_n + h[f(x_n, z_n) - f(x_n, y_n)]$$

To be more straightforward, use the Lipschtz condition and apply theorem (4.3.2) to get

$$\max_{0 \le n \le N} |z_n - y_n| \le e^{(b-x_0)K} |\epsilon|$$

There is a constant $\hat{c} \ge 0$ such that

$$\max_{0 \le n \le N} |z_n - y_n| \le \hat{c} |\epsilon|$$

This analog shows that error must be less than the received perturbed error in startup for getting stable method. Thus, Euler method is a stable under the solution of the initial value problem Now, we turn to examine the performance of Euler's method on model problem. We have

$$y'(x) = \lambda y(x), \ x \geq 0 \tag{4.4.1}$$

$$y(0) = 1$$

When it is applied to the model problem, the numerical method is displayed

$$y_{n+1} = y_n + h\lambda y_n = (1 + h\lambda) y_n, \ n \geq 0 \tag{4.4.2}$$

$$y_0 = 1$$

We investigate the case λ to find the region of stability of Euler methods. Recursively, it is obtained

$$y_n = (1 + h\lambda)^n y_0 \tag{4.4.3}$$

It can be written for a fixed node point $x_n = nh \equiv \bar{x}$, as $n \longrightarrow \infty$

$$y_n = \left(1 + \frac{\lambda \bar{x}}{n}\right)^n \longrightarrow e^{\lambda \bar{x}}$$

Since exact solution is decaying exponentially, numerical method also has same behavior. To show the stability of Explicit Euler method, we need that

$$|1 + h\lambda| < 1$$

for λ is real and negative. Thus, in the complex plane radius is 1 with centre 1. The point $h\lambda$ is in region of absolutely stable under specified interval with Re(λ). Then the condition indicates the interval that Euler method is stable. It becomes

$$-2 < h\lambda < 0$$

namely

$$0 < h < \frac{2}{\lambda}$$

This also satisfied the convergence of Euler method since in the notation (4.4.3), $y_n \longrightarrow 0$ as $n \longrightarrow \infty$ for any choice of step size holds. Thus, in Explicit Euler method, step size should be chosen excessively small to provide the stability. The stability region of Explicit Euler method is straightforward in the figure 4.1

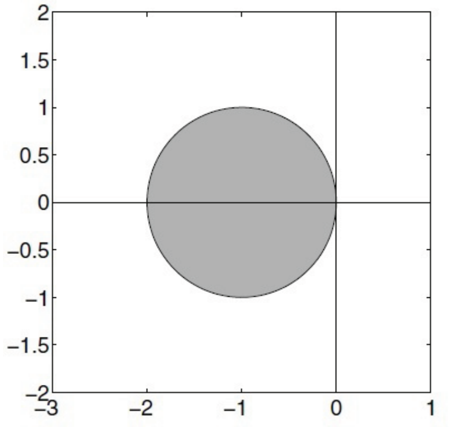

Figure 4.1. The region of absolute stability of Forward Euler Method

On the other hand, we show the implicit Euler method to understand how methods behaves in the model problem. Implicit Euler method states that

$$y_{n+1} = y_n + hf(x_{n+1}, y_{n+1})$$

After that substitute model problem to numerical method to obtain

$$y_{n+1} = y_n + h\lambda y_{n+1}$$
$$y_{n+1} = \frac{1}{1-h\lambda} y_n$$

and by using induction

$$y_n = \left(\frac{1}{1-h\lambda} \right)^n y_0$$

Therefore, it is clear that for $\mathrm{Re}(\lambda) < 0$ implicit Euler method is stable if and only if it satisfies

$$\left(\frac{1}{1-h\lambda} \right) < 1$$

So, $|1-h\lambda| > 1$ for any step size $h > 0$. Thus, we should choose step size h large to ensure the stability. It is straightforward in figure (4.2)

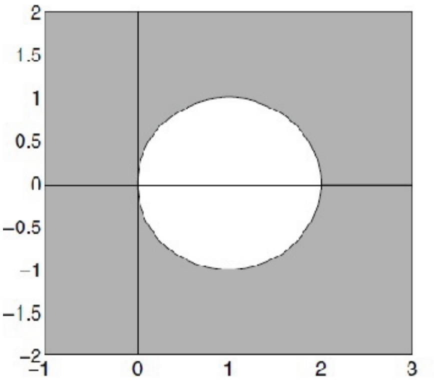

Figure 4.2. The region of absolute stability of Backward Euler Method

Consequently, when numerical method is absolutely stable, there is no restriction on h. If the values of λ that is real and negative has the large magnitude, then step size should be chosen excessively small to satisfies $h\lambda$. Hereby, it belong to the region of absolute stability. This also affect the approximation solution. It provides the truncation error to be small. Even though Euler's method is absolute stable, we cannot make any comment for Euler Method about weak-stable. Consider root condition (2.30) and take $p = 0$

$$\rho(r) = r - a_0$$

and root is

$$r_0 = a_0$$

From the condition (2.14) of theorem 2.0.1, manifestly $r_0 = 1$. This shows that Euler method is absolute stable, but we cannot apply the definition 2.0.5 which is related with relatively stable and weak stable. Because it has one root.

4.5. Rounding Error Accumulation of Euler's Method

"Integer mode" and "floating point mode" provide to represent the numbers. Integer mode is used for integer numbers and floating point mode is used for real numbers. We concern with floating point mode. In numerical methods, numbers have greatly varying size. In order to facilitate the operations, the limitations on the numbers of digit are required. Rounding error is caused due to this modes. It affect the accuracy in the numerical solution. Now, we analyze the error for Euler's method. After numerical method applied the to any differential equation, rounding error is being performed.

Denote δ_n which is called rounding error. It is the precision of the arithmetic and affected the approximate solution $\{y_n\}$. We have

$$y_{n+1} = y_n + hf(x_n, y_n) + \delta_n, \quad n = 0, 1, \ldots, N(h) - 1 \tag{4.5.1}$$

Assume that

$$|\delta_n| \leq cu. \max_{x_0 \leq x \leq b} |y(x)| = cu. ||y||_\infty \tag{4.5.2}$$

where $u = 2.2x10^{-16}$ is the *machine epsilon* of the computer and the magnitude of c is 1 or larger 1. Let $\delta(h)$be a bound on the rounding errors

$$\delta(h) = \max_{0 \leq n \leq N(h)-1} |\delta_n| \tag{4.5.3}$$

To see the effect of rounding error subtract the approximate solution (4.5.1) from the exact solution to have

$$y(x_{n+1}) - y_{n+1} = y(x_n) - y_n + h\left[f(x_n, y(x_n)) - f(x_n, y_n)\right] + \frac{1}{2}h^2 y''(\xi_n) - \delta_n \qquad (4.5.4)$$

$$y(x_0) - y_0 = 0$$

This is equivalent to the form

$$y(x_{n+1}) - y_{n+1} = y(x_n) - y_n + h\left[f(x_n, y(x_n)) - f(x_n, y_n)\right] + h\tau(h) - \delta_n$$

where $T_{n+1} = \frac{1}{2}h^2 y^{"}(\xi_n)$ is the local truncation error. It is same as the notation

$$\tau(h) = \frac{1}{h}T_{n+1}$$

The error term of (4.5.4) can be written as follows

$$h\tau(h) - \delta(h) = h\left[\tau(h) - \frac{\delta_n}{h}\right] \qquad (4.5.5)$$

Setting $e_n = y(x_n) - y_n$, take bounds of notation (4.5.4) using the Lipschtz condition

$$|e_{n+1}| \le |e_n| + hK|e_n| + h\tau(h) - \delta(h) \qquad (4.5.6)$$

$$|e_{n+1}| \le (1 + hK)|e_n| + h\tau(h) - \delta(h) \qquad (4.5.1)$$

By an inductive argument and applying the proof of the theorem (4.3.2) we obtain,

$$|e_n| \le e^{(b-x_0)K}|e_0| + \frac{e^{(b-x_0)K}}{hK}h\left[\tau(h) - \frac{\delta_n}{h}\right]$$

where $e_0 = 0$ is getting from (4.5.4). So, we reach the last form as

$$|e_n| \leq \frac{e^{(b-x_0)K}}{K}\left[\frac{1}{2}h\left\|y''\right\|_\infty - \frac{cu}{h}\|y\|_\infty\right]$$

$$|y(x_n) - y_n| \leq c_1\left\{\frac{1}{2}h\left\|y''\right\|_\infty - \frac{cu}{h}\|y\|_\infty\right\} = E(h) \qquad (4.5.7)$$

It is clear from the above operations that error will decrease as h decrease at the beginning; but at the critical point the optimum value of h^*, the error will increase. At the second term in brackets on the right hand side of (4.5.7) affect the error despite the fact that machine epsilon u is small. The figure (4.3) also demonstrate the above explanation.

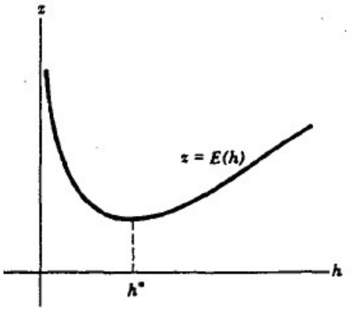

Figure 4.3. Rounding Error Curve for (4.5.7)

4.6. Euler’s Method for Systems

The numerical solution for system is same as for a single equation. Numerical method is applied to each equation in the system. It is shown in detailed in Chapter 6 that is written in a matrix form. To be more clarity, we consider Euler’s method for a system of two differential equations. Let

$$y_1'(x) = f_1(x, y_1, y_2)$$

$$y_2'(x) = f_2(x, y_1, y_2)$$

with initial conditions

$$y_1(x_0) = y_{1,0}$$

$$y_2(x_0) = y_{2,0}$$

In order to obtain system of Euler’s method, consider the Taylor expansion for each equation.

$$y_1(x_{n+1}) = y_1(x_n) + hf_1(x_n, y_{1,n}, y_{2,n}) + \frac{h^2}{2} y_1''(\xi_n)$$
$$y_2(x_{n+1}) = y_2(x_n) + hf_2(x_n, y_{1,n}), y_{2,n}) + \frac{h^2}{2} y_2''(\zeta_n)$$

for some ξ_n, ζ_n in $[x_n, x_{n+1}]$. We get Euler's method for a system of two equation by ignoring the error terms and considering the initial conditions

$$y_{1,n+1} = y_{1,n} + hf_1(x_n, y_{1,n}, y_{2,n})$$

$$y_{2,n+1} = y_{2,n} + hf_2(x_n, y_{1,n}, y_{2,n})$$

This indicates the general form of system in matrices format as

$$\mathbf{y}_{n+1} = \mathbf{y}_n + h\mathbf{f}(x_n, \mathbf{y}_n)$$

Chapter 5

RUNGE-KUTTA METHOD

5.1. Introduction

The Runge-Kutta method is popular method for solving initial value problem. It is most accurate and stable method. It arise when Leonhard Euler have made improvements on Euler method to produce Improved Euler method. Then, Runge is realized this method which is similar method with the second order Runge Kutta method. A few years later in 1989 Runge acquired Fourth Order of Runge Kutta method and afterwards, it is developed by Heun(1900) and Kutta(1901). Fourth Order Runge-Kutta method intend to increase accuracy to get better approximated solution. This means that the aim of this method is to achieve higher accuracy and to find explicit method of higher order. In this section, we discuss the formulation of method, concept of convergence, stability, consistency for RK4 method. In spite of the fact that Runge Kutta methods are all explicit, implicit Runge Kutta method is also observed. It has the same idea of Euler method. Euler method is the first order accurate; in addition it require only a single evaluation of $f(x_n,y_n)$ to obtain y_{n+1} from y_n. In contrast, Runge Kutta method has higher accuracy. It re-evaluates the function f at two consecutive points (x_n,y_n) and (x_{n+1},y_{n+1}). It requires four evaluations per step. Due to this, Runge-Kutta method is quite accurate, and it has faster rates of convergence.

5.2. Problem of Runge-Kutta Method

Analyze the initial value problem to find the solution of ordinary differential equations

$$y'(x) = f(x, y(x)) \tag{5.2.1}$$

$$y(x_0) = y_0$$

The general form Runge-Kutta method is

$$y_{n+1} = y_n + hF(x_n, y_n; h),\ n = 0, \ldots, N-1 \tag{5.3.2}$$

where $F(.,.;.)$ is called an *increment function* on the interval $[x_n, x_{n+1}]$. It can be defined in general form as

$$F(x, y; h) = b_1k_1 + b_2k_2 + \ldots + b_nk_n \tag{5.2.3}$$

where b_n's are constant and k_n's are

$$\begin{aligned} k_1 &= f(x_n, y_n) \\ k_2 &= f(x_n + c_2h, y_n + a_{21}k_1h) \\ k_3 &= f(x_n + c_3h, y_n + a_{31}k_1h + a_{32}k_2h) \\ &\vdots \\ k_s &= f(x_n + c_sh, y_n + a_{s,1}k_1h + a_{s,2}k_2h + \ldots + a_{s,s-1}k_{s-1}h) \end{aligned} \tag{5.2.4}$$

where c's and a's are constants. Here, each of function k's are represent slope of the solution which are approximated to $y(x)$. These coefficients a_{ij}, c_j, b_j must be satisfied the system of Runge-Kutta method. These are usually arranged in *Butcher tableau* or (*partitioned tableau*)

$$\begin{array}{c|c} c & A \\ \hline & b^T \end{array}$$

The coefficient b^T is a vector of quadrature weights, and a_{ij} $(i, j = 1, \ldots, s)$ indicates the matrix A.

