AN EFFICIENT IMPLEMENTATION TECHNIQUE FOR IMPLICIT RUNGE-KUTTA GAUSS METHODS IN SOLVING MATHEMATICAL STIFF PROBLEMS

MOHD HAFIZUL BIN MUHAMMAD

05-4506832 pustaka.upsi.edu.my Perpustakaan Tuanku Bainun Kampus Sultan Abdul Jalil Shah PustakaTBainun ptbupsi

THESIS SUBMITTED IN FULFILLMENT OF THE REQUIREMENT FOR THE DEGREE OF MASTER OF SCIENCE (APPLIED MATHEMATICS) (MASTER BY RESEARCH)

FACULTY OF SCIENCE AND MATHEMATICS
UNIVERSITI PENDIDIKAN SULTAN IDRIS

2018

05-4506832 pustaka.upsi.edu.my Perpustakaan Tuanku Bainun Kampus Sultan Abdul Jalil Shah PustakaTBainun ptbupsi

ABSTRACT

This research is aimed to produce an efficient implementation technique for the 2-stage (G2) and 3-stage (G3) implicit Runge-Kutta Gauss methods in solving mathematical stiff problems. Both methods are constructed by using Maple software and have been implemented by using Matlab software numerically. This research applied four implementation strategies which are full Newton without compensated summation (FNWSN), full Newton with compensated summation (FNCS), simplified Newton without compensated summation (SNWCS) and simplified Newton with compensated summation (SNCS). Comparison have been done with the implementations of Hairer and Wanner scheme, Cooper and Butcher scheme, and González scheme. Results for stiff test problems showed that SNCS is the most efficient technique in solving some real life mathematical problems such as the Kepler, Oregonator, Van der Pol, HIRES and Brusselator problems. According to the numerical results, the implementation of G2 using SNCS by the Hairer and Wanner scheme is the most efficient technique for solving Kepler and Brusselator problems, while SNCS by the González scheme is the most efficient technique for solving other problems. On the contrary for G3, SNCS by the Hairer and Wanner scheme gives the most efficient technique for solving Kepler and Van der Pol problems, while SNCS by the González scheme gives the most efficient technique for solving other problems. In conclusion, for both G2 and G3 methods, SNCS plays an important role to improve the efficiency of implicit Runge-Kutta Gauss methods in solving mathematical stiff problems. As for the implications, the implementation technique used in this research can be extended during tertiary education on the subject numerical ordinary differential equations that focusses on implementation schemes by other researchers as well as to some other implicit Runge-Kutta methods.

TEKNIK PELAKSANAAN YANG EFISIEN BAGI KAEDAH RUNGE-KUTTA GAUSS TERSIRAT DALAM MENYELESAIKAN MASALAH MATEMATIK KAKU

ABSTRAK

Kajian ini bertujuan untuk menghasilkan satu teknik perlaksanaan yang efisien bagi kaedah peringkat-2 (G2) dan peringkat-3 (G3) kaedah tersirat Runge-Kutta Gauss dalam menyelesaikan masalah matematik kaku. Kedua-dua kaedah ini diterbitkan menggunakan perisian Maple and dilaksanakan menggunakan perisian Matlab secara berangka. Kajian ini menggunakan empat strategi perlaksanaan iaitu Newton penuh tanpa lebihan penjumlahan (FNWCS), Newton penuh bersama lebihan penjumlahan (FNCS), Newton yang dipermudahkan tanpa lebihan penjumlahan (SNWCS), Newton yang dipermudahkan bersama lebihan penjumlahan (SNCS). Dalam perlaksanaan ini juga, perbandingan telah dibuat ke atas perlaksanaan skim Hairer dan Wanner, skim Cooper dan Butcher dan skim González. Berdasarkan keputusan masalah ujian matematik kaku, SNCS adalah teknik perlaksanaan yang paling efisien yang telah digunakan dalam menyelesaikan masalah matematik sebenar seperti masalah Kepler, Oregonator, Van der Pol, HIRES dan Brusselator. Berdasarkan keputusan masalah berangka, teknik perlaksanaan G2 menggunakan SNCS skim Hairer dan Wanner adalah teknik yang paling efisien dalam menyelesaikan masalah Kepler dan Brusselator manakala SNCS skim González adalah teknik yang paling efisien dalam menyelesaikan masalah-masalah lain. Berbeza pula bagi G3, SNCS skim Hairer dan Wanner adalah teknik perlaksanaan paling efisien dalam menyelesaikan masalah Kepler and Van der Pol manakala SNCS skim González adalah teknik perlaksanaan paling efisien dalam menyelesaikan masalah yang lain. Kesimpulannya, bagi kedua-dua kaedah G2 dan G3, SNCS adalah sangat penting untuk meningkatkan keefisienan dalam menyelesaikan masalah matematik kaku. Sebagai implikasi, teknik perlaksanaan yang digunakan dalam kajian ini boleh dikembangkan dalam pengajian tinggi dalam subjek penyelesaian berangka persamaan pembezaan biasa yang menumpukan teknik skim pelaksanaan oleh skim penyelidik lain dan juga kepada beberapa kaedah Runge-Kutta tersirat yang lain.

