A NEW CLASS OF EXPLICIT RUNGE-KUTTA METHOD FOR INTEGRATION OF INITIAL VALUE PROBLEMS IN ORDINARY DIFFERENTIAL EQUATIONS

BY

ROTIFA EBUN EBENEZER

IMC/99/4084

A THESIS IN THE DEPARTMENT OF MATHEMATICAL SCIENCES SUBMITTED TO THE SCHOOL OF POSTGRADUATE STUDIES TOWARDS A PARTIAL FULFILMENT OF THE REQUIREMENT FOR THE AWARD OF THE DEGREE OF MASTERS OF TECHNOLOGY (M. TECH.) IN MATHEMATICS OF FEDERAL UNIVERSITY OF TECHNOLOGY, AKURE, NIGERIA.

JANUARY, 2007

CERTIFICATION

This work has not been presented elsewhere for the award of a degree, or any other purpose.

Candidate's Name: ROTIFA, EBUN EBENEZER

Signature:---------- **Date:** 23/01/07

We hereby certify that this work been carried out by **Rotifa** Ebun Ebenezer and submitted to the Department of Mathematical Sciences of the Federal University of Technology, Akure, in partial fulfillment of the award of M. Tech. Degree in Mathematics. To the best of our knowledge it has not been submitted elsewhere for the award of a degree.

Major Supervisor's Name: Dr. R.A. Ademiluyi

Signature:---------- **Date:** 12-02-07

Co-supervisor's Name: Prof. D.O. Awoyemi

Signature:---------- **Date:**----------

DEDICATION

This research work is dedicated to the Almighty God.

ACKNOWLEDGEMENT

There has been a burning desire to acquire more knowledge in the field of science and technology. This desire would not have come to reality but for the encouragement offered me by my wife, Mrs. Rotifa Oluwafunmilayo.

Also, I wish to express my profound gratitude to Dr. R.A. Ademiluyi, my able lecturer and supervisor who enriched me with relevant books and journals. He always sacrificed his precious time to read through the project and make valuable corrections and suggestions.

My special thanks go to the head of department, Prof. S.T. Oni, Prof. D.O. Awoyemi my co-supervisor and other lecturers in the department for their instructions and guidance which contributed in no small measure to the success of this project.

I would not forget my children Oluwaseun, Oluwayomi who prayerfully and morally stood by me to the completion of this programme. Finally, I am grateful to God for seeing me through.

TABLE OF CONTENTS

		PAGES
Certifications		i.
Dedication		ii.
Acknowledgement		iii.
Table of Content		iv.
List of Figures		vii.
List of Tables		viii.
Abstract		xi.
CHAPTER ONE		
	Introduction	
1.1	Preamble	1
1.2	Existing One-Step methods for solution of ODES	3
1.3	Problems associated with one-step methods	3
1.4	Motivation	4
1.5	Research Objectives	4
1.6	Research Methodology	5
1.7	Expected Contribution	5
CHAPTER TWO		
	Literature Review	

2.1 Basic Concepts and Principles 7

2.1.1 The Steplength or Mesh Size 7

Finite difference scheme 8

2.1.2 Principles of One-step Schemes 8

(iii) One-step iteration Schemes

(iv) Examples of one-step schemes

2.1.3 Properties of One-Step Schemes 11

CHAPTER THREE

The Proposed Schemes 15

3.1 Derivation of the Schemes 16

3.1.1 One-Stage Schemes 18

3.1.2 Two-Stage Schemes 22

3.1.3 Three-Stage Schemes 26

CHAPTER FOUR

Basic Properties of the New Schemes 32

4.1 Order of Accuracy and Error Constant of the Schemes 35

4.2 Consistency 37

4.3 Convergence of the Method 40

4.3.1 One-Stage Scheme 40

4.3.2 General Case 42

4.4 Absolute Stability Properties of the Method 44

CHAPTER FIVE

	Application and Numerical Results	52
5.1	Algorithm of the Method	52
5.2	Program Flow Chart	55
5.3	Programming Implementation	58
5.4	Numerical Experiment and Results	58
5.5	Discussion of Results	60

CHAPTER SIX

6.1	Summary	77
6.2	Limitations	77
6.3	Recommendations	78
6.4	Contribution to Knowledge	78
	References	79
	Appendix	81

LIST OF FIGURES

		PAGES
Figure 4:	A-Stability region of Two-Step Schemes	51
Figure 5.1:	Flow Chart of the Implementation	57

LIST OF TABLES

		PAGES
Table 1:	Numerical Solution of Problem 1 by One-Stage Scheme with $\lambda = -10$	61
Table 2:	Numerical Solution of Problem 1 by One-Stage Scheme with $\lambda = -100$	62
Table 3:	Numerical Solution of Problem 1 by Two-Stage Scheme with $\lambda = -10$	63
Table 4:	Numerical Solution of Problem 1 by Three-Stage Scheme with $\lambda = -100$	64
Table 5:	Numerical Solution of Problem 1 by Three-Stage Scheme With $\lambda = -1000$	65
Table 6:	Numerical Solution of Problem 2 by One-Stage Scheme	66
Table 7:	Numerical Solution of Problem 2 by Two-Stage Scheme	67
Table 8:	Numerical Solution of Problem 2 by Three-Stage Scheme	68
Table 9:	Numerical Solution of Problem 3 by One-Stage Scheme	69
Table 10:	Numerical Solution of Problem 3 by Two-Stage Scheme	70
Table 11:	Numerical Solution of Problem 3 by Three-Stage Scheme	71
Table 12:	Numerical Solution of Problem 1 by One-Stage Scheme Using Richardson Error estimate technique	72

Table 13: Numerical Solution of Problem 1 by Two-Stage Scheme Using Richardson Error estimate technique 73

Table 14: Numerical Solution of Problem 1 by Three-Stage Scheme Using Richardson Error estimate technique 73

Table 15: Comparative analysis of the proposed Scheme and classical Runge-Kutta Method on Problem 2. 75

Table 16: Comparative analysis of the proposed Scheme and Classical Runge-Kutta Method on Problem 3. 76

ABSTRACT

In this thesis, a new class of explicit Runge-Kutta Schemes are developed to solve non-stiff and stiff initial value problems in ordinary differential equations.

The method is motivated by the variety of its application in the solution of problems arising from such areas as: population dynamics, pharmaco-kinectic theory, chemical and nuclear reactions, electrical transmission network and other dynamic processes in industries. Its development, analysis and implementation adopts Taylor series expansion, Richardson extrapolation techniques and fortran programming language respectively. The developed schemes are found to be consistent , convergent and A-stable. Numerical results and comparative analysis with some standard methods show that the new schemes are accurate, efficient and effective.

CHAPTER ONE

INTRODUCTION

1.1 PREAMBLE

Numerical problems are encountered in the various branches of human activities such as science, engineering, management and technology. These include:

(i) The study of the rate of decomposition of a radio active substance.

(ii) The study of chemicals reactions of two different elements.

(iii) The population growth of a given community.

(iv) Determination of the quantity of electric charge flowing across an electric circuit.

(v) The study of the rate of growth of the economy of a country.

The mathematical formulation of these problems often leads to first order ordinary differential equations of the form:

$$y' = f(x, y), a \leq x \leq b \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (1.1)$$

Thus, a first order ordinary differential equation is any equation of the form (1.1) which contains a function y(x) and its derivative y' with respect to variable x. Derivative is the rate of change of one variable y in relation to another variable x. Variable y is called the dependent variable and x, the independent variable.

The most general form of an ordinary differential equations is of the form:

$$F(x, y, y', y'' \ldots y^{(n)}) = 0 \quad \ldots\ldots\ldots\ldots\ldots\ldots (1.2)$$

where n is the order of its highest derivatives. The order of a differential equation is the order of the highest derivatives involved in the equation. The degree of a differential equation is the exponent or power to which the highest derivatives is

raised after rationalization. If no product of the dependent variable $y_{(x)}$ with itself or any of the derivatives occurs; the equation is said to be linear, otherwise, it is non-linear.

The differential equation (1.1) together with initial conditions

$y_{(xo)} = y_{(o)}$ ... (1.3) is called a first order initial value problem in ODEs. In order that equation (1.1) with condition (1.3) may have a solution, the following conditions must be satisfied by f(x, y), namely

(i) f(x, y) must be a real valued function

(ii) f(x, y) must be defined and continuous at all points (x, y) in the region D defined by $D = \{(x,y)/a \le x \le b, -\infty < y < \infty\}$ and contains initial point

(x_o, y_o).

There exist a real constant L, such that for any $x \in (a,b)(x,y),(x,y^*) \in D$

$$\left|f(x,y) - f(x,y^*)\right| \le L\left|y - y^*\right| \quad \ldots\ldots\ldots\ldots\ldots\ldots (1.4)$$

where L is the so called Lipschitz constant. This means that f and its partial derivatives df must be continuous.

When equation (1.1) together with equation (1.3) satisfy conditions (i) – (iii), then we shall be sure that it has a unique solution $y_{(x)}$ for every $x \in (a,b)$. [Apostol (1965)].

We can choose the Lipschitz constant L to be

$$L = \sup_{(x,y) \in D} \left|\frac{df(x,y)}{dy}\right| \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (1.6)$$

where df/dy is the Jacobian of f with respect to y.

This class of first-order ordinary differential equations are widely used to solve problems arising from areas such as heat and matter transfer, chemical and nuclear reactions, pharmaco-kinectic theory, electrical or electronic transmission network and other dynamic process in industries.

1.2 EXISTING ONE-STEP METHODS FOR SOLUTION OF ODES

There are various one-step algorithms designed to solve both linear and non-linear initial value problems. These include:

(i) Euler's schemes

(ii) Runge-Kutta methods

(iii) Taylor's series expansion technique as discussed in Lambert (1973)

(iv) Rational Runge-Kutta method by Hong Yuanfu (1982).

1.3 PROBLEMS ASSOCIATED WITH EXPLICIT METHODS

The problem associated with explicit methods are first discovered by Dahlquist (1963). He observed that no explicit one-step method or multistep methods of any order can be A-stable. He observed that only implicit multi-step method can be A-stable.

Unfortunately, implicitness introduces the problems of solving at each integration step, a set of simultaneous linear equations. Other problems include small region of absolute stability associated with the existing explicit methods.

The decline of interest, in the explicit method may be due to the barrier identified by Dahlquist, (1963). However, this barrier has prompted many researchers including Butcher, (1964), Gear (1974), to introduce a less rigorous general conditions such as A(o)-stability, A(o)-stability and L-stability. The work of Hong Yuanfu, (1982), Okunbor, (1985), Ademiluyi (2005) confirmed that there exists some explicit class of Runge-Kutta methods that are A-stable. Based on the above reasons, we proposed a new class of Explicit Runge-Kutta scheme defined as:

$$y_{n+r} = \frac{y_{n+r-1}}{1 + y_{n+r-1}\sum_{i=1}^{r} w_i k_i} \qquad (1.7)$$

where $k_i = hgf(x_{n+r}, z_n)$, $k_i = hgt(x_n + dih, z_n + \sum_{j=1}^{i} b_{ij} k_j)$

This $di = \sum b_{ij}$ formula is developed along the line of thought of Hong Yuanfu, (1982), Babatola (2000), Ademiluyi (2001, 2005).

1.4 MOTIVATION

The method is Explicit, consistent Convergent and A-stable. It is easy to implement on the computer than implicit method. It has variety of applications in solving non-stiff and stiff ordinary differential equations.

1.5 RESEARCH OBJECTIVES

The main objective of this study is to develop an explicit comput methods (1.7) for numerical solution of both non-stiff and stiff first-order o differential equations. Specifically, we

(i) determine the numerical values of the parameters wi, bij as to ensure tha is consistent convergent and A-stable.

(ii) analyse the consistency, order of accuracy, error term, Convergenc stability of the method

(iii) determine the interval of absolute stability of the method

(iv) code the algorithms in a computer programming language (FORTRAN)

(v) implement the programme with specific sample problems on a r computer to test its applicability and suitability.

1.6 RESEARCH METHODOLOGY

To achieve the above objectives, we adopt the Taylor's series expansio y_{n+1}, k_i's and g in two variables in power series of h so as to completely express local truncation error T_{n+1} in powers of h, so as to be able to single out the orde accuracy of the method

The resultant formula was analysed for consistency, convergence and stability adopting Dahlquist (1963) model equation. The algorithms was coded in Fort programming language and implemented on a micro-computer to confirm t workability and accuracy of the newly developed schemes with sample problems.

1.7 EXPECTED CONTRIBUTION

A new class of explicit Runge-Kutta method which is A-stable and can solve bo non-stiff and stiff ODES is developed. The new methods may be developed into multipurpose code for the solution of all kinds of problems arising from populati dynamics economic system, pharmaco-kinetics theory and other dynamic processes.

CHAPTER TWO

LITERATURE REVIEW

According to Dahlquist (1963) postulate, he observed that no explicit one-step method or multistep methods can be A- stable. Any method which is not A-stable cannot accurately solve stiff equations.

However, the introduction of A(o)-stability, A(α)-stability and L-stability conditions (Gear (1974), Butcher (1964)) and A-stability, A(α)-stability Hong Yuanfu (1982) has helped in establishing schemes that cope with stiff equations.

2.1 BASIC CONCEPTS AND PRINCIPLES

The purpose of this section is to briefly explain the basic concepts and principles that are relevant to the schemes. These concepts and principles include:

(i) The mesh-points and mesh size

(ii) Finite difference schemes

(iii) One-step iteration schemes

(iv) Properties of one-step schemes

2.1.1 THE STEPLENGTH OF MESH SIZE

Numberical integration method for solving initial value problems in ordinary differential equations are based on the principles of discretization. Approximate values of unknown function $y_{(x)}$ are sought on a certain discrete points defined by

xi, i = 0, 1, 2 ..... where the sequence of points xi, i = 0(1)N in the interval (a,b) by

$$a = x_0 < x_1 < x_2 \ldots\ldots X_N = b \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots (2.1.1);$$

and $h = \dfrac{b-a}{N}$ and $x_{i+1} = x_{i+h}$, i = 0(1)N ..... (2.1.2),

h is called stepsize or mesh size and the points x_is are called mesh points.

FINITE DIFFERENCE SCHEMES

In this technique, we obtained approximate values of an unknown function $y_i = y_{(x)}$ satisfying the initial value problem of (1.3) at the sequence of points defined by

$$X_{i+1} = Xi + h,\ i = 0(1)N,\ h > o$$

The principle is as follows:

If x_{i-1}, x_i and x_{i+1}, $i = 0(1)n$ are any three points in the interval (a,b) then

$$D y_i = \frac{y_{i+1} - y_i}{x_{i+1} - x_i}$$

$$= \frac{y_{i+1} - y_i}{h} \quad \ldots\ldots\ldots\ldots\ldots \quad (2.1.3)$$

$$D^2 y_i = \frac{y_{i+1} - 2y_i + y_{i-1}}{h^2} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (2.1.4)$$

are called finite difference approximations to the derivatives y' and y'' of y.

Putting equation (2.1.3) into the differential equation (1.3); for example, we obtain

$$\frac{y_{i+1} - y_i}{h} = f(x_i, y_i) \ldots\ldots\ldots\ldots\ldots\ldots\ldots (2.1.5)$$

Hence, the process of replacing the differential equation (1.3) with a finite difference scheme is called finite difference method. The values are expected to approach the exact solution $y_{(x)}$ as h tends to zero.

2.1.2 PRINCIPLES OF ONE-STEP SCHEMES

ONE-STEP INTERATION SCHEMES

The general one-step schemes for solution of the differential equation of type (1.3) is the method in which the approximation y_{n+1} can be generated from the knowledge of y_n at x_n. Generally, one-step schemes are of the form $y_{n+1} = y_n + h\phi(x_n, y_n, h)$ ..... (2.1.6)

where $\Phi(x_n, y_n, h)$ is a function of the arguments x,y,h. This function $\Phi(x_n, y_n, h)$ is called the increment function. One-step schemes of type (2.1.6) are self starting and well suited to computer manipulations. This class of method includes Euler's scheme, Taylor's series scheme and Runge-Kutta scheme. The one-step schemes require one value y_i, i =0(1)n at the single point $x = x_i$ in order to proceed into calculating the y_{i+1} at

$x = x_{i+1}$.

Unfortunately, the type of equation mentioned above are restricted to those problems in which f is well behaved. Their performance on stiff equations are known to be very poor Conte and de Boor (1972, 1965 and 1980).

Examples of one-step schemes are:

(i) Euler's scheme

$$y_{n+1} = y_n + hf(x_n, y_n) \quad \ldots\ldots\ldots\ldots\ldots \quad (2.2)$$

(ii) The Taylor's series scheme

$$y_{n+1} = y_n + hf(x_n, y_n) + \frac{h^2}{2} f'(x_1, y_1) + 0(H^3) \quad \ldots\ldots\ldots (2.3)$$

(iii) Runge-Kutta schemes

$$y_{n+1} = y_n + \sum_{i=1}^{R} c_i k_i \quad \ldots\ldots\ldots\ldots\ldots\ldots (2.4)$$

where $k_i = hf(x_n + a_i h, y_n + \sum_{j=1}^{i} b_{ij} k_j)$ and R – is the stage of the method.

