Copyright Statement

The digital copy of this thesis is protected by the Copyright Act 1994 (New Zealand). This thesis may be consulted by you, provided you comply with the provisions of the Act and the following conditions of use:

- Any use you make of these documents or images must be for research or private study purposes only, and you may not make them available to any other person.

- Authors control the copyright of their thesis. You will recognise the author's right to be identified as the author of this thesis, and due acknowledgement will be made to the author where appropriate.

- You will obtain the author's permission before publishing any material from their thesis.

To request permissions please use the Feedback form on our webpage. http://researchspace.auckland.ac.nz/feedback

General copyright and disclaimer

In addition to the above conditions, authors give their consent for the digital copy of their work to be used subject to the conditions specified on the Library Thesis Consent Form
The Effective Order of Singly–Implicit Methods for Stiff Differential Equations

David Jen Lung Chen

A thesis submitted for the degree of
Doctor of Philosophy at The University of Auckland

Department of Mathematics
June, 1998
ABSTRACT

Singly-implicit Runge-Kutta methods (SIRK) are designed for stiff differential equations. The existing code STRIDE based on these methods has been shown to be efficient for stiff problems, especially for high dimensional problems. However, SIRK methods with order greater than 2 possess the undesirable property that some of their abscissae are outside the integration interval. In order to improve the numerical behaviour of SIRK methods, we need to overcome this drawback. While retaining the original advantages of SIRK methods as much as possible, it would be advantageous to have more free parameters in choosing the coefficients for these methods.

Recently, two generalizations of SIRK methods were introduced to overcome this difficulty. One is the so-called "DESI" (Diagonally Extended Singly-Implicit Runge-Kutta) method in which some additional diagonally implicit stages are added to the corresponding classical SIRK method. It turns out that there is more freedom in choosing the abscissae because of these extra stages. The other generalization is the so-called "ESIRK" (Effective order Singly-Implicit Runge-Kutta) method which adopts the idea of "effective order" so that the desirable free parameters come from "perturbed" initial values. The first approach has been verified to be a successful generalization. The existing variable order code DESI was shown to be more efficient than STRIDE, and competes well with the BDF (Backward Differentiation Formulae) code LSODE for many stiff problems (Butcher, Cash, Diamantakis [24] 1996).

For the second approach, the numerical behaviour of ESIRK methods with variable stepsize, is closely related to the choice of the abscissae. In this thesis, it is shown that the classical SIRK methods are not the best choice with respect to the local truncation error. We analyze the numerical behaviour of the ESIRK methods both theoretically and experimentally. The choices of the abscissae for these methods are investigated. It is found that except when \( s = 2 \) (\( s \) is the number of stages in the method), the numerical results obtained with equally spaced abscissae in \([0,1]\) are better than the corresponding SIRK methods for
Several alternative choices are also given. Some experimental variable-stepsize ESIRK codes are designed and are compared with the famous IRK codes SDIRK4, RADAU5 and the BDF code LSODE. The numerical results show that ESIRK methods are successful generalizations of the SIRK methods and are good candidates as solvers for stiff problems.

In attempting to increase the efficiency of ESIRK methods, the idea of adding some additional diagonal stages is proposed. The generalizations of the ESIRK methods, called "EDES" (Effective order Diagonally Extended Singly-Implicit Runge-Kutta) methods, are shown to be promising in solving stiff problems and are also successful generalizations of DESI and ESIRK methods.
ACKNOWLEDGEMENTS

This study has been completed in the Mathematics Department at The University of Auckland. First, I am very grateful to my supervisor Prof. John C. Butcher, who has spent a lot of time guiding and encouraging me. I wish to express my deep gratitude to him for his kindness, patience and valuable guidance during this study. In fact, I have had many advantages from this supervision as he is a pioneer in this field. One of these benefits is that I have had many chances to discuss my work with experts in this field visiting Auckland. I would like to express my appreciation to these visitors. In particular, I wish to thank Prof. Kevin Burrage for his useful comments on this thesis, and Dr. Michalis Diamantakis for sharing his knowledge and discussing my work with me.

Secondly, I wish to express my deep appreciation to my advisor Dr. Robert Chan, for his encouragement, patience and valuable advice during this study. I would also like to thank my advisor Dr. Philip Sharp for his useful advice and comments. The regular workshops organized by John and Robert have given me many valuable opportunities to present parts of my work. I also wish to extend my thanks to all the participants in these workshops, especially Dr. Allison Heard and my fellow doctoral students, Tina Chan and Anjana Singh. My special thanks also to Dorothy Brown and Nicolette Goodwin, who have kindly read through the manuscript of this thesis.

The Mathematics Department of The University of Auckland is a wonderful place to work in. I would like to thank all the staff in this department for the tremendous research and study environment they have created. I would also like to express my thanks to my badminton partners, who have created wonderful opportunities for relaxation during the past three years.

In addition, I wish to express my gratitude to Ling-Tung Technical Institute in Taiwan for their financial support in paying the tuition fees during this study. The financial aid for me to attend the conferences "Numdiff 8" in Germany and "SciCADE97" in Italy came from the New Zealand Mathematical Society, the Graduate Research Committee and the Mathematics Department of The

v
University of Auckland, and the Marsden Funds of Prof. John Butcher. I would like to express my appreciation for the opportunity to attend these conferences.

I am very grateful to my parents Chen Tsang Chou and Weng Hsueh, who have not only given me financial assistance, but have also spent a lot of time looking after my children. I wish to express my deep appreciation to them for their encouragement and consideration. I would like to thank my brother Lance Chen and my sister Sandy Chen for their financial support and their consideration.

Finally, I would like to express my deep gratitude to my wife, Tina Ming-hua Chan, who is always considerate and patient. I am grateful to her and my children for their understanding and love.
# Contents

1 Introduction \hspace{10cm} 1
  \hspace{1cm} 1.1 Review of singly-implicit Runge-Kutta methods \hspace{1cm} 1
  \hspace{1cm} 1.2 The aims and the framework of this thesis \hspace{1cm} 9

2 Numerical methods for stiff differential equations \hspace{10cm} 11
  \hspace{1cm} 2.1 Stiff problems \hspace{1cm} 12
  \hspace{1cm} 2.2 Stability requirements \hspace{1cm} 17
  \hspace{1cm} 2.3 Implicit Runge–Kutta methods \hspace{1cm} 23

3 Singly–Implicit Methods \hspace{10cm} 37
  \hspace{1cm} 3.1 Butcher’s transformation \hspace{1cm} 38
  \hspace{1cm} 3.2 SIRK methods and their stability \hspace{1cm} 44
  \hspace{1cm} 3.3 The transformation matrix \hspace{1cm} 48
  \hspace{1cm} 3.4 DESI methods \hspace{1cm} 52

4 The effective order of SIRK methods \hspace{10cm} 63
  \hspace{1cm} 4.1 Effective order \hspace{1cm} 64
  \hspace{1cm} 4.2 Effective order conditions for SIRK methods \hspace{1cm} 76
## CONTENTS

4.3 Doubly Companion Matrices ........................................... 87
4.4 ESIRK methods .......................................................... 99
4.5 Variable stepsize for ESIRK methods .............................. 113
4.6 Study of a systematic stepsize change pattern .................... 118
  4.6.1 Local truncation error .......................................... 122
  4.6.2 Stability considerations ....................................... 142
4.7 Error estimation for changing stepsize and order ................. 150
4.8 Implementation and some numerical results ....................... 166

5 The design of an EDESI integrator ................................... 191
  5.1 Construction of EDESI methods .................................. 192
  5.2 Variable step-size for EDESI methods .......................... 206
  5.3 Implementation ..................................................... 211
  5.4 Numerical results .................................................. 214

A Mathematica and Matlab programs ................................. 245
  4.1 Transformation matrix $T = WV$ for ESIRK methods .......... 245
  4.2 Stability function $R(z, r)$ ...................................... 249
  4.3 Local truncation error ............................................ 253
    A.3.1 Error ratio $\varepsilon(r, c)$ ................................ 253
    A.3.2 Normalized error constant $\hat{C}(r, c)$ ................. 255

Bibliography ................................................................. 258
List of Figures

1.1 Runge-Kutta methods for stiff problems ........................................ 8
2.1 The components of the Robertson problem (2.6) .............................. 15
2.2 The first component of the Van der Pol problem (2.7) with $\theta = 10^6$ 17
2.3 The second component of (2.7) with $\theta = 10^6$ .............................. 17
4.1 Integration procedure for effective order methods ............................ 66
4.2 Effective order method $\phi$ and the starting method $\psi$ ..................... 67
4.3 Order-2 SIRK: --, $[\frac{7-4\sqrt{2}}{3}, 1]$: +, $[0, 1]$: · · ·, order-3 SIRK: --,
                  $[0, \frac{1}{5}, 1]$: ×, $[0, \frac{1}{2}, 1]$: ○ .............................................. 109
4.4 Order-2 SIRK: --, $[\frac{7-4\sqrt{2}}{3}, 1]$: +, $[0, 1]$: · · ·, order-3 SIRK: --,
                  $[0, \frac{1}{5}, 1]$: ×, $[0, \frac{1}{2}, 1]$: ○ .............................................. 109
4.5 Variable stepsize scheme for ESIRK methods .................................. 113
4.6 Stepsize changing pattern for studying stability function of variable
                  stepsize ESIRK methods ....................................................... 120
4.7 $s = 2$, Graph of $\bar{e}(c_1) = \frac{\partial^2}{\partial r^2} e(r, c_1)|_{r=1}$ ...................... 124
4.8 $s = 2$, error ratio $e(r, c_1)$ of variable stepsize against constant stepsize 125
4.9 $s = 3$, $c_1 = 0$, $y$-axis : $\bar{e}(c_2) = \frac{\partial^2}{\partial r^2} e(r, c_2)|_{r=1}$, $x$-axis : $c_2$ .............. 126
4.10 $s = 3$, error ratio $e(r, c)$ of variable stepsize/constant stepsize ......... 126
4.11 $s = 3$, $c_1 = -1$, $y$-axis : $\varepsilon(c_2) = \frac{\partial^2}{\partial c_1^2} \varepsilon(r, c_2)|_{r=1}$, $x$-axis : $c_2$ ....... 127

4.12 $s = 3$, error ratio $\varepsilon(r, c)$ of variable stepsize/constant stepsize ....... 127

4.13 The error ratio $\varepsilon(r, c)$ for ESIRK methods $(0, \frac{1}{2}, 1), (-1, 0, 1)$. The curve $\varepsilon(r, c)$ for $(-1, 0, 1)$ is concave down when $r \in (0.844405, 1.18427)$ and is only smaller than the the curve of $(0, \frac{1}{2}, 1)$, when $r \in (0.65, 1.53846)$. ....................... 128

4.14 Order-2 SIRK, ESIRK methods $(0, 1), \left(\frac{7-4\sqrt{3}}{3}, 1\right)$ ....................... 129

4.15 Order-2 SIRK, ESIRK methods $(0, 1), \left(\frac{7-4\sqrt{3}}{3}, 1\right)$ ....................... 131

4.16 Order-3 SIRK, ESIRK methods $(0, \frac{1}{2}, 1), (0, \frac{1}{3}, 1)$ ....................... 131

4.17 Second derivative of the normalized error constant $\hat{C}(r, c_1)|_{r=1}$ for $s = 2$ .......................... 133

4.18 Normalized error constant $\hat{C}(r, c)$ for $s = 2$ SIRK, ESIRK methods 134

4.19 Normalized error constant $\hat{C}(r, c)$ of SIRK, ESIRK $E_2 = \left(\frac{7-4\sqrt{3}}{3}, 1\right)$, $E_2 = \left(\frac{5}{4}, 1\right)$ for $s = 2$ ....................... 134

4.20 ContourPlot of $\frac{\partial^2}{\partial c_1^2} \hat{C}(r, c_1, c_2)|_{r=1}$, $\hat{C}(r)$ is the normalized error constant for $s = 3$ ESIRK, $y$-axis : $c_2$, $x$-axis : $c_1$ .......................... 135

4.21 $s = 3$, Normalized error constants $\hat{C}(r)$ of SIRK, ESIRK $E_1 : (0, \frac{1}{2}, 1), E_2 : (0, \frac{1}{2}, 1), E_3 : (0, -\frac{1}{2}, 1), E_5 : (-\frac{3}{2}, -1, 1), E_4 : (-1, -\frac{5}{16}, 1)$ ....................... 136

4.22 $s = 4$, Normalized error constants $\hat{C}(r)$ of SIRK, ESIRK $E_1 : (0, \frac{1}{3}, \frac{2}{3}, 1), E_2 : (-1, -\frac{1}{2}, 0, 1), E_3 : (-1, -\frac{1}{2}, 1), E_4 : (-2, -1, 0, 1)$ 139

4.23 Normalized error constants $\hat{C}(r)$ of order-5 SIRK and ESIRK methods $E_1 : (0, \frac{1}{4}, \frac{2}{4}, \frac{3}{4}, 1)$, and $E_2 : (-1, -\frac{1}{2}, 0, \frac{1}{2}, 1)$, and $E_3 : (-2, -1, -\frac{1}{2}, 0, 1)$, and $E_4 : (-3, -2, -1, 0, 1)$ ....................... 140

4.24 Normalized error constants $\hat{C}(r)$ of order-6 SIRK and ESIRK methods $E_1 : (0, \frac{1}{3}, \frac{2}{3}, \frac{3}{3}, \frac{4}{3}, 1)$, and $E_2 : (-1, -\frac{2}{3}, -\frac{1}{3}, \frac{1}{3}, \frac{2}{3}, 1)$, and $E_3 : (-2, -\frac{3}{2}, -1, -\frac{1}{2}, 0, 1)$, and $E_4 : (-4, -3, -2, -1, 0, 1)$ ....................... 140
4.25 Normalized error constants $\tilde{C}(r)$ of order-8 SIRK, ESIRK methods

$E_1 : (0, \frac{1}{7}, \frac{2}{7}, \frac{3}{7}, \frac{4}{7}, \frac{5}{7}, \frac{6}{7}, 1)$, $E_2 : (-1, -\frac{3}{4}, -\frac{2}{4}, -\frac{1}{4}, \frac{1}{4}, \frac{2}{4}, \frac{3}{4}, 1)$, and $E_3 : (-2, -\frac{5}{3}, -\frac{4}{3}, -\frac{2}{3}, -\frac{1}{3}, \frac{1}{3}, \frac{2}{3}, 1)$, $E_4 : (-3, -\frac{5}{2}, -2, -\frac{5}{2}, -1, -\frac{1}{2}, 0, 1)$

4.26 Error estimation of order-2 ESIRK for the Kaps problem

4.27 Work/Precision diagrams for order-4 ESIRK methods, A-stable:

- $A(\alpha = 1.56)$-stable: --

4.28 Work/Precision diagrams for order-6 ESIRK methods, A-stable:

- $A(\alpha = 1.49)$-stable: --

4.29 Work/Precision diagrams for order 4, 5, 6 ESIRK methods, order

4: +, order 5: o, order 6: x

4.30 Work/precision diagram of ESIRK4 and SDIRK4

4.31 Work/precision diagram of ESIRK5, LSODE and RADAU5

4.32 Work/precision diagram of ESIRK5, LSODE and RADAU5

4.33 Work/precision diagram of ESIRK5, LSODE and RADAU5

5.1 Error vs. stepsize for order-2 singly-implicit methods with constant stepsize

5.2 Flops vs. error for order-2 singly-implicit methods with constant stepsize

5.3 Work/Precision diagram (flops/maximum error) for EDESI methods solving the Kaps problem, $s = p + 1$: --, $s = p + 2$: --

5.4 Work/precision diagrams of order-2 singly-implicit methods, SIRK:

+", DESI: *, ESIRK: o, EDESI: x

5.5 Work/precision diagrams of order-3 singly-implicit methods, SIRK:

+", DESI: *, ESIRK: o, EDESI: x

5.6 Work/precision diagrams of order-4 singly-implicit methods, SIRK:

+", DESI: *, ESIRK: o, EDESI: x
List of Tables

2.1 Summary of the stability, stiff accuracy and orders of some A-stable one-step methods ........................................ 27

3.1 Total approximate number of operations for solving $N$ dimensional systems ....................................................... 42

3.2 Total flops for solving the Kaps problem (3.9) ............................ 44

3.3 $\lambda$ for L-stability with $s$-stage SIRK methods ........................................ 47

3.4 Error constants of SIRK and DESI .............................................. 58

3.5 Numerical results for Kaps (3.9) using order-2 SIRK and DESI ... 60

4.1 $F(t)(y)$, $\gamma(t)$, $\alpha(t)$ for $\rho(t) \leq 4$ ................................. 73

4.2 Number of order conditions for effective order $\geq s$ ................. 76

4.3 Absolute value of error constant for ESIRK methods .................. 107

4.4 The global error for the Kaps problem using $s = 2,3$ SIRK and ESIRK methods with constant stepsize ...................... 110

4.5 The average iteration number for the Kaps problem using $s = 2$ SIRK and ESIRK methods, $tol$: tolerance for stopping the iteration.111

4.6 The average iteration number for the Kaps problem using $s = 3$ SIRK and ESIRK methods, $tol$: tolerance for stopping the iteration.112
## LIST OF TABLES

4.7 Order 2, 3 SIRK, ESIRK methods with stepsize changing pattern $h, rh, h_{\ldots}$ .................................................. 130

4.8 $s = 2, 3$, Normalized error ratio (ESIRK/SIRK) for solving the Kaps problem ................................................... 137

4.9 Normalized error ratio (ESIRK/SIRK) using order-3 methods for DETEST $A_1, D_1$ .................................................. 139

4.10 $r$-intervals for A-stability for order 2, 3 ESIRK methods .......................... 145

4.11 $r$-intervals for A-stability for order 4, 5 ESIRK methods .......................... 146

4.12 $r$-intervals for A-stability for order 6, 8 ESIRK methods .......................... 147

4.13 Proper abscissae for ESIRK methods ...................................... 149

4.14 Numerical results for stiff DETEST $A_1, A_2, A_3, D_1, D_2, D_3$ with order-2 SIRK, ESIRK methods ........................................ 151

4.15 Numerical results for A-group of stiff DETEST with order-3 SIRK, ESIRK methods ........................................ 152

4.16 Numerical results for B-group of stiff DETEST with order-3 SIRK, ESIRK methods ........................................ 153

4.17 Results for C-group of stiff DETEST with order-3 SIRK, ESIRK methods ........................................ 154

4.18 Numerical results for D-group of stiff DETEST with order-3 SIRK, ESIRK methods ........................................ 155

4.19 Numerical results for E-group of stiff DETEST with order-3 SIRK, ESIRK methods ........................................ 156

4.20 Results when using previous solution value (A), predictor (4.73) (B) and predictor (4.74) (C) with order-2 ESIRK method ............... 172

4.21 Results when using previous solution value (A), predictor (4.73) (B) and predictor (4.74) (C) with order-3 ESIRK method ............... 173
4.22  $\frac{1}{\lambda}$ of A($\alpha$)-stable ESIRK methods with $\alpha \geq 1.45$, the value of $\alpha$ is given in parentheses  

4.23 Numerical results for Van der Pol (2.7) by testing order-4 L-stable, A($\alpha$)-stable ESIRKs and order-4 L-stable SDIRK  

4.24 Numerical results for Van der Pol (2.7) problem by testing LSODE, RADAU5 and order-5 L-stable ESIRK  

4.25 Numerical results for Robertson problem (2.6) by testing LSODE, RADAU5 and order-5 L-stable ESIRK  

4.26 Numerical results for Oregonator (4.77) by testing LSODE, RADAU5 and order-5 L-stable ESIRK  

5.1 Numerical results for the Kaps problem using order 2 SIRK, DESI, ESIRK and EDESI ($s = p + 1, s = p + 2$) with constant stepsize  

5.2 Results when using predictor (4.74) (A) and previous solution value (B) with order 2, 3 EDESI  

5.3 Efficiency measurement of some singly-implicit methods  

5.4 Summary of SIRK and DESI  

5.5 Summary of ESIRK and EDESI  

5.6 Numerical results for Curtis (5.23) by testing order-2 singly-implicit methods  

5.7 Numerical results for Prothero-Robinson (2.5) by testing order-2 singly-implicit methods  

5.8 Numerical results for Kaps (3.9) by testing order-2 singly-implicit methods  

5.9 Numerical results for Oregonator (4.77) by testing order-2 singly-implicit methods  

5.10 Numerical results for Robertson (2.6) by testing order-2 singly-implicit methods  

xv
LIST OF TABLES

5.11 Numerical results for Van der Pol (2.7) by testing order-2 singly-implicit methods ........................................ 231
5.12 Numerical results for Prothero-Robinson by testing order-3 methods 232
5.13 Numerical results for Curtis problem by testing order-3 methods . 233
5.14 Numerical results for Kaps by testing order-3 methods ........... 234
5.15 Numerical results for Oregonator by testing order-3 methods ... 235
5.16 Numerical results for Robertson by testing order-3 methods .... 236
5.17 Numerical results for Van der Pol by testing order-3 methods ... 237
5.18 Numerical results for Prothero-Robinson by testing order-4 methods 238
5.19 Numerical results for Curtis problem by testing order-4 methods . 239
5.20 Numerical results for Kaps by testing order-4 methods .......... 240
5.21 Numerical results for Oregonator by testing order-4 methods .... 241
5.22 Numerical results for Robertson by testing order-4 methods .... 242
5.23 Numerical results for Van der Pol by testing order-4 methods ... 243
Chapter 1

Introduction

1.1 Review of singly-implicit Runge-Kutta methods

Numerical methods for the solution of the initial value problem (IVP)
\[ y'(x) = f(x, y(x)), \quad y(x_0) = y_0, \quad f: \mathbb{R} \times \mathbb{R}^N \to \mathbb{R}^N, \]
mainly lie in one of two large classes: multistep methods and multistage methods. Multistep methods advance the solution approximation by using a linear combination of the past approximations to the solution. Instead of using past values, multistage methods compute the derivative \( f \) several times within each step. Furthermore, by combining these two types of methods, many generalizations can be found. For example, Hybrid methods which are generalized multistep methods allowing a few evaluations of \( f \) with one step, Rosenbrock type methods which are generalized Runge-Kutta methods linearizing the nonlinear systems of the implicit Runge-Kutta methods in order to reduce the computational cost, high derivative multistep methods using higher derivatives, and general linear methods combining the essential features of linear multistep methods and Runge-Kutta methods.

These methods can also be divided into two main groups: explicit methods and implicit methods. A easy and quick way to distinguish these two types of meth-
ods is that the implicit methods need an iteration scheme, usually Newton type iteration, during the integration, whereas the explicit methods do not. Therefore, the computations for implicit methods are more expensive than for explicit methods. In addition to the implementation of the methods, accuracy and stability are two further factors for judging the efficacy of a method. In the past forty years, numerical analysts have focussed on these three topics in order to obtain high accuracy (or high order; we will discuss this in the next chapter) methods with good stability and low computational cost. Unfortunately, it is not easy to achieve all these goals with the same type of method. Linear multistep methods, especially explicit methods, have advantages in terms of computational cost, but they have barriers for high accuracy and good stability. On the other hand, implicit Runge-Kutta methods have high accuracy and good stability, but their computational costs are high.

In the early fifties, a special phenomenon for some IVPs was discovered which is now referred to as "stiffness". The characteristic widely-varying time constants for stiff problems force some traditional methods to use very small stepsizes in some integration intervals. This discovery led to a new concept of stability. A property known as "A-stability" was introduced to characterize methods that are suitable for stiff problems. Many investigations mainly concerning methods that possess A-stability have been made. A remarkable result of these studies is that no explicit methods can be A-stable. Therefore, amongst these classical numerical methods, good candidates for stiff problems are limited to implicit methods. Furthermore, another important result is that the maximum order of an A-stable linear multistep method is two. This also guides the study of the methods for stiff problems.

Amongst the class of linear multistep methods used at present, the only ones which have stability good enough to cope with the problem of stiffness are the BDF methods (Backward Differentiation Formulae). Unlike the traditional multistep methods, BDF formulae are based on the numerical differentiation of a given function rather than using the quadrature formulae to approximate the integral. In spite of the fact that linear multistep methods have poor stability properties, codes based on BDF are very efficient and perform well in practice.
One of the reasons for this success is that the implementation cost for implicit linear multistep methods is lower than implicit Runge-Kutta methods. For any implicit multistep method, at every integration step, the total costs for the solution of a nonlinear system of \( N \) equations require \( O(N^3) \) operations for LU factorization and \( O(N^2) \) operations for back substitutions. Another reason for this success is that “almost A-stable” BDF methods can be constructed up to order 6. For many stiff problems whose eigenvalues do not have large imaginary components, the stability requirement does not limit the stepsize.

According to [28], there are several “first inventors” for implicit Runge-Kutta methods (IRK). Cauchy found them by “integration”. Hammer and Hollingsworth found them by the idea of “collocation” (quadratic function) [46]. Butcher explores them with the famous so-called “simplifying order conditions” and gives useful systematic theories [7]. No matter which approach, they all represent an important role in the study of implicit Runge-Kutta methods. But it is undeniable that the implicit Runge-Kutta methods which have been widely noticed and used are based on the third approach. Compared with implicit linear multistep methods, Runge-Kutta methods have more chance of obtaining better stability. Furthermore, it has been shown that for any accuracy requirement, it is possible to find A-stable IRK methods. Therefore, it is natural to use IRK methods as candidates for stiff problems.

If \( h \) is the stepsize, then for each integration step, the process of an \( s \)-stage IRK method is of the form

\[
Y_i = y_0 + h \sum_{j=1}^{s} a_{ij} f(x_0 + c_i h, Y_j) , \quad i = 1, \ldots, s , \tag{1.1}
\]

\[
y_1 = y_0 + h \sum_{i=1}^{s} b_i f(x_0 + c_i h, Y_i) , \tag{1.2}
\]

or denoted by the “Butcher tableau”,

\[
c \begin{bmatrix} A \\ b^T \end{bmatrix} , \tag{1.3}
\]

where \( c = [c_1, c_2, \ldots, c_s]^T , A = [a_{ij}]_{s \times s} , \) and \( b^T = [b_1, b_2, \ldots, b_s] \). In spite of their excellent stability properties, IRK methods are expensive to implement relative
to linear multistep methods because the nonlinear system (1.1) of an s-stage IRK method now has \( sN \) equations. This means that using the Newton method at every iteration step, the total operations become \( O(s^3N^3) + O(s^2N^2) \). The costs increase rapidly as \( s \) or \( N \) increases.

The aim of reducing the computational cost for IRK methods leads to the so-called "DIRK" (or SDIRK) methods. The idea of the DIRK methods is trivial. Instead of using a full coefficient matrix \( A \) in (1.3), one can use a lower triangular matrix \( A \) for the method. In this case, the nonlinear system (1.1) can be solved stage by stage sequentially. The total operation number can be reduced from \( O(s^3N^3) + O(s^2N^2) \) to \( O(sN^3) + O(sN^2) \). This is a considerable saving, especially for large dimension systems. Furthermore, when we solve the nonlinear system (1.1) using the Newton iteration, for \( i = 1, 2, \ldots, s \), the \( N \)-dimensional iteration matrix is of the form

\[
I_N - h a_{ii} \frac{\partial f}{\partial y}.
\]

If all diagonal element, \( a_{ii} \), of \( A \) are equal, then the Jacobian \( \frac{\partial f}{\partial y} \) and so the iteration matrix (if \( h \) is constant or is only changed moderately), can be kept constant over all stages or even over several steps. In this case, the savings can be reduced further to \( O(N^3) + O(sN^2) \).

Sometimes, you gain something and you lose something else. The DIRK method is an example of this. The computational cost has been cut down for IRK methods by using DIRK methods, but the accuracy and stability are affected because of the simplification of the coefficient matrix \( A \). The fact that the maximum attainable order for an \( A \)-stable s-stage IRK method is \( 2s \), whereas there are only four cases: \( s = 1, 2, 3, 5 \), for an s-stage SDIRK method to have the maximum attainable order \( s + 1 \). This is one of the main reasons why the existing codes based on SDIRK methods cannot compete with the codes based on IRK methods for many stiff problems. Another important disadvantage for DIRK methods is the lack of accuracy of the stage approximations. Unlike multistep methods, the multistage methods have to calculate several stages before they advance the initial values to the numerical solutions for each integration step. For IRK methods, the output values are obtained from the combination of the input values and
1.1. REVIEW OF SINGLY-IMPlicit RUNGE-KUTTA METHODS

the derivatives of the internal stages, the accuracy of the stage values affect the accuracy of the numerical solution. Some investigations have shown that for some stiff problems, the accuracy of the numerical solutions obtained is not the expected accuracy, and is closely related to the accuracy of the internal stage approximations. The very poor accuracy of the stage value for DIRK methods is no doubt a serious handicap. In fact, many existing famous IRK methods also have similar drawbacks. Therefore, the search for high accuracy (for both output value and stage value) and high stability while retaining computational advantages leads to the “Singly-Implicit Runge-Kutta” (SIRK) methods which are our main concern in this thesis.

The accuracy of a numerical solution can be interpreted in terms of the difference between the exact solution and the numerical solution. This is, a numerical solution $y_{n+1}$ is said to be of order $p$ at the integrated point $x_n$ if

$$y(x_n) - y_n = O(h^{p+1}),$$

where $y(x_n)$ is the exact solution. Due to the implementation and accuracy considerations, the design of an $s$-stage SIRK method is based on the following two assumptions:

1. the coefficient matrix $A$ has a one point spectrum property, and is written as $\sigma(A) = \{\lambda\}$;
2. the order of the stage values and the order of the output value each equals $s$.

The reason for designing SIRK methods with the property $\sigma(A) = \{\lambda\}$ is because the $A$ matrix can be transformed to a Jordan canonical form with the same diagonal elements and the same bi-diagonal elements. By using some transformation matrices, we can still have the iteration matrix $I_N - h\lambda \frac{\partial f}{\partial y}$ for each stage iteration. Hence, the overall computational cost is similar to SDIRK methods but also includes the extra transformation costs $O(3s^2N)$. It turns out that A-stable $s$-stage SIRK methods can be obtained for $s = 1, 2, \ldots, 6, 8$ and have highly accurate stage values (same order as the output value) without too much extra
computational cost. The improvement achieved is more apparent for large dimensional systems. Furthermore, for order higher than 8, almost A-stable SIRK methods are available. The existing variable order code STRIDE based on SIRK methods from order 1 up to order 14 has been shown to be efficient for large problems, especially for hyperbolic differential equations.

Although all important factors affecting the efficiency of a method are considered by SIRK methods, there is another difficulty which appears when SIRK methods try to retain high accuracy for stage values. Because the order of the stage approximations is equal to the stage number for SIRK methods, the choices of the abscissae for SIRK methods are limited. The abscissae $c_1, c_2, \ldots, c_s$ of an $s$-stage SIRK method must satisfy

$$c_i = \lambda \xi_i, \quad \xi_i \text{ is the zero of the } s \text{ degree Laguerre polynomial } L_s(x),$$

for all $i = 1, 2, \ldots, s$. It turns out that, for $s > 2$, some of the abscissae are greater than 1. This may cause some difficulties in the numerical behaviour of algorithms based on these methods, especially for nonlinear problems.

To overcome this difficulty, more freedom in choosing the abscissae is necessary. The question raised now is how to obtain more free parameters in choosing the coefficients for SIRK methods while still retaining the existing advantages. This difficulty was partly solved by a combination of the SDIRK and SIRK methods. Using SIRK methods to retain the high stage order and using the SDIRK methods to obtain some free parameters. These methods are the so-called "DESI" (Diagonally Extended Singly Implicit Rung-Kutta) methods. The coefficient matrix $A$ of a DESI method has a singly-implicit block and a diagonally-implicit block. The method still has a single-implicitness property and the order and stage number equal to the stage number. Because of these additional stages, DESI methods have more freedom in choosing their coefficients. In particular, the abscissae of the diagonal stages can be chosen for convenience. Furthermore, because of this extra freedom, the local truncation errors of the DESI methods are lowered. The magnitude of this reduction depends on the number of the additional diagonal stages. It has been shown that the existing variable order DESI code based on the DESI formulae with three additional stages are more efficient than SIRK methods.
and competitive with the BDF methods even for many small problems. Because of high accuracy, good stability and the implementation advantage, DESI methods have potential for PDE and DAE (Differential Algebraic Equation) problems.

Another approach to removing the restriction on the abscissae for SIRK methods is the use of "effective order". The effective order of SIRK methods is defined in terms of the ability of the method to conform to some perturbations of the exact solution, $\psi(y(x))$. The mapping $\psi$ can be a notional one-step method. Thus, if $\phi$ denotes the specified method, for $\phi$ to have effective order $p$ it is necessary that $\|\phi(\psi(y(x_0))) - \psi(y(x_0 + h))\| = O(h^{p+1})$. These perturbations introduce some free parameters, so the limitation on the abscissae is removed. The abscissae for the effective order SIRK (ESIRK) methods can be chosen as we wish as long as they are distinct. It is clear that ESIRK methods need a starting procedure to produce the perturbed initial value. This is usually done by using the associated classical SIRK methods. Because all the stage approximations for an $s$-stage ESIRK method are designed to be of the classical order $s$, an approximation to the exact solution can be obtained at any time. In addition to removing the restriction on abscissae, a remarkable outcome for ESIRK methods is the reduction of the local error when these methods are applied using a special stepsize changing scheme. Therefore, the adoption of effective order is verified to be beneficial.

Adding additional stages or applying effective order to SIRK methods seems to be a successful generalization of the classical SIRK methods. It is interesting to combine all these ideas together and to examine their numerical behaviour. Hence, we arrived at the latest member of the families of SIRK methods, EDESI (Effective order Diagonally Extended Singly Implicit Runge-Kutta) methods. Unlike DESI methods, the distinct abscissae for EDESI methods can be chosen for convenience. The local errors become smaller than the corresponding ESIRK methods. They also benefit from the variable stepsize. Similar to ESIRK methods, some numerical results have shown that EDESI methods are also good candidates for stiff problems.

Although some numerical results show that for some stiff problems, especially for small dimensional problems, some Rosenbrock type methods, some extrapolation
methods or some IRK methods are more efficient than singly-implicit methods, it is believed that good stability and high accuracy support the families of singly-implicit methods as appropriate choices for stiff problems. In particular, the improvement in computer technology will make these methods more attractive.

The diagram for the development of these methods together with the names associated with the method is shown in Figure 1.1. The number in the parentheses following the method is the year the specified method was introduced.

![Diagram of Runge-Kutta methods for stiff problems]

**Figure 1.1: Runge-Kutta methods for stiff problems**

The reference papers related to the singly-implicit methods are given as follows.

1. IRK: Butcher [8].
2. DIRK: Alexander [1], Crouzeix [31], Nørsett [55].
3. SIRK: Burrage [2], Butcher [14], Burrage, Butcher and Chipman [5], [6].
4. DESI: Butcher and Cash [23], Butcher, Cash and Diamantakis [24], Cash and Diamantakis [29], Diamantakis [35].
5. ESIRK: Butcher and Chartier [20], [21], Butcher and Chen [25].
6. EDESI: Butcher and Diamantakis [26].
1.2 The aims and the framework of this thesis

The main aims of this thesis can be summarized as follows.

(1) To investigate the numerical behaviour of the effective order singly-implicit methods using both theoretical studies and practical experiments.

(2) To study the choice of the abscissae for ESIRK methods.

(3) To design some experimental variable stepsize ESIRK codes and examine their efficiency by comparing the numerical results with some existing IRK codes, such as \textit{SDIRK}, \textit{RADAU5}, and the BDF code \textit{LSODE}.

In chapter two, we discuss the stiffness of IVPs and give some examples which we will use later in our experiments. Some details of IRK methods are also reviewed. In chapter three, we consider the singly-implicit methods including SIRK methods and DESI methods. We discuss how to reduce the computational costs for IRK methods by the use of a transformation matrix and give some different approaches to achieve this reduction.

In chapter four, we focus on the application of the effective order for SIRK. Some frameworks for this application is made; this includes the derivation of the effective order conditions, the construction of the ESIRK methods, and the systematic stepsize changing scheme. Because the perturbation solution flow is changed when this special stepsize changing scheme is used in ESIRK methods, we focus on the study of the numerical behaviour of the ESIRK methods with variable stepsize. Based on this investigation, some suitable choices of the abscissae for ESIRK methods are given. Some implementation details including several different ways for local error estimation, and the initial values prediction for Newton iterations are also studied. In the final part, we show some numerical results obtained using these ESIRK codes.

In the last chapter, the construction of the EDESI methods together with some related implementations are discussed. We also show some numerical comparisons obtained by using SIRK, DESI methods together with their corresponding
effective order applications, ESIRK, EDESI methods. Therefore, the influences caused by the use of effective order can be analysed.
Chapter 2

Numerical methods for stiff differential equations

Stiff differential equations are important in numerical analysis because they frequently arise in practical problems such as chemical kinetics, electrical circuits, atmospheric pollution, mathematical biology and computer aided design techniques. Furthermore, they are difficult to solve by traditional numerical methods. For example, explicit methods do not work well for these equations because they do not have infinite stability regions which are a necessity for stiff problems. In this chapter, we first discuss the aspects of the phenomenon of stiffness and give some examples. Because stiff equations require new concepts of stability, we therefore focus on stability theory in section 2. Implicit Runge-Kutta methods (IRK) are ideal for stiff problems because of their good stability and high accuracy. In section 3, we study some well-known existing IRK methods together with some related properties including the simplifying assumptions of the order conditions, the constructions of these methods, the concept of collocation methods, the stability properties and implementation for IRK methods.
CHAPTER 2. NUMERICAL METHODS FOR STIFF DIFFERENTIAL EQUATIONS

2.1 Stiff problems

Suppose we are dealing with the initial value problem

\[ \begin{align*}
  y'(x) &= f(x, y(x)), \quad x \in [x_0, b] \subset \mathbb{R}, \\
  y(x_0) &= y_0,
\end{align*} \]  
(2.1)

where \( f : \mathbb{R} \times \mathbb{R}^N \to \mathbb{R}^N \). It is easy to express (2.1) in autonomous form

\[ \begin{align*}
  z'(x) &= g(z(x)), \quad x \in [x_0, b] \subset \mathbb{R}, \\
  z(x_0) &= z_0,
\end{align*} \]  
(2.2)

where \( g : \mathbb{R}^{N+1} \to \mathbb{R}^{N+1} \) and \( z_0 = [y_0, x_0]^T \) by defining

\[ \begin{align*}
  z_i(x) &= y_i(x), \quad i = 1, \ldots, N, \\
  z_{n+1} &= x.
\end{align*} \]

Therefore, without loss in generality, we will consider only the autonomous initial value problem in this thesis.

To start working on the stiffness of differential equations, it is appropriate to study the linear homogeneous system

\[ y'(x) = Ay(x), \]  
(2.3)

where \( A \) is a constant \( N \times N \) matrix. The special case in which \( A \) is diagonal is remarkably typical if \( A \) has distinct eigenvalues. Suppose \( D = P^{-1}AP \), where \( D = \text{diag}(\lambda_1, \lambda_2, \ldots, \lambda_N) \). If we define \( z = P^{-1}y(x) \), then (2.3) is transformed into an equivalent system

\[ z'(x) = Dz(x). \]  
(2.4)

In other words, one can verify the solution behaviour of (2.3) by studying the solution of (2.4). The solution of (2.4) can be immediately written as

\[ z(x) = [c_1 \exp(\lambda_1(x - x_0)), \ldots, c_N \exp(\lambda_N(x - x_0))]^T, \]
where \( c_i = z_i(x_0), i = 1, \ldots, N \). If there is an eigenvalue of a component with a very negative real part compared with the others, say \( \lambda_m \), then when \( x - x_0 \) increases, this component \( \exp(\lambda_m(x - x_0)) \) decays to zero rapidly. Therefore the contribution from this component to the solution will become increasingly less significant. A linear constant coefficient system whose solution contains such a component is referred to as a "stiff" differential equation. We first give an example to illustrate a stiff linear differential equation.

**Example 2.1** Consider the system

\[
y'(x) = Ay(x), \quad y(0) = [y_{10}, y_{20}]^T, \quad \text{where} \quad A = \begin{bmatrix} -2 & 1 \\ 998 & -999 \end{bmatrix}.
\]

The two eigenvalues are \(-1, -1000\). We can rearrange this system to the equivalent system

\[
z'(x) = Dz(x), \quad \text{where} \quad D = \begin{bmatrix} -1 & 0 \\ 0 & -1000 \end{bmatrix},
\]

by defining

\[
z(x) = P^{-1}y(x), \quad \text{where} \quad P^{-1} = \frac{1}{999} \begin{bmatrix} 998 & 1 \\ 1 & -1 \end{bmatrix}.
\]

Therefore the solution of the original system can be written as

\[
y(x) = P\text{diag}(\exp(-x), \exp(-1000x))P^{-1}y(0).
\]

In this case, stiffness happens because the solution of the differential equation contains the rapidly decaying component \( \exp(-1000x) \). If a numerical method, such as an explicit method which has a finite stability region is used to solve this system, we will find that stability requirements will force \( 1000h \) to be bounded, \( h \) is the stepsize. For example, the Euler method has the bound \((0, 2)\) for \( 1000h \) in order to fulfill the stability requirement (we will discuss this in the following subsection). It is clear that a very small stepsize, \( h < 0.002 \), will be needed.
although the first component does not have this restriction. After a few steps, the magnitude of the second component will be so small compared to the first component which contains the significant information we are interested in.