CONTENTS

05-4506832 pustaka.upsi.edu.my Perpustakaan Tuanku Bainun Kampus Sultan Abdul Jalil Shah PustakaTBainun ptbupsi

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

1.1 Introduction to Numerical ODEs

Ordinary differential equations (ODEs) represent a mathematical model for many systems in various discipline of knowledge. Unfortunately, most of the ODEs have no equations for exact solution therefore numerical methods play as an important technique to approximate the solution for the ODEs system. Nowadays, highly accurate solution for many kinds of complicated ODEs can be obtained by numerical approximation with the help of sophisticated software for computational mathematics.

1.1.1 Ordinary Differential Equations

Differential equations can be used to solve many system in real life problems including chemical, physical and biological processes. Ordinary differential equations are a type of differential equations that consist of derivatives of unknown solution with respect to only one independent variable.

First order ODEs can be written in the following form:

$$y' = f(x,y), \;\; y(x_0) = y_0, \;\; f : [x_0,x_n] \times \mathbb{R}^n \to \mathbb{R}^n. \tag{1.1}$$

f is *autonomous* if it is a function of only y. But it is called *non autonomous* if f is explicitly depends on x. In the equation, x is time variable or known as the independent variable and y is the dependent variable, x_0 is the initial time and y_0 is the initial value. Function f is used to determine the unknown function y satisfying the ODEs. $\mathbb{R}^n$ is a set of real number and N is a set of positive integers. If the value of x_0 and y_0 are given, then equation (1.1) is known as the initial value problems.

Ordinary differential equation also can be solved analytically. However, analytical approach are difficult to solve stiff ODEs problem. Numerical method is required to solve this kind of ODEs problem. Therefore, approximation of a solution can be performed when the exact solution of the ODEs problem is unknown.

Runge-Kutta (RK) methods, linear multistep methods and general linear methods are among the popular numerical methods used nowadays. Butcher mentioned that Runge-Kutta methods only involve one step method due to the finding by Runge, Heun and Kutta (Butcher, 2016). One of a simple RK method is the explicit Euler method.

In 1883, Bashforth and Adam invented the linear multistep method (Bashforth, 1883). This method is an extended Euler method whereby the several previous solutions and derivative values are used in approximating the solution at a particular point.

In 1996, Butcher proposed general linear methods (Butcher, 2016). It is formulated by Runge-Kutta and linear multistep methods in natural mean. General linear methods are advantages in terms of the capabilities of having one to large number of input quantity and stage as its inherent from Runge-Kutta and linear multistep methods. It is also known as multistage-multivalue methods. Multivalue method is a method that collect input in vectors forms at the beginning of step and a similar collection is passed on as an output from the current step and as input into the following step. Multistage method is a computation in forming the output quantities. In this research, only implicit RK method will be focused. A brief introduction to Runge-Kutta methods is given in subsection 1.1.2

In some applications such as in science and engineering, there exist a complicated phenomena that involve very wide time scales for example the problems that involve energy conservation, combustion, density, pressure and temperature conditions (Faou et al., 2004) and (Kadoura et al., 2014). For mathematicall modelling, stiffness is a special parameter that can be found in the ODEs system. A system is stiff when it involves different components that changing rapidly and slowly together. In literature, some researchers defined stiff problem as a system that covers *widely differing time constant* or mathematically is a system *with a large Lipschitz constant* (Kadoura et al., 2014). It means that if the partial derivatives $\frac{\partial f}{\partial y}$ are continuous and bounded, the Lipschitz constant is defined by the following equation.

$$L = \left\| \frac{\partial f}{\partial y} \right\| > \rho, \tag{1.2}$$

where ρ is defined by

$$\rho = \|\lambda_i\|, \;\; i = 1,2,...n. \tag{1.3}$$

and λ is the eigenvalues of $\frac{\partial f}{\partial y}$.

To understand stiffness, consider the Prothero Robinson (PR) problem which is given in equation (1.4).

$$y' = \lambda(y - g(x)) + g'(x), \quad y(0) = g(0), \tag{1.4}$$

where $g(x) = \sin(x)$ with exact solution $y(x) = g(x)$ and λ is stiffness parameter. When λ is large negative number such -1000000, PR problem is considered as a stiff problem resulting in using a much smaller stepsize to achieve convergence solution and as requested by stability (Gorgey, 2012) and (Butcher, 2016). A detailed explanation on stiff problems is given in section 1.2

Any type of numerical methods have errors that can spoil the solution or cause less efficient and less accurate solution. Generally, common error is divided into two types. Namely local and global errors. Local error is a type of error that is produced by numerical method in a single step where the value at the beginning of that step is assumed to be exact. Global error is another type of error that is caused by the accumulation of local errors after n steps. The accumulation does not means that the summation of local errors at each n steps but it is bounded by the sum of the bounds on the local errors (Butcher, 2016). Local errors, l_n can be defined by

$$l_n = u_n(x_n) - y_n, \tag{1.5}$$

where, u_n is the solution curve and y_n is exact solution curve. The global error, ε_n is given by

$$\varepsilon_n = y(x_n) - y_n, \tag{1.6}$$

where $y(x_n)$ is the solution curve at n steps. Equation (1.6), ε_n can be rewrite as

$$\varepsilon_n = y(x_n) - u_n(x_n) + l_n, \tag{1.7}$$

where ε_n is the actual error after n steps. Hence, there are two components of global error, one is due to the local errors at the present step and the other is due to the local errors at the previous steps.

Beside these errors, there is another error known as round-off errors. These errors can destroy the numerical solutions if it is significant in numerical approximation. Detailed about round-off errors is given in chapter 4 on section 4.1.4. In the next section, a brief introduction on Runge-Kutta methods is given.