(iv) Rational Runge-Kutta scheme

$$y_{n+1} = \frac{y_n + \sum_{i=1}^{r} w_i k_i}{1 + y_n \sum_{i=1}^{r} v_i H_i} \qquad \text{(2.4a)}$$

RUNGE-KUTTA SCHEMES

Runge-Kutta scheme is one of the oldest numerical methods for the solution of ordinary differential equation (ODES).

These schemes was first proposed by Kutta (1901) and later improved by Runge (1913). The R-stage Runge-Kutta schemes are often divided into three classes, namely;

(i) Explicit: R-K if $B = (bij) = 0$, for $j \geq i$

(ii) Semi-Explicit, R-K if $B = (bij) = 0$, for $j > i$

(iii) Implicit: R-K if $bij \neq 0$, for at least one $j \geq i$.

RATIONAL RUNGE-KUTTA SCHEMES

The inadequacies of Explicit Runge-Kutta schemes, motivated numerical analysts like Hong Yuanfu (1982) to search for other numerical methods that can perform better. This led to the development of the Rational Runge-Kutta schemes.

$$y_{n+1} = \frac{y_n + \sum_{i=1}^{r} w_i k_i}{1 + y_n \sum_{i=1}^{r} v_i H_i} \qquad \text{(2.5)}$$

where $k_i = hf(x_n, y_n)$

$$k_i = hf(x_n + c_{i-1}h, y_n + \sum_{j=1}^{i} a_{i-1j} k_j)$$

$$H_i = hg(x_n, z_n)$$

$$H_i = hg(x_n + d_{i-1}h, z_n + \sum_{j=1}^{i} b_{i-1j} k_j)$$

$$g(x_n, z_n) = -z_n^2 f(x_n, z_n), z_n = {}^1/y_n$$

During analysis of the schemes, he observed that the schemes are A-stable and its region of absolute stability is large. However, the stability property of the schemes encouraged Okunbor (1985) to extend the schemes to family of order four. Their performance on stiff ODES are satisfactory. However, this method requires a lot of computer manipulation and storage facilities. To reduce this computational labour, a new variant of the method is considered, particularly, one-stage, two-stage and three-stage scheme of order one, two and three respectively in chapter three. The methods are A-stable and suitable for the numerical approximation of IVP of ordinary differential equations.

2.1.2 PROPERTIES OF ONE-STEP SCHEMES

The basic properties of one-step schemes includes order of accuracy, consistency, convergence and stability.

The ability of a numerical scheme to reliably control the global error generated is a major aspect of numerical computing that needs to be considered for new method. Thus, the global discretization error associated with (2.2) is expressed as:

$$\ell_{n+1} = y(x_{n+1}) - y_{n+1} \quad \ldots\ldots (2.6)$$

where y_{n+1} is the numerical approximation to the theoretical solution $y(x_{n+1})$ at step x_{n+1}. It is required that the error be made as small as possible as h is sufficiently close to zero. To make the concept clearer the following definition are given.

Definition 1: The local truncation error T_{n+1} associated with one-step schemes (2.1.6) is defined as the amount by which the theoretical solution $y(x_n)$ of the initial value problem (1.3) fails to satisfy the difference equation (2.1.6). That is,

$$T_{n+1} = y(x_{n+1}) - y(x_n) - h\,\Phi\,(x_n, y_{(xn)}, h) \ldots\ldots\ldots (2.7)$$

The relationship between the global error defined by (2.6) and the local truncation error defined in (2.7).

$$\left|\ell_{n+1}\right| \leq K\left|T_{n+1}\right| \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad (2.8)$$

where k is a constant. From (2.7), it is clear that the local discretization error is directly proportional to the truncation error introduced at each step particularly when the derivation and computation of the local truncation error is rigorous and all previous solutions are exact. This establishes the linear convergence of one-step schemes.

Definition 2:

The integration formula (2.1.6) is said to be consistent with the IVP (1.3) if the increment function $\Phi(x,y,h)$ satisfies the condition.

$$\Phi(x_n, y_n, 0) = f(x, y) \quad \ldots\ldots\ldots\ldots\ldots\ldots \quad (2.9).$$

as h tends to zero. For further information consult (Lambert, 1973).

Definition 3:

The one step scheme (2.1.6) is convergent provided that for an arbitrary initial solution vector y_o and arbitrary point $x_o \in (a, b)$, there exists a global error, ℓ_n such that

$$\lim_{n \to \infty} \ell_n = Lt_{(\infty)}(y_{(\infty)} - y_n) = 0 \quad \ldots\ldots\ldots\ldots\ldots\ldots \quad (2.10)$$
$$n \to \infty \qquad n \to \infty$$

Definition 4:

The integration formula (2.1.6) is said to be of order P if P is the largest positive integer such that the local truncation error (L.t.e) t_{n+1} satisfies

$$t_{n+1} = 0\,(h^{p+1}) \quad \ldots\ldots\ldots\ldots\ldots\ldots \quad (2.11)$$

where $0(h^{p+1})$ implies the existence of finite constants C and $h_0 > 0$ such that

$t_{n+1} \leq C h^{p+1}$ ..... (2.11a) for all $h \leq h_o$

Definition 5:

One-step scheme is said to be stable, if for any initial error ℓ_n, there exist a constant K and $h_o > 0$ such that when (2.1.6) is applied to initial value problem (1.3) with step size $h \in (0, h_o)$, the ultimate error ℓ_n satisfies the following inequality.

$$\ell_n \leq K\ell_o, 0 < K < 1 \quad \text{.....} \quad (2.12)$$

one-step scheme is said to absolutely stable for a given step size h_o and for initial value problem (1.3), if the errors tend to zero, as the step size decreases.

A numerical method is said to be accurate, if its numerical solution do not deviate significantly from the corresponding values of the exact solution, while a numerical solution that is not stable is said to have unbounded discretization error, Dahlquist (1959).

Instability exists in various forms, but the two basic forms are: Inherent instability and Induced instability.

Inherent instability is a bye-product of transforming of the real situation into differential equation; while the induced instability is the characteristics of the numerical methods. To further explain the concepts, let us consider the following scalar initial value problem.

$$y' = \lambda y, y_{(xo)} = y_o, a \leq x \leq b \quad \text{.....} \quad (2.13)$$

with $R_e(\lambda) < 0$. Its theoretical solution is

$$y_{(x)} = y_o \ell^{\lambda x} \quad \text{.....} \quad (2.14)$$

Now suppose, we slightly change the initial condition in (2.13) by $\beta > 0$, so that the initial condition becomes

$$y_o + \beta \quad \text{.....} \quad (2.15)$$

Thus, the solution (2.14) modifies into

$$y_{(x)} = (y_o + \beta)e^{\lambda x} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad (2.16)$$

Irrespective of the integration schemes used, and for h > o, no matter how small, it is seer that the second term $\beta e^{\lambda x}$ in (2.16) grows exponentially as the computation proceeds if

$$R_e(\lambda) > o.$$

Then, the solution $y(x) = y_o e^{\lambda x}$ ..... (2.17)

Will become unstable for slight change in the initial condition. This kind of instability is saic to be inherent.

A differential equation may be stable while the numerical scheme gives unstable solution duc to truncation error, round off error and error propagation. This class of instability is termec induced instability.

It arises as a result of:

(i) the process of derivation of the scheme.

(ii) Implementation process that adopts finite iteration steps instead of infinite iteration process during computer implementation of the scheme.

This is normally detected by applying the integration scheme to solve the scalar stability test equation (2.13). Its instability will show in the form of spurious exponentials and it is minimized by reducing the stepsize.

CHAPTER THREE

THE PROPOSED SCHEMES

Rational functions are quotients of polynomials. They are more adequate as function approximation and capable of yielding better results compared with polynomial approximation schemes (Ademiluyi, 1987). Since polynomial do not have singularities, (Fatunla, 1978), it breeds endless generation of smooth derivatives; they are not really suited for problems with singularities as such, poor results are usually obtained.

Therefore, the principal target of rational approximation are the problems with singularities. Hence it constitutes a much richer class of functions than polynomials. This greatly increases their prospect for accurate approximation. This property of rational functions perhaps motivated Hong Yuanfu (1982) to propose the rational scheme (2.5) which is the basis of the newly proposed schemes defined in (1.7) as:

$$y_{n+r} = \frac{y_{n+r-1}}{1 + y_{n+r-1}\sum_{i=1}^{r} w_i k_i}$$

where Ki = hg(x_n + dih, z_n + $\sum_{i=1}^{r}$ bij kj)

The new scheme can be classified into three categories; namely; explicit, semi-implicit and implicit family of methods.

(i) Explicit: The method is explicit when $\beta = \{b_{ij}\}$, i, j = 1(i)r

1(i)r is zero for j≥I in which case β is a lower triangular matrix, with zero diagonal elements. This represent the Yuanfu's family of methods.

(ii) Semi-Implicit: It is semi-implicit, if $\beta = \{b_{ij}\}$, i,j=1(i)r, is zero for j>I, in this case β is a bounded matrix with non-zero diagonal elements.

(iii) Implicit: The method is implicit, if the elements of matrix β is different from zero for at least one j>i. This implies that β is an upper triangular matrix.

In this chapter, we developed explicit families of one-stage, two-stage and three-stag schemes of order one, two and three respectively for non-stiff and stiff ordinary differenti equations. This family of explicit rational Runge-Kutta schemes perform well on non-sti ODES and on certain class of mildly stiff equations. However, when they are applied t sophisticated stiff equations such as stiff osciliatory equations that is, ODES whose eige values are close to imaginary axis, their performance were found to be very poor. The highe the stage of the methods the poorer the stability, Babatola (2000). Other problems inherent in explicit rational Runge-Kutta schemes includes:

(i) large numbers of functions evaluation per integration,
(ii) difficult derivation process
(iii) uneasy analysis, and
(iv) difficult implementation process

In order to reduce the computational labour, the 2R-function evaluation per ste associated with RR-K method (2.5), is replaced by R-functions evaluation per integratio step in the new method.

3.1 DERIVATION OF THE NEW METHOD

In order to achieve the above objectives, we redefine formula (2.5) as:

$$y_{n+1} = \frac{y_{n+1,\,1}}{1 + y_{n+1,\,1}\sum_{i=1}^{r} w_i k_i}, i = 1(1)r \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad (3.1)$$

where $k_i = hg(x_n + d_i h, z_n + \sum_{j=1}^{s} b_{ij} k_j)$ ..... (3.1a)

and $k_i = hg(x_n, z_n)$ with g defined as:

$$g(x_n, z_n) = -z_n^2 f(x_n, y_n), z_n = -y_n^{-1} \quad \ldots\ldots\ldots\ldots\ldots\ldots \quad (3.1b)$$

with constraints

$$d_i = \sum_{j=1}^{i} b_{ij}, i = 1(1)r \quad \text{..... (3.1c)}$$

From the above schemes, there is need to solve systems of equations involving undetermined parameters w_i, d_i, b_{ij}'s which are obtained on the basis of the local truncation error

$$T_{n+r} = y_{n+1}\left[1 + y_n \sum_{i=1}^{r} w_i k_i\right] - y_n \quad \text{.....(3.2)}$$

in line with the order that is required.

From (3.2), it can be seen that the accuracy of the method depends on the values of parameters d_i, b_{ij}, w_i which are determined by adopting the steps below:

(i) obtain the Taylor's series expansion of y_{n+1}, about x_n and the functions k_i's about (x_n, z_n) in two variables in powers of h.

(ii) Insert these expansions into equation (3.2).

(iii) Collect equal powers of h, so that we can single out the order of accuracy of the formula. This steps transform the error equation (3.2) into expression of the form:

$$T_{n+1} = c_0 y_n + c_1 h y_n' + c_2 h^2 y_n'' - - - c_p h^p y^{(p)}{}_n + o(h^{p+1}) \quad \text{..... (3.3)}$$

where c_p depends on the parameters r, d_i, w_i. These parameters are then chosen as to ensure that the resultant algorithm shall have,

(a) adequate high order of accuracy,

(b) minimum bound of local truncation error;

(c) maximum interval of absolute stability,

(d) minimum or adequate computer storage space requirement.

To permit the solution of the equation, there is need to choose some of the parameters as free parameters. These free parameters are chosen arbitrarily in the

spirit of King (1966), Gill (1951) and Blum (1952). This is because the numbe sof parameters normally exceeds the number of equations.

3. 1.1 ONE-STAGE SCHEME

Numerical method (3.1) is explicit if bij=0 for j>i

Setting r=1 in (3.1),we have a new class of one-step explicit Runge-Kutta formula of the form:

$$y_{n+1} = \frac{y_n}{1 + y_n w_1 k_1} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad (3.4)$$

with $d_1 = b_{11} = o$, $k_1 = hg(x_n, z_n)$

$$g(x_n, z_n) = -z^2_n f(x_n, y_n).$$

The corresponding truncation error is

$$T_{n+1} = y_{n+1}[1 + y_n w_1 k_1] - y_n \quad \ldots\ldots\ldots\ldots\ldots\ldots \quad (3.4a)$$

To express (3.4a) completely in powers of h, so as to single out the order of accuracy of the method as much as possible we adopt the steps (i) – (iii) above. Thus the Taylor series expansion of y_{n+1} about (x_n, y_n) is

$$y_{n+1} = y_n + hy'_n + \frac{h^2}{2} y''_n + \frac{h^3}{6} y'''_n + o(h^3) \quad \ldots\ldots\ldots\ldots\ldots\ldots \quad (3.5)$$

Let us determine the differentials of $g(x_n, z_n)$ in terms of g and its partial derivatives about x_n.

Now; $g(x_n, z_n) = f_n$

$$g' = \left(d/dx + g\, d/dz\right)g$$

$$= g_x + gg_z$$

$$g'' = \left(\frac{d^2}{dx^2} + gx\, d/dz + g\, d/dxdz + g^2 \frac{d^2}{dz^2}\right)g \ .$$

$$g'' = \left(\frac{d^2}{dx^2} + gx\, d/dz + g\, d/dxdz + g^2 \frac{d^2}{dz^2} \right) g$$

$$= \quad g_{xx} + 2gg_{xz} + g_x g_z + g^2 g_{zz} + gg^2 z$$

$$g''' = \left(\frac{d}{dx} + g\, d/dz \right)^3$$

$$= \quad \frac{d^3}{dx^3} + 3\frac{d^3}{dx^3}(g\, d/dz) + (3\, d/dx + (g\, d/dz))^2$$

$$+ \quad 3\frac{d}{dx}(g\, d/dz)^2 + (g\, d/dz)^3 \quad \ldots\ldots\ldots\ldots\ldots \quad (3.5a)$$

We now express the derivatives $y^{(i)}$ in terms of f and g about x_n, to obtain

$$y' \quad = f_n$$

$$y'' \quad = g_x + gg_z = F$$

$$y''' = g_{xx} + 2gg_{xz} + g^2 g_{zz}$$

$$+ g_z (g_x + gg_z) = G + F_{gz}$$

$$y^{iv} = g_{xxx} + 3gg_{xxz} + 3g^2 g_{xzz}$$

$$+ g^3 g_{zzz} + g_z(g_{xx} + 2gg_{xz} + g^2 g_{zz})$$

$$+ (g_x + gg_z)(3g_{xz} + 3gg_z + g^2 z)$$

$$= \quad (B + Ggz + F(g^2 z + 3g_{xz} + 3gg_z)$$

where

$$F = g_z + gg_z$$

$$G = g_{xx} + 2gg_{xz} + g^2 g_{zz}$$

$$Fgz = g_x g_z + gg^2 z$$

$$B = g_{xxx} + 3gg_{xxz} + 3g^2 g_{xzz} + g^3 gzzz \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad (3.5b)$$

Similarly, expanding k_1 about (x_n, z_n), we have,

$$k_1 = hg_n \quad \ldots\ldots\ldots\ldots\ldots\ldots \quad (3.6)$$

with $g(x_n, z_n) = -z_n^2 f(x_n, y_n)$

$$zn = \frac{1}{y_n},\ z_n^2 = -1/y^2 n = 1/y^2_n f(x_n, y_n) \quad \text{.....(3.6a)}$$

and $g(x_n, z_n) = g_n$ ..... (3.6b)

where f and g are sufficiently differentiable functions.