In the above Example 2.1, we note that the stiffness is independent of the choice of the initial conditions $y(0) = [y_{10}, y_{20}]^T$. We may use different methods or different stepsizes to solve the two components separately from this system. However this may not be possible for many systems. Furthermore, stiffness may happen when a system is one-dimensional. Consider, for example, the famous Prothero-Robinson differential equation,

$$y'(x) = \lambda(y(x) - g(x)) + g'(x), \quad \text{Re}(\lambda) \ll 0, \quad (2.5)$$

where $g(x)$ is a smooth function. Equation (2.5) is easily rearranged as

$$(y(x) - g(x))' = \lambda(y(x) - g(x)), \quad (2.5)$$

with exact solution

$$y(x) = g(x) + C_0 \exp(\lambda x), \quad C_0 = y(x_0) - g(x_0). \quad (2.5)$$

Since Re($\lambda$) $\ll 0$, the second term of this solution will decay to zero rapidly and become insignificant compared with the first term. Obviously, the more negative $\lambda$ is, the more stiff (2.5) is. In fact, (2.5) is used in studying the so called “order reduction” phenomenon by Prothero and Robinson ([58] 1974). They found that when applying some implicit Runge-Kutta methods with infinite stability regions to equation (2.5), the obtained order of the numerical solutions is lower than the expected order. These also lead to the concept of the so-called “stiff accuracy” (this is discussed in the following section). However, the discovery of stiffness using a similar type of equation to (2.5) leads to the use of implicit methods for stiff problems first proposed by Curtiss and Hirschfelder in 1952 (cf. [45]).

For some stiff nonlinear systems, the various eigenvalues can be negatively large at some points in the integration interval. We now give some examples of this phenomenon.
Example 2.2 The three dimensional Robertson chemical kinetics problem ([60] 1966) with real eigenvalues is a mildly stiff problem and is very often used for testing methods. This problem describes the concentrations of three components in a chemical reaction. On physical grounds, the three components have non-negative values and the system will eventually tend to some equilibrium state.

Robertson chemical kinetics

\[
\begin{align*}
y_1' &= -0.04y_1 + 10^4 y_2 y_3, & y_1(0) &= 1 \\
y_2' &= 0.04y_1 - 10^4 y_2 y_3 - 3 \times 10^7 y_2^2, & y_2(0) &= 0 \\
y_3' &= 3 \times 10^7 y_2^2, & y_3(0) &= 0
\end{align*}
\] (2.6)

Since \( \sum_{i=1}^{3} y_i' = 0 \), the sum of these three components is one during all integration. The Jacobian is found to be

\[
J(y) = \begin{pmatrix}
-0.04 & 10^4 y_3 & 10^4 y_2 \\
0.04 & -10^4 y_3 - 6 \times 10^7 y_2 & -10^4 y_2 \\
0 & 6 \times 10^7 y_2 & 0
\end{pmatrix}
\]
At $x = 0$, the eigenvalues are $-0.04, 0, 0$ and stiffness does not occur. In the neighbourhood of the equilibrium $y_1 = 1, y_2 = 3.65 \times 10^{-5}, y_3 = 0$, the eigenvalues become $0, -0.0405, -2189.6$, and this leads to the stiffness situation. If we use the Euler method in such circumstances, the stepsize is forced to be bounded $u_{\text{max}}$. In fact, the concentrations will tend to the equilibrium values $y_1(x) = 0, y_2(x) = 0, y_3(x) = 1$, when $x \to \infty$. The solution flows of these three components are shown in Figure 2.1. Although physically no solution component can be zero, it is difficult to avoid the integration method producing negative solution values for this problem. As a matter of fact, the second component becoming negative has in practice been extremely hard to deal with if the integration interval is large enough, say $(0, x_{\text{end}}), x_{\text{end}} = 10^6, 10^{11}, 10^{12}, \ldots$. □

**Example 2.3** Another remarkable nonlinear stiff problem is the Van der Pol oscillator (cf. [43], [45]) with complex eigenvalues.

Van der Pol oscillator

\[
\begin{align*}
  y_1' &= y_2, & y_1(0) &= 2 \\
  y_2' &= \theta(1 - y_1^2)y_2 - y_1, & y_2(0) &= 0
\end{align*}
\]  

(2.7)

This problem has a periodic solution with very different properties along the trajectory. The parameter $\theta$ varies the properties of the problem. Increasing $\theta$ is accompanied by increasing stiffness. It is thus called the stiffness parameter. For $\theta = 10^6$, the first and second components of the solution with integration interval $(0, 2)$ are shown in Figure 2.2 and Figure 2.3 respectively. The oscillations of the first component appear periodically. Because $\theta$ is very large, the derivative of $y_2'$ with respect to $y_2$ is very large negative in the neighbourhood of $y_2' = 0$, for $|y_1| > 1$. Therefore, the solution will rapidly approach the neighbourhood of $y_2' = 0$. Hence $y_2 = \frac{y_1}{\theta(1 - y_1^2)}$. This implies $y_2 \approx 0$ when $y_1 > 1$ (refer to Figure 2.3). Furthermore, $y_1'$ depends on $y_2$. Hence, $y_1'$ (so as $y_2'$) tends to infinity when $y_1 \approx \pm 1$ (refer to Figure 2.2). These also cause the stepsize chosen to be very small in order to satisfy the error test. Therefore, the main problem when a code is trying to solve this problem numerically is that the difficulty is not only from
2.2. STABILITY REQUIREMENTS

Figure 2.2: The first component of the Van der Pol problem (2.7) with $\theta = 10^6$

Figure 2.3: The second component of (2.7) with $\theta = 10^6$

the stiffness but also the change which comes from the behaviour of the solution flow.

Following Gear [41], we note that the stiffness phenomenon may happen in some particular initial value problems with some particular initial values and the error tolerance. This kind of system is called pseudo-stiff by Lambert [50]. The methodology for solving stiff problems is to use methods with good stability properties, this will be discussed in the following subsection.

2.2 Stability requirements

Suppose the exact solution of the ordinary differential equation

$$y'(x) = f(y(x)), \quad y(x_i) = y_i, \quad i = 0, 1, 2, \ldots, k - 1.$$  \hspace{1cm} (2.8)

exists and is unique and the method used has the general form

$$\sum_{i=0}^{k} \alpha_i y_{n+i} = h \sum_{i=0}^{k} \beta_i f(y_{n+i}), \quad \alpha_k = 1, \quad |\alpha_0| + |\beta_0| > 0.$$  \hspace{1cm} (2.9)

We note that when $\beta_k = 0$, the method is explicit, otherwise it is implicit. The minimum requirement for (2.9) to solve (2.8) is that the numerical solutions
produced by this method must converge to the expected exact solutions as the stepsize \( h \) tends to zero. For any numerical method, there are two conditions which have to hold in order to meet the "convergence" requirement. One of these two conditions is called "consistency" which simply means that when we use a method to solve an ordinary differential equation, if the initial value for any point \( x_i \) used is the exact solution, then the difference (local error) of the exact solution at \( x_i + h \) and the numerical solution tends to zero as \( h \) tends to zero. This guarantees the local accuracy of the numerical solution. However, it is not enough to ensure that the numerical approximation will be able to converge to the exact solution after many integration steps because the truncation error may accumulate to be very large and overflow. Therefore another condition to ensure the numerical solution is bounded is necessary. Thus the method must satisfy the so-called "stability" (zero-stability) requirement. It is easy to specify consistency and stability by the use of generating polynomials \( \rho, \sigma \) which are defined as follows:

\[
\rho(\omega) = \sum_{i=0}^{k} \alpha_i \omega^i, \quad \sigma(\omega) = \sum_{i=0}^{k} \beta_i \omega^i.
\]

The concept of "order" of a numerical method is important and is defined in the following manner.

**Definition 2.1** A numerical method is of order \( p \) if it satisfies

\[
y(x_n) - y_n = O(h^{p+1}), \quad n = 1, \ldots
\]  

(2.10)

where \( y(x_n) \) is the exact solution of an IVP at \( x_n \) and \( y_n \) is the \( n \)-th step numerical solution obtained using the exact starting values, that is, if the local truncation error behaves like \( O(h^{p+1}) \) at the \( n \)-th step.

The consistency of a method means that its order is at least 1 in order to satisfy the local accuracy requirement. While assuming constant stepsize \( h \), we can use the Taylor series to expand the exact solution \( y(x_{n+k}) \) about \( x_n \) and from (2.8), (2.9), we can also expand the numerical solution \( y_{n+k} \). If we ask that the order must be bigger than 0, then by (2.10), the coefficients of \( h \) in these two Taylor
2.2. STABILITY REQUIREMENTS

series have to agree up to the first power of $h$. It is found that

$$\sum_{i=0}^{k} \alpha_i = 0, \quad (h^0 \text{ term})$$
$$\sum_{i=0}^{k} i\alpha_i = \sum_{i=0}^{k} \beta_i, \quad (h^1 \text{ term})$$

By using generating polynomials $\rho, \sigma$, this can be rewritten as

$$\rho(1) = 0, \quad \rho'(1) = \sigma(1). \quad (2.11)$$

(2.11) is an easy way to examine if a method is consistent or not. For stability, a method must satisfy the so-called "root condition".

**Definition 2.2** The generating polynomial $\rho(r)$ does not have roots whose Moduli are bigger than 1 and whose roots are simple when their Moduli is 1.

We now turn our attention to the property of stability. Explicit methods are not desirable for solving stiff problems because the stepsize used is much smaller than what accuracy requires. Implicit methods have a common property which guarantees the convergence of the numerical solutions in some sense when stiffness appears. We are going to take the Euler and implicit Euler methods as examples. As we have seen in the last section, it is possible to transform (2.3) to an equivalent system (2.4). That is, for each distinct eigenvalue $\lambda$, we will have the simple form

$$y'(x) = \lambda y(x). \quad (2.12)$$

Hence, we would like to ask how we can have an acceptable behaviour for (2.12). Firstly, we apply the Euler method

$$y_n = y_{n-1} + hf(y_{n-1}) \quad (2.13)$$

to (2.12) with constant stepsize $h$. We then have the approximate solution at $x_n = x_0 + nh$,

$$y_n = (1 + h\lambda)y_{n-1}, \quad (2.14)$$
CHAPTER 2. NUMERICAL METHODS FOR STIFF DIFFERENTIAL EQUATIONS

and this leads to the numerical solution

\[ y_n = (1 + h\lambda)^n y_0, \]

if \( y_0 \) is the initial value. For convenience, we write \( z = h\lambda \). It is clear that \( y_0 \exp(nh\lambda) = y_0 \exp(nz) \) is the exact solution at \( x_n \) for (2.12). The question now is how we can have a bounded behaviour for \((1 + z)^n\), as \( n \to \infty \) when \( \exp(nz) \) is bounded. This will be equivalent to requiring \(|1 + z| \leq 1\) if \( \text{Re}(z) \leq 0\). In other words, if one applies the Euler method to some specified problem with some eigenvalue \( \lambda_t \), and if the step-size \( h \) used satisfies \( z = h\lambda_t \in S = \{ z \in \mathbb{C}; |1 + z| < 1 \}, \forall t \), then the numerical solution may be guaranteed not to be divergent. \( S \) here is called the "stability region" of the Euler method. For stiff problems with some very large negative eigenvalues, it is natural to require the stability region to be as large as possible. Unfortunately, the stability region of the Euler method is just the disc with centre at \(-1\) and radius \( 1 \).

Let us now look at the implicit Euler method,

\[ y_n = y_{n-1} + hf(y_n). \]  

(2.15)

When we apply it to (2.12), this yields

\[ y_n = y_{n-1} + zy_n. \]

Hence, \( y_n = (1 - z)^{-1} y_{n-1} \). To obtain a bounded solution sequence, \(|1 - z| \geq 1\) must be satisfied. It is easy to see that the stability region \( S \) here is much larger than that for the Euler method. It is the complement in the complex plane of the interior of the disc with centre \( 1 \) and radius \( 1 \). Because the stability region includes the whole left half-plane, there is a bounded approximation no matter how large the negative eigenvalue. This also means that we do not need to force the step-size to be too small for convergence. Therefore, the basic requirement for a numerical method in solving stiff problems without high computing costs is widely referred to in the following famous property which was introduced by Dahlquist in 1963 [33].

**Definition 2.3** (Dahlquist 1963). A method whose stability region contains the whole of the left half-plane is called **A-stable**.
The ratio $y_n/y_{n-1}$ of these two one-step methods (2.13), (2.15) is known as the "stability function" and is noted as $R(z)$. It is easy to see that the stability functions $R(z)$ for the Euler method and the implicit Euler method are $1 + z$ and $(1 - z)^{-1}$ respectively. Because in examining the accuracy and stability of Runge–Kutta methods, the stability function $R(z)$ plays an important role, we will discuss it in more detail in section 2.3.

A-stable methods seem to be ideal for stiff problems because their stability region includes the left half-plane. For very stiff problems, where the eigenvalue has a very large negative real part, the exact solution of the corresponding stiff component is damped out very fast. In this case, the methods used are supposed to have a stability function that satisfies

$$R(z) \to 0, \quad \text{when } |z| = |h\lambda| \to \infty.$$  

When some A-stable methods are applied to such a system, the numerical solutions decay slowly even when $|h\lambda|$ is very large, for example, the trapezoidal rule

\[
\begin{array}{ccc}
0 & 0 & 0 \\
1 & \frac{1}{2} & \frac{1}{2} \\
\frac{1}{2} & \frac{1}{2} \\
\end{array}
\]

which has a stability function equal to $(1 + \frac{z}{2})/(1 - \frac{z}{2})$. Hence a stronger stability property named "L-stability" was introduced by Ehle [36] in 1969.

**Definition 2.4** A method is said to be L-stable if it is A-stable and the stability function satisfies

$$\lim_{|z| \to \infty} R(z) = 0.$$  

We need to notice that the condition of L-stability requires the method be A-stable and the rapidly decaying solutions to be correctly damped. But this still causes the rapidly increasing solutions to be damped when we try to solve a system with large positive eigenvalues ([50], [22]). It is clear that the implicit Euler method is L-stable. For Runge-Kutta methods, the stability functions of
CHAPTER 2. NUMERICAL METHODS FOR STIFF DIFFERENTIAL EQUATIONS

the explicit methods are polynomials (discussed in section 2.3), and hence we cannot have an infinite stability region. A-stability (therefore L-stability) is not possible. In the case of implicit methods, the stability functions are rational functions. In order to have L-stability for these methods, the degree of the numerator must be less than the degree of the denominator. For linear multistep methods, in 1963, Dahlquist presented another famous theorem which had a great influence on the numerical analysis of ODEs. This result is well known as the second Dahlquist barrier.

**Theorem 2.5** *(Dahlquist 1963)* No explicit linear multistep method can be A-stable. Furthermore, an A-stable linear multistep method must be of order less than or equal to 2.

This theorem limits the use of higher order multistep methods for stiff problems. Due to the computation advantage of linear multistep methods, some alternative stability theorems were proposed by several authors. For many practical problems whose eigenvalues are not very close to the imaginary axis, a method having a stability region which has part of the left half-plane and includes the negative real axis is enough to solve those problems. This leads to the following weaker stability property introduced by Widlund ([64] 1967).

**Definition 2.6** A method is called $A(\alpha)$-stable if its stability region includes

$$\{ z \in \mathbb{C}; |\arg(-z)| \leq \alpha \}.$$ 

Another similar weakening of A-stability is discussed by Gear ([40] 1969) and is called “stiff stability”.

**Definition 2.7** A method is stiffly stable if its stability region includes $\{ z \in \mathbb{C}; \text{Re}(z) < -D \}$ and $\{ z \in \mathbb{C}; -D \leq \text{Re}(z) < 0, -c \leq \text{Im}(z) \leq c \}$, where $D, c$ are positive real numbers.

As mentioned in the last section, in 1952, Curtiss and Hirschfelder introduced a type of method called “backward differentiation formulae” (or BDF) whose
stability properties are promising for stiff problems amongst the class of linear multistep methods. The BDF formula is given as follows:

\[
\sum_{i=1}^{k} \frac{1}{k} \nabla^i y_{n+1} = hf_{n+1},
\]

(2.16)

where \( \nabla^i \) is the backward difference operator defined by \( \nabla^0 y_n = y_n, \quad \nabla^{i+1} y_n = \nabla^i y_n - \nabla^i y_{n-1}. \)

We note that (2.16) is the implicit Euler method when \( k = 1 \). BDF methods of order up to 6 have been shown to be stiffly stable for some \( D \) and \( c \) values [39]. Below is the table of \( A(\alpha) \)-stable and stiffly-stable BDF methods with the values \( \alpha \) and \( D \) [45].

<table>
<thead>
<tr>
<th>( k )</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \alpha )</td>
<td>90°</td>
<td>90°</td>
<td>86.03°</td>
<td>73.35°</td>
<td>51.84°</td>
<td>17.84°</td>
</tr>
<tr>
<td>( D )</td>
<td>0</td>
<td>0</td>
<td>0.083</td>
<td>0.667</td>
<td>2.327</td>
<td>6.075</td>
</tr>
</tbody>
</table>

Because of the fast convergence and low computational cost of the BDF method (only one function evaluation required per iteration), many present codes based on BDF formula give quite satisfactory numerical performances for stiff problems. However, the lack of stability and lower accuracy (large error constants) are drawbacks for these methods. It is natural to pay attention to the implicit Runge-Kutta methods which have good stability and high accuracy. It is hoped that for most (very) stiff problems, the numerical results obtained by IRK methods are more satisfactory because of their excellent stabilities.

### 2.3 Implicit Runge–Kutta methods

Amongst one step methods, explicit Runge-Kutta methods are natural candidates for practical implementation because of their low computational cost. However, their finite stability regions makes them unsuitable for stiff problems, and for high orders of accuracy, they need many more stages than implicit methods of the
same order. Furthermore, implicit methods are closely related to the very efficient Rosenbrock methods. Undoubtedly IRK methods are a valuable preliminary to working with Rosenbrock methods.

In this section, we will devote ourselves to implicit Runge-Kutta methods which are suitable for solving stiff problems because of their good stability properties. We note that A-stable IRK methods of this type exist for all orders.

As usual, we will write the standard initial value problem in the autonomous form

\[ y'(x) = f(y(x)), \quad y(x_0) = y_0. \]  

(2.17)

We consider an \( s \)-stage Runge-Kutta method in the form

\[ Y_i = y_0 + h \sum_{j=1}^{s} a_{ij} f(Y_j), \quad i = 1, \ldots, s, \]

\[ y_1 = y_0 + h \sum_{i=1}^{s} b_i f(Y_i), \]

or in a more compact notation (Let \( \otimes \) is the Kronecker product),

\[ Y = e \otimes y_0 + h(A \otimes I_N) F, \]

(2.18)

\[ y_1 = y_0 + h(b^T \otimes I_N) F, \]

(2.19)

where \( e = [1, 1, \ldots, 1]^T \), \( Y = [Y_1, Y_2, \ldots, Y_s]^T \) and \( F = [f(Y_1), f(Y_2), \ldots, f(Y_s)]^T \),

\[ A e = c, \quad b^T e = 1. \]

If \( A \) satisfies \( a_{ij} = 0, \quad i \leq j \), the method is explicit, otherwise we call the method implicit (semi-implicit when \( a_{ij} = 0, \quad i < j \) and at least one \( a_{ij} \neq 0 \)).

Before we go through the family of implicit methods, we shall discuss the so-called “simplifying assumptions” \( C(q), B(p), D(r) \) introduced by Butcher in 1963 [7]. \( C(q) \) is called the stage order condition. If an \( s \)-stage method satisfies \( C(q) \), then it satisfies

\[ \sum_{j=1}^{s} a_{ij} c_j^{k-1} = \frac{1}{k} c_i^k, \quad i = 1, 2, \ldots, s; \quad k = 1, 2, \ldots, q, \]

(2.20)
2.3. IMPLICIT RUNGE-KUTTA METHODS

or in a compact form

\[ A c^{k-1} = \frac{c^k}{k}, \quad k = 1, 2, \ldots, q. \]  

(2.21)

The "stage order" of the method refers to the highest \( q \) such that (2.20) holds when \( k \leq q \). By comparing both the Taylor expansions of (1.1) and \( y(x_0 + c_i h) \), we have

\[ \dot{Y}_i = y(x_0 + c_i h) + O(h^{q+1}), \quad i = 1, \ldots, s, \]

which means every approximation of the internal stages can have at most order \( s \). In a similar way, the output approximation \( y_1 \) will satisfy

\[ y_1 = y(x_0 + h) + O(h^{p+1}), \]

if the method also satisfies \( B(p) \),

\[ b^T c^{k-1} = \sum_{i=1}^{s} b_i c_i^{k-1} = \frac{1}{k}, \quad k = 1, 2, \ldots, p. \]  

(2.22)

Satisfying \( D(r) \) means the method satisfies

\[ \sum_{j=1}^{s} b_j c_j^{k-1} a_{ij} = \frac{b_j}{k} (1 - c_j^k); \quad i = 1, 2, \ldots, s, \quad k = 1, 2, \ldots, r. \]

or

\[ b^T c^{k-1} A = \frac{1}{k} (b^T - b^T c^k), \quad c = \text{diag}(c_1, c_2, \ldots, c_s). \]

If the \( C(q) \) and \( D(q) \) conditions hold for high enough values of \( q \), the number of order conditions is greatly reduced.

The families of these methods we are discussing are all based on quadrature rules. The well known \( s \)-stage "Gauss method" is of order \( 2s \) and its abscissae are the zeros of the shifted Legendre polynomial of degree \( s \) on the interval \([0, 1]\), that is, zeros of \( \frac{d^s}{dx^s}(x^s(x - 1)^s) \). Besides, the \( s \)-stage Gauss method satisfies \( B(2s), C(s), D(s) \). The methods can be determined easily using these simplifying assumptions. For better stability properties, we seek methods which have order slightly lower than \( 2s \). The Radau type methods have order \( 2s - 1 \) while the Lobatto type methods have order \( 2s - 2 \) [15]. The choice of abscissae is given in the following theorem [9].
CHAPTER 2. NUMERICAL METHODS FOR STIFF DIFFERENTIAL EQUATIONS

Theorem 2.8 (1) For the Radau I case, \( c_1, c_2, \ldots, c_s \) are the distinct roots of 
\[
(d/dx)^s - x^s (x-1)^{s-1} = 0, \text{ with } c_1 = 0 \text{ and } c_s < 1.
\]

(2) For the Radau II case, \( c_1, c_2, \ldots, c_s \) are the distinct roots of 
\[
(d/dx)^{s-1} x^{s-1} (x-1)^s = 0, \text{ with } c_1 > 0 \text{ and } c_s = 1.
\]

(3) For the Lobatto III case, \( c_1, c_2, \ldots, c_s \) are the distinct roots of 
\[
(d/dx)^{s-2} x^{s-1} (x-1)^{s-1} = 0, \text{ with } c_1 = 0 \text{ and } c_s = 1.
\]

We note that the 1-stage Gauss method, 1-stage Radau IIA method and 2-stage Lobatto IIIA method are the implicit Mid point Rule, the Implicit Euler and the Trapezoidal Rule respectively.

In the study of stiffness using the problem (2.5), Prothero and Robinson show that the global error depends not only on the local error but also on the error contributed by the computations of the stage values. If the method has stage order less than the order of the method, the global error will be dominated by the error produced by the stage values. Therefore the accuracy of the solutions obtained appear to be unrelated to the expected order. This phenomenon is the so-called "order reduction". One way to reduce this damage is to use "stiffly accurate" methods. A method satisfying \( a_{sj} = b_j, \forall j \), is called stiffly accurate. Because the error produced by the calculations of the internal stages vanish as \(|z| = |h\lambda| \to \infty\), the stiffly accurate methods will be able to give an asymptotic result as \( z \) tends to infinity. In fact, an A-stable Implicit Runge-Kutta method with nonsingular \( A \) satisfying the stiffly accurate property is L-stable. For completeness, we list the related simplifying assumptions, order, stiff accuracy for \( s \)-stage Gauss, Radau, Lobatto type methods in Table 2.1.

As we can see the Gauss and Lobatto IIIA methods (for the same number of stage evaluations) have a higher order of accuracy than the others, but the Gauss method is not L-stable nor is it stiffly accurate. Hence, for very stiff problems, the local truncation errors of the Gauss method are not damped out quickly when \(|h\lambda|\) becomes large, and for some special type of problem, such as (2.5), the
2.3. IMPLICIT RUNGE-KUTTA METHODS

Table 2.1: Summary of the stability, stiff accuracy and orders of some A-stable one-step methods

<table>
<thead>
<tr>
<th>method</th>
<th>simplifying assumptions</th>
<th>stiffly accurate</th>
<th>order</th>
<th>stage order</th>
<th>L-stable</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gauss</td>
<td>$B(2s), C(s), D(s)$</td>
<td></td>
<td>$2s$</td>
<td>$s$</td>
<td></td>
</tr>
<tr>
<td>Radau IA</td>
<td>$B(2s - 1), C(s - 1), D(s)$</td>
<td>$2s - 1$</td>
<td>$s - 1$</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>Radau IIA</td>
<td>$B(2s - 1), C(s), D(s - 1)$</td>
<td>✓</td>
<td>$2s - 1$</td>
<td>$s$</td>
<td>✓</td>
</tr>
<tr>
<td>Lobatto IIIA</td>
<td>$B(2s - 2), C(s), D(s - 2)$</td>
<td>✓</td>
<td>$2s - 2$</td>
<td>$s$</td>
<td></td>
</tr>
<tr>
<td>Lobatto IIIB</td>
<td>$B(2s - 2), C(s - 2), D(s)$</td>
<td></td>
<td>$2s - 2$</td>
<td>$s - 2$</td>
<td></td>
</tr>
<tr>
<td>Lobatto IIIIC</td>
<td>$B(2s - 2), C(s - 1), D(s - 1)$</td>
<td>✓</td>
<td>$2s - 2$</td>
<td>$s - 1$</td>
<td>✓</td>
</tr>
</tbody>
</table>

method suffers order reduction more than the Lobatto IIIA method. Although Lobatto IIIA methods are significantly more accurate than Radau II methods, it was pointed out by Prothero and Robinson that for equation (2.5), they are only stable for a restricted range of stepsizes because they are not S-stable or strongly S-stable (a special stability property for equation (2.5) as defined in [58]). As a consequence, for both stability and accuracy considerations, the Radau IIA methods are recommended. As a matter of fact, the present famous implicit Runge-Kutta code RADAU5 is based on the 3-stage, fifth order Radau IIA formulae which has the corresponding Butcher table,

\[
\begin{array}{c|cccc}
4-\sqrt{6} & 88-7\sqrt{6} & 296-169\sqrt{6} & -2+3\sqrt{6} \\
10 & 360 & 1800 & 225 \\
4+\sqrt{6} & 88+7\sqrt{6} & -2-3\sqrt{6} \\
10 & 296+169\sqrt{6} & 360 & 225 \\
1 & 16-\sqrt{6} & 16+\sqrt{6} & 1 \\
36 & 36 & 9 \\
16-\sqrt{6} & 16+\sqrt{6} & 1 \\
36 & 36 & 9
\end{array}
\]  

(2.23)

Another concept related to implicit Runge-Kutta methods is called “collocation”. For initial value problems $y'(x) = f(y(x)), \ y(x_0) = y_0$, if the numerical solution of a method is given by

\[ y_1 = p(x_0 + h), \]
CHAPTER 2. NUMERICAL METHODS FOR STIFF DIFFERENTIAL EQUATIONS

where \( p(x) \) is defined by

\[
\begin{align*}
\begin{cases}
p(x_0) &= y_0, & \text{initial condition} \\
p'(x_0 + c_i h) &= f(p(x_0 + c_i h)), & i = 1, 2, \ldots, s,
\end{cases}
\end{align*}
\]

then the method is called a collocation method. The relation between collocation methods and implicit Runge-Kutta methods was pointed out by K. Wright ([66] 1970). If the coefficients of an \( s \)-stage implicit Runge-Kutta method, \( a_{ij}, b_j \), are chosen to be

\[
a_{ij} = \int_0^{c_i} l_j(t) dt, \quad b_j = \int_0^1 l_j(t) dt, \quad i, j = 1, 2, \ldots, s, \quad (2.24)
\]

where \( l_j(t) \) are the Lagrange polynomials,

\[
l_j(t) = \prod_{i \neq j}^s \frac{t - c_i}{c_j - c_i},
\]

then the method is equivalent to a collocation method. Because the coefficient matrix is determined uniquely if the method satisfies \( C(s) \), we note that the choices \( a_{ij}, b_j \) in (2.24) satisfy \( C(s), B(s) \) respectively. In other words, an \( s \)-stage implicit Runge-Kutta method which satisfies \( C(s) \) and \( B(s) \) is a collocation method. This can be also verified as follows.

Suppose \( p(x) \) is a polynomial of degree \( s \), and the \( c_1, c_2, \ldots, c_s \) are distinct. If \( p(x) \) satisfies the initial condition

\[ p(x_0) = y(x_0), \]

then the output approximation \( y_1 \) can be found using

\[ y_1 = p(x_0 + h), \]

if \( c_1, c_2, \ldots, c_s \) satisfy the \( s \) equations

\[
p'(x_0 + c_j h) = \frac{1}{h} \frac{\partial}{\partial c} p(x_0 + ch)|_{c=c_j} = f(p(x_0 + c_j h)), \quad j = 1, 2, \ldots, s. \quad (2.25)
\]

If we write \( p(x_0 + c h) \) as a polynomial in \( c \) as follows

\[
p(x_0 + ch) = y(x_0) + h(v_1 c + v_2 c^2 + \cdots + v_s c^s), \quad (2.26)
\]

28
2.3. IMPLICIT RUNGE-KUTTA METHODS

and

\[ Y_i = p(x_0 + c_i h), \quad i = 1, 2, \ldots, s, \]

then by (2.25), we have

\[ \begin{align*}
    v_1 + 2v_2 c_j + \cdots + sv_j c_j^{s-1} &= hf(Y_j), \\
    j &= 1, \ldots, s
\end{align*} \]

(2.27)

If we multiply both sides of (2.27) by \( a_{ij} \) and then summed from \( j = 1 \) to \( s \) and use the \( C(s) \) condition (2.20) and (2.26), we then have

\[ \begin{align*}
    v_1 c_i + v_2 c_i^2 + \cdots + v_s c_i^s &= \sum_{j=1}^{s} a_{ij} hf(Y_j),
\end{align*} \]

which implies that

\[ Y_i = p(x_0 + c_i h) = y(x_0) + h \sum_{j=1}^{s} a_{ij} f(Y_j). \]

In the same way, if we multiply both sides of (2.27) by \( b_j \) and then summed from \( j = 1 \) to \( s \) and use \( B(s) \), we then have

\[ y_1 = y(x_0) + h \sum_{j=1}^{s} b_j f(Y_j). \]

It turns out that of the methods shown in Table 2.1, the Gauss, Radau IIA and Lobatto IIIA methods are collocation methods, while the Radau IA, Lobatto IIIB and Lobatto IIIC are not.

It is clear that order reduction does not occur when the stage order is equal to the order of the method. Unfortunately, none of the methods shown in Table 2.1 satisfies this condition. Furthermore, in the study of the error behaviour when applying the numerical methods to the nonlinear stiff problems, a theory derived by Frank, Schneid and Überhuber ([38] 1981) shows that for some nonlinear problems (classified by satisfying the one-sided Lipschitz condition) with some IRK methods (classified by satisfying the so-called \( B \)-stability), the order of accuracy obtained is independent of the stiffness of the problem. In [22], two types of order are identified. The order of \( B \)-consistency measures the behaviour of the local
error and the order of $B$-convergence measures the behaviour of the global error. It turns out that both the orders of $B$-consistency and $B$-convergence are closely related to the stage order of the method ([38], [15], [45]). As a consequence, the collocation methods should be preferred, especially methods which have the quadrature order equal to the stage order in this sense.

In an attempt to discuss the stability of Runge-Kutta methods, we again consider the standard test equation

$$y'(x) = \lambda y(x), \quad y(x_0) = y_0 \quad \lambda \in \mathbb{C}, \quad \text{Re}(\lambda) < 0. \quad (2.28)$$

It is easy to see that if $y(x_0) = y(0) = 1$, the exact solution of (2.28) is $y(x) = \exp(\lambda x)$ which decays to zero. If we apply the general Runge–Kutta method to (2.28), we will obtain a one-step difference equation of the form

$$y_{n+1} = R(z)y_n, \quad z = h\lambda. \quad (2.29)$$

$R(z)$ here is called the stability function of the specified method. It can be shown that the sequence of numerical solutions $\{y_n\}$ can converge to zero if $z$ is in the set $\{z \in \mathbb{C} : |R(z)| < 1\}$, called the stability region of the method.

For a general Runge–Kutta method, the stability function can be derived as follows. If (1.1) is applied to (2.28), and we assume the system dimension is one, the Runge-Kutta formulae can be written as

$$Y = y_0e + zAY, \quad (2.30)$$
$$y_1 = y_0 + zb^T Y, \quad (2.31)$$

where $Y = [Y_1, Y_2, \ldots, Y_s]^T$ and $z = h\lambda$. Solving for $Y$ in (2.30) and substituting into (2.31) gives

$$y_1 = (1 + zb^T(I - zA)^{-1}e) y_0. \quad (2.32)$$

According to (2.29), for any dimensional system, it can be concluded that the following result holds.

**Lemma 2.9** The stability function for an $s$-stage Runge-Kutta method (1.1), (1.2) is

$$R(z) = 1 + zb^T(I - zA)^{-1}e. \quad (2.32)$$
Furthermore, (2.30), (2.31) can be rewritten as
\[
\begin{pmatrix}
I - zA & 0 \\
-zb^T & 1
\end{pmatrix}
\begin{pmatrix}
Y \\
y_1
\end{pmatrix} =
\begin{pmatrix}
y_0e \\
y_0
\end{pmatrix}.
\]
We can apply Cramer's rule to find \( y_1 \) in terms of \( y_0 \). The denominator and the numerator of \( R(z) \) can each be derived. This leads to an alternative form for \( R(z) \).

**Lemma 2.10** The stability function of an \( s \)-stage Runge-Kutta method (1.1), (1.2) satisfies
\[
R(z) = \frac{\det(I - zA + zeb^T)}{\det(I - zA)}.
\] (2.33)

If \( A \) is a strictly lower triangular matrix, then \( \det(I - zA) = 1 \). This means that the stability function \( R(z) \) of an explicit Runge-Kutta method will be a polynomial. Because the method is consistent, \( R(z) \) is not a constant. Hence, \( R(z) \) cannot be bounded by 1 when \( |z| \to \infty \).

**Lemma 2.11** There is no \( A \)-stable explicit Runge-Kutta method.

In the study of stability and accuracy, the stability function \( R(z) \) plays an important role. For example, from (2.28), the exact solution can be written as \( y(x_0 + h) = y(x_0) \exp(z) \) and the output approximation can be written as \( y_1 = y(x_0)R(z) \). These lead to the following result.

**Lemma 2.12** If a method is of order \( p \), then \( R(z) \) satisfies
\[
\exp(z) - R(z) = Cz^{p+1} + O(z^{p+2}), \quad |z| \to \infty,
\] (2.34)
where \( C \) is the error constant of the method.

Since \( R(z) \) is easy to obtain, (2.34) is very useful for finding the error constant. Furthermore, the fact that the stability region \( \{ z \in \mathbb{C} : |R(z)| \leq 1 \} \) includes
the whole left half plane implies that $|R(iy)| \leq 1$, \( \forall y \in \mathbb{R} \) and \( R(z) \) is analytic when \( \text{Re}(z) < 0 \). Using \( R(z) = \frac{P(z)}{Q(z)} \), it follows that

\[
|R(iy)| \leq 1 \iff \frac{|P(iy)|}{|Q(iy)|} \leq 1,
\]

\[
\iff |Q(iy)|^2 - |P(iy)|^2 \geq 0,
\]

\[
\iff Q(iy)Q(-iy) - P(iy)P(-iy) \geq 0.
\]

Here \( E(y) = Q(iy)Q(-iy) - P(iy)P(-iy) \), the so-called "\( E \)-polynomial", is an even function in \( y \) with real coefficients. We note that the \( E \)-polynomial for Gauss methods is zero because \( P(z) = \exp(z)Q(z) + O(z^{2s+1}) \) for small \( |z| \). In this thesis, we mainly consider the stability function \( R(z) \) of the IRK methods without poles in the region of the complex plane \( \text{Re}(z) < 0 \). Consequently, by the maximum principle, the property that \( R(z) \) is analytic for \( \text{Re}(z) < 0 \) and \( E(y) \geq 0, \ \forall y \in \mathbb{R} \), is not only a sufficient but also a necessary condition to ensure the method is A-stable.

Another remarkable technique for the study of linear stability is provided by the theory of "order stars" introduced by G. Wanner, E. Hairer and S.P. Nørsett ([63], 1978). The idea of order stars is to study not the original stability function \( R(z) \) but instead \( R(z)\exp(-z) \). In other words, the original stability region is replaced by

\[
S = \{ z \in \mathbb{C} : |R(z)| > |\exp(z)| \}.
\]

If \( I \) denotes the set of purely imaginary numbers, then the following result gives a criterion based on order stars.

**Lemma 2.13** A method is A-stable if and only if the stability function \( R(z) \) has no poles in the left half-plane and \( S \cap I = \emptyset \).

The advantage in this appearance is that the behaviour when \( |z| \) is large is determined by the exponential factor. The behaviour at the origin, at \( z = \pm \infty \) and at the poles and zeros largely determines the order star. Many previously intractable problems have now been solved using this theory.
2.3. IMPLICIT RUNGE-KUTTA METHODS

Linear stability plays a central role in the choice of numerical methods suitable for stiff problems. However, some unstable behaviour can still arise when applying some A-stable methods to certain nonlinear stiff problems. It was proposed by Dahlquist to consider a nonlinear test problem to study the stability for linear multistep methods and this leads to the so-called concept of “G-stability”. For Runge-Kutta methods, Butcher introduced a similar stronger stability property known as “B-stability” ([12] 1975) for nonlinear problems.

**Definition 2.14** Let \{...y_{n-1}, y_n,...\} and \{...z_{n-1}, z_n,...\} be two sequences of approximate solutions of \(y'(x) = f(y(x))\) computed using an implicit Runge-Kutta method with a fixed stepsize. Let \((\cdot, \cdot)\) denote an inner product on \(\mathbb{R}^N\) and let \(||\cdot||\) be the corresponding norm. The Runge-Kutta method is said to be B-stable if for any \(f\) satisfying

\[
(f(u) - f(v), u - v) \leq 0 \quad \text{for all} \quad u, v \in \mathbb{R}^N, \tag{2.35}
\]

it follows that

\[||y_n - z_n|| \leq ||y_{n-1} - z_{n-1}||.\]

This definition is known to be stronger than that of A-stability because of the requirement for the sequence \{||y_n - z_n||\} to be non-increasing rather than only bounded. The relation (2.35) is called the “contractivity condition”. In [4], Burrage and Butcher extended the system of the above definition to the non-autonomous form \(y'(x) = f(x, y(x))\) and arrived at the so-called concept of “BN-stability”. The sufficient condition for BN-stability is related to the so-called condition for “algebraic stability” introduced by Burrage and Butcher ([4] 1979), and also by Crouzeix ([32] 1979). It has been shown that BN-stability is equivalent to algebraic stability for any non-confluent Runge-Kutta method (by non-confluent me mean that the abscissae are distinct). A number of further investigations have been carried out on these ideas, for example, the equivalence between B-stability, BN-stability and algebraic stability for any irreducible Runge-Kutta method. For more details, one can refer to [15] and [45].

Although implicit Runge-Kutta methods have good stability and accuracy for stiff problems, unlike explicit Runge-Kutta methods, a complicated iteration scheme
for the solution of the internal stages is necessary. The computation of the stage values is typically the most expensive component in their implementation. If the dimension of the initial value system is $N$, then the nonlinear stage value system (2.18) for an $s$-stage Runge-Kutta method can be written as

$$Z = h(A \otimes I_N)F,$$  \hfill (2.36)

where $Z$ is the $sN$ dimensional vector made up from subvectors $Y_i - y_0$. Because implicit Runge-Kutta methods are proposed for the solution of stiff problems, functional iteration is not feasible for solving (2.36) and the "Newton-Raphson" algorithm is used instead [51]. In this case, the linear system of the Newton method can be written as

$$M(\Delta Z) = -Z + h(A \otimes I_N)F,$$

where $M$ is the Jacobian matrix, and $\Delta Z$ is the Newton update. That is $Z + \Delta Z$ becomes the next approximation. Hence, in each iteration of a full Newton-Raphson scheme, the total computation cost will include

1. Evaluation of $F$,
2. Evaluation of $-Z + h(A \otimes I_N)F$,
3. $LU$ factorization of the Jacobian matrix $M$,
4. Back substitution to yield the Newton update.

