

Block Runge-Kutta Methods for Non-Stiff Initial Value Problems

A thesis presented for the degree of
Doctor of Philosophy of the University of London
and the
Diploma of Membership of Imperial College
by

Maria Isabel Coutinho Vieira

Department of Mathematics
Imperial College of Science, Technology and Medicine
Queen's Gate, London SW7 2BZ

December 1994

Abstract

Explicit Runge-Kutta methods are very widely used for the numerical integration of non-stiff initial value problems of the form:

$$\frac{dy}{dx} = f(x, y), \quad y(a) = y_0. \quad (0.1)$$

Recent work on Runge-Kutta methods has suggested several ways in which existing formulae can be improved. In particular, Dormand and Prince have developed a powerful approach to allow estimation of the global truncation error. Higham and Hall have investigated a new stability concept which is particularly appropriate when the steplength of integration is controlled by stability rather than accuracy and numerous authors have investigated the possibility of providing an interpolant to define a continuous solution which can be computed at “off-step” points. In another development, Cash has considered the possibility of applying block Runge-Kutta methods to non-stiff problems of the form (0.1). Such formulae can be regarded either as explicit Runge-Kutta methods integrating forward over two steps or standard Runge-Kutta methods using more than the minimum number of function evaluations. A particular 5(4) block method has been shown to be very competitive with standard methods of the same order. In this thesis we carry out a detailed investigation of block formulae. In particular we investigate the 5(4) case more thoroughly and also derive 6(5) and 7(6) formulae. In addition we derive block formulae which have good stability in the sense of Higham and Hall and which have proved to be very effective for mildly stiff problems. Finally we develop block formulae to compute global error estimates. Such formulae are particularly appropriate for this task since they contain many free parameters and also it is straightforward to derive an interpolant of the desired form. Extensive numerical results are given to demonstrate the efficiency of the new methods.

to my Parents

Acknowledgements

This thesis would never have been possible without the help and encouragement of my supervisor, Professor Jeff Cash, to whom I express my sincere gratitude. I also would like to thank Professor John Butcher for his support over these years of work.

My thanks go to the portuguese speaking community at Imperial College, particularly: Rosário, Guida, Anabela, Heloisa. I would also like to thank Simon, Nairo, Marwan, Theo, Poppy and Arlene for their support and friendship throughout my post graduate years at Imperial College – thanks!

Further, Phillip and Michalis for all their computing help and expertise.

Lastly (but not least) I am in great debt to ISCAP (Instituto Superior de Contabilidade e Administração do Porto) for allowing me to come here. Also, thanks to JNICT (Junta Nacional de Investigação Científica e Tecnológica) for their financial support.

Table of Contents

Chapter 1.	Introduction	10
1.1	Initial Value Problems	10
1.2	Numerical Methods	12
1.3	Stability and Convergence	17
1.4	Stiffness	20
1.5	Comparison Between RKM and LMM	23
Chapter 2.	Runge-Kutta Methods - A Survey	25
2.1	Introduction	25
2.2	Order versus Stages	26
2.3	Some Examples of Low Order Formulae	28
2.4	Butcher's Analysis and Relation to Rooted Trees	30
2.5	Simplifying Assumptions	32
2.6	Local Error Estimation	38
2.7	Stability	42
2.8	Interpolation	45
2.9	Main Codes Used at Present	46
Chapter 3.	Block Explicit Runge-Kutta Methods	47
3.1	Introduction	47
3.2	Accuracy Criteria	51
3.3	Stability Criteria	53

Table of Contents

3.4	Order 2	54
3.4.1	Block Formula of Order 2 with 3 Stages	56
3.4.2	Block Formula of Order 2 with 4 Stages	59
3.5	Order 3	65
3.5.1	Block Formula of Order 3 with 5 Stages	67
3.5.2	Block Formula of Order 3 with 6 Stages	72
3.6	Order 4	77
3.6.1	Block Formula of Order 4 with 8 Stages	78
3.7	Order 5	87
3.7.1	Block Formula of Order 5 with 9 Stages	89
3.8	Order 6	95
3.8.1	Block Formula of Order 6 with 12 Stages	96
3.9	Order 7	104
3.9.1	Block Formula of Order 7 with 16 Stages	104
Chapter 4.	Numerical Results	115
4.1	Introduction	115
4.2	The Problems	121
4.3	The Output	121
Chapter 5.	Global Error Estimation	145
5.1	Introduction	145
5.2	Existing Methods	146
5.3	Cost of the Global Error Estimation	150
5.4	Global Error Estimation for Block Methods	151
5.4.1	Order 3 with 5 Stages	151
5.4.2	Order 3 with 6 Stages	153
5.4.3	Order 4 with 7 Stages	155

Table of Contents

5.4.4	Order 4 with 8 Stages	158
5.4.5	Order 5 with 8 Stages	163
Chapter 6.	Numerical Results	167
6.1	Results for Order 4 with 7 Stages	168
6.2	Results for Order 4 with 8 Stages	171
6.3	Results for Order 5 with 8 Stages	174
6.4	Results for Order 5 with 9 Stages	178
Chapter 7.	An Alternative Approach to Stability	183
7.1	Introduction	183
7.2	Order 2 with 3 Stages	185
7.3	Order 2 with 4 Stages	186
7.4	Order 3 with 5 Stages	190
7.5	Order 3 with 6 Stages	193
Appendix A.	Code to Implement Block 6(5) formula (3.61)	196

List of Figures

2.1	Stability of explicit Runge-Kutta methods of orders 1 to 4	44
3.1	Stability region of order 2 with 3 stages	60
3.2	Stability region of order 2 with 4 stages	64
3.3	Stability region of order 3 with 5 stages	71
3.4	Stability region of order 3 with 6 stages	76
3.5	Stability region of order 4 with 8 stages	86
3.6	Stability region of order 5 with 9 stages	92
3.7	Interpolation error for order 5	94
3.8	Stability region of order 6 with 12 stages	101
3.9	Interpolation error for order 6	103
3.10	Stability region of order 7 with 16 stages	109
5.1	Stability region of order 3 with 5 stages	153
5.2	Stability region of order 3 with 6 stages	156
5.3	Stability region of order 4 with 7 stages	159
5.4	Stability region of order 4 with 8 stages	162
5.5	Stability region of order 5 with 8 stages	166
7.1	Stability region of (7.7)	187
7.2	Stability region of (7.10)	189
7.3	Stability region of (7.12)	192
7.4	Stability region of (7.14)	195

List of Tables

2.1	Order equations for a sixth order Runge-Kutta method . . .	37
4.2	Results on DETEST for block 5(4) formula (3.52)	123
4.3	Results on DETEST for the existing block 5(4) formula in [Cash89]	128
4.4	Results on DETEST for block 6(5) formula (3.61)	133
4.5	Results on DETEST for block 7(6) formula (3.72)	138
4.6	Results on DETEST over all groups	143
5.1	Extra order conditions that must be satisfied for each order p to obtain asymptotically correct global error estimates . . .	149

Chapter 1

Introduction

1.1 Initial Value Problems

Initial value problems for ordinary differential equations occur very frequently in a variety of application areas such as control theory, chemical engineering, biology, electrical and civil engineering. Such a problem consists of a first order system of ordinary differential equations together with a set of conditions which are all specified at the same initial point.

The main aim of this thesis is to find efficient numerical methods for the approximate solution of the general first order, non-stiff, initial value problem of the form:

$$y' = f(x, y), \quad y(a) = y_0, \quad x \in [a, b] \quad (1.1)$$

where $y \in \mathbb{R}^m$, $y' \equiv \frac{dy}{dx}$ and $f : \mathbb{R} \times \mathbb{R}^m \longrightarrow \mathbb{R}^m$. Here y is called the *dependent* variable and x the *independent* variable. If f does not depend on x , the problem is said to be *autonomous*, and to be *non-autonomous* otherwise. It is easy to write a non-autonomous problem in autonomous form simply by taking $y_{m+1} \equiv x$ (where y_{m+1} is the $(m+1)^{th}$ component of y) and adding the extra equation $y'_{m+1} = 1$ with initial condition $y_{m+1}(a) = a$ to (1.1). We now get a new $(m+1)$ -dimensional system which is clearly autonomous. If we have a system of order higher than one it can always be reduced to the form (1.1). Consider for example the p^{th} order, m -dimensional system of ordinary differential equations of the form:

$$y^{(p)} = f(x, y^{(0)}, y^{(1)}, \dots, y^{(p-1)}), \quad (1.2)$$

1. Introduction

where $f : \mathbb{R} \times \underbrace{\mathbb{R}^> \times \dots \times \mathbb{R}^>}_{\substack{| \\ \approx \triangleright \sim}} \longrightarrow \mathbb{R}^>$ and $y^{(p)}$ denotes the p^{th} derivative of y with respect to x . We now define a new set of variables $Y_i \in \mathbb{R}^>$, $\triangleright = \# \neq, \dots, \vdash$, as follows:

$$\begin{aligned} Y_1 &= y && (\equiv y^{(0)}) \\ Y_2 &= Y_1' && (\equiv y^{(1)}) \\ Y_3 &= Y_2' && (\equiv y^{(2)}) \\ &\vdots && \vdots \\ Y_p &= Y_{p-1}' && (\equiv y^{(p-1)}). \end{aligned}$$

The last $p - 1$ of these equations together with (1.2) give:

$$\begin{aligned} Y_1' &= Y_2 \\ Y_2' &= Y_3 \\ &\vdots \\ Y_{p-1}' &= Y_p \\ Y_p' &= f(x, Y_1, Y_2, \dots, Y_p), \end{aligned}$$

which is a first order system of dimension pm .

In view of this we confine our attention to (1.1) from now on. The first question we need to investigate concerns the existence of a solution to our problem. A theorem that gives a sufficient condition for the existence and uniqueness of a solution to (1.1), see for example [Lambert91], is:

Theorem 1.1 *Let $f(x, y)$ be defined and continuous for all (x, y) in $D = \{(x, y) : a \leq x \leq b, -\infty < y^s < \infty, 1 \leq s \leq m, a, b \text{ finite}\}$. If f satisfies a Lipschitz condition on D :*

$$\|f(x, y) - f(x, y^*)\| \leq L \|y - y^*\|, \quad (1.3)$$

where L is a Lipschitz constant, then for any $y_0 \in \mathbb{R}^>$, there exists a unique solution, $y(x)$, of the initial value problem (1.1), where $y(x)$ is continuous and differentiable for all $(x, y) \in D$ and $y(x) = (y^1, y^2, \dots, y^m)^T$.

Using the Mean Value Theorem we may choose, if f is differentiable,

$$L = \sup_{(x,y) \in D} \left\| \frac{\partial f(x, y)}{\partial y} \right\|,$$

where $\frac{\partial f(x, y)}{\partial y} \equiv J$ is the Jacobian matrix.

1. Introduction

In what follows we assume that a unique solution of the problem (1.1) exists and that the problem is “*well posed*”. We say that a problem is “well posed” if “small” perturbations in the initial condition produce correspondingly “small” changes in the solution. Although most initial value problems have a unique solution often it can not be found analytically. In view of this it is normally necessary in practical situations to resort to numerical methods and we will discuss these in the next section.

1.2 Numerical Methods

There are two widely used classes of numerical methods for the solution of problems of the form (1.1), namely one-step methods and linear multistep methods. Both of these classes of methods involve discretising the interval $[a, b]$ into points x_i , $i = 0, 1, 2, \dots, N$, such that $x_0 = a$, $x_{i+1} = x_i + h_i$ and $x_N = b$. The aim then is to get numerical approximations y_i to the true solution $y(x_i)$ at each grid point x_i . In what follows we will discuss these two classes of methods and we start off with the one-step case.

Runge-Kutta methods (RKM) are a class of one-step methods which have the general form:

$$y_{n+1} = y_n + h_n \sum_{j=1}^s b_j k_j, \quad (1.4)$$

where

$$k_j = f(x_n + c_j h_n, y_n + h_n \sum_{i=1}^s a_{ji} k_i), \quad j = 1, 2, \dots, s. \quad (1.5)$$

Here s is the number of stages and h_n the steplength of integration. Note that each k_j defines a function evaluation so that an s -stage method requires exactly s function evaluations. A more compact way of writing a Runge-Kutta formula is by means of the so called Butcher’s table:

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

1. Introduction

where c and b are s -dimensional vectors and A is an $s \times s$ matrix defined by:

$$c = (c_1, c_2, \dots, c_s)^T, \quad b = (b_1, b_2, \dots, b_s)^T, \quad A = [a_{ij}].$$

If A is strictly lower triangular we say that the method is *explicit*; if A is lower triangular we say that the method is *semi-implicit* and if A is not lower triangular we say that the method is *implicit*. If we denote by $y(x_{n+1})$ the exact solution of the differential equation at $x_{n+1} = x_n + h_n$ we have by Taylor expansion about x_n :

$$y(x_{n+1}) = y(x_n) + h_n y'(x_n) + \frac{h_n^2}{2} y''(x_n) + \dots .$$

Definition 1.1 *Assuming the localizing assumption: $y_n = y(x_n)$, i.e., the numerical solution is exact at the beginning of each step, we say that the Runge-Kutta method defined by (1.4) is of order p if the Taylor expansions for $y(x_{n+1})$ and y_{n+1} agree up to and including the terms in h_n^p .*

We define the *local truncation error (LTE)* associated with a Runge-Kutta method by the difference between these two Taylor expansions. This is in effect the amount by which the true solution fails to satisfy (1.4) and can be written as:

$$LTE = y(x_{n+1}) - y_{n+1} = h_n^{p+1} \phi_p + O(h_n^{p+2}), \quad (1.6)$$

where ϕ_p is the principal error function of the Runge-Kutta formula. This error function can be written as $\phi_p = \sum_{j=1}^{N_p} T_j D_j$ where T_j are polynomials in the coefficients of the formula, D_j are elementary differentials [**Butcher63a**] depending on the problem and its solution and N_p is an integer depending on p . The values of N_p are given in the table below.

p	1	2	3	4	5	6	7	8	9
N_p	1	1	2	4	9	20	48	115	286

These polynomials give rise to a set of nonlinear algebraic equations defining the so called *order conditions*. The order conditions are complicated to derive from the Taylor expansions, but fortunately an analysis developed by J.C. Butcher [**Butcher87**] based on the use of rooted trees makes them quite

1. Introduction

easy to write down. The order conditions define simultaneous nonlinear algebraic equations in the coefficients c_i , b_i and a_{ij} . For high order formulae, these order conditions become extremely difficult to solve. One can understand why the construction of high order Runge-Kutta formulae is not an easy task if we consider the next table.

order p	1	2	3	4	5	6	7	8	9	10
<i>no. of order conditions</i>	1	2	4	8	17	37	85	200	486	1205

In order to solve these equations it is often necessary to impose certain simplifying assumptions and we return to these later in this thesis.

We now consider *linear multistep methods* (LMM). A general linear k -step method can be defined by the equation:

$$\sum_{j=0}^k \alpha_j y_{n+j} = h_n \sum_{j=0}^k \beta_j f(x_{n+j}, y_{n+j}) \quad (1.7)$$

where α_j and β_j are constants that satisfy:

$$\alpha_k = 1, \quad |\alpha_0| + |\beta_0| \neq 0.$$

The condition on α_k is simply a normalising condition while the condition on α_0 and β_0 ensures that (1.7) can not be rewritten as a linear $(k - 1)$ -step method.

A linear multistep method is said to be explicit if $\beta_k = 0$ and implicit otherwise.

Associated with a linear multistep method, it is usual to define the *first* and *second characteristic polynomials* as:

$$\rho(\xi) = \sum_{j=0}^k \alpha_j \xi^j, \quad \sigma(\xi) = \sum_{j=0}^k \beta_j \xi^j, \quad (1.8)$$

where $\xi \in \mathbb{C}$ is a dummy variable.

It is also convenient to define the *linear difference operator* L by:

$$L[z(x); h_n] = \sum_{j=0}^k [\alpha_j z(x + jh_n) - h_n \beta_j z'(x + jh_n)], \quad (1.9)$$

1. Introduction

where $z(x) \in C^1[a, b]$ is an arbitrary function.

If we choose $z(x)$ to be differentiable as often as we need, expand $z(x + jh_n)$ and $z'(x + jh_n)$ in a Taylor series about x , and collect terms, we obtain:

$$L[z(x); h_n] = C_0 z(x) + C_1 h_n z^{(1)}(x) + \cdots + C_m h_n^m z^{(m)}(x) + \cdots \quad (1.10)$$

where the C_m are recursively defined by:

$$C_0 = \sum_{j=0}^k \alpha_j = \rho(1)$$

$$C_1 = \sum_{j=0}^k (j\alpha_j - \beta_j) = \rho'(1) - \sigma(1)$$

$$C_m = \sum_{j=0}^k \left(\frac{j^m}{m!} \alpha_j - \frac{j^{m-1}}{(m-1)!} \beta_j \right), \quad m = 2, 3, \dots$$

Definition 1.2 *The method (1.7) is said to be of order p if in (1.10) the constants $C_0 = C_1 = C_2 = \cdots = C_p = 0$ and $C_{p+1} \neq 0$.*

The local truncation error of a linear multistep method at the point x_{n+k} is defined by $L[y(x_n); h_n]$, where $y(x)$ is the exact solution of the initial value problem (1.1). If we make the localizing assumption that $y_{n+j} = y(x_{n+j})$, $j = 0, 1, \dots, k-1$, i.e., all back values are exact, and also assuming that $y(x) \in C^{p+1}[a, b]$, where p is the order of the method, it can be shown that:

$$LTE = C_{p+1} h_n^{p+1} y^{(p+1)}(x_n) + O(h_n^{p+2}). \quad (1.11)$$

The term $C_{p+1} h_n^{p+1} y^{(p+1)}(x_n)$ is referred to as the principal local truncation error and the constant C_{p+1} as the error constant or principal error constant. As in the case of Runge-Kutta methods, the local truncation error provides a measure of the difference between the true solution and the numerical solution over a single step on the assumption that previously computed solutions are exact. Note however that the form of the local truncation error for linear multistep methods is relatively simple containing as it does just a single term. This is in contrast to high order Runge-Kutta methods which generally have very complicated expressions for their local truncation error.

1. Introduction

A widely used class of linear multistep methods is the class of *Adams methods*. These can be written in the form:

$$y_{n+k} - y_{n+k-1} = h_n \sum_{j=0}^k \beta_j f(x_{n+j}, y_{n+j}). \quad (1.12)$$

The explicit formulae are known as *Adams-Bashforth* methods while the implicit ones are known as *Adams-Moulton* methods. A popular technique used to solve an initial value problem using these two types of formulae is the *predictor-corrector* approach. We can explain this by considering the Adams-Bashforth formula:

$$y_{n+k} = y_{n+k-1} + h_n \sum_{j=0}^{k-1} \bar{\beta}_j f_{n+j} \quad (1.13)$$

and the Adams-Moulton formula:

$$y_{n+k} = y_{n+k-1} + h_n \sum_{j=0}^k \beta_j f_{n+j}. \quad (1.14)$$

Assuming that the data (x_i, y_i) is known for $0 \leq i \leq k-1$, a predictor-corrector approach is as follows.

- (P) Use the Adams-Bashforth formula to predict a solution $y_{n+k}^{(0)}$.
- (E) Evaluate $f_{n+k}^{(0)} \equiv f(x_{n+k}, y_{n+k}^{(0)})$.
- (C) Use the iteration scheme

$$y_{n+k}^{(i)} = y_{n+k-1} + h_n \beta_k f_{n+k}^{(i-1)} + h_n \sum_{j=0}^{k-1} \beta_j f_{n+j}, \quad i = 1, 2, \dots, M$$

- (E) $f_{n+k}^{(i)} \equiv f(x_{n+k}, y_{n+k}^{(i)})$

to find a sequence of corrected solutions.

Often this approach is implemented with $M = 1$ and in this case the formulae are said to be used in PECE mode.

In the next section we examine the concepts of stability and convergence of numerical methods.

1.3 Stability and Convergence

In analysing the stability of numerical methods it is normal to study the *scalar test equation*:

$$y' = \lambda y, \quad y(0) = 1, \quad \lambda \in \mathbb{C}, \quad \Re(\lambda) < \infty, \quad (1.15)$$

with constant stepsize.

Although this is a very simple model problem it does help us to derive methods which work well in practice for many problems of interest.

If we apply any one-step numerical method with a steplength h to (1.15) we obtain a difference equation of the form:

$$y_{n+1} = R(z)y_n, \quad (1.16)$$

where $z = \lambda h$. Here $R(z)$ is called the *stability function* of the method.

Definition 1.3 *A numerical method is said to be absolutely stable for those values of z that satisfy:*

$$|R(z)| < 1. \quad (1.17)$$

Definition 1.4 *The region of absolute stability, \mathfrak{R}_A , of a numerical method is that part of the complex z -plane for which (1.17) is satisfied for all $z \in \mathfrak{R}_A$. These regions are symmetric about the real axis.*

The function $R(z)$ obtained by applying a Runge-Kutta method to (1.15) can be expressed in either of the following ways [**Lambert91**]:

$$R(z) = 1 + zb^T(I - zA)^{-1}e \quad (1.18)$$

or

$$R(z) = \frac{\det(I - zA + zeb^T)}{\det(I - zA)}, \quad (1.19)$$

where $e = (1, 1, \dots, 1)^T$. When the method is explicit, in which case A is a strictly lower triangular matrix, the matrix $I - zA$ is also lower triangular, with all the diagonal elements equal to one. It follows immediately that

1. Introduction

$\det(I - zA) = 1$ in this case and this means that, for an explicit method, $R(z)$ is a polynomial in z .

When we apply a linear multistep method to the scalar test equation (1.15) we get a k^{th} order constant coefficients difference equation which are of the form Ar^m where r satisfies solutions of:

$$R(r, z) \equiv \rho(r) - z\sigma(r) = 0. \quad (1.20)$$

We now consider how the property of stability is related to that of convergence. Most numerical methods for solving the initial value problem (1.1) can be written in the form:

$$\sum_{j=0}^k \alpha_j y_{n+j} = h_n \phi_f(y_{n+k}, y_{n+k-1}, \dots, y_n, x_n; h_n). \quad (1.21)$$

Theorem 1.1 gives us conditions for the existence of a solution of the initial value problem (1.1). Now, we are interested in knowing if the numerical method used to obtain such a solution is convergent. Assuming for the time being that $h_n \equiv h$ is a constant we have:

Definition 1.5 *The numerical method (1.21) is said to be convergent if, for all initial value problems satisfying the hypotheses of Theorem 1.1, the following condition is satisfied at an arbitrary fixed point $x_n = a + nh$:*

$$\lim_{h \rightarrow 0} \|y(x_n) - y_n\| = 0.$$

Given a particular numerical method it is very difficult to check directly whether it satisfies this condition. What we would like is a necessary and sufficient condition for convergence which is easy to check in practice. To obtain this we first define the concept of consistency.

Definition 1.6 *The numerical method (1.21) is said to be consistent of order p if, for all initial value problems satisfying the hypotheses of Theorem 1.1, the following condition is satisfied:*

$$\frac{1}{h_n} \sum_{i=0}^k \alpha_i y(x_{n+i}) - \phi_f(y(x_{n+k}), y(x_{n+k-1}), \dots, y(x_n), x_n; h_n) = O(h_n^p).$$

1. Introduction

In order to state the fundamental theorem on convergence we need to give some further definitions.

Definition 1.7 *The numerical method (1.21) is said to satisfy the root condition if all of the roots of the first characteristic polynomial have modulus less than or equal to one, with those of modulus one being simple.*

The following theorem gives a necessary and sufficient condition for a numerical method to be *zero-stable* (defined for example in [Lambert91]).

Theorem 1.2 *The method (1.21) is zero-stable iff it satisfies the root condition.*

We can now state the fundamental result (see for example [Lambert91]):

Theorem 1.3 *The necessary and sufficient conditions for the numerical method (1.21) to be convergent are that it is zero-stable and consistent.*

Since Runge-Kutta methods are one-step in nature they automatically satisfy the root condition. This means that providing the coefficients b_i , a_{ij} , c_i are chosen so that the method is consistent, a Runge-Kutta method is also convergent. Thus in the case of Runge-Kutta methods it is trivial to get convergence. In the case of linear multistep methods it is not so simple and this can be seen by the following theorem:

Theorem 1.4 *(first Dahlquist barrier)*

No zero-stable linear k -step method can have order exceeding $k+1$ when k is odd and $k+2$ when k is even.

We see that the condition that a linear multistep method should be convergent imposes a severe restriction on the attainable order.

1.4 Stiffness

Many of the initial value problems that occur in practice fall into two distinct (but not very well defined) classes known as stiff and non-stiff problems.

The concept of stiffness is a property of the differential system itself. However in practice, whether or not a problem appears to be stiff depends on the method being used to solve it, since with an appropriate choice of methods one hardly notices that there are any difficulties in finding the solution to a stiff system. Stiffness cannot be defined in a very precise mathematical way. To get the “feeling” of what it is let us consider the following problem:

$$y' = Ay + \varphi(x), \quad y(a) = y_0, \quad (1.22)$$

where $y, \varphi \in \mathbb{R}^n$ and A is an $m \times m$ matrix of constant coefficients with distinct eigenvalues $\lambda_i \in \mathbb{C}$, $\lambda_i \neq \lambda_j, \dots$. The restriction to distinct eigenvalues is not crucial but makes the analysis more straightforward. The analytic solution of (1.22) has the form:

$$y(x) = \sum_{i=1}^m c_i e^{\lambda_i x} v_i + \psi(x), \quad (1.23)$$

where the c_i are constants and $\psi(x)$ is a particular integral. Let us assume that $\text{Re}(\lambda_i) < 0$, $i = 1, 2, \dots, m$, and that

$$\frac{\max |\text{Re}(\lambda_i)|}{\min |\text{Re}(\lambda_i)|} \gg 1.$$

Here $\frac{\max |\text{Re}(\lambda_i)|}{\min |\text{Re}(\lambda_i)|}$ defines the *stiffness ratio*. If we compare the two components of the solution: $e^{(-\max |\text{Re}(\lambda_i)|)x}$ and $e^{(-\min |\text{Re}(\lambda_i)|)x}$ the first decays very fast while the second decays slowly in comparison.

In practice the aim is often to reach the *steady state solution*, $\psi(x)$. Using a code based on a method with a finite region of absolute stability the stepsize will be restricted since the stability region is “small” and some of the λ_i are “very” negative. Integrating with this small stepsize will be very time consuming, and we will have to integrate over a long distance until we get to the

1. Introduction

steady state solution. Let us consider an example. Consider the second order differential equation:

$$y'' - (\lambda - 1)y' - \lambda y = 0. \quad (1.24)$$

This can be transformed into a system of two first order differential equations to give:

$$y' = z$$

$$z' = (\lambda - 1)z + \lambda y \quad \text{or}$$

$$\begin{pmatrix} y' \\ z' \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ \lambda & \lambda - 1 \end{pmatrix} \begin{pmatrix} y \\ z \end{pmatrix}. \quad (1.25)$$

The analytic solution is given by:

$$y = Ae^{\lambda x} + Be^{-x}, \quad A, B \in \mathbb{R}.$$

Suppose we use Euler's method to solve (1.25). The interval of absolute stability of this method is $(-2, 0)$. The restrictions on the stepsize of integration needed to ensure stability are:

$$\begin{aligned} 0 &> -h > -2 \\ 0 &> h\lambda > -2. \end{aligned} \quad (1.26)$$

Suppose for example that $\lambda = -1$. In this case the restriction (1.26) is not at all severe. However if for example $\lambda = -1000$ then the second equation of (1.26) would need a very small stepsize for it to be satisfied. If, however, we look at the solution:

$$y = Ae^{-1000x} + Be^{-x}$$

we can see that the first component goes to zero much quicker than the second one. In contrast, if we consider the equation $y' = -y$, which has approximately the same solution after a small time, we just need to have $h < 2$ for stability.

The concept of stiffness is defined in [Lambert91] in the following way:

Definition 1.8 *If a numerical method with a finite region of absolute stability, applied to a system with any initial conditions, is forced to use in a certain interval of integration a steplength which is excessively small in relation to the smoothness of the exact solution in that interval, then the system is said to be stiff in that interval.*

1. Introduction

As we have seen, the ability of a numerical method to successfully integrate a stiff system efficiently depends among other things on the size of its stability region. Ideally we would like a numerical method to have an infinite region of absolute stability and we give the following definitions which impose this requirement.

Definition 1.9 *A method is said to be A-stable, see for example [Lambert91], if its region of absolute stability, \mathfrak{R}_A , contains the whole complex left half plane $\text{Re}(z) \leq 0$ i.e.:*

$$\mathfrak{R}_A \supseteq \{z \in \mathbb{C} : \mathbb{R}(F) \leq \kappa, F = \lambda \approx\}.$$

Definition 1.10 *A one-step method is said to be L-stable if it is A-stable and in addition when applied to the scalar test equation the stability function $R(z)$ satisfies:*

$$\lim_{\text{Re}(z) \rightarrow -\infty} |R(z)| = 0. \quad (1.27)$$

The most widely used methods for the integration of stiff systems are the *backward differentiation formulae* (BDF) which are from the class of linear multistep methods. Although these have been successfully applied to a wide class of stiff problems they are restricted by the second Dahlquist barrier which says that an A-stable linear multistep method cannot have order exceeding 2. As a result of this, many classes of L-stable implicit Runge-Kutta formulae have been proposed but these have been found to be relatively expensive. In an attempt to get over these problems there has recently been some interest in the use of block methods. In particular block implicit methods have been considered for stiff problems in [Butcher90],[Cash83b]. Results so far indicate that this approach is likely to be particularly efficient for stiff problems combining as it does high stage order with excellent stability. In this thesis we extend these ideas to block explicit methods suitable for non-stiff problems. Some early work on this approach was given in [Cash83a] and later on in this thesis we will develop this approach to high order, computationally efficient formulae. Before considering these methods, which can be regarded as a combination of linear multistep and Runge-Kutta methods, in detail we first examine the relative merits of these two classes of formulae.

1.5 Comparison Between RKM and LMM

RKM	LMM
self-starting	not self-starting except at order ≤ 2
easy to change stepsize	expensive to change stepsize
for problems with frequent discontinuities it is easy to start at any order	need to restart at order 1 and then build up the order to a reasonable level
expensive to estimate the LTE due to the number of function evaluations required; difficult to construct a variable order code	easy to estimate the LTE; easy to construct a variable order code
low overhead	high overhead
no natural interpolation polynomial available; expensive to obtain solutions at “off-step” points; can restrict the stepsize to hit output points or derive an interpolant usually at the cost of extra function evaluations	natural underlying interpolation polynomial available; easy to obtain solutions at “off-step” points; stepsize determined just by accuracy requirements
need more than 2 function evaluations per step for high order: difficult to obtain efficient high order methods	only 2 function evaluations per step are needed at all orders if implemented in a PECE mode

Here “overhead” denotes all computation effort apart from function evaluations.

Many attempts have been made to derive a class of methods which is in some sense between these two classes. Examples of these are: general linear methods [**Butcher87**], hybrid methods [**Lambert73**] and block explicit Runge-Kutta methods [**Cash85**],[**Cash86**],[**Cash89**]. One of the major purposes of our investigation is to develop a new class of methods which has none of the drawbacks associated with each method but has the advantages of both.

1. Introduction

Ideally we would like a method which is self-starting, has free interpolation, in the sense that no extra function evaluations are necessary, has good stability properties and for which it is easy to compute local error estimates and to change order.

Chapter 2

Runge-Kutta Methods - A Survey

2.1 Introduction

In this section we consider in more detail the use of explicit Runge-Kutta methods for the numerical solution of (1.1). The simplest numerical method is Euler's method which for constant h is:

$$y_{n+1} = y_n + hf(x_n, y_n). \quad (2.1)$$

This method has order 1 which is not sufficiently high for it to be efficient for the integration of general systems of the form (1.1) except possibly when very low accuracy is required. As we saw in the previous chapter the two main ways of obtaining higher order are either to consider a multistep approach or to consider a one-step Runge-Kutta approach. In this section we concentrate on the Runge-Kutta approach.

Runge (1895) first had the idea of getting methods of order greater than 1 by keeping the one-step formulation but at the cost of losing linearity. His initial ideas were further developed by Heun (1900) and Kutta (1901).

An s -stage Runge-Kutta method can be written in either of the following two equivalent ways:

$$y_{n+1} = y_n + h \sum_{j=1}^s b_j k_j, \quad (2.2)$$

where

$$k_j = f(x_n + c_j h, y_n + h \sum_{i=1}^s a_{ji} k_i), \quad j = 1, 2, \dots, s, \quad (2.3)$$

2. Runge-Kutta Methods - A Survey

or

$$y_{n+1} = y_n + h \sum_{j=1}^s b_j f(x_n + c_j h, Y_j), \quad (2.4)$$

with

$$k_j = f(x_n + c_j h, Y_j), \quad j = 1, 2, \dots, s, \quad (2.5)$$

where

$$Y_j = y_n + h \sum_{i=1}^s a_{ji} f(x_n + c_i h, Y_i), \quad j = 1, 2, \dots, s. \quad (2.6)$$

As explained in the previous chapter, in order to derive a Runge-Kutta method of order p we need to satisfy certain nonlinear order conditions. In the next section we examine the link between the integers p and s .

2.2 Order versus Stages

An s -stage explicit Runge-Kutta method has $s(s+1)/2$ free parameters. If it is to be of order p it must satisfy $\sum_{i=1}^p N_i$ order conditions. Butcher [Butcher87] has given a table that relates the number of parameters, the number of order conditions and the number of stages for each order and we reproduce it below.

s	1	2	3	4	5	6	7	8	9
$s(s+1)/2$	1	3	6	10	15	21	28	36	45
$\sum_{p=1}^s N_p$	1	2	4	8	17	37	85	200	485

Of particular interest is the number of stages s required to get a particular order p formula. Butcher [Butcher87] has given the following theorems relating to explicit Runge-Kutta methods.

Theorem 2.1 *An explicit Runge-Kutta method with s -stages has order not exceeding s .*

Theorem 2.2 *For an explicit Runge-Kutta method with $s \leq 4$ stages, the greatest attainable order is s .*

2. Runge-Kutta Methods - A Survey

Theorem 2.3 *There is no p -stage, p^{th} order explicit Runge-Kutta method with $p \geq 5$.*

Theorem 2.4 *For a given $p \geq 6$ there exists an explicit Runge-Kutta method with s -stages where $s = \frac{p^2 - 7p + 20}{2}$. Furthermore, if $p \geq 10$, a method of the required order exists with $s = \frac{p^2 - 7p + 10}{2}$ stages.*

Theorem 2.5 *For a given p there exists an explicit Runge-Kutta method of order p with s -stages where*

$$s = \begin{cases} \frac{3p^2 - 10p + 24}{8} & p \text{ even,} \\ \frac{3p^2 - 4p + 9}{8} & p \text{ odd.} \end{cases}$$

The known results for explicit Runge-Kutta methods can be summarised in the following table:

Order	Minimum number of stages	
	Lower bound	Upper bound
1		1
2		2
3		3
4		4
5		6
6		7
7		9
8		11
9	12	17
10	13	17

2.3 Some Examples of Low Order Formulae

In this section we will give some well known explicit Runge-Kutta methods of order 1 to 4.

Euler's method is the only order 1 explicit Runge-Kutta method with 1 stage. Using Butcher's matrix notation this formula can be written as:

$$\begin{array}{c|c} 0 & 0 \\ \hline & 1 \end{array} \quad (2.7)$$

As examples of an order 2 method with 2 stages, we have the Modified Euler method:

$$\begin{array}{c|cc} 0 & 0 & \\ \frac{1}{2} & \frac{1}{2} & 0 \\ \hline & 0 & 1 \end{array} \quad (2.8)$$

and the Improved Euler method:

$$\begin{array}{c|cc} 0 & 0 & \\ 1 & 1 & 0 \\ \hline & \frac{1}{2} & \frac{1}{2} \end{array} \quad (2.9)$$

For order 3 with 3 stages there are two well known formulae: Heun's third

2. Runge-Kutta Methods - A Survey

order formula given by:

$$\begin{array}{c|ccc}
 0 & 0 & & \\
 \frac{1}{3} & \frac{1}{3} & 0 & \\
 \frac{2}{3} & 0 & \frac{2}{3} & 0 \\
 \hline
 & \frac{1}{4} & 0 & \frac{3}{4}
 \end{array} \tag{2.10}$$

and Kutta's third order formula:

$$\begin{array}{c|ccc}
 0 & 0 & & \\
 \frac{1}{2} & \frac{1}{2} & 0 & \\
 1 & -1 & 2 & 0 \\
 \hline
 & \frac{1}{6} & \frac{2}{3} & \frac{1}{6}
 \end{array} \tag{2.11}$$

For order 4 with 4 stages by far the most well known method is the classical Runge-Kutta method:

$$\begin{array}{c|cccc}
 0 & 0 & & & \\
 \frac{1}{2} & \frac{1}{2} & 0 & & \\
 \frac{1}{2} & 0 & \frac{1}{2} & 0 & \\
 1 & 0 & 0 & 1 & 0 \\
 \hline
 & \frac{1}{6} & \frac{1}{3} & \frac{1}{3} & \frac{1}{6}
 \end{array} \tag{2.12}$$

The method was derived in 1901 by Kutta in the pre-computer era which accounts for the simplicity of the coefficients and the large number of zeros.

Many methods with order greater than 4 exist and can be found, for example in [Butcher87] together with the references given therein.

2.4 Butcher's Analysis and Relation to Rooted Trees

As we have seen in previous sections, the order conditions for high order explicit Runge-Kutta formulae are not easy to derive from first principles using Taylor expansions. However by using the theory of rooted trees, Butcher [Butcher87] has considerably simplified this analysis. Based on graph theory Butcher has developed an approach which gives a correspondence between trees and order conditions. In what follows we give a brief introduction to this analysis.

The trees used in Butcher's theory are built up from a unique root and a set of nodes joined by lines which are referred to as branches. It is very easy to build up the trees from the lower order ones just by adding some branches on to the "free" nodes. Of course we have to make sure that we are not considering equivalent trees. For example the trees with just one or two nodes are:



From the tree with two nodes we have two alternatives, either add another node to the root to give the tree:



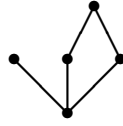
or add another branch to the top node to give:



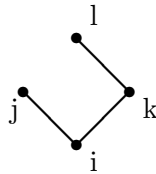
Thus new trees can be created simply by adding new branches and nodes to the existing nodes. There is a constraint that doesn't permit branches to grow

2. Runge-Kutta Methods - A Survey

together again, an example of this being:



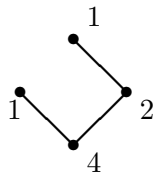
Such a tree would not be allowed. The number of nodes corresponds to the order. To obtain the order conditions from the trees we will define the polynomials in a_{ij} , b_i , c_i by $\Phi(t)$ and the constants by $\gamma(t)$ for each tree t . The necessary and sufficient conditions for a method to be of order p are that $\Phi(t) = \frac{1}{\gamma(t)}$ for all trees t with no more than p nodes (cf. [Butcher87]). The polynomial $\Phi(t)$ is obtained by giving a label i, j, k, \dots to each of the nodes of the tree, starting from the bottom and going up, i.e., i is the label of the root. To each branch we assign the label a_{ij} where i and j are the labels of the two nodes that are connected by this branch. Then we form the product $b_i \prod a_{jk}$, and sum over i, j, k, \dots from 1 to s . For example if we consider the tree:



the corresponding polynomial is $\sum_{i,j,k,l=1}^s b_i a_{ij} a_{ik} a_{kl}$ which can be simplified to

$$\sum_{i,j=1}^s b_i c_i a_{ij} c_j \text{ by making use of the row-sum conditions } \sum_{j=1}^s a_{ij} = c_i, \quad 1 \leq i \leq s.$$

If we want to know the value of $\gamma(t)$, which is called the *density of the tree* t , for a particular tree t , we label the terminal nodes with 1 and all the others with a value $i + 1$, where i is the number of branches going outward from that branch. Then the value of $\gamma(t)$ is the product of the numbers attached to each node. If we consider the same tree, for example, then following these rules we have:



so that $\gamma(t) = 8$. We can define the *height of a tree* t , $H(t)$, as the length of the longest possible path in the tree plus 1, and *width of a tree* t , $w(t)$, as

the number of terminal nodes. In the case of the previous tree $H(t) = 3$ and $w(t) = 2$. Table 2.1 gives the order conditions, the height, the width and the corresponding trees for orders 1 to 6. All subscripts run from 1 to s .

2.5 Simplifying Assumptions

Since it is difficult to solve the nonlinear algebraic equations defining a high order Runge-Kutta method it is often necessary to use simplifying assumptions to reduce the complexity of the equations. The effect that these simplifying assumptions have is to reduce the number of nonlinear algebraic order conditions by forcing some of the equations to become equivalent. There are two kinds of simplifying assumptions that are commonly used namely: *row-simplifying assumptions* and *column-simplifying assumptions*. The row-simplifying assumptions are for example:

$$\begin{aligned} \sum_{j=1}^s a_{ij}c_j &= \frac{1}{2} c_i^2, \quad i \neq 2, \\ b_2 &= 0. \end{aligned} \tag{2.13}$$

If we consider for example an order 6 method, when the row-simplifying assumptions are satisfied together with the order conditions of the form:

$$\sum_{i=1}^s b_i c_i^j = \frac{1}{j+1}, \quad j = 0, 1, \dots, 5 \tag{2.14}$$

the order conditions in the first column of the table below will “disappear” since they become equivalent to equations in the second column.




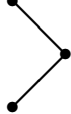
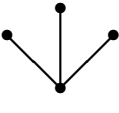
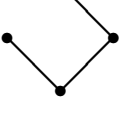
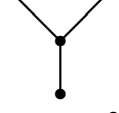
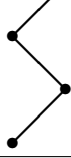
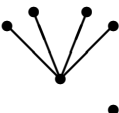
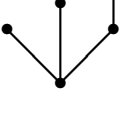
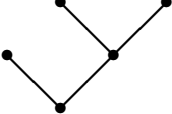
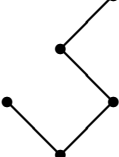
4	$\sum b_i a_{ij} c_j = \frac{1}{6}$	3	$\sum b_i c_i^2 = \frac{1}{3}$
6	$\sum b_i c_i a_{ij} c_j = \frac{1}{8}$	5	$\sum b_i c_i^3 = \frac{1}{4}$
10	$\sum b_i c_i^2 a_{ij} c_j = \frac{1}{10}$	9	$\sum b_i c_i^4 = \frac{1}{5}$
13	$\sum b_i a_{ij} c_j a_{ik} c_k = \frac{1}{20}$	10	$\sum b_i c_i^2 a_{ij} c_j = \frac{1}{10}$
19	$\sum b_i c_i^3 a_{ij} c_j = \frac{1}{12}$	18	$\sum b_i c_i^5 = \frac{1}{6}$
20	$\sum b_i c_i a_{ij} c_j a_{ik} c_k = \frac{1}{24}$	19	$\sum b_i c_i^3 a_{ij} c_j = \frac{1}{12}$

The column-simplifying assumptions are:

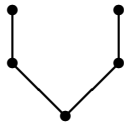
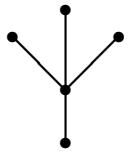
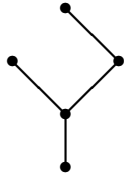
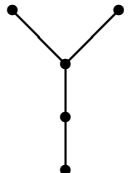
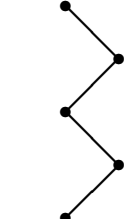
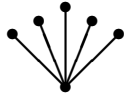
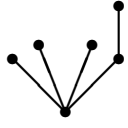
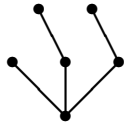
$$\sum_{i=1}^s b_i a_{ij} = b_j(1 - c_j), \quad j = 1, 2, \dots, s. \tag{2.15}$$

In the table below we show which order conditions will “disappear”.

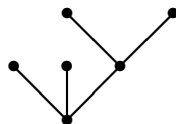
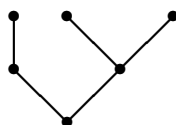
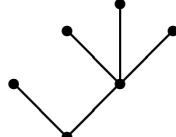
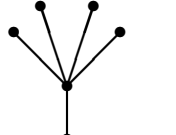
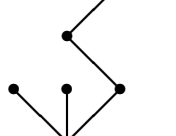
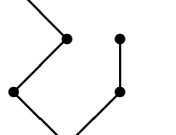
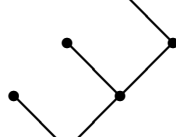
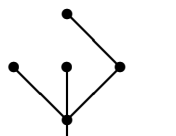
2. Runge-Kutta Methods - A Survey

		t	Order Condition	w(t)	H(t)
1	$\tau_1^{(1)}$		$\sum b_i = 1$	0	1
2	$\tau_1^{(2)}$		$\sum b_i c_i = \frac{1}{2}$	1	2
3	$\tau_1^{(3)}$		$\sum b_i c_i^2 = \frac{1}{3}$	2	2
4	$\tau_2^{(3)}$		$\sum b_i a_{ij} c_j = \frac{1}{6}$	1	3
5	$\tau_1^{(4)}$		$\sum b_i c_i^3 = \frac{1}{4}$	3	2
6	$\tau_2^{(4)}$		$\sum b_i c_i a_{ij} c_j = \frac{1}{8}$	2	3
7	$\tau_3^{(4)}$		$\sum b_i a_{ij} c_j^2 = \frac{1}{12}$	2	3
8	$\tau_4^{(4)}$		$\sum b_i a_{ij} a_{jk} c_k = \frac{1}{24}$	1	4
9	$\tau_1^{(5)}$		$\sum b_i c_i^4 = \frac{1}{5}$	4	2
10	$\tau_2^{(5)}$		$\sum b_i c_i^2 a_{ij} c_j = \frac{1}{10}$	3	3
11	$\tau_3^{(5)}$		$\sum b_i c_i a_{ij} c_j^2 = \frac{1}{15}$	3	3
12	$\tau_4^{(5)}$		$\sum b_i c_i a_{ij} a_{jk} c_k = \frac{1}{30}$	2	4

2. Runge-Kutta Methods - A Survey

		t	Order Condition	w(t)	H(t)
13	$\tau_5^{(5)}$		$\sum b_i a_{ij} c_j a_{ik} c_k = \frac{1}{20}$	2	3
14	$\tau_6^{(5)}$		$\sum b_i a_{ij} c_j^3 = \frac{1}{20}$	3	3
15	$\tau_7^{(5)}$		$\sum b_i a_{ij} c_j a_{jk} c_k = \frac{1}{40}$	2	4
16	$\tau_8^{(5)}$		$\sum b_i a_{ij} a_{jk} c_k^2 = \frac{1}{60}$	2	4
17	$\tau_9^{(5)}$		$\sum b_i a_{ij} a_{jk} a_{kl} c_l = \frac{1}{120}$	1	5
18	$\tau_1^{(6)}$		$\sum b_i c_i^5 = \frac{1}{6}$	5	2
19	$\tau_2^{(6)}$		$\sum b_i c_i^3 a_{ij} c_j = \frac{1}{12}$	4	3
20	$\tau_3^{(6)}$		$\sum b_i c_i a_{ij} c_j a_{ik} c_k = \frac{1}{24}$	3	3

2. Runge-Kutta Methods - A Survey

		t	Order Condition	w(t)	H(t)
21	$\tau_4^{(6)}$		$\sum b_i c_i^2 a_{ij} c_j^2 = \frac{1}{18}$	4	3
22	$\tau_5^{(6)}$		$\sum b_i a_{ij} c_j^2 a_{ik} c_k = \frac{1}{36}$	3	3
23	$\tau_6^{(6)}$		$\sum b_i c_i a_{ij} c_j^3 = \frac{1}{24}$	4	3
24	$\tau_7^{(6)}$		$\sum b_i a_{ij} c_j^4 = \frac{1}{30}$	4	3
25	$\tau_8^{(6)}$		$\sum b_i c_i^2 a_{ij} a_{jk} c_k = \frac{1}{36}$	3	4
26	$\tau_9^{(6)}$		$\sum b_i a_{ij} a_{ik} c_k a_{jl} c_l = \frac{1}{72}$	2	4
27	$\tau_{10}^{(6)}$		$\sum b_i c_i a_{ij} c_j a_{jk} c_k = \frac{1}{48}$	3	4
28	$\tau_{11}^{(6)}$		$\sum b_i a_{ij} c_j^2 a_{jk} c_k = \frac{1}{60}$	3	4

2. Runge-Kutta Methods - A Survey

		t	Order Condition	w(t)	H(t)
29	$\tau_{12}^{(6)}$		$\sum b_i a_{ij} a_{jk} c_k a_{jl} c_l = \frac{1}{120}$	2	4
30	$\tau_{13}^{(6)}$		$\sum b_i c_i a_{ij} a_{jk} c_k^2 = \frac{1}{72}$	3	4
31	$\tau_{14}^{(6)}$		$\sum b_i a_{ij} c_j a_{jk} c_k^2 = \frac{1}{90}$	3	4
32	$\tau_{15}^{(6)}$		$\sum b_i a_{ij} a_{jk} c_k^3 = \frac{1}{120}$	3	4
33	$\tau_{16}^{(6)}$		$\sum b_i c_i a_{ij} a_{jk} a_{kl} c_l = \frac{1}{144}$	2	5
34	$\tau_{17}^{(6)}$		$\sum b_i a_{ij} c_j a_{jk} a_{kl} c_l = \frac{1}{180}$	2	5

2. Runge-Kutta Methods - A Survey

		t	Order Condition	w(t)	H(t)
35	$\tau_{18}^{(6)}$		$\sum b_i a_{ij} a_{jk} c_k a_{kl} c_l = \frac{1}{240}$	2	5
36	$\tau_{19}^{(6)}$		$\sum b_i a_{ij} a_{jk} a_{kl} c_l^2 = \frac{1}{360}$	2	5
37	$\tau_{20}^{(6)}$		$\sum b_i a_{ij} a_{jk} a_{kl} a_{lm} c_m = \frac{1}{720}$	1	6

Table 2.1: Order equations for a sixth order Runge-Kutta method

2. Runge-Kutta Methods - A Survey

4	$\sum b_i a_{ij} c_j = \frac{1}{6}$	2-3	$\sum b_i c_i = \frac{1}{2} - \sum b_i c_i^2 = \frac{1}{3}$
7	$\sum b_i a_{ij} c_j^2 = \frac{1}{12}$	3-5	$\sum b_i c_i^2 = \frac{1}{3} - \sum b_i c_i^3 = \frac{1}{4}$
14	$\sum b_i a_{ij} c_j^3 = \frac{1}{20}$	5-9	$\sum b_i c_i^3 = \frac{1}{4} - \sum b_i c_i^4 = \frac{1}{5}$
24	$\sum b_i a_{ij} c_j^4 = \frac{1}{30}$	9-18	$\sum b_i c_i^4 = \frac{1}{5} - \sum b_i c_i^5 = \frac{1}{6}$

By making the simplifying assumptions (2.13), (2.15) we would be able to derive higher order methods in a very straightforward way. We will return to these simplifying assumptions in the next chapter where we will extend this approach to derive high order block explicit Runge-Kutta formulae.

2.6 Local Error Estimation

The local truncation error of high order Runge-Kutta formulae is difficult to estimate directly because of its complicated form. As we showed earlier, the LTE of an order p Runge-Kutta method is given by:

$$LTE = y(x_{n+1}) - y_{n+1} = h_n^{p+1} \phi_p + O(h_n^{p+2}), \quad (2.16)$$

where ϕ_p , the error function, is given by: $\phi_p = \sum_{j=1}^{N_p} T_j D_j$. The number of terms in this sum increases rapidly with p . The problem we are faced with is to establish some practical way in which the LTE can be estimated. The first approach developed was *Richardson extrapolation* otherwise known as the *deferred approach to the limit*, see for example [Lambert91]. This approach consists of parallel integration with different stepsizes. Starting from the point (x_n, y_n) let y_{n+1} and y_{n+1}^* be two numerical approximations to the true solution, $y(x_{n+1})$, obtained using stepsizes h_n and $h_n/2$, respectively with the same p^{th} order formula. The local truncation error in y_{n+1} is given by (2.16). Considering the second approximation we get:

$$LTE = y(x_{n+1}) - y_{n+1}^* = 2 \left(\frac{h_n}{2} \right)^{p+1} \phi_p + O(h_n^{p+2}). \quad (2.17)$$

We can obtain an estimate for the LTE by subtracting equation (2.16) from (2.17):

$$\phi_p h_n^{p+1} \cong \frac{2^p (y_{n+1} - y_{n+1}^*)}{1 - 2^p}. \quad (2.18)$$

2. Runge-Kutta Methods - A Survey

Assuming that the method has s -stages this procedure will require a total of $3s - 1$ function evaluations. Although this procedure often gives a good approximation to the LTE it is relatively expensive and so it is natural to consider cheaper alternatives. One possibility is to use the idea of embedding. This approach consists of defining two methods of consecutive orders with the same coefficients a_{ij} and c_i , but with different b_i . Assuming that y_{n+1} and \bar{y}_{n+1} are the two estimates of $y(x_{n+1})$ given by the main method and the embedded method, respectively, then we can write:

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

$$\bar{y}_{n+1} = y_n + h_n \sum_{i=1}^s \bar{b}_i k_i.$$

By evaluating the difference between the results computed by each method we get an estimate of the local truncation error in the lower order method. Thus when deriving an order p Runge-Kutta method we will also find an order $p - 1$ method for the purpose of error estimation. Usually we denote the pair of formulae by (p, q) or $p(q)$ where p is the order of the main method and q is the order of the embedded one. In the formulae developed in this thesis we will consider $q = p - 1$. The error estimate can be used to monitor the local error and to control the stepsize of integration. Again if we consider y_{n+1} and \bar{y}_{n+1} as two approximations to the true solution, where y_{n+1}, \bar{y}_{n+1} are obtained by the formulae of order $p, p - 1$, respectively, then we can write:

$$y(x_{n+1}) - y_{n+1} = h_n^{p+1} \phi_p + O(h_n^{p+2}), \quad (2.19)$$

$$y(x_{n+1}) - \bar{y}_{n+1} = h_n^p \phi_{p-1} + O(h_n^{p+1}). \quad (2.20)$$

Subtracting these two equations and neglecting $O(h_n^{p+1})$, we get:

$$h_n^p \phi_{p-1} \cong y_{n+1} - \bar{y}_{n+1}. \quad (2.21)$$

Suppose we wish to choose a step h_{n+1} so that

$$\|LTE\| = TOL, \quad (2.22)$$

where TOL is some user defined local error tolerance. The next step is given by:

$$h_{n+1} = h_n \left(\frac{TOL}{\|E_{n+1}\|_\infty} \right)^{1/p}, \quad (2.23)$$

2. Runge-Kutta Methods - A Survey

where $\|E_{n+1}\|_\infty \equiv \|y_{n+1} - \bar{y}_{n+1}\|_\infty$. However since this estimate assumes that the terms in h_n^{p+1} are negligible we introduce a safety factor SF into our formula. Hence we actually choose h_{n+1} as

$$h_{n+1} = SF h_n \left(\frac{TOL}{\|E_{n+1}\|_\infty} \right)^{1/p}. \quad (2.24)$$

An alternative way of choosing the step is:

$$h_{n+1} = SF h_n \left(\frac{TOL}{\left\| \frac{E_{n+1}}{h_n} \right\|_\infty} \right)^{1/p}. \quad (2.25)$$

Equation (2.24) gives the error per step control (EPS) and equation (2.25) gives the error per unit step control (EPUS). This technique of estimating the error is called *Fehlberg embedding*. If the solution \bar{y}_{n+1} is acceptable on accuracy grounds we actually accept the asymptotically more accurate solution y_{n+1} and this technique is known as *local extrapolation*. An extension to this error estimation technique comes from considering *First Same As Last* (FSAL) methods. Such methods come from assuming that the last row of the matrix A is formed by the vector b . In this approach there are $s + 1$ stages but the last function evaluation of each step will be the same as the first function evaluation of the following one. That is $a_{s+1,i} = b_i$. If we denote by k_{s+1}^n the last evaluation during the step from x_n to x_{n+1} , then

$$k_{s+1}^n = f(x_n + h, y_n + h \sum_{i=1}^s a_{s+1,i} k_i^n). \quad (2.26)$$

If we go another step forward, x_{n+1} to x_{n+2} , there is no need to evaluate the first k of the next step as it is given by:

$$k_1^{n+1} = f(x_n + h, y_n + h \sum_{i=1}^s b_i k_i^n) = k_{s+1}^n. \quad (2.27)$$

Important contributions to this area of deriving embedded Runge-Kutta pairs were made by E. Fehlberg, J. H. Verner and R. England. In [Fehlberg68] and [Fehlberg69] Fehlberg derives formulae of orders 1-8 with stepsize control. Together with each order p formula an order $p + 1$ formula is derived for the purpose of error estimation. These formulae require more function evaluations per step than conventional Runge-Kutta formulae without stepsize control. However they require less function evaluations than conventional formulae with

2. Runge-Kutta Methods - A Survey

Richardson extrapolation. Simplifying assumptions and the FSAL technique for the error estimation were used by Fehlberg. The formulae presented by Fehlberg have several free parameters and these were chosen so that the leading terms of the LTE are small. However these methods for order greater than 4 are unsatisfactory because they give error estimates which are identically zero when solving quadrature problems of the form $y' = f(x)$.

One of the first people to consider embedded formulae was England [**England69**] who gives an example of a 5(4) pair with 6 stages, which is the minimum number of stages possible for an order 4 method with an error estimate. The advantage of this method is that b_5 and b_6 are both zero, which make it efficient when the error estimate is not required.

Many pairs of embedded formulae have been derived by Verner [**Cooper72**], [**Verner78**], [**Verner79**], [**Verner91a**] and [**Verner91b**]. In particular, in [**Verner78**], pairs of formulae of order 6(5), 7(6), 8(7) and 9(8) were developed. These overcome the disadvantage of the methods presented by Fehlberg in that they do not give zero error estimates for quadrature problems.

Dormand and Prince [**Dormand80**], [**Prince81**] have also given examples of embedded formulae of orders 5(4), 6(5) and 8(7). Their aim is to develop pairs of formulae that have 'small' principal local truncation errors and 'good' regions of absolute stability. They use the quantities:

$$\begin{aligned}
 A^{(p+1)} &= \|T^{(p+1)}\| \\
 B^{(p+1)} &= \frac{\|\bar{T}^{(p+1)}\|}{\|\bar{T}^{(p)}\|} \\
 C^{(p+1)} &= \frac{\|\bar{T}^{(p+1)} - T^{(p+1)}\|}{\|\bar{T}^{(p)}\|}
 \end{aligned} \tag{2.28}$$

to determine the efficiency of the $p(p-1)$ formulae, where $T^{(i)}$ are the polynomials corresponding to the i^{th} order conditions. The bar is associated with the order $p-1$ formula. For a discussion of these norms and how they are used to design methods the reader is referred to [**Dormand80**], [**Prince81**].

One of the more efficient 5(4) formulae which has been derived by Dormand

2. Runge-Kutta Methods - A Survey

and Prince [Dormand80] is given below.

$$\begin{array}{c|cccccc}
 0 & 0 & & & & & \\
 \frac{1}{5} & \frac{1}{5} & 0 & & & & \\
 \frac{3}{10} & \frac{3}{40} & \frac{9}{40} & 0 & & & \\
 \frac{3}{5} & \frac{3}{10} & -\frac{9}{10} & \frac{6}{5} & 0 & & \\
 \frac{2}{3} & \frac{226}{729} & -\frac{25}{27} & \frac{880}{729} & \frac{55}{729} & 0 & \\
 1 & -\frac{181}{270} & \frac{5}{2} & -\frac{266}{297} & -\frac{91}{27} & \frac{189}{55} & 0 \\
 \hline
 & \frac{19}{216} & 0 & \frac{1000}{2079} & -\frac{125}{216} & \frac{81}{88} & \frac{5}{56} \\
 \hline
 & \frac{31}{540} & 0 & \frac{190}{297} & -\frac{145}{108} & \frac{351}{220} & \frac{1}{20}
 \end{array} \tag{2.29}$$

In this case the quantities (2.28) are $A^{(6)} \cong 0.1227 \times 10^{-2}$, $B^{(6)} \cong 1.2808$, $C^{(6)} \cong 1.3436$.

2.7 Stability

As we saw in the previous chapter it is common to study stability using the scalar test equation $y' = \lambda y$. If we consider an explicit Runge-Kutta method of order p with s stages and if we expand the exact solution $y(x_{n+1})$, we have:

$$y(x_{n+1}) = y(x_n) + h\lambda y(x_n) + \frac{1}{2!}h^2\lambda^2 y(x_n) + \cdots + \frac{1}{p!}h^p\lambda^p y(x_n) + O(h^{p+1}).$$

Under the localizing assumption that $y_n = y(x_n)$, the value y_{n+1} given by the method applied to the test equation differs from the the previous Taylor

2. Runge-Kutta Methods - A Survey

expansion by $O(h^{p+1})$ terms. So we can write:

$$y_{n+1} = y_n \left(1 + h\lambda + \frac{1}{2!}h^2\lambda^2 + \cdots + \frac{1}{p!}h^p\lambda^p \right) + O(h^{p+1}).$$

The stability function is given by:

$$\begin{aligned} R(z) &= 1 + z \sum_i b_i + z^2 \sum_{ij} b_i a_{ij} + z^3 \sum_{ijk} b_i a_{ij} a_{jk} + \cdots \\ &= 1 + z \sum_i b_i + z^2 \sum_i b_i c_i + z^3 \sum_{ij} b_i a_{ij} c_j + \cdots \end{aligned} \quad (2.30)$$

which is a polynomial of degree $\leq s$. If a Runge-Kutta method is of order p , then:

$$R(z) = 1 + z + \frac{1}{2!}z^2 + \frac{1}{3!}z^3 + \cdots + \frac{1}{p!}z^p + O(z^{p+1}). \quad (2.31)$$

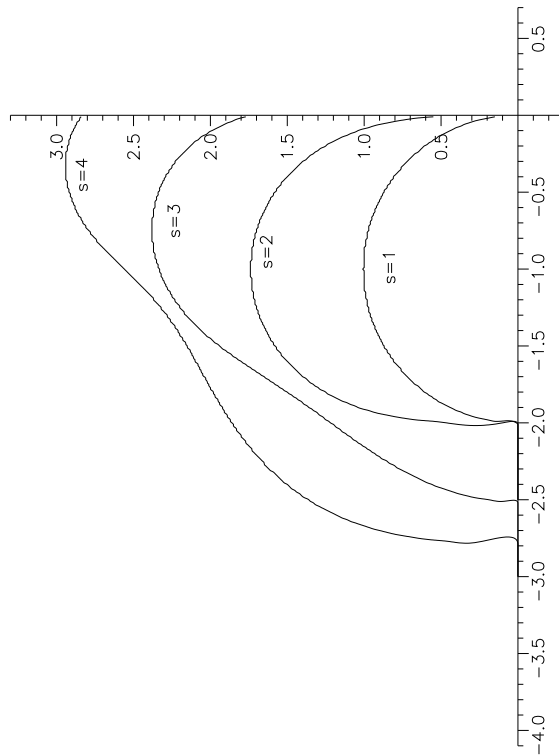
Consequently we can say that all explicit Runge-Kutta methods with $p = s$ possess the following stability function:

$$R(z) = \frac{y_{n+1}}{y_n} = 1 + z + \frac{1}{2!}z^2 + \cdots + \frac{1}{s!}z^s. \quad (2.32)$$

When we compare linear multistep methods with Runge-Kutta methods we can see that in both cases implicit methods can have larger stability regions than explicit methods. But as the order increases, for explicit linear multistep methods the stability regions tend to decrease; while for Runge-Kutta methods the opposite happens, as the order increases the stability regions become larger. When deriving high order Runge-Kutta methods we often have many free coefficients available. When $p > 4$ we can choose some of these coefficients to improve the stability of our formulae. We will return to this point later.

In Figure 2.1 we have plotted the stability regions of the explicit Runge-Kutta methods of orders $p \equiv s = 1, 2, 3, 4$.

2. Runge-Kutta Methods - A Survey



[r]

Figure 2.1: Stability of explicit Runge-Kutta methods of orders 1 to 4

2.8 Interpolation

A numerical method for the integration of (1.1) typically finds values of y and its derivative at a discrete set of points $\{x_n\}$. Often, however, solution values are required at “off-step” points and to compute these it is most efficient to develop an interpolating polynomial. When using Adams methods, solutions at “off-step” points can be obtained simply by evaluating a natural underlying polynomial at the appropriate point. This in turn means that the steplength of integration used with these formulae is determined by accuracy requirements alone and is practically independent of the distribution of the output points. In contrast, explicit Runge-Kutta methods do not normally have such an interpolating polynomial available and this means that most implementations of such methods choose the stepsize of integration so as to hit all output points exactly. This can, of course, lead to an extremely expensive integration if there are many output points and this generally makes Runge-Kutta methods uncompetitive with Adams methods in this situation. Recently a lot of attention has been given to developing codes that include interpolating polynomials [**Dormand86**],[**Gladwell**],[**Shampine85b**],[**Shampine86**],[**Shampine92**],[**Enright86**]. With high order Runge-Kutta methods there is no natural interpolant and it is a challenging problem to derive interpolants with the appropriate order. In the case of block explicit Runge-Kutta methods going ν steps forward, we have $\nu + 1$ solutions available: $y_n, y_{n+1}, \dots, y_{n+\nu}$. If we now fit a ν^{th} order interpolating polynomial $P(x)$ through the points (x_i, y_i) , $i = n, n + 1, \dots, n + \nu$, we can compute an order ν solution y_τ at any intermediate point x_τ as $y_\tau = P(x_\tau)$. This is the procedure followed in [**Cash85**],[**Cash86**],[**Cash89**]. The ability to derive high order interpolating polynomials is a big advantage of block explicit Runge-Kutta methods. Enright et al. [**Enright86**] describe a general procedure to construct interpolants for Runge-Kutta formulae. They study how many extra function evaluations are necessary to construct these polynomials. The number of stages required for a C^1 -interpolant with various $p(p-1)$ Runge-Kutta formulae has been given by Verner [**Verner93**]. In particular he found the following:

2. Runge-Kutta Methods - A Survey

$p(p - 1)$	stages	stages for a C^1 -interpolant of order p
6(5)	8	12
7(6)	10	16
8(7)	13	21
9(8)	16	26

We note from this table that the number of function evaluations required to obtain the desired interpolants rises rapidly with order.

