KWAME NKRUMAH UNIVERSITY OF SCIENCE AND TECHNOLOGY, KUMASI

COLLEGE OF SCIENCE

DEPARTMENT OF MATHEMATICS

AN ANALYSIS OF RUNGE-KUTTA METHOD IN NON-NEWTONIAN CALCULUS

BY

ZAKARIA ADNAN

A THESIS SUBMITTED TO THE DEPARTMENT OF MATHEMATICS, KWAME NKRUMAH UNIVERSITY OF SCIENCE AND TECHNOLOGY, KUMASI, IN PARTIAL FULFILMENT OF THE REQUIREMENT FOR THE AWARDS OF MASTER OF SCIENCE DEGREE IN INDUSTRIAL MATHEMATICS

JUNE, 2015

DECLARATION

I hereby declare that this submission is my own work towards Master of Science degree and that to the best of my knowledge, it contains no material previously published by another person nor materials which have been accepted for the award of any other degree of any university, except for references cited from extracts, scripts, text books, journals, papers and other sources which have been duly acknowledged.

ZAKARIA ADNAN ……………………………… ……………………….

(PG8146712) Signature Date

I declare that I have supervised the candidate's research work and I confirmed that the student has my permission to present it for assessment.

PROF. I.K DONTWI …………………….. ………………………….

(Supervisor) Signature Date

Certified by:

PROF. S. AMPONSAH ……………………….. …………………………..

(Head of Department) Signature Date

DEDICATION

This project is dedicated to Almighty Allah, my parents and family members.

i

ACKNOWLEGEMENT

One cannot complete a research of this nature without certain amount of guidance, directions and support from others.

Sincere thanks first of all goes to the Almight Allah for making this thesis a success.

My sincere gratitude also goes to my supervisor, Prof. I.K Dontwi for his patience, support and assistance on this project. I feel especially blessed to have had such a wonderful supervisor to help see this thesis to fruition. Thank you!

Special thanks and gratitude are also extended to all the lectures of the Mathematics department, KNUST for impacting in me further and advance knowledge of Mathematics and its applications.

Many thanks to my parents and all other family members especially Alhaji Zakaria Yakubu, Alhaji Karimu Yakubu, Hajia Sirina Yakubu , Menuna Bello for their immense support in my educational career. Allah blesses you all.

Finally, I am most grateful to Mr. Zakaria Amjad, my brother whose immense contribution and support helped to the final outcome of this project work.

ABSTRACT

Non-Newtonian Calculus or Multiplicative calculus can be used as a tool wherever a problem is of exponential (relational) in nature. In this thesis the derivation of the Runge-Kutta Method in the framework of non-Newtonian (Multiplicative) Calculus is presented. The non-Newtonian (Multiplicative) Runge-Kutta Methods of orders 2, 3, and 4 are developed and discussed. The non-Newtonian (Multiplicative) Runge-Kutta Method is tested on some selected examples where the solutions of the ordinary differential equations are known using developed Matlab codes. The results show that for certain family of initial value problem the non-Newtonian (Multiplicative) Runge-Kutta method gives better results to the Ordinary Runge-Kutta method.

TABLE OF CONTENTS

Content
Page

LIST OF TABLES

LIST OF FIGURES

CHAPTER ONE

INTRODUCTION

1.1 Background of the study

Isaac Newton and Gottfried Wilhelm Leibnitz in the second half of the 17th century independently laid the foundations of differential and integral calculus, the most applicable mathematical theory. Differentiation and integration - these two operations are the basic in calculus and analysis. Sometimes this calculus is also referred as Newtonian calculus. The basic operations of Newtonian calculus (differentiation and integration) are the infinitesimal versions of the arithmetic operation of subtraction and addition.

Michael Grossman and Robert Katz in the period from 1967 to 1970 indicated in their work [1] that infinitely many calculi can be generated independently. Bigeometric calculus [2] which was later on introduced by Grossman which gave definitions of a new kind of derivative and integral, changing from the roles of subtraction and addition to division and multiplication, and thus established a new calculus, called non-Newtonian calculus(Multiplicative calculus). Bashirov et al in [3] gave the theoretical background of non-Newtonian (Multiplicative) Calculus. NonNewtonian (Multiplicative) Calculus is based on division and multiplication. It is, sometimes called exponential or alternative calculus as well.

Definition of the derivative of non-Newtonian (Multiplicative) Calculus of function $f(x)$ with

respect to x is as follows: $\frac{}{} = \lim_{\to} \frac{}{}$ and with this definition, a non-

Newtonian (Multiplicative) Runge Kutta Method on the complete theory of [3] will be presented, taking into consideration the Multiplicative Runge-Kutta Method formulated by Aniszewska in [4].

1.2 Problem Statement

In numerical analysis, the Runge-Kutta methods are important family of implicit and explicit iterative methods for the approximation of solutions of ordinary differential equations. In the areas of biology, finance, actuarial science, demography, economics etc. there exist numerous mathematical models base on differential equations which are quite hard to solve using standard solution methods for ordinary differential equations. Generally in Science and Engineering the 4^{th} order Runge-Kutta method is preferred because it gives the most accurate approximation to initial value problems with a reasonable computational effort. Since the solutions of many of these problems are exponential or multiplicative in nature and not additive, the solutions of these problems can be accomplished by using Non-Newtonian (Multiplicative) differential equations. Therefore the Non-Newtonian (Multiplicative) calculus builds the proper framework for the solutions of these problems and it is self-evident to use it also for numerical approximations.

1.3 Research Objectives

The main objectives of this research is to:

1. Develop a Non-Newtonian (Multiplicative) Runge-Kutta Methods for the orders 2, 3 and 4.
2. Compare the Non-Newtonian(Multiplicative) Runge-Kutta Methods for the orders 2, 3, and 4 and the corresponding order of ordinary Runge-Kutta Methods.

1.4 Justification

This research is intended to serve the following purpose:

1. It will serve as a partial fulfillment of the requirement for the Master of Science Degree in **Industrial Mathematics** at the University of Science and Technology (KNUST), Kumasi.
2. It will help further research in this new kind of calculus, since Non-Newtonian calculus is still in its infancy.

1.5 Structure of the thesis

The rest of the thesis includes four chapters. In chapter two, definitions, basic concepts and theorems of non-Newtonian (Multiplicative) Calculus necessary for the understanding of the consequent chapters are explained. In chapter three, the derivations of the Ordinary Runge-Kutta methods of 2nd, 3rd and 4th orders are explicitly discussed using the basic ideas formulated by Carl David Tolme Runge and Martin Wilhelm Kutta(Runge and Kutta). With the basic knowledge in chapter two and the derivations of the Ordinary Runge-Kutta methods, nonNewtonian (Multiplicative) Runge-Kutta methods of 2nd, 3rd and 4th orders are derived. In the next chapter (chapter four) examples will be solved using Ordinary Runge-Kutta methods and non-Newtonian (Multiplicative) Runge-Kutta methods and the results in both cases analyzed and compared. Matlab codes for Ordinary and non-Newtonian (Multiplicative) Runge-Kutta methods would also be formulated. The concluding chapter (chapter five) takes into consideration the results obtained in chapter four and summaries the important findings in that chapter and also provides general conclusions regarding non-Newtonian (Multiplicative) Runge-Kutta method.

CHAPTER TWO

REVIEW OF SOME BASIC DEFINITIONS AND CONCEPTS

2.1. Non-Newtonian (Multiplicative) calculus derivatives

The Non-Newtonian (Multiplicative) derivative is defined by

$$^{*} = \lim_{h\to 0}\left(\frac{f(x+h)}{f(x)}\right)^{\frac{1}{h}} \text{, where } \neq 0 \qquad (2.1)$$

Comparing the non-Newtonian (Multiplicative) derivative (2.1) with the definition of the ordinary derivative

$$= \lim_{h\to 0}\frac{f(x+h)-f(x)}{h} \qquad (2.2)$$

it can be observe that the ordinary difference $f(x+h)-f(x)$, in the ordinary derivative, is replaced by the ratio $\frac{f(x+h)}{f(x)}$ (exponential difference) in the non-Newtonian (Multiplicative) derivative and the division by h is replaced by the exponential ratio (i.e. the reciprocal power $\frac{1}{h}$). Thus the # non-Newtonian (Multiplicative) derivative tells us how many times increases at the instant or how many times changes at . It differs from the derivative , which tells us the rate of change of f at x.

The limit, i.e. $\lim_{h\to 0}\left(\frac{f(x+h)}{f(x)}\right)^{\frac{1}{h}}$ in (2.1) is called the non-Newtonian (Multiplicative) derivative

or, *derivative of f at x and it can be denoted by either $f^*(x)$ or $\frac{d^*f}{dx}$. If $f^*(x)$ exists for all $x \in A$

from some open set $A \subseteq \mathbb{R}$ (where $\mathbb{R}$ denotes real line) then the function $f^*: A \to \mathbb{R}$ is well-defined

[3]. The function f^* is called the *derivative of $f: A \to \mathbb{R}$ The *derivative of f^* is

called the second *derivative of $f(x)$ and it is denoted by f^{**} or $\frac{d^{**}f}{dx}$. In the same vein, the

nth *derivative of $f(x)$ can also be defined, which is denoted by $f^{*(n)}$ or $\frac{d^{*(n)}f}{dx}$ for $n = 0, 1, 2 \ldots$

where $f^{*(0)} = f$.

If $f(x)$ is a positive function on A and its derivative at x exists, then we can calculate

$$f^* = \lim_{h\to 0}\left(\frac{f(x+h)}{f(x)}\right)^{\frac{1}{h}}$$

$$= \lim_{h\to 0}\left(\frac{f(x+h)-f(x)}{f(x)}+1\right)^{\frac{1}{h}}$$

$$= \lim_{h\to 0}\left(1+\frac{f(x+h)-f(x)}{f(x)}\right)^{\frac{f(x)}{f(x+h)-f(x)}\times\frac{f(x+h)-f(x)}{h}\times\frac{1}{f(x)}}$$

$$= \lim_{h\to 0}\left(\left(1+\frac{f(x+h)-f(x)}{f(x)}\right)^{\frac{f(x)}{f(x+h)-f(x)}}\right)^{\frac{f(x+h)-f(x)}{h}\times\frac{1}{f(x)}}$$

01

= ___0 = 2* 1

* = 2* 1 , where ln =ln .

Similarly, if the second derivative of *f (x)* exists, then ** = 2* *1 = 2* 11 .Which is easily obtained by substitution [3], [19].

If ≠0 and exists, then ln ʺ exists and repeating the procedure *n* times, for being positive function and its *nth* derivatives at *x* exists, then * * also exists and

* * = 2* + , for ,=0,1,2… (2.3)

Note that in (2.3), for ,=0 is included as well because

= 56, that is * = 2* 7 = .

Thus it can be concluded that, the function :&→ℝis *differentiable at *x* or on *A* if it is positive and differentiable at *x* or on *A* respectively [3].

Deriving a formula similar to Newton's binomial formula, * can be express in terms of * *. Using the *nth* non-Newtonian (Multiplicative) derivative of *f*, we have

ln * * = ln * = ln 8 * 8 = ln * 8 * 8 .

By using

= ln * ,

The second derivative in terms of non-Newtonian (Multiplicative) derivative for ʺ can be calculated as,

= ln * + ln ** ,

and the third derivative in terms of non-Newtonian (Multiplicative) derivative for ‴ using the second derivative can also be calculated as,

‴ = ln * +2 ln ** + ln *** .

Repeating this procedure *n* times, a formula for the *nth* derivative can be formulated (or derived) as follows;

$$^{(n)}(x) = \sum_{k=0}^{n-1} \frac{(n-1)!}{k!(n-k-1)!} f^{(k)}(x)(\ln f^{*(n-k)})(x) .$$

For the constant function =:>0 on the interval *(a, b)*, where $a < b$, we have

$f^*(x) = e^{(\ln c)'} = e^0 = 1$, ∈=,>.

Conversely, if * =1 for every ∈=,> then

$$^* = \frac{f'(x)}{f(x)} = e^{(lnf)'(x)} = 1, \text{ implies that } =:\neq 0.$$

It can be observed easily that =:?,@A.>0 for ∈=,>. Thus, a function is a positive constant on an open interval if and only if its *derivative on this interval is identically equal 1. It can also be noted that, a neutral element 0 appears for additive derivative and for multiplicative derivative, a neutral element 1 appears [3].

The following are some properties (or rules) of the *differentiation which can be easily derived / proved:

* = * # . #1 Power Rule (2.7)

?h^* = $^*h^{\#1}$ Chain Rule (2.8)

+B* $= f^*(x)^{\frac{f(x)}{f(x)+g(x)}} . g^*(x)^{\frac{g(x)}{f(x)+g(x)}}$ Sum Rule (2.9)

where :>0 is a constant, and B are *differentiable, *h* is differentiable. Derivation of equations (2.4) to (2.9) are as follows:

Constant Multiple Rule:

: * = 56 H 1 = IK0J×MH 1NO

: * = * Constant Multiple Rule (2.4)

B* = *B* Product Rule (2.5)

B⁄ * = */B* Quotient Rule (2.6)

$$= e^{\frac{f'(x)}{f(x)}} = e^{lnf^*(x)} = f^* .$$

Product Rule:

B* = M2* P N1 = 561 56P 1

= 561 × 56P 1 = * B* .

Quotient Rule:

B⁄ * = 56 G0 ——)) 1 = I G___ 0 Q 01 G G G R 1 0___ SO

$= Q \quad \frac{f'(x)}{} \frac{g'(x)}{} \quad 0 \quad G \quad S = e^{(lnf^*(x) - lng^*(x))} = e^{\ln(\frac{f^*(x)}{g^*(x)})}$

$= \frac{f^*(x)}{^*(x)} = f^*$ /B* P .

Power Rule:

1 J T MU T V+0

* = 56 T = 0 N

J T Q#1 01

= 0 T $f(x)$ 2* # 0 S

#1 2* # 01 T1 * T ——)

= #1 × * # = * # . #1 . =0 = 2* × 2*

Sum Rule:

Chain Rule:

J 1

1 J 01T ×#1 P1

?h^* = 56 W# = 0MT N 1M# N#1 = 0 T +B*

= 56 P

1 = 0 FG

= #1 2* *# = 2* *#T1 = *h#1 .

01 G1 0 2* (x) * G 2*P*

$$= e^{(\frac{(\ )}{f(x)+g(x)} + \frac{(\ )}{f(x)+g(x)})} = e^{(\frac{(\ )}{f(x)+g(x)}} \times \frac{(x)}{f(x)+g(x)}$$

$$= 2^{*} {}_{*} \frac{0}{0 \quad FG} \times 2^{*}P_{*} \frac{G}{0 \quad FG}$$

$$= f^{*}(x)^{\frac{(\ )}{f(x)+g(x)}} \ {}^{*}(x)^{\frac{g(x)}{f(x)+g(x)}} = f^{*}(x)^{\frac{f(x)}{f(x)+g(x)}} . g^{*}(x)^{\frac{g(x)}{f(x)+g(x)}} \times B.$$

The first order non-Newtonian (Multiplicative) differential equation which contains the *derivative of y is in the form

$$^{*} \quad = , \qquad (2.10)$$ and

its equivalent to the ordinary differential equation is

$$= \quad \ln , \quad . \qquad (2.11)$$

Theorem 1 [3](Multiplicative Mean Value Theorem). If the function is continuous on the close interval [=,>] and has a *derivative at every point of the open interval =,> , then there exists =<:<> such that

$$_{*}: = \left(\frac{f(b)}{f(a)} \right)^{\frac{1}{b-a}} \quad \text{or} \quad \frac{f(b)}{f(a)} = {}_{*}:[\quad \backslash$$

Theorem 2 [3](Multiplicative Taylor's Theorem for One Variable). Let & be an open interval and let :&→ℝ be ,+1 times *differentiable on &. Then for any , $+h$∈&, there exists a number]∈ 0,1 such that

$$+h = \prod_{m=0}^{n} \left(f^{*(m)}(x) \right)^{\frac{h^{m}}{m!}} . \left(f^{*(n+1)}(x+\theta h) \right)^{\frac{h^{n+1}}{(n+1)!}}.$$

Consider the function , of two variables defined on some open subset of ℝ×ℝ =ℝ.

The partial *derivative of can be defined, considering fixed, and it is denoted by * or ^

__* . In a similar way, the partial *derivative of in can be defined, which is denoted by _* ^

or __. Higher order partial *derivatives of can also be defined. That is * or ^____* R , _* or

^*

^_ ^ ^

^* R

____^ ^_ , __* or ^____^_^_* R .

Theorem 3 [3](Multiplicative Chain Rule). Let be a function of two variables and ` with continuous partial *derivatives. If and ` are differentiable functions on =,> such that

,` is defined for every ∈=,>, then

$$\frac{d^* f(y(x), z(x))}{dx} = f_y^*(y(x), z(x))^{y'(x)} \, f_z^*(y(x), z(x))^{z'(x)}.$$

Theorem 4 [3](Multiplicative Taylor's Theorem for Two Variable). Let & be an open subset of ℝ. Assume that the function :&→ℝhas all partial *derivatives of order ,+1 on &. Then for every , , +*h*, +a ∈& such that the line segment connecting these two points belongs to &, there exists a number]∈0,1 , such that

$$f(x+h, y+k) = \prod_{m=0}^{n}\prod_{i=0}^{m} f^{*(m)}_{x^i y^{m-i}}(x,y)^{\frac{h^i k^{m-i}}{i!(m-i)!}} \prod_{i=0}^{n+1} f^{*(n+1)}_{x^i y^{n+1-i}}(x+\theta h, y+\theta k)^{\frac{h^i k^{n+1-i}}{i!(n+1-i)!}}.$$

CHAPTER THREE

METHODOLOGY

3.1 Ordinary Runge-Kutta Methods

These techniques were designed around 1900 by the German mathematicians Carl David Tolme Runge and Martin Wilhelm Kutta. The methods named after Carl Runge and Wilhelm Kutta are formulated to imitate the Taylor series method without requiring analytic differentiation of the original differential equation. In numerical analysis, the Runge-Kutta methods are family of explicit and implicit iterative methods for the approximation of solutions of ordinary differential equations. These methods are used to find the solutions of the ordinary differential equations of the form

$$= , , \qquad = , \tag{3.1}$$

where $= \frac{}{\$}$ $\$$–. Sometimes, for convenience, the in is omitted.