It is easy to see that the cost of each iteration grows rapidly as $s$ or $N$ increases. The $LU$ factorizations and the back substitutions need approximately $s^3N^3$ and $s^2N^2$ operations respectively. Hence for high order methods, the cost is increasingly expensive. In practice, the cost of the third item can be lowered because the Jacobian matrix $M$ can be kept unchanged over several steps and the Jacobian required for each stage can also be kept constant for all stages or even for several steps. That is $M$ can be written in the form

$$M = I_s \otimes I_N - hA \otimes J,$$  \hfill (2.37)
where $J = \frac{\partial f}{\partial y}$. In order to reduce the cost further, we need to simplify the system (2.37). It is easy to see that the key to this simplification is to consider a special type of coefficient matrix $A$ and to incorporate transformations into the computations. This leads to singly-implicit Runge-Kutta methods.
Chapter 3

Singly–Implicit Methods

Although Implicit Runge-Kutta methods have good stability and accuracy, they impose higher computational costs because of the non-triangular coefficient matrices. One of the approaches proposed to reduce these costs is to use the coefficient matrix \( A = [a_{ij}]_{N \times N} \) which has the property \( a_{ij} = 0 \), where \( i < j \). Therefore they retain the implicit \( A \) matrix for solving stiff problems. By using this, the factorizations can be reduced from \( s^3N^3 \) (suppose the system is \( N \) dimensional) to \( sN^3 \). Such methods are called semi-implicit methods or diagonally implicit methods. When the diagonal elements of the \( A \) matrix are equal (so-called SDIRK), the operations will reduce to \( N^3 \). However, their stage order is only one. For some very stiff problems this may cause order reduction. Following Butcher’s idea ([13] 1976), Burrage ([2] 1978) introduced another approach in which the coefficient matrix \( A \) has a one point spectrum property (SIRK). These methods can also have high stage order.

In this chapter, we mainly consider SIRK methods. In the first section, we will discuss how to reduce the operations by the use of linear algebra. The family of singly-implicit Runge-Kutta methods and their stability properties are discussed in the second section. In the third section, it is shown how to construct particular SIRK methods, along with appropriate transformation matrices. Because of stability requirements, for order \( \geq 3 \), some abscissae of the SIRK methods are outside the integration interval. Another member of SIRK methods used to over-
CHAPTER 3. SINGLY-IMPLICIT METHODS

come this flaw and take computational advantage from SDIRK is introduced by Butcher and Cash ([23] 1990). The main idea for this combination (SDIRK and SIRK) is to add some additional diagonal stages to the classical SIRK methods such that the coefficient matrix still has a one point spectrum. It turns out that the error constants of these new methods can be much smaller than the classical SIRK methods and still have good stability and high stage order. These methods were also implemented by Butcher, Cash and Diamantakis (so-called DESI method) in 1996 [24]. These will be discussed in the last section.

3.1 Butcher’s transformation

The first attempt to reduce the computational costs for IRK methods is to seek some implicit Runge-Kutta method which has a lower triangular matrix. This was pointed out by Butcher in 1963 [8]. He called these methods “semi-implicit” methods. Further related work has been done by several persons, Alt, 1971, Nørsett, ([55] 1974), Crouzeix ([31] 1975) and Alexander ([1] 1977). In the Newton-type iteration, we have only a N-dimensional system which needs to be solved for each iteration. Crouzeix also pointed out the usefulness of this idea for solving linear differential equations arising from discretization of linear parabolic partial differential equations by the finite element method [31]. In the work of Alexander, he called these methods DIRK which are the same as the well-known SDIRK. Some new order 2,3 methods with good stability have been derived. He also showed that no 4-stage A-stable method has order 5. In fact, for SDIRK methods, we can only have stage number $s = 1, 2, 3, 5$ A-stable methods which can have the maximum attainable order of $p = s + 1$. For the case of L-stable methods, order $p = s, s = 1, 2, \ldots, 8$ methods are available except for $s = 7$.

Because the study of the stability and the error behaviour for these methods are very similar to SIRK methods, we will not consider these problems until after the study of SIRK methods. In practice, the stiffly-accurate SDIRK methods are preferred (hence L-stable). The existing code SIMPLE implemented by Nørsett and Thomsen ([56] 1984) is based on the 3-stage third order B-stable SDIRK methods. Another code $SDIRK$ implemented by Hairer and Wanner ([45] 1987).
is based on the 5-stage fourth-order L-stable SDIRK methods as shown in the corresponding Butcher table,

\begin{align*}
\begin{array}{c|ccc}
1/4 & 1/4 & \\
3/4 & 1/2 & 1/4 & \\
11/20 & 17/50 & -1/25 & 1/4 & \\
1/2 & 371/1360 & -137/2720 & 15/544 & 1/4 & \\
1 & 25/24 & 49/48 & 125/16 & -85/12 & 1/4 & \\
\end{array}
\end{align*}

(3.1)

Although SDIRK methods have computational advantages and retain good stability, their stage order is only one, and this is no doubt a disadvantage. The question of how to increase the stage order without losing the original advantages leads to the development of SIRK methods.

As mentioned in last chapter, when we are using an s-stage implicit Runge-Kutta method (1.1) to solve the N dimensional ordinary differential equation (2.8) (cf. section 2.3), the sN × sN nonlinear system and the equations for output approximation can be written

\begin{align*}
Y &= e \otimes y_0 + h(A \otimes I_N)F, \quad (3.2) \\
y_1 &= y_0 + h(b^T \otimes I_N)F. \quad (3.3)
\end{align*}

It is suggested that for accuracy and efficiency considerations, the modified Newton iterations should be used to solve (3.2). That is the Jacobian matrix \( J = f_y(e \otimes y_0) \) for each iteration in one step (several steps can also be achieved). If we assume \( Z = Y - e \otimes y_0 \), then solving (3.2) is equivalent to solving

\[ G(Z) = Z - hAf(Z + e \otimes y_0) = 0. \]

Let \( \Delta Z \) be the Newton update. That is, \( Z^{[l+1]} = Z^{[l]} + \Delta Z \), where \( Z^{[l]} \) is the value in the \( l \)-th iteration. Using the Newton method to solve this nonlinear system yields

\[ [(I_s \otimes I_N) - hA \otimes J] \Delta Z = -Z + h(A \otimes I_N)F. \quad (3.4) \]
CHAPTER 3. SINGLY-IMPLICIT METHODS

where $\otimes$ is the Kronecker product and $F = f(Z + e \otimes y_0)$. The nonlinear Jacobian matrix on the left hand side of (3.4) has $sN \times sN$ equations and this will take about $s^3 N^3 + O(N^2)$ operations for the LU factorization and $s^2 N^2 + O(N)$ operations for back substitutions. These computation costs are high, especially for large dimensional systems. In order to overcome these difficulties, Butcher ([13] 1976) proposed to reduce the computational costs by using the Jordan canonical form of $A$ on the left hand side of (3.4) instead of using $A$. That is, for any $A$ matrix, there exists a nonsingular matrix $T$ such that $T^{-1} A T = \tilde{A}$, where $\tilde{A}$ is the Jordan canonical form of $A$. Hence (3.4) can be transformed to

$$[(I_s \otimes I_N) - h\tilde{A} \otimes J] \Delta \tilde{Z} = -\tilde{Z} + h(\tilde{A} \otimes I_N) \tilde{F},$$  \hspace{1cm} (3.5)

where

$$\tilde{Z} = (T^{-1} \otimes I_N) Z,$$
$$\tilde{F} = (T^{-1} \otimes I_N) F,$$
$$\Delta Z = (T \otimes I_N) \Delta \tilde{Z}.$$

Because of these transformations (so-called "Butcher transformation"), the operations for the LU factorizations will become proportional to $dN^3$, $d$ is the number of distinct eigenvalues of $A$, and the operations for the back substitutions are reduced to $sN^2$. But there are three transformation costs ($s^2 N$) for each $\Delta \tilde{Z}, \tilde{Z}$ and the update $\Delta Z = (T \otimes I_N) \Delta \tilde{Z}$ which need to be taken into account. For large systems, these are reasonable savings.

From this approach, one can see that the biggest reduction in the computational costs occurs when the coefficient matrix $A$ has a one point spectrum. The factorization cost will then reduce to $N^3$. Methods with this property are called "SIRK" (singly-implicit Runge-Kutta) methods and they will be discussed in the next section. Meanwhile, when we are dealing with low dimensional systems, the transformation costs become crucial. Codes based on SIRK methods may not be attractive because of the high transformation costs. To fix this weakness, Butcher ([16] 1988) considered reforming (3.5) to reduce the transformation count. Because the full transformation matrix $T$ causes high operation costs, it is possible to make the transformation matrix a lower triangular matrix by

40
assuming \( T = UL \), where \( U \) is a unit upper triangular matrix. Then when we substitute \( T = UL \) into \( T^{-1}AT = \hat{A} \), we have

\[
(UL)^{-1}AUL = L^{-1}U^{-1}AUL = \hat{A},
\]

this implies

\[
U^{-1}AU = L\hat{A}L^{-1} = \hat{A}.
\] (3.6)

Here \( \hat{A} \) is a lower triangular matrix with one single diagonal element \( \lambda \). When \( U \) is chosen to be the new transformation matrix, the nonlinear system (3.4) will become

\[
[(I_s \otimes I_N) - h\hat{A} \otimes J]\Delta \hat{Z} = -\hat{Z} + h(\hat{A} \otimes I_N)\hat{F},
\] (3.7)

where

\[
\hat{Z} = (U^{-1} \otimes I_N)Z,
\hat{F} = (U^{-1} \otimes I_N)F,
\Delta Z = (U \otimes I_N)\Delta \hat{Z}.
\]

In this case, the costs of LU factorization remain the same and the costs for back substitution will become \( sN^2 + \frac{s(s-1)N^2}{2} \) because the \( \hat{A} \) matrix is a lower triangular matrix. But each of the transformation operations is reduced to about \( \frac{1}{2}s(s-1)N \). This is a considerable improvement particularly in solving small dimensional systems. A Matlab subroutine "u1" which factorizes a matrix \( T = UL \) can be found in Appendix A.1.

A further idea to improve the calculations in (3.7) is that if we assume \( M = \lambda \hat{A}^{-1} \), then we have

\[
M = \begin{bmatrix}
1 & 0 & 0 & \cdots & 0 \\
m_{21} & 1 & 0 & \cdots & 0 \\
m_{31} & m_{32} & 1 & \cdots & 0 \\
\vdots & \vdots & \vdots & \ddots & \vdots \\
m_{s1} & m_{s2} & m_{s3} & \cdots & 1
\end{bmatrix}.
\]
 CHAPTER 3. SINGLY-IMPICIT METHODS

Table 3.1: Total approximate number of operations for solving $N$ dimensional systems

<table>
<thead>
<tr>
<th>Method</th>
<th>$N=2$</th>
<th>$N=5$</th>
<th>$N=10$</th>
<th>$N=50$</th>
<th>$N=250$</th>
</tr>
</thead>
<tbody>
<tr>
<td>IRK</td>
<td>252</td>
<td>3600</td>
<td>27900</td>
<td>3397500</td>
<td>422437500</td>
</tr>
<tr>
<td>SIRK with (3.5)</td>
<td>86</td>
<td>410</td>
<td>1870</td>
<td>141350</td>
<td>16006750</td>
</tr>
<tr>
<td>SIRK with (3.7)</td>
<td>62</td>
<td>350</td>
<td>1750</td>
<td>140750</td>
<td>16003750</td>
</tr>
<tr>
<td>SIRK with (3.8)</td>
<td>56</td>
<td>290</td>
<td>1480</td>
<td>133400</td>
<td>15817000</td>
</tr>
</tbody>
</table>

Multiplying both sides of (3.7) by $M$ gives

$$[(M \otimes I_N) - h\lambda I_s \otimes J] \Delta \hat{Z} = -\hat{Z} + (h\lambda I_s \otimes I_N) \hat{F},$$  \hspace{1cm} (3.8)

where

$$\hat{Z} = (M \otimes I_N) \hat{Z}.$$  

The advantage of this modification is to reduce the operations for the back substitution. Note that the entries in the left-lower part of the matrix of the left hand side in (3.7) have changed from $-\bar{a}_{ij} h J$ to $m_{ij} I_N$. It will help to reduce the costs from $\frac{N^2}{2}$ to $\frac{N^2}{2}$. But in this case we also need another triangular matrix transformation. Table 3.1 shows the total operations produced by using an order-3 IRK method and SIRK methods with the nonlinear system (3.4), (3.5), (3.7), (3.8) to solve different dimensional systems. Clearly, SIRK is superior to IRK. For small systems, one can see the cost is reduced by using (3.7), but for large systems, using (3.5) and (3.7) have similar performances; while adopting (3.8) is the best option among them for all systems. We give an example to illustrate the savings from these transformations.
Example 3.1 Consider the order-2 SIRK method with corresponding tableau

\[
\begin{array}{c|cc}
3 - 2\sqrt{2} & 5 - 3\sqrt{2} & 7 - 5\sqrt{2} \\
1 & 1 + 1\sqrt{2} & 1 - 1\sqrt{2} \\
\hline
1 + 1\sqrt{2} & 3 - 1\sqrt{2} & 1 + 1\sqrt{2} \\
\end{array}
\]

If \( T \) is the transformation matrix satisfying \( \widetilde{A} = T^{-1}AT \), then we will have

\[
\widetilde{A} = \begin{bmatrix}
\lambda & 0 \\
-\lambda & \lambda
\end{bmatrix} = \begin{bmatrix}
1 - \frac{\sqrt{2}}{2} & 0 \\
\frac{\sqrt{2}}{2} - 1 & 1 - \frac{\sqrt{2}}{2}
\end{bmatrix},
\]

where

\[
T = \begin{bmatrix}
1 & 1 - 1 + \sqrt{2} \\
1 & -1 - \sqrt{2}
\end{bmatrix},
\]

and \( \lambda \) is the only eigenvalue of \( A \). This matrix \( T \) can be factorized to be the product of a unit upper triangular matrix \( U \) and a lower triangular matrix \( L \). That is

\[
T = UL = \begin{bmatrix}
1 & -3 + 2\sqrt{2} \\
0 & 1
\end{bmatrix} \begin{bmatrix}
4 - 2\sqrt{2} & 0 \\
1 & -1 - \sqrt{2}
\end{bmatrix}.
\]

Then \( \widetilde{A} \) can be derived from (3.6), \( LAL^{-1} = \widetilde{A} \).

\[
\widetilde{A} = \begin{bmatrix}
1 - \frac{\sqrt{2}}{2} & 0 \\
\frac{1}{4} + \frac{\sqrt{2}}{4} & 1 - \frac{\sqrt{2}}{2}
\end{bmatrix} = \lambda \begin{bmatrix}
1 & 0 \\
1 + \frac{3}{4}\sqrt{2} & 1
\end{bmatrix},
\]

a lower triangular matrix. To use (3.8), we need to assume \( M = \lambda \widetilde{A}^{-1} \). Hence we have

\[
M = \begin{bmatrix}
1 & 0 \\
-1 - \frac{3}{4}\sqrt{2} & 1
\end{bmatrix}.
\]
Table 3.2: Total flops for solving the Kaps problem (3.9)

<table>
<thead>
<tr>
<th>method</th>
<th>number of flops</th>
<th>global error</th>
</tr>
</thead>
<tbody>
<tr>
<td>IRK</td>
<td>89274</td>
<td>$1.852 \times 10^{-7}$</td>
</tr>
<tr>
<td>SIRK with (3.5)</td>
<td>46198</td>
<td>$1.852 \times 10^{-7}$</td>
</tr>
<tr>
<td>SIRK with (3.7)</td>
<td>42166</td>
<td>$1.852 \times 10^{-7}$</td>
</tr>
<tr>
<td>SIRK with (3.8)</td>
<td>41414</td>
<td>$1.852 \times 10^{-7}$</td>
</tr>
</tbody>
</table>

Experiment 3.1 We now test the 2-dimensional Kaps problem [49] with constant step-size.

The Kaps problem

\[
\begin{align*}
y_1'(x) &= -1002y_1(x) + 1000y_2(x)^2, & x \in [0, 10], \\
y_2'(x) &= y_1(x) - y_2(x)(1 + y_2(x)), \\
y_1(0) &= 1, & y_1(x) = \exp(-2x), \\
y_2(0) &= 1, & y_2(x) = \exp(-x),
\end{align*}
\]  

(3.9)

The step number is chosen to be 100 and the integration interval is [0, 10]. As shown in Table 3.2, using the transformations (3.5), (3.7), (3.8) really does lower computational costs and we do not lose accuracy.

3.2 SIRK methods and their stability

As we mentioned in the last section, by the use of some proper transformations, the single-implicitness of the coefficient matrix $A$ is more attractive in reducing the computational costs. Therefore, a special family of implicit Runge-Kutta methods was introduced by Burrage ([2] 1978). To avoid order reduction and to
make it easier to construct higher order methods and an error estimator, high stage order methods are proposed for solving stiff problems. Hence, in addition to having the condition: the method has a one-point spectrum property, the standard s-stage SIRK methods have stage order and order each equal to s as well. And this leads to SIRK methods being collocation methods.

We now consider the stability property of SIRK methods. According to (2.32), (2.33) in the last chapter, the stability function \( R(z) \) for an s-stage SIRK method can be written as

\[
R(z) = \frac{P(z)}{(1 - \lambda z)^s},
\]

where \( \lambda \) is the eigenvalue of \( A \) and the degree of \( P(z) \) is at most \( s \). The following result given in [15] is used to derive \( P(z) \) and the error constant for the method.

**Lemma 3.1** If for \( n = -1, -2, \ldots \), define \( L_s^{(n)} \) by \( L_s^{(-1)}(0) = L_s^{(-2)}(0) = \cdots = 0 \) with the derivative of \( L_s^{(n-1)} \) always equal to \( L_s^{(n)} \) and \( L_s^{(0)} \) identified with \( L_s \), then we have

\[
(1 - z)^s \exp(\theta z) = (-1)^s \sum_{k=0}^{\infty} L_s^{(s-k)}(\theta) z^k, \tag{3.10}
\]

where

\[
L_s(z) = \sum_{k=0}^{s} \binom{s}{k} (-z)^k k!, \tag{3.11}
\]

is the Laguerre polynomial of degree \( s \).

Because the order of an s-stage SIRK method is equal to \( s \), \( R(z) \) satisfies

\[
R(z) = \exp(z) + O(z^{s+1}).
\]

Thus we can have

\[
P(z) = (1 - \lambda z)^s \exp(z) + O(z^{s+1}).
\]

\( P(z) \) can be derived by the use of Lemma 3.1. Let \( \theta = \frac{1}{\lambda} \) and substitute \( \lambda z \) for \( z \) in (3.10). We then have

\[
P(z) = (-1)^s \sum_{k=0}^{s} \lambda^k L_s^{(s-k)} \left( \frac{1}{\lambda} \right) z^k. \tag{3.12}
\]
Also from
\[ \exp(z) - R(z) = C_{s+1}z^{s+1} + O(z^{s+2}), \]  
(3.13)
the error constant can be obtained by multiplying both sides of (3.13) by \((1 - \lambda z)^s\). Because the degree of \(P(z) \leq s\), the error constant \(C_{s+1}\) will be the coefficient of \(z^{s+1}\) in \((1 - \lambda z)^s \exp(z)\). Let \(L_s^{(-1)}\) be the anti-derivative of \(L_s\). From Lemma 3.1, we find
\[ C_{s+1} = (-1)^s \lambda^{s+1} L_s^{(-1)} \left( \frac{1}{\lambda} \right). \]  
(3.14)

Using some properties of standard Laguerre polynomials,
\[ L_{s+1}^{(1)}(z) = L_s^{(1)}(z) - L_s(z), \]  
(3.15)
\[ zL_s^{(1)}(z) = sL_s(z) - sL_{s-1}(z), \]  
(3.16)
we simplify (3.14) as follows,
\[
C_{s+1} = (-1)^s \lambda^{s+1} L_s^{(-1)} \left( \frac{1}{\lambda} \right) = (-1)^s \lambda^{s+1} \int_0^{1/\lambda} L_s(x) dx \\
= (-1)^s \lambda^{s+1} \left( L_{s+1} \left( \frac{1}{\lambda} \right) - L_s \left( \frac{1}{\lambda} \right) \right) \\
= \frac{\lambda^s (-1)^{s+1}}{s+1} L_{s+1}^{(1)} \left( \frac{1}{\lambda} \right). \]  
(3.17)

It can be seen that the stability property and the error constant are closely related to the choice of \(\lambda\). From (3.17), we will be able to have a one order higher \((s+1)\) method if we let \(\frac{1}{\lambda}\) be a zero of \(L_{s+1}^{(1)}\). Meanwhile, \(\lambda\) must satisfy the stability requirement as well. It turns out that there are only 4 cases \((s = 1, 2, 3, 5)\) which can give one order higher as well as A-stability. An alternative approach is quite promising in constructing methods for practical purposes. It is clear from (3.12) that the error constant for \(z^s\) is
\[ C_s = (-1)^s \lambda^s L_s \left( \frac{1}{\lambda} \right). \]

If we choose \(\frac{1}{\lambda}\) to be a zero of \(L_s\), then the method obtained is order \(s\) but L-stable because the degree of \(P(z)\) in (3.12) is less than \(s\). In this case, (3.12) will become
\[
P(z) = (-1)^s \sum_{k=0}^{s-1} \lambda^k L_s^{(s-k)} \left( \frac{1}{\lambda} \right) z^k. \]  
(3.18)
3.2. SIRK METHODS AND THEIR STABILITY

Using the E-polynomial mentioned in Chapter 2, we are able to find the bounds for $\lambda$ for the method to be stable. We give an example to demonstrate this.

Table 3.3: $\lambda$ for L-stability with s-stage SIRK methods

<table>
<thead>
<tr>
<th>s</th>
<th>$\lambda$</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>$1 \pm \frac{\sqrt{2}}{2}$</td>
</tr>
<tr>
<td>3</td>
<td>0.435866521508458994</td>
</tr>
<tr>
<td>4</td>
<td>0.572816062482134886</td>
</tr>
<tr>
<td>5</td>
<td>0.278053841136452325</td>
</tr>
<tr>
<td>6</td>
<td>0.334142367068050436</td>
</tr>
<tr>
<td>8</td>
<td>0.234373159605583558</td>
</tr>
</tbody>
</table>

**Example 3.2** Take $s = 2$, from (3.11) and (3.18),

$$P(z) = L_2^{(2)} \left( \frac{1}{\lambda} \right) + L_2^{(1)} \left( \frac{1}{\lambda} \right) \lambda z = 1 + (1 - 2\lambda)z.$$ 

And with $Q(z) = (1 - \lambda z)^2$,

$$E(y) = Q(iy)Q(-iy) - P(iy)P(-iy) = (1 + \lambda^2 y^2)^2 - (1 + (1 - 2\lambda)^2 y^2),$$

$$= (-1 + 4\lambda - 2\lambda^2)y^2 + \lambda^4 y^4.$$ 

This implies

$$E(y) \geq 0, \quad \forall y \iff 1 - \frac{\sqrt{2}}{2} \leq \lambda \leq 1 + \frac{\sqrt{2}}{2}.$$ 

Because the two zeros of $L_2^{(1)}(\frac{1}{2})$ are $1 - \frac{\sqrt{2}}{2}$ and $1 + \frac{\sqrt{2}}{2}$, we have two options of $\lambda$ for L-stability and the order is 2 when $s = 2$. 

When $s = 7$, $\lambda$ is not available for L-stability. The choices of $\lambda$ for L-stability for order up to 8 has been given in [2]. Table 3.3 shows the value of $\lambda$ for L-stability.
3.3 The transformation matrix

Once the crucial value, the single eigenvalue $\lambda$ of $A$, has been determined by stability, it is natural to hope that we can evaluate the coefficients of the method together with the transformation matrix for practical use. This work has been done by Butcher ([14] 1979). Because the high stage-order property is proposed, we will from now on consider only SIRK methods which are collocation methods. We assume that the abscissae are distinct, and that the $s$-stage method satisfies $C(s)$ and $B(s)$. From the single-implicitness of $A$ and the condition $C(s)$, it is found that the abscissae chosen must be proportional to the zeros of the Laguerre polynomial $L_s$. This can be verified by the following theorem [15].

**Theorem 3.2** For an $s$-stage SIRK method with $\sigma(A) = \{\lambda\}$, the abscissae $c_1, \ldots, c_s$ satisfy

$$c_i = \lambda \xi_i, \quad i = 1, 2, \ldots, s,$$

where $\xi_i$ are the zeros of the $s$-degree Laguerre polynomial $L_s$.

**Proof:** If we let $c^k, k = 0, 1, \ldots$, denote the component-wise $k$-th power of $c$, and use the stage-order condition (2.21), we have for $k = 0, 1, \ldots, s$,

$$A^k e = A^{k-1}(Ae) = A^{k-1}c = \frac{1}{2} A^{k-2}c^2 = \ldots = \frac{1}{(k-1)!} A^{k-1}c = \frac{c^k}{k!}.$$

By the singly-implicitness of $A$ and the Cayley-Hamilton theorem, we have

$$0 = (A - \lambda I)^s e = \sum_{k=0}^{s} \binom{s}{k} (-1)^k \lambda^k A^{s-k} e = \sum_{k=0}^{s} \binom{s}{k} \frac{(-1)^k}{(s-k)!} \lambda^k c^{s-k}.$$

Multiplying both sides of the above equation by $\lambda^{-s}$, we have each component of $c$ satisfies the equation

$$\sum_{k=0}^{s} \frac{(-1)^{s-k}}{(s-k)!} \binom{s}{k} \left( \frac{c_i}{\lambda} \right)^{s-k} e = 0.$$

This is equivalent to $L(c_i/\lambda) = 0$. \qed
Furthermore, the transformation matrix can be found using (3.15). The stage order conditions (2.20),

\[
\sum_{j=1}^{s} a_{ij} c_j^{k-1} = \frac{1}{k} c_i^k, \quad k = 1, 2 \ldots, s,
\]

are equivalent to

\[
\sum_{j=1}^{s} a_{ij} p(c_j) = \int_0^{c_i} p(c) dc,
\]

(3.19)

where \( p(x) \) is any polynomial with degree \( \leq s - 1 \). Again, we let \( c_j = \lambda \xi_j, \quad j = 1, \ldots, s \), \( \xi_j \) is the zero of the Laguerre polynomial \( L_s \). Since \( L_k(0) = L_{k+1}(0) = 1 \), from (3.15) and (3.19), for \( k = 0, 1, \ldots, s - 1 \), we have

\[
\sum_{j=1}^{s} a_{ij} L_k(\xi_j) = \sum_{j=1}^{s} a_{ij} L_k \left( \frac{c_j}{\lambda} \right) = \int_0^{c_i} L_k \left( \frac{c}{\lambda} \right) dc = \int_0^{\xi_i} \lambda L_k(\xi) d\xi = \lambda \int_0^{\xi_i} \left( L_k(\xi) - L_{k+1}(\xi) \right) d\xi = \lambda L_k(\xi_i) - \lambda L_{k+1}(\xi_i).
\]

Hence, if the transformation \( T \) is chosen to be

\[
T = [v_1, v_2, \ldots, v_s], \quad \text{where} \quad v_k = \begin{bmatrix} L_{k-1}^{(1)}(\xi_1) \\ L_{k-1}^{(2)}(\xi_2) \\ \vdots \\ L_{k-1}^{(s)}(\xi_s) \end{bmatrix},
\]

(3.20)

for \( k = 1, \ldots, s \), then the coefficient matrix \( A \) can be transformed to the Jordan canonical matrix which has \( \lambda \) in the diagonal and \(-\lambda \) in the sub-diagonal. This is

\[
T^{-1}AT = \begin{bmatrix} \lambda & 0 & 0 & \cdots & 0 & 0 \\ -\lambda & \lambda & 0 & \cdots & 0 & 0 \\ 0 & -\lambda & \lambda & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & -\lambda & \lambda \end{bmatrix}
\]
CHAPTER 3. SINGLY-IMPLICIT METHODS

From the L-stability requirement, the eigenvalue $\lambda$ must satisfy $L_s(\frac{1}{3}) = 0$. From Theorem 3.2, this will mean that one of the abscissae must be chosen to be 1. For the standard SIRK method, we set $b_j = a_{ij}$ when $c_i = 1, j = 1, 2, \ldots, s$. It can thus be concluded that the procedures for constructing an $s$-stage standard SIRK method and the corresponding transformation matrix are as follows:

1. Choose $\lambda$ for stability from Table 3.3;
2. Choose the abscissae $c_i = \lambda \xi_i, i = 1, \ldots, s$, where $\xi_i, i = 1, \ldots, s$ are the zeros of $L_s$;
3. Form $C(s)$ to determine the coefficient matrix $A$;
4. $b^T = e^T_j A$, where $e_j$ is the zero vector except the $j$-th component is 1 ($j$ is the value for which $c_j = 1$);
5. Find the transformation matrix $T$ from (3.20).

We give some SIRK examples and their associated transformation matrices below.

**Example 3.3** It is easy to see that the order-1 L-stable SIRK method is the implicit Euler method, so we begin with the order-2 SIRK method. In this case, the eigenvalue $\lambda$ has been chosen to be $\lambda = 1/\xi_2 = 1 - \frac{\sqrt{2}}{2}$ (method with another eigenvalue $1 + \frac{\sqrt{2}}{2}$ is also $A$-stable, but it has larger error constant) so that $c_2 = 1$. Hence $c_1 = \lambda \xi_1 = \lambda(2 - \sqrt{2}) = 3 - 2\sqrt{2}$. By $C(2)$, we have the following equations

\[
\begin{align*}
a_{11} + a_{12} &= c_1, & a_{21} + a_{22} &= c_2, \\
a_{11}c_1 + a_{12}c_2 &= \frac{c_1^2}{2}, & a_{21}c_1 + a_{22}c_2 &= \frac{c_2^2}{2}.
\end{align*}
\]

Therefore, the coefficient matrix $A$ of the order-2 method is easy to determine uniquely. The weight $b^T$ is the second row of $A$. We have the corresponding Butcher tableau,

\[
\begin{array}{c|ccc}
3 - 2\sqrt{2} & 5 - 3\sqrt{2} & 7 - 5\sqrt{2} \\
1 & 1 + \sqrt{2} & 3 - \sqrt{2} \\
1 & 1 + \sqrt{2} & 3 - \sqrt{2} \\
\end{array}
\]

(3.21)
From (3.20), the transformation matrix \( T \) is given by

\[
T = [v_1, v_2] = \begin{bmatrix}
L_0(\xi_1) & L_1(\xi_1) \\
L_0(\xi_2) & L_1(\xi_2)
\end{bmatrix} = \begin{bmatrix}
1 & -1 + \sqrt{2} \\
1 & -1 - \sqrt{2}
\end{bmatrix}.
\]

\[\square\]

**Example 3.4** The order 3 SIRK method has \( \lambda \approx \frac{1}{\xi} = 0.435866521508459 \).

The abscissae of the method are obtained by

\[
c_1 = \frac{\xi_1}{\xi_2} \approx 0.1812220979694, c_2 = 1.00000, c_3 = \frac{\xi_3}{\xi_2} \approx 2.7415764837792,
\]

where \( \xi_1, \xi_2, \xi_3 \) are the roots of \( L_3(x) \). In a similar way, it is easy to derive the coefficient matrix \( A \) by \( C(3) \). Approximately, the corresponding tableau is of the form

\[
\begin{array}{ccc}
c_1 & 0.208637 & -0.030875 & 0.003460 \\
c_2 & 0.574386 & 0.442669 & -0.017056 \\
c_3 & 0.154424 & 1.930859 & 0.656292 \\
\end{array}
\]

\[
\begin{array}{c}
0.574386 \\
0.442669 \\
-0.017056
\end{array}
\]

(3.22)

The transformation matrix \( T \) is given by

\[
T = \begin{bmatrix}
L_0(\xi_1) & L_1(\xi_1) & L_2(\xi_1) \\
L_0(\xi_2) & L_1(\xi_2) & L_2(\xi_2) \\
L_0(\xi_3) & L_1(\xi_3) & L_2(\xi_3)
\end{bmatrix} = \begin{bmatrix}
1 & 0.584225 & 0.254885 \\
1 & -1.294280 & -0.956700 \\
1 & -5.289945 & 8.201814
\end{bmatrix}.
\]

\[\square\]

We note that one of the abscissae for order-3 SIRK method is greater than 1 which means that one of the approximated derivatives is outside the integration interval. This is no doubt a handicap. As a matter of fact, if we insist in using A-stable methods, when the order is higher than 2, some of the abscissae are always larger than 1. For example, the largest abscissae goes to 5.3816 for order 4 and 3.1482, 5.34056 for order 5, 6 respectively. An alternative way is to use \( A(\alpha) \)-stable methods rather than A-stable methods. Furthermore, the
stability regions of rational approximations to the exponential function where the approximation function has a single pole has been studied by Wolfbrandt [65]. The existing variable order code, \textit{STIRDE}, implemented by Burrage, Butcher and Chipman ([5] 1979) has adopted the $A(\alpha)$-stable SIRK methods up to order 15 with $\alpha \geq 1.45$ ([6] 1980). We also remark that because of good stability and stiff-accuracy, the SIRK methods are also reasonable alternatives for DAE problems (Nilsen [53]).

3.4 DESI methods

Low stage order (only one) makes SDIRK methods unlikely candidates for highly accurate codes. Meanwhile SIRK methods are inferior for small dimensional systems because of their relatively high transformation costs. Furthermore, higher order SIRK methods have some abscissae outside the integration interval. To overcome these difficulties, we need to have some extra free parameters. In 1990, Butcher and Cash [23] combined those two concepts (SDIRK and SIRK) together. They arrived at the so-called DESI (Diagonally Extended Singly Implicit) methods. The main skeleton of DESI methods is to use the SIRK methods to create the input value for the following several diagonal stages within one step. Because of those extra internal stages, DESI can have more freedom in determining its coefficients, such as the abscissae. It turns out that their error constants can be much smaller than the error constants of the classical SIRK methods. This also means that the DESI methods will integrate further forward than their corresponding SIRK methods. Therefore, the overall computational costs can be reduced considerably. On the other hand, the stage values being $p$ (if there are $p$ singly-implicit stages) will be an advantage over SDIRK methods.