2.9 Main Codes Used at Present

Early codes based on $p(p - 1)$ pairs and using local extrapolation were developed from formulae derived by England and Fehlberg. Shampine and Watts [**Shampine76a**] used a Fehlberg 5(4) pair with 6 stages in their code RKF45. This was one of the first very widely used Runge-Kutta codes. Hull, Enright and Jackson [**Hull76**] used a 6(5) pair with 8 stages. The code derived was called DVERK. In the NAG library a 5(4) formula developed by Merson is used. The subroutine is called D02PAF but note that the order 5 formula becomes of order 3 when the equations are not linear differential equations with constant coefficients. The codes RKF45, D02PAF, XRK and DVERK use error per step and local extrapolation.

Chapter 3

Block Explicit Runge-Kutta Methods

3.1 Introduction

In this chapter we consider the derivation of block explicit Runge-Kutta formulae. The aim is to derive methods which are more efficient than conventional Runge-Kutta methods of the same order by deriving formulae which use information over several integration steps. Thus in one sense these methods could be regarded as multistep versions of Runge-Kutta methods. In an attempt to derive a general class of formulae which include both Runge-Kutta and linear multistep formulae, Butcher [Butcher66] proposed *General Linear Methods* (GLM).

A general linear method can be written in the form (A, B, C) such that the formula for $y_i^{(n)}$, $i = 1, 2, \dots, N$ is given by:

$$y_i^{(n+1)} = \sum_{j=1}^N a_{ij} y_j^{(n)} + h_n \sum_{j=1}^N b_{ij} f(x_j, y_j^{(n+1)}) + h_n \sum_{j=1}^N c_{ij} f(x_j, y_j^{(n)}), \quad (3.1)$$

where A, B, C are $N \times N$ matrices with elements a_{ij} , b_{ij} , c_{ij} , respectively.

In Chapter 2 we showed that if we define $Y_j = y_n + h_n \sum_{i=1}^s a_{ji} k_i$, $j = 1, 2, \dots, s$, then an s -stage Runge-Kutta method can be written in the form:

$$y_{n+1} = y_n + h_n \sum_{j=1}^s b_j f(x_n + c_j h_n, Y_j). \quad (3.2)$$

It can be seen that Runge-Kutta methods belong to the class of general linear methods if we consider $N = s + 1$ and identify $y_1^{(n)}, y_2^{(n)}, \dots, y_N^{(n)}$ with

3. Block Explicit Runge-Kutta Methods

$Y_1, Y_2, \dots, Y_s, y_n$ and if the matrices A , B and C are given by:

$$A = \begin{bmatrix} 0 & \dots & 0 & 1 \\ \vdots & \ddots & \vdots & \vdots \\ 0 & \dots & 0 & 1 \end{bmatrix}, \quad B = \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1s} & 0 \\ a_{21} & a_{22} & \dots & a_{2s} & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ a_{s1} & a_{s2} & \dots & a_{ss} & 0 \\ b_1 & b_2 & \dots & b_s & 0 \end{bmatrix}, \quad C = 0.$$

If, in addition, we compute an embedded solution \bar{y}_{n+1} defined by:

$$\bar{y}_{n+1} = y_n + h_n \sum_{j=1}^s \bar{b}_j f(x_n + c_j h_n, Y_j), \quad (3.3)$$

this corresponds to $N = s + 2$ with the matrices A , B now defined by:

$$A = \begin{bmatrix} 0 & \dots & 0 & 1 & 0 \\ 0 & \dots & 0 & 1 & 0 \\ \vdots & \ddots & \vdots & \vdots & \vdots \\ 0 & \dots & 0 & 1 & 0 \end{bmatrix}, \quad B = \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1s} & 0 & 0 \\ a_{21} & a_{22} & \dots & a_{2s} & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ a_{s1} & a_{s2} & \dots & a_{ss} & 0 & 0 \\ b_1 & b_2 & \dots & b_s & 0 & 0 \\ \bar{b}_1 & \bar{b}_2 & \dots & \bar{b}_s & 0 & 0 \end{bmatrix}.$$

In this chapter we will focus our attention on block explicit Runge-Kutta methods. These methods can be thought of in two different ways. Either they can be regarded as Runge-Kutta formulae defined over more than one steplength or they can be regarded as standard explicit Runge-Kutta formulae with more than the minimum number of function evaluations. Of course these methods are more expensive per step than standard methods but we hope for better stability and better accuracy than with conventional methods. The two major aims, improvement of accuracy and stability, have to be balanced against the need to avoid expending unnecessary computational effort. The hope is that because of the extra accuracy of these methods they will be competitive with standard explicit Runge-Kutta formulae.

In what follows we will regard block explicit Runge-Kutta formulae as multistep formulae. Such formulae have all the characteristics of standard explicit Runge-Kutta formulae except that they are no longer single step in nature in the sense that a p^{th} order block formula produces a p^{th} order approximation to the solution at several points instead of at one point only. These formulae are

3. Block Explicit Runge-Kutta Methods

still self-starting and it is easy to change the steplength of integration. We also hope to be able to change the order easily. The results obtained in [Cash83a] indicate the superiority of the variable step/variable order approach over conventional fixed order Runge-Kutta formulae. Block Runge-Kutta methods are able to compute intermediate solutions at “off-step” points with relatively little additional cost due to the fact that it is straightforward to derive an interpolating polynomial. Block explicit Runge-Kutta formulae, with s -stages, have the general form:

$$y_{n+i} = y_n + h_n \sum_{j=1}^s b_{ij} k_j, \quad 1 \leq i \leq \nu, \quad (3.4)$$

where

$$k_j = f(x_n + c_j h_n, y_n + h_n \sum_{i=1}^{j-1} a_{ji} k_i). \quad (3.5)$$

For consistency we require:

$$\sum_{j=1}^s b_{ij} = i, \quad 1 \leq i \leq \nu. \quad (3.6)$$

Written in this form it is clear that our formulae are a combination of linear multistep and Runge-Kutta methods. The first step of the block is a standard explicit Runge-Kutta formula but, unlike with standard formulae, the second solution y_{n+2} is computed using information from the interval $[x_n, x_{n+1}]$ as well as from $[x_{n+1}, x_{n+2}]$. However, while it is in a sense much more natural to consider our formulae as block formulae it is easier to analyse them using the one-step framework because we are then able to appeal to Butcher’s analysis. Thus from now on when analysing the order of our methods we will regard them as standard one-step Runge-Kutta methods using more than the minimum number of function evaluations but having enhanced accuracy and stability properties. In addition our formulae viewed in this way have the novel property that they produce accurate approximations at certain points within the range (x_n, x_{n+1}) as well as at x_{n+1} .

We will consider block formulae of order p which integrate ν steps forward using a total of s function evaluations and so it is natural to speak of them as requiring s/ν function evaluations per step. At each point in the block we have available a cheap estimate of the local truncation error in the solution of lower order by use of the Fehlberg embedding approach.

3. Block Explicit Runge-Kutta Methods

Some examples of block Runge-Kutta methods do exist in the literature [Cash85],[Cash86],[Cash89] but the only one that has been analysed in any great detail is a 6(4) formula given in [Cash89]. Experience has shown that for low accuracy requirements it is often advantageous to use low order formulae while for efficiency over a wide range of medium and strict tolerances it is often advantageous to look for high order formulae. In this thesis we will systematically examine formulae of orders 2 to 7. We stop at order 7 because the derivation of formulae of order higher than 7 rapidly becomes more difficult and we do not feel going beyond order 7 is a realistic aim for this thesis. As well as deriving efficient block formulae we will also look at the problem of deriving good embedded formulae for the purpose of error estimation as well as deriving good interpolants. In the case of a block method going ν steps forward, we have $\nu + 1$ solutions available: $y_n, y_{n+1}, \dots, y_{n+\nu}$. In addition, if we construct the block in a suitable way, we will have $y'_n, y'_{n+1}, \dots, y'_{n+\nu}$ available as well. We can now fit an interpolating polynomial of degree $2\nu + 1$ through these data and, providing $2\nu + 1 \geq p$ which is the order of the method, we can compute a p^{th} order approximation at any point in the interval at no extra cost by evaluating the interpolating polynomial at this point.

Associated with (3.4)–(3.5) we will derive an embedded formula of the form:

$$\bar{y}_{n+i} = y_n + h_n \sum_{j=1}^s \bar{b}_{ij} k_j, \quad 1 \leq i \leq \nu, \quad (3.7)$$

where again

$$\sum_{j=1}^s \bar{b}_{ij} = i, \quad 1 \leq i \leq \nu. \quad (3.8)$$

This produces a lower order solution, \bar{y}_{n+i} , that is used for the purpose of local error estimation. Such a method can be represented in a Butcher's table in the following way:

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

In this thesis we confine our attention to the case of going two steps forward, $\nu = 2$, so in order to simplify the notation the b 's will appear with only one subscript. It is of course possible to derive formulae with $\nu > 2$ but we leave this for future research.

3. Block Explicit Runge-Kutta Methods

In deriving block formulae the hope is that we can choose the coefficients so that we will get better stability and better accuracy than with conventional methods. We will seek to get sufficient accuracy so that the modified method is more efficient than standard methods even though it takes more function evaluations per step. The block formula integrates forward over a step νh and so, in order to be able to use Butcher's analysis, it is convenient to "normalise" the formula to a step h by dividing all the elements of the Butcher's matrix by a factor ν .

When comparing the theoretical properties of the block and standard Runge-Kutta formulae of a given order we will also take into account the fact that they use a different number of function evaluations. To compensate for this we will introduce a scaling factor when comparing stability regions and accuracy. We will come back to this point later. When deriving our formulae we will look for positive values of the b_i if possible since very negative values will lead to non desirable bounds on the rounding error produced in a single step.

In what follows we will consider the case where $h_n \equiv h$ is fixed.

3.2 Accuracy Criteria

In this section we discuss how we compare the accuracy properties of the conventional and block p^{th} order Runge-Kutta formulae. We assume that the conventional method of order p uses s_c function evaluations per step and that the block method of order p uses s_b function evaluations per step. Since both methods are of order p it follows that we can write their local truncation errors in the form:

$$LTE_{conv} = h^{p+1} \sum_{i=1}^{N_p} T_i D_i + O(h^{p+2}),$$

$$LTE_{block} = h^{p+1} \sum_{i=1}^{N_p} \hat{T}_i D_i + O(h^{p+2}).$$

As explained previously, the D_i are elementary differentials while the T_i and \hat{T}_i are polynomials depending on the coefficients of the Runge-Kutta methods. We now wish to derive a method to compare the local accuracies of these two

3. Block Explicit Runge-Kutta Methods

formulae given that they require different numbers of function evaluations. In a sense we will compare their accuracy per function evaluation. We will do this by considering what happens when we use each method to integrate from x_n to x_{n+i} using the same number of function evaluations. More specifically let $r = \text{lcm}(s_c, s_b)$. It then follows that the conventional method uses r function evaluations to go $\frac{r}{s_c}$ steps forward each of length $\frac{s_c}{s_b}h$ while the block method uses r function evaluations to go $\frac{r}{s_b}$ steps forward of length h . Note that in this case both methods have gone $\frac{r}{s_b}h$ forward using r functions. We compare the accuracy of the methods by comparing the accuracies achieved at $\hat{x} \equiv x_n + \frac{r}{s_b}h$. To do this we scale the two previous equations for the LTE to get:

$\overline{LTE}_{conv} \equiv$ local error in the solution at \hat{x} using the conventional method,

$$\overline{LTE}_{conv} = \frac{r}{s_c} \left(\frac{s_c}{s_b} h \right)^{p+1} \sum_{i=1}^{N_p} T_i D_i + O(h^{p+2}),$$

$\overline{LTE}_{block} \equiv$ local error in the solution at \hat{x} using the block method,

$$\overline{LTE}_{block} = \frac{r}{s_b} h^{p+1} \sum_{i=1}^{N_p} \hat{T}_i D_i + O(h^{p+2}).$$

In order for the block method to have better accuracy than the conventional method we require that:

$$| \overline{LTE}_{block} | < | \overline{LTE}_{conv} |$$

i.e.

$$\left| \frac{r}{s_b} h^{p+1} \sum_{i=1}^{N_p} \hat{T}_i D_i \right| < \left| \frac{r}{s_c} \left(\frac{s_c}{s_b} h \right)^{p+1} \sum_{i=1}^{N_p} T_i D_i \right|.$$

This can be simplified to the requirement that

$$\left| \sum_{i=1}^{N_p} \hat{T}_i D_i \right| < \left| \left(\frac{s_c}{s_b} \right)^p \sum_{i=1}^{N_p} T_i D_i \right|.$$

Since both these sums contain many terms we need to introduce a norm to measure their size. We follow the approach of Shampine [Shampine86] and measure the size of the local truncation error as:

$$\|T\| = \left(\sum_{i=1}^{N_p} T_i^2 \right)^{1/2}.$$

3. Block Explicit Runge-Kutta Methods

Thus the accuracy requirement for our block method is:

$$\|\hat{T}\| < \left(\frac{s_c}{s_b}\right)^p \|T\|. \quad (3.9)$$

In deriving our block formulae it is this accuracy requirement that we will attempt to satisfy.

3.3 Stability Criteria

We now consider the absolute stability properties of our methods. All conventional explicit Runge-Kutta methods have a stability polynomial of the form (2.31). It immediately follows that all p -stage explicit Runge-Kutta formulae of order p , for $p = 1, 2, 3, 4$, have the same interval of absolute stability as shown in the following table:

order	stability polynomial	interval of absolute stability
1	$1 + z$	$(-2, 0)$
2	$1 + z + \frac{1}{2}z^2$	$(-2, 0)$
3	$1 + z + \frac{1}{2}z^2 + \frac{1}{6}z^3$	$(-2.51, 0)$
4	$1 + z + \frac{1}{2}z^2 + \frac{1}{6}z^3 + \frac{1}{24}z^4$	$(-2.78, 0)$

The corresponding regions of absolute stability are well known and were given in Chapter 2.

To compare the stability regions of the conventional and block methods we scale them to take into consideration the different number of function evaluations used. We believe that this is the fairest way to compare them. If we consider again the case where the conventional method of order p uses s_c function evaluations per step and the block method of order p uses s_b function evaluations per step, we will require that the block method will have a real interval of stability which is at least s_b/s_c times that of the conventional method. Thus what we are doing in effect is to measure stability per function evaluation. We would like to find nice round stability regions for the block methods. Ideally we want them to inclose completely the stability regions of

3. Block Explicit Runge-Kutta Methods

the conventional methods but we know from [Hairer91] that it is not possible because of the following theorem:

Theorem 3.1 *If $R_1(z)$ and $R_2(z)$ are the stability functions of degrees s_1 and s_2 of two explicit Runge-Kutta methods of orders ≥ 1 , then one scaled stability domain can never completely contain another.*

We have now established the aims of this investigation. What we wish to do is to obtain high order block explicit Runge-Kutta formulae. These should contain embedded formulae for the purpose of error estimation. They should also generate accurate solutions at several points to allow interpolation. In addition they should satisfy the accuracy and stability requirements just described to make them competitive with standard methods. In what follows we will derive block methods of increasing orders starting at order 2.

3.4 Order 2

A conventional explicit Runge-Kutta method of order 2 requires two function evaluations per step and has the form:

$$\begin{aligned} k_1 &= f(x_n, y_n) \\ k_2 &= f(x_n + c_2h, y_n + hc_2k_1) \\ y_{n+1} &= y_n + h(b_1k_1 + b_2k_2). \end{aligned}$$

The corresponding Butcher's table is:

$$\begin{array}{c|cc} 0 & 0 & \\ c_2 & c_2 & 0 \\ \hline & b_1 & b_2 \end{array} \quad (3.10)$$

For this method to be second order the coefficients have to satisfy the following order relations:

$$\begin{cases} b_1 + b_2 = 1 \\ b_2c_2 = \frac{1}{2}. \end{cases}$$

The local truncation error is:

$$LTE = h^3(T_1D_1 + T_2D_2) + O(h^4),$$

3. Block Explicit Runge-Kutta Methods

where the error terms are given by:

$$T_1 = \sum_{i=1}^2 b_i c_i^2 - \frac{1}{3} = b_2 c_2^2 - \frac{1}{3}$$

$$T_2 = \sum_{i,j=1}^2 b_i a_{ij} c_j - \frac{1}{6} = -\frac{1}{6}.$$

If we apply this formula to the test equation $y' = \lambda y$, and put $z = \lambda h$, the stability polynomial is: $\frac{y_{n+1}}{y_n} = 1 + z + \frac{1}{2}z^2$. It is easy to show, using this polynomial, that the interval of absolute stability of the standard second order method is: $(-2, 0)$.

Going two steps forward with a standard second order method takes four function evaluations (this corresponds to applying the same method twice). To rewrite this in the framework of a one-step formula we divide all the coefficients by the number of steps, which in this case is two. The corresponding Butcher's table is:

$$\begin{array}{c|ccc}
 0 & 0 & & \\
 \frac{c_2}{2} & \frac{c_2}{2} & 0 & \\
 \frac{1}{2} & \frac{b_1}{2} & \frac{b_2}{2} & 0 \\
 \frac{1+c_2}{2} & \frac{b_1}{2} & \frac{b_2}{2} & \frac{c_2}{2} & 0 \\
 \hline
 & \frac{b_1}{2} & \frac{b_2}{2} & \frac{b_1}{2} & \frac{b_2}{2}
 \end{array} \tag{3.11}$$

The stability polynomial associated with this is: $\frac{y_{n+1}}{y_n} = (1 + \frac{1}{2}z + \frac{1}{8}z^2)^2$. This corresponds to the interval of absolute stability: $(-4, 0)$.

The order relations in this case are:

$$\left\{ \begin{array}{l} \mathbf{b_1} + b_2 = 1 \\ \mathbf{b_2}c_2 = \frac{1}{2}. \end{array} \right.$$

We have used bold face for the variables that we choose to be defined by each particular equation.

3. Block Explicit Runge-Kutta Methods

The local truncation error is:

$$LTE = h^3(T_1D_1 + T_2D_2) + O(h^4),$$

where the error terms are given by:

$$T_1 = \sum_{i=1}^4 b_i c_i^2 - \frac{1}{3} = \frac{1}{8}c_2 - \frac{1}{12}$$

$$T_2 = \sum_{i,j=1}^4 b_i a_{ij} c_j - \frac{1}{6} = -\frac{1}{24}.$$

3.4.1 Block Formula of Order 2 with 3 Stages

We saw above that if we apply a standard second order explicit Runge-Kutta method twice we obtain a two-step method which uses four function evaluations. However there are only 3 free coefficients available since we have used exactly the same formula twice. It is natural to ask whether we could take the second step using a totally different formula and whether there would be any advantage in doing this. We will seek to obtain improved efficiency by examining a formula which uses only three functions to advance two steps. This has the form:

$$\begin{array}{c|ccc} 0 & 0 & & \\ \frac{c_2}{2} & \frac{c_2}{2} & 0 & \\ c_3 & a_{31} & a_{32} & 0 \\ \hline & \hat{b}_1 & \hat{b}_2 & \hat{b}_3 \end{array} \quad (3.12)$$

For this formula the row-sum condition is: $c_3 = \mathbf{a}_{31} + a_{32}$.

The associated stability polynomial is: $\frac{y_{n+1}}{y_n} = 1 + z + \frac{1}{2}z^2 + tz^3$, where $t = \hat{b}_3 \mathbf{a}_{32} \frac{c_2}{2}$. We wish to choose the coefficients so that the interval of absolute stability is at least as good as the scaled stability interval: $\frac{3}{4}(-4, 0) = (-3, 0)$. This ensures that the block formula has a stability interval per function evaluation which is at least as good as that of the conventional formula.

Using Butcher's analysis it follows that the conditions for (3.12) to be of order 2 are:

$$\begin{cases} \hat{\mathbf{b}}_1 + \hat{b}_2 + \hat{b}_3 = 1 \\ \hat{\mathbf{b}}_2 \frac{c_2}{2} + \hat{b}_3 c_3 = \frac{1}{2}. \end{cases}$$

3. Block Explicit Runge-Kutta Methods

We have 7 parameters that remain to be determined. If we consider the row-sum condition and the order conditions we get 3 equations to be solved and this specifies 3 of the parameters. Thus we have just four parameters left free: $c_2, c_3, \hat{b}_3, a_{32}$, say. These will be chosen to satisfy the stability and accuracy requirements described earlier.

The error terms associated with formula (3.12) are:

$$\hat{T}_1 = \sum_{i=1}^3 \hat{b}_i c_i^2 - \frac{1}{3} = \hat{b}_2 \frac{c_2^2}{4} + \hat{b}_3 c_3^2 - \frac{1}{3}$$

$$\hat{T}_2 = \sum_{i,j=1}^3 \hat{b}_i a_{ij} c_j - \frac{1}{6} = \hat{b}_3 a_{32} \frac{c_2}{2} - \frac{1}{6} = t - \frac{1}{6}.$$

The condition for accuracy, corresponding to (3.9), in this case is:

$$\left(\hat{T}_1^2 + \hat{T}_2^2\right)^{\frac{1}{2}} < \left(\frac{4}{3}\right)^2 (T_1^2 + T_2^2)^{\frac{1}{2}}. \quad (3.13)$$

Thus what we are looking for is a Runge-Kutta formula which has an interval of absolute stability at least $(-3, 0)$ and which satisfies the accuracy condition (3.13). Many such formulae have been found. In what follows we give an example where the $\hat{b}_i, c_i \in [0, 1]$ and $t = \frac{329}{5000}$:

$$\begin{array}{c|ccc} 0 & 0 & & \\ \frac{1}{20} & \frac{1}{20} & 0 & \\ \frac{7}{10} & -\frac{59}{50} & \frac{47}{25} & 0 \\ \hline & \frac{1}{10} & \frac{1}{5} & \frac{7}{10} \end{array} \quad (3.14)$$

Here the ratio between the right hand side and the left hand side of (3.13) is equal to 1.4411. We are now in a position to clarify some of the remarks made earlier. If we examine equation (3.14) we see that it is a standard Runge-Kutta formula of order 2 requiring 3 function evaluations. However we know that we can derive a second order formula with only 2 function evaluations. Therefore, what we need to do is to ensure that (3.14) has sufficiently good accuracy and stability so that it is still competitive with the standard formula.

3. Block Explicit Runge-Kutta Methods

For the purpose of local error estimation we need to find an embedded formula of order 1. We have at our disposal 3 function evaluations. Starting with (3.14) we add a new row of b 's to get an embedded formula. The associated stability polynomial is: $\frac{y_{n+1}}{y_n} = 1 + z + w_1 z^2 + w_2 z^3$, where $w_1 = \bar{\mathbf{b}}_2 \frac{c_2}{2} + \bar{\mathbf{b}}_3 c_3$ and $w_2 = \bar{\mathbf{b}}_3 a_{32} \frac{c_2}{2}$. The order 1 condition, $\sum_{i=1}^3 \bar{b}_i = 1$, gives us \bar{b}_1 . We are now left with 2 free parameters: \bar{b}_2 and \bar{b}_3 . These will be determined by finding a good stability region. Then we will check if the method also has good accuracy. The error term in the embedded formula is given by $\bar{T}_1 = \sum_{i=1}^3 \bar{b}_i c_i - \frac{1}{2}$. After some numerical searching we choose $w_1 = \frac{9}{20}$ and $w_2 = \frac{11}{200}$. The formula obtained is:

$$\begin{array}{c|ccc}
 0 & 0 & & \\
 \frac{1}{20} & \frac{1}{20} & 0 & \\
 \frac{7}{10} & -\frac{59}{50} & \frac{47}{25} & 0 \\
 \hline
 & \frac{1}{10} & \frac{1}{5} & \frac{7}{10} \\
 \hline
 & -\frac{37}{94} & \frac{38}{47} & \frac{55}{94}
 \end{array} \tag{3.15}$$

From now on we will denote the 2-norm of the i^{th} order error coefficients of the embedded formula by $\|\bar{T}^{(i)}\|$ and by $\|\tilde{T}^{(i)}\|$ the correspondent norm for the embedded conventional formula.

If we consider as conventional embedded formula of order 1 the following

3. Block Explicit Runge-Kutta Methods

formula:

$$\begin{array}{c|cccc}
 0 & 0 & & & \\
 \frac{1}{20} & \frac{1}{20} & 0 & & \\
 \frac{1}{2} & \frac{1}{4} & \frac{1}{4} & 0 & \\
 \frac{21}{20} & \frac{1}{4} & \frac{1}{4} & \frac{1}{20} & 0 \\
 \hline
 & \frac{1}{4} & \frac{1}{4} & \frac{1}{4} & \frac{1}{4}
 \end{array} \tag{3.16}$$

Here $\|\tilde{T}^{(2)}\| = \frac{9}{40}$. We are now in a position to compare it with the one of the block formula obtained. For the embedded formula (3.15): $\|\bar{T}^{(2)}\| = \frac{1}{20}$. The comparison is done using the condition:

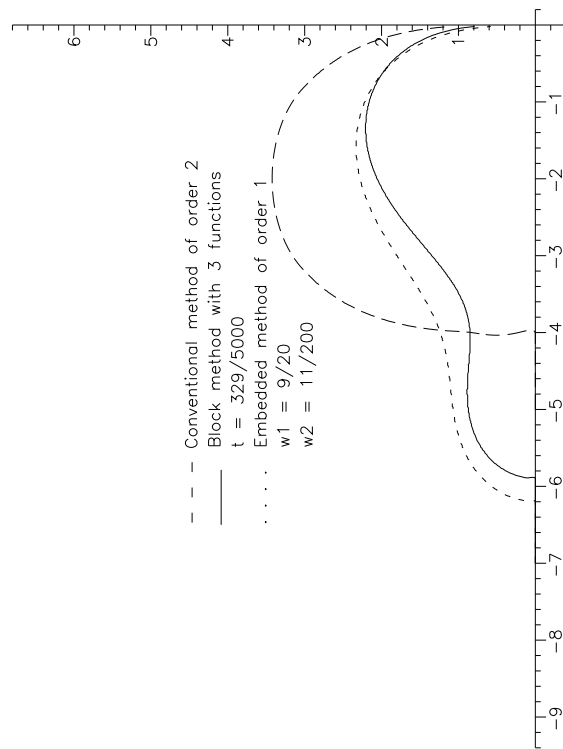
$$\|\bar{T}^{(p)}\| < 2 \left(\frac{1}{2}\right)^p \|\tilde{T}^{(p)}\|. \tag{3.17}$$

In this case the we have $\frac{1}{20} < 2 \left(\frac{1}{2}\right)^2 \frac{9}{40}$. The ratio between the right hand side and the left hand side of the previous equation is equal to $\frac{9}{4}$; this shows that the block formula is better. The plots of the stability regions of the three methods are given in Figure 3.1, where the dotted line corresponds to the conventional method, the continuous one to the block method and the small dotted line to the embedded formula.

3.4.2 Block Formula of Order 2 with 4 Stages

Although we found a good formula in terms of stability and accuracy using 3 function evaluations, we will try to get even better results by using an extra function evaluation. That is we investigate the possibility of going two steps

3. Block Explicit Runge-Kutta Methods



[r]

Figure 3.1: Stability region of order 2 with 3 stages

3. Block Explicit Runge-Kutta Methods

forward using 4 function evaluations. This Runge-Kutta formula has the form:

$$\begin{array}{c|cccc}
 0 & 0 & & & \\
 \frac{c_2}{2} & \frac{c_2}{2} & 0 & & \\
 c_3 & a_{31} & a_{32} & 0 & \\
 c_4 & a_{41} & a_{42} & a_{43} & 0 \\
 \hline
 & \hat{b}_1 & \hat{b}_2 & \hat{b}_3 & \hat{b}_4
 \end{array} \tag{3.18}$$

For this formula the row-sum conditions are:

$$c_3 = \mathbf{a}_{31} + a_{32}$$

$$c_4 = \mathbf{a}_{41} + a_{42} + a_{43}.$$

The associated stability polynomial is: $\frac{y_{n+1}}{y_n} = 1 + z + \frac{1}{2}z^2 + t_1z^3 + t_2z^4$, where $t_1 = \hat{b}_3a_{32}\frac{c_2}{2} + \hat{b}_4(\mathbf{a}_{42}\frac{c_2}{2} + a_{43}c_3)$ and $t_2 = \hat{b}_4\mathbf{a}_{43}a_{32}\frac{c_2}{2}$. As both the block and conventional formulae use the same number of function evaluations we do not need to scale the stability intervals when comparing them. We will try to derive a modified method with a stability interval at least as good as the conventional one.

Using Butcher's analysis it follows that the conditions for (3.18) to be of order 2 are:

$$\begin{cases} \hat{b}_1 + \hat{b}_2 + \hat{b}_3 + \hat{b}_4 = 1 \\ \hat{b}_2\frac{c_2}{2} + \hat{b}_3c_3 + \hat{b}_4c_4 = \frac{1}{2}. \end{cases}$$

We have 12 parameters that remain to be determined. If we consider the row-sum conditions and the order conditions, four of these parameters are defined. Now we just have 8 free parameters that are chosen to satisfy the stability and accuracy requirements.

The error terms associated with (3.18) are:

$$\hat{T}_1 = \sum_{i=1}^4 b_i c_i^2 - \frac{1}{3} = \hat{b}_2 \frac{c_2^2}{4} + \hat{b}_3 c_3^2 + \hat{b}_4 c_4^2 - \frac{1}{3}$$

$$\hat{T}_2 = \sum_{i,j=1}^4 b_i a_{ij} c_j - \frac{1}{6} = \hat{b}_3 a_{32} \frac{c_2}{2} + \hat{b}_4 (a_{42} \frac{c_2}{2} + a_{43} c_3) - \frac{1}{6} = t_1 - \frac{1}{6}.$$

The comparison for accuracy (3.9) in this case is:

$$\left(\hat{T}_1^2 + \hat{T}_2^2 \right)^{\frac{1}{2}} < \left(T_1^2 + T_2^2 \right)^{\frac{1}{2}}. \tag{3.19}$$

3. Block Explicit Runge-Kutta Methods

After some experimentation we choose our formula with $t_1 = \frac{3}{20}$ and $t_2 = \frac{1}{64}$ which satisfies both the accuracy and the stability requirements. Our final formula is:

$$\begin{array}{c|cccc}
 0 & 0 & & & \\
 \frac{1}{20} & \frac{1}{20} & 0 & & \\
 \frac{1}{5} & -\frac{1}{10} & \frac{3}{10} & 0 & \\
 \frac{7}{10} & \frac{123}{140} & -\frac{5}{3} & \frac{125}{84} & 0 \\
 \hline
 & \frac{1}{10} & \frac{1}{5} & 0 & \frac{7}{10}
 \end{array} \tag{3.20}$$

Here the ratio between the right hand side and the left hand side of (3.19) is equal to 4.713.

For the purpose of local error estimation we seek an embedded formula of order 1 using the four function evaluations that are available. We will add on to (3.20) a new row of \bar{b} 's to get the embedded formula. The associated stability polynomial is: $\frac{y_{n+1}}{y_n} = 1 + z + w_1 z^2 + w_2 z^3 + w_3 z^4$, where $w_1 = \bar{\mathbf{b}}_2 \frac{c_2}{2} + \bar{b}_3 c_3 + \bar{b}_4 c_4$, $w_2 = \bar{\mathbf{b}}_3 a_{32} \frac{c_2}{2} + \bar{b}_4 (a_{42} c_2 + a_{43} c_3)$ and $w_3 = \bar{\mathbf{b}}_4 a_{43} a_{32} \frac{c_2}{2}$.

The order 1 condition, $\sum_{i=1}^4 \bar{b}_i = 1$, gives us \bar{b}_1 . We are now left with 3 free parameters: \bar{b}_2 , \bar{b}_3 and \bar{b}_4 . These will be determined by finding a good stability region. Then we will check if the method also has good accuracy. The error term in the embedded formula is given by $\bar{T}_1 = \sum_{i=1}^4 \bar{b}_i c_i - \frac{1}{2}$. After some numerical searching we choose $w_1 = \frac{49}{100}$, $w_2 = \frac{3}{20}$ and $w_3 = \frac{1}{64}$. The formula

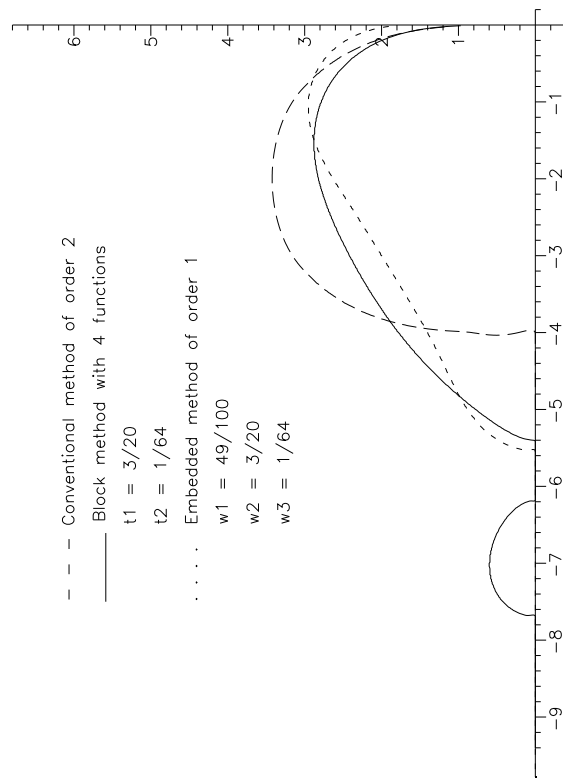
3. Block Explicit Runge-Kutta Methods

obtained is:

$$\begin{array}{c|cccc}
 0 & 0 & & & \\
 \frac{1}{20} & \frac{1}{20} & 0 & & \\
 \frac{1}{5} & -\frac{1}{10} & \frac{3}{10} & 0 & \\
 \frac{7}{10} & \frac{123}{140} & -\frac{5}{3} & \frac{125}{84} & 0 \\
 \hline
 & \frac{1}{10} & \frac{1}{5} & 0 & \frac{7}{10} \\
 \hline
 & \frac{3}{10} & 0 & 0 & \frac{7}{10}
 \end{array} \tag{3.21}$$

The error term in the embedded formula is: $||\bar{T}^{(2)}|| = \frac{1}{100}$. The plots of the stability regions of the three methods are given in Figure 3.2, where again the dotted line corresponds to the conventional method, the continuous one to the block method and the small dotted line to the embedded formula.

3. Block Explicit Runge-Kutta Methods



[r]

Figure 3.2: Stability region of order 2 with 4 stages

3. *Block Explicit Runge-Kutta Methods*

It is easily seen that the results obtained with four function evaluations are better than those obtained with just three. We gain in real stability but lose a little bit on the imaginary axis. It would have been ideal if we had the stability region of the conventional method completely inclosed in the stability region of the block method. This however it is not possible as we have shown earlier. In terms of accuracy there are no doubts that the modified method is better.

3.5 Order 3

We now go on to consider the derivation of third order block methods. In this case we are going to consider a slightly different approach as we will use a known formula for the conventional method. We have chosen a 3(2) formula from [Bogacki89] to compute the approximate solution y_{n+1} at the first point of the block. This formula uses a FSAL technique in that it uses the first evaluation of the next step for the embedded method and is given by:

3. Block Explicit Runge-Kutta Methods

$$\begin{array}{c|ccc}
 0 & 0 & & \\
 \frac{1}{2} & \frac{1}{2} & 0 & \\
 \frac{3}{4} & 0 & \frac{3}{4} & 0 \\
 \hline
 1 & \frac{2}{9} & \frac{1}{3} & \frac{4}{9} & 0 \\
 \hline
 & \frac{7}{24} & \frac{1}{4} & \frac{1}{3} & \frac{1}{8}
 \end{array} \tag{3.22}$$

Using the same approach as in the order two case we will consider going two steps forward but using the one-step formulation. The corresponding Butcher's table is:

$$\begin{array}{c|cccc}
 0 & 0 & & & \\
 \frac{1}{4} & \frac{1}{4} & 0 & & \\
 \frac{3}{8} & 0 & \frac{3}{8} & 0 & \\
 \frac{1}{2} & \frac{1}{9} & \frac{1}{6} & \frac{2}{9} & 0 \\
 \frac{3}{4} & \frac{1}{9} & \frac{1}{6} & \frac{2}{9} & \frac{1}{4} & 0 \\
 \frac{7}{8} & \frac{1}{9} & \frac{1}{6} & \frac{2}{9} & 0 & \frac{3}{8} & 0 \\
 \hline
 & \frac{1}{9} & \frac{1}{6} & \frac{2}{9} & \frac{1}{9} & \frac{1}{6} & \frac{2}{9}
 \end{array} \tag{3.23}$$

It can be seen that this method uses six function evaluations per step. The associated stability polynomial is: $\frac{y_{n+1}}{y_n} = 1 + z + \frac{1}{2}z^2 + \frac{1}{6}z^3 + \frac{7}{192}z^4 + \frac{1}{192}z^5 +$

3. Block Explicit Runge-Kutta Methods

$\frac{1}{2304}z^6$. Which corresponds to the interval of absolute stability: $(-5.02, 0)$. The norm of the coefficients of the local truncation error of the embedded formula is: $\|\tilde{T}^{(3)}\| = \frac{\sqrt{5}}{48} \cong 0.466 \times 10^{-1}$.

The problem we have to consider now is what is the minimum number of function evaluations necessary to derive an efficient block formula. We will try to find a block formula which uses five function evaluations. We are saving one function evaluation per step when compared with the conventional method. Again we are going to consider the block formula as a one-step Runge-Kutta formula, and so we will scale it so that we can apply Butcher's analysis.

3.5.1 Block Formula of Order 3 with 5 Stages

The block formula we consider is:

$$\begin{array}{c|ccccc}
 0 & 0 & & & & \\
 \frac{1}{4} & \frac{1}{4} & 0 & & & \\
 \frac{3}{8} & 0 & \frac{3}{8} & 0 & & \\
 c_4 & a_{41} & a_{42} & a_{43} & 0 & \\
 c_5 & a_{51} & a_{52} & a_{53} & a_{54} & 0 \\
 \hline
 & \hat{b}_1 & \hat{b}_2 & \hat{b}_3 & \hat{b}_4 & \hat{b}_5
 \end{array} \tag{3.24}$$

We have 14 parameters to be determined. For this formula the row-sum conditions are:

$$c_4 = \mathbf{a}_{41} + a_{42} + a_{43}$$

$$c_5 = \mathbf{a}_{51} + a_{52} + a_{53} + a_{54}.$$

The associated stability polynomial is: $\frac{y_{n+1}}{y_n} = 1 + z + \frac{1}{2}z^2 + \frac{1}{6}z^3 + t_1z^4 + t_2z^5$, where $t_1 = \hat{b}_4a_{43}a_{32}c_2 + \hat{b}_5\mathbf{a}_{53}a_{32}c_2 + \hat{b}_5a_{54}(a_{42}c_2 + a_{43}c_3)$ and $t_2 = \hat{b}_5a_{54}\mathbf{a}_{43}a_{32}c_2$. The scaled stability interval in this case is: $\frac{5}{6}(-5.02, 0) \cong (-4.18333, 0)$.

3. Block Explicit Runge-Kutta Methods

Using Butcher's analysis it follows that the conditions for (3.24) to be of order 3 are:

$$\begin{cases} \hat{\mathbf{b}}_1 + \hat{b}_2 + \hat{b}_3 + \hat{b}_4 + \hat{b}_5 = 1 \\ \hat{\mathbf{b}}_2 c_2 + \hat{b}_3 c_3 + \hat{b}_4 c_4 + \hat{b}_5 c_5 = \frac{1}{2} \\ \hat{b}_2 c_2^2 + \hat{\mathbf{b}}_3 c_3^2 + \hat{b}_4 c_4^2 + \hat{b}_5 c_5^2 = \frac{1}{3} \\ \hat{b}_3 a_{32} c_2 + \hat{b}_4 (a_{42} c_2 + a_{43} c_3) + \hat{b}_5 (a_{52} c_2 + a_{53} c_3 + a_{54} c_4) = \frac{1}{6}. \end{cases}$$

The corresponding error terms are:

$$\begin{aligned} \hat{T}_1 &= \sum_{i=1}^5 \hat{b}_i c_i^3 - \frac{1}{4} \\ \hat{T}_2 &= \sum_{i,j=1}^5 \hat{b}_i c_i a_{ij} c_j - \frac{1}{8} \\ \hat{T}_3 &= \sum_{i,j=1}^5 \hat{b}_i a_{ij} c_j^2 - \frac{1}{12} \\ \hat{T}_4 &= \sum_{i,j,k=1}^5 \hat{b}_i a_{ij} a_{jk} c_k - \frac{1}{24} = t_1 - \frac{1}{24}. \end{aligned}$$

Following the approach considered in [Bogacki89] we will choose four of the free coefficients so that the method is of order 4. This implies that $t_1 = \frac{1}{24}$ and that the condition (3.9) of better accuracy is automatically verified. We now have just 4 free coefficients. These will be chosen to make the order 5 coefficients \hat{T}_i , $i = 1, 2, \dots, 9$ of the error term small and to get better stability than the conventional method.

3. Block Explicit Runge-Kutta Methods

An example found with $\hat{b}_i, c_i \in [0, 1]$, $t_1 = \frac{1}{24}$ and $t_2 = \frac{41}{10000}$ is given in the table below:

$$\begin{array}{c|ccccc}
 0 & 0 & & & & \\
 \frac{1}{4} & \frac{1}{4} & 0 & & & \\
 \frac{3}{8} & 0 & \frac{3}{8} & 0 & & \\
 \frac{4}{5} & -\frac{40664}{209375} & \frac{1276}{3125} & \frac{122672}{209375} & 0 & \\
 1 & \frac{347}{120} & -\frac{989}{330} & -\frac{194}{255} & \frac{8375}{4488} & 0 \\
 \hline
 & \frac{11}{120} & \frac{32}{165} & \frac{128}{425} & \frac{1675}{4488} & \frac{1}{25}
 \end{array} \tag{3.25}$$

This formula has $\|\hat{T}^{(5)}\| \cong 0.0085$.

As we have mentioned before, we are interested in obtaining an embedded formula of order 2 for the purpose of error estimation. If we consider the block formula (3.25) we will add a new row of b 's. The associated stability polynomial is: $\frac{y_{n+1}}{y_n} = 1 + z + \frac{1}{2}z^2 + w_1z^3 + w_2z^4 + w_3z^5$, where
 $w_1 = \bar{\mathbf{b}}_3a_{32}c_2 + \bar{b}_4(a_{42}c_2 + a_{43}c_3) + \bar{b}_5(a_{52}c_2 + a_{53}c_3 + a_{54}c_4)$,
 $w_2 = \bar{\mathbf{b}}_4a_{43}a_{32}c_2 + \bar{b}_5(a_{53}a_{32}c_2 + a_{54}(a_{42}c_2 + a_{43}c_3))$ and
 $w_3 = \bar{\mathbf{b}}_5a_{54}a_{43}a_{32}c_2$.

To be an order 2 formula it has to satisfy the following order conditions:

$$\begin{cases} \bar{\mathbf{b}}_1 + \bar{b}_2 + \bar{b}_3 + \bar{b}_4 + \bar{b}_5 = 1 \\ \bar{\mathbf{b}}_2c_2 + \bar{b}_3c_3 + \bar{b}_4c_4 + \bar{b}_5c_5 = \frac{1}{2}. \end{cases}$$

With these two order conditions we can define \bar{b}_1 and \bar{b}_2 , say. In this case we have left just 3 free parameters: $\bar{b}_3, \bar{b}_4, \bar{b}_5$. These will be determined by finding a good stability region. Then we will check if the method also has good accuracy. The error terms are given by:

$$\bar{T}_1 = \sum_{i=1}^5 \bar{b}_i c_i^2 - \frac{1}{3}$$

3. Block Explicit Runge-Kutta Methods

$$\bar{T}_2 = \sum_{i,j=1}^5 \bar{b}_i a_{ij} c_j - \frac{1}{6} = w_1 - \frac{1}{6}.$$

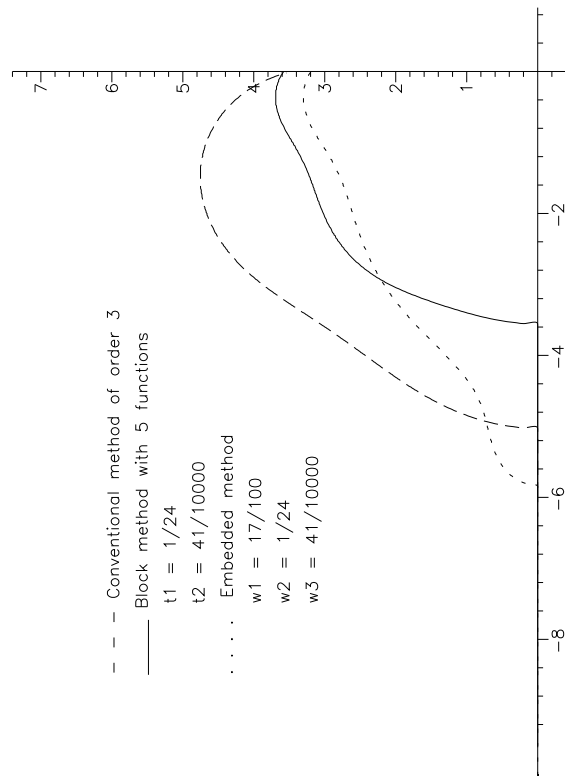
After some numerical searching we choose a formula where $w_1 = \frac{17}{100}$, $w_2 = \frac{1}{24}$ and $w_3 = \frac{41}{1000}$ that is given by:

$$\begin{array}{c|cccccc}
 0 & 0 & & & & & \\
 \frac{1}{4} & \frac{1}{4} & 0 & & & & \\
 \frac{3}{8} & 0 & \frac{3}{8} & 0 & & & \\
 \frac{4}{5} & -\frac{40664}{209375} & \frac{1276}{3125} & \frac{122672}{209375} & 0 & & \\
 1 & \frac{347}{120} & -\frac{989}{330} & -\frac{194}{255} & \frac{8375}{4488} & 0 & \\
 \hline
 & \frac{11}{120} & \frac{32}{165} & \frac{128}{425} & \frac{1675}{4488} & \frac{1}{25} & \\
 \hline
 & \frac{197}{1800} & \frac{116}{825} & \frac{1288}{3825} & \frac{1675}{4488} & \frac{1}{25} &
 \end{array} \tag{3.26}$$

For this formula $\|\bar{T}^{(3)}\| \cong 0.118 \times 10^{-4}$. The plots of the stability regions of the three methods are given in Figure 3.3.

In the case of the order 2 formulae we found that we could find a good formula using 3 function evaluations but we found an even better block formula using 4 function evaluations. We now investigate whether this carries over to third order formulae by examining a block formula with 6 stages.

3. Block Explicit Runge-Kutta Methods



[r]

Figure 3.3: Stability region of order 3 with 5 stages

3.5.2 Block Formula of Order 3 with 6 Stages

The formula which we will consider has the general form:

$$\begin{array}{c|cccccc}
 0 & 0 & & & & & \\
 \frac{1}{4} & \frac{1}{4} & 0 & & & & \\
 \frac{3}{8} & 0 & \frac{3}{8} & 0 & & & \\
 c_4 & a_{41} & a_{42} & a_{43} & 0 & & \\
 c_5 & a_{51} & a_{52} & a_{53} & a_{54} & 0 & \\
 c_6 & a_{61} & a_{62} & a_{63} & a_{64} & a_{65} & 0 \\
 \hline
 & \hat{b}_1 & \hat{b}_2 & \hat{b}_3 & \hat{b}_4 & \hat{b}_5 & \hat{b}_6
 \end{array} \tag{3.27}$$

We have 21 parameters to be determined. For this formula the row-sum conditions are:

$$\begin{aligned}
 c_4 &= \mathbf{a}_{41} + a_{42} + a_{43} \\
 c_5 &= \mathbf{a}_{51} + a_{52} + a_{53} + a_{54} \\
 c_6 &= \mathbf{a}_{61} + a_{62} + a_{63} + a_{64} + a_{65}.
 \end{aligned}$$

The associated stability polynomial is: $\frac{y_{n+1}}{y_n} = 1 + z + \frac{1}{2}z^2 + \frac{1}{6}z^3 + t_1z^4 + t_2z^5 + t_3z^6$, where $t_1 = \hat{b}_4a_{43}a_{32}c_2 + \hat{b}_5(\mathbf{a}_{53}a_{32}c_2 + a_{54}(a_{42}c_2 + a_{43}c_3)) + \hat{b}_6(a_{63}a_{32}c_2 + a_{64}(a_{42}c_2 + a_{43}c_3) + a_{65}(a_{52}c_2 + a_{53}c_3 + a_{54}c_4))$, $t_2 = \hat{b}_5a_{54}\mathbf{a}_{43}a_{32}c_2 + \hat{b}_6(a_{64}a_{43}a_{32}c_2 + a_{65}(a_{53}a_{32}c_2 + a_{54}(a_{42}c_2 + a_{43}c_3)))$ and $t_3 = \hat{b}_6a_{65}a_{54}a_{43}a_{32}c_2$. Recall that the stability interval of the conventional method is: $(-5.02, 0)$. As we are using the same number of function evaluations as with the conventional method there is no need to perform any scaling. This means that we will seek a block method with an interval of absolute stability as least $(-5.02, 0)$.

Using Butcher's analysis it follows that the conditions for (3.27) to be of

3. Block Explicit Runge-Kutta Methods

order 3 are:

$$\left\{ \begin{array}{l} \hat{\mathbf{b}}_1 + \hat{b}_2 + \hat{b}_3 + \hat{b}_4 + \hat{b}_5 + \hat{b}_6 = 1 \\ \hat{\mathbf{b}}_2 c_2 + \hat{b}_3 c_3 + \hat{b}_4 c_4 + \hat{b}_5 c_5 + \hat{b}_6 c_6 = \frac{1}{2} \\ \hat{b}_2 c_2^2 + \hat{\mathbf{b}}_3 c_3^2 + \hat{b}_4 c_4^2 + \hat{b}_5 c_5^2 + \hat{b}_6 c_6^2 = \frac{1}{3} \\ \hat{b}_3 a_{32} c_2 + \hat{b}_4 (a_{42} c_2 + a_{43} c_3) + \hat{b}_5 (a_{52} c_2 + a_{53} c_3 + a_{54} c_4) + \hat{b}_6 \sum_{i=2}^5 a_{6i} c_i = \frac{1}{6}. \end{array} \right.$$

The corresponding error terms are:

$$\begin{aligned} \hat{T}_1 &= \sum_{i=1}^6 \hat{b}_i c_i^3 - \frac{1}{4} \\ \hat{T}_2 &= \sum_{i,j=1}^6 \hat{b}_i c_i a_{ij} c_j - \frac{1}{8} \\ \hat{T}_3 &= \sum_{i,j=1}^6 \hat{b}_i a_{ij} c_j^2 - \frac{1}{12} \\ \hat{T}_4 &= \sum_{i,j,k=1}^6 \hat{b}_i a_{ij} a_{jk} c_k - \frac{1}{24} = t_1 - \frac{1}{24}. \end{aligned}$$

Following the approach of [Bogacki89] we will choose four of the free coefficients so that the method is of order 4. This implies that $t_1 = \frac{1}{24}$ and that the condition of better accuracy is automatically satisfied. This means that we just have 10 free coefficients at our disposal. These will to be chosen to make the order 5 coefficients \hat{T}_i , $i = 1, 2, \dots, 9$ of the error term small and to get better stability than the conventional method.

An example found with $\hat{b}_i, c_i \in [0, 1]$, $t_1 = \frac{1}{24}$, $t_2 = \frac{4}{625}$ and $t_3 = \frac{1}{2500}$ is

3. Block Explicit Runge-Kutta Methods

given in the table below:

0	0					
$\frac{1}{4}$	$\frac{1}{4}$	0				
$\frac{3}{8}$	0	$\frac{3}{8}$	0			
$\frac{1}{2}$	$\frac{2}{5}$	$-\frac{2}{5}$	$\frac{1}{2}$	0		
$\frac{7}{10}$	$\frac{10425679}{17769375}$	$-\frac{2048576}{1974375}$	$\frac{7930427}{7107750}$	$\frac{128}{3645}$	0	
$\frac{9}{10}$	$\frac{121679348}{479773125}$	$\frac{48781612}{53308125}$	$\frac{866266321}{959546250}$	$-\frac{500325547}{319848750}$	$\frac{9}{10}$	0
	$\frac{1238}{9375}$	$\frac{172}{9375}$	$\frac{3104}{9375}$	$\frac{8569}{37500}$	$\frac{1}{50}$	$\frac{27}{100}$

For this formula $\|\hat{T}^{(5)}\| \cong 0.2797 \times 10^{-2}$.

As we have explained before, we are interested in obtaining an embedded formula of order 2 for the purpose of error estimation. If we consider the block formula (3.28) we will add a new row of b 's. The corresponding stability polynomial is: $\frac{y_{n+1}}{y_n} = 1 + z + \frac{1}{2}z^2 + w_1z^3 + w_2z^4 + w_3z^5 + w_4z^6$, where $w_1 = \bar{\mathbf{b}}_3 a_{32} c_2 + \bar{b}_4(a_{42}c_2 + a_{43}c_3) + \bar{b}_5(a_{52}c_2 + a_{53}c_3 + a_{54}c_4) + \bar{b}_6(a_{62}c_2 + a_{63}c_3 + a_{64}c_4 + a_{65}c_5)$, $w_2 = \bar{\mathbf{b}}_4 a_{43} a_{32} c_2 + \bar{b}_5(a_{53} a_{32} c_2 + a_{54}(a_{42}c_2 + a_{43}c_3)) + \bar{b}_6(a_{63} a_{32} c_2 + a_{64}(a_{42}c_2 + a_{43}c_3) + a_{65}(a_{52}c_2 + a_{53}c_3 + a_{54}c_4))$, $w_3 = \bar{\mathbf{b}}_5(a_{54} a_{43} a_{32} c_2) + \bar{b}_6(a_{64}(a_{42}c_2 + a_{43}c_3) + a_{65} \sum_{i=2}^4 a_{5i} c_i)$ and $w_4 = \bar{\mathbf{b}}_6 a_{65} a_{54} a_{43} a_{32} c_2$.

To be an order 2 formula it has to satisfy the following order relations:

$$\begin{cases} \bar{\mathbf{b}}_1 + \bar{b}_2 + \bar{b}_3 + \bar{b}_4 + \bar{b}_5 + \bar{b}_6 = 1 \\ \bar{\mathbf{b}}_2 c_2 + \bar{b}_3 c_3 + \bar{b}_4 c_4 + \bar{b}_5 c_5 + \bar{b}_6 c_6 = \frac{1}{2}. \end{cases}$$

We have 6 coefficients left to be determined. With these two order conditions we can define \bar{b}_1 and \bar{b}_2 , say. In this case we just have left 4 free parameters: $\bar{b}_3, \bar{b}_4, \bar{b}_5, \bar{b}_6$. These will be determined by finding a good stability region. Then we will check if the method also has good accuracy.

3. Block Explicit Runge-Kutta Methods

The error terms are given by:

$$\bar{T}_1 = \sum_{i=1}^6 \bar{b}_i c_i^2 - \frac{1}{3}$$

$$\bar{T}_2 = \sum_{i,j=1}^6 \bar{b}_i a_{ij} c_j - \frac{1}{6} = w_1 - \frac{1}{6}.$$

After some numerical searching we choose a formula where $w_1 = \frac{17}{100}$, $w_2 = \frac{421}{10000}$, $w_3 = \frac{4}{625}$ and $w_4 = \frac{1}{2500}$ that is given by:

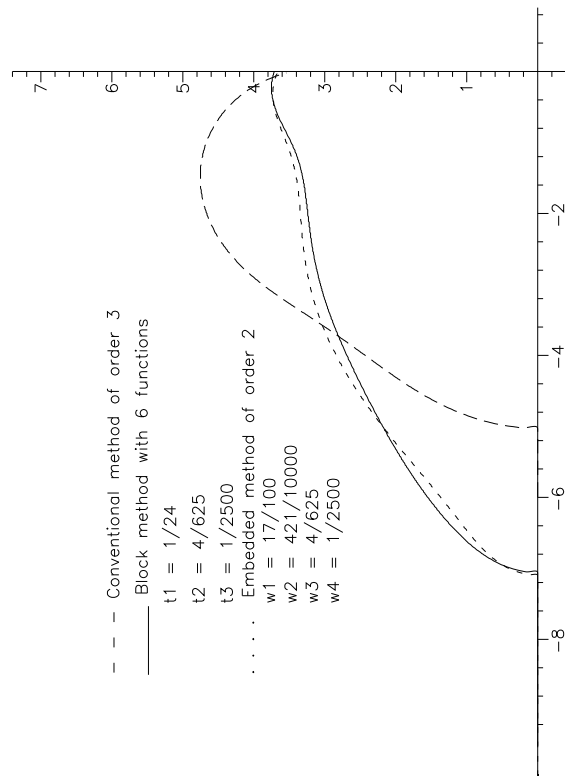
0	0					
$\frac{1}{4}$	$\frac{1}{4}$	0				
$\frac{3}{8}$	0	$\frac{3}{8}$	0			
$\frac{1}{2}$	$\frac{2}{5}$	$-\frac{2}{5}$	$\frac{1}{2}$	0		
$\frac{7}{10}$	$\frac{10425679}{17769375}$	$-\frac{2048576}{1974375}$	$\frac{7930427}{7107750}$	$\frac{128}{3645}$	0	
$\frac{9}{10}$	$-\frac{121679348}{479773125}$	$\frac{48781612}{53308125}$	$\frac{866266321}{959546250}$	$-\frac{500325547}{319848750}$	$\frac{9}{10}$	0
	$\frac{1238}{9375}$	$\frac{172}{9375}$	$\frac{3104}{9375}$	$\frac{8569}{37500}$	$\frac{1}{50}$	$\frac{27}{100}$
	$\frac{71448709057}{1155009375000}$	$\frac{40413713857}{128334375000}$	$\frac{122433793049}{288752343750}$	$-\frac{529874581163}{1540012500000}$	$\frac{1065841}{3900000}$	$\frac{27}{100}$

(3.29)

For this formula $\|\bar{T}^{(3)}\| \cong 0.01300$. The plots of the stability regions of the three methods are given in Figure 3.4.

When comparing the results obtained with 5 and 6 stages we can see that we have improved the real stability. In terms of embedded the new block formula is slightly worse than the block formula with five stages but it is still better than the conventional formula, as if we consider (3.17), $0.01300 < \frac{1}{4} \frac{\sqrt{5}}{48}$.

3. Block Explicit Runge-Kutta Methods



[r]

Figure 3.4: Stability region of order 3 with 6 stages

3.6 Order 4

We now consider the derivation of a fourth order block method. We have chosen a 4(3) formula from [Dormand84] to compute the approximate solution y_{n+1} at the first point of the block. This formula is denoted by RK4(3)5G in [Dormand84]. It requires five function evaluations per step and can be written as:

$$\begin{array}{c|cccccc}
 0 & 0 & & & & & \\
 \frac{5+\sqrt{5}}{10} & \frac{5+\sqrt{5}}{10} & 0 & & & & \\
 \frac{5-\sqrt{5}}{10} & \frac{\sqrt{5}}{10} & \frac{5-2\sqrt{5}}{10} & 0 & & & \\
 \frac{5+\sqrt{5}}{10} & -\frac{\sqrt{5}}{10} & 0 & \frac{5+2\sqrt{5}}{10} & 0 & & \\
 1 & 1 & \frac{-5+2\sqrt{5}}{2} & \frac{-\sqrt{5}}{2} & \frac{5-\sqrt{5}}{2} & 0 & \\
 \hline
 & \frac{1}{12} & 0 & \frac{5}{12} & \frac{5}{12} & \frac{1}{12} & \\
 \hline
 & 0 & 0 & \frac{5+\sqrt{5}}{12} & \frac{5-\sqrt{5}}{12} & \frac{1}{6} &
 \end{array} \tag{3.30}$$

Using the same approach as before we will consider going two steps forward but using the one-step formulation to analyse the method. The corresponding

3. Block Explicit Runge-Kutta Methods

Butcher's table is:

0	0										
$\frac{5 + \sqrt{5}}{20}$	$\frac{5 + \sqrt{5}}{20}$	0									
$\frac{5 - \sqrt{5}}{20}$	$\frac{\sqrt{5}}{20}$	$\frac{5 - 2\sqrt{5}}{20}$	0								
$\frac{5 + \sqrt{5}}{20}$	$-\frac{\sqrt{5}}{20}$	0	$\frac{5 + 2\sqrt{5}}{20}$	0							
$\frac{1}{2}$	$\frac{1}{2}$	$\frac{-5 + 2\sqrt{5}}{4}$	$-\frac{\sqrt{5}}{4}$	$\frac{5 - \sqrt{5}}{4}$	0						
$\frac{1}{2}$	$\frac{1}{24}$	0	$\frac{5}{24}$	$\frac{5}{24}$	$\frac{1}{24}$	0					
$\frac{15 + \sqrt{5}}{20}$	$\frac{1}{24}$	0	$\frac{5}{24}$	$\frac{5}{24}$	$\frac{1}{24}$	$\frac{5 + \sqrt{5}}{20}$	0				
$\frac{15 - \sqrt{5}}{20}$	$\frac{1}{24}$	0	$\frac{5}{24}$	$\frac{5}{24}$	$\frac{1}{24}$	$\frac{\sqrt{5}}{20}$	$\frac{5 - 2\sqrt{5}}{20}$	0			
$\frac{15 + \sqrt{5}}{20}$	$\frac{1}{24}$	0	$\frac{5}{24}$	$\frac{5}{24}$	$\frac{1}{24}$	$-\frac{\sqrt{5}}{20}$	0	$\frac{5 + 2\sqrt{5}}{20}$	0		
1	$\frac{1}{24}$	0	$\frac{5}{24}$	$\frac{5}{24}$	$\frac{1}{24}$	$\frac{1}{2}$	$\frac{-5 + 2\sqrt{5}}{4}$	$-\frac{\sqrt{5}}{4}$	$\frac{5 - \sqrt{5}}{4}$	0	
	$\frac{1}{24}$	0	$\frac{5}{24}$	$\frac{5}{24}$	$\frac{1}{24}$	$\frac{1}{24}$	0	$\frac{5}{24}$	$\frac{5}{24}$	$\frac{1}{24}$	(3.31)

As it is easily seen, this method uses ten function evaluations per step. The associated stability polynomial is: $\frac{y_{n+1}}{y_n} = 1 + z + \frac{1}{2}z^2 + \frac{1}{6}z^3 + \frac{1}{24}z^4 + \frac{31}{3840}z^5 + \frac{7}{5760}z^6 + \frac{13}{92160}z^7 + \frac{1}{81920}z^8 + \frac{1}{1474560}z^9 + \frac{1}{58982400}z^{10}$. Using this polynomial it is easy to show that the interval of absolute stability of this method is: $(-11.78, 0)$. The norm of the coefficients of the local truncation error of the embedded formula is: $\|\tilde{T}^{(4)}\| \cong 0.0430$.

3.6.1 Block Formula of Order 4 with 8 Stages

The problem we have to consider now is what is the minimum number of function evaluations necessary to derive an efficient two-step block formula of order 4. We will start by trying to find a block formula which uses eight function evaluations. Again we will consider the block formula as a one-step Runge-Kutta formula and so we have to scale the coefficients so that we can

3. Block Explicit Runge-Kutta Methods

formula requires eight function evaluations per step while the conventional one requires 10 functions.

Using Butcher's analysis it follows that the conditions for (3.32) to be of order 4 are:

$$\left\{ \begin{array}{l} \sum_{i=1}^8 \hat{b}_i = 1 \\ \sum_{i=1}^8 \hat{b}_i c_i = \frac{1}{2} \\ \sum_{i=1}^8 \hat{b}_i c_i^2 = \frac{1}{3} \\ \sum_{i,j=1}^8 \hat{b}_i a_{ij} c_j = \frac{1}{6} \\ \sum_{i=1}^8 \hat{b}_i c_i^3 = \frac{1}{4} \\ \sum_{i,j=1}^8 \hat{b}_i c_i a_{ij} c_j = \frac{1}{8} \\ \sum_{i,j=1}^8 \hat{b}_i a_{ij} c_j^2 = \frac{1}{12} \\ \sum_{i,j,k=1}^8 \hat{b}_i a_{ij} a_{jk} c_k = \frac{1}{24}. \end{array} \right.$$

The corresponding error terms are:

$$\begin{aligned} \hat{T}_1 &= \sum_{i=1}^8 \hat{b}_i c_i^4 - \frac{1}{5} \\ \hat{T}_2 &= \sum_{i,j=1}^8 \hat{b}_i c_i^2 a_{ij} c_j - \frac{1}{10} \\ \hat{T}_3 &= \sum_{i,j,k=1}^8 \hat{b}_i a_{ij} c_j a_{ik} c_k - \frac{1}{20} \\ \hat{T}_4 &= \sum_{i,j=1}^8 \hat{b}_i c_i a_{ij} c_j^2 - \frac{1}{15} \\ \hat{T}_5 &= \sum_{i,j=1}^8 \hat{b}_i a_{ij} c_j^3 - \frac{1}{20} \\ \hat{T}_6 &= \sum_{i,j=1}^8 \hat{b}_i c_i a_{ij} a_{jk} c_k - \frac{1}{30} \end{aligned}$$

3. Block Explicit Runge-Kutta Methods

$$\begin{aligned}\hat{T}_7 &= \sum_{i,j,k=1}^8 \hat{b}_i a_{ij} c_j a_{jk} c_k - \frac{1}{40} \\ \hat{T}_8 &= \sum_{i,j,k=1}^8 \hat{b}_i a_{ij} a_{jk} c_k^2 - \frac{1}{60} \\ \hat{T}_9 &= \sum_{i,j,k,m=1}^8 \hat{b}_i a_{ij} a_{jk} a_{km} c_m - \frac{1}{120} = t_5 - \frac{1}{120}.\end{aligned}$$

As we have so many free parameters we can make the method to be of order 5 and in this case it automatically has better asymptotic accuracy than the conventional one. This implies that $t_5 = \frac{1}{120}$ and that the condition of better accuracy is automatically verified. We will choose the free coefficients so that we get better stability than the conventional formula and to make the order 6 coefficients of the error term small. As we have mentioned before, when the order gets higher the order conditions become more difficult to solve and there is a need to introduce simplifying assumptions. In order to derive an order 4 block formula we will use the following simplifying assumptions. The row-simplifying assumptions are:

$$\sum_{j=2}^{i-1} a_{ij} c_j = \frac{1}{2} c_i^2, \quad i = 7, 8 \quad (3.35)$$

(for $i = 3, 4, 5, 6$ they are already satisfied) and

$$\hat{b}_2 = 0. \quad (3.36)$$

The column-simplifying assumptions are:

$$\sum_{i=1}^8 \hat{b}_i a_{ij} = \hat{b}_j (1 - c_j), \quad j = 2, 3, \dots, 8. \quad (3.37)$$

If we subtract the order conditions (7) and (11) of Table 2.1 we get:

$$\sum_{i=1}^8 \hat{b}_i (1 - c_i) a_{ij} c_j^2 = \frac{1}{60}. \quad (3.38)$$

If we subtract the order conditions (8) and (12) of Table 2.1 we get:

$$\sum_{i=1}^8 \hat{b}_i (1 - c_i) a_{ij} a_{jk} c_k = \frac{1}{120}. \quad (3.39)$$

The coefficients of the block method were found in the following way:

1) $\hat{b}_2 = 0$ is defined by equation (3.36); $c_8 = 1$ is defined by equation (3.37)

3. Block Explicit Runge-Kutta Methods

when $j = 8$.