Equation (3.1) can be more briefly written as $' = ,$. Since , is just the slope of the desired exact solution of (3.1), one has approximately

$$\frac{\_ \# \_}{\#} \approx , \quad , \quad \text{for } h \neq 0 \tag{3.2}$$

or

$$+h \approx \qquad +h , \tag{3.3}$$

By using the initial condition $=$, the solution of the equation (3.1) thus takes the form

$$_c = _c + _c, \qquad _c h, \quad \text{for } d = 0,1,2\ldots.. \tag{3.4}$$

where $h = _c - _c$.

The Taylor series methods are known to have the desirable property of high-order local truncation error (the ability to keep the errors small), but also has the disadvantage of requiring repeated differentiation of the function , [12]. The computation and evaluation of the high derivatives of , may become very long and are, therefore, error-prone and tedious, this can be a serious obstacle to using this method. In the Taylor series method, each of these high order derivatives is evaluated at the point $_c$ in order to evaluate $_c$. Runge-Kutta methods have the high-order local truncation error of the Taylor methods but eliminate the need to compute and evaluate the derivatives of , . Basically, the aim of Runge-Kutta methods is to eliminate the need for repeated differentiation of the differential equations [15].

3.1.1 Runge-Kutta Second-Order Method

To arrive at the second-order Runge-Kutta method for the solution of the differential equation (3.1), we start with the Taylor series expansion of the form for $+h$ which is

$$+h = \quad +h + \frac{\#}{!}{}^{R} + \frac{\#}{g!}{}^{f} + \cdots + \frac{\#}{j!}{}^{i}\ {}^{j} +]h . \qquad (3.5)$$

The first step in deriving the 2nd order Runge-Kutta method is to consider the second order Taylor series formula. In applying the Taylor series we shall use the following subscripts to

denote partial derivatives, i.e., $= \hat{}\hat{}\_\_$, $_\_ = \hat{}\_\_\_\hat{}$, $= \hat{}\hat{}\_\_\_{}^{R}{}_{R}$, $_\_ = \hat{}\ \hat{}\_\_\_\_\_{}^{\hat{}R}$ and $\_\_ = \hat{}\_\_\_\_{}^{\hat{}R}{}_{R}$.

The first derivative in (3.5), i.e., can be replaced by the right-hand side of the differential equation (3.1). Thus differentiating we have

$$= , + \_ ,$$

$$= , + \_ , , \qquad (3.6)$$

since $= ,$ and taking the partial derivatives of , with respect to and .

By substituting equation (3.6) into equation (3.5), we get the second-order Taylor expansion as

$$+h = \quad +h , + \#\_\_{}^{R} , + \quad \_ , , +k\ h^{g}$$

$$= \quad +h , + \#\_\_{}^{R} , + \#\_\_{}^{R} \quad \_ , , +k\ h^{g} \qquad (3.7)$$

The Runge-Kutta method can be written as a linear combination of the function , evaluated at certain points in the step. The results of the second-order method is then evaluated in a sequence of operations of the form

+h = += ha += ha (3.8)

where

a = , (3.9)

a = +lh, +mha (3.10)

The constants = , = , l, m are determined by comparing equation (3.7) with equation (3.8). To do this we need to find the Taylor series expansion for +lh, +mha which is

a = +lh, +mha = , + , lh+ _ , mha +k h (3.11)

Now we substitute equation (3.11) into equation (3.8) to get

$$y(x+h) = y(x) + a_1 hf(x,y) + a_2 h(f(x,y) + f_x(x,y)ph + f_y(x,y)qhf(x,y)) + O(h^2)$$

$$= y(x) + (a_1 + a_2)hf(x,y) + a_2 h^2 pf_x(x,y) + a_2 h^2 qf_y(x,y)f(x,y) + O(h^3) \quad (3.12)$$

since a = , .

Comparing equations (3.7) and (3.12), we find that they are identical if

= += =1 = l=_ = m=_ (3.13)

Thus in (3.13) we have three equations in four unknown parameters, here we have more than one choice in finding the unknown parameters. Some of the popular choices and the names associated with the resulting formulas are:

= =_ = =_ l=1 m=1 Heun's method

= =0 = =1 l=_ m=_ Modified Euler's method

= =_ = =_ l= g_ m= g_ Ralston's method g g n n

Choosing the modified Euler's method, and substitute the corresponding parameters into equation (3.8) yields the second-order Runge-Kutta method

$+h =$ $+h + \#\_$ $, + \#\_ ,$

(3.14)

or, equivalently,

$$+h = \quad +a \qquad (3.15)$$

where

$$a \quad = h \, , \qquad (3.16)$$

$$a \quad = h + \#\_ \quad , +\_a \qquad (3.17)$$

which is the same as

$$+h = \quad +a \, h \qquad (3.18)$$

where

$$a = , \qquad (3.19)$$

$$a = +\_h, +\_a \, h \qquad (3.20)$$

For the Heun's method, the second-order Runge-Kutta method is

$$+h = \quad + a + a \, h \quad - \; 1 \; - \qquad (3.21)$$ where

$$a = , \qquad (3.22)$$

$$a = +h, +a \, h \qquad (3.23)$$

And the second-order Runge-Kutta method for the Ralston's method is

$$+h = \quad + a + a \, h \qquad (3.24) \qquad - \; 1 \; -$$

g g

where

$$a = , \qquad (3.25)$$

$$a = + {}^{g}\_ h, + {}^{g}\_ a \, h \qquad (3.26)$$

n n

3.1.2 Runge-Kutta Third-Order Method

The third-order Runge-Kutta method can be obtained from the Taylor series along the same lines as the second-order method. To derive the third-order Runge-Kutta method, we use the Taylor series expansion for the order 3 which is of the form

+h = +h , +_h ! +_h^g g! +k h^n (3.27)

From the expansion we need to evaluate and thus differentiating we get

$$y'''(x) = f_{xx}(x, y) + 2 f_{xy}(x, y) f(x, y) + f_{yy}(x, y) f(x, y)^2$$
$$+ f_y(x, y)(f_x(x, y) + f_y(x, y) f(x, y)) \qquad (3.28)$$

and by substituting equations (3.6) and (3.28) into equation (3.27) we have

$$y(x + h) = y(x) + hf(x, y) + \frac{1}{2!}h^2(f_x(x, y) + f_y(x, y) f(x, y))$$
$$+ \frac{1}{3!}h^3(f_{xx}(x, y) + 2 f_{xy}(x, y) f(x, y)$$
$$+ f_{yy}(x, y) f(x, y)^2 + f_y(x, y)(f_x(x, y) + f_y(x, y) f(x, y))) \qquad (3.29)$$

The third-order Runge-Kutta method can then be formulated as

+h = += a += a +=$_g$a$_g$ (3.30) where

a =h , (3.31)

a =h +l h, +m ha (3.32)

a$_g$ =h +l h, +m ha +m$_g$$h$a (3.33)

From here the Taylor series expansions for a and a$_g$ are evaluated as

$k_2 = hf(x + p_1h, y + q_1k_1)$

$$= h[f(x, y) + p_1hf_x(x, y) + q_1hf_y(x, y) f(x, y) + \frac{1}{2}\{(p_1h)^2 f_{xx}(x, y) + 2p_1q_1h^2 f_{xy}(x, y)$$

$$+ (q_1 hf(x, y))^2 f_{yy}(x, y)\}]$$

$$= h[f(x, y) + h(p_1 f_x(x, y) + q_1 f_y(x, y) f(x, y)) + \frac{h^2}{2}(p_1^2 f_{xx}(x, y) + 2p_1 q_1 f_{xy}(x, y) f(x, y) + q_1^2 f_{yy}(x, y) f(x, y)^2)] \tag{3.34}$$

$$k_3 = hf(x + p_2 h, y + q_2 k_1 + q_3 k_2))$$

$$= h[f(x, y) + p_2 hf_x(x, y) + (q_2 hf(x, y) + q_3 k_2) f_y(x, y) + \frac{1}{2}\{(p_2 h)^2 f_{xx}(x, y)$$
$$+ 2p_2 h(q_2 hf(x, y) + q_3 k_2) f_{xy}(x, y) + ((q_2 hf(x, y))^2 + 2q_2 q_3 hk_2 f(x, y)$$
$$+ (q_3 k_2)^2) f_{yy}(x, y)\}]$$

$$= h[f(x, y) + p_2 hf_x(x, y) + q_2 hf_y(x, y) f(x, y) + q_3 k_2 f_y(x, y) + \frac{h^2}{2}(p_2^2 f_{xx}(x, y)$$
$$+ 2p_2 q_2 f_{xy}(x, y) f(x, y) + q_2^2 f_{yy}(x, y) f(x, y)) + \frac{1}{2}(2p_2 hq_3 k_2 f_{xy}(x, y)$$
$$+ 2q_2 q_3 hk_2 f_{yy}(x, y) f(x, y) + q_3^2 k_2^2 f_{yy}(x, y))]$$

$$= h[f(x, y) + p_2 hf_x(x, y) + q_2 hf_y(x, y) f(x, y) + \frac{h^2}{2}(p_2^2 f_{xx}(x, y)$$
$$+ 2p_2 q_2 f_{xy}(x, y) f(x, y) + q_2^2 f_{yy}(x, y) f(x, y)^2) + q_3 f_y(x, y) k_2 + \frac{1}{2}(2p_2 hq_3 f_{xy}(x, y)$$
$$+ 2q_2 q_3 hf_{yy}(x, y) f(x, y)) k_2 + \frac{1}{2} q_3^2 f_{yy}(x, y) k_2^2]$$

$$= h[f(x, y) + h(p_2 f_x(x, y) + q_2 f_y(x, y) f(x, y)) + \frac{h^2}{2}(p_2^2 f_{xx}(x, y)$$
$$+ 2p_2 q_2 f_{xy}(x, y) f(x, y) + q_2^2 f_{yy}(x, y) f(x, y)^2) + q_3 f_y(x, y) \times h\{f(x, y) + h(p_1 f_x(x, y)$$
$$+ q_1 f_y(x, y) f(x, y)) + O(h^2)\} + \frac{1}{2}(2p_2 hq_3 f_{xy}(x, y) + 2q_2 q_3 hf_{yy}(x, y) f(x, y)) \times h\{f(x, y)$$
$$+ O(h)\} + \frac{1}{2} q_3^2 f_{yy}(x, y) \times [h\{f(x, y) + O(h)\}]^2]$$

h^2 2

$$
\begin{aligned}
&= h\Big[f(x,y) + h\big(p_2 f_x(x,y) + q_2 f_y(x,y) f(x,y) + q_3 f_y(x,y) f(x,y)\big) + \frac{h^2}{2}\big(p_2^2 f_{xx}(x,y) \\
&\quad + 2p_2 q_2 f_{xy}(x,y) f(x,y) + q_2^2 f_{yy}(x,y) f(x,y)^2\big) + h^2 q_3 f_y(x,y)\big(p_1 f_x(x,y) \\
&\quad + q_1 f_y(x,y) f(x,y)\big) + \frac{h^2}{2}\big(2p_2 q_3 f_{xy}(x,y) f(x,y) + 2q_2 q_3 f_{yy}(x,y) f(x,y)^2\big) \\
&\quad + \frac{h^2}{2} q_3^2 f_{yy}(x,y) f(x,y)^2 \Big]
\end{aligned}
$$

$$
\begin{aligned}
&= h\Big[f(x,y) + h\big(p_2 f_x(x,y) + q_2 f_y(x,y) f(x,y) + q_3 f_y(x,y) f(x,y)\big) \\
&\quad + \frac{h^2}{2}\big(p_2^2 f_{xx}(x,y) + 2p_2 q_2 f_{xy}(x,y) f(x,y)\big) + h^2 q_3 f_y(x,y)\big(p_1 f_x(x,y) + q_1 f_y(x,y) f(x,y)\big) \\
&\quad + \frac{h^2}{2}\big(2p_2 q_3 f_{xy}(x,y) f(x,y)\big) + \frac{h^2}{2}\big(q_3^2 + 2q_2 q_3 + q_2^2\big) f_{yy}(x,y) f(x,y)^2 \Big]
\end{aligned}
$$

$$
\begin{aligned}
k_3 &= h\Big[f(x,y) + h\big(p_2 f_x(x,y) + (q_2 + q_3) f_y(x,y) f(x,y)\big) + \frac{1}{2} h^2\big(p_2^2 f_{xx}(x,y) \\
&\quad + 2p_2(q_2 + q_3) f_{xy}(x,y) f(x,y)\big) + h^2\big(p_2 q_3 f_{xy}(x,y) + q_2 q_3 f_y(x,y)^2 f(x,y)\big) \\
&\quad + \frac{1}{2} h^2 (q_2 + q_3)^2 f(x,y)^2 f_{yy}(x,y)\Big]
\end{aligned}
\tag{3.35}
$$

equations (3.31), (3.34) and (3.35) are substituted into equation (3.30) to get

$$
\begin{aligned}
y(x+h) &= y(x) + a_1 \times hf(x,y) + a_2 \times \Big[hf(x,y) + h^2\big(p_1 f_x(x,y) + q_1 f_y(x,y) f(x,y)\big) \\
&\quad + \frac{h^3}{2}\big(p_1^2 f_{xx}(x,y) + 2p_1 q_1 f_{xy}(x,y) f(x,y) + q_1^2 f_{yy}(x,y) f(x,y)^2\big)\Big] \\
&\quad + a_3 \times \Big[hf(x,y) + h^2\{p_2 f_x(x,y) + (q_2 + q_3) f_y(x,y) f(x,y)\} + \frac{h^3}{2}\{p_2^2 f_{xx}(x,y) \\
&\quad + 2p_3(q_2 + q_3) f_{xy}(x,y) f(x,y)\} + h^3\big(p_1 q_3 f_{xy}(x,y) + q_1 q_3 f_y^2(x,y) f(x,y)\big) \\
&\quad + \frac{h^3}{2}(q_2 + q_3)^2 f_{yy}(x,y) f(x,y)^2\Big]
\end{aligned}
$$

$$= y(x) + hf(x, y)\times(a_1 + a_2 + a_3) + h^2 f_x(x, y)\times (a_2 p_1 + a_3 p_2)$$

$$+ h^2 f_y(x, y) f(x, y)\times\{a_2 q_1 + a_3(q_2 + q_3)\} + h^3 f_{xx}(x, y)\times \frac{1}{2}(a_2 p_1^2 + a_3 p_2^2)$$

$$+ h^3 f_{xy}(x, y) f(x, y)\times\{a_2 p_1 q_1 + a_3 p_2(q_2 + q_3)\} + h^3 f_{yy}(x, y) f(x, y)^2 \times \frac{1}{2}\{a_2 q_1^2 + a_3(q_2 + q_3)^2\}$$

$$+ h^3 f_{xy}(x, y)\times(a_3 p_1 q_3) + h^3 f_y(x, y)^2 f(x, y)\times (a_3 q_1 q_3) \qquad (3.36)$$

the result of (3.36) is then compared with equation (3.29) to help us find the values of the constants = , = , =g, l , m , l , m and mg. The comparison give us the following equations

= += +=g =1 (3.37)

= l +=gl =_ (3.38)

= m +=gm +mg=_ (3.39)

_= l +=gl = o_ (3.40)

= l m +=gl m +mg = g_ (3.41)

_= m +=gm +m = o_ (3.42)

=gl mg = o_ (3.43)

=gm mg = o_ (3.44)

If we choose l =1 and l =m =_ , then we obtain = ==g = o_, =g = n o_ ,m =−1

and mg =2. Thus the third-order Runge-Kutta method formulas are as follows:

+h = + o_ a +4a +ag (3.45) where

a =ℎ , (3.46)

a =ℎ + #_ , +_a (3.47) a_g =ℎ +ℎ

, −a +2a (3.48) which is the same as

+ℎ = +a +4a +a − o g ℎ (3.49)

where

a = , (3.50)

a = + #_, + #_a (3.51) a_g = +ℎ ,

−ℎa +2ℎa (3.52)

3.1.3 Runge-Kutta Fourth-Order Method

In deriving the fourth-order Runge-Kutta method we shall use a different procedure. The construction of the fourth-order Runge-Kutta method starts with the fourth order Taylor series expansion of the form

+ℎ = +ℎ , +_ℎ ! +_ℎ^g g! +_ℎ^n n! n +k ℎ^q (3.53)

where n n is evaluated to get

= , +3 _, , +3 __, , + __, , g

+3 _, +3 __, , + _, , . (3.54)

By substituting equations (3.6), (3.28) and (3.54) into equation (3.53) we have

+ℎ = +ℎ , +_ℎ , + _, ,

+ o_ ℎ^g , +2 _, , + __, , + _,

+ n__ ℎ^n M , +3 _, , +3 __, ,

+ __, , g +3 _, +3 __, ,

+ _ , , N (3.55)

In general a Runge-Kutta method of order @ can be written as

+ℎ = +ℎ∑ᵗ cu =c ac +k ℎᵗ (3.56) where

ac = M +lc ℎ , +ℎ∑ᵗ vu mcv av N (3.57) which are

the increments obtained evaluating the derivatives of at the d-th order [21].

The derivation for the ordinary 4th order Runge-Kutta method is developed using equation (3.56) with @=4, at the starting point, the midpoint and the end point of any interval , +ℎ , thus we choose lc: l =0, l =_ , lg =_ , ln =0 and mcv: m =_ , mg =_ , mng =1.