The $s$-stage, order-$p$ DESI method with the corresponding tableau can be written
as

\[
\begin{array}{c|cccccccc}
& a_{11} & a_{12} & \cdots & a_{1p} & 0 & 0 & \cdots & 0 \\
c_1 & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\
c_{p} & a_{p1} & a_{p2} & \cdots & a_{pp} & 0 & 0 & \cdots & 0 \\
c_{p+1} & a_{p+1,1} & a_{p+1,2} & \cdots & a_{p+1,p} & \lambda & 0 & \cdots & 0 \\
c_{p+2} & a_{p+2,1} & a_{p+2,2} & \cdots & a_{p+2,p} & a_{p+2,p+1} & \lambda & \cdots & 0 \\
\vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\
c_{s} & a_{s1} & a_{s2} & \cdots & a_{sp} & a_{s,p+1} & a_{s,p+2} & \cdots & \lambda \\
\hline
b_1 & b_2 & \cdots & b_p & b_{p+1} & b_{p+2} & \cdots & b_s
\end{array}
\]

where the \( p \times p \) matrix

\[
A_p = \begin{bmatrix}
  a_{11} & a_{12} & \cdots & a_{1p} \\
  a_{21} & a_{22} & \cdots & a_{2p} \\
  \vdots & \vdots & \ddots & \vdots \\
  a_{p1} & a_{p2} & \cdots & a_{pp}
\end{bmatrix},
\]

has a one-point spectrum \( \{\lambda\} \) which is the same as the one appearing in the diagonal block. Hence we will not have any new LU factorization. In order to retain \( p \) as the stage order, \( c_1, \ldots, c_p \) are chosen as \( \lambda \xi_1, \ldots, \lambda \xi_p \) respectively, where \( \xi_1, \xi_2, \ldots, \xi_p \) are the zeros of the Laguerre polynomial \( L_p \). The matrix \( A_p \) thus can be found using the stage order conditions

\[
\sum_{j=1}^{p} a_{ij} c_j^{k-1} = \frac{1}{k} c_i^k, \quad k = 1, 2, \ldots, p. \quad (3.24)
\]

The methods are proposed to be strongly stable at infinity, and therefore an A-stable method can be L-stable. The last abscissa \( c_s \) is always chosen to be 1 and \( b_i = a_{si}, \quad i = 1, \ldots, s \), retains stiff accuracy. The other coefficients are determined to satisfy the stability requirement and the stage-order conditions. We give an example to illustrate how to construct DESI methods.
Example 3.5 Consider an order $p = 2$ DESI method $(A, b^T, c)_s$. Let $s = p + 1$ first, suppose $\lambda$ is the eigenvalue satisfying $\sigma(A) = \{\lambda\}$, and because of the $C(2)$ condition, the $c_i = \lambda \xi_i, i = 1, 2$, where $\xi_1, \xi_2$ are the zeros of the Laguerre polynomial $L_2$. Since $c_3$ has been set to be 1 and $b_i = a_{3i}, i = 1, 2, 3$, the corresponding Butcher tableau will be of the form

\[
\begin{array}{ccc}
   c_1 & a_{11} & a_{12} & 0 \\
   c_2 & a_{21} & a_{22} & 0 \\
   1 & a_{31} & a_{32} & \lambda \\
   a_{31} & a_{32} & \lambda \\
\end{array}
\]

Since the $a_{ij}$ can be determined immediately by $C(2)$ once $\lambda$ is given, all we have to do now is to specify $\lambda$ by the stability constraint. Adopting the $E$-polynomial scheme, we need to derive the stability function $R(z)$ of the method. Because of the singly-implicitness and the $L$-stability, we have

\[
R(z) = \frac{P(z)}{Q(z)},
\]

where

\[
P(z) = 1 + \alpha_1 z + \alpha_2 z^2, \quad Q(z) = (1 - \lambda z)^3.
\]

It is clear that $P(z)$ can be found using $(1 - \lambda z)^3 \exp(z)$. We have

\[
P(z) = 1 + (1 - 3\lambda)z + \left(\frac{1}{2} - 3\lambda + 3\lambda^2\right)z^2.
\]

Hence, the $E$-polynomial is

\[
E(y) = Q(iy)Q(-iy) - P(iy)P(-iy)
\]

\[
= \left(-\frac{1}{4} + 3\lambda - 12\lambda^2 + 18\lambda^3 - 6\lambda^4\right)y^4 + \lambda^6 y^6.
\]

It is easy to see that

\[
E(y) \geq 0, \quad \forall y \geq 0
\]

\[
\Leftrightarrow \frac{1}{4} \left(3 + \sqrt{3} - \sqrt{8 + \frac{14\sqrt{3}}{3}}\right) \leq \lambda \leq \frac{1}{4} \left(3 + \sqrt{3} + \sqrt{8 + \frac{14\sqrt{3}}{3}}\right)
\]

\[
\Leftrightarrow 0.1804253064293985641 \leq \lambda \leq 2.1856000973550400826.
\]
For the purpose of choosing the smaller abscissae and smaller error constant, $\lambda$ is chosen as

$$\lambda = 0.1804253064293985641.$$  

Therefore, it follows that

$$c_1 = 0.10569069751102009486,$$

$$c_2 = 0.6160105282065741617.$$  

By the $C(2)$ condition, the coefficients of the $A$ matrix are found to be

$$a_{11} = 0.1166353275924543058,$$

$$a_{12} = -0.0109446300814342109,$$

$$a_{21} = 0.371795242940231339,$$

$$a_{22} = 0.244215285266342822,$$

$$a_{31} = 0.363089841262914082,$$

$$a_{32} = 0.456484852307687354.$$  

In this case ($p = 2, s = 3$), it is also easy to see that the error constant $C_3$ is the coefficient of $z^3$ in $(1 - \lambda z)^3 \exp(z)$. By (3.12), we find

$$C_3 = (-1)^3 \lambda^3 L_3 \left( \frac{1}{\lambda} \right) = -0.012184856917075198.$$  

We can also consider adding one more diagonal stage to the above method to obtain the method with $p = 2, s = 4$. In this case, we have the coefficient, say $\alpha_3$, of $z^3$ in the numerator of the stability function needing to be specified. In a similar way, the $E$-polynomial can be derived as

$$E(y) = (1 - \lambda iy)^4 (1 + \lambda iy)^4 - P(iy) P(-iy)$$

$$= (1 + \lambda^2 y^2)^4 - P(iy) P(-iy),$$

where

$$P(z) = 1 + (1 - 4\lambda)z + \left( \frac{1}{2} - 4\lambda + 6\lambda^2 \right) z^2 + \alpha_3 z^3.$$
CHAPTER 3. SINGLY–IMPLICIT METHODS

We find
\[ E(y) = \left( -\frac{1}{4} + 2\alpha_3 + (4 - 8\alpha_3)\lambda - 22\lambda^2 + 48\lambda^3 - 30\lambda^4 \right) y^4 \]
\[ + (-\alpha_3 + 4\lambda^6)y^6 + \lambda^8 y^8. \]

It can then be shown that
\[ E(y) \geq 0, \quad \forall y \geq 0 \]
\[ \Leftrightarrow 0.129945766237072504344 \leq \lambda \leq 3.284267796136022544 \]
and \[ -0.00573385343830948 \leq \alpha_3 \leq 0.00573385343830948. \]

In order to minimize the error constant and place the abscissae inside the integration interval as much as we can, we choose \( \lambda = 0.129945766237072504344 \) and \( \alpha_3 = 0.00573385343830948. \) The abscissae \( c_1, c_2 \) are given by \( c_i = \lambda \zeta_i, i = 1, 2. \) \( c_3 \) is a free parameter and \( c_4 = 1. \) In this example, we choose \( c_3 = \frac{\lambda}{\lambda_t} \) where \( \lambda_t \) is the eigenvalue for the case of \( p = 2, s = 3 \) (The same as in the existing DESI code, but it is not necessary). The coefficients in the first three rows of the coefficient matrix can be obtained by the \( C(2) \) condition. Furthermore, \( a_{41}, a_{42}, a_{43} \) must satisfy the \( C(2) \) condition and the stability requirement, that is
\[ 1 + x e_5^T A (I - z A)^{-1} e = \frac{P(z)}{(1 - \lambda z)^2}. \]

Hence, the second-order L-stable DESI method with 4 stages has the following corresponding Butcher tableau
\[
\begin{array}{c|ccc}
1 & a_{11} & a_{12} & 0 & 0 \\
c_2 & a_{21} & a_{22} & 0 & 0 \\
c_3 & a_{31} & a_{32} & \lambda & 0 \\
1 & a_{41} & a_{42} & a_{43} & \lambda \\
\end{array}
\]
(3.25)

where
\[
\begin{align*}
c_1 &= 0.07612046748871324, \\c_2 &= 0.4436625974595767,
\end{align*}
\]

56
The error constant $C_3$ is the coefficient of $z^3$ in $(1 - \lambda z)^4 \exp(z) - \alpha_3$, and hence we find (we discuss this later)

$$C_3 = \lambda^3 L'_4\left(\frac{1}{\lambda}\right) - \alpha_3 = -0.006420312248818325.$$

The related coefficients $1/\lambda$ and $(1/\lambda^i)\alpha_i$ of DESI formulae for order $p = 1, 2, \ldots, 8$ and $s = p + n, n = 1, 2, \ldots, 6$ are given in [34]. From the above example, we also note that for order 2 DESI methods the error constants for $s = 3$, $s = 4$ are respectively 0.012185 and 0.00642 approximately, while the error constant for order-2 SIRK method is 0.04044. Therefore, from the point of view of local error, we clearly have an advantage of adding some extra stages to the classical SIRK methods. Generally, for $s = p + 1$, the error constant $C_{p+1}$ for an order-$p$ DESI method can be found using

$$\frac{P(z)}{(1 - \lambda z)^{p+1}} = \exp(z) + C_{p+1}z^{p+1} + O(z^{p+2}),$$

where $\deg(P(z)) = p$. This implies

$$C_{p+1} = (-1)^{p+1}\lambda^{p+1}L_{p+1}\left(\frac{1}{\lambda}\right).$$

For $s \geq p + 2$, because the numerator of the stability function $P(z)$ has the term $\alpha_{p+1}z^{p+1}$, the error constant $C_{p+1}$ is the coefficient of $z^{p+1}$ in $(1 - \lambda z)^s \exp(z)$
CHAPTER 3. SINGLY-IMPLICIT METHODS

Table 3.4: Error constants of SIRK and DESI

<table>
<thead>
<tr>
<th>p</th>
<th>SIRK</th>
<th>s = p + 1</th>
<th>s = p + 2</th>
<th>s = p + 3</th>
<th>s = p + 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>4.4044 \times 10^{-2}</td>
<td>1.2185 \times 10^{-2}</td>
<td>6.4203 \times 10^{-3}</td>
<td>3.6824 \times 10^{-3}</td>
<td>2.5535 \times 10^{-3}</td>
</tr>
<tr>
<td>3</td>
<td>2.5897 \times 10^{-2}</td>
<td>3.7902 \times 10^{-4}</td>
<td>3.5969 \times 10^{-4}</td>
<td>4.6262 \times 10^{-6}</td>
<td>2.6417 \times 10^{-5}</td>
</tr>
<tr>
<td>4</td>
<td>1.1242 \times 10^{-3}</td>
<td>8.8406 \times 10^{-4}</td>
<td>1.6388 \times 10^{-4}</td>
<td>8.0272 \times 10^{-6}</td>
<td>4.1801 \times 10^{-5}</td>
</tr>
<tr>
<td>5</td>
<td>5.3005 \times 10^{-4}</td>
<td>4.5630 \times 10^{-5}</td>
<td>1.7491 \times 10^{-5}</td>
<td>2.4229 \times 10^{-6}</td>
<td>1.3785 \times 10^{-6}</td>
</tr>
<tr>
<td>6</td>
<td>1.5712 \times 10^{-5}</td>
<td>2.7750 \times 10^{-5}</td>
<td>2.7630 \times 10^{-7}</td>
<td>1.6085 \times 10^{-6}</td>
<td>8.5710 \times 10^{-7}</td>
</tr>
<tr>
<td>7</td>
<td>5.7932 \times 10^{-6}</td>
<td>1.1977 \times 10^{-6}</td>
<td>3.5152 \times 10^{-7}</td>
<td>8.3174 \times 10^{-8}</td>
<td>3.5883 \times 10^{-8}</td>
</tr>
<tr>
<td>8</td>
<td>1.3036 \times 10^{-7}</td>
<td>1.9774 \times 10^{-7}</td>
<td>4.8152 \times 10^{-8}</td>
<td>1.6909 \times 10^{-9}</td>
<td>2.7833 \times 10^{-9}</td>
</tr>
</tbody>
</table>

minus $\alpha_{p+1}$. By (3.12), we have

$$C_{p+1} = (-1)^s \lambda^{p+1} L_s^{(s-p-1)} \left( \frac{1}{\lambda} \right) - \alpha_{p+1}.$$

In general, as shown in Table 3.4, when the number of stages increases the error constant decreases. But there are some exceptions, for example, when $p = 3$, the error constant for $s = 6$ is smaller than for $s = 7$.

Although we can reduce the magnitude of the error constant by adding some extra stages to SIRK methods, it does not follow that DESI methods are more efficient than SIRK methods because we need more stages in one step for DESI.

One way to measure the efficiency is to use the length of the stepsize for one flop. Suppose $C_{p,s}$ denotes the error constant for an $s$-stage, order-$p$ DESI method ($s = p$ is the case of SIRK). If we use the standard stepsize prediction, then after integrating at $x_n$, the new stepsize $h_{new}$ is approximately given by

$$h_{new} = \left( \frac{tol}{|C_{p,s}| \|y^{(p+1)}(x_n)\|} \right)^{\frac{1}{p+1}},$$

where tol is the tolerance used by user. Suppose we use the modified Newton method to solve the nonlinear system for DESI and SIRK. For comparison
purposes, if we consider the solution of a linear stiff system with constant coefficients and assume the iteration matrix is updated at each integration step, then we should have only one iteration for convergence. Using an order-$p$, $s$-stage method to solve a problem with $N$ dimensions, the total flops per step are produced approximately by factorizations $(N^3/3)$, back substitutions $(sN^2)$, matrix transformations $(2p^2N)$, function evaluations $(N)$ and Jacobian evaluations $(N^2)$. Let $T$ denote the total flops in one step, the distance forward per flop, say $\tilde{h}_{\text{new}}$, is given by

$$\tilde{h}_{\text{new}} = \frac{h_{\text{new}}}{T}.$$ 

Therefore, $\tilde{h}_{\text{new}}$ is proportional to $1/T(C_{p,s})^{p+1}$. We see that the larger $\tilde{h}_{\text{new}}$ is, the more efficient the method is. For example, if we take $\tilde{h}_{\text{new}}$ for $s = 3, p = 2$ DESI and for $p = 2$ SIRK respectively, we find their ratio $r$ (DESI/SIRK) to be $r = 1.491$ which means DESI is more efficient than SIRK. In the case of $p = 2, s = p + 2$, the ratio goes to $r = 1.847$. A table containing these ratios for $p = 1, 2, \ldots, 8$ and $s = p + n, n = 1, 2, 3, 4$ is given in [34]. Generally, the more extra stages that are added, the more efficient the DESI is. But when $p = 6, 8$ and $s = p + 1$, the ratio is less than 1.

**Experiment 3.2** We now present some numerical results for solving the Kaps problem (3.9) by using an order-2 SIRK method and order-2 DESI methods with $s = 3, 4$ (mentioned in Example 3.5). In Table 3.5, the maximum global errors, average iterations, step numbers and total flops are given. The $s = 4, p = 2$ DESI method has the best performances in accuracy and efficiency because it has the smallest error constant. Both DESI methods are better than SIRK method. We also note that it is easier to have an asymptotic error estimator for DESI methods because of the extra stages. For SIRK methods, we modify the idea using $h^p y^{(p)}(x_n) - h^p y^{(p)}(x_{n-1})$ to estimate $h^{p+1} y^{(p+1)}(x_n)$ to allow for variable stepsize (we will discuss this further in section 4.7). All methods in this experiment use $Y_i = y_{n-1}$ as the starting value for the Newton iterations.

As we have seen the DESI methods have some advantages over the classical SIRK methods because of the additional diagonal stages. In practice, owing to the
Table 3.5: Numerical results for Kaps (3.9) using order-2 SIRK and DESI

<table>
<thead>
<tr>
<th>method</th>
<th>SIRK</th>
<th>$s = p + 1$</th>
<th>$s = p + 2$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>$tol = 10^{-4}$</td>
<td>$tol = 10^{-5}$</td>
</tr>
<tr>
<td>maxerr</td>
<td>$1.8552 \times 10^{-4}$</td>
<td>$9.8891 \times 10^{-5}$</td>
<td>$9.1282 \times 10^{-5}$</td>
</tr>
<tr>
<td>averit</td>
<td>2.6842</td>
<td>2.7262</td>
<td>2.6413</td>
</tr>
<tr>
<td>nstep</td>
<td>38</td>
<td>28</td>
<td>23</td>
</tr>
<tr>
<td>flops</td>
<td>62686</td>
<td>51785</td>
<td>46639</td>
</tr>
</tbody>
</table>
3.4. DESI METHODS

efficiency considerations and because we need to obtain reliable error estimates for the method itself and for the method of next highest order, \( s \geq p + 3 \) is suggested. In fact, the existing variable stepsize/variable order DESI stiff solver is based on \( s = p + 3 \) formulae [24], [29]. It has been shown that the code DESI is much more efficient than the SIRK code STRIDE and competitive with the BDF code LSODE [34]. Furthermore, DESI methods still retain high stage order and good stability. These advantages ensure that DESI methods are good candidates not only for solving stiff ordinary differential equations, but also differential algebraic equations of index-2 and partial differential equations ([19], [35]).

We also wish to remark that all the abscissae of DESI methods can be placed inside the integration interval. The difficulty for SIRK methods that have abscissae greater than 1 can be removed by using DESI formulae. However, in the existing DESI code, for \( p \geq 5 \), the case \( s = p + 3 \) is not sufficient to ensure that all the \( c_i \) are in \([0, 1]\). Hence, it is suspected that the numerical results for the existing DESI code can be improved by adjusting these abscissae. It is also known that for SIRK methods, we cannot have an L-stable method of order 7, but this quirk can be avoided as well.

The better performance in efficiency for DESI methods over standard SIRK methods is based on their smaller local truncation errors. However, these smaller error constants allow DESI methods to take large steps. This can be a serious handicap in problems with sharp solution flows or frequent discontinuities. Consequently, when we use the modified Newton method to solve the nonlinear system, a more conservative stopping criterion and a good predictor are necessary.
Chapter 4

The effective order of SIRK methods

In the previous chapter, we have seen that SIRK methods with $s \geq 3$ possess the undesirable property that some of their abscissae, because of the A-stability requirement, lie outside the integration interval. Instead of using the extra stages mentioned in section 3.4 to overcome this disadvantage, Butcher and Chartier ([20], [21] 1996) introduced the effective order SIRK methods. The introduced perturbation of the initial value offers several free parameters so that we are able to remove the restrictions on the abscissae. By adopting the more general definition of order, "effective order", these new methods allow more freedom in the choice of their parameters without losing the high stage order property. This chapter is devoted to the study of these methods.

The concept of effective order and its relative order conditions are discussed in the first section. In order to have a systematic approach to the construction of these methods, a special type of matrix is discussed in the second section. In addition to constructing these methods, the local error and stability of the method will be the content of the third section. Variable stepsize implementation is a difficulty for effective order methods. In section 4.4, we consider this difficulty, and give some examples to illustrate it. To examine whether there is any disadvantage when changing stepsize, we focus on a study of a systematic stepsize change.
CHAPTER 4. THE EFFECTIVE ORDER OF SIRK METHODS

pattern. In the study of local truncation error and stability, we will consider the performances of these methods using variable stepsize in section 4.5. Several ways of estimating error and some numerical results are given in the last two sections respectively.

4.1 Effective order

In order to overcome the famous "difficulty" shown in the following theorem, Butcher introduced "effective order" ([10] 1969).

**Theorem 4.1** When \( s \geq 5 \), no \( s \)-stage explicit Runge Kutta method can attain order \( s \).

This is the well-known "Butcher barrier". With the aid of this new interpretation of order, Butcher was able to construct 5-stage explicit Runge-Kutta methods which were of order 5. In that design, two different 5-stage explicit Runge-Kutta methods of order 3 are used to integrate the first step and the final step respectively, and a 5-stage explicit method of order 4 is used to integrate the remaining \( n - 2 \) steps (if total integration steps are \( n \)). It turns out that after this combination, the final output numerical solution will be of order 5. Because the order conditions for effective order methods involve the composition of two methods, we shall discuss the product rule for Runge-Kutta methods and the inversion of a Runge-Kutta method before we consider the main content of effective order.

Let \( \mathbf{a} = (A, b, c)_s \) and \( \mathbf{\bar{a}} = (\bar{A}, \bar{b}, \bar{c})_\bar{s} \) be two Runge-Kutta methods with \( s \) and \( \bar{s} \) stages respectively. If a numerical solution produced by \( \mathbf{a} \) is the input approximation for \( \mathbf{\bar{a}} \), then the internal stages for the two-step procedure, denoted by \( \mathbf{aa} \), can be expressed as

\[
Y_i = y_0 + h \sum_{j=1}^{s} a_{ij} f(Y_j), \quad i = 1, \ldots, s,
\]

\[
\overline{Y}_k = y_0 + h \sum_{i=1}^{s} b_i f(Y_i) + h \sum_{j=1}^{s} \bar{a}_{kj} f(\overline{Y}_j), \quad k = 1, \ldots, \bar{s},
\]
4.1. EFFECTIVE ORDER

\[ y_1 = y_0 + h \sum_{i=1}^{s} b_i f(Y_i) + h \sum_{j=1}^{s} \tilde{b}_j f(\tilde{Y}_j), \]

where \( Y_i, \tilde{Y}_j \) are the internal stage values for methods \( \mathbf{a}, \tilde{\mathbf{a}} \) respectively. Hence, the corresponding Butcher tableau for the composite method \( \mathbf{a}\tilde{\mathbf{a}} \) is

\[
\begin{array}{c|cc}
    c & A & 0 \\
    d + \tilde{c} & eb^T & \tilde{A} \\
    b^T & \tilde{b}^T
\end{array}
\]

where \( d = \sum_{i=1}^{s} b_i \). For the inverse of a Runge-Kutta method, suppose \( y_0 \) and \( y_1 \) are the input and output of a method \( \mathbf{a} \) respectively. If the initial value for a method is \( y_1 \) and the output value is \( y_0 \), then this method can be derived using (1.1) and (1.2). Therefore we have

\[
Y_i = y_1 + h \sum_{j=1}^{s} (a_{ij} - b_j) f(Y_j), \quad i = 1, \ldots, s,
\]

\[
y_0 = y_1 + h \sum_{i=1}^{s} (-b_i) f(Y_i),
\]

which has the tableau

\[
\begin{array}{c|cc}
    c - e & A - eb^T \\
    -b^T
\end{array}
\]

A method defined in this way will be denoted by \( \mathbf{a}^{-1} \) and it is easy to verify that the result obtained by using \( \mathbf{a}^{-1}\mathbf{a} \) or \( \mathbf{a}\mathbf{a}^{-1} \) is the starting value itself. Thus, if we use a sufficiently small stepsize, we have

\[
\mathbf{b}(\mathbf{a}\mathbf{a}^{-1}) = (\mathbf{a}\mathbf{a}^{-1})\mathbf{b} = \mathbf{b},
\]

where \( \mathbf{b} \) is an arbitrary Runge-Kutta method. According to [10], the definition of effective order for a Runge-Kutta method is given as follows:

**Definition 4.2** A method \( \mathbf{a} \) is of effective order \( s \) if there is a method \( \mathbf{d} \) such that \( \mathbf{d}\mathbf{a}\mathbf{d}^{-1} \) is of order \( s \).
CHAPTER 4. THE EFFECTIVE ORDER OF SIRK METHODS

From this definition, we can see the meaning of the classical order obtained by the method itself is generalized to that of the order obtained by combining three methods, one starting method d (or perturbation method), followed by the method a itself, then followed by the inverse of the starting method d\(^{-1}\). By introducing the starting method, one can have more free parameters to build up a 5-stage procedure of order 5, without having extra steps. This is because if we integrate a problem using the combination method dad\(^{-1}\), we have after \(n\) steps,

\[(d\text{ad}^{-1})^n = \text{da}^n\text{d}^{-1} = (\text{da})^{n-2}(\text{ad}^{-1}).\]

Let \(b = \text{da}\) and \(c = \text{ad}^{-1}\) be the starting and finishing methods respectively, then there are only \(n\) steps used in this integration. If the method \(a\) is of effective order \(s\), then ba\(^{n-2}\)c will be of order \(s\). The diagram of the integration procedure for effective order methods is shown in Figure 4.1. For \(p = 5\), three 5 stage methods \(b, a, c\) are given in [10] and [15]. In fact, they are not unique because of the extra free parameters.

![Figure 4.1: Integration procedure for effective order methods](image)

An alternative way of defining effective order is to use function mappings. When an order \(p\) method is used to integrate a differential equation \(y'(x) = f(y(x))\), the local error is

\[y(x_n) - y_n = O(h^{p+1}).\]

If we use the function \(\phi(y(x_{n-1}))\) to represent the numerical solution obtained using a method after the first step, then the above equation can be written as

\[y(x_{n-1} + h) - \phi(y(x_{n-1})) = O(h^{p+1}).\]
Similarly, if we assume that \( \psi \) and \( \phi \) are the mappings associated with the starting method \( d \) and method \( a \) of effective order \( p \) respectively, then from the definition of effective order, we have

\[
y(x_{n-1} + h) - \psi^{-1}(\phi(\psi(y(x_{n-1})))) = O(h^{p+1}).
\]

This is equivalent to the following definition for effective order.

**Definition 4.3** A method is said to have effective order \( p \) at \( x_n \), if it satisfies

\[
\psi(y(x_{n-1} + h)) - \phi(\psi(y(x_{n-1}))) = O(h^{p+1}),
\]

where \( \psi \) and \( \phi \) are the mappings associated with the starting method and the method itself.

![Figure 4.2: Effective order method \( \phi \) and the starting method \( \psi \)](image)

From this definition, one can treat the "exact solution" at \( x_n \) for an effective order method \( \phi \) as the perturbation using the starting method \( \psi \) at \( x_n \). This is illustrated in Figure 4.2, where the original input is \( y(x_{n-1}) \) and the numerical solution obtained by using method \( \psi \) follow by \( \phi \) is \( \phi(\psi(y(x_{n-1}))) \). The solution obtained using the mapping \( \pi \) (the so-called Picard integral) simply means the exact solution at \( x_{n-1} + h \) is \( y(x_n) \). Consequently, for effective order, the exact solution at \( x_n \) can be regarded as the perturbation after using the starting method \( \psi \) at \( y(x_n) \).
CHAPTER 4. THE EFFECTIVE ORDER OF SIRK METHODS

As we have seen, the effective order applied to explicit Runge-Kutta methods is to overcome the stage barrier for high-order methods. In other words, the number of order conditions for effective order methods can be reduced because of the new interpretation of order. In order to simplify the count of the order conditions, it is necessary to use the “Butcher Theorem”. We give an example below to demonstrate it before we state the main theorem.

Example 4.1 To find the order conditions for a Runge-Kutta method with a specified order, it is necessary to expand both the solution to the differential equation and the computed approximation by the Taylor series. We have the exact solution for (2.17) near an initial point \((x_{n-1}, y_{n-1})\) is given by the expansion

\[
y(x_{n-1} + h) = y(x_{n-1}) + hy'(x_{n-1}) + \frac{h^2}{2} y''(x_{n-1}) + \frac{h^3}{3!} y'''(x_{n-1}) + \frac{h^4}{4!} y^{(4)}(x_{n-1}) + O(h^5)
\]

\[
= y_{n-1} + hf + \frac{h^2}{2} f'(f) + \frac{h^3}{6} (f''(f, f) + f'(f'))
\]

\[
+ \frac{h^4}{24} (f'''(f, f, f) + 3f''(f, f') + f''(f')) + f'(f'(f'(f')))) + O(h^5),
\]

where \(f = f(y_{n-1}), f' = f'(y_{n-1}), f'' = f''(y_{n-1}),\) and \(f''' = f'''(y_{n-1}).\) In the above expansion, we can use a “rooted tree” to express the corresponding differential behaviour. This is the so-called elementary differential. A recursively generated rooted tree is defined as follows.

1. \(\tau\) is the tree with one vertex, and

2. \([t_1, t_2, \ldots, t_m]\) is the tree formed by joining the roots of \(t_i, i = 1, 2, \ldots, m,\) to a common root.

The elementary differentials, \(F(t)(y),\) are defined recursively by

\[
F(t)(y(x)) = \begin{cases} 
  f, & t = \tau, \\
  f^{(m)}(F(t_1), \ldots, F(t_m))(y(x)), & t = [t_1, t_2, \ldots, t_m].
\end{cases}
\]

68
As a consequence, from (4.1), for an expansion up to $h^4$, we have 8 trees related to the elementary differentials. We list these 8 trees and the way of deriving their corresponding elementary differentials in the following diagram.

Let

$$K_i = f(Y_i), \quad i = 1, 2, \ldots, s,$$

(1.1), (1.2) can be expressed by

$$Y_i = y_{n-1} + h \sum_{j=1}^{s} a_{ij} K_j, \quad i = 1, 2, \ldots, s,$$

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

We assume the various quantities above are functions of $h$ and expand $K_i$ with respect to $h = 0$. We have

$$y_n = y_{n-1} + h \sum_{i=1}^{s} b_i \left( K_i(0) + h K'_i(0) + \frac{h^2}{2} K''_i(0) + \frac{h^3}{3!} K'''_i(0) \right) + O(h^5).$$
CHAPTER 4. THE EFFECTIVE ORDER OF SIRK METHODS

where

\[
\begin{align*}
K_0(0) &= f, \\
K_0'(0) &= c_i f'(f), \\
K_1''(0) &= c_i^2 f''(f, f) + 2(Ac_i) f'(f'), \\
K_1'''(0) &= c_i^3 f'''(f, f, f) + 6(cAc_i) f''(f, f') + 3(Ac^2_i) f'(f''(f, f)) \\
&\quad + 6(A^2 c_i) f'(f'(f'))).
\end{align*}
\]

and this yields

\[
y_n = y_{n-1} + h(b^T e) f + b^T \left((3b^T c^2 f''(f, f) + (6b^T Ac) f'(f'))\right) \\
+ b^T \left((4b^T c^3 f'''(f, f, f) + 3(8b^T cAc) f''(f, f')) + (12b^T Ac^2) f'(f''(f, f))\right) \\
+ (24b^T A^2 c) f'(f'(f')) + O(h^5). 
\]

After comparing the terms in (4.1) and (4.2), we have the method is of order 4 if and only if

\[
b^T e = 1, \quad b^T c = \frac{1}{2}, \quad b^T c^2 = \frac{1}{3}, \quad b^T Ac = \frac{1}{6}, \\
b^T c^3 = \frac{1}{4}, \quad b^T cAc = \frac{1}{8}, \quad b^T Ac^2 = \frac{1}{12}, \quad b^T A^2 c = \frac{1}{24}.
\]

Therefore we have 8 order conditions for constructing 4-th order methods. When the order is higher, the corresponding number of rooted trees is increasingly greater. For example, we have 17 and 37 order conditions for order 5 and 6 respectively.

We can derive the order conditions for Runge-Kutta methods by the study of rooted trees. As a matter of fact, the relations in (4.3) have been studied by Butcher systematically ([7] 1963). If \( T \) denotes the set of rooted trees and \( \rho(t) \) the number of vertices of \( t \), then for any \( t \in T, \rho(t) \leq p \), the "elementary weight" \( \alpha(t) \) depends only on the method and is defined by \( \alpha(t) = b^T \Phi(t) \) with \( \Phi \) defined recursively by

\[
\Phi(t) = \begin{cases} 
\epsilon, & t = \tau, \\
(A\Phi(t_1)) \ast (A\Phi(t_2)) \ast \cdots \ast (A\Phi(t_m)), & t = [t_1, t_2, \ldots, t_m].
\end{cases}
\]

70
where * denotes the component-by-component product. "γ(t)" is the "density" of t and is defined recursively by

\[ γ(t) = \begin{cases} 
1, & t = τ, \\
n(t)γ(t_1)γ(t_2) \cdots γ(t_m), & t = [t_1, t_2, \ldots, t_m].
\end{cases} \]

Furthermore, if we define the "symmetry" of t by \( σ(τ) = 1 \) and \( σ(t) = m_1!m_2! \cdots m_k! \), where \( m_1 \) of \( t_1, t_2, \ldots, t_m \) are identical of one kind, \( m_2 \) of \( t_1, t_2, \ldots, t_m \) are identical of second kind, \ldots, and \( m_k \) are identical of the \( k \)-th kind, then we have the Taylor series for the exact solution is given by

\[ y(x_0 + h) = y(x_0) + \sum_{t \in T} h^ρ(t) \frac{1}{σ(t)γ(t)} F(t)(y_0), \tag{4.4} \]

and the corresponding Taylor series for the computed result is

\[ y_1 = y(x_0) + \sum_{t \in T} h^ρ(t) \frac{α(t)}{σ(t)} F(t)(y_0). \tag{4.5} \]

Comparing the two series (4.4) and (4.5), it is easy to find a corresponding order condition for each \( t \). For the case of order 4, these will be

\[ α(t_i) = \frac{1}{γ(t_i)}, \quad i = 1, 2, \ldots, 8, \]

where \( α(t_i) \), for all \( i = 1, 2, \ldots, 8 \), are given as the same as the left hand side of the equations in (4.3), and the \( γ(t_i), i = 1, 2, \ldots, 8 \), are the denominators of the right hand side of the equations in (4.3). As a matter of fact, for a given tree \( t_i \), \( ρ(t) \leq p \), the elementary weight \( α(t_i) \) and the density \( γ(t_i) \) can be found by the following rules.

1. For finding the elementary weight \( α(t_i) \):
   (1) The root of \( t_i \) stands for \( b^T \).
   (2) The terminal vertices stand for \( c \).
   (3) The other vertices stand for \( A \).

2. For finding the density \( γ(t_i) \);
CHAPTER 4. THE EFFECTIVE ORDER OF SIRK METHODS

(1) The number attached to the terminal vertices is 1.

(2) The number attached to the non-terminal vertices is the number of vertices of the outward branches growing from this vertex plus 1 (the vertex itself).

(3) The density is obtained by multiplying the numbers attached to all vertices.

For example, using these rules, the $\alpha(t)$ and $\gamma(t)$ of the tree $t$ can be given as follows.

\[
\alpha(t) = b^T ((Ac)(Ac)), \quad \gamma(t) = 1 \times 2 \times 2 \times 4 \times 7 = 112,
\]

Order condition: $b^T ((Ac)(Ac)) = \frac{1}{112}$.

Table 4.1 shows the corresponding elementary differential, density and the elementary weight for each tree $t$, $\rho(t) \leq 4$.

The "order" of a Runge-Kutta method is given in the following theorem.

**Theorem 4.4** A Runge-Kutta method is of order $p$ if and only if

\[
\alpha(t) = \frac{1}{\gamma(t)},
\]

for all $t$ satisfying $\rho(t) \leq p$.

Referring to [15], [27] and [44], the weight functions of two methods can be used to derive the elementary weight function of their combination. This is based on the following theorem.
### 4.1. EFFECTIVE ORDER

Table 4.1: $F(t)(y), \gamma(t), \alpha(t)$ for $\rho(t) \leq 4$

<table>
<thead>
<tr>
<th>tree</th>
<th>$t$</th>
<th>order</th>
<th>$F(t)(y)$</th>
<th>$\gamma(t)$</th>
<th>$\alpha(t)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$t_1$</td>
<td>$\tau$</td>
<td>1</td>
<td>$f$</td>
<td>1</td>
<td>$b^T e$</td>
</tr>
<tr>
<td>$t_2$</td>
<td>$[\tau]$</td>
<td>2</td>
<td>$f'(f)$</td>
<td>2</td>
<td>$b^T c$</td>
</tr>
<tr>
<td>$t_3$</td>
<td>$[\tau^2]$</td>
<td>3</td>
<td>$f''(f, f)$</td>
<td>3</td>
<td>$b^T c^2$</td>
</tr>
<tr>
<td>$t_4$</td>
<td>$[[\tau]]$</td>
<td>3</td>
<td>$f'(f'(f))$</td>
<td>6</td>
<td>$b^T A c$</td>
</tr>
<tr>
<td>$t_5$</td>
<td>$[\tau^3]$</td>
<td>4</td>
<td>$f''(f, f, f)$</td>
<td>4</td>
<td>$b^T c^3$</td>
</tr>
<tr>
<td>$t_6$</td>
<td>$[\tau[\tau]]$</td>
<td>4</td>
<td>$f''(f, f'(f))$</td>
<td>8</td>
<td>$b^T c A c$</td>
</tr>
<tr>
<td>$t_7$</td>
<td>$[[\tau^2]]$</td>
<td>4</td>
<td>$f'(f''(f, f))$</td>
<td>12</td>
<td>$b^T A c^2$</td>
</tr>
<tr>
<td>$t_8$</td>
<td>$[[[\tau]]]$</td>
<td>4</td>
<td>$f'(f'(f'(f)))$</td>
<td>24</td>
<td>$b^T A^2 c$</td>
</tr>
</tbody>
</table>

**Theorem 4.5** The weight function $(\alpha \beta)(t)$ for a composition $a \bar{a}$ of two methods $a, \bar{a}$ is given by

$$(\alpha \beta)(t) = \alpha(t) + \sum_{u \subset t} \beta(u)\alpha(t - u) + \beta(t),$$

where $\alpha(t), \beta(t)$ are the weights of $a, \bar{a}$ respectively, and $u$ is the subtree of $t$ ($u$ shares the same root as $t$) and $t - u$ is a tree formed from $t$ by taking off its subtree $u$.

An example to illustrate the above theorem is given as follows.

**Example 4.2** For methods of order 4, we denote $\alpha_i = \alpha(t_i), \beta_i = \beta(t_i), i = 1, 2, \ldots, 8$, then the weight $(\alpha \beta)(t)$ for the 8 trees in example 4.1 can be expressed in terms of $\alpha(t), \beta(t)$ as follows

$$(\alpha \beta)(t_1) = \alpha_1 + \beta_1,$$

$$(\alpha \beta)(t_2) = \alpha_2 + \beta_1 \alpha_1 + \beta_2,$$

$$(\alpha \beta)(t_3) = \alpha_3 + \beta_1 \alpha_1^2 + 2\beta_2 \alpha_1 + \beta_3,$$

73
CHAPTER 4. THE EFFECTIVE ORDER OF SIRK METHODS

\[(\alpha\beta)(t_4) = \alpha_4 + \beta_1\alpha_2 + \beta_2\alpha_1 + \beta_4,\]
\[(\alpha\beta)(t_5) = \alpha_5 + \beta_1\alpha_1^3 + 3\beta_2\alpha_1^2 + 3\beta_3\alpha_1 + \beta_5,\]
\[(\alpha\beta)(t_6) = \alpha_6 + \beta_1\alpha_1\alpha_2 + \beta_2(\alpha_1^2 + \alpha_2) + 3\beta_4\alpha_1 + \beta_6,\]
\[(\alpha\beta)(t_7) = \alpha_7 + \beta_1\alpha_3 + \beta_2\alpha_1^2 + 2\beta_4\alpha_1 + \beta_7,\]
\[(\alpha\beta)(t_8) = \alpha_8 + \beta_1\alpha_4 + \beta_2\alpha_2 + \beta_4\alpha_1 + \beta_8.\]

\[\square\]

Suppose \(E\) is the weight function of a method of infinite order, that is, \(E(t)\) satisfies
\[E(t) = \frac{1}{\gamma(t)}, \quad \forall t \in T.\]

On the other hand, using Theorems 4.4, 4.5 and Definition 4.3, we can use the weight function of Runge-Kutta methods to obtain an alternative effective order definition.

**Definition 4.6** Let \(\alpha(t), \beta(t)\) be the weights of a starting method and the method \(a\) respectively. If \(a\) satisfies
\[(\alpha\beta)(t) = E\alpha(t), \quad \rho(t) \leq p, \quad (4.6)\]
then \(a\) is a method of effective order \(p\).

Therefore, by (4.6), for given \(p\), the order conditions of a method with effective order \(p\) can be derived easily. We give an example to show how to derive the order conditions for Runge-Kutta methods with effective order 4.

**Example 4.3** Suppose \(\alpha(t), \beta(t)\) are the weights of a starting method and the effective order method \(a\) respectively. We want to examine (4.6) for \(t = t_1, t_2, \ldots, t_8\). By Example 4.2, we can derive those conditions for \(\beta_i, \quad i = 1, 2, \ldots, 8\). First, for \(t_1\), we have
\[(\alpha\beta)(t_1) = E\alpha(t_1),\]
\[\Rightarrow \alpha_1 + \beta_1 = E(t_1) + \alpha_1,\]
\[\Rightarrow \beta_1 = E(t_1) = 1.\]
4.1. EFFECTIVE ORDER

and for \( t_2 \),

\[
(\alpha \beta)(t_2) = E\alpha(t_2),
\]

\[
\Rightarrow \alpha_2 + \beta_1 \alpha_1 + \beta_2 = E(t_2) + \alpha_1 E(t_1) + \alpha_2,
\]

\[
\Rightarrow \beta_1 \alpha_1 + \beta_2 = \frac{1}{2} + \alpha_1,
\]

\[
\Rightarrow \beta_2 = \frac{1}{2}.
\]

In a similar way, we can derive

\[
\beta_4 = \frac{1}{6}, \quad \beta_8 = \frac{1}{24},
\]

and \( \beta_3, \beta_5, \beta_6, \beta_7 \) satisfy the condition

\[
\beta_3 - \beta_5 + 2\beta_6 - \beta_7 = \frac{1}{4}.
\]

We note that the number of order conditions for a method with effective order 4 is now \( 5 \). If we compare it with the number of the classical order conditions, \( 8 \), this generalized order definition gives more freedom in choosing a method and also explains the reason why Butcher can construct 5-stage explicit methods which can attain order 5 by the aid of starting and finishing schemes. We also note that the order conditions are the same as the classical ones when \( t \) is the so-called "tall tree". That is \( t = t_1, t_2, t_4, t_8 \) in this example. In fact, the number of conditions for a Runge-Kutta method to have effective order \( s \) has been studied by Butcher and Sanz-Serna ([27] 1990). They show that if \( v_s \) denotes the number of rooted trees with \( s \) vertices, then the number of order conditions for effective order \( s \) will be \( v_s + 1 \). Table 4.2 gives \( v_s, e_s \) with \( s \) up to 10. Here \( o_s \) is the number of the classical order conditions. One can see the number of order conditions for methods with classical order 5 is 17, but the number of coefficients in an explicit method with 5 stages is 15 (assume methods satisfy \( C(1) \)). However, the number of effective order conditions for a method of order 5 is only 10 (There are free parameters when we want to determine the coefficients for methods). Once the method is found, the starting and finishing methods can be determined using (4.6). Recall that \( \text{dad}^{-1} \) is a method of order 5, the starting method is \( b = da = Ed \) (\( E \) is the method of infinite order), and the finishing method is
CHAPTER 4. THE EFFECTIVE ORDER OF SIRK METHODS

Table 4.2: Number of order conditions for effective order $\geq s$

<table>
<thead>
<tr>
<th>$s$</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>$u_s$</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>4</td>
<td>9</td>
<td>20</td>
<td>48</td>
<td>115</td>
<td>286</td>
<td>719</td>
</tr>
<tr>
<td>$e_s$</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>5</td>
<td>10</td>
<td>21</td>
<td>49</td>
<td>116</td>
<td>287</td>
<td>720</td>
</tr>
<tr>
<td>$o_s$</td>
<td>1</td>
<td>2</td>
<td>4</td>
<td>8</td>
<td>17</td>
<td>37</td>
<td>85</td>
<td>200</td>
<td>486</td>
<td>1205</td>
</tr>
</tbody>
</table>

$c = ad^{-1} = d^{-1}E$. Using (4.6), the weight functions of $b, c$ for $\rho(t) \leq 5$ can be derived. If the weight of $d$ for $t_1$ is assumed to be 0, then $b$ and $c$ retain 5 stages. Hence, it will not have any extra steps.

4.2 Effective order conditions for SIRK methods

For explicit Runge Kutta methods, the application of effective order reduces the number of order conditions and makes it possible to break the Butcher barrier. In this section, we will discuss another approach which is to use this idea to overcome one of the difficulties of singly-implicit Runge Kutta methods. Recalling Theorem 3.2, because of the stability requirement and the fact that SIRK methods are collocation methods, the choices of abscissae of the methods are restricted to be proportional to the roots of the Laguerre polynomial. It turns out that when $s \geq 3$, some of the components of $c$ are greater than 1. In order to retain high stage order, a way of overcoming this difficulty is to introduce more free parameters. The concept of effective order meets this requirement.

Before applying effective order to SIRK methods, it is natural to ask what kind of conditions the new method should satisfy? First, the new methods should have the single-implicitness property for computational savings. Secondly, the new methods must have good stability for solving stiff problems. Finally, they should have high stage order so as to prevent order reduction for some special problems.
Due to the above requirements, we aim to construct these new methods to satisfy the following aims:

(1) new methods are generalizations of classical SIRK methods;
(2) they have the same stability properties as SIRK methods;
(3) they have stage order equal to the stage number.