1.1.2 Introduction to Runge-Kutta Methods

Runge-Kutta methods are developed to find the approximate solution for ordinary differential equations. In general, the Runge-Kutta methods can be defined as

$$Y_i = y_{n-1} + h\sum_{j=1}^{s} a_{ij} f(x_{n-1} + c_j h, Y_j), \tag{1.8}$$

$$y_n = y_{n-1} + h\sum_{j=1}^{s} b_j f(x_{n-1} + c_j h, Y_j), \tag{1.9}$$

where $i, j = 1, 2, ..., s$, s is the number of stage. Y_i represents the internal stage values and y_n represent the updated of y at the n^{th} step. a is the coefficient used to find the internal stages using the linear combinations of the stage derivatives. b represents the quadrature weights which indicates how the approximation to the solution depends on the derivatives of the internal stages. c is the vector of abscissas which indicates the positions within the step of the stage values. The coefficient a and c must hold the row-sum condition such as given in the Table 1.1. The coefficient in the general equa-

tion (1.8) and (1.9) can be represented by a partitioned tableau known as the Butcher tableau (Butcher, 2016) of the form

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

where A is a matrix that consist of the a values of RK methods.

Table 1.1
Row-Sum Condition of Explicit RK and Implicit RK Methods

Explicit RK	Implicit RK
$\sum_{j=1}^{i-1} a_{ij} = c_i$	$\sum_{j=1}^{s} a_{ij} = c_i$
$i, j = 2, ..., s$	$i, j = 1, ..., s$

The row-sum condition differentiate the implicit RK (IRK) and the explicit RK (ERK). The sum over j is goes up to $i-1$ for the explicit RK and form the triangular matrix A contains the coefficient a (Deuflhard and Bornemann, 2012). The classical Runge-Kutta method of order-4 (RK4) is an example of popular ERK. In the implicit RK, the sum over j is up to s and the coefficient matrix A is not triangular. Implicit methods can be divided into several other categories, for example, fully-implicit if matrix A is not lower triangular, semi-implicit if A is lower triangular with at least one non-zero diagonal element, diagonal implicit if A is lower triangular with all the diagonal elements are equal and non-zero diagonally implicit Rune-Kutta (DIRK) and singly implicit if A is matrix with a single non-zero eigenvalue singly implicit Runge-Kutta (SIRK). Table 1.2 describe these properties.

Explicit methods are easy to implement as the internal stages can be calculated sequentially without depending on later stages and are also easy to code. However, explicit methods cannot be used to solve stiff problems since they have poor stability behavior (refer to section 1.2). On the other hand, implicit methods are suitable for

Table 1.2
Butcher Tableau of Explicit RK and Implicit RK Methods

Explicit RK

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

Implicit RK

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

solving stiff problems but are more costly to implement. The implementation of implicit methods is discussed in Chapter 3 on section 3.4.

Some explicit methods are the Euler's method, explicit midpoint rule, explicit trapezoidal rule and other higher order explicit methods. The explicit Euler method is the simplest explicit Runge-Kutta method which is of order-1. The examples of implicit methods are the implicit Euler method, implicit midpoint rule, implicit trapezoidal rule, Gauss methods, Lobatto methods and other higher order implicit methods. In fact, only 2-stage (G2) and 3-stage (G3) Gauss methods are involved in this research. Table 1.3 gives the Butcher tableau for the 2-stage and 3-stage Gauss methods.

Table 1.3
Butcher Tableau of G2 and G3 Methods

2-stage Gauss

$$\begin{array}{c|cc} \frac{1}{2}-\frac{\sqrt{3}}{6} & \frac{1}{4} & \frac{1}{4}-\frac{\sqrt{3}}{6} \\ \frac{1}{2}+\frac{\sqrt{3}}{6} & \frac{1}{4}+\frac{\sqrt{3}}{6} & \frac{1}{4} \\ \hline & \frac{1}{2} & \frac{1}{2} \end{array}$$

3-stage Gauss

$$\begin{array}{c|ccc} \frac{1}{2}-\frac{\sqrt{15}}{10} & \frac{5}{36} & \frac{2}{9}-\frac{\sqrt{15}}{15} & \frac{5}{36}-\frac{\sqrt{15}}{30} \\ \frac{1}{2} & \frac{5}{36}+\frac{\sqrt{15}}{24} & \frac{2}{9} & \frac{5}{36}-\frac{\sqrt{15}}{24} \\ \frac{1}{2}+\frac{\sqrt{15}}{10} & \frac{5}{36}+\frac{\sqrt{15}}{30} & \frac{2}{9}+\frac{\sqrt{15}}{15} & \frac{5}{36} \\ \hline & \frac{5}{18} & \frac{4}{9} & \frac{5}{18} \end{array}$$

For the 2-stage Gauss method, the defining equations are given in equation (1.10), (1.11) and (1.12) while for the 3-stage Gauss method the defining equations are given in equation (1.13), (1.14) and (1.15).