- 2) c_7, \hat{b}_6 and \hat{b}_8 are chosen arbitrarily.
- 3) Solve the linear system of equations: $\sum_{i=1}^8 \hat{b}_i c_i^j = \frac{1}{j+1}, \quad j = 1, 2, 3, 4$ for $\hat{b}_3, \hat{b}_4, \hat{b}_5, \hat{b}_7$.
- 4) Set $\hat{b}_1 = 1 - \sum_{i=2}^8 \hat{b}_i$.
- 5) a_{75} and a_{76} are chosen arbitrarily.
- 6) a_{74} is given by (3.39).
- 7) a_{73} is given by (3.38).
- 8) a_{72} is given by (3.35) with $i = 7$.
- 9) a_{71} is given by the first of the row-sum conditions (3.33).
- 10) equations (3.37) with $j = 2, 3, 4, 5, 6, 7$ are solved for $a_{82}, a_{83}, a_{84}, a_{85}, a_{86}, a_{87}$.
- 11) a_{81} is given by the second of the row-sum conditions (3.34).

By adopting this approach we were able to find formulae with the desired accuracy. The choice of the free parameters was made by picking the “nicest” stability region. By “nicest” we mean a stability region that is as round as possible and going as far as possible in the real axis. The example chosen has $t_5 = \frac{1}{120}, t_6 = \frac{10165 + 153\sqrt{5}}{8908800}, t_7 = \frac{19}{204800}$ and $t_8 = \frac{69}{23756800}$ which produces the formula:

$$\begin{array}{c|cccccccc}
 0 & 0 & & & & & & & \\
 \frac{5+\sqrt{5}}{20} & \frac{5+\sqrt{5}}{20} & 0 & & & & & & \\
 \frac{5-\sqrt{5}}{20} & \frac{\sqrt{5}}{20} & \frac{5-2\sqrt{5}}{20} & 0 & & & & & \\
 \frac{5+\sqrt{5}}{20} & -\frac{\sqrt{5}}{20} & 0 & \frac{5+2\sqrt{5}}{20} & 0 & & & & \\
 \frac{1}{2} & \frac{1}{2} & \frac{-5+2\sqrt{5}}{4} & -\frac{\sqrt{5}}{4} & \frac{5-\sqrt{5}}{4} & 0 & & & \\
 \frac{1}{2} & \frac{1}{24} & 0 & \frac{5}{24} & \frac{5}{24} & \frac{1}{24} & 0 & & \\
 \frac{4}{5} & -\frac{394}{675} & \frac{1293-571\sqrt{5}}{675} & \frac{2(139+112\sqrt{5})}{675} & \frac{-1015+347\sqrt{5}}{675} & \frac{1}{10} & \frac{23}{50} & 0 & \\
 1 & \frac{79}{348} & \frac{-135+61\sqrt{5}}{36} & \frac{-205+289\sqrt{5}}{522} & \frac{3505-2347\sqrt{5}}{1044} & \frac{833}{2088} & \frac{119}{232} & \frac{75}{116} & 0 \\
 \hline
 & \frac{1}{480} & 0 & \frac{43+33\sqrt{5}}{348} & \frac{43-33\sqrt{5}}{348} & \frac{2}{15} & \frac{3}{10} & \frac{225}{928} & \frac{3}{40}
 \end{array} \tag{3.40}$$

For this formula $\|\hat{T}^{(6)}\| \cong 0.14918 \times 10^{-2}$.

3. Block Explicit Runge-Kutta Methods

The next task is to find a good embedded formula of order 3. If we consider the block formula (3.40) we will add a new row of b's. The corresponding stability polynomial is: $\frac{y_{n+1}}{y_n} = 1 + z + \frac{1}{2}z^2 + \frac{1}{6}z^3 + w_4z^4 + w_5z^5 + w_6z^6 + w_7z^7 + w_8z^8$, where $w_4 = \sum_{i,j,k=1}^8 \bar{b}_i a_{ij} a_{jk} c_k$, $w_5 = \sum_{i,j,k,l=1}^8 \bar{b}_i a_{ij} a_{jk} a_{kl} c_l$, $w_6 = \sum_{i,j,k,l,m=1}^8 \bar{b}_i a_{ij} a_{jk} a_{kl} a_{lm} c_m$, $w_7 = \sum_{i,j,k,l,m,n=1}^8 \bar{b}_i a_{ij} a_{jk} a_{kl} a_{lm} a_{mn} c_n$ and $w_8 = \sum_{i,j,k,l,m,n,o=1}^8 \bar{b}_i a_{ij} a_{jk} a_{kl} a_{lm} a_{mn} a_{no} c_o$.

To be an order 3 formula it has to satisfy the following order relations:

$$\left\{ \begin{array}{l} \sum_{i=1}^8 \bar{b}_i = 1 \\ \sum_{i=1}^8 \bar{b}_i c_i = \frac{1}{2} \\ \sum_{i=1}^8 \bar{b}_i c_i^2 = \frac{1}{3} \\ \sum_{i,j=1}^8 \bar{b}_i a_{ij} c_j = \frac{1}{6}. \end{array} \right.$$

We have 8 coefficients to be determined. With these four order conditions we can define \bar{b}_1 , \bar{b}_2 , \bar{b}_3 and \bar{b}_5 , say. In this case we just have left 4 free parameters: \bar{b}_4 , \bar{b}_6 , \bar{b}_7 , \bar{b}_8 . These will be determined by finding a good stability region. Then we will check if the method has also good accuracy. The error terms are given by:

$$\begin{aligned} \bar{T}_1 &= \sum_{i=1}^8 \bar{b}_i c_i^3 - \frac{1}{4} \\ \bar{T}_2 &= \sum_{i,j=1}^8 \bar{b}_i c_i a_{ij} c_j - \frac{1}{8} \\ \bar{T}_3 &= \sum_{i,j=1}^8 \bar{b}_i a_{ij} c_j^2 - \frac{1}{12} \\ \bar{T}_4 &= \sum_{i,j,k=1}^8 \bar{b}_i a_{ij} a_{jk} c_k - \frac{1}{24} = w_4 - \frac{1}{24}. \end{aligned}$$

After some numerical searching we choose a formula where

$$w_4 = \frac{213821 + 1817\sqrt{5}}{5760000}, w_5 = \frac{5584021 + 57222\sqrt{5}}{1002240000}, w_6 = \frac{226871 + 1224\sqrt{5}}{445440000},$$

3. Block Explicit Runge-Kutta Methods

$$w_7 = \frac{10913}{445440000} \text{ and } w_8 = \frac{69}{148480000}.$$

3. Block Explicit Runge-Kutta Methods

This is given by:

0	0							
$\frac{5 + \sqrt{5}}{20}$	$\frac{5 + \sqrt{5}}{20}$	0						
$\frac{5 - \sqrt{5}}{20}$	$\frac{\sqrt{5}}{20}$	$\frac{5 - 2\sqrt{5}}{20}$	0					
$\frac{5 + \sqrt{5}}{20}$	$-\frac{\sqrt{5}}{20}$	0	$\frac{5 + 2\sqrt{5}}{20}$	0				
$\frac{1}{2}$	$\frac{1}{2}$	$\frac{-5 + 2\sqrt{5}}{4}$	$-\frac{\sqrt{5}}{4}$	$\frac{5 - \sqrt{5}}{4}$	0			
$\frac{1}{2}$	$\frac{1}{24}$	0	$\frac{5}{24}$	$\frac{5}{24}$	$\frac{1}{24}$	0	(3.41)	
$\frac{4}{5}$	$-\frac{394}{675}$	$\frac{1293 - 571\sqrt{5}}{675}$	$\frac{2(139 + 112\sqrt{5})}{675}$	$\frac{-1015 + 347\sqrt{5}}{675}$	$\frac{1}{10}$	$\frac{23}{50}$	0	
1	$\frac{79}{348}$	$\frac{-135 + 61\sqrt{5}}{36}$	$\frac{-205 + 289\sqrt{5}}{522}$	$\frac{3505 - 2347\sqrt{5}}{1044}$	$\frac{833}{2088}$	$\frac{119}{232}$	$\frac{75}{116}$	0
	$\frac{1}{480}$	0	$\frac{43 + 33\sqrt{5}}{348}$	$\frac{43 - 33\sqrt{5}}{348}$	$\frac{2}{15}$	$\frac{3}{10}$	$\frac{225}{928}$	$\frac{3}{40}$
	$\frac{503}{1200}$	0	$\frac{-1760 + 37\sqrt{5}}{6000}$	$\frac{-1760 - 37\sqrt{5}}{6000}$	$\frac{67}{250}$	$\frac{11}{16}$	$\frac{1}{5}$	$\frac{3}{250}$

For this formula $\|\bar{T}^{(4)}\| \cong 0.037539$. The plots of the stability regions of the three methods are given in Figure 3.5.

We use the FSAL technique, i.e.

$$a_{6i} = b_i, \quad 1 \leq i \leq 5, \quad (3.42)$$

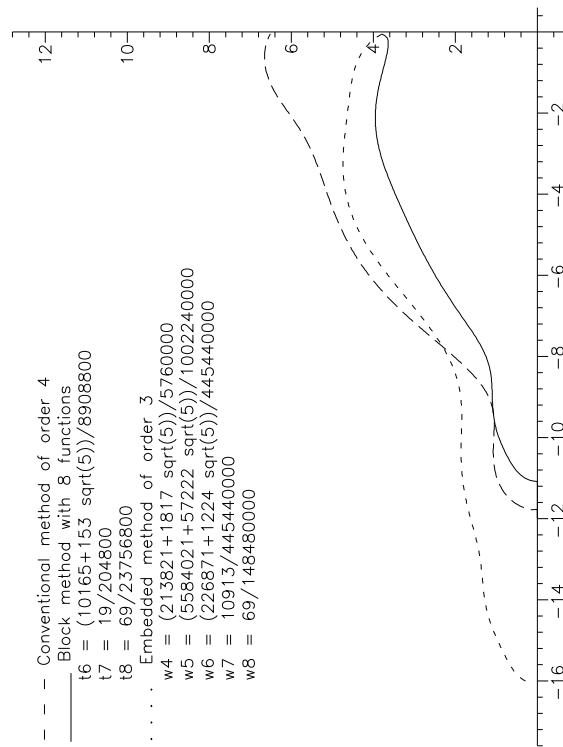
to allow us to construct a quintic Hermite interpolating polynomial without the need to compute extra function evaluations. We have available the following data (x_n, y_n, k_1) , $(x_{n+1/2}, y_{n+1/2}, k_6)$ and (x_{n+1}, y_{n+1}, k_9) , where $k_1 = y'_n$, $k_6 = y'_{n+1/2}$ and $k_9 = y'_{n+1}$, with $y'_{n+i} \equiv f(x_{n+i}, y_{n+i})$, $i = 0, \frac{1}{2}, 1$. Note that k_9 is needed to start the integration at the next block (i.e. k_9 of the current block is k_1 of the next block (FSAL)). The function k_9 is only computed if the block is successful, and of course, interpolation is only carried out after a successful block. We wish to fit an interpolating polynomial of the form:

$$p(x) = \sum_{i=0}^5 a_i x^i \quad (3.43)$$

such that at $x = x_{n+i}$, $i = 0, \frac{1}{2}, 1$, we have:

$$\begin{aligned} p(x) &= y_{n+i} \\ p'(x) &= f(x_{n+i}, y_{n+i}). \end{aligned}$$

3. Block Explicit Runge-Kutta Methods



[r]

Figure 3.5: Stability region of order 4 with 8 stages

3. Block Explicit Runge-Kutta Methods

Denoting the formulae to compute $y_{n+1/2}$, y_{n+1} by:

$$y_{n+1/2} = y_n + h \sum_{i=1}^5 b_{1/2i} k_i \quad (3.44)$$

$$y_{n+1} = y_n + h \sum_{i=1}^8 b_{1i} k_i \quad (3.45)$$

it is straightforward to show that at $x = x_n + \sigma h$:

$$\begin{aligned} p(x) = y_n &+ \sigma h k_1 + 4\sigma^2 \left[h \sum_{i=1}^5 b_{1/2i} k_i - \frac{1}{2} h k_1 \right] \\ &+ 4\sigma^2 \left(\sigma - \frac{1}{2} \right) \left[h k_6 - 4h \sum_{i=1}^5 b_{1/2i} k_i + h k_1 \right] \\ &+ 4\sigma^2 \left(\sigma - \frac{1}{2} \right)^2 \left[h \sum_{i=1}^8 b_{1i} k_i - 2h k_6 - h k_1 + 4h \sum_{i=1}^5 b_{1/2i} k_i \right] \\ &+ 4\sigma^2 \left(\sigma - \frac{1}{2} \right)^2 (\sigma - 1) \left[h k_9 + 4h k_6 + h k_1 - 6h \sum_{i=1}^8 b_{1i} k_i \right]. \end{aligned} \quad (3.46)$$

Since we are controlling the error in a fourth order solution and our interpolating polynomial is of order 5 we would expect it to give satisfactory approximations at “off-step” points. This has been found to be the case in practice.

Numerical experiments with this formula have shown it to be very efficient for the solution of non-stiff problems at low to moderate tolerances. In view of this we will not investigate the possibility of deriving fourth order block formulae with 9 function evaluations but instead go on to consider higher order block formulae.

3.7 Order 5

A fifth order block explicit Runge-Kutta formula has already been considered in some detail in [Cash86]. In this section we use the general algorithm presented in [Cash86] to derive alternative block explicit Runge-Kutta formulae of order 5 with 9 stages. We have tried several standard formulae for computing the solution at the first point in the block i.e. the well known formula RKF45 as well as RK5(4)6M, RK5(4)7M and RK5(4)7S taken from [Dormand80]. As an example we consider the formula RK5(4)6M given by:

3. Block Explicit Runge-Kutta Methods

0	0										
$\frac{1}{5}$	$\frac{1}{5}$	0									
$\frac{3}{10}$	$\frac{3}{40}$	$\frac{9}{40}$	0								
$\frac{3}{5}$	$\frac{3}{10}$	$-\frac{9}{10}$	$\frac{6}{5}$	0							
$\frac{2}{3}$	$\frac{226}{729}$	$-\frac{25}{27}$	$\frac{880}{729}$	$\frac{55}{729}$	0						
1	$-\frac{181}{270}$	$\frac{5}{2}$	$-\frac{266}{297}$	$-\frac{91}{27}$	$\frac{189}{55}$	0					
	$\frac{19}{216}$	0	$\frac{1000}{2079}$	$-\frac{125}{216}$	$\frac{81}{88}$	$\frac{5}{56}$					
	$\frac{31}{540}$	0	$\frac{190}{297}$	$-\frac{145}{108}$	$\frac{351}{220}$	$\frac{1}{20}$					

(3.47)

Using the same approach as before we will consider going two steps forward with the standard formula (3.47) but using the one-step formulation to analyse it. The corresponding Butcher's table is:

0	0											
$\frac{1}{10}$	$\frac{1}{10}$	0										
$\frac{3}{20}$	$\frac{3}{80}$	$\frac{9}{80}$	0									
$\frac{3}{10}$	$\frac{3}{20}$	$-\frac{9}{20}$	$\frac{3}{5}$	0								
$\frac{1}{3}$	$\frac{113}{729}$	$-\frac{25}{54}$	$\frac{440}{729}$	$\frac{55}{1458}$	0							
$\frac{1}{2}$	$-\frac{181}{540}$	$\frac{5}{4}$	$-\frac{133}{297}$	$-\frac{91}{54}$	$\frac{189}{110}$	0						
$\frac{1}{2}$	$\frac{19}{432}$	0	$\frac{500}{2079}$	$-\frac{125}{432}$	$\frac{81}{176}$	$\frac{5}{112}$	0					
$\frac{3}{5}$	$\frac{19}{432}$	0	$\frac{500}{2079}$	$-\frac{125}{432}$	$\frac{81}{176}$	$\frac{5}{112}$	$\frac{1}{10}$	0				
$\frac{13}{20}$	$\frac{19}{432}$	0	$\frac{500}{2079}$	$-\frac{125}{432}$	$\frac{81}{176}$	$\frac{5}{112}$	$\frac{3}{80}$	$\frac{9}{80}$	0			
$\frac{4}{5}$	$\frac{19}{432}$	0	$\frac{500}{2079}$	$-\frac{125}{432}$	$\frac{81}{176}$	$\frac{5}{112}$	$\frac{3}{20}$	$-\frac{9}{20}$	$\frac{3}{5}$	0		
$\frac{5}{6}$	$\frac{19}{432}$	0	$\frac{500}{2079}$	$-\frac{125}{432}$	$\frac{81}{176}$	$\frac{5}{112}$	$\frac{113}{729}$	$-\frac{25}{54}$	$\frac{440}{729}$	$\frac{55}{1458}$	0	
1	$\frac{19}{432}$	0	$\frac{500}{2079}$	$-\frac{125}{432}$	$\frac{81}{176}$	$\frac{5}{112}$	$-\frac{181}{540}$	$\frac{5}{4}$	$-\frac{133}{297}$	$-\frac{91}{54}$	$\frac{189}{110}$	0
	$\frac{19}{432}$	0	$\frac{500}{2079}$	$-\frac{125}{432}$	$\frac{81}{176}$	$\frac{5}{112}$	$\frac{19}{432}$	0	$\frac{500}{2079}$	$-\frac{125}{432}$	$\frac{81}{176}$	$\frac{5}{112}$

(3.48)

As it is easily seen this method uses twelve function evaluations per step. The associated stability polynomial is: $\frac{y_{n+1}}{y_n} = 1 + z + \frac{1}{2}z^2 + \frac{1}{6}z^3 + \frac{1}{24}z^4 + \frac{1}{120}z^5 +$

3. Block Explicit Runge-Kutta Methods

$\frac{319}{230400}z^6 + \frac{89}{460800}z^7 + \frac{83}{3686400}z^8 + \frac{1}{460800}z^9 + \frac{1}{5898240}z^{10} + \frac{1}{98304000}z^{11} + \frac{1}{2621440000}z^{12}$. Using this polynomial it is straightforward to show that the interval of absolute stability of this formula is $(-7.4, 0)$. The norm of the coefficients of the local truncation error of the embedded formula is: $\|\tilde{T}^{(5)}\| \cong 0.527 \times 10^{-2}$.

3.7.1 Block Formula of Order 5 with 9 Stages

To derive our block formula of order 5 we will use the same number of function evaluations as was used in [Cash86], that is 9 function evaluations. The aim of investigating this approach is to find formulae with better accuracy and stability than that given in [Cash86]. If we use the FSAL technique and

3. Block Explicit Runge-Kutta Methods

$t_7 = \sum_{i,j,k,l,m,n=1}^s \hat{b}_i a_{ij} a_{jk} a_{kl} a_{lm} a_{mn} c_n$, $t_8 = \sum_{i,j,k,l,m,n,o=1}^s \hat{b}_i a_{ij} a_{jk} a_{kl} a_{lm} a_{mn} a_{no} c_o$ and $t_9 = \hat{b}_9 a_{98} a_{87} a_{76} a_{65} a_{54} a_{43} a_{32} c_2$. We wish to choose the coefficients so that the interval of absolute stability is at least as good as: $\frac{9}{12}(-7.4, 0) \cong (-5.55, 0)$ since this reflects the fact that the conventional formula uses 12 function evaluations while the block one uses 9.

The conditions for (3.49) to be of order 5 are well known and given in Table 2.1.

Since we have so many free parameters we can choose the method used to compute the solutions at the second point of the block to have order 6 and in this case it automatically has better asymptotic accuracy than the conventional one. This implies that $t_6 = \frac{1}{720}$. We will also choose the free coefficients so that we get better stability than the conventional formula. Again we will have to make use of simplifying assumptions to allow us to solve the order conditions.

The parameters of the block formula (3.49) were found following the procedure given in Cash [Cash86].

After some numerical searching we found $t_7 = \frac{23947551443}{152845056000000}$,

$t_8 = \frac{8602784557}{407586816000000}$, $t_9 = \frac{7}{3072000}$ which gives a formula with

$$\|\hat{T}^{(7)}\| \cong 0.3595 \times 10^{-7}.$$

The next step is to find a good embedded formula of order 4. The corresponding stability polynomial is: $\frac{y_{n+1}}{y_n} = 1 + z + \frac{1}{2}z^2 + \frac{1}{6}z^3 + \frac{1}{24}z^4 + w_5z^5 +$

$w_6z^6 + w_7z^7 + w_8z^8 + w_9z^9$, where $w_5 = \sum_{i,j,k,l=1}^s \bar{b}_i a_{ij} a_{jk} a_{kl} c_l$,

$w_6 = \sum_{i,j,k,l,m=1}^s \bar{b}_i a_{ij} a_{jk} a_{kl} a_{lm} c_m$, $w_7 = \sum_{i,j,k,l,m,n=1}^s \bar{b}_i a_{ij} a_{jk} a_{kl} a_{lm} a_{mn} c_n$,

$w_8 = \sum_{i,j,k,l,m,n,o=1}^s \bar{b}_i a_{ij} a_{jk} a_{kl} a_{lm} a_{mn} a_{no} c_o$ and $w_9 = \bar{b}_9 a_{98} a_{87} a_{76} a_{65} a_{54} a_{43} a_{32} c_2$.

We now have 9 coefficients to be determined. With the eight order 4 conditions we can define: $\bar{b}_1, \bar{b}_2, \bar{b}_3, \bar{b}_4, \bar{b}_5, \bar{b}_6, \bar{b}_7$ and \bar{b}_8 , say. In this case we just have left 1 free parameter: \bar{b}_9 . This will be determined by finding a good stability region. Then we will check if the method also has good accuracy.

3. Block Explicit Runge-Kutta Methods

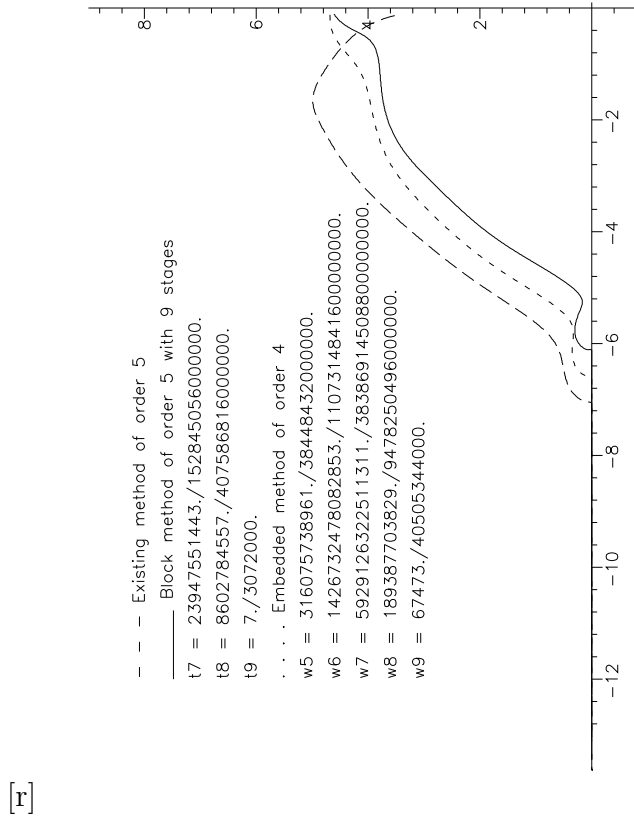


Figure 3.6: Stability region of order 5 with 9 stages

After some numerical searching we choose a formula where

$$w_5 = \frac{316075738961}{38448432000000}, w_6 = \frac{1426732478082853}{1107314841600000000},$$

$$w_7 = \frac{5929126322511311}{38386914508800000000}, w_8 = \frac{189387703829}{9478250496000000}, w_9 = \frac{67473}{40505344000}.$$

This formula is given by (3.52) and for this formula $\|\bar{T}^{(5)}\| \cong 0.4355 \times 10^{-6}$. We note that the new block formula has better accuracy and stability than the conventional formula. The plots of the stability regions of the three methods are given in Figure 3.6.

We use the FSAL technique, i.e.

$$a_{7i} = b_i, \quad 1 \leq i \leq 6, \quad (3.53)$$

to allow us to construct a quintic Hermite interpolating polynomial without the need to compute extra function evaluations. We have available the following

3. Block Explicit Runge-Kutta Methods

data (x_n, y_n, k_1) , $(x_{n+1/2}, y_{n+1/2}, k_7)$ and $(x_{n+1}, y_{n+1}, k_{10})$, where $k_1 = y'_n$, $k_7 = y'_{n+1/2}$ and $k_{10} = y'_{n+1}$, with $y'_{n+i} \equiv f(x_{n+i}, y_{n+i})$, $i = 0, \frac{1}{2}, 1$. Note that k_{10} is needed to start the integration at the next block (i.e. k_{10} of the current block is k_1 of the next block (FSAL)). The function k_{10} is only computed if the block is successful, and of course, interpolation is only carried out after a successful block. We now examine the quality of our interpolating polynomial. We wish to fit an interpolating polynomial of the form:

$$p(x) = \sum_{i=0}^5 a_i x^i \quad (3.54)$$

such that at $x = x_{n+i}$, $i = 0, \frac{1}{2}, 1$, we have:

$$\begin{aligned} p(x) &= y_{n+i} \\ p'(x) &= f(x_{n+i}, y_{n+i}). \end{aligned}$$

Denoting the formulae to compute $y_{n+1/2}$, y_{n+1} by:

$$y_{n+1/2} = y_n + h \sum_{i=1}^6 b_{1/2i} k_i \quad (3.55)$$

$$y_{n+1} = y_n + h \sum_{i=1}^9 b_{1i} k_i \quad (3.56)$$

it is straightforward to show that at $x = x_n + \sigma h$:

$$\begin{aligned} p(x) = y_n &+ \sigma h k_1 + 4\sigma^2 \left[h \sum_{i=1}^6 b_{1/2i} k_i - \frac{1}{2} h k_1 \right] \\ &+ 4\sigma^2 \left(\sigma - \frac{1}{2} \right) \left[h k_7 - 4h \sum_{i=1}^6 b_{1/2i} k_i + h k_1 \right] \\ &+ 4\sigma^2 \left(\sigma - \frac{1}{2} \right)^2 \left[h \sum_{i=1}^9 b_{1i} k_i - 2h k_7 - h k_1 + 4h \sum_{i=1}^6 b_{1/2i} k_i \right] \\ &+ 4\sigma^2 \left(\sigma - \frac{1}{2} \right)^2 (\sigma - 1) \left[h k_{10} + 4h k_7 + h k_1 - 6h \sum_{i=1}^9 b_{1i} k_i \right]. \end{aligned} \quad (3.57)$$

Denoting the 2-norm of the local error coefficients of (3.57) by $T_6(\sigma)$ it is instructive to plot the quantity:

$$R = \frac{T_6(\sigma)}{T_6(\frac{1}{2})} \quad (3.58)$$

[r]

Figure 3.7: Interpolation error for order 5

for σ in the range $[0, 1]$. For further discussion of this see [Cash86]. Recall that the interpolant is of order 5 while the global error in the solution is also of order 5. However the integration is advanced at order 6 and the stepsize of integration is chosen for the order 4 formula. The plot of (3.58) for $\sigma \in [0, 1]$ is given in Figure 3.7.

We have incorporated the block 5(4) formula described in this section in a FORTRAN program and we give the results obtained in the next chapter.

3.8 Order 6

In this section we describe the derivation of an explicit block 6(5) Runge-Kutta formula. We will base our block formula on a standard 6(5) pair requiring eight function evaluations per step. Such methods have been investigated in detail by Verner and Sharp [**Verner91b**] and they have found one of the most efficient 6(5) formulae to be that given in the next table.

3. Block Explicit Runge-Kutta Methods

0	0								
$\frac{1}{12}$	$\frac{1}{12}$	0							
$\frac{2}{15}$	$\frac{2}{75}$	$\frac{8}{75}$	0						
$\frac{1}{5}$	$\frac{1}{20}$	0	$\frac{3}{20}$	0					
$\frac{8}{15}$	$\frac{88}{135}$	0	$-\frac{112}{45}$	$\frac{64}{27}$	0				
$\frac{13}{19}$	$-\frac{408551}{260642}$	0	$\frac{3426735}{521284}$	$-\frac{650026}{130321}$	$\frac{347139}{521284}$	0			
$\frac{19}{20}$	$\frac{1296313}{565760}$	0	$-\frac{48507}{5120}$	$\frac{3310503}{400384}$	$-\frac{761805}{748544}$	$\frac{197436315}{223814656}$	0		
1	$\frac{103039}{16796}$	0	$-\frac{105}{4}$	$\frac{8856}{391}$	$-\frac{13797}{3655}$	$\frac{53582508}{22075469}$	$-\frac{1792}{9595}$	0	
	$\frac{1385}{23712}$	0	0	$\frac{515}{1656}$	$\frac{2511}{9632}$	$\frac{17332693}{93496104}$	$\frac{4352}{17271}$	$-\frac{17}{252}$	
	$\frac{1249}{23712}$	0	0	$\frac{61}{184}$	$\frac{1269}{6880}$	$\frac{8731507}{31165368}$	$\frac{4352}{28785}$	0	0

(3.59)

Using the same approach as before we will consider going two steps forward with the standard formula (3.59) but using the one-step formulation to analyse it. The method uses sixteen function evaluations per step. The associated stability polynomial is: $\frac{y_{n+1}}{y_n} = 1 + z + \frac{1}{2}z^2 + \frac{1}{6}z^3 + \frac{1}{24}z^4 + \frac{1}{120}z^5 + \frac{1}{720}z^6 + \frac{2171}{10944000}z^7 + \frac{347}{14008320}z^8 + \frac{2873}{1050624000}z^9 + \frac{281}{1050624000}z^{10} + \frac{97}{4202496000}z^{11} + \frac{1759}{100859904000}z^{12} + \frac{23}{201719808000}z^{13} + \frac{12089}{1916338176000000}z^{14} + \frac{67}{239542272000000}z^{15} + \frac{1}{119771136000000}z^{16}$. Using this polynomial it is straightforward to show that the interval of absolute stability of this formula is $(-8.73, 0)$. The norm of the coefficients of the local truncation error of the embedded formula is: $\|\tilde{T}^{(6)}\| \cong 0.149 \times 10^{-2}$.

3.8.1 Block Formula of Order 6 with 12 Stages

Considering (3.59) as the formula to compute the solution at the first point of the block we now add on an extra row such that:

$$a_{9i} = b_i, \quad 1 \leq i \leq 8. \quad (3.60)$$

This ensures that $k_9 \equiv y'_{n+1/2}$ and it allows us to derive a Hermite polynomial through the data $(y_n, y_{n+1/2}, y_{n+1}, y'_n, y'_{n+1/2}, y'_{n+1})$ for use as a fifth order

3. Block Explicit Runge-Kutta Methods

interpolating polynomial. The question now remains as to how many extra functions are needed to complete a block 6(5) formula. We have tried deriving formulae with both 10 and 11 function evaluations and found the order conditions to be inconsistent. However a block formula is possible with 12 function evaluations and in what follows we explain how such a formula can be derived. The order conditions for an order 6 formula are well known and can be found in Table 2.1. In order to be able to solve these equations we have found it necessary to make the usual simplifying assumptions. It turns out that, after these simplifying assumptions have been made, we still have sufficient free parameters to make the second formula in the block (i.e. the one used to compute the solution at x_{n+1}) of order 7. In this case it automatically has better asymptotic accuracy than the conventional one. In terms of stability we will look for an interval of absolute stability that is at least as good as: $\frac{12}{16}(-8.73, 0) \cong (-6.55, 0)$. Thus as a direct extension of the approach considered earlier, the block formula will be 6(5) at the first step and 7(5) at the second step with a free fifth order interpolating polynomial. To derive our formula of order 7 we need to solve a total of 78 nonlinear algebraic equations. In what follows we will describe a simple general procedure which allows us to solve these equations.

1) Make the simplifying assumptions:

$$\begin{aligned}\hat{b}_2 &= \hat{b}_3 = 0 \\ a_{i2} &= 0, \quad i = 10, 11, 12 \\ \sum_i \hat{b}_i c_i a_{i3} &= 0 \\ \sum_i \hat{b}_i a_{ij} &= \hat{b}_j (1 - c_j), \quad 1 \leq j \leq 12.\end{aligned}$$

2) Set $c_{12} = 1$. Choose $\hat{b}_4, \hat{b}_5, \hat{b}_9, c_{10}, c_{11}$ arbitrarily.

3) Solve the linear equations $\sum_{i=1}^6 \hat{b}_i c_i^j = \frac{1}{j+1}$, $j = 1, 2, 3, 4, 5, 6$ for $\hat{b}_6, \hat{b}_7, \hat{b}_8,$

$$\hat{b}_{10}, \hat{b}_{11}, \hat{b}_{12}.$$

$$\text{Set } \hat{b}_1 = 1 - \sum_{i=2}^{12} \hat{b}_i.$$

4) Choose $a_{10,3}, a_{11,3}, a_{12,3}$ to satisfy: $\sum_i \hat{b}_i a_{i3} = 0, \sum_i \hat{b}_i c_i a_{i3} = 0, \sum_i \hat{b}_i c_i^2 a_{i3} = 0.$

0.

5) Choose $a_{10,8}$ and $a_{10,9}$ arbitrarily and solve:

3. Block Explicit Runge-Kutta Methods

$$\begin{aligned}\sum_i a_{10,i}c_i &= \frac{1}{2}c_i^2 \\ \sum_i a_{10,i}c_i^2 &= \frac{1}{3}c_i^3 \\ \sum_i \hat{b}_i(1-c_i)(c_i-c_{11})a_{ij}c_j^3 &= \frac{1}{168} - \frac{c_{11}}{120} \\ \sum_{ij} \hat{b}_i(1-c_i)(c_i-c_{11})a_{ij}a_{j3} &= 0\end{aligned}$$

for $a_{10,4}$, $a_{10,5}$, $a_{10,6}$, $a_{10,7}$.

Set $a_{10,1} = c_{10} - \sum_{i=3}^9 a_{10,i}$.

6) Choose $a_{11,8}$ arbitrarily.

Solve

$$\begin{aligned}\sum_i a_{11,i}c_i &= \frac{1}{2}c_{11}^2 \\ \sum_i a_{11,i}c_i^2 &= \frac{1}{3}c_{11}^3 \\ \sum_i a_{11,i}c_i^3 &= \frac{1}{b_{11}(c_{11}-1)} \left(\frac{1}{24} - \hat{b}_i(1-c_i)c_i^3 - \sum_{j=1}^{10} \hat{b}_j(c_j-1)a_{ji}c_i^3 \right) \\ \sum_{ij} \hat{b}_i(1-c_i)a_{ij}a_{j3} &= 0 \\ \sum_{ij} \hat{b}_i(1-c_i)a_{ij}c_j^4 &= \frac{1}{210} \\ \sum_{ijk} \hat{b}_i(1-c_i)a_{ij}a_{jk}c_k^3 &= \frac{1}{840}\end{aligned}$$

for $a_{11,10}$, $a_{11,9}$, $a_{11,7}$, $a_{11,6}$, $a_{11,5}$, $a_{11,4}$.

Set $a_{11,1} = c_{11} - \sum_{i=3}^{10} a_{11,i}$.

7) Set $a_{12,j} = [\hat{b}_j(1-c_j) - \sum_{k=j+1}^{11} \hat{b}_k a_{kj}] / \hat{b}_{12}$, $j = 4, 5, \dots, 10$

$$a_{12,11} = \hat{b}_{11}(1-c_{11}) / \hat{b}_{12}$$

$$a_{12,1} = c_{12} - \sum_{j=3}^{11} a_{12,j}$$

This defines all the coefficients of our main formula of order 7. It is important to note that all the equations which need to be solved are *linear* so it is simple to construct a computer program to carry out this general procedure.

3. Block Explicit Runge-Kutta Methods

Having obtained the main formula we can derive an embedded formula of order 5 in the following way:

1) Choose $\bar{b}_4, \bar{b}_9, \bar{b}_{12}$ arbitrarily.

Set $\bar{b}_2 = \bar{b}_3 = 0$.

2) Solve:

$$\begin{aligned}\sum_i \bar{b}_i c_i^j &= \frac{1}{j+1}, \quad j = 1, 2, 3, 4 \\ \sum_i \bar{b}_i a_{i3} &= 0 \\ \sum_{ij} \bar{b}_i a_{ij} c_j^3 &= \frac{1}{20}\end{aligned}$$

for $\bar{b}_5, \bar{b}_6, \bar{b}_7, \bar{b}_8, \bar{b}_{10}, \bar{b}_{11}$.

$$\text{Put } \bar{b}_1 = 1 - \sum_{i=2}^{12} \bar{b}_i.$$

Using this approach we have developed the block formula (3.61), where

$$\begin{aligned}a_{10,1} &= -\frac{15514400620094897541}{146323163457536000000}, \quad a_{10,3} = \frac{14894129938336353}{296200735744000000}, \quad a_{10,4} = -\frac{1115465796694125137}{5109462691584000000}, \\ a_{10,5} &= \frac{2570129433088854921}{127366316369920000000}, \quad a_{10,6} = \frac{4715356027351248054167}{96158384701380352000000}, \quad a_{10,7} = -\frac{261974217902055743}{8326306814835000000}, \\ a_{11,1} &= \frac{45043408253882515066518381}{18347755643649694995200000}, \quad a_{11,3} = -\frac{27917699597648811}{13580628435200000}, \quad a_{11,4} = -\frac{1506088107154654995594000251}{29898670633882600144000000}, \\ a_{11,5} &= -\frac{8259724559381291201457887499}{2445516266521375718300000000}, \quad a_{11,6} = -\frac{32220126226752270243394813467141}{4726537326891457620284268800000}, \\ a_{11,7} &= -\frac{245211708686956024569238294}{60903053823901059014453125}, \quad a_{11,9} = \frac{49383719169866734171599}{9099595410312095696000}, \quad a_{11,10} = \frac{39388790671769555952}{2843623565722529905}, \\ a_{12,1} &= -\frac{1019761775615731879569301491872119}{74627838689488457066330149783296}, \quad a_{12,3} = \frac{1354611699555}{185033261984}, \\ a_{12,4} &= \frac{6975021330674121332266184865803}{193031971419054713509400972224}, \quad a_{12,5} = \frac{201256172007798122954183274296301}{10104804069067038046318207415552}, \\ a_{12,6} &= \frac{63908462135618415595781597790625}{1588918512078788463336823077824}, \quad a_{12,7} = \frac{5234832269273922385292285155580}{251649696435574725016671512023}, \\ a_{12,8} &= -\frac{3988339351014871459225909175}{2098173602381029494667401872}, \quad a_{12,9} = -\frac{896812812789916578125}{33651744427453381544}, \quad a_{12,10} = -\frac{1229024787195280000000}{149630077900640928651}, \\ a_{12,11} &= \frac{206630455251489062500}{215773357006517336541}, \quad \hat{b}_1 = \frac{15570496384}{257777690625}, \quad \hat{b}_6 = -\frac{11242116232463771}{41967407100937500}, \\ \hat{b}_7 &= -\frac{54840487616}{194961524625}, \quad \hat{b}_8 = \frac{408061607}{11965275000}, \quad \hat{b}_{10} = \frac{592401471488000}{1290745082732553}, \quad \hat{b}_{11} = \frac{16975785544000}{68176788371811}, \\ \hat{b}_{12} &= \frac{11564578874}{306736171875}, \quad \bar{b}_1 = -\frac{835201624659198460204559}{34713141956439124815000000}, \\ \bar{b}_5 &= \frac{1017751370513896071}{6514327279724200000}, \quad \bar{b}_6 = \frac{49794680976565711400765612263}{383704456051829135363595000000}, \\ \bar{b}_7 &= -\frac{37330322369529825525437}{281294170179680815395000}, \quad \bar{b}_8 = \frac{7028842195201371181033}{201410122330496065000000}, \\ \bar{b}_{10} &= -\frac{2020332036756821187243464}{8846484592512498482993925}, \quad \bar{b}_{11} = \frac{10579467130236170324548567}{39572833674995279521619400}.\end{aligned}$$

If we denote by $\|\bar{T}^{(6)}\|$ the 2-norm of the sixth order error coefficients of the fifth order embedded formula, and similarly for $\|\bar{T}^{(7)}\|$, we have:

$$\|\bar{T}^{(6)}\| \cong 0.3848 \times 10^{-4} \quad (3.62)$$

$$\|\bar{T}^{(7)}\| \cong 0.8927 \times 10^{-4}. \quad (3.63)$$

3. Block Explicit Runge-Kutta Methods

If we compare the norms of the error coefficients of the conventional embedded methods and for the block one taking into account the number of function evaluations we can see that the block method is better. That is from (3.17)

$$0.3848 \times 10^{-4} < \frac{1}{32} 0.149 \times 10^{-2}.$$

The stability polynomial of (3.61) is given by: $\frac{y_{n+1}}{y_n} = 1 + z + \frac{1}{2}z^2 + \frac{1}{6}z^3 + \frac{1}{24}z^4 + \frac{1}{120}z^5 + \frac{1}{720}z^6 + \frac{1}{5040}z^7 + t_8z^8 + t_9z^9 + t_{10}z^{10} + t_{11}z^{11} + t_{12}z^{12}$ where

$$t_8 = \frac{2946653115863302012161227347}{130251653287964194390508160000000}, t_9 = \frac{575019705456366763941826487}{271357611016592071646892000000000},$$

$$t_{10} = \frac{2416954365804905612083830349}{13025165328796419439050816000000000}, t_{11} = \frac{287254211438861}{23874888645878860800000}, t_{12} = \frac{3760360903}{10971915737995800000}.$$

Similarly the stability polynomial of the embedded formula is: $\frac{y_{n+1}}{y_n} = 1 + z +$

$$\frac{1}{2}z^2 + \frac{1}{6}z^3 + \frac{1}{24}z^4 + \frac{1}{120}z^5 + w_6z^6 + w_7z^7 + w_8z^8 + w_9z^9 + w_{10}z^{10} + w_{11}z^{11} + w_{12}z^{12}$$

where

$$w_6 = \frac{7009720887340857621639192551}{5036131621243879342938393600000},$$

$$w_7 = \frac{17179258985771627442908701698103665042884443}{87700454598755911235606176220951178240000000000},$$

$$w_8 = \frac{47822392642194935658788261616851110298714111}{2192511364968897780890154405523779456000000000000},$$

$$w_9 = \frac{17952852596111441688929459321041236101031331}{8770045459875591123560617622095117824000000000000},$$

$$w_{10} = \frac{277969671489666781486198267172254921}{5752214566458178937693071615797248000000000},$$

$$w_{11} = \frac{21794484201703114763191297129}{2009416516876307857832419046400000000},$$

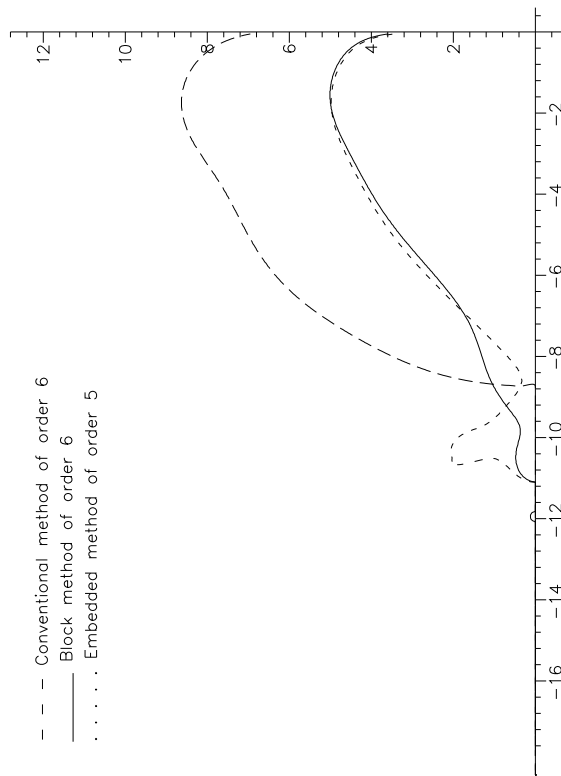
$$w_{12} = \frac{301306438074681}{1023013030594582798937600}.$$

The regions of absolute stability of the three formulae are given in Figure 3.8. Again we see that the block formula has superior accuracy and stability to the conventional one.

We use the FSAL technique, i.e.

$$a_{9i} = b_i, \quad 1 \leq i \leq 8, \quad (3.64)$$

to allow us to construct a quintic Hermite interpolating polynomial without the need to compute extra function evaluations. We have available the following data (x_n, y_n, k_1) , $(x_{n+1/2}, y_{n+1/2}, k_9)$ and $(x_{n+1}, y_{n+1}, k_{13})$, where $k_1 = y'_n$, $k_9 = y'_{n+1/2}$ and $k_{13} = y'_{n+1}$, with $y'_{n+i} \equiv f(x_{n+i}, y_{n+i})$, $i = 0, \frac{1}{2}, 1$. Note that k_{13} is needed to start the integration at the next block (i.e. k_{13} of the current block is k_1 of the next block (FSAL)). The function k_{13} is only computed if the block is successful, and of course, interpolation is only carried out after a successful block. We now examine the quality of our interpolating polynomial. We wish



[r]

Figure 3.8: Stability region of order 6 with 12 stages

3. Block Explicit Runge-Kutta Methods

to fit an interpolating polynomial of the form:

$$p(x) = \sum_{i=0}^5 a_i x^i \quad (3.65)$$

such that at $x = x_{n+i}$, $i = 0, \frac{1}{2}, 1$, we have:

$$\begin{aligned} p(x) &= y_{n+i} \\ p'(x) &= f(x_{n+i}, y_{n+i}). \end{aligned}$$

Denoting the formulae to compute $y_{n+1/2}$, y_{n+1} by:

$$y_{n+1/2} = y_n + h \sum_{i=1}^8 b_{1/2i} k_i \quad (3.66)$$

$$y_{n+1} = y_n + h \sum_{i=1}^{12} b_{1i} k_i \quad (3.67)$$

it is straightforward to show that at $x = x_n + \sigma h$:

$$\begin{aligned} p(x) = y_n &+ \sigma h k_1 + 4\sigma^2 \left[h \sum_{i=1}^8 b_{1/2i} k_i - \frac{1}{2} h k_1 \right] \\ &+ 4\sigma^2 \left(\sigma - \frac{1}{2} \right) \left[h k_9 - 4h \sum_{i=1}^8 b_{1/2i} k_i + h k_1 \right] \\ &+ 4\sigma^2 \left(\sigma - \frac{1}{2} \right)^2 \left[h \sum_{i=1}^{12} b_{1i} k_i - 2h k_9 - h k_1 + 4h \sum_{i=1}^8 b_{1/2i} k_i \right] \\ &+ 4\sigma^2 \left(\sigma - \frac{1}{2} \right)^2 \left(\sigma - 1 \right) \left[h k_{13} + 4h k_9 + h k_1 - 6h \sum_{i=1}^{12} b_{1i} k_i \right]. \end{aligned} \quad (3.68)$$

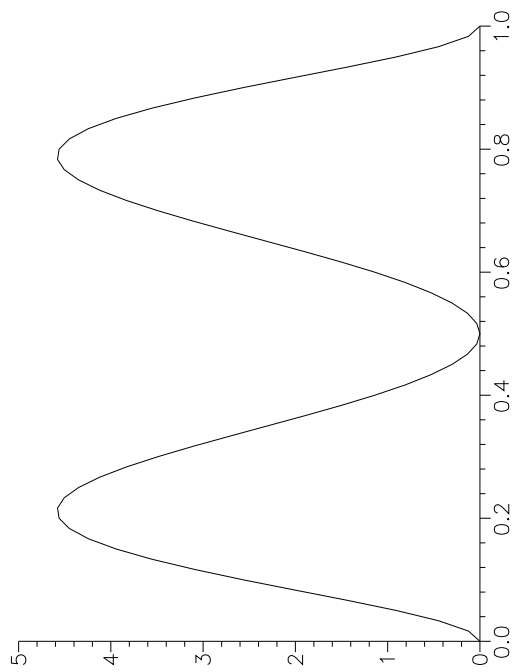
Denoting the 2-norm of the local error coefficients of (3.68) by $\hat{T}_6(\sigma)$ and TE the error in the embedded order 5 formula, it is instructive to plot the quantity:

$$R = \frac{\hat{T}_6(\sigma)}{TE} \quad (3.69)$$

for σ in the range $[0, 1]$ and $TE = 0.46875 \times 10^{-4}$. Recall that the interpolant is of order 5 while the global error in the solution is also of order 5. The plot of (3.69) for $\sigma \in [0, 1]$ is given in Figure 3.9. By looking at Figure 3.9 we can see that the ratio (3.69) is zero for $\sigma = 1/2$ which can be explained by the fact that the method is of order 7 on the second step.

We have incorporated the block 6(5) formula described in this section in a FORTRAN program and we give the results obtained in the next chapter.

3. Block Explicit Runge-Kutta Methods



[r]

Figure 3.9: Interpolation error for order 6

3.9 Order 7

Finally in this section we consider the derivation of a block 7(6) Runge-Kutta formula. As in the previous sections we will use a standard formula for the first step in the block. Again we use a 7(6) formula suggested by Verner and Sharp [Verner91b] which is given by (3.70).

Using the same approach as before we will consider going two steps forward with the standard formula (3.70) but using the one-step formulation to analyse it. The method uses twenty two function evaluations per step. The interval of absolute stability of this formula is $(-9.2, 0)$. The norm of the coefficients of the local truncation error of the embedded formulae is: $\|\tilde{T}^{(7)}\| \cong 0.277 \times 10^{-3}$.

3.9.1 Block Formula of Order 7 with 16 Stages

Considering (3.70) as the formula to compute the solution at the first point of the block we now add an extra row so that:

$$a_{12,i} = b_i, \quad 1 \leq i \leq 11. \quad (3.71)$$

This will serve as a step towards getting an interpolating polynomial. It is now not appropriate to use a fifth order Hermite polynomial as our interpolant since the embedded formula has order 6. Thus we need to generate one more pair of data of the required order. We are again faced with the problem of determining the minimum number of function evaluations which will allow us to derive a block formula. It was quickly found that the order relations become inconsistent with either 13 or 14 stages and we were not able to find an efficient formula with 15 stages. Hence we sought a block formula with 16 stages. After making the standard simplifying assumptions we found that we were able to make the main formula at the second step of the block of order 8. In this case it automatically has better asymptotic accuracy than the conventional one. In terms of stability we will look for an interval of absolute stability that is at least as good as: $\frac{16}{22}(-9.2, 0) \cong (-6.69, 0)$. This leaves us 200 equations to solve in the unknowns and in what follows we will describe a general approach which will both solve these equations and allow us to derive an interpolating

3. Block Explicit Runge-Kutta Methods

polynomial.

1) Set $c_{16} = 1$, choose c_{13} , c_{14} , c_{15} arbitrarily.

2) Set $\hat{b}_2 = \hat{b}_3 = \hat{b}_4 = \hat{b}_5 = 0$.

Choose $\hat{b}_6, \hat{b}_7, \hat{b}_9, \hat{b}_{11}$ arbitrarily.

Solve $\sum_i \hat{b}_i c_i^j = \frac{1}{j+1}$, $j = 1, 2, \dots, 7$ for $\hat{b}_8, \hat{b}_{10}, \hat{b}_{12}, \hat{b}_{13}, \hat{b}_{14}, \hat{b}_{15}, \hat{b}_{16}$.

Set $\hat{b}_1 = 1 - \sum_{i=2}^{16} \hat{b}_i$.

3) Set $a_{i2} = 0$, $i = 13, 14, 15, 16$ and $a_{14,j} = 0$, $j = 3, 4, 5$.

4) Choose $a_{13,j}, a_{15,j}, a_{16,j}$, $3 \leq j \leq 5$ to satisfy:

$$\begin{aligned} \sum_{ij} \hat{b}_i a_{ij} &= 0 \\ \sum_{ij} \hat{b}_i c_i a_{ij} &= 0 \\ \sum_{ij} \hat{b}_i c_i^2 a_{ij} &= 0. \end{aligned}$$

5) Choose $a_{13,10}, a_{13,11}, a_{13,12}$ arbitrarily.

Define $a_{13,6}, a_{13,7}, a_{13,8}, a_{13,9}$ as the solution of:

$$\begin{aligned} \sum_i a_{13,i} c_i &= \frac{1}{2} c_{13}^2 \\ \sum_i a_{13,i} c_i^2 &= \frac{1}{3} c_{13}^3 \\ \sum_i a_{13,i} c_i^3 &= \frac{1}{4} c_{13}^4 \\ \sum_{ij} \hat{b}_i (c_{15} - c_i) (1 - c_i) a_{ij} c_j^4 &= \frac{c_{15}}{210} - \frac{1}{280} \end{aligned}$$

where in the last equation we use $a_{14,j} c_j^4 = \frac{1}{5} c_{14}^5$.

Set $a_{13,1} = c_3 - \sum_{i=3}^{12} a_{13,i}$.

6) Choose $a_{14,10}, a_{14,11}, a_{14,12}, a_{14,13}$ arbitrarily.

Define $a_{14,6}, a_{14,7}, a_{14,8}, a_{14,9}$ as the solution of: $\sum_i a_{14,i} c_i^j = \frac{c_{14}^{j+1}}{j+1}$,

$j = 1, 2, 3, 4$.

Put $a_{14,1} = c_{14} - \sum_{i=2}^{13} a_{14,i}$.

This gives a solution at $x_n + c_{14}h$ which is fifth order. We use hk_{14} , which

3. Block Explicit Runge-Kutta Methods

is sixth order accurate, as the extra piece of information needed to derive an order 6 interpolating polynomial.

7) Choose $a_{15,11}$ arbitrarily.

Choose $a_{15,6}, a_{15,7}, a_{15,8}, a_{15,9}, a_{15,10}, a_{15,12}, a_{15,13}, a_{15,14}$ to satisfy:

$$\begin{aligned}\sum_i a_{15,i} c_i^j &= \frac{c_{15}^j}{j+1}, \quad j = 1, 2, 3 \\ \sum_{ij} \hat{b}_i (1 - c_i) a_{ij} c_j^4 &= \frac{1}{210} \\ \sum_{ij} \hat{b}_i (1 - c_i) a_{ij} c_j^5 &= \frac{1}{336} \\ \sum_{ij} \hat{b}_i (1 - c_i) a_{ij} a_{j4} &= 0 \\ \sum_{ij} \hat{b}_i (1 - c_i) a_{ij} a_{j5} &= 0 \\ \sum_{ijk} \hat{b}_i (1 - c_i) a_{ij} a_{jk} c_k^4 &= \frac{1}{1680}.\end{aligned}$$

$$\text{Set } a_{15,1} = c_{15} - \sum_{i=2}^{14} a_{15,i}.$$

$$8) \text{ Set } a_{16,i} = [\hat{b}_i (1 - c_i) - \sum_{j=1}^{14} \hat{b}_{j+1} a_{j+1,i}] / \hat{b}_{16} \text{ for } i = 6, 7, \dots, 14.$$

$$\text{Set } a_{16,1} = c_{16} - \sum_{i=2}^{15} a_{16,i}.$$

This gives the main formula of order 8 for use in the second step of the block.

The derivation of a satisfactory embedded method proved elusive and we eventually decided to investigate this problem further using a FSAL approach. In this approach we evaluate $f(x_{n+1}, y_{n+1})$ and use this in the error estimating formula. The process for obtaining an embedded formula is now as follows:

- 1) Choose $\bar{b}_2 = \bar{b}_3 = \bar{b}_4 = \bar{b}_5 = 0$; $\bar{b}_{11}, \bar{b}_{13}, \bar{b}_{16}, \bar{b}_{17}$ are arbitrary.
- 2) Solve:

3. Block Explicit Runge-Kutta Methods

$$\sum_i \bar{b}_i c_i^j = \frac{1}{j+1}, \quad j = 1, 2, 3, 4, 5$$

$$\sum_{ij} \bar{b}_i a_{ij} c_j^4 = \frac{1}{30}$$

$$\sum_{ijk} \bar{b}_i a_{ij} a_{jk} c_k^3 = \frac{1}{120}$$

$$\sum_{ij} \bar{b}_i a_{ij} a_{j3} = 0$$

for $\bar{b}_6, \bar{b}_7, \bar{b}_8, \bar{b}_9, \bar{b}_{10}, \bar{b}_{12}, \bar{b}_{14}, \bar{b}_{15}$.

Put $\bar{b}_1 = 1 - \sum_{i=2}^{17} \bar{b}_i$.

Using this approach we have developed the 7(6) formula given by (3.72), where

$$c_9 = \frac{237}{617}, a_{91} = -\frac{4333962560861949}{57227634421028480}, a_{94} = -\frac{27356602518987831}{22891053768411392}, a_{95} = -\frac{5577622585179759}{14306908605257120},$$

$$a_{96} = \frac{4440533975079489}{22891053768411392}, a_{97} = \frac{2074327658036397}{1788363575657140}, a_{98} = \frac{308892413477097}{447090893914285}, c_{10} = \frac{11}{24},$$

$$a_{10,1} = \frac{1815462473}{351645204480}, a_{10,4} = -\frac{777713915}{1780482048}, a_{10,5} = -\frac{257595283}{1112801280}, a_{10,6} = -\frac{979078639}{27894218752}, a_{10,7} = \frac{545934213}{1214808064},$$

$$a_{10,8} = \frac{8188786287}{16673472512}, a_{10,9} = \frac{1751987622862169}{811027050254336}, c_{11} = \frac{1}{2}, a_{11,1} = \frac{327182831}{686807040}, a_{11,4} = \frac{7216785}{1159168},$$

$$a_{11,5} = \frac{1512153}{724480}, a_{11,6} = \frac{49732231}{54480896}, a_{11,7} = -\frac{3029901427}{605031360}, a_{11,8} = -\frac{9462199113}{2768056960},$$

$$a_{11,9} = -\frac{978946598211850411}{887810363554111872}, a_{11,10} = \frac{18304}{56049}, c_{12} = \frac{1}{2}, a_{12,1} = \frac{79217}{3185280}, a_{12,6} = \frac{17449}{210560},$$

$$a_{12,7} = \frac{29672}{233835}, a_{12,8} = \frac{9178839}{68467840}, a_{12,9} = \frac{2262011668669933529}{47104143493869221760}, a_{12,10} = \frac{144896}{1961715}, a_{12,11} = \frac{283}{30030},$$

$$a_{13,1} = -\frac{16503122408637426677}{109648565258856000000}, a_{13,4} = \frac{24555474755981397}{185060869635200000}, a_{13,5} = \frac{4610947253184729}{23132608704400000},$$

$$a_{13,6} = \frac{145645901889159194799}{43489304364272000000}, a_{13,7} = \frac{103573641759666357783}{160988498702183750000}, a_{13,8} = -\frac{204550745468344851807}{129976345157847500000},$$

$$a_{13,9} = -\frac{1612208699761528921182290701}{644853018980887474933500000}, a_{14,1} = \frac{72164600657}{30336000000}, a_{14,6} = -\frac{92105097257}{6016000000},$$

$$a_{14,7} = -\frac{76919966307}{11135000000}, a_{14,8} = \frac{7437900253617}{575360000000}, a_{14,9} = \frac{17767337197914206641}{2854539666816000000}, c_{15} = \frac{9}{10},$$

$$a_{15,1} = -\frac{18838714840418524990382629657}{2580054938309736648949760000}, a_{15,4} = \frac{289165772537125353}{310125225342400000}, a_{15,5} = \frac{1437011962705341}{38765653167800000},$$

$$a_{15,6} = \frac{123212497129116514623272201073}{3488566803813510428672000000}, a_{15,7} = \frac{135383657408601045877075644539}{6456979946885544983920000000},$$

$$a_{15,8} = -\frac{12273779868834760615374262570809}{333640591130675093125120000000}, a_{15,9} = -\frac{572132403289242553128797419327915332007}{8662708413820027965813008740556800000},$$

$$a_{15,10} = -\frac{388437364400036611380569}{221169448835273653000000}, a_{15,12} = -\frac{677475438499160866047}{128065691276939000000}, a_{15,13} = \frac{143326519697888343}{163924084834481920},$$

$$a_{15,14} = \frac{18086781996351}{35450710388080}, a_{16,4} = -\frac{196238220767379}{114600302474240}, a_{16,5} = -\frac{1963131649839}{14325037809280},$$

$$a_{16,6} = -\frac{288562310516196559164561361}{1015187536746853789779200}, a_{16,7} = -\frac{254918544967545154070726573}{11274049090767503606227200},$$

$$a_{16,8} = \frac{52020562695681700924420855113}{131611984226148262358466560}, a_{16,9} = -\frac{293155315322366424159250797417253432079}{58232429521412295450728661552938188800},$$

$$a_{16,10} = \frac{248662283826691565124769}{42137679167509885621200}, a_{16,11} = -\frac{25409998768951}{21924141346800}, a_{16,12} = \frac{5054663354626003}{709860038346096},$$

$$a_{16,13} = -\frac{322108096988125}{501077674126656}, a_{16,14} = -\frac{1780880625}{499240469}, a_{16,15} = \frac{41762532500}{168389489169},$$

$$\hat{b}_1 = \frac{146385607}{4598415360}, \hat{b}_8 = \frac{68604171399}{328548147200}, \hat{b}_{10} = \frac{434785856}{3143124985}, \hat{b}_{12} = \frac{685403}{7257600},$$

$$\hat{b}_{13} = \frac{239624375}{2013548544}, \hat{b}_{14} = \frac{862373125}{4951425024}, \hat{b}_{15} = \frac{803125625}{8446975488}, \hat{b}_{16} = \frac{1929059}{50319360},$$

$$\bar{b}_1 = \frac{1655185626226806853089092657}{45979086882909209356800000000}, \bar{b}_6 = -\frac{5260181875887495939571}{39975341707791050000000}, \bar{b}_7 = \frac{40780549337809515925271}{370594715008906575000000},$$

$$\bar{b}_8 = \frac{1388745561198239373879212247099}{6597146087828361523744000000000}, \bar{b}_9 = \frac{126558157169211507187885557673943}{595593689498269186530599970000000},$$

3. Block Explicit Runge-Kutta Methods

$$\bar{b}_{10} = \frac{155434837957709352022489693}{7630585229601587484850781250}, \bar{b}_{12} = \frac{965490517645254411979891}{11022987841127064000000000},$$

$$\bar{b}_{14} = \frac{2749813988117521591057411}{15040646449461056286720000}, \bar{b}_{15} = \frac{3049610597201809751405423}{34211826902749245995520000}.$$

Again denoting the 2-norm of the seventh order error coefficients of the embedded formula by $\|\bar{T}^{(7)}\|$ and similarly for $\|\bar{T}^{(8)}\|$, we have:

$$\|\bar{T}^{(7)}\| \cong 0.47099 \times 10^{-5} \quad (3.73)$$

$$\|\bar{T}^{(8)}\| \cong 0.90102 \times 10^{-5}. \quad (3.74)$$

If we compare $\|\bar{T}^{(7)}\|$ and the scaled $\|\tilde{T}^{(7)}\|$ using (3.17) we can see that the block method has slightly worse accuracy, but if we take into the number of function evaluations we can see that the block method is better.