Because a stage order equal to \( s \) is proposed for \( s \)-stage methods, the coefficient matrix \( A \) can be determined by the stage-order conditions. That is

\[
\sum_{j=1}^{s} a_{ij}c_{j}^{k-1} = \frac{1}{k} c_{i}^{k}, \quad k = 1, 2, \ldots, s \tag{4.7}
\]

But for single-implicitness,

\[
\det(zI - A) = (z - \lambda)^s
\]

and consequently, the \( c_{i} \) are determined uniquely once the eigenvalue \( \lambda \) is given. This is the reason why the classical SIRK methods have the restriction on their abscissae. We recall the effective order Definition 4.3: assume that \( \psi \) and \( \phi \) represent the mappings associated with the starting method and the method itself. If the method satisfies

\[
\psi(y(x_{n-1} + h)) - \phi(\psi(y(x_{n-1}))) = O(h^{s+1}),
\]

then we say the method has effective order \( s \). By using this generalized concept of order, we will be able to introduce more free parameters from the starting method \( \psi \). In other words, the input value \( y(x_{n-1}) \) at the \( n \)-th step can now be replaced by \( \psi(y(x_{n-1})) \). This can be done easily because all we need to do is to add the starting procedure and ensure the numerical solution obtained after this step is \( \psi(y(x_{n-1} + h)) + O(h^{s+1}) \). The question now is what kind of input value \( \psi(y(x_{n-1})) \) should take? For Runge-Kutta methods, the calculations for the internal stages \( Y_{i} \) are given by

\[
Y_{i} = y(x_{n-1}) + h \sum_{j=1}^{s} a_{ij}f(Y_{j}), \quad i = 1, 2, \ldots, s. \tag{4.8}
\]
CHAPTER 4. THE EFFECTIVE ORDER OF SIRK METHODS

For each \( i \), using Taylor series to expand the right hand side of (4.8) at \( x_{n-1} \) and comparing it with the Taylor series of the exact solution \( y(x_{n-1} + c_i h) \) yields the stage-order conditions (4.7). Because we need the free parameters to appear in (4.7), it is natural and convenient to assume \( \psi(y(x_{n-1})) \) has the value

\[
y(x_{n-1}) + \alpha_1 y'(x_{n-1}) + \alpha_2 h^2 y''(x_{n-1}) + \cdots + \alpha_s h^s y^{(s)}(x_{n-1}).
\]  

(4.9)

Hence, \( s \) free parameters are introduced. The abscissae \( c_i \) can be chosen as we wish as long as they are distinct. Clearly, from the point of view of effective order, (4.9) can be treated as the “exact solution” for SIRK methods with effective order \( s \). This plays an important role in the study of effective order of SIRK methods [20]. For convenience, we give the following notation. Let

\[
B_{s,h}(\alpha, y(x_{n-1})) = y(x_{n-1}) + \alpha_1 y'(x_{n-1}) + \alpha_2 h^2 y''(x_{n-1}) + \cdots + \alpha_s h^s y^{(s)}(x_{n-1}),
\]

where \( y(x_{n-1}) \) is the exact solution at the \( n \)-th step and \( \alpha = [\alpha_1, \alpha_2, \ldots, \alpha_s]^T \in \mathbb{R}^s \). \( B_{s,h}(\alpha, y(x_{n-1})) \) will be called the “Butchered solution”. The definition for a SIRK method which has effective order \( s \) can now be given.

**Definition 4.7** An \( s \)-stage SIRK method is said to have effective order \( s \) at \( x_{n-1} \) if it satisfies

\[
B_{s,h}(\alpha, y(x_n)) - \phi(B_{s,h}(\alpha, y(x_{n-1}))) = O(h^{s+1}),
\]

(4.10)

where \( \phi \) is the associated mapping of the method.

Now, we consider the derivation of order conditions for the new methods. The methods we proposed here have stage order equal to their order. Hence each internal stage value \( Y_i \) in an \( s \) stage method satisfies the \( C(s) \) condition

\[
Y_i = y(x_{n-1} + c_i h) + O(h^{s+1}), \quad i = 1, 2, \ldots, s.
\]

(4.11)

This leads to

\[
hf(Y_i) = hy'(x_{n-1} + c_i h) + O(h^{s+2}), \quad i = 1, 2, \ldots, s.
\]

(4.12)
It is convenient to write the modified Vandermonde matrix \( W \), the Nordsieck vector \( \zeta^{[n-1]} \), and the vector of stage derivatives \( F \) as follow:

\[
W = \begin{bmatrix} 1 & c_1 & \frac{c_1^2}{2} & \cdots & \frac{c_1^{s-1}}{(s-1)!} \\ 1 & c_2 & \frac{c_2^2}{2} & \cdots & \frac{c_2^{s-1}}{(s-1)!} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & c_s & \frac{c_s^2}{2} & \cdots & \frac{c_s^{s-1}}{(s-1)!} \end{bmatrix}, \quad \zeta^{[n-1]} = \begin{bmatrix} h y'(x_{n-1}) \\ h^2 y''(x_{n-1}) \\ \vdots \\ h^s y^{(s)}(x_{n-1}) \end{bmatrix}, \quad h F = \begin{bmatrix} h f(Y_1) \\ h f(Y_2) \\ \vdots \\ h f(Y_s) \end{bmatrix}.
\]

Let \( N \) denote the dimension of the system. Using the Taylor series on the right hand side of (4.12) at \( x_{n-1} \), we have

\[
h F = (W \otimes I_N) \zeta^{[n-1]} + O(h^{s+1}). \tag{4.13}
\]

By using a Taylor series, it is also easy to see that the Butchered solution can be expressed as

\[
B_{s,h}(\alpha, y(x_{n-1})) = y(x_{n-1}) + (\alpha^T \otimes I_N) \zeta^{[n-1]}, \tag{4.14}
\]

and

\[
y(x_{n-1} + h) = y(x_{n-1}) + (\tau^T \otimes I_N) \zeta^{[n-1]} + O(h^{s+1}), \tag{4.15}
\]

where \( \tau^T = [1, \frac{1}{2}, \ldots, \frac{1}{s!}] \). The Nordsieck vector at \( x_{n-1} + h \) can be written in terms of the Nordsieck vector at \( x_{n-1} \)

\[
\zeta^{[n]} = (E \otimes I_N) \zeta^{[n-1]} + O(h^{s+1}), \tag{4.16}
\]

where

\[
E = \begin{bmatrix} 1 & 1 & \frac{1}{2!} & \cdots & \frac{1}{(s-1)!} \\ 0 & 1 & 1 & \cdots & \frac{1}{(s-2)!} \\ 0 & 0 & 1 & \cdots & \frac{1}{(s-3)!} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 1 \end{bmatrix}.
\]

For the new method, the input value at \( x_{n-1} \) is now taken to be the Butchered solution \( B_{s,h}(\alpha, y(x_{n-1})) \). From (4.11), (4.12), the following theorem can be derived.
CHAPTER 4. THE EFFECTIVE ORDER OF SIRK METHODS

**Theorem 4.8** If the Butchered solution for an \( s \)-stage SIRK method with effective order \( s \) at \( x_{n-1} \) is \( B_{s,h}(\alpha, y(x_{n-1})) \), then the coefficient matrix \( A = [a_{ij}] \) and the abscissae \( c_i, i = 1, 2, \ldots, s \), of the method satisfy

\[
\sum_{j=1}^{s} a_{ij} \frac{c_{j}^{k-1}}{(k-1)!} + \alpha_k = \frac{c_{i}^{k}}{k!}, \quad i = 1, 2, \ldots, s; \quad k = 1, 2, \ldots, s, \tag{4.17}
\]

or, equivalently, in matrix form

\[
A \frac{c_{i}^{k-1}}{(k-1)!} + \alpha_k e = \frac{c_{i}^{k}}{k!}, \quad k = 1, 2, \ldots, s. \tag{4.18}
\]

**Proof:** From the calculation of the internal stages for a Runge-Kutta method and using the Butchered solution \( B_{s,h}(\alpha, y(x_{n-1})) \) for the input value, we have

\[
Y_i = B_{s,h}(\alpha, y(x_{n-1})) + h \sum_{j=1}^{s} a_{ij} f(Y_j), \quad i = 1, 2, \ldots, s.
\]

By (4.11), (4.12) and the use of the Taylor series, for each \( i = 1, 2, \ldots, s \), the left hand side of the above equations can be written as

\[
Y_i = y(x_{n-1} + c_i h) + O(h^{s+1}) = y(x_{n-1}) + \sum_{k=1}^{s} \frac{c_{i}^{k}}{k!} h^k y^{(k)}(x_{n-1}) + O(h^{s+1}), \tag{4.19}
\]

and the right hand side can be written as

\[
B_{s,h}(\alpha, y(x_{n-1})) + h \sum_{j=1}^{s} a_{ij} y'(x_{n-1} + c_j h) + O(h^{s+2})
\]

\[
= y(x_{n-1}) + \sum_{k=1}^{s} \alpha_k h^k y^{(k)}(x_{n-1}) + \sum_{k=1}^{s} \frac{c_{i}^{k-1}}{(k-1)!} h^k y^{(k)}(x_{n-1}) + O(h^{s+1}), \tag{4.20}
\]

respectively. Equating coefficients of the terms \( h^k y^{(k)}(x_{n-1}), k = 1, 2, \ldots, s \) in (4.19), (4.20) yields (4.17).

The stage values \( Y_i \) obtained in this way are approximations to \( y(x_{n-1} + c_i h) \) up to \( O(h^{s+1}) \). Furthermore, we want the new methods to have the same stability properties as SIRK methods. Hence, the single eigenvalue \( \lambda \) of the \( A \) matrix of an \( s \)-stage method is chosen to be the same as the one for the \( s \)-stage SIRK method (see Table 3.3). In addition to having the \( s^2 \) conditions in (4.17), the
single-implicitness gives another $s$ conditions. Consequently, once the distinct abscessae $c_1, c_2, \ldots, c_s$ are chosen, the coefficient matrix $A$ of the new method will be determined uniquely. Because the classical SIRK methods are collocation methods, it is easy to see that $\alpha_k = 0$, for all $k = 1, 2, \ldots, s$, if the abscessae are chosen to be the same as for SIRK methods. Therefore, we have the following corollary.

**Corollary 4.9** If the abscessae $c_1, c_2, \ldots, c_s$ for an $s$-stage SIRK method with effective order $s$ are chosen to be the same as the associated order-$s$ SIRK method, then the Butchered solution $B_{s,h}(\alpha, y(x_{n-1}))$ satisfies

$$B_{s,h}(\alpha, y(x_{n-1})) = y(x_{n-1}).$$

**Proof:** By Theorem 3.2 in chapter 3, the coefficient matrix $A$ of an order-$s$ SIRK method satisfies

$$\sum_{j=1}^{s} a_{ij} \frac{c_j^{k-1}}{(k-1)!} = \frac{c_i^k}{k!}, \quad i = 1, 2, \ldots, s; \quad k = 1, 2, \ldots, s.$$

By (4.17), $\alpha^T$ is a zero vector. From (4.14), the result follows. \(\square\)

The next aim should be the derivation of the weight vector $b^T$ for the new methods. From Definition 4.7, when the input value at $x_{n-1}$ is the Butchered solution $B_{s,h}(\alpha, y(x_{n-1}))$, after this step, the output value proposed is an approximation to $B_{s,h}(\alpha, y(x_{n-1} + h))$. Therefore, $b^T$ can be determined by the following theorem.

**Theorem 4.10** If the Butchered solution of an $s$-stage SIRK method with effective order $s$ at $x_{n-1}$ is $B_{s,h}(\alpha, y(x_{n-1}))$ and $\alpha_0 = 1$, then the weights $b^T = [b_1, b_2, \ldots, b_s]$ and the abscessae $c_i, i = 1, 2, \ldots, s$, of the method satisfy

$$\sum_{i=1}^{s} b_i \frac{c_i^{k-1}}{(k-1)!} = \sum_{j=1}^{k} \frac{1}{(k-j+1)!} \alpha_{j-1}, \quad k = 1, 2, \ldots, s. \quad (4.21)$$

or, in matrix form,

$$b^T W = \tau^T + \alpha^T (E - I_s). \quad (4.22)$$
**CHAPTER 4. THE EFFECTIVE ORDER OF SIRK METHODS**

**Proof:** Without loss of generality, we assume the dimension of the system is one. From the definition of the effective order of SIRK methods, the output value at \( x_n = x_{n-1} + h \) is an approximation to the Butchered solution \( B_{s,h}(\alpha, y(x_{n-1} + h)) \). Hence, we have

\[
B_{s,h}(\alpha, y(x_{n-1} + h)) = B_{s,h}(\alpha, y(x_{n-1})) + h \sum_{i=1}^{s} b_i f(Y_i) + O(h^{s+1}).
\]

From (4.14), we find

\[
B_{s,h}(\alpha, y(x_{n-1} + h)) = y(x_{n-1} + h) + \alpha^T \zeta^{[n]} \tag{4.23}
= y(x_{n-1}) + \alpha^T \zeta^{[n-1]} + h \sum_{i=1}^{s} b_i f(Y_i) + O(h^{s+1}).
\]

By (4.15), (4.16), the left hand side of (4.23) can be written as

\[
y(x_{n-1}) + \tau^T \zeta^{[n-1]} + \alpha^T E \zeta^{[n-1]} + O(h^{s+1}).
\]

If \( b^T = [b_1, b_2, \ldots, b_s] \), the right hand side of (4.23) can be written as

\[
y(x_{n-1}) + \alpha^T \zeta^{[n-1]} + b^T h F + O(h^{s+1}) =
= y(x_{n-1}) + \alpha^T \zeta^{[n-1]} + b^T W \zeta^{[n-1]} + O(h^{s+1}) \tag{4.13},
\]

Equating the two equations, we have

\[
\tau^T + \alpha^T (E - I_s) = b^T W.
\]

In order to obtain the modified initial value for the new method, we need a perturbed initial value for the differential equation in the first step and to undo the perturbation in the last step (cf. Figure 4.1). In other words, we have to find the starting and finishing methods for the whole integration procedure. For the finishing methods, the new methods have an advantage over effective order explicit methods. For the \( s \)-stage SIRK methods with effective order \( s \), the stage order is also \( s \). We can then take any stage value \( Y_i \) to be the output value as long as \( c_i = 1 \) for some \( i \). Consequently, without any finishing procedure and any extra computational costs, the numerical solution with classical order \( s \) can
be obtained easily. For a starting method, the perturbations $\alpha_i$ will be equal to 0 if the abscissae $c_i$ are chosen to be the same as for the SIRK methods. Therefore, SIRK methods are special cases of the corresponding effective order methods. This persuades us to use SIRK methods to produce the initial value at the first step. Because the initial value at $x_{n-1}$ is the Butchered solution $B_{s,h}(\alpha, y(x_{n-1}))$, the weight vector $\bar{b}^T$ of the SIRK method should be modified to fulfill this requirement.

**Theorem 4.11** If the starting method for an $s$-stage SIRK method with effective order $s$ is the $s$-stage SIRK method with modified weight vector $\bar{b}^T$, and the stepsize used at the first step is the same as for the SIRK method, then $\bar{b}^T$ and the abscissae $c_i, i = 1, 2, \ldots, s$, of the starting SIRK method satisfy

$$\bar{b}^T W = \tau^T + \alpha^T E. \tag{4.24}$$

**Proof:** Without loss of generality, we also assume the dimension of the system is one. If the initial value for the system we are solving is $y(x_0)$, then we use the starting method to produce an approximation to the Butchered solution $B_{s,h}(\alpha, y(x_0 + h))$ at $x_1 = x_0 + h$. Therefore,

$$B_{s,h}(\alpha, y(x_0 + h)) = y(x_0) + h \sum_{i=1}^{s} \bar{b}_i f(Y_i) + O(h^{s+1}).$$

From (4.14), we have

$$B_{s,h}(\alpha, y(x_0 + h)) = y(x_0 + h) + \alpha^T \zeta^{[1]} \tag{4.25}$$

$$= y(x_0) + h \sum_{i=1}^{s} \bar{b}_i f(Y_i) + O(h^{s+1}).$$

The left hand side of (4.25) can be written as

$$y(x_0) + \tau^T \zeta^{[0]} + \alpha^T E \zeta^{[0]} + O(h^{s+1}), \quad \text{by} \quad (4.15), (4.16)$$

If $\bar{b}^T = [\bar{b}_1, \bar{b}_2, \ldots, \bar{b}_s]$, by (4.13), the right hand side of (4.25) can be written as

$$y(x_0) + \bar{b}^T h F + O(h^{s+1}) = y(x_0) + \bar{b}^T W \zeta^{[0]} + O(h^{s+1}).$$
Equating these two equations, we have
\[ r^T + \alpha^T E = b^T W. \]

In Theorem 4.8, the generalized stage-order conditions have been derived. In Theorem 4.10, 4.11, for the effective order requirement, the modified order conditions are given for the new methods and the starting methods respectively. We are now in a position to give an example for constructing a starting method as well as an effective order method by adopting these results.

**Example 4.4** We give an example for \( s = 3 \). Let the effective order method be denoted by \((A, b, c)_3\) and let \( c_1, c_2 \) and \( c_3 \) be distinct. Our purpose is to find \( A = [a_{ij}]_{3 \times 3}, b^T = [b_1, b_2, b_3], \) and \( c = [c_1, c_2, c_3]^T \). We divide the discussion of details in three parts.

(1) **Order conditions**

From the generalized stage-order conditions (4.18), we have the following equations

\[ Ae + \alpha_1 = c, \quad Ac + \alpha_2 = \frac{c^2}{2}, \quad A_{2!}^2 + \alpha_3 = \frac{c^3}{3!}. \]

Furthermore, we have to consider the stability of this method. At least, for the case of constant stepsize, we aim to build up the new method which has the same stability property as SIRK methods. This means that if we choose the eigenvalue \( \lambda \) from Table 3.3 for \( s = 3 \) (cf. section 3.2), \( \lambda \approx 0.435866521508458994 \), the characteristic polynomial \( p(z) \) of \( A \) satisfies

\[ p(z) = \det(zI_3 - A) = (z - \lambda)^3. \]

If we choose any three distinct abscessae, say \( c = [c_1, c_2, c_3]^T \), then we have the matrix \( A \) has its entries as follows.

\[
\begin{align*}
a_{11} &= \frac{12\alpha_3 - 2c_1^3 - 6\alpha_2c_2 + 3c_1^2c_2 - 6\alpha_2c_3 + 3c_1^2c_3 + 6\alpha_1c_2c_3 - 6c_1c_2c_3}{6(-c_1 + c_2)(c_1 - c_3)}, \\
a_{12} &= \frac{12\alpha_3 - 6\alpha_2c_1 + c_1^3 - 6\alpha_2c_3 + 6\alpha_1c_1c_3 - 3c_1^2c_3}{6(-c_1 + c_2)(-c_2 + c_3)},
\end{align*}
\]
4.2. **EFFECTIVE ORDER CONDITIONS FOR SIRK METHODS**

\[
a_{13} = -\frac{12\alpha_3 - 6\alpha_2 c_1 + c_1^3 - 6\alpha_2 c_2 + 6\alpha_1 c_1 c_2 - 3c_1^2 c_2}{6(c_1 - c_2)(c_2 - c_3)},
\]

\[
a_{21} = -\frac{12\alpha_3 + 6\alpha_2 c_2 - c_2^3 + 6\alpha_2 c_3 - 6\alpha_1 c_2 c_3 + 3c_2^2 c_3}{6(c_1 - c_2)(c_1 - c_3)},
\]

\[
a_{22} = \frac{12\alpha_3 - 6\alpha_2 c_1 + 3c_1 c_2^2 - 2c_2^3 - 6\alpha_2 c_3 + 6\alpha_1 c_1 c_3 - 6c_1 c_2 c_3 + 3c_2^2 c_3}{6(-c_1 + c_2)(-c_2 + c_3)},
\]

\[
a_{23} = -\frac{12\alpha_3 - 6\alpha_2 c_1 + c_1^3 - 6\alpha_2 c_2 + 6\alpha_1 c_1 c_2 - 3c_1^2 c_2}{6(c_1 - c_3)(c_2 - c_3)},
\]

\[
a_{31} = \frac{12\alpha_3 - 6\alpha_2 c_2 - 6\alpha_2 c_3 + 6\alpha_1 c_2 c_3 + c_3^3}{6(-c_1 + c_2)(c_1 - c_3)},
\]

\[
a_{32} = \frac{12\alpha_3 - 6\alpha_2 c_1 + c_3^3 - 6\alpha_2 c_3 + 6\alpha_1 c_1 c_3 - 3c_1 c_3^2}{6(-c_1 + c_2)(-c_2 + c_3)},
\]

\[
a_{33} = -\frac{12\alpha_3 - 6\alpha_2 c_1 - 6\alpha_2 c_2 + 6\alpha_1 c_1 c_2 - 6c_1 c_2 c_3 + 3c_1 c_3^2 + 3c_2 c_3^2 - 2c_3^3}{6(c_1 - c_3)(c_2 - c_3)},
\]

with the characteristic polynomial

\[
p(z) = (\alpha_3 - \frac{1}{3} \alpha_2 (c_1 + c_2 + c_3) + \frac{1}{6} \alpha_1 (c_1 c_2 + c_2 c_3 + c_1 c_3) - \frac{1}{6} c_1 c_2 c_3)
\]
\[+ \left( \alpha_2 - \frac{1}{3} \alpha_1 (c_1 + c_2 + c_3) + \frac{1}{6} (c_1 c_2 + c_2 c_3 + c_1 c_3) \right) z
\]
\[+ \left( \alpha_1 - \frac{1}{3} (c_1 + c_2 + c_3) \right) z^2 + z^3.
\]

Equating the coefficients of \(p(z)\) and \((z - \lambda)^3\), we find

\[
\alpha_1 = \frac{1}{3}(c_1 + c_2 + c_3) - 3\lambda,
\]

\[
\alpha_2 = \frac{1}{9}(c_1 + c_2 + c_3)^2 + \frac{1}{6}(-c_1 c_2 - c_1 c_3 - c_2 c_3) - (c_1 + c_2 + c_3)\lambda + 3\lambda^2,
\]

\[
\alpha_3 = \frac{1}{18}c_1 c_2 c_3 + \frac{1}{27}(c_1^3 + c_2^3 + c_3^3) - \frac{1}{3} \left( c_1^2 + c_2^2 + c_3^2 + \frac{1}{2} (c_1 c_2 + c_2 c_3 + c_1 c_3) \right) \lambda + (c_1 + c_2 + c_3)\lambda^2 - \lambda^3.
\]

It is easy to see that \(\alpha_1, \alpha_2, \alpha_3\) can be determined once the three distinct abscissae \(c_1, c_2, c_3\) are chosen. The coefficient matrix \(A\) can be determined uniquely. For example, for \(c = [0, \frac{1}{2}, 1]^T\), we have

\[
\alpha_1 = \frac{1}{2} - 3\lambda = -0.80759956452537698,
\]

\[
\alpha_2 = \frac{1}{6} - \frac{3}{2} \lambda + 3\lambda^2 = 0.08280575811963002,
\]

85
CHAPTER 4. THE EFFECTIVE ORDER OF SIRK METHODS

\[ \alpha_3 = \frac{1}{24} - \frac{1}{2} \lambda + \frac{3}{2} \lambda^2 - \lambda^3 = 0.02589708465063307. \]

The coefficient matrix \( A \) for the method is then given by

\[
A = \begin{bmatrix}
\frac{1}{2} - \frac{11}{2} \lambda + 15 \lambda^2 & -\frac{5}{3} + 14 \lambda - 24 \lambda^2 & \frac{2}{3} - \frac{11}{2} \lambda + 9 \lambda^2 \\
\frac{17}{24} - \frac{11}{2} \lambda + 15 \lambda^2 & -\frac{4}{3} + 14 \lambda - 24 \lambda^2 & \frac{5}{8} - \frac{11}{2} \lambda + 9 \lambda^2 \\
\frac{2}{3} - \frac{11}{2} \lambda + 15 \lambda^2 & -1 + 14 \lambda - 24 \lambda^2 & \frac{5}{6} - \frac{11}{2} \lambda + 9 \lambda^2
\end{bmatrix},
\]

(2) Weight vector \( b^T \)

After we have the perturbations \( \alpha^T = [\alpha_1, \alpha_2, \alpha_3] \), the weights \( b^T \) can be found using (4.22),

\[
b^T W = \tau^T + \alpha^T (E - I_3).
\]

In other words, \( b^T \) can also be determined uniquely once \( c_1, c_2, c_3 \) are given. In the special case \( c = [0, \frac{1}{2}, 1] \), we have

\[
\tau^T = [1, \frac{1}{2!}, \frac{1}{3!}], \quad W = \begin{bmatrix} 1 & 0 & 0 \\ 1 & \frac{1}{2} & \frac{1}{8} \\ 1 & 1 & \frac{1}{2} \end{bmatrix}, \quad E = \begin{bmatrix} 1 & 1 & \frac{1}{2} \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{bmatrix}.
\]

It can then be shown that

\[
b_1 = \frac{1}{3} - 3 \lambda + 12 \lambda^2, \quad b_2 = -\frac{2}{3} + 12 \lambda - 24 \lambda^2, \quad b_3 = \frac{4}{3} - 9 \lambda + 12 \lambda^2.
\]

Therefore, the new method has the corresponding tableau

<table>
<thead>
<tr>
<th>( a )</th>
<th>( b )</th>
<th>( c )</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0.952428500281735</td>
<td>-0.12404635552734553</td>
</tr>
<tr>
<td>( \frac{1}{2} )</td>
<td>1.160761833615068</td>
<td>0.2092869780598781</td>
</tr>
<tr>
<td>1</td>
<td>1.119095166948401</td>
<td>0.5426203113932113</td>
</tr>
<tr>
<td>1.305489263670564</td>
<td>0.0042206017096269</td>
<td>-0.3097098653801905</td>
</tr>
</tbody>
</table>
(3) Starting method

Suppose the starting method is denoted by \((\bar{A}, \bar{b}, \bar{c})_3\). The abscissae \(\bar{c} = [\bar{c}_1, \bar{c}_2, \bar{c}_3]^T\) and the coefficient matrix \(\bar{A}\) are kept unchanged from the classical SIRK method. Now \(\bar{c}_1 = \frac{\xi_1}{\xi_2} \approx 0.1812220979694, \bar{c}_2 = 1.00000, \bar{c}_3 = \frac{\xi_3}{\xi_2} \approx 2.7415764837792, \xi_1, \xi_2, \xi_3\) are the roots of the degree-3 Laguerre polynomial \(L_3(x)\) (cf. section 3.2). The matrix \(\tilde{A}\) is given by the \(C(3)\) conditions (see section 3.3).

\[
\tilde{A} = W_1 W^{-1},
\]

where \(W_1 = [\bar{c}, \frac{1}{2} \bar{c}^2, \frac{1}{6} \bar{c}^3]\) and \(W = [e, \bar{c}, \frac{1}{2} \bar{c}^2]\). From (4.24)

\[
\tilde{b}^T W = \tau^T + \alpha^T E,
\]

and the weight \(\tilde{b}^T\) can be expressed in terms of \(c_i\). Hence, for the special case \(c = [0, \frac{1}{2}, 1]^T\), the corresponding tableau of the starting method is approximately given by

\[
\begin{array}{cccc}
\bar{c}_1 & 0.208637205597339 & -0.030875105117538 & 0.0034601093171387 \\
\bar{c}_2 & 0.574386497347735 & 0.442669941606176 & -0.0170564389539104 \\
\bar{c}_3 & 0.154424213232075 & 1.930859853225261 & 0.6562924173218653 \\
\end{array}
\]

\[
\begin{array}{cc}
0.530301350609535 & -0.347664899818779 \\
0.0097639846838670 & \\
\end{array}
\]

\[\blacksquare\]

### 4.3 Doubly Companion Matrices

From Example 4.4, if the distinct abscissae \(c_i\) are given, then we can find the perturbations \(\alpha_i\). The coefficient matrix for a SIRK method with effective order \(s\) can be determined using

\[
A \frac{c^{k-1}}{(k-1)!} + \alpha_k e = \frac{c^k}{k!}, \quad k = 1, 2, \ldots, s.
\]

To express the above system in matrix form, we use the modified Vandermonde matrix \(W\) and the vector \(\alpha^T = [\alpha_1, \alpha_2, \ldots, \alpha_s]\), and \(e = [1, 1, \ldots, 1]^T\). We also
write $\tilde{W}$ as $W$ with the first column removed and the additional column $e^s/s!$ added as the last column. With this notation, (4.26) can be written as

$$AW + e\alpha^T = \tilde{W}.$$  

It is easy to see that

$$AW + e\alpha^T = AW + We_1\alpha^T,$$

where $e_1 = [1, 0, \ldots, 0]^T$, and $\tilde{W}$ can be separated into

$$\tilde{W} = WK + \bar{W},$$

where

$$K = \begin{bmatrix} 0 & 0 & 0 & \cdots & 0 & 0 \\ 1 & 0 & 0 & \cdots & 0 & 0 \\ 0 & 1 & 0 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 1 & 0 \end{bmatrix}, \quad \bar{W} = \begin{bmatrix} 0 & 0 & 0 & \cdots & 0 & e_1^T \\ 0 & 0 & 0 & \cdots & 0 & e_2^T \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 0 & e_s^T \end{bmatrix}. $$

Hence, we have

$$AW + We_1\alpha^T = WK + \bar{W},$$

$$\Rightarrow \quad AW = WK - We_1\alpha^T + \bar{W}.$$  

It is natural to ask whether $\bar{W}$ can be factorized in the form $\bar{W} = WX$. If so, we can find the matrix which has similarity with the coefficient matrix $A$. Because the last column of $\bar{W}$ is $\bar{w}_s = ([e_1^T, e_2^T, \ldots, e_s^T]^T$, if we let $\beta = [\beta_s, \beta_{s-1}, \ldots, \beta_1]^T$ and assume $\beta$ satisfies

$$\bar{w}_s = -W\beta,$$  

then we have

$$\bar{W} = \bar{w}_s e_s^T = -W\beta e_s^T,$$
4.3. DOUBLY COMPANION MATRICES

where \( e_s^T = [0, 0, \ldots, 1] \). It can then be shown that

\[
AW = WK - We_1\alpha^T + \bar{W} = WK - We_1\alpha^T - W\beta e_s^T
= W(K - e_1\alpha^T - \beta e_s^T),
\]

which implies

\[
W^{-1}AW = (K - e_1\alpha^T - \beta e_s^T),
\]

(4.28)

and leads to the following result.

**Theorem 4.12** Suppose \( c_1, c_2, \ldots, c_s \) are \( s \) distinct abscissae for an \( s \)-stage SIRK method with effective order \( s \), and \( B_{s,h}(\alpha, y(x_{n-1})) \) is the Butchered solution. If \( \beta_0 = 1 \) and \( \beta_1, \beta_2, \ldots, \beta_s \) are defined by

\[
\prod_{i=1}^{s}(x - c_i) = \sum_{i=0}^{s} \frac{s!}{i!} \beta_{s-i}x^i,
\]

(4.29)

then

\[
W^{-1}AW = C,
\]

where \( W \) is the modified Vandermonde matrix, and \( C \) is the matrix having the form

\[
C = \begin{bmatrix}
-\alpha_1 & -\alpha_2 & -\alpha_3 & \cdots & -\alpha_s & -\beta_s \\
1 & 0 & 0 & \cdots & -\beta_{s-1} \\
0 & 1 & 0 & \cdots & -\beta_{s-2} \\
\vdots & \vdots & \vdots & \ddots & \vdots \\
0 & 0 & 0 & \cdots & -\beta_1 
\end{bmatrix}
\]

(4.30)

**Proof:** From (4.27), we have

\[
\frac{c_i^s}{s!} = -\sum_{k=0}^{s-1} \frac{c_i^{s-k-1}}{(s-k-1)!} \beta_{k+1}
\Rightarrow \sum_{k=0}^{s} \frac{c_i^{s-k}}{(s-k)!} \beta_k = 0, \quad i = 1, 2, \ldots, s.
\]
CHAPTER 4. THE EFFECTIVE ORDER OF SIRK METHODS

If

\[ p(x) = s! \sum_{k=0}^{s} \frac{x^{s-k}}{(s-k)!} \beta_k, \]

then \( p(x) \) is a polynomial of degree \( s \), and \( c_1, c_2, \ldots, c_s \) are \( s \) distinct zeros of \( p \). Hence

\[ p(x) = \prod_{i=1}^{s} (x - c_i) = s! \sum_{k=0}^{s} \frac{x^{s-k}}{(s-k)!} \beta_k = \sum_{i=0}^{s} \frac{s!}{i!} \beta_{s-i} x^i. \]

It is easy to see that

\[ C = K - e_1 \alpha^T - \beta e_3^T, \]

and from (4.28), the result follows. \( \square \)

An example to demonstrate Theorem 4.12 is given as follows.

**Example 4.5** If we choose the abscissae \( c = [0, \frac{1}{2}, 1]^T \) which is the same as the one in Example 4.4, then the perturbations are given by

\[ \alpha_1 = \frac{1}{2} - 3\lambda, \quad \alpha_2 = \frac{1}{6} - \frac{3}{2} \lambda + 3\lambda^2, \quad \alpha_3 = -\frac{1}{8} + \lambda - \frac{3}{2} \lambda^2, \]

where \( \lambda = 0.435866521508459 \) is a zero of \( \lambda^3 - 3\lambda^2 + \frac{3}{2} \lambda - \frac{1}{6} = 0 \) (We note that the two expressions for \( \alpha_3 \) are equal). From (4.29),

\[ x(x - \frac{1}{2})(x - 1) = x^3 + 3\beta_1 x^2 + 6\beta_2 x + 6\beta_3. \]

Hence we have

\[ \beta_1 = -\frac{1}{2}, \quad \beta_2 = \frac{1}{12}, \quad \beta_3 = 0. \]

The \( C \) mentioned in Theorem 4.12 is given by

\[
C = \begin{bmatrix}
-\frac{1}{2} + 3\lambda & -\frac{1}{6} + \frac{3}{2} \lambda - 3\lambda^2 & \frac{1}{8} - \lambda + \frac{3}{2} \lambda^2 \\
1 & 0 & -\frac{1}{12} \\
0 & 1 & \frac{1}{2}
\end{bmatrix}.
\]

\( \square \)
By Theorem 4.12, if the perturbation vector $\alpha^T$ is given, then we can obtain the coefficient matrix $A$ from its similarity with $C$. Furthermore, since we know that $A$ and $C$ have the same characteristic polynomial, it is hoped that instead of equating the coefficients of $z$ in $(z - \lambda)^3$ and the characteristic polynomial of $A$ to find $\alpha^T$, we may be able to use the characteristic polynomial of the matrix $C$ to find a formula for $\alpha^T$.

Let $P(z) = \det(zI - C)$ be the characteristic polynomial of $C$. If $\beta_i = 0, i = 1, 2, \ldots, s$, then

$$P(z) = z^s + \alpha_1 z^{s-1} + \cdots + \alpha_s.$$  

Also, if $\alpha_i = 0, i = 1, 2, \ldots, s$, then

$$P(z) = z^s + \beta_1 z^{s-1} + \cdots + \beta_s.$$  

In each case, $C$ is called the "doubly companion matrix" of $P(z)$. In our case, $C$ has been named the "doubly companion matrix" [21]. Let $\lambda$ be one of the eigenvalues of $P(z)$, and let $v$ be the corresponding eigenvector. It is obvious that the last component of $v$ cannot be zero (otherwise $v = 0$). Without loss of generality, let the last component of $v$ be 1. Also note that every component of $v$ is a function of $\lambda$. We assume that $v = [B_{s-1}(\lambda), B_{s-2}(\lambda), \ldots, B_1(\lambda), B_0(\lambda)]^T$, where $B_0(\lambda) = 1$. From the last component up to the second component of $Cv = \lambda v$, the following recurrence relation can be found

$$
\begin{cases}
B_0(\lambda) = 1, \\
B_k(\lambda) = \lambda B_{k-1}(\lambda) + \beta_k, & k = 1, 2, \ldots, s - 1.
\end{cases}
$$  

If we define

$$B_s(\lambda) = \lambda B_{s-1}(\lambda) + \beta_s,$$  

then the first component of $Cv = \lambda v$ yields

$$- \sum_{k=1}^{s-1} \alpha_k B_{s-k}(\lambda) + (-\alpha_s - \beta_s)B_0(\lambda) = \lambda B_{s-1}.$$  

It can also be shown that

$$\lambda B_{s-1} + \beta_s + \sum_{k=1}^{s} \alpha_k B_{s-k}(\lambda) = B_s(\lambda) + \sum_{k=1}^{s} \alpha_k B_{s-k}(\lambda) = 0.$$  

(4.33)
It is clear that the left hand side of (4.33) gives the characteristic polynomial of the doubly companion matrix $C$; in other words,

$$P(z) = B_s(z) + \sum_{k=1}^{s} \alpha_k B_{s-k}(z).$$

(4.34)

Hence from (4.31), (4.32), we have the following result.

**Theorem 4.13** Let $P(z)$ be the characteristic polynomial of the $s \times s$ doubly companion matrix $C$, $\alpha_0 = 1, \beta_0 = 1$, and let

$$A(z) = \sum_{k=0}^{s} \alpha_k z^{s-k}, \quad B(z) = \sum_{j=0}^{s} \beta_j z^{s-j},$$

with $B_k(z)$ defined by

$$B_0(z) = 1, \quad B_k(z) = \sum_{j=0}^{k} \beta_j z^{k-j}, \quad k = 1, 2, \ldots, s.$$

Then

(1) $P(z)$ is the polynomial $z^{-s} A(z) B(z)$ where terms of negative power have been deleted.

(2) The eigenvector corresponding to an eigenvalue $\lambda$ of $C$ is given by

$$[B_{s-1}(\lambda), B_{s-2}(\lambda), \ldots, B_1(\lambda), 1]^T.$$

**Proof:** It is easy to see $B_s(z) = B(z)$ and the first result can be derived by the following.

$$z^{-s} A(z) B(z) = z^{-s} \left( \sum_{k=0}^{s} \alpha_k z^{s-k} \right) B(z) = z^{-s} \left( z^s + \sum_{k=1}^{s} \alpha_k z^{s-k} \right) B(z)$$

$$= \left( 1 + \sum_{k=1}^{s} \alpha_k z^{-k} \right) \left( \sum_{j=0}^{s} \beta_j z^{s-j} \right) = B_s(z) + \left( \sum_{k=1}^{s} \alpha_k z^{-k} \sum_{j=0}^{s} \beta_j z^{s-j} \right),$$

and for $k = 1, 2, \ldots, s$, we have

$$\alpha_k z^{-k} \sum_{j=0}^{s} \beta_j z^{s-j} = \alpha_k \sum_{j=0}^{s} \beta_j z^{s-j-k} = \alpha_k \left( \sum_{j=0}^{s-k} \beta_j z^{s-k-j} + O(z^{-1}) \right)$$

$$= \alpha_k \left( B_{s-k}(z) + O(z^{-1}) \right).$$
Hence, we can write
\[ z^{-s}A(z)B(z) = B_s(z) + \sum_{k=1}^{s} \alpha_k B_{s-k}(z) + O(z^{-1}), \]
from (4.34), and the first result follows. If \( v \) denotes the eigenvector mentioned in the theorem, then \( Cv = \lambda v \) yields the second result. \( \square \)

Similar derivations to the argument given before Theorem 4.13, gives the row eigenvectors.

**Corollary 4.14** The row eigenvector corresponding to an eigenvalue \( \lambda \) of the doubly companion matrix \( C \) is given by
\[ [A_0(\lambda), A_1(\lambda), A_2(\lambda), \ldots, A_{s-1}(\lambda)]^T, \]
where
\[ A_0(\lambda) = 1, \quad A_k(\lambda) = \lambda A_{k-1}(\lambda) + \alpha_k, \quad k = 1, 2, \ldots, s - 1. \]

Now, we give an example to show how to find the characteristic polynomial for a given doubly companion matrix \( C \).

**Example 4.6** Take the doubly companion matrix \( C \) in Example 4.5,
\[ C = \begin{bmatrix}
-\frac{1}{2} + 3\lambda & -\frac{1}{6} + \frac{3}{2}\lambda - 3\lambda^2 & \frac{1}{8} - \lambda + \frac{3}{2}\lambda^2 \\
1 & 0 & -\frac{1}{12} \\
0 & 1 & \frac{1}{2}
\end{bmatrix}. \]

We have
\[ A(z) = z^3 + \left(\frac{1}{2} - 3\lambda\right)z^2 + \left(\frac{1}{6} - \frac{3}{2}\lambda + 3\lambda^2\right)z + \frac{1}{24} - \frac{1}{2}\lambda + \frac{3}{2}\lambda^2 - \lambda^3, \]
\[ B(z) = z^3 - \frac{1}{2}z^2 + \frac{1}{12}z. \]

By Theorem 4.13, \( P(z) \) can be found from
\[ z^{-3}A(z)B(z) = z^{-3}(z^6 - 3\lambda z^5 + 3\lambda^2 z^4 - \lambda^3 z^3) + O(z^{-1}) \]
\[ = (z - \lambda)^3 + O(z^{-1}). \]

Therefore \( P(z) = (z - \lambda)^3 \). This has also verified the similarity between \( A \) and \( C \). \( \square \)
CHAPTER 4. THE EFFECTIVE ORDER OF SIRK METHODS

The result shown in Theorem 4.13 gives a formula for finding the characteristic polynomial of a given doubly companion matrix. For an SIRK method with effective order \( s \), the coefficient matrix has one single \( s \)-fold eigenvalue \( \lambda \). In other words, the characteristic polynomial of the corresponding doubly companion matrix is given by \((z - \lambda)^s\). After the \( \beta_i \) are found using (4.29), we can use the following Theorem 4.15 to find the perturbations \( \alpha^T \).