The stage equations of 2-stage Gauss are defined by

$$Y_1 = y_{n-1} + h\left(\frac{1}{4}\right)F_1 + h\left(\frac{1}{4} - \frac{\sqrt{3}}{6}\right)F_2,$$

$$Y_2 = y_{n-1} + h\left(\frac{1}{4} + \frac{\sqrt{3}}{6}\right)F_1 + h\left(\frac{1}{4}\right)F_2. \tag{1.10}$$

The internal stage derivative equations of the 2-stage Gauss are defined by

$$\begin{aligned} F_1 &= f\left(x_{n-1} + h\left(\frac{1}{2} - \frac{\sqrt{3}}{6}\right), Y_1\right), \\ F_2 &= f\left(x_{n-1} + h\left(\frac{1}{2} + \frac{\sqrt{3}}{6}\right), Y_2\right). \end{aligned} \tag{1.11}$$

The update equation of the 2-stage Gauss is defined by

$$y_n = y_{n-1} + h\left(\frac{1}{2}\right)F_1 + h\left(\frac{1}{2}\right)F_2. \tag{1.12}$$

The equations (1.13) are the stage equations of the 3-stage Gauss method.

$$\begin{aligned} Y_1 &= y_{n-1} + h\left(\frac{5}{36}\right)F_1 + h\left(\frac{2}{9} - \frac{\sqrt{15}}{15}\right)F_2 + h\left(\frac{5}{36} - \frac{\sqrt{15}}{30}\right)F_3, \\ Y_2 &= y_{n-1} + h\left(\frac{5}{36} + \frac{\sqrt{15}}{24}\right)F_1 + h\left(\frac{2}{9}\right)F_2 + h\left(\frac{5}{36} - \frac{\sqrt{15}}{24}\right)F_3, \\ Y_3 &= y_{n-1} + h\left(\frac{5}{36} + \frac{\sqrt{15}}{30}\right)F_1 + h\left(\frac{2}{9} + \frac{\sqrt{15}}{15}\right)F_2 + h\left(\frac{5}{36}\right)F_3. \end{aligned} \tag{1.13}$$

The internal stage derivative equations of 3-stage Gauss are defined by

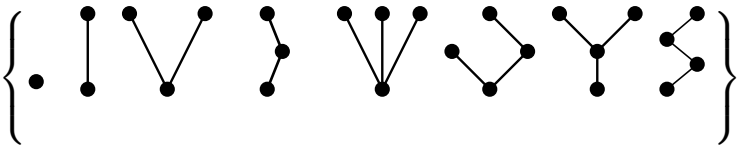
$$\begin{aligned} F_1 &= f\left(x_{n-1} + h\left(\frac{1}{2} - \frac{\sqrt{15}}{10}\right), Y_1\right), \\ F_2 &= f\left(x_{n-1} + h\left(\frac{1}{2}\right), Y_2\right), \\ F_3 &= f\left(x_{n-1} + h\left(\frac{1}{2} + \frac{\sqrt{15}}{10}\right), Y_3\right). \end{aligned} \tag{1.14}$$

The update equation of 3-stage Gauss is defined by

$$y_n = y_{n-1} + h\left(\frac{5}{18}\right) F_1 + h\left(\frac{4}{9}\right) F_2 + h\left(\frac{5}{18}\right) F_3. \tag{1.15}$$

Next, in the following section, the theory of Runge-Kutta methods such as the elementary weights, elementary differentials and the order conditions are given for the set of rooted trees (Butcher, 2016).

Let T denote the set of rooted trees up to order-4

$T =$  (1.16)

It is important to understand the theory of Runge-Kutta methods such as elementary differential, elementary weight and order conditions. The elementary differentials can be derived using chain rule. The following subsection shows the derivation up to fourth derivative. It also can be represented by the root trees t as defined in equation (1.16). The tree can be labeled using the coefficients A, b^T and c. The tree and the labels are also represent the order and order conditions of the Runge-Kutta methods respectively.

1.1.2.1 Elementary Differentials

The elementary differential are defined using chain rule and are given up to the fourth derivative as follows

$$y' = f(y(x)) = \mathbf{f},$$
$$y'' = f'(y(x))y' = f'(y(x))(f(y(x))) = \mathbf{f}'\mathbf{f},$$
$$y''' = f''(y(x))(f(y(x)), f(y(x))) + f'(y(x))(f'(y(x))(f(y(x)))) = \mathbf{f}''(\mathbf{f},\mathbf{f}) + \mathbf{f}'\mathbf{f}\mathbf{f},$$

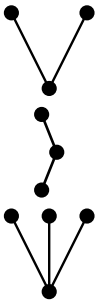
$$\begin{aligned} y^{(4)} &= f'''(y(x))(f(y(x)), f(y(x)), f(y(x))) + 3f''(y(x))(f(y(x)), f'(y(x))f(y(x))) \\ &\quad + f'(y(x))f''(y(x))(f(y(x))(f(y(x)))) + f'(y(x))f'(y(x))f'(y(x))f(y(x)), \\ &= \mathbf{f}'''(\mathbf{f},\mathbf{f},\mathbf{f}) + \mathbf{f}''(\mathbf{f},\mathbf{f}'\mathbf{f}) + \mathbf{f}'\mathbf{f}''(\mathbf{f},\mathbf{f}) + \mathbf{f}'\mathbf{f}'\mathbf{f}'\mathbf{f}. \end{aligned}$$

The expressions $\mathbf{f}, \mathbf{f}'\mathbf{f}, \ldots$ are known as the elementary differentials. Elementary differentials are related to the rooted-trees. Hence, for up to order four, the trees can be constructed such as

$$y' = \mathbf{f}$$

$$y'' = \mathbf{f}'\mathbf{f}$$

$$y''' = \mathbf{f}''(\mathbf{f},\mathbf{f}) + \mathbf{f}'\mathbf{f}'\mathbf{f}$$