The stability polynomial of (3.72) is given by: $\frac{y_{n+1}}{y_n} = 1 + z + \frac{1}{2}z^2 + \frac{1}{6}z^3 + \frac{1}{24}z^4 + \frac{1}{120}z^5 + \frac{1}{720}z^6 + \frac{1}{5040}z^7 + \frac{1}{40320}z^8 + t_9z^9 + t_{10}z^{10} + t_{11}z^{11} + t_{12}z^{12} +$

$t_{13}z^{13} + t_{14}z^{14} + t_{15}z^{15} + t_{16}z^{16}$, where

$$t_9 = \frac{2956495693948759414382062974899534609}{1149721021792062794912966941978342195200000},$$

$$t_{10} = \frac{57119301224445593017607353058862910481/242426889737869240755934172337147582873600000}{242426889737869240755934172337147582873600000},$$

$$t_{11} = \frac{16149773190975359245258495744507324577}{745019222121256691103602578401965742489600000},$$

$$t_{12} = \frac{10205887914691155087060717819143139491}{7187244260463888078881813109289551868723200000},$$

$$t_{13} = \frac{5684634476904456344926771552140275819}{122183152427886097340990822857922381768294400000},$$

$$t_{14} = \frac{1931191664898246540045349}{934228619082280484138640086138880000},$$

$$t_{15} = \frac{1221376616886944221}{12072818096166774555249463001088},$$

$$t_{16} = \frac{7145326289585}{3610138266186826514876596224}.$$

Similarly the stability polynomial of the embedded formula is given by: $\frac{y_{n+1}}{y_n} =$

$$1 + z + \frac{1}{2}z^2 + \frac{1}{6}z^3 + \frac{1}{24}z^4 + \frac{1}{120}z^5 + \frac{1}{720}z^6 + w_7z^7 + w_8z^8 + w_9z^9 + w_{10}z^{10} +$$

$w_{11}z^{11} + w_{12}z^{12} + w_{13}z^{13} + w_{14}z^{14} + w_{15}z^{15} + w_{16}z^{16} + w_{17}z^{17}$, where

$$w_7 = \frac{68935169103120591433599851320584799694302772669237669}{34700757934925575443228016455776616978453626880000000000} = 0.00019865609054532727845,$$

$$w_8 = \frac{16886297638591549069105857214050102330793898133769950940387}{682258133665947724779977697987180447894219761123328000000000000} = 0.000024750599231198822256,$$

$$w_9 = \frac{315323837589232019036510052162508050860027658938208236891}{122734267431969431971001279002984842478325248032768000000000000} = 2.56915891695865101011 \times 10^{-6},$$

$$w_{10} = \frac{154154179042289646908562606377451119645183441829344548315969}{649509743249982233990538768483795786395297212589408256000000000000} = 2.37339286506990922619 \times 10^{-7},$$

$$w_{11} = \frac{98682922768976836227240459157946582224783263550167827906}{463935530892844452850384834631282704568069437563863040000000000000} = 2.12708266984986987253 \times 10^{-8},$$

$$w_{12} = \frac{903549852585175553871523457953392598548661395252713051377}{649509743249982233990538768483795786395297212589408256000000000000} = 1.39112594071959715537 \times 10^{-9},$$

$$w_{13} = \frac{4382205581722325724160738944379146263811039281977}{84704151973529173783778618160688445509155894067200000000000} = 5.173542830688872555 \times 10^{-11},$$

3. Block Explicit Runge-Kutta Methods

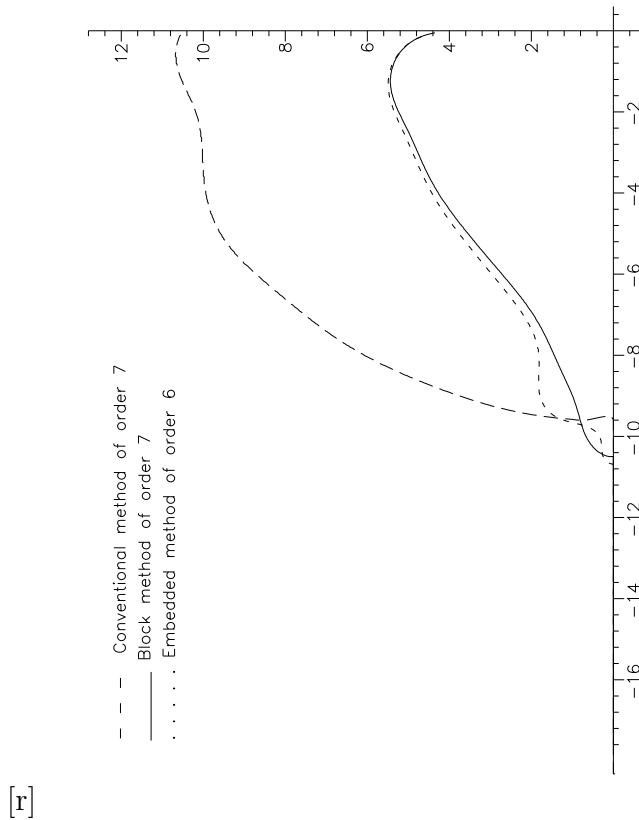


Figure 3.10: Stability region of order 7 with 16 stages

$$w_{14} = \frac{1482797468124650939042960945425092550523190323}{677633215788233390270228945285507564073247152537600000000} = 2.18820068670901572197 \times 10^{-12},$$

$$w_{15} = \frac{66199580176796393237025088880424772349}{679869988048914318377691550486608506058178560000000} = 9.7370940533462053705 \times 10^{-14},$$

$$w_{16} = -\frac{216206040385060736501051}{264649754588333885872442884629725184000} = -8.1695159975255607927 \times 10^{-16},$$

$$w_{17} = -\frac{-1429065257917}{90253456654670662871914905600} = -1.58339116404695081203 \times 10^{-17}.$$

The regions of absolute stability of the three formulae are given in Figure 3.10.

In what follows we will base our interpolating polynomial on the following data (x_n, y_n, y'_n) , $(x_{n+1/2}, y_{n+1/2}, y'_{n+1/2})$, $(x_{n+1}, y_{n+1}, y'_{n+1})$, $(x_{n+c_{14}h}, y'_{n+c_{14}h})$. We need extra data because we want an interpolating polynomial of order 6. When the block method was derived this was taken into account and we pointed out that the solution at $x_{n+c_{14}h}$ is of order 5 and consequently that hk_{14} is of order 6. As an interpolating polynomial for use with this formulae we used the

3. Block Explicit Runge-Kutta Methods

following sixth order polynomial:

$$\begin{aligned}
 p(x) = y_n & + \sigma h k_1 + 4\sigma^2 \left[h \sum_{i=1}^{11} b_{1/2i} k_i - \frac{1}{2} h k_1 \right] \\
 & + 4\sigma^2 \left(\sigma - \frac{1}{2} \right) \left[h k_{12} - 4h \sum_{i=1}^{11} b_{1/2i} k_i + h k_1 \right] \\
 & + 4\sigma^2 \left(\sigma - \frac{1}{2} \right)^2 \left[h \sum_{i=1}^{16} b_{1i} k_i - 2h k_{12} - h k_1 + 4h \sum_{i=1}^{11} b_{1/2i} k_i \right] \\
 & + 4\sigma^2 \left(\sigma - \frac{1}{2} \right)^2 (\sigma - 1) \left[h k_{17} + 4h k_{12} - h k_1 - 6h \sum_{i=1}^{16} b_{1i} k_i \right] \\
 & + \sigma^2 \left(\sigma - \frac{1}{2} \right)^2 (\sigma - 1)^2 [\alpha],
 \end{aligned} \tag{3.75}$$

where α is a complicated expression that can be obtained by the condition $p'(x_{n+c_{14}h}) = y'_{n+c_{14}}$. A theoretical investigation of the polynomial shows that its accuracy is sometimes poor and we feel more work needs to be done to derive a really good interpolating polynomial. This is borne out by the results given in the next chapter.

3. Block Explicit Runge-Kutta Methods

0	0								
$\frac{1}{10}$	$\frac{1}{10}$	0							
$\frac{3}{20}$	$\frac{3}{80}$	$\frac{9}{80}$	0						
$\frac{3}{10}$	$\frac{3}{20}$	$-\frac{9}{20}$	$\frac{3}{5}$	0					
$\frac{1}{3}$	$\frac{113}{729}$	$-\frac{25}{54}$	$\frac{440}{729}$	$\frac{55}{1458}$	0				
$\frac{1}{2}$	$-\frac{181}{540}$	$\frac{5}{4}$	$-\frac{133}{297}$	$-\frac{91}{54}$	$\frac{189}{110}$	0			
$\frac{1}{2}$	$\frac{19}{432}$	0	$\frac{500}{2079}$	$-\frac{125}{432}$	$\frac{81}{176}$	$\frac{5}{112}$	0		
$\frac{39}{50}$	$\frac{5127870379}{11192500000}$	$-\frac{3008889}{11192500}$	$\frac{1}{4}$	$\frac{24241166971}{4477000000}$	$-\frac{671546919267}{111925000000}$	$-\frac{37436178987}{111925000000}$	$\frac{117021996}{69953125}$	0	
1	$\frac{148673568911}{293233374720}$	$\frac{185135}{435116}$	$-\frac{10453369921}{25657920288}$	$-\frac{1251502231253}{58646674944}$	$\frac{2993530119}{116779520}$	$\frac{101178152323}{45614080512}$	$-\frac{1535355}{217558}$	$\frac{108109375}{106024464}$	
	$\frac{47531}{852930}$	0	$\frac{2806400}{16395939}$	$\frac{34025}{367416}$	$\frac{1}{10}$	$\frac{11891}{119070}$	$\frac{2}{21}$	$\frac{31796875}{100304568}$	
	$\frac{48843143}{817903008}$	0	$\frac{37739453}{908344206}$	$\frac{288489667}{251662464}$	$-\frac{824645}{949344}$	$\frac{3}{20}$	$\frac{1}{20}$	$\frac{25492320125}{68703852672}$	

[1]

(3.52)

3. Block Explicit Runge-Kutta Methods

0	0												
$\frac{1}{24}$	$\frac{1}{24}$	0											
$\frac{1}{15}$	$\frac{1}{75}$	$\frac{4}{75}$	0										
$\frac{1}{10}$	$\frac{1}{40}$	0	$\frac{3}{40}$	0									
$\frac{4}{15}$	$\frac{44}{135}$	0	$-\frac{56}{45}$	$\frac{32}{27}$	0								
$\frac{13}{38}$	$-\frac{408551}{521284}$	0	$\frac{3426735}{1042568}$	$-\frac{325013}{130321}$	$\frac{347139}{1042568}$	0							
$\frac{19}{40}$	$\frac{1296313}{1131520}$	0	$-\frac{48507}{10240}$	$\frac{3310503}{800768}$	$-\frac{761805}{1497088}$	$\frac{197436315}{447629312}$	0						
$\frac{1}{2}$	$\frac{103039}{33592}$	0	$-\frac{105}{8}$	$\frac{4428}{391}$	$-\frac{13797}{7310}$	$\frac{26791254}{22075469}$	$-\frac{896}{9595}$	0					
$\frac{1}{2}$	$\frac{1385}{47424}$	0	0	$\frac{515}{3312}$	$\frac{2511}{19264}$	$\frac{17332693}{186992208}$	$\frac{2176}{17271}$	$-\frac{17}{504}$	0				
$\frac{93}{400}$	$a_{10,1}$	0	$a_{10,3}$	$a_{10,4}$	$a_{10,5}$	$a_{10,6}$	$a_{10,7}$	$\frac{7}{800}$	$\frac{3}{400}$	0			
$\frac{171}{200}$	$a_{11,1}$	0	$a_{11,3}$	$a_{11,4}$	$a_{11,5}$	$a_{11,6}$	$a_{11,7}$	$\frac{87}{200}$	$a_{11,9}$	$a_{11,10}$	0		
1	$a_{12,1}$	0	$a_{12,3}$	$a_{12,4}$	$a_{12,5}$	$a_{12,6}$	$a_{12,7}$	$a_{12,8}$	$a_{12,9}$	$a_{12,10}$	$a_{12,11}$	0	
	\hat{b}_1	0	0	$\frac{9}{1000}$	0	\hat{b}_6	\hat{b}_7	\hat{b}_8	$\frac{7}{10}$	\hat{b}_{10}	\hat{b}_{11}	\hat{b}_{12}	
	\bar{b}_1	0	0	$\frac{653}{2000}$	\bar{b}_5	\bar{b}_6	\bar{b}_7	\bar{b}_8	$\frac{219}{500}$	\bar{b}_{10}	\bar{b}_{11}	$\frac{81}{2500}$	

[1]

(3.61)

3. Block Explicit Runge-Kutta Methods

0	0									
$\frac{1}{12}$	$\frac{1}{12}$	0								
$\frac{4}{27}$	$\frac{4}{243}$	$\frac{32}{243}$	0							
$\frac{2}{9}$	$\frac{1}{18}$	0	$\frac{1}{6}$	0						
$\frac{5}{9}$	$\frac{5}{9}$	0	$-\frac{25}{12}$	$\frac{25}{12}$	0					
$\frac{2}{3}$	$\frac{1}{15}$	0	0	$\frac{1}{3}$	$\frac{4}{15}$	0				
$\frac{1}{6}$	$\frac{319}{3840}$	0	0	$\frac{161}{1536}$	$-\frac{41}{960}$	$\frac{11}{512}$	0			
$\frac{4}{9}$	$\frac{245}{5184}$	0	0	$\frac{1627}{10368}$	$\frac{151}{1296}$	$-\frac{445}{10368}$	$\frac{1}{6}$			
$\frac{474}{617}$	$-\frac{4333962560861949}{28613817210514240}$	0	0	$-\frac{27356602518987831}{11445526884205696}$	$-\frac{5577622585179759}{7153454302628560}$	$\frac{4440533975079489}{11445526884205696}$	$\frac{2074327658036397}{894181787828570}$	617	447	
$\frac{11}{12}$	$\frac{1815462473}{175822602240}$	0	0	$-\frac{777713915}{890241024}$	$-\frac{257595283}{556400640}$	$-\frac{979078639}{13947109376}$	$\frac{545934213}{607404032}$	8	8	
1	$\frac{327182831}{343403520}$	0	0	$\frac{7216785}{579584}$	$\frac{1512153}{362240}$	$\frac{49732231}{27240448}$	$-\frac{3029901427}{302515680}$	$-$	$-$	
	$\frac{79217}{1592640}$	0	0	0	0	$\frac{17449}{105280}$	$\frac{59344}{233835}$			
	$\frac{40147}{834240}$	0	0	0	0	$\frac{11521}{45120}$	$\frac{5216}{20043}$			

[1]

(3.70)

Chapter 4

Numerical Results

4.1 Introduction

In this section we present numerical results to compare some of the algorithms derived in the previous sections. The algorithms which will be mainly of interest to us are:

- 1) the block 5(4) formula (3.52)
- 2) the block 5(4) formula derived in [Cash89]
- 3) the block 6(5) formula (3.61). The code to implement this appears in the Appendix
and
- 4) the block 7(6) formula (3.72).

We have written general purpose codes for all of these four formulae and in this section we will examine the performance of these codes. We examine first the interpolation properties of our formulae by considering a simple test problem. Following [Cash89] we consider the numerical integration of

$$y' = -y, \quad y(0) = 1, \quad 0 \leq x \leq 20 \quad (4.1)$$

and ask for output at the points $x = 1, 2, \dots, 20$. Since the steplengths of integration chosen in the various codes will not normally hit these output points exactly this will test the interpolation capabilities of the codes. We ran this problem with error tolerances 10^{-6} and 10^{-10} and the results obtained are given in the following tables.

4. Numerical Results

x	y	dy/dx	Error in y	Error in dy/dx	No. of func. evals.
1.0	3.678795×10^{-1}	-3.678795×10^{-1}	2.62×10^{-8}	-2.40×10^{-8}	37
2.0	1.353353×10^{-1}	-1.353351×10^{-1}	5.16×10^{-8}	2.00×10^{-7}	55
3.0	4.978713×10^{-2}	-4.978697×10^{-2}	6.35×10^{-8}	1.03×10^{-7}	73
4.0	1.831575×10^{-2}	-1.831575×10^{-2}	1.08×10^{-7}	-1.07×10^{-7}	91
5.0	6.738049×10^{-3}	-6.738057×10^{-3}	1.02×10^{-7}	-1.10×10^{-7}	100
6.0	2.478879×10^{-3}	-2.478875×10^{-3}	1.27×10^{-7}	-1.23×10^{-7}	109
7.0	9.122050×10^{-4}	-9.122993×10^{-4}	3.23×10^{-7}	-4.17×10^{-7}	123
8.0	3.364938×10^{-4}	-3.350651×10^{-4}	1.03×10^{-6}	3.98×10^{-7}	123
9.0	1.238228×10^{-4}	-1.236102×10^{-4}	4.13×10^{-7}	-2.00×10^{-7}	132
10.0	4.572067×10^{-5}	-4.563094×10^{-5}	3.21×10^{-7}	-2.31×10^{-7}	141
11.0	1.689637×10^{-5}	-1.630331×10^{-5}	1.95×10^{-7}	3.98×10^{-7}	141
12.0	6.495237×10^{-6}	-6.454500×10^{-6}	3.51×10^{-7}	-3.10×10^{-7}	150
13.0	2.390270×10^{-6}	-2.347949×10^{-6}	1.30×10^{-7}	-8.76×10^{-8}	150
14.0	9.998818×10^{-7}	-9.627178×10^{-7}	1.68×10^{-7}	-1.31×10^{-7}	150
15.0	3.865091×10^{-7}	-3.809986×10^{-7}	8.06×10^{-8}	-7.51×10^{-8}	159
16.0	1.435180×10^{-7}	-9.403426×10^{-8}	3.10×10^{-8}	1.85×10^{-8}	159
17.0	1.768391×10^{-7}	7.471149×10^{-8}	1.35×10^{-7}	1.16×10^{-7}	159
18.0	9.516586×10^{-8}	-9.446951×10^{-8}	7.99×10^{-8}	-7.92×10^{-8}	168
19.0	3.519033×10^{-8}	-3.257032×10^{-8}	2.96×10^{-8}	-2.70×10^{-8}	168
20.0	1.684292×10^{-8}	-1.684292×10^{-8}	1.48×10^{-8}	-1.48×10^{-8}	168

Table 4.1a: Results obtained by (3.52) with Tol = 10^{-6}

x	y	dy/dx	Error in y	Error in dy/dx	No. of func. evals.
1.0	3.678794×10^{-1}	-3.678794×10^{-1}	5.28×10^{-13}	-6.96×10^{-12}	172
2.0	1.353353×10^{-1}	-1.353353×10^{-1}	4.48×10^{-13}	2.69×10^{-12}	316
3.0	4.978707×10^{-2}	-4.978707×10^{-2}	3.20×10^{-13}	-5.04×10^{-13}	460
4.0	1.831564×10^{-2}	-1.831564×10^{-2}	2.39×10^{-13}	4.57×10^{-13}	577
5.0	6.737947×10^{-3}	-6.737947×10^{-3}	2.13×10^{-13}	-1.92×10^{-13}	685
6.0	2.478752×10^{-3}	-2.478752×10^{-3}	2.15×10^{-13}	1.08×10^{-12}	766
7.0	9.118820×10^{-4}	-9.118820×10^{-4}	2.34×10^{-13}	-1.26×10^{-13}	838
8.0	3.354626×10^{-4}	-3.354626×10^{-4}	2.97×10^{-13}	-9.63×10^{-13}	901
9.0	1.234098×10^{-4}	-1.234098×10^{-4}	3.40×10^{-13}	-3.40×10^{-13}	946
10.0	4.539993×10^{-5}	-4.539993×10^{-5}	4.62×10^{-13}	2.42×10^{-12}	982
11.0	1.670170×10^{-5}	-1.670170×10^{-5}	4.94×10^{-13}	-4.69×10^{-13}	1018
12.0	6.144213×10^{-6}	-6.144213×10^{-6}	6.07×10^{-13}	-5.63×10^{-13}	1045
13.0	2.260330×10^{-6}	-2.260326×10^{-6}	8.97×10^{-13}	3.45×10^{-12}	1063
14.0	8.315299×10^{-7}	-8.315246×10^{-7}	1.15×10^{-12}	4.08×10^{-12}	1081
15.0	3.059034×10^{-7}	-3.059034×10^{-7}	1.05×10^{-12}	-1.07×10^{-12}	1099
16.0	1.125371×10^{-7}	-1.125285×10^{-7}	1.93×10^{-12}	6.65×10^{-12}	1108
17.0	4.140283×10^{-8}	-4.139225×10^{-8}	3.45×10^{-12}	7.13×10^{-12}	1117
18.0	1.523356×10^{-8}	-1.521483×10^{-8}	3.58×10^{-12}	1.52×10^{-11}	1126
19.0	5.607641×10^{-9}	-5.610920×10^{-9}	4.84×10^{-12}	-8.12×10^{-12}	1135
20.0	2.075757×10^{-9}	-2.075757×10^{-9}	1.46×10^{-11}	-1.46×10^{-11}	1135

Table 4.1b: Results obtained by (3.52) with Tol = 10^{-10}

4. Numerical Results

x	y	dy/dx	Error in y	Error in dy/dx	No. of func. evals.
1.0	3.678795×10^{-1}	-3.678798×10^{-1}	3.86×10^{-8}	-4.05×10^{-7}	37
2.0	1.353353×10^{-1}	-1.353356×10^{-1}	4.02×10^{-8}	-3.35×10^{-7}	55
3.0	4.978717×10^{-2}	-4.978717×10^{-2}	9.72×10^{-8}	-1.05×10^{-7}	73
4.0	1.831566×10^{-2}	-1.831568×10^{-2}	2.19×10^{-8}	-4.61×10^{-8}	82
5.0	6.738002×10^{-3}	-6.737588×10^{-3}	5.53×10^{-8}	3.59×10^{-7}	91
6.0	2.478839×10^{-3}	-2.478353×10^{-3}	8.67×10^{-8}	3.99×10^{-7}	100
7.0	9.119403×10^{-4}	-9.115701×10^{-4}	5.83×10^{-8}	3.12×10^{-7}	109
8.0	3.354782×10^{-4}	-3.354545×10^{-4}	1.56×10^{-8}	8.15×10^{-9}	118
9.0	1.234819×10^{-4}	-1.236060×10^{-4}	7.21×10^{-8}	-1.96×10^{-7}	127
10.0	4.548547×10^{-5}	-4.513088×10^{-5}	8.55×10^{-8}	2.69×10^{-7}	127
11.0	1.673939×10^{-5}	-1.682914×10^{-5}	3.77×10^{-8}	-1.27×10^{-7}	136
12.0	6.207683×10^{-6}	-5.947948×10^{-6}	6.35×10^{-8}	1.96×10^{-7}	136
13.0	2.332549×10^{-6}	-2.374541×10^{-6}	7.22×10^{-8}	-1.14×10^{-7}	145
14.0	8.302290×10^{-7}	-8.401229×10^{-7}	-1.30×10^{-9}	-8.59×10^{-9}	145
15.0	3.831348×10^{-7}	-1.770379×10^{-7}	7.72×10^{-8}	1.29×10^{-7}	145
16.0	1.789708×10^{-7}	-1.928159×10^{-7}	6.64×10^{-8}	-8.03×10^{-8}	154
17.0	4.811991×10^{-8}	-8.040055×10^{-8}	6.72×10^{-9}	-3.90×10^{-8}	154
18.0	1.130332×10^{-8}	6.470540×10^{-9}	-3.93×10^{-9}	2.17×10^{-8}	154
19.0	5.190446×10^{-8}	6.092684×10^{-8}	4.63×10^{-8}	6.65×10^{-8}	154
20.0	7.225827×10^{-8}	-7.225827×10^{-8}	7.02×10^{-8}	-7.02×10^{-8}	154

Table 4.1c: Results obtained by [Cash89] with Tol = 10^{-6}

x	y	dy/dx	Error in y	Error in dy/dx	No. of func. evals.
1.0	3.678794×10^{-1}	-3.678794×10^{-1}	1.11×10^{-12}	-3.11×10^{-11}	145
2.0	1.353353×10^{-1}	-1.353353×10^{-1}	8.39×10^{-13}	1.91×10^{-11}	262
3.0	4.978707×10^{-2}	-4.978707×10^{-2}	4.60×10^{-13}	3.62×10^{-13}	379
4.0	1.831564×10^{-2}	-1.831564×10^{-2}	3.63×10^{-13}	5.99×10^{-12}	478
5.0	6.737947×10^{-3}	-6.737947×10^{-3}	2.75×10^{-13}	-4.49×10^{-12}	568
6.0	2.478752×10^{-3}	-2.478752×10^{-3}	2.68×10^{-13}	-5.08×10^{-12}	640
7.0	9.118820×10^{-4}	-9.118820×10^{-4}	4.09×10^{-13}	7.91×10^{-12}	694
8.0	3.354626×10^{-4}	-3.354626×10^{-4}	2.37×10^{-13}	-2.51×10^{-12}	748
9.0	1.234098×10^{-4}	-1.234098×10^{-4}	5.81×10^{-13}	7.31×10^{-12}	784
10.0	4.539993×10^{-5}	-4.539993×10^{-5}	2.92×10^{-13}	2.96×10^{-12}	820
11.0	1.670170×10^{-5}	-1.670169×10^{-5}	5.62×10^{-13}	6.66×10^{-12}	847
12.0	6.144213×10^{-6}	-6.144214×10^{-6}	3.08×10^{-13}	-1.48×10^{-12}	874
13.0	2.260330×10^{-6}	-2.260328×10^{-6}	3.31×10^{-13}	1.84×10^{-12}	892
14.0	8.315290×10^{-7}	-8.315298×10^{-7}	3.22×10^{-13}	-1.09×10^{-12}	910
15.0	3.059037×10^{-7}	-3.059045×10^{-7}	1.42×10^{-12}	-2.17×10^{-12}	928
16.0	1.125357×10^{-7}	-1.125381×10^{-7}	5.57×10^{-13}	-2.91×10^{-12}	937
17.0	4.139972×10^{-8}	-4.140079×10^{-8}	3.38×10^{-13}	-1.42×10^{-12}	946
18.0	1.523039×10^{-8}	-1.523193×10^{-8}	4.11×10^{-13}	-1.95×10^{-12}	955
19.0	5.603930×10^{-9}	-5.605182×10^{-9}	1.13×10^{-12}	-2.39×10^{-12}	964
20.0	2.062401×10^{-9}	-2.062401×10^{-9}	1.25×10^{-12}	-1.25×10^{-12}	964

Table 4.1d: Results obtained by [Cash89] with Tol = 10^{-10}

4. Numerical Results

x	y	dy/dx	Error in y	Error in dy/dx	No. of func. evals.
1.0	3.678793×10^{-1}	-3.678795×10^{-1}	-1.75×10^{-7}	-7.08×10^{-8}	37
2.0	1.353351×10^{-1}	-1.353344×10^{-1}	-1.98×10^{-7}	8.69×10^{-7}	49
3.0	4.978699×10^{-2}	-4.978690×10^{-2}	-8.07×10^{-8}	1.71×10^{-7}	61
4.0	1.831558×10^{-2}	-1.831568×10^{-2}	-6.37×10^{-8}	-3.63×10^{-8}	73
5.0	6.737884×10^{-3}	-6.737725×10^{-3}	-6.35×10^{-8}	2.22×10^{-7}	85
6.0	2.478642×10^{-3}	-2.478600×10^{-3}	-1.11×10^{-7}	1.52×10^{-7}	97
7.0	9.117892×10^{-4}	-9.117479×10^{-4}	-9.28×10^{-8}	1.34×10^{-7}	97
8.0	3.354227×10^{-4}	-3.355305×10^{-4}	-3.99×10^{-8}	-6.78×10^{-8}	109
9.0	1.233262×10^{-4}	-1.234051×10^{-4}	-8.36×10^{-8}	4.74×10^{-9}	121
10.0	4.536499×10^{-5}	-4.547018×10^{-5}	-3.49×10^{-8}	-7.02×10^{-8}	121
11.0	1.663181×10^{-5}	-1.667836×10^{-5}	-6.99×10^{-8}	2.33×10^{-8}	133
12.0	6.120112×10^{-6}	-6.101551×10^{-6}	-2.41×10^{-8}	4.27×10^{-8}	133
13.0	2.204480×10^{-6}	-2.333517×10^{-6}	-5.58×10^{-8}	-7.32×10^{-8}	133
14.0	7.846551×10^{-7}	-8.012427×10^{-7}	-4.69×10^{-8}	3.03×10^{-8}	145
15.0	2.883335×10^{-7}	-2.790063×10^{-7}	-1.76×10^{-8}	2.69×10^{-8}	145
16.0	1.016587×10^{-7}	-1.320843×10^{-7}	-1.09×10^{-8}	-1.95×10^{-8}	145
17.0	-2.905544×10^{-9}	-6.673656×10^{-8}	-4.43×10^{-8}	-2.53×10^{-8}	145
18.0	-9.989863×10^{-9}	9.998449×10^{-9}	-2.52×10^{-8}	2.52×10^{-8}	157
19.0	-3.675084×10^{-9}	3.701829×10^{-9}	-9.28×10^{-9}	9.30×10^{-9}	157
20.0	-1.324905×10^{-9}	1.324905×10^{-9}	-3.39×10^{-9}	3.39×10^{-9}	157

Table 4.1e: Results obtained by (3.61) with Tol = 10^{-6}

x	y	dy/dx	Error in y	Error in dy/dx	No. of func. evals.
1.0	3.678794×10^{-1}	-3.678794×10^{-1}	-2.84×10^{-11}	-1.18×10^{-9}	85
2.0	1.353353×10^{-1}	-1.353353×10^{-1}	-3.89×10^{-11}	5.49×10^{-10}	145
3.0	4.978707×10^{-2}	-4.978707×10^{-2}	-3.75×10^{-11}	3.96×10^{-10}	205
4.0	1.831564×10^{-2}	-1.831564×10^{-2}	-1.35×10^{-11}	-7.34×10^{-10}	253
5.0	6.737947×10^{-3}	-6.737947×10^{-3}	-2.71×10^{-11}	4.39×10^{-10}	289
6.0	2.478752×10^{-3}	-2.478753×10^{-3}	-2.53×10^{-11}	-4.22×10^{-10}	325
7.0	9.118820×10^{-4}	-9.118820×10^{-4}	-7.22×10^{-12}	-7.38×10^{-11}	349
8.0	3.354626×10^{-4}	-3.354628×10^{-4}	-1.34×10^{-11}	-1.99×10^{-10}	373
9.0	1.234098×10^{-4}	-1.234097×10^{-4}	-1.09×10^{-11}	1.30×10^{-10}	397
10.0	4.539992×10^{-5}	-4.539989×10^{-5}	-1.08×10^{-11}	4.30×10^{-11}	409
11.0	1.670168×10^{-5}	-1.670160×10^{-5}	-1.59×10^{-11}	1.00×10^{-10}	433
12.0	6.144205×10^{-6}	-6.144222×10^{-6}	-7.22×10^{-12}	-9.21×10^{-12}	445
13.0	2.260322×10^{-6}	-2.260352×10^{-6}	-7.44×10^{-12}	-2.29×10^{-11}	457
14.0	8.315214×10^{-7}	-8.315064×10^{-7}	-7.28×10^{-12}	2.23×10^{-11}	469
15.0	3.058877×10^{-7}	-3.058801×10^{-7}	-1.46×10^{-11}	2.22×10^{-11}	481
16.0	1.125223×10^{-7}	-1.125169×10^{-7}	-1.29×10^{-11}	1.83×10^{-11}	481
17.0	4.139414×10^{-8}	-4.140620×10^{-8}	-5.24×10^{-12}	-6.82×10^{-12}	493
18.0	1.521844×10^{-8}	-1.523044×10^{-8}	-1.15×10^{-11}	-4.64×10^{-13}	505
19.0	5.598568×10^{-9}	-5.609182×10^{-9}	-4.23×10^{-12}	-6.39×10^{-12}	505
20.0	2.051466×10^{-9}	-2.051466×10^{-9}	-9.69×10^{-12}	9.69×10^{-12}	517

Table 4.1f: Results obtained by (3.61) with Tol = 10^{-10}

4. Numerical Results

x	y	dy/dx	Error in y	Error in dy/dx	No. of func. evals.
1.0	3.678743×10^{-1}	-3.679115×10^{-1}	-5.11×10^{-6}	-3.20×10^{-5}	33
2.0	1.353352×10^{-1}	-1.353403×10^{-1}	-8.51×10^{-8}	-5.03×10^{-6}	49
3.0	4.978691×10^{-2}	-4.977954×10^{-2}	-1.59×10^{-7}	7.52×10^{-6}	65
4.0	1.831097×10^{-2}	-1.831538×10^{-2}	-4.67×10^{-6}	2.54×10^{-7}	81
5.0	6.735659×10^{-3}	-6.713698×10^{-3}	-2.29×10^{-6}	2.42×10^{-5}	81
6.0	2.478795×10^{-3}	-2.480126×10^{-3}	4.31×10^{-8}	-1.37×10^{-6}	97
7.0	9.092017×10^{-4}	-9.383050×10^{-4}	-2.68×10^{-6}	-2.64×10^{-5}	113
8.0	3.349487×10^{-4}	-3.451567×10^{-4}	-5.14×10^{-7}	-9.69×10^{-6}	113
9.0	1.230634×10^{-4}	-1.360633×10^{-4}	-3.46×10^{-7}	-1.27×10^{-5}	129
10.0	4.346781×10^{-5}	-3.186505×10^{-5}	-1.93×10^{-6}	1.35×10^{-5}	129
11.0	8.874360×10^{-6}	-1.930213×10^{-5}	-7.83×10^{-6}	-2.60×10^{-6}	129
12.0	1.242496×10^{-6}	-2.060702×10^{-5}	-4.90×10^{-6}	-1.45×10^{-5}	145
13.0	1.796438×10^{-8}	8.051153×10^{-6}	-2.24×10^{-6}	1.03×10^{-5}	145
14.0	-3.797029×10^{-6}	-1.219377×10^{-5}	-4.63×10^{-6}	-1.14×10^{-5}	145
15.0	-1.301559×10^{-6}	1.422742×10^{-5}	-1.61×10^{-6}	1.45×10^{-5}	145
16.0	-2.962715×10^{-6}	-2.368309×10^{-6}	-3.08×10^{-6}	-2.26×10^{-6}	161
17.0	-1.163415×10^{-6}	3.496337×10^{-6}	-1.20×10^{-6}	3.54×10^{-6}	161
18.0	-5.821986×10^{-7}	-2.815079×10^{-6}	-5.97×10^{-7}	-2.80×10^{-6}	161
19.0	-3.314375×10^{-6}	1.785070×10^{-7}	-3.32×10^{-6}	1.84×10^{-7}	161
20.0	1.948512×10^{-8}	-1.948512×10^{-8}	1.74×10^{-8}	-1.74×10^{-8}	161

Table 4.1g: Results obtained by (3.72) with Tol = 10^{-6}

x	y	dy/dx	Error in y	Error in dy/dx	No. of func. evals.
1.0	3.678793×10^{-1}	-3.678797×10^{-1}	-1.36×10^{-7}	-2.38×10^{-7}	65
2.0	1.353352×10^{-1}	-1.353355×10^{-1}	-7.27×10^{-8}	-2.47×10^{-7}	97
3.0	4.978703×10^{-2}	-4.978700×10^{-2}	-3.52×10^{-8}	7.32×10^{-8}	129
4.0	1.831562×10^{-2}	-1.831563×10^{-2}	-1.74×10^{-8}	5.30×10^{-9}	161
5.0	6.737940×10^{-3}	-6.737929×10^{-3}	-6.55×10^{-9}	1.84×10^{-8}	177
6.0	2.478749×10^{-3}	-2.478745×10^{-3}	-3.12×10^{-9}	7.58×10^{-9}	193
7.0	9.118808×10^{-4}	-9.118793×10^{-4}	-1.13×10^{-9}	2.70×10^{-9}	225
8.0	3.354621×10^{-4}	-3.354611×10^{-4}	-5.18×10^{-10}	1.57×10^{-9}	241
9.0	1.234096×10^{-4}	-1.234094×10^{-4}	-2.01×10^{-10}	4.53×10^{-10}	257
10.0	4.539982×10^{-5}	-4.539978×10^{-5}	-1.05×10^{-10}	1.49×10^{-10}	257
11.0	1.670165×10^{-5}	-1.670171×10^{-5}	-5.03×10^{-11}	-1.34×10^{-11}	273
12.0	6.144190×10^{-6}	-6.144136×10^{-6}	-2.27×10^{-11}	7.66×10^{-11}	289
13.0	2.260318×10^{-6}	-2.260292×10^{-6}	-1.15×10^{-11}	3.78×10^{-11}	289
14.0	8.315186×10^{-7}	-8.314925×10^{-7}	-1.01×10^{-11}	3.62×10^{-11}	305
15.0	3.058899×10^{-7}	-3.058966×10^{-7}	-1.24×10^{-11}	5.76×10^{-12}	305
16.0	1.125251×10^{-7}	-1.125199×10^{-7}	-1.01×10^{-11}	1.53×10^{-11}	321
17.0	4.139533×10^{-8}	-4.140511×10^{-8}	-4.05×10^{-12}	-5.74×10^{-12}	321
18.0	1.522141×10^{-8}	-1.522961×10^{-8}	-8.57×10^{-12}	3.66×10^{-13}	321
19.0	5.598881×10^{-9}	-5.611355×10^{-9}	-3.92×10^{-12}	-8.56×10^{-12}	337
20.0	2.055029×10^{-9}	-2.055029×10^{-9}	-6.12×10^{-12}	6.12×10^{-12}	337

Table 4.1h: Results obtained by (3.72) with Tol = 10^{-10}

4. Numerical Results

It can be seen from these results that the behaviour of the interpolants associated with the first three methods is very satisfactory. For these methods, the maximum error in y and y' is always at about the level of the requested tolerance. Furthermore, as would be expected, the 6(5) formula is considerably more efficient than the 5(4) codes at a tolerance of 10^{-10} . The performance of the interpolant associated with (3.72) is disappointing as would be expected from the remarks at the end of the previous chapter. Although the results given by the interpolant were often satisfactory, there were occasions particularly when the requested tolerance was 10^{-10} , that unacceptably large errors occurred. This confirms the theoretical analysis of the previous section which showed that the interpolants for the fifth and sixth order formulae were satisfactory whereas further investigations of an interpolant for the order seven formula is required. However if we simply look at the results at $x = 20$ we see that (3.72) performs very well and is easily the most efficient formula at a tolerance of 10^{-10} .

Our second set of numerical results was obtained by running the four codes on the DETEST test set. The DETEST package can be used to test almost any initial value code and reports numerous statistics which allow a preliminary comparison of different codes on a large set of test problems. Although it is very difficult to compare different codes, the performance of a code on the DETEST test set is widely seen as a reasonable indication of its performance. A first version of DETEST, which is used for non-stiff problems, is described in Hull et al. [Hull72]. Further numerical testing was described in [Enright76]. The testing program was further extended to deal with stiff problems by Enright et al. [Enright75]. A final version, which contains options for both the stiff and non-stiff case, was given by Enright and Pryce [Enright87] and this is the version we shall use.

4.2 The Problems

The DETEST package contains five classes of test problems which are labeled A-E as follows:

- A - linear with real eigenvalues
- B - linear with non real eigenvalues
- C - nonlinear coupling
- D - nonlinear with real eigenvalues
- E - nonlinear with non real eigenvalues

Each of these subclasses contains five problems, making a total of 25 problems in all, and we shall label these A_1, A_2, \dots, A_5 and similarly for the other classes.

4.3 The Output

DETEST contains various options which control the level of detail in the output. In our testing we set the options so as to compare the computational effort required to obtain a given global error. As well as measuring the time taken to achieve a task (efficiency) we also wanted to measure how well the task was achieved (reliability).

One difficulty with the comparison of codes is that most initial value codes attempt to control the local error whereas a user is often interested in the global error. The way in which DETEST deals with this problem is as follows: the user specifies a sequence of local errors (in our experiments we used $10^{-2}, 10^{-3}, \dots, 10^{-9}$) and DETEST runs the problems with these local errors recording the time taken, functions required, number of steps, the global error obtained together with various statistics on the reliability. A least squares fit is then applied to this data and this allows an estimate of the work required to obtain the maximum global error of say $10^{-2}, 10^{-3}, 10^{-4}$ which is output as “Expected Accuracy”. Also given is the predicted local tolerance that should be imposed to obtain this global accuracy and this appears as “Log10 Tol”.

4. Numerical Results

In this way we are able to compare the performance of different codes in their attempts to achieve a specified global accuracy. The cost statistics are the “Time”, that is the total time, in seconds, necessary to solve the problems. The machine used was an IBM RS6000; “FCN Calls” is the number of function evaluations used to solve the problems. “No. of Steps” is the number of steps taken to solve the problems. “Maximum Loc Err” is the maximum local error per step expressed in units of the tolerance; “Fraction deceived” is the fraction of steps on which the true local error per step exceeded the tolerance; “Fraction bad deceived” is the fraction of steps on which the true local error exceeded five times the tolerance. In what follows we give the *efficiency* statistics of the four codes we are comparing. We start with the block 5(4) code based on the formula (3.52).

4. Numerical Results

**Table 4.2: Results on DETEST for block 5(4)
formula (3.52)**

	Expected Accuracy	log10 Tol	Time	FCN Calls	No. of Steps
A1	10^{-3}	-2.80	.002	69	7
	10^{-4}	-3.84	.003	94	9
	10^{-5}	-4.88	.003	120	12
	10^{-6}	-5.92	.005	164	17
	10^{-7}	-6.96	.005	248	26
	10^{-8}	-8.00	.008	361	40
	10^{-9}	-9.04	.014	544	60
	10^{-10}	-10.08	.024	848	93
A2	10^{-5}	-2.93	.002	62	6
	10^{-6}	-3.98	.002	81	8
	10^{-7}	-5.04	.003	119	13
	10^{-8}	-6.09	.005	169	18
	10^{-9}	-7.14	.007	253	28
	10^{-10}	-8.20	.012	392	43
	10^{-11}	-9.25	.017	592	65
A3	10^{-2}	-2.26	.006	175	15
	10^{-3}	-3.04	.008	250	21
	10^{-4}	-3.83	.010	321	30
	10^{-5}	-4.61	.013	482	44
	10^{-6}	-5.40	.018	666	61
	10^{-7}	-6.18	.026	877	83
	10^{-8}	-6.97	.035	1178	114
	10^{-9}	-7.76	.046	1621	164
	10^{-10}	-8.54	.064	2217	227
	10^{-11}	-9.33	.087	3059	317
	A4	10^{-3}	-2.38	.003	51
10^{-4}		-3.32	.002	70	5
10^{-5}		-4.26	.004	98	9
10^{-6}		-5.20	.005	136	13
10^{-7}		-6.14	.007	192	18
10^{-8}		-7.08	.010	260	26
10^{-9}		-8.02	.013	377	39
10^{-10}		-8.96	.020	557	57
10^{-11}		-9.90	.029	801	85
A5		10^{-4}	-2.58	.003	56
	10^{-5}	-3.64	.002	75	8
	10^{-6}	-4.69	.003	112	11
	10^{-7}	-5.74	.005	159	16
	10^{-8}	-6.80	.006	235	25
	10^{-9}	-7.85	.010	359	38
	10^{-10}	-8.91	.015	552	59
	10^{-11}	-9.96	.021	834	91

Table 4.2a: Results for (3.52) on Class A

4. Numerical Results

	Expected Accuracy	log10 Tol	Time	FCN Calls	No. of Steps
B1	10^{-2}	-2.37	.012	362	32
	10^{-3}	-3.46	.017	534	49
	10^{-4}	-4.55	.025	812	78
	10^{-5}	-5.63	.039	1238	120
	10^{-6}	-6.72	.058	1852	188
	10^{-7}	-7.81	.093	2866	304
	10^{-8}	-8.90	.137	4392	471
B2	10^{-3}	-2.66	.006	170	14
	10^{-4}	-3.74	.006	183	16
	10^{-5}	-4.82	.007	194	19
	10^{-6}	-5.90	.010	241	24
	10^{-7}	-6.98	.012	319	34
	10^{-8}	-8.06	.019	482	53
	10^{-9}	-9.14	.029	748	83
B3	10^{-2}	-1.92	.003	61	6
	10^{-3}	-2.81	.003	85	8
	10^{-4}	-3.70	.003	96	10
	10^{-5}	-4.59	.005	126	13
	10^{-6}	-5.48	.007	174	18
	10^{-7}	-6.37	.009	246	26
	10^{-8}	-7.26	.013	351	37
	10^{-9}	-8.15	.018	493	53
	10^{-10}	-9.04	.026	698	75
	10^{-11}	-9.93	.037	1019	112
	B4	10^{-2}	-2.52	.008	205
10^{-3}		-3.49	.013	310	34
10^{-4}		-4.46	.021	474	52
10^{-5}		-5.43	.030	707	78
10^{-6}		-6.41	.044	1069	118
10^{-7}		-7.38	.069	1668	185
10^{-8}		-8.35	.104	2532	281
10^{-9}		-9.32	.160	3821	424
B5		10^{-2}	-2.10	.007	167
	10^{-3}	-2.95	.008	235	21
	10^{-4}	-3.81	.011	279	30
	10^{-5}	-4.66	.015	396	43
	10^{-6}	-5.52	.022	562	62
	10^{-7}	-6.37	.033	811	90
	10^{-8}	-7.23	.048	1194	132
	10^{-9}	-8.09	.066	1735	192
	10^{-10}	-8.94	.093	2436	270
	10^{-11}	-9.80	.145	3631	403

Table 4.2b: Results for (3.52) on Class B

4. Numerical Results

	Expected Accuracy	log10 Tol	Time	FCN Calls	No. of Steps
C1	10^{-3}	-2.68	.008	90	9
	10^{-4}	-3.61	.010	113	12
	10^{-5}	-4.54	.013	151	16
	10^{-6}	-5.48	.018	206	22
	10^{-7}	-6.41	.026	299	33
	10^{-8}	-7.34	.038	446	49
	10^{-9}	-8.27	.057	652	72
	10^{-10}	-9.20	.082	955	106
C2	10^{-2}	-1.92	.038	429	33
	10^{-3}	-2.90	.042	476	35
	10^{-4}	-3.89	.033	366	35
	10^{-5}	-4.88	.039	427	39
	10^{-6}	-5.87	.043	480	41
	10^{-7}	-6.85	.049	534	50
	10^{-8}	-7.84	.061	681	68
	10^{-9}	-8.83	.085	915	95
	10^{-10}	-9.82	.125	1302	140
C3	10^{-2}	-1.97	.027	181	14
	10^{-3}	-2.86	.027	168	16
	10^{-4}	-3.75	.025	194	17
	10^{-5}	-4.65	.026	215	19
	10^{-6}	-5.54	.029	232	22
	10^{-7}	-6.44	.035	276	28
	10^{-8}	-7.33	.047	365	39
	10^{-9}	-8.22	.066	501	55
	10^{-10}	-9.12	.094	720	79
		10^{-11}	-10.01	.134	1058
C4	10^{-2}	-1.99	.117	193	15
	10^{-3}	-2.89	.107	174	16
	10^{-4}	-3.78	.114	182	17
	10^{-5}	-4.67	.124	210	19
	10^{-6}	-5.57	.136	232	22
	10^{-7}	-6.46	.169	278	28
	10^{-8}	-7.36	.224	368	39
	10^{-9}	-8.25	.300	507	56
	10^{-10}	-9.14	.431	730	81
		10^{-11}	-10.04	.625	1068
C5	10^{-2}	-2.11	.024	37	4
	10^{-3}	-3.00	.031	45	4
	10^{-4}	-3.89	.042	62	6
	10^{-5}	-4.78	.054	85	9
	10^{-6}	-5.67	.072	109	12
	10^{-7}	-6.56	.100	153	16
	10^{-8}	-7.45	.147	225	24
	10^{-9}	-8.35	.212	323	35
	10^{-10}	-9.24	.302	459	50

Table 4.2c: Results for (3.52) on Class C

4. Numerical Results

	Expected Accuracy	log10 Tol	Time	FCN Calls	No. of Steps
D1	10^{-1}	-2.35	.008	191	18
	10^{-2}	-3.16	.012	249	25
	10^{-3}	-3.97	.015	329	35
	10^{-4}	-4.78	.023	466	51
	10^{-5}	-5.58	.031	646	71
	10^{-6}	-6.39	.044	916	101
	10^{-7}	-7.20	.063	1315	146
	10^{-8}	-8.01	.089	1869	207
	10^{-9}	-8.81	.121	2610	289
	10^{-10}	-9.62	.178	3781	420
D2	10^{-1}	-2.08	.012	252	20
	10^{-2}	-2.96	.013	293	26
	10^{-3}	-3.83	.019	395	37
	10^{-4}	-4.71	.026	539	54
	10^{-5}	-5.59	.036	753	76
	10^{-6}	-6.47	.050	1054	110
	10^{-7}	-7.35	.070	1505	163
	10^{-8}	-8.23	.101	2191	238
	10^{-9}	-9.11	.151	3196	342
	10^{-10}	-9.99	.224	4733	506
D3	10^{-1}	-2.15	.015	342	27
	10^{-2}	-3.07	.019	425	37
	10^{-3}	-3.99	.027	604	52
	10^{-4}	-4.91	.037	790	76
	10^{-5}	-5.83	.051	1057	108
	10^{-6}	-6.75	.070	1499	162
	10^{-7}	-7.68	.103	2243	247
	10^{-8}	-8.60	.152	3285	364
	10^{-9}	-9.52	.231	4923	546
	D4	10^{-1}	-2.77	.026	555
10^{-2}		-3.69	.035	767	66
10^{-3}		-4.61	.047	1014	93
10^{-4}		-5.54	.065	1413	132
10^{-5}		-6.46	.087	1894	192
10^{-6}		-7.38	.122	2925	288
10^{-7}		-8.31	.181	3878	429
D5	10^0	-2.58	.036	804	69
	10^{-1}	-3.72	.052	1188	102
	10^{-2}	-4.85	.079	1736	156
	10^{-3}	-5.99	.116	2586	236
	10^{-4}	-7.13	.161	3482	380

Table 4.2d: Results for (3.52) on Class D

4. Numerical Results

	Expected Accuracy	log10 Tol	Time	FCN Calls	No. of Steps	
E1	10^{-2}	-2.25	.005	102	11	
	10^{-3}	-3.16	.007	144	15	
	10^{-4}	-4.07	.010	197	21	
	10^{-5}	-4.98	.015	287	31	
	10^{-6}	-5.89	.019	409	45	
	10^{-7}	-6.80	.029	610	67	
	10^{-8}	-7.71	.045	925	102	
	10^{-9}	-8.62	.063	1346	149	
	10^{-10}	-9.53	.095	2007	222	
	E2	10^{-3}	-2.67	.017	462	42
10^{-4}		-3.60	.024	655	62	
10^{-5}		-4.53	.035	961	94	
10^{-6}		-5.46	.051	1418	138	
10^{-7}		-6.39	.071	2025	203	
10^{-8}		-7.32	.105	2962	306	
10^{-9}		-8.25	.160	4411	458	
10^{-10}		-9.18	.231	6305	668	
E3		10^{-3}	-2.20	.015	249	25
		10^{-4}	-3.36	.026	411	43
	10^{-5}	-4.53	.051	683	73	
	10^{-6}	-5.69	.075	1103	119	
	10^{-7}	-6.85	.111	1811	198	
	10^{-8}	-8.01	.184	3038	334	
	10^{-9}	-9.17	.309	4960	548	
	E4	10^{-3}	-2.47	.001	32	3
10^{-4}		-3.32	.002	36	3	
10^{-5}		-4.17	.003	57	5	
10^{-6}		-5.03	.002	72	7	
10^{-7}		-5.88	.003	95	9	
10^{-8}		-6.73	.005	119	12	
10^{-9}		-7.58	.006	163	18	
10^{-10}		-8.44	.008	229	25	
10^{-11}		-9.29	.011	318	35	
E5		10^{-4}	-2.75	.002	59	4
	10^{-5}	-3.67	.004	84	6	
	10^{-6}	-4.58	.005	87	8	
	10^{-7}	-5.50	.005	104	11	
	10^{-8}	-6.42	.006	156	17	
	10^{-9}	-7.33	.009	234	25	
	10^{-10}	-8.25	.016	345	38	
	10^{-11}	-9.17	.021	501	55	
	10^{-12}	-10.08	.035	732	81	

Table 4.2e: Results for (3.52) on Class E

4. Numerical Results

Table 4.3: Results on DETEST for the existing block 5(4) formula in [Cash89]

	Expected Accuracy	log10 Tol	Time	FCN Calls	No. of Steps
A1	10^{-3}	-2.20	.002	56	6
	10^{-4}	-3.20	.002	69	7
	10^{-5}	-4.19	.002	96	10
	10^{-6}	-5.19	.004	124	13
	10^{-7}	-6.18	.005	167	18
	10^{-8}	-7.18	.006	243	26
	10^{-9}	-8.18	.009	350	38
	10^{-10}	-9.17	.013	511	56
A2	10^{-3}	-2.73	.001	43	4
	10^{-4}	-3.56	.002	51	5
	10^{-5}	-4.38	.002	65	7
	10^{-6}	-5.21	.002	87	9
	10^{-7}	-6.04	.003	110	12
	10^{-8}	-6.86	.005	140	15
	10^{-9}	-7.69	.005	188	20
	10^{-10}	-8.52	.007	249	27
A3	10^{-1}	-2.62	.003	152	14
	10^{-2}	-3.54	.006	221	20
	10^{-3}	-4.45	.010	320	29
	10^{-4}	-5.37	.013	460	41
	10^{-5}	-6.28	.017	637	59
	10^{-6}	-7.20	.026	892	86
	10^{-7}	-8.11	.037	1324	128
	10^{-8}	-9.03	.054	1913	183
A4	10^{-3}	-2.78	.001	35	3
	10^{-4}	-3.73	.002	53	5
	10^{-5}	-4.67	.003	78	8
	10^{-6}	-5.62	.003	109	11
	10^{-7}	-6.56	.005	157	16
	10^{-8}	-7.51	.008	230	24
	10^{-9}	-8.45	.012	337	35
	10^{-10}	-9.40	.018	497	52
A5	10^{-3}	-2.39	.001	40	4
	10^{-4}	-3.50	.001	55	6
	10^{-5}	-4.62	.003	75	8
	10^{-6}	-5.74	.003	101	11
	10^{-7}	-6.85	.003	155	17
	10^{-8}	-7.97	.007	232	25
	10^{-9}	-9.08	.010	362	39

Table 4.3a: Results for [Cash89] on Class A

4. Numerical Results

	Expected Accuracy	log10 Tol	Time	FCN Calls	No. of Steps
B1	10^0	-2.04	.008	267	25
	10^{-1}	-2.93	.010	318	28
	10^{-2}	-3.82	.013	438	39
	10^{-3}	-4.72	.019	578	55
	10^{-4}	-5.61	.027	794	76
	10^{-5}	-6.50	.035	1101	110
	10^{-6}	-7.39	.049	1574	163
	10^{-7}	-8.28	.072	2301	240
B2	10^{-2}	-1.92	.005	113	11
	10^{-3}	-2.83	.004	121	12
	10^{-4}	-3.75	.005	136	14
	10^{-5}	-4.66	.006	164	17
	10^{-6}	-5.58	.009	210	21
	10^{-7}	-6.49	.010	261	27
	10^{-8}	-7.41	.013	347	38
	10^{-9}	-8.32	.019	490	54
	10^{-10}	-9.24	.026	687	76
	B3	10^{-2}	-2.52	.002	64
10^{-3}		-3.32	.002	81	8
10^{-4}		-4.12	.004	104	11
10^{-5}		-4.92	.005	133	14
10^{-6}		-5.72	.005	168	18
10^{-7}		-6.52	.008	222	24
10^{-8}		-7.32	.012	299	33
10^{-9}		-8.12	.016	399	44
10^{-10}		-8.92	.021	540	59
10^{-11}		-9.71	.026	760	83
B4		10^{-1}	-2.90	.007	190
	10^{-2}	-3.92	.011	255	27
	10^{-3}	-4.94	.013	363	40
	10^{-4}	-5.96	.021	552	59
	10^{-5}	-6.97	.035	860	92
	10^{-6}	-7.99	.057	1338	143
	10^{-7}	-9.01	.095	2265	225
	10^{-8}	-10.03	.152	3531	354
B5	10^{-1}	-2.49	.005	150	13
	10^{-2}	-3.37	.007	201	18
	10^{-3}	-4.24	.010	258	24
	10^{-4}	-5.11	.014	358	35
	10^{-5}	-5.98	.018	453	46
	10^{-6}	-6.85	.024	620	68
	10^{-7}	-7.72	.034	916	101
	10^{-8}	-8.60	.053	1356	146
	10^{-9}	-9.47	.080	2019	216

Table 4.3b: Results for [Cash89] on Class B

4. Numerical Results

	Expected Accuracy	log10 Tol	Time	FCN Calls	No. of Steps
C1	10^{-2}	-1.92	.007	77	7
	10^{-3}	-2.85	.007	81	8
	10^{-4}	-3.77	.008	102	11
	10^{-5}	-4.69	.013	146	16
	10^{-6}	-5.61	.019	196	21
	10^{-7}	-6.54	.025	274	30
	10^{-8}	-7.46	.036	403	44
	10^{-9}	-8.38	.053	582	64
	10^{-10}	-9.30	.077	841	93
	C2	10^{-2}	-2.60	.024	285
10^{-3}		-3.52	.026	295	30
10^{-4}		-4.45	.027	313	31
10^{-5}		-5.37	.030	333	33
10^{-6}		-6.30	.033	366	39
10^{-7}		-7.22	.044	443	47
10^{-8}		-8.14	.068	602	65
10^{-9}		-9.07	.077	795	86
10^{-10}		-9.99	.100	1137	125
C3		10^{-2}	-2.18	.018	144
	10^{-3}	-3.03	.017	140	14
	10^{-4}	-3.88	.024	161	16
	10^{-5}	-4.73	.025	181	18
	10^{-6}	-5.58	.031	212	21
	10^{-7}	-6.43	.036	251	26
	10^{-8}	-7.27	.044	316	35
	10^{-9}	-8.12	.062	435	48
	10^{-10}	-8.97	.075	580	64
	10^{-11}	-9.82	.100	843	93
C4	10^{-2}	-1.99	.095	145	14
	10^{-3}	-2.87	.085	140	14
	10^{-4}	-3.75	.092	158	16
	10^{-5}	-4.63	.108	179	18
	10^{-6}	-5.51	.125	209	21
	10^{-7}	-6.39	.143	249	26
	10^{-8}	-7.26	.178	315	34
	10^{-9}	-8.14	.252	439	48
	10^{-10}	-9.02	.370	592	65
	10^{-11}	-9.90	.517	869	96
C5	10^{-2}	-2.80	.023	35	3
	10^{-3}	-3.79	.037	51	5
	10^{-4}	-4.79	.046	69	7
	10^{-5}	-5.79	.062	94	10
	10^{-6}	-6.78	.093	142	15
	10^{-7}	-7.78	.145	216	23
	10^{-8}	-8.77	.210	318	35
	10^{-9}	-9.77	.334	487	54

Table 4.3c: Results for [Cash89] on Class C

4. Numerical Results

	Expected Accuracy	log10 Tol	Time	FCN Calls	No. of Steps
D1	10^0	-2.30	.008	161	16
	10^{-1}	-3.17	.008	187	20
	10^{-2}	-4.05	.014	268	29
	10^{-3}	-4.93	.018	387	42
	10^{-4}	-5.80	.026	548	60
	10^{-5}	-6.68	.038	805	89
	10^{-6}	-7.55	.058	1204	133
	10^{-7}	-8.43	.083	1743	193
	10^{-8}	-9.31	.118	2524	280
D2	10^0	-2.44	.009	185	17
	10^{-1}	-3.33	.013	247	24
	10^{-2}	-4.23	.017	344	34
	10^{-3}	-5.12	.023	477	48
	10^{-4}	-6.01	.032	649	68
	10^{-5}	-6.91	.042	918	101
	10^{-6}	-7.80	.065	1386	153
	10^{-7}	-8.70	.095	2009	223
	10^{-8}	-9.59	.142	2999	333
D3	10^1	-1.95	.012	265	20
	10^0	-2.85	.013	307	25
	10^{-1}	-3.74	.016	360	33
	10^{-2}	-4.64	.021	463	45
	10^{-3}	-5.54	.030	656	63
	10^{-4}	-6.44	.043	932	91
	10^{-5}	-7.33	.059	1289	135
	10^{-6}	-8.23	.081	1798	198
	10^{-7}	-9.13	.125	2601	286
D4	10^1	-2.11	.017	360	27
	10^0	-3.08	.017	382	35
	10^{-1}	-4.05	.024	527	47
	10^{-2}	-5.01	.033	709	66
	10^{-3}	-5.98	.044	962	91
	10^{-4}	-6.94	.065	1435	137
	10^{-5}	-7.91	.087	1976	205
	10^{-6}	-8.88	.135	2901	303

Table 4.3d: Results for [Cash89] on Class D

4. Numerical Results

	Expected Accuracy	log10 Tol	Time	FCN Calls	No. of Steps
E1	10^{-2}	-2.72	.005	92	10
	10^{-3}	-3.61	.007	132	14
	10^{-4}	-4.50	.010	194	21
	10^{-5}	-5.38	.014	279	30
	10^{-6}	-6.27	.019	403	44
	10^{-7}	-7.16	.028	589	65
	10^{-8}	-8.05	.039	867	96
	10^{-9}	-8.94	.059	1243	138
	10^{-10}	-9.82	.084	1870	207
	E2	10^{-1}	-2.03	.010	300
10^{-2}		-3.04	.017	482	39
10^{-3}		-4.05	.022	594	55
10^{-4}		-5.06	.029	845	77
10^{-5}		-6.07	.048	1289	112
10^{-6}		-7.08	.062	1707	169
10^{-7}		-8.09	.092	2595	263
10^{-8}		-9.10	.141	3948	397
E3	10^{-2}	-2.82	.012	213	22
	10^{-3}	-3.82	.019	308	33
	10^{-4}	-4.83	.028	453	49
	10^{-5}	-5.84	.044	694	72
	10^{-6}	-6.85	.064	1025	111
	10^{-7}	-7.85	.100	1620	176
	10^{-8}	-8.86	.165	2583	271
	10^{-9}	-9.87	.240	4072	428
E4	10^{-3}	-2.72	.002	28	3
	10^{-4}	-3.62	.002	42	4
	10^{-5}	-4.51	.002	55	5
	10^{-6}	-5.41	.002	72	7
	10^{-7}	-6.30	.003	100	10
	10^{-8}	-7.20	.004	138	14
	10^{-9}	-8.09	.007	193	19
	10^{-10}	-8.99	.010	251	26
	10^{-11}	-9.88	.013	355	38
	E5	10^{-2}	-2.43	.002	35
10^{-3}		-3.37	.002	50	4
10^{-4}		-4.30	.002	60	5
10^{-5}		-5.24	.002	74	6
10^{-6}		-6.17	.005	111	9
10^{-7}		-7.10	.007	137	13
10^{-8}		-8.04	.008	184	20
10^{-9}		-8.97	.012	276	30
10^{-10}		-9.90	.021	434	48

Table 4.3e: Results for [Cash89] on Class E

4. Numerical Results

**Table 4.4: Results on DETEST for block 6(5)
formula (3.61)**

	Expected Accuracy	log10 Tol	Time	FCN Calls	No. of Steps
A1	10^{-3}	-2.11	.002	63	5
	10^{-4}	-3.09	.003	86	7
	10^{-5}	-4.08	.002	99	8
	10^{-6}	-5.06	.003	134	11
	10^{-7}	-6.04	.003	159	13
	10^{-8}	-7.03	.007	218	18
	10^{-9}	-8.01	.007	289	24
	10^{-10}	-8.99	.010	372	30
	10^{-11}	-9.97	.013	513	42
A2	10^{-4}	-1.95	.002	60	4
	10^{-5}	-3.02	.003	73	6
	10^{-6}	-4.09	.003	99	8
	10^{-7}	-5.17	.003	125	10
	10^{-8}	-6.24	.004	156	12
	10^{-9}	-7.31	.006	215	17
	10^{-10}	-8.39	.008	297	24
A3	10^{-2}	-2.55	.007	220	14
	10^{-3}	-3.50	.009	289	19
	10^{-4}	-4.44	.010	356	25
	10^{-5}	-5.38	.011	448	33
	10^{-6}	-6.32	.016	640	45
	10^{-7}	-7.26	.024	868	63
	10^{-8}	-8.20	.034	1184	86
	10^{-9}	-9.14	.045	1604	117
	10^{-10}	-10.08	.061	2129	164
A4	10^{-3}	-2.73	.002	50	3
	10^{-4}	-3.73	.003	67	4
	10^{-5}	-4.72	.003	94	6
	10^{-6}	-5.71	.005	127	8
	10^{-7}	-6.71	.004	168	11
	10^{-8}	-7.70	.007	215	15
	10^{-9}	-8.69	.011	292	21
A5	10^{-3}	-1.92	.002	48	3
	10^{-4}	-2.99	.002	60	4
	10^{-5}	-4.07	.002	85	7
	10^{-6}	-5.14	.003	102	8
	10^{-7}	-6.21	.003	140	11
	10^{-8}	-7.29	.004	189	15
	10^{-9}	-8.36	.007	267	22
	10^{-10}	-9.44	.010	370	30

Table 4.4a: Results for (3.61) on Class A

4. Numerical Results

	Expected Accuracy	log10 Tol	Time	FCN Calls	No. of Steps
B1	10^{-2}	-2.99	.015	522	32
	10^{-3}	-4.10	.020	646	44
	10^{-4}	-5.20	.027	874	62
	10^{-5}	-6.31	.037	1149	83
	10^{-6}	-7.42	.051	1639	121
	10^{-7}	-8.52	.072	2265	172
B2	10^{-3}	-2.57	.004	122	9
	10^{-4}	-3.50	.005	134	10
	10^{-5}	-4.43	.006	152	12
	10^{-6}	-5.35	.007	181	15
	10^{-7}	-6.28	.008	224	18
	10^{-8}	-7.20	.011	296	24
	10^{-9}	-8.13	.016	388	32
	10^{-10}	-9.05	.020	503	41
	10^{-11}	-9.98	.027	680	56
	B3	10^{-3}	-2.20	.003	65
10^{-4}		-3.18	.003	89	7
10^{-5}		-4.17	.004	115	9
10^{-6}		-5.15	.007	152	12
10^{-7}		-6.14	.007	199	15
10^{-8}		-7.12	.010	258	20
10^{-9}		-8.11	.012	343	27
10^{-10}		-9.09	.017	457	37
10^{-11}		-10.08	.024	645	51
B4		10^{-1}	-2.51	.009	199
	10^{-2}	-3.35	.011	262	21
	10^{-3}	-4.18	.014	347	28
	10^{-4}	-5.02	.018	448	37
	10^{-5}	-5.86	.024	579	48
	10^{-6}	-6.70	.034	785	65
	10^{-7}	-7.53	.050	1063	88
	10^{-8}	-8.37	.062	1411	117
	10^{-9}	-9.21	.072	1868	155
	10^{-10}	-10.05	.096	2521	210
B5	10^{-3}	-2.83	.009	254	16
	10^{-4}	-3.79	.014	329	22
	10^{-5}	-4.76	.018	425	30
	10^{-6}	-5.72	.020	545	40
	10^{-7}	-6.68	.027	690	53
	10^{-8}	-7.64	.037	912	74
	10^{-9}	-8.60	.048	1225	102
	10^{-10}	-9.56	.068	1713	142

Table 4.4b: Results for (3.61) on Class B

4. Numerical Results

	Expected Accuracy	log10 Tol	Time	FCN Calls	No. of Steps
C1	10^{-3}	-2.51	.008	91	7
	10^{-4}	-3.47	.010	113	9
	10^{-5}	-4.44	.012	148	12
	10^{-6}	-5.41	.016	188	15
	10^{-7}	-6.37	.022	243	20
	10^{-8}	-7.34	.030	329	27
	10^{-9}	-8.31	.039	449	37
	10^{-10}	-9.27	.053	599	49
C2	10^{-2}	-2.01	.025	302	21
	10^{-3}	-2.94	.027	313	21
	10^{-4}	-3.86	.024	296	20
	10^{-5}	-4.78	.027	307	21
	10^{-6}	-5.70	.031	339	24
	10^{-7}	-6.62	.036	377	28
	10^{-8}	-7.55	.045	450	35
	10^{-9}	-8.47	.055	557	44
	10^{-10}	-9.39	.068	707	57
C3	10^{-3}	-2.19	.017	127	9
	10^{-4}	-3.17	.020	149	11
	10^{-5}	-4.16	.021	166	12
	10^{-6}	-5.14	.025	191	15
	10^{-7}	-6.12	.029	222	17
	10^{-8}	-7.11	.036	275	22
	10^{-9}	-8.09	.046	370	30
	10^{-10}	-9.07	.062	482	40
	10^{-11}	-10.06	.090	671	55
C4	10^{-3}	-2.47	.086	144	10
	10^{-4}	-3.44	.096	160	11
	10^{-5}	-4.40	.108	181	13
	10^{-6}	-5.36	.117	194	15
	10^{-7}	-6.33	.138	224	18
	10^{-8}	-7.29	.180	292	24
	10^{-9}	-8.26	.242	388	32
	10^{-10}	-9.22	.296	511	42
C5	10^{-3}	-2.74	.030	45	3
	10^{-4}	-3.70	.038	57	4
	10^{-5}	-4.66	.051	76	6
	10^{-6}	-5.62	.066	99	8
	10^{-7}	-6.59	.091	137	11
	10^{-8}	-7.55	.125	189	15
	10^{-9}	-8.51	.167	253	21
	10^{-10}	-9.47	.235	351	29