**Theorem 4.15** Let \( B_{s,n}(\alpha, y(x_{n-1})) \) be the Butchered solution for an SIRK method with effective order \( s \). If \( \lambda \) is the corresponding eigenvalue of the \( s \)-stage SIRK method, \( \alpha_0 = 1, \beta_0 = 1 \), and \( \beta_1, \ldots, \beta_s \) are defined by

\[
\prod_{i=1}^{s} (1 - c_i z) = s! \sum_{i=0}^{s} \frac{\beta_i z^i}{(s - i)!},
\]

(4.35)

then the perturbations \( \alpha^T = [\alpha_1, \alpha_2, \ldots, \alpha_s] \) are given by the relation

\[
1 + \alpha_1 z + \alpha_2 z^2 + \cdots + \alpha_s z^s = (1 - \lambda)^s(1 + \beta_1 z + \beta_2 z^2 + \cdots + \beta_s z^s)^{-1} + O(z^{s+1}).
\]

(4.36)

**Proof:** By Theorem 4.12, the coefficient matrix \( A \) satisfies

\[
W^{-1} AW = C,
\]

where \( W \) is the modified Vandermonde matrix and \( C \) is the corresponding doubly companion matrix. Because \((z - \lambda)^s\) is the characteristic polynomial of \( A \), it is also the characteristic polynomial of \( C \). By Theorem 4.13, we have

\[
z^{-s} \left( \sum_{k=0}^{s} \alpha_k z^{s-k} \right) \left( \sum_{k=0}^{s} \beta_k z^{s-k} \right) = (z - \lambda)^s + O(z^{-1}),
\]

which implies

\[
z^{-s}(z^s + \alpha_1 z^{s-1} + \alpha_2 z^{s-2} + \cdots + \alpha_s)(z^s + \beta_1 z^{s-1} + \beta_2 z^{s-2} + \cdots + \beta_s) = (z - \lambda)^s + O(z^{-1}).
\]

This is equivalent to

\[
1 + \alpha_1 z + \alpha_2 z^2 + \cdots + \alpha_s z^s = (1 - \lambda)^s(1 + \beta_1 z + \beta_2 z^2 + \cdots + \beta_s z^s)^{-1} + O(z^{s+1}).
\]
With the aid of the doubly companion matrix, we can use the results shown in Theorem (4.15) to find the perturbations for methods. We give an example to illustrate this.

**Example 4.7** Take an order-3 method again. We choose the abscissae $c = [-1, 0, 1]^T$ this time. For order-3 methods, the eigenvalue $\lambda = 0.435866521508458994$. Using (4.35), we have

$$(1 + z)(1 - z) = \beta_0 + 3\beta_1 z + 6\beta_2 z^2 + 6\beta_3 z^3.$$ 

Hence

$$\beta_1 = 0, \quad \beta_2 = -\frac{1}{6}, \quad \beta_3 = 0$$

From (4.36), we have

$$1 + \alpha_1 z + \alpha_2 z^2 + \alpha_3 z^3 = (1 - \lambda z)^3 \left(1 - \frac{1}{6} z^2\right)^{-1} + O(z^4) = 1 - 3\lambda z + \left(\frac{1}{6} + 3\lambda^2\right) z^2 + \left(-\frac{1}{2} - \lambda^3\right) z^3.$$ 

Hence

$$\alpha_1 = -3\lambda, \quad \alpha_2 = \frac{1}{6} + 3\lambda^2, \quad \alpha_3 = -\frac{1}{2} - \lambda^3.$$ 

Another application of doubly companion matrices is to find the transformation matrix for this special class of singly-implicit methods. This is based on the following theorem [21].

**Theorem 4.16** Let $B_k(\lambda)$ be defined by

$$B_0(\lambda) = 1, \quad B_k(\lambda) = \sum_{j=0}^{k} \beta_j \lambda^{k-j}, \quad k = 1, 2, \ldots, s.$$
and if a doubly companion matrix $C$ has characteristic polynomial $(z - \lambda)^s$, then

$$V^{-1}CV = \begin{bmatrix} \lambda & 0 & 0 & \ldots & 0 \\ 1 & \lambda & 0 & \ldots & 0 \\ 0 & 1 & \lambda & \ldots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \ldots & \lambda \end{bmatrix},$$

where $V = [v_{s-1}, v_{s-2}, \ldots, v_1, v_0]$, and $v_k = \frac{1}{k!} \frac{d^k}{d\lambda^k} [B_{s-1}(\lambda), B_{s-2}(\lambda), \ldots, B_1(\lambda), 1]^T$, for $k = 0, 1, \ldots, s - 1$.

**Proof:** From Theorem 4.13, we know that $v_k$ defined in this way implies that

$$Cv_0 = \lambda v_0. \quad (4.37)$$

Because $C$ has a single $s$-fold eigenvalue $\lambda$, we can differentiate up to $s - 1$ times both sides of (4.37) with respect to $\lambda$, yielding

$$Cv_0^{(k)} = \lambda v_0^{(k)} + kv_0^{(k-1)}, \quad k = 1, 2, \ldots, s - 1.$$

From the definition of $v_k$ we have

$$k!v_k = v_0^{(k)}, \quad k = 1, 2, \ldots, s - 1.$$ 

we have

$$Ck!v_k = \lambda k!v_k + k(k - 1)!v_{k-1}, \quad k = 1, 2, \ldots, s - 1, \quad Cv_k = \lambda v_k + v_{k-1}, \quad k = 1, 2, \ldots, s - 1,$$

and the result follows. \qed

The results from Theorem 4.16 show that the doubly companion matrix $C$ is similar to a matrix which has Jordan block form with a single diagonal element $\lambda$. Consequently, the idea of Butcher’s transformation can be adapted to these new methods. Ideally, we propose to use (3.8) in section 3.1 for implementation. First, we shall seek the transformation matrix $T$ such that $T^{-1}AT = \tilde{A}$, where
4.3. **DOUBLY COMPANION MATRICES**

$\tilde{A}$ is the Jordan canonical form of the coefficient matrix $A$. In our case, from Theorem 4.16, we have

$$V^{-1}CV = \tilde{A} = \lambda I_s + K,$$

where

$$K = \begin{bmatrix}
0 & 0 & \ldots & 0 \\
1 & 0 & \ldots & 0 \\
0 & 1 & \ldots & 0 \\
\vdots & \vdots & \ddots & \vdots \\
0 & 0 & \ldots & 1 \\
0 & 0 & \ldots & 1
\end{bmatrix}.$$

Using the similarity between $A$ and $C$, that is $W^{-1}AW = C$, we have

$$V^{-1}(W^{-1}AW)V = (WV)^{-1}A(WV) = \lambda I_s + K.$$ 

Hence we denote $T = WV$. In order to use (3.8), we need to factorize $T = WV$ into $UL$, where $U$ is a unit upper triangular matrix and $L$ is a lower triangular matrix. Hence, it follows that

$$(UL)^{-1}AUL = L^{-1}(U^{-1}AU)L = \lambda I_s + K,$$

and

$$U^{-1}AU = L(\lambda I_s + K)L^{-1} = \tilde{A},$$

where $\tilde{A}$ is a lower triangular matrix with one single diagonal entry $\lambda$. Now, the transformation matrix we used is $U$. Finally, the matrix $M$ proposed is given by $M = \lambda \tilde{A}^{-1}$. When $\Delta Z$ is the Newton update which satisfies $Z[i+1] = Z[i] + \Delta Z$, $Z[i]$ where is the value in the $l$-th iteration, according to (3.8), the formula for the Newton decrements is given by

$$(I_n - h\lambda J)\Delta \hat{Z}_i = -\hat{Z}_i + h\lambda \hat{F}_i + \sum_{j<i} m_{ij}(\hat{Z}_j - \Delta \hat{Z}_j), \quad i = 1, 2, \ldots, s,$$

where

$$\hat{Z} = (M \otimes I_n) \hat{Z} = (M \otimes I_n)(U^{-1} \otimes I_n)Z,$$

$$\hat{F} = (U^{-1} \otimes I_n)F,$$
and the update $\Delta Z$ is given by

$$\Delta Z = (U \otimes I_n) \Delta \tilde{Z}.$$ 

A Mathematica subroutine "Twv" in Appendix A.1 can be used to find the transformation matrix $T$ for these methods, and a Matlab program "u1" used to factorize $T = UL$ is also available in Appendix A.1. We now give an example to demonstrate Theorem 4.16 and its applications.

Example 4.8 Choose abscissae $c = [0, 1/2, 1]^T$ for an order-3 method, from Example 4.5, $\lambda = 0.435866521508458994$, a zero of $L_3(1_\lambda) = 0$ and

$$\beta_1 = -\frac{1}{2}, \quad \beta_2 = \frac{1}{12}, \quad \beta_3 = 0.$$ 

Since

$$v_k = \frac{1}{k!} \frac{d^k}{d\lambda^k} [B_2(\lambda), B_1(\lambda), 1]^T, \quad k = 0, 1, 2,$$

and

$$B_0(\lambda) = 1, \quad B_1(\lambda) = \lambda + \beta_1 = \lambda - \frac{1}{2},$$

$$B_2(\lambda) = \lambda^2 + \beta_1 \lambda + \beta_2 = \lambda^2 - \frac{1}{2} \lambda + \frac{1}{12},$$

the transformation matrix $V$ in Theorem 4.16 is given by

$$V = [v_2, v_1, v_0]$$

$$= \begin{bmatrix} 1 & 2 & -\frac{1}{2} + \lambda & \frac{1}{12} - \frac{1}{2} \lambda + \lambda^2 \\ 0 & 1 & -\frac{1}{2} + \lambda \\ 0 & 0 & 1 \end{bmatrix}.$$ 

The Vandermonde matrix $W$ is

$$W = \begin{bmatrix} 1 & 0 & 0 \\ 1 & \frac{1}{2} & \frac{1}{8} \\ 1 & 1 & \frac{1}{2} \end{bmatrix}.$$
Hence the transformation matrix $T$ is given by

$$T = WV = \begin{bmatrix} 1 & -\frac{1}{2} + 2\lambda & \frac{1}{12} - \frac{1}{2} \lambda + \lambda^2 \\ 1 & 2\lambda & -\frac{1}{24} + \lambda^2 \\ 1 & \frac{1}{2} + 2\lambda & \frac{1}{12} + \frac{1}{2} \lambda + \lambda^2 \end{bmatrix}.$$ 

Factorizing $T = UL$ with $U$ as a unit upper triangular matrix and $L$ as a lower triangular matrix, we have

$$U = \begin{bmatrix} 1 & -\frac{4+48\lambda^2}{1+12\lambda+24\lambda^2} & \frac{1-6\lambda+12\lambda^2}{1+6\lambda+12\lambda^2} \\ 0 & 1 & -\frac{1+24\lambda^2}{2+12\lambda+24\lambda^2} \\ 0 & 0 & 1 \end{bmatrix},$$

and

$$L = \begin{bmatrix} \frac{6}{1+12\lambda+24\lambda^2} & 0 & 0 \\ \frac{3+12\lambda}{2+12\lambda+24\lambda^2} & \frac{1+24\lambda^2}{4+24\lambda+48\lambda^2} & 0 \\ 1 & \frac{1}{2} + 2\lambda & \frac{1}{12} + \frac{1}{2} \lambda + \lambda^2 \end{bmatrix}.$$ 

Hence

$$\hat{A} = L(\lambda I_s + K)L^{-1} = \begin{bmatrix} \lambda & 0 & 0 \\ \frac{(1+12\lambda+24\lambda^2)^2}{24(1+6\lambda+12\lambda^2)} & \lambda & 0 \\ \frac{1}{2} \lambda + 3\lambda^2 + 4\lambda^3 & \frac{(1+6\lambda+12\lambda^2)^2}{3+36\lambda+72\lambda^2} & \lambda \end{bmatrix},$$

and this implies

$$M = \lambda \hat{A}^{-1} = \begin{bmatrix} 1 & 0 & 0 \\ -\frac{(1+12\lambda+24\lambda^2)^2}{24(1+6\lambda+12\lambda^2)} & 1 & 0 \\ 1 + \frac{1}{72}\lambda^2 + \frac{1}{4}\lambda + \lambda - \frac{(1+6\lambda+12\lambda^2)^2}{\lambda(3+36\lambda+72\lambda^2)} & \frac{1}{\lambda(3+36\lambda+72\lambda^2)} & 1 \end{bmatrix}. $$

\[4.4 \text{ ESIRK methods}\]

Following the previous three sections, we will call an $s$-stage SIRK method with effective order $s$ an "ESIRK" method. In other words, if the input at $x_{n-1}$ is
CHAPTER 4. THE EFFECTIVE ORDER OF SIRK METHODS

$B_{s,h}(a, y(x_{n-1}))$, and the numerical solution after the integration of this step is an approximation to $B_{s,h}(a, y(x_n))$, then an $s$-stage ESIRK method satisfies

$$B_{s,h}(a, y(x_n)) - \phi(B_{s,h}(a, y(x_{n-1}))) = O(h^{s+1}),$$

(4.38)

where $\phi$ is the mapping associated with the method. In this section, we summarize the systematic construction of ESIRK methods and we also consider their local error and stability properties.

From the last section, we know that if the eigenvalue $\lambda$ and the abscissae $c_i$ for an $s$-stage ESIRK method are given, we will be able to use Theorem 4.15 to find the perturbation vector $\alpha^T$. Once $\alpha^T$ is known, then Theorem 4.12 and Theorem 4.10 can be used to find the coefficient matrix $A$ and the weights $b^T$. Thus, the systematic construction of ESIRK methods, together with their transformation matrices is as follows:

1. Select $\lambda$ from Table 3.3 to satisfy the stability condition;
2. Choose distinct $c_1, c_2, \ldots, c_s$ for convenience;
3. Use the formula:
   $$ (1 - c_1 z)(1 - c_2 z) \cdots (1 - c_s z) = 1 + s\beta_1 z + s(s-1)\beta_2 z^2 + \cdots + s!\beta_s, $$
   to find the $\beta_1, \beta_2, \ldots, \beta_s$;
4. Use the formula:
   $$ 1 + \alpha_1 z + \alpha_2 z^2 + \cdots + \alpha_s z^s = (1 + \beta_1 z + \beta_2 z^2 + \cdots + \beta_s z^s)^{-1}(1 - \lambda z)^s + O(z^{s+1}), $$
   to find the $\alpha_1, \alpha_2, \ldots, \alpha_s$;
5. Use $W^{-1}AW = C$ to find the matrix $A$,
   where $C$ is the "doubly companion matrix" (4.30), and $W$ is the "Vandermonde matrix";
6. Use the formula
   $$ b^T W = \tau^T + \alpha^T (E - I_s), $$
   (4.39)
   $$ = \left[ 1, \frac{1}{2!} + \alpha_1, \frac{1}{3!} + \frac{1}{2!} \alpha_1 + \alpha_2, \ldots, \frac{1}{s!} + \cdots + \alpha_{s-1} \right], $$
   (4.40)
   to find $b^T$.

100
4.4. **ESIRK METHODS**

(7) Use \( V = [v_{s-1}, v_{s-2}, \ldots, v_1, v_0] \), to find the matrix \( V \), where \( v_k = \frac{1}{k!} \frac{d^k}{d\lambda^k} [B_{s-1}(\lambda), B_{s-2}(\lambda), \ldots, B_1(\lambda), 1]^T \), for \( k = 0, 1, \ldots, s - 1 \), and

\[
B_0 = 1, \quad B_k(\lambda) = \sum_{j=0}^{k} \beta_j \lambda^{k-j}, \quad k = 1, 2, \ldots, s; \quad (4.41)
\]

(8) Use \( T = WV \) to find the matrix \( T \), then from the factorization

\[
T = UL,
\]

where \( U \) is a unit upper triangular matrix and \( L \) is a lower triangular matrix, we find the transformation matrix \( U \);

(9) Calculate

\[
\hat{A} = L(\lambda I_s + K)L^{-1},
\]

then use \( M = \lambda \hat{A}^{-1} \) to find the matrix \( M \) for Newton iterations.

The procedures for deriving the corresponding starting method can be written as follows:

(1) The abscissae and the coefficient matrix are the same as the associated SIRK method; (3.3)

(2) Find the weights \( \vec{b}^T \) from

\[
\vec{b}^T W = T^T + \alpha^T E = \begin{bmatrix} 1 + \alpha_1, \frac{1}{2!} + \alpha_1 + \alpha_2, \ldots, \frac{1}{s!} + \cdots + \alpha_{s-1} + \alpha_s \end{bmatrix}.
\]

We now present some examples for constructing some order 2, 3, 4 ESIRK methods and their associated starting methods and transformation matrices.

**Example 4.9** We begin with order-1 ESIRK methods with \( \lambda = 1 \). In this case, \( \beta = -1 \) and \( \alpha_i = c_i - \lambda \). The standard choice has \( c_i = 1 \) in order that the first stage value \( Y_s \) can be the numerical output. Hence the perturbation \( \alpha_1 = 0 \). The
coefficient matrix $A$ and the weight $b^T$ are unchanged. The associated Butcher tableau is given by
\[ \begin{array}{cc}
1 & 1 \\
1 & 1 \\
\end{array} \]
Therefore the order-1 ESIRK method is the implicit Euler method. It is also clear that we do not need either a starting method or a transformation matrix in this case. \( \Box \)

**Example 4.10** For the order-2 ESIRK methods, the eigenvalue is $\lambda = 1 - \sqrt{\frac{3}{2}}$.

For starting methods, the vector $c$ and the coefficient matrix $A$ of the classical SIRK method will not be changed but we need to modify the $b^T$ vector in accordance with the required perturbation of ESIRK input values. Denote the modified weight vector by $\tilde{b}^T = [\tilde{b}_1, \tilde{b}_2]$. $[\tilde{b}_1, \tilde{b}_2][\epsilon, \tilde{c}] = \left[ 1 + \alpha_1, \frac{1}{2} + \alpha_1 + \alpha_2 \right]$, where $\epsilon = [3 - 2\sqrt{2}, 1]^T$. The method is given by the tableau
\[
\begin{array}{c|cc}
3 - 2\sqrt{2} & \frac{5 - 3\sqrt{2}}{4} & \frac{7 - 5\sqrt{2}}{4} \\
1 & \frac{1 + \sqrt{2}}{4} & \frac{3 - \sqrt{2}}{4} \\
\end{array}
\]
where
\[
\tilde{b}_1 = \frac{-1 + \epsilon_1^2 - 4(\epsilon_1 + 1)\lambda + 4\lambda^2}{4(-1 + \epsilon_1)}, \\
\tilde{b}_2 = \frac{(-1 + \epsilon_1)(5 + \epsilon_1) + 4(3 - \epsilon_1)\lambda - 4\lambda^2}{4(-1 + \epsilon_1)}.
\]
For ESIRK methods, for arbitrary $c_1$ and $c_2 = 1$, we will express the coefficient matrix $A$ and weights $b^T$ in terms of $c_1$ and $\lambda$. In this case, $\beta_1 = \frac{-1 - c_1}{2}$, $\beta_2 = \frac{c_1}{2}$ and $\alpha_1 = \frac{1 + c_1}{2} - 2\lambda$, $\alpha_2 = \frac{1 + c_1^2}{4} - (1 + c_1)\lambda + \lambda^2$ and the coefficients in the method are given by
\[
a_{11} = \frac{(1 - c_1)^2 + 4(c_1 - 1)\lambda - 4\lambda^2}{4(-1 + c_1)}, \\
a_{12} = \frac{(-1 + c_1 + 2\lambda)^2}{4(-1 + c_1)},
\]
102
4.4. ESIRK METHODS

\[ a_{21} = \frac{(1 - c_1 + 2\lambda)^2}{4(1 - c_1)}, \quad a_{22} = \frac{-(1 - c_1)^2 + 4(c_1 - 1)\lambda + 4\lambda^2}{4(-1 + c_1)}, \]
\[ b_1 = \frac{c_1 - 4\lambda}{2(-1 + c_1)}, \quad b_2 = \frac{-2 + c_1 + 4\lambda}{2(-1 + c_1)}. \]

For the special case \( c_1 = 0, c_2 = 1 \), the corresponding tableau is given by

\[
\begin{array}{c|cc}
0 & 9 - 6\sqrt{2} & -3 + 2\sqrt{2} \\
1 & 11 - 6\sqrt{2} & -1 + 2\sqrt{2} \\
2 & -2 - \sqrt{2} & -1 + \sqrt{2}
\end{array}
\]

From \( T = WV \) and (4.41), its associated transformation matrix \( T \) is given by

\[
T = WV = \begin{bmatrix} 1 & -\frac{1}{2} + \lambda \\ 1 & \frac{1}{2} + \lambda \end{bmatrix}.
\]

**Example 4.11** For an order-3 ESIRK method, \( \lambda \approx 0.435866521508459 \) and \( c = [0, \frac{1}{3}, 1] \). In this case, \( \beta_1 = -\frac{2}{5}, \beta_2 = \frac{3}{30}, \beta_3 = 0, \) and \( \alpha_1 = \frac{2}{5} - 3\lambda, \alpha_2 = \frac{19}{150} - \frac{6}{5}\lambda + 3\lambda^2 \) and \( \alpha_3 = \frac{14}{375} - \frac{19}{50}\lambda + \frac{6}{5}\lambda^2 - \lambda^3. \)

The starting method will be given by \( A = W_1 W^{-1}, W_1 = [\tilde{c}_1, \frac{1}{2} \tilde{c}_2, \frac{1}{6} \tilde{c}_3], \) and \( \tilde{c} = [\tilde{c}_1, \tilde{c}_2, \tilde{c}_3]^T, A \) with the same as the classical order-3 method. (cf. Example 3.22). From (4.39), \( \tilde{b}^T = [0.446293, -0.376275, 0.022382]. \) The ESIRK method is then given by the tableau

\[
\begin{array}{c|cccc}
0 & -\frac{1}{75} - \frac{2}{5}\lambda + 6\lambda^2 + 10\lambda^3 & -\frac{13}{40} + \frac{11}{4}\lambda - \frac{15}{4}\lambda^2 - \frac{25}{2}\lambda^3 & -\frac{37}{600} + \frac{13}{20}\lambda - \frac{9}{4}\lambda^2 + \frac{5}{2}\lambda^3 \\
\frac{1}{5} & \frac{2}{25} - \frac{2}{5}\lambda + 6\lambda^2 + 10\lambda^3 & -\frac{13}{60} + \frac{11}{4}\lambda - \frac{15}{4}\lambda^2 - \frac{25}{2}\lambda^3 & -\frac{38}{600} + \frac{13}{20}\lambda - \frac{9}{4}\lambda^2 + \frac{5}{2}\lambda^3 \\
1 & -\frac{26}{75} - \frac{2}{5}\lambda + 6\lambda^2 + 10\lambda^3 & \frac{43}{60} + \frac{11}{4}\lambda - \frac{15}{4}\lambda^2 - \frac{25}{2}\lambda^3 & \frac{23}{100} + \frac{13}{20}\lambda - \frac{9}{4}\lambda^2 + \frac{5}{2}\lambda^3 \\
\frac{8}{15} & -9\lambda + 30\lambda^2 & -\frac{13}{24} + 15\lambda - \frac{75}{2}\lambda^2 & \frac{121}{120} - 6\lambda + \frac{15}{2}\lambda^2
\end{array}
\]

The corresponding transformation matrix is found to be

\[
T = \begin{bmatrix}
1 & -\frac{2}{5} + 2\lambda & \frac{1}{30} - \frac{2}{5}\lambda + \lambda^2 \\
1 & -\frac{1}{5} + 2\lambda & -\frac{1}{75} - \frac{1}{5}\lambda + \lambda^2 \\
1 & \frac{3}{5} + 2\lambda & \frac{2}{15} - \frac{3}{5}\lambda + \lambda^2
\end{bmatrix}.
\]
**Example 4.12** An order-4 ESIRK method with $\lambda \approx 0.572816062482134886$ and $c = [0, \frac{1}{3}, \frac{2}{3}, 1]$. In this case, $\beta_1 = -\frac{1}{2}, \beta_2 = \frac{11}{108}, \beta_3 = -\frac{1}{108}, \beta_4 = 0$ and $\alpha_1 = \frac{1}{2} - 4\lambda, \alpha_2 = \frac{4}{27} - 2\lambda + 6\lambda^2, \alpha_3 = \frac{7}{216} - \frac{16}{27}\lambda + 3\lambda^2 - 4\lambda^3, \alpha_4 = \frac{67}{13664} - \frac{7}{54}\lambda + \frac{8}{9}\lambda^2 - 2\lambda^3 + \lambda^4$. The starting method is given approximately by the tableau

\[
\begin{array}{cccccc}
 & c_1 & c_2 & c_3 & c_4 \\
0 & 0.217815 & -0.040348 & 0.007937 & -0.000642 \\
\frac{1}{3} & 0.561831 & 0.468284 & -0.032332 & 0.002217 \\
\frac{2}{3} & 0.323301 & 1.597585 & 0.692960 & -0.015132 \\
1 & 1.373687 & -0.796387 & 3.892135 & 0.912268 \\
\hline
& -0.164079 & -0.873444 & 0.266428 & -0.020185
\end{array}
\]

where $c_1 = 0.18476, c_2 = 1.00000, c_3 = 2.59867, c_4 = 5.38199$. The associated ESIRK method is given by the tableau

\[
\begin{array}{cccc}
 & a_{11} & a_{12} & a_{13} & a_{14} \\
0 &  &  &  &  \\
\frac{1}{3} &  &  &  &  \\
\frac{2}{3} &  &  &  &  \\
1 &  &  &  &  \\
\hline
& b_1 & b_2 & b_3 & b_4
\end{array}
\]

where

\[
\begin{align*}
a_{11} &= -\frac{49}{432} + \frac{5}{6}\lambda + 3\lambda^2 + 18\lambda^3 + 27\lambda^4, \\
a_{12} &= -\frac{49}{144} + \frac{11}{6}\lambda + 9\lambda^2 - 18\lambda^3 - 81\lambda^4, \\
a_{13} &= -\frac{5}{144} + \frac{11}{6}\lambda - 9\lambda^2 - 18\lambda^3 + 81\lambda^4, \\
a_{14} &= -\frac{5}{432} + \frac{1}{6}\lambda - 3\lambda^2 + 18\lambda^3 - 27\lambda^4, \\
b_1 &= -\frac{7}{12} + 11\lambda - 54\lambda^2 + 108\lambda^3,
\end{align*}
\]
4.4. ESIRK METHODS

\[ b_2 = \frac{7}{3} - 39\lambda + 216\lambda^2 - 324\lambda^3, \]
\[ b_3 = -\frac{35}{12} + 57\lambda - 270\lambda^2 + 324\lambda^3, \]
\[ b_4 = \frac{13}{6} - 29\lambda + 108\lambda^2 - 108\lambda^3. \]

Approximately, this will be

<table>
<thead>
<tr>
<th></th>
<th>7.25639</th>
<th>-8.44076</th>
<th>3.39984</th>
<th>-0.424197</th>
</tr>
</thead>
<tbody>
<tr>
<td>1/3</td>
<td>7.38139</td>
<td>-8.17688</td>
<td>3.33039</td>
<td>-0.410308</td>
</tr>
<tr>
<td>2/3</td>
<td>7.36750</td>
<td>-7.99632</td>
<td>3.51095</td>
<td>-0.424197</td>
</tr>
<tr>
<td>1</td>
<td>7.38139</td>
<td>-8.06576</td>
<td>3.77484</td>
<td>-0.299197</td>
</tr>
<tr>
<td>-----</td>
<td>---------</td>
<td>----------</td>
<td>---------</td>
<td>------------</td>
</tr>
<tr>
<td></td>
<td>8.29801</td>
<td>-10.0292</td>
<td>2.03818</td>
<td>0.693020</td>
</tr>
</tbody>
</table>

The transformation matrix is found to be

\[
T = \begin{bmatrix}
1 & -\frac{1}{2} + 6\lambda & \frac{11}{108} - \lambda + 3\lambda^2 & -\frac{1}{108} + \frac{11}{108}\lambda - \frac{1}{2}\lambda^2 + \lambda^3 \\
1 & -\frac{1}{6} + 3\lambda & -\frac{1}{108} - \frac{1}{3}\lambda + 3\lambda^2 & \frac{1}{324} - \frac{1}{108}\lambda - \frac{1}{6}\lambda^2 + \lambda^3 \\
1 & \frac{1}{6} + 3\lambda & -\frac{1}{108} + \frac{1}{3}\lambda + 3\lambda^2 & -\frac{1}{324} - \frac{1}{108}\lambda + \frac{1}{6}\lambda^2 + \lambda^3 \\
1 & \frac{1}{2} + 3\lambda & \frac{11}{108} + \lambda + 3\lambda^2 & \frac{1}{108} + \frac{11}{108}\lambda + \frac{1}{2}\lambda^2 + \lambda^3 \\
\end{bmatrix}
\]

ESIRK methods are designed to be the generalization of classical SIRK methods. They are aimed to have similar error behaviour and stability properties as the SIRK methods. Recall that the classical definition of Local Truncation Error for a Runge–Kutta method at \( x_{n-1} + h \) is

\[ y(x_{n-1} + h) - \phi(y(x_{n-1})) = O(h^{s+1}), \]

where \( \phi \) is the mapping associated with an order-\( s \) Runge–Kutta method. Because ESIRK methods satisfy only effective order conditions, this is replaced by

\[ \psi(y(x_{n-1} + h)) - \phi(\psi(y(x_{n-1}))) = O(h^{s+1}), \]

where \( \psi \) is the mapping associated with the starting method. It is appropriate to give the definition of Local Truncation Error for ESIRK methods as follows:
CHAPTER 4. THE EFFECTIVE ORDER OF SIRK METHODS

**Definition 4.17** The Local Truncation Error at $x_{n-1}$ for an $s$-stage ESIRK method $(A, b, c)_s$ is defined by:

$$
\delta(x_{n-1}) = B_{s,h}(\alpha, y(x_{n-1} + h)) - B_{s,h}(\alpha, y(x_{n-1})) - h \sum_{j=1}^{s} b_j f(Y_j).
$$

where $B_{s,h}(\alpha, y(x_{n-1}))$ is the Butchered solution at $x_{n-1}$.

By this definition, we have

$$
\delta(x_{n-1}) = \sum_{k=0}^{s} \alpha_k h^k y^{(k)}(x_{n-1} + h) - \sum_{k=0}^{s} \alpha_k h^k y^{(k)}(x_{n-1}) - h \sum_{j=1}^{s} b_j f(Y_j)
$$

$$
= \hat{C} h^{s+1} y^{(s+1)}(x_{n-1}) + O(h^{s+2}).
$$

(4.43)

The error constant $\hat{C}$ can be derived from stage order conditions and Taylor expansion. We give the following result.

**Theorem 4.18** The error constant for an ESIRK method is independent of the choice of (distinct) abscissae. In particular, it is the same as for the corresponding SIRK method.

**Proof:** It is easy to see that

$$
h b_j f(Y_j) = b_j \left( \sum_{k=1}^{s} \frac{c_{j-1}^{k-1}}{(k-1)!} h^k y^{(k)}(x_{n-1}) + \frac{c_s^j}{s!} h^{s+1} y^{(s+1)}(x_{n-1}) \right) + O(h^{s+2})
$$

for $j = 1, \ldots, s$, and

$$
\alpha_k h^k y^{(k)}(x_{n-1} + h) = \alpha_k \sum_{l=0}^{s+1-k} \frac{1}{l!} h^{k+l} y^{(k+l)}(x_{n-1}) + O(h^{s+2})
$$

for $k = 0, \ldots, s$. Sum the coefficients of terms in $h^{s+1} y^{(s+1)}$ and use the fact that $k + l = s + 1$ to give the result

$$
\hat{C} = \sum_{k=0}^{s} \frac{\alpha_k}{(s + 1 - k)!} - \sum_{j=1}^{s} \frac{b_j c_j^s}{s!} = \frac{1}{s!} \left( \sum_{k=0}^{s} \frac{s!}{(s + 1 - k)!} \alpha_k - \sum_{j=1}^{s} b_j c_j^s \right).
$$
Table 4.3: Absolute value of error constant for ESIRK methods

<table>
<thead>
<tr>
<th>s</th>
<th>C</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>0.04044011451988</td>
</tr>
<tr>
<td>3</td>
<td>0.02589708465063</td>
</tr>
<tr>
<td>4</td>
<td>0.02725898678972</td>
</tr>
<tr>
<td>5</td>
<td>0.00053004823515</td>
</tr>
<tr>
<td>6</td>
<td>0.00034135127283</td>
</tr>
<tr>
<td>8</td>
<td>0.00000268206244</td>
</tr>
</tbody>
</table>

Also using (4.17) from Theorem 4.8, (4.35), (4.36) from Theorem 4.15 and standard properties of Laguerre polynomials $L_s(x)$ (cf. (3.15), (3.16)), we can simplify $\hat{C}$ as follows:

$$\hat{C} = \sum_{k=0}^{s} \frac{1}{(s-k+1)!} \binom{s}{k} (-1)^k \lambda^k = \sum_{j=0}^{s} \frac{1}{(j+1)!} \binom{s}{j} (-1)^{s-j} \lambda^{s-j}$$

$$= \lambda^{s+1} (-1)^s \int_0^1 L_s(x) dx = \lambda^{s+1} (-1)^{s+1} \left( L_{s+1} \left( \frac{1}{\lambda} \right) - L_s \left( \frac{1}{\lambda} \right) \right)$$

$$= \frac{\lambda^{s} (-1)^{s+1}}{s+1} L_{s+1}^{(1)} \left( \frac{1}{\lambda} \right).$$

This is the same as the error constant of an $s$-stage SIRK method (see (3.17)).

Taking the eigenvalue $\lambda$ in Table 3.3 for L-stability, we list the error constants for $s = 2, 3, \ldots, 6$ and $s = 8$ ESIRK methods in Table 4.3.

Because the method for changing stepsize for ESIRK methods is unnatural, in addition to the error constant, the accuracy of ESIRK methods will also depend on some other factors (we will discuss this in the following two sections). But, at this stage, from the point of view of the generalization of SIRK methods, we learn that ESIRK methods will not lose accuracy when constant stepsize is used. We also note that the way of designing ESIRK methods ensures that their stability properties are the same as the associated SIRK methods when the stepsize is
constant. This can be seen in the following theorems.

**Theorem 4.19** When applying an ESIRK method to the linear test equation \( y' = \lambda y \) with \( h\lambda = z \), the Butchered solution \( B_{s,n}(\alpha, y(x_{n-1})) \) satisfies

\[
B_{s,n}(\alpha, y(x_{n-1})) = \phi(z)y(x_{n-1}),
\]

where \( \phi(z) = \alpha_0 + \alpha_1z + \alpha_2z^2 + \cdots + \alpha_sz^s \), \( \alpha_0 = 1 \).

**Proof:** From the definition of \( B_{s,n}(\alpha, y(x_{n-1})) \), we have

\[
B_{s,n}(\alpha, y(x_{n-1})) = \sum_{i=0}^{s} \alpha_i h^i y^{(i)}(x_{n-1}) = \sum_{i=0}^{s} \alpha_i (h\lambda)^i y(x_{n-1}) = y(x_{n-1}) \sum_{i=0}^{s} \alpha_i z^i.
\]

From the theorem above, it is easy to derive the following result.

**Theorem 4.20** For an ESIRK method with given \( \lambda \) and order \( p = s \), the stability function is independent of the choice of \( c_1, c_2, \ldots, c_s \). In particular, the stability function is the same as that for the corresponding SIRK method.

**Proof:** Let \( R(z) \) and \( \overline{R}(z) \) be stability functions for the SIRK and ESIRK methods of order \( s \) respectively. Suppose

\[
R(z) = \frac{P(z)}{(1 - \lambda z)^s}, \quad \overline{R}(z) = \frac{\overline{P}(z)}{(1 - \lambda z)^s}.
\]

We must show that \( P(z) = \overline{P}(z) \). However, this follows from

\[
\frac{P(z)}{(1 - \lambda z)^s} = \exp(z) + O(z^{p+1}), \quad \frac{\overline{P}(z)}{(1 - \lambda z)^s} = \exp(z) + O(z^{p+1}),
\]

and the fact that \( P \) and \( \overline{P} \) have the same degree \( s \).

**Experiment 4.1** From the literature, we test some order 2, 3 SIRK and ESIRK methods on the Prothero-Robinson problem (2.5) with \( \lambda = -100 \) and the Kaps problem (3.9). For \( s = 2 \), the abscissae for ESIRK methods are chosen as \([0, 1]\) and \([\frac{7-4\sqrt{2}}{3}, 1]\) (The reason for the second choice will be discussed in the next
4.4. ESIRK METHODS

For $s = 3$, the abscissae for ESIRK methods are chosen to be $[0, 1/2, 1]$ and $[0, 1/5, 1]$. We can see that the error behaviour for SIRK and ESIRK methods are approximately the same if the stepsize is small enough (cf. Figure 4.3, 4.4). The accuracy is independent of the choice of the distinct abscissae, they have similar error performance as their associated SIRK methods. (cf. Table 4.4).

Because the abscissae for ESIRK methods can be placed distinctly inside the integration interval, it is hoped that the problem of the abscissae lying outside the interval $[0, 1]$ which appears in classical SIRK methods can be overcome by using ESIRK methods. Therefore, we also list in a table the difference of the average iteration numbers for different choices of abscissae and SIRK methods. The tolerances $tol$ for controlling the Newton iterations have been chosen as $10^{-4}, 10^{-6}$ for $s = 2$ and $10^{-6}, 10^{-8}$ for $s = 3$. When $s = 2$, the ESIRK method $[7 - 4\sqrt{2}, 1]$ still has fewer iterations than the others, the choice $[0, 1]$ has a similar number of iterations as the SIRK method (cf. Table 4.5). In the case $s = 3$, two ESIRK methods have fewer iterations than SIRK method when stepsize $h$ becomes smaller. The choice $[0, 1/2, 1]$ gives faster convergence than the others (cf. Table 4.6). Therefore, effective order methods seem to have an advantage over their corresponding SIRK methods in the convergence of the Newton iterations.

Figure 4.3: Order-2 SIRK: $- -$, $[7 - 4\sqrt{2}, 1]$: $+$, $[0, 1]$: $\cdots$, order-3 SIRK: $[7 - 4\sqrt{2}, 1]$: $+$, $[0, 1]$: $\cdots$, order-3 SIRK: $- -$, $[0, 1/2, 1]$: $\times$, $[0, 1/5, 1]$: $\circ$
### Table 4.4: The global error for the Kaps problem using $s = 2, 3$ SIRK and ESIRK methods with constant stepsize

<table>
<thead>
<tr>
<th></th>
<th>$s = 2$</th>
<th>$s = 3$</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>stepsize $h$</strong></td>
<td>$5.0000 \times 10^{-1}$</td>
<td>$5.0000 \times 10^{-1}$</td>
</tr>
<tr>
<td>SIRK</td>
<td>$4.6513 \times 10^{-6}$</td>
<td>$1.1433 \times 10^{-6}$</td>
</tr>
<tr>
<td>ESIRK$\left[\frac{7-4\sqrt{2}}{3}, 1\right]$</td>
<td>$4.4323 \times 10^{-6}$</td>
<td>$1.2216 \times 10^{-6}$</td>
</tr>
<tr>
<td>ESIRK$[0, 1]$</td>
<td>$4.9157 \times 10^{-6}$</td>
<td>$1.1784 \times 10^{-6}$</td>
</tr>
<tr>
<td><strong>stepsize $h$</strong></td>
<td>$6.2500 \times 10^{-2}$</td>
<td>$6.2500 \times 10^{-2}$</td>
</tr>
<tr>
<td>SIRK</td>
<td>$7.2134 \times 10^{-8}$</td>
<td>$7.7721 \times 10^{-9}$</td>
</tr>
<tr>
<td>ESIRK$\left[\frac{7-4\sqrt{2}}{3}, 1\right]$</td>
<td>$7.1647 \times 10^{-8}$</td>
<td>$7.2429 \times 10^{-9}$</td>
</tr>
<tr>
<td>ESIRK$[0, 1]$</td>
<td>$7.2728 \times 10^{-8}$</td>
<td>$7.2879 \times 10^{-9}$</td>
</tr>
</tbody>
</table>
### 4.4. ESIRK METHODS

Table 4.5: The average iteration number for the Kaps problem using $s = 2$ SIRK and ESIRK methods, $tol$: tolerance for stopping the iteration.