5.3. Explicit Runge Kutta Method

The family of $s-$stage Explicit Runge Kutta method can be formed as shown in the following:

$$y_{n+1} = y_n + h \begin{bmatrix} b_1 f(x_n, y_n) + b_2 f(x_n + c_2 h, f(x_n, y_n + h[a_{21}k_1])) + \ldots \\ +b_{s-1} f(x_n + c_{s-1}h, y_n + h[a_{s-1,1}k_1 + a_{s-1,2}k_2 + \ldots + a_{s-1,s-2}]) \\ +b_s f(x_n + c_s h, y_n + h[a_{s,1}k_1 + a_{s,2}k_2 + \ldots + a_{s,s-1}]) \end{bmatrix} \tag{5.3.1}$$

$$y_{n+1} = y_n + h \sum_{i=1}^{s} b_i k_i$$

where

$$k_1 = f(x_n, y_n),$$

$$k_i = f(x_n + c_i h, y_n + h \sum_{j=1}^{s-1} a_{i,j} k_j),\ i = 1, 2, \ldots, s \tag{5.3.2}$$

where $h = x_{n+1} - x_n$. The coefficients $\left\{c_i, a_{i,j}, b_j\right\}$ determine the numerical method. This form was developed for higher accuracy methods. However, formulas can be defined different notations as

$$z_i = y_n + h \sum_{j=1}^{i-1} a_{i,j} f(x_n + c_j h, z_j), \quad i = 1, 2, ..., s \tag{5.3.4}$$

$$y_{n+1} = y_n + h \sum_{j=1}^{s} b_j f\left(x_n + c_j h, z_j\right) \tag{4.3.5}$$

The coefficients $\left\{c_i, a_{i,j}, b_j\right\}$ for Explicit Runge Kutta methods can be simplified by using the following Butcher Tableau

$$\begin{array}{c|ccccc}
0 & & & & & \\
c_2 & a_{21} & & & & \\
c_3 & a_{31} & a_{32} & & & \\
\vdots & \vdots & \vdots & \ddots & & \\
c_s & a_{s1} & a_{s2} & \ldots & \ldots & a_{s,s-1} \\
\hline
 & b_1 & b_2 & \ldots & b_{s-1} & b_s
\end{array}$$

If Runge-Kutta method is consistent, then the coefficients $\left\{a_{i,j}\right\}$ and $\left\{c_j\right\}$ must satisfy the condition

$$\sum_{j=1}^{i-1} a_{i,j} = c_i, \; i = 2, \ldots, s \tag{5.3.5}$$

The simplest Runge–Kutta method is the (Explicit) Euler method. The formula of Euler method which is the first order, is mentioned previous chapter as $y_{n+1} = y_n + f(x_n, y_n)$. The corresponding tableau is

$$\begin{array}{c|c} 0 & 0 \\ \hline & 1 \end{array}$$

The other example is given the midpoint method which is a second-order method

$$y_{n+1} = y_n + hf(x_n + \frac{1}{2}h, y_n + \frac{1}{2}hf(x_n, y_n)$$

and its Butcher tableau is shown as

$$\begin{array}{c|cc} 0 & 0 & \\ \frac{1}{2} & \frac{1}{2} & 0 \\ \hline & 0 & 1 \end{array}$$

In order to evaluate $y(x_{n+1})$, each of higher order derivatives is required to expand at the point x_i in the Taylor series. It is convenient to analyze any Runge-Kutta method by using some formulae of derivatives which is

$$
\begin{aligned}
y'(x) &= f(y(x)) \\
y''(x) &= f'(y(x))\, y'(x) \implies f'(y(x))\, f(y(x)) \\
y'''(x) &= f''(y(x))\big(f(y(x)), y'(x)\big) + f'(y(x)) f'(y(x)) y'(x) \\
&\implies f''(y(x))(f(y(x)),\ f(y(x))) + f'(y(x)) f'(y(x)) f(y(x))
\end{aligned}
$$

It can be simplified to avoid complication by taking $f = f(y(x))$, $f_y = f'(y(x))$, $f_{yy} = f''(y(x))$. We obtain

$$
\begin{aligned}
y'(x) &= f \\
y''(x) &= f_y f \\
y'''(x) &= f_{yy} f^2 + f_y^2 f \\
y''''(x) &= f_{yyyy} f^3 + 4 f_{yy} f_y f^2 + f_y^3 f
\end{aligned}
\tag{5.3.6}
$$

In order to generate the second order Runge Kutta method, we observe Taylor expansions for $y(x_{n+1})$ which has the form

$$
y(x_{n+1}) = y(x_n) + hy'(x_n) + \frac{h^2}{2!}y''(x_n) + \frac{h^3}{3!}y'''(x_n) + \ldots + \frac{h^p}{p!}y^{(p)}(x_n+) \tag{5.3.7}
$$

So as to attain the second order Runge Kutta method, we should take the derivation of $f(x,y)$ for the functions k_1, k_2. Consider the second order Taylor series and use notations of (5.3.5)–(5.3.6) to construct the Runge Kutta method with order 2. Thus, instead of $y'(x_n)$, we can write $f(x,y)$ and we need to evaluate $y''(x_n)$ with respect to

y. After substituting, we takes form

$$y(x_{n+1}) = y(x_n) + hf + \frac{1}{2}h^2 f_y f + O(h^3) \tag{5.3.8}$$

The approximate solution of second order Runge Kutta method can be obtained

$$y_{n+1} = y_n + h\,[b_1 k_1 + b_2 k_2] \tag{5.3.9}$$

where

$$\begin{aligned} k_1 &= f(x,y) = f \qquad (5.3.10) \\ k_2 &= f(x_n + c_2 h, y_n + h a_{21} k_1) = f + h a_{21} f f_y + h^2 a_{12} f f_x \end{aligned}$$

Then, replace the slope of solution to form of the approximation solution y_{n+1} and neglect function of f that depend on x to obtain

$$y_{n+1} = y_n + h\Big[b_1 f + b_2 \Big(f + h a_{21} f f_y\Big)\Big] \tag{5.3.11}$$

Now, we match (5.3.8) with Taylor series (5.3.7). This satisfy the following equations

$$\begin{aligned} b_1 + b_2 &= 1 \qquad (5.3.12) \\ b_2 a_{21} &= \frac{1}{2} \end{aligned}$$

So, we have two conditions with three unknowns values. If we choose $b_1 = 0$, $b_2 = 1$ and $a_{21} = \frac{1}{2}$, Runge Kutta method with order 2

$$y_{n+1} = y_n + hf\left(x_n + \frac{1}{2}h, y_n + \frac{1}{2}hf(x_{n,}y_n)\right)$$

This is called *Modified Euler method.* Another particular solution of (4.3.12) is Heun's method since $b_1 = \frac{1}{2}, b_2 = \frac{1}{2}$ and $a_{21} = 1$. The resulting method is

$$y_{n+1} = y_n + \frac{h}{2}f(x_n, y_n + f(x_n + h, y_n + hf(x_n, y_n)))$$

It is possible to acquire third order Runge Kutta method by using the same construction. Expansion of the term will include the order h^3.Thus, for the Runge Kutta method of order third, we consider third order Taylor expansion which has the form

$$y(x_{n+1}) = y(x_n) + hy^{'}(x_n) + \frac{h^2}{2!}y^{''}(x_n) + \frac{h^3}{3!}y^{'''}(x_n) + O(h^4)$$

It is required to find $y^{''}(x_n)$, $y^{'''}(x_n)$ and also we need to expand k_1, k_2, k_3 by using derive expansion. The numerical solution is written as

$$y_{n+1} = y_n + h[b_1k_1 + b_2k_2 + b_3k_3] \tag{5.3.13}$$

where

$$\begin{aligned} k_1 &= f(x,y) \\ k_2 &= f(x_n + c_2 h, y_n + h a_{21} k_1) \qquad (5.3.14) \\ k_3 &= f(x_n + c_3 h, y_n + h\,[a_{31} k_1 + a_{32} k_2]) \end{aligned}$$

Expanding k_2 as a Taylor series we obtain

$$k_2 = f + hc_2 f_x + ha_{21} k_1 f_y + h^2 a_{12} f f_x + \frac{1}{2} h^2 a_{21}^2 \left[f_{xx} + 2k_1 f_{xy} + f_{yy} k_1^2 \right] + O(h^3) \quad (5.3.15)$$

and substitute k_1 to equation (5.3.15) and ignore the function which is depending of x to obtain

$$k_2 = f + ha_{21} f f_y + \frac{1}{2} h^2 a_{21}^2 f_{yy} f^2 + O(h^3)$$

Similarly, in order to obtain k_3, same process is applied in turn

$$k_3 = f + h\,(a_{31} k_1 + a_{32} k_2)\, f_y + \frac{1}{2} h^2 \left(a_{31}^2 k_1^2 + a_{32}^2 k_2^2 \right) f_{yy}$$

Then by substituting k_i's ($i = 1,2,3$) to the numerical solution, we obtain

$$\begin{aligned} y_{n+1} &= y_n + h\,(b_1 + b_2 + b_3)\, f + h^2\,(b_2 a_{21} + b_3 a_{31} + b_3 a_{32})\, f f_y \\ &+ \frac{h^3}{2} \left(b_2 a_{21}^2 + b_3 a_{31}^2 + b_3 a_{32}^2 \right) f^2 f_{yy} + h^3 b_3 a_{32} a_{21} f f_y f_y \end{aligned}$$

Ultimately, we match the Taylor expansion of exact solution with the expansion of solution. We have

$$y(x_{n+1})-y_{n+1} = h(b_1+b_2+b_3-1)f+h^2\left(b_2a_{21}+b_3a_{31}+b_3a_{32}-\frac{1}{2}\right)ff_y \quad (5.3.16)$$
$$+\frac{h^3}{2}\left(b_2a_{21}^2+b_3a_{31}^2+b_3a_{32}^2-\frac{1}{3}\right)f^2f_{yy}+h^3(b_3a_{32}a_{21}-\frac{1}{6})ff_yf_y$$

It is clear that set of functions

$$\begin{aligned} b_1+b_2+b_3 &= 1 \\ b_2a_{21}+b_3a_{31}+b_3a_{32} &= \frac{1}{2} \\ b_2a_{21}^2+b_3a_{31}^2+b_3a_{32}^2 &= \frac{1}{3} \\ b_3a_{32}a_{21} &= \frac{1}{6} \end{aligned} \quad (5.3.17)$$

There are four equations with six unknowns. By taking convenient value for unknowns, we get through two particular solution of third order Runge Kutta method:

1. ([28])Since $b_1=\frac{1}{4}, b_2=0, b_3=\frac{3}{4}, a_{21}=\frac{1}{3}, a_{31}=\frac{2}{3}, a_{32}=\frac{2}{3}$, the result gives *Heun's third order formula*

$$\begin{aligned} y_{n+1} &= y_n+\frac{h}{4}(k_1+3k_3) \\ k_1 &= f(x,y) \\ k_2 &= f(x+\frac{1}{3}h, y+\frac{1}{3}hk_1) \\ k_3 &= f(x+\frac{2}{3}h, y+\frac{2}{3}hk_2) \end{aligned}$$

2. ([29])Since $b_1 = \frac{1}{6}, b_2 = \frac{2}{3}, b_3 = \frac{1}{6}, a_{21} = \frac{1}{2}, a_{31} = 1, a_{32} = 2$, the result gives *Kutta's third order formula*

$$\begin{aligned}
y_{n+1} &= y_n + \frac{h}{6}(k_1 + 4k_2 + k_3)\\
k_1 &= f(x,y)\\
k_2 &= f(x + \frac{1}{2}h, y + \frac{1}{2}hk_1)\\
k_2 &= f(x + h, y - hk_1 + 2hk_2)
\end{aligned}$$

5.4. Implicit Runge-Kutta Method

The implicit method is more different than explicit methods. It is such a complicated method; but in solving differential equation, it has helpful numerical stability. The form of s- stage Runge Kutta method is defined as

$$y_{n+1} = y_n + h\sum_{i=1}^{s} b_i k_i$$

where

$$k_i = f(x_n + c_i h, y_n + h\sum_{j=1}^{s} a_{ij}k_j), \quad i = 1, \ldots, s \tag{5.4.2}$$

It has a Butcher tableau

$$
\begin{array}{c|cccc}
c_1 & a_{11} & a_{12} & \dots & a_{1s} \\
c_2 & a_{21} & a_{22} & \dots & a_{2s} \\
\vdots & \vdots & \vdots & \ddots & \vdots \\
c_s & a_{s1} & a_{s2} & \dots & a_{ss} \\
\hline
 & b_1 & b_2 & ... & b_s
\end{array}
$$

If the equation is a system of m differential equation, then the coefficients will be system of sm nonlinear equation. It is clear that s nonlinear equations of system has s unknowns k_i. We now introduce a famous Runge Kutta method.

5.5. Fourth Order Runge-Kutta Method

The most popular method *Fourth Order Runge-Kutta method* is obtained as

$$y_{n+1} = y_n + \frac{1}{6}(k_1 + 2k_2 + 2k_3 + k_4)$$

where

$$
\begin{aligned}
k_1 &= f(x_n, y_n) \\
k_2 &= f(x_n + \frac{1}{2}h, y_n + \frac{1}{2}k_1 h) \\
k_3 &= f(x_n + \frac{1}{2}h, y_n + \frac{1}{2}k_2 h) \\
k_4 &= f(x_n + h, y_n + k_3 h)
\end{aligned}
\tag{5.5.2}
$$

Fourth Order Runge Kutta method is the well-known sample of all Runge Kutta methods. Its tableau is

$$
\begin{array}{c|cccc}
0 & & & & \\
\frac{1}{2} & \frac{1}{2} & & & \\
\frac{1}{2} & 0 & \frac{1}{2} & & \\
1 & 0 & 0 & 1 & \\
\hline
 & \frac{1}{6} & \frac{1}{3} & \frac{1}{3} & \frac{1}{6}
\end{array}
$$

It is clear that RK4 method is consistent over the condition (5.3.8). The local truncation error is $O(h^5)$. Also, it has four evaluation of function f in each step. We can construct the function of f same as the second order method. First we begin with Taylor expansion of exact solution

$$y(x_{n+1}) = y(x_n) + hy'(x_n) + \frac{1}{2}h^2 y''(x_n) + \frac{1}{3!}h^3 y'''(x_n) + \frac{1}{4!}h^4 y''''(x_n) + O(h^5) \quad (5.5.3)$$

Then we replaced the first derivative, second derivative, third derivative, and fourth derivative by using differentiation of (5.2.1). The expression will be interpreted

$$\begin{aligned} y(x_{n+1}) = y(x_n) + hf + \frac{1}{2}h^2 f f_y + \frac{1}{3!}h^3 \left[f_{yy} f^{\,2} + f_y^2 f \right] \quad (5.5.4) \\ + \frac{1}{4!}h^4 \left[f_{yyyy} f^3 + 4 f_{yy} f_y f^2 + f_y^3 f \right] + O(h^5) \end{aligned}$$

To contrast the approximate and exact solution, we need to indicate the expansion of numerical method. By assuming

$$y_{n+1} = y_n + h\left[b_1 k_1 + b_2 k_2 + b_3 k_3 + b_4 k_4\right] \quad (5.5.5)$$

with

$$
\begin{aligned}
k_1 &= f(x,y) \\
k_2 &= f(x_n + c_2 h, y_n + h a_{21} k_1) \qquad (5.5.6) \\
k_3 &= f(x_n + c_3 h, y_n + h[a_{31} k_1 + a_{32} k_2]) \\
k_4 &= f(x_n + c_4 h, y_n + h[a_{41} k_1 + a_{42} k_2 + a_{43} k_3]
\end{aligned}
$$

In order to extend k_i's we use again Taylor expansion formulae

$$\sum_{m=0}^{\infty} \sum_{j=0}^{m} \frac{1}{j!(m-j)!} D_1^j D_2^{(m-j)} f(x,y) h^j k^{(m-j)} \qquad (5.5.7)$$