Table 4.4c: Results for (3.61) on Class C

4. Numerical Results

	Expected Accuracy	log10 Tol	Time	FCN Calls	No. of Steps
D1	10^{-1}	-2.27	.008	186	13
	10^{-2}	-3.23	.012	247	18
	10^{-3}	-4.19	.016	300	24
	10^{-4}	-5.16	.020	419	34
	10^{-5}	-6.12	.028	570	47
	10^{-6}	-7.08	.039	821	68
	10^{-7}	-8.04	.054	1170	97
	10^{-8}	-9.00	.075	1598	133
	10^{-9}	-9.96	.121	2314	192
D2	10^{-1}	-2.77	.014	307	19
	10^{-2}	-3.68	.018	373	25
	10^{-3}	-4.60	.022	463	34
	10^{-4}	-5.52	.029	630	45
	10^{-5}	-6.44	.039	850	62
	10^{-6}	-7.36	.052	1100	85
	10^{-7}	-8.28	.068	1473	117
	10^{-8}	-9.20	.098	2043	162
	D3	10^{-1}	-2.74	.020	422
10^{-2}		-3.73	.024	548	34
10^{-3}		-4.71	.031	723	46
10^{-4}		-5.70	.041	949	61
10^{-5}		-6.68	.057	1236	83
10^{-6}		-7.67	.074	1585	115
10^{-7}		-8.65	.104	2185	159
10^{-8}		-9.64	.138	2937	225
D4		10^0	-2.39	.026	547
	10^{-1}	-3.41	.032	683	42
	10^{-2}	-4.42	.040	896	56
	10^{-3}	-5.44	.053	1201	75
	10^{-4}	-6.45	.073	1596	101
	10^{-5}	-7.47	.099	2098	142
	10^{-6}	-8.49	.130	2874	197
D5	10^1	-2.08	.036	822	47
	10^0	-3.07	.042	917	60
	10^{-1}	-4.06	.057	1225	77
	10^{-2}	-5.05	.068	1567	101
	10^{-3}	-6.04	.096	2072	131
	10^{-4}	-7.03	.123	2830	182
	10^{-5}	-8.02	.167	3674	253

Table 4.4d: Results for (3.61) on Class D

4. Numerical Results

	Expected Accuracy	log10 Tol	Time	FCN Calls	No. of Steps
E1	10^{-2}	-2.35	.006	105	8
	10^{-3}	-3.33	.007	136	11
	10^{-4}	-4.30	.009	187	15
	10^{-5}	-5.28	.011	249	20
	10^{-6}	-6.25	.016	331	27
	10^{-7}	-7.23	.022	462	38
	10^{-8}	-8.20	.032	647	53
	10^{-9}	-9.18	.043	893	74
E2	10^{-2}	-2.44	.014	480	31
	10^{-3}	-3.33	.018	578	40
	10^{-4}	-4.22	.026	773	54
	10^{-5}	-5.11	.041	1151	74
	10^{-6}	-5.99	.050	1434	93
	10^{-7}	-6.88	.068	1884	125
	10^{-8}	-7.77	.085	2346	168
	10^{-9}	-8.65	.111	3141	224
	10^{-10}	-9.54	.146	4099	306
	E3	10^{-2}	-2.21	.015	257
10^{-3}		-3.22	.022	387	27
10^{-4}		-4.23	.030	517	37
10^{-5}		-5.24	.040	677	52
10^{-6}		-6.25	.060	966	72
10^{-7}		-7.26	.078	1351	105
10^{-8}		-8.27	.113	1898	150
10^{-9}		-9.28	.169	2629	212
E4	10^{-3}	-2.03	.003	47	3
	10^{-4}	-3.05	.000	38	3
	10^{-5}	-4.08	.003	72	5
	10^{-6}	-5.10	.002	75	6
	10^{-7}	-6.13	.005	99	7
	10^{-8}	-7.15	.004	124	9
	10^{-9}	-8.18	.005	162	12
	10^{-10}	-9.20	.007	225	17
E5	10^{-4}	-2.81	.003	57	3
	10^{-5}	-3.93	.003	81	4
	10^{-6}	-5.05	.005	109	7
	10^{-7}	-6.17	.007	166	9
	10^{-8}	-7.29	.007	177	11
	10^{-9}	-8.42	.010	213	15
	10^{-10}	-9.54	.013	279	21

Table 4.4e: Results for (3.61) on Class E

4. Numerical Results

**Table 4.5: Results on DETEST for block 7(6)
formula (3.72)**

	Expected Accuracy	log10 Tol	Time	FCN Calls	No. of Steps
A1	10^{-4}	-2.58	.003	99	6
	10^{-5}	-3.57	.003	122	7
	10^{-6}	-4.57	.004	147	9
	10^{-7}	-5.56	.004	179	11
	10^{-8}	-6.56	.004	219	13
	10^{-9}	-7.55	.007	285	17
	10^{-10}	-8.55	.009	356	22
	10^{-11}	-9.54	.012	454	28
A2	10^{-4}	-2.16	.003	67	4
	10^{-5}	-3.27	.003	85	5
	10^{-6}	-4.37	.002	102	6
	10^{-7}	-5.48	.004	120	7
	10^{-8}	-6.59	.005	147	9
	10^{-9}	-7.70	.005	194	12
	10^{-10}	-8.81	.008	260	16
	10^{-11}	-9.92	.010	346	21
A3	10^{-2}	-1.91	.009	222	11
	10^{-3}	-2.94	.005	255	14
	10^{-4}	-3.98	.010	329	18
	10^{-5}	-5.01	.008	371	23
	10^{-6}	-6.05	.017	580	32
	10^{-7}	-7.08	.024	788	45
	10^{-8}	-8.12	.030	1027	60
	10^{-9}	-9.15	.036	1319	79
A4	10^{-4}	-2.61	.003	58	3
	10^{-5}	-3.89	.003	79	4
	10^{-6}	-5.18	.002	105	6
	10^{-7}	-6.47	.006	160	8
	10^{-8}	-7.76	.008	225	12
	10^{-9}	-9.05	.012	301	17
A5	10^{-4}	-2.37	.002	70	4
	10^{-5}	-3.52	.002	89	5
	10^{-6}	-4.66	.004	107	6
	10^{-7}	-5.81	.004	138	8
	10^{-8}	-6.96	.003	175	10
	10^{-9}	-8.11	.007	231	14
	10^{-10}	-9.26	.009	309	18

Table 4.5a: Results for (3.72) on Class A

4. Numerical Results

	Expected Accuracy	log10 Tol	Time	FCN Calls	No. of Steps
B1	10^{-1}	-2.40	.017	526	26
	10^{-2}	-3.33	.018	580	30
	10^{-3}	-4.26	.023	728	38
	10^{-4}	-5.19	.027	828	48
	10^{-5}	-6.12	.034	1041	59
	10^{-6}	-7.05	.043	1278	76
	10^{-7}	-7.97	.056	1669	101
	10^{-8}	-8.90	.066	2084	127
B2	10^{-3}	-2.49	.008	178	10
	10^{-4}	-3.34	.008	192	11
	10^{-5}	-4.19	.009	212	12
	10^{-6}	-5.04	.010	251	15
	10^{-7}	-5.90	.010	256	15
	10^{-8}	-6.75	.013	305	19
	10^{-9}	-7.60	.015	378	23
	10^{-10}	-8.46	.019	468	29
	10^{-11}	-9.31	.024	578	36
	B3	10^{-4}	-2.79	.005	109
10^{-5}		-3.81	.006	138	8
10^{-6}		-4.83	.007	171	10
10^{-7}		-5.85	.008	231	13
10^{-8}		-6.87	.010	282	16
10^{-9}		-7.89	.013	360	21
10^{-10}		-8.91	.016	427	26
10^{-11}		-9.93	.021	566	35
B4	10^{-2}	-2.85	.011	263	16
	10^{-3}	-3.88	.016	343	21
	10^{-4}	-4.91	.018	455	28
	10^{-5}	-5.94	.025	585	36
	10^{-6}	-6.98	.038	811	50
	10^{-7}	-8.01	.050	1139	71
	10^{-8}	-9.04	.061	1495	93
	10^{-9}	-10.07	.097	2073	129
	B5	10^{-2}	-2.64	.008	207
10^{-3}		-3.60	.011	272	16
10^{-4}		-4.56	.015	358	22
10^{-5}		-5.51	.018	450	28
10^{-6}		-6.47	.024	572	35
10^{-7}		-7.43	.031	753	47
10^{-8}		-8.39	.039	974	60
10^{-9}		-9.35	.051	1265	79

Table 4.5b: Results for (3.72) on Class B

4. Numerical Results

	Expected Accuracy	log10 Tol	Time	FCN Calls	No. of Steps
C1	10^{-4}	-2.75	.012	125	7
	10^{-5}	-3.78	.014	153	9
	10^{-6}	-4.80	.019	199	12
	10^{-7}	-5.83	.023	248	15
	10^{-8}	-6.85	.032	325	20
	10^{-9}	-7.88	.042	421	26
	10^{-10}	-8.90	.052	533	33
	10^{-11}	-9.93	.069	722	45
C2	10^{-3}	-2.82	.045	453	22
	10^{-4}	-3.79	.045	451	24
	10^{-5}	-4.76	.047	492	25
	10^{-6}	-5.73	.042	467	25
	10^{-7}	-6.70	.045	497	27
	10^{-8}	-7.67	.056	570	33
	10^{-9}	-8.64	.070	679	40
	10^{-10}	-9.61	.083	820	49
C3	10^{-3}	-2.26	.027	203	11
	10^{-4}	-3.14	.030	207	11
	10^{-5}	-4.03	.033	236	13
	10^{-6}	-4.92	.038	264	14
	10^{-7}	-5.81	.038	267	15
	10^{-8}	-6.69	.044	304	18
	10^{-9}	-7.58	.051	376	23
	10^{-10}	-8.47	.062	461	28
C4	10^{-3}	-2.24	.133	203	11
	10^{-4}	-3.13	.137	207	11
	10^{-5}	-4.02	.159	235	13
	10^{-6}	-4.91	.173	263	14
	10^{-7}	-5.80	.188	279	16
	10^{-8}	-6.68	.197	309	19
	10^{-9}	-7.57	.277	376	23
	10^{-10}	-8.46	.333	461	28
C5	10^{-3}	-2.10	.034	50	3
	10^{-4}	-3.22	.047	68	4
	10^{-5}	-4.34	.061	86	5
	10^{-6}	-5.46	.073	104	6
	10^{-7}	-6.57	.086	131	8
	10^{-8}	-7.69	.121	178	11
	10^{-9}	-8.81	.160	231	14
	10^{-10}	-9.93	.213	315	19

Table 4.5c: Results for (3.72) on Class C

4. Numerical Results

	Expected Accuracy	log10 Tol	Time	FCN Calls	No. of Steps
D1	10^{-1}	-2.44	.013	252	15
	10^{-2}	-3.25	.016	313	19
	10^{-3}	-4.07	.019	394	24
	10^{-4}	-4.89	.026	499	31
	10^{-5}	-5.71	.031	603	37
	10^{-6}	-6.53	.041	767	47
	10^{-7}	-7.35	.051	992	61
	10^{-8}	-8.17	.060	1256	78
	10^{-9}	-8.98	.079	1531	95
	10^{-10}	-9.80	.105	2012	125
D2	10^{-1}	-2.57	.016	345	18
	10^{-2}	-3.45	.020	441	24
	10^{-3}	-4.33	.028	590	31
	10^{-4}	-5.21	.036	689	38
	10^{-5}	-6.09	.038	735	44
	10^{-6}	-6.97	.047	938	58
	10^{-7}	-7.85	.064	1258	78
	10^{-8}	-8.73	.083	1593	99
	10^{-9}	-9.61	.106	2098	131
	D3	10^0	-1.96	.020	384
10^{-1}		-2.88	.026	501	26
10^{-2}		-3.79	.027	605	32
10^{-3}		-4.71	.038	762	41
10^{-4}		-5.62	.047	936	51
10^{-5}		-6.54	.054	1129	63
10^{-6}		-7.46	.065	1349	81
10^{-7}		-8.37	.082	1655	103
10^{-8}		-9.29	.108	2152	134
D4		10^0	-2.45	.033	702
	10^{-1}	-3.46	.037	812	40
	10^{-2}	-4.47	.046	971	50
	10^{-3}	-5.49	.055	1135	61
	10^{-4}	-6.50	.069	1429	79
	10^{-5}	-7.51	.085	1760	102
	10^{-6}	-8.53	.115	2206	133
D5	10^1	-2.09	.045	995	44
	10^0	-3.32	.052	1037	54
	10^{-1}	-4.55	.074	1486	77
	10^{-2}	-5.77	.102	2022	101
	10^{-3}	-7.00	.111	2410	132

Table 4.5d: Results for (3.72) on Class D

4. Numerical Results

	Expected Accuracy	log10 Tol	Time	FCN Calls	No. of Steps
E1	10^{-2}	-2.06	.007	115	7
	10^{-3}	-3.02	.005	161	10
	10^{-4}	-3.98	.008	207	12
	10^{-5}	-4.93	.015	268	16
	10^{-6}	-5.89	.016	329	20
	10^{-7}	-6.85	.022	431	26
	10^{-8}	-7.80	.028	590	36
	10^{-9}	-8.76	.038	759	47
	10^{-10}	-9.72	.052	1007	62
	E2	10^{-1}	-2.01	.015	457
10^{-2}		-2.89	.018	558	29
10^{-3}		-3.78	.029	741	39
10^{-4}		-4.67	.034	906	50
10^{-5}		-5.56	.040	1089	61
10^{-6}		-6.44	.049	1415	79
10^{-7}		-7.33	.066	1808	103
10^{-8}		-8.22	.089	2202	132
10^{-9}		-9.11	.114	2898	169
10^{-10}		-10.00	.142	3703	222
E3	10^{-2}	-2.00	.015	267	15
	10^{-3}	-3.10	.021	336	19
	10^{-4}	-4.20	.027	466	27
	10^{-5}	-5.31	.038	635	38
	10^{-6}	-6.41	.051	848	51
	10^{-7}	-7.51	.070	1193	72
	10^{-8}	-8.61	.095	1696	100
	10^{-9}	-9.71	.137	2366	141
	E4	10^{-3}	-2.38	.002	49
10^{-4}		-3.38	.002	55	3
10^{-5}		-4.38	.002	71	4
10^{-6}		-5.37	.003	86	5
10^{-7}		-6.37	.004	102	6
10^{-8}		-7.37	.005	124	7
10^{-9}		-8.37	.006	156	9
10^{-10}		-9.37	.008	194	12
E5	10^{-3}	-2.06	.002	49	3
	10^{-4}	-3.10	.003	52	3
	10^{-5}	-4.14	.004	87	4
	10^{-6}	-5.19	.005	123	6
	10^{-7}	-6.23	.006	131	6
	10^{-8}	-7.28	.008	161	8
	10^{-9}	-8.32	.008	169	9
	10^{-10}	-9.37	.009	210	12

Table 4.5e: Results for (3.72) on Class E

4. Numerical Results

Table 4.6: Results on DETEST over all groups

log10 Tol	Time	FCN Calls	No. of Steps	Maximum loc err	Fraction received	Fraction bad received
-2.00	.376	4765	417	1.234	.002	.000
-3.00	.437	6172	571	.903	.000	.000
-4.00	.541	8339	799	1.133	.001	.000
-5.00	.741	11798	1174	1.337	.002	.000
-6.00	.973	16802	1679	1.056	.001	.000
-7.00	1.378	23894	2563	1.279	.001	.000
-8.00	2.095	36818	4007	1.183	.000	.000
-9.00	3.106	54688	5970	.996	.000	.000
-10.00	4.858	85334	9387	1.211	.000	.000

Table 4.6a: Summary of results on DETEST for the block 5(4) method

log10 Tol	Time	FCN Calls	No. of Steps	Maximum loc err	Fraction received	Fraction bad received
-2.00	.283	3700	339	20.962	.195	.038
-3.00	.310	4575	428	27.496	.149	.031
-4.00	.411	6019	588	12.865	.116	.010
-5.00	.535	8150	815	5.831	.096	.001
-6.00	.729	11527	1145	4.790	.054	.000
-7.00	1.001	16483	1710	1.952	.005	.000
-8.00	1.488	24616	2629	5.366	.001	.000
-9.00	2.266	37219	3915	.748	.000	.000
-10.00	3.431	56719	6124	.399	.000	.000

Table 4.6b: Summary of results on DETEST for the block 5(4) existing method

log10 Tol	Time	FCN Calls	No. of Steps	Maximum loc err	Fraction received	Fraction bad received
-2.00	.330	4970	328	9.481	.043	.003
-3.00	.400	6130	422	4.367	.021	.000
-4.00	.495	7804	547	4.090	.031	.000
-5.00	.614	10246	734	6.742	.018	.001
-6.00	.791	13446	953	5.754	.009	.001
-7.00	1.068	18173	1317	.716	.000	.000
-8.00	1.452	24142	1848	.428	.000	.000
-9.00	1.953	33633	2523	.372	.000	.000
-10.00	2.728	44882	3613	.324	.000	.000

Table 4.6c: Summary of results on DETEST for the block 6(5) method

log10 Tol	Time	FCN Calls	No. of Steps	Maximum loc err	Fraction received	Fraction bad received
-2.00	.479	6293	329	10.542	.046	.006
-3.00	.519	6943	385	4.210	.034	.000
-4.00	.635	8893	495	4.615	.016	.000
-5.00	.786	11043	627	3.436	.011	.000
-6.00	.901	13411	762	1.987	.009	.000
-7.00	1.089	16957	996	1.908	.006	.000
-8.00	1.484	21741	1324	1.357	.002	.000
-9.00	1.841	27731	1693	2.132	.001	.000
-10.00	2.484	37441	2311	.329	.000	.000

Table 4.6d: Summary of results on DETEST for the block 7(6) method

4. Numerical Results

We examine first of all the reliability of the four block methods. We can measure this by examining the columns “Fraction deceived” and “Fraction bad deceived”. As explained earlier, this is a measure of the fraction of the total number of accepted steps for which the local error estimate was greater than the true local error, five times the true local error, respectively. As can be seen all four block methods are very reliable. The number of steps on which the code was badly deceived in particular is very small. However out of all the codes the new 5(4) code is by far the most reliable. The maximum local error is only 1.337 times the true error and the fraction of steps on which the code is deceived is very small. At the stricter tolerances the new 6(5) and 7(6) codes are also extremely reliable. These results certainly indicate that the block codes have the desired level of reliability.

The performance of the codes on the DETEST set is exactly what would be expected. Overall the performances of the two block 5(4) methods are very similar. One particular formula is better on one class of problems while the other formula is better on another class. Generally the 6(5) block formula is better than the 5(4) formulae at moderate and strict tolerances. Finally the 7(6) formula is generally better than the 6(5) formula for very strict tolerances. This bears out what is generally held to be true in that high order methods are often preferable for strict tolerances. We feel that the new codes that we have developed are extremely competitive with existing codes, c.f. the numerical results given in [Cash89], and we are at present putting these codes in NETLIB for general use.

Chapter 5

Global Error Estimation

5.1 Introduction

Codes for solving initial value problems typically attempt to keep an estimate of the local truncation error less than a user defined maximum. However the user is often more interested in the size of the global error, that is the difference between the true solution of the initial value problem and the solution generated by the code. Unfortunately there is often no proportionality between the specified local error and the global error actually obtained. For this reason it is difficult to estimate the global error cheaply and reliably. One obvious way of estimating global error is by extrapolation. However this can be expensive often tripling the overall cost. A second approach, proposed by Dormand and Prince [**Dormand84**], [**Dormand89b**], [**Dormand85**], [**Dormand86**], [**Dormand89a**], involves integrating an error equation using a Runge-Kutta triple. In what follows we will describe these approaches and show that the Dormand and Prince approach is often very reliable and produces global error estimates with a reasonable cost.

5.2 Existing Methods

In [Dormand89a] a survey of recent advances in global error estimation is given. In what follows we will give a short description of some of the methods used to estimate the global error when solving a problem of the form (1.1) by an embedded explicit Runge-Kutta pair $p(q)$, where $p > q$. The global error at x_n is given by:

$$\varepsilon_n = y_n - y(x_n), \quad n = 0, 1, \dots \quad (5.1)$$

where y_n is the numerical approximation to the true solution $y(x_n)$ of the initial value problem at $x = x_n$. We start of by considering the Richardson extrapolation technique.

1. The classical method of global error estimation is *Richardson extrapolation*. In this technique, two separate solutions are computed one with a stepsize h and the other with a step $h/2$. Suppose that we are using an explicit Runge-Kutta method of order p and denote the solutions obtained with step h and $h/2$ by y^h , $y^{h/2}$ respectively. Assuming that the principal local truncation error of the numerical method is:

$$LTE = h^{p+1}\phi_p + O(h^{p+2}) \quad (5.2)$$

we have after $k + 1$ steps

$$y(x_{k+1}) - y_{k+1}^h = (k + 1)h^{p+1} \phi_p + O(h^{p+2}) \quad (5.3)$$

$$y(x_{k+1}) - y_{k+1}^{h/2} = 2(k + 1)\left(\frac{h}{2}\right)^{p+1} \phi_p + O(h^{p+2}). \quad (5.4)$$

From relations (5.3) and (5.4) we have:

$$y_{k+1}^{h/2} - y_{k+1}^h = \left(1 - \frac{1}{2^p}\right) (k + 1)h^{p+1} \phi_p.$$

So

$$y(x_{k+1}) - y_{k+1}^h = \frac{2^p(y_{k+1}^{h/2} - y_{k+1}^h)}{2^p - 1}. \quad (5.5)$$

This technique is expensive as we need to compute two completely separate solutions. If the basic p^{th} order Runge-Kutta method uses s function evaluations per step then the total cost of this technique is $3s$ functions per step.

5. Global Error Estimation

The second global error estimation technique that we wish to consider is due to Zadunaisky [zad76].

2. The *Zadunaisky technique* to estimate the global error is summarised in [Dormand84]. We denote the problem (1.1) that we wish to solve by the “main problem” and carry out the following steps:

- (a) use an order p Runge-Kutta method to solve the main problem (MP) on an interval $I = [x_0, x_N]$, giving a numerical solution y_i at x_i , $i = 0, 1, \dots, N$.
- (b) Construct a neighbouring problem (NP), with known true solution $y_h(x)$, which is in some sense close to the MP.
- (c) Using the same order p method and step sequence as in (a), solve the NP giving a numerical solution y_{hi} at x_i , $i = 0, 1, \dots, N$.
- (d) At any point $x_n \in I$ calculate the global error

$$\varepsilon_{hn} = y_{hn} - y_h(x_n) \tag{5.6}$$

for the NP and use this as an estimate of ε_n , the global error in the MP.

There are many ways in which the neighbouring problem can be constructed. Normally these are based on approximating $y(x)$ by an interpolating polynomial. For example we could take as the NP:

$$y'_h(x) = f(x, y_h(x)) + d_h(x) \tag{5.7}$$

where $d_h(x) \equiv P'(x) - f(x, P(x))$ and $P(x)$ is a polynomial in x . The important point to write about (5.7) is that it has a known true solution $y_h(x) = P(x)$. The analysis given by Zadunaisky shows that the known global error committed in integrating (5.7) is a good approximation to the global error committed in integrating the main equation. There are many ways of defining the polynomial, $P(x)$, and among those that have been suggested are:

- (i) interpolation of y_i values;
- (ii) interpolation of f_i values;
- (iii) interpolation using y_i and f_i values, over a block of ν steps for $i = 0, 1, \dots, N$.

There are several problems with the Zadunaisky approach. One is the need to integrate both the (MP) and (NP) using exactly the same step sequence and error control. Another is the fact that the interpolating polynomials, being of high degree and defined over several past steps, are often highly oscillatory. Practical experience shows that Zadunaisky’s approach often produces poor

5. Global Error Estimation

results when the interpolating polynomial has high degree. To avoid these drawbacks an alternative approach was proposed by Dormand and Prince.

3. The Dormand and Prince approach, [Dormand86], [Dormand89a] consists of using an embedded pair of explicit Runge-Kutta formulae together with a third *dense formula* which is used to provide a continuous solution. We write these three formulae as:

$$y_{n+1} = y_n + h_n \sum_{i=1}^s b_i k_i \quad (5.8)$$

$$\bar{y}_{n+1} = y_n + h_n \sum_{i=1}^{\bar{s}} \bar{b}_i k_i \quad (5.9)$$

$$y_{n+\sigma}^* = y_n + \sigma h_n \sum_{i=1}^{s^*} b_i^* k_i \quad (5.10)$$

where $k_i = f(x_n + c_i h_n, y_n + h_n \sum_{j=1}^{i-1} a_{ij} k_j)$, $i = 1, 2, \dots, \text{Max}(s, s^*)$, $\sigma \in (0, 1)$.

Formula (5.10) can be used to provide estimates $y_{n+\sigma}^*$ of $y(x_n + \sigma h_n)$ for any point σ in the range $[0,1]$. We will assume that the main formula (5.8) is of order p , the embedded formula (5.9) is of order q while the dense formula (5.10) is of order p^* . The way in which these three formulae are used is as follows.

- (i) Use formulae (5.8), (5.9) to integrate the initial value problem (1.1) so as to obtain an estimate \hat{y}_{n+1} of $y(x_{n+1})$ which satisfies the local error criterion.
- (ii) Use the dense formula (5.10) to obtain the required non-mesh point values $y_{n+\sigma}^*$ needed to compute the interpolating polynomial $P(x)$.
- (iii) Use an error integrator to integrate the neighbouring problem

$$\varepsilon'_h(x) = P'(x) - f[P(x) - \varepsilon_h(x)] \quad (5.11)$$

to give the global error estimate $\varepsilon_h(x)$.

An analysis of the accuracy requirements of the error integrator has been carried out by Dormand and Prince. Suppose that the order of (5.10) is p^* , the degree of the interpolant is m and the order of (5.8) is p . Then certain additional accuracy requirements need to be satisfied by an error integrator of order p if it is to provide asymptotically correct estimates of the global error. The form taken by these accuracy requirements is that coefficients of certain higher order differentials should vanish. Denoting the order m of the

5. Global Error Estimation

p	\bar{m}	h^p	h^{p+1}
2	1	$\tau_1^{(3)}$	$\tau_i^{(3)}, i=1,2; \tau_i^{(4)}, i=1,3$
	2	$\tau_1^{(j)}, j=3,4,5$	$\tau_i^{(3)}, i=1,2; \tau_i^{(4)}, i=1,3; \tau_i^{(5)}, i=1,5; \tau_7^{(6)}$
	3	$\tau_1^{(j)}, j=3, \dots, 7$	too many to include
3	1	$\tau_1^{(4)}$	$\tau_i^{(4)}, i=1,3; \tau_i^{(5)}, i=1,5$
	2	$\tau_1^{(j)}, j=4,5$	$\tau_i^{(4)}, i=1,3; \tau_i^{(5)}, i=1,5; \tau_7^{(6)}$
	3	$\tau_1^{(j)}, j=4, \dots, 7$	too many to include
4	1	$\tau_i^{(5)}, i=1,5$	$\tau_i^{(5)}, i=1,4,5,8; \tau_i^{(6)}, i=1,6,7,15$
	2	$\tau_1^{(5)}$	$\tau_i^{(5)}, i=1,5; \tau_i^{(6)}, i=1,7$
	3	$\tau_1^{(j)}, j=5,6,7$	too many to include
5	1	$\tau_i^{(6)}, i=1,6,7,15$	too many to include
	2	$\tau_1^{(6)}$	$\tau_i^{(6)}, i=1,7; \tau_i^{(7)}, i=1,11$
	3	$\tau_1^{(6)}, \tau_1^{(7)}$	$\tau_i^{(6)}, i=1,7; \tau_i^{(7)}, i=1,11; \tau_{15}^{(8)}$
6	1	$\tau_i^{(7)}, i=1,10,11,29$	too many to include
	2	$\tau_i^{(7)}, i=1,11$	$\tau_i^{(7)}, i=1,10,11,29; \tau_i^{(8)}, i=1,14,15,53$
	3	$\tau_1^{(7)}$	$\tau_i^{(7)}, i=1,11; \tau_i^{(8)}, i=1,15$
	4	$\tau_1^{(j)}, j=7,8,9$	$\tau_i^{(7)}, i=1,11; \tau_i^{(8)}, i=1,15; \tau_i^{(9)}, i=1,22; \tau_{30}^{(10)}$

Table 5.1: Extra order conditions that must be satisfied for each order p to obtain asymptotically correct global error estimates

interpolating polynomial by $m = 2\bar{m} + 1$ these accuracy conditions are given in Table 5.1. In Table 5.1 we give two columns of elementary differentials. The first, under the heading h^p , lists those elementary differentials that need to vanish in order for the first term of the global error to be estimated. The second column gives the differentials that need to vanish for the second term of the global error to be estimated.

5.3 Cost of the Global Error Estimation

In what follows we will compare the cost of two of the methods we have proposed for global error estimation – namely Richardson extrapolation and the Dormand and Prince approach. We will measure the cost initially by counting the number of function evaluations. Let N_S and N_R be the number of steps which are successful and the number of rejected steps respectively. Also let s be the number of stages of the main formula, s^* the number of stages of the dense formula and \bar{s} the number of stages of the estimator formula. We will assume that a FSAL technique is used so that the last stage in the error estimator formula is the first stage of the main formula. Under these conditions the cost in terms of the number of function evaluations of applying the integrator to compute the main solution (i.e. without the error estimator) is:

$$\text{Main Solution} \quad C_M = (s-1)(N_S + N_R) + 1$$

$$\text{Richardson Extrapolation} \quad C_E = 2(s-1)N_S$$

$$\text{Dormand and Prince} \quad C_{DP} = s^*N_S.$$

Note that the error estimate is not applied at failed steps. The cost of the Dormand and Prince method comes from the $s^* - s$ stages to compute the interpolant. We note that the direct method of Dormand and Prince is cheaper than the extrapolation method as a method of error estimation if the cost of function evaluations dominates the total computational cost. This is normally the case with problems of practical interest. However the Dormand and Prince approach also involves the evaluation of $P(x)$ and $P'(x)$ where $P(x)$ denotes the polynomial given by the dense formula. In cases where function evaluations are cheap these evaluations of $P(x)$ will be a significant part of the overall computational cost.

5.4 Global Error Estimation for Block Methods

In this section we will extend the approach of Dormand and Prince to block Runge-Kutta methods. We will consider the case where we solve the initial value problem (1.1) by one of the explicit block formulae described in Chapter 3 and we wish to derive a global error estimator. The neighbouring problem will be defined by means of the interpolating polynomial associated with the particular block formula we are using. The global error integrator needs to satisfy certain extra order conditions to make the global error estimate asymptotically correct and these conditions are given in Table 5.1. Here p is the order of the main method, m is the degree of the polynomial, $\tau_j^{(i)}$ represent the order conditions where the superscript represents the order and the subscript represents one of the elementary differentials of each order. Using the free parameters we will try to find methods that have a good stability region and good accuracy. We have applied this technique to the block methods derived in Chapter 3 for orders 3, 4 and 5 and in what follows we give the global error estimators that we have derived.

5.4.1 Order 3 with 5 Stages

In what follows we will consider a third order block formula with five stages:

$$\begin{array}{c|ccccc}
 0 & 0 & & & & \\
 c_2 & c_2 & 0 & & & \\
 c_3 & a_{31} & a_{32} & 0 & & \\
 c_4 & a_{41} & a_{42} & a_{43} & 0 & \\
 c_5 & a_{51} & a_{52} & a_{53} & a_{54} & 0 \\
 \hline
 & b_1 & b_2 & b_3 & b_4 & b_5
 \end{array} \tag{5.12}$$

which will be used for the purpose of global error estimation. The stability polynomial associated with (5.12) is: $\frac{y_{n+1}}{y_n} = 1 + z + \frac{1}{2}z^2 + \frac{1}{6}z^3 + u_4z^4 + u_5z^5$, where $u_4 = b_4a_{43}a_{32}c_2 + b_5(a_{53}a_{32}c_2 + a_{54}(a_{42}c_2 + a_{43}c_3))$ and $u_5 = b_5a_{54}a_{43}a_{32}c_2$.

5. Global Error Estimation

We have 18 free parameters to be determined. The row-sum conditions are:

$$c_3 = \sum_{i=1}^2 a_{3i} \quad (5.13)$$

$$c_4 = \sum_{i=1}^3 a_{4i} \quad (5.14)$$

$$c_5 = \sum_{i=1}^4 a_{5i}. \quad (5.15)$$

We want the method (5.12) to be of order 3 so it has to satisfy the following order conditions: $\tau_1^{(1)}$, $\tau_1^{(2)}$, $\tau_1^{(3)}$, $\tau_2^{(3)}$. By looking at Table 5.1 with $p = 3$ and $\bar{m} = 2$ we also have to satisfy $\tau_1^{(4)}$, $\tau_1^{(5)}$, $\tau_3^{(4)}$, $\tau_5^{(5)}$, and $\tau_7^{(6)}$ in order to be able to obtain an asymptotically correct global error estimate which estimates the first two terms of the global error. To solve these equations the free parameters were chosen in the following way:

- 1) choose c_2 , c_3 , c_4 and c_5 arbitrarily;
- 2) solve $\tau_1^{(i)}$ with $i = 1, 2, 3, 4, 5$ for b_i ;
- 3) choose a_{42} , a_{52} and a_{53} arbitrarily;
- 4) solve $\tau_5^{(5)}$ for a_{54} ;
- 5) solve $\tau_3^{(4)}$ for a_{43} ;
- 6) solve $\tau_2^{(3)}$ for a_{32} ;
- 7) solve the row-sum conditions for a_{31} , a_{41} and a_{51} ;
- 8) evaluate u_4 and u_5 ;
- 9) make sure that $\tau_7^{(6)}$ is satisfied;
- 10) try to choose the free coefficients so that the square root of the sum of the squares of $\tau_2^{(4)}$ and $\tau_4^{(4)}$ should be as small as possible.

The formula found using this approach is given below.

$$\begin{array}{c|cccccc}
 0 & 0 & & & & & \\
 \frac{1}{10} & \frac{1}{10} & 0 & & & & \\
 1 & \frac{1067}{272} & -\frac{795}{272} & 0 & & & \\
 \frac{4}{5} & \frac{2723}{1350} & \frac{29}{10} & -\frac{56}{675} & 0 & & \\
 \frac{2}{5} & -\frac{183}{98} & \frac{19}{10} & 0 & \frac{18}{49} & 0 & \\
 \hline
 & \frac{1}{32} & \frac{100}{576} & \frac{17}{324} & \frac{75}{224} & \frac{175}{432} &
 \end{array} \quad (5.16)$$

5. Global Error Estimation

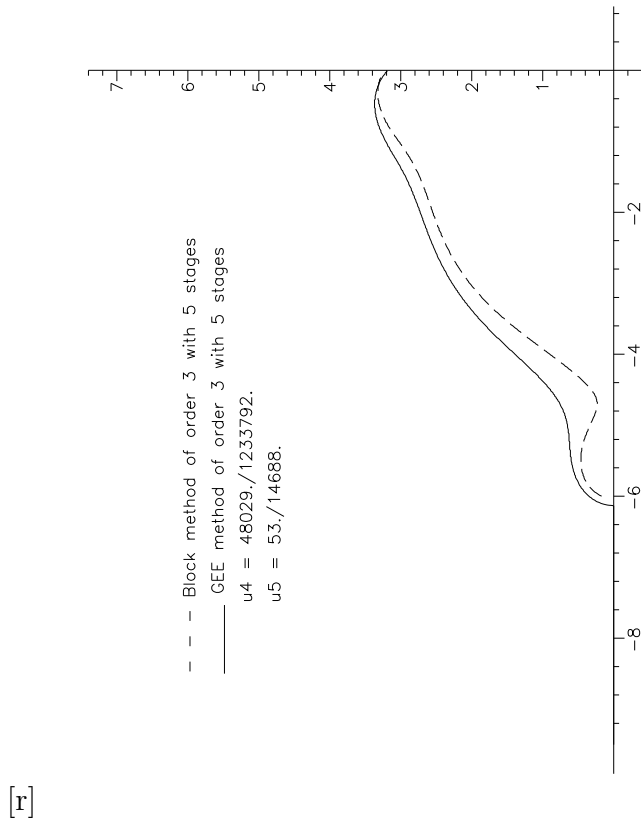


Figure 5.1: Stability region of order 3 with 5 stages

Here $u_4 = \frac{48029}{1233792}$, $u_5 = \frac{53}{14688}$ and $\sqrt{(\tau_2^{(4)})^2 + (\tau_4^{(4)})^2} \cong 0.00703$. The graph of the stability regions of the block method of order 3 (3.26) and of (5.16) is given in Figure 5.1.

5.4.2 Order 3 with 6 Stages

In the previous section we derived a global error estimator with 5 stages. In what follows we consider if there is an advantage to be gained in allowing the error estimator to have 6 stages. We consider an error estimator of the

5. Global Error Estimation

form:

$$\begin{array}{c|cccccc}
 0 & 0 & & & & & \\
 c_2 & c_2 & 0 & & & & \\
 c_3 & a_{31} & a_{32} & 0 & & & \\
 c_4 & a_{41} & a_{42} & a_{43} & 0 & & \\
 c_5 & a_{51} & a_{52} & a_{53} & a_{54} & 0 & \\
 c_6 & a_{61} & a_{62} & a_{63} & a_{64} & a_{65} & 0 \\
 \hline
 & b_1 & b_2 & b_3 & b_4 & b_5 & b_6
 \end{array} \tag{5.17}$$

The stability polynomial associated with (5.17) is: $\frac{y_{n+1}}{y_n} = 1 + z + \frac{1}{2}z^2 + \frac{1}{6}z^3 + u_4z^4 + u_5z^5 + u_6z^6$, where $u_4 = \sum_{i,j,k=1}^s b_i a_{ij} a_{jk} c_k$, $u_5 = \sum_{i,j,k,l=1}^s b_i a_{ij} a_{jk} a_{kl} c_l$ and $u_6 = b_6 a_{65} a_{54} a_{43} a_{32} c_2$. We have 25 free parameters to be determined. The row-sum conditions are:

$$c_3 = \sum_{i=1}^2 a_{3i} \tag{5.18}$$

$$c_4 = \sum_{i=1}^3 a_{4i} \tag{5.19}$$

$$c_5 = \sum_{i=1}^4 a_{5i} \tag{5.20}$$

$$c_6 = \sum_{i=1}^5 a_{6i}. \tag{5.21}$$

We want the method to be of order 3 so it has to satisfy the following order conditions: $\tau_1^{(1)}, \tau_1^{(2)}, \tau_1^{(3)}, \tau_2^{(3)}$. By looking at Table 5.1 with $p = 3$ and $\bar{m} = 2$ we also have to satisfy $\tau_1^{(4)}, \tau_1^{(5)}, \tau_3^{(4)}, \tau_5^{(5)}$, and $\tau_7^{(6)}$. The parameters were chosen in the following way:

- 1) choose c_2, c_3, c_4, c_5, c_6 and b_6 arbitrarily;
- 2) solve $\tau_1^{(i)}$ with $i = 1, 2, 3, 4, 5$ for b_i ;
- 3) choose $a_{42}, a_{52}, a_{53}, a_{62}, a_{63}$ and a_{64} arbitrarily;
- 4) solve $\tau_7^{(6)}$ for a_{65} ;
- 5) solve $\tau_5^{(5)}$ for a_{54} ;
- 6) solve $\tau_3^{(4)}$ for a_{43} ;
- 7) solve $\tau_2^{(3)}$ for a_{32} ;
- 8) solve the row-sum conditions for a_{31}, a_{41}, a_{51} and a_{61} ;
- 9) evaluate u_4, u_5 and u_6 ;

5. Global Error Estimation

10) try to choose the free coefficients so that the square root of the sum of the squares of $\tau_2^{(4)}$ and $\tau_4^{(4)}$ should be as small as possible.

The formula found is:

$$\begin{array}{c|cccccc}
 0 & 0 & & & & & \\
 \frac{1}{10} & \frac{1}{10} & 0 & & & & \\
 \frac{1}{4} & \frac{1}{4} & 0 & 0 & & & \\
 \frac{1}{2} & \frac{1667}{7332} & 0 & \frac{1999}{7332} & 0 & & \\
 \frac{3}{4} & \frac{133}{2560} & 0 & \frac{1}{2} & \frac{507}{2560} & 0 & \\
 1 & \frac{109}{270} & 0 & -\frac{1}{2} & \frac{4}{5} & \frac{8}{27} & 0 \\
 \hline
 & -\frac{173}{90} & \frac{625}{117} & -\frac{224}{45} & \frac{47}{15} & -\frac{512}{585} & \frac{3}{10}
 \end{array} \tag{5.22}$$

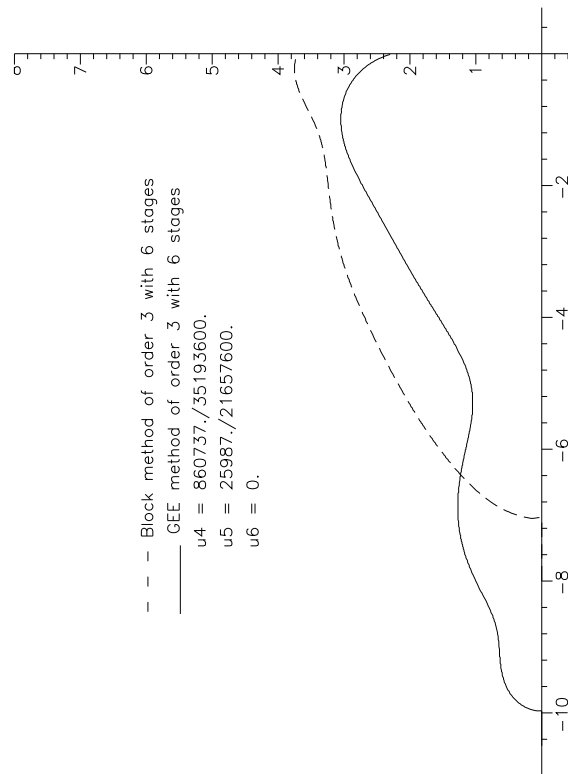
Here $u_4 = \frac{860737}{35193600}$, $u_5 = \frac{25987}{21657600}$, $u_6 = 0$ and $\sqrt{(\tau_2^{(4)})^2 + (\tau_4^{(4)})^2} \cong 0.555 \times 10^{-3}$. The graph of the stability regions of the block method of order 3 (3.29) and of (5.22) is given in Figure 5.2. As can be seen, by allowing one additional function evaluation we can find formulae which are considerably more accurate and which have much better stability regions than formulae with five functions. Hence (5.22) will be the third order method of choice.

5.4.3 Order 4 with 7 Stages

In this section we will consider the derivation of an error estimator for use with a fourth order block formula. We will consider first of all a formula of the form:

$$\begin{array}{c|ccccccc}
 0 & 0 & & & & & \\
 c_2 & c_2 & 0 & & & & \\
 c_3 & a_{31} & a_{32} & 0 & & & \\
 c_4 & a_{41} & a_{42} & a_{43} & 0 & & \\
 c_5 & a_{51} & a_{52} & a_{53} & a_{54} & 0 & \\
 c_6 & a_{61} & a_{62} & a_{63} & a_{64} & a_{65} & 0 \\
 c_7 & a_{71} & a_{72} & a_{73} & a_{74} & a_{75} & a_{76} & 0 \\
 \hline
 & b_1 & b_2 & b_3 & b_4 & b_5 & b_6 & b_7
 \end{array} \tag{5.23}$$

5. Global Error Estimation



[r]

Figure 5.2: Stability region of order 3 with 6 stages

5. Global Error Estimation

The stability polynomial associated with (5.23) is: $\frac{y_{n+1}}{y_n} = 1 + z + \frac{1}{2}z^2 + \frac{1}{6}z^3 + \frac{1}{24}z^4 + u_5z^5 + u_6z^6 + u_7z^7$, where $u_5 = \sum_{i,j,k,l=1}^7 b_i a_{ij} a_{jk} a_{kl} c_l = \tau_9^{(5)}$,
 $u_6 = \sum_{i,j,k,l,m=1}^7 b_i a_{ij} a_{jk} a_{kl} a_{lm} c_m$ and $u_7 = \sum_{i,j,k,l,m,n=1}^7 b_i a_{ij} a_{jk} a_{kl} a_{lm} a_{mn} c_n$. We have 33 free parameters to be determined. The row-sum conditions are:

$$c_3 = \sum_{i=1}^2 a_{3i} \quad (5.24)$$

$$c_4 = \sum_{i=1}^3 a_{4i} \quad (5.25)$$

$$c_5 = \sum_{i=1}^4 a_{5i} \quad (5.26)$$

$$c_6 = \sum_{i=1}^5 a_{6i} \quad (5.27)$$

$$c_7 = \sum_{i=1}^6 a_{7i}. \quad (5.28)$$

We want the method to be of order 4 so it has to satisfy the following order conditions: $\tau_1^{(1)}$, $\tau_1^{(2)}$, $\tau_1^{(3)}$, $\tau_2^{(3)}$, $\tau_1^{(4)}$, $\tau_2^{(4)}$, $\tau_3^{(4)}$, $\tau_4^{(4)}$. By looking at Table 5.1 with $p = 4$ and $\bar{m} = 2$ we also have to satisfy $\tau_1^{(5)}$, $\tau_5^{(5)}$, $\tau_1^{(6)}$, and $\tau_7^{(6)}$ to obtain a formula which estimates the first two terms of the global error. We will use the usual row and column-simplifying assumptions. The parameters were chosen in the following way:

- 1) set $b_2 = 0$ and $c_7 = 1$;
- 2) choose c_2 , c_3 , c_4 , c_5 and c_6 arbitrarily;
- 3) solve $\tau_1^{(i)}$ with $i = 1, 2, 3, 4, 5$ for b_1 , b_3 , b_4 , b_5 , b_6 and b_7 ;
- 4) choose a_{43} , a_{53} , a_{54} , a_{63} , a_{64} and a_{65} arbitrarily;
- 5) solve $\sum_j a_{ij} c_j = \frac{1}{2} c_i^2$ with $i = 3, 4, 5, 6$ for a_{32} , a_{42} , a_{52} and a_{62} ;
- 6) solve the simplifying assumptions of the form $\sum_{i=1}^s b_i a_{ij} = b_j(1 - c_j)$ with $j = 3, 4, 5, 6$ for a_{7j} ;
- 7) get a_{72} from $\sum_j a_{ij} c_j = \frac{1}{2} c_i^2$ for $i = 7$;
- 8) solve the row-sum conditions for a_{31} , a_{41} , a_{51} , a_{61} and a_{71} ;
- 9) evaluate u_5 , u_6 and u_7 ;

5. Global Error Estimation

10) try to choose the free coefficients so that the square root of the sum of the squares of $\tau_2^{(5)}, \tau_3^{(5)}, \tau_4^{(5)}, \tau_6^{(5)}, \tau_7^{(5)}, \tau_8^{(5)}$ and $\tau_9^{(5)}$ should be as small as possible.

The formula found is:

$$\begin{array}{c|ccccccc}
 0 & 0 & & & & & & \\
 \frac{1}{10} & \frac{1}{10} & 0 & & & & & \\
 \frac{1}{5} & 0 & \frac{1}{5} & 0 & & & & \\
 \frac{2}{5} & \frac{3}{10} & \frac{3}{5} & \frac{1}{10} & 0 & & & \\
 \frac{3}{5} & \frac{11}{25} & \frac{23}{25} & -\frac{1}{5} & \frac{8}{25} & 0 & & \\
 \frac{4}{5} & \frac{67}{250} & -\frac{78}{125} & \frac{7}{10} & \frac{39}{250} & \frac{3}{10} & 0 & \\
 1 & \frac{359}{190} & \frac{221}{95} & \frac{25}{38} & \frac{23}{190} & -\frac{5}{38} & \frac{15}{19} & 0 \\
 \hline
 & \frac{19}{288} & 0 & \frac{25}{96} & \frac{25}{144} & \frac{25}{144} & \frac{25}{96} & \frac{19}{288}
 \end{array} \tag{5.29}$$

Here $u_5 = \frac{59}{10000}$, $u_6 = \frac{2867}{7200000}$, $u_7 = \frac{1}{100000}$ and $\sqrt{\left(\sum_{i=1}^9 \tau_i^{(5)}\right)^2} \cong 0.004243$.

The graph of the stability regions of the block method of order 4 (3.41) and of (5.29) is given in Figure 5.3.

5.4.4 Order 4 with 8 Stages

As in the third order case we will consider whether there is any advantage to be gained by allowing our global error estimator to have one extra function evaluation. With this in mind we consider:

$$\begin{array}{c|ccccccc}
 0 & 0 & & & & & & \\
 c_2 & c_2 & 0 & & & & & \\
 c_3 & a_{31} & a_{32} & 0 & & & & \\
 c_4 & a_{41} & a_{42} & a_{43} & 0 & & & \\
 c_5 & a_{51} & a_{52} & a_{53} & a_{54} & 0 & & \\
 c_6 & a_{61} & a_{62} & a_{63} & a_{64} & a_{65} & 0 & \\
 c_7 & a_{71} & a_{72} & a_{73} & a_{74} & a_{75} & a_{76} & 0 \\
 c_8 & a_{81} & a_{82} & a_{83} & a_{84} & a_{85} & a_{86} & a_{87} & 0 \\
 \hline
 & b_1 & b_2 & b_3 & b_4 & b_5 & b_6 & b_7 & b_8
 \end{array} \tag{5.30}$$

5. Global Error Estimation

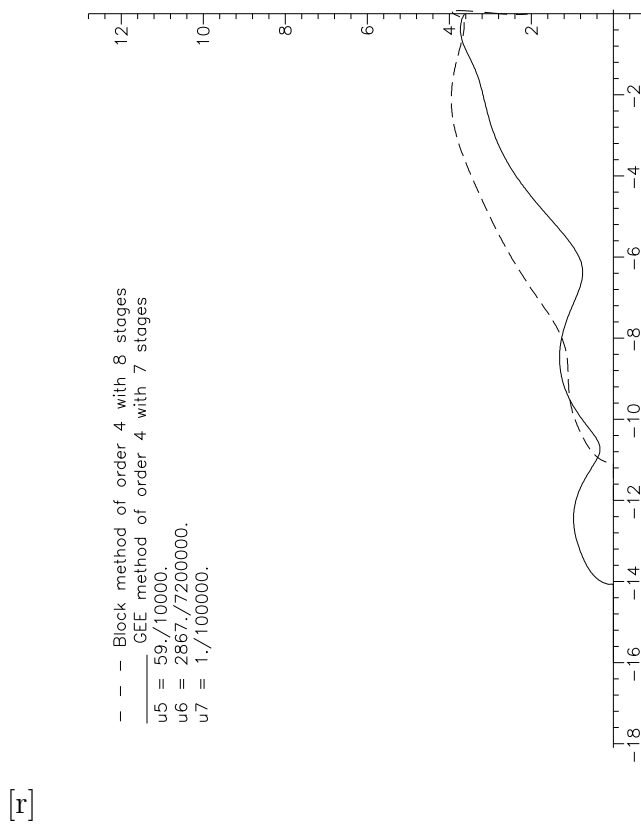


Figure 5.3: Stability region of order 4 with 7 stages

5. Global Error Estimation

The stability polynomial associated with (5.30) is: $\frac{y_{n+1}}{y_n} = 1 + z + \frac{1}{2}z^2 + \frac{1}{6}z^3 + \frac{1}{24}z^4 + u_5z^5 + u_6z^6 + u_7z^7 + u_8z^8$, where $u_5 = \sum_{i,j,k,l=1}^7 b_i a_{ij} a_{jk} a_{kl} c_l = \tau_9^{(5)}$,
 $u_6 = \sum_{i,j,k,l,m=1}^8 b_i a_{ij} a_{jk} a_{kl} a_{lm} c_m$, $u_7 = \sum_{i,j,k,l,m,n=1}^8 b_i a_{ij} a_{jk} a_{kl} a_{lm} a_{mn} c_n$ and
 $u_8 = \sum_{i,j,k,l,m,n,o=1}^8 b_i a_{ij} a_{jk} a_{kl} a_{lm} a_{mn} a_{no} c_o$. We have 42 free parameters to be determined. The row-sum conditions are:

$$c_3 = \sum_{i=1}^2 a_{3i} \tag{5.31}$$

$$c_4 = \sum_{i=1}^3 a_{4i} \tag{5.32}$$

$$c_5 = \sum_{i=1}^4 a_{5i} \tag{5.33}$$

$$c_6 = \sum_{i=1}^5 a_{6i} \tag{5.34}$$

$$c_7 = \sum_{i=1}^6 a_{7i} \tag{5.35}$$

$$c_8 = \sum_{i=1}^7 a_{8i}. \tag{5.36}$$

We want the method to be of order 4 so it has to satisfy the following order conditions: $\tau_1^{(1)}$, $\tau_1^{(2)}$, $\tau_1^{(3)}$, $\tau_2^{(3)}$, $\tau_1^{(4)}$, $\tau_2^{(4)}$, $\tau_3^{(4)}$, $\tau_4^{(4)}$. By looking at Table 5.1 with $p = 4$ and $\bar{m} = 2$ we also have to satisfy $\tau_1^{(5)}$, $\tau_5^{(5)}$, $\tau_1^{(6)}$, and $\tau_7^{(6)}$ to obtain a formula which estimates the first two terms of the global error. We will use the usual row- and column-simplifying assumptions. The parameters were chosen in the following way:

- 1) set $b_2 = 0$ and $c_8 = 1$;
- 2) choose c_2 , c_3 , c_4 , c_5 , c_6 , c_7 and b_8 arbitrarily;
- 3) solve $\tau_1^{(i)}$ with $i = 1, 2, 3, 4, 5$ for b_1 , b_3 , b_4 , b_5 , b_6 and b_7 ;
- 4) choose a_{43} , a_{53} , a_{54} , a_{63} , a_{64} , a_{65} , a_{73} , a_{74} and a_{75} arbitrarily;
- 5) solve $\sum_j a_{ij} c_j = \frac{1}{2} c_i^2$ with $i = 3, \dots, 7$ for a_{i2} ;
- 6) solve $\sum_i b_i a_{ij} = b_j (1 - c_j)$ with $j = 3, \dots, 7$ for a_{8j} ;

5. Global Error Estimation

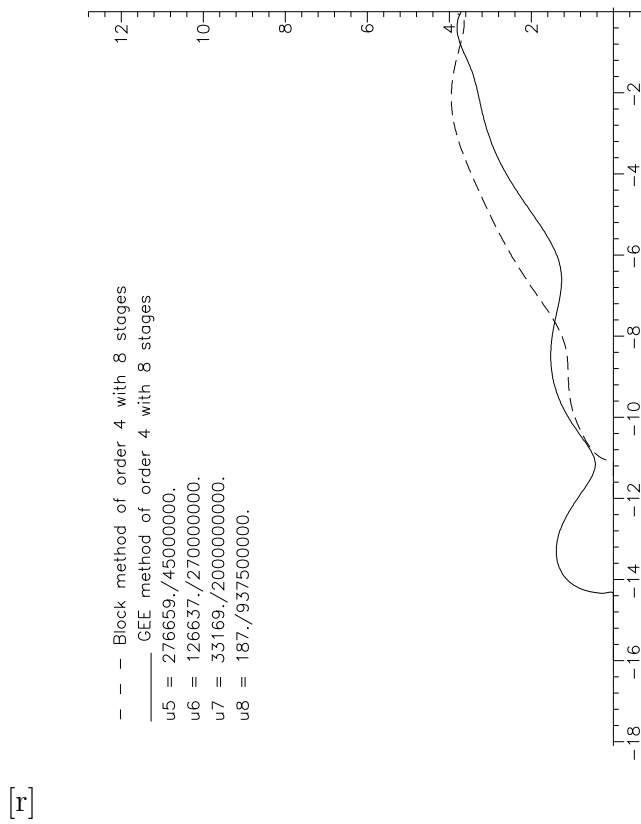


Figure 5.4: Stability region of order 4 with 8 stages

5. Global Error Estimation

$\tau_3^{(5)}, \tau_4^{(5)}, \tau_5^{(5)}, \tau_6^{(5)}, \tau_7^{(5)}, \tau_8^{(5)}, \tau_9^{(5)}$. By looking at Table 5.1 with $p = 5$ and $\bar{m} = 2$ we also have to satisfy $\tau_1^{(6)}, \tau_7^{(6)}, \tau_1^{(7)}$, and $\tau_{11}^{(7)}$ if the global error estimator is to give an asymptotically correct estimate of the first two terms in the global error. We will use the usual row- and column-simplifying assumptions. The parameters were chosen in the following way:

- 1) set $b_2 = 0$ and $c_8 = 1$;
- 2) choose $c_2, c_3, c_4, c_5, c_6, c_7, a_{42}, a_{5,2}, a_{6,2}, a_{54}, a_{64}, a_{65}$ and a_{75} arbitrarily;
- 3) solve $\tau_1^{(i)}$ with $i = 1, 2, 3, 4, 5, 6, 7$ for $b_1, b_3, b_4, b_5, b_6, b_7$ and b_8 ;
- 4) solve $\sum_j a_{ij}c_j = \frac{1}{2}c_i^2$ with $i = 3$ for a_{32} , with $i = 4, 5, 6$ for a_{i3} ;
- 5) solve $\sum_i b_i c_i a_{i2} = 0$ for a_{72} ;
- 6) solve $\sum_{ij} b_i(1 - c_i)a_{ij}c_j^2 = \frac{1}{60}$ for a_{76} ;
- 7) solve $\sum_i b_i a_{ij} = b_j(1 - c_j)$ with $j = 3, \dots, 7$ for a_{8j} ;
- 8) solve $\sum_j a_{ij}c_j = \frac{1}{2}c_i^2$ with $i = 8$ for a_{82} ;
- 9) solve the row-sum conditions for a_{j1} with $j = 1, \dots, 8$;
- 10) evaluate u_6, u_7 and u_8 ;
- 11) try to choose the free coefficients so that the square root of the sum of the squares of $\tau_i^{(6)}$, with $i = 1, 20$ should be as small as possible.

5. Global Error Estimation

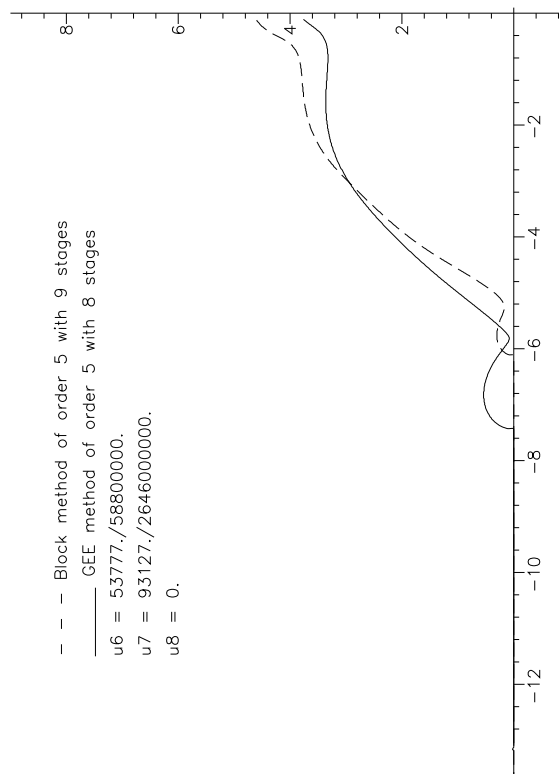
Using this approach we found the following formula:

0	0								
$\frac{1}{20}$	$\frac{1}{20}$	0							
$\frac{1}{10}$	0	$\frac{1}{10}$	0						
$\frac{1}{5}$	0	0	$\frac{1}{5}$	0					
$\frac{2}{5}$	$-\frac{2}{5}$	0	$\frac{4}{5}$	0	0				
$\frac{3}{5}$	$-\frac{3}{5}$	0	1	0	$\frac{1}{5}$	0			
$\frac{4}{5}$	$\frac{31441}{25350}$	$-\frac{1408}{4225}$	$-\frac{26232}{21125}$	$\frac{7}{10}$	$-\frac{3}{10}$	$\frac{93127}{126750}$	0		
1	$-\frac{45477}{15218}$	$\frac{1408}{1087}$	$\frac{142833}{38045}$	$-\frac{7965}{2174}$	$\frac{73719}{15218}$	$-\frac{246621}{76090}$	$\frac{7605}{7609}$	0	
	$\frac{23}{2016}$	0	$\frac{880}{3969}$	$-\frac{25}{2016}$	$\frac{1075}{3024}$	$\frac{65}{1008}$	$\frac{4225}{14112}$	$\frac{1087}{18144}$	

(5.45)

Here $u_6 = \frac{53777}{58800000}$, $u_7 = \frac{93127}{2646000000}$, $u_8 = 0$ and $\sqrt{\left(\sum_{i=1}^{20} \tau_i^{(6)}\right)^2} \cong 0.253 \times 10^{-5}$. The graph of the stability regions of the block method of order 5 (3.52) and of (5.45) is given in Figure 5.5.

In the next chapter we will examine the performance of the various global error estimators derived in this chapter by applying them to some test problems.



[r]

Figure 5.5: Stability region of order 5 with 8 stages

Chapter 6

Numerical Results

In this chapter we present some numerical results to compare certain of the global error estimation algorithms derived in the previous chapters. In particular we will consider the order 4 and 5 formulae derived earlier for the purpose of global error estimation. We have written a code which implements our block formulae and the associated global error estimator. This code runs each algorithm for nine different problems giving statistics concerning the global error and the cost involved. For each of the nine problems we present a table with the tolerance, the end point of integration, the true solution, the global error, the estimated error, the difference and the ratio between the global error and the estimated error, the number of function evaluations, the number of steps and the number of rejected steps involved in the process. The nine problems used are:

Problem 1: $y' = -y, \quad y(0) = 1.$

Problem 2: $y' = -\frac{1}{2}y^3, \quad y(0) = 1.$

Problem 3: $y' = y \cos x, \quad y(0) = 1.$

Problem 4: $y' = \frac{y}{4}(1 - \frac{y}{2}), \quad y(0) = 1.$

Problem 5: $y' = (y - \sin x) + \cos x, \quad y(0) = 0.$

6. Numerical Results

Problem 6:

$$\begin{cases} y_1' = y_2, & y_1(0) = 30 \\ y_2' = \frac{4}{125} - \frac{2}{5}y_2^2, & y_2(0) = 0. \end{cases}$$

Problem 7:

$$\begin{cases} y_1' = y_2, & y_1(0) = 0 \\ y_2' = \frac{\sqrt{1+y_2^2}}{25-x}, & y_2(0) = 0. \end{cases}$$

Problem 8:

$$\begin{cases} y_1' = y_2 - \frac{x^2}{5}, & y_1(0) = 0 \\ y_2' = -y_1 + \frac{2}{5}x, & y_2(0) = 1. \end{cases}$$

Problem 9:

$$\begin{cases} y_1' = y_3, & y_1(0) = 1 \\ y_2' = y_4, & y_2(0) = 0 \\ y_3' = -\frac{y_1}{(\sqrt{y_1^2 + y_2^2})^3}, & y_3(0) = 0 \\ y_4' = -\frac{y_2}{(\sqrt{y_1^2 + y_2^2})^3}, & y_4(0) = 1. \end{cases}$$

6.1 Results for Order 4 with 7 Stages

For the order 4 main formula we have used (3.41) derived in Chapter 3. For the error estimator we have used (5.29) obtained in Chapter 5. The results are given in the following 9 tables for each of the problems.