<table>
<thead>
<tr>
<th>stepsize $h$</th>
<th>$5.0000 \times 10^{-1}$</th>
<th>$2.5000 \times 10^{-1}$</th>
<th>$1.2500 \times 10^{-1}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$tol = 10^{-4}$</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SIRK</td>
<td>2.2000</td>
<td>2.0500</td>
<td>1.9500</td>
</tr>
<tr>
<td>ESIRK$\left[\frac{7-4\sqrt{2}}{3}, 1\right]$</td>
<td>2.1000</td>
<td>2.0500</td>
<td>1.9250</td>
</tr>
<tr>
<td>ESIRK$[0, 1]$</td>
<td>2.2000</td>
<td>2.1000</td>
<td>1.9750</td>
</tr>
<tr>
<td>$tol = 10^{-6}$</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SIRK</td>
<td>2.8000</td>
<td>2.6500</td>
<td>2.5250</td>
</tr>
<tr>
<td>ESIRK$\left[\frac{7-4\sqrt{2}}{3}, 1\right]$</td>
<td>2.7000</td>
<td>2.6000</td>
<td>2.4750</td>
</tr>
<tr>
<td>ESIRK$[0, 1]$</td>
<td>2.8000</td>
<td>2.6500</td>
<td>2.5250</td>
</tr>
<tr>
<td>stepsize $h$</td>
<td>$6.2500 \times 10^{-2}$</td>
<td>$3.1250 \times 10^{-2}$</td>
<td>$1.5625 \times 10^{-2}$</td>
</tr>
<tr>
<td>$tol = 10^{-4}$</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SIRK</td>
<td>1.8125</td>
<td>1.6812</td>
<td>1.5406</td>
</tr>
<tr>
<td>ESIRK$\left[\frac{7-4\sqrt{2}}{3}, 1\right]$</td>
<td>1.7875</td>
<td>1.6500</td>
<td>1.5125</td>
</tr>
<tr>
<td>ESIRK$[0, 1]$</td>
<td>1.8375</td>
<td>1.7000</td>
<td>1.5594</td>
</tr>
<tr>
<td>$tol = 10^{-6}$</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SIRK</td>
<td>2.4125</td>
<td>2.3375</td>
<td>2.2344</td>
</tr>
<tr>
<td>ESIRK$\left[\frac{7-4\sqrt{2}}{3}, 1\right]$</td>
<td>2.3875</td>
<td>2.3250</td>
<td>2.2000</td>
</tr>
<tr>
<td>ESIRK$[0, 1]$</td>
<td>2.4125</td>
<td>2.3500</td>
<td>2.2500</td>
</tr>
</tbody>
</table>
CHAPTER 4. THE EFFECTIVE ORDER OF SIRK METHODS

Table 4.6: The average iteration number for the Kaps problem using $s = 3$ SIRK and ESIRK methods, $tol$: tolerance for stopping the iteration.

<table>
<thead>
<tr>
<th>stepsize $h$</th>
<th>$5.0000 \times 10^{-1}$</th>
<th>$2.5000 \times 10^{-1}$</th>
<th>$1.2500 \times 10^{-1}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$tol = 10^{-6}$</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SIRK</td>
<td>2.9000</td>
<td>2.8000</td>
<td>2.7000</td>
</tr>
<tr>
<td>ESIRK$[0, \frac{1}{5}, 1]$</td>
<td>2.9000</td>
<td>2.8000</td>
<td>2.6500</td>
</tr>
<tr>
<td>ESIRK$[0, \frac{1}{2}, 1]$</td>
<td>2.9000</td>
<td>2.7500</td>
<td>2.6500</td>
</tr>
<tr>
<td>$tol = 10^{-8}$</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SIRK</td>
<td>3.4000</td>
<td>3.3000</td>
<td>3.1000</td>
</tr>
<tr>
<td>ESIRK$[0, \frac{1}{5}, 1]$</td>
<td>3.4000</td>
<td>3.3000</td>
<td>3.0500</td>
</tr>
<tr>
<td>ESIRK$[0, \frac{1}{2}, 1]$</td>
<td>3.4000</td>
<td>3.2500</td>
<td>3.0250</td>
</tr>
<tr>
<td>$tol = 10^{-2}$</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SIRK</td>
<td>6.2500 $\times 10^{-2}$</td>
<td>3.1250 $\times 10^{-2}$</td>
<td>1.5625 $\times 10^{-2}$</td>
</tr>
<tr>
<td>$tol = 10^{-6}$</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SIRK</td>
<td>2.5750</td>
<td>2.4375</td>
<td>2.3719</td>
</tr>
<tr>
<td>ESIRK$[0, \frac{1}{5}, 1]$</td>
<td>2.5125</td>
<td>2.4062</td>
<td>2.3375</td>
</tr>
<tr>
<td>ESIRK$[0, \frac{1}{2}, 1]$</td>
<td>2.5125</td>
<td>2.4000</td>
<td>2.3312</td>
</tr>
<tr>
<td>$tol = 10^{-8}$</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SIRK</td>
<td>2.9625</td>
<td>2.8250</td>
<td>2.6875</td>
</tr>
<tr>
<td>ESIRK$[0, \frac{1}{5}, 1]$</td>
<td>2.9000</td>
<td>2.7625</td>
<td>2.6156</td>
</tr>
<tr>
<td>ESIRK$[0, \frac{1}{2}, 1]$</td>
<td>2.8875</td>
<td>2.7500</td>
<td>2.6031</td>
</tr>
</tbody>
</table>
4.5 Variable stepsize for ESIRK methods

In applying effective order to singly-implicit methods, it is obvious that the stepsize chosen is crucial to the starting method and to the method itself. Restricting attention to constant stepsize is not appropriate because of the many problems for which numerical behaviour change over the region of integration. However, serious difficulties arise when variable stepsize is used for effective order methods. This partly explains why effective order has been ignored for 27 years!

Recall that an \( s \)-stage ESIRK method of order \( s \) (denoted by the mapping \( \phi_h \)) has to satisfy

\[
B_{s,h}(\alpha, y(x_n)) - \phi_h(B_{s,h}(\alpha, y(x_{n-1}))) = O(h^{s+1}),
\]

where \( B_{s,h}(\alpha, y(x_{n-1})) \) is the associated Butchered solution. In interpreting (4.44), it is taken for granted that the same stepsize \( h \) is used for

1. the perturbation of the incoming approximation \( B_{s,h}(\alpha, y(x_{n-1})) \),
2. the step represented by \( \phi_h \),
3. the stepsize expected in the following step.

\[
B_{s,h}(\alpha, y(x_{n-1})) = y(x_{n-1}) + \alpha_1 h y'(x_{n-1}) + \cdots + \alpha_s h^s y^{(s)}(x_{n-1})
\]

\[
B_{s,rh}(\alpha, y(x_{n-1} + h)) = y(x_{n-1} + h) + \alpha_1 r h y'(x_{n-1} + h) + \cdots + \alpha_s r^s h^s y^{(s)}(x_{n-1} + h)
\]

Figure 4.5: Variable stepsize scheme for ESIRK methods
CHAPTER 4. THE EFFECTIVE ORDER OF SIRK METHODS

In order to have a new stepsize (say \( rh \)) for the next step, we need to have an approximation to the Butchered solution \( B_{s,rh}(\alpha, y(x_{n-1} + h)) \) at the end of the current step. In other words, suppose the stepsize used for the present step is \( h \), and at the end of this step we decide to choose the stepsize for the next step to be \( rh \), then we have to scale \( h \) to \( rh \) in the outgoing perturbation (cf. Figure 4.5). Below we give a second-order method to explain how to achieve this goal.

Example 4.13 For an order-2 ESIRK method with the perturbation \( \alpha_1 = \frac{-3 + 2\sqrt{2}}{2} \), \( \alpha_2 = \frac{3 - 2\sqrt{2}}{4} \) and the corresponding tableau

<table>
<thead>
<tr>
<th></th>
<th>( \frac{9 - 6\sqrt{2}}{4} )</th>
<th>( \frac{-3 + 2\sqrt{2}}{4} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>( \frac{11 - 6\sqrt{2}}{4} )</td>
<td>( \frac{-1 + 2\sqrt{2}}{4} )</td>
</tr>
<tr>
<td>1</td>
<td>2 - ( \sqrt{2} )</td>
<td>-1 + ( \sqrt{2} )</td>
</tr>
</tbody>
</table>

the input value at the \( n \)-th step is an approximation to the Butchered solution

\[
B_{s,h}(\alpha, y(x_{n-1})) = y(x_{n-1}) + \alpha_1 h y'(x_{n-1}) + \alpha_2 h^2 y''(x_{n-1}).
\]

After this step, if the stepsize selected for the next step is \( rh \), we aim to produce the output value as an approximation to the Butchered solution

\[
B_{s,rh}(\alpha, y(x_{n-1} + h)) = y(x_{n-1} + h) + \alpha_1 rh y'(x_{n-1} + h) + \alpha_2 r^2 h^2 y''(x_{n-1} + h).
\]

It is clear that we have to approximate \( B_{s,rh}(\alpha, y(x_{n-1} + h)) \) by the use of the derivatives \( hf(Y_1), hf(Y_2) \). First, we approximate \( y(x_{n-1} + h) \) using the Taylor series, then it is easy to see that

\[
y(x_{n-1} + h) = y(x_{n-1}) + hy'(x_{n-1}) + \frac{h^2}{2} y''(x_{n-1}) + O(h^3).
\]

From the \( C(2) \) condition, we have

\[
\begin{align*}
hf(Y_1) &= hy'(x_{n-1}) + c_1 h^2 y''(x_{n-1}) + O(h^3) \\
&= hy'(x_{n-1}) + O(h^3), \\
\end{align*}
\]

\[
\begin{align*}
hf(Y_2) &= hy'(x_{n-1}) + c_2 h^2 y''(x_{n-1}) + O(h^3) \\
&= hy'(x_{n-1}) + h^2 y''(x_{n-1}) + O(h^3).
\end{align*}
\]

114
This implies that
\[
\begin{align*}
h y'(x_{n-1}) &= hf(Y_1) + O(h^3), \\
h^2 y''(x_{n-1}) &= hf(Y_2) - hf(Y_1) + O(h^3).
\end{align*}
\]
Hence we have the approximation to \(y(x_{n-1} + h)\),
\[
y(x_{n-1} + h) = y(x_{n-1}) + \frac{1}{2} hf(Y_1) + \frac{1}{2} hf(Y_2) + O(h^3).
\]
Similarly, we can approximate \(h y'(x_{n-1} + h), h^2 y''(x_{n-1} + h)\) as follows
\[
\begin{align*}
h y'(x_{n-1} + h) &= hf(Y_2) + O(h^3), \\
h^2 y''(x_{n-1} + h) &= -hf(Y_1) + hf(Y_2) + O(h^3).
\end{align*}
\]
In order to obtain \(B_{s,sh}(\alpha, y(x_{n-1} + h))\), we need to multiply \(h y'(x_{n-1} + h), h^2 y''(x_{n-1} + h)\) by \(r\alpha_1\) and \(r^2\alpha^2\) respectively. Therefore, we have
\[
\begin{align*}
B_{s,sh}(\alpha, y(x_{n-1} + h)) &= y(x_{n-1} + h) + \alpha_1 h y'(x_{n-1} + h) + \alpha_2 h^2 y''(x_{n-1} + h) \\
&= y(x_{n-1}) + \frac{1}{2} hf(Y_1) + \frac{1}{2} hf(Y_2) + \alpha_1 hf(Y_1) \\
&\quad + \alpha_2 r^2 (-hf(Y_1) + hf(Y_2)) + O(h^3) \\
&= y(x_{n-1}) + \left(\frac{1}{2} - \alpha_2 r^2\right) hf(Y_1) \\
&\quad + \left(\frac{1}{2} + \alpha_1 r + \alpha_2 r^2\right) hf(Y_2) + O(h^3).
\end{align*}
\]
Because the input value for the present step is not an approximation of \(y(x_{n-1})\) but of \(B_{s,h}(\alpha, y(x_{n-1}))\), by the relation (4.14),
\[
\begin{align*}
y(x_{n-1}) &= B_{s,h}(\alpha, y(x_{n-1})) - \alpha_1 h y'(x_{n-1}) - \alpha_2 h^2 y''(x_{n-1}) + O(h^3) \\
&= B_{s,h}(\alpha, y(x_{n-1})) + (\alpha_2 - \alpha_1) hf(Y_1) - \alpha_2 hf(Y_2) + O(h^3).
\end{align*}
\]
Thus, (4.45) can be written as follows
\[
B_{s,sh}(\alpha, y(x_{n-1} + h)) = B_{s,h}(\alpha, y(x_{n-1})) + \bar{b}_1(r) hf(Y_1) + \bar{b}_2(r) hf(Y_2),
\]
where
\[
\begin{align*}
\bar{b}_1(r) &= \frac{(2\sqrt{2} - 3) r^2 + 11 - 6\sqrt{2}}{4}, \\
\bar{b}_2(r) &= \frac{(3 - 2\sqrt{2}) r^2 + (4\sqrt{2} - 6) r + (2\sqrt{2} - 1)}{4}.
\end{align*}
\]
We note that \( \bar{b}_1(1), \bar{b}_2(1) \) are identical with the weights given by Theorem 4.10. From the above example, we know that we will not have any extra computational cost when we use such a stepsize changing scheme for ESIRK methods. The task now is to seek a formula for the weight function \( b^T(r) \), then a formula for \( B_{s,rh}(\alpha, y(x_{n-1} + h)) \). In fact, by the use of the Nordsieck vector mentioned in section 4.2, it is easy to derive such a formula. This is verified in the following result.

**Lemma 4.21** Let \( R \) be an \( s \times s \) matrix which has the form

\[
R = \begin{bmatrix}
  r & 0 & \ldots & 0 \\
  0 & r^2 & \ldots & 0 \\
  \vdots & \vdots & \ddots & \vdots \\
  0 & 0 & \ldots & r^s 
\end{bmatrix}
\]

For an \( s \)-stage ESIRK method, the Butchered solution \( B_{s,rh}(\alpha, y(x_{n-1} + h)) \) satisfies

\[
B_{s,rh}(\alpha, y(x_{n-1} + h)) = B_{s,h}(\alpha, y(x_{n-1})) + h \left( b^T(r) \otimes I_N \right) F, \tag{4.46}
\]

where \( b^T(r) = (\tau^T - \alpha^T + \alpha^T RE)W^{-1} \) and \( N \) is the dimension of the system.

**Proof:** Without loss of generality, let \( N = 1 \). Recalling the Nordsieck vector

\[
\zeta^{[n-1]} = \begin{bmatrix}
  hy'(x_{n-1}) \\
  h^2y''(x_{n-1}) \\
  \vdots \\
  h^s y^{(s)}(x_{n-1})
\end{bmatrix},
\]

we have

\[
B_{s,rh}(\alpha, y(x_{n-1} + h)) = B_{s,h}(R\alpha, y(x_{n-1} + h)) \\
= y(x_{n-1} + h) + \alpha^T R \zeta^{[n]} + O(h^{s+1}) \quad \text{by (4.14)} \\
= y(x_{n-1}) + \tau^T \zeta^{[n-1]} + \alpha^T R \zeta^{[n]} + O(h^{s+1}) \quad \text{by (4.15)}
\]
4.5. VARIABLE STEPSIZE FOR ESIRK METHODS

\[ y(x_{n-1}) + \tau^T \zeta^{[n-1]} + \alpha^T RE \zeta^{[n-1]} + O(h^{s+1}) \quad \text{by (4.16)} \]

\[ B_{s,h}(\alpha, y(x_{n-1})) - \alpha^T \zeta^{[n-1]} + \tau^T \zeta^{[n-1]} + \alpha^T RE \zeta^{[n-1]} + O(h^{s+1}) \quad \text{by (4.14)} \]

Then from (4.13),

\[ hF = (W \otimes I_N)\zeta^{[n-1]} + O(h^{s+1}), \]

and the result follows.

Lemma 4.21 gives a general formula for the weight function \( b^T(r) \) for ESIRK methods to achieve a stepsize change proposal. It simplifies the complex task discussed in Example 4.13. We give an example to illustrate Lemma 4.21.

**Example 4.14** We show how to obtain variable stepsize with a 3-stage ESIRK method:

| 0 | \( \frac{1}{2} - \frac{11}{2} \lambda + 15 \lambda^2 \) | \( \frac{5}{3} + 14 \lambda - 24 \lambda^2 \) | \( \frac{7}{3} - \frac{11}{2} \lambda + 9 \lambda^2 \)
| \( \frac{1}{2} \) | \( \frac{17}{24} - \frac{11}{2} \lambda + 15 \lambda^2 \) | \( \frac{4}{3} + 14 \lambda - 24 \lambda^2 \) | \( \frac{5}{8} - \frac{11}{2} \lambda + 9 \lambda^2 \)
| 1 | \( \frac{2}{3} - \frac{11}{2} \lambda + 15 \lambda^2 \) | \( \frac{5}{6} - \frac{11}{2} \lambda + 9 \lambda^2 \) | \( \frac{5}{6} - \frac{11}{2} \lambda + 9 \lambda^2 \)
| \( \frac{1}{3} - 3 \lambda + 12 \lambda^2 \) | \( \frac{2}{3} + 12 \lambda - 24 \lambda^2 \) | \( \frac{4}{3} - 9 \lambda + 12 \lambda^2 \)

where \( \lambda \approx 0.435866521508459 \) and \( \alpha_1 = \frac{1}{2} - 3 \lambda, \alpha_2 = \frac{1}{8} - \frac{3}{2} \lambda + 3 \lambda^2, \alpha_3 = \frac{1}{24} - \frac{1}{2} \lambda + \frac{3}{2} \lambda^2 - \lambda^3. \)

It is easy to find the associated matrices as follows.

\[ \tau^T = \begin{bmatrix} 1, \frac{1}{2!}, \frac{1}{3!} \end{bmatrix}, \quad R = \begin{bmatrix} r & 0 & 0 \\ 0 & r^2 & 0 \\ 0 & 0 & r^3 \end{bmatrix} \]

and

\[ E = \begin{bmatrix} 1 & 1 & \frac{1}{2!} \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{bmatrix}, \quad W^{-1} = \begin{bmatrix} 1 & 0 & 0 \\ -3 & 4 & -1 \\ 4 & -8 & 4 \end{bmatrix} \]
CHAPTER 4. THE EFFECTIVE ORDER OF SIRK METHODS

From Lemma 4.21,

\[ b^T(r) = (T - \alpha^T + \alpha^T RE)W^{-1} \]
\[ = \left( \left[ 1, \frac{1}{21}, \frac{1}{3!} \right] - [\alpha_1, \alpha_2, \alpha_3] + [\alpha_1, \alpha_2, \alpha_3] RE \right) W^{-1} \]
\[ = [b_1(r), b_2(r), b_3(r)], \]

where

\[ b_1(r) = \frac{1}{6} - \alpha_1 + (3 + r^2)\alpha_2 + (-4 + 4r^3)\alpha_3, \]
\[ b_2(r) = \frac{2}{3} + (-4 - 4r^2)\alpha_2 + (8 - 8r^3)\alpha_3, \]
\[ b_3(r) = \frac{1}{6} + r\alpha_1 + (1 + 3r^2)\alpha_2 + (-4 + 4r^3)\alpha_3. \]

We note that the stepsize changing scheme for ESIRK methods has the stepsize ratio \( r \) and the perturbation \( \alpha_i \) involved in the formula. Furthermore, after the \( n \)-th step, the output value is an approximation to \( B_{s,rh}(\alpha, y(x_{n-1} + h)) \) rather than to \( B_{s,h}(\alpha, y(x_{n-1} + h)) \). This means that the stability function for this step depends on the stepsize ratio and perturbations as well. In the next section, we focus on the influence of the local error and the stability properties caused by using variable stepsize in ESIRK methods.

4.6 Study of a systematic stepsize change pattern

As mentioned in the last section, the stepsize changing scheme for ESIRK methods is unnatural. It is interesting to examine whether there is any disadvantage when variable stepsize is used in these new methods. If the \( n \)-th step with stepsize \( h \) is followed by a step with stepsize \( rh \), then the stability function for this step is given by

\[ \tilde{R}(z, r) = \frac{y_n}{B_{s,h}(\alpha, y(x_{n-1})))}, \]

(4.47)
where \( y_n = B_{s,rh}(\alpha, y(x_{n-1}+h)) + O(h^{s+1}) \). If we apply an \( s \)-stage ESIRK method to the test equation \( y' = \lambda y \) and write \( z = h\lambda \), then from Theorem 4.19, we have

\[
B_{s,h}(\alpha, y(x_{n-1})) = y(x_{n-1})\phi(z),
\]

where

\[
\phi(z) = 1 + \alpha_1 z + \alpha_2 z^2 + \cdots + \alpha_s z^s.
\]

It can then be shown that

\[
B_{s,rh}(\alpha, y(x_{n-1} + h)) = y(x_{n-1} + h)\phi(rz) = y(x_{n-1})\exp(z)\phi(rz).
\]  

(4.48)

Suppose that \( y(x_{n-1}) \neq 0 \). Substituting (4.48) into (4.47) then yields

\[
\hat{R}(z, r)\phi(z) = \exp(z)\phi(rz) + O(z^{s+1}).
\]  

(4.49)

Because of the single-implicitness of \( A \), the following result can be proved.

**Theorem 4.22** For an \( s \)-stage ESIRK method, if the stepsize for the present step and the next step are \( h \) and \( rh \) respectively, then the stability function \( \hat{R}(z, r) \) for this step is

\[
\hat{R}(z, r) = \frac{P(z, r)}{(1 - \lambda z)^s},
\]

where

\[
P(z, r) = \frac{(1 - \lambda z)^s \exp(z)\phi(rz)}{\phi(z)} + O(z^{s+1}).
\]  

(4.50)

**Proof:** From (4.49), we have

\[
\hat{R}(z, r) = \frac{\exp(z)\phi(rz)}{\phi(z)} + O(z^{s+1}) = \frac{P(z, r)}{(1 - \lambda z)^s} + O(z^{s+1}).
\]

\( \square \)

Of course, we cannot use the stability function \( R(z, r) \) in Theorem 4.22 to study the error constant and the stability because the output value for this step is

\( B_{s,rh}(\alpha, y(x_{n-1} + h)) \) rather than \( B_{s,h}(\alpha, y(x_{n-1} + h)) \). For ESIRK methods, the perturbed solution flow for stepsize \( rh \) has a totally different behaviour from the
solution flow for steps size $h$ if $r \neq 1$. In order to study the truncation error and stability behaviour of ESIRK methods under variable stepsize, we need to go back to the solution flow for stepsize $h$ once we start changing stepsize from $h$ to $rh$. Therefore, in the following theoretical study, we will use a simple stepsize changing pattern, from $h$ to $rh$, then back to $h$. That is, $h, rh, h, rh, \ldots, h$. To prepare to go back to stepsize $h$, the output value at the second step is $B_{s,h}(\alpha, y(x_{n+1}))$ if the input value at $x_{n-1}$ is $B_{s,h}(\alpha, y(x_{n-1}))$ (cf. Figure 4.6).

![Figure 4.6: Stepsize changing pattern for studying stability function of variable stepsize ESIRK methods](image)

Because the stepsize ratio of changing stepsize (from $rh$ to $h$) is $\frac{1}{r}$, the stability function for this step will be $\hat{R}(rz, \frac{1}{r})$. Therefore, the stability function $\hat{R}_2(z,r)$ for these two steps is given by

$$\hat{R}_2(z,r) = \hat{R}(z,r)\hat{R}\left(rz, \frac{1}{r}\right) = \frac{P(z,r)}{(1-\lambda z)^2} \frac{P(rz, \frac{1}{r})}{(1-\lambda rz)^2}. \quad (4.51)$$

From (4.50) and (4.51), it is clear that $\hat{R}_2(z,r)$ depends on the ratio $r$ and the perturbations $\alpha$ (Because $\alpha$ determines $\phi(z)$). In other words, $\hat{R}_2(z,r)$ also depends on the choice of abscissae. The illustration of this two-step stability function is given in the following example.
Example 4.15 Take order-2 ESIRK methods with \( c = [c_1, 1]^T \) as an example. In this case, \( \lambda = 1 - \frac{\sqrt{2}}{2} \), and by (4.36) in Theorem 4.15, \( \phi(z) = 1 + \alpha_1 z + \alpha_2 z^2 \), where

\[
\alpha_1 = -\frac{3}{2} + \sqrt{2} + \frac{1}{2}, \quad \alpha_2 = \frac{1}{4}(-1 + c_1)(-3 + 2\sqrt{2} + c_1).
\]

By (4.50), we have the numerators \( P(z, r) \) and \( P(rz, \frac{1}{r}) \) of \( \tilde{R}_2(z, r) \) as follows.

\[
P(z, r) = \frac{(1 - \lambda z)^2 \exp(z) \phi(rz)}{\phi(z)} + O(z^3)
= 1 + \left( \frac{1}{2} - \frac{c_1}{2} - \frac{3r}{2} + \sqrt{2r} + \frac{c_1 r}{2} \right) z
+ \frac{1}{4}(-1 + c_1)(-3 + 2\sqrt{2} + c_1)(-1 + r)r^2 + O(z^3), \tag{4.52}
\]

\[
P \left( rz, \frac{1}{r} \right) = \frac{(1 - \lambda rz)^2 \exp(rz) \phi(z)}{\phi(rz)} + O(z^3)
= 1 + \left( \frac{-3}{2} - \frac{c_1 r}{2} + \frac{r}{2} + \sqrt{2} + \frac{c_1}{2} \right) z
+ \frac{1}{4}(-1 + c_1)(-3 + 2\sqrt{2} + c_1)(1 - r)r^2 + O(z^3). \tag{4.53}
\]

Therefore, by substituting (4.52), (4.53) into \( \tilde{R}_2(z, r) \), we obtain

\[
\tilde{R}_2(z, r) = \frac{P(z, r) P(rz, \frac{1}{r})}{(1 - \lambda z)^s (1 - \lambda rz)^s}
= 1 + (1 + r)z + \frac{(1 + r)^2}{2} z^2
+ \frac{1}{8} \left( f_1(c_1) + f_2(c_1)r + f_2(c_1)r^2 + f_1(c_1)r^3 \right) z^3 + O(z^4),
\]

where

\[
f_1(c_1) = -1 + 2\sqrt{2} - 7c_1 + 4\sqrt{2}c_1 + 5c_1^2 - 2\sqrt{2}c_1^2 - c_1^3,
\]

\[
f_2(c_1) = 1 + 2\sqrt{2} + 7c_1 - 4\sqrt{2}c_1 - 5c_1^2 + 2\sqrt{2}c_1^2 + c_1^3.
\]

It is clear that the error constant depends on \( r \) and \( c_1 \) because \( \tilde{C}_2 \) involves \( c_1 \)
and \( r \). If \( \tilde{C}_2(c_1, r) \) is the error constant for these two steps with the total stepsize \( (1 + r)h \), then \( \tilde{C}_2(c_1, r) \) satisfies

\[
\tilde{R}_2(z, r) = \exp(-(1 + r)z) + \tilde{C}_2(c_1, r)z^3 + O(z^4).
\]

121
CHAPTER 4. THE EFFECTIVE ORDER OF SIRK METHODS

If we let \( r = 1 \) (constant stepsize for two steps), then we find

\[
\widehat{C}_2 \approx 0.08088022904,
\]

which is just twice the error constant for the second order ESIRK method and is independent of the abscissae.

We give Mathematica subroutines "stabfun" and "twostep" in Appendix A.2 for finding the stability function in Theorem 4.22 and the stability function over two steps shown in (4.51). Using this special stepsize changing pattern assumption, we discuss the local truncation error as well as the stability property of ESIRK methods with variable stepsize in the following two subsections.

4.6.1 Local truncation error

In this subsection, we introduce the study of the error ratio (variable stepsize/constant stepsize) obtained by the two-step stability function \( \widehat{R}_2(z, r) \). By using the error ratio, we examine whether there is any loss because of the unnatural stepsize changing formulae used in the ESIRK methods. From Theorem 4.19, we have

\[
y(x_{n-1}) \phi(z) \widehat{R}_2(z, r) = y(x_{n-1}) \exp((1 + r)z) \phi(z) + O(z^{s+1}).
\]

Hence, the error constant \( \widehat{C}_2(r) \) for these two steps can be derived using

\[
\widehat{R}_2(z, r) \exp(-(1 + r)z) = 1 + \widehat{C}_2(r) z^{s+1} + O(z^{s+2}).
\]

Using (4.54) it is easy to see that \( \widehat{C}_2(r) \) is a symmetric polynomial in the sense that

\[
\widehat{C}_2(r) = \widehat{C}_2 \left( \frac{1}{r} \right) r^{s+1}.
\]

It is natural that we choose \( h_c = \frac{(1+r)}{2} h \) and compare the error introduced by the constant stepsize \( h_c \) over two steps and by variable stepsize change from \( h \) to \( rh \). Since \( \widehat{C}_2(r) \) is the error constant for these two steps with total stepsize \( h + rh \), \( \widehat{C}_2(1) \) is the error constant for these two steps with constant stepsize \( h \). Therefore,
by (4.55), the error constant with constant stepsize $h_\varepsilon$ is $\hat{C}_2(1)(\frac{1+r}{2})^{s+1}$. We can then use the error ratio

$$\varepsilon(r) = \frac{\hat{C}_2(r)}{\hat{C}_2(1)(\frac{1+r}{2})^{s+1}}, \quad (4.56)$$

to compare the local error behaviour between variable stepsize and constant stepsize. If $\varepsilon(r) < 1$, then we might have some benefit from variable stepsize. However, if $\varepsilon(r) > 1$, then the variable stepsize produces a loss of performance. The Mathematica subroutines "errorate" given in Appendix A.3.1 are used to investigate this error ratio. Because of the symmetry of the error constant $\hat{C}_2(1)$, we find

$$\varepsilon\left(\frac{1}{r}\right) = \frac{\hat{C}_2\left(\frac{1}{r}\right)}{\hat{C}_2(1)(\frac{1+r}{2})^{s+1}} = \frac{\hat{C}_2(r)}{\hat{C}_2(1)(\frac{1+r}{2})^{s+1}} = \varepsilon(r). \quad (4.57)$$

When we differentiate (4.57) with respect to $r$, we have

$$\frac{\partial}{\partial r} \varepsilon(1) = 0,$$

which means $\varepsilon(1)$ is a local extreme value of $\varepsilon(r)$. Because $\varepsilon(1)$ is independent of the choice of abscissae, we can consider the ESIRK methods with different stages. As the order gets higher, there are more free parameters in $\varepsilon(r)$. The assumption $c_s = 1$ will always be made.

In the case $s = 2$, there is only one extra parameter $c_1$ involved in $\varepsilon(r)$. We will write $\varepsilon(r, c_1)$ instead of $\varepsilon(r)$. From (4.54) and (4.56), we find

$$\varepsilon(r, c_1) = \frac{f_1(c_1) + f_2(c_1)r + f_1(c_1)r^2}{(-4 + 3\sqrt{2})(1 + r)^2},$$

where

$$f_1(c_1) = -7 + 6\sqrt{2} - 21c_1 + 12\sqrt{2}c_1 + 15c_1^2 - 6\sqrt{2}c_1^2 - 3c_1^3,$$

$$f_2(c_1) = -2 + 42c_1 - 24\sqrt{2}c_1 - 30c_1^2 + 12\sqrt{2}c_1^2 + 6c_1^3.$$

As discussed previously, $\varepsilon(r, c_1)$ has a critical point at $r = 1$. We can use the second derivative $\frac{\partial^2}{\partial r^2} \varepsilon(r, c_1)|_{r=1}$ to examine whether $\varepsilon(1, c_1)$ is a minimum or a maximum. It follows that

$$\frac{\partial^2}{\partial r^2} \varepsilon(r, c_1)|_{r=1} = \frac{1}{4}(6 + 2\sqrt{2} + (-12 - 15\sqrt{2})c_1$$

$$+ (24 + 21\sqrt{2})c_1^2 + (-12 - 9\sqrt{2})c_1^3),$$

123
and it is found that

$$\varepsilon(c_1) = \frac{\partial^2}{\partial r^2} \varepsilon(r, c_1)|_{r=1} > 0, \ \forall \ c_1 \in (-\infty, 1).$$

Therefore, $\varepsilon(r, c_1)$ has a local minimum at $(1, c_1)$ for all $c_1 \in (-1, 1)$. This also means that we cannot have any advantage from variable stepsize in the case of order 2. Furthermore, we can use the derivative $\frac{\partial}{\partial c_1} \varepsilon(c_1)$ to find the optimal choice for $c_1$. We have

$$\frac{\partial}{\partial c_1} \varepsilon(c_1) = 0 \Leftrightarrow c_1 = \frac{7 - 4\sqrt{2}}{3}, \quad c_1 = 1.$$

Hence, $\varepsilon(c_1)$ has the minimum value at $c_1 = \frac{7 - 4\sqrt{2}}{3} \approx 0.44715$. In other words, the choice $c_1 = \frac{7 - 4\sqrt{2}}{3}$ will give the most accurate result among $c_1 \in (-\infty, 1)$ under this stepsize changing pattern. The behaviour of $\varepsilon(c_1)$ for $c_1 \in (0, 1.5)$ is shown in Figure 4.7. It is also found that

$$\varepsilon(3 - 2\sqrt{2}) \geq \varepsilon(c_1), \quad \forall \ c_1 \in (3 - 2\sqrt{2}, 1),$$

and

$$\varepsilon(3 - 2\sqrt{2}) \leq \varepsilon(c_1), \quad \forall \ c_1 \in [0, 3 - 2\sqrt{2}).$$

Figure 4.7: $s = 2$, Graph of $\varepsilon(c_1) = \frac{\partial^2}{\partial r^2} \varepsilon(r, c_1)|_{r=1}$

124
The set \((3 - 2\sqrt{2}, 1)\) is the abscissae of SIRK method. We plot the error ratio 
\(\varepsilon(r, 3 - 2\sqrt{2}), \varepsilon(r, 0), \varepsilon(r, \frac{7-4\sqrt{2}}{3})\), which are functions of the stepsize ratio \(r\), in Figure 4.8. If \(c_1\) is chosen in \((0, 3 - 2\sqrt{2})\), then the curve \(\varepsilon(r, c_1)\) will appear in between the curves \(\varepsilon(r, 3 - 2\sqrt{2}), \varepsilon(r, 0)\), and if \(c_1\) is chosen in \((3 - 2\sqrt{2}, 1)\), the curve \(\varepsilon(r, c_1)\) will appear in between the curves \(\varepsilon(r, 3 - 2\sqrt{2}), \varepsilon(r, \frac{7-4\sqrt{2}}{3})\). We can see the best performance when changing stepsize is for \(c_1 = \frac{7-4\sqrt{2}}{3}\).

![Figure 4.8](image)

Figure 4.8: \(s = 2\), error ratio \(\varepsilon(r, c_1)\) of variable stepsize against constant stepsize

Therefore, we have the following conclusions for \(s = 2\).

(1) Constant stepsize is better than variable stepsize.

(2) For variable stepsize, a SIRK method is better than an ESIRK method when \(c_1 \in [0, 3 - 2\sqrt{2})\) while an ESIRK method is better than a SIRK method when \(c_1 \in (3 - 2\sqrt{2}, 1)\).

(3) For variable stepsize, \(c_1 = \frac{7-4\sqrt{2}}{3}\) is the optimal choice.

For convenience, from now on, we will use \((c_1, c_2, \ldots, c_s)\) to denote the abscissae for a specified method. For example: ESIRK method \((0, \frac{1}{2}, 1)\) means the ESIRK
CHAPTER 4. THE EFFECTIVE ORDER OF SIRK METHODS

method with abscissae $c_1 = 0, c_2 = \frac{1}{2}, c_3 = 1$. In the case of $s = 3$, there are $c_1, c_2$ involved in the error ratio. If we let $c_1 = 0$, we have

$$
\varepsilon(c_2) = \frac{\partial^2}{\partial r^2} \varepsilon(r, c_2)|_{r=1} \leq 0, \quad \forall \ c_2 \in I = (-1.003942, 0.436130),
$$

and,

$$
\frac{\partial}{\partial c_2} \varepsilon(c_2) = 0 \quad \text{if and only if} \quad c_2 = -0.461103, \ 1.20947, \ 2.193733.
$$

The plot of $\varepsilon(c_2)$ is shown in Figure 4.9.

Since $\varepsilon(c_2) \leq 0, \forall \ c_2 \in I$, $\varepsilon(r, c_2)$ has a local maximum at $(1, c_2)$. Thus, we have $\varepsilon(r, c_2) \leq \varepsilon(1, c_2) = 1$, for all $c_2 \in I$. In Figure 4.10, the error ratios $\varepsilon(r)$ for the SIRK method and ESIRK methods $(0, \frac{1}{2}, 1), (0, \frac{1}{5}, 1)$ are plotted as functions of $r$. It is clear that the SIRK method and ESIRK method $(0, \frac{1}{2}, 1)$ cannot have advantages from variable stepsize because $\varepsilon(r) \geq 1$. However, the ESIRK method $(0, \frac{1}{5}, 1)$ is better than the SIRK method. As shown in Figure 4.10, when $c_2$ is chosen to be $\frac{1}{5}$, the error ratio $\varepsilon(r) \leq 1$, for all $r \in (0, 2)$. Therefore, in this case, the local error is reduced by changing stepsize. We also note that all curves are very smooth. Furthermore, in Figure 4.9, $\frac{\partial^2}{\partial r^2} \varepsilon(r, c_2)|_{r=1}$ is negative if $c_2 \in I$, and

![Figure 4.9: $s = 3, c_1 = 0, y$-axis : $\varepsilon(c_2) = \frac{\partial^2}{\partial r^2} \varepsilon(r, c_2)|_{r=1}, x$-axis : $c_2$](image)

![Figure 4.10: $s = 3, error ratio \varepsilon(r, c)$ of variable stepsize/constant stepsize](image)
the minimum value is at \( c_2 = -0.461103 \). Therefore \((0, -0.461103, 1)\) is the best option in this case. If we let \( c_1 = -1 \), it can then be shown that

\[
\varepsilon(c_2) \leq 0, \quad \forall c_2 \in (-2.502257, 0.002599),
\]

and,

\[
\frac{\partial}{\partial c_2} \varepsilon(c_2) = 0 \quad \text{if and only if} \quad c_2 = -1.561489.
\]

The graph of \( \varepsilon(c_2) \) is given in Figure 4.11. In Figure 4.12, we plot the error ratio \( \varepsilon(r, c) \) for four ESIRK methods: \((0, 0.2, 1)\), \((-1, 0, 1)\), \((-1, -0.5, 1)\), \((-1, -1.5615, 1)\). We note that the error ratios \( \varepsilon(r, c) \) for the ESIRK methods \((-1, -0.5, 1)\), \((-1, -1.5615, 1)\) are less than 1 for most of the \( r \)-interval \((0, 2)\) except when \( r \) is near the origin; the curves increase quickly and become larger than 1. The method \((-1, -1.5615, 1)\) falls more steeply than the method \((-1, -0.5, 1)\) does and the \( r \)-interval for \( \varepsilon(r, c) \) less than 1 is slightly smaller than that for the method \((-1, -0.5, 1)\). In fact, if both \( c_1 \) and \( c_2 \) are chosen to be large negative numbers, the curve rises more sharply and the \( r \)-interval for \( \varepsilon(r, c) \) less than 1 becomes smaller. The error ratio \( \varepsilon(r, c) \) for method \((-1, 0, 1)\) is less than 1 only for \( r \in (0.844405, 1.18427) \). Compared with them, the curve of the method \((0, 0.2, 1)\)
is smoother and is less than 1 for all $r \in (0, 2)$. If we consider the error behaviour near $r = 1$, then the best choice of the abscissae is $(-1, -1.561489, 1)$.

Therefore, for the case of $s = 3$, a few conclusions can be made.

(1) It is easy to choose ESIRK methods which can benefit from variable stepsize. For example, if the abscissae are chosen to be $(0, c_2, 1)$ and $c_2 \in (-1.003943, 0.43613)$.

(2) For variable stepsize, the choice $(0, \frac{1}{2}, 1)$ for an ESIRK method is better than the SIRK method. If we consider the error behaviour when $r$ is close to 1, then the more negative the numbers $c_1, c_2$ are, the smaller $\varepsilon(c)$ is.

In a similar way, for $s > 3$, it is not difficult to find some ESIRK methods which have an advantage from the point of view of variable stepsize. Because the error behaviour is closely related to $r$, it is difficult to decide which choice of the abscissae is optimal.

![Figure 4.13: The error ratio $\varepsilon(r, c)$ for ESIRK methods $(0, \frac{1}{2}, 1), (-1, 0, 1)$. The curve $\varepsilon(r, c)$ for $(-1, 0, 1)$ is concave down when $r \in (0.844405, 1.18427)$ and is only smaller than the the curve of $(0, \frac{1}{2}, 1)$, when $r \in (0.65, 1.53846)$.

For example, in the case of $s = 3$, the choice $(-1, 0, 1)$ can have advantages for variable stepsize when $r \in (0.844405, 1.18427)$, whereas the choice $(0, \frac{1}{2}, 1)$ does
not have any advantage. However, the choice $(0, \frac{1}{2}, 1)$ is better than $(-1, 0, 1)$ when $r \not\in (0.65, 1.53846)$ (cf. Figure 4.13). Furthermore, we still need to take stability into account. Consequently, in addition to considering the ratio of variable stepsize and constant stepsize, it is relevant to examine whether the ESIRK methods are better than SIRK methods because of variable stepsize.