Let us express g and its partial derivatives to facilitate comparison of coefficients. That is,

$$g_n = \frac{-f_n}{y^2_n}, gx = \frac{-f_x}{y^2_n},$$

$$g_{xx} = \frac{-f_{xx}}{y^2_n}, gz = \frac{-2f_n}{y_n} + fy, g_{xz} = \frac{-2f_x}{y_n} + f_{xy},$$

$$g_{xxz} = \frac{-2f_{xx}}{y_n} + f_{xxy},$$

$$g_{zz} = -2f_n - y^2_n f_{yy}$$

$$g_{xzz} = -2f_x - y^2_n f_{xyy}$$

$$g_{zzz} = 4y^2_n + 6y^2_n f_{yy} + y^4_n f_{yyy}, \quad \text{..... (3.6c)}$$

Now, substituting equation (3.5) in (3.4a), we obtain,

$$T_{n+1} = \left[y_n + hy'_n + \frac{h^2}{2}y''_n + \frac{h^3}{6}y'''_n + o(h^4)\right]\left[1 + w_1 k_1 y_n\right] - y_n \quad \text{.....} \quad (3.7)$$

Substituting (3.6) in (3.7), and adopting steps (i) – (iii) as mentioned earlier, we have,

$$T_{n+1} = \left(y_n + hy'_n + \frac{h^2}{2}y''_n + \frac{h^3}{6}y'''_n + o(h^4)\right)\left[1 + w_1 hg(x_n, z_n) y_n\right] - y_n \quad \text{.....} \quad (3.7a)$$

Simplify (3.7a)

$$T_{n+1} = y_n + hy'_n + \frac{h^2}{2}y''_n + \frac{h^3}{6}y'''_n + o(h^4)$$

$$+ \quad y_n * (y_n + hy'_n + \frac{h^2}{2}y''_n + \frac{h^3}{6}y'''_n + o(h^4)) w_1 hg(x_n, z_n)$$

$$= hy'_n + \frac{h^2}{2}y''_n + \frac{h^3}{6}y'''_n + o(h^4) + y_n{}^2 w_1 hg(x_n, z_n)$$

$$+ hy'_n w_1 hg(x_n, z_n)y_n + \frac{h^2}{2}y''_n w_1 hg(x_n, z_n)y_n$$

$$+ \frac{h^3}{6}y'''_n w_1 hg(x_n, z_n)y_n + o(h^4) - y_n \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad (3.7b)$$

Collecting equal powers of h, we have,

$$hy'_n + y^2{}_n w_1 hg(x_n, z_n)$$

$$+ \frac{h^2}{2}y''_n + h^2 y'_n w_1 y_n g(x_n, z_n) \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad (3.7c)$$

$$+ \frac{h^3}{6}y'''_n + \frac{h^3}{2}y''_n w_1 y_n g(x_n, z_n) + o(h^4)$$

Equation (3.4a) can now be expressed in form (3.3) as

$$T_{n+1} = h(1 - w_1)f_n + h^2 \left(\tfrac{1}{2}y'' + y'_n w_1 y_n g(x_n, z_n)\right.$$

$$+ h^3\left(\tfrac{1}{6}y'''_n + \tfrac{1}{2}y''_n w_1 y_n g(x_n, z_n)\right) + o(h^4)$$

Imposing accuracy of order one on T_{n+1}, then

$T_{n+1} = o(h^2)$ and

$(1 - w_1)f_n = o$

Since $f_n \neq o$, then

$1 - w_1 = o \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad (3.8)$

solving (3.8) we have

$w_1 = 1.$

Using this value in (3.4), we obtain a one-stage formula of the form:

$$y_{n+1} = \frac{y^2_n}{y_n - hf_n} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad (3.9)$$

which incidentally coincides with the inverse Euler formula proposed by Fatunla (1982) with principal error term,

$$T_{n+1} = \tfrac{1}{2}\left(\frac{g_x + gg_z - f^2_n}{y_n - hf_n}\right)h^2$$

3.1.2 TWO-STAGE SCHEME

Setting r=2 in (3.1), we have a general 2-stage Explicit Runge-Kutta scheme in the form:

$$y_{n+2} = \frac{y_{n+1}}{1 + y_{n+1}(w_1k_1 + w_2k_2)} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad (3.10)$$

with local trunction error

$$T_{n+2} = y_{n+2}[1 + y_{n+1}(w_1k_1 + w_2k_2)] - y_{n+1} \quad \ldots\ldots\ldots\ldots\ldots\ldots \quad (3.10a)$$

where $k_2 = hg(x_n + d_2h, z_n + + b_{21}k_1 + b_{22}k_2)$

since $b_{22} = o$, $d_2 = b_{21}$, then

$$k_2 = hg(x_n + d_2h, z_n + b_{21}k_1)$$

with $g(x_n, z_n) = -z^2_n f(x_n, y_n)$, $z_n = {}^1\!/_{y_n}$ and $k_1 = hg(x_n, z_n)$.

Since the method is explicit hence,

$$d_1 = b_{11} + b_{12} = o$$

$$d_2 = b_{21} \neq o,\ b_{22} = o$$

Removing brackets in (3.10a), we obtained

$$T_{n+2} = y_{n+2} + y_{n+2}\, y_{n+1}\, w_1k_1 + y_{n+2}\, y_{n+1}\, w_2k_2 - y_{n+1} \quad \ldots\ldots\ldots\ldots\ldots \quad (3.10c)$$

The Taylor series expansion of y_{n+2} about (x_{n+1}) yields

$$y_{n+2} = y_{n+1} + hy'_{n+1} + \frac{h^2}{2} y''_{n+1} + \frac{h^3}{6} y'''_{n+1} + o(h^4) \quad \ldots\ldots\ldots\ldots\ldots\ldots \quad (3.11)$$

Substituting (3.11) in (3.10c), we have,

$$T_{n+2} = y_{n+1} + hy'_{n+1} + \frac{h^2}{2} y''_{n+1} + \frac{h^3}{6} y'''_{n+1} + o(h^4)$$

$$+ \quad y^2_{n+1} [w_1k_1 + w_2k_2]$$

$$+ \quad hy'_{n+1} [w_1k_1 + w_2k_2]y_{n+1}$$

$$+ \quad \frac{h^2}{2} y''_{n+1} [w_1k_1 + w_2k_2]y_{n+1}$$

$$+ \quad \frac{h^3}{6} y'''_{n+1} [w_1k_1 + w_2k_2]\, y_{n+1} - y_{n+1} \quad \ldots\ldots\ldots\ldots\ldots\ldots \quad (3.12)$$

The Taylor's series expansion of k_1 and k_2 about (x_{n+1}, z_{n+1}), give

$$k_1 = hg(x_{n+1}, z_{n+1})$$

$$k_2 = hg_n + h^2d_2F + \tfrac{1}{2} d^2{}_2h\,(G) + o(h^3) \quad \ldots\ldots\ldots\ldots\ldots\ldots \quad (3.13)$$

Substituting (3.13) in (3.12), we have,

$$T_{n+2} = hy'_{n+1} + y^2{}_{n+1}(w_1 + w_2)hg(x_{n+1}, z_{n+1})$$

$$+ \quad \frac{h^2}{2} y''_{n+1} + y^2{}_{n+1}\, w_2\, hg\,(x_n + d_2h,\, z_n + b_{21}k_1)$$

$$+ \quad hy'_{n+1} [w_1\, hg(x_n, z_n) + w_2h^2d_2F + w_2 \frac{h^3}{2} d_2{}^2\,(G)]\, y_{n+1}$$

$$+ \quad \frac{h^2}{2} y''_{n+1} [w_1hg(x_n, z_n) + w_2h^2d_2F + w_2 \frac{h^3}{2} d^2{}_2(/G)] y_{n+1}$$

$$\frac{h^3}{6} y'''_{n+1} + \frac{h^3}{6} y''_{n+1} [w_1\, hg(x_n, z_n) + w_2h^2d_2F + w_2 \frac{h^3}{2} d^2{}_2\,(G)]\, y_{n+1} + 0(h^4)$$

collecting equal powers of h. we get,

$$T_{n+2} = h[1 - (w_1 + w_2)f_{n+1} + h^2[\tfrac{1}{2} - w_2 d_2)F + 0(h^3) \quad \ldots\ldots\ldots\ldots\ldots\ldots \quad (3.14)$$

Imposing accuracy of order two on T_{n+2}, that is,

$T_{n+2} = o(h^3)$; then

$[1 - (w_1 + w_2)f_{n+1} = o$

$[\tfrac{1}{2} - w_2 d_2]\ F = o$

Since $f_{n+1} \neq o$, and $F \neq o$, then

$[1 - (w_1 + w_2)] = o$, and

$(\tfrac{1}{2} - w_2 d_2) = o \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad (3.15)$

with $c_o = c_1 = c_2 = o$, but $c_3 \neq o$

The local truncation error term is

$$T_{n+2} = \frac{1}{6}(G + Fgz)(w_1 + w_2) - \tfrac{1}{2}(w_2 d_2^2)\left(\frac{2f_n}{y^2_{n+1}}\right)$$

$$x\ (g_x - f^2_n)\, h^3 + o(h^4) \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad (3.15a)$$

Simplify (3.15), we obtain a system of non-linear equations

$w_1 + w_2 = 1$

$w_2 d_2 = \tfrac{1}{2} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad (3.15b)$

From equation (3.15b), we obtained families of two-stage method of order 2.

Examples of two-stage scheme of order 2, p=2 are:

Case I

$w_1 = o$, $w_2 = 1$, $d_2 = b_{21} = \tfrac{1}{2}$

$$y_{n+2} = \frac{y_{n+1}}{1 + y_{n+1} k_2} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad (3.16)$$

with $k_1 = hg(x_n, z_n)$

$k_2 = hg(x_n + \frac{1}{2} h, z_n + \frac{1}{2} k_1)$

Case II

$w_1 = \frac{1}{4}, w_2 = \frac{3}{4}$

$d_1 = b_{11} + b_{12} = o, b_{22} = o, d_2 = b_{21} = 1.$

Equation (3.10) yields

$$y_{n+2} = \frac{y_{n+1}}{1 + \frac{y_{n+1}}{4}(k_1 + 3k_2)} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad (3.17)$$

with $k_1 = hg(x_n, z_n)$

$k_2 = hg(x_n + h, z_n + k_1)$

Case III

Setting $w_1 = w_2 = \frac{1}{2}$, $d_2 = b_{21} = 1$

$d_1 = b_{11} + b_{12} = o$

Equation (3.10) become,

$$y_{n+2} = \frac{y_{n+1}}{1 + \frac{y_{n+1}}{2}(k_1 + k_2)} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad (3.18)$$

with $k_1 = hg(x_n, z_n)$

$k_2 = hg(x_n + h, z_n + k_1)$

Case IV

Setting $w_1 = \frac{3}{4}$, $w_2 = \frac{1}{4}$

$d_1 = b_{11} + b_{12} = o, b_{22} = o, d_2 = b_{21} = 1$

Equation (3.10) yields

$$y_{n+2} = \frac{y_{n+1}}{1 + \frac{y_{n+1}}{4}(3k_1 + k_2)} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad (3.19)$$

with $k_1 = hg(x_n, z_n)$

$k_2 = hg(x_n + h, z_n + k_1)$.

3.1.3 THREE-STAGE SCHEME

Setting r=3 in equation (3.1), we have a general 3-stage Explicit Runge-Kutta method of the form:

$$y_{n+3} = \frac{y_{n+2}}{1 + y_{n+2}(w_1k_1 + w_2k_2 + w_3k_3)} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad (3.20)$$

The corresponding local truncation error is

$$T_{n+3} = y_{n+3}[1 + y_{n+2}(w_1k_1 + w_2k_2 + w_3k_3] - y_{n+2} \quad \ldots\ldots\ldots\ldots\ldots \quad (3.21)$$

With $k_i = hg(x_n + d_ih, z_n + \sum_{j=1}^{3} b_{ij}k_j)$, i=1, 2, 3

$g(x_n, z_n) = -z^2_n f(x_n, z_n)$, $z_n = 1/y_n$.

$d_i = \sum_{j=1}^{3} b_{ij}$, i=1(3)r

where $k_1 = hg(x_n, z_n)$

$k_2 = hg(x_n + d_2h, z_n + b_{21}k_1 + b_{22}k_2)$

$k_3 = hg(x_n + d_3h, z_n + (d_3 - b_{32})k_1 + b_{32}k_2)$

with the following constraints

$d_1 = b_{11} + b_{12}$

$d_2 = b_{21} + b_{22}$

$$d_3 = b_{31} + b_{32}$$

The method is explicit, hence

$$d_1 = b_{11} + b_{12} = o$$

$$d_2 = b_{21} \neq o$$

$$d_3 = b_{31} + b_{32} \neq o$$

$$b_{22} = b_{33} = o.$$

Simplifying (3.21), we obtain

$$T_{n+3} = y_{n+3} + y_{n+3}\, y_{n+2}\, w_1k_1 + y_{n+3}\, y_{n+2}\, w_2k_2 + y_{n+3}\, y_{n+2}\, w_3\, k_3 - y_{n+2} \ldots\ldots \quad (3.22)$$

The Taylor series expansion of y_{n+3} about (x_{n+2}) yields

$$y_{n+3} = y_{n+2} + hy'_{n+2} + hy'_{n+2} + \frac{h^2}{2} y''_{n+2} + \frac{h^3}{6} y'''_{n+2} + o(h^4) \ldots\ldots\ldots\ldots \quad (3.23)$$

Substituting (3.23) in (3.22), we obtained

$$T_{n+3} = y_{n+2} + hy'_{n+2} + \frac{h^2}{2} y''_{n+2} + \frac{h^3}{6} y'''_{n+2} + o(h^4)$$

$$+ \quad y^2{}_{n+2}[w_1k_1 + w_2k_2 + w_3k_3]$$

$$+ \quad hy'_{n+2}[w_3k_1 + w_2k_2 + w_3k_3]y_{n+2}$$

$$+ \quad \frac{h^2}{2} y''_{n+2}\,[w_1k_1 + w_2k_2 + w_3k_3]y_{n+2}$$

$$+ \quad \frac{h^3}{6} y'''_{n+2}[w_1k_1 + w_2k_2 + w_3k_3]y_{n+2} + 0(h^4) - y_{n+2} \ldots\ldots\ldots\ldots \quad (3.24)$$

The Taylor series expansion of k_1, k_2 and k_3 about (x_{n+2}, z_{n+2}), we have,

$$K_1 = hg(x_{n+2}, z_{n+2})$$

$$K_2 = hg_n + h^2d_2F + \tfrac{1}{2} d_2{}^2 h^3(G) + 0(h^3)$$

$$K_3 = hg_n + h^2F\,(w_2d_2 + w_3d_3)$$

$$+ \quad \frac{h^3}{2}[(2w_3d_2b_{32})]\,Fgz + \frac{h^3}{6}(w_2d_2^2 + w_3d^2_3)\,G \quad \ldots\ldots\ldots \quad (3.25)$$

Substituting (3.25) in (3.24), we have

$$T_{n+3} = hy'_{n+2} + y_{n+2}{}^2\,[\,w_1 + w_2\,w_3]\,hg(x_{n+2}, z_{n+2})$$

$$+ \quad \frac{h^2}{2}y''_{n+2} + y^2_{n+2}\,[w_2d_2 + w_3d_3]h^2Fg(x_{n+2}, z_{n+2})$$

$$+ \quad hy'_{n+2}\,[w_1\,hg(x_{n+2}, z_{n+2}) + w_2h^2d_2F + w_2\frac{h^3}{2}d_2{}^2\,(G)]\,y_{n+2}$$

$$+ \quad \frac{h^2}{2}y''_{n+2}\,[w_1\,hg\,(x_{n+2}, z_{n+2}) + w_2h^2d_2F + w_3\frac{h^3}{2}d_2{}^2\,(G)]\,y_{n+2}$$

$$+ \quad \frac{h^3}{6}y'''_{n+2} + \frac{h^3}{6}y'''_{n+2}\,[w_1\,hg\,(x_{n+2}, z_{n+2}) + w_2h^2d_2F + w_3\frac{h^3}{2}d_2{}^2\,(G)]\,y_{n+2} + 0(h^4)\ldots.\,(3.26)$$

Collecting equal powers of h, we get

$$T_{n+3} = h[1 - (w_1 + w_2 + w_3)f_{n+2}$$

$$+ \quad h^2[\tfrac{1}{2} - (w_2d_2 + w_2d_2]F$$

$$+ \quad h^3[1/6 - (w_3d_2\,b_{32})Fgz$$

$$+ \quad h^3[1/6 - w_2d^2_2 + w_3d^2_3)]G$$

$$+ \text{ higher order terms } \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad (3.27)$$

Equation (3.27) is expressed in the form:

$$T_{n+3} = c_0y_{n+2} + c_1hy'_{n+2} + c_2h^2y''_{n+2} + c_3h^3y'''_{n+2}\ldots c_{p+2}h^{n+2} + o(h^{n+2}) \ldots\ldots\ldots \quad (3.28)$$

where $c_o = c_1 = c_2 = c_3 = o$

but $c_4 \neq o$.