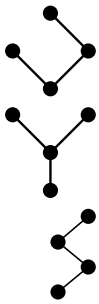


$$y^{(4)} = \mathbf{f}'''(\mathbf{f},\mathbf{f},\mathbf{f})$$

$$+\mathbf{f}''(\mathbf{f},\mathbf{f}'\mathbf{f})$$

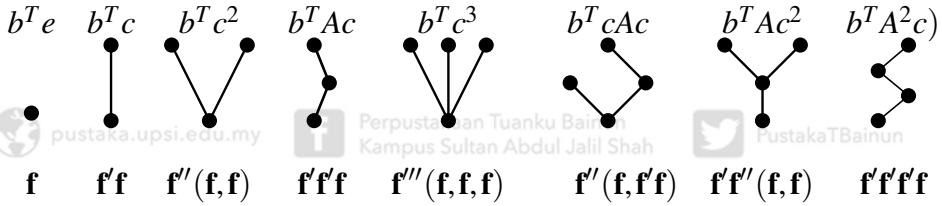
$$+\mathbf{f}'\mathbf{f}''(\mathbf{f},\mathbf{f})$$

$$+\mathbf{f}'\mathbf{f}'\mathbf{f}'\mathbf{f}$$



1.1.2.2 Elementary Weight and Order Conditions

A tree can be labeled using the coefficients A, b^T and c. The root is labelled with b^T , terminal vertices with c and non-terminal vertices with A. Corresponding to each t is a real number called the elementary weights and are denoted by $\Phi(t)$ (Butcher, 2016).



Comparison of successive term in Taylor series expansions of computed solution with the exact solutions is used to derived the order conditions of Runge-Kutta methods. The order conditions are used to investigate the error in carrying out a single step of a Runge-Kutta method. The exact solution at x_n, $y(x_n) = y(x_{n-1}+h)$ to order p is represented by the Taylor series expansion.

The formal Taylor expansion of the solution at $x_n + h$ is given by

$$\begin{aligned} y(x_{n-1}+h) &= y(x_{n-1}) + hy'(x_{n-1}) + \frac{h^2}{2!}y''(x_{n-1}) + \ldots + \frac{h^p}{p!}y^{(p)}(x_{n-1} + O(h^{p+1}), \\ &= y(x_{n-1}) + \sum_{k=1}^{p} \frac{h^k}{k!}y^{(k)}(x_{n-1}) + O(h^{p+1}). \end{aligned}$$

The Taylor series expansion of the exact solution with $y(x_0) = y_0$ has the following

form (Butcher, 2016):

$$\begin{aligned} y(x_{n-1}+h) &= y(x_{n-1}) + \sum_{k=1}^{p} \frac{h^k}{k!} \sum_{r(t)=k} \alpha(t) F(t)(y(x_{n-1})) + O(h^{p+1}), \\ &= y(x_{n-1}) + \sum_{k=1}^{p} \frac{h^k}{k!} \sum_{r(t)=k} \frac{k}{\sigma(t)\gamma(t)} F(t)(y(x_{n-1})) + O(h^{p+1}), \\ &= y(x_{n-1}) + \sum_{k=1}^{p} h^k \sum_{1} \frac{1}{\sigma(t)\gamma(t)} F(t)(y(x_{n-1})) + O(h^{p+1}), \end{aligned}$$

For all $t \in T$,

$$y(x_{n-1}+h) = y(x_{n-1}) + \sum_{t \in T} h^{r(t)} \frac{1}{\sigma(t)\gamma(t)} F(t)(y(x_{n-1})) + O(h^{p+1}).$$

The number of ways of labeling with an ordered set $\alpha(t)$ is given by

$$\alpha(t) = \frac{r(t)!}{\sigma(t)\gamma(t)},$$

and the number of ways labeling with an unordered set $\beta(t)$ with

$$\beta(t) = \frac{r(t)!}{\sigma(t)}.$$

From $\alpha(t)$ and $\beta(t)$, it is obtain

$$y_n(h) = \sum_{t \in T} \alpha(t) \frac{h^{r(t)}}{r(t)!} \gamma(t) \Phi(t) F(t)(y_0).$$

Since

$$y(x+h) = \sum_{t \in T} \alpha(t) \frac{h^{r(t)}}{r(t)!} F(t)(y_0).$$

Then $\gamma(t)\Phi(t) = 1$. Hence, the order conditions will be as follows (Butcher, 2016)

$$\text{Order } \; p = \Phi(t) = \frac{1}{\gamma(t)} \;\; \text{for } \; t : r(t) \leq p, \tag{1.17}$$

for $t : r(t) \leq p$, where

$r(t)$	Order of t, it is convenient with number of vertices.
$\sigma(t)$	Symmetry of t, it is convenient with order of automorphism group.
$\gamma(t)$	Density of t.
$\alpha(t)$	Number of ways of labeling with an ordered set.
$\beta(t)$	Number of ways labeling with an unordered set.
$F(t)(y_0)$	Elementary differential.