Where mcv =0 otherwise.

Starting, we define the following quantities

+ℎ = +ℎ , (3.58)

+ℎ = +ℎ + # , + # ;− − (3.59)

+ℎ = +ℎ + # , + # [+ # , + # ,] − − (3.60)

then we define

a = , (3.61)

a = + # , + # a = , + # _ − $, (3.62) $

ag = Q + # , + # + # , + # a S= , + # _ $ w , + # Q$ $, Sx − —

= , + # Q$ $, S+ # n R — ($ $R R , (3.63)

an = Q +ℎ , +ℎ + # , + # a S − =+ℎ , +ℎ [+ # _ , + # _ + # _ , + # _ ,]

= , +ℎ _ $,

$ + #_ __$ [, + # $,]

$ − (\$

= , +ℎQ$__$, S+ #R I$$R R , O+ #n f I$$f f , O — — (3.64) where

__$$, = ^__^ , + ^_ __^ , = , + _ ,

If we express the general formula for the fourth order Runge-Kutta method as

+ℎ = += ℎa += ℎa +=gℎag +=nℎan (3.65)

Then by substituting the equations (3.61) to (3.64) into equation (3.65) we have

+ℎ = += ℎ , += ℎ , + #_ __ $, $

+ # $ #R $R — =gℎy , +_Q$, S+ n I$ R , Oz

+=nℎy , +ℎQ$__$, S+ #R I$$RR , O+ #nf I$$ff , Oz — —

= += ℎ , += ℎ , + \f#R Q $$, S+=nℎ ,

+ \{# I$$ R , O+ \{n# I $$ f , O — — $

= + = += +=g +=n , ℎ+ \R + \f +=n — Q$$, Sℎ

Q \ \ $R \ $f __$, S

+ \f# Q $, S+ \f# I $ R , O+=nℎ , — +=nℎ
$ n $

f R { f

R f R

= n£+ £l$ R , Oh^g + n£l$ f , Oh^n (3.66)

By comparing equation (3.55) and equation (3.66) we get the following equations

= += +=g +=n =1 (3.67)

= + =g +=n = − 2 − (3.68) − n =g + =n = o

(3.69) − − − n =n = n (3.70) − —

which when solved gives = = o , = = g , =g = g and =n = − o . − − −

We therefore get the 4th order Runge-Kutta method as

+h = + o a +2a +2ag +an h − (3.71) where

a = , (3.72)

a = +h , +ha (3.73) − ag = − +_h , +_ha (3.74)

an = +h , +$h$ag (3.75)

In the Runge-Kutta methods, the introduction the notation a ,a ,ag,an into the methods is to eliminate the need for successive nesting in the second variable of , .That is, instead of writing the equation (3.71) as

+h = + #_ , + #_ +_h, +_h , o g

+ #_ Q +_h, +_h +_h, +_h , S g

+ #_ l +h, +h Q +_h, +_h +_h, +_h , SO o a is introduced to take care of , , a takes care of +_h, +_h , , whiles a_g takes care of Q +_h, +_h +

_h, +_h , S and also a_n taking care of

I +h, +h Q +_h, +_h +_h, +_h , SO . Therefore, for easy numerical

Computation, the 4th order Runge-Kutta method is expressed in the form of equation (3.71).

3.2 Non-Newtonian (Multiplicative) Runge-Kutta Methods

In this section, we will derive the non-Newtonian (multiplicative) Runge-Kutta methods that will be used to find suitable approximations to the solution of non-Newtonian(multiplicative) initial value problems of the form

* = , , (3.76) or

* = , (3.77)

with initial condition

= . (3.78)

3.2.1 Non-Newtonian (multiplicative) Runge-Kutta Second-Order Method

In analogy to the ordinary second-order Runge-Kutta method, we will derive the second-order non-Newtonian (multiplicative) Runge-Kutta method for solution of the differential equation (3.76).We start with the second order multiplicative Taylor expansion for +h which is given as

+h = · * # · ** TR R! ·... (3.79) from Theorem 2

[3](Multiplicative Taylor's Theorem for One Variable). By substitute the right hand side of equation (3.77) into equation (3.79) the expansion becomes

+h = · , # · ** TR R! ·... (3.80)

We can express ** as

** = * * = , * = , * (3.81)

Recall that is a function of .Thus applying Theorem 3 [3](multiplicative chain rule) to ** we have

** = $__$* , = *, · _*,–1 (3.82)

or

$$y^{**}(x) = f_x^*(x,y) \cdot f^*_{y,y(x)\cdot \ln f(x,y)}$$ (3.83)

since $= y(x) \cdot \ln(y^*$ and * = , .

Substituting equation (3.83) into equation (3.80), then the multiplicative Taylor expansion becomes

+h = · , # · *, · _*,_ ·2* ,– IR R ·...

= · , IR # · *, R · _*,_ ·2* ,–IRR ·... (3.84)

where *, denotes the non-Newtonian(multiplicative) partial derivative with respect to and _*, with respect to respectively. We then formulate the second order non-Newtonian (multiplicative) Runge-Kutta method as

+h = ·a\# ·a[# (3.85)

where

a = , (3.86)

a $= f(x + ph\,, y \cdot k_1$}# (3.87) To compare equation (3.85) with equation (3.84), we need to find the multiplicative Taylor series for a and by the application of the chain rule of multiplicative derivative

a = , · *,~#· $f^*_{y,y(x)qh\cdot \ln k}$J

= , · *,~# · _*,$_{y(x)qh\cdot \ln f(x,y)}$ (3.88)

Then by substituting equations (3.88) and (3.86) into equation (3.85) we get the non-Newtonian (multiplicative) second order Runge-Kutta method as

+h = · , \# · , · *,~# · _*,$y(x)qh$·ln $f(x,y)$ [#

= · , \ [# · *, bph^2 · f^*_y,_ ·2* ,_ []#R (3.89)

Now comparing equation (3.89) with equation (3.84) yields

=+>=1 (3.90)

>l=_ (3.91)

>m=_ (3.92)

Of course, we have infinitely many solutions of the equations (3.90), (3.91) and (3.92), as the number of unknowns is greater than the number of equations. One possible choice of the parameters [9] =,>,l, and m is == , >=_ , l=1 and m=1. This gives as the formulae for the second order non-Newtonian (multiplicative) Runge-Kutta method as

$$x + h) = y(x) \cdot k_1^{\frac{h}{2}} \cdot k_2^{\frac{h}{2}} \quad (3.93)$$

where

a = , (3.94)

a = +h, ·a# (3.95) Depending on the problem the parameters =,>,l, and m can also be chosen differently in order to satisfy the equations (3.90), (3.91) and (3.92).

3.2.2 Non-Newtonian (multiplicative) Runge-Kutta Third-Order Method

To derive the third order non-Newtonian (multiplicative) Runge-Kutta method, we shall use the multiplicative Taylor expansion for +h up to order 3 which is of the form

+h = · * # · ** TR R! · *** Tf f! ·..., (3.96)

Remembering that ** = * * = , * we can write *** as

*** = * ** = , ** (3.97)

Noting that * = M , N can also be expressed as * = , .

From equation (3.96) we can evaluate *** by the application of multiplicative chain rule to get

*** = $__** ,
$

= __ __$*$* ,
$ $

$$= —\left(f_x\,(x, y) \cdot f_y^*\right.$$ $* * ,_1 N
$

= * , · _* ,–1 · _* ,–1 · __* ,–1 R · _* ,–11

= * , · _* ,–1 · __* ,–1 R · _* ,–11 (3.98) Since ** = $__$* , = *, · _*,–1 and partial multiplicative derivatives are commutative [16].

We substitute equations (3.98), (3.82) and (3.77) into equation (3.96) to get the multiplicative Taylor expansion up to order 3 for $+h$ as

$+h$ = $\cdot\,(f(x, y))^h \cdot (f_x^*(x, y) \cdot f_y^*$,_1 TR!R

Tf
$\cdot\,[\,f_{xx}^*(x, y) \cdot f_{yx}^*$,_1 · __* ,_1 R · _* ,–11] f! ·...

TR 1 TR Tf R_1 Tf
= · , # · *, R · _*, R · *, •· _*, •

1 RTf 11 Tf
· _* , ______• · _*, ______• ·... (3.99)

The third-order non-Newtonian (multiplicative) Runge-Kutta method can be depicted as

$$+h = \quad \cdot a^{\backslash\#} \cdot a^{[\#} \cdot a_g^{H\#} \tag{3.100}$$

where

$$a = , \tag{3.101}$$

$$a \quad = f(x + ph , y \cdot k_1{}^{\}\#} \tag{3.102}$$

$$a_g = +|_1 h , y \cdot k_1^{q_1 h} \cdot k_{2}^{q}{}_{R\#} \tag{3.103}$$

To make the comparison between equation (3.99) and equation (3.100), we need to find the multiplicative Taylor expansion for a and a_g which are

a = , · ∗,~# · _∗,_8J€T 1~#

= , · ∗,~# · _∗,~8J€T_1# (3.104)

€JT8€RT 1 •

ag = , · ∗,~J# · _∗,_8J R ~J# · ∗ , JRT R · _∗ ,~J# R_8J€JT8R€RT 1

· _∗ ,• JT R,J€RJT,€RRT 1 R · _∗,• JT R,J€RJT,R€RT 11

= , · ∗,~J# · _∗,~J8J€JT8R€RT_1# · ∗ ,•JRRTR · _∗ ,~JR8J€JT8R€RT#R_1

•RJ,R€J JT,R€R RTTR 1 R •R €

· _∗ , R · _∗, J,JJT,R€RRTTR 11

(3.105) that is by the application of the chain rule multiplicative derivative.

Now we substitute equations (3.101), (3.104) and (3.105) into equation (3.100) to get the third order non-Newtonian (multiplicative) Runge-Kutta method as

$$+h = \quad \cdot (f(x,y))^{ah} \cdot [\, f(x,y) \cdot f^{*}_{x},_{\sim\#} \cdot f^{*}_{y},_{\sim 8_J \text{€T}\_{}^{1}\#}]_{[\#}$$

$\cdot [f(x,y) \cdot f_x^*$,~J# · _∗,~J8J€JT8R€RT_1# · ∗ ,•RJRTR

∗ ,~JR8J€JT8R€RT#R_1 · __∗ ,•JR,R€J JT,RR€R RTTR 1 R

· _

•RJQ,€JJT,€RRTSTR 11

· _*, R]H#

= · , \# · , [# · ∗,~[#R $\cdot f_y^*$,[~8J€T#R_1 · , H#

· ∗,H~J#R · _∗,H~J8J€JT8R€RT#R_1 · ∗ ,K•JRRTf · _∗ ,H~JR8J€JT8R€RT#f_1

K•R R€

∗ J,J JT,R€R RTTf 1 R K•R €

· __, R · _∗, J,JJT,R€RRTTf 11

= · , \ [H # · ∗, ~[H~J#R · _∗, [~8J€TH~J8J€JT8R€RT#R_1

· *,K•RJTf · ∗ ,H~JR8J€JT8R€RT#f_1 · __∗ ,K• JR,JR€JT,RR€R RTTf 1 R

R _

K•R €

· _*, J,JJT’RR€RTTf 11 (3.106)

Comparison of the powers of , and its partial derivatives in equation (3.99) with equation (3.106) up to order 3 in h gives

=+>+:=1 (3.107)

l>+:l =_ (3.108)

>la}# +:l a}J#a}R# =_ (3.109)

_:l =_ (3.110)

o

:l a}J#a}R# =_ (3.111)

o

_:l a}J#a}R# =_ (3.112) o

_:l a}J#a}R# =_ (3.113) o

As before, the number of solutions are infinitely many. By choosing [8], l =1, m=m =0 and l= ,we get == , >=_ , := , Therefore the 3rd order non-Newtonian g o g

(multiplicative) Rung-Kutta method takes the form

1 1 1

•# ·aR# ·a g f# (3.114)

+h = ·a

Where

a = , (3.115)

a = +_h , (3.116) g a$_g$ = +h

, (3.117)

3.2.3 Non-Newtonian (multiplicative) Runge-Kutta Fourth-Order Method

In analogy to the above described 3rd order non-Newtonian (multiplicative) Runge-Kutta method [16] , we will now derive the 4rd order non-Newtonian (multiplicative) Runge-Kutta method. Therefore we start by using the multiplicative Taylor expansion for +h up to order 4 which is of the form

TR Tf $\frac{h^4}{4!}$ · ...

+h = · * # . ** R! · *** f! · * n (3.118)

Recall that * = M , N= , and using the knowledge in equations (3.81) and (3.97) we can therefore express * n as

* n = * *** = , *** = , *** (3.119)

By the application of multiplicative chain rule, equation (3.119) results to

$$= \left(f^*_{xx}(x,y) \cdot f^*_{yx} \cdots \right) \qquad (3.120)$$

We now substitute equations (3.77), (3.82), (3.98) and (3.120) into equation (3.118) to get the multiplicative Taylor expansion for $+h$ up to order 4 as

$$+h = \cdots \left[f^*_{xx}(x,y) \cdot f^*_{yx} \cdots \right] \cdot \left[f^*_{xxx}(x,y) \cdot f^*_{yxx} \cdots \right]^{\frac{h^4}{4!}} \cdot \ldots$$

· _* , R{ · __* , ______R{ · _* , R{ · __* , R{

111 T{

· _*, R{ ·... (3.121)

The fourth order non-Newtonian (multiplicative) Runge-Kutta method has the form

+h = ·a\# ·a[# ·ag H# ·an $# (3.122) where

a = , (3.123)

a $= f(x + ph , y \cdot k_1$}# (3.124) ag

= +l1 $h , y \cdot k_1^{q_1 h} \cdot k_2^{q}$R# (3.125) an = +l

2 $h , y \cdot k_1^{q_3 h} \cdot k_2^{q_4 h} \cdot k_3^{q}$f# (3.126) In order to compare equation (3.122) with equation (3.121) we need to expand also an using the multiplicative Taylor theorem since that of a and ag are the equations (3.104) and (3.105). By the application of the chain rule multiplicative derivative the Taylor expansion for an is

€fT

an = , · *,~R# · _*,_8J 8R€{T8f€fT 1~R# · * , •RRT R

· _* , ~R# R_8J€fT8R€{T8f€fT 1 · __* ,•______RT R,J€fTR,R€{T,f€fT 1 R

•RT R,J€fT,€R{T,€ffT 11 •RT f •RT f,€JfT,€R{T,f€fT 1

· _*, ______R · * , • · _* , ______R

•RT f,J€fT,€R{T,f€fT 1 R •RT f,J€fT,€R{T,f€fT 1 f

· __* , ______R · __* , ______•

•RT f,J€fT,R€{T,f€fT 11 •RT f ,€JfT,R€{T,€ffT 1,J€fT,€R{T,f€fT 11

· _* , ______R · __* , ______ R

•RT f,J€fT,€R{T,€ffT 111

· _* , ______•

,~R# · _, ~R8J€fT8R€{T8f€fT_1# · * ,•RRRTR

= , ·

R8J€fT8R€{T8f€fT _1#R ·

__* ,•RR,R€J fT,RR€{TR,fR€fT 1 RTR

· _* , ~R

•RR,€JfT,€R{T,f€fT 11 TR •fRTf •Rf,€JfT,€R{T,f€fT 1Tf

· _*, R · * , • · _* , R

•fRTf,R€J fT,RR€{T,R€f fT 1 R •fR,Jf€ fT,Rf€{T,f€f fT 1 f Tf

· __* , R · ___* , •

•fR,€JfT,€R{T,f€fT 11Tf •fR,€JfT,€R{T,f€fT

1,J€fT,R€{T,€fJT 11Tf

· _* , R · __* , R

•fR,€JfT,€R{T,f€fT 111Tf

$$\cdot \_^*, \quad \bullet \tag{3.127}$$

Then by substituting equations (3.123), (3.104), (3.105), and (3.127) in equation (3.122), we get the non-Newtonian (multiplicative) Runge-Kutta expansion for the comparison with the multiplicative Taylor expansion of equation (3.121) as