Here, expansion is only depending on y, so we neglect x which appears in argument of f. After expansion is carry out, substitute k_i's to numerical method. Afterwards, we have to match the expansions (5.5.4) and (5.5.5). We achieve ([16])

$$b_1 + b_2 + b_3 + b_4 = 1 \qquad (5.5.8a)$$

$$b_2 c_2 + b_3 c_3 + b_4 c_4 = \frac{1}{2} \qquad (5.5.8b)$$

$$b_2 c_2^2 + b_3 c_3^2 + b_4 c_4^2 = \frac{1}{3} \qquad (5.5.8c)$$

$$b_3 a_{32} c_2 + b_4 a_{42} c_2 + b_4 a_{43} c_3 = \frac{1}{6} \qquad (5.5.8d)$$

$$b_2 c_2^3 + b_3 c_3^3 + b_4 c_4^3 = \frac{1}{4} \qquad (5.5.8e)$$

$$b_3 c_3 a_{32} c_2 + b_4 c_4 a_{42} c_2 + b_4 c_4 a_{43} c_3 = \frac{1}{8} \qquad (5.5.8f)$$

$$b_3 a_{32} c_2^2 + b_4 a_{42} c_2^2 + b_4 a_{43} c_3^2 = \frac{1}{12} \qquad (5.5.8g)$$

$$b_4 a_{43} a_{32} c_2 = \frac{1}{24} \qquad (5.5.8h)$$

In place of above a system of 8 equations in 10 unknowns is obtained. To show the application is so complicated for Fourth Order Runge Kutta method. In contrast, It can be solved by taking some appropriate value. Thus, since $c_4 = 1$ is taken, b_2, b_3, b_4 obtain for equation (5.5.8b),(5.5.8c) and (5.5.8c) ; then solve for a_{32}, a_{42}, a_{43} from equation (5.5.8d), (5.5.8f), (5.5.8g); finally substituting to h. Many solution and $s-$ stages families of solution is arise these condition. There are two well known fourth order methods

$$y_{n+1} = y_n + \frac{h}{6}(k_1 + 2k_2 + 2k_3 + k_4)$$

$$\begin{aligned}
k_1 &= f(x_n, y_n) \\
k_2 &= f(x_n + \frac{1}{2}h, y_n + \frac{1}{2}k_1 h) \\
k_3 &= f(x_n + \frac{1}{2}h, y_n + \frac{1}{2}k_2 h) \\
k_4 &= f(x_n + h, y_n + k_3 h)
\end{aligned}$$

and

$$y_{n+1} = y_n + \frac{h}{8}(k_1 + 3k_2 + 3k_3 + k_4)$$

$$k_1 = f(x_n, y_n)$$

$$k_2 = f(x_n + \frac{1}{3}h, y_n + \frac{1}{3}hk_1)$$

$$k_3 = f(x_n + \frac{2}{3}h, y_n - \frac{1}{3}hk_1 + hk_2)$$

$$k_4 = f(x_n + h, y_n + hk_1 - hk_2 + hk_3)$$

5.6. Stability of Runge-Kutta Method

Runge-Kutta method has the same property of stability. Observe that the model problem

$$y' = \lambda y \tag{5.6.1}$$

$$y(0) = y_0$$

with real and negative λ. Denote $\mathbf{z}^T = [z_1, z_2, \ldots, z_s]$ and $\mathbf{e}^T = [1, 1, \ldots, 1]$ is the $s-$dimensional vector. Apply the general form of $s-$stages to model problem to get

$$\mathbf{z}_n = y_n\mathbf{e} + h\lambda\mathbf{A}\mathbf{z}_n \tag{5.6.2}$$

$$y_{n+1} = y_n + h\lambda\mathbf{b}^T\mathbf{z}_n$$

After some operations applied, it gives

$$y_{n+1} = y_n + h\lambda\mathbf{b}^T(1 - h\lambda\mathbf{A})^{-1}\mathbf{e}y_n = \left(1 + h\lambda\mathbf{b}^T(1 - h\lambda\mathbf{A})^{-1}\mathbf{e}\right)y_n \tag{5.6.3}$$

Then the method has stability function if

$$R(\eta) = \left(1 + h\lambda \mathbf{b}^T (1 - h\lambda \mathbf{A})^{-1} \mathbf{e}\right) \tag{5.6.4}$$

The Runge-Kutta method is *A−Stable* if modulus of function is less than 1 for $\mathrm{Re}\,(h\lambda)$<0 which is

$$|R(\eta)| < 1 \tag{5.6.5}$$

for all complex $h\lambda$.

Definition 5.6.1 *([24]) The method (5.3.2) is said to have **order** p if p is the largest integer for which*

$$y(x+h) - y(x) - hF(x, y(x); h) = O(h^{p+1}) \tag{5.6.6}$$

holds, where $y(x)$ is known as the theoretical solution of the initial value problem.

Definition 5.6.2 *([26]) The method (5.3.2) is said to be **consistent** with initial value problem if*

$$F(x, y, 0) \equiv f(x, y) \tag{5.6.7}$$

The method is **zero stable** if $F(x, y(x); h) = 0$ as $h \longrightarrow 0$. It can be shown as follows:

$$y_{n+1} = y_n + \frac{1}{6}(k_1 + 2k_2 + 2k_3 + k_4) \tag{5.6.8}$$

This equation becomes

$$y_{n+1} - y_n = 0 \tag{5.6.9}$$

If we apply the root condition, the characteristic equation is satisfied

$$r - 1 = 0$$

$$r = 1$$

Then root conditions hold when the modulus of root is less than or equal to 1 which shown as $|r| \leq 1$. We can say that the fourth order of Runge Kutta method is zero stable.

To confirm that Runge Kutta method have greater accuracy than Euler method, we need to calculate the local truncation error. Assume that

$$F(y_n, x_n; h) = \sum_{i=1}^{s} b_i k_i$$

RK4 is consistent if $F(y_n, x_n; 0) = f(x_n, y_n))$ then we must approve this rule. Replacing the consistency condition into $F(y_n, x_n; h) = \sum_{i=1}^{s} b_i k_i$, we get

$$\sum_{i=1}^{s} b_i = 1$$

which is the necessary condition to satisfy the consistency condition for Runge-Kutta method. Furthermore, we can define the local truncation error is

$$T_{n+1} = y(x_{n+1}) - y(x_n) - hF(y(x_n), x_n; h)$$

The comparison of definition (5.6.2) and the local truncation error gives us

$$T_{n+1} = O(h^{p+1})$$

where p is the order of the method. It is clear that $T_{n+1} = O(h^5)$ is the local truncation error of Fourth Order Runge Kutta method. Order of method assure the accuracy of the method. For RK4 method, the local truncation error is appeared $O(h^5)$. This means that the order of RK4 method is four. In case that RK4 is consistent and zero stable, RK4 converges to analytical solution.

The definition of convergence is given in Chapter 2. To investigate the convergence of Runge-Kutta method, consistency and stability condition must be hold. Consider the form of method

$$y_{n+1} = y_n + hF(x_n, y_n; h), \quad n \geq 0 \tag{5.6.10}$$

which refers to the numerical solution of initial value problem (5.2.1). Using the truncation error

$$T_{n+1} = y(x_{n+1}) - y(x_n) - hF(x, y(x), h; f) \tag{5.6.11}$$

we define

$$\tau_n(y) = \frac{1}{h}T_{n+1}(y)$$

In order to show the convergence of the solution (5.6.10), we need to have $\tau_n(y) \longrightarrow 0$ as $h \longrightarrow 0$ since

$$\tau_n(y) = \frac{y(x_{n+1}) - y_n}{h} - F(x, y(x), h; f) \qquad (5.6.12)$$

We need that

$$F\,(x_n, y(x_n), h; f) \longrightarrow y^{'}(x) = f(x, y(x)) \;\; as\; h \longrightarrow 0$$

This shows that

$$\eta(h) = sup \;\; _{x_0 \le x \le b,\; -\infty < y < \infty} |f(x, y) - F(x, y, h; f)| \qquad (5.6.1)$$

and since

$$\eta(h) \longrightarrow 0 \quad \text{as} \;\; h \longrightarrow 0 \qquad (5.6.14)$$

the consistency condition is hold. We can also show the consistency result by other way. Rewrite (5.6.12) in the form

$$y(x_{n+1}) = y(x_n) + hF(x_n, y(x_n), h; f) + h\tau_n(y) \qquad (5.6.15)$$

then define

$$\tau(h) = \max_{x_0 \le x \le b} |\tau_n(y)|$$

The condition can be used for consistency condition while $\tau(h) \longrightarrow 0$ as $h \longrightarrow 0$. Apart from that we need to check the Lipschitz condition on F;

$$|F(x,y,h;f) - F(x,z,h;f)| \le L|y-z| \tag{5.6.16}$$

for all $x_0 \le x \le b$, $-\infty < y, z < \infty$ and $h > 0$.

Theorem 5.6.3 *([10])Assume that the Runge-Kutta method satisfies the Lipschitz condition. Then, for the initial value problem, the solution $\{y_n\}$ satisfies*

$$\max_{x_0 \le x_n \le b} |y(x_n) - y_n| \le e^{(b-x_0)L} |y(x_0) - y_0| + \left[\frac{e^{(b-x_0)L} - 1}{L}\right] \tau(h) \tag{5.6.17}$$

where

$$\tau(h) \equiv \max_{x_0 \le x_n \le b} |\tau_n(y)| \tag{5.6.18}$$

If the consistency condition (5.6.14) is also satisfied, then the numerical solution $\{y_n\}$ converges to $y(x)$.

Proof. ([11])Subtract (5.6.15) from (5.6.10) to obtain

$$e_{n+1} = e_n + h\left[F(x_n, y(x_n), h; f) - F(x_n, y_n, h; f)\right] + h\tau_n(y) \tag{5.6.19}$$

in which $e_n = y(x_n) - y_n$. Apply the Lipschitz condition (5.6.16) and use (5.6.18) to get

$$|e_{n+1}| \leq (1 + hL)|e_n| + h\tau(h), \quad x_0 \leq x_N \leq b \tag{5.6.20}$$

As with the convergence proof in Theorem 4.3.2 for the Euler method, given in Section 4.3 of Chapter 3, this leads easily to the result (5.6.9). In most cases, it is known by direct computation that $\tau(h) \longrightarrow 0$ as $h \longrightarrow 0$, and in that case, convergence of $\{y_n\}$ to $y(x)$ is immediately proved. But all that we need to know is that (5.6.14) is satisfied. To see this, write

$$\begin{aligned}
h\tau_n(y) &= y(x_{n+1}) - y(x_n) - hF(x_n, y(x_n), h; f) \\
&= hy'(x_n) + \frac{1}{2}h^2 y''(\xi_n) - hF(x_n, y(x_n); h), \quad y'(x_n) = f(x_n, y(x_n)) \\
h|\tau_n(y)| &\leq h\eta(h) + \frac{1}{2}h^2 \left\|y''\right\|_\infty \\
\tau(h) &\leq \eta(h) + \frac{1}{2}h \left\|y''\right\|_\infty
\end{aligned}$$

Thus $\tau(h) \longrightarrow 0$ as $h \longrightarrow 0$, completing the proof. ■

5.6.1. Absolute Stability of Runge-Kutta Method

The adequate technique is to consider model problem

$$y' = \lambda y \tag{5.7.1}$$

$$y(0) = y_0$$

The analytical solution of initial value problem is $y^{'}(x) = y_0 \exp(\lambda x)$ where λ negative real number. Here, analytical solution converge to 0 as $x \longrightarrow +\infty$ at the exponential part. Note that this condition also is necessary for the solution of Runge Kutta method. We investigate that what condition on step size the solution of Runge Kutta method will satisfy the same behavior. By choosing the appropriate step size in the numerical solution of initial value problem on the interval $[x_0, x_m]$ with $x_m \gg x_0$, the numerical solution reproduce desired status. When the numerical solution is applied to the model problem, the interval of line $h\lambda$ must satisfy the absolute stability condition. This shows that solution of Runge Kutta method tends to zero as $x \longrightarrow \infty$. Now, perform the Runge Kutta fourth order method to model problem to obtain

$$\begin{aligned}
k_1 &= f(x,y) = \lambda y \\
k_2 &= f(x_n + c_2 h, y_n + h a_{21} k_1) = \lambda(y + h a_{21} \lambda y) = \\
&= \lambda y (1 + a_{21} h\lambda) \\
k_3 &= f(x_n + c_3 h, y_n + h[a_{31} k_1 + a_{32} k_2]) \\
&= \lambda(y + h[a_{31}\lambda y + a_{32}\lambda y(1 + a_{21} h\lambda)]) \\
&= \lambda y \Big(1 + h\lambda[a_{31} + a_{32}] + h^2\lambda^2 a_{32} a_{21}\Big) \\
k_4 &= f(x_n + c_4 h, y_n + h[a_{41} k_1 + a_{42} k_2 + a_{43} k_3] = \\
&= \lambda\Big(y + h[a_{41}\lambda y] + h a_{42}\lambda y[1 + a_{21} h\lambda] + h a_{43}\lambda y\Big[1 + h\lambda[a_{31} + a_{32}] + h^2\lambda^2 a_{32} a_{21}\Big]\Big) \\
&= \lambda y \Big(1 + h\lambda a_{41} + h\lambda a_{42} + h^2\lambda^2 a_{42} a_{21} + h\lambda a_{43} + h^2\lambda^2 a_{43}[a_{31} + a_{32}] + h^3\lambda^3 a_{43} a_{32} a_{21}\Big)
\end{aligned} \tag{5.7.2}$$

Assuming $\bar{h} = h\lambda$ and $\sum_{j=1}^{i-1} a_{ij} = c_i$, $(i = 1,..,s)$, then rewrite (5.7.2)

$$\begin{aligned}
k_1 &= \lambda y \\
k_2 &= \lambda y\left(1 + a_{21}\bar{h}\right) \\
k_3 &= \lambda y\left(1 + \bar{h}c_3 + \bar{h}^2 a_{32}a_{21}\right) \\
k_4 &= \lambda y\left(1 + \bar{h}c_4 + \bar{h}^2 a_{42}a_{21} + \bar{h}^2 a_{43}c_3 + \bar{h}^3 a_{43}a_{32}a_{21}\right)
\end{aligned} \tag{5.7.3}$$

Afterwards, substitute into (5.2.3) to obtain

$$F(x, y; h) = b_1k_1 + b_2k_2 + b_3k_3 + b_4k_4$$

and

$$\begin{aligned}
y_{n+1} &= y_n + \bar{h}\begin{bmatrix} b_1 + b_2\left(1 + a_{21}\bar{h}\right) + b_3\left(1 + \bar{h}c_3 + \bar{h}^2 a_{32}a_{21}\right) \\ +b_4\left(1 + \bar{h}c_4 + \bar{h}^2 a_{42}a_{21} + \bar{h}^2 a_{43}c_3 + \bar{h}^3 a_{43}a_{32}a_{21}\right) \end{bmatrix} y_n \\
y_{n+1} - y_n &= \bar{h}\begin{bmatrix} (b_1 + b_2 + b_3 + b_4) + (b_2a_{21} + b_3c_3 + b_4c_4)\bar{h} \\ +(b_3a_{32}a_{21} + b_4a_{42}a_{21} + b_4a_{43}c_3)\bar{h}^2 + b_4a_{43}a_{32}a_{21}\bar{h}^3 \end{bmatrix} y_n \\
y_{n+1}/y_n &= 1 + (b_1 + b_2 + b_3 + b_4)\bar{h} + (b_2a_{21} + b_3c_3 + b_4c_4)\bar{h}^2 \\
&\quad + (b_3a_{32}a_{21} + b_4a_{42}a_{21} + b_4a_{43}c_3)\bar{h}^3 + b_4a_{43}a_{32}a_{21}\bar{h}^4
\end{aligned} \tag{5.7.4}$$

Now, we turn to the conditions (5.5.8a), (5.5.8b), (5.5.8d), (5.5.8h). It is evident that the difference equation satisfies these conditions; so it is obtained

$$y_{n+1} = \left(1 + \bar{h} + \frac{1}{2}\bar{h}^2 + \frac{1}{6}\bar{h}^3 + \frac{1}{24}\bar{h}^4\right) y_n \tag{5.7.5}$$

Consequently, method is *absolutely stable*, that is, y_n tends to 0 if and only if

$$\left|1 + \bar{h} + \frac{1}{2}\bar{h}^2 + \frac{1}{6}\bar{h}^3 + \frac{1}{24}\bar{h}^4\right| < 1 \tag{5.7.6}$$

The plot of this function towards $\bar{h}$ pose that *the interval of absolute stability* in $\bar{h} \epsilon [-2.78, 0]$.