Brief detail of trees up to order-4 is given in Table 1.4. As the order increases, the number of order conditions increases rapidly and becomes unmanageable. For this reason, simplifying assumptions are introduced to simplify the order conditions (Butcher, 1963). In matrix form they are given by

$$B(p): \quad b^T c^{k-1} = \frac{1}{k}, \quad k = 1, ..., p, \tag{1.18}$$

$$C(q): \quad Ac^{k-1} = \frac{c^k}{k}, \quad k = 1, ..., q, \tag{1.19}$$

$$D(r): \quad b^T C^{k-1} A = \frac{1}{k}[b^T - b^T C^k], \quad k = 1, ..., r, \tag{1.20}$$

where $C = \text{diag}(c_1, ..., c_s)$. The $B(p)$ condition refers to the bushy trees ($\curlyvee$, $\curlyvee\!\!\curlyvee$, etc). The minimum of p and q when $B(p)$ and $C(q)$ hold are called the stage order.

To understand better about the simplifying assumption, consider Example 1.1.1.

Example 1.1.1

Verify that 2-Stage Gauss Method is Order-4.

2-stage Gauss method has $s = 2$. So it must satisfy $B(4)$ and $C(2)$.

$$B(p): \quad b^T c^{k-1} = \frac{1}{k},$$

$$b^T e = 1,$$

$$b^T c = \frac{1}{2},$$

$$b^T c^2 = \frac{1}{3},$$
$$b^T c^3 = \frac{1}{4}.$$

The computation are given in Appendix A referring to the Maple sheet eq1 till eq4.

$$C(q) : Ac^{k-1} = \frac{c^k}{k},$$
$$C(1) : Ae = c,$$
$$Ae = \begin{bmatrix} \frac{1}{4} & \frac{1}{4} - \frac{\sqrt{3}}{6} \\ \frac{1}{4} + \frac{\sqrt{3}}{6} & \frac{1}{4} \end{bmatrix} \begin{bmatrix} 1 \\ 1 \end{bmatrix} = c,$$
$$= \begin{bmatrix} \frac{1}{2} - \frac{\sqrt{3}}{6} \\ \frac{1}{2} + \frac{\sqrt{3}}{6} \end{bmatrix}.$$

$$C(2) : Ac = \frac{c^2}{2},$$
$$Ac = \begin{bmatrix} \frac{1}{4} & \frac{1}{4} - \frac{\sqrt{3}}{6} \\ \frac{1}{4} + \frac{\sqrt{3}}{6} & \frac{1}{4} \end{bmatrix} \begin{bmatrix} \frac{1}{2} - \frac{\sqrt{3}}{6} \\ \frac{1}{2} + \frac{\sqrt{3}}{6} \end{bmatrix},$$
$$= \begin{bmatrix} \frac{1}{6} - \frac{\sqrt{3}}{12} \\ \frac{1}{6} + \frac{\sqrt{3}}{12} \end{bmatrix},$$

$$\frac{c^2}{2} = \begin{bmatrix} \frac{(\frac{1}{2} - \frac{\sqrt{3}}{6})^2}{2} \\ \frac{(\frac{1}{2} + \frac{\sqrt{3}}{6})^2}{2} \end{bmatrix},$$
$$= \begin{bmatrix} \frac{1}{6} - \frac{\sqrt{3}}{12} \\ \frac{1}{6} + \frac{\sqrt{3}}{12} \end{bmatrix}.$$

Table 1.4
Elementary Weights and Order Conditions Up To Order-4 (Butcher and Hojjati, 2005)

$r(t)$	t	$\Phi(t)$	Order conditions
1		$\sum_{i=1}^{s} b_i = 1$	$b^T e = 1$
2		$\sum_{i=1}^{s} b_i c_i = \frac{1}{2}$	$b^T c = \frac{1}{2}$
3		$\sum_{i=1}^{s} b_i c_i^2 = \frac{1}{3}$	$b^T c^2 = \frac{1}{3}$
3		$\sum_{i=1}^{s} b_i a_{ij} c_j = \frac{1}{6}$	$b^T Ac = \frac{1}{6}$
4		$\sum_{i=1}^{s} b_i c_i^3 = \frac{1}{4}$	$b^T c^3 = \frac{1}{4}$
4		$\sum_{i=1}^{s} b_i c_i a_{ij} c_j = \frac{1}{8}$	$b^T (cAc) = \frac{1}{8}$
4		$\sum_{i=1}^{s} b_i a_{ij} c_j^2 = \frac{1}{12}$	$b^T Ac^2 = \frac{1}{12}$
4		$\sum_{i=1}^{s} b_i a_{ij} a_{jk} c_k = \frac{1}{24}$	$b^T A^2 c = \frac{1}{24}$

Several IRK methods will be used in solving several ODEs problems. Some ODEs problems have the equation of exact solution. The difference between the approximate solution and the exact solution is called the error of approximation. Normally, the efficiency of the methods can be represented by the graph of the error versus the CPU time. In addition, the efficiency can be improved by proper method of implementation. Thus, several implementations were tested in solving several ODEs problems including chemistry and physics problems.