Problem 1

Tolerance	End Point	Numerical Solution	Global Error	Estimated Error	Difference	Ratio	No. of F. Eval.	No. of Steps	No. of Reject.
1.0×10^{-03}	5.00	$.6717 \times 10^{-02}$	$-.2063 \times 10^{-04}$	$-.1736 \times 10^{-04}$	$.3273 \times 10^{-05}$.84	91	6	0
1.0×10^{-04}	5.00	$.6736 \times 10^{-02}$	$-.1938 \times 10^{-05}$	$-.1746 \times 10^{-05}$	$.1922 \times 10^{-06}$.90	151	10	0
1.0×10^{-05}	5.00	$.6738 \times 10^{-02}$	$-.1546 \times 10^{-06}$	$-.1459 \times 10^{-06}$	$.8789 \times 10^{-08}$.94	241	16	0
1.0×10^{-06}	5.00	$.6738 \times 10^{-02}$	$-.8448 \times 10^{-08}$	$-.8187 \times 10^{-08}$	$.2604 \times 10^{-09}$.97	391	26	0
1.0×10^{-07}	5.00	$.6738 \times 10^{-02}$	$-.5260 \times 10^{-09}$	$-.5170 \times 10^{-09}$	$.9077 \times 10^{-11}$.98	661	44	0
1.0×10^{-08}	5.00	$.6738 \times 10^{-02}$	$-.3335 \times 10^{-10}$	$-.3301 \times 10^{-10}$	$.3346 \times 10^{-12}$.99	1156	77	0
1.0×10^{-09}	5.00	$.6738 \times 10^{-02}$	$-.1968 \times 10^{-11}$	$-.1957 \times 10^{-11}$	$.1118 \times 10^{-13}$.99	2026	135	0
1.0×10^{-10}	5.00	$.6738 \times 10^{-02}$	$-.1132 \times 10^{-12}$	$-.1128 \times 10^{-12}$	$.3742 \times 10^{-15}$	1.00	3571	238	0
1.0×10^{-11}	5.00	$.6738 \times 10^{-02}$	$-.6544 \times 10^{-14}$	$-.6525 \times 10^{-14}$	$.1892 \times 10^{-16}$	1.00	6339	422	1

6. Numerical Results

Problem 2

Tolerance	End Point	Numerical Solution	Global Error	Estimated Error	Difference	Ratio	No. of F. Eval.	No. of Steps	No. of Reject.
1.0×10^{-03}	10.00	$.3015 \times 10^{+00}$	$-.5873 \times 10^{-05}$	$-.6028 \times 10^{-05}$	$-.1553 \times 10^{-06}$	1.03	121	8	0
1.0×10^{-04}	10.00	$.3015 \times 10^{+00}$	$-.5123 \times 10^{-06}$	$-.5214 \times 10^{-06}$	$-.9022 \times 10^{-08}$	1.02	181	12	0
1.0×10^{-05}	10.00	$.3015 \times 10^{+00}$	$-.3733 \times 10^{-07}$	$-.3774 \times 10^{-07}$	$-.4091 \times 10^{-09}$	1.01	286	19	0
1.0×10^{-06}	10.00	$.3015 \times 10^{+00}$	$-.2701 \times 10^{-08}$	$-.2719 \times 10^{-08}$	$-.1795 \times 10^{-10}$	1.01	481	32	0
1.0×10^{-07}	10.00	$.3015 \times 10^{+00}$	$-.1661 \times 10^{-09}$	$-.1667 \times 10^{-09}$	$-.6403 \times 10^{-12}$	1.00	796	53	0
1.0×10^{-08}	10.00	$.3015 \times 10^{+00}$	$-.1025 \times 10^{-10}$	$-.1028 \times 10^{-10}$	$-.2260 \times 10^{-13}$	1.00	1366	91	0
1.0×10^{-09}	10.00	$.3015 \times 10^{+00}$	$-.6067 \times 10^{-12}$	$-.6076 \times 10^{-12}$	$-.8899 \times 10^{-15}$	1.00	2386	159	0
1.0×10^{-10}	10.00	$.3015 \times 10^{+00}$	$-.3531 \times 10^{-13}$	$-.3525 \times 10^{-13}$	$.5727 \times 10^{-16}$	1.00	4209	280	1
1.0×10^{-11}	10.00	$.3015 \times 10^{+00}$	$-.1998 \times 10^{-14}$	$-.2028 \times 10^{-14}$	$-.2973 \times 10^{-16}$	1.01	7449	496	1

Problem 3

Tolerance	End Point	Numerical Solution	Global Error	Estimated Error	Difference	Ratio	No. of F. Eval.	No. of Steps	No. of Reject.
1.0×10^{-03}	20.00	$.2492 \times 10^{+01}$	$.3790 \times 10^{-03}$	$.3676 \times 10^{-03}$	$-.1143 \times 10^{-04}$.97	675	38	13
1.0×10^{-04}	20.00	$.2492 \times 10^{+01}$	$.1291 \times 10^{-04}$	$.1255 \times 10^{-04}$	$-.3625 \times 10^{-06}$.97	1135	66	18
1.0×10^{-05}	20.00	$.2492 \times 10^{+01}$	$-.2531 \times 10^{-06}$	$-.2609 \times 10^{-06}$	$-.7828 \times 10^{-08}$	1.03	1857	112	22
1.0×10^{-06}	20.00	$.2492 \times 10^{+01}$	$-.1062 \times 10^{-06}$	$-.1062 \times 10^{-06}$	$.1041 \times 10^{-10}$	1.00	3095	194	23
1.0×10^{-07}	20.00	$.2492 \times 10^{+01}$	$-.6980 \times 10^{-08}$	$-.6987 \times 10^{-08}$	$-.6969 \times 10^{-11}$	1.00	5278	339	24
1.0×10^{-08}	20.00	$.2492 \times 10^{+01}$	$-.6833 \times 10^{-09}$	$-.6835 \times 10^{-09}$	$-.2329 \times 10^{-12}$	1.00	9187	598	27
1.0×10^{-09}	20.00	$.2492 \times 10^{+01}$	$-.4898 \times 10^{-10}$	$-.4902 \times 10^{-10}$	$-.3992 \times 10^{-13}$	1.00	16080	1057	28
1.0×10^{-10}	20.00	$.2492 \times 10^{+01}$	$-.3751 \times 10^{-11}$	$-.3719 \times 10^{-11}$	$.3217 \times 10^{-13}$.99	28296	1873	25
1.0×10^{-11}	20.00	$.2492 \times 10^{+01}$	$-.3473 \times 10^{-12}$	$-.2646 \times 10^{-12}$	$.8268 \times 10^{-13}$.76	50099	3326	26

Problem 4

Tolerance	End Point	Numerical Solution	Global Error	Estimated Error	Difference	Ratio	No. of F. Eval.	No. of Steps	No. of Reject.
1.0×10^{-03}	20.00	$.1773 \times 10^{+02}$	$-.2210 \times 10^{-03}$	$-.2408 \times 10^{-03}$	$-.1981 \times 10^{-04}$	1.09	122	7	2
1.0×10^{-04}	20.00	$.1773 \times 10^{+02}$	$-.8469 \times 10^{-05}$	$-.9295 \times 10^{-05}$	$-.8254 \times 10^{-06}$	1.10	190	11	3
1.0×10^{-05}	20.00	$.1773 \times 10^{+02}$	$-.5907 \times 10^{-06}$	$-.6416 \times 10^{-06}$	$-.5088 \times 10^{-07}$	1.09	318	19	4
1.0×10^{-06}	20.00	$.1773 \times 10^{+02}$	$-.3723 \times 10^{-07}$	$-.3937 \times 10^{-07}$	$-.2135 \times 10^{-08}$	1.06	521	32	5
1.0×10^{-07}	20.00	$.1773 \times 10^{+02}$	$-.2264 \times 10^{-08}$	$-.2369 \times 10^{-08}$	$-.1047 \times 10^{-09}$	1.05	843	54	4
1.0×10^{-08}	20.00	$.1773 \times 10^{+02}$	$-.1414 \times 10^{-09}$	$-.1454 \times 10^{-09}$	$-.4046 \times 10^{-11}$	1.03	1450	95	3
1.0×10^{-09}	20.00	$.1773 \times 10^{+02}$	$-.8836 \times 10^{-11}$	$-.9086 \times 10^{-11}$	$-.2506 \times 10^{-12}$	1.03	2538	167	4
1.0×10^{-10}	20.00	$.1773 \times 10^{+02}$	$-.5080 \times 10^{-12}$	$-.5116 \times 10^{-12}$	$-.3533 \times 10^{-14}$	1.01	4458	295	4
1.0×10^{-11}	20.00	$.1773 \times 10^{+02}$	$-.3553 \times 10^{-13}$	$-.3052 \times 10^{-13}$	$.5012 \times 10^{-14}$.86	7878	523	4

Problem 5

Tolerance	End Point	Numerical Solution	Global Error	Estimated Error	Difference	Ratio	No. of F. Eval.	No. of Steps	No. of Reject.
1.0×10^{-03}	20.00	$-.6515 \times 10^{+04}$	$-.6516 \times 10^{+04}$	$-.6444 \times 10^{+04}$	$.7197 \times 10^{+02}$.99	506	31	5
1.0×10^{-04}	20.00	$-.4039 \times 10^{+03}$	$-.4049 \times 10^{+03}$	$-.3953 \times 10^{+03}$	$.9589 \times 10^{+01}$.98	868	53	9
1.0×10^{-05}	20.00	$-.2438 \times 10^{+02}$	$-.2529 \times 10^{+02}$	$-.2481 \times 10^{+02}$	$.4744 \times 10^{+00}$.98	1476	93	10
1.0×10^{-06}	20.00	$-.9893 \times 10^{+00}$	$-.1902 \times 10^{+01}$	$-.1877 \times 10^{+01}$	$.2538 \times 10^{-01}$.99	2473	160	9
1.0×10^{-07}	20.00	$.8096 \times 10^{+00}$	$-.1033 \times 10^{+00}$	$-.1025 \times 10^{+00}$	$.8140 \times 10^{-03}$.99	4357	284	12
1.0×10^{-08}	20.00	$.9067 \times 10^{+00}$	$-.6280 \times 10^{-02}$	$-.6248 \times 10^{-02}$	$.3149 \times 10^{-04}$.99	7628	501	14
1.0×10^{-09}	20.00	$.9126 \times 10^{+00}$	$-.3897 \times 10^{-03}$	$-.3884 \times 10^{-03}$	$.1275 \times 10^{-05}$	1.00	13417	888	12
1.0×10^{-10}	20.00	$.9129 \times 10^{+00}$	$-.2464 \times 10^{-04}$	$-.2454 \times 10^{-04}$	$.1035 \times 10^{-06}$	1.00	23745	1576	13
1.0×10^{-11}	20.00	$.9129 \times 10^{+00}$	$-.1591 \times 10^{-05}$	$-.1451 \times 10^{-05}$	$.1404 \times 10^{-06}$.91	42083	2798	14

6. Numerical Results

Problem 6

Tolerance	End Point	Numerical Solution	Global Error	Estimated Error	Difference	Ratio	No. of F. Eval.	No. of Steps	No. of Reject.
1.0×10^{-03}	10.00	$.3134 \times 10^{+02}$ $.2295 \times 10^{+00}$	$-.2048 \times 10^{-03}$ $-.1472 \times 10^{-04}$	$-.2027 \times 10^{-03}$ $-.1499 \times 10^{-04}$	$.2153 \times 10^{-05}$ $-.2730 \times 10^{-06}$.99 1.02	54	3	1
1.0×10^{-04}	10.00	$.3134 \times 10^{+02}$ $.2295 \times 10^{+00}$	$-.1150 \times 10^{-04}$ $-.9778 \times 10^{-06}$	$-.1143 \times 10^{-04}$ $-.1002 \times 10^{-05}$	$.6831 \times 10^{-07}$ $-.2373 \times 10^{-07}$.99 1.02	92	5	2
1.0×10^{-05}	10.00	$.3134 \times 10^{+02}$ $.2295 \times 10^{+00}$	$-.8474 \times 10^{-06}$ $-.5444 \times 10^{-07}$	$-.8445 \times 10^{-06}$ $-.5534 \times 10^{-07}$	$.2896 \times 10^{-08}$ $-.9022 \times 10^{-09}$	1.00 1.02	122	7	2
1.0×10^{-06}	10.00	$.3134 \times 10^{+02}$ $.2295 \times 10^{+00}$	$-.6413 \times 10^{-07}$ $-.3877 \times 10^{-08}$	$-.6390 \times 10^{-07}$ $-.3934 \times 10^{-08}$	$.2308 \times 10^{-09}$ $-.5660 \times 10^{-10}$	1.00 1.01	182	11	2
1.0×10^{-07}	10.00	$.3134 \times 10^{+02}$ $.2295 \times 10^{+00}$	$-.4822 \times 10^{-08}$ $-.2855 \times 10^{-09}$	$-.4802 \times 10^{-08}$ $-.2894 \times 10^{-09}$	$.1934 \times 10^{-10}$ $-.3972 \times 10^{-11}$	1.00 1.01	294	19	1
1.0×10^{-08}	10.00	$.3134 \times 10^{+02}$ $.2295 \times 10^{+00}$	$-.2210 \times 10^{-09}$ $-.1680 \times 10^{-10}$	$-.2203 \times 10^{-09}$ $-.1695 \times 10^{-10}$	$.6536 \times 10^{-12}$ $-.1467 \times 10^{-12}$	1.00 1.01	497	32	2
1.0×10^{-09}	10.00	$.3134 \times 10^{+02}$ $.2295 \times 10^{+00}$	$-.1299 \times 10^{-10}$ $-.9942 \times 10^{-12}$	$-.1296 \times 10^{-10}$ $-.9993 \times 10^{-12}$	$.2990 \times 10^{-13}$ $-.5131 \times 10^{-14}$	1.00 1.01	834	55	1
1.0×10^{-10}	10.00	$.3134 \times 10^{+02}$ $.2295 \times 10^{+00}$	$-.7390 \times 10^{-12}$ $-.5770 \times 10^{-13}$	$-.7289 \times 10^{-12}$ $-.5784 \times 10^{-13}$	$.1002 \times 10^{-13}$ $-.1323 \times 10^{-15}$.99 1.00	1441	96	0
1.0×10^{-11}	10.00	$.3134 \times 10^{+02}$ $.2295 \times 10^{+00}$	$-.4263 \times 10^{-13}$ $-.3414 \times 10^{-14}$	$-.3761 \times 10^{-13}$ $-.3383 \times 10^{-14}$	$.5026 \times 10^{-14}$ $.3126 \times 10^{-16}$.88 .99	2551	170	0

Problem 7

Tolerance	End Point	Numerical Solution	Global Error	Estimated Error	Difference	Ratio	No. of F. Eval.	No. of Steps	No. of Reject.
1.0×10^{-03}	10.00	$.2385 \times 10^{+01}$ $.5333 \times 10^{+00}$	$-.4466 \times 10^{-04}$ $.3817 \times 10^{-05}$	$-.4290 \times 10^{-04}$ $.3762 \times 10^{-05}$	$.1761 \times 10^{-05}$ $-.5463 \times 10^{-07}$.96 .99	61	4	0
1.0×10^{-04}	10.00	$.2385 \times 10^{+01}$ $.5333 \times 10^{+00}$	$-.1462 \times 10^{-05}$ $.1442 \times 10^{-06}$	$-.1431 \times 10^{-05}$ $.1428 \times 10^{-06}$	$.3100 \times 10^{-07}$ $-.1449 \times 10^{-08}$.98 .99	76	5	0
1.0×10^{-05}	10.00	$.2385 \times 10^{+01}$ $.5333 \times 10^{+00}$	$-.1024 \times 10^{-06}$ $.9363 \times 10^{-08}$	$-.1012 \times 10^{-06}$ $.9305 \times 10^{-08}$	$.1206 \times 10^{-08}$ $-.5854 \times 10^{-10}$.99 .99	121	8	0
1.0×10^{-06}	10.00	$.2385 \times 10^{+01}$ $.5333 \times 10^{+00}$	$-.4905 \times 10^{-08}$ $.4665 \times 10^{-09}$	$-.4873 \times 10^{-08}$ $.4648 \times 10^{-09}$	$.3249 \times 10^{-10}$ $-.1708 \times 10^{-11}$.99 1.00	211	14	0
1.0×10^{-07}	10.00	$.2385 \times 10^{+01}$ $.5333 \times 10^{+00}$	$-.2785 \times 10^{-09}$ $.2612 \times 10^{-10}$	$-.2775 \times 10^{-09}$ $.2607 \times 10^{-10}$	$.1024 \times 10^{-11}$ $-.5466 \times 10^{-13}$	1.00 1.00	361	24	0
1.0×10^{-08}	10.00	$.2385 \times 10^{+01}$ $.5333 \times 10^{+00}$	$-.1585 \times 10^{-10}$ $.1472 \times 10^{-11}$	$-.1582 \times 10^{-10}$ $.1470 \times 10^{-11}$	$.3225 \times 10^{-13}$ $-.1774 \times 10^{-14}$	1.00 1.00	631	42	0
1.0×10^{-09}	10.00	$.2385 \times 10^{+01}$ $.5333 \times 10^{+00}$	$-.8984 \times 10^{-12}$ $.8293 \times 10^{-13}$	$-.8982 \times 10^{-12}$ $.8288 \times 10^{-13}$	$.1801 \times 10^{-15}$ $-.5602 \times 10^{-16}$	1.00 1.00	1111	74	0
1.0×10^{-10}	10.00	$.2385 \times 10^{+01}$ $.5333 \times 10^{+00}$	$-.5107 \times 10^{-13}$ $.4774 \times 10^{-14}$	$-.5081 \times 10^{-13}$ $.4678 \times 10^{-14}$	$.2647 \times 10^{-15}$ $-.9641 \times 10^{-16}$.99 .98	1951	130	0
1.0×10^{-11}	10.00	$.2385 \times 10^{+01}$ $.5333 \times 10^{+00}$	$-.3109 \times 10^{-14}$ $.1110 \times 10^{-15}$	$-.2782 \times 10^{-14}$ $.2666 \times 10^{-15}$	$.3269 \times 10^{-15}$ $.1556 \times 10^{-15}$.89 2.40	3466	231	0

Problem 8

Tolerance	End Point	Numerical Solution	Global Error	Estimated Error	Difference	Ratio	No. of F. Eval.	No. of Steps	No. of Reject.
1.0×10^{-03}	10.00	$-.5441 \times 10^{+00}$ $.1916 \times 10^{+02}$	$-.1007 \times 10^{-03}$ $-.5662 \times 10^{-03}$	$-.1406 \times 10^{-03}$ $-.5603 \times 10^{-03}$	$-.3984 \times 10^{-04}$ $.5854 \times 10^{-05}$	1.40 .99	227	14	2
1.0×10^{-04}	10.00	$-.5440 \times 10^{+00}$ $.1916 \times 10^{+02}$	$-.1273 \times 10^{-04}$ $-.3636 \times 10^{-04}$	$-.1438 \times 10^{-04}$ $-.3581 \times 10^{-04}$	$-.1654 \times 10^{-05}$ $.5505 \times 10^{-06}$	1.13 .98	369	24	1
1.0×10^{-05}	10.00	$-.5440 \times 10^{+00}$ $.1916 \times 10^{+02}$	$-.1046 \times 10^{-05}$ $-.2288 \times 10^{-05}$	$-.1107 \times 10^{-05}$ $-.2259 \times 10^{-05}$	$-.6079 \times 10^{-07}$ $.2883 \times 10^{-07}$	1.06 .99	632	41	2
1.0×10^{-06}	10.00	$-.5440 \times 10^{+00}$ $.1916 \times 10^{+02}$	$-.7012 \times 10^{-07}$ $-.1325 \times 10^{-06}$	$-.7212 \times 10^{-07}$ $-.1313 \times 10^{-06}$	$-.2006 \times 10^{-08}$ $.1137 \times 10^{-08}$	1.03 .99	1090	71	3
1.0×10^{-07}	10.00	$-.5440 \times 10^{+00}$ $.1916 \times 10^{+02}$	$-.4484 \times 10^{-08}$ $-.7835 \times 10^{-08}$	$-.4553 \times 10^{-08}$ $-.7792 \times 10^{-08}$	$-.6948 \times 10^{-10}$ $.4305 \times 10^{-10}$	1.02 .99	1877	124	2
1.0×10^{-08}	10.00	$-.5440 \times 10^{+00}$ $.1916 \times 10^{+02}$	$-.2847 \times 10^{-09}$ $-.4753 \times 10^{-09}$	$-.2874 \times 10^{-09}$ $-.4736 \times 10^{-09}$	$-.2700 \times 10^{-11}$ $.1687 \times 10^{-11}$	1.01 1.00	3294	219	1
1.0×10^{-09}	10.00	$-.5440 \times 10^{+00}$ $.1916 \times 10^{+02}$	$-.1652 \times 10^{-10}$ $-.2692 \times 10^{-10}$	$-.1661 \times 10^{-10}$ $-.2685 \times 10^{-10}$	$-.8963 \times 10^{-13}$ $.7053 \times 10^{-13}$	1.01 1.00	5829	388	1
1.0×10^{-10}	10.00	$-.5440 \times 10^{+00}$ $.1916 \times 10^{+02}$	$-.9331 \times 10^{-12}$ $-.1492 \times 10^{-11}$	$-.9521 \times 10^{-12}$ $-.1523 \times 10^{-11}$	$-.1892 \times 10^{-13}$ $-.3082 \times 10^{-13}$	1.02 1.02	10329	688	1
1.0×10^{-11}	10.00	$-.5440 \times 10^{+00}$ $.1916 \times 10^{+02}$	$-.7538 \times 10^{-13}$ $-.6750 \times 10^{-13}$	$-.5453 \times 10^{-13}$ $-.8633 \times 10^{-13}$	$.2085 \times 10^{-13}$ $-.1883 \times 10^{-13}$.72 1.28	18339	1222	1

6. Numerical Results

Problem 9

Tolerance	End Point	Numerical Solution	Global Error	Estimated Error	Difference	Ratio	No. of F. Eval.	No. of Steps	No. of Reject.
1.0×10^{-03}	20.00	$.2519 \times 10^{+00}$	$-.3202 \times 10^{-01}$	$-.3169 \times 10^{-01}$	$-.3221 \times 10^{-03}$.99	581	36	5
		$.9329 \times 10^{+00}$	$-.9846 \times 10^{-02}$	$-.9755 \times 10^{-02}$	$.9100 \times 10^{-04}$.99			
		$-.9697 \times 10^{+00}$	$.9095 \times 10^{-02}$	$.9032 \times 10^{-02}$	$-.6310 \times 10^{-04}$.99			
1.0×10^{-04}	20.00	$.3631 \times 10^{+00}$	$.3432 \times 10^{-01}$	$.3372 \times 10^{-01}$	$-.6025 \times 10^{-03}$.98	939	62	1
		$.2214 \times 10^{+00}$	$.1533 \times 10^{-02}$	$.1539 \times 10^{-02}$	$.6072 \times 10^{-05}$	1.00			
		$.9423 \times 10^{+00}$	$-.4419 \times 10^{-03}$	$-.4451 \times 10^{-03}$	$-.3213 \times 10^{-05}$	1.01			
1.0×10^{-05}	20.00	$-.9784 \times 10^{+00}$	$.4133 \times 10^{-03}$	$.4136 \times 10^{-03}$	$.3274 \times 10^{-06}$	1.00	1659	110	1
		$.3305 \times 10^{+00}$	$.1659 \times 10^{-02}$	$.1664 \times 10^{-02}$	$.4630 \times 10^{-05}$	1.00			
		$.2200 \times 10^{+00}$	$.7610 \times 10^{-04}$	$.7641 \times 10^{-04}$	$.3070 \times 10^{-06}$	1.00			
1.0×10^{-06}	20.00	$.9427 \times 10^{+00}$	$-.2181 \times 10^{-04}$	$-.2193 \times 10^{-04}$	$-.1279 \times 10^{-06}$	1.01	2934	195	1
		$-.9787 \times 10^{+00}$	$.2063 \times 10^{-04}$	$.2068 \times 10^{-04}$	$.4698 \times 10^{-07}$	1.00			
		$.3289 \times 10^{+00}$	$.8277 \times 10^{-04}$	$.8308 \times 10^{-04}$	$.3144 \times 10^{-06}$	1.00			
1.0×10^{-07}	20.00	$.2199 \times 10^{+00}$	$.4078 \times 10^{-05}$	$.4087 \times 10^{-05}$	$.9629 \times 10^{-08}$	1.00	5214	347	1
		$.9427 \times 10^{+00}$	$-.1161 \times 10^{-05}$	$-.1165 \times 10^{-05}$	$-.4016 \times 10^{-08}$	1.00			
		$-.9788 \times 10^{+00}$	$.1114 \times 10^{-05}$	$.1116 \times 10^{-05}$	$.1509 \times 10^{-08}$	1.00			
1.0×10^{-08}	20.00	$.3288 \times 10^{+00}$	$.4444 \times 10^{-05}$	$.4454 \times 10^{-05}$	$.1024 \times 10^{-07}$	1.00	9264	617	1
		$.2199 \times 10^{+00}$	$.2253 \times 10^{-06}$	$.2256 \times 10^{-06}$	$.2896 \times 10^{-09}$	1.00			
		$.9427 \times 10^{+00}$	$-.6384 \times 10^{-07}$	$-.6397 \times 10^{-07}$	$-.1237 \times 10^{-09}$	1.00			
1.0×10^{-09}	20.00	$-.9788 \times 10^{+00}$	$.6188 \times 10^{-07}$	$.6193 \times 10^{-07}$	$.4376 \times 10^{-10}$	1.00	16464	1097	1
		$.3288 \times 10^{+00}$	$.2456 \times 10^{-06}$	$.2459 \times 10^{-06}$	$.3127 \times 10^{-09}$	1.00			
		$.2199 \times 10^{+00}$	$.1259 \times 10^{-07}$	$.1260 \times 10^{-07}$	$.8740 \times 10^{-11}$	1.00			
1.0×10^{-10}	20.00	$.9427 \times 10^{+00}$	$-.3556 \times 10^{-08}$	$-.3560 \times 10^{-08}$	$-.3818 \times 10^{-11}$	1.00	29259	1950	1
		$-.9788 \times 10^{+00}$	$.3468 \times 10^{-08}$	$.3469 \times 10^{-08}$	$.1266 \times 10^{-11}$	1.00			
		$.3288 \times 10^{+00}$	$.1372 \times 10^{-07}$	$.1373 \times 10^{-07}$	$.9524 \times 10^{-11}$	1.00			
1.0×10^{-11}	20.00	$.2199 \times 10^{+00}$	$.7059 \times 10^{-09}$	$.7063 \times 10^{-09}$	$.3600 \times 10^{-12}$	1.00	52014	3467	1
		$.9427 \times 10^{+00}$	$-.1991 \times 10^{-09}$	$-.1992 \times 10^{-09}$	$-.1455 \times 10^{-12}$	1.00			
		$-.9788 \times 10^{+00}$	$.1948 \times 10^{-09}$	$.1949 \times 10^{-09}$	$.6278 \times 10^{-13}$	1.00			
1.0×10^{-10}	20.00	$.3288 \times 10^{+00}$	$.7696 \times 10^{-09}$	$.7700 \times 10^{-09}$	$.3906 \times 10^{-12}$	1.00	29259	1950	1
		$.2199 \times 10^{+00}$	$.3955 \times 10^{-10}$	$.3966 \times 10^{-10}$	$.1144 \times 10^{-12}$	1.00			
		$.9427 \times 10^{+00}$	$-.1114 \times 10^{-10}$	$-.1117 \times 10^{-10}$	$-.3023 \times 10^{-13}$	1.00			
1.0×10^{-11}	20.00	$-.9788 \times 10^{+00}$	$.1092 \times 10^{-10}$	$.1096 \times 10^{-10}$	$.3262 \times 10^{-13}$	1.00	29259	1950	1
		$.3288 \times 10^{+00}$	$.4312 \times 10^{-10}$	$.4324 \times 10^{-10}$	$.1184 \times 10^{-12}$	1.00			
		$.2199 \times 10^{+00}$	$.2296 \times 10^{-11}$	$.2224 \times 10^{-11}$	$-.7208 \times 10^{-13}$.97			
1.0×10^{-11}	20.00	$.9427 \times 10^{+00}$	$-.6526 \times 10^{-12}$	$-.6262 \times 10^{-12}$	$.2644 \times 10^{-13}$.96	52014	3467	1
		$-.9788 \times 10^{+00}$	$.6311 \times 10^{-12}$	$.6148 \times 10^{-12}$	$-.1628 \times 10^{-13}$.97			
		$.3288 \times 10^{+00}$	$.2506 \times 10^{-11}$	$.2425 \times 10^{-11}$	$-.8140 \times 10^{-13}$.97			

6.2 Results for Order 4 with 8 Stages

For the order 4 main formula we have used (3.41) derived in Chapter 3. For the error estimator we have used (5.37) obtained in Chapter 5. The results are given in the following 9 tables for each of the problems.

Problem 1

Tolerance	End Point	Numerical Solution	Global Error	Estimated Error	Difference	Ratio	No. of F. Eval.	No. of Steps	No. of Reject.
1.0×10^{-03}	5.00	$.6717 \times 10^{-02}$	$-.2063 \times 10^{-04}$	$-.1831 \times 10^{-04}$	$.2316 \times 10^{-05}$.89	97	6	0
1.0×10^{-04}	5.00	$.6736 \times 10^{-02}$	$-.1938 \times 10^{-05}$	$-.1786 \times 10^{-05}$	$.1525 \times 10^{-06}$.92	161	10	0
1.0×10^{-05}	5.00	$.6738 \times 10^{-02}$	$-.1546 \times 10^{-06}$	$-.1469 \times 10^{-06}$	$.7738 \times 10^{-08}$.95	257	16	0
1.0×10^{-06}	5.00	$.6738 \times 10^{-02}$	$-.8448 \times 10^{-08}$	$-.8200 \times 10^{-08}$	$.2479 \times 10^{-09}$.97	417	26	0
1.0×10^{-07}	5.00	$.6738 \times 10^{-02}$	$-.5260 \times 10^{-09}$	$-.5170 \times 10^{-09}$	$.9033 \times 10^{-11}$.98	705	44	0
1.0×10^{-08}	5.00	$.6738 \times 10^{-02}$	$-.3335 \times 10^{-10}$	$-.3301 \times 10^{-10}$	$.3411 \times 10^{-12}$.99	1233	77	0
1.0×10^{-09}	5.00	$.6738 \times 10^{-02}$	$-.1968 \times 10^{-11}$	$-.1957 \times 10^{-11}$	$.1157 \times 10^{-13}$.99	2161	135	0
1.0×10^{-10}	5.00	$.6738 \times 10^{-02}$	$-.1132 \times 10^{-12}$	$-.1128 \times 10^{-12}$	$.3920 \times 10^{-15}$	1.00	3809	238	0
1.0×10^{-11}	5.00	$.6738 \times 10^{-02}$	$-.6544 \times 10^{-14}$	$-.6523 \times 10^{-14}$	$.2167 \times 10^{-16}$	1.00	6761	422	1

6. Numerical Results

Problem 2

Tolerance	End Point	Numerical Solution	Global Error	Estimated Error	Difference	Ratio	No. of F. Eval.	No. of Steps	No. of Reject.
1.0×10^{-03}	10.00	$.3015 \times 10^{+00}$	$-.5873 \times 10^{-05}$	$-.6098 \times 10^{-05}$	$-.2249 \times 10^{-06}$	1.04	129	8	0
1.0×10^{-04}	10.00	$.3015 \times 10^{+00}$	$-.5123 \times 10^{-06}$	$-.5251 \times 10^{-06}$	$-.1274 \times 10^{-07}$	1.02	193	12	0
1.0×10^{-05}	10.00	$.3015 \times 10^{+00}$	$-.3733 \times 10^{-07}$	$-.3790 \times 10^{-07}$	$-.5656 \times 10^{-09}$	1.02	305	19	0
1.0×10^{-06}	10.00	$.3015 \times 10^{+00}$	$-.2701 \times 10^{-08}$	$-.2725 \times 10^{-08}$	$-.2442 \times 10^{-10}$	1.01	513	32	0
1.0×10^{-07}	10.00	$.3015 \times 10^{+00}$	$-.1661 \times 10^{-09}$	$-.1669 \times 10^{-09}$	$-.8613 \times 10^{-12}$	1.01	849	53	0
1.0×10^{-08}	10.00	$.3015 \times 10^{+00}$	$-.1025 \times 10^{-10}$	$-.1029 \times 10^{-10}$	$-.3020 \times 10^{-13}$	1.00	1457	91	0
1.0×10^{-09}	10.00	$.3015 \times 10^{+00}$	$-.6067 \times 10^{-12}$	$-.6078 \times 10^{-12}$	$-.1136 \times 10^{-14}$	1.00	2545	159	0
1.0×10^{-10}	10.00	$.3015 \times 10^{+00}$	$-.3531 \times 10^{-13}$	$-.3525 \times 10^{-13}$	$.5350 \times 10^{-16}$	1.00	4489	280	1
1.0×10^{-11}	10.00	$.3015 \times 10^{+00}$	$-.1998 \times 10^{-14}$	$-.2016 \times 10^{-14}$	$-.1741 \times 10^{-16}$	1.01	7945	496	1

Problem 3

Tolerance	End Point	Numerical Solution	Global Error	Estimated Error	Difference	Ratio	No. of F. Eval.	No. of Steps	No. of Reject.
1.0×10^{-03}	20.00	$.2492 \times 10^{+01}$	$.3790 \times 10^{-03}$	$.3600 \times 10^{-03}$	$-.1902 \times 10^{-04}$.95	713	38	13
1.0×10^{-04}	20.00	$.2492 \times 10^{+01}$	$.1291 \times 10^{-04}$	$.1236 \times 10^{-04}$	$-.5548 \times 10^{-06}$.96	1201	66	18
1.0×10^{-05}	20.00	$.2492 \times 10^{+01}$	$-.2531 \times 10^{-06}$	$-.2640 \times 10^{-06}$	$-.1088 \times 10^{-07}$	1.04	1969	112	22
1.0×10^{-06}	20.00	$.2492 \times 10^{+01}$	$-.1062 \times 10^{-06}$	$-.1062 \times 10^{-06}$	$-.2520 \times 10^{-11}$	1.00	3289	194	23
1.0×10^{-07}	20.00	$.2492 \times 10^{+01}$	$-.6980 \times 10^{-08}$	$-.6989 \times 10^{-08}$	$-.9570 \times 10^{-11}$	1.00	5617	339	24
1.0×10^{-08}	20.00	$.2492 \times 10^{+01}$	$-.6833 \times 10^{-09}$	$-.6836 \times 10^{-09}$	$-.3107 \times 10^{-12}$	1.00	9785	598	27
1.0×10^{-09}	20.00	$.2492 \times 10^{+01}$	$-.4898 \times 10^{-10}$	$-.4902 \times 10^{-10}$	$-.4101 \times 10^{-13}$	1.00	17137	1057	28
1.0×10^{-10}	20.00	$.2492 \times 10^{+01}$	$-.3751 \times 10^{-11}$	$-.3719 \times 10^{-11}$	$.3201 \times 10^{-13}$.99	30169	1873	25
1.0×10^{-11}	20.00	$.2492 \times 10^{+01}$	$-.3473 \times 10^{-12}$	$-.2648 \times 10^{-12}$	$.8244 \times 10^{-13}$.76	53425	3326	26

Problem 4

Tolerance	End Point	Numerical Solution	Global Error	Estimated Error	Difference	Ratio	No. of F. Eval.	No. of Steps	No. of Reject.
1.0×10^{-03}	20.00	$.1773 \times 10^{+02}$	$-.2210 \times 10^{-03}$	$-.2444 \times 10^{-03}$	$-.2342 \times 10^{-04}$	1.11	129	7	2
1.0×10^{-04}	20.00	$.1773 \times 10^{+02}$	$-.8469 \times 10^{-05}$	$-.9353 \times 10^{-05}$	$-.8832 \times 10^{-06}$	1.10	201	11	3
1.0×10^{-05}	20.00	$.1773 \times 10^{+02}$	$-.5907 \times 10^{-06}$	$-.6462 \times 10^{-06}$	$-.5547 \times 10^{-07}$	1.09	337	19	4
1.0×10^{-06}	20.00	$.1773 \times 10^{+02}$	$-.3723 \times 10^{-07}$	$-.3956 \times 10^{-07}$	$-.2334 \times 10^{-08}$	1.06	553	32	5
1.0×10^{-07}	20.00	$.1773 \times 10^{+02}$	$-.2264 \times 10^{-08}$	$-.2378 \times 10^{-08}$	$-.1139 \times 10^{-09}$	1.05	897	54	4
1.0×10^{-08}	20.00	$.1773 \times 10^{+02}$	$-.1414 \times 10^{-09}$	$-.1457 \times 10^{-09}$	$-.4397 \times 10^{-11}$	1.03	1545	95	3
1.0×10^{-09}	20.00	$.1773 \times 10^{+02}$	$-.8836 \times 10^{-11}$	$-.9110 \times 10^{-11}$	$-.2748 \times 10^{-12}$	1.03	2705	167	4
1.0×10^{-10}	20.00	$.1773 \times 10^{+02}$	$-.5080 \times 10^{-12}$	$-.5129 \times 10^{-12}$	$-.4911 \times 10^{-14}$	1.01	4753	295	4
1.0×10^{-11}	20.00	$.1773 \times 10^{+02}$	$-.3553 \times 10^{-13}$	$-.3122 \times 10^{-13}$	$.4307 \times 10^{-14}$.88	8401	523	4

Problem 5

Tolerance	End Point	Numerical Solution	Global Error	Estimated Error	Difference	Ratio	No. of F. Eval.	No. of Steps	No. of Reject.
1.0×10^{-03}	20.00	$-.6515 \times 10^{+04}$	$-.6516 \times 10^{+04}$	$-.6596 \times 10^{+04}$	$-.7952 \times 10^{+02}$	1.01	537	31	5
1.0×10^{-04}	20.00	$-.4039 \times 10^{+03}$	$-.4049 \times 10^{+03}$	$-.3979 \times 10^{+03}$	$.6929 \times 10^{+01}$.98	921	53	9
1.0×10^{-05}	20.00	$-.2438 \times 10^{+02}$	$-.2529 \times 10^{+02}$	$-.2485 \times 10^{+02}$	$.4355 \times 10^{+00}$.98	1569	93	10
1.0×10^{-06}	20.00	$-.9893 \times 10^{+00}$	$-.1902 \times 10^{+01}$	$-.1877 \times 10^{+01}$	$.2487 \times 10^{-01}$.99	2633	160	9
1.0×10^{-07}	20.00	$.8096 \times 10^{+00}$	$-.1033 \times 10^{+00}$	$-.1025 \times 10^{+00}$	$.8304 \times 10^{-03}$.99	4641	284	12
1.0×10^{-08}	20.00	$.9067 \times 10^{+00}$	$-.6280 \times 10^{-02}$	$-.6247 \times 10^{-02}$	$.3274 \times 10^{-04}$.99	8129	501	14
1.0×10^{-09}	20.00	$.9126 \times 10^{+00}$	$-.3897 \times 10^{-03}$	$-.3883 \times 10^{-03}$	$.1332 \times 10^{-05}$	1.00	14305	888	12
1.0×10^{-10}	20.00	$.9129 \times 10^{+00}$	$-.2464 \times 10^{-04}$	$-.2455 \times 10^{-04}$	$.9237 \times 10^{-07}$	1.00	25321	1576	13
1.0×10^{-11}	20.00	$.9129 \times 10^{+00}$	$-.1591 \times 10^{-05}$	$-.1467 \times 10^{-05}$	$.1240 \times 10^{-06}$.92	44881	2798	14

6. Numerical Results

Problem 6

Tolerance	End Point	Numerical Solution	Global Error	Estimated Error	Difference	Ratio	No. of F. Eval.	No. of Steps	No. of Reject.
1.0×10^{-03}	10.00	$.3134 \times 10^{+02}$ $.2295 \times 10^{+00}$	$-.2048 \times 10^{-03}$ $-.1472 \times 10^{-04}$	$-.2010 \times 10^{-03}$ $-.1535 \times 10^{-04}$	$.3868 \times 10^{-05}$ $-.6292 \times 10^{-06}$.98 1.04	57	3	1
1.0×10^{-04}	10.00	$.3134 \times 10^{+02}$ $.2295 \times 10^{+00}$	$-.1150 \times 10^{-04}$ $-.9778 \times 10^{-06}$	$-.1139 \times 10^{-04}$ $-.1014 \times 10^{-05}$	$.1077 \times 10^{-06}$ $-.3585 \times 10^{-07}$.99 1.04	97	5	2
1.0×10^{-05}	10.00	$.3134 \times 10^{+02}$ $.2295 \times 10^{+00}$	$-.8474 \times 10^{-06}$ $-.5444 \times 10^{-07}$	$-.8437 \times 10^{-06}$ $-.5558 \times 10^{-07}$	$.3688 \times 10^{-08}$ $-.1142 \times 10^{-08}$	1.00 1.02	129	7	2
1.0×10^{-06}	10.00	$.3134 \times 10^{+02}$ $.2295 \times 10^{+00}$	$-.6413 \times 10^{-07}$ $-.3877 \times 10^{-08}$	$-.6385 \times 10^{-07}$ $-.3943 \times 10^{-08}$	$.2756 \times 10^{-09}$ $-.6651 \times 10^{-10}$	1.00 1.02	193	11	2
1.0×10^{-07}	10.00	$.3134 \times 10^{+02}$ $.2295 \times 10^{+00}$	$-.4822 \times 10^{-08}$ $-.2855 \times 10^{-09}$	$-.4799 \times 10^{-08}$ $-.2900 \times 10^{-09}$	$.2274 \times 10^{-10}$ $-.4530 \times 10^{-11}$	1.00 1.02	313	19	1
1.0×10^{-08}	10.00	$.3134 \times 10^{+02}$ $.2295 \times 10^{+00}$	$-.2210 \times 10^{-09}$ $-.1680 \times 10^{-10}$	$-.2202 \times 10^{-09}$ $-.1697 \times 10^{-10}$	$.7503 \times 10^{-12}$ $-.1650 \times 10^{-12}$	1.00 1.01	529	32	2
1.0×10^{-09}	10.00	$.3134 \times 10^{+02}$ $.2295 \times 10^{+00}$	$-.1299 \times 10^{-10}$ $-.9942 \times 10^{-12}$	$-.1295 \times 10^{-10}$ $-.9999 \times 10^{-12}$	$.3296 \times 10^{-13}$ $-.5723 \times 10^{-14}$	1.00 1.01	889	55	1
1.0×10^{-10}	10.00	$.3134 \times 10^{+02}$ $.2295 \times 10^{+00}$	$-.7390 \times 10^{-12}$ $-.5770 \times 10^{-13}$	$-.7290 \times 10^{-12}$ $-.5786 \times 10^{-13}$	$.9961 \times 10^{-14}$ $-.1589 \times 10^{-15}$.99 1.00	1537	96	0
1.0×10^{-11}	10.00	$.3134 \times 10^{+02}$ $.2295 \times 10^{+00}$	$-.4263 \times 10^{-13}$ $-.3414 \times 10^{-14}$	$-.3776 \times 10^{-13}$ $-.3390 \times 10^{-14}$	$.4870 \times 10^{-14}$ $.2369 \times 10^{-16}$.89 .99	2721	170	0

Problem 7

Tolerance	End Point	Numerical Solution	Global Error	Estimated Error	Difference	Ratio	No. of F. Eval.	No. of Steps	No. of Reject.
1.0×10^{-03}	10.00	$.2385 \times 10^{+01}$ $.5333 \times 10^{+00}$	$-.4466 \times 10^{-04}$ $.3817 \times 10^{-05}$	$-.4233 \times 10^{-04}$ $.3746 \times 10^{-05}$	$.2335 \times 10^{-05}$ $-.7103 \times 10^{-07}$.95 .98	65	4	0
1.0×10^{-04}	10.00	$.2385 \times 10^{+01}$ $.5333 \times 10^{+00}$	$-.1462 \times 10^{-05}$ $.1442 \times 10^{-06}$	$-.1421 \times 10^{-05}$ $.1423 \times 10^{-06}$	$.4084 \times 10^{-07}$ $-.1900 \times 10^{-08}$.97 .99	81	5	0
1.0×10^{-05}	10.00	$.2385 \times 10^{+01}$ $.5333 \times 10^{+00}$	$-.1024 \times 10^{-06}$ $.9363 \times 10^{-08}$	$-.1008 \times 10^{-06}$ $.9287 \times 10^{-08}$	$.1581 \times 10^{-08}$ $-.7637 \times 10^{-10}$.98 .99	129	8	0
1.0×10^{-06}	10.00	$.2385 \times 10^{+01}$ $.5333 \times 10^{+00}$	$-.4905 \times 10^{-08}$ $.4665 \times 10^{-09}$	$-.4863 \times 10^{-08}$ $.4642 \times 10^{-09}$	$.4248 \times 10^{-10}$ $-.2225 \times 10^{-11}$.99 1.00	225	14	0
1.0×10^{-07}	10.00	$.2385 \times 10^{+01}$ $.5333 \times 10^{+00}$	$-.2785 \times 10^{-09}$ $.2612 \times 10^{-10}$	$-.2772 \times 10^{-09}$ $.2605 \times 10^{-10}$	$.1336 \times 10^{-11}$ $-.7126 \times 10^{-13}$	1.00 1.00	385	24	0
1.0×10^{-08}	10.00	$.2385 \times 10^{+01}$ $.5333 \times 10^{+00}$	$-.1585 \times 10^{-10}$ $.1472 \times 10^{-11}$	$-.1581 \times 10^{-10}$ $.1469 \times 10^{-11}$	$.4178 \times 10^{-13}$ $-.2338 \times 10^{-14}$	1.00 1.00	673	42	0
1.0×10^{-09}	10.00	$.2385 \times 10^{+01}$ $.5333 \times 10^{+00}$	$-.8984 \times 10^{-12}$ $.8293 \times 10^{-13}$	$-.8983 \times 10^{-12}$ $.8282 \times 10^{-13}$	$.1165 \times 10^{-15}$ $-.1100 \times 10^{-15}$	1.00 1.00	1185	74	0
1.0×10^{-10}	10.00	$.2385 \times 10^{+01}$ $.5333 \times 10^{+00}$	$-.5107 \times 10^{-13}$ $.4774 \times 10^{-14}$	$-.5115 \times 10^{-13}$ $.4633 \times 10^{-14}$	$-.8098 \times 10^{-16}$ $-.1411 \times 10^{-15}$	1.00 .97	2081	130	0
1.0×10^{-11}	10.00	$.2385 \times 10^{+01}$ $.5333 \times 10^{+00}$	$-.3109 \times 10^{-14}$ $.1110 \times 10^{-15}$	$-.3099 \times 10^{-14}$ $.2278 \times 10^{-15}$	$.9945 \times 10^{-17}$ $.1168 \times 10^{-15}$	1.00 2.05	3697	231	0

Problem 8

Tolerance	End Point	Numerical Solution	Global Error	Estimated Error	Difference	Ratio	No. of F. Eval.	No. of Steps	No. of Reject.
1.0×10^{-03}	10.00	$-.5441 \times 10^{+00}$ $.1916 \times 10^{+02}$	$-.1007 \times 10^{-03}$ $-.5662 \times 10^{-03}$	$-.1428 \times 10^{-03}$ $-.5485 \times 10^{-03}$	$-.4204 \times 10^{-04}$ $.1770 \times 10^{-04}$	1.42 .97	241	14	2
1.0×10^{-04}	10.00	$-.5440 \times 10^{+00}$ $.1916 \times 10^{+02}$	$-.1273 \times 10^{-04}$ $-.3636 \times 10^{-04}$	$-.1441 \times 10^{-04}$ $-.3552 \times 10^{-04}$	$-.1682 \times 10^{-05}$ $.8426 \times 10^{-06}$	1.13 .98	393	24	1
1.0×10^{-05}	10.00	$-.5440 \times 10^{+00}$ $.1916 \times 10^{+02}$	$-.1046 \times 10^{-05}$ $-.2288 \times 10^{-05}$	$-.1108 \times 10^{-05}$ $-.2252 \times 10^{-05}$	$-.6194 \times 10^{-07}$ $.3626 \times 10^{-07}$	1.06 .98	673	41	2
1.0×10^{-06}	10.00	$-.5440 \times 10^{+00}$ $.1916 \times 10^{+02}$	$-.7012 \times 10^{-07}$ $-.1325 \times 10^{-06}$	$-.7218 \times 10^{-07}$ $-.1312 \times 10^{-06}$	$-.2064 \times 10^{-08}$ $.1311 \times 10^{-08}$	1.03 .99	1161	71	3
1.0×10^{-07}	10.00	$-.5440 \times 10^{+00}$ $.1916 \times 10^{+02}$	$-.4484 \times 10^{-08}$ $-.7835 \times 10^{-08}$	$-.4556 \times 10^{-08}$ $-.7788 \times 10^{-08}$	$-.7219 \times 10^{-10}$ $.4762 \times 10^{-10}$	1.02 .99	2001	124	2
1.0×10^{-08}	10.00	$-.5440 \times 10^{+00}$ $.1916 \times 10^{+02}$	$-.2847 \times 10^{-09}$ $-.4753 \times 10^{-09}$	$-.2875 \times 10^{-09}$ $-.4735 \times 10^{-09}$	$-.2826 \times 10^{-11}$ $.1829 \times 10^{-11}$	1.01 1.00	3513	219	1
1.0×10^{-09}	10.00	$-.5440 \times 10^{+00}$ $.1916 \times 10^{+02}$	$-.1652 \times 10^{-10}$ $-.2692 \times 10^{-10}$	$-.1661 \times 10^{-10}$ $-.2685 \times 10^{-10}$	$-.9376 \times 10^{-13}$ $.7381 \times 10^{-13}$	1.01 1.00	6217	388	1
1.0×10^{-10}	10.00	$-.5440 \times 10^{+00}$ $.1916 \times 10^{+02}$	$-.9331 \times 10^{-12}$ $-.1492 \times 10^{-11}$	$-.9521 \times 10^{-12}$ $-.1523 \times 10^{-11}$	$-.1900 \times 10^{-13}$ $-.3110 \times 10^{-13}$	1.02 1.02	11017	688	1
1.0×10^{-11}	10.00	$-.5440 \times 10^{+00}$ $.1916 \times 10^{+02}$	$-.7538 \times 10^{-13}$ $-.6750 \times 10^{-13}$	$-.5434 \times 10^{-13}$ $-.8675 \times 10^{-13}$	$.2105 \times 10^{-13}$ $-.1925 \times 10^{-13}$.72 1.29	19561	1222	1

6. Numerical Results

Problem 9

Tolerance	End Point	Numerical Solution	Global Error	Estimated Error	Difference	Ratio	No. of F. Eval.	No. of Steps	No. of Reject.
1.0×10^{-03}	20.00	$.2519 \times 10^{+00}$	$.3202 \times 10^{-01}$	$.3119 \times 10^{-01}$	$-.8290 \times 10^{-03}$.97	617	36	5
		$.9329 \times 10^{+00}$	$-.9846 \times 10^{-02}$	$-.9668 \times 10^{-02}$	$.1777 \times 10^{-03}$.98			
		$-.9697 \times 10^{+00}$	$.9095 \times 10^{-02}$	$.8847 \times 10^{-02}$	$-.2486 \times 10^{-03}$.97			
1.0×10^{-04}	20.00	$.3631 \times 10^{+00}$	$.3432 \times 10^{-01}$	$.3325 \times 10^{-01}$	$-.1074 \times 10^{-02}$.97	1001	62	1
		$.2214 \times 10^{+00}$	$.1533 \times 10^{-02}$	$.1534 \times 10^{-02}$	$.9800 \times 10^{-06}$	1.00			
		$.9423 \times 10^{+00}$	$-.4419 \times 10^{-03}$	$-.4441 \times 10^{-03}$	$-.2181 \times 10^{-05}$	1.00			
1.0×10^{-05}	20.00	$.3288 \times 10^{+00}$	$.4133 \times 10^{-03}$	$.4119 \times 10^{-03}$	$-.1371 \times 10^{-05}$	1.00	1769	110	1
		$-.9784 \times 10^{+00}$	$.1659 \times 10^{-02}$	$.1659 \times 10^{-02}$	$-.2532 \times 10^{-06}$	1.00			
		$.3305 \times 10^{+00}$	$.7610 \times 10^{-04}$	$.7636 \times 10^{-04}$	$.2549 \times 10^{-06}$	1.00			
1.0×10^{-06}	20.00	$.2200 \times 10^{+00}$	$-.2181 \times 10^{-04}$	$-.2193 \times 10^{-04}$	$-.1210 \times 10^{-06}$	1.01	3129	195	1
		$.9427 \times 10^{+00}$	$.2063 \times 10^{-04}$	$.2065 \times 10^{-04}$	$.2656 \times 10^{-07}$	1.00			
		$-.9787 \times 10^{+00}$	$.8277 \times 10^{-04}$	$.8303 \times 10^{-04}$	$.2636 \times 10^{-06}$	1.00			
1.0×10^{-07}	20.00	$.2199 \times 10^{+00}$	$.4078 \times 10^{-05}$	$.4087 \times 10^{-05}$	$.9245 \times 10^{-08}$	1.00	5561	347	1
		$.9427 \times 10^{+00}$	$-.1161 \times 10^{-05}$	$-.1165 \times 10^{-05}$	$-.4130 \times 10^{-08}$	1.00			
		$-.9788 \times 10^{+00}$	$.1114 \times 10^{-05}$	$.1115 \times 10^{-05}$	$.1219 \times 10^{-08}$	1.00			
1.0×10^{-08}	20.00	$.3288 \times 10^{+00}$	$.4444 \times 10^{-05}$	$.4453 \times 10^{-05}$	$.9908 \times 10^{-08}$	1.00	9881	617	1
		$.2199 \times 10^{+00}$	$.2253 \times 10^{-06}$	$.2256 \times 10^{-06}$	$.2962 \times 10^{-09}$	1.00			
		$.9427 \times 10^{+00}$	$-.6384 \times 10^{-07}$	$-.6398 \times 10^{-07}$	$-.1324 \times 10^{-09}$	1.00			
1.0×10^{-09}	20.00	$-.9788 \times 10^{+00}$	$.6188 \times 10^{-07}$	$.6192 \times 10^{-07}$	$.3974 \times 10^{-10}$	1.00	17561	1097	1
		$.3288 \times 10^{+00}$	$.2456 \times 10^{-06}$	$.2459 \times 10^{-06}$	$.3215 \times 10^{-09}$	1.00			
		$.2199 \times 10^{+00}$	$.1259 \times 10^{-07}$	$.1260 \times 10^{-07}$	$.9277 \times 10^{-11}$	1.00			
1.0×10^{-10}	20.00	$.9427 \times 10^{+00}$	$-.3556 \times 10^{-08}$	$-.3560 \times 10^{-08}$	$-.4182 \times 10^{-11}$	1.00	31209	1950	1
		$-.9788 \times 10^{+00}$	$.3468 \times 10^{-08}$	$.3469 \times 10^{-08}$	$.1228 \times 10^{-11}$	1.00			
		$.3288 \times 10^{+00}$	$.1372 \times 10^{-07}$	$.1373 \times 10^{-07}$	$.1015 \times 10^{-10}$	1.00			
1.0×10^{-11}	20.00	$.2199 \times 10^{+00}$	$.7059 \times 10^{-09}$	$.7063 \times 10^{-09}$	$.3921 \times 10^{-12}$	1.00	55481	3467	1
		$.9427 \times 10^{+00}$	$-.1991 \times 10^{-09}$	$-.1992 \times 10^{-09}$	$-.1612 \times 10^{-12}$	1.00			
		$-.9788 \times 10^{+00}$	$.1948 \times 10^{-09}$	$.1949 \times 10^{-09}$	$.6572 \times 10^{-13}$	1.00			
1.0×10^{-10}	20.00	$.3288 \times 10^{+00}$	$.7696 \times 10^{-09}$	$.7700 \times 10^{-09}$	$.4259 \times 10^{-12}$	1.00	31209	1950	1
		$.2199 \times 10^{+00}$	$.3955 \times 10^{-10}$	$.3967 \times 10^{-10}$	$.1224 \times 10^{-12}$	1.00			
		$.9427 \times 10^{+00}$	$-.1114 \times 10^{-10}$	$-.1118 \times 10^{-10}$	$-.3261 \times 10^{-13}$	1.00			
1.0×10^{-11}	20.00	$-.9788 \times 10^{+00}$	$.1092 \times 10^{-10}$	$.1096 \times 10^{-10}$	$.3470 \times 10^{-13}$	1.00	31209	1950	1
		$.3288 \times 10^{+00}$	$.4312 \times 10^{-10}$	$.4325 \times 10^{-10}$	$.1268 \times 10^{-12}$	1.00			
		$.2199 \times 10^{+00}$	$.2296 \times 10^{-11}$	$.2228 \times 10^{-11}$	$-.6772 \times 10^{-13}$.97			
1.0×10^{-11}	20.00	$.9427 \times 10^{+00}$	$-.6526 \times 10^{-12}$	$-.6274 \times 10^{-12}$	$.2514 \times 10^{-13}$.96	55481	3467	1
		$-.9788 \times 10^{+00}$	$.6311 \times 10^{-12}$	$.6159 \times 10^{-12}$	$-.1513 \times 10^{-13}$.98			
		$.3288 \times 10^{+00}$	$.2506 \times 10^{-11}$	$.2429 \times 10^{-11}$	$-.7684 \times 10^{-13}$.97			

6.3 Results for Order 5 with 8 Stages

For the order 5 main formula we have used (3.52) derived in Chapter 3. For the error estimator we have used (5.45) obtained in Chapter 5. The results are given in the following 9 tables for each of the problems.

Problem 1

Tolerance	End Point	Numerical Solution	Global Error	Estimated Error	Difference	Ratio	No. of F. Eval.	No. of Steps	No. of Reject.
1.0×10^{-03}	5.00	$.6881 \times 10^{-02}$	$.1426 \times 10^{-03}$	$.1663 \times 10^{-03}$	$.2367 \times 10^{-04}$	1.17	69	4	0
1.0×10^{-04}	5.00	$.6767 \times 10^{-02}$	$.2946 \times 10^{-04}$	$.3295 \times 10^{-04}$	$.3490 \times 10^{-05}$	1.12	86	5	0
1.0×10^{-05}	5.00	$.6743 \times 10^{-02}$	$.4582 \times 10^{-05}$	$.4936 \times 10^{-05}$	$.3536 \times 10^{-06}$	1.08	120	7	0
1.0×10^{-06}	5.00	$.6738 \times 10^{-02}$	$.1890 \times 10^{-06}$	$.1909 \times 10^{-06}$	$.1843 \times 10^{-08}$	1.01	171	10	0
1.0×10^{-07}	5.00	$.6738 \times 10^{-02}$	$.1353 \times 10^{-07}$	$.1346 \times 10^{-07}$	$-.7141 \times 10^{-10}$.99	273	16	0
1.0×10^{-08}	5.00	$.6738 \times 10^{-02}$	$.7489 \times 10^{-09}$	$.7414 \times 10^{-09}$	$-.7521 \times 10^{-11}$.99	409	24	0
1.0×10^{-09}	5.00	$.6738 \times 10^{-02}$	$.4148 \times 10^{-10}$	$.4110 \times 10^{-10}$	$-.3759 \times 10^{-12}$.99	613	36	0
1.0×10^{-10}	5.00	$.6738 \times 10^{-02}$	$.2921 \times 10^{-11}$	$.2900 \times 10^{-11}$	$-.2052 \times 10^{-13}$.99	970	57	0
1.0×10^{-11}	5.00	$.6738 \times 10^{-02}$	$.1736 \times 10^{-12}$	$.1727 \times 10^{-12}$	$-.8475 \times 10^{-15}$	1.00	1497	88	0

6. Numerical Results

Problem 2

Tolerance	End Point	Numerical Solution	Global Error	Estimated Error	Difference	Ratio	No. of F. Eval.	No. of Steps	No. of Reject.
1.0×10^{-03}	10.00	$.3015 \times 10^{+00}$	$.4907 \times 10^{-05}$	$.3192 \times 10^{-05}$	$-.1715 \times 10^{-05}$.65	103	6	0
1.0×10^{-04}	10.00	$.3015 \times 10^{+00}$	$.5707 \times 10^{-06}$	$.4384 \times 10^{-06}$	$-.1324 \times 10^{-06}$.77	120	7	0
1.0×10^{-05}	10.00	$.3015 \times 10^{+00}$	$.8511 \times 10^{-07}$	$.7057 \times 10^{-07}$	$-.1454 \times 10^{-07}$.83	171	10	0
1.0×10^{-06}	10.00	$.3015 \times 10^{+00}$	$.9423 \times 10^{-08}$	$.8281 \times 10^{-08}$	$-.1142 \times 10^{-08}$.88	239	14	0
1.0×10^{-07}	10.00	$.3015 \times 10^{+00}$	$.8246 \times 10^{-09}$	$.7561 \times 10^{-09}$	$-.6852 \times 10^{-10}$.92	324	19	0
1.0×10^{-08}	10.00	$.3015 \times 10^{+00}$	$.6625 \times 10^{-10}$	$.6253 \times 10^{-10}$	$-.3721 \times 10^{-11}$.94	494	29	0
1.0×10^{-09}	10.00	$.3015 \times 10^{+00}$	$.5083 \times 10^{-11}$	$.4895 \times 10^{-11}$	$-.1883 \times 10^{-12}$.96	732	43	0
1.0×10^{-10}	10.00	$.3015 \times 10^{+00}$	$.3584 \times 10^{-12}$	$.3497 \times 10^{-12}$	$-.8649 \times 10^{-14}$.98	1140	67	0
1.0×10^{-11}	10.00	$.3015 \times 10^{+00}$	$.2463 \times 10^{-13}$	$.2425 \times 10^{-13}$	$-.3828 \times 10^{-15}$.98	1769	104	0

Problem 3

Tolerance	End Point	Numerical Solution	Global Error	Estimated Error	Difference	Ratio	No. of F. Eval.	No. of Steps	No. of Reject.
1.0×10^{-03}	20.00	$.2493 \times 10^{+01}$	$.1619 \times 10^{-02}$	$-.2393 \times 10^{-02}$	$-.4012 \times 10^{-02}$	-1.48	362	17	8
1.0×10^{-04}	20.00	$.2492 \times 10^{+01}$	$.1973 \times 10^{-03}$	$.7895 \times 10^{-04}$	$-.1184 \times 10^{-03}$.40	524	26	9
1.0×10^{-05}	20.00	$.2492 \times 10^{+01}$	$.5331 \times 10^{-05}$	$.3994 \times 10^{-05}$	$-.1337 \times 10^{-05}$.75	807	40	14
1.0×10^{-06}	20.00	$.2492 \times 10^{+01}$	$.1790 \times 10^{-06}$	$-.1816 \times 10^{-06}$	$-.3606 \times 10^{-06}$	-1.01	1252	63	20
1.0×10^{-07}	20.00	$.2492 \times 10^{+01}$	$.4910 \times 10^{-09}$	$-.3630 \times 10^{-07}$	$-.3679 \times 10^{-07}$	-73.94	1841	95	25
1.0×10^{-08}	20.00	$.2492 \times 10^{+01}$	$-.6031 \times 10^{-09}$	$-.3271 \times 10^{-08}$	$-.2668 \times 10^{-08}$	5.42	2761	147	29
1.0×10^{-09}	20.00	$.2492 \times 10^{+01}$	$-.8577 \times 10^{-11}$	$-.1002 \times 10^{-09}$	$-.9165 \times 10^{-10}$	11.69	4147	228	30
1.0×10^{-10}	20.00	$.2492 \times 10^{+01}$	$-.1735 \times 10^{-12}$	$-.4248 \times 10^{-11}$	$-.4075 \times 10^{-11}$	24.48	6376	357	34
1.0×10^{-11}	20.00	$.2492 \times 10^{+01}$	$-.9853 \times 10^{-13}$	$-.3353 \times 10^{-12}$	$-.2368 \times 10^{-12}$	3.40	9783	559	31

Problem 4

Tolerance	End Point	Numerical Solution	Global Error	Estimated Error	Difference	Ratio	No. of F. Eval.	No. of Steps	No. of Reject.
1.0×10^{-03}	20.00	$.1773 \times 10^{+02}$	$-.3086 \times 10^{-02}$	$-.2216 \times 10^{-02}$	$.8697 \times 10^{-03}$.72	95	5	1
1.0×10^{-04}	20.00	$.1773 \times 10^{+02}$	$-.6208 \times 10^{-03}$	$-.4403 \times 10^{-03}$	$.1805 \times 10^{-03}$.71	121	6	2
1.0×10^{-05}	20.00	$.1773 \times 10^{+02}$	$.3888 \times 10^{-05}$	$.6482 \times 10^{-05}$	$.2595 \times 10^{-05}$	1.67	138	7	2
1.0×10^{-06}	20.00	$.1773 \times 10^{+02}$	$-.1820 \times 10^{-06}$	$-.6151 \times 10^{-07}$	$.1205 \times 10^{-06}$.34	215	11	3
1.0×10^{-07}	20.00	$.1773 \times 10^{+02}$	$-.1576 \times 10^{-07}$	$-.1058 \times 10^{-07}$	$.5184 \times 10^{-08}$.67	335	17	5
1.0×10^{-08}	20.00	$.1773 \times 10^{+02}$	$-.1688 \times 10^{-08}$	$-.1410 \times 10^{-08}$	$.2779 \times 10^{-09}$.84	453	25	3
1.0×10^{-09}	20.00	$.1773 \times 10^{+02}$	$-.1260 \times 10^{-09}$	$-.1143 \times 10^{-09}$	$.1166 \times 10^{-10}$.91	709	39	5
1.0×10^{-10}	20.00	$.1773 \times 10^{+02}$	$-.1234 \times 10^{-10}$	$-.1180 \times 10^{-10}$	$.5408 \times 10^{-12}$.96	1057	60	4
1.0×10^{-11}	20.00	$.1773 \times 10^{+02}$	$-.1329 \times 10^{-11}$	$-.1289 \times 10^{-11}$	$.4038 \times 10^{-13}$.97	1644	94	5

Problem 5

Tolerance	End Point	Numerical Solution	Global Error	Estimated Error	Difference	Ratio	No. of F. Eval.	No. of Steps	No. of Reject.
1.0×10^{-03}	20.00	$-.9930 \times 10^{+04}$	$-.9931 \times 10^{+04}$	$-.1997 \times 10^{+04}$	$.7934 \times 10^{+04}$.20	283	15	3
1.0×10^{-04}	20.00	$.8209 \times 10^{+03}$	$.8200 \times 10^{+03}$	$.1647 \times 10^{+04}$	$.8271 \times 10^{+03}$	2.01	437	23	5
1.0×10^{-05}	20.00	$.5603 \times 10^{+02}$	$.5512 \times 10^{+02}$	$.8225 \times 10^{+02}$	$.2714 \times 10^{+02}$	1.49	669	34	10
1.0×10^{-06}	20.00	$.2050 \times 10^{+02}$	$.1959 \times 10^{+02}$	$.2265 \times 10^{+02}$	$.3062 \times 10^{+01}$	1.16	949	51	9
1.0×10^{-07}	20.00	$.2882 \times 10^{+01}$	$.1969 \times 10^{+01}$	$.2143 \times 10^{+01}$	$.1739 \times 10^{+00}$	1.09	1400	77	10
1.0×10^{-08}	20.00	$.1044 \times 10^{+01}$	$.1311 \times 10^{+00}$	$.1376 \times 10^{+00}$	$.6419 \times 10^{-02}$	1.05	2062	117	8
1.0×10^{-09}	20.00	$.9282 \times 10^{+00}$	$.1527 \times 10^{-01}$	$.1581 \times 10^{-01}$	$.5365 \times 10^{-03}$	1.04	3203	182	12
1.0×10^{-10}	20.00	$.9139 \times 10^{+00}$	$.9705 \times 10^{-03}$	$.9907 \times 10^{-03}$	$.2021 \times 10^{-04}$	1.02	4937	284	12
1.0×10^{-11}	20.00	$.9130 \times 10^{+00}$	$.7013 \times 10^{-04}$	$.7107 \times 10^{-04}$	$.9409 \times 10^{-06}$	1.01	7691	446	12