Chapter 6

SYSTEM OF ORDINARY DIFFERENTIAL EQUATIONS

6.1. Introduction

More application problems include a system of several equation. The solution of a system of ordinary differential equation is required in engineering and science that has more complicated situations. The initial value problem of m differential equation's system can be put into a form as

$$\mathbf{y}'(x) = \mathbf{f}(x, \mathbf{y}(x)), \quad \mathbf{y}(x_0) = \mathbf{y}_0 \tag{6.1}$$

We can write as follows

$$\begin{aligned}
y_1'(x) &= f_1(x, y_1(x), y_2(x), ..., y_m(x)), \qquad y_1(x_0) = y_{1,0} \\
y_2'(x) &= f_2(x, y_1(x), y_2(x), ..., y_m(x)), \qquad y_2(x_0) = y_{2,0} \\
&\vdots \\
y_m'(x) &= f_m(x, y_1(x), y_2(x), ..., y_m(x)), \qquad y_m(x_0) = y_{m,0}
\end{aligned}$$

with the given some interval $x_0 \leq x \leq b$. The general form of system can be represented the solution and the differential equation by using the column vector. Indicate,

$$\mathbf{y}(x) = \begin{bmatrix} y_1(x) \\ y_2(x) \\ \vdots \\ y_m(x) \end{bmatrix}, \mathbf{y}_0 = \begin{bmatrix} y_{1,0} \\ y_{2,0} \\ \vdots \\ y_{m,0} \end{bmatrix}, \mathbf{f}(x, \mathbf{y}) = \begin{bmatrix} f_1(x, y_1, y_2, ..., y_m) \\ f_2(x, y_1, y_2, ..., y_m) \\ \vdots \\ f_m(x, y_1, y_2, ..., y_m) \end{bmatrix}$$

with $\mathbf{y} = [y_1, y_2, ..., y_m]^T$.

Example 6.1.1 *([13])The initial value problem*

$$y_1' = y_1(x) - 2y_2(x) + 4\cos(x) - 2\sin(x)\,, \;\; y_1(0) = 1,$$

$$y_2' = 3y_1(x) - 4y_2(x) + 5\cos(x) - 5\sin(x), \;\; y_2(0) = 2$$

has the solution

$$y_1(x) = \cos(x) + \sin(x), \qquad y_2(x) = 2\cos(x)$$

System can be written as

$$\mathbf{y}'(x) = A\mathbf{y}(x) + \Phi(x), \qquad \mathbf{y}(0) = \mathbf{y}_0$$

with

$$\mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \end{bmatrix}, A = \begin{bmatrix} 1 & -2 \\ 3 & -4 \end{bmatrix},$$

$$\Phi(x) = \begin{bmatrix} 4\cos(x) - 2\sin(x) \\ 5\cos(x) - 5\sin(x) \end{bmatrix}, \mathbf{y}_0 = \begin{bmatrix} 1 \\ 2 \end{bmatrix}$$

In the notation above it can be represented as

$$\mathbf{f}(x,\mathbf{y}) = A\mathbf{y} + \mathbf{\Phi}(x), \quad \mathbf{y} = [y_1, y_2]^{\mathbf{T}}$$

6.2. Stability Theory for System

Stability means that small disturbance in the initial value problems causes a small change in the solution. Here, we consider the numerical methods for solving the initial value problems that are numerically stable. Small change in initial value problem will cause a small change in the numerical solution for any sufficiently small step size.

To examine the stability of system keep in view the initial value problem

$$y'(x) = f(x, y(x)),$$

$$y(x_0) = y_0$$

Instead of this, we consider the stability of numerical method for the model problem

$$y'(x) = \lambda y(x) + g(x)$$

$$y(0) = 1$$

Stability and convergence can be answered for this problem. Expand $y'(x) = f(x,y)$ to get

$$f(x,y(x)) = f(x_0,y_0) + f_x(x_0,y_0)(x - x_0) + f_y(x_0,y_0)(y(x) - y_0)$$

Thus,

$$\begin{aligned} y'(x) &= f(x,y(x)) \\ &= \lambda(y(x) - y_0) + g(x) \end{aligned}$$

with

$$g(x) = f(x_0,y_0) + f_x(x_0,y_0)(x - x_0),$$

$$\lambda = f_y(x_0,y_0)$$

Let $V(x) = y(x) - y_0$, then we obtain

$$V'(x) = \lambda V(x) + g(x)$$

which is called **model equation** for the initial value problem. Make small change in the initial value problem and see the difference in the solution

$$V_\epsilon'(x) = \lambda V_\epsilon(x) + g(x)$$

$$V_\epsilon(x_0) = V_0 + \epsilon$$

Here, $g(x)$ will be cancelled. Because we are interested in the differ of the solutions and we get model problem. Subtracting them we get

$$V_\epsilon'(x) - V^{'}(x) = \lambda(V_\epsilon(x) - V(x))$$

$$V_\epsilon(x_0) - V(x_0) = \epsilon$$

Take $W = V_\epsilon'(x) - V'(x)$ to obtain

$$W' = \lambda W$$

$$W(x_0) = \epsilon$$

These operations demonstrate the stability and convergence of model equation. Now, we examine the more general problem for system by considering the model equation. It is similar with the above analysis.

The initial value problem of m differential equations of system describe as

$$\mathbf{y}' = \mathbf{f}(x, \mathbf{y}), \quad x_0 \le x \le b$$

$$\mathbf{y}(x_0) = \mathbf{y}_0$$

The final version of the model problem is defined

$$\mathbf{y}'(x) = \wedge\mathbf{y}(x) + \mathbf{g}(x)$$

with $\wedge = \mathbf{f}_y(x_{0,}\mathbf{y}_0)$. if f is differentiable, then $f_y(x, \mathbf{y})$ indicate a Jacobian matrix.

$$\mathbf{f}_y(x, \mathbf{y})_{i,j} = \frac{\partial f_i(x, y_1, y_2, ..., y_m)}{\partial y_j}, \qquad 1 \leq i, j \leq m$$

This system can be written as

$$\mathbf{y}' = \wedge\mathbf{y} + \mathbf{g}(x)$$

which can be reduced to equivalent system

$$z_i' = \lambda_i z_i + \gamma_i(x), \;\; 1 \leq i \leq m$$

with $\lambda_1, \lambda_2, ..., \lambda_m$ the eigenvalues of $\wedge = \mathbf{f}_y(x_0, \mathbf{y}(\mathbf{x}_0))$.

To investigate the stability of multistep method (3), examine the special model equation case

$$y'(x) = \lambda y(x),$$

$$y_0 = 1$$

After applied it to the multistep method, it becomes

$$y_{n+1} = \sum_{j=0}^{p} a_j y_{n-j} + h\lambda \sum_{j=-1}^{p} b_j y_{n-j} \ , n \geq p$$

$$y_{n+1} = \sum_{j=0}^{p} a_j y_{n-j} + h\lambda b_{-1} y_{n+1} + h\lambda \sum_{j=0}^{p} b_j y_{n-j}$$

we get

$$(1 - h\lambda b_{-1}) y_{n+1} = \sum_{j=0}^{p} \left(a_j + h\lambda b_j\right) y_{n-j}$$

This is called a linear difference equation of $(p+1)$.We investigate a general solution in a form

$$y_n = r^n \ , n \geq 0$$

Substitute this form to the method and multiply by r^{p-n}

$$r^{p+1} = \sum_{j=0}^{p} a_j r^{p-j} + h\lambda \sum_{j=-1}^{p} b_j r^{p-j}$$

which is called **characteristic equation**. Denote the second characteristic roots,

$$\sigma(r) = \sum_{j=-1}^{p} b_j r^{p-j} = b_{-1} r^{p+1} + \sum_{j=0}^{p} r^{p-j}$$

and remind that the first root of polynomials

$$\rho(r) = r^{p+1} - \sum_{j=0}^{p} a_j r^{p-j}$$

Thus first and second characteristic polynomial of (2.3) can be defined by

$$\pi(r, h\lambda) = \rho(r) - h\lambda\sigma(r) = 0$$

The stability of numerical methods will be analyzed, when the step size is not chosen very small. The model problem is the instructive technique to state the region of the stability for any methods. We will investigate the case λ for this.

In general multistep methods (2.3), the characteristic equation is represented as

$$r^{p+1} - \sum_{j=0}^{p} a_j r^{p-j} - h\lambda \sum_{j=-1}^{p} b_j r^{p-j} = 0$$

for the determined model equation.

All roots of characteristic equation have magnitude 1 to satisfy the absolute stability. We can find $h\lambda$ from the characteristic equation.

$$r^{p+1} - \sum_{j=0}^{p} a_j r^{p-j} = h\lambda \sum_{j=-1}^{p} b_j r^{p-j}$$
$$h\lambda = \frac{r^{p+1} - \sum_{j=0}^{p} a_j r^{p-j}}{\sum_{j=-1}^{p} b_j r^{p-j}}$$

This shows the region of the stability of method. When the constant λ is real, then λ is negative $(\lambda < 0)$; or when λ is complex, assume that $\mathrm{Re}(\lambda) < 0$ in stable differential equation problems. The true solution of the given model problem is

$$y(x) = e^{\lambda x}$$

Considering the above cases, the solution of model problem tends to zero as x tends to infinity.

$$y(x) \longrightarrow 0 \text{ as } x \longrightarrow \infty$$

When the any numerical method is applied to model problem, the approximated solution satisfied

$$y(x_n) \longrightarrow 0 \;\text{ as } x_n \longrightarrow \infty$$

If $h\lambda$ is satisfied for above definition in any numerical method, then it is called region of absolute stability of the numerical method.

Chapter 7

NUMERICAL EXPERIMENTS ON SIMPLE SYSTEMS

7.1. Introduction

Numerical experiments is mentioned to prove which numerical methods converge faster to analytic solutions. Numerical experiments are investigated over an example. In comparing different numerical methods, numerical experiments must be done using the same of ordinary differential equations.([17]). For comparison bases, Hull and Enright (1976) pointed out some assumptions that must be undertaken. Assumptions include assuming that the method is modelled to integrate between initial values specified. Another assumption is assuming that local error is observed by keeping the absolute error under the specified error tolerance. Differential equation is inspected two varied numerical methods.

The given differential equations are analyzed for Explicit Euler method, Explicit Runge-Kutta method. Analytical solution of ODE is calculated as well. Each one of all is examined at varied step sizes. Step size is started with 0.1 and continued with halved. Afterward, the absolute error is identified.

Initially, exact solution of DE is computed and then approximated solution is calculated by using Matlab software for each numerical method at different step size. Afterwards, absolute error are computed by taking difference analytical solution to approximated solutions and presented in the same table at each step size. Ultimately,errors at each step size of method are compared with other numerical methods. All numbers in the table. The computation process, errors and according to graphs of each steps can be found in Appendix A and Appendix B.

7.2. Numerical Experiments on Stiff System

7.2.1. Numerical Experiments on Explicit Euler method

Consider the following example of stiff system;

$$\begin{aligned}\frac{du}{dx} &= 8u(x) - 5v(x) + 10w(x)\\ \frac{dv}{dx} &= 2u(x) + 1v(x) + 2w(x)\\ \frac{dw}{dx} &= -4u(x) + 4v(x) + 6w(x)\end{aligned}$$

where initial conditions are $u(0) = 2, v(0) = 2, w(0) = -3$. It can be shown in general form as $y(0) = [2,2,-3]^T$. Its theoretical solution is given

$$u(x) = 6e^{2x} - 4e^{3x},$$

$$v(x) = -4e^{3x} + 6e^{2x},$$

$$w(x) = -e^{-2x} + 3e^{2x}$$

where $\mathbf{y}(x) = [u(x), v(x), w(x)]^T$.[36]

Explicit Euler method and Explicit Fourth Order Runge Kutta method of numerical methods is considered to solve system of ordinary differential equations. The system is computed in Matlab software to evaluate values of the approximated and analytical solution for Explicit Euler method and Explicit Fourth Order Runge Kutta method. Initially, Explicit Euler method and Explicit Fourth Order Runge Kutta method is computed for the given system at different step size. The system and analytical solutions is entered into Matlab software for each method separately. Then analytical solution and approximated solution is evaluated with step sizes $0.1, 0.05, 0.025, 0.0125$, 0.00625 and 0.003125. Afterwards, the absolute errors which is obtain by differ approximated solution from analytical solution, are calculated at specified step size. Error tables with exact solution, approximated solution, and graphs, which is related with exact and approximated solution, are shown in Appendix 2. In this table, we can see how the approximated solutions behave when the step size is reduced by half. We can compare the exact solution with approximate solution and identify the safe step size ,which the approximated solution tends to analytical solution, through this table. It can be determined the proximity of the approximated solution to exact solution since the step size is changed.

In order to compute the approximated solution, step size is started with 0.1 for Explicit Euler method. As can be seen from the table in step size $0.1, 0.05$, 0.025 and 0.0125, the resulted of approximated solution is not closeness to exact solution. So, these step sizes do not lie in the region of absolute stability. The difference between exact solution and approximated solution must be excessively small. The computed approximated solutions have much difference from analytical solutions as it seen in the table. Nevertheless, the step size will lie inside the region of absolute stability since $h < 0.003125$. The approximated solution is closer to exact solution. This shows that the small step size provides the better approximation.