Thus the principal local truncation error for family of three-stage schemes explicit Runge-

Kutta method of order three is

$$T_{n+3} = \frac{-h^4}{2^4}\left[B + Ggz + F(g^2 z + 3g_{xz} + 3gg_z)\right]$$

$$+ \ o(h^5) + 1/6\,(G + Fgz)\,(w_1 + w_2 + w_3)\left(\frac{-2f_n}{y_{n+2}} - \frac{-2f_n^3}{y_{n+2}} + \frac{2f_n^2}{y_{n+2}} - gz\right)$$

$$+ \ \tfrac{1}{2}\,(w_2 d_2^2)\left(\frac{-2f}{y_{n+2}}\right)\left(gx - f_n^2\right)$$

Imposing accuracy of order 3, on T_{n+3}, that is

$$T_{n+3} = 0(h^4)$$

Equation (3.27) becomes

$$[1 - w_1 + w_2 + w_3)] = o$$

$$[\tfrac{1}{2} - (w_2 d_2 + w_3 d_3)] = o$$

$$[1/6 - (w_2 d_2 b_{32})] = o$$

$$[1/6 - (w_2 d_2^2 + w_3 d_3^2)] = o \quad \ldots\ldots\ldots\ldots\ldots\ldots \quad (3.29$$

Simplifying (3.29), we obtained a system of non-linear equations of the form:

$$w_1 + w_2 + w_3 = 1$$

$$w_2 d_2 + w_3 d_3 = \tfrac{1}{2}$$

$$w_3 d_2 b_{32} = 1/6$$

$$w_2 d_2^2 + w_3 d_3^2 = 1/3$$

Solving the above equations, we obtain family of three-stage schemes of order 3 as:

Case I

$$w_1 = \tfrac{1}{4},\ w_2 = o,\ w_3 = \tfrac{3}{4}$$

$$d_2 = 1/3,\ d_3 = 2/3,\ b_{32} = 2/3,\ b_{21} = 1/3$$

Thus, a 3-stage formula of order 3 is

$$y_{n+3} = \frac{y_{n+2}}{1 + \frac{y_{n+2}}{4}(k_1 + 3k_3)} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad (3.30)$$

with

$k_1 = hg(x_n, z_n)$

$k_2 = hg(x_n + 1/3h, z_n + 1/3k_1)$

$k_3 = hg(x_n + 2/3h, z_n + 2/3k_2)$

Case II

$w_1 = 1/6, w_2 = 2/3, w_3 = 1/6$

$d_2 = b_{21} = ½, d_3 = b_{32} = 2, b_{31} = -1$

yielding a 3-stage method of order 3 as:

$$y_{n+3} = \frac{y_{n+2}}{1 + \frac{y_{n+2}}{6}(k_1 + 4k_2 + k_3)} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad (3.31)$$

with

$k_1 = hg(x_n, z_n)$

$k_2 = hg(x_n + ½ h, z_n + ½ k_1)$

$k_3 = hg(x_n + h, z_n - k_1 + 2k_2)$

Case III

$w_1 = 2/9, w_2 = 1/3, w_3 = 4/9$

$d_1 = 0, d_2 = b_{21} = ½, d_3 = b_{32} = ¼$

Giving a 3-stage formula of order 3 as:

$$y_{n+3} = \frac{y_{n+2}}{1 + \frac{y_{n+2}}{9}(2k_1 + 3k_2 + 4k_3)} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad (3.32)$$

with

$k_1 = hg(x_n, z_n)$

$k_2 = hg(x_n + \frac{1}{2} h, z_n + \frac{1}{2} k_1)$

$k_3 = hg(x_n + \frac{3}{4} h, z_n + \frac{3}{4} k_2)$

CHAPTER FOUR

BASIC PROPERTIES OF THE SCHEMES

Based on the mode of derivation of the new schemes errors can be generated when they are adopted for approximation of solutions of ordinary differential equations. The magnitude of these errors determine the degree of accuracy of the schemes. Their effect can be so disastrous in the sense that it can make the solution unstable.

Errors of numerical approximation techniques for IVPs ordinary differential equations arises from different sources; namely modeling and solution processes. Existing literature, [Henrici (1962), Fatunla (1970), Lambert (1973)] grouped the errors into round off error, truncation error, and discretization error respectively. Round-off errors is an error introduced as a result of the computing devices such as the computing formula and the computer Arithmetic. This can be expressed mathematically as:

$$R_{n+1} = y_{n+1} - p_{n+1} \quad \ldots\ldots\ldots\ldots\ldots\ldots \quad (4.1)$$

Where y_{n+1} is the expected solution of the finite difference equation (3.4). While p_{n+1} is the computer output at the $n+1^{th}$ iteration. It is the amount by which the computed approximation p_{n+1} differs from the expected approximation y_{n+1} by the schemes (3.4) at point x_{n+1}. The accumulation of these errors depend on many factors namely; storage and manipulation of numbers that is, the way and manners machine operation are performed.

Truncation error can be defined as the error introduced as a result of ignoring some of higher terms of the power series (Taylor or Binomial) during the development of the new schemes. Mathematically, truncation errors are defined as the amount by which the true solution $y_{(x_{n+1})}$ of the differential equation (1.3) fails to satisfy the difference equation (3.4). For example, the truncation error of the scheme (3.4) can be defined as:

$$T_{n+1} = y_{n+1}[1 + y_n w_1 k_1 y_n] - y_n \quad \ldots\ldots\ldots\ldots\ldots \quad (4.2)$$

With $k_1 = hg(x_n, z_n)$

$g(x_n, z_n) = -z_n^3 f(x_n, y_n)$, $z_n = 1/y_n$.

The bound for this local truncation error of the scheme (3.4) can be obtained by adopting Taylor series expansion for y_{n+1} and k_i's about (x_n, y_n). Following Lambert's approach it can be written in the form:

$$T_{n+1} = \psi(x_n, y_{(xn)})\, h^{p+1} + O(h^{p+2}) \quad \ldots\ldots\ldots\ldots\ldots\ldots \quad (4.3)$$

It is assumed that $y_{(x)}$ is sufficiently differentiable, while P is the order of the method. In (4.3) $\Psi(x_n, y_{(xn)})$ represents the principal error function and $\psi(x_n, y_{(xn)})h^{p+1}$ is called principal local truncation error associated with the scheme. It is the error introduced at each step of the integration of the schemes. These values are often taken as the measure of deviation or dispersion of the numerical solution from the exact solution of the ordinary differential equations.

Error introduced as a result of transforming a differential equation (1.3) into finite difference equation (3.1) leads to error ℓn often called discretization error. Mathematically, it is defined as

$$\ell_{n+1} = y_{(xn+1} - y_{n+1}) \quad \ldots\ldots\ldots\ldots\ldots\ldots \quad (4.6)$$

where $y(x_{n+1})$, y_{n+1} are respectively exact and approximate solution at x_{n+1}.

Since the numerical solution of the scheme (3.4) involves iteration process, there will be propagation of error from step to step. Error propagation means the process by which errors mentioned earlier grow or decay from step to step as n tends to infinity. When iterating with a numerical scheme, we obtained a sequence of values y_i, i=1(1)n. If the value of y_1 has an error and since y_2 depends on y_1, y_2 will be subject to error and so on; errors are accumulated and the final solution may be in serious inaccuracy. The accuracy of the numerical techniques will depends on the magnitude of these errors. The smaller the error, the better is the numerical solution. A method is said to be accurate if the magnitude of its error is small and inaccurate otherwise.

The minimum basic property requirement for a good numerical scheme includes high order of accuracy consistency, convergence and stability. The analysis of these properties will help whether the new scheme have the capability to handle the types of problems envisaged. Therefore, in order to guarantee the quality of approximation by a numerical scheme, it is necessary to have estimate of these errors.

The error estimation procedure for the schemes is based on Richardson extrapolation technique, whereby the local truncation error is estimated from the difference between two predictions using the same scheme with different step sizes h and $\frac{h}{2}$.

If y_{n+1} designate the solution by single step size h, the local discretization error as defined by (4.6) can be written as suggested by Lambert (1973) as

$$y_{(n+1)} - y_{n+1} = \varphi(x_n, y_{(n)}, h)h^p + 0(h^{p+1}) \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (4.7)$$

where P is refers to the order of accuracy and h as step size.

Similarly, suppose we compute another approximation M_{n+1} to $y_{(n+1)}$ by applying scheme (3.1) with $^{h}/2$ as step size then it follows that

$$y_{(n+1)} - M_{n+1} = \varphi(x_n, y_{(n)}, \tfrac{h}{2})(\frac{h}{2})^p + 0(h^{p+1}) \quad \ldots\ldots\ldots\ldots\ldots\ldots \quad (4.8)$$

subtracting (4.7) from (4.8) and simplifying, we obtained

$$\varphi(x_n, y_{(n)})h^{p+1} = [y_{n+1} - M_{n+1}]\left[\frac{2^{p+1}}{2^{p+1}-1}\right] \quad \ldots\ldots\ldots\ldots\ldots\ldots \quad (4.9)$$

Therefore the local discretization error of the scheme can be estimated to be

$$\ell_{n+1} = [y_{n+1} - M_{n+1}]\left[\frac{2^{p+1}}{2^{p+1}-1}\right] \quad \ldots\ldots\ldots\ldots\ldots\ldots (4.9a)$$

The approximation y_{n+1} from step x_n to x_{n+1} is accepted as a good approximation to the exact solution. If the global error is less than error tolerance (T_{ol}). That is,

$|\ell_{n+1}| \leq T_{ol}$ ..... (4.9b)

Experts like Lambert (1973), Gear (1971), Hindmash (1983) have found this form of error estimate to be adequate for ordinary differential equations. This estimate can then be used to choose reasonable step size that will accelerate convergence of the sheme.

4.1 ORDER OF ACCURACY AND ERROR CONSTANT OF THE SCHEMES

Setting r=1, we can estimate the order and the error term of our method. The corresponding error equation (3.4) is

$T_{n+1} = y_{n+1}[1 + w_1 k_1 r_n] - y_n$ ..... (4.10)

But $T_{n+1} = c_0 y_{(n)} + c_1 h y'_{(n)} + c_2 h^2 y'' \ldots\ldots + c_p h^p y_n^p + 0(h^{p+1})$ ..... (4.10a)

Simplify (4.10) and collecting equal powers of h, we have an expression similar to equation (4.10a).

Definition

According to Lambert (1979) a numerical formula of type (3.4) is said to be of order P, if P is the largest positive integer for which the constants $c_p = o$, P = 0, 1, 2, 3 ... m

Thus imposing accuracy of order one on T_{n+1}, we have, $T_{n+1} = 0(h^2)$, then equation (3.8) become, $c_0 = 0$, $c_1 = (1 - w_1)$, setting w1=1, $c_0 = 0$, $c_1 = 1 - 1 = 0$

But $c_2 = (\frac{1}{2} - 0) = \frac{1}{2} \neq 0$.

That is $c_2 = c_{p+1} = \frac{1}{2} \neq 0$ Hence, p+1 =2, p=1

The result above shows that the method (3.4) is of order p=1 with principal truncation error constant $C_2 = \frac{1}{2}$.

Similarly, setting r=2, imposing accuracy of order two on T_{n+2}, we have,

$T_{n+2} = 0(h^3)$, equation (3.15) yields,

$c_1 = (1 - (w_1 + w_2)$ and

$c_2 = (\frac{1}{2} - w_2 d_2)$

Using the value of w_1, w_2, d_2 we have

$C_1 = 1-1 = 0$

$C_2 = \frac{1}{2} - \frac{1}{2}, 1 = 0$

$C_0 = 0, c_1 = 0, c_2 = 0$, but $c_3 \neq 0$

The local truncation error term is

$$C_3 = \left[\frac{1}{6} - \frac{1}{2}(w_2 d_2^2)\right]$$

$C_3 = (1/6 - \frac{1}{4}) = -1/12$

That is $c_3 = c_{p+1} = -1/12 \neq 0$

Hence, $c_3 = c_{p+1} = 3, p+1 = 3, p = 2$

The order p of the scheme (3.10) is 2 and the error constant $c_{p+1} = -1/12$.

Correspondingly, setting $r = 3$, imposing accuracy of order three on T_{n+1}, that is,

$T_{n+1} = 0(h^3)$, equation (3.29) becomes

$C_0 = 0$

$C_1 = (1 - w_1 + w_2 + w_3)$

$C_2 = [1/2 - (w_2 d_2 + w_3 d_3)]$

$C_3 = [1/6 - (w_2 d_2 b_{1,0})]$

$C_3 = [1/6 - 1/2(w_2 d_2^2 + w_3 d_3^2)]$

substituting for the values of w_1, w_2, w_3, d_2 and d_3, we get into the equations, we get

$c_1 = 1 - 1 = 0$

$c_2 = \frac{1}{2} - \frac{1}{2} \cdot 1 = 0$

$c_3 = 1/6 - 1/6 = 0$ but $c_4 \neq 0$

The local truncation error term is

$C_4 = (-1/24 + 1/6 + \frac{1}{4}) = 3/8$

That is $c_4 = c_{p+1} = 3/8 \neq 0$

Hence $c_3 = c_{p+1} = 4$, $p + 1 = 4$, $p = 3$

$Cp+1 = c_4 = 3/8 \neq 0$.

The results obtained shows that the order p of the scheme (3.20) is 3 and the error constant $c_{p+1} = 3/8$. Since the schemes are of order $p = 3 \geq 1$, the schemes are consistent, (Lambert (1973)).

4.2 CONSISTENCY

Consistency is the property that ensures that the formula exactly approximates the differential equation.

4.2.1 ONE-STAGE SCHEME

Here, we demonstrate the consistency of the proposed one-stage scheme.

$$y_{n+1} = \frac{y_n^2}{y_n - hf_n} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad (4.12)$$

To achieve this, we subtract y_n from both sides, to get,

$$y_{n+1} - y_n = \frac{hy_n y_n}{y_n - hy_n'} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad (4.13)$$

Dividing both sides by h, we get

$$\frac{y_{n+1} - y_n}{h} = \frac{y_n y_n'}{y_n - hy_n'} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad (4.14)$$

Take limit on both sides of (4.14) as h tends to zero, equation (4.14) we have

$$y'_n = f(x_n, y_n)$$

suggesting that the method is consistent.

4.2.2 TWO-STAGE SCHEME

Recall the two-stage scheme in (3.16) is

$$y_{n+2} = \frac{y_{n+1}}{1 + y_{n+1}k_2} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad (4.15$$

Subtracting y_{n+1} from both sides, we get,

$$y_{n+2} - y_{n+1} = \frac{y_{n+1}}{1 + y_{n+1}k_2} - y_{n+1}$$

$$y_{n+2} - y_{n+1} = \frac{y_{n+1}[1 - 1 - y_{n+1}k_2]}{1 + y_{n+1}k_2}$$

$$y_{n+2} - y_{n+1} = \frac{-y^2_{n+1}k_2}{1 + y_{n+1}k_2}$$

dividing both sides by h, and take limit as h tends to zero on both sides to get.

$$\frac{y_{n+2} - y_{n+1}}{h} = \frac{-y^2_{n+1}hg(x_n + d_2h, z_n + b_{21}k_1)}{1 + y_{n+1}hg(x_n + d_2h, z_n + b_{21}k_1)} \times \frac{1}{h}$$

as h tends to zero, equation (4.15) becomes

$$\frac{y_{n+2} - y_{n+1}}{h} = -y^2_{n+1}g(x_n, z_n)$$

$$= \frac{y^2_{n+1}}{y^2_{n+1}} f_{n+1}$$

$$y'_{n+1} = f_{n+1}$$

4.2.3 General Case

Recall that equation (3.1) is

$$y_{n+r} = \frac{y_{n+r-1}}{1 + y_{n+r-1}\sum_{i=1}^{r} w_i k_i}$$

Subtracting y_{n+r-1} from both sides, we get,

$$y_{n+r} - y_{n+r-1} = \frac{y_{n+r-1}}{1 + y_{n+r-1}\sum_{i=1}^{r} w_i k_i} - y_{n+r-1}$$

$$= \frac{y_{n+r-1} - y_{n+r-1} - y^2_{n+r-1}\sum_{i=1}^{r} w_i k_i}{1 + y_{n+r-1}\sum_{i=1}^{r} w_i k_i}$$

$$= -\frac{-y^2_{n+r-1}\sum_{i=1}^{r} w_i k_i}{1 + y_{n+r-1}\sum_{i=1}^{r} w_i k_i}$$

with $k_i = hg(x_n + d_i h, z_n + \sum_{j=1}^{i} b_{ij} k_j)$

substituting for k_i, and divide both sides by h, to obtain

$$\frac{y_{n+r} - y_{n+r-1}}{h} = -\frac{y^2_{n+r-1}\sum_{i=1}^{r} w_i hg(x_n + d_i h, z_n + \sum_{j=1}^{i} b_{ij} k_j)}{1 + y_{n+r-1}\sum_{i=1}^{r} w_i hg(x_n + d_i h, z_n + \sum_{j=1}^{i} b_{ij} k_j)} \times \frac{1}{h}$$

as h tends to zero, the above equation becomes

$$y'_{n+r-1} = -\frac{y^2_{n+r-1}\sum_{i=1}^{r} w_i g(x_n + d_i h, z_n + \sum_{j=1}^{i} b_{ij} k_j)}{1}$$

$$y'_{n+r-1} = -y^2_{n+r-1}\sum_{i=1}^{r} w_i g(x_{n+r-1} + z_{n+r-1})$$

But $\sum_{i=1}^{r} w_i = 1$, and $g(x_{n+r-1}, z_{n+r-1} = \frac{f(x_{n+r-1}, y_{n+r-1}}{y^2_{n+r-1}}$ ..... (4.16)

substitute (4.16) in the above equation, we obtained,

$$y'_{n+r-1} = f(x_{n+r-1}, y_{n+r-1})$$

4.3 CONVERGENCE OF THE METHOD

As mentioned earlier, any error introduced at any stage of the computation which is not bounded can produce unstable numerical results, hence it is necessary to examine whether our methods are convergent. This section therefore examines the behaviour of the error generated by the proposed schemes.