In many cases, ordinary differential equation plays an important rule in solving a simple linear equation. Several analytical methods can be used to solve the equations such as separable variable, factorisation, substitution and other methods. However, analytical solution for nonlinear equations are always hard to solve. On the other hand,

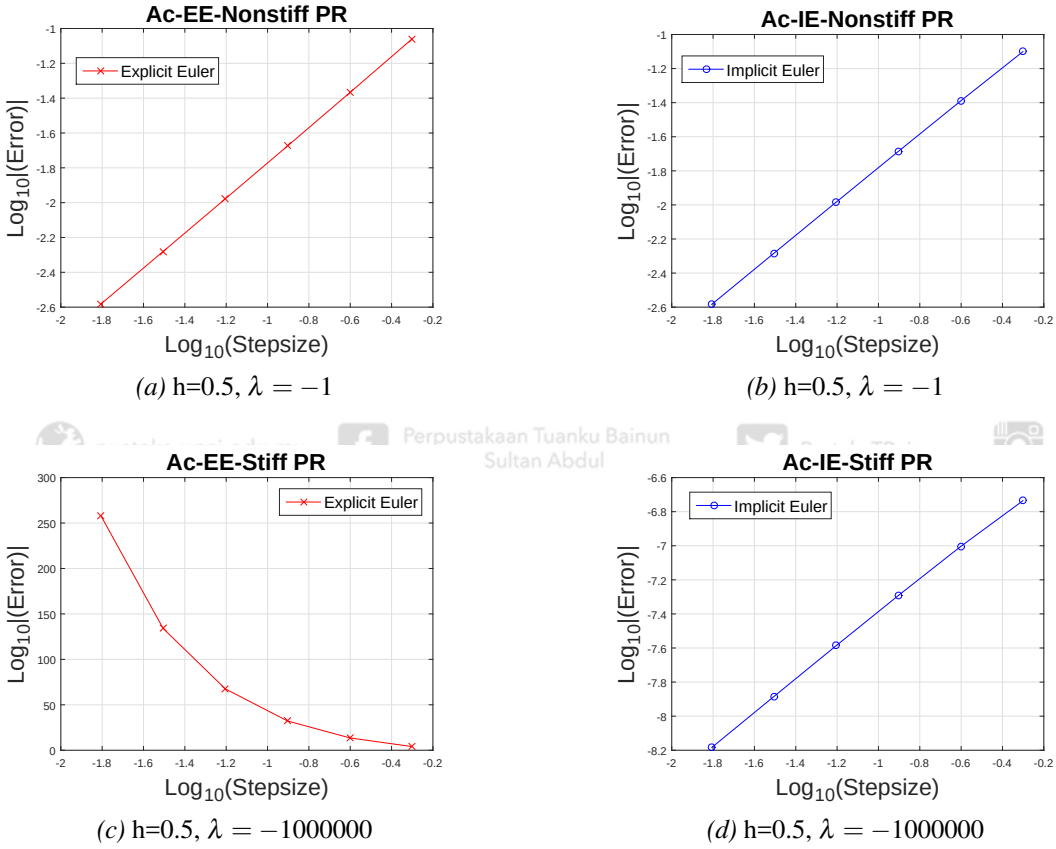
several type of numerical methods mentioned earlier is also quite important since it can solve for an approximate solution of the nonlinear equations whenever the exact solution is unknown. The combination of a good implementation and very small error can result a good approximate solution which is very close to the exact solution. The explicit and implicit Runge-Kutta methods are able to produce a good approximate solution for certain problems depend on the nature of equations.

The explicit and implicit Runge-Kutta methods are differ in term of the equations, coefficient and steps. Although explicit methods are easier to implement compared to the implicit methods, the methods need more time to obtain the approximate solution (Cash, 1975). The implementation is not significant when the time taken by explicit methods are more than double the time consumed by the implicit methods. The difference of processing time occurs because of the internal stage equations of the explicit methods depend on each other. The second stage equation need the value of the first stage equation and so on. On the other hand, for implicit method, every internal stage equations are independent which contribute to the shorten processing time. Besides, the explicit Runge-Kutta is less stability compare to implicit Runge-Kutta (Shampine, 1984).

1.2 Problem Statement

The explicit method is very easy to implementation as the internal stages can be calculated directly without depending on the later stages. Beside that, this method also incurs a cheap implementation cost. The stability of the explicit methods is classified as not A-stable (Iserles, 2009). Thus, the explicit methods cannot be used to solve stiff problems compared to the implicit methods as they have poor stability (Sanderse and Koren, 2012).

To clarify the property of stiffness, let use explicit Euler (EE) and implicit Euler (IE) to solve the PR problem. The Prothero-Robinson (PR) problem is consider stiff at -1000000. Figure 1.1 shows the accuracy of the EE and IE methods for the Prothero-Robinson problem with λ equal to -1 and -1000000 and stepsize h=0.5. It is observed that when λ=-1, EE and IE can solve stiff PR problem with good accuracy as shown in the Figure 1.1(a) and Figure 1.1(b).



(a) h=0.5, $\lambda = -1$ *(b)* h=0.5, $\lambda = -1$

(c) h=0.5, $\lambda = -1000000$ *(d)* h=0.5, $\lambda = -1000000$

Figure 1.1. The Accuracy of Explicit Euler and Implicit Euler for Nonstiff and Stiff Prothero Robinson Problem

However, when the value of λ=-1000000, Figure 1.1(c) shows the accuracy of EE was destroyed means the approximation was not efficient but IE method maintains with good accuracy. The destruction of the accuracy is because of the problem behave with the large value of stiffness parameter λ. Since λ is a large negative value, the first terms decay very quickly, leaving $y(x) = g(x)$. Therefore, a problem is stiff if the ability to find reliable numerical approximations to its solution hinges on stability rather than

accuracy alone, and when some components of the solution decay much more rapidly than others. For example, the explicit Euler is not A-stable or even A(α) stable for any $\alpha < \frac{\pi}{2}$. This is true for all explicit methods since none of the explicit can be A(α)-stable. Therefore, none of the explicit methods are suitable for solving stiff problems.