Application of Explicit Euler method is not occasionally preferred to use in stiff systems. The reason is that cannot give the accurate approximated solution in fixed error tolerance well. In order to achieve certain approximation, the step size must be taken very small. This cause more iterations and high computation.

7.2.2. Numerical Experiments on Explicit Fourth Order Runge-Kutta method

The same system is computed in Matlab software. It is taken for comparing the Explicit Euler method and Explicit Runge Kutta method. The same technique which is implemented for Explicit Euler method, is applied to RK4. The computation of Explicit Runge Kutta of fourth order method in same stiff system is also given at different step sizes in Table for given systems. Step size is started from 0.1 and computation progress is proceed of half. Evaluation of analytical solution is same and the approximated solution is computed by using the step sizes $0.1, 0.05, 0.025, 0.0125$ respectively. In addition, local error of numerical method is generated by taking modulus. Error tables with graphic can be found in Appendix 2. As it is seen in table, at the step size $h = 0.1$, errors in system lie in the region of absolute stability. Thus,absolute error is started to behave as approximate solutions But, the approximation of $u(x), v(x), w(x)$ is not close to analytical solution. It is also appreciable in absolute error which is not small absolute error. Again, when step sizes are taken 0.05 and 0.025, we can see in the table that absolute error is started to behave as approximate solutions. These results indicate that the acceptable approximations, and step sizes are in the region of the absolute stability. if the step size is used less than 0.0125, the approximate solution will have same solutions with analytical solutions. It is also noticeable from the error evaluation that have excessively small difference. This shows that approximated solution turn into the exact solution at step size $h = 0.0125$.

Chapter 8

COMPARISON OF EULER METHOD AND RUNGE-KUTTA METHOD WITH ADVANTAGES AND DISADVANTAGES

1. **EULER METHOD:**

   Advantages:

   Euler's method is the simplest of all linear multi-step method to obtain the approximated solution of the specified initial value problem. It has the one-step techniques, and it can be easily programmed. Most of the ordinary differential equations can be solved conjecturally with numerical method and approximated solution y_{n+1} can be obtained from y_n. The derivation of Euler's method can be revealed by constructing Taylor series. Approximated solution can be acquired at each step before progressing to the next step. In addition, it has merely single computation of function f in each time (step-size). The error analysis, which involve local and global truncation error, and remainder term can be obtained smoothly. Consequently, Euler's method has the practical solution techniques in order to solve the complicated differential equations.

   Disadvantages:

   In spite of the simplicity, it is restricted to use. The reason is; it generate large error in each successive step during the computation which is the accumulated error. In order to avoid the formation of larger error, step-size should be taken excessively small.Therefore, it needs high computation of time. Additively, approximated solution converge slower to analytical solution. This means that the order of method is 1 and the error is observable $O\left(h^2\right)$. It is a slow rate of convergence.

2. **RUNGE-KUTTA METHOD:**

   Advantages:

   The idea of families of Runge Kutta method is too complicated, but higher order provides much better approximated solutions than Euler method. The most popular Runge-Kutta method is the method of order four. It is good choice to get more accurate and more efficient solutions for solving the specified ordinary differential equations.The approximated solution converge faster to exact solution and the order of RK4 is 4 and the truncation error is $O\left(h^5\right)$.

   Disadvantages:

   Method is re-evaluating the function f at each time to obtain the predictable solution. It requires four evaluation per step. So, the computation of function may take long time. The derivation of Runge-Kutta method is obtained from Taylor series, but it is tedious to calculate higher derivative. To avoid this, the function f is evaluated at more points.

Chapter 9

CONCLUSION

In this thesis, we have discussed numerical methods for solving systems of ordinary differential equations. Some necessary conditions and definitions are given to examine the numerical methods. After that, by considering these conceptions and definitions, Euler methods and Runge Kutta method of order 4 are developed and, derivations and stabilities of both methods are discussed. Afterwards, the system of ODE is given and, we discussed the efficiency of two methods at the different step size for systems using Tables of approximated solutions with exact solutions. At the end, we gave information about the advantages and disadvantages of Euler's method and Runge Kutta method. Consequently, we see that in the Euler's method excessively small step size converge to analytical solution. Therefore, large number of computation is needed. In contrast, Runge Kutta method gives better results and it converge faster to analytical solution and has less iteration to get accuracy solution.

REFERENCES

[1] Atkinson, K. E., and Han, W., and Steward, D.,(1961), *Numerical Solutions of Ordinary Differential Equations*, John Wiley & Sons, Inc., Hoboken, New Jersey, Canada

[2] Atkinson, K. E., and Han, W., and Steward, D.,(1961), *Numerical Solutions of Ordinary Differential Equations*, John Wiley & Sons, Inc., Hoboken, New Jersey, Canada

[3] Atkinson, K. E., and Han, W., and Steward, D.,(1961), *Numerical Solutions of Ordinary Differential Equations*, John Wiley & Sons, Inc., Hoboken, New Jersey, Canada

[4] Atkinson, K. E., and Han, W., and Steward, D.,(1961), *Numerical Solutions of Ordinary Differential Equations*, John Wiley & Sons, Inc., Hoboken, New Jersey, Canada

[5] Atkinson, K. E., and Han, W., and Steward, D.,(1961), *Numerical Solutions of Ordinary Differential Equations*, John Wiley & Sons, Inc., Hoboken, New Jersey, Canada

[6] Atkinson, K. E., and Han, W., and Steward, D.,(1961), *Numerical Solutions of Ordinary Differential Equations*, John Wiley & Sons, Inc., Hoboken, New Jersey, Canada

[7] Atkinson, K. E., and Han, W., and Steward, D.,(1961), *Numerical Solutions of Ordinary Differential Equations*, John Wiley & Sons, Inc., Hoboken, New Jersey, Canada

[8] Atkinson, K. E., and Han, W., and Steward, D.,(1961), *Numerical Solutions of Ordinary Differential Equations*, John Wiley & Sons, Inc., Hoboken, New Jersey, Canada

[9] Atkinson, K. E., and Han, W., and Steward, D.,(1961), *Numerical Solutions of Ordinary Differential Equations*, John Wiley & Sons, Inc., Hoboken, New Jersey, Canada

[10] Atkinson, K. E., and Han, W., and Steward, D.,(1961) *Numerical Solutions of Ordinary Differential Equations*, John Wiley & Sons, Inc., Hoboken, New Jersey, Canada

[11] Atkinson, K. E., and Han, W., and Steward, D.,(1961), *Numerical Solutions of Ordinary Differential Equations*, John Wiley & Sons, Inc., Hoboken, New Jersey, Canada

[12] Atkinson, K. E., and Han, W., and Steward, D.,(1961), *Numerical Solutions of Ordinary Differential Equations*, John Wiley & Sons, Inc., Hoboken, New Jersey, Canada

[13] Atkinson, K. E., and Han, W., and Steward, D.,(1961) *Numerical Solutions of Ordinary Differential Equations*, John Wiley & Sons, Inc., Hoboken, New Jersey, Canada

[14] Atkinson, K. E., and Han, W., and Steward, D.,(1961) *Numerical Solutions of Ordinary Differential Equations*, John Wiley & Sons, Inc., Hoboken, New Jersey, Canada

[15] Atkinson, K. E., and Han, W., and Steward, D.,(1961), *Numerical Solutions of Ordinary Differential Equations*, John Wiley & Sons, Inc., Hoboken, New Jersey, Canada

[16] Butcher, J., C., (1933), *Numerical Methods for Ordinary Differential Equations,*John Wiley & Sons, The University of Auckland, New Zealand

[17] Hull, T.E. and Enright,W.H., (1976), The Results on Initial Value Methods for Non-Stiff Ordinary Differential Equations, *Society for Industrial and Applied Mathematics Journal on Numerical Analysis,* 13(6), pp,944-961.

[18] Lambert J.D., (1973), *Computational Methods in Ordinary Differential Equations*, London

[19] Lambert J.D., (1973), *Computational Methods in Ordinary Differential Equations*, London

[20] Lambert J.D., (1973), *Computational Methods in Ordinary Differential Equations*, London

[21] Lambert J.D., (1973), *Computational Methods in Ordinary Differential Equations*, London

[22] Lambert J.D., (1973), *Computational Methods in Ordinary Differential Equations*, London

[23] Lambert J.D., (1973), *Computational Methods in Ordinary Differential Equations*, London

[24] Lambert J.D., (1973), *Computational Methods in Ordinary Differential Equations*, London

[25] Lambert J.D., (1973), *Computational Methods in Ordinary Differential Equations*, London

[26] Lambert J.D., (1973), *Computational Methods in Ordinary Differential Equations*, London

[27] Lambert J.D., (1973), *Computational Methods in Ordinary Differential Equations*, London

[28] Lambert J.D., (1973), *Computational Methods in Ordinary Differential Equations*, London

[29] Lambert J.D., (1973), *Computational Methods in Ordinary Differential Equations*, London

[30] Lambert J.D., (1973), *Computational Methods in Ordinary Differential Equations*, London

[31] http://www.math.uiowa.edu/~atkinson/m171.dir/sec_6-3.pdf

[32] http://people.maths.ox.ac.uk/suli/nsodes.pdf

[33] http://people.maths.ox.ac.uk/suli/nsodes.pdf

[34] http://people.maths.ox.ac.uk/suli/nsodes.pdf

[35] http://people.maths.ox.ac.uk/suli/nsodes.pdf

[36] http://plato.asu.edu/MAT275/odes_matlab/Chapter12.pdf

[37] http://www.siam.org/books/ot98/sample/OT98Chapter7.pdf

APPENDICES

Appendix A. Explicit Euler method

.

Table 1. Explicit Euler method with h=0.1 for n=10

i	h	$u_{\text{app}}(x)$	$v_{\text{app}}(x)$	$w_{\text{app}}(x)$	$u(x)$	$v(x)$	$w(x)$	$\lvert u(x)-u_{\text{app}}(x)\rvert$	$\lvert v(x)-v_{\text{app}}(x)\rvert$	$\lvert w(x)-w_{\text{app}}(x)\rvert$
0	0	2	2	−3						
1	0.1	−0.4	2	−1.2	−0.487051	1.92898	−1.24818	0.0870507	0.0710187	0.0481762
2	0.2	−2.92	1.88	0.48	−3.26655	1.66247	0.453554	0.346555	0.217527	0.0264462
3	0.3	−5.716	1.58	2.112	−6.54554	1.0943	2.17349	0.829543	0.4857	0.0614866
4	0.4	−8.9668	1.0172	3.7632	−10.5845	0.0727779	3.98065	1.61769	0.944422	0.217449
5	0.5	−12.8856	0.0782	5.49888	−15.7195	−1.61707	5.94757	2.83384	1.69527	0.448689
6	0.6	−17.7344	−1.39133	7.38509	−22.3914	−4.27789	8.15319	4.65705	2.88656	0.768097
7	0.7	−23.8411	−3.60032	9.49125	−31.1851	−8.33348	10.686	7.34398	4.73316	1.19477
8	0.8	−31.6226	−6.83033	11.8928	−42.8813	−14.3745	13.6477	11.2587	7.54418	1.7549
9	0.9	−41.6127	−11.4593	14.674	−58.5271	−23.221	17.1571	16.9144	11.7617	2.48311
10	1	−54.4992	−17.993	17.931	−79.5301	−36.0078	21.3552	25.031	18.0148	3.42419

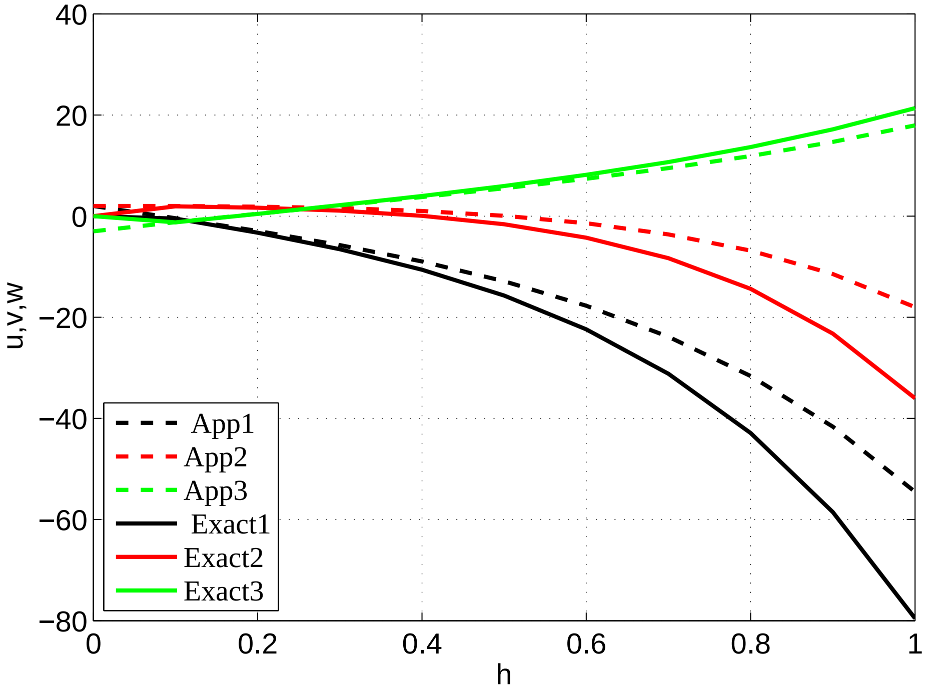

Figure 1. Graph of approximated solution and exact solution by using Explicit Euler method with $h = 0.1$

Table 2. Explicit Euler method with h=0.05 for n=20

i	h	$u_{\text{app}}(x)$	$v_{\text{app}}(x)$	$w_{\text{app}}(x)$	$u(x)$	$v(x)$	$w(x)$	$\lvert u(x)-u_{\text{app}}(x)\rvert$	$\lvert v(x)-v_{\text{app}}(x)\rvert$	$\lvert w(x)-w_{\text{app}}(x)\rvert$
0	0	2	2	−3						
2	0.1	−0.43	1.97	−1.23	−0.487051	1.92898	−1.24818	0.0570507	0.0410187	0.0181762
4	0.2	−3.05943	1.78857	0.4557	−3.26655	1.66247	0.453554	0.20713	0.126102	0.00214618
6	0.3	−6.0636	1.37712	2.12604	−6.54554	1.0943	2.17349	0.481946	0.282823	0.0474496
8	0.4	−9.65329	0.625441	3.84796	−10.5845	0.0727779	3.98065	0.931206	0.552664	0.132686
10	0.5	−14.0902	−0.619776	5.68916	−15.7195	−1.61707	5.94757	1.62932	0.997289	0.258412
12	0.6	−19.7064	−2.57043	7.72071	−22.3914	−4.27789	8.15319	2.685	1.70746	0.432478
14	0.7	−26.9302	−5.51783	10.0199	−31.1851	−8.33348	10.686	4.25488	2.81565	0.66613
16	0.8	−36.3187	−9.86065	12.6731	−42.8813	−14.3745	13.6477	6.56266	4.51387	0.974611
18	0.9	−48.6012	−16.1423	15.7792	−58.5271	−23.221	17.1571	9.92589	7.07873	1.37796
20	1	−64.7367	−25.1011	19.453	−79.5301	−36.0078	21.3552	14.7934	10.9067	1.90212