$+h =$ $\cdot (f(x,y))^{ah} \cdot (f(x,y) \cdot f_x^*$,~# · _*,~8J€T _1# [#

•RJTR

$\cdot (f(x,y) \cdot f_x^*$,~J# · _*,~J8J€JT8R€RT_1# · * , R

· _* ,~JR8J€JT8R€RT#R_1 · __* ,• RJ,JR€JT,RR€R RTTR1 R

· _*,•RJ ,€JJT,R€RRTTR 11 H# $\cdot (f(x,y) \cdot f_x^*$,~R#

$\cdot f_y^*,$ ~R8J€fT8R€{T8f€ƒT_1# · * ,•RRRTR

R8J€fT8R€{T8f€ƒT _1#R · __* ,•

R,R€J fT,RR€{TR,R€f ƒT 1 RTR

R

· _* , ~R

•RR,€JfT,€R{T,f€ƒT 11 TR •RfTf •Rf,€JfT,€R{T,f€ƒT

1Tf

· _*, R · * , • · _* , R

•fRTf,R€J fT,RR€{T,R€f ƒT 1 R •fR,Jf€ fT,Rf€{T,f€f ƒT 1 f Tf

· __*, R · ___*, •

•fR,€JfT,€R{T,f€ƒT 11Tf •fR,€JfT,R€{T,f€ƒT

1,J€fT,R€{T,€fƒT 11Tf

· _*, R · __*, R

•fR,€JfT,€R{T,f€ƒT 111Tf

· _*, • $#

= · , \# · , [# · *,~[#R · _*,[~8J€T#R_1 · , H#

· *,H~ J#R · _*,H~J8J€JT8R€RT#R_1 · * ,K•JRRTf · _* ,H~JR8J€JT8R€RT#f_1

· __* , J,J JT,R€RR RTTf 1 R · _*,K•RJ,J€JT,R€RRTTf 11 · , $# · *,$~R#R K•R R€

*

· „•RRTf * ,$~RR8J€fT8R€{T8f€ƒT#f _1

_ ,$~R8J€fT8R€{T8f€ƒT_1#R · * , R · _

„•RR,R€J fT,RR€{T,R€f ƒT 1 RTf „•RR,€JfT,€R{T,f€ƒT 11

Tf

· __*, R $\cdot f_y^*,$ R

„•fRT{ „•fR,€JfT,€R{T,f€ƒT 1T{ „•Rf,R€J fT,R€R {T,R€f ƒT 1 R T{

· * , • · _* , R · __* , R

„•fR,f€J fT,f€R {T,ff€ƒT 1 f T{ „•fR,€JfT,R€{T,€ffT 11T{

· ___*, • · _*, R

„•fR,€JfT,R€{T,€ffT1,J€fT,R€{T,f€fT 11T{
„•fR,€JfT,€R{T,f€fT 111T{

· _* , R · _*, •

= · , \ [H $ # · *, ~[H~J$~R#R

· _*, [~8J€TH~J8J€JT8R€RT$~R8J€fT8R€{T8f€fT#R_1 · * ,K•JRF„•R RRTf · _*

,H~JR8J€JT8R€RT$~RR8J€fT8R€{T8f€fT#f_1

K•RJ,R€J JT,R€R RTF„•RR,R€J fT,R€R {T,R€f fTTf 1 R

· _* , R

K•RJ,€JJT,€RRTF „•RR,€JfT,€R{T,f€fTTf 11 „•fRT{

· _*, R · *, •

„•fR,€JfT,R€{T,€ffT 1T{ „•Rf,R€J fT,RR€{T,R€f fT 1 R T{

$\cdot f^*_{yxx}$, R · _*, R

„•fR,f€J fT,f€R {T,ff€fT 1 f T{ „•fR,€JfT,R€{T,€ffT 11T{

· _* , • · _*, R

„•fR,R€J fT,R€R {T,fR€fT 1 11T{
„•fR,€JfT,R€{T,€ffT 111T{

· _*, R · _*, • (3.128)

Therefore we get by comparison of equation (3.128) with equation (3.121), the following set of equations

=+>+:+...=1 (3.129)

l>+:l +...l =_ (3.130)

>la}# +:l a}J#a}R# +...l a}f#a}{#ag}f# =_ (3.131)

_ :l +...l =_ (3.132) o

:l a}J#a}R# +...l a}f#a}{#ag}f# = o_ (3.133)

_M:l a}J#a}R# +...l a}f#a}{#a g}f#N= o_ (3.134) _

M:l a}J#a}R# +...l a}f#a}{#ag}f#N= o_ (3.135)

_...lg =__ (3.136)
o n

_M...lga}f#a}{#ag}f#N= n__g (3.137)

_M...lga}f#a}{#ag}f#N= n__g (3.138)

o_ M...lgag}f#ag}{#agg}f#N= n__ (3.139)

_M...lga}f#a}{#ag}f#N= n__g (3.140) _

M...lga}f#a}{#ag}f#N= n__g (3.141)

o_ M...lga}f#a}{#ag}f#N= n__ (3.142)

As =, >, : and ... are determined by the choices of l, l , l , m, m , m , mg, mn and mq. By

selecting, l =1, l=_ , l =_ , m=m =m =mg =mn = mq =0 and solving the set of equations gives ...= , :=_ , >=_ and ==_ . We therefore get the 4th order non-Newtonian n g o n

(multiplicative) Runge-Kutta method as

1 1 1 1

{# ·a•# ·a g f# ·a n {# (3.143)

+h = ·a where

a = , (3.144)

a = +_h, (3.145) ag = +_h

, (3.146) $a_n = +h$

, (3.147)

Depending on the problem the parameters can also be chosen differently in order to satisfy the equations (3.129) to (3.142).

CHAPTER FOUR

ANALYSIS AND INTERPRETATIONS OF NUMERICAL SIMULATIONS OF ORDINARYAND NON-NEWTONIAN (MULTIPLICATIVE) RUNGE-KUTTA METHODS

In this chapter, we will compare the ordinary and non-Newtonian (multiplicative) Runge-kutta Methods using examples of ordinary and non-Newtonian (multiplicative) differential equations with their analytical solutions [17].

4.1 COMPUTATIONAL COMPARISON

We begin by considering the solution for non-Newtonian (multiplicative) initial value problem

$$^{*} = \quad , \quad 1 = 1 \tag{4.1}$$

The ordinary differential equation corresponding to equation (4.1) is

$$= 2 \,, \quad 1 = 1 \tag{4.2}$$

The exact solution for both equations (4.1) and (4.2) is

$$= {}^{R} \tag{4.3}$$

We first solve the equation (4.1) by using the second-order non-Newtonian (multiplicative) Runge-Kutta method and equation (4.2) by also using the second-order ordinary Runge-Kutta method. In the following table we compare the results of the non-Newtonian (multiplicative) Runge-Kutta method and the ordinary Runge-Kutta method, using a step size of h=0.1 and ,=6 points. The results in tabular form and the graphs are as follows

	W ‡U^	U \H%	‰Š*PU 8Š%%\	‹ Œ=Ad• ‹‹ d, % ? ,?, −• •	‹ Œ=Ad• ‹‹ d, % ? ”, B −a’AA=
1	1	1	1	0	0
1.1	1.23367805996	1.23367805996	1.23100000000	0	0.2171
1.2	1.55270721851	1.55270721851	1.54527430000	0	0.4787
1.3	1.99371553324	1.99371553324	1.97795110400	0	0.7907
1.4	2.61169647342	2.61169647342	2.58142398583	0	1.1591
1.5	3.49034295746	3.49034295746	3.43484275554	6.36 ^{n}	1.5901
1.6	4.75882124514	4.75882124514	4.65936419770	3.73 ^{n}	2.0900

Table 4.1: Comparison of the results of non-Newtonian (multiplicative) and ordinary second-order Runge-Kutta methods with the exact values and their relative errors in percentage for * = and =2

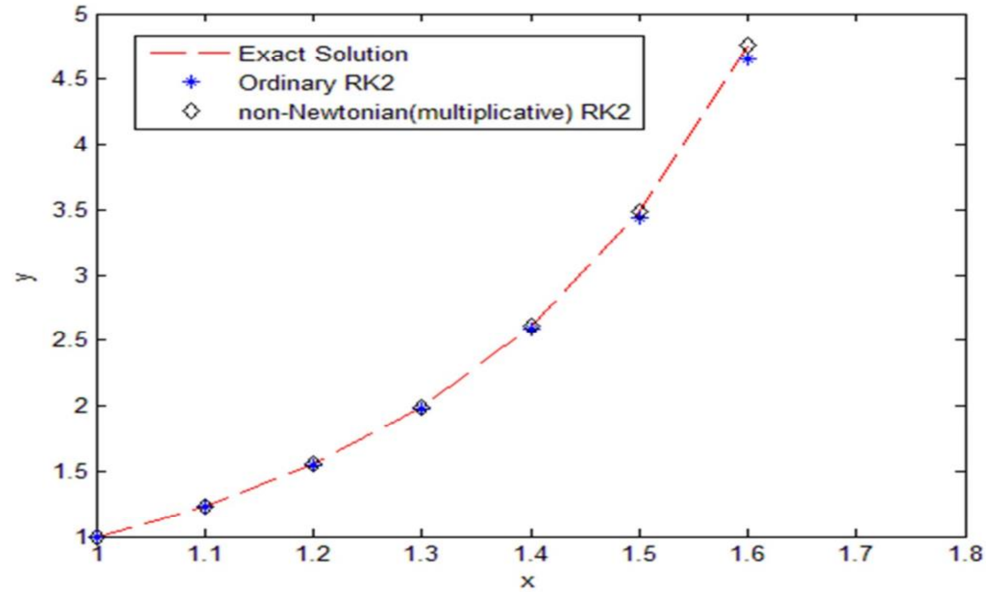

Figure 4.1: Graphs of non-Newtonian (multiplicative) and ordinary second-order Runge-Kutta methods and the exact solution for * = and =2

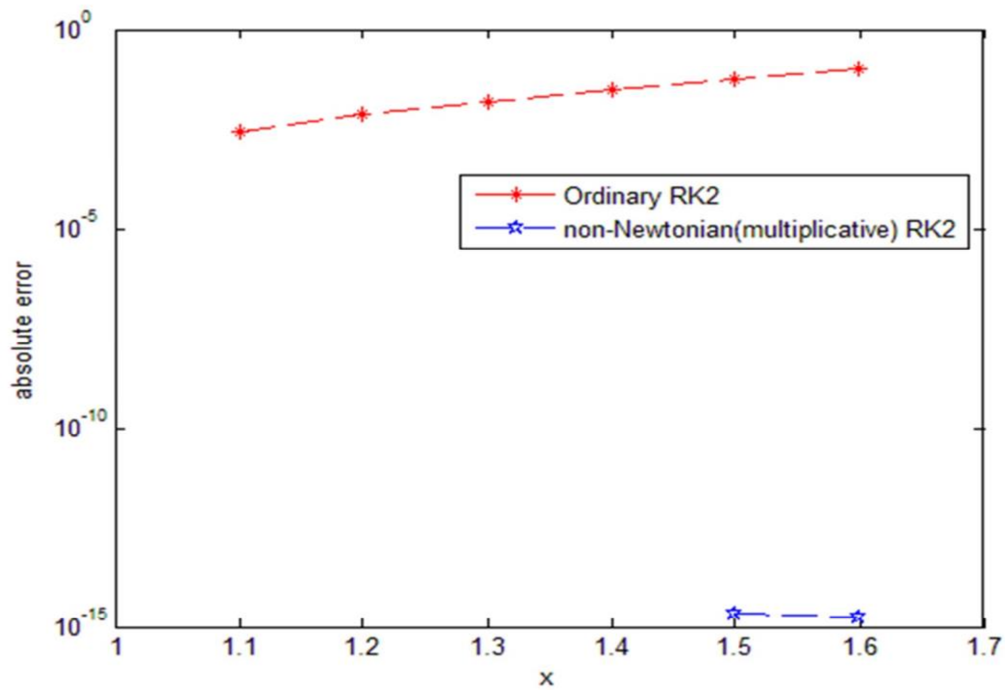

Figure 4.2: Approximation error containing the second-order non-Newtonian (multiplicative) and the ordinary Runge-Kutta methods of table 4.1

The comparison of the results presented in the table 4.1 shows that the non-Newtonian (multiplicative) second-order Runge-Kutta method gives better results compared to the ordinary second-order Runge-Kutta method. Since the error terms of non-Newtonian (multiplicative) Rung-Kutta method is smaller than that of the ordinary Runge-Kutta method. This is also represented in figure 4.2

The following table gives us the numerical results and the relative errors in percentage for the non-Newtonian(multiplicative) Runge-Kutta third-order method and the ordinary third- order Runge-Kutta method compared to the exact result. The graphs are also below.

	W ‡U^	U \H%	‰Š*PU 8Š%%\	‹ Œ=Ad• ‹‹ d, % ? ,?, −• •	‹ Œ=Ad• ‹‹ d, % ? ʻʻ, B −a'AA=
1	1	1	1	0	0
1.1	1.23573590445	1.23367805996	1.23360666667	0.1668	0.0058
1.2	1.55789154498	1.55270721851	1.55248247634	0.3339	0.0145
1.3	2.00370907394	1.99371553324	1.99318050195	0.5013	0.0268
1.4	2.62916595013	2.61169647342	2.61055354578	0.6689	0.0438
1.5	3.51955067859	3.49034295746	3.48803020727	0.8368	0.0663
1.6	4.80664819378	4.75882124514	4.75427818665	1.0050	0.0955

Table 4.2a: Comparison of the results of non-Newtonian (multiplicative) and ordinary third-order Runge-Kutta methods with the exact values and their relative errors in percentage

for * = and =2

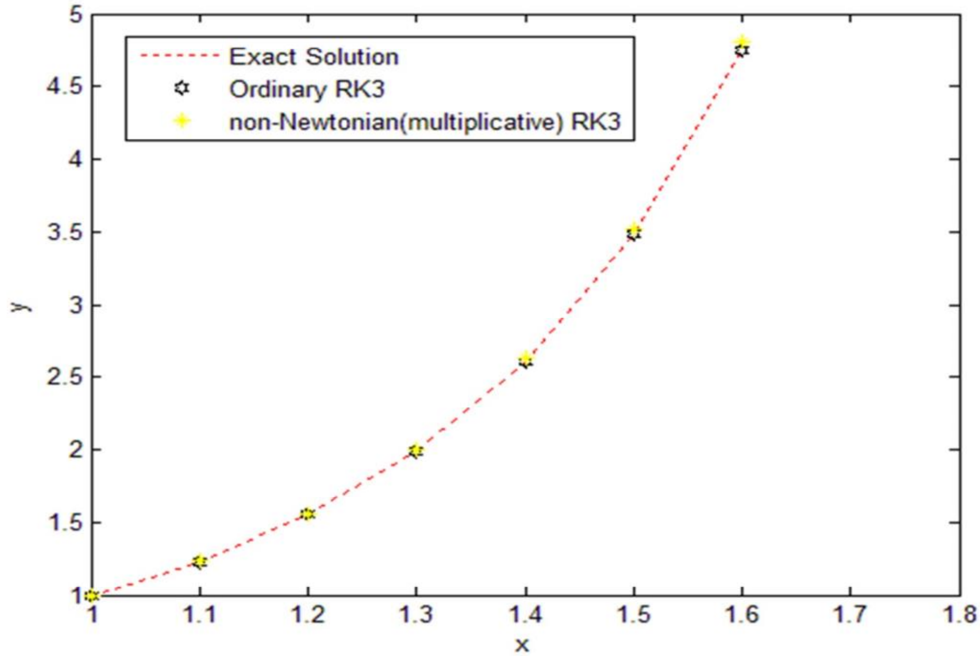

Figure 4.3a: Graphs of non-Newtonian (multiplicative) and ordinary third-order Runge-Kutta methods and the exact solution for * = and =2

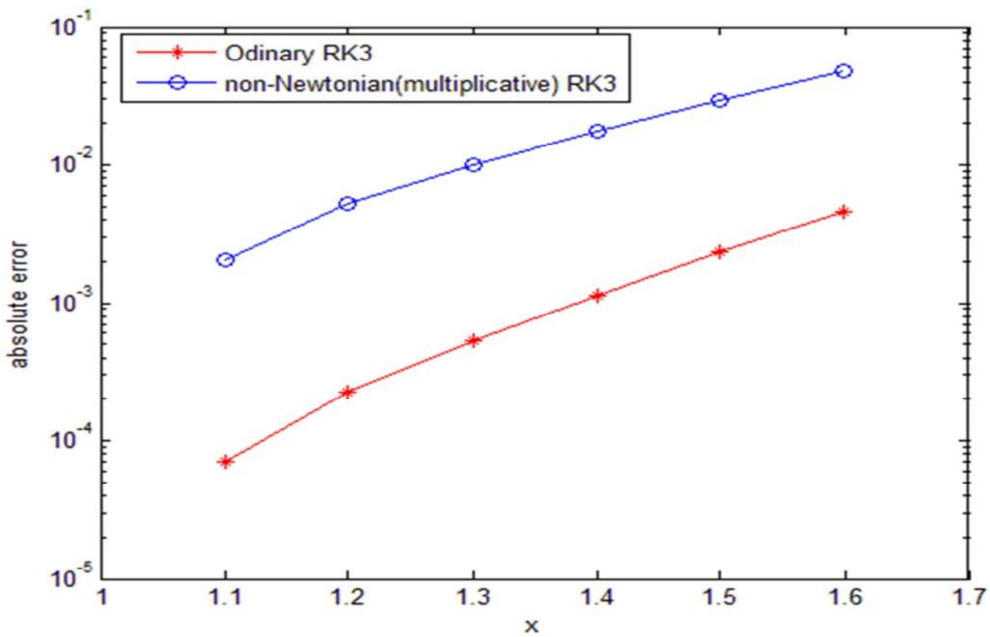

Figure 4.4a: Approximation error containing the third-order non-Newtonian (multiplicative) and the ordinary Runge-Kutta methods of table 4.2a

The comparison of the results presented in the table 4.2a shows that the ordinary third-order Runge-kutta method gives better results compared to the non-Newtonian (multiplicative) third-order Runge-Kutta method. Since the error terms of the ordinary Runge-Kutta method is smaller than that of the non-Newtonian (multiplicative) Rung-Kutta method. This is also represented in figure 4.4a.

We now set our parameters in the case of the non-Newtonian (multiplicative) third-order Rung-Kutta method in equation (3.103) to :=_ , $=_{o}$, $==_{g}$, l=_ , l_{o} =1, m =−1, m =2,

and m= which is the parameters widely used for the third-order ordinary Runge-Kutta method.

Table 4.2b below gives us the comparison of the results for the parameters chosen and also the graphs are shown in figures 4.3b and 4.4b.