4.3.1 One-Stage Scheme

Recall that the one-stage scheme discussed in chapter three is

$$y_{n+1} = \frac{y_n^2}{y_n - hf_n} \qquad (4.17)$$

The theoretical solution $y_{(x)}$ to the equation (1.3) is seen to satisfy

$$y_{(n+1)} = \frac{y^2(x_n)}{y(x_n) - hy'(x_n)} + T_{n+1} \qquad (4.18)$$

where T_{n+1} is the local truncation error substracting (4.17) from (4.18) yields.

$$y_{(n+1)} - y_{n+1} = \frac{y^2(x_n)}{y(x_n) - hy'(x_n)} - \frac{y_n^2}{y_n - hy_n'} + T_{n+1} \qquad (4.19)$$

adopting the definition in Chapter 3,

$$y_{(n+1)} - y_{n+1} = \ell_{n+1}$$

simplify (4.19), we get,

$$\ell_{n+1} = \frac{y^2(x_n)(y_n - hy_n') - y^2{}_n(y(x_n) - hy'(x_n)}{(y(x_n) - hy'(x_n)(y_n - hy_n')} + T_{n+1} \qquad (4.20)$$

Simplifying (4.20), we obtain,

$$\ell_{n+1} = \frac{y_n^2 \ell_n}{(y_{n-1})(y_{n-1})} + T_{n+1} \qquad (4.21)$$

Taking modulus on both sides (4.21) yields

$$\left|\ell_{n+1}\right| \leq \left|\frac{y_n^{\,2}}{y^{2}_{n-1}}\right|\left|\ell_n\right| + T_{n+1} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad (4.21a)$$

Setting $\left|\frac{y_n^{\,2}}{y_{n-1}^{\,2}}\right| = \left|\left(\frac{y_n}{y_{n-1}}\right)^2\right| = k$ ..... (4.21b)

Then, the inequality (4.21a) becomes,

$$\left|\ell_{n+1}\right| \leq k\left|\ell_n\right| + \left|T_{n+1}\right| \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad (4.21c)$$

Let T = $Sup\left|T_{n+1}\right| and E_{n+1} = Sup\left|\ell_{n+1}\right|, o \leq n \leq \infty$

Therefore,

$E_{n+1} < k E_n + T,\ n = o, 1, 2 \ldots$

$E_1 < k E_o + T$

$E_2 < k E_1 + T$

Substituting for the value of E_1, we get,

$E_2 < k E_1 + T = k (K E_o + T) + T$

$E_2 < k^2 E_o + k T + T$

Also $E_3 < K E_2 + T$

Substituting for the value of E_2, we get,

$E_3 < K^3 E_o + + K^2 T + KT + T$

Continuing in this way, we shall obtain

$$E_n < K^n E_o + \sum_{i=1}^{n} k^{i-1} T .$$

Since K < 1, Then as n → ∞

Lim E_n → o. Hence ℓ_n → o,

This indicates that the schemes converges. According to Henne (1962), Lambert (1973) and Fatunla (1987), a consistent and convergent method is stable. Hence, the new scheme is stable.

4.3.2 General Case

Recall that the general scheme is

$$y_{n+r} = \frac{y_{n+r-1}}{1 + y_{n+r-1}\sum_{i=1}^{r} w_i k_i} \qquad \text{(4.22)}$$

The theoretical solution $y_{(x)}$ is seen to satisfy this equation

$$y_{(n+r)} = \frac{y(x_{n+r-1})}{1 + y(x_{n+r-1})\sum_{i=1}^{r} w_i k_i} + T_{n+r} \qquad \text{(4.23)}$$

Subtracting (4.22) from (4.23), we get

$$y_{(n+r)} - y_{n+r} = \frac{y(x_{n+r-1})}{1 + y(x_{n+r-1})\sum_{i=1}^{r} w_i k_i} - \frac{y_{n+r-1}}{1 + y_{n+r-1}\sum_{i=1}^{r} w_i k_i} + T_{n+r} \qquad \text{(4.24)}$$

Adopting definition (2.6) in (4.24) and simplify, we get,

$$y(x_{n+r}) - y_{n+r} = \ell_{n+r}$$

$$\ell_{n+r} = \frac{y(x_{n+r-1}) - y_{n+r-1}}{(1 + y(x_{n+r-1})(1 + y_{n+r-1})\sum_{i=1}^{r} w_i k_i} + T_{n+r}$$

Simplifying (4.25), we obtained

$$\ell_{n+r} = \frac{\ell_{n+r-1}}{(1 + y_{(n+r-1)})(1 + y_{n+r-1})\sum_{i=1}^{r} w_i k_i} + T_{n+r} \qquad \text{(4.26)}$$

Let $P = (1 + y(x_{n+r-1})\sum_{i=1}^{r} w_i k_i$, and

$$Q = (1 + y_{n+r-1})\sum_{i=1}^{r} w_i k_i$$

Substituting for P and Q in (4.25), we obtain

$$\ell_{n+r} = \frac{\ell_{n+r-1}}{PQ} + T_{n+r} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad (4.26)$$

with p > o, Q > O,

Taking modulus on both sides of (4.26), we get

$$|\ell_{n+r}| \leq \left|\frac{\ell_{n+r-1}}{PQ}\right| + |T_{n+r}| \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad (4.27)$$

Setting $\left|\frac{1}{PQ}\right| = \left|\frac{1}{PQ}\right| = \frac{1}{PQ} = K$

Then

$$|\ell_{n+r}| \leq K|\ell_{n+r-1}| + |T_{n+r}| \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad (4.28)$$

Let T = Sup (T_{n+r}) and K < 1.

Similarly setting

E_{n+r} = Sup ℓ_{n+1}, then the inequality (4.28) modified into

$E_{n+r} \leq K\ E_{n+r-1} + T$, hence,

Setting r=1, we get,

$E_{n+r} \leq K\ E_n + T$

$E_{n+2} \leq K^2 En + KT + T$

$E_{n+3} \leq K^3 E_n + K^2 T + KT + T$

Continuing in this way, it can be shown that

$$E_{n+r} \leq K^r E_n + \sum_{r=n}^{r-1} K^r T$$

Since K < 1, hence

As $n \rightarrow \infty$

$E_{n+r} \rightarrow 0$

Hence $\ell_{n+r} \rightarrow 0, n \rightarrow \infty$.

This shows that the general method is convergent.

4.4 ABSOLUTE STABILITY PROPERTIES OF THE METHOD

To understand the meaning of absolute stability, we adopt the following definitions;

Definition 1: A numerical method is said to be A-Stable if its region of absolute stability contains the whole of the left-hand half-plane Re h λ<0 (Lambert, 1973).

Definition 2: The scheme (3.4) is said to be absolutely stable if a point (z, $\mu(z)$) lies in the complex Z-plane, that is, if the stability function satisfies,

$$|\mu(Z)| < 1 \qquad \text{.....(4.31)}$$

The corresponding region R of absolute stability of the scheme can be defined as:

$$R = \{Z: |\mu(Z)| < 1\} \qquad \text{.....(4.32)}$$

Definition 3:

The numerical scheme (3.4) is said to be A-stable if the region of absolute stability AS, specified in (4.32) includes the entire half of the complex plane denoted by

$$AS = \{z/ZEC \text{ and } Re(z) < 0\} \qquad \text{.....} \quad (4.33)$$

A-stability property is one of the *desirable properties for any numerical methods* both stiff and non-stiff ordinary differential equations as suggested by Dahlquist (1963).

To determine the suitability of the new schemes, we apply them to solve the Dahlquist (1963) stability scalar test, initial value problem.

$$y' = \lambda y, y_{(x=0)} = y_{(x=0)} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad (4.34)$$

under the assumption that $Re(\lambda) \leq 0$, (λ is a complex constant with negative real part).

4.4.1 ONE-STEP SCHEME

A one-step application of the scheme (3.9) to the stability test equation (4.34) yields

$$y_{n+1} = \left[\frac{1}{1-\lambda h}\right] y_n \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad (4.35)$$

where $\frac{1}{1-\lambda h}$ is called stability function.

For the convergence of the solution of (4.35)

$$|\ \mu(z)\ | = |\ \frac{1}{1-\lambda h}\ | < 1 \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad (4.36)$$

setting $z = \lambda h$, then equation (4.36) becomes

$$\left|\frac{1}{1-z}\right| < 1 \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad (4.37)$$

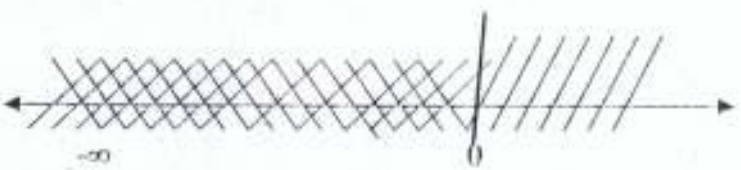
Interpretation of (4.37) yields

$$-1 < \frac{1}{1-z} < 1$$

Simplifying, we get

$z < 2$ and $z < 0$, the stabiliy region is the intersection of the two sets as shown

Fig 4.1



The double shaded region $(-\infty, 0)$ represents the region of absolute stability (AS).

Fig 4.1: A-stability region of the one-stage scheme

This implies that the interval of A-stability of the method is $(-\infty, 0)$. The analysis indicates that the scheme (3.9) is A-stable because its region of absolute stability include the entire half lefthand of the complex plane (Dahlquist (1963)).

4.4.1 TWO-STAGE SCHEME

The Two-stage scheme (3.16) is

$$Y_{n+2} = \frac{y_{n+1}}{1 + y_{n+1}k_2}$$

Applying this formula to the Dahlquist (1963) test equation (4.34), we obtained,

$$y_{n+2} = \frac{y_{n+1}}{1 + y_{n+1}hg(x_n + d_2h, z_n + b_{21}k_1)} \quad \ldots\ldots\ldots\ldots\ldots\ldots \quad (4.38)$$

Equation (4.38) modified into:

$$y_{n+2} = \frac{y_{n+1}}{1 + y_{n+1}\lambda h^2 g(x_{n+1}, z_{n+1})} \quad \ldots\ldots\ldots\ldots\ldots\ldots \quad (4.39)$$

But $g(x_{n+1}, z_{n+1}) = \frac{-1}{y_{n+1}^2} f(x_{n+1}, y_{n+1})$ ..... (4.40)

Substituting (4.40) in (4.39) and simplify, we get,

$$y_{n+2} = \frac{1}{1 - \lambda h^2} \quad \ldots\ldots\ldots\ldots\ldots\ldots \quad (4.41)$$

The stability function of (4.41) is

$$\mu(z) = \frac{1}{1 - z^2} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad (4.42)$$

Equation (4.42) gives a stable solution if

$$|\mu(z)| < 1, z < o,$$

That is $\left|\frac{1}{1-z^2}\right| < 1$ ..... (4.43)

Simplifying equation (4.43), we obtained,

$$-1 < \frac{1}{1-z^2} < 1$$

$$-1\,(1 - z^2) < 1 < 1 - z^2$$

$$-1\,(1 - z^2) < 1$$

$$-1 - z^2 < 1$$

$$z < \sqrt{2}$$

$(-\infty, \sqrt{2})$ and

$$1 < 1 - z^2$$

$$1 + z^{2 < 1}$$

$$z^2 < o$$

$$z < o$$

$(-\infty, o)$

Meaning that the interval of A-stability of the scheme (3.16) is $(-\infty, o)$.

Recall the Two-stage scheme (3.18) is

$$y_{n+2} = \frac{y_{n+1}}{1 + \frac{y_{n+1}}{2}(k_1 + k_2)}$$

Applying equation (4.34) in (3.18), we have,

$$y_{n+2} = \frac{y_{n+1}}{1 + \frac{y_{n+1}}{2}[hg(x_{n+1}, z_{n+1}) + hg(x_n + d_2 h, z_n + b_{21} k_1]} \quad \ldots\ldots \quad (4.44)$$

Equation (4.44) modified into:

$$y_{n+2} = \frac{y_{n+1}}{1 + \frac{y_{n+1}}{2}\lambda(h + h^2) g(x_{n+1}, z_{n+1})} \quad \ldots\ldots \quad (4.45)$$

But $g(x_{n+1}, z_{n+1}) = \frac{-1}{y_{n+1}^2} f(x_{n+1}, z_{n+1})$ ..... (4.46)

Substituting (4.46) in (4.45) and simplified, we get,

$$y_{n+2} = \frac{1}{1 - \frac{1}{2}\lambda(h + h^2)} \quad \ldots\ldots \quad (4.47)$$

The stability function of (4.47) is

$$\mu(z) = \frac{1}{1 - \frac{z}{2} - \frac{z^2}{2}} \quad \ldots\ldots \quad (4.48)$$

Then, equation (4.48) gives a stable solution

$|\mu(z)| < 1, z < o$. That is, if $\left| \frac{1}{1 - \frac{z}{2} - \frac{z^2}{2}} \right| < 1$

Simplifying (4.48), we have,

$$-1 < \frac{1}{1 - \frac{z}{2} - \frac{z^2}{2}} < 1$$

$$-1(1 - \frac{z}{2} - \frac{z^2}{2}) < 1 < 1 - \frac{z}{2} - \frac{z^2}{2}$$

$$-1+\frac{z}{2}+\frac{z^2}{2}<1$$

$$\frac{z^2}{2}+\frac{z}{2}<2$$

$$z^2+z<4$$

$$z^2+z-4<0.$$

Applying the general formula to solve the equation, we have,

$$z=\frac{-b\pm\sqrt{b^2-4ac}}{2a}, a=1, b=1, c=-4$$

$$=\frac{-1\pm\sqrt{1+4x1x4}}{2}$$

$$=\frac{-1\pm\sqrt{17}}{2}$$

$$=\frac{-1\pm4.1}{2}$$

z = -2.5 or 1.5

z < 1.5

$(-\infty, 1.5)$ and

$$1<1-\frac{z}{2}-\frac{z^2}{2}$$

$$\frac{z}{2}+\frac{z^2}{2}+1<1$$

$$\frac{z}{2}+\frac{z^2}{2}<0$$

$$z^2+z<0$$

$$z<0$$

The interval of A-stability of the scheme (3.18) is $(-\infty, 0)$ (see fig. 4.2) The stability property of three-stage scheme follows the same procedure. Hence the methods are convergent and A-stable.

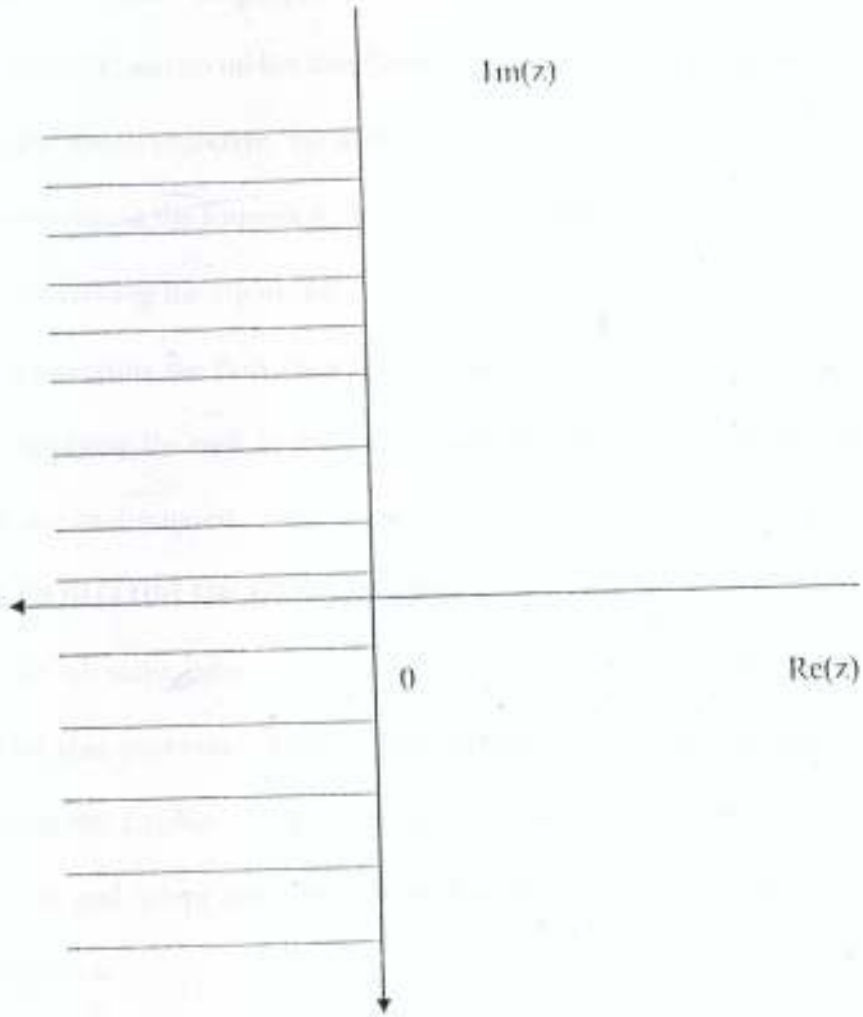


Fig. 4.2: A-Stability Region of two stage Scheme

CHAPTER FIVE

APPLICATION OF THE SCHEME

In order to make the scheme problem applicable there is need to translate the new numerical formula (3.4) into computer programme. This involves the writing of the formula (3.4) in computer algorithm called pseudo-code.