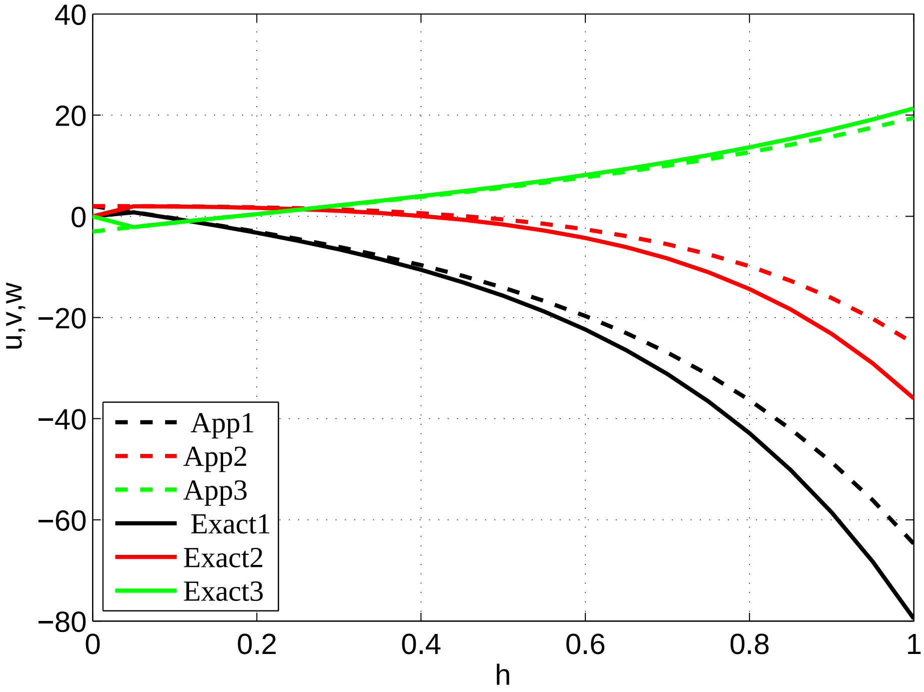

Figure 2. Graph of approximated solution and exact solution by using Explicit Euler method with $h = 0.05$

Table 3. Explicit Euler method with h=0.025 for n=40

i	h	$u_{\text{app}}(x)$	$v_{\text{app}}(x)$	$w_{\text{app}}(x)$	$u(x)$	$v(x)$	$w(x)$	$\left\|u(x)-u_{\text{app}}(x)\right\|$	$\left\|v(x)-v_{\text{app}}(x)\right\|$	$\left\|w(x)-w_{\text{app}}(x)\right\|$
0	2	2	2	−3						
4	0.1	−0.4548	1.95116	−1.2405	−0.4871	1.92898	−1.2482	0.03221	0.02218	0.00766
8	0.2	−3.1534	1.73082	0.45184	−3.2666	1.66247	0.45355	0.11317	0.06835	0.00171
12	0.3	−6.285	1.24802	2.14541	−6.5455	1.0943	2.17349	0.26059	0.15372	0.02808
16	0.4	−10.082	0.37408	3.90786	−10.585	0.07278	3.98065	0.50208	0.3013	0.07279
20	0.5	−14.841	−1.0716	5.80898	−15.72	−1.6171	5.94757	0.87899	0.54545	0.13859
24	0.6	−20.94	−3.3409	7.92337	−22.391	−4.2779	8.15319	1.45186	0.93699	0.22982
28	0.7	−28.877	−6.783	10.3334	−31.185	−8.3335	10.686	2.30827	1.55046	0.35259
32	0.8	−39.308	−11.88	13.1326	−42.881	−14.375	13.6477	3.57381	2.49438	0.51516
36	0.9	−53.1	−19.295	16.4288	−58.527	−23.221	17.1571	5.42767	3.9258	0.72838
40	1	−71.406	−29.937	20.3489	−79.53	−36.008	21.3552	8.12425	6.07079	1.00626

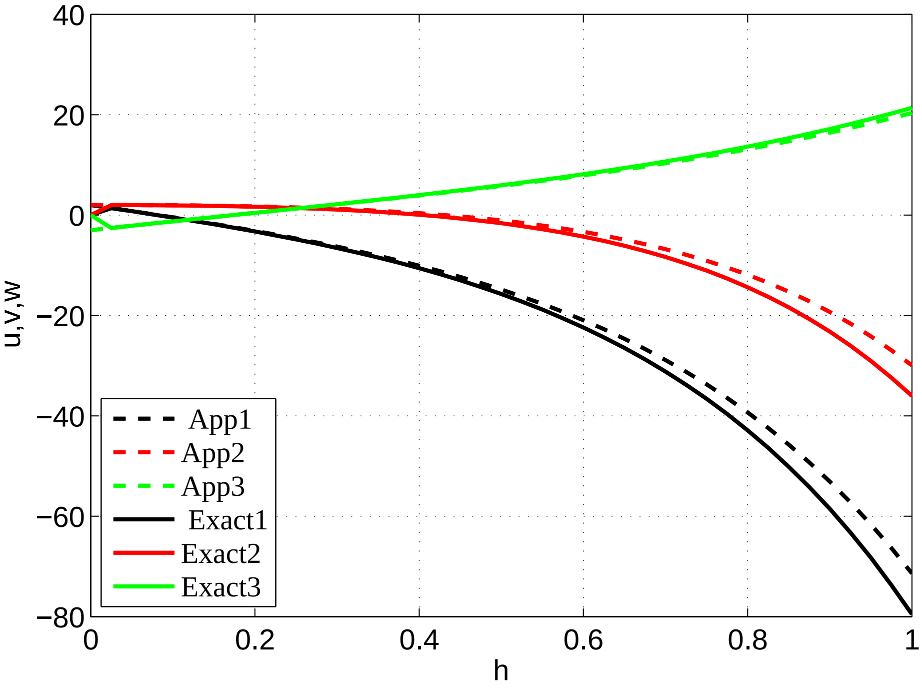

Figure 3. Graph of approximated solution and exact solution by using Explicit Euler method with $h = 0.025$

Table 4. Explicit Euler method with h=0.0125 for n=80

i	h	$u_{\text{app}}(x)$	$v_{\text{app}}(x)$	$w_{\text{app}}(x)$	$u(x)$	$v(x)$	$w(x)$	$\lvert u(x)-u_{\text{app}}(x)\rvert$	$\lvert v(x)-v_{\text{app}}(x)\rvert$	$\lvert w(x)-w_{\text{app}}(x)\rvert$
0	0	2	2	−3						
8	0.1	−0.46997	1.94053	−1.2447	−0.48705	1.92898	−1.24818	0.017078	0.011553	0.003474
16	0.2	−3.20739	1.69812	0.451996	−3.26655	1.66247	0.453554	0.059165	0.03565	0.001558
24	0.3	−6.4099	1.1746	2.15833	−6.54554	1.0943	2.17349	0.135639	0.080303	0.015158
32	0.4	−10.3234	0.230441	3.94258	−10.5845	0.072778	3.98065	0.261089	0.157663	0.038073
40	0.5	−15.2621	−1.33113	5.8758	−15.7195	−1.61707	5.94757	0.457359	0.285933	0.071772
48	0.6	−21.6349	−3.78579	8.03466	−22.3914	−4.27789	8.15319	0.756507	0.492101	0.118523
56	0.7	−29.9801	−7.51763	10.5045	−31.1851	−8.33348	10.686	1.205	0.815851	0.181523
64	0.8	−41.0117	−13.0594	13.3826	−42.8813	−14.3745	13.6477	1.86965	1.31511	0.265073
72	0.9	−55.6811	−21.1471	16.7823	−58.5271	−23.221	17.1571	2.84601	2.07393	0.374821
80	1	−75.26	−32.7942	20.8371	−79.5301	−36.0078	21.3552	4.27015	3.21361	0.51808

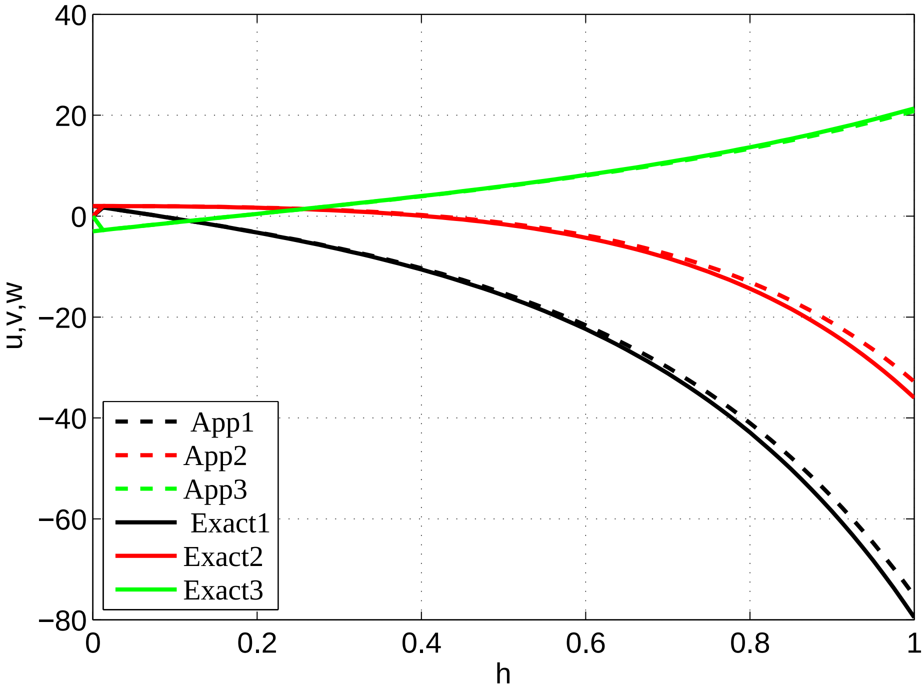

Figure 4. Graph of approximated solution and exact solution by using Explicit Euler method with $h = 0.0125$

Table 5. Explicit Euler method with h=0.00625 for n=160

i	h	$u_{\text{app}}(x)$	$v_{\text{app}}(x)$	$w_{\text{app}}(x)$	$u(x)$	$v(x)$	$w(x)$	$\left\|u(x)-u_{\text{app}}(x)\right\|$	$\left\|v(x)-v_{\text{app}}(x)\right\|$	$\left\|w(x)-w_{\text{app}}(x)\right\|$
0	0	2	2	−3						
16	0.1	−0.478261	1.93488	−1.24653	−0.487051	1.92898	−1.24818	0.00878967	0.00589864	0.0016486
32	0.2	−3.2363	1.68069	0.452598	−3.26655	1.66247	0.453554	0.0302534	0.0182149	0.000955953
48	0.3	−6.47632	1.13536	2.16562	−6.54554	1.0943	2.17349	0.069219	0.041064	0.00786322
64	0.4	−10.4513	0.153474	3.96118	−10.5845	0.0727779	3.98065	0.133188	0.0806957	0.0194665
80	0.5	−15.4861	−1.47058	5.91105	−15.7195	−1.61707	5.94757	0.233399	0.146485	0.0365229
96	0.6	−22.0051	−4.02554	8.09299	−22.3914	−4.27789	8.15319	0.386359	0.252353	0.0601948
112	0.7	−30.5691	−7.91468	10.5939	−31.1851	−8.33348	10.686	0.616025	0.418796	0.0921156
128	0.8	−41.9244	−13.6987	13.5132	−42.8813	−14.3745	13.6477	0.956895	0.675778	0.134482
144	0.9	−57.0688	−22.1542	16.967	−58.5271	−23.221	17.1571	1.45837	1.06683	0.190177
160	1	−77.3393	−34.353	21.0922	−79.5301	−36.0078	21.3552	2.19089	1.65485	0.262934

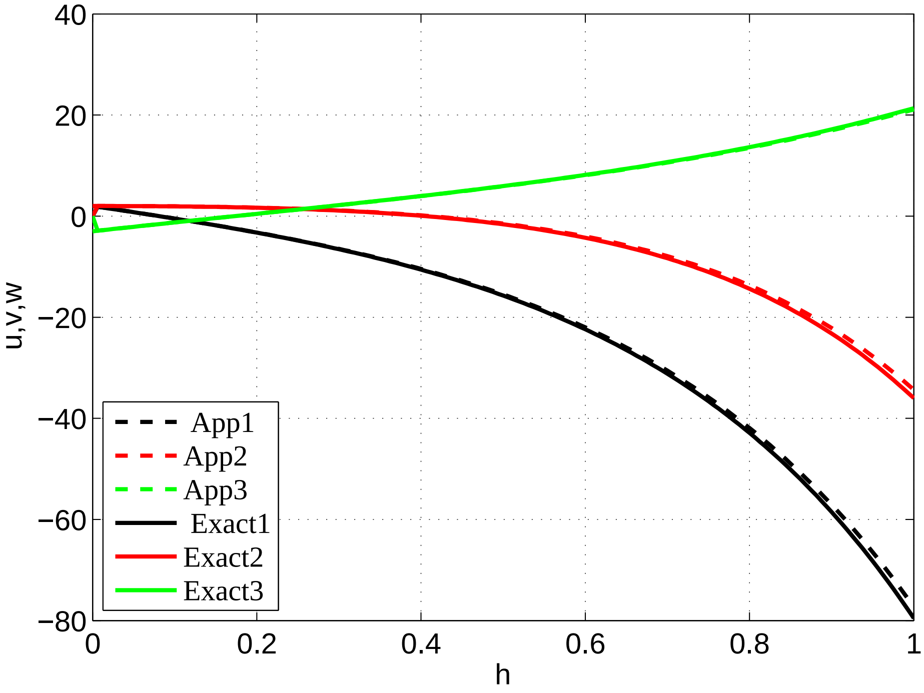

Figure 5. Graph of approximated solution and exact solution by using Explicit Euler method with $h = 0.00625$