6. Numerical Results

Problem 6

Tolerance	End Point	Numerical Solution	Global Error	Estimated Error	Difference	Ratio	No. of F. Eval.	No. of Steps	No. of Reject.
1.0×10^{-03}	10.00	$.3134 \times 10^{+02}$ $.2295 \times 10^{+00}$	$.2027 \times 10^{-03}$ $-.1805 \times 10^{-04}$	$.1736 \times 10^{-03}$ $.4763 \times 10^{-04}$	$-.2907 \times 10^{-04}$ $.6568 \times 10^{-04}$.86 -2.64	52	3	0
1.0×10^{-04}	10.00	$.3134 \times 10^{+02}$ $.2295 \times 10^{+00}$	$.1583 \times 10^{-04}$ $-.4714 \times 10^{-05}$	$.2170 \times 10^{-04}$ $-.2401 \times 10^{-05}$	$.5869 \times 10^{-05}$ $.2313 \times 10^{-05}$	1.37 .51	69	4	0
1.0×10^{-05}	10.00	$.3134 \times 10^{+02}$ $.2295 \times 10^{+00}$	$-.1919 \times 10^{-05}$ $.4539 \times 10^{-06}$	$-.1693 \times 10^{-05}$ $.5795 \times 10^{-06}$	$.2265 \times 10^{-06}$ $.1256 \times 10^{-06}$.88 1.28	69	4	0
1.0×10^{-06}	10.00	$.3134 \times 10^{+02}$ $.2295 \times 10^{+00}$	$-.1048 \times 10^{-06}$ $.2729 \times 10^{-07}$	$-.8398 \times 10^{-07}$ $.3050 \times 10^{-07}$	$.2086 \times 10^{-07}$ $.3205 \times 10^{-08}$.80 1.12	95	5	1
1.0×10^{-07}	10.00	$.3134 \times 10^{+02}$ $.2295 \times 10^{+00}$	$-.1785 \times 10^{-08}$ $.7713 \times 10^{-09}$	$-.4515 \times 10^{-09}$ $.9346 \times 10^{-09}$	$.1333 \times 10^{-08}$ $.1633 \times 10^{-09}$.25 1.21	129	7	1
1.0×10^{-08}	10.00	$.3134 \times 10^{+02}$ $.2295 \times 10^{+00}$	$-.4571 \times 10^{-11}$ $.4110 \times 10^{-10}$	$-.6835 \times 10^{-10}$ $.4807 \times 10^{-10}$	$.7292 \times 10^{-10}$ $.6971 \times 10^{-11}$	-14.95 1.17	180	10	1
1.0×10^{-09}	10.00	$.3134 \times 10^{+02}$ $.2295 \times 10^{+00}$	$.1417 \times 10^{-10}$ $.1151 \times 10^{-11}$	$.1876 \times 10^{-10}$ $.1489 \times 10^{-11}$	$.4591 \times 10^{-11}$ $.3373 \times 10^{-12}$	1.32 1.29	265	15	1
1.0×10^{-10}	10.00	$.3134 \times 10^{+02}$ $.2295 \times 10^{+00}$	$.1611 \times 10^{-11}$ $.3140 \times 10^{-13}$	$.1837 \times 10^{-11}$ $.5811 \times 10^{-13}$	$.2260 \times 10^{-12}$ $.2671 \times 10^{-13}$	1.14 1.85	384	22	1
1.0×10^{-11}	10.00	$.3134 \times 10^{+02}$ $.2295 \times 10^{+00}$	$.1925 \times 10^{-12}$ $-.3278 \times 10^{-14}$	$.2055 \times 10^{-12}$ $-.1186 \times 10^{-14}$	$.1296 \times 10^{-13}$ $.2092 \times 10^{-14}$	1.07 .36	571	33	1

Problem 7

Tolerance	End Point	Numerical Solution	Global Error	Estimated Error	Difference	Ratio	No. of F. Eval.	No. of Steps	No. of Reject.
1.0×10^{-03}	10.00	$.2385 \times 10^{+01}$ $.5333 \times 10^{+00}$	$-.1373 \times 10^{-03}$ $.1195 \times 10^{-04}$	$-.1224 \times 10^{-03}$ $.1806 \times 10^{-04}$	$.1489 \times 10^{-04}$ $.6109 \times 10^{-05}$.89 1.51	52	3	0
1.0×10^{-04}	10.00	$.2385 \times 10^{+01}$ $.5333 \times 10^{+00}$	$-.1769 \times 10^{-04}$ $.2031 \times 10^{-05}$	$-.1422 \times 10^{-04}$ $.2721 \times 10^{-05}$	$.3476 \times 10^{-05}$ $.6897 \times 10^{-06}$.80 1.34	69	4	0
1.0×10^{-05}	10.00	$.2385 \times 10^{+01}$ $.5333 \times 10^{+00}$	$-.9310 \times 10^{-06}$ $.1127 \times 10^{-06}$	$-.8455 \times 10^{-06}$ $.1344 \times 10^{-06}$	$.8548 \times 10^{-07}$ $.2179 \times 10^{-07}$.91 1.19	78	4	1
1.0×10^{-06}	10.00	$.2385 \times 10^{+01}$ $.5333 \times 10^{+00}$	$-.6816 \times 10^{-07}$ $.8437 \times 10^{-08}$	$-.6479 \times 10^{-07}$ $.9503 \times 10^{-08}$	$.3365 \times 10^{-08}$ $.1066 \times 10^{-08}$.95 1.13	86	5	0
1.0×10^{-07}	10.00	$.2385 \times 10^{+01}$ $.5333 \times 10^{+00}$	$-.400 \times 10^{-08}$ $.5100 \times 10^{-09}$	$-.3887 \times 10^{-08}$ $.5495 \times 10^{-09}$	$.1129 \times 10^{-09}$ $.3952 \times 10^{-10}$.97 1.08	120	7	0
1.0×10^{-08}	10.00	$.2385 \times 10^{+01}$ $.5333 \times 10^{+00}$	$-.2469 \times 10^{-09}$ $.3179 \times 10^{-10}$	$-.2429 \times 10^{-09}$ $.3331 \times 10^{-10}$	$.4004 \times 10^{-11}$ $.1515 \times 10^{-11}$.98 1.05	171	10	0
1.0×10^{-09}	10.00	$.2385 \times 10^{+01}$ $.5333 \times 10^{+00}$	$-.1821 \times 10^{-10}$ $.2227 \times 10^{-11}$	$-.1809 \times 10^{-10}$ $.2296 \times 10^{-11}$	$.1220 \times 10^{-12}$ $.6903 \times 10^{-13}$.99 1.03	273	16	0
1.0×10^{-10}	10.00	$.2385 \times 10^{+01}$ $.5333 \times 10^{+00}$	$-.1053 \times 10^{-11}$ $.1334 \times 10^{-12}$	$-.1048 \times 10^{-11}$ $.1359 \times 10^{-12}$	$.5152 \times 10^{-14}$ $.2566 \times 10^{-14}$	1.00 1.02	409	24	0
1.0×10^{-11}	10.00	$.2385 \times 10^{+01}$ $.5333 \times 10^{+00}$	$-.6521 \times 10^{-13}$ $.8335 \times 10^{-14}$	$-.6501 \times 10^{-13}$ $.8435 \times 10^{-14}$	$.2032 \times 10^{-15}$ $.1006 \times 10^{-15}$	1.00 1.01	630	37	0

Problem 8

Tolerance	End Point	Numerical Solution	Global Error	Estimated Error	Difference	Ratio	No. of F. Eval.	No. of Steps	No. of Reject.
1.0×10^{-03}	10.00	$-.5466 \times 10^{+00}$ $.1916 \times 10^{+02}$	$-.2601 \times 10^{-02}$ $-.2224 \times 10^{-02}$	$-.2070 \times 10^{-02}$ $-.1552 \times 10^{-02}$	$.5310 \times 10^{-03}$ $.6727 \times 10^{-03}$.80 .70	155	8	2
1.0×10^{-04}	10.00	$-.5443 \times 10^{+00}$ $.1916 \times 10^{+02}$	$-.3051 \times 10^{-07}$ $-.9344 \times 10^{-04}$	$-.2721 \times 10^{-03}$ $-.6366 \times 10^{-04}$	$.3298 \times 10^{-04}$ $.2978 \times 10^{-04}$.89 .68	232	12	3
1.0×10^{-05}	10.00	$-.5440 \times 10^{+00}$ $.1916 \times 10^{+02}$	$-.2244 \times 10^{-04}$ $.2744 \times 10^{-07}$	$-.2124 \times 10^{-04}$ $.1041 \times 10^{-05}$	$.1205 \times 10^{-05}$ $.1013 \times 10^{-05}$.95 37.93	343	18	4
1.0×10^{-06}	10.00	$-.5440 \times 10^{+00}$ $.1916 \times 10^{+02}$	$-.1911 \times 10^{-05}$ $.4695 \times 10^{-06}$	$-.1852 \times 10^{-05}$ $.5185 \times 10^{-06}$	$.5839 \times 10^{-07}$ $.4903 \times 10^{-07}$.97 1.10	496	27	4
1.0×10^{-07}	10.00	$-.5440 \times 10^{+00}$ $.1916 \times 10^{+02}$	$-.1287 \times 10^{-06}$ $.4940 \times 10^{-07}$	$-.1265 \times 10^{-06}$ $.5156 \times 10^{-07}$	$.2182 \times 10^{-08}$ $.2158 \times 10^{-08}$.98 1.04	725	41	3
1.0×10^{-08}	10.00	$-.5440 \times 10^{+00}$ $.1916 \times 10^{+02}$	$-.9202 \times 10^{-08}$ $.4640 \times 10^{-08}$	$-.9106 \times 10^{-08}$ $.4748 \times 10^{-08}$	$.9673 \times 10^{-10}$ $.1078 \times 10^{-09}$.99 1.02	1107	64	2
1.0×10^{-09}	10.00	$-.5440 \times 10^{+00}$ $.1916 \times 10^{+02}$	$-.5994 \times 10^{-09}$ $.3469 \times 10^{-09}$	$-.5955 \times 10^{-09}$ $.3517 \times 10^{-09}$	$.3852 \times 10^{-11}$ $.4787 \times 10^{-11}$.99 1.01	1693	99	1
1.0×10^{-10}	10.00	$-.5440 \times 10^{+00}$ $.1916 \times 10^{+02}$	$-.3827 \times 10^{-10}$ $.2453 \times 10^{-10}$	$-.3812 \times 10^{-10}$ $.2473 \times 10^{-10}$	$.1525 \times 10^{-12}$ $.2042 \times 10^{-12}$	1.00 1.01	2653	156	0
1.0×10^{-11}	10.00	$-.5440 \times 10^{+00}$ $.1916 \times 10^{+02}$	$-.2368 \times 10^{-11}$ $.1601 \times 10^{-11}$	$-.2362 \times 10^{-11}$ $.1609 \times 10^{-11}$	$.5836 \times 10^{-14}$ $.8227 \times 10^{-14}$	1.00 1.01	4200	247	0

6. Numerical Results

Problem 9

Tolerance	End Point	Numerical Solution	Global Error	Estimated Error	Difference	Ratio	No. of F. Eval.	No. of Steps	No. of Reject.
1.0×10^{-03}	20.00	$.2537 \times 10^{+00}$	$.3384 \times 10^{-01}$	$.3008 \times 10^{-01}$	$-.3758 \times 10^{-02}$.89	412	21	6
		$.9338 \times 10^{+00}$	$-.8913 \times 10^{-02}$	$-.8051 \times 10^{-02}$	$.8621 \times 10^{-03}$.90			
1.0×10^{-04}	20.00	$-.9673 \times 10^{+00}$	$.1148 \times 10^{-01}$	$.1014 \times 10^{-01}$	$-.1341 \times 10^{-02}$.88	600	31	8
		$.3669 \times 10^{+00}$	$.3815 \times 10^{-01}$	$.3228 \times 10^{-01}$	$-.5870 \times 10^{-02}$.85			
1.0×10^{-05}	20.00	$.2225 \times 10^{+00}$	$.2627 \times 10^{-02}$	$.2792 \times 10^{-02}$	$.1645 \times 10^{-03}$	1.06	863	47	7
		$.9422 \times 10^{+00}$	$-.5280 \times 10^{-03}$	$-.5455 \times 10^{-03}$	$-.1754 \times 10^{-04}$	1.03			
1.0×10^{-06}	20.00	$-.9787 \times 10^{+00}$	$.9372 \times 10^{-03}$	$.9927 \times 10^{-03}$	$.5548 \times 10^{-04}$	1.06	1331	74	8
		$.3316 \times 10^{+00}$	$.2831 \times 10^{-02}$	$.2904 \times 10^{-02}$	$.7320 \times 10^{-04}$	1.03			
1.0×10^{-07}	20.00	$.2201 \times 10^{+00}$	$.1764 \times 10^{-03}$	$.1835 \times 10^{-03}$	$.7043 \times 10^{-05}$	1.04	1965	115	1
		$.9427 \times 10^{+00}$	$-.3178 \times 10^{-04}$	$-.3186 \times 10^{-04}$	$-.7344 \times 10^{-07}$	1.00			
1.0×10^{-08}	20.00	$-.9788 \times 10^{+00}$	$.6589 \times 10^{-04}$	$.6870 \times 10^{-04}$	$.2808 \times 10^{-05}$	1.04	3121	183	1
		$.3290 \times 10^{+00}$	$.1865 \times 10^{-03}$	$.1889 \times 10^{-03}$	$.2394 \times 10^{-05}$	1.01			
1.0×10^{-09}	20.00	$.2199 \times 10^{+00}$	$.7858 \times 10^{-05}$	$.8183 \times 10^{-05}$	$.3246 \times 10^{-06}$	1.04	4923	289	1
		$.9427 \times 10^{+00}$	$-.1158 \times 10^{-05}$	$-.1191 \times 10^{-05}$	$-.3257 \times 10^{-07}$	1.03			
1.0×10^{-10}	20.00	$-.9788 \times 10^{+00}$	$.3193 \times 10^{-05}$	$.3301 \times 10^{-05}$	$.1074 \times 10^{-06}$	1.03	7796	458	1
		$.3288 \times 10^{+00}$	$.8154 \times 10^{-05}$	$.8314 \times 10^{-05}$	$.1603 \times 10^{-06}$	1.02			
1.0×10^{-11}	20.00	$.2199 \times 10^{+00}$	$.4038 \times 10^{-06}$	$.4181 \times 10^{-06}$	$.1425 \times 10^{-07}$	1.04	12352	726	1
		$.9427 \times 10^{+00}$	$-.4266 \times 10^{-07}$	$-.4428 \times 10^{-07}$	$-.1623 \times 10^{-08}$	1.04			
1.0×10^{-03}	20.00	$-.9788 \times 10^{+00}$	$.1805 \times 10^{-06}$	$.1850 \times 10^{-06}$	$.4511 \times 10^{-08}$	1.02	4923	289	1
		$.3288 \times 10^{+00}$	$.4107 \times 10^{-06}$	$.4181 \times 10^{-06}$	$.7345 \times 10^{-08}$	1.02			
1.0×10^{-04}	20.00	$.2199 \times 10^{+00}$	$.1711 \times 10^{-07}$	$.1761 \times 10^{-07}$	$.5056 \times 10^{-09}$	1.03	3121	183	1
		$.9427 \times 10^{+00}$	$-.6528 \times 10^{-09}$	$-.7120 \times 10^{-09}$	$-.5915 \times 10^{-10}$	1.09			
1.0×10^{-05}	20.00	$-.9788 \times 10^{+00}$	$.8852 \times 10^{-08}$	$.9005 \times 10^{-08}$	$.1521 \times 10^{-09}$	1.02	4923	289	1
		$.3288 \times 10^{+00}$	$.1712 \times 10^{-07}$	$.1736 \times 10^{-07}$	$.2437 \times 10^{-09}$	1.01			
1.0×10^{-06}	20.00	$.2199 \times 10^{+00}$	$.7438 \times 10^{-09}$	$.7623 \times 10^{-09}$	$.1848 \times 10^{-10}$	1.02	7796	458	1
		$.9427 \times 10^{+00}$	$.4401 \times 10^{-10}$	$.4188 \times 10^{-10}$	$-.2131 \times 10^{-11}$.95			
1.0×10^{-07}	20.00	$-.9788 \times 10^{+00}$	$.4603 \times 10^{-09}$	$.4657 \times 10^{-09}$	$.5367 \times 10^{-11}$	1.01	4923	289	1
		$.3288 \times 10^{+00}$	$.7284 \times 10^{-09}$	$.7366 \times 10^{-09}$	$.8222 \times 10^{-11}$	1.01			
1.0×10^{-08}	20.00	$.2199 \times 10^{+00}$	$.3350 \times 10^{-10}$	$.3419 \times 10^{-10}$	$.6906 \times 10^{-12}$	1.02	7796	458	1
		$.9427 \times 10^{+00}$	$.6283 \times 10^{-11}$	$.6205 \times 10^{-11}$	$-.7739 \times 10^{-13}$.99			
1.0×10^{-09}	20.00	$-.9788 \times 10^{+00}$	$.2522 \times 10^{-10}$	$.2541 \times 10^{-10}$	$.1955 \times 10^{-12}$	1.01	7796	458	1
		$.3288 \times 10^{+00}$	$.3191 \times 10^{-10}$	$.3219 \times 10^{-10}$	$.2836 \times 10^{-12}$	1.01			
1.0×10^{-10}	20.00	$.2199 \times 10^{+00}$	$.1572 \times 10^{-11}$	$.1599 \times 10^{-11}$	$.2632 \times 10^{-13}$	1.02	12352	726	1
		$.9427 \times 10^{+00}$	$.5404 \times 10^{-12}$	$.5375 \times 10^{-12}$	$-.2872 \times 10^{-14}$.99			
1.0×10^{-11}	20.00	$-.9788 \times 10^{+00}$	$.1439 \times 10^{-11}$	$.1447 \times 10^{-11}$	$.7317 \times 10^{-14}$	1.01	12352	726	1
		$.3288 \times 10^{+00}$	$.1447 \times 10^{-11}$	$.1457 \times 10^{-11}$	$.1011 \times 10^{-13}$	1.01			

In deriving these results we have used an order 5 global error estimator with a block 6(4) formula. This block formula is 5(4) at the first step and 6(4) at the second step of the block. However the global error estimator that we have derived is appropriate for a 5(4) formula. In view of this we would expect the global error to sometimes be overestimated since the global error with a 6(4) formula will often be less than with a 5(4) formula. The numerical results that we have presented bear out these expectations. It would be of interest to see how our global error estimator behaves on a block 5(4) formula which is 5(4) on both steps and this we do in the next section.

6.4 Results for Order 5 with 9 Stages

In this section we will derive a new block 5(4) pair. We will use the same approach as we did in Chapter 3 with the only difference that we will use the free coefficients to make the formula at the second step in the block 5(4) rather than 6(4). The parameters were chosen in the following way:

- 1) set $\hat{b}_2 = 0$ and $c_9 = 1$;
 - 2) choose $c_8, \hat{b}_6, \hat{b}_8, \hat{b}_9$ arbitrarily;
 - 3) solve the order conditions of the form $\sum_{i=1}^9 \hat{b}_i c_i^j = \frac{1}{j+1}$, $j = 0, 1, 2, 3, 4$
for $\hat{b}_1, \hat{b}_3, \hat{b}_4, \hat{b}_5, \hat{b}_7$;
 - 4) choose a_{85}, a_{86}, a_{87} arbitrarily;
 - 5) a_{84} is given by $\sum_{i=1}^9 \hat{b}_i (1 - c_i) a_{ij} a_{jk} c_k = \frac{1}{120}$;
 - 6) a_{83} is given by $\sum_{i=1}^9 \hat{b}_i (1 - c_i) a_{ij} c_j^2 = \frac{1}{60}$;
 - 7) a_{82} is given by $\sum_{j=2}^8 a_{ij} c_j = \frac{1}{2} c_i^2$ with $i = 8$;
 - 8) a_{81} is given by the row-sum condition $c_8 = \sum_{i=1}^7 a_{8i}$;
 - 9) a_{9j} are obtained from $\sum_{i=1}^9 \hat{b}_i a_{ij} = \hat{b}_j (1 - c_j)$, for $j = 2, 3, \dots, 8$;
 - 10) a_{91} is given by the row-sum condition $c_9 = \sum_{i=1}^8 a_{9i}$.
- The \bar{b} of the embedded formula were obtained by:
- 11) setting $\bar{b}_2 = 0$;
 - 12) choosing $\bar{b}_6, \bar{b}_7, \bar{b}_8, \bar{b}_9$ arbitrarily;
 - 13) solving the order conditions of the form $\sum_{i=1}^9 \bar{b}_i c_i^j = \frac{1}{j+1}$, $j = 0, 1, 2, 3$
for $\bar{b}_1, \bar{b}_3, \bar{b}_4, \bar{b}_5$.

6. Numerical Results

The formula found is:

$$\begin{array}{c|cccccccccc}
 0 & 0 & & & & & & & & & \\
 \frac{1}{10} & \frac{1}{10} & 0 & & & & & & & & \\
 \frac{3}{20} & \frac{3}{80} & \frac{9}{80} & 0 & & & & & & & \\
 \frac{3}{10} & \frac{3}{20} & -\frac{9}{20} & \frac{3}{5} & 0 & & & & & & \\
 \frac{1}{3} & \frac{113}{729} & -\frac{25}{54} & \frac{440}{729} & \frac{55}{1458} & 0 & & & & & \\
 \frac{1}{2} & -\frac{181}{540} & \frac{5}{4} & -\frac{133}{297} & -\frac{91}{54} & \frac{189}{110} & 0 & & & & \\
 \frac{1}{2} & \frac{19}{432} & 0 & \frac{500}{2079} & -\frac{125}{432} & \frac{81}{176} & \frac{5}{112} & 0 & & & \\
 \frac{7}{10} & \frac{551}{1260} & \frac{167}{105} & -\frac{113}{35} & \frac{479}{252} & 0 & 0 & 0 & 0 & & \\
 1 & \frac{44159}{75600} & -\frac{331}{70} & \frac{36437}{72765} & \frac{746587}{15120} & -\frac{1528551}{30800} & \frac{617}{3920} & \frac{1343}{350} & \frac{9}{10} & 0 & \\
 \hline
 & \frac{1169}{6750} & 0 & -\frac{7648}{10395} & \frac{1877}{270} & -\frac{9153}{1375} & \frac{1}{10} & \frac{1343}{1750} & \frac{3}{10} & \frac{1}{10} & \\
 \hline
 & \frac{587}{270} & 0 & -\frac{3392}{297} & \frac{1541}{27} & -\frac{5481}{110} & 1 & 1 & 1 & 0 & \\
 \end{array} \tag{6.1}$$

For the global error estimator we have used (5.45) obtained in Chapter 5. The results are given in the following 9 tables for each of the test problems described earlier in this chapter.

Problem 1

Tolerance	End Point	Numerical Solution	Global Error	Estimated Error	Difference	Ratio	No. of F. Eval.	No. of Steps	No. of Reject.
1.0×10^{-03}	5.00	$.6731 \times 10^{-02}$	$-.6796 \times 10^{-05}$	$-.7436 \times 10^{-05}$	$-.6398 \times 10^{-06}$	1.09	154	9	0
1.0×10^{-04}	5.00	$.6737 \times 10^{-02}$	$-.4934 \times 10^{-06}$	$-.5109 \times 10^{-06}$	$-.1748 \times 10^{-07}$	1.04	205	12	0
1.0×10^{-05}	5.00	$.6738 \times 10^{-02}$	$-.4292 \times 10^{-07}$	$-.4359 \times 10^{-07}$	$-.6752 \times 10^{-09}$	1.02	324	19	0
1.0×10^{-06}	5.00	$.6738 \times 10^{-02}$	$-.4049 \times 10^{-08}$	$-.4075 \times 10^{-08}$	$-.2661 \times 10^{-10}$	1.01	528	31	0
1.0×10^{-07}	5.00	$.6738 \times 10^{-02}$	$-.2541 \times 10^{-09}$	$-.2546 \times 10^{-09}$	$-.5782 \times 10^{-12}$	1.00	868	51	0
1.0×10^{-08}	5.00	$.6738 \times 10^{-02}$	$-.1647 \times 10^{-10}$	$-.1649 \times 10^{-10}$	$-.1283 \times 10^{-13}$	1.00	1480	87	0
1.0×10^{-09}	5.00	$.6738 \times 10^{-02}$	$-.1067 \times 10^{-11}$	$-.1067 \times 10^{-11}$	$-.2859 \times 10^{-15}$	1.00	2585	152	0
1.0×10^{-10}	5.00	$.6738 \times 10^{-02}$	$-.6173 \times 10^{-13}$	$-.6173 \times 10^{-13}$	$-.5330 \times 10^{-17}$	1.00	4523	266	0
1.0×10^{-11}	5.00	$.6738 \times 10^{-02}$	$-.3650 \times 10^{-14}$	$-.3650 \times 10^{-14}$	$-.1027 \times 10^{-18}$	1.00	800	470	1

6. Numerical Results

Problem 2

Tolerance	End Point	Numerical Solution	Global Error	Estimated Error	Difference	Ratio	No. of F. Eval.	No. of Steps	No. of Reject.
1.0×10^{-03}	10.00	$.3015 \times 10^{+00}$	$-.2217 \times 10^{-05}$	$-.2252 \times 10^{-05}$	$-.3538 \times 10^{-07}$	1.02	154	9	0
1.0×10^{-04}	10.00	$.3015 \times 10^{+00}$	$-.3815 \times 10^{-06}$	$-.3848 \times 10^{-06}$	$-.3333 \times 10^{-08}$	1.01	222	13	0
1.0×10^{-05}	10.00	$.3015 \times 10^{+00}$	$-.4319 \times 10^{-07}$	$-.4337 \times 10^{-07}$	$-.1727 \times 10^{-09}$	1.00	307	18	0
1.0×10^{-06}	10.00	$.3015 \times 10^{+00}$	$-.4767 \times 10^{-08}$	$-.4775 \times 10^{-08}$	$-.8321 \times 10^{-11}$	1.00	477	28	0
1.0×10^{-07}	10.00	$.3015 \times 10^{+00}$	$-.4022 \times 10^{-09}$	$-.4025 \times 10^{-09}$	$-.2733 \times 10^{-12}$	1.00	749	44	0
1.0×10^{-08}	10.00	$.3015 \times 10^{+00}$	$-.3039 \times 10^{-10}$	$-.3039 \times 10^{-10}$	$-.7475 \times 10^{-14}$	1.00	1225	72	0
1.0×10^{-09}	10.00	$.3015 \times 10^{+00}$	$-.2082 \times 10^{-11}$	$-.2082 \times 10^{-11}$	$-.1784 \times 10^{-15}$	1.00	2092	123	0
1.0×10^{-10}	10.00	$.3015 \times 10^{+00}$	$-.1304 \times 10^{-12}$	$-.1304 \times 10^{-12}$	$-.3729 \times 10^{-17}$	1.00	3605	212	0
1.0×10^{-11}	10.00	$.3015 \times 10^{+00}$	$-.7844 \times 10^{-14}$	$-.7844 \times 10^{-14}$	$-.7318 \times 10^{-19}$	1.00	6317	371	1

Problem 3

Tolerance	End Point	Numerical Solution	Global Error	Estimated Error	Difference	Ratio	No. of F. Eval.	No. of Steps	No. of Reject.
1.0×10^{-03}	20.00	$.2490 \times 10^{+01}$	$-.1353 \times 10^{-02}$	$-.1209 \times 10^{-02}$	$.1441 \times 10^{-03}$.89	645	31	13
1.0×10^{-04}	20.00	$.2491 \times 10^{+01}$	$-.1861 \times 10^{-03}$	$-.1882 \times 10^{-03}$	$-.2081 \times 10^{-05}$	1.01	960	49	14
1.0×10^{-05}	20.00	$.2492 \times 10^{+01}$	$-.1512 \times 10^{-04}$	$-.1527 \times 10^{-04}$	$-.1491 \times 10^{-06}$	1.01	1507	78	20
1.0×10^{-06}	20.00	$.2492 \times 10^{+01}$	$-.3886 \times 10^{-06}$	$-.3971 \times 10^{-06}$	$-.8489 \times 10^{-08}$	1.02	2394	127	26
1.0×10^{-07}	20.00	$.2492 \times 10^{+01}$	$.2867 \times 10^{-07}$	$.2822 \times 10^{-07}$	$-.4510 \times 10^{-09}$.98	3909	214	30
1.0×10^{-08}	20.00	$.2492 \times 10^{+01}$	$.3811 \times 10^{-08}$	$.3772 \times 10^{-08}$	$-.3823 \times 10^{-10}$.99	6624	370	37
1.0×10^{-09}	20.00	$.2492 \times 10^{+01}$	$.5696 \times 10^{-09}$	$.5682 \times 10^{-09}$	$-.1416 \times 10^{-11}$	1.00	11411	650	40
1.0×10^{-10}	20.00	$.2492 \times 10^{+01}$	$.1976 \times 10^{-09}$	$.1975 \times 10^{-09}$	$-.1067 \times 10^{-12}$	1.00	19911	1150	40
1.0×10^{-11}	20.00	$.2492 \times 10^{+01}$	$.4287 \times 10^{-10}$	$.4286 \times 10^{-10}$	$-.1187 \times 10^{-13}$	1.00	34909	2037	31

Problem 4

Tolerance	End Point	Numerical Solution	Global Error	Estimated Error	Difference	Ratio	No. of F. Eval.	No. of Steps	No. of Reject.
1.0×10^{-03}	20.00	$.1773 \times 10^{+02}$	$-.5497 \times 10^{-03}$	$-.5742 \times 10^{-03}$	$-.2450 \times 10^{-04}$	1.04	146	8	1
1.0×10^{-04}	20.00	$.1773 \times 10^{+02}$	$-.1045 \times 10^{-03}$	$-.1075 \times 10^{-03}$	$-.2939 \times 10^{-05}$	1.03	214	12	1
1.0×10^{-05}	20.00	$.1773 \times 10^{+02}$	$-.1028 \times 10^{-04}$	$-.1038 \times 10^{-04}$	$-.1010 \times 10^{-06}$	1.01	299	17	1
1.0×10^{-06}	20.00	$.1773 \times 10^{+02}$	$-.1097 \times 10^{-05}$	$-.1101 \times 10^{-05}$	$-.3867 \times 10^{-08}$	1.00	487	27	3
1.0×10^{-07}	20.00	$.1773 \times 10^{+02}$	$-.1196 \times 10^{-06}$	$-.1198 \times 10^{-06}$	$-.1824 \times 10^{-09}$	1.00	784	45	2
1.0×10^{-08}	20.00	$.1773 \times 10^{+02}$	$-.5904 \times 10^{-08}$	$-.5906 \times 10^{-08}$	$-.1533 \times 10^{-11}$	1.00	1328	77	2
1.0×10^{-09}	20.00	$.1773 \times 10^{+02}$	$-.4568 \times 10^{-09}$	$-.4568 \times 10^{-09}$	$-.2835 \times 10^{-13}$	1.00	2306	134	3
1.0×10^{-10}	20.00	$.1773 \times 10^{+02}$	$-.4222 \times 10^{-10}$	$-.4222 \times 10^{-10}$	$-.5629 \times 10^{-15}$	1.00	4014	235	2
1.0×10^{-11}	20.00	$.1773 \times 10^{+02}$	$-.3495 \times 10^{-11}$	$-.3495 \times 10^{-11}$	$-.9514 \times 10^{-18}$	1.00	7066	414	3

Problem 5

Tolerance	End Point	Numerical Solution	Global Error	Estimated Error	Difference	Ratio	No. of F. Eval.	No. of Steps	No. of Reject.
1.0×10^{-03}	20.00	$-.2543 \times 10^{+05}$	$-.2543 \times 10^{+05}$	$-.2711 \times 10^{+05}$	$-.1672 \times 10^{+04}$	1.07	489	25	7
1.0×10^{-04}	20.00	$-.2510 \times 10^{+04}$	$-.2511 \times 10^{+04}$	$-.2590 \times 10^{+04}$	$-.7906 \times 10^{+02}$	1.03	975	52	10
1.0×10^{-05}	20.00	$-.1977 \times 10^{+03}$	$-.1986 \times 10^{+03}$	$-.2012 \times 10^{+03}$	$-.2672 \times 10^{+01}$	1.01	1637	92	8
1.0×10^{-06}	20.00	$-.1177 \times 10^{+02}$	$-.1269 \times 10^{+02}$	$-.1276 \times 10^{+02}$	$-.7258 \times 10^{-01}$	1.01	2677	150	14
1.0×10^{-07}	20.00	$.2159 \times 10^{+00}$	$-.6970 \times 10^{+00}$	$-.6985 \times 10^{+00}$	$-.1524 \times 10^{-02}$	1.00	4692	268	15
1.0×10^{-08}	20.00	$.8798 \times 10^{+00}$	$-.3315 \times 10^{-01}$	$-.3318 \times 10^{-01}$	$-.2408 \times 10^{-04}$	1.00	8091	469	13
1.0×10^{-09}	20.00	$.9115 \times 10^{+00}$	$-.1444 \times 10^{-02}$	$-.1445 \times 10^{-02}$	$-.3141 \times 10^{-06}$	1.00	14272	831	16
1.0×10^{-10}	20.00	$.9129 \times 10^{+00}$	$-.6495 \times 10^{-04}$	$-.6496 \times 10^{-04}$	$-.3916 \times 10^{-08}$	1.00	25184	1475	12
1.0×10^{-11}	20.00	$.9129 \times 10^{+00}$	$-.2692 \times 10^{-05}$	$-.2692 \times 10^{-05}$	$-.2933 \times 10^{-10}$	1.00	44667	2620	14

6. Numerical Results

Problem 6

Tolerance	End Point	Numerical Solution	Global Error	Estimated Error	Difference	Ratio	No. of F. Eval.	No. of Steps	No. of Reject.
1.0×10^{-03}	10.00	$.3134 \times 10^{+02}$ $.2295 \times 10^{+00}$	$.5288 \times 10^{-04}$ $-.3542 \times 10^{-04}$	$.5314 \times 10^{-04}$ $-.3666 \times 10^{-04}$	$.2604 \times 10^{-06}$ $-.1244 \times 10^{-05}$	1.00 1.04	78	4	1
1.0×10^{-04}	10.00	$.3134 \times 10^{+02}$ $.2295 \times 10^{+00}$	$.3648 \times 10^{-04}$ $-.1048 \times 10^{-04}$	$.3775 \times 10^{-04}$ $-.1092 \times 10^{-04}$	$.1264 \times 10^{-05}$ $-.4434 \times 10^{-06}$	1.03 1.04	78	4	1
1.0×10^{-05}	10.00	$.3134 \times 10^{+02}$ $.2295 \times 10^{+00}$	$.1077 \times 10^{-05}$ $-.7585 \times 10^{-06}$	$.1104 \times 10^{-05}$ $-.7667 \times 10^{-06}$	$.2685 \times 10^{-07}$ $-.8191 \times 10^{-08}$	1.02 1.01	95	5	1
1.0×10^{-06}	10.00	$.3134 \times 10^{+02}$ $.2295 \times 10^{+00}$	$.2248 \times 10^{-06}$ $-.8853 \times 10^{-07}$	$.2261 \times 10^{-06}$ $-.8908 \times 10^{-07}$	$.1305 \times 10^{-08}$ $-.5421 \times 10^{-09}$	1.01 1.01	155	8	2
1.0×10^{-07}	10.00	$.3134 \times 10^{+02}$ $.2295 \times 10^{+00}$	$.3895 \times 10^{-08}$ $-.7166 \times 10^{-08}$	$.3897 \times 10^{-08}$ $-.7178 \times 10^{-08}$	$.1968 \times 10^{-11}$ $-.1254 \times 10^{-10}$	1.00 1.00	223	12	2
1.0×10^{-08}	10.00	$.3134 \times 10^{+02}$ $.2295 \times 10^{+00}$	$-.1122 \times 10^{-08}$ $-.4792 \times 10^{-09}$	$-.1122 \times 10^{-08}$ $-.4794 \times 10^{-09}$	$-.7365 \times 10^{-12}$ $-.1942 \times 10^{-12}$	1.00 1.00	376	21	2
1.0×10^{-09}	10.00	$.3134 \times 10^{+02}$ $.2295 \times 10^{+00}$	$-.1877 \times 10^{-09}$ $-.3415 \times 10^{-10}$	$-.1877 \times 10^{-09}$ $-.3415 \times 10^{-10}$	$-.3292 \times 10^{-13}$ $-.3093 \times 10^{-14}$	1.00 1.00	631	36	2
1.0×10^{-10}	10.00	$.3134 \times 10^{+02}$ $.2295 \times 10^{+00}$	$-.1952 \times 10^{-10}$ $-.2193 \times 10^{-11}$	$-.1952 \times 10^{-10}$ $-.2193 \times 10^{-11}$	$-.9820 \times 10^{-15}$ $-.2713 \times 10^{-16}$	1.00 1.00	1098	64	1
1.0×10^{-11}	10.00	$.3134 \times 10^{+02}$ $.2295 \times 10^{+00}$	$-.1676 \times 10^{-11}$ $-.1370 \times 10^{-12}$	$-.1676 \times 10^{-11}$ $-.1370 \times 10^{-12}$	$-.2962 \times 10^{-16}$ $.9340 \times 10^{-19}$	1.00 1.00	1939	114	0

Problem 7

Tolerance	End Point	Numerical Solution	Global Error	Estimated Error	Difference	Ratio	No. of F. Eval.	No. of Steps	No. of Reject.
1.0×10^{-03}	10.00	$.2385 \times 10^{+01}$ $.5333 \times 10^{+00}$	$-.1942 \times 10^{-04}$ $.3334 \times 10^{-05}$	$-.1912 \times 10^{-04}$ $.3366 \times 10^{-05}$	$.2962 \times 10^{-06}$ $.3184 \times 10^{-07}$.98 1.01	69	4	0
1.0×10^{-04}	10.00	$.2385 \times 10^{+01}$ $.5333 \times 10^{+00}$	$-.2894 \times 10^{-05}$ $.5081 \times 10^{-06}$	$-.2876 \times 10^{-05}$ $.5105 \times 10^{-06}$	$.1733 \times 10^{-07}$ $.2437 \times 10^{-08}$.99 1.00	69	4	0
1.0×10^{-05}	10.00	$.2385 \times 10^{+01}$ $.5333 \times 10^{+00}$	$-.5111 \times 10^{-06}$ $.7741 \times 10^{-07}$	$-.5103 \times 10^{-06}$ $.7760 \times 10^{-07}$	$.7397 \times 10^{-09}$ $.1900 \times 10^{-09}$	1.00 1.00	103	6	0
1.0×10^{-06}	10.00	$.2385 \times 10^{+01}$ $.5333 \times 10^{+00}$	$-.5158 \times 10^{-07}$ $.8554 \times 10^{-08}$	$-.5155 \times 10^{-07}$ $.8563 \times 10^{-08}$	$.3486 \times 10^{-10}$ $.8582 \times 10^{-11}$	1.00 1.00	137	8	0
1.0×10^{-07}	10.00	$.2385 \times 10^{+01}$ $.5333 \times 10^{+00}$	$-.1250 \times 10^{-07}$ $.1832 \times 10^{-08}$	$-.1250 \times 10^{-07}$ $.1834 \times 10^{-08}$	$.2881 \times 10^{-11}$ $.1080 \times 10^{-11}$	1.00 1.00	188	11	0
1.0×10^{-08}	10.00	$.2385 \times 10^{+01}$ $.5333 \times 10^{+00}$	$-.4802 \times 10^{-08}$ $.6964 \times 10^{-09}$	$-.4801 \times 10^{-08}$ $.6967 \times 10^{-09}$	$.8477 \times 10^{-12}$ $.3264 \times 10^{-12}$	1.00 1.00	300	16	3
1.0×10^{-09}	10.00	$.2385 \times 10^{+01}$ $.5333 \times 10^{+00}$	$-.4700 \times 10^{-10}$ $.2240 \times 10^{-10}$	$-.4697 \times 10^{-10}$ $.2240 \times 10^{-10}$	$.2875 \times 10^{-13}$ $.2918 \times 10^{-14}$	1.00 1.00	555	31	3
1.0×10^{-10}	10.00	$.2385 \times 10^{+01}$ $.5333 \times 10^{+00}$	$.2430 \times 10^{-12}$ $.1151 \times 10^{-11}$	$.2437 \times 10^{-12}$ $.1151 \times 10^{-11}$	$.7281 \times 10^{-15}$ $.5099 \times 10^{-16}$	1.00 1.00	1014	58	3
1.0×10^{-11}	10.00	$.2385 \times 10^{+01}$ $.5333 \times 10^{+00}$	$.5864 \times 10^{-13}$ $.3462 \times 10^{-13}$	$.5865 \times 10^{-13}$ $.3462 \times 10^{-13}$	$.7529 \times 10^{-17}$ $.4120 \times 10^{-18}$	1.00 1.00	1864	108	3

Problem 8

Tolerance	End Point	Numerical Solution	Global Error	Estimated Error	Difference	Ratio	No. of F. Eval.	No. of Steps	No. of Reject.
1.0×10^{-03}	10.00	$-.5440 \times 10^{+00}$ $.1916 \times 10^{+02}$	$-.2711 \times 10^{-04}$ $-.3124 \times 10^{-03}$	$-.2481 \times 10^{-04}$ $-.2963 \times 10^{-03}$	$.2303 \times 10^{-05}$ $.1611 \times 10^{-04}$.92 .95	317	17	3
1.0×10^{-04}	10.00	$-.5440 \times 10^{+00}$ $.1916 \times 10^{+02}$	$-.6010 \times 10^{-05}$ $-.2102 \times 10^{-04}$	$-.5893 \times 10^{-05}$ $-.2065 \times 10^{-04}$	$.1171 \times 10^{-06}$ $.3676 \times 10^{-06}$.98 .98	478	27	2
1.0×10^{-05}	10.00	$-.5440 \times 10^{+00}$ $.1916 \times 10^{+02}$	$-.5910 \times 10^{-06}$ $-.1433 \times 10^{-05}$	$-.5870 \times 10^{-06}$ $-.1424 \times 10^{-05}$	$.4031 \times 10^{-08}$ $.9318 \times 10^{-08}$.99 .99	792	46	1
1.0×10^{-06}	10.00	$-.5440 \times 10^{+00}$ $.1916 \times 10^{+02}$	$-.4272 \times 10^{-07}$ $-.8494 \times 10^{-07}$	$-.4262 \times 10^{-07}$ $-.8476 \times 10^{-07}$	$.9700 \times 10^{-10}$ $.1868 \times 10^{-09}$	1.00 1.00	1353	79	1
1.0×10^{-07}	10.00	$-.5440 \times 10^{+00}$ $.1916 \times 10^{+02}$	$-.2771 \times 10^{-08}$ $-.4976 \times 10^{-08}$	$-.2769 \times 10^{-08}$ $-.4972 \times 10^{-08}$	$.2107 \times 10^{-11}$ $.3707 \times 10^{-11}$	1.00 1.00	2382	139	2
1.0×10^{-08}	10.00	$-.5440 \times 10^{+00}$ $.1916 \times 10^{+02}$	$-.1658 \times 10^{-09}$ $-.2812 \times 10^{-09}$	$-.1658 \times 10^{-09}$ $-.2811 \times 10^{-09}$	$.4022 \times 10^{-13}$ $.6708 \times 10^{-13}$	1.00 1.00	4192	246	1
1.0×10^{-09}	10.00	$-.5440 \times 10^{+00}$ $.1916 \times 10^{+02}$	$-.9740 \times 10^{-11}$ $-.1602 \times 10^{-10}$	$-.9739 \times 10^{-11}$ $-.1602 \times 10^{-10}$	$.7621 \times 10^{-15}$ $.1236 \times 10^{-14}$	1.00 1.00	7430	437	0
1.0×10^{-10}	10.00	$-.5440 \times 10^{+00}$ $.1916 \times 10^{+02}$	$-.5523 \times 10^{-12}$ $-.8927 \times 10^{-12}$	$-.5523 \times 10^{-12}$ $-.8927 \times 10^{-12}$	$.1362 \times 10^{-16}$ $.2173 \times 10^{-16}$	1.00 1.00	13176	775	0
1.0×10^{-11}	10.00	$-.5440 \times 10^{+00}$ $.1916 \times 10^{+02}$	$-.3127 \times 10^{-13}$ $-.5005 \times 10^{-13}$	$-.3127 \times 10^{-13}$ $-.5005 \times 10^{-13}$	$.2434 \times 10^{-18}$ $.3849 \times 10^{-18}$	1.00 1.00	23436	1378	1

6. Numerical Results

Problem 9

Tolerance	End Point	Numerical Solution	Global Error	Estimated Error	Difference	Ratio	No. of F. Eval.	No. of Steps	No. of Reject.
1.0×10^{-03}	20.00	$.2245 \times 10^{+00}$	$.4630 \times 10^{-02}$	$.4383 \times 10^{-02}$	$-.2466 \times 10^{-03}$.95	726	40	5
		$.9409 \times 10^{+00}$	$-.1789 \times 10^{-02}$	$-.1760 \times 10^{-02}$	$.2883 \times 10^{-04}$.98			
		$-.9779 \times 10^{+00}$	$.8647 \times 10^{-03}$	$.7630 \times 10^{-03}$	$-.1017 \times 10^{-03}$.88			
1.0×10^{-04}	20.00	$.3344 \times 10^{+00}$	$.5618 \times 10^{-02}$	$.5380 \times 10^{-02}$	$-.2374 \times 10^{-03}$.96	1098	64	1
		$.2207 \times 10^{+00}$	$.7761 \times 10^{-03}$	$.7664 \times 10^{-03}$	$-.9698 \times 10^{-05}$.99			
		$.9425 \times 10^{+00}$	$-.2414 \times 10^{-03}$	$-.2395 \times 10^{-03}$	$.1950 \times 10^{-05}$.99			
1.0×10^{-05}	20.00	$-.9786 \times 10^{+00}$	$.1920 \times 10^{-03}$	$.1887 \times 10^{-03}$	$-.3284 \times 10^{-05}$.98	1889	110	2
		$.3297 \times 10^{+00}$	$.8707 \times 10^{-03}$	$.8609 \times 10^{-03}$	$-.9759 \times 10^{-05}$.99			
		$.2199 \times 10^{+00}$	$.6191 \times 10^{-04}$	$.6172 \times 10^{-04}$	$-.1917 \times 10^{-06}$	1.00			
1.0×10^{-06}	20.00	$.9427 \times 10^{+00}$	$-.1793 \times 10^{-04}$	$-.1788 \times 10^{-04}$	$.4500 \times 10^{-07}$	1.00	3317	194	2
		$-.9787 \times 10^{+00}$	$.1654 \times 10^{-04}$	$.1648 \times 10^{-04}$	$-.5773 \times 10^{-07}$	1.00			
		$.3289 \times 10^{+00}$	$.6807 \times 10^{-04}$	$.6788 \times 10^{-04}$	$-.1927 \times 10^{-06}$	1.00			
1.0×10^{-07}	20.00	$.2199 \times 10^{+00}$	$.3917 \times 10^{-05}$	$.3914 \times 10^{-05}$	$-.3481 \times 10^{-08}$	1.00	5867	344	2
		$.9427 \times 10^{+00}$	$-.1111 \times 10^{-05}$	$-.1110 \times 10^{-05}$	$.8853 \times 10^{-09}$	1.00			
		$-.9788 \times 10^{+00}$	$.1072 \times 10^{-05}$	$.1071 \times 10^{-05}$	$-.9735 \times 10^{-09}$	1.00			
1.0×10^{-08}	20.00	$.3288 \times 10^{+00}$	$.4285 \times 10^{-05}$	$.4281 \times 10^{-05}$	$-.3497 \times 10^{-08}$	1.00	10423	612	2
		$.2199 \times 10^{+00}$	$.2286 \times 10^{-06}$	$.2286 \times 10^{-06}$	$-.6198 \times 10^{-10}$	1.00			
		$.9427 \times 10^{+00}$	$-.6433 \times 10^{-07}$	$-.6432 \times 10^{-07}$	$.1623 \times 10^{-10}$	1.00			
1.0×10^{-09}	20.00	$-.9788 \times 10^{+00}$	$.6318 \times 10^{-07}$	$.6317 \times 10^{-07}$	$-.1685 \times 10^{-10}$	1.00	18498	1087	2
		$.3288 \times 10^{+00}$	$.2497 \times 10^{-06}$	$.2496 \times 10^{-06}$	$-.6234 \times 10^{-10}$	1.00			
		$.2199 \times 10^{+00}$	$.1307 \times 10^{-07}$	$.1307 \times 10^{-07}$	$-.1100 \times 10^{-11}$	1.00			
1.0×10^{-10}	20.00	$.9427 \times 10^{+00}$	$-.3664 \times 10^{-08}$	$-.3664 \times 10^{-08}$	$.2924 \times 10^{-12}$	1.00	32906	1934	3
		$-.9788 \times 10^{+00}$	$.3632 \times 10^{-08}$	$.3632 \times 10^{-08}$	$-.2941 \times 10^{-12}$	1.00			
		$.3288 \times 10^{+00}$	$.1427 \times 10^{-07}$	$.1427 \times 10^{-07}$	$-.1107 \times 10^{-11}$	1.00			
1.0×10^{-11}	20.00	$.2199 \times 10^{+00}$	$.7411 \times 10^{-09}$	$.7411 \times 10^{-09}$	$-.1949 \times 10^{-13}$	1.00	58474	3438	3
		$.9427 \times 10^{+00}$	$-.2072 \times 10^{-09}$	$-.2072 \times 10^{-09}$	$.5224 \times 10^{-14}$	1.00			
		$-.9788 \times 10^{+00}$	$.2064 \times 10^{-09}$	$.2064 \times 10^{-09}$	$-.5166 \times 10^{-14}$	1.00			
1.0×10^{-10}	20.00	$.3288 \times 10^{+00}$	$.8085 \times 10^{-09}$	$.8085 \times 10^{-09}$	$-.1962 \times 10^{-13}$	1.00	32906	1934	3
		$.2199 \times 10^{+00}$	$.4176 \times 10^{-10}$	$.4176 \times 10^{-10}$	$-.3433 \times 10^{-15}$	1.00			
		$.9427 \times 10^{+00}$	$-.1166 \times 10^{-10}$	$-.1166 \times 10^{-10}$	$.9229 \times 10^{-16}$	1.00			
1.0×10^{-11}	20.00	$-.9788 \times 10^{+00}$	$.1165 \times 10^{-10}$	$.1165 \times 10^{-10}$	$-.9058 \times 10^{-16}$	1.00	32906	1934	3
		$.3288 \times 10^{+00}$	$.4556 \times 10^{-10}$	$.4556 \times 10^{-10}$	$-.3453 \times 10^{-15}$	1.00			
		$.2199 \times 10^{+00}$	$.2354 \times 10^{-11}$	$.2354 \times 10^{-11}$	$-.6101 \times 10^{-17}$	1.00			
1.0×10^{-11}	20.00	$.9427 \times 10^{+00}$	$-.6570 \times 10^{-12}$	$-.6570 \times 10^{-12}$	$.1644 \times 10^{-17}$	1.00	58474	3438	3
		$-.9788 \times 10^{+00}$	$.6573 \times 10^{-12}$	$.6573 \times 10^{-12}$	$-.1605 \times 10^{-17}$	1.00			
		$.3288 \times 10^{+00}$	$.2568 \times 10^{-11}$	$.2568 \times 10^{-11}$	$-.6138 \times 10^{-17}$	1.00			

It can be seen that the results obtained with this global error estimator are excellent. Often the global error is correctly estimated to at least two decimal places. The underlying numerical results are also very efficient with there being very few rejected steps. We feel that this is an excellent method of global error estimation which should be an option in subroutine libraries.

Chapter 7

An Alternative Approach to Stability

7.1 Introduction

In this chapter we will study a different concept of stability that was first considered by Higham and Hall [Hall88], [Higham91], [Higham90], [Higham89]. This theory can be used to determine whether a Runge-Kutta algorithm will perform smoothly when stability rather than accuracy restricts the stepsize. In the case of stiff problems we use formulae with very large stability regions which means that the stepsize is normally not restricted by stability requirements. Indeed when dealing with stiff problems it is normal to use formulae with infinite regions of absolute stability and in this case stability concepts such as A- and L-stability are appropriate. In the case of mildly stiff problems we use explicit methods since these seem to be the most efficient and in this case the stepsize is often limited by stability. When the algorithm attempts to use a step that is outside the stability region of the method then the error estimator will normally indicate a failure and another step is attempted which is hopefully inside the stability region. We consider the case where we wish to solve an initial value problem using a pair of explicit Runge-Kutta formulae of orders $p(p-1)$. The solution obtained by the main order p method is given by:

$$y_{n+1} = y_n + h \sum_{i=1}^s b_i k_i, \quad (7.1)$$

and the solution obtained by the embedded order $p-1$ method is given by:

$$\bar{y}_{n+1} = y_n + h \sum_{i=1}^s \bar{b}_i k_i. \quad (7.2)$$

7. An Alternative Approach to Stability

The new stability concept which we will describe is again based on the scalar test equation (1.15). If we apply the main method of order p (7.1) to the scalar test equation $y' = \lambda y$ we obtain $y_{n+1} = R(z)y_n$, where $z = \lambda h$. If we do the same with the embedded method we get $\bar{y}_{n+1} = \bar{R}(z)y_n$. If we now evaluate the difference $E(z) = R(z) - \bar{R}(z)$ and define $err_n = E(z)y_n$, we can control the stepsize using:

$$h_{n+1} = h_n SF \left(\frac{TOL}{|err_n|} \right)^\alpha \quad (7.3)$$

where TOL is a local tolerance given by the user and $\alpha = \frac{1}{p}$. Equation (7.3) defines the normal way of choosing the stepsize where SF is a safety factor, often equal to about 0.9, to account for the fact that we have neglected certain high order terms. Let us define the matrix

$$C = \begin{bmatrix} 1 & u \\ -\alpha & 1 - \alpha v \end{bmatrix}, \quad (7.4)$$

where $u = Re \left(\frac{R'(z)}{R(z)} z \right)$ and $v = Re \left(\frac{E'(z)}{E(z)} z \right)$. We are now in a position to state the proposition (see [Hairer91]) below:

Theorem 7.1 *The step control mechanism (7.3) is stable for $z = \lambda h$ on the boundary of the stability region iff the spectral radius of C in (7.4) satisfies:*

$$\rho(C) < 1.$$

If this inequality is satisfied the method is called SC-stable at z .

The matrix C is independent of the given differential equation and of the given tolerance. It is therefore a characteristic of the numerical method and the boundary of its stability region. In what follows we will examine the possibility of deriving block SC-stable methods. A convenient test problem for testing the performances of SC-stable methods is:

$$\begin{aligned} y_1' &= -2000(y_1 \cos x + y_2 \sin x + 1), & y_1(0) &= 1 \\ y_2' &= -2000(-y_1 \sin x + y_2 \cos x + 1), & y_2(0) &= 0 \end{aligned} \quad (7.5)$$

for $0 \leq x \leq 1.57$. For this range of x the eigenvalues of the Jacobian of (7.5) move slowly on a large circle from -2000 to $\pm 2000i$. For such a problem the eigenvalues have a sufficiently large modulus that for most tolerances the

7. An Alternative Approach to Stability

stepsize will be restricted by stability rather than by accuracy. In such circumstances we expect an SC-stable method to be superior to one which does not have this stability property. In what follows we will derive some SC-stable methods and test them on the problem (7.5).

7.2 Order 2 with 3 Stages

We will consider a general 2(1) pair of formulae using three function evaluations given by:

$$\begin{array}{c|ccc}
 0 & 0 & & \\
 c_2 & c_2 & 0 & \\
 c_3 & a_{31} & a_{32} & 0 \\
 \hline
 & b_1 & b_2 & b_3 \\
 \hline
 & \bar{b}_1 & \bar{b}_2 & \bar{b}_3
 \end{array} \tag{7.6}$$

The stability polynomial of the main order 2 formula is: $R(z) = 1 + z + \frac{1}{2}z^2 + tz^3$, where $t = b_3a_{32}c_2$. It immediately follows that $R'(z) = 1 + z + 3tz^2$. Similarly, for the embedded order 1 formula, $\bar{R}(z) = 1 + z + w_2z^2 + w_3z^3$, where $w_2 = \bar{b}_2c_2 + \bar{b}_3c_3$ and $w_3 = \bar{b}_3a_{32}c_2$. Now we can evaluate the difference $E(z) = R(z) - \bar{R}(z) = (\frac{1}{2} - w_2)z^2 + (t - w_3)z^3$. The first derivative is: $E'(z) = 2(\frac{1}{2} - w_2)z + 3(t - w_3)z^2$. If we take into consideration the order conditions and the row-sum conditions that the methods have satisfy if they are to have the required orders we have 6 free parameters left at our disposal. After some numerical searching we derived the method with $t = \frac{329}{5000}$, $w_1 = \frac{1893}{4700}$, $w_2 = \frac{77}{2000}$, which give improved regions of SC-stability:

$$\begin{array}{c|ccc}
 0 & 0 & & \\
 \frac{7}{200} & \frac{7}{200} & 0 & \\
 \frac{493}{700} & \frac{1387}{700} & \frac{94}{35} & 0 \\
 \hline
 & \frac{1}{10} & \frac{1}{5} & \frac{7}{10} \\
 \hline
 & \frac{503}{188} & \frac{307}{94} & \frac{77}{188}
 \end{array} \tag{7.7}$$

7. An Alternative Approach to Stability

When we apply this method to solve problem (7.5) we obtained 321 rejections when we considered $SF = 0.9$ and 156 when $SF = 0.8$, in both cases the tolerance used was $TOL = 10^{-6}$. We wish to compare this performance with that of a standard Runge-Kutta method.

As a conventional Runge-Kutta formula we consider the following 2(1) pair:

$$\begin{array}{c|cc}
 0 & 0 & \\
 \frac{1}{2} & \frac{1}{2} & 0 \\
 \hline
 & 0 & 1 \\
 \hline
 & 1 & 0
 \end{array} \tag{7.8}$$

When we use this pair to solve the same problem (7.5) we have 198 rejections for $SF = 0.8$ and 437 for $SF = 0.9$. This shows that we have a smaller number of rejections, in both cases, than when we used the block formulae. The graph of the stability regions of the order 2 method (7.7) is given in Figure 7.1, where the dotted line corresponds to the absolute stability region and the thick line to the SC-stability region. Although we have reduced the number of rejections by a small amount the small decrease is disappointing. Hence we will again consider a 2(1) pair but now with four function evaluations to see if we get a more dramatic decrease in the number of rejections.

7.3 Order 2 with 4 Stages

We will consider a general 2(1) pair of formulae using four function evaluations given by:

$$\begin{array}{c|cccc}
 0 & 0 & & & \\
 c_2 & c_2 & 0 & & \\
 c_3 & a_{31} & a_{32} & 0 & \\
 c_4 & a_{41} & a_{42} & a_{43} & 0 \\
 \hline
 & b_1 & b_2 & b_3 & b_4 \\
 \hline
 & \bar{b}_1 & \bar{b}_2 & \bar{b}_3 & \bar{b}_4
 \end{array} \tag{7.9}$$

The stability polynomial of the main order 2 formula is: $R(z) = 1 + z + \frac{1}{2}z^2 + t_3z^3 + t_4z^4$, where $t_3 = b_3a_{32}c_2 + b_4(a_{42}c_2 + a_{43}c_3)$ and $t_4 = b_4a_{43}a_{32}c_2$.

7. An Alternative Approach to Stability

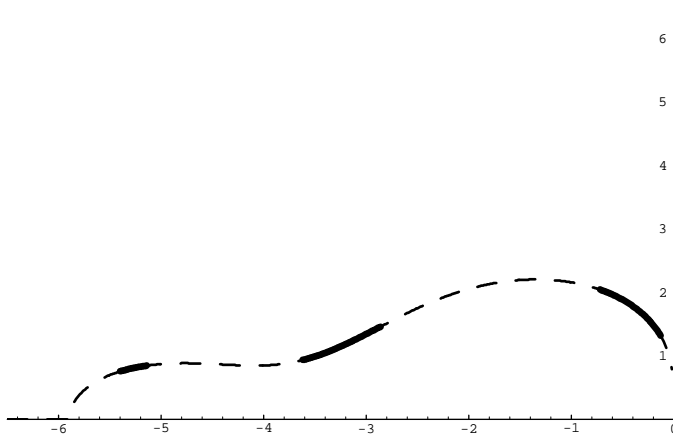


Figure 7.1: Stability region of (7.7)

7. An Alternative Approach to Stability

Differentiating this expression we obtain $R'(z) = 1 + z + 3t_3z^2 + 4t_4z^3$. Similarly, for the embedded order 1 formula, $\bar{R}(z) = 1 + z + w_2z^2 + w_3z^3 + w_4z^4$, where $w_2 = \bar{b}_2c_2 + \bar{b}_3c_3 + \bar{b}_4c_4$, $w_3 = \bar{b}_3a_{32}c_2 + \bar{b}_4(a_{42}c_2 + a_{43}c_3)$ and $w_4 = \bar{b}_4a_{43}a_{32}c_2$. Now we can evaluate the difference $E(z) = R(z) - \bar{R}(z) = (\frac{1}{2} - w_2)z^2 + (t_3 - w_3)z^3 + (t_4 - w_4)z^4$. The first derivative is: $E'(z) = 2(\frac{1}{2} - w_2)z + 3(t_3 - w_3)z^2 + 4(t_4 - w_4)z^3$. We have 11 free parameters at our disposal. After some numerical searching we found the following block 2(1) formula with $t_1 = \frac{3}{20}$, $t_2 = \frac{1}{64}$, $w_1 = \frac{49}{100}$, $w_2 = \frac{3}{20}$, $w_3 = \frac{1}{64}$:

$$\begin{array}{c|cccc}
 0 & 0 & & & \\
 \frac{1}{20} & \frac{1}{20} & 0 & & \\
 \frac{1}{5} & -\frac{1}{10} & \frac{3}{10} & 0 & \\
 \frac{7}{10} & \frac{123}{140} & -\frac{5}{3} & \frac{125}{84} & 0 \\
 \hline
 & \frac{1}{10} & \frac{1}{5} & 0 & \frac{7}{10} \\
 \hline
 & \frac{3}{10} & 0 & 0 & \frac{7}{10}
 \end{array} \tag{7.10}$$

When we applied this method to solve problem (7.5) we obtained 1 rejection for both the cases $SF = 0.8$ and $SF = 0.9$ when the tolerance used was $TOL = 10^{-6}$. When we use the conventional method (7.8) to solve the same problem (7.5) we have 198 rejections for $SF = 0.8$ and 437 for $SF = 0.9$. This gives a very large decrease in the number of rejections and is a much better result than was obtained in the case of three functions evaluations. The graph of the stability region of the order 2 method (7.10) is given in Figure 7.2, where again the dotted line corresponds to the absolute stability region and the thick line to the SC-stability region.

7. An Alternative Approach to Stability

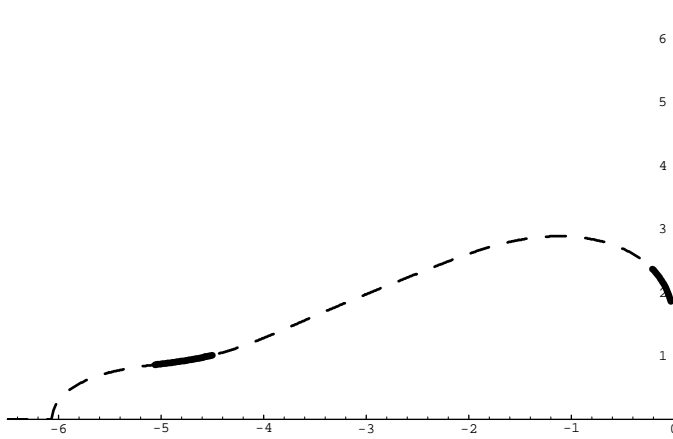


Figure 7.2: Stability region of (7.10)

7.4 Order 3 with 5 Stages

We now consider a general 3(2) pair of formulae using five function evaluations given by:

$$\begin{array}{c|ccccc}
 0 & 0 & & & & \\
 c_2 & c_2 & 0 & & & \\
 c_3 & a_{31} & a_{32} & 0 & & \\
 c_4 & a_{41} & a_{42} & a_{43} & 0 & \\
 c_5 & a_{51} & a_{52} & a_{53} & a_{54} & 0 \\
 \hline
 & b_1 & b_2 & b_3 & b_4 & b_5 \\
 \hline
 & \bar{b}_1 & \bar{b}_2 & \bar{b}_3 & \bar{b}_4 & \bar{b}_5
 \end{array} \tag{7.11}$$

The stability polynomial of the main order 3 formula is: $R(z) = 1 + z + \frac{1}{2}z^2 + \frac{1}{6}z^3 + t_4z^4 + t_5z^5$, where $t_4 = b_4a_{43}a_{32}c_2 + b_5(a_{53}a_{32}c_2 + a_{54}(a_{42}c_2 + a_{43}c_3))$ and $t_5 = b_5a_{54}a_{43}a_{32}c_2$. Differentiating this expression we obtain $R'(z) = 1 + z + \frac{1}{2}z^2 + 4t_4z^3 + 5t_5z^4$. Similarly for the embedded order 2 formula,

$\bar{R}(z) = 1 + z + \frac{1}{2}z^2 + w_3z^3 + w_4z^4 + w_5z^5$, where $w_3 = \sum_{ij=1}^5 \bar{b}_i a_{ij} c_j$, $w_4 =$

$\sum_{ijk=1}^5 \bar{b}_i a_{ij} a_{jk} c_k$ and $w_5 = \sum_{ijkl=1}^5 \bar{b}_i a_{ij} a_{jk} a_{kl} c_l$. Now if we evaluate the difference

we obtain $E(z) = R(z) - \bar{R}(z) = (\frac{1}{6} - w_3)z^3 + (t_4 - w_4)z^4 + (t_5 - w_5)z^5$.

The first derivative is: $E'(z) = 3(\frac{1}{6} - w_3)z^2 + 4(t_4 - w_4)z^3 + 5(t_5 - w_5)z^4$.