	W ‡Uˆ	U \H%	‰Š*PU 8Š%%\	‹ Œ=Ad• ‹‹ d, % ? ,?, –• •	‹ Œ=Ad• ‹‹ d, % ? ”, B −a'AA=
1	1	1	1	0	0
1.1	1.23367805996	1.23367805996	1.23360666667	0	0.0058
1.2	1.55270721851	1.55270721851	1.55248247634	0	0.0145
1.3	1.99371553324	1.99371553324	1.99318050195	0	0.0268
1.4	2.61169647342	2.61169647342	2.61055354578	0	0.0438
1.5	3.49034295746	3.49034295746	3.48803020727	6.36 ^{n}	0.0663
1.6	4.75882124514	4.75882124514	4.75427818665	3.73 ^{n}	0.0955

Table 4.2b: Comparison of the results of non-Newtonian (multiplicative) and ordinary third-order Runge-Kutta methods with the exact values and their relative errors in percentage

for * = and =2

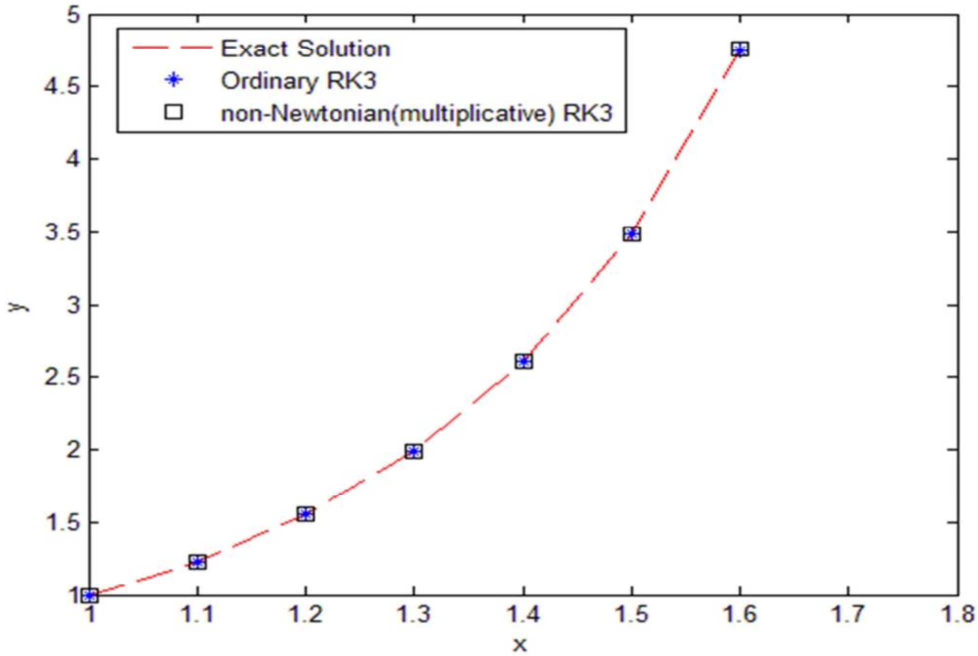

Figure 4.3b: Graphs of non-Newtonian (multiplicative) and ordinary third-order Runge-Kutta methods and the exact solution for * = and =2

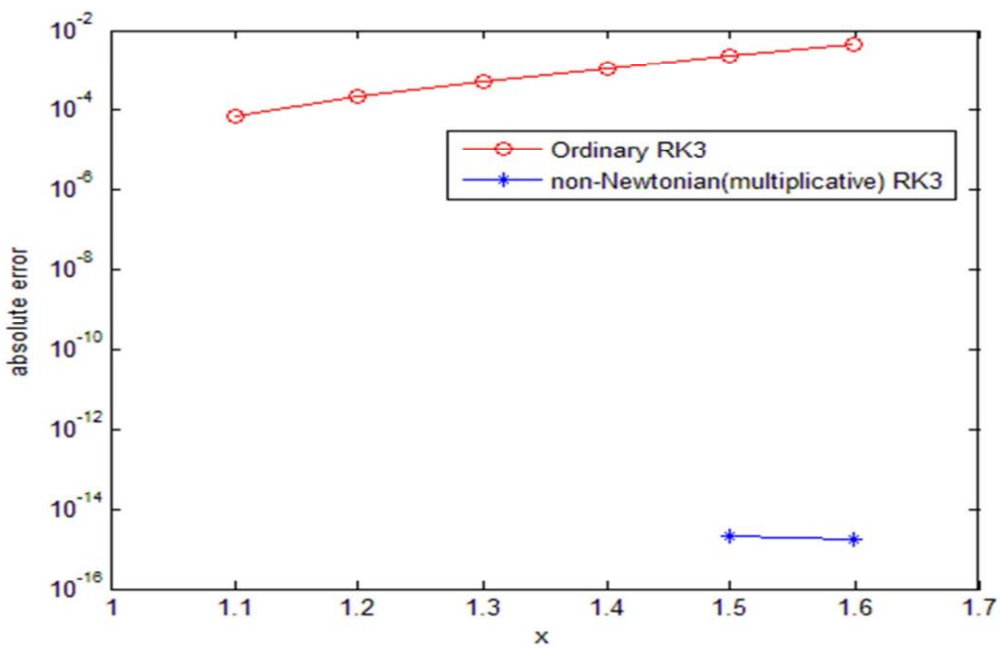

Figure 4.4b: Approximation error containing the third-order non-Newtonian (multiplicative) and the ordinary Runge-Kutta methods of table 4.2b

We observe that the comparison of results in table 4.2b shows that the non-Newtonian (multiplicative) third-order Runge-Kutta method gives us better results as compared to the ordinary third-order Runge-Kutta method for the same parameters. This can also be notice in figure 4.4b.

The following table gives us the numerical results and the relative errors in percentage for the non-Newtonian(multiplicative) Runge-Kutta fourth-order method and the ordinary fourth- order Runge-Kutta method compared to the exact result. The graphs are also below.

	W ‡Uˆ	U \H%	‰Š*PU 8Š%%\	‹ Œ=Ad• ‹‹ d, % ? ,?, –• •	‹ Œ=Ad• ‹‹ d, % ? ‘’, B –a’AA=
1	1	1	1	0	0
1.1	1.23367805996	1.23367805996	1.23367806171	0	1.62 ˝
1.2	1.55270721851	1.55270721851	1.55270731751	0	6.38 ᵒ
1.3	1.99371553324	1.99371553324	1.99371555011	0	8.53 ˝
1.4	2.61169647342	2.61169647342	2.61169654914	0	2.91 ᵒ
1.5	3.49034295746	3.49034295746	3.49034350924	2.22 ᵍ	1.58 ᑫ
1.6	4.75882124514	4.75882124514	4.75882143391	1.78 ᵍ	3.95 ᵒ

Table 4.3: Comparison of the results of non-Newtonian (multiplicative) and ordinary fourth-order Runge-Kutta methods with the exact values and their relative errors in percentage for * = and =2

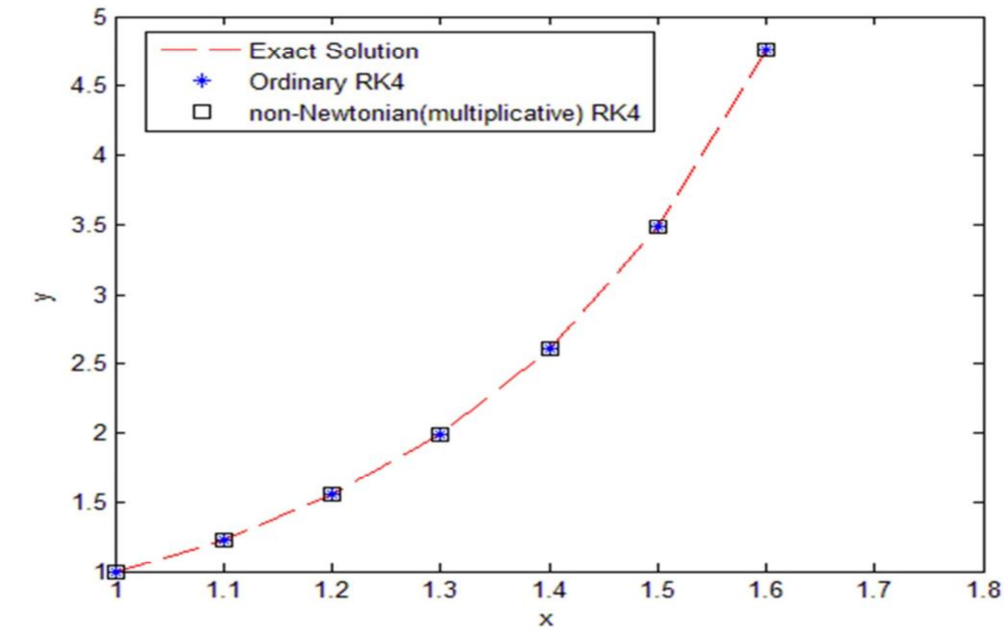

Figure 4.5: Graphs of non-Newtonian (multiplicative) and ordinary fourth-order Runge-Kutta methods and the exact solution for * = and =2

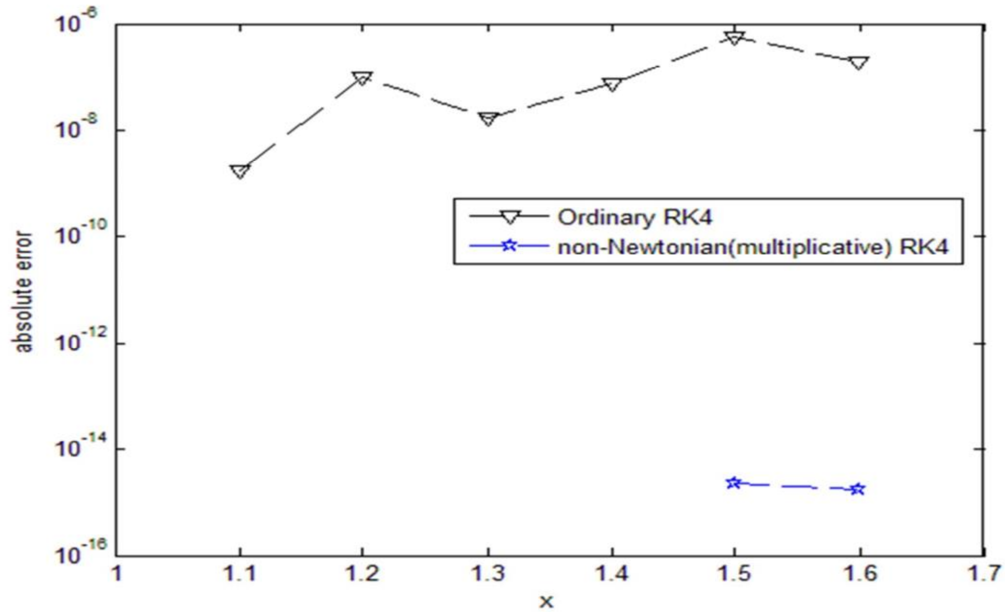

Figure 4.6: Approximation error containing the fourth-order non-Newtonian (multiplicative) and the ordinary Runge-Kutta methods of table 4.3

Since the figure 4.5 do not tell us much, it can be observed from the table 4.3 that the error terms of the fourth-order non-Newtonian (multiplicative) Runge-Kutta method is much smaller as compared to that of the ordinary fourth- order Runge-Kutta method. We can therefore say that the non-Newtonian (multiplicative) fourth-order Runge-Kutta method gives us better solutions.

We now consider the solution for non-Newtonian (multiplicative) initial value problem

$$^{*} \quad = \underline{\quad}, \quad 0 = 2 \qquad (4.4)$$

The ordinary differential equation corresponding to equation (4.4) is

$$= - , \quad 0 = 2 \qquad (4.5)$$

The exact solution for both equations (4.4) and (4.5) is

$$= + + 1 \qquad (4.6)$$

Let now check the difference between the non-Newtonian (multiplicative) and the ordinary Runge-Kutta method by comparing the results of the equations (4.4) and (4.5) for ,=8 points and h=0.5 (i.e., the step size). The results in tabular form for second-order non-Newtonian (multiplicative) Runge-Kutta method and the second-order ordinary Runge-Kutta method and the graphs are below.

	W ‡U^	U \H%	‰Š*PU 8Š%%\	‹ Œ=Ad• ‹‹ d, % ? ,?, –• •	‹ Œ=Ad• ‹‹ d, % ? ’’,B –a’AA=
0	2	2	2	0	0
0.5	3.17478214527	3.14872127070	3.12500000000	0.8277	0.7534
1.0	4.77881007482	4.71828182846	4.64062500000	1.2828	1.6459
1.5	7.09246039743	6.98168907034	6.79101562500	1.5866	2.7311
2.0	10.57646361810	10.38905609893	9.97290039063	1.8039	4.0057
2.5	15.98994535541	15.68249396070	14.83096313477	1.9605	5.4298
3.0	24.58436887273	24.08553692319	22.41281509399	2.0711	6.9449

3.5	38.42306811421	37.61545195869	34.42082452774	2.1470	8.4929
4.0	60.90797140470	59.59815003314	53.62133985758	2.1978	10.0285

Table 4.4: Comparison of the results of non-Newtonian (multiplicative) and ordinary second-order Runge-Kutta methods with the exact values and their relative errors in percentage

for * = ___ and = –

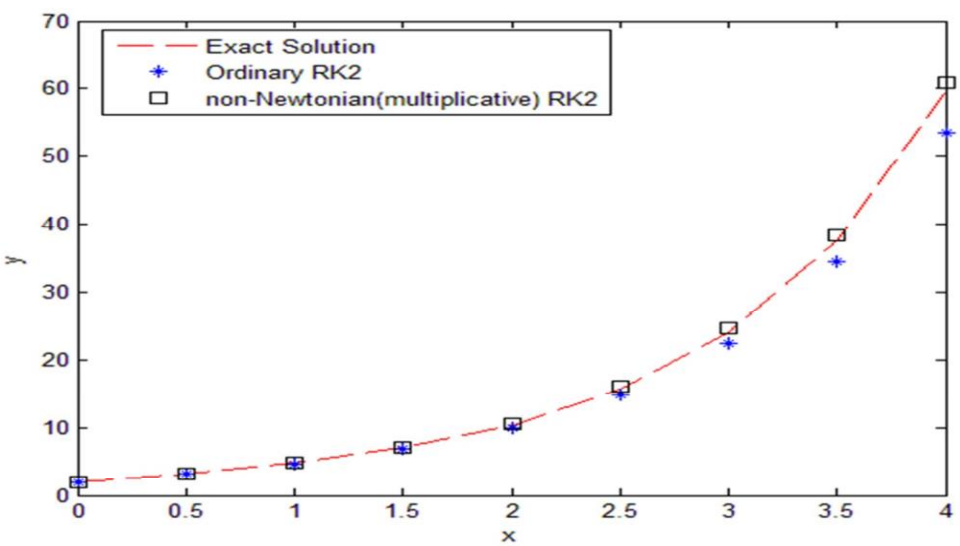

Figure 4.7: Graphs of non-Newtonian (multiplicative) and ordinary second-order Runge-Kutta

methods and the exact solution for for $y^{*}(x) = e^{\frac{y-x}{y}}$ and $y'(x) = y - x$

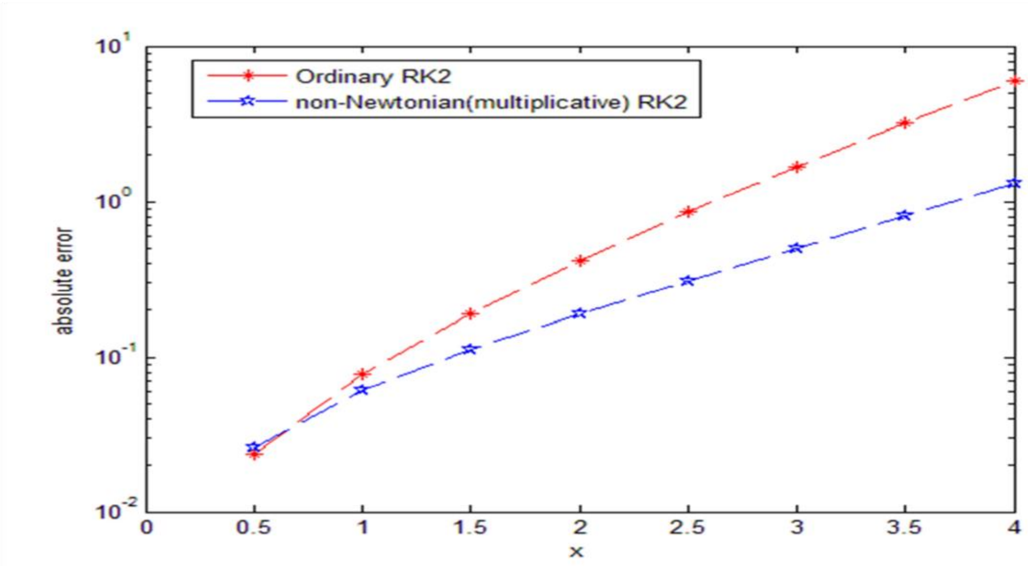

Figure 4.8: Approximation error containing the second-order non-Newtonian (multiplicative) and the ordinary Runge-Kutta methods of table 4.4

The comparison of the results presented in the table 4.4 shows that the non-Newtonian (multiplicative) second-order Runge-Kutta method gives better results compared to the ordinary second-order Runge-Kutta method. Since the error terms of non-Newtonian (multiplicative) Rung-Kutta method is smaller than that of the ordinary Runge-Kutta method.

The following table gives us the numerical results and the relative errors in percentage for the non-Newtonian(multiplicative) Runge-Kutta third-order method and the ordinary third- order Runge-Kutta method compared to the exact result. The graphs are also below.