Table 6. Explicit Euler method with h=0.003125 for n=320

i	h	$u_{\text{app}}(x)$	$v_{\text{app}}(x)$	$w_{\text{app}}(x)$	$u(x)$	$v(x)$	$w(x)$	$\left\|u(x)-u_{\text{app}}(x)\right\|$	$\left\|v(x)-v_{\text{app}}(x)\right\|$	$\left\|w(x)-w_{\text{app}}(x)\right\|$
0	0	2	2	−3						
32	0.1	−0.482592	1.93196	−1.24737	−0.487051	1.92898	−1.24818	0.00445843	0.00298071	0.000802203
64	0.2	−3.25126	1.67168	0.453031	−3.26655	1.66247	0.453554	0.0152979	0.00920782	0.000522396
96	0.3	−6.51057	1.11507	2.16948	−6.54554	1.0943	2.17349	0.0349677	0.0207671	0.00400324
128	0.4	−10.5172	0.113606	3.97081	−10.5845	0.0727779	3.98065	0.0672722	0.0408286	0.009842
160	0.5	−15.6016	−1.54291	5.92915	−15.7195	−1.61707	5.94757	0.117913	0.0741511	0.018423
192	0.6	−22.1962	−4.15008	8.12285	−22.3914	−4.27789	8.15319	0.195268	0.127806	0.0303346
224	0.7	−30.8736	−8.12127	10.6396	−31.1851	−8.33348	10.686	0.311503	0.212212	0.0464025
256	0.8	−42.3972	−14.0319	13.58	−42.8813	−14.3745	13.6477	0.484152	0.342612	0.0677365
288	0.9	−57.7888	−22.6799	17.0614	−58.5271	−23.221	17.1571	0.738337	0.541163	0.0957938
320	1	−78.4202	−35.1679	21.2227	−79.5301	−36.0078	21.3552	1.10991	0.839907	0.132461

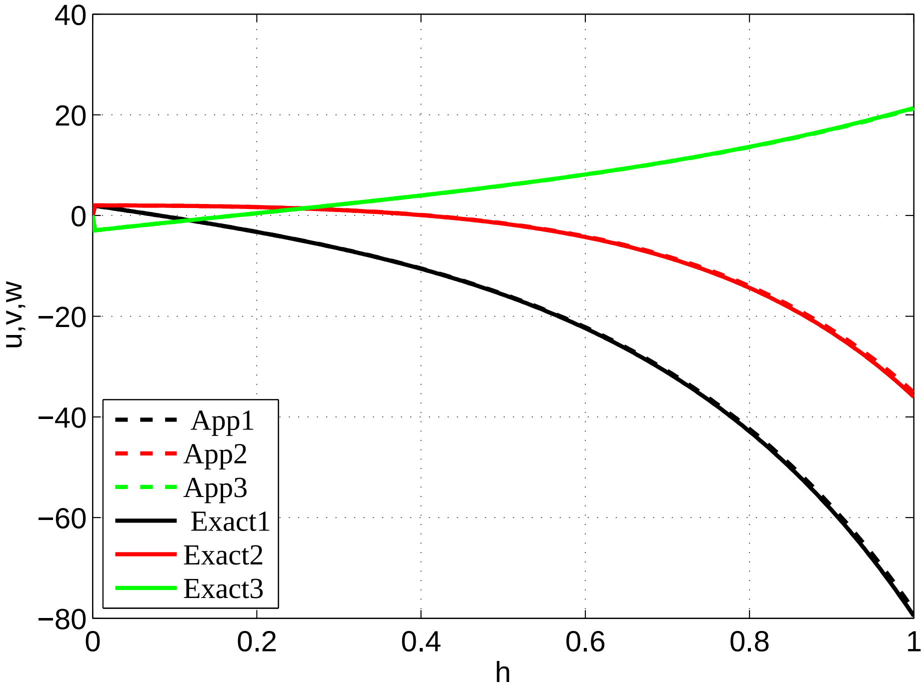

Figure 6. Graph of approximated solution and exact solution by using Explicit Euler method with $h = 0.003125$

Table 7. Explicit Euler method with h= 0.0015625 for n=640

i	h	$u_{\text{app}}(x)$	$v_{\text{app}}(x)$	$w_{\text{app}}(x)$	$u(x)$	$v(x)$	$w(x)$	$\left\|u(x)-u_{\text{app}}(x)\right\|$	$\left\|v(x)-v_{\text{app}}(x)\right\|$	$\left\|w(x)-w_{\text{app}}(x)\right\|$
0	0	2	2	−3						
64	0.1	−0.484805	1.93048	−1.24778	−0.487051	1.92898	−1.24818	0.00224524	0.00149832	0.000395578
128	0.2	−3.25886	1.6671	0.453281	−3.26655	1.66247	0.453554	0.00769218	0.00462937	0.000272326
192	0.3	−6.52797	1.10474	2.17147	−6.54554	1.0943	2.17349	0.0175745	0.0104432	0.00201961
256	0.4	−10.5507	0.0933143	3.9757	−10.5845	0.0727779	3.98065	0.033808	0.0205364	0.00494835
320	0.5	−15.6602	−1.57976	5.93832	−15.7195	−1.61707	5.94757	0.0592642	0.0373064	0.00925219
384	0.6	−22.2933	−4.21357	8.13796	−22.3914	−4.27789	8.15319	0.0981639	0.0643174	0.0152271
448	0.7	−31.0285	−8.22666	10.6627	−31.1851	−8.33348	10.686	0.156638	0.106822	0.0232882
512	0.8	−42.6378	−14.202	13.6137	−42.8813	−14.3745	13.6477	0.243526	0.172509	0.0339934
576	0.9	−58.1556	−22.9485	17.1091	−58.5271	−23.221	17.1571	0.371497	0.272556	0.0480751
640	1	−78.9715	−35.5847	21.2887	−79.5301	−36.0078	21.3552	0.558639	0.423136	0.0664819

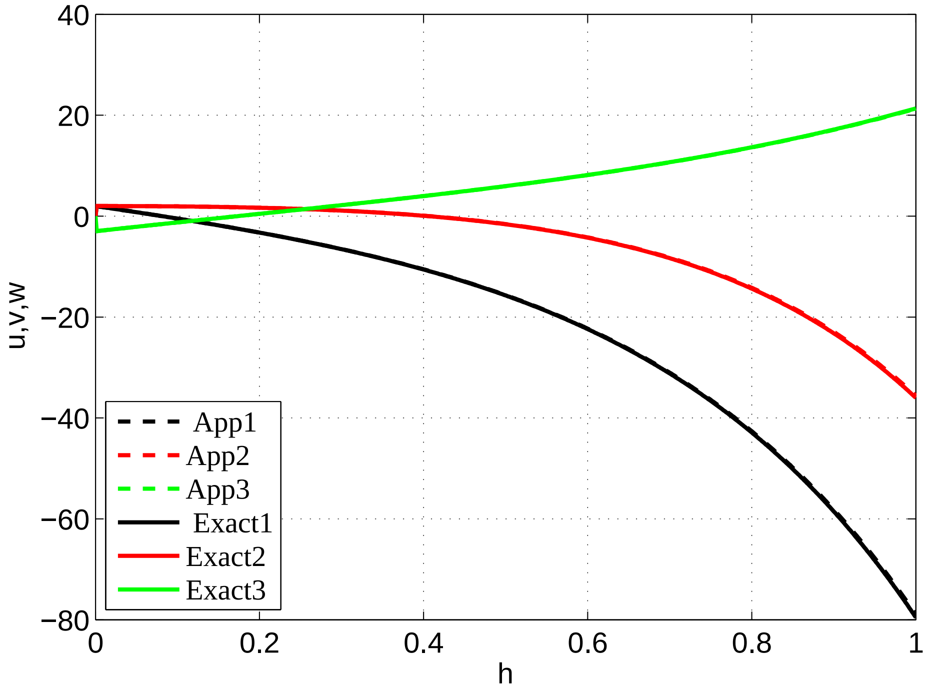

Figure 7. Graph of approximated solution and exact solution by using Explicit Euler method with $h = 0.0015625$

Appendix B. Fourth Order Runge Kutta method

.

Table 8. Fourth Order Runge Kutta method with h=0.1 for n=10

i	h	$u_{\text{app}}(x)$	$v_{\text{app}}(x)$	$w_{\text{app}}(x)$	$u(x)$	$v(x)$	$w(x)$	$\lvert u(x)-u_{\text{app}}(x)\rvert$	$\lvert v(x)-v_{\text{app}}(x)\rvert$	$\lvert w(x)-w_{\text{app}}(x)\rvert$
0	0	2	2	−3						
1	0.1	−0.48695	1.92905	−1.2482	−0.48705	1.929	−1.24818	0.000101	$6.87E-05$	$2.38E-05$
2	0.2	−3.2663	1.662663	0.453508	−3.26656	1.6625	0.453554	0.000255	0.00019	$4.56E-05$
3	0.3	−6.54505	1.094692	2.173418	−6.54554	1.0943	2.173487	0.000497	0.000392	$6.82E-05$
4	0.4	−10.5836	0.073496	3.980555	−10.5845	0.072778	3.980649	0.000872	0.000718	$9.43E-05$
5	0.5	−15.718	−1.61584	5.947442	−15.7195	−1.6171	5.947569	0.00145	0.001231	0.000127
6	0.6	−22.3891	−4.27587	8.153016	−22.3914	−4.2779	8.153185	0.002326	0.002022	0.000169
7	0.7	−31.1815	−8.33026	10.68579	−31.1851	−8.3335	10.68602	0.003642	0.003225	0.000225
8	0.8	−42.8757	−14.3695	13.64742	−42.8813	−14.375	13.64772	0.005598	0.005031	0.000299
9	0.9	−58.5187	−23.2133	17.15675	−58.5271	−23.221	17.15715	0.008483	0.007717	0.000397
10	1	−79.5174	−35.9961	21.35463	−79.5301	−36.008	21.35516	0.012707	0.01168	0.000526

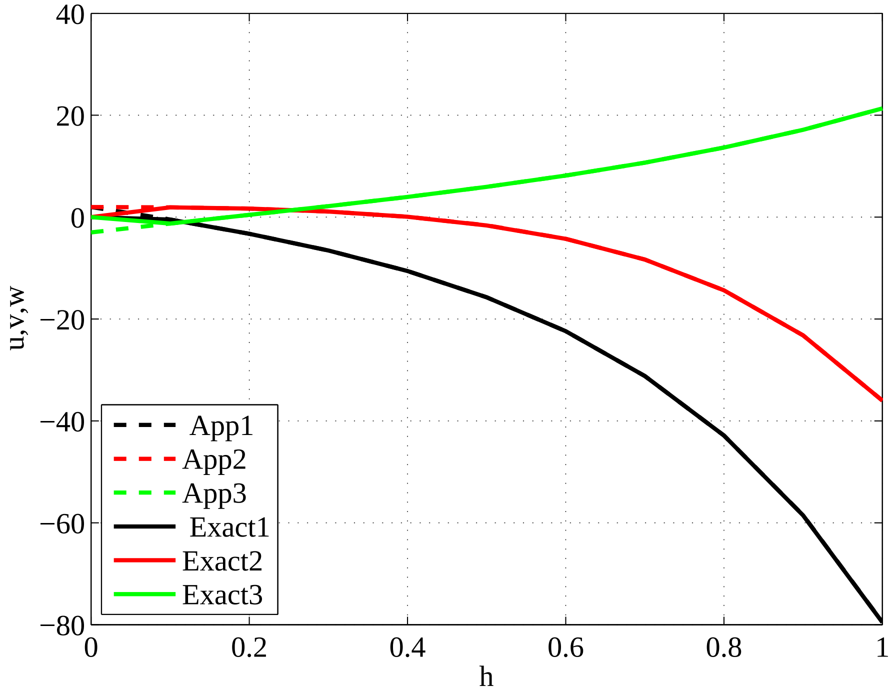

Figure 8. Graph of approximated solution and exact solution by using Explicit RK4 method with $h = 0.1$

Table 9. Fourth Order Runge Kutta method with h=0.05 for n=20

i	h	$u_{\text{app}}(x)$	$v_{\text{app}}(x)$	$w_{\text{app}}(x)$	$u(x)$	$v(x)$	$w(x)$	$\left\|u(x)-u_{\text{app}}(x)\right\|$	$\left\|v(x)-v_{\text{app}}(x)\right\|$	$\left\|w(x)-w_{\text{app}}(x)\right\|$
0	0	2	2	−3						
2	0.1	−0.4870438	1.928986	−1.248178	−0.4870507	1.929	−1.248176	$6.92E-06$	$4.91E-06$	$1.45E-06$
4	0.2	−3.266537	1.662487	0.453551	−3.266555	1.6625	0.4535538	$1.77E-05$	$1.35E-05$	$2.83E-06$
6	0.3	−6.545508	1.094328	2.173482	−6.545543	1.0943	2.173487	$3.48E-05$	$2.79E-05$	$4.30E-06$
8	0.4	−10.58443	0.07282903	3.980643	−10.58449	0.072778	3.980649	$6.13E-05$	$5.12E-05$	$6.05E-06$
10	0.5	−15.71938	−1.616978	5.947561	−15.71948	−1.6171	5.947569	0.000102135	$8.76E-05$	$8.25E-06$
12	0.6	−22.39126	−4.277744	8.153174	−22.39142	−4.2779	8.153185	0.000164166	0.000143872	$1.11E-05$
14	0.7	−31.18484	−8.333251	10.686	−31.1851	−8.3335	10.68602	0.000257318	0.000229322	$1.49E-05$
16	0.8	−42.88093	−14.37415	13.6477	−42.88133	−14.375	13.64772	0.000395824	0.000357608	$2.00E-05$
18	0.9	−58.52653	−23.22049	17.15712	−58.52713	−23.221	17.15715	0.000600045	0.000548329	$2.67E-05$
20	1	−79.52924	−36.00698	21.35512	−79.53014	−36.008	21.35516	0.000899018	0.000829558	$3.55E-05$

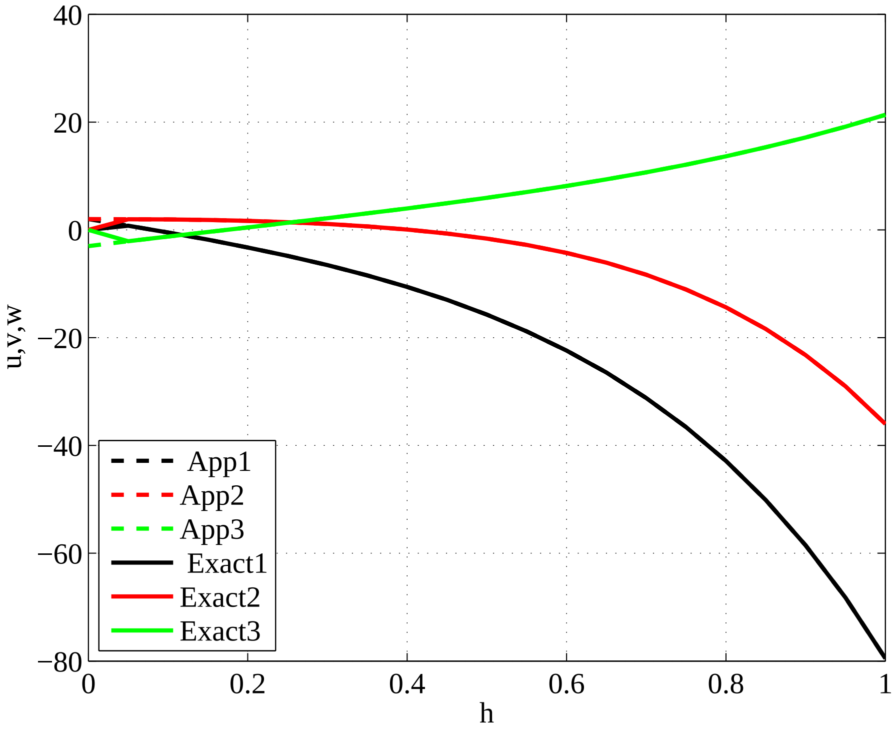

Figure 9. Graph of approximated solution and exact solution by using Explicit RK4 method with $h = 0.05$