A detailed discussion on stiffness is given by Shampine and Gear (1979). They discuss ways of determining whether equations are stiff and the importance of stiff equations. Furthermore they comment that if the system is known to be very stable, the governing equations are likely to be stiff if some variables are known to change on time scales very different from others. Further reading on A-stability can be found in Butcher (2008).

The implicit methods play an important role in solving a stiff problems and differential algebraic equations. However, the implicit methods are expensive and difficult to implement as the stage-by-stage scheme is no longer available and need to be replaced by an iterative computation. Even it is difficult to implement, the implicit methods gives a fewer stages for the same order and better stability if compared to the explicit methods. Due to this better stability, the implicit methods are widely used in the applications of physics, chemistry and medical problems.

To construct a good method for solving stiff problems, one need to consider three important criteria such as high accuracy, good stability and a low implementation cost. Gauss methods have good stability as well as high accuracy, but the Gauss methods are very expensive to implement because of the implicit structure of their matrix's coefficients. In addition to that, the Gauss methods have different and complex eigenvalues which makes the implementation a little bit tricky.

For the family of Gauss methods, a group of researchers from Spain suggested a new implementation scheme to solve stiff problems but their investigations did not employed compensated summation (González-Pinto et al., 1994). Although some researchers have proposed a new implementation schemes of higher order IRK methods that minimize the computational cost, the modifications does not involve compensated summation (refer to section 4.2.3). Besides, there is also researcher claimed new efficient implementation based on Gauss methods that is proposed in 2014 but it solved linear and nonlinear ODEs for nonstiff problems (Agam and Yahaya, 2014). Therefore, in this research it will be of interest to use simplified Newton and compensated summation techniques into some implementation schemes proposed by some researchers such as the standard implementation scheme by Hairer and Wanner (1999), Cooper and Butcher (1983), González-Pinto et al. (1994) and González-Pinto et al. (1995) in solving some real life problems.

1.3 Research Objectives

The objectives of this research are

1. to study the effect of round-off errors for 2-stage and 3-stage Gauss methods in solving stiff problems.
2. to investigate the implementation of simplified Newton and full Newton with and without compensated summation.
3. to compare the implementation ideas from several researchers such as Cooper and Butcher (1983), González-Pinto et al. (1994), González-Pinto et al. (1995) and standard method by Hairer and Wanner (1999) for the 2-stage and 3-stage Gauss methods.
4. to find the most efficient implementation strategy for the 2-stage and 3-stage Gauss methods in solving stiff problems.

1.4 Research Questions

In this research, several questions of interest are wished to attempt. Some of these are

1. How are the effect of round-off errors in solving stiff problems by the 2-stage and 3-stage Gauss methods?
2. How are the implementation of simplified Newton and full Newton with and without compensated summation are done?
3. What is the difference between the implementation ideas by Cooper and Butcher (1983), González-Pinto et al. (1994), González-Pinto et al. (1995) and the standard method?
4. What is the most efficient implementation strategy for solving stiff problems?

1.5 Significant of Research

At the end of the research, it is hoped that

1. The computational cost for the implicit methods can be reduced by using the most efficient implementation strategy suggested.
2. The most efficient implementation strategy can be identified for the Gauss methods in solving stiff problems.
3. Researchers have broad knowledge on the implementation techniques for implicit Runge-Kutta methods.
4. Researchers can start using implicit Runge-Kutta methods which is proven to be more reliable and efficient in solving real life problems and widen the applications to real life problems.
5. The round-off errors can be reduced by using compensated summation technique.

1.6 Scope of Study

This research focuses on the implicit Runge-Kutta methods. The implicit methods that will be emphasized in this research are the 2-stage and 3-stage Gauss methods. The construction of the 2-stage and 3-stage Gauss methods are done using Maple 2016. Preliminary study is about understanding the ideas of implementation for implicit methods recommended by various researchers including the standard/common implementation method. The first stage in this research is to perform test problems using Prothero Robinson and Kaps problem. The best implementation strategies will be selected to solve real life problems in the numerical experiment. The numerical experiments are done on real life problems such as Kepler, Oregonator, Van der Pol, HIRES and Brusselator. All the numerical experiments are performed using MATLAB 2016 software.

1.7 Thesis Outline

There are 6 chapters in this thesis.

Chapter 1 is about the Introduction. This chapter include the background of this study, problem statement, objectives, significant of research and scope of this research.

Chapter 2 is the literature review. In this chapter, some history of implementation of Runge-Kutta methods, the implementation of 2-stage and 3-stage Gauss methods.

In Chapter 3, discussion on the construction of 2-stage and 3-stage Gauss methods, the implementation of implicit Runge-Kutta methods by other researchers, implementation of 2-stage and 3-stage Gauss methods in MATLAB.

Chapter 4 gives the test problems on the Prothero Robinson and Kaps problems. The best implementation of every strategies in the test problems was chosen for numerical experiment in solving real life problems in Chapter 5. The numerical results are given by the efficiency diagrams. The efficiency is measured using computational time versus error with certain stepsize.

Chapter 5 gives the numerical analysis of this research. This chapter give all the numerical results for real-life problems. The numerical results are also given by the efficiency diagrams.

Lastly, Chapter 6 gives the conclusions and the future work.