Some of the computer languages that are available are FORTRAN, BASIC, PASCAL, CLIPPER, DBASE and so on but this thesis adopts the Fortran programming language

To achieve the above objective, the following steps are adopted.

(i) re-writing the formula in an algorithmic form

(ii) translating the algorithm into a computer flow chart

(iii) converting the flow chart into computer code

(iv) applying the code to solve some sample problems on a digital computer

The results are as discussed.

5.1 ALGORITHM OF THE METHOD

A set of steps taken to obtain the solution of a given problem is defined as the algorithm of that problem. Thus in this section, we develop the numerical algorithm for implementing the Explicit Runge-Kutta schemes described in chapter three, most especially formula (3.4) and adopt the error estimation discussed in chapter four and the step size control measures.

The algorithm is given below:

STEP 1: Declaration of variables

STEP 2: Define function

F(x,y), y exact

STEP 3: Selection of input values

X_o, X_{last}, y_o, h_{old}, t_{ol},

STEP 4: Initialise variables by setting

n = o (counter)

x = X_o

y = y_o

H = h_{old}

P = o

STEP 5: Compute the approximate values of $y_{(x)}$, by adopting routine EXPRK (x, y, h_{old}, t_{ol}, y_{n+1})

For i = 1, 2, 3 do

Set $k_i^{(P)}$ = 0.0

and for j = 1, 2, 3 define

$d_i = D, b_{ij} = \beta$

Set z_n = $-1/y_n$

z_{ni} = z_n + sum $\beta * K_i^{(p)}$

x_i = x_n + D.h

y_i = $1/z_{ni}$

Set $k_i^{(p+1)}$ = $-z^2_{ni} hf(x_i, y_i)$

While [abs $(k_i^{(p+1)} - k_i^{(p)})$] < T_{ol}

Then, for I = 1, 2, 3 do

$Q^{(p+1)}$ = $1 + y_n.sumw_i k_i^{(p+1)}$

y_{n+1} = $[Q^{(p+1)}]^{-1}$

x_{n+1} = $x_n + h$

$y_{(xn+1)}$ = $y_{exact}\ (x_{n+1}, y_{n+1})$

Return the results

Else, for i = 1, 2, 3, do

Set $k_i^{(p)} = k_i^{(P+1)}$

Repeat Step 5.

STEP 6: Estimate the local truncation error (L.T.E)

Using subroutine ADAPT (x, y, h, t_{ol}, L_{te}, H_{new}, y_{n+1})

Set $y_{new} = y + hf\ (x, y)$

D_{new} = Abs (t_{ol}*y_{new})

Call EXPRK (x_o, y_o, h, t_{ol}, y_1)

H_{new} = 0.5 x h_{old}

Call EXPRK (x_o, y_o, h_2, t_{ol}, y_2)

SET x_1 = $x_o + h_{new}$

Call EXPRK (x_1, y_2, h_2, t_{ol}, y_3)

SET

$$Lte = [y_3 - y_1]\left[\frac{2^{r+1}}{2^{r+1}-1}\right]$$

D_{old} = Abs ($y_3 - y_1$)

$$D_{NP} = \left(D_{new}/D_{old}\right)$$

While ($D_{old} < D_{new}$)

Then

SET $h_{new} = h_{old}$ x $(D_{NP})^{0.2}$

ELSE

SET

$$H_{new} = h_{old} \times (D_{NP})^{0.25}$$

STEP 7: While (L.T.E < T_{ol})

Then

Return the results

ELSE

STEP 8: Adjust the size and replace h_{old} by h_{new}

And repeat step 6 and 7.

STEP 9: Output the results

STEP 10: Stopping Criterion while ($x_n < x_{last}$), Then

SET:

$x_n = x_{n+1}$

$y_n = y_{n+1}$

$h_{old} = h_{new}$

$N = n+1$

and repeat step 5 – 10

ELSE

STEP 11: STOP

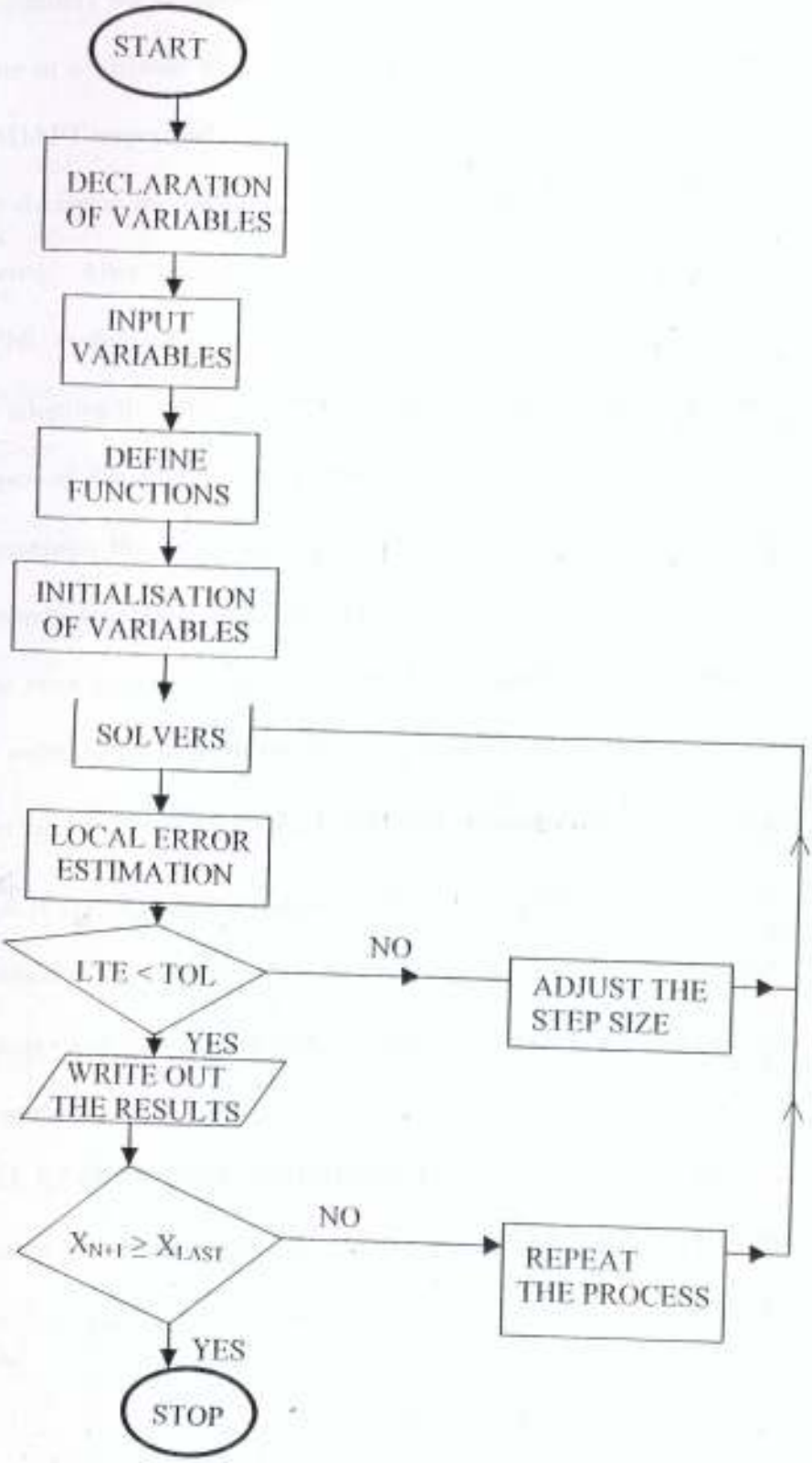
5.2 PROGRAM FLOW CHART

A computer flow chart is a diagrammatic representation of the algorithm or the plan of solution of a problem. It describes the process of solution, the relevant operations, computations, point of decision and other information at the point of solution.

Flow charts are of particular interest because of its documenting feature. They can be constructed by using geometrical symbols, such as squares, rectangles, diamonds shapes or circles (see Fig. 5.1). Each symbol represent some activities which could be input/output of data, taking a decision, terminating the solution and so on.

These symbols are joined by directed line segments to indicate direction of flow. The flow chart of the above algorithm is given below:

Fig. 5.1: FLOW CHAR TOF THE IMPLEMENTATION



5.3 PROGRAMMING IMPLEMENTATION

This section considers the computer implementation of the above algorithm. The implementation is done in a variable step size. The program is in three modules namely: FUNC, EXPRK and ADAPT respectively.

The program starts by declaring the value of variables in double precision mode in order to reduce the round off error. After this, the program chooses initial estimate for the variables and the subroutine FUNC evaluates f(x, y).

This was followed by adopting the solver called EXPRK to call on subroute FUNC to supply the value of the slopes of the integral curve of the solution; and on the receipt of the estimates, EXPRK generates the approximates solution y_{n+1} to $y_{(x)}$ at x_{n+1} and called on routine ADAPT to estimate the values of the error (LTE) associated with the computation.

On the receipt of the error estimate from the subroutines ADAPT, then EXPRK test for convergence of the solution by comparing the magnitude of the error with allowable tolerance. As soon as the condition $|LTE| < T_{ol}$ is met, the program will ask whether the end point is being reached, if yes, it will output result. Otherwise, EXPRK will go the next step to generate the next round of approximation to the solution. The process will be repeated and continued until the upper end point is reached. When the program reached the end point (x_{last}) it will stop the process.

5.3 NUMERICAL EXPERIMENT AND RESULTS

To test the performance of the proposed scheme on non-stiff and stiff IVPs of ordinary differential equations, we consider the following sample problems taken from Okunbor (1985).

Problem 1:

Consider initial value problem

$$y' = \lambda(y - x^3) + 3x^2, y_{(o)} = 1 \quad \ldots\ldots\ldots\ldots\ldots\ldots \quad (5.1)$$

where λ is a complex constant with negative real part i.e. $R_e(\lambda) < o$.

The theoretical solution is

$$y_{(x)} = x^3 + e^{\lambda x} \quad \ldots\ldots\ldots\ldots\ldots\ldots \quad (5.2)$$

which has two parts, the transient part $e^{\lambda x}$ which decays rapidly with respect to x while the second part x^3n varies slowly. This behaviour is characteristics of stiff equations.

The numerical results for $\lambda_1 = -10, \lambda_2 = -100, \lambda_3 = -1000$, are shown in table 1-5.

Problem 2:

The second sample problem considered is the initial value problem

$$y' = 4xy^{1/2}, y_{(o)} = 1 \quad \ldots\ldots\ldots\ldots\ldots\ldots (5.3)$$

defined in the interval $0 \leq x \leq 1$,

The numerical approximation of the solution by the new scheme with stepsize h = 0.1 is as shown in tables 6, 7 and 8.

Problem 3:

Third problem solved is the linear mildly stiff IVP in ODES of the form:

$$y' = -100y, y_{(o)} = 1 \quad \ldots\ldots\ldots\ldots\ldots\ldots \quad (5.4)$$

The problem is taken from Jain (1979). The exact solution is:

$y_{(x)} = e^{-100x}$ in the interval $0 \leq x \leq 1$, steplength h = 0.0001.

The problem was solved by method with stepsize h = 0.0001.

The results is as shown in tables 9, 10 and 11.

5.5 DISCUSSION OF RESULTS

From the results in tables 1, 2, 3 ... 11, we observed that the proposed schemes converges rapidly. The numerical solutions tends to the theoretical solution as the steplength tends to zero. The schemes are accurate, stable and convergent. The schemes compares favourably with the existing Runge-Kutta schemes of the same order.

TABLE 1

NUMERICAL SOLUTION OF PROBLEM 1 BY THE ONE-STAGE SCHEMES WITH $\lambda = -10$.

Variable Step Size	Exact Solution	Numerical Solution by the proposed One-Stage Scheme		Numerical Solution by One-Stage Classical R-K method of order one	
h	$Y(x_n)$	ONE-STAGE y_n	Error e_n	y_n	Error e_n
0.001	0.999000499	0.999000999	0.0000005	0.999000000	0.000000499
0.0005	0.999500125	0.999019029	0.000481096	0.99899104	0.000509085
0.00025	0.999750031	0.999010926	0.000739105	0.998982937	0.000767094
0.000125	0.999875007	0.999003634	0.000871373	0.998975646	0.000899361
0.0000625	0.999937502	0.998997071	0.000940431	0.998969082	0.00096842
0.00003125	0.99996875	0.998991172	0.000977578	0998963183	0.001005567
0.000015625	0.999984375	0.998985867	0.000998508	0.998957877	0.001026498
0.000007812	0.99992187	0.9989810797	0.000940773	0.998953108	0.000968762
0.000003908	0.999996093	0.99897681	0.001019283	0.998948821	0.001047272
0.000001953	0.999998046	0.99897296	0.001025086	0.998944971	0.01053075

TABLE 2

NUMERICAL SOLUTION OF PROBLEM 1 BY THE ONE-STAGE SCHEMES

WITH $\lambda = -100$

Variable Step Size	Exact Solution	Numerical Solution by the one stage scheme and error		Numerical Solution by one stage classical R-K method and error	
h	$Y(x_n)$	Y_n	Error e_n	y_n	Error e_n
0.01	0.990049833	0.990099009	0.000049176	0.990000000	0.000049833
0.005	0.995012479	0.990009409	0.00500307	0.9899104	0.005102079
0.0025	0.997503122	0.989928388	0.007574734	0.989829372	0.00767375
0.00125	0.99875078	0.989855486	0.008895294	0.989756464	0.008994316
0.000625	0.999375195	0.98978985	0.009585345	0.989690824	0.009684371
0.0003125	0.999687548	0.989730858	0.00995669	0.989631829	0.010055719
0.00015625	0.999843762	0.989677808	0.010165954	0.989578775	0.010264987
0.000078125	0.999921878	0.989630114	0.010291764	0.989530179	0.010390799
0.000039062	0.999960938	0.989587251	0.010373687	0.989488214	0.010472724
0.000019531	0.999980468	0.989555165	0.010425303	0.989449709	0.010530759

TABLE 3

NUMERICAL SOLUTION OF PROBLEM 1 BY TWO-STAGE SCHEMES WITH

$\lambda = -10$.

Variable Step Size	Exact Solution	Numerical Solution by the Proposed 2-stage scheme of order 2		Numerical Solution by the 2-stage classical R-K method of order 2	
h	$Y(x_n)$	y_n	Error E_1	y_n	Error E_2
0.001	0.999000499	0.999000999	0.0000005	0.999000500	0.000000000
0.0005	0.999500125	0.998004986	0.001495139	0.998001999	0.001498126
0.00025	0.999750031	0.9997007479	0.002742552	0.997004496	0.002745535
0.000125	0.999875007	0.996010968	0.003864039	0.996006642	0.003868365
0.0000625	0.999937502	0.995015953	0.004921549	0.995011133	0.004926369
0.00003125	0.99996875	0.99402143	0.00594732	0.994016619	0.005952131
0.000015625	0.999984375	0.9930279	0.006956475	0.993023099	0.006961276
0.000007812	0.999992187	0.992035362	0.007956825	0.992030572	0.007961615
0.000003906	0.999996093	0.991043814	0.008952279	0.9910383496	0.008957744
0.000001953	0.999998046	0.990053257	0.009944789	0.990048494	0.009949552

TABLE 4

NUMERICAL SOLUTION OF PROBLEM 1 BY THE TWO-STAGE SCHEMES

WITH $\lambda = -100$.

Step Size	Exact Solution	Numerical Solution by the Proposed 3-stage scheme of order 3		Numerical Solution by the 3-stage classical R-K method of order 3	
h	$y(x_n)$	y_n	Error E_n	y_n	Error E_n
0.01	0.990049833	0.999000999	0.008951166	0.999001099	0.008951265
0.005	0.995012479	0.989119683	0.005892796	0.999000099	0.00398762
0.0025	0.997503122	0.979431928	0.018071194	0.99899909	0.001495968
0.00125	0.99875078	0.96932107	0.02942971	0.998998979	0.000248199
0.000625	0.999375199	0.960614796	0.038760403	0.998997854	0.000377345
0.0003125	0.999687548	0.951474789	0.048212759	0.998996704	0.000690844
0.00015625	0.999843762	0.942507073	0.057336689	0.998995499	0.000848263
0.000078125	0.999921878	0.933706821	0.066215057	0.998994271	0.000927607
0.000039062	0.999960938	0.925069385	0.074891553	0.998993041	0.000967897
0.000019531	0.999980468	0.916590289	0.083390179	0.998991805	0.000988663

TABLE 5

NUMERICAL SOLUTION OF PROBLEM 1 BYTHE THREE-STAGE SCHEMES

WITH $\lambda = -1000$.