Table 10. Fourth Order Runge Kutta method with h=0.025 for n=40

i	h	$u_{\text{app}}(x)$	$v_{\text{app}}(x)$	$w_{\text{app}}(x)$	$u(x)$	$v(x)$	$w(x)$	$\lvert u(x)-u_{\text{app}}(x)\rvert$	$\lvert v(x)-v_{\text{app}}(x)\rvert$	$\lvert w(x)-w_{\text{app}}(x)\rvert$
0	0	2	2	−3						
4	0.1	−0.4871	1.92898	−1.2482	−0.4871	1.929	−1.2482	$4.55E-07$	$3.28E-07$	$9.00E-08$
8	0.2	−3.2666	1.66247	0.45355	−3.2666	1.6625	0.45355	$1.17E-06$	$9.04E-07$	$1.77E-07$
12	0.3	−6.5455	1.0943	2.17349	−6.5455	1.0943	2.17349	$2.30E-06$	$1.87E-06$	$2.71E-07$
16	0.4	−10.584	0.07278	3.98065	−10.584	0.07278	3.98065	$4.06E-06$	$3.41E-06$	$3.84E-07$
20	0.5	−15.719	−1.6171	5.94757	−15.719	−1.6171	5.94757	$6.78E-06$	$5.85E-06$	$5.27E-07$
24	0.6	−22.391	−4.2779	8.15319	−22.391	−4.2779	8.15319	$1.09E-05$	$9.60E-06$	$7.15E-07$
28	0.7	−31.185	−8.3335	10.686	−31.185	−8.3335	10.686	$1.71E-05$	$1.53E-05$	$9.63E-07$
32	0.8	−42.881	−14.374	13.6477	−42.881	−14.375	13.6477	$2.63E-05$	$2.38E-05$	$1.29E-06$
36	0.9	−58.527	−23.221	17.1572	−58.527	−23.221	17.1572	$3.99E-05$	$3.65E-05$	$1.73E-06$
40	1	−79.53	−36.008	21.3552	−79.53	−36.008	21.3552	$5.98E-05$	$5.53E-05$	$2.30E-06$

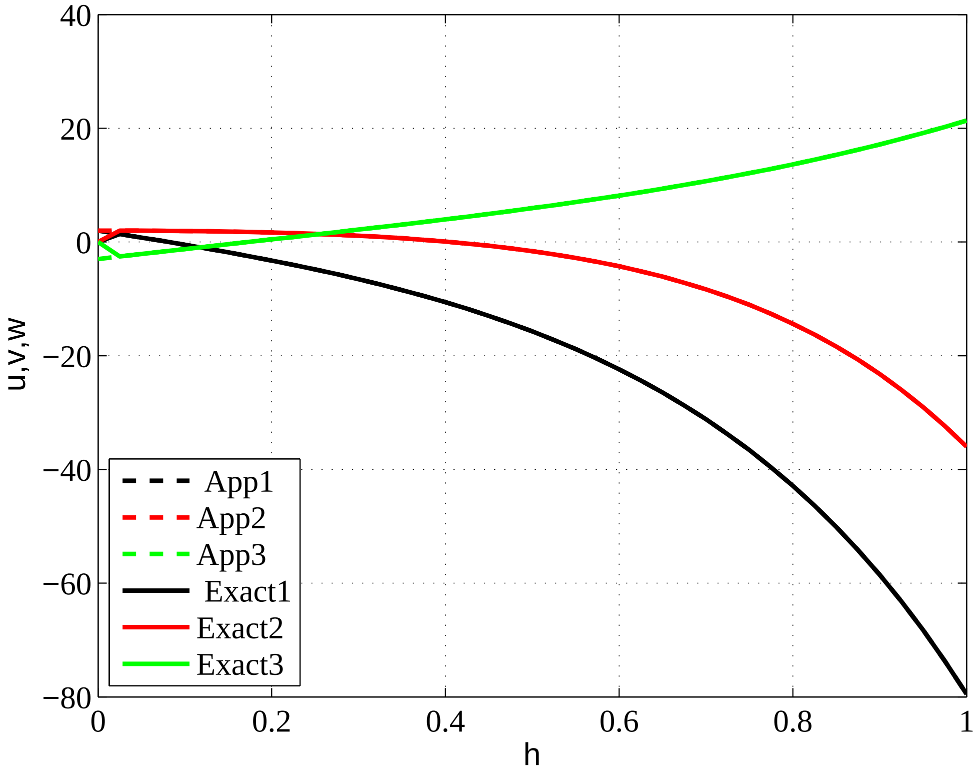

Figure 10. Graph of approximated solution and exact solution by using Explicit RK4 method with $h = 0.025$

Table 11. Fourth Order Runge Kutta method with h=0.0125 for n=80

i	h	$u_{\text{app}}(x)$	$v_{\text{app}}(x)$	$w_{\text{app}}(x)$	$u(x)$	$v(x)$	$w(x)$	$\lvert u(x)-u_{\text{app}}(x)\rvert$	$\lvert v(x)-v_{\text{app}}(x)\rvert$	$\lvert w(x)-w_{\text{app}}(x)\rvert$
0	0	2	2	−3						
8	0.1	−0.4870507	1.928981	−1.248176	−0.4870507	1.929	−1.248176	$2.91E-08$	$2.12E-08$	$5.60E-09$
16	0.2	−3.266555	1.662473	0.4535538	−3.266555	1.6625	0.4535538	$7.52E-08$	$5.84E-08$	$1.11E-08$
24	0.3	−6.545542	1.0943	2.173487	−6.545543	1.0943	2.173487	$1.48E-07$	$1.21E-07$	$1.70E-08$
32	0.4	−10.58449	0.0727781	3.980649	−10.58449	0.072778	3.980649	$2.62E-07$	$2.20E-07$	$2.42E-08$
40	0.5	−15.71948	−1.617065	5.947569	−15.71948	−1.6171	5.947569	$4.37E-07$	$3.78E-07$	$3.33E-08$
48	0.6	−22.39142	−4.277888	8.153185	−22.39142	−4.2779	8.153185	$7.03E-07$	$6.20E-07$	$4.53E-08$
56	0.7	−31.1851	−8.333479	10.68602	−31.1851	−8.3335	10.68602	$1.10E-06$	$9.87E-07$	$6.12E-08$
64	0.8	−42.88132	−14.37451	13.64772	−42.88133	−14.375	13.64772	$1.70E-06$	$1.54E-06$	$8.22E-08$
72	0.9	−58.52713	−23.22104	17.15715	−58.52713	−23.221	17.15715	$2.57E-06$	$2.36E-06$	$1.10E-07$
80	1	−79.53013	−36.00781	21.35516	−79.53014	−36.008	21.35516	$3.86E-06$	$3.57E-06$	$1.47E-07$

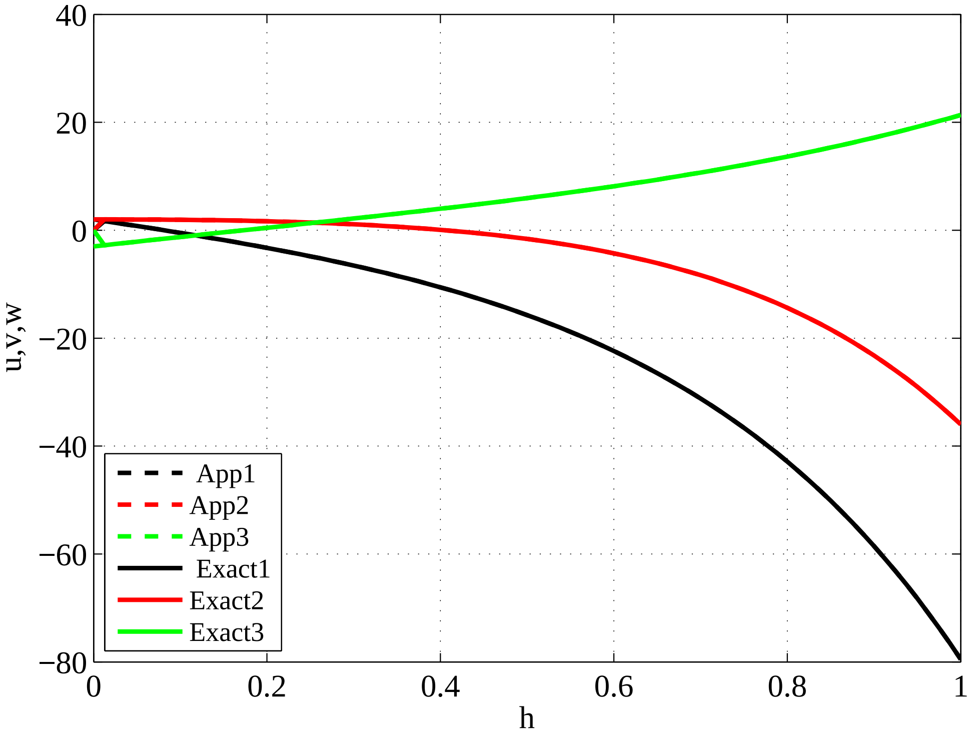

Figure 11. Graph of approximated solution and exact solution by using Explicit RK4 method with $h = 0.0125$

Table 12. Fourth Order Runge Kutta method with h=0.00625 for n=160

i	h	$u_{\text{app}}(x)$	$v_{\text{app}}(x)$	$w_{\text{app}}(x)$	$u(x)$	$v(x)$	$w(x)$	$\lvert u(x)-u_{\text{app}}(x)\rvert$	$\lvert v(x)-v_{\text{app}}(x)\rvert$	$\lvert w(x)-w_{\text{app}}(x)\rvert$
0	0	2	2	−3						
16	0.1	−0.4870507	1.928981	−1.248176	−0.4870507	1.929	−1.248176	$1.84E-09$	$1.35E-09$	$3.50E-10$
32	0.2	−3.266555	1.662473	0.4535538	−3.266555	1.6625	0.4535538	$4.77E-09$	$3.71E-09$	$6.91E-10$
48	0.3	−6.545543	1.0943	2.173487	−6.545543	1.0943	2.173487	$9.38E-09$	$7.66E-09$	$1.07E-09$
64	0.4	−10.58449	0.07277789	3.980649	−10.58449	0.072778	3.980649	$1.66E-08$	$1.40E-08$	$1.52E-09$
80	0.5	−15.71948	−1.617065	5.947569	−15.71948	−1.6171	5.947569	$2.77E-08$	$2.40E-08$	$2.10E-09$
96	0.6	−22.39142	−4.277888	8.153185	−22.39142	−4.2779	8.153185	$4.46E-08$	$3.94E-08$	$2.85E-09$
112	0.7	−31.1851	−8.33348	10.68602	−31.1851	−8.3335	10.68602	$7.00E-08$	$6.27E-08$	$3.86E-09$
128	0.8	−42.88133	−14.37451	13.64772	−42.88133	−14.375	13.64772	$1.08E-07$	$9.77E-08$	$5.19E-09$
144	0.9	−58.52713	−23.22104	17.15715	−58.52713	−23.221	17.15715	$1.63E-07$	$1.50E-07$	$6.94E-09$
160	1	−79.53014	−36.00781	21.35516	−79.53014	−36.008	21.35516	$2.45E-07$	$2.27E-07$	$9.26E-09$

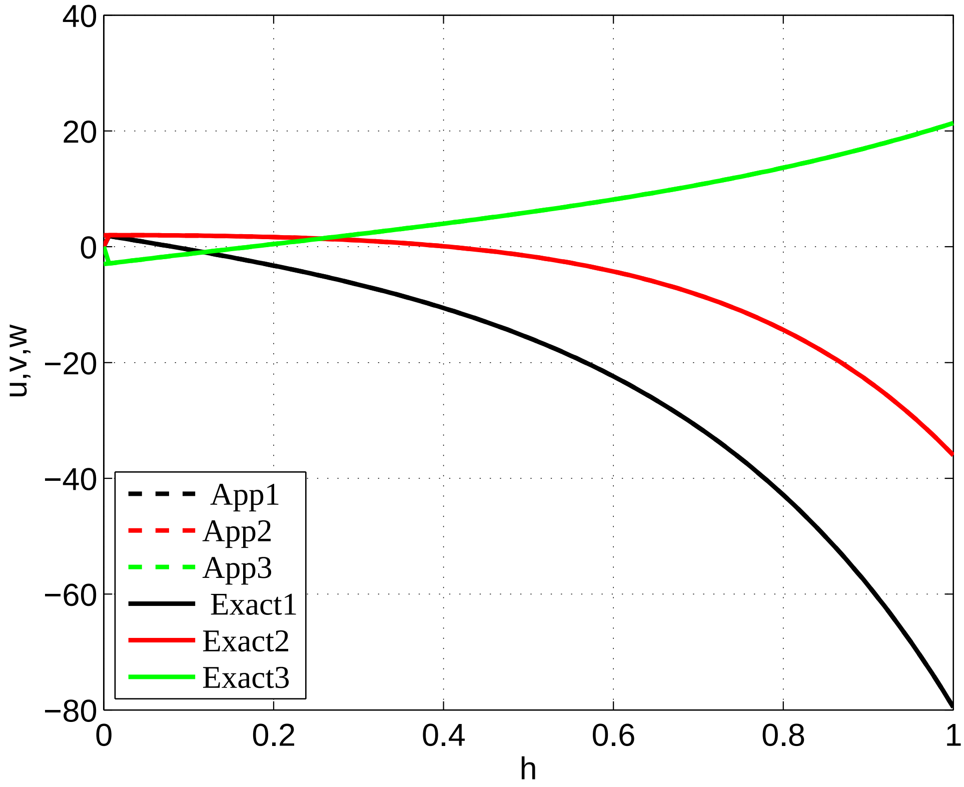

Figure 12. Graph of approximated solution and exact solution by using Explicit RK4 method with $h = 0.00625$