	W ‡Uˆ	U \H%	‰Š*PU 8Š%%\	‹ Œ=Ad• ‹‹ d, % ? ,?, −• •	‹ Œ=Ad• ‹‹ d, % ? ‘’,B −a’AA=
0	2	2	2	0	0
0.5	3.06556077620	3.14872127070	3.14583333333	2.6411	0.0917

1.0	4.44201393774	4.71828182846	4.70876736111	5.8553	0.2017
1.5	6.33263140475	6.98168907034	6.95817961516	9.2966	0.3367
2.0	9.06347686742	10.38905609893	10.33742061662	12.7594	0.4970
2.5	13.16852684291	15.68249396070	15.57617143152	16.0304	0.6780
3.0	19.52766092692	24.08553692319	23.87536548104	18.9237	0.8726
3.5	29.59335417801	37.61545195869	37.21153902089	21.3266	1.0738
4.0	45.76350814946	59.59815003314	58.83774130521	23.2132	1.2759

Table 4.5a: Comparison of the results of non-Newtonian (multiplicative) and ordinary third-order Runge-Kutta methods with the exact values and their relative errors in percentage

for * = ___ and = –

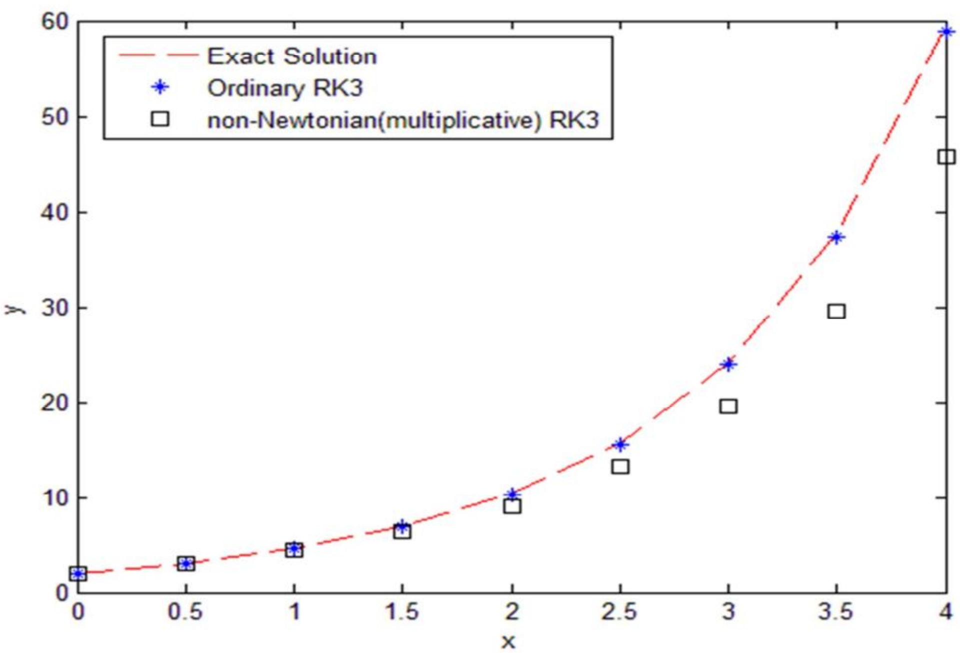

Figure 4.9a: Graphs of non-Newtonian (multiplicative) and ordinary third-order Runge-Kutta

methods and the exact solution for for $y^*(x) = e^{\frac{y-x}{y}}$ and $y'(x) = y - x$

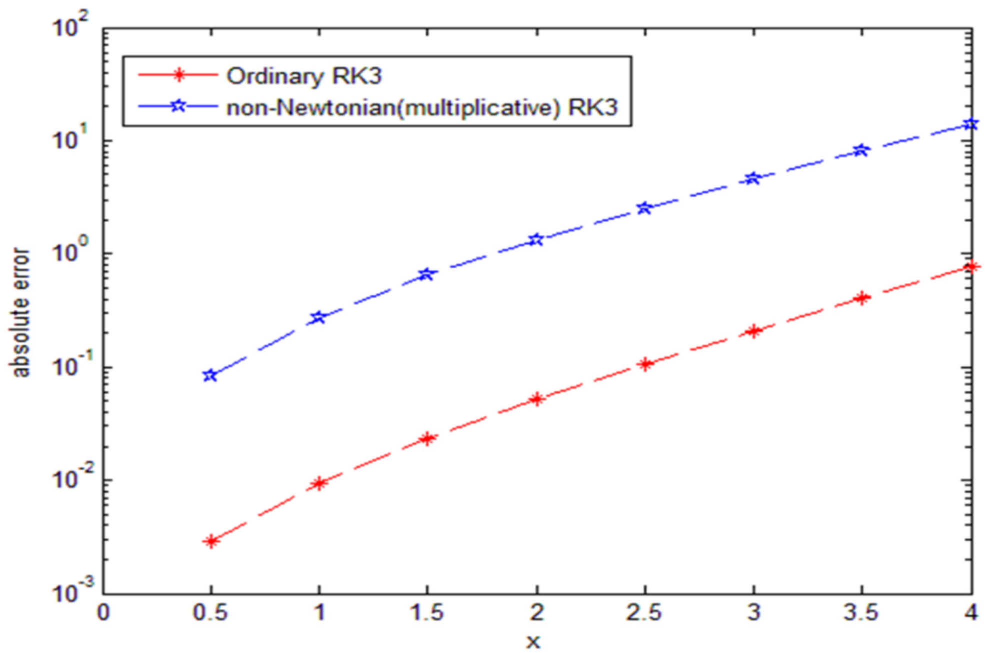

Figure 4.10a: Approximation error containing the second-order non-Newtonian (multiplicative) and the ordinary Runge-Kutta methods of table 4.5a

The comparison of the results in the table 4.5a shows that the ordinary third-order Runge-kutta method gives better results compared to the non-Newtonian (multiplicative) third-order Runge-Kutta method. Since the error terms of the ordinary Runge-Kutta methodis smaller than that of the non-Newtonian (multiplicative) Rung-Kutta method. This is also represented in figure 4.10a.

We once again set our parameters in the case of the non-Newtonian (multiplicative) third-order Rung-Kutta method in equation (3.103) to :=_ , = , ==_ , l=_ , l =1, m =−1, o g o

m =2, and m=_ which is the parameters widely used for the third-order ordinary Runge-Kutta method.

Table 4.5b below gives us the comparison of the results for the parameters chosen and also the graphs are shown in figures 4.9b and 4.10b.

	W ‡U˄	U\H%	‰Š*PU 8Š%%\	‹ Œ=Ad• ‹‹ d, % ? ,?, –• •	‹ Œ=Ad• ‹‹ d, % ? '',B –a'AA=
0	2	2	2	0	0
0.5	3.14800558240	3.14872127070	3.14583333333	0.0227	0.0917
1.0	4.71708394307	4.71828182846	4.70876736111	0.0254	0.2017
1.5	6.97977792898	6.98168907034	6.95817961516	0.0274	0.3367
2.0	10.38584525716	10.38905609893	10.33742061662	0.0309	0.4970
2.5	15.67696794504	15.68249396070	15.57617143152	0.0352	0.6780
3.0	24.07605221819	24.08553692319	23.87536548105	0.0394	0.8726
3.5	37.59936285780	37.61545195869	37.21153902089	0.0428	1.0738
4.0	59.57115847503	59.59815003314	58.83774130521	0.0453	1.2759

Table 4.5b: Comparison of the results of non-Newtonian (multiplicative) and ordinary third-order Runge-Kutta methods with the exact values and their relative errors in percentage

for * = ___ and = –

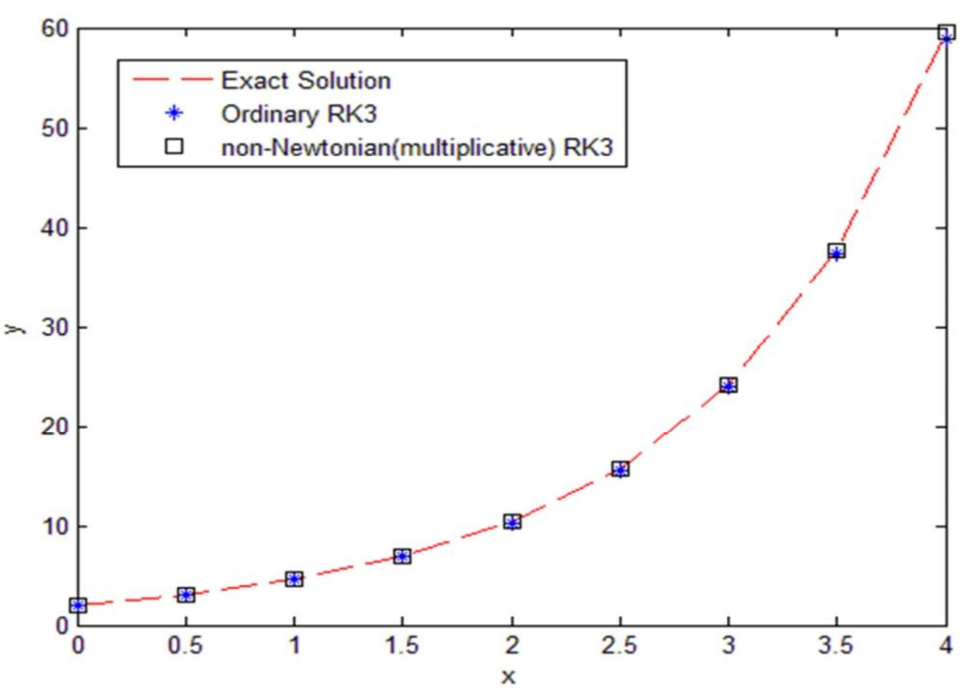

Figure 4.9b: Graphs of non-Newtonian (multiplicative) and ordinary third-order Runge-Kutta methods and the exact solution for for $y^*(x) = e^{\frac{y-x}{y}}$ and $y'(x) = y - x$

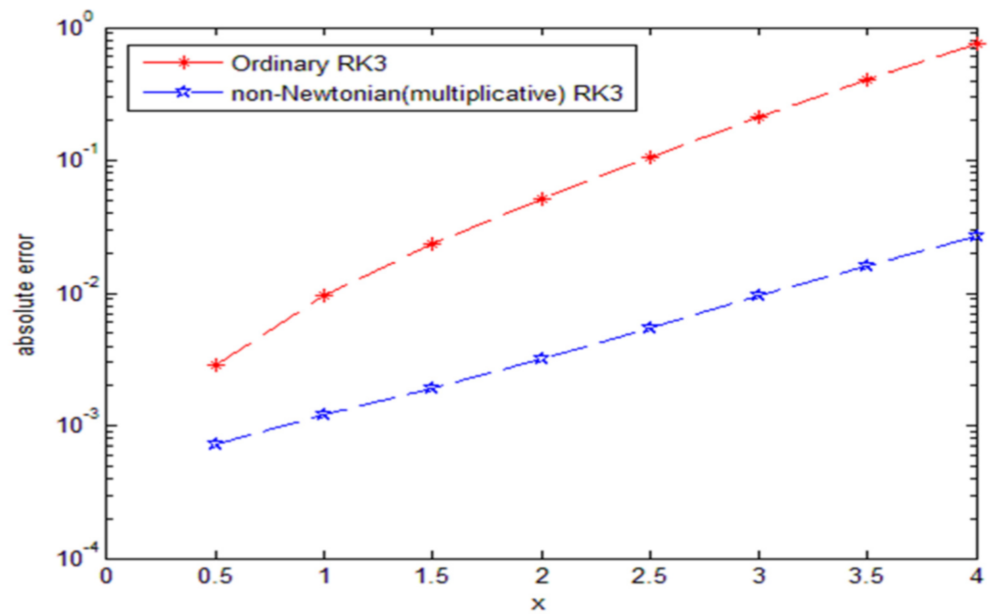

Figure 4.10b: Approximation error containing the third-order non-Newtonian (multiplicative) and the ordinary Runge-Kutta methods of table 4.5b

We can observe that the comparison of results in table 4.5b shows that the non-Newtonian (multiplicative) third-order Runge-Kutta method gives us better results as compared to the ordinary third-order Runge-Kutta method for the same parameters. This we can also observe in figure 4.10b.

The following table gives us the numerical results and the relative errors in percentage for the non-Newtonian(multiplicative) Runge-Kutta fourth-order method and the ordinary fourth- order Runge-Kutta method compared to the exact result. The graphs are also below.

	W ‡U^	U \H%	‰Š*PU 8Š%%\	‹ Œ=Ad• ‹‹ d, % ? ,?, −• •	‹ Œ=Ad• ‹‹ d, % ? ʺ,B −a'AA=

0	2	2	2	0	0
0.5	3.09766059727	3.14872127070	3.14843750000	1.6216	0.0090
1.0	4.52486653575	4.71828182846	4.71734619141	4.0993	0.0198
1.5	6.49779222100	6.98168907034	6.97937536240	6.9309	0.331
2.0	9.36333319349	10.38905609893	10.38397032395	9.8731	0.0490
2.5	13.68981076465	15.68249396070	15.67201358089	12.7064	0.0668
3.0	20.41382059959	24.08553692319	24.06480363724	15.2445	0.0861
3.5	31.08138908999	37.61545195869	37.57557474577	17.3707	0.1060
4.0	48.24459155490	59.59815003314	59.52301774498	19.0502	0.1261

Table 4.6a: Comparison of the results of non-Newtonian (multiplicative) and ordinary fourth-order Runge-Kutta methods with the exact values and their relative errors in percentage

for * = ___ and = −

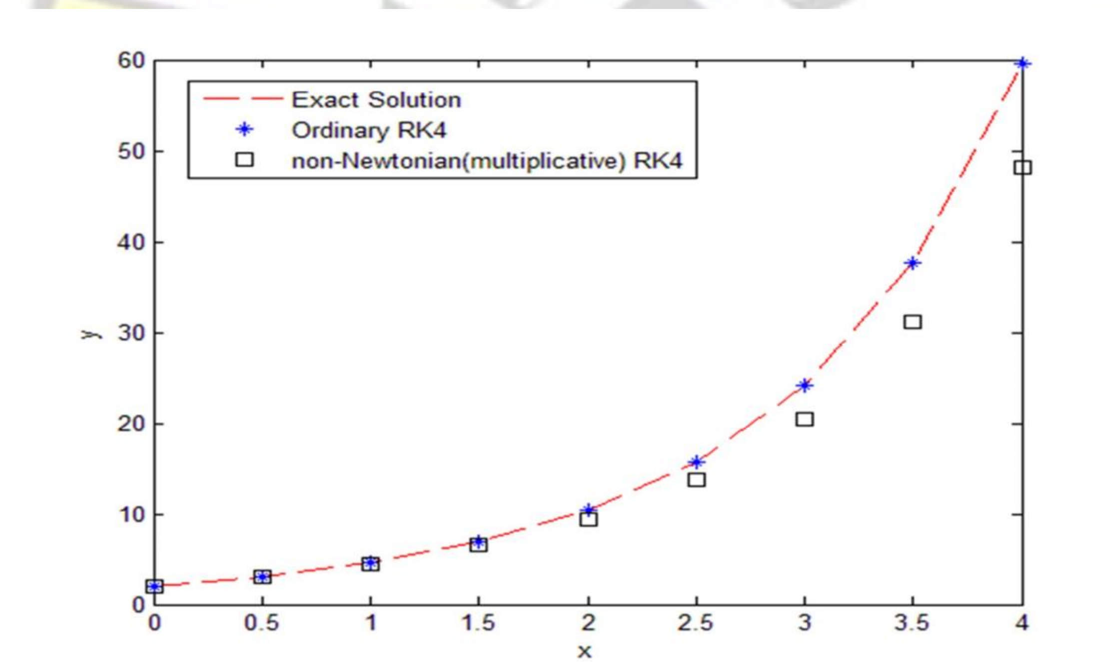

Figure 4.11a: Graphs of non-Newtonian (multiplicative) and ordinary fourth-order Runge-Kutta

methods and the exact solution for for $y^{*}(x) = e^{\frac{y-x}{y}}$ and $y'(x) = y - x$

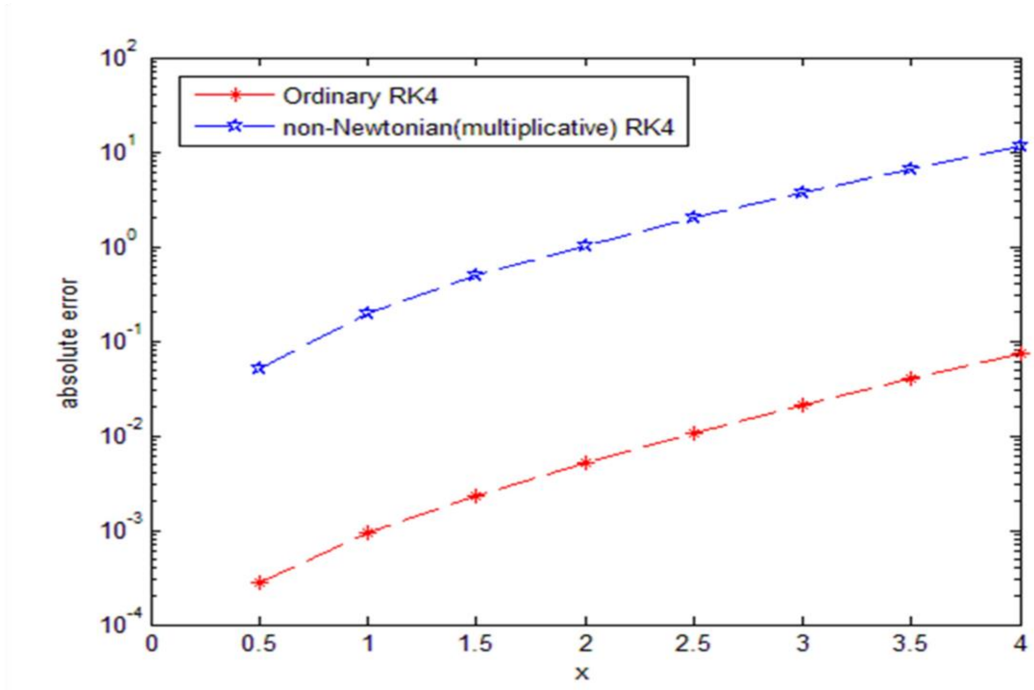

Figure 4.12a: Approximation error containing the fourth-order non-Newtonian (multiplicative) and the ordinary Runge-Kutta methods of table 4.6a

We can see from table 4.6a that the ordinary fourth-order Runge-Kutta method gives us better solutions than the non-Newtonian (multiplicative) Runge-Kutta method. This can also be clearly seen in figures 4.11a and 4.12a.