Variable Step size	Exact Solution	Numerical Solution and error by the proposed 3 stage scheme		Numerical solution and error by the 3-stage classical R-K method	
h	$y(x_n)$	y_n	Error E_n	y_n	Error E_n
0.1	0.904837418	0.999000999	-0.024163581	0.999001099	0.008951266
0.05	0.951229424	0.98119683	-0.02967406	0.998999009	0.00398653
0.025	0.975309912	0.988142292	-0.01283238	0.998998989	0.001485867
0.0125	0.9875798	0.987166831	0.000410969	0.998997864	0.000247084
0.00625	0.99376949	0.986193295	0.007576195	0.99899675	0.000378448
0.003125	0.996879877	0.985221676	0.011658201	0.998995704	0.002120224
0.0015625	0.99843872	0.98425197	0.01418675	0.998954940	0.000888822
0.00078125	0.99219055	0.983284172	0.016319279	0.998942860	0.000979018
0.000390625	0.999609451	0.982318274	0.017291106	0.998937621	0.001023317
0.00019531	0.999804709	0.981354272	0.018450484	0.998921542	0.001058926

TABLE 6

NUMERICAL SOLUTION OF PROBLEM 2 BY ONE-STAGE SCHEMES

Step Size	Exact Solution	Numerical Solution by One-Stage Scheme		Numerical Solution by one-stage classical R-K method of order one	
x	$y(x_n)$	y_n	Error E_n	y_n	Error E_n
0.00	1.00000000	1.00000000	0.000000000	1.000000000	0.000000000
0.10	1.02010000	1.041666667	0.021566667	1.040000	0.0199
0.20	1.08160000	1.130260646	0.048660646	1.080792156	0.000807844
0.30	1.18810000	1.222232398	0.034132398	1.205545541	0.017445541
0.40	1.34560000	1.371051285	0.025451285	1.381221282	0.035621282
0.50	1.56250000	1.588050082	0.025550082	1.616272024	0.053772024
0.60	1.84960000	1.887631698	0.038031698	1.921390474	0.071790474
0.70	2.22010000	2.287161794	0.067061794	2.309510318	0.089410318
0.80	2.68960000	2.806829341	0.117229341	2.795816658	0.106216658
0.90	3.27610000	3.469520981	0.193420981	3.397761704	0.121661704
1.00	4.00000000	4.300729226	0.300729226	4.135082445	0.135082445

TABLE 7

NUMERICAL SOLUTION OF PROBLEM 2 BY THE TWO-STAGE SCHEMES

Step Size	**Exact Solution**	**Numerical Solution by 2-stage scheme of order 2**		**Numerical solution by the 2-stage classical R-K method of order 2**	
X	**$y(x_n)$**	**y_n**	**Error E_n**	**y_n**	**Error E_n**
0.00	1.00000000	1.000000000	0.00000000	1.000000000	0.000000000
0.10	1.02010000	1.041666667	0.0215666667	1.020000000	0.0001
0.20	1.08160000	1.062915946	0.018684054	1.081389262	0.000210738
0.30	1.18810000	1.210513565	0.022413565	1.187734666	0.000365334
0.40	1.34560000	1.389126338	0.043526338	1.344985881	0.000614119
0.50	1.56250000	1.573134002	0.010634002	1.56147984	0.00102016
0.60	1.84960000	1.832273071	0.017326929	1.84794486	0.00165514
0.70	2.22010000	2.182707451	0.037392549	2.217504127	0.002595873
0.80	2.68960000	2.771141127	0.081541127	2.68567846	0.00392154
0.90	3.27610000	3.649099906	0.372999906	3.270388416	0.00521584
1.00	4.00000000	4.941190135	0.941190135	3.991955811	0.008044189

TABLE 8

NUMERICAL SOLUTION OF PROBLEM 2 BY THE THREE-STAGE SCHEMES

Step Size	Exact Solution	Numerical Solution by the 3-stage scheme of order 3		Numerical Solution by the 3-stage classical R-K method of order 3	
x	$y(x_n)$	y_n	Error E_n	y_n	Error E_n
0.00	1.00000000	1.0000000	.00000000	1.00000000	0.000000000
0.10	1.02010000	1.04166667	0.02156667	1.004286524	0.015813476
0.20	1.08160000	1.062915946	0.018684054	1.062915946	0.018684054
0.30	1.18810000	1.08340855	0.10469145	1.055797815	0.132302185
0.40	1.34560000	1.14881417	0.19678583	1.050542811	0.295057189
0.50	1.56250000	1.27095237	0.29154763	1.042177686	0.520322314
0.60	1.84960000	1.484751492	0.364848508	1.03084787	0.81875213
0.70	2.22010000	1.84914248	0.37095752	1.017567706	1.202532294
0.80	2.68960000	2.497480123	0.192119877	1.003536891	1.686063109
0.90	3.27610000	3.746227027	0.470127027	0.990190056	2.285909944
1.00	4.00000000	6.521358813	2.521358813	0.978864797	3.21135203

TABLE 9

NUMERICAL SOLUTION OF PROBLEM 3 BY ONE-STAGE SCHEMES

Step Size	Exact Solution	Numerical solution by one-stage scheme		Numerical Solution by one-stage classical R-K method of order one	
h	$y(x_n)$	y_n	Error E_n	y_n	Error E_n
0.01	0.990049833	0.990099009	0.000049176	0.99000000	0.000049833
0.005	0.995012479	0.980296048	0.014716431	0.98010000	0.014912499
0.0025	0.997503122	0.970590144	0.026912978	0.97029900	0.027204122
0.00125	0.99875078	0.960980341	0.037770439	0.96059601	0.03815477
0.000625	0.999375195	0.990099009	0.009276186	0.950990049	0.048385146
0.0003125	0.999687548	0.980296049	0.019391499	0.941480144	0.058207404
0.00015625	0.999843762	0.970590148	0.029253614	0.932065342	0.06777842
0.00078125	0.999921878	0.960980344	0.038941534	0.922744688	0.07717719
0.000039062	0.999960938	0.951465688	0.04849525	0.913517241	0.086443697
0.000019531	0.999980468	0.942045236	0.057935232	0.904382068	0.0955984

TABLE 10

NUMERICAL SOLUTION OF PROBLEM 3 BY THE TWO-STAGE SCHMES

Step Size	Exact Solution	Numerical solution by the 2-stage scheme of order 2		Numerical Solution by the 2-stage classical R-K method of order 2	
h	$y(x_n)$	y_n	Error E_n	y_n	Error E_n
0.01	0.990049833	0.990099009	0.000049176	0.994950005	0.004900172
0.005	0.995012479	0.985080267	0.009932212	0.989926002	0.000123831
0.0025	0.997503122	0.975367836	0.022135286	0.979926759	0.017576363
0.00125	0.99875078	0.965727422	0.033023358	0.970032923	0.028717857
0.000625	0.999375195	0.956182622	0.043192573	0.965133771	0.034241424
0.0003125	0.999687548	0.946732492	0.052955056	0.955386885	0.044300663
0.00015625	0.999843762	0.938355346	0.061488416	0.945738432	0.05410533
0.00078125	0.999921878	0.929082045	0.070839833	0.940962925	0.058958953
0.000039062	0.999960938	0.919962746	0.080017722	0.93146014	0.068500798
0.000019531	0.999980468	0.910871894	0.089108574	0.926756732	0.073223736

TABLE 11

NUMERICAL SOLUTION OF PROBEM 3 BY THE THREE-STAGE SCHEMES

Step Size	Exact Solution	Numerical solution by the 3-stage scheme of order 3		Numerical Solution by the 3-stage classical R-K method of order 3	
h	$y(x_n)$	y_n	Error E_n	y_n	Error E_n
0.01	0.990049833	0.990099009	0.000049176	0.990099009	0.000049176
0.005	0.995012479	0.985080267	0.009932212	0.99009809	0.004914389
0.0025	0.997503122	0.97547248	0.022030642	0.99009708	0.007406042
0.00125	0.99875078	0.965955665	0.032795115	0.99009607	0.008654715
0.000625	0.999375195	0.956440852	0.042834343	0.990095051	0.009280144
0.0003125	0.999687548	0.947210257	0.052477291	0.990094021	0.009593527
0.00015625	0.999843762	0.938221515	0.061622247	0.990093009	0.009750753
0.00078125	0.999921878	0.929068582	0.070150198	0.990091959	0.009829919
0.000039062	0.999960938	0.920072981	0.079887957	0.990090906	0.009870032
0.000019531	0.999980468	0.911135472	0.088844996	0.990089839	0.009890629

TABLE 12

NUMERICAL SOLUTION OF PROBLEM 1 BY ONE STAGE SCHEME USING RICHARDSON ERROR ESTIMATE TECHNIQUE

H	Y EXACT	NUMERIVAL VALUE, Y_N	VALUES OF Y_L	L.T.E	ERROR
0.001	0.999000499	0.999000999	0.999500249	0.000665666	0.0000005
0.0005	0.999500125	0.999019029	0.998501746	0.000068971	0.000481096
0.00025	0.999750031	0.999010926	0.998002746	0.00134424	0.000739105
0.000125	0.999875007	0.999003634	0.997503995	0.006661853	0.000871373
0.0000625	0.999937502	0.998997071	0.997005492	0.002655438	0.000940431
0.00003125	0.99996875	0.998991172	0.996507239	0.00331191	0.000977578
0.000015625	0.999984375	0.998985867	0.991549491	0.000991516	0.000998508
0.000007812	0.99992187	0.9989810797	0.991053964	0.001056948	0.000940791
0.000003906	0.99996093	0.99897681	0.990558683	0.001122416	0.001019283
0.000001953	0.999998046	0.99897296	0.990063651	0.001187907	0.001025086

TABLE 13

NUMERICAL SOLUTION OF PROBLEM 1 BY TWO STAGE SCHEME USING RICHARDSON ERROR ESTIMATE TECHNIQUE

H	Y EXACT	NUMERIVAL VALUE, Y_N	VALUES OF Y_L	L.T.E	ERROR
0.001	0.999000999	0.999000999	0.999500249	-0.000570571	0.000000000
0.0005	0.999500125	0.998004986	0.998502371	-0.00056844	0.001495026
0.00025	0.999750031	0.997007479	0.998250313	-0.001420381	0.002742552
0.000125	0.999875007	0.996010968	0.998247571	-0.002556117	0.003864039
0.0000625	0.999937502	0.995015953	0.998220139	-0.003661926	0.004921549
0.00003125	0.99996875	0.99402143	0.997968336	-0.004510749	0.00594732
0.000015625	0.999984375	0.9930279	0.997716661	-0.005458584	0.006956475
0.000007812	0.99992187	0.992035362	0.995206869	-0.003624579	0.007956825
0.000003906	0.99996093	0.991043814	0.992709676	-0.001903838	0.008952279
0.000001956	0.999998046	0.990053257	0.990224977	-0.000196251	0.009944789

TABLE 14

NUMERICAL SOLUTION OF PROBLEM 1 BY THE THREE STAGE SCHEME USING RICHARDSON ERROR ESTIMATE TECHNIQUE

STEP SIZE	Y EXACT	NUMERICAL SOLUTION Y_N	RICHARDSON'S EXTRAPOLATION SOLUTION Y_R	L.T.E	DISCRETIZATION ERROR (e_n)
0.001	0.990049833	0.999000999	0.999500249	-0.000532533	0.008951166
0.005	0.995012479	0.989119683	0.998863792	-0.010393716	0.005892796
0.0025	0.997503122	0.979431928	0.998726622	-0.020581006	0.018071194
0.00125	0.99875078	0.96932107	0.998589491	-0.031219649	0.02942971
0.000625	0.999375199	0.960614796	0.998452397	-0.040360107	0.038760403
0.0003125	0.999687548	0.951474789	0.998315341	-0.049963255	0.048212759
0.00015625	0.999843762	0.942507073	0.998178323	-0.059382666	0.057336689
0.00007812	0.999921878	0.933706821	0.997995326	-0.068574405	0.066215057
0.00003906	0.999960938	0.925069385	0.997858395	-0.07764161	0.074891553
0.00001953	0.999980468	0.916590289	0.997721494	-0.081131205	0.083390179

TABLE 15

COMPARATIVE ANALYSIS OF THE PROPOSED SCHEME AND CLASSICAL RUNGE KUTTA METHOD ON PROBLEM 2

PROPOSED SCHEME					RUNGE KUTTA METHOD		
X	Y EXACT	ONE-STAGE	TWO-STAGE	3-STAGE	ONE-STAGE	TWO-STAGE	3-STAGE
0.00	1.00000000	1.0000000	1.00000000	1.00000000	1.0000000	1.0000000	1.0000000
0.10	1.02010000	1.041666667	1.041666667	1.04166667	1.040000	1.020000000	1.004286524
0.20	1.08160000	1.130260646	1.062915946	1.062915946	1.080792156	1.081389262	1.062915946
0.30	1.18810000	1.222232398	1.210513565	1.08340855	1.205545541	1.187734666	1.155797815
0.40	1.3456000	1.346051285	1.342126338	1.34001417	1.341221282	1.344985881	1.320542811
0.50	1.56250000	1.568050082	1.553134002	1.27095237	1.566272024	1.56147984	1.542177686
0.60	1.84960000	1.857631698	1.832273071	1.834751492	1.841390474	1.84094486	1.83084787
0.70	2.22010000	2.227161794	1.182707451	2.04914248	2.309510318	2.217504127	2.117567706
0.80	2.68960000	2.606829341	2.471141127	2.437480123	2.695816658	2.68567846	2.503553689
0.9	3.27610000	3.269520981	3.249099906	3.246227027	3.397761704	3.270388416	3.260186001
1.00	4.0000000	4.100729226	4.041190135	3.90012783	4.135082445	3.991955811	3.978864797

TABLE 16

COMPARATIVE ANALYSIS OF THE PROPOSED SCHEME AND CLASSICAL RUNGE KUTTA METHOD ON PROBLEM 3

Variable Step	Exact Solution	PROPOSED SCHEME			CLASSICAL RUNGE KUTTA METHOD		
x	$y(x_n)$	ONE-STAGE	TWO-STAGE	3-STAGE	ONE-STAGE	TWO-STAGE	3-STAGE
0.01	0.990049833	0.990099009	0.990099009	0.990099009	0.990000000	0.994950005	0.990099009
0.005	0.995012479	0.980296048	0.985080267	0.985080267	0.98010000	0.989926002	0.99009809
0.0025	0.997503122	0.970590144	0.975367836	0.97547248	0.97029900	0.979926759	0.99009708
0.00125	0.99875078	0.960980341	0.966727422	0.965955665	0.96059601	0.970032923	0.99009607
0.000625	0.999375195	0.990099009	0.956182622	0.956540852	0.950990049	0.965133771	0.990095051
0.0003125	0.999687548	0.980296049	0.946732492	0.947210257	0.941480144	0.955386885	0.990094021
0.00015625	0.999843762	0.970590148	0.938355346	0.938221515	0.932065342	0.945738432	0.990093009
0.000078125	0.999921878	0.960980344	0.929082045	0.929068582	0.922744688	0.940962925	0.990091959
0.000039062	0.999960938	0.951465688	0.919962746	0.920072981	0.913517241	0.93146014	0.990090906
0.000019531	0.999980468	0.942045236	0.910871894	0.911135472	0.904382068	0.926756732	0.990089839

CHAPTER SIX

GENERAL CONCLUSION

6.1 SUMMARY

The project has considered the design, analysis and implementation of a new class of explicit Runge-Kutta scheme for numerical solution of first order initial value problems in ordinary differential equation. The new schemes are consistent, convergent and A-stable. Comparative analysis of the proposed schemes with classical Runge-Kutta. Tables 15 – 16 indicates that, the performance of the new schemes is better than the existing Runge-Kutta scheme. Results in Tables 1 – 11 of the proposed schemes are accurate. The advantages of the new schemes over the existing Runge Kutta schemes, are that:

The interval of absolute stability of the proposed one- stage scheme is $(-\infty, 0)$ which is larger compared to the absolute stability interval of $(-2, 0)$ for the one- stage R-K method.

That is: the new class of explicit Runge- Kutta schemes is A- stable while the conventional Runge- Kutta methods of the same stage and order is not A- stable.

6.2 LIMITATIONS

One obvious drawback of the new scheme is that it involves more functions evaluation than the existing R- K methods.

Also its derivation and implementation is difficult and time-nconsuming but its high accuracy and A- stability compensate for these difficulties

6.2 RECOMMENDATIONS

(i) Adopting strategies which can help to find appropriate balance between the step size h, the order of accuracy, and stability of the methods to achieve more accuracy, efficient and effective schemes are essential.

These strategies include the choice of step length h as to satisfy

$h = \frac{y_n}{y_n'} \neq 0$. This value of h will over step the point of singularity.

(ii) It is important to double precision arithmetic to minimize the effects of round-off error.

(iii) Further research should look into higher stage methods with a view to identify the general order of the method.