We have 18 free parameters at our disposal. After some numerical searching we found a block formula with $t_4 = \frac{450899}{14112000}$, $t_5 = \frac{54899}{23520000}$, $w_3 = \frac{10891}{75600}$,

7. An Alternative Approach to Stability

$w_4 = \frac{54899}{3528000}$, $w_5 = 0$. This is given by:

$$\begin{array}{cccccc}
 0 & 0 & & & & \\
 \frac{1}{4} & \frac{1}{4} & 0 & & & \\
 \frac{3}{8} & \frac{425}{4704} & \frac{1339}{4704} & 0 & & \\
 \frac{4}{5} & \frac{40664}{209375} & \frac{1276}{3125} & \frac{122672}{209375} & 0 & \\
 1 & \frac{347}{120} & \frac{989}{330} & \frac{194}{255} & \frac{8375}{4488} & 0 \\
 \hline
 & \frac{17}{120} & \frac{1}{165} & \frac{196}{425} & \frac{1675}{4488} & \frac{3}{100} \\
 \hline
 & \frac{19}{1800} & \frac{248}{825} & \frac{1288}{3825} & \frac{1675}{4488} & 0
 \end{array} \tag{7.12}$$

When we apply this method to solve problem (7.5) we obtained 87 rejections when we consider $SF = 0.8$ and 191 when $SF = 0.9$ in both cases the tolerance used was $TOL = 10^{-6}$. If we consider as a conventional Runge-Kutta formula the 3(2) pair that we have used in Chapter 3 as the first step of our block formula (3.22), when we use this pair to solve the same problem (7.5) we have 275 rejections for $SF = 0.8$ and 409 for $SF = 0.9$. This shows that we have a smaller number of rejections, in both cases, when we used the block formulae. The graph of the stability region of the order 3 method (7.12) is given in Figure 7.3, where the dotted line corresponds to the absolute stability region and the thick line to the SC-stability region.

7. An Alternative Approach to Stability

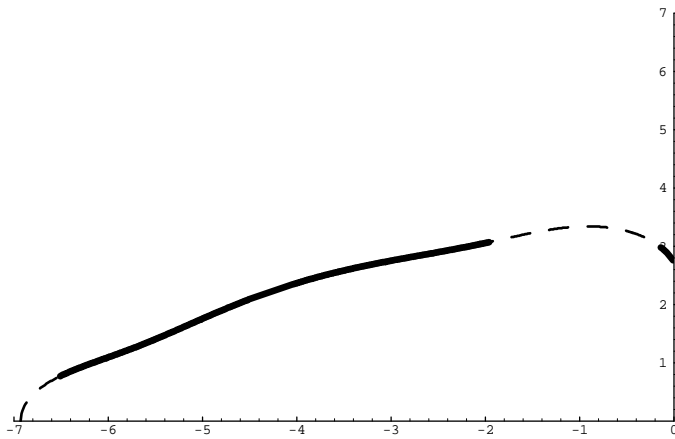


Figure 7.3: Stability region of (7.12)

7.5 Order 3 with 6 Stages

Finally we will consider a general 3(2) pair of formulae using six function evaluations given by:

$$\begin{array}{c|cccccc}
 0 & 0 & & & & & \\
 c_2 & c_2 & 0 & & & & \\
 c_3 & a_{31} & a_{32} & 0 & & & \\
 c_4 & a_{41} & a_{42} & a_{43} & 0 & & \\
 c_5 & a_{51} & a_{52} & a_{53} & a_{54} & 0 & \\
 c_6 & a_{61} & a_{62} & a_{63} & a_{64} & a_{65} & 0 \\
 \hline
 & b_1 & b_2 & b_3 & b_4 & b_5 & b_6 \\
 \hline
 & \bar{b}_1 & \bar{b}_2 & \bar{b}_3 & \bar{b}_4 & \bar{b}_5 & \bar{b}_6
 \end{array} \tag{7.13}$$

The stability polynomial of the main order 3 formula is: $R(z) = 1 + z + \frac{1}{2}z^2 + \frac{1}{6}z^3 + t_4z^4 + t_5z^5 + t_6z^6$, where $t_4 = \sum_{i,j,k=1}^6 b_i a_{ij} a_{jk} c_k$, $t_5 = \sum_{i,j,k,l=1}^6 b_i a_{ij} a_{jk} a_{kl} c_l$ and $t_6 = b_6 a_{65} a_{54} a_{43} a_{32} c_2$. Differentiating this expression we obtain $R'(z) = 1 + z + \frac{1}{2}z^2 + 4t_4z^3 + 5t_5z^4 + 6t_6z^5$. Similarly for the embedded order 2 formula,

$$\bar{R}(z) = 1 + z + \frac{1}{2}z^2 + w_3z^3 + w_4z^4 + w_5z^5 + w_6z^6, \text{ where } w_3 = \sum_{ij=1}^6 \bar{b}_i a_{ij} c_j,$$

$$w_4 = \sum_{ijk=1}^6 \bar{b}_i a_{ij} a_{jk} c_k, w_5 = \sum_{ijkl=1}^6 \bar{b}_i a_{ij} a_{jk} a_{kl} c_l \text{ and } w_6 = \sum_{ijklm=1}^6 \bar{b}_i a_{ij} a_{jk} a_{kl} a_{lm} c_m.$$

Now we can evaluate the difference $E(z) = R(z) - \bar{R}(z) = (\frac{1}{6} - w_3)z^3 + (t_4 - w_4)z^4 + (t_5 - w_5)z^5 + (t_6 - w_6)z^6$. The first derivative is: $E'(z) = 3(\frac{1}{6} - w_3)z^2 + 4(t_4 - w_4)z^3 + 5(t_5 - w_5)z^4 + 6(t_6 - w_6)z^5$. We have 25 free parameters at our disposal. After some numerical searching we found a block formula with $t_4 = \frac{1}{24}$, $t_5 = \frac{4}{625}$, $t_6 = \frac{1}{2500}$, $w_3 = \frac{17}{100}$, $w_4 = \frac{421}{10000}$, $w_5 = \frac{4}{625}$, $w_6 = \frac{1}{2500}$.

7. An Alternative Approach to Stability

This is given by:

$$\begin{array}{c|cccccc}
 0 & 0 & & & & \\
 \frac{1}{4} & \frac{1}{4} & 0 & & & \\
 \frac{3}{8} & 0 & \frac{3}{8} & 0 & & \\
 \frac{1}{2} & \frac{2}{5} & -\frac{2}{5} & \frac{1}{2} & 0 & \\
 \frac{7}{10} & \frac{10425679}{17769375} & -\frac{2048576}{1974375} & \frac{7930427}{7107750} & \frac{128}{3645} & 0 \\
 \frac{9}{10} & -\frac{121679348}{479773125} & \frac{48781612}{53308125} & \frac{866266321}{959546250} & -\frac{500325547}{319848750} & \frac{9}{10} & 0 \\
 \hline
 & \frac{1238}{9375} & \frac{172}{9375} & \frac{3104}{9375} & \frac{8569}{37500} & \frac{1}{50} & \frac{27}{100} \\
 \hline
 & \frac{71448709057}{1155009375000} & \frac{40413713857}{128334375000} & \frac{122433793049}{288752343750} & -\frac{529874581163}{1540012500000} & \frac{1065841}{3900000} & \frac{27}{100}
 \end{array} \tag{7.14}$$

When we apply this method to solve problem (7.5) we have obtained 8 rejections when we consider $SF = 0.8$ and 9 when $SF = 0.9$ in both cases the tolerance used was $TOL = 10^{-6}$. This shows that we have a much smaller number of rejections, in both cases, when we used the block formulae. The graph of the stability region of the order 3 method (7.14) is given in Figure 7.4, where the dotted line corresponds to the absolute stability region and the thick line to the SC-stability region.

These results show that by using a block approach we can find formulae with slightly improved regions of SC-stability which are very efficient for mildly stiff problems. By considering blocks with more functions we could find even higher order SC-stable formulae and we leave this as a topic for future research. However it should be said that the improvements in SC-stability are disappointing although the numerical results obtained are considerably better. More work is needed to understand this.

7. An Alternative Approach to Stability

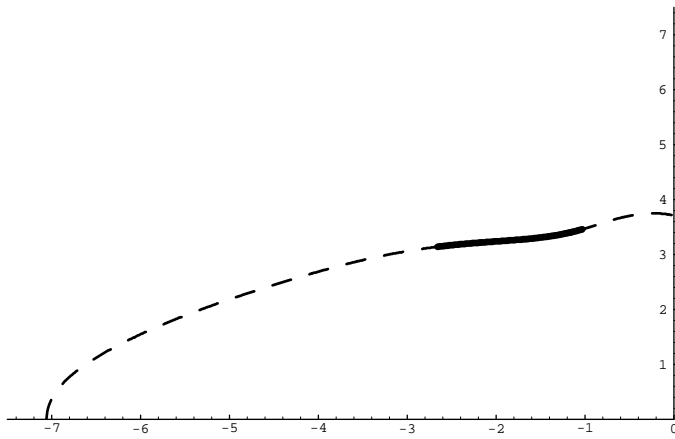


Figure 7.4: Stability region of (7.14)

Appendix A

Code to Implement Block 6(5) formula (3.61)

```
C PROGRAM RUN (INPUT,OUTPUT,RESULT,TAPE5=INPUT,TAPE6=RESULT)
C
C PROGRAM TO INTEGRATE DY/DT=-Y, Y(0)=1 IN THE RANGE 0.LE.T.LE.20
C WITH OUTPUT REQUIRED AT T=1.0,2.0,.....,19.0,20.0.
C
C THE DIMENSION STATEMENT HAS BEEN SET UP FOR 10 DIFFERENTIAL
C EQUATIONS.
C
C DIMENSION Y(10),YP(10),WORK(116),IWORK(5)
C EXTERNAL FCN
C open(1,file='65.res')
C iout=1
C
C SET INPUT PARAMETERS TO SUBROUTINE BRKF56
C
C NEQN = 1
C Y(1) = 1.0E+0
C T = 0.0E+0
C IFLAG = -1
C DEL = 1.0E+0
C K = 1
C TOUT = K*DEL
C TEND = 20.0E+0
C RELERR = 1.E-11
C ABSERR = 1.E-10
C WRITE(IOUT, 50)
C
C INPUT PARAMETERS SET.
C
10 CALL BRKF56 (FCN,NEQN,Y,T,TOUT,TEND,YP,RELERR,ABSERR,IFLAG,WORK,
* IWORK)
```

A. Code to Implement Block 6(5) formula (3.61)

```

      IF (IFLAG.LE.2) GO TO 30
C
C   IF IFLAG IS GREATER THAN 2 WE HAVE AN INPUT ERROR.
C
      WRITE (IOUT,20) IFLAG,NEQN,IWORK(1),T,TOUT,RELERR,ABSERR
20  FORMAT (1X,'FAILURE',5X,'IFLAG,NEQN,NUM F EVAL    = ',2I5,I10/
      *      13X,'T,TOUT    = ',2F13.4,/13X,'RELERR,ABSERR = ',1P2E10.1)
      IF (IFLAG .EQ. 3) THEN
          WRITE (IOUT,25)
25  FORMAT (1X,'RELATIVE ERROR TOLERANCE INCREASED TO CONTINUE')
          GO TO 10
      ELSE
          STOP
      END IF
30  CONTINUE
      IF (T.LT.TOUT) GO TO 10
C
C   THE INTEGRATION HAS PASSED THE OUTPUT POINT T=TOUT.
C
      A = EXP(-T)
      B = -A
      E1 = Y(1)-A
      E2 = YP(1)-B
C
C   E1 AND E2 ARE THE ERRORS IN THE INTERPOLATED SOLUTION
C   AND DERIVATIVE
C
      WRITE (IOUT,60) T,Y(1),YP(1),E1,E2,IWORK(1)
C
C   SET NEW VALUE OF THE OUTPUT POINT.
C
      K = K + 1
      TOUT = K*DEL
      IF(ABS(TOUT-TEND).LT.ABSERR) TOUT=TEND
      IF (T.NE.TEND) GO TO 10
50  FORMAT(5X,1HT, 11X, 6H      Y, 14X,
      * 10H      DY/DT, 6X, 10HERROR IN Y, 3X,
      * 14HERROR IN DY/DT)
60  FORMAT (1X,F8.4,1P2E20.10,2E10.2,I6)
      END
      SUBROUTINE FCN (T,Y,YP)
      DIMENSION Y(1),YP(1)
      YP(1) = -Y(1)
      RETURN
      END
      SUBROUTINE BRKF56(FCN,NEQN,Y,T,TOUT,TEND,YP,RELERR,ABSERR,IFLAG,
      *                  WORK,IWORK)
C
C   BLOCK FEHLBERG FIFTH-SIXTH ORDER RUNGE-KUTTA METHOD ADVANCING
C   A BLOCK OF TWO EQUAL STEPS. THE FORMULA AT THE FIRST STEP IS

```

A. Code to Implement Block 6(5) formula (3.61)

```
C      6(5) WHILE AT THE SECOND STEP IT IS 7(5).
C
C      BRKF56 IS PRIMARILY DESIGNED TO SOLVE NON-STIFF AND MILDLY STIFF
C      INITIAL VALUE ORDINARY DIFFERENTIAL EQUATIONS WHEN DERIVATIVE
C      EVALUATIONS ARE INEXPENSIVE. BRKF56 USES INTERPOLATION TO PRODUCE
C      OUTPUT AT 'OFF-STEP POINTS' EFFICIENTLY. BRKF SHOULD GENERALLY
C      NOT BE USED WHEN THE USER IS DEMANDING HIGH ACCURACY. IN SUCH
C      CASES A GOOD ADAMS CODE WILL OFTEN BE MORE EFFICIENT.
C
C*****
C ABSTRACT
C*****
C
C      SUBROUTINE BRKF56 INTEGRATES A SYSTEM OF NEQN FIRST ORDER
C      ORDINARY DIFFERENTIAL EQUATIONS OF THE FORM
C
C              DY(I)/DT = FCN(T,Y(1),Y(2),...,Y(NEQN))
C
C      WHERE THE Y(I) ARE KNOWN AT TIME T.
C      TYPICALLY THE SUBROUTINE IS USED TO INTEGRATE FROM T TO TEND
C      (WHILE RETURNING ANSWERS AT SPECIFIED OUTPUT POINTS TOUT), BUT
C      IT CAN ALSO BE USED AS A ONE-BLOCK INTEGRATOR TO ADVANCE THE
C      SOLUTION A SINGLE BLOCK STEP IN THE DIRECTION OF TEND. ON RETURN
C      THE PARAMETERS IN THE CALL LIST ARE SET FOR CONTINUING THE
C      INTEGRATION. THE USER HAS ONLY TO CALL BRKF56 AGAIN (AND PERHAPS
C      DEFINE A NEW VALUE FOR TOUT). ACTUALLY, BRKF56 IS AN INTERFACING
C      ROUTINE WHICH CALLS SUBROUTINE RKFC FOR THE SOLUTION. SUBROUTINE
C      RKFC COMPUTES AN APPROXIMATE SOLUTION OVER ONE BLOCK OF LENGTH 2H.
C      BRKF IS PARTICULARLY USEFUL WHEN OUTPUT IS REQUIRED AT MANY
C      'OFF-STEP POINTS' SINCE THE OUTPUT VALUES CAN BE OBTAINED BY
C      INTERPOLATION. THIS IS IN CONTRAST TO MANY OTHER RUNGE-KUTTA
C      PROGRAMS WHICH CHOOSE THE STEP SEQUENCE SO AS TO HIT ALL OUTPUT
C      POINTS EXACTLY AND SO BECOME INEFFICIENT WHEN OUTPUT IS REQUIRED
C      AT MANY POINTS WITHIN A STEP.
C      BRKF56 USES THE (6,5); (7,5) BLOCK FORMULA DESCRIBED IN
C      J.R. CASH and I. VIEIRA,
C      BLOCK 6(5) AND 7(6) EXPLICIT RUNGE-KUTTA FORMULAE (TO APPEAR).
C
C      THE PARAMETERS REPRESENT-
C      FCN -- SUBROUTINE FCN(T,Y,YP) TO EVALUATE DERIVATIVES
C              YP(I)=DY(I)/DT
C      NEQN -- NUMBER OF DIFFERENTIAL EQUATIONS TO BE INTEGRATED.
C      Y(*) -- APPROXIMATION TO THE SOLUTION VECTOR AT T.
C      T -- INDEPENDENT VARIABLE.
C      TOUT -- THE NEXT POINT WHERE INTERMEDIATE OUTPUT IS
C              REQUIRED. APPROXIMATE SOLUTIONS AT THESE POINTS
C              WILL BE OBTAINED BY INTERPOLATION.
C      TEND -- END OF THE INTEGRATION RANGE. THIS WILL BE HIT EXACTLY.
C      YP(*) -- APPROXIMATION TO THE DERIVATIVE VECTOR DY/DT AT T.
C      RELERR,ABSERR -- RELATIVE AND ABSOLUTE ERROR TOLERANCES FOR
```

A. Code to Implement Block 6(5) formula (3.61)

```
C          LOCAL ERROR TEST. AT THE NTH POINT OF EACH BLOCK (N=1,2)
C          THE CODE REQUIRES THAT
C              ABS(LOCAL ERROR)/N .LE. RELERR*ABS(Y) + ABSERR
C          FOR EACH COMPONENT OF THE LOCAL ERROR AND SOLUTION VECTORS
C          IFLAG -- INDICATOR FOR STATUS OF INTEGRATION.
C          WORK(*) -- ARRAY TO HOLD INFORMATION INTERNAL TO BRKF56 WHICH
C              IS NECESSARY FOR SUBSEQUENT CALLS. MUST BE DIMENSIONED
C              AT LEAST 6+11*NEQN
C          IWORK(*) -- INTEGER ARRAY USED TO HOLD INFORMATION INTERNAL TO
C              BRKF56 WHICH IS NECESSARY FOR SUBSEQUENT CALLS. MUST BE
C              DIMENSIONED AT LEAST 5
C
C
C*****
C FIRST CALL TO BRKF56
C*****
C
C          SUBROUTINE RKFC CONTAINS A MACHINE-DEPENDENT PARAMETER U.
C          THE SINGLE-PRECISION VERSION OF THIS PACKAGE CONTAINS A VALUE
C          SUITABLE FOR AN RS6000, AND THE DOUBLE-PRECISION VERSION ALSO CONTAINS
C          A VALUE SUITABLE FOR AN RS6000. IF THE PACKAGE IS RUN ON ANY
C          OTHER MACHINE, THE USER SHOULD FIRST RESET U, WHICH IS DEFINED IN
C          A DATA STATEMENT, CLEARLY MARKED IN SUBROUTINE RKFC. APPROPRIATE
C          VALUES OF U FOR OTHER MAJOR MACHINES ARE SPECIFIED IN THE COMMENTS
C          OF RKFC.
C
C          THE USER MUST PROVIDE STORAGE IN HIS CALLING PROGRAM FOR THE ARRAYS
C          IN THE CALL LIST - Y(NEQN) , YP(NEQN) , WORK(6+11*NEQN)
C          IWORK(5), DECLARE FCN IN AN EXTERNAL STATEMENT, SUPPLY SUBROUTINE
C          FCN(T,Y,YP) AND INITIALIZE THE FOLLOWING PARAMETERS-
C
C          NEQN -- NUMBER OF EQUATIONS TO BE INTEGRATED. (NEQN .GE. 1)
C          Y(*) -- VECTOR OF INITIAL CONDITIONS.
C          T -- STARTING POINT OF INTEGRATION , MUST BE A VARIABLE.
C          TOUT -- OUTPUT POINT AT WHICH SOLUTION, AND POSSIBLY THE
C              DERIVATIVE, IS DESIRED.
C          TEND -- END OF THE RANGE OF INTEGRATION. IF THE SOLUTION IS
C              REQUIRED ONLY AT TEND THEN THE USER SHOULD SET TOUT=TEND.
C          RELERR,ABSERR -- RELATIVE AND ABSOLUTE LOCAL ERROR TOLERANCES
C              WHICH MUST BE NON-NEGATIVE. RELERR MUST BE A VARIABLE WHILE
C              ABSERR MAY BE A CONSTANT. THE CODE SHOULD NORMALLY NOT BE
C              USED WITH RELATIVE ERROR CONTROL SMALLER THAN ABOUT 1.E-8,
C              UNLESS AN APPROPRIATE NONZERO ABSOLUTE TOLERANCE IS GIVEN.
C              TO AVOID LIMITING PRECISION DIFFICULTIES THE CODE REQUIRES
C              RELERR TO BE LARGER THAN AN INTERNALLY COMPUTED RELATIVE
C              ERROR PARAMETER WHICH IS MACHINE DEPENDENT. IN PARTICULAR,
C              PURE ABSOLUTE ERROR IS NOT PERMITTED. IF A SMALLER THAN
C              ALLOWABLE VALUE OF RELERR IS ATTEMPTED, BRKF56 INCREASES
C              RELERR APPROPRIATELY AND RETURNS CONTROL TO THE USER BEFORE
C              CONTINUING THE INTEGRATION.
```

A. Code to Implement Block 6(5) formula (3.61)

```

C      IFLAG -- +1,-1 INDICATOR TO INITIALIZE THE CODE FOR EACH NEW
C      PROBLEM. NORMAL INPUT IS +1. THE USER SHOULD SET IFLAG=-1
C      ONLY WHEN ONE-BLOCK INTEGRATOR CONTROL IS ESSENTIAL. IN
C      THIS CASE, BRKF56 ATTEMPTS TO ADVANCE THE SOLUTION A
C      SINGLE BLOCK IN THE DIRECTION OF TEND EACH TIME IT IS
C      CALLED. SINCE THIS MODE OF OPERATION RESULTS IN EXTRA
C      COMPUTING OVERHEAD, IT SHOULD BE AVOIDED UNLESS NEEDED.
C
C
C*****
C  OUTPUT FROM BRKF56
C*****
C
C  Y(*) -- COMPUTED SOLUTION APPROXIMATION AT T.
C  T -- VALUE OF THE INDEPENDENT VARIABLE WHERE THE SOLUTION IS
C      REPORTED.
C  IFLAG = 2 -- SUCCESSFUL RETURN. EITHER THE INTEGRATION REACHED
C      T=TEND OR A SUCCESSFUL INTERPOLATION HAS BEEN
C      PERFORMED AT T=TOUT. IF T.EQ.TEND THEN THE
C      INTEGRATION IS FINISHED. IF NOT, THE CODE SHOULD BE
C      CALLED WITH THE NEXT VALUE OF TOUT AND WITH IFLAG=+2
C      FOR NORMAL INTEGRATION OR IFLAG=-2 FOR ONE-BLOCK
C      INTEGRATION.
C  =-2 -- A SINGLE SUCCESSFUL BLOCK IN THE DIRECTION OF TEND
C      HAS BEEN TAKEN. NORMAL MODE FOR CONTINUING
C      INTEGRATION ONE BLOCK AT A TIME.
C  = 3 -- INTEGRATION WAS NOT COMPLETED BECAUSE RELATIVE ERROR
C      TOLERANCE WAS TOO SMALL. RELERR HAS BEEN INCREASED
C      APPROPRIATELY FOR CONTINUING.
C  = 4 -- INTEGRATION WAS NOT COMPLETED BECAUSE MORE THAN
C      18000 DERIVATIVE EVALUATIONS WERE NEEDED. THIS
C      IS APPROXIMATELY 1500 BLOCKS.
C  = 5 -- INTEGRATION WAS NOT COMPLETED BECAUSE SOLUTION
C      VANISHED MAKING A PURE RELATIVE ERROR TEST
C      IMPOSSIBLE. MUST USE NON-ZERO ABSERR TO CONTINUE.
C      USING THE ONE-BLOCK INTEGRATION MODE FOR ONE BLOCK
C      IS A GOOD WAY TO PROCEED.
C  = 6 -- INTEGRATION WAS NOT COMPLETED BECAUSE REQUESTED
C      ACCURACY COULD NOT BE ACHIEVED USING SMALLEST
C      ALLOWABLE STEPSIZE. USER MUST INCREASE THE ERROR
C      TOLERANCE BEFORE CONTINUED INTEGRATION CAN BE
C      ATTEMPTED.
C  = 7 -- INVALID INPUT PARAMETERS
C      THIS INDICATOR OCCURS IF ANY OF THE FOLLOWING IS
C      SATISFIED -  NEQN .LE. 0
C                  T=TEND
C                  RELERR OR ABSERR .LT. 0.
C                  ABS(IFLAG) .LT. 1 OR .GT.7
C  WORK(*),IWORK(*) -- INFORMATION WHICH IS USUALLY OF NO INTEREST
C      TO THE USER BUT NECESSARY FOR SUBSEQUENT CALLS.

```


A. Code to Implement Block 6(5) formula (3.61)

```
C          WORK(1),...,WORK(NEQN) CONTAIN THE SOLUTION VECTOR
C          AND WORK(NEQN+1),...,WORK(2*NEQN) CONTAIN THE
C          DERIVATIVE VECTOR AT THE END POINT (WHICH IS ITSELF
C          CONTAINED IN WORK(2*NEQN+1)) OF THE BLOCK STEP JUST
C          COMPUTED. WORK(2*NEQN+2) CONTAINS THE STEPSIZE H
C          JUST USED. (THIS IS THE STEPSIZE BEING USED BY THE
C          INTERPOLANT OVER THIS BLOCK.) WORK(2*NEQN+3)
C          CONTAINS THE STEPSIZE H TO BE ATTEMPTED ON THE NEXT
C          BLOCK. IWORK(1) CONTAINS THE DERIVATIVE EVALUATION
C          COUNTER.
C
C
C
C*****
C  SUBSEQUENT CALLS TO BRKF56
C*****
C
C  SUBROUTINE BRKF56 RETURNS WITH ALL INFORMATION NEEDED TO CONTINUE
C  THE INTEGRATION. AFTER THE CODE REPORTS A SUCCESSFUL SOLUTION AT
C  TOUT (INDICATED BY IFLAG=2), THE USER NEEDS TO DEFINE A NEW TOUT
C  BEFORE SIMPLY CALLING BRKF56 AGAIN TO CONTINUE IN THE NORMAL MODE.
C  (BUT THE USER MUST FIRST RESET IFLAG TO -2 TO CONTINUE IN THE
C  ONE-BLOCK INTEGRATOR MODE.)
C  IF THE INTEGRATION WAS NOT COMPLETED BUT THE USER STILL WANTS TO
C  CONTINUE (IFLAG=3,4), HE JUST CALLS BRKF56 AGAIN. IN THE CASE
C  IFLAG=3 THE RELERR PARAMETER HAS BEEN ADJUSTED APPROPRIATELY FOR
C  CONTINUING THE INTEGRATION. IN THE CASE OF IFLAG=4 THE FUNCTION
C  COUNTER WILL BE RESET TO 0 AND ANOTHER 18000 FUNCTION EVALUATIONS
C  ARE ALLOWED.
C  HOWEVER, IN THE CASE IFLAG=5, THE USER MUST FIRST ALTER THE ERROR
C  CRITERION TO USE A POSITIVE VALUE OF ABSERR BEFORE INTEGRATION CAN
C  PROCEED. IF HE DOES NOT, EXECUTION IS TERMINATED.
C  ALSO, IN THE CASE IFLAG=6, IT IS NECESSARY FOR THE USER TO RESET
C  IFLAG TO 2 (OR -2 WHEN THE ONE-BLOCK INTEGRATION MODE IS BEING
C  USED) AS WELL AS INCREASING EITHER ABSERR, RELERR OR BOTH BEFORE
C  THE INTEGRATION CAN BE CONTINUED. IF THIS IS NOT DONE, EXECUTION
C  WILL BE TERMINATED. THE OCCURRENCE OF IFLAG=6 INDICATES A TROUBLE
C  SPOT (SOLUTION IS CHANGING RAPIDLY, SINGULARITY MAY BE PRESENT) AND
C  IT OFTEN IS INADVISABLE TO CONTINUE.
C  IF IFLAG=7 IS OBTAINED, INTEGRATION CAN NOT BE CONTINUED UNLESS
C  THE INVALID INPUT PARAMETERS ARE CORRECTED.
C  IT SHOULD BE NOTED THAT THE ARRAYS WORK, IWORK CONTAIN INFORMATION
C  REQUIRED FOR SUBSEQUENT INTEGRATION. ACCORDINGLY, WORK AND IWORK
C  SHOULD NOT BE ALTERED.
C
C
C
C*****
C  USER CALLS TO THE INTERPOLANT ROUTINE EXTRA
C*****
C
C  SUBROUTINE EXTRA CAN ALSO BE CALLED BY THE USER, IN CONJUNCTION
```

A. Code to Implement Block 6(5) formula (3.61)

```

C   WITH USAGE OF BRKF56, TO PROVIDE APPROXIMATE SOLUTIONS AT 'OFF-STEP
C   POINTS' BY USE OF THE INTERPOLATING POLYNOMIAL. WHILE BRKF56
C   HANDLES THE USUAL SITUATIONS, IT CAN BE HELPFUL TO THE USER TO BE
C   ABLE TO ACCESS THE INTERPOLANT DIRECTLY, SUCH AS WHEN DOING ROOT
C   FINDING. ALSO, IT IS POSSIBLE THAT THE USER MAY HAVE SOME NEED FOR
C   EXTRAPOLATING OUTSIDE OF THE BLOCK STEP ON WHICH THE UNDERLYING
C   INTERPOLANT IS BASED. BRKF56 WILL NOT DO THIS.
C   THE FORM OF THE USAGE CALL IS
C
C   CALL EXTRA ( NEQN, WORK(6*NEQN+4), WORK(NEQN+1), WORK(2*NEQN+1),
C               WORK(2*NEQN+2), WORK(2*NEQN+4), WORK(3*NEQN+4),
C               WORK(4*NEQN+4), WORK(5*NEQN+4), TEX, YEX, YPEX )
C
C   WHERE YEX AND YPEX ARE THE SOLUTION VECTOR AND DERIVATIVE VECTOR
C   APPROXIMATIONS DEFINED BY THE INTERPOLANT AT THE POINT TEX, AND
C   WORK IS THE WORKING ARRAY SET UP BY BRKF56.
C*****
C
C   LOGICAL ENDPNT, BLKOUT
C   DIMENSION Y(NEQN), YP(NEQN), WORK(*), IWORK(5)
C   EXTERNAL FCN
C
C   COMPUTE INDICES FOR THE SPLITTING OF THE WORK ARRAY
C
C   KW = 1
C   KWP = KW + NEQN
C   KX = KWP + NEQN
C   KHI = KX + 1
C   KH = KHI + 1
C   KY1 = KH + 1
C   KY2 = KY1 + NEQN
C   KF1 = KY2 + NEQN
C   KF2 = KF1 + NEQN
C   KF3 = KF2 + NEQN
C   KF4 = KF3 + NEQN
C   KF5 = KF4 + NEQN
C   KF6 = KF5 + NEQN
C   KF7 = KF6 + NEQN
C   KSR = KF7 + NEQN
C   KSA = KSR + 1
C   KT = KSA + 1
C   K ? = KT + 1
C   THE WORK SPACE TOTALS 6 + 11*NEQN .
C
C   IF (ABS(IFLAG) .NE. 1) THEN
C       ENDPNT = (IWORK(4) .EQ. -1)
C       BLKOUT = (IWORK(5) .EQ. -1)
C   END IF
C
C*****

```

A. Code to Implement Block 6(5) formula (3.61)

```

C   THIS INTERFACING ROUTINE MERELY RELIEVES THE USER OF A LONG
C   CALLING LIST VIA THE SPLITTING APART OF TWO WORKING STORAGE
C   ARRAYS. IF THIS IS NOT COMPATIBLE WITH THE USERS COMPILER,
C   HE MUST USE RKFC DIRECTLY.
C*****
C
C   CALL RKFC (FCN,NEQN,Y,T,TOUT,TEND,YP,RELERR,ABSERR,IFLAG,WORK(KW),
*       WORK(KWP),WORK(KX),WORK(KHI),WORK(KH),WORK(KY1),WORK(KY2),
*       WORK(KF1),WORK(KF2),WORK(KF3),WORK(KF4),
*       WORK(KF5),WORK(KF6),WORK(KF7),WORK(KSR),
*       WORK(KSA),WORK(KT),IWORK(1),IWORK(2),IWORK(3),ENDPNT,BLKOUT)

IWORK(4) = 0
IF (ENDPNT) IWORK(4) = -1
IWORK(5) = 0
IF (BLKOUT) IWORK(5) = -1

RETURN
END
SUBROUTINE RKFC (FCN,NEQN,Y,T,TOUT,TEND,YP,RELERR,ABSERR,IFLAG,
*       W,WP,X,HINT,H,Y1,Y2,F1,F2,F3,F4,
*       F5,F6,F7,SAVRE,SAVAE,
*       TOLD,NFE,JFLAG,KFLAG,ENDPNT,BLKOUT)
C
C   TWO STEP BLOCK RUNGE-KUTTA FEHLBERG METHOD.
C   A STANDARD 6(5) FORMULA IS USED AT THE FIRST POINT IN THE BLOCK
C   AND A 7(5) FORMULA IS USED AT THE SECOND POINT.
C
LOGICAL HFAILD,ENDPNT,INTERP,BLKOUT
DIMENSION Y(NEQN),YP(NEQN),Y1(NEQN),Y2(NEQN),F1(NEQN),F2(NEQN),
*       F3(NEQN),F4(NEQN),F5(NEQN),F6(NEQN),F7(NEQN),
*       W(NEQN),WP(NEQN)
EXTERNAL FCN
PARAMETER( C2=1.E+0/12.E+0, C3=2.E+0/15.E+0, C4=1.E+0/5.E+0,
*       C5=8.E+0/15.E+0, C6=13.E+0/19.E+0, C7=19.E+0/20.E+0,
*       C8=1.E+0,C9=1.0E+0,C10=0.465E+0,c11=1.71E+0,C12=2.0E+0)
PARAMETER( A21=C2 )
PARAMETER( A31=2.E+0/75.E+0, A32=8.E+0/75.E+0 )
PARAMETER( A41=1.E+0/20.E+0, A42=0.E+0, A43=3.E+0/20.E+0 )
PARAMETER( A51=88.E+0/135.E+0, A52=0.E+0,
*       A53=-112.E+0/45.E+0, A54=64.E+0/27.E+0 )
PARAMETER( A61=-408551.E+0/260642.E+0, A62=0.E+0,
*       A63=3426735.E+0/521284.E+0,A64=-650026.E+0/130321.E+0,
*       A65=347139.E+0/521284.E+0 )
PARAMETER( A71=1296313.E+0/565760.E+0, A72=0.E+0,
*       A73=-48507.E+0/5120.E+0, A74=3310503.E+0/400384.E+0,
*       A75=-761805.E+0/748544.0E+0,
*       A76=197436315.E+0/223814656.E+0 )
PARAMETER( A81=103039.E+0/16796.E+0, A82=0.E+0,A83=-105.E+0/4.E+0,
*       A84=8856.E+0/391.E+0, A85=-13797.E+0/3655.0E+0,

```

A. Code to Implement Block 6(5) formula (3.61)

```

*          A86=53582508.E+0/22075469.E+0, A87=-1792.E+0/9595.E+0 )
PARAMETER( B1B=1385.E+0/23712.E+0, B3B=0.E+0,
*          B4B=515.E+0/1656.E+0, B5B=2511.E+0/9632.E+0,
*          B6B=17332693.E+0/93496104.E+0,
*          B7B=4352.E+0/17271.E+0, B8B=-17.E+0/252.E+0 )
PARAMETER( ERC11=B1B-1249.OE+0/23712.OE+0,ERC13=0.OE+0 ,
*          ERC14=B4B-61.OE+0/184.OE+0,ERC15=B5B-1269.E+0/6880.E+0,
*          ERC16=B6B-8731507.OE+0/31165368.OE+0,
*          ERC17=B7B-4352.OE+0/28785.OE+0,ERC18=B8B)

C
C   THE ABOVE DEFINE THE COEFFICIENTS FOR BRKF67 USED TO GENERATE
C   THE SOLUTION AT THE FIRST BLOCK POINT. BELOW ARE ADDITIONAL
C   COEFFICIENTS NEEDED TO GENERATE THE SOLUTION AT THE SECOND BLOCK
C   POINT.
C
PARAMETER( B1=31140992768.E+0/257777690625.E+0,
*          B2=0.E+0, B3=0.E+0,
*          B4=9.E+0/500.E+0,
*          B5=0.E+0,
*          B6=-11242116232463771.E+0/20983703550468750.E+0,
*          B7=-109680975232.E+0/194961524625.E+0,
*          B8=408061607.E+0/5982637500.E+0,
*          B9=7.E+0/5.E+0,
*          B10=1184802942976000.E+0/1290745082732553.E+0,
*          B11=33951571088000.E+0/68176788371811.E+0,
*          B12=23129157748.E+0/306736171875.E+0 )
PARAMETER( A91=1385.E+0/23712.E+0, A92=0.E+0, A93=0.E+0,
*          A94=515.E+0/1656.E+0,
*          A95=2511.E+0/9632.E+0, A96=17332693.E+0/93496104.E+0,
*          A97=4352.E+0/17271.E+0, A98=-17.E+0/252.E+0 )
PARAMETER( A101=-15514400620094897541.E+0/
*          73161581728768000000.E+0,
*          A102=0.E+0,
*          A103=14894129938336353.E+0/14810036787200000.E+0,
*          A104=-1115465796694125137.E+0/2554731345792000000.E+0,
*          A105=2570129433088854921.E+0/63683158184960000000.E+0,
*          A106=4715356027351248054167.E+0/
*          48079192350690176000000.E+0,
*          A107=-261974217902055743.E+0/4163153407417500000.E+0,
*          A108=7.E+0/400.E+0, A109=3.E+0/200.E+0 )
PARAMETER( A111=45043408253882515066518381.E+0/
*          9173877821824847497600000.E+0,
*          A112=0.E+0,
*          A113=-27917699597648811.E+0/6790314217600000.E+0,
*          A114=-1506088107154654995594000251.E+0/
*          149493353169413000720000000.E+0,
*          A115=-8259724559381291201457887499.E+0/
*          1222758133260687859150000000.E+0,
*          A116=-32220126226752270243394813467141.E+0/
*          2363268663445728810142134400000.E+0,

```

A. Code to Implement Block 6(5) formula (3.61)

```
*      A117=-490423417373912049138476588.E+0/  
*      60903053823901059014453125.E+0,  
*      A118=87.E+0/100.E+0,  
*      A119=49383719169866734171599.E+0/  
*      4549797705156047848000.E+0,  
*      A1110=78777581343539111904.E+0/2843623565722529905.E+0)  
PARAMETER( A121=-1019761775615731879569301491872119.E+0/  
*      37313919344744228533165074891648.E+0,  
*      A122=0.E+0,  
*      A123=1354611699555.E+0/92516630992.E+0,  
*      A124=6975021330674121332266184865803.E+0/  
*      96515985709527356754700486112.E+0,  
*      A125=201256172007798122954183274296301.E+0/  
*      5052402034533519023159103707776.E+0,  
*      A126=63908462135618415595781597790625.E+0/  
*      794459256039394231668411538912.E+0,  
*      A127=10469664538547844770584570311160.E+0/  
*      251649696435574725016671512023.E+0,  
*      A128=-3988339351014871459225909175.E+0/  
*      1049086801190514747333700936.E+0,  
*      A129=-896812812789916578125.E+0/  
*      16825872213726690772.E+0,  
*      A1210=-24580495743905600000000.E+0/  
*      149630077900640928651.E+0,  
*      A1211=413260910502978125000.E+0/  
*      215773357006517336541.E+0 )
```

```
C  
C      NEXT WE DEFINE COEFFICIENTS FOR THE ERROR ESTIMATE FORMULA.  
C
```

```
PARAMETER( ERC21=B1+835201624659198460204559.E+0/  
*      17356570978219562407500000.E+0,  
*      ERC24=B4-653.E+0/1000.E+0,  
*      ERC25=B5-1017751370513896071.E+0/3257163639862100000.E+0,  
*      ERC26=B6-49794680976565711400765612263.E+0/  
*      191852228025914567681797500000.E+0,  
*      ERC27=B7+37330322369529825525437.E+0/  
*      140647085089840407697500.E+0,  
*      ERC28=B8-7028842195201371181033.E+0/  
*      100705061165248032500000.E+0,  
*      ERC29=B9-219.E+0/250.E+0,  
*      ERC210=B10+4040664073513642374486928.E+0/  
*      8846484592512498482993925.E+0,  
*      ERC211=B11-10579467130236170324548567.E+0/  
*      19786416837497639760809700.E+0,  
*      ERC212=B12-81.E+0/1250.E+0)
```

```
C  
C      IN WHAT FOLLOWS WE GIVE DECIMAL APPROXIMATIONS TO THE  
C      COEFFICIENTS NEEDED TO COMPUTE THE SOLUTION OVER THE SECOND  
C      STEP OF THE BLOCK.
```

A. Code to Implement Block 6(5) formula (3.61)

```

C     PARAMETER( B1=.6040280811830318E-01 *2.0E+0,
C     *         B3=0.E+0,
C     *         B4=18.E+0/1000.E+0,
C     *         B5=0.E+0,
C     *         B6=-0.2678773126348223E+0 *2.0E+0,
C     *         B7=-.2812887708059000E+0*2.0E+0,
C     *         B8=.3410382252257749E-01 *2.0E+0,
C     *         B9=0.6999999993294478E+0*2.0E+0,
C     *         B10=.4589608586644637E+0 *2.0E+0,
C     *         B11=.2489965565907767E+0*2.0E+0,
C     *         B12=.3770203821515350E-01*2.0E+0 )
C     PARAMETER( A91=1385.E+0/23712.E+0, A92=0.E+0, A93=0.E+0,
C     *         A94=515.E+0/1656.E+0,
C     *         A95=2511.E+0/9632.E+0, A96=17332693.E+0/93496104.E+0,
C     *         A97=4352.E+0/17271.E+0, A98=-17.E+0/252.E+0 )
C     PARAMETER( A101=-.1060283313668945*2.0E+0,
C     *         A102=0.E+0,
C     *         A103=-.5028390666228310E+0*2.0E+0,
C     *         A104=-.2183137239644424E+0*2.0E+0,
C     *         A105=.2017903668165691E-01*2.0E+0,
C     *         A106=.4903738749274709E-01*2.0E+0,
C     *         A107=-.3146343546589802E-01*2.0E+0,
C     *         A108=7.E+0/400.E+0, A109=3.E+0/200.E+0 )
C     PARAMETER( A111=2.454981901744750E+0*2.0E+0,
C     *         A112=0.E+0,
C     *         A113=-2.055700074490705E+0*2.0E+0,
C     *         A114=-5.037307978015044E+0*2.0E+0,
C     *         A115=-3.377497272161788E+0*2.0E+0,
C     *         A116=-6.816856414875826E+0*2.0E+0,
C     *         A117=-4.026262930441851E+0*2.0E+0,
C     *         A118=87.E+0/100.E+0,
C     *         A119=5.427023606512188E+0 *2.0E+0,
C     *         A1110=13.85161916172827E+0*2.0E+0 )
C     PARAMETER( A121=-13.66462958579018E+0*2.0E+0,
C     *         A122=0.E+0,
C     *         A123=7.320908944413973E+0*2.0E+0,
C     *         A124=36.13402129703147E+0*2.0E+0,
C     *         A125=19.91688010637204E+0*2.0E+0,
C     *         A126=40.22135923602411E+0*2.0E+0,
C     *         A127=20.80206065944549E+0*2.0E+0,
C     *         A128=-1.900862411510558E+0 *2.0E+0,
C     *         A129=-26.64981642636537E+0*2.0E+0,
C     *         A1210=-82.13754911021002E+0 *2.0E+0,
C     *         A1211=.9576272905890594E+0*2.0E+0 )
C
C     NEXT WE DEFINE COEFFICIENTS FOR THE ERROR ESTIMATE FORMULA.
C
C     PARAMETER( ERC21=B1+.2406009878306810E-01*2.0E+0,
C     *         ERC24=B4-653.E+0/1000.E+0,
C     *         ERC25=-.1562327629983161E+0*2.0E+0,

```

A. Code to Implement Block 6(5) formula (3.61)

```

C   *   ERC26=B6-.1297735307587656E+0*2.0E+0,
C   *   ERC27=B7+.1327091971902782E+0*2.0E+0,
C   *   ERC28=B8-.3489816056431624E-01*2.0E+0,
C   *   ERC29=B9-219.E+0/250.E+0,
C   *   ERC210=B10+.2283768204767345E+0*2.0E+0,
C   *   ERC211=B11-.2673416621286829E+0*2.0E+0,
C   *   ERC212=B12-81.E+0/1250.E+0)

C   THE COEFFICIENTS OF THE RUNGE-KUTTA FORMULA ARE NOW SET.
C
C*****
C
C   THE COMPUTER UNIT ROUNDOFF ERROR U IS THE SMALLEST POSITIVE VALUE
C   REPRESENTABLE IN THE MACHINE SUCH THAT 1.+ U .GT. 1.
C
C           VALUES TO BE USED ARE
C           U = 9.5E-7           FOR IBM 360/370
C           U = 1.2E-7           FOR DEC VAX (SINGLE PRECISION)
C           U = 1.5E-8           FOR UNIVAC 1108
C           U = 7.5E-9           FOR PDP-10
C           U = 7.1E-15          FOR CDC 6000 SERIES
C           U = 2.2D-16          FOR IBM 360/370 (DOUBLE PRECISION)
C           U = 2.8D-17          FOR DEC VAX (DOUBLE PRECISION)
C
C   DATA U / .60E-7 /
C   DATA U / 3.6E-15 /
C
C*****
C
C   REMIN IS A TOLERANCE THRESHOLD WHICH IS ALSO DETERMINED BY THE
C   INTEGRATION METHOD. IN PARTICULAR, A SIXTH ORDER METHOD WILL
C   GENERALLY NOT BE CAPABLE OF DELIVERING ACCURACIES NEAR LIMITING
C   PRECISION ON COMPUTERS WITH LONG WORDLENGTHS. THIS DOES NOT HAVE TO
C   BE CHANGED FOR DIFFERENT MACHINES.
C
C   DATA REMIN / 1.E-12 /
C
C*****
C
C   THE EXPENSE IS CONTROLLED BY RESTRICTING THE NUMBER
C   OF FUNCTION EVALUATIONS TO BE APPROXIMATELY MAXNFE.
C
C   DATA MAXNFE / 18000 /
C
C*****
C
C   CHECK INPUT PARAMETERS
C
C   IF (NEQN.LT.1) GO TO 10

```


A. Code to Implement Block 6(5) formula (3.61)

```
C
C   RESET FUNCTION EVALUATION COUNTER
C
C   50 NFE = 0
C       IF (MFLAG.EQ.2) GO TO 70
C
C   RESET FLAG VALUE FROM PREVIOUS CALL
C
C   60 IFLAG = JFLAG
C       IF (KFLAG.EQ.3) MFLAG = ABS(IFLAG)
C
C   SAVE INPUT IFLAG AND SET CONTINUATION FLAG VALUE FOR SUBSEQUENT
C   INPUT CHECKING
C
C   70 JFLAG = IFLAG
C       KFLAG = 0
C
C   SAVE RELERR AND ABSERR FOR CHECKING INPUT ON SUBSEQUENT CALLS
C
C       SAVRE = RELERR
C       SAVAE = ABSERR
C
C   RESTRICT RELATIVE ERROR TOLERANCE TO BE AT LEAST AS LARGE AS
C   2U+REMIN TO AVOID LIMITING PRECISION DIFFICULTIES ARISING FROM
C   IMPOSSIBLE ACCURACY REQUESTS. IF TOLERANCE TOO SMALL, INCREASE
C   AND RETURN.
C
C       RER = 2.0E+0*U+REMIN
C       IF (RELERR .LT. RER) THEN
C
C           RELATIVE ERROR TOLERANCE TOO SMALL
C
C           RELERR = RER
C           IFLAG = 3
C           KFLAG = 3
C           RETURN
C       END IF
C
C   U26 = 26.0E+0*U
C   IF (MFLAG.NE.1) GO TO 100
C
C*****
C
C   INITIALIZATION --
C           DEFINE INTEGRATION INDEPENDENT VARIABLE X
C           EVALUATE INITIAL DERIVATIVES
C           SET UP WORKING ARRAYS FOR INTEGRATION VARIABLES
C           SET COUNTER FOR FUNCTION EVALUATIONS,NFE
C           ESTIMATE STARTING STEPSIZE
C
```

A. Code to Implement Block 6(5) formula (3.61)

```

X = T
ENDPNT = .FALSE.
BLKOUT = .FALSE.

A = T
CALL FCN (A,Y,YP)
NFE = 1
DO 80 N = 1, NEQN
    W(N) = Y(N)
    WP(N) = YP(N)
80 CONTINUE
C
C   COMPUTE INITIAL STEPLENGTH.
C
    DT = TOUT - T
    IF (DT .EQ. 0.0E+0) DT = TEND - T
    H = ABS(DT)
    TOLN = 0.0E+0
    DO 90 K = 1, NEQN
        TOL = RELERR*ABS(Y(K))+ABSERR
        IF (TOL.LE.0.0E+0) GO TO 90
        TOLN = TOL
        YPK = ABS(YP(K))
        IF (YPK*H**6.GT.TOL) H = (TOL/YPK)**0.167E+0
90 CONTINUE
    IF (TOLN.LE.0.0E+0) H = 0.0E+0
    H = MAX(H,U26*MAX(ABS(T),ABS(DT)))
    JFLAG = SIGN(2,IFLAG)
    H = SIGN(H,DT)
C
C   INITIAL STEPLENGTH NOW COMPUTED. COMPUTE FIRST SOLUTION.
C
C*****
C
100 CONTINUE
C
C   TO AVOID PREMATURE UNDERFLOW IN THE ERROR TOLERANCE FUNCTION,
C   SCALE THE ERROR TOLERANCES.
C
    SCALE = 2.0E+0/RELERR
    AE = SCALE*ABSERR
C
C   SET SAFETY FACTOR FOR STEPSIZE ADJUSTMENT, BASED ON TOLERANCES.
C
    TOLER = MAX(ABSERR,RELERR)
    SF = 0.85E+0
    IF (TOLER .GE. 1.E-5) SF = 0.8E+0
    IF (TOLER .LE. 1.E-9) SF = 0.9E+0
C
C   RESTORE INTEGRATION VARIABLE TO END OF LAST BLOCK STEP TAKEN

```

A. Code to Implement Block 6(5) formula (3.61)

```

C   AND SET THE SIGN OF THE DIRECTION OF INTEGRATION.
C
      T = X
      DTSIGN = SIGN(1.0E+0,TEND-T)
C
C   HAVE WE ALREADY INTEGRATED PAST THE PRESENT DATA OUTPUT POINT?
C   IF SO JUMP TO 390 AND PERFORM INTERPOLATION.
C   IF NOT, SEE IF WE HAVE REACHED THE END POINT OF INTEGRATION.
C   IF NOT, SEE IF RESULTS AT THE END OF THE BLOCK STEP NEED TO BE
C   REPORTED.
C
      IF ((TOUT-T)*DTSIGN .LE. 0.0E+0) THEN
        IF (TOUT .EQ. T) THEN
          IFLAG = 2
          ENDPNT = .FALSE.
          GO TO 400
        ELSE
          GO TO 390
        END IF
      END IF
      IF (ENDPNT) THEN
        IFLAG = 2
        ENDPNT = .FALSE.
        GO TO 400
      END IF
      IF (IFLAG .EQ. -2 .AND. BLKOUT) THEN
        BLKOUT = .FALSE.
        GO TO 400
      END IF
C
C*****
C*****
C   BLOCK BY BLOCK INTEGRATION
C
      150 CONTINUE
C
C   SEE IF WE ARE TOO CLOSE TO THE END POINT. IF SO, DO LINEAR
C   EXTRAPOLATION AND RETURN.
C
      IF (ABS(TEND-T) .LE. U26*ABS(T)) THEN
        IF (DTSIGN*(TEND-TOUT) .GT. 0.0E+0) THEN
          DT = TOUT - T
          ENDPNT = .FALSE.
          T = TOUT
        ELSE
          DT = TEND - T
          ENDPNT = .TRUE.
          T = TEND
          X = TEND
        END IF

```

A. Code to Implement Block 6(5) formula (3.61)

```

DO 160 K = 1, NEQN
    Y(K) = W(K)+DT*WP(K)
160  CONTINUE
    CALL FCN (T,Y,YP)
    NFE = NFE+1
    IF (ENDPNT) THEN
        DO 170 N=1,NEQN
            W(N) = Y(N)
            WP(N) = YP(N)
170  CONTINUE
        ENDPNT = .FALSE.
    END IF
    IFLAG = 2
    RETURN
END IF
C
C
C  SET SMALLEST ALLOWABLE STEPSIZE AND STEP FAILURE FLAG.
C  ADJUST STEPSIZE IF NECESSARY TO HIT THE END POINT OF INTEGRATION.
C
    HMIN = U26*ABS(T)
    HFAILD = .FALSE.
    HSTOP = 0.5E+0*(TEND-T)
    IF (ABS(HSTOP) .LE. ABS(H)) THEN
        ENDPNT = .TRUE.
        H = HSTOP
    END IF
C
C*****
C  CORE INTEGRATOR FOR A SINGLE BLOCK
C*****
C  THE TOLERANCES HAVE BEEN SCALED TO AVOID PREMATURE UNDERFLOW IN
C  COMPUTING THE ERROR TOLERANCE FUNCTION ERRTOL.
C  TO AVOID PROBLEMS WITH ZERO CROSSINGS,RELATIVE ERROR IS MEASURED
C  USING THE AVERAGE OF THE MAGNITUDES OF THE SOLUTION AT THE
C  BEGINNING AND END POINTS OF A BLOCK.
C  TO DISTINGUISH THE VARIOUS ARGUMENTS, H IS NOT PERMITTED
C  TO BECOME SMALLER THAN 26 UNITS OF ROUND OFF IN T.
C  PRACTICAL LIMITS ON THE CHANGE IN THE STEPSIZE ARE ENFORCED TO
C  SMOOTH THE STEPSIZE SELECTION PROCESS AND TO AVOID EXCESSIVE
C  CHATTERING ON PROBLEMS HAVING DISCONTINUITIES.
C*****
C
C
C  TEST NUMBER OF DERIVATIVE FUNCTION EVALUATIONS.
C  IF OKAY,TRY TO ADVANCE THE INTEGRATION FROM T TO T+H
C
200 IF (NFE .GT. MAXNFE) THEN
C
C  TOO MUCH WORK

```

A. Code to Implement Block 6(5) formula (3.61)

```

C
      IFLAG = 4
      KFLAG = 4
      GO TO 400
END IF

C
C   ADVANCE AN APPROXIMATE SOLUTION OVER ONE STEP OF LENGTH H
C
      DO 210 N = 1, NEQN
          Y1(N) = W(N)+A21*H*WP(N)
210 CONTINUE
          CALL FCN (T+C2*H,Y1,F2)
      DO 220 N = 1, NEQN
          Y1(N) = W(N)+H*(A31*WP(N)+A32*F2(N))
220 CONTINUE
          CALL FCN (T+C3*H,Y1,F3)
      DO 230 N = 1, NEQN
          Y1(N) = W(N)+H*(A41*WP(N)+A43*F3(N))
230 CONTINUE
          CALL FCN (T+C4*H,Y1,F4)
      DO 240 N = 1, NEQN
          Y1(N) = W(N)+H*(A51*WP(N)+A53*F3(N)+A54*F4(N))
240 CONTINUE
          CALL FCN (T+C5*H,Y1,F5)
      DO 241 N = 1,NEQN
          Y1(N)=W(N)+H*(A61*WP(N)+A63*F3(N)+A64*F4(N)
+           +A65*F5(N))
241 continue
          CALL FCN (T+C6*H,Y1,F6)
      DO 242 N=1,NEQN
          Y1(N)=W(N)+h*(A71*WP(N)+A73*F3(N)+A74*F4(N)+
+           A75*F5(N)+A76*F6(N))
242 continue
          CALL FCN (T+C7*H,Y1,Y)
      DO 250 N = 1, NEQN
          Y1(N) = W(N)+H*(A81*WP(N)+A83*F3(N)+A84*F4(N)+
*           A85*f5(N)+A86*F6(N)+A87*Y(N))
250 CONTINUE
          CALL FCN (T+C8*H,Y1,YP)
          NFE = NFE+7
          EEOET = 0.0E+0
      DO 270 N = 1, NEQN
          Y1(N) = W(N)+H*(B1B*WP(N)+B4B*F4(N)+B5B*F5(N)+
*           B6B*F6(N)+B7B*Y(N)+B8B*YP(N))
          ERRTOL = ABS(W(N))+ABS(Y1(N))+AE
          IF (ERRTOL .LE. 0.0E+0) THEN
C
C           INAPPROPRIATE ERROR TOLERANCE
C
          IFLAG = 5

```

A. Code to Implement Block 6(5) formula (3.61)

```

        KFLAG = 5
        GO TO 400
    END IF
    EZ = ABS(H*(ERC11*WP(N)+ERC14*F4(N)+ERC15*F5(N)+
*       ERC16*F6(N)+ERC17*Y(N)+B8B*YP(N)))
    EEOET = MAX(EEOET,EZ/ERRTOL)
270 CONTINUE
    ESTTOL = EEOET*SCALE
C
C CHECK THE ERROR ESTIMATE. IF STEPLENGTH HAS FAILED FOR FIRST STEP
C IN THE BLOCK, GO AND COMPUTE A SMALLER STEP FOR A RE-TRY.
C
    IF (ESTTOL .GT. 1.0E+0) GO TO 320
C
C
C INTEGRATION SUCCESSFUL FOR FIRST STEP. NOW INTEGRATE OVER
C THE SECOND STEP IN THE BLOCK
C
    CALL FCN (T+H,Y1,F1)
    DO 290 N = 1, NEQN
        Y2(N) = W(N)+H*(A101*WP(N)+A103*F3(N)+A104*F4(N)+
*       A105*F5(N)+A106*F6(N)+A107*Y(N)+A108*YP(N)+A109*f1(N))
290 CONTINUE
    CALL FCN (T+C10*H,Y2,F7)
    DO 300 N = 1, NEQN
        Y2(N) = W(N)+H*(A111*WP(N)+A113*F3(N)+A114*F4(N)+
*       A115*F5(N)+A116*F6(N)+A117*Y(N)+A118*YP(N)+A119*F1(N)
*       +A1110*F7(N))
300 CONTINUE
    CALL FCN (T+C11*H,Y2,F2)
    DO 301 N=1,NEQN
        Y2(N)=W(N)+H*(A121*WP(N)+A123*F3(N)+A124*F4(N)+A125*
+       F5(N)+A126*F6(N)+A127*Y(N)+A128*YP(N)+A129*F1(N)+A1210*F7(N)
+       +A1211*F2(N))
301 CONTINUE
    CALL FCN(T+2*H,Y2,F3)
    NFE = NFE+4
    EEOET = 0.0E+0
    DO 310 N = 1, NEQN
        EE = ABS(H*(ERC21*WP(N)+ERC24*F4(N)+ERC25*F5(N)+
*       ERC26*F6(N)+ERC27*Y(N)+ERC28*YP(N)+ERC29*F1(N)
*       + ERC210*F7(N)+ERC211*F2(N)+ERC212*F3(N)))
        Y2(N) = W(N)+H*(B1*WP(N)+B4*F4(N)+B5*F5(N)+B6*F6(N)+
*       B7*Y(N)+B8*YP(N)+B9*F1(N)+B10*F7(N)+B11*F2(N)
*       + B12*F3(N))
        IF (ERRTOL .LE. 0.0E+0) THEN
C
C     INAPPROPRIATE ERROR TOLERANCE
C
        IFLAG = 5

```

A. Code to Implement Block 6(5) formula (3.61)

```

        KFLAG = 5
        GO TO 400
    END IF
    EEOET = MAX(EEOET,EE/ERRTOL)
310 CONTINUE
    ESTTOL = EEOET*SCALE
    DO 315 N = 1, NEQN
        Y1(N) = B1B*WP(N)+B4B*F4(N)+B5B*F5(N)+
*           B6B*F6(N)+B7B*Y(N)+B8B*YP(N)
        F3(N) = B1*WP(N)+B4*F4(N)+B5*F5(N)+B6*F6(N)+
*           B7*Y(N)+B8*YP(N)+B9*F1(N)+B10*F7(N)+
*           B11*F2(N)+B12*F3(N)
315 CONTINUE
C
C CHECK THE ERROR ESTIMATE OVER THE SECOND STEP OF THE BLOCK.
C
    IF (ESTTOL .LE. 1.0E+0) GO TO 330
C
C UNSUCCESSFUL BLOCK
C REDUCE THE STEPSIZE , TRY AGAIN
C THE DECREASE IS LIMITED TO A FACTOR OF ABOUT 1/10
C
320 HFAILD = .TRUE.
    S = 0.1E+0
    ENDPNT = .FALSE.
    IF (ESTTOL.LT.1.0E+6) S = SF/ESTTOL**0.167E+0
    H = S*H
    IF (ABS(H) .LT. HMIN) THEN
C
C REQUESTED ERROR UNATTAINABLE AT SMALLEST ALLOWABLE STEPSIZE
C
        IFLAG = 6
        KFLAG = 6
        GO TO 400
    END IF
    GO TO 200
C
C SUCCESSFUL BLOCK. CHECK FOR NEED TO INTERPOLATE.
C STORE SOLUTION AT T+2*H.
C
330 T = T+2.0E+0*H
    IF (ENDPNT) T = TEND
    TOLD = X
    X = T
    HINT = H
    INTERP = .FALSE.
    IF ((T-TOUT)*DTSIGN .GE. 0.0E+0) INTERP = .TRUE.
    IF (INTERP) THEN
        DO 340 N=1,NEQN
            SWAP = W(N)

```

A. Code to Implement Block 6(5) formula (3.61)

```

      W(N) = Y2(N)
      Y2(N) = SWAP
      F2(N) = WP(N)
340   CONTINUE
      ELSE
      DO 350 N=1,NEQN
      W(N) = Y2(N)
350   CONTINUE
      END IF
C
      A = T
      CALL FCN (A,W,WP)
      NFE = NFE+1
C
C   CHOOSE NEXT STEPSIZE
C   THE INCREASE IS LIMITED TO A FACTOR OF ABOUT 10
C   IF STEP FAILURE HAS JUST OCCURED, NEXT STEP IS NOT ALLOWED
C   TO INCREASE.
C
      S = 10.0E+0
      IF (ESTTOL.GT.1.E-6) S = SF/ESTTOL**0.167E+0
      IF (HFAILD) S = MIN(S,1.0E+0)
      H = SIGN(MAX(S*ABS(H),HMIN),H)
C
C   HAVE WE INTEGRATED PAST AN OUTPUT POINT?
C   IF SO, CALL THE INTERPOLATION ROUTINE AT T=TOUT.
C   IF NOT, SEE IF WE ARE AT THE END POINT OF INTEGRATION.
C   OTHERWISE, CHECK IF USER WANTS SOLUTIONS AT THE END OF THE BLOCK
C   STEP, OR ELSE CONTINUE THE INTEGRATION.
C
      IF (INTERP) GO TO 390
      IF (ENDPNT) THEN
      IFLAG = 2
      ENDPNT = .FALSE.
      GO TO 400
      END IF
      IF (IFLAG .LT. 0) THEN
      IFLAG = -2
      GO TO 400
      ELSE
      GO TO 150
      END IF
C
C   INTERPOLATE TO GET DATA AT OFF-STEP POINT AND RETURN.
C
C   THE SECOND OF THE THREE CHANGES NEEDED TO MAKE THE INTERPOLANT
C   DERIVATIVE MORE STABLE IS TO REPLACE STATEMENT 390 BY THE
C   VERSION IN THE COMMENT STATEMENT BELOW
C
C 390 CALL EXTRA (NEQN,W,WP,X,HINT,Y1,Y2,F1,F2,TOUT,Y,YP)
```


A. Code to Implement Block 6(5) formula (3.61)

```

390 CALL EXTRA (NEQN,F3,WP,X,HINT,Y1,Y2,F1,F2,TOUT,Y,YP)
      T = TOUT
      IF (IFLAG .LT. 0 .AND. TOUT .NE. X) BLKOUT = .TRUE.
      IFLAG = 2
      RETURN
C
C RETURN WITH THE SOLUTION AT THE END OF THE LAST SUCCESSFUL
C BLOCK STEP.
C
400 DO 410 N=1,NEQN
      Y(N) = W(N)
      YP(N) = WP(N)
410 CONTINUE
      RETURN
C
      END

      SUBROUTINE EXTRA (NEQN,YINC2,F2,T,H,YINC1,Y,F1,F,TEX,YEX,YPEX)
C
C T IS THE INDEPENDENT VARIABLE;
C Y IS THE VALUE AT T-2H; THE VALUES AT T-H AND T ARE
C Y + H*YINC1 AND Y + H*YINC2 RESPECTIVELY;
C F,F1 AND F2 ARE THE DERIVATIVES AT T-2H, T-H AND T.
C TEX IS THE POINT WHERE THE OUTPUT IS REQUIRED;
C YEX WILL HOLD THE APPROXIMATE SOLUTION AT TEX AND YPEX WILL
C HOLD THE DERIVATIVE APPROXIMATION AT THIS POINT.
C
      DIMENSION Y(NEQN),YINC1(NEQN),YINC2(NEQN),F(NEQN),F1(NEQN),
*           F2(NEQN),YEX(NEQN),YPEX(NEQN)
C
C PERFORM QUINTIC INTERPOLATION BASED ON THE VALUES
C Y(N),Y(N)+H*YINC1(N),Y(N)+H*YINC2(N),F(N),F1(N),F2(N) AT THE
C POINTS T-2H, T-H, T. THIS POLYNOMIAL IS EVALUATED AT THE POINT
C TEX TO OBTAIN THE APPROXIMATE VALUE OF Y(TEX) AND THIS IS STORED
C IN THE ARRAY YEX. THE DERIVATIVE OF THE INTERPOLATING POLYNOMIAL
C IS ALSO EVALUATED AT TEX TO OBTAIN AN APPROXIMATION TO DY/DT
C AT THIS POINT AND THIS APPROXIMATION IS STORED IN YPEX.
C
C
      SIG = (TEX-T)/H+2.0E+0
      DO 10 N = 1, NEQN
          HF = H*F(N)
          HF1 = H*F1(N)
          HF2 = H*F2(N)
          A1 = 2.0E+0*(HF2+HF)-6.0E+0*H*YINC2(N)+8.0E+0*HF1
          A2 = H*YINC2(N)+4.0E+0*H*YINC1(N)-4.0E+0*HF1-2.0E+0*HF
          A3 = HF1+HF-2.0E+0*H*YINC1(N)
          A4 = H*YINC1(N)-HF

```

A. Code to Implement Block 6(5) formula (3.61)

```
YEX(N) = (((A1*0.125E+0*(SIG-2.0E+0)+0.25E+0*A2)*(SIG-1.0E+0)+
*
A3)*(SIG-1.0E+0)+A4)*SIG+HF)*SIG+Y(N)
YPEX(N) = (((0.625E+0*A1*SIG+(A2-2.0E+0*A1))*SIG+(1.875E+0*A1-
*
1.5E+0*A2+3.0E+0*A3))*SIG+(0.5E+0*(A2-A1)-
*
2.0E+0*(A3-A4)))*SIG+HF)/H
10 CONTINUE
RETURN
END
```