We now choose the parameters ...= , :=_ , >= , ==_ , l =1, l= , l =_ , m =
o g g o

m $=m_g=m_n=0$, $m_q=1$ and m=_ . Which is also the set of parameters widely used for the ordinary Runge-Kutta method and substitute into equation (3.125).

Table 4.6b below gives us the comparison of the results for the parameters chosen and also the graphs are presented in figures 4.11b and 4.12b.

	W ‡Uˆ	U \H%	‰Š*PU 8Š%%\	‹ Œ=Ad• ‹‹ d, % ? ,?, −• •	‹ Œ=Ad• ‹‹ d, % ? ‘’,B −a’AA=
0	2	2	2	0	0
0.5	3.14878031996	3.14872127070	3.14843750000	0.0019	0.0090
1.0	4.71842749578	4.71828182846	4.71734619141	0.0031	0.0198
1.5	6.98197000266	6.98168907034	6.97937536240	0.0040	0.0331
2.0	10.38954950716	10.38905609893	10.38397032395	0.0047	0.0490
2.5	15.68332726836	15.68249396070	15.67201358089	0.0053	0.0668
3.0	24.08692224948	24.085536923188	24.06480363724	0.0058	0.0861
3.5	37.61774103632	37.61545195869	37.57557474577	0.0061	0.1060
4.0	59.60192377380	59.59815003314	59.52301774498	0.0063	0.1261

Table 4.6b: Comparison of the results of non-Newtonian (multiplicative) and ordinary fourth-order Runge-Kutta methods with the exact values and their relative errors in percentage for * = ___ and = −

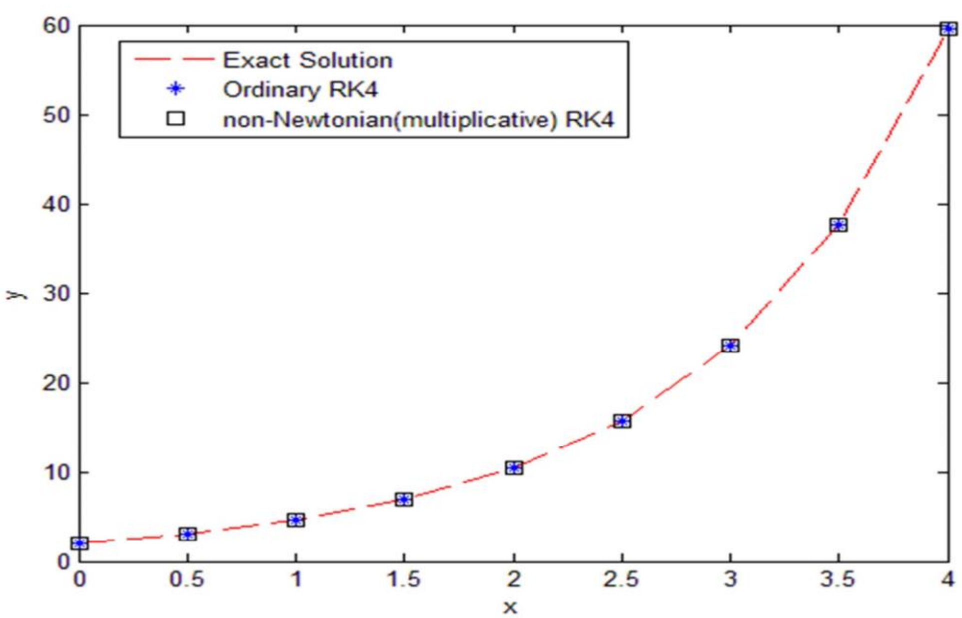

Figure 4.11b: Graphs of non-Newtonian (multiplicative) and ordinary fourth-order Runge-Kutta

methods and the exact solution for for $y^*(x) = e^{\frac{y-x}{y}}$ and $y'(x) = y - x$

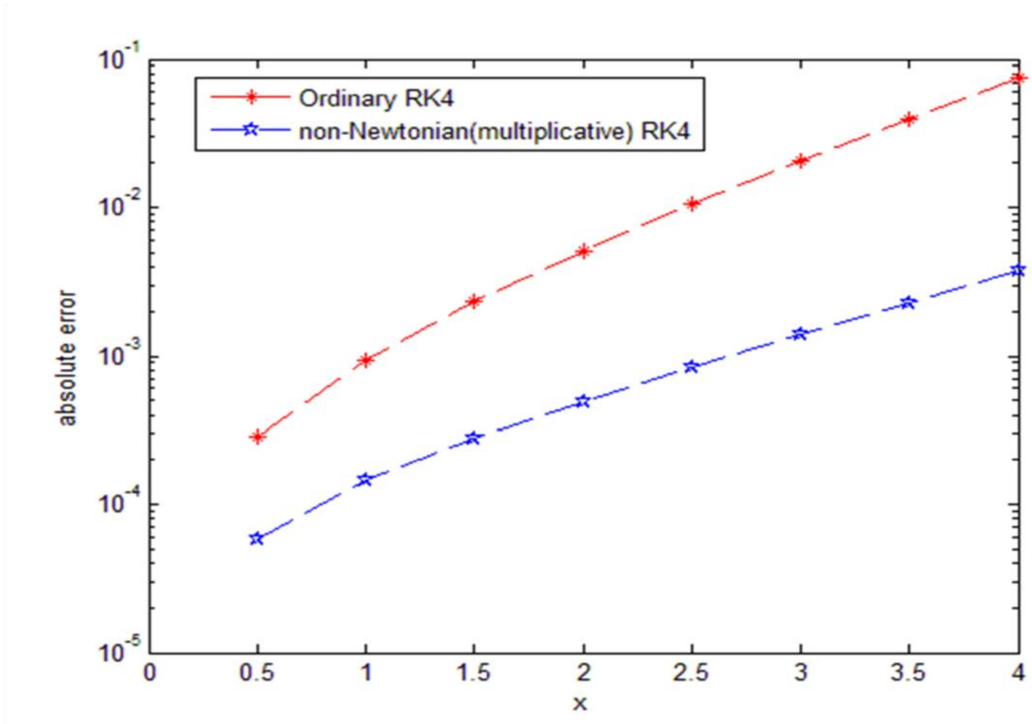

Figure 4.12b: Approximation error containing the fourth-order non-Newtonian (multiplicative) and the ordinary Runge-Kutta methods of table 4.6b

From table 4.6b we observe that the change of the parameters for the non-Newtonian (multiplicative) Runge-Kutta method gave us better results as compared to the ordinary Runge-Kutta method for the same parameters. This can also be notice in figure 4.12b. Thus the non-Newtonian (multiplicative) method gives us better solutions than the ordinary Runge-Kutta method.

We finally consider the solution for non-Newtonian (multiplicative) initial value problem

$$^{*} \quad =, \quad 0 = 1 \qquad (4.7)$$

The ordinary differential equation corresponding to equation (4.7) is

$$= 1 - \quad , \quad 0 = 1 \qquad (4.8)$$

The exact solution for both equations (4.7) and (4.8) is

$$= Q \quad \frac{R}{RS} \tag{4.9}$$

Using h=0.25 and ,=8, we now check the solutions of non-Newtonian (multiplicative) and the ordinary Runge-Kutta method by comparing the results of the equations (4.7) and (4.8). The following table compares the results of the non-Newtonian (multiplicative) Runge-Kutta method and the ordinary Runge-Kutta method of the second order. The graphs are also presented below.

	W ‡U^	U \H%	‰Š*PU 8Š%%\	‹ Œ=Ad• ‹‹ d, % ? ,?, –• •	‹ Œ=Ad• ‹‹ d, % ? '',B –a'AA=
0	1	1	1	0	0
0.25	1.24452010777	1.24452010777	1.24609375000	0	0.1264
0.50	1.45499141462	1.45499141464	1.45904922485	0	0.2789
0.75	1.59799544995	1.59799544995	1.60438420624	0	0.3998
1.00	1.64872127070	1.64872127070	1.65608799413	0	0.4468
1.25	1.59799544995	1.59799544995	1.60433524434	0	0.3967
1.50	1.45499141462	1.45499141462	1.458629016072	6.104 ^{n}	0.2500
1.75	1.24452010777	1.24452010777	1.24496265630	0	0.0356
2.00	1.00000000000	1.00000000000	0.99815853596	0	0.1841

Table 4.7: Comparison of the results of non-Newtonian (multiplicative) and ordinary second-order Runge-Kutta methods with the exact values and their relative errors in percentage for * = and = 1−

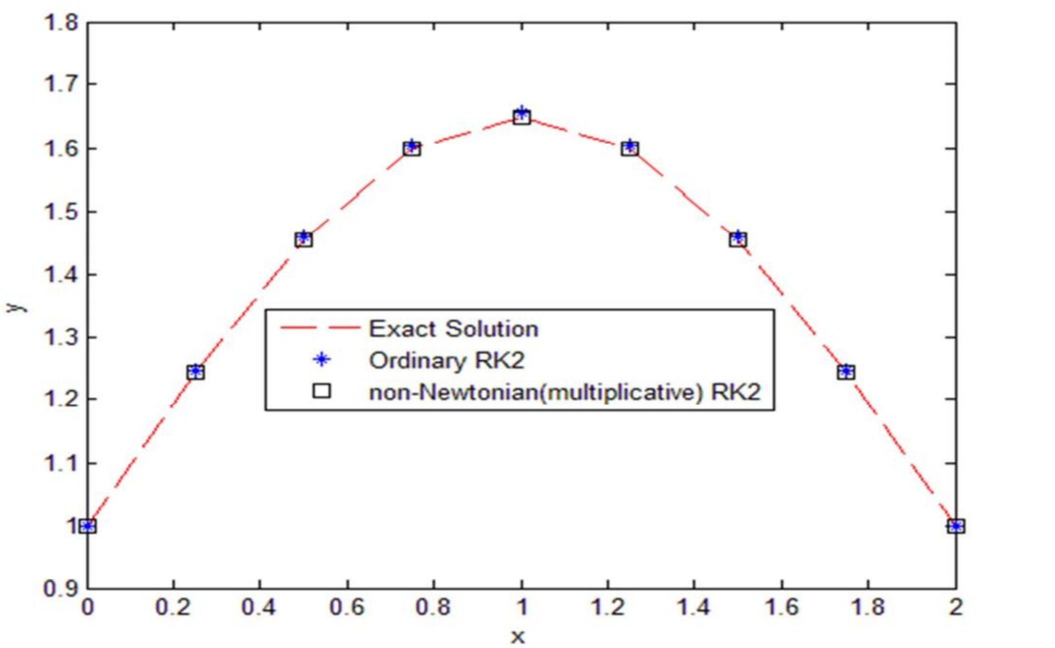

Figure 4.13: Graphs of non-Newtonian (multiplicative) and ordinary second-order Runge-Kutta methods and the exact solution for * = and = 1−

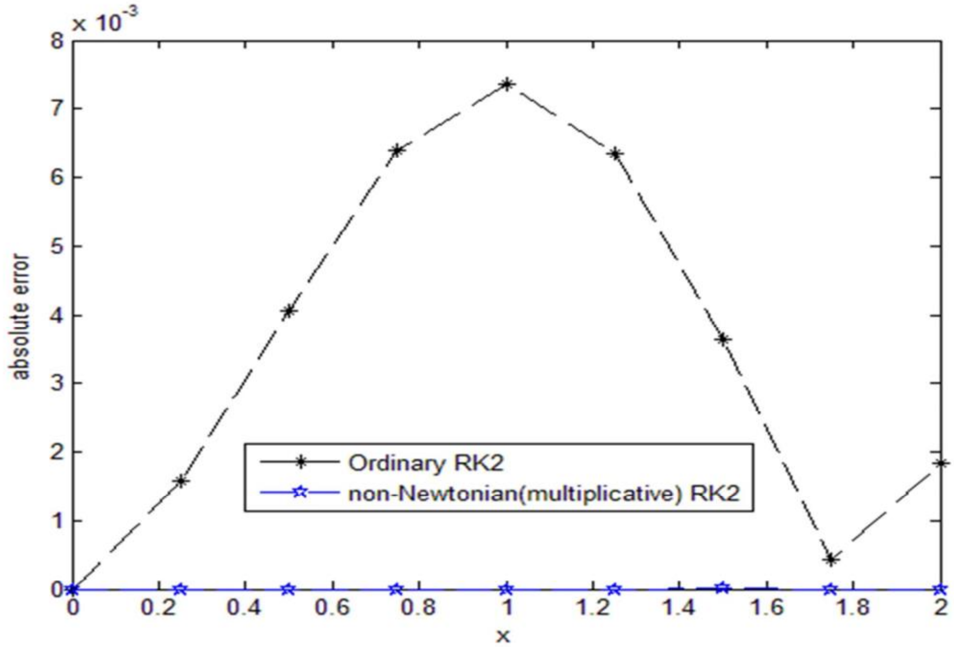

Figure 4.14: Approximation error containing the second-order non-Newtonian (multiplicative) and the ordinary Runge-Kutta methods of table 4.7

The comparison of the results presented in the table 4.7 shows that the non-Newtonian (multiplicative) second-order Runge-Kutta method gives better results compared to the ordinary second-order Runge-Kutta method. Since the error terms of non-Newtonian (multiplicative) Rung-Kutta method is smaller than that of the ordinary Runge-Kutta method.

In the following table we compare the results of the non-Newtonian (multiplicative) and ordinary third-order Runge-Kutta methods. The results in tabular form and the graphs are as follows

	W ‡U^	U \H%	‰Š*PU 8Š%%\	‹ Œ=Ad• ‹‹ d, % ? ,?, –• •	‹ Œ=Ad• ‹‹ d, % ? ‘’,B –a’AA=
0	1	1	1	0	0
0.25	1.24452010777	1.24452010777	1.24454752604	0	0.0022
0.50	1.45499141462	1.45499141462	1.45516248213	0	0.0118
0.75	1.59799544995	1.59799544995	1.59839319682	0	0.0249
1.00	1.64872127070	1.64872127070	1.64938360479	0	0.0402
1.25	1.59799544995	1.59799544995	1.59891418459	0	0.0575
1.50	1.45499141462	1.45499141462	1.45610751812	0	0.0767
1.75	1.24452010777	1.24452010777	1.24571372756	0	0.0959
2.00	1.00000000000	1.00000000000	1.00109237139	0	0.1092

Table 4.8: Comparison of the results of non-Newtonian (multiplicative) and ordinary third- order Runge-Kutta methods with the exact values and their relative errors in percentage

for * = and = 1–

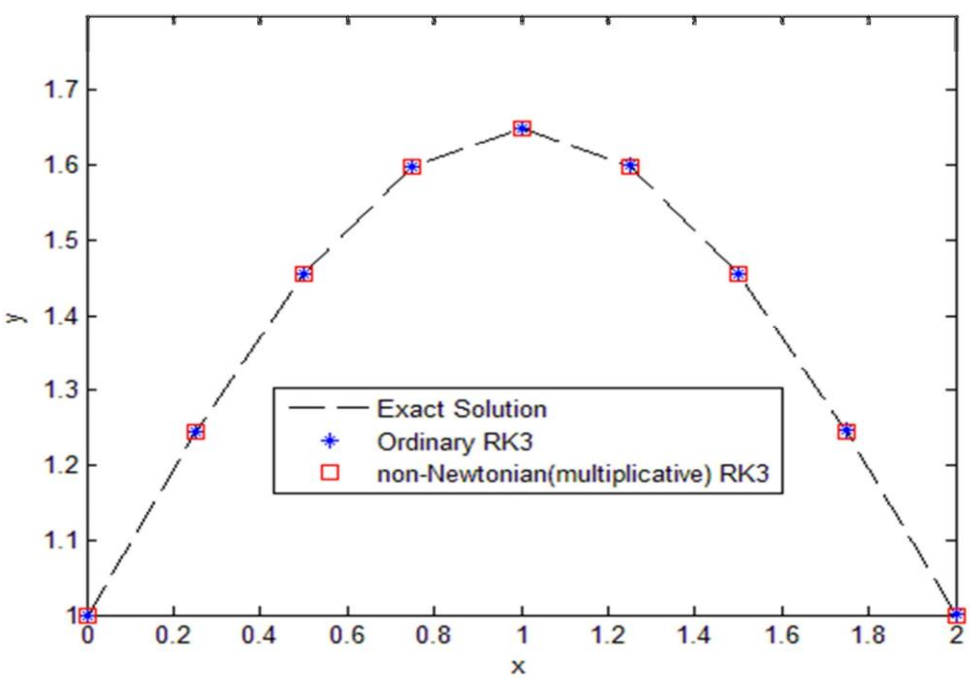

Figure 4.15: Graphs of non-Newtonian (multiplicative) and ordinary third-order Runge-Kutta

methods and the exact solution for * = and = 1−

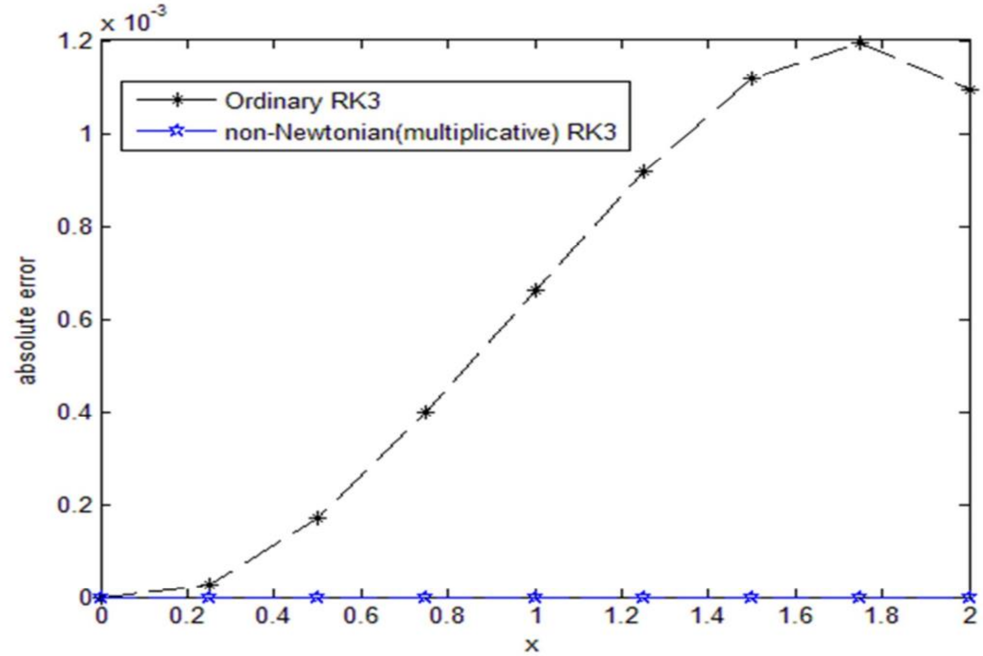

Figure 4.16: Approximation error containing the third-order non-Newtonian (multiplicative) and the ordinary Runge-Kutta methods of table 4.8

By comparing the error terms of the non-Newtonian (multiplicative) third-order Runge-Kutta method and the ordinary third-order Runge-Kutta method of table 4.8, we can observe that the non-Newtonian (multiplicative) Runge-Kutta method gives better results than the ordinary Runge-Kutta method.