6.3 CONTRIBUTION TO KNOWLEDGE

The new scheme shows that there exists explicit one-step scheme that are A-stable contrary to Dahlquist (1963) postulate. A new class of Explicit Runge-Kutta schemss valid for integration of non-stiff and stiff IVPs in ODES have been developed. The schemes will be quite suitable and useful in the solution of ODE problems arising from nuclear reaction processes, economic system, population model and other dynamic process such as heat and matter transfer, diffusion and pharmaco kinectic theories, that lead to differential of this type.

Consequently, the new scheme will be easier to apply for solution of stiff ODES compared to implicit methods which involves the solution of system of Linear algebraic equation per unit step.

REFERENCES

Ademiluyi, R.A. (1987), "New hybrid methods for systems of stiff ordinary differential equations". Ph.D. Dissertation, University of Benin, Benin-City.

Ademiluyi, R.A. (2002), "A new one-step scheme for the integration of ordinary differential equations". Journal NMS, vol. 21. Pg 1-69

Ademiluyi, R.A. et. Al. (2002), "A new class of implicit Rational Runge-Kutta methods for the integration of stiff ordinary differential equation" Journal NMS, vol. 21, Pg 27-3

Babatola, P.O. (2000), "Implicit Rational Runge-Kutta methods for solving stiff systems of ordinary differential equations", M.Tech. Thesis, Federal University of Technology, Akure.

Birkoff, G. and **Varga, R.S.** (1965), "Discretization Errors for Well-set Cauchy problems", Int. Journals of Mathematics and Physics, vol. 44. Pg 1-23

Blum, E.K. (1952), "A modification of Runge-Kutta Fourth-Order method", Maths Computation, vol. 16. Pg 176-187

Buthcer, J.C. (1964), "Implicit Runge-Kutta Processes" Math Computation vol. 18. Pg 50-64

Dahlquist, D. (1963), "A special stability problem for linear multistep methods" BIT 3, Pg. 27-43.

Fatunla, S.O. (1980), "Numerical integration for stiff and highly osciliatory problem ordinary differential equation", Maths Computation vol. 34, pg. 374-390.

Gear, C.W. (1971), "DIFSUB for solution of ordinary differential equation", Communication of ACM, vol. 14. Pg 185-190

Gill, S. (1951), A process for step-by-step integration of differential equations in an Automatic Digital Computing Machine", Cambridge, vol. 47. pg 95-108

Hong Yuanfu (1982), "A class of A-stable or A(α)-stable Explicit Schemes", Computational and assymptatic methods for Boundary and interior layer, Proceeding of BAILJI Conferences Trinity College, Dublin, Pg. 236-248.

Henrici, P. (1962), "Discrete variable variable methods in ordinary differential equations", John Wiley Inc., New York.

Henrici, P. (1963), "Error propagation for Difference methods, John Wiley and Sons.

Jain, M.K. (1979), "Numerical Solution of ordinary differential equations", Wiley Eastern Limited.

King, R. (1966), "Runge-Kutta methods with Cnstrained Mnimum Error Bounds", Maths Comp. Vol. 20. pg 286-291

Lambert, J.D. (1973), "Computational methods in ordinary Differential Equations", University of Dundee, Scotland. pg 1-264

Lambert, J.D. and **B. Shaw** (1966), "A generalization of multistep methods for ordinary differential equations", Numerical Math. Comp. Vol. 20. pg 250-263

Okunbor, D.I. (1985), "Explicit Runge-Kutta Schemes for stiff systems of ordinary differential equations", M.Sc. Thesis, University of Benin, Benin- City.

Ralston, A. (1962), "Runge-Kutta with Minimum error bounds", Maths Computation vol. 16. pg 431-437

Richardson, L.F. (1927), "The deferred approach to the limit, 1-single Lattice", Trans. Roy. Soc. London, 226, 229-349.

APPENDIX I

```
C     THIS PROGRAM SOLVES STIFF INITIAL VALUE PROBLEM
      Y' = λ(Y − E(X)) + E'(X)

C     WHERE λ IS A COMPLEX NUMBER WITH NEGATIVE REAL PART
C     COMPARING ONE-STAGE EXPLICIT RUNGE
C     KUTTA OF ORDER ONE AND CLASSICAL
C     RUNGE KUTTA OF ORDER ONE
      EXPLICIT REAL *8(A-H, T, X-Z)
C     INITIALISE X, Y, H, T_OL, X_L
.     OPEN (UNIT = 3, FILE = MATHS OUT, 'STATUS = 'NEW')
      OPEN (UNIT = 1, FILE = 'MATHS.DATA', STATUS = 'NEW')
.     WRITE (3, 1)

1     FORMAT (12X, 'RESULTS OF EXPLICIT RUNGE KUTTA SCHEME OF
      ORDER ONE AND CLASSICAL RUNGE-KUTTA SCHEME OF ORDER ONE')
      WRITRE (3, 103)

103   FORMAT ('/, ' ..... '/,)
      WRITE (3,7)

7     FORMAT (6X, 'H, '20X, 'YEXACT', '13X, 'ORDER ONE',
      13X, 'E2', 13X, 'CLASSICAL RUNGE KUTTA OF ORDER ONE', 3X, 'E3')
      WRITE (3, 102)

102   FORMAT ('/, ..... '/,)
      H1    =    0.1D-3

8     HMIN  =    0.1953125D-5
      TOL   =    1.0D-3
      XL    =    1.0D0
      XO    =    0.0D0
      YO    =    1.0D0

C     INVOKE SUBROUTINE EXPRK AND RUNKUT
C     TO COMPUTE APPROXIMATE VALUE OF Y
      CALL EXPRK (XO, YO, H1, TOL, Y(xn), E2, YN)
      CALL RUNKUT (XO, YO, H1, Y(xn), E3, YP)

C     CHANGE THE STEPSIZE TO HALF OF THE STEPSIZE
      HN = 0.5D0 *H1
C     WRITE OUT RESULTS
      WRITE (3, 10), H1, Y(xn), Y, E2, YP, E3

10    FORMAT (D14, 8, 7X, 5(D16.8, 4X)
```

```
      Write (3, 101)

101   FORMAT ('/, 18('-'), '/, , 5(20('-'), '/,))
      IF (H1.LE.HMIN) THEN
      STOP
      ELSE
      H1 = HN
      GO TO 8
      END IF
       END

C     FUNCTION SUBPROGRAM TO COMPUTE THE FUNCTION OF F     FUNCTION
      F(x,y)
      EXPLICIT REAL *8(A-H, T, X-Z)
      F = IODO (Y-X*3) + 3DO*X*X
      RETURN
      END

C     FUNCTION SUBPROGRAM TO COMPUTE THE FUNCTION OF F     FUNCATION
      F(x, y)
      EXPLICIT REAL *8(A-H, T, X-Z)
      F = 100D0*(Y-X**3DO) + 3DO*X*X
      RETURN
      END
C     FUNCTION SUBPROGRAM TO COMPUTE THE FUNCTION OF F     FUNCTION
      F(x, y)
      EXPLICIT REAL *8(A-H, T, X-Z)
      F = -1000D0*(Y-X**3DO) + 3DO*X*X
      RETURN
      END
C     FUNCTION SUBPROGRAM TO COMPUTE
      YEXACT FUNCTION YEXACT (x,y)
      EXPLICIT REAL 88(A-H, T, X-Z)
      YEXACT = (X**3.0D0) + EXP(-10:0*X)
      RETURN
      END
C     SUBROUTINE EXPRK TO COMPUTE THE APPROXIMATE VALUE OF Y,
      USING
      RUNGE KUTTA METHOD OF ORDER ONE
      SUBROUTINE, (X, Y, H, Yn, ℓn)
      EXPLICIT REAL *8(A-H, T, X-Z)
      Yn+1 = Yn**2/Yn – H* y'n
      ℓn  =  ABS(Yn – Y)
```

```
C     SUBROUTINE EXPRK2 TO COMPUTE THE APPROXIMATE VALUE OF Y,
      USING TWO STAGE EXPLICIT RUNGE KUTTA OF ORDER TWO
      SUBROUTINE EXPRK (X, Y, H, TOL, TP, E, YN)
      EXPLICIT REAL *8(A-H, T, X-Z)
      AH1   =     0.0D0
      AH2   =     0.0D0
      B11   =     0.0D0
      B12   =     0.0D0
      B21   =     0.5D0
      B22   =     0.0D0
      D1    =     0.0D0
      D2    =     0.5D0

12    X1    =     X + D1*H
      X2    =     X + D2*H
      Z0    =     1.ODO/Y
      Z1    =     ZO + (B11*AH1) + (B12*AH2)
      Z2    =     Z0 + (B21*AH1) + (B22*AH2)
      Y1    =     1.0D0/Z1
      Y2    =     1.0D0/Z2
      EH1   =     -Z1*Z1*F(X1, Y1)*H
      EH2   =     -Z2*Z2*F(X2,Y2)*H
      A1    =     ABS (EH1 – AH1)
      A2    =     ABS (EH2 – AH2)
      IF (A1.LE.TOL AND A2.LE.TOL) THEN
      TSUM = 0.5D0*(EH1 + EH2)
      YN    =     Y/(1.0D0 + Y*TSUM)
      XN    =     X + H
      YP    =     YEXACT (XN, YN)
      E     =     ABS (YP – YN)
      ELSE
      AH1   =     EH1
      AH2   =     EH2
      GO TO 12
      END IF
      RETURN
      END

C     SUBROUTINE RUNKUT TO COMPUTE
C     APPROXIMATE VALUE OF Y USING
C     CLASSICAL RUNGE KUTTA SCHEME OR ORDER TWO.
      SUBROUTINE RUNKUT2 (X, Y, H, YP, E, YN)
      EXPLICIT REAL *8(A-H, T, X-Z)
      AK1   =     H*F(X,Y)
      X2    =     X + 1.0D-3*H)
      Y2    =     Y + (1.0D – 3*H*AK1)
```

```
      AK2   =     H*F(X2, Y2)
      TSUM  =     (AK1 + (2.0D-3*AK2))/2
      YN    =     Y + TSUM
      XN    =     X + H
      YP    =     YEXACT (XN, YN)
      E     =     ABS (YP – YN)
      RETURN
      END

C     SUBROUTINE EXPRK3 TO COMPUTE THE APPROXIMATE VALUE OF Y,
      USING THREE STAGE EXPLICIT RUNGE KUTTA OF ORDER THREE
      SUBROUTINE EXPRK (, Y, H, TOL, YP, E, YN)
      EXPLICIT REAL *8(A-H, T, X-Z)
      AH1   =     0.0D0
      AH2   =     0.0D0
      AH3   =     0.0D0
      B11   =     0.0D0
      B12   =     0.0D0
      B13   =     0.0D0
      B21   =     0.5D0
      B22   =     0.0D0
      B31   =     1.0D0
      B32   =     2.0D0
      B33   =     0.0D0
      D1    =     0.0D0
      D2    =     0.5D0
      D3    =     1.0D0

14    X1    =     X + D1*H
      X2    =     X + D2*H
      Z0    =     1.0D0/Y
      Z1    =     Z0 + (B11*AH1) + (B12*AH2) + (B13*AH3)
      Z2    =     Z0 + (B21*AH1) + (B22*AH2) + (B22*AH3)
      Z3    =     Z0 + (B31*AH1) + (B32*AH2) + (B33*AH3)
      Y1    =     1.0D0/Z1
      Y2    =     1.0D0/Z2
      Y3    =     1.0D0/Z3.
      EH1   =     -Z1*Z1*F(X1, Y1)
      EH2   =     -Z2*Z2F(X2, Y2)
      EH3   =     -Z3*Z3*F(X3, Y3)
      A1    =     ABS (EH1 – AH1)
      A2    =     ABS (EH2 – AH2)
      A3    =     ABS (EH3 – AH3)
IF (AI.LE.TL, A2.LE.TOL AND A3.LE.TOL)
THEN
TSUM = 1.5DO*(EH1 + EH2 + EH3)
```

```
YN    =    Y/(1.0D0 + Y*TSUM)
XN    =    X + H
YP    =    YEXACT (XN, YN)
E     =    (YP – YN)
ELSE
AH1   =    EH1
AH2   =    EH2
AH3   =    EH3
GO TO 14
END IF
RETURN
END

C     SUBROUTINE RUNKUT TO COMPUTE
      APPROXIMATE VALUES OF Y USING

C     CLASSICAL RUNGE KUTTA SCHEME OF ORDER THREE
      SUBROUTINE RUNGKUT 3(X, Y, H, TP, E, YN)
      EXPLICIT REAL *8(A-H, T, X-Z)
      AK1   =    H*F (X, Y)
      X2    =    X + 0.5D0*H
      Y2    =    Y + (0.5D*AK1)
      AK2   =    H*F(X2, Y2)
      Y2    =    Y + (1.0D0*AK2)
      AK2   =    H*F(X2, Y2)
      Y3    =    Y + (1.0D0*AK2)
      AK3   =    H*F(X2, Y3)
      TSUM  =    AK1 + (2.0D0*(AK2 + AK3))/6.0D0
      YN    =    Y + TSUM
      XN    =    X + H
      YP    =    YEXACT (XN, YN)
      E     =    ABS (YP – YN)
      RETURN
      END
```

APPENDIX 2

```
C     THIS PROGRAM SOLVES STIFF INITIAL
C     INITIAL VALUE PROBLEM Y' = λ(Y – EX) + E'(X)
C     USING TWO STAGE EXPLICIT RUNGE KUTTA OF ORDER TWO
C     USING ERROR ESTIMATES TO ADJUST THE STEP SIZE. THE STRATEGY
      IS TO
C     CONTROL THE STEPSIZE TO GET MINIMUM ERROR
      EXPLICIT REAL*8(A – H, T – Z)
      OPEN (UNIT = 3, FILE = 'MATHS 2 OUT', STATUS = 'NEW')
      WRITE (8,*) 'RETURN OF TWO STAGE EXPLICIT RUNGE KUTTA OF ORDER
      TWO, ADOPTING  RICHARDSON EXTRAPOLATION METHOD TO ESTIMATE
      THE LOCAL TRUNCATION ERROR AND CONTROL THE STEP SIZE
      WRITE (8, 100)

100   FORMAT ('/,, ' ..... '/,)
      WRITE (8, 11)
      FORMT (2X, ' H, YEXACT, VALUE OF
           Y,* NUMERICAL VALUE OF Y, LOCAL ERROR
           ERROR
      WRITE (8, 120)

120   FORMAT ('/,, ..... '/,)
      XO    =     0.0D0
      YO    =     1.0D0
      HI    =     0.1D-1
      TOL   =     0.1D-2
      XL    =     1.0D0

2     XP    =     XO + HI
      CALL ADAPT (X, Y, TOL, HN, LTE,)
      CALL EXPRK2 (XO, YO, HI, TOL, YNP)
      YN    =     YEXACT (XP, YNP)
      E     =     ABS (YN – YNP)
      WRITE (8, 3) HN, XP, YN, YNP, LTE, E.

3     FORMAT (2 X, 6 (16.8, 4X)
      WRITE (8, 100)

100   FORMAT ('/,, 6(20 ('-'), '/,))
      IF (XP. GE. XL) STOP
      XO    =     XP
      YO    =YNP
      GO TO 2
      END
C     SUBROUTINE ADAPT TO CONTROL THE STEP SIZE
```

```
      SUBROUTINE ADAPT (X, Y, T_OL, H_N, LTE)
      EXPLICIT REAL *8 (A – H, T – Z)
      H     =     0.1D-3

7     D_n   =     T_OL*Y_NP)
      H_2   =     0.5D0*H
      XPI   =     X + H_2
      CALL EXPRK (X, Y, H, T, YP_1)
      CALL EXPRK (X, Y, H_2, T, YP_2)
      CALL EXPRK (XP_1, YP_2, H_2, T, YP_3)
      SET LTE = (Y_N – Y_1) ( 2^(r+1) / (2^(r+1) – 1) )
      DP    =     ABS (YP_3 – YP_1)
      DNP   =     ABS (D_N/D_P)
      EA    =     (16.0D0*D_P)/15.0D0
                        IF (DP.LE.DN) THEN
      HN    =     H*ABS (DNP**0.2DO)
      ELSE
      HN    =     H*ABS (DNP**0.25D0)
      END IF
      IF (LTE.LE.E) THEN
      GO TO 8
      ELSE
      H = HN
      GO TO 7
      END IF

8     RETURN
      END

C     SUBROUTINE EXPRK TO COMPUTE NUMERICAL VALUES OF Y
      SUBROUTINE EXPRK (X, Y, H, T, Y_P)
      EXPLICIT REAL *8(A – H, T – Z)
C     INITIALISE THE VARIABLES
      AH_1  =     0.D0
      AH_2  =     0.D0
C     FUNCTION SUBPROGRM TO COMPUTE F
      FUNCTION F(x, y)
      EXPLICIT REAL *8(A – H, T – Z)
      SET
      Y_n+2 =     Y/(1.0d0 + Y*(-z_1*z_1*F (X_1, Y_1)* H + (-z_2*z_2
                  F(X_2, Y_2)H))/2.0D0
      E     =     ABS (Y_P – Y_N)
      RETURN
      END
```