The following table compares the results of the non-Newtonian (multiplicative) and ordinary fourth-order Runge-Kutta methods. Below are the results in tabular form and the graphs.

	W ‡Uˆ	U \H%	‰Š*PU 8Š%%\	‹ Œ=Ad• ‹‹ d, % ? ,?, −• •	‹ Œ=Ad• ‹‹ d, % ? ’’,B −a’AA=
0	1	1	1	0	0
0.25	1.24452010777	1.24452010777	1.24451382955	0	5.045 ^{n}
0.50	1.45499141462	1.45499141462	1.45498184673	0	6.576 ^{n}
0.75	1.59799544995	1.59799544995	1.59798433214	0	6.957 ^{n}
1.00	1.64872127070	1.64872127070	1.64870973608	0	6.996 ^{n}
1.25	1.59799544995	1.59799544995	1.59798420510	0	7.037 ^{n}
1.50	1.45499141462	1.45499141462	1.45498069954	0	7.364 ^{n}
1.75	1.24452010777	1.24452010777	1.24451059773	0	7.642 ^{n}
2.00	1.00000000000	1.00000000000	0.99999494634	0	5.054 ^{n}

Table 4.9: Comparison of the results of non-Newtonian (multiplicative) and ordinary fourth-order Runge-Kutta methods with the exact values and their relative errors in percentage for * = and = 1−

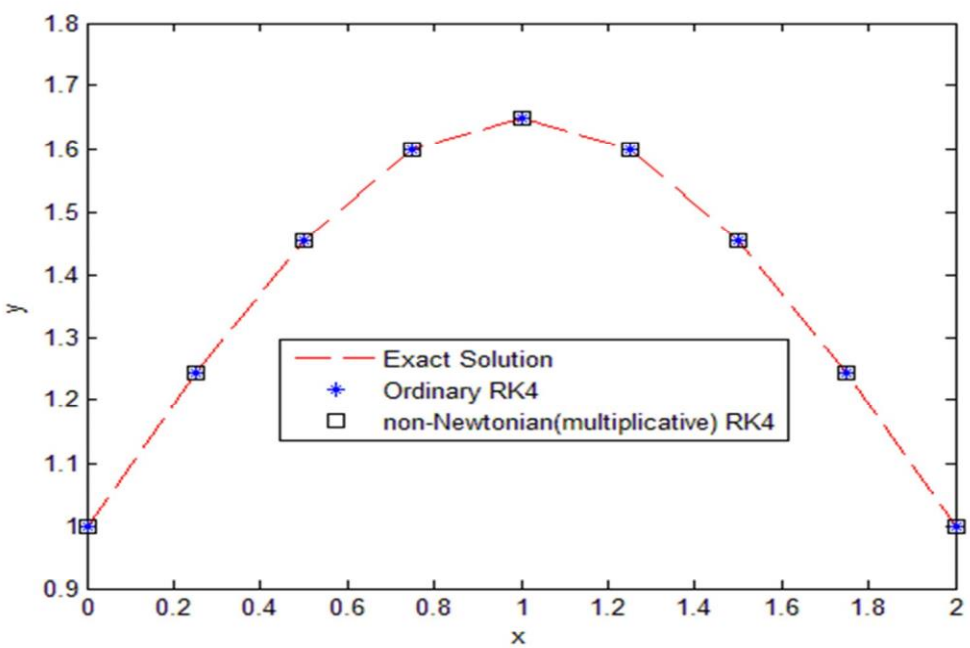

Figure 4.17: Graphs of non-Newtonian (multiplicative) and ordinary fourth-order Runge-Kutta

methods and the exact solution for $*$ = and = 1−

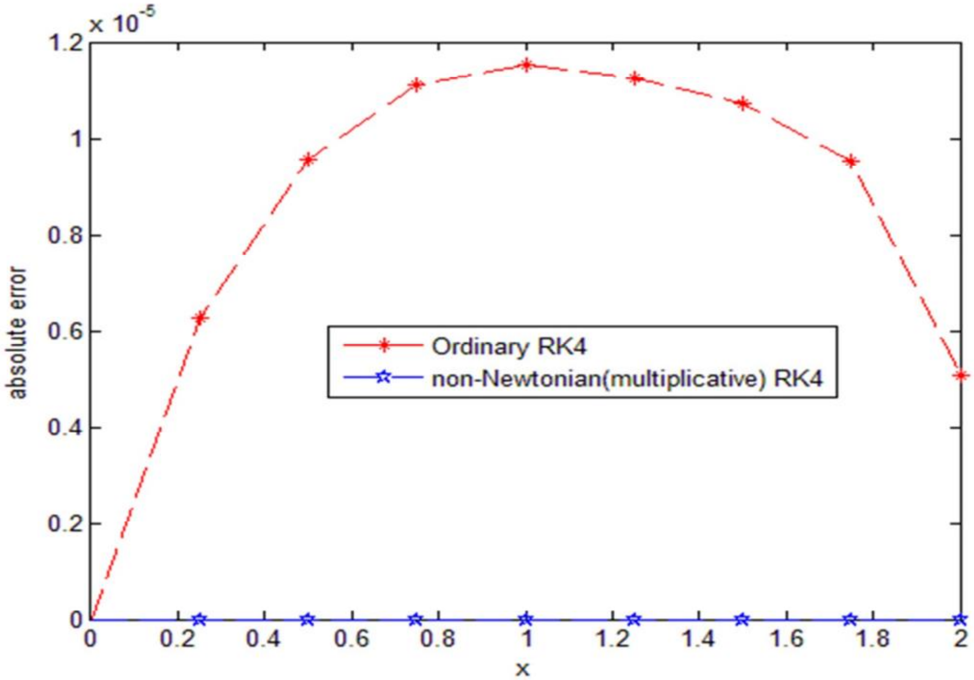

Figure 4.18: Approximation error containing the fourth-order non-Newtonian (multiplicative) and the ordinary Runge-Kutta methods of table 4.9

Comparing the results and error terms of table 4.9, we see that the non-Newtonian (multiplicative) Runge-Kutta method gives us better solutions than the ordinary Runge-Kutta method.

CHAPTER FIVE

CONCLUSION AND RECOMMENDATION

5.1 Conclusion

In this thesis, a new kind of calculus called the non-Newtonian (Multiplicative) calculus is discussed. The non-Newtonian (Multiplicative) derivative definition is given in terms of non-Newtonian (Multiplicative) calculus and on the basis of the definition of the derivative, the definition of both the non-Newtonian (Multiplicative) Taylor series and chain rule were defined. With the definitions, we derived the non-Newtonian (Multiplicative) Runge-Kutta methods of the 2nd order, 3rd order and 4th order for the solution of non-Newtonian (Multiplicative) initial value problems of the form

* = , , = , where the starting point is and the initial value is . The derivation of the 2nd, 3rd and 4th order ordinary Runge-Kutta methods were also discussed. The non-Newtonian (Multiplicative) Runge-Kutta method and the ordinary Runge-Kutta method were both tested on some differential equations and the numerical results compared. We observed that the error terms of the non- Newtonian (Multiplicative) Runge-Kutta method in most cases are significantly less compared to the ones of the ordinary Runge-Kutta method. Furthermore, we also observed that different parameters were used depending on the problem to arrive at the desired results. At the end it was observe that the non-Newtonian (Multiplicative) Runge-Kutta method gave better results as compared to the ordinary Runge-Kutta method.

5.2 Recommendation

It is recommend that, future project must be carried out on the family of problems where the non-Newtonian (Multiplicative) Runge-Kutta method gives better results to the ordinary Runge-Kutta method.

REFERENCES

[1] Michael Grossman and Robert Katz. Non-Newtonian calculus. Lee Press, Pigeon Cove, Mass., 1972.

[2] Michael Grossman. Bigeometric calculus. Archimedes Foundation, Rockport, Mass., 1983. A system with a scale-free derivative.

[3] A. Bashirov, E. Kurpınar, A.Özyapıcı, Multiplicative calculus and its applications, Journal of Mathematical Analysis and Applications 337 (1) (2008) 36-48. [4] Dorota Aniszewska. Multiplicative runge–kutta methods. Nonlinear Dynamics, 50(1-2):265–272, October 2007.

[5] A. Bashirov and M. Riza. Complex multiplicative calculus. arXiv preprint arXiv:1103.1462, 2011

[6] E. Mısırlı, Y. Gurefe, Multiplicative adams bashforth-moulton methods, Numerical Algorithms 57 (4) (2011) 425-439.

[7] A. E. Bashirov, M. Riza, On complex multiplicative differentiation,TWMS Journal of Applied and Engineering Mathematics 1 (1) (2011) 75-85.

[8] Agamirza E. Bashirov, Emine Mısırlı, Yücel Tandoğdu, Ali Özyapıcı, On modeling with multiplicative differential equations, Appl. Math. J. Chinese Univ. 2011, 26(4): 425-438. [9] M. Riza and H. Aktöre. The Runge-Kutta Method in Geometric Multiplicative Calculus. ArXiv e-prints 1311.6108, November 2013.

[10] M. Riza, A. Özyapıcı, and E. Mısırlı. Multiplicative finite difference methods. Quarterly of AppliedMathematics, 67(4):745, 2009.

[11] Dorota Aniszewska and Marek Rybaczuk. Chaos in multiplicative systems. In Chaotic systems, pages 9–16. World Sci. Publ., Hackensack, NJ, 2010.

[12] J. Stoer and R. Bulirsch. Introduction to numerical analysis, Springer-Verlag,

New York, third edition, 2002.
[13] S.R. Otto and J.P. Denier, An Introduction to Programming and Numerical Methods in MATLAB, Springer-Verlag London Limited 2005.

[14] Andrew Knight, BASICS OF MATLAB and Beyond, CRC Press LLC 2000. ISBN 0-8493-2039-9.

[15] Richard Hamming. Numerical Methods for Scientists and Engineers . Dover, New Y ork, 1987. ISBN 0486652416.

[16] Hatice Aktöre, " Multiplicative Runge-Kutta Methods," Eastern Mediterranean University Gazimağusa, North Cyprus, February 2011.

[17] Shampine, L. F., and M. K. Gordon, Computer Solution of Ordinary Differential Equation: the Initial Value Problem, W. H. Freeman and Company, San Francisco, 1975.

[18] Luc Florack and Hans van Assen, Multiplicative calculus in biomedical image analysis. Journal of Mathematical Imaging and Vision, 2011.

[19] Campbell, D. (1999), 'Multiplicative calculus and student projects',Primus IX, 327–333.

[20] Diana Andrada Filip and Cyrille Piatecki, " A non-newtonian examination of the theory of Exogenous economic growth", 2010.

[21] http://www.wikipedia//Runge-Kutta methods-Wikipedia, the free encyclopedia.htm

APPENDICES

The following MATLAB codes make , steps from to to approximate a solution of the

Initial value problem = , , = and * = , , = .

For each ordinary differential and non-Newtonian (multiplicative) initial value problem, we need only define the appropriate function , in the file **f.m** [13], [14].

APPENDIX I: Second-Order Ordinary Runge-Kutta

```
% rungekutta2.m          function [X2,Y2]
= rungekutta2(x, xf, y, n)
     h = (xf – x) / n;
     X2 = x; Y2 = y;
     for  i = 1: n
k1 = f(x,y);          k2 =
f(x+h/2, y+h*k1/2);          k =
k2;          y = y+h*k;
x = x+h;
     X2 = [X2; x];
Y2 = [Y2; y];                              end
```

APPENDIX II:

Third-Order Ordinary Runge-Kutta

```
   % rungekutta3.m
   function [X3,Y3] = rungekutta3(x, xf, y, n)
h = (xf – x) / n;          X3 = x; Y3 = y;
for  i = 1: n          k1 = f(x,y);          k2
= f(x+h/2, y+h*k1/2);          k3 = f(x+h, y-
h*k1+2*h*k2);          k = (k1+4*k2+k3) /
6;          y = y+h*k;          x = x+h;
     X3 = [X3; x];
Y3 = [Y3; y];
   end
   %%%%%%%%%%%%%%%%%%%%
```

% f.m

function y(x)p = f(x,y)   y(x)p

= 2*x*y;   % ( )l = ( ) **APPENDIX III:**

Fourth-Order Ordinary Runge-Kutta

% rungekutta4.m

Function [X4,Y4] = rungekutta4(x, xf, y, n)

h = (xf – x) / n;   X4 = x;

Y4 = y;   for i = 1: n

k1 = f(x,y);   k2 = f(x+h/2,

y+h*k1/2);   k3 = f(x+h/2,

y+h*k2/2);   k4 = f(x+h,

y+h*k3);   k =

(k1+2*k2+2*k3+k4) / 6;   y =

y+h*k;   x = x+h;

X4 = [X4; x];

Y4 = [Y4; y];   end

%%%%%%%%%%%%%%%%%%%%%%

% f.m

function y(x)p = f(x,y)   y(x)p

= y - x;   % ( )l = ( ) **APPENDIX IV:**

Second-Order non-Newtonian

(Multiplicative) Runge-Kutta

% nonnewrk2.m   function [X2,Y2]

= nonnewrk2(x, xf, y, n)

```
h = (xf – x) / n;
X2 = x; Y2 = y;          for  i = 1:
n                k1 = f(x,y);
k2 = f(x+h, y*k1.^(h));
               k = (k1.^(1/2)*k2.^(1/2));
               y = y.*(k.^h);
x = x+h;
              X2 = [X2; x];
Y2 = [Y2; y];          end
```

APPENDIX V: Third-Order non-Newtonian (Multiplicative) Runge-Kutta

```
% nonnewrk3.m          function [X3,Y3] =
nonnewrk3(x, xf, y, n)
     h = (xf – x) / n;
X3 = x; Y3 = y;          for
i = 1: n               k1 =
f(x,y);               k2 =
f(x+h/2, y);
```

```
k3=f(x+h,y);
k = (k1.^(1/6)*k2.^(1/2)*k3.^(1/3));
y = y.*(k.^h);
x = x+h;
X3 = [X3; x];
Y3 = [Y3; y];
end
%%%%%%%%%%%%%%%%%%%%
% f.m
function y(x)p = f(x,y)        y(x)p =
exp(2*x);   % ( )l = *( )
```

APPENDIX VI: Third-Order non-Newtonian (Multiplicative) Runge-Kutta

```
% nonnewrk3.m
function[X3,Y3]=nonnewrk3(x,xf,y,n)
h=(xf-x)/n;        X3=x;
Y3=y;    for i=1:n        k1=f(x,y);
k2=f(x+h/2,y.*(k1.^(h/2)));
k3=f(x+h,y.*(k1.^(-h))*(k2.^(2.*h)));
k=(k1.^(1/6)*k2.^(2/3)*k3.^(1/6));
y=y.*(k.^h);        x=x+h;
X3=[X3;x];
Y3=[Y3;y];     end
%%%%%%%%%%%%%%%%%%%%
```

% f.m

function y(x)p = f(x,y) y(x)p = exp(1-x); % ( )I = *( ) **APPENDIX VII: Fourth-Order non-Newtonian (Multiplicative) Runge-Kutta**

% nonnewrk4.m function [X4,Y4] = nonnewrk4(x, xf, y, n)

h = (xf – x) / n;

X4 = x; Y4 = y; for i = 1: n k1 = f(x,y); k2 = f(x+h/2, y);

k3 = f(x+h/2,y);

k4 = f(x+h,y);

k = (k1.^(1/4)*k2.^(1/6)*k3.^(1/3)*k4.^(1/4));

y = y.*(k.^h);

x = x+h;

X4 = [X4; x];

Y4 = [Y4; y];

end

%%%%%%%%%%%%%%%%%%

% f.m

function y(x)p = f(x,y) y(x)p = exp((y - x) / y); % ( )I = *( ) **APPENDIX VIII:**

Fourth-Order non-Newtonian (Multiplicative)

Runge-Kutta

```
    % nonnewrk4.m           function [X4,Y4] =
nonnewrk4(x, xf, y, n)
        h = (xf – x) / n;
X4 = x; Y4 = y;
for  i = 1: n
k1 = f(x,y);
            k2=f(x+h/2,y.*(k1.^(h/2)));
            k3=f(x+h/2,y.*(k2.^(h/2)));
            k4=f(x+h,y*(k3.^h));
           k = (k1.^(1/6)*k2.^(1/3)*k3.^(1/3)*k4.^(1/6));
             y =
y.*(k.^h);              x =
x+h;
           X4 = [X4; x];
Y4 = [Y4; y];          end
%%%%%%%%%%%%%%%%%%%%
        % f.m
         function y(x)p = f(x,y)                     y(x)p =
exp( (y - x) / y );    % ( )l = *( )
```