**Experiment 4.2** We present some numerical results for the Prothero-Robinson problem (2.5) using order-2 methods with the fixed stepsize pattern $(h, rh, h, rh, \ldots)$. We choose the SIRK method and ESIRK methods $(0, 1), (\frac{7-4\sqrt{2}}{3}, 1)$ to test the problem. The numerical results for constant stepsize together with variable stepsize are given in Figure 4.14. The stepsize changing ratio $r$ is chosen to be 2.8 for A-stability (The stability will be considered in more detail in the next subsection). If $h$ is small enough, then all methods have similar accuracy. Furthermore, constant stepsize has better performance than variable stepsize for all methods. For variable stepsize, the method $(\frac{7-4\sqrt{2}}{3}, 1)$ performs better than the others. These results confirm our expectations. Symbols indicating the specified method and the end point global error ratios for variable stepsize against constant stepsize are presented in Table 4.7.

![Figure 4.14: Order-2 SIRK, ESIRK methods (0, 1), (\frac{7-4\sqrt{2}}{3}, 1)](image)

We also use order 2, 3 SIRK, ESIRK methods with this stepsize changing pattern to solve the Kaps problem (3.9). For the fixed pattern of stepsize changing,
CHAPTER 4. THE EFFECTIVE ORDER OF SIRK METHODS

Table 4.7: Order 2,3 SIRK, ESIRK methods with stepsize changing pattern $h, rh, h, \ldots$

<table>
<thead>
<tr>
<th>$s$</th>
<th>Abscissae</th>
<th>Error ratio</th>
<th>Constant</th>
<th>Variable</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>SIRK</td>
<td>1.6742</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>2</td>
<td>(0, 1)</td>
<td>2.1610</td>
<td>- *</td>
<td>- +</td>
</tr>
<tr>
<td>2</td>
<td>$(2^{-4} \sqrt{2}, 1)$</td>
<td>1.4271</td>
<td>- x</td>
<td>- o</td>
</tr>
</tbody>
</table>

Symbols and global error ratio (variable/constant) in Figure 4.15, 4.16

<table>
<thead>
<tr>
<th>$s$</th>
<th>Abscissae</th>
<th>Error ratio</th>
<th>Constant</th>
<th>Variable</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>SIRK</td>
<td>1.6733</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>2</td>
<td>(0, 1)</td>
<td>2.1501</td>
<td>- *</td>
<td>- +</td>
</tr>
<tr>
<td>2</td>
<td>$(2^{-4} \sqrt{2}, 1)$</td>
<td>1.4393</td>
<td>- x</td>
<td>- o</td>
</tr>
<tr>
<td>3</td>
<td>SIRK</td>
<td>1.4075</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>3</td>
<td>(0, $\frac{1}{5}$, 1)</td>
<td>0.7658</td>
<td>- x</td>
<td>- o</td>
</tr>
<tr>
<td>3</td>
<td>(0, $\frac{1}{2}$, 1)</td>
<td>1.0580</td>
<td>- *</td>
<td>- +</td>
</tr>
</tbody>
</table>

The stepsize changing ratios are chosen to be 2.8, 1.7 for $s = 2$ and $s = 3$ respectively (for stability). In the case of $s = 2$, the results obtained have the same conclusion as for the Prothero-Robinson problem. Using constant stepsize is still better than using variable stepsize for all methods: SIRK, ESIRK (0, 1), $(2^{-4} \sqrt{2}, 1)$. The choice $(2^{-4} \sqrt{2}, 1)$ has the best performance for variable stepsize. For $s = 3$, SIRK and ESIRK (0, $\frac{1}{2}$, 1) do not have advantages for variable stepsize, but the choice (0, $\frac{1}{5}$, 1) performs better when changing stepsize. The curves derived from the global error against stepsize plot for these methods are given in Figure 4.15 and Figure 4.16. These results conclude our investigations about error ratio mentioned in the previous part of this section. The global error ratio (variable/constant) and the symbols used in Figure 4.15, 4.16 are shown in
4.6. STUDY OF A SYSTEMATIC STEPSIZE CHANGE PATTERN

Figure 4.15: Order-2 SIRK, ESIRK methods $(0, 1), (\frac{7-4\sqrt{2}}{3}, 1)$

Figure 4.16: Order-3 SIRK, ESIRK methods $(0, \frac{1}{2}, 1), (0, \frac{1}{5}, 1)$

Table 4.7.

From this study of the error ratio, it is encouraging to discover that the accuracy is closely related to the choice of abscissae but that the standard SIRK choice is not the best. ESIRK methods might have been expected to have some disadvantages because of the unnatural way stepsize is changed, but from the point of view of truncation errors, this is not the case. However, it is complicated to optimize the abscissae for $s \geq 3$. Even though the error ratios obtained using equal spacing in $[0, 1]$ are always greater than 1, these ESIRK methods still perform better than the associated classical SIRK methods ($s = 3, \ldots, 6, 8$). The question raised now is whether the choice of equal spacing for the abscissae in $[0, 1]$ is good enough for ESIRK methods? Are there other suitable choices? We will keep these questions in mind during the following discussion.

Similar to the error ratio, using the normalized error constant obtained from the stability function $\hat{R}_2(z, r)$, we will be able to study the local truncation error for ESIRK methods as well. Refer to (4.54), the stability function $\hat{R}_2(z, r)$ of these two steps with stepsize changing from $h$ to $rh$, then back to $h$, satisfies

$$\hat{R}_2(z, r) \exp(-(1 + r)z) = 1 + \hat{C}_2(r)z^{s+1} + O(z^{s+2}).$$

Using the error constant $\hat{C}_2(r)$ for these two steps, we can obtain the normalized
error constant for each step. If \( \hat{C} \) denotes the normalized error constant for each step, then the error constant for the stepsize change, \( h \) followed by \( rh \), is \( \hat{C}(1 + r^{s+1}) \). Therefore, \( \hat{C} \) is given by

\[
\hat{C} = \frac{\hat{C}_2(r)}{1 + r^{s+1}},
\]

(4.58)

We will write \( \hat{C}(r) \) instead of \( \hat{C} \). When the perturbation \( \alpha^T = 0 \), the stepsize changing formulae for ESIRK methods is independent of the stepsize changing ratio \( r \). Therefore, if the abscissae is chosen to be the same as for the SIRK methods, then the normalized error constant \( \hat{C}(r) \) is the error constant of the SIRK method. For ESIRK methods, the normalized error constant \( \hat{C}(r) \) is determined not only by the stepsize ratio \( r \), but also by the choice of the abscissae. It follows from (4.55) that

\[
\hat{C}(r) = \hat{C}\left(\frac{1}{r}\right),
\]

because of the symmetry of the \( \hat{C}_2(r) \). This also implies

\[
\frac{\partial}{\partial r} \hat{C}(1) = 0.
\]

Hence, \( \hat{C}(r) \) has a local extreme value at \( r = 1 \), and \( \hat{C}(1) \) is the error constant of the SIRK method. The Mathematica subroutine "nec" used to investigate this normalized error constant is given in Appendix A.3.2. In the following investigation, we will always assume \( c_s = 1 \).

From (4.56), (4.58), the error constant \( \hat{C}(r) \) has the same numerator as the error ratio \( \varepsilon(r) \) but has denominator \( 1 + r^{s+1} \). In the case of \( s = 2 \), we will have

\[
\hat{C}(r, c_1) = \frac{f_1(c_1) + f_2(c_1) r + f_1(c_1) r^2}{24(1 - r + r^2)},
\]

where

\[
\begin{align*}
f_1(c_1) & = -7 + 6\sqrt{2} - 21c_1 + 12\sqrt{2}c_1 + 15c_1^2 - 6\sqrt{2}c_1^2 - 3c_1^3, \\
f_2(c_1) & = -2 + 42c_1 - 24\sqrt{2}c_1 - 30c_1^2 + 12\sqrt{2}c_1^2 + 6c_1^3.
\end{align*}
\]

We can also use the second derivative \( \frac{\partial^2}{\partial r^2} \hat{C}(r, c_1)_{|r=1} \) to study the error behaviour. We have

\[
\frac{\partial^2}{\partial r^2} \hat{C}(r, c_1)_{|r=1} > 0, \quad \forall c_1 \in (0, 3 - 2\sqrt{2}),
\]

132
4.6. STUDY OF A SYSTEMATIC STEPSIZE CHANGE PATTERN

\[ \frac{\partial^2}{\partial r^2} \hat{C}(r, c_1)|_{r=1} < 0, \quad \forall c_1 \in (3 - 2\sqrt{2}, 1), \]

which means the error constant is smaller than that for the SIRK method when \( c_1 \in (3 - 2\sqrt{2}, 1) \). The graph of \( \frac{\partial^2}{\partial r^2} \hat{C}(r, c_1)|_{r=1} \) as a function of \( c_1 \) is given in Figure 4.17.

![Graph showing the second derivative of the normalized error constant \( \hat{C}(r, c_1)|_{r=1} \) for \( s = 2 \)](image)

Figure 4.17: Second derivative of the normalized error constant \( \hat{C}(r, c_1)|_{r=1} \) for \( s = 2 \)

Furthermore, we have

\[ \frac{\partial}{\partial c_1} \left( \frac{\partial^2}{\partial r^2} \hat{C}(r, c_1)|_{r=1} \right) = 0 \Leftrightarrow c_1 = \frac{7 - 4\sqrt{2}}{3}, \quad c_1 = 1. \]

Therefore, the second derivative \( \frac{\partial^2}{\partial r^2} \hat{C}(r, c_1)|_{r=1} \) has a local minimum value at \( c_1 = \frac{7 - 4\sqrt{2}}{3} \). For the error behaviour near \( r = 1 \), the optimal choice for \( c_1 \) is \( \frac{7 - 4\sqrt{2}}{3} \) which is the same as the one found using the error ratio \( \varepsilon(r) \). The normalized error constant \( \hat{C}(r) \) of the SIRK method and ESIRK methods \((0, 1), (0.1, 1), (\frac{7 - 4\sqrt{2}}{3}, 1), (0.25, 1)\) as a function of \( r \in (0, 2) \) is given in Figure 4.18. The line parallel to the \( x \)-axis is the error constant for the SIRK method. The choices \((0, 1)\) and \((0.1, 1)\) produce convex curves. The concave curves correspond to the choices \((0.25, 1), (\frac{7 - 4\sqrt{2}}{3}, 1)\). The reason that those curves are concave above or below the
error constant of the SIRK method is due to the second derivative \( \frac{\partial^2}{\partial c^2} \hat{C}(r, c_1)|_{r=1} \) being positive or negative. When \( c_1 = 0.25 \in (3 - 2\sqrt{2}, 1) \), the second derivative is negative. For all methods here, the curve for \( c_1 = \frac{7-4\sqrt{2}}{3} \) has the best performance with variable stepsize. We also note that even if the stepsize changing ratio \( r \) moves from 0 to 2, the error constant for \( c_1 = \frac{7-4\sqrt{2}}{3} \) is still inside the interval \((-0.04044, 0.04044)\) \((0.04044 \text{ is the error constant for SIRK})\) which means \( c_1 = \frac{7-4\sqrt{2}}{3} \) is also the optimal choice when \( r \in (0, 2) \). (cf. Figure 4.19).

When \( s \geq 3 \), because we have more unknowns \( c_i \) involved in \( \hat{C}(r) \), it is not easy to find a standard way to optimize the normalized error constant \( \hat{C}(r) \). However, the error behaviour when \( r \) is near 1 is crucial. The best way to do the optimization is to use the second derivative \( \frac{\partial^2}{\partial c^2} \hat{C}(r, c)|_{r=1} \), \( c = (c_1, c_2, \ldots, c_{s-1}) \).

For example, \( s = 2 \), the error constant is positive, the minimum of the second derivative \( \frac{\partial^2}{\partial c^2} \hat{C}(r, c_1)|_{r=1} \) is \( c_1 = \frac{7-4\sqrt{2}}{3} \) which is the optimal choice for an order-2 ESIRK method. We use this idea to seek the “optimal” choice of the abscissae for ESIRK methods. Mathematica programs “Secder”, “mnsort3” used to search for the optimum value can be found in Appendix A.3.2. We intend to search the minimum (or maximum, depending on the sign of the error constant) value
of \( \frac{\partial^2}{\partial r^2} \hat{C}(r, c)|_{r=1} \) within a chosen interval. For example, when \( s = 3 \), the error constant is negative \((-0.02589708465063)\), and we try to find the maximum value for \( \frac{\partial^2}{\partial r^2} \hat{C}(r, c_1, c_2)|_{r=1} \) when \( c_1, c_2 \) are chosen from \([-1, 0.9]\) (stepsize is 0.1). It can be shown that \( \frac{\partial^2}{\partial r^2} \hat{C}(r, c_1, c_2)|_{r=1} \) has a maximum value at \( c_1 = -1, c_2 = -0.9 \). In this case, in order to obtain a more accurate answer when searching for the maximum of \( \frac{\partial^2}{\partial r^2} \hat{C}(r, c_1, c_2)|_{r=1} \) with a chosen interval, we also use the “ContourPlot” available in Mathematica to examine the answer. When \( c_1, c_2 \) both tend to be more negative, the value of \( \frac{\partial^2}{\partial r^2} \hat{C}(r, c_1, c_2)|_{r=1} \) is increasing. The contour plot produced by Mathematica is shown in Figure 4.20. The colour becomes lighter when \( c_1, c_2 \) both tend to \(-2\). This means that the value of \( \frac{\partial^2}{\partial r^2} \hat{C}(r, c_1, c_2)|_{r=1} \) is getting larger. The curve \( \hat{C}(r) \) gets more concave when \( c_1, c_2 \) are both getting closer to \(-2\). From the point of view of truncation error, we may have the greatest advantage from variable stepsize.

![ContourPlot](image)

Figure 4.20: ContourPlot of \( \frac{\partial^2}{\partial r^2} \hat{C}(r, c_1, c_2)|_{r=1} \). \( \hat{C}(r) \) is the normalized error constant for \( s = 3 \) ESIRK, \( y \)-axis : \( c_2 \), \( x \)-axis : \( c_1 \).

Although we have verified the distribution area for the abscissae in minimizing (or maximizing) \( \frac{\partial^2}{\partial r^2} \hat{C}(r, c)|_{r=1} \), it does not mean that those abscissae chosen in this way have an overwhelming advantage. We can just be sure that they can perform

135
CHAPTER 4. THE EFFECTIVE ORDER OF SIRK METHODS

Figure 4.21: $s = 3$, Normalized error constants $\hat{C}(r)$ of SIRK, ESIRK $E_1 : (0, \frac{1}{2}, 1)$, $E_2 : (0, \frac{1}{5}, 1)$, $E_3 : (0, -\frac{1}{2}, 1)$, $E_5 : (-\frac{3}{2}, -1, 1)$, $E_4 : (-1, -\frac{9}{10}, 1)$

better when $r$ is very close to 1. Another message we can have is that we may be able to find some better choices for the abscissae than the choice of equal spacing in [0, 1] in the sense of the local truncation error. Hence, we need to know the error behaviour when the magnitude of $r$ changes moderately. We plot the error constant $\hat{C}(r)$ of the $s = 3$ SIRK and ESIRK methods $(0, \frac{1}{2}, 1)$, $(0, -\frac{1}{2}, 1)$, $(0, -\frac{1}{5}, 1)$, $(-1.5, -1, 1)$, $(-1, -0.9, 1)$ in Figure 4.21. The horizontal line is the error constant for the SIRK method. All ESIRK methods perform better when $r$ is near 1. The most concave curve is for the choice $(-1.5, -1, 1)$, then $(-1, -0.9, 1)$. This is because they both have a larger value of $\frac{\partial^2}{\partial r^2} \hat{C}(r, c)|_{r=1}$. But the curves for $(-1.5, -1, 1)$ and $(-1, -0.9, 1)$ exceed 0.025897 (absolute value of the error constant for SIRK) when $r > 1.5$ or $r < 0.65$. The curves of $(0, \frac{1}{2}, 1)$, $(0, \frac{1}{5}, 1)$, and $(0, -\frac{1}{2}, 1)$ are smoother than the other two and the error constants are always between the error bound $(-0.025897, 0.025897)$ when $r \in (0, 2)$. Hence, the choice of equal spacing $(0, \frac{1}{2}, 1)$ seems quite satisfactory, and the choices $(0, \frac{1}{5}, 1)$, $(0, -\frac{1}{2}, 1)$ are considerably better.
Table 4.8: $s = 2, 3$, Normalized error ratio (ESIRK/SIRK) for solving the Kaps problem

<table>
<thead>
<tr>
<th>$s = 2$</th>
<th>$(0, 1)$</th>
<th>$(\frac{7-4\sqrt{2}}{3}, 1)$</th>
<th>$s = 3$</th>
<th>$(0, \frac{1}{2}, 1)$</th>
<th>$(0, \frac{1}{5}, 1)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$tol$</td>
<td>Error ratio</td>
<td>Error ratio</td>
<td>$tol$</td>
<td>Error ratio</td>
<td>Error ratio</td>
</tr>
<tr>
<td>$10^{-2}$</td>
<td>1.1259</td>
<td>1.0647</td>
<td>$10^{-2}$</td>
<td>0.6632</td>
<td>0.3394</td>
</tr>
<tr>
<td>$10^{-3}$</td>
<td>1.0243</td>
<td>1.0243</td>
<td>$10^{-3}$</td>
<td>0.7000</td>
<td>0.4842</td>
</tr>
<tr>
<td>$10^{-4}$</td>
<td>1.0234</td>
<td>0.9600</td>
<td>$10^{-4}$</td>
<td>0.8300</td>
<td>0.6448</td>
</tr>
<tr>
<td>$10^{-5}$</td>
<td>1.0112</td>
<td>0.9499</td>
<td>$10^{-5}$</td>
<td>0.9421</td>
<td>0.8333</td>
</tr>
<tr>
<td>$10^{-6}$</td>
<td>0.9916</td>
<td>0.9762</td>
<td>$10^{-6}$</td>
<td>0.9631</td>
<td>0.9011</td>
</tr>
<tr>
<td>$10^{-7}$</td>
<td>1.0029</td>
<td>1.0011</td>
<td>$10^{-7}$</td>
<td>0.9504</td>
<td>0.9404</td>
</tr>
<tr>
<td>$10^{-8}$</td>
<td>-</td>
<td>-</td>
<td>$10^{-8}$</td>
<td>1.1719</td>
<td>0.9647</td>
</tr>
<tr>
<td>$10^{-9}$</td>
<td>-</td>
<td>-</td>
<td>$10^{-9}$</td>
<td>0.9826</td>
<td>0.9793</td>
</tr>
</tbody>
</table>

For implementation, it is not efficient and appropriate to use the fixed stepsize changing pattern for variable stepsize. We should use error estimation to control error. From now on, we will use the error estimation for error control in our experiments. There are several ways to achieve error estimation for ESIRK methods, we will discuss it in the next section. We now present some numerical results using order 2, 3 ESIRK methods and the SIRK method with similar error estimators.

**Experiment 4.3** First, we test the Kaps problem (3.9) with the order-2 SIRK and ESIRK methods $(0, 1), (\frac{7-4\sqrt{2}}{3}, 1)$, and with the order-3 SIRK and ESIRK methods $(0, \frac{1}{2}, 1), (0, \frac{1}{5}, 1)$. We give the normalized maximum global error ratios (ESIRK/SIRK) in Tables 4.8. For order 2, all choices seem to have similar performance. $(\frac{7-4\sqrt{2}}{3}, 1)$ is still better than the choice $(0, 1)$. For $s = 3$, it is clear that two ESIRK choices perform better than the SIRK method, especially the choice $(0, \frac{1}{5}, 1)$.

**Experiment 4.4** We present another numerical result for solving the stiff DETEST package [37]. Because the results obtained are quite similar, we just give
the results for testing $A_1$ (linear), $D_1$ (non-linear) problems. The stiff components of $A_1$ (4.59) happen in the third and the fourth equations. They have eigenvalues $-100, -90$ respectively.

**DETEST $A_1$ Linear with real eigenvalues**

\[
\begin{align*}
y'_1(x) &= -0.5y_1(x), & y_1(0) &= 1, \\
y'_2(x) &= -y_2(x), & y_2(0) &= 1, \\
y'_3(x) &= -100y_3(x), & y_3(0) &= 1, \\
y'_4(x) &= -90y_4(x), & y_4(0) &= 1,
\end{align*}
\]

(4.59) The other one is the non-linear system $D_1$ (4.60) (nuclear reactor theory: Liniger, Willoughby 1967 [51]) whose last two components have the varying eigenvalues from $-0.17$ to $-0.012$ and from $-60$ to $-11$.

**DETEST $D_1$ Non-linear with real eigenvalues**

\[
\begin{align*}
y'_1(x) &= 0.2(y_2(x) - y_1(x)), & y_1(0) &= 0, \\
y'_2(x) &= 10y_1(x) - (60 - 0.125y_3(x))y_2 + 0.125y_3, & y_2(0) &= 0, \\
y'_3(x) &= 1, & y_3(0) &= 0, \\
x_f &= 400, & h_0 &= 1.7 \times 10^{-2}.
\end{align*}
\]

(4.60) We choose the order-3 SIRK and ESIRK methods $(0, \frac{1}{2}, 1), (0, \frac{1}{5}, 1), (-6, 0, 1)$ in this test. The normalized maximum error ratios of an ESIRK method against the SIRK method are shown in Table 4.9. The choice $(-0.6, 0, 1)$ has better performance than the two sets $(0, \frac{1}{2}, 1), (0, \frac{1}{5}, 1)$. For both problems, all ESIRK methods perform better than the SIRK method. The choice $(0, \frac{1}{5}, 1)$ is still better than $(0, \frac{1}{2}, 1)$, and $(-0.6, 0, 1)$ is the best amongst them. We also note that ESIRK methods can have more advantages over the SIRK method when solving non-linear problems.
4.6. **STUDY OF A SYSTEMATIC STEPSIZE CHANGE PATTERN**

Table 4.9: Normalized error ratio (ESIRK/SIRK) using order-3 methods for DETEST $A_1, D_1$

<table>
<thead>
<tr>
<th>DETEST</th>
<th>$A_1$</th>
<th>$D_1$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$s = 3$</td>
<td>$(0, \frac{1}{2}, 1)$, $(0, \frac{1}{3}, 1)$, $(-0.6, 0, 1)$</td>
<td>$(0, \frac{1}{2}, 1)$, $(0, \frac{1}{3}, 1)$, $(-0.6, 0, 1)$</td>
</tr>
<tr>
<td>$tol$</td>
<td>ratio</td>
<td>ratio</td>
</tr>
<tr>
<td>$10^{-4}$</td>
<td>0.8893</td>
<td>0.6629</td>
</tr>
<tr>
<td>$10^{-5}$</td>
<td>0.9569</td>
<td>0.8905</td>
</tr>
<tr>
<td>$10^{-6}$</td>
<td>1.0123</td>
<td>0.9502</td>
</tr>
<tr>
<td>$10^{-7}$</td>
<td>1.0081</td>
<td>0.9950</td>
</tr>
</tbody>
</table>

Figure 4.22: $s = 4$, Normalized error constants $\hat{C}(r)$ of SIRK, ESIRK $E_1 : (0, \frac{1}{3}, \frac{2}{3}, 1)$, $E_2 : (-1, -\frac{1}{3}, 0, 1)$, $E_3 : (-1, -\frac{1}{2}, -\frac{1}{2}, 1)$, $E_4 : (-2, -1, 0, 1)$

For $s = 4$, we plot the error constant $\hat{C}(r)$ (function of $r$) for the SIRK and ESIRK methods $(0, \frac{1}{3}, \frac{2}{3}, 1)$, $(-1, -0.5, 0, 1)$, $(-1, -0.5, 0.5, 1)$, $(-2, -1, 0, 1)$ in Figure 4.22. We note that the error constant for the SIRK method is positive. In this case, the most convex curve is found to be given by $(-2, -1, 0, 1)$, but the curve quickly turns up and exceeds the error bound $2.7259 \times 10^{-2}$ when $r > 1.5$ or $r < 0.7$. The curve for the ESIRK method $(0, \frac{1}{3}, \frac{2}{3}, 1)$ is still smooth and inside the error bound. The two choices $(-1, -0.5, 0, 1)$, $(-1, -0.5, 0.5, 1)$
perform reasonably better than \((0, \frac{1}{3}, \frac{2}{3}, 1)\) when \(0.3 \leq r \leq 2\). Both their curves turn up when \(r\) is around 0.5.

Figure 4.23: Normalized error constants \(\tilde{C}(r)\) of order-5 SIRK and ESIRK methods \(E_1 : (0, \frac{1}{4}, \frac{2}{4}, \frac{3}{4}, 1)\), and \(E_2 : (-1, -\frac{1}{2}, 0, \frac{1}{2}, 1)\), and \(E_3 : (-2, -1, -\frac{1}{2}, 0, 1)\), and \(E_4 : (-3, -2, -1, 0, 1)\)

Figure 4.24: Normalized error constants \(\tilde{C}(r)\) of order-6 SIRK and ESIRK methods \(E_1 : (0, \frac{1}{5}, \frac{2}{5}, \frac{3}{5}, \frac{4}{5}, 1)\), and \(E_2 : (-1, -\frac{2}{3}, -\frac{1}{3}, \frac{1}{3}, \frac{2}{3}, 1)\), and \(E_3 : (-2, -\frac{3}{2}, -1, -\frac{1}{2}, 0, 1)\), and \(E_4 : (-4, -3, -2, -1, 0, 1)\)

For \(s = 5\), we also have positive error constant \(5.3005 \times 10^{-4}\) for the SIRK method. Figure 4.23 shows that except for the choice \((-3, -2, -1, 0, 1)\), the other curves are convex. Although \((-2, -1, -\frac{1}{2}, 0, 1)\) has the smallest second derivative amongst them, it becomes smaller than \(-5.3005 \times 10^{-4}\) when \(r < 0.75\) or \(r > 1.4\). The equally spaced choice \((0, \frac{1}{4}, \frac{2}{4}, \frac{3}{4}, 1)\) still performs well, the other equally spaced choice \((-1, -\frac{1}{2}, 0, \frac{1}{2}, 1)\) is also a good option. In Figure 4.24, we have a very similar graph to that for \(s = 5\) except that the error constant for the \(s = 6\) SIRK method is negative. Amongst these choices, when \(r\) is near 1, only the choice \((-4, -3, -2, -1, 0, 1)\) is worse than the SIRK method; the others perform better. \((-2, -\frac{1}{2}, -1, -\frac{1}{2}, 0, 1)\) has the largest second derivative but its curve exceed the error bound \(3.4135 \times 10^{-4}\) when \(r < 0.65\) or \(r > 1.65\). For a wider \(r\)-interval, the two choices of equal spacing in \((0, 1)\) and \((-1, 1)\) still have
better performance.

In the investigation of \( s = 8 \), we use the choices which are equally spaced in \((0, 1), (-1, 1), (-2, 1), (-3, 1)\) respectively. The error constant is \(2.682 \times 10^{-6}\). In Figure 4.25, all choices perform better than the SIRK method when \( r \) is close to 1. \((-3, -\frac{5}{2}, -2, -\frac{3}{2}, -1, -\frac{1}{2}, 0, 1)\) is the most convex curve, followed by \((-2, -\frac{5}{3}, -\frac{4}{3}, -\frac{2}{3}, -\frac{1}{3}, \frac{1}{3}, \frac{2}{3}, 1)\). Both their curves exceed the error bound after \( r < 0.55 \) or \( r > 1.75 \). The choice of equal spacing in \((0,1)\) and \((-1,1)\) still have satisfactory performance. The latter may have more benefit from changing stepsize when \( 0.5 \leq r \leq 2 \).

![Normalized error constants](image)

**Figure 4.25:** Normalized error constants \( \hat{C}(r) \) of order-8 SIRK, ESIRK methods

- \( E_1 : (0, \frac{1}{7}, \frac{2}{7}, \frac{3}{7}, \frac{4}{7}, \frac{5}{7}, \frac{6}{7}, 1) \),
- \( E_2 : (-1, -\frac{3}{4}, -\frac{2}{4}, -\frac{1}{4}, \frac{1}{4}, \frac{2}{4}, \frac{3}{4}, 1) \),
- \( E_3 : (-2, -\frac{5}{3}, -\frac{4}{3}, -\frac{2}{3}, -\frac{1}{3}, \frac{1}{3}, \frac{2}{3}, 1) \),
- \( E_4 : (-3, -\frac{5}{2}, -2, -\frac{3}{2}, -1, -\frac{1}{2}, 0, 1) \)

The study of the normalized error constant for ESIRK methods has shown that it is not difficult to find some ESIRK methods to be more accurate than their associated SIRK methods by appropriate choice of the abscissae. Although it is hard to decide which choice of abscissae is the optimal, we can still give some recommendations for choosing the abscissae in the cases \( s = 2, 3, \ldots, 6, 8 \). First, the choice of equal spacing in \([0, 1]\) is a reasonably good option. For \( s = 2 \), the
choice \( \left( \frac{7 - 4 \sqrt{2}}{3}, 1 \right) \) is the best option. For \( s = 3, 4, 5, 6, 8 \), the error performance considered should be for most of the \( r \)-interval \((0, 2)\) rather than only for the interval around \( r = 1 \). Therefore, for \( s = 3 \), equal spacing in \([-0.5, 1]\), and for \( s = 4, 5, 6, 8 \) equal spacing in \([-1, 1]\) are recommended. It is hoped that these suggested abscissae can also have reasonable stability for stiff problems. In the following discussion, we will focus on the stability properties of ESIRK methods with variable stepsize.

### 4.6.2 Stability considerations

ESIRK methods are designed for stiff problems. From the previous discussions, their performances in local error behaviour do not become worse when we use the special stepsize changing scheme for these methods. For the stability requirement, it is important to examine the stability properties of these methods.

Recalling the two-step stability function \( \widehat{R}_2(z, r) \), the stability study can be achieved by way of the \( E \)-polynomial mentioned in section 2.4. Because of the special way in which stepsize is changed for the ESIRK methods, \( \widehat{R}_2(z, r) \) depends on the stepsize changing ratio \( r \) and the abscissae. Therefore, the \( E \)-polynomial also depends on \( r \) and the abscissae. Suppose that \( \widehat{R}_2(z, r) \) has the form

\[
\widehat{R}_2(z, r) = \frac{P(z, r)}{Q(z, r)}.
\]

The \( E \)-polynomial \( \xi(y, r) \) of \( \widehat{R}_2(z, r) \) is defined as

\[
\xi(y, r) = Q(iy, r)Q(-iy, r) - P(iy, r)P(-iy, r).
\]

From (4.51),

\[
\widehat{R}_2(z, r) = \frac{P(z, r) \frac{P(z, \frac{1}{r})}{(1 - \lambda z)^s (1 - \lambda rz)^s}}{(1 - \lambda z)^s (1 - \lambda rz)^s}.
\]

By Theorem 4.22, \( \xi(y, r) \) can be found using

\[
\xi(y, r) = (1 + y^2 \lambda^2)^s (1 + y^2 r^2 \lambda^2)^s - P(iy, r)P(-iy, r)P(iy, r)P\left(-iy, \frac{1}{r}\right) P\left(iy, \frac{1}{r}\right).
\]

142
where $P(z, r)$ satisfies
\[
P(z, r) = \frac{(1 - \lambda z)^s \exp(z) \phi(r z)}{\phi(z)} + O(z^{s+1}).
\]

It is easy to see that $\xi(y, r)$ is an even polynomial of degree $4s$ which depends on $r$ and the abscissae. Thus, we can use the fact that an ESIRK method is A-stable if and only if
\[
\xi(y, r) \geq 0, \quad \forall y \in \mathbb{R},
\]
to find the relation between the stability properties and the ratio $r$ and different choices of abscissae for the ESIRK methods. We give an example with $s = 2$.

**Example 4.16** By Example 4.15, we have the numerators $P(z, r)$ and $P(rz, \frac{1}{r})$, where
\[
P(z, r) = 1 + \left( \frac{1}{2} - \frac{c_1}{2} - \frac{3r}{2} + \sqrt{2r} + \frac{c_1r}{2} \right) z + \frac{1}{4} (-1 + c_1) (-3 + 2\sqrt{2} + c_1) (-1 + r) rz^2 + O(z^3),
\]
\[
P(rz, \frac{1}{r}) = 1 + \left( \frac{-3}{2} - \frac{c_1r}{2} + \frac{r}{2} + \sqrt{2} + \frac{c_1}{2} \right) z + \frac{1}{4} (-1 + c_1) (-3 + 2\sqrt{2} + c_1) (1 - r) z^2 + O(z^3).
\]

Hence, the $E$-polynomial $\xi(y, r)$ is given by
\[
\xi(y, r) = \left( 1 - iy + \frac{\sqrt{2}}{2} iy \right)^2 \left( 1 - iry + \frac{\sqrt{2}}{2} iry \right)^2
\]
\[
- P(iy, r) P(-iy, r) P \left( iry, \frac{1}{r} \right) P \left( -iry, \frac{1}{r} \right).
\]

It is easy to see that $\xi(y, r)$ is a polynomial of degree 8. Even when $s = 2$, evaluating $\xi(y, r)$ is very tedious. For convenience, we assume $c_1 = 0$. Because $\tilde{R}_2(z, r)$ has the symmetric property, the coefficients of $y^i$, $i = 4, 6, 8$ are symmetric polynomials in $r$. Therefore, we have
\[
\xi(y, r) = p_1(r) y^4 + p_2(r) y^6 + p_3(r) y^8,
\]

143
CHAPTER 4. THE EFFECTIVE ORDER OF SIRK METHODS

where

\[ p_1(r) = \frac{1}{8} \left[ 20 - 14\sqrt{2} + (17 - 12\sqrt{2})r + (-6 + 4\sqrt{2})r^2 + (17 - 12\sqrt{2})r^3 \right. \]
\[ \left. + (20 - 14\sqrt{2})r^4 \right], \]

\[ p_2(r) = \frac{1}{64} \left[ -17 + 12\sqrt{2} + (34 - 24\sqrt{2})r + (809 - 572\sqrt{2})r^2 + (1516 - 1072\sqrt{2})r^3 \right. \]
\[ \left. + (809 - 572\sqrt{2})r^4 + (34 - 24\sqrt{2})r^5 + (-17 + 12\sqrt{2})r^6 \right], \]

\[ p_3(r) = \frac{1}{256} \left[ (-577 + 408\sqrt{2})r^2 + (2308 + 1632\sqrt{2})r^3 + (5770 - 4080\sqrt{2})r^4 \right. \]
\[ \left. + (2308 + 1632\sqrt{2})r^5 + (-577 + 408\sqrt{2})r^6 \right], \]

Therefore the ESIRK method \((0, 1)\) is A-stable if \(r\) satisfies

\[ p_1(r) \geq 0 \quad \text{and} \quad p_2(r) \geq 0 \quad \text{and} \quad p_3(r) \geq 0. \] (4.61)

we find

\[ p_1(r) \geq 0, \quad \forall r \in \mathbb{R}, \]

\[ p_2(r) \geq 0 \iff r \in [0.355047, 2.81653], \]

\[ p_3(r) \geq 0 \iff r \in [0.171573, 5.82843]. \]

Consequently, under the assumption of the stepsize changing pattern, the method \((0, 1)\) is A-stable if the stepsize changing ratio lies in the interval \(0.355047 < r < 2.81653\).

Note that (4.61) in the above example is not necessary. In the case \(s = 2\), we also find that \(r \in [0.355047, 5.82843]\) gives acceptable stability (also using \(p_1(r) \geq 0\) and \(p_3(r) \geq 0 \) and \(p_2(r) \leq 0\) and \((p_2^2(r) - 4p_1(r)p_3(r) \leq 0)\), we have additional interval \([2.81653, 5.82843]\) for A-stability).

In the following discussion, we will always assume \(c_s = 1\) for each \(s\). In the case of \(s = 2\), we have only \(r, c_1\) occurring in the \(E\)-polynomial \(\xi(y, r)\), and it is easier to study the stability for some specified \(c_1\) numerically. The \(r\)-interval for A-stability for some order 2, 3 ESIRK methods together with the SIRK methods are shown in Table 4.10. It is found that the worst choice of \(c_1\) is 0.471255 and when \(c_1\) is chosen closer to \(3 - 2\sqrt{2}\) or 1, the stability performance is better. For example, for
4.6. Study of a Systematic Stepsize Change Pattern

Table 4.10: $r$-intervals for A-stability for order 2, 3 ESIRK methods

<table>
<thead>
<tr>
<th>$s$</th>
<th>Abscissae</th>
<th>$r$-interval</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>$(3 - 2\sqrt{2}, 1)$ (SIRK)</td>
<td>$(0, \infty)$</td>
</tr>
<tr>
<td>2</td>
<td>$(0, 1)$</td>
<td>[0.355047, 2.81653]</td>
</tr>
<tr>
<td>2</td>
<td>$(0.171579, 1)$</td>
<td>[0.138999, 7.19429]</td>
</tr>
<tr>
<td>2</td>
<td>$(\frac{7-4\sqrt{2}}{3}, 1)$</td>
<td>[0.235576, 4.24491]</td>
</tr>
<tr>
<td>2</td>
<td>$(0.471255, 1)$</td>
<td>[0.236458, 4.22908]</td>
</tr>
<tr>
<td>3</td>
<td>$(2.7415165, 0.1812222, 1)$ (SIRK)</td>
<td>$(0, \infty)$</td>
</tr>
<tr>
<td>3</td>
<td>$(0, \frac{1}{2}, 1)$</td>
<td>[0.444787, 2.24827]</td>
</tr>
<tr>
<td>3</td>
<td>$(0, 0.43613, 1)$</td>
<td>[0.475459, 2.10307]</td>
</tr>
<tr>
<td>3</td>
<td>$(0, \frac{1}{5}, 1)$</td>
<td>[0.559114, 1.78855]</td>
</tr>
<tr>
<td>3</td>
<td>$(-\frac{1}{2}, 0, 1)$</td>
<td>[0.605993, 1.65018]</td>
</tr>
<tr>
<td>3</td>
<td>$(-1, 0, 1)$</td>
<td>[0.577686, 1.73105]</td>
</tr>
<tr>
<td>3</td>
<td>$(-1, -\frac{1}{2}, 1)$</td>
<td>[0.750625, 1.33222]</td>
</tr>
<tr>
<td>3</td>
<td>$(-1, -\frac{9}{10}, 1)$</td>
<td>[0.823996, 1.21360]</td>
</tr>
<tr>
<td>3</td>
<td>$(-1.5, -1, 1)$</td>
<td>[0.852326, 1.61230]</td>
</tr>
</tbody>
</table>

$c_1 = 0.171579$ in Table 4.10, we have a larger $r$-interval than the others and when $c_1 \geq 0.628274$ the stability interval will include $(0.2, 5)$. In the discussion of the local truncation error, the best choice $c_1 = \frac{7-4\sqrt{2}}{3}$ still has the quite promising $r$-interval $[0.235576, 4.24491]$. In the case $s = 3$, the stability property of the choice $(0, c_2, 1)$ improves with increasing $c_2$. For example, when $c_2 \geq 0.788764$ the stability interval includes $(0.25, 4)$. Both classical SIRK methods have unbounded $r$-intervals because their stability function in the changing stepsize situation is independent of $r$. We also note that when $c_1, c_2$ are becoming more negative, the $r$-interval becomes smaller. (cf. Table 4.10).
Table 4.11: $r$-intervals for A-stability for order 4, 5 ESIRK methods

<table>
<thead>
<tr>
<th>$s$</th>
<th>Abscissae</th>
<th>$r$-interval</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>$(0, \frac{1}{3}, \frac{2}{3}, 1)$</td>
<td>$[0.368532, 2.71347]$</td>
</tr>
<tr>
<td>4</td>
<td>$(-1, 0, 0.5, 1)$</td>
<td>$[0.378112, 2.64472]$</td>
</tr>
<tr>
<td>4</td>
<td>$(-1, -0.5, 0.5, 1)$</td>
<td>$[0.452341, 2.21072]$</td>
</tr>
<tr>
<td>4</td>
<td>$(-1, -0.5, 0, 1)$</td>
<td>$[0.401487, 2.49074]$</td>
</tr>
<tr>
<td>4</td>
<td>$(-1, -0.9, -0.8, 1)$</td>
<td>$[0.686243, 1.45721]$</td>
</tr>
<tr>
<td>4</td>
<td>$(-2, -1, 0, 1)$</td>
<td>$[0.680843, 1.46877]$</td>
</tr>
<tr>
<td>5</td>
<td>$(0, \frac{1}{4}, \frac{2}{4}, \frac{3}{4}, 1)$</td>
<td>$[0.707239, 1.41395]$</td>
</tr>
<tr>
<td>5</td>
<td>$(-1, -\frac{1}{2}, 0, \frac{1}{2}, 1)$</td>
<td>$[0.788534, 1.26818]$</td>
</tr>
<tr>
<td>5</td>
<td>$(-1, -0.9, -0.8, -0.7, 1)$</td>
<td>$[0.897066, 1.11475]$</td>
</tr>
<tr>
<td>5</td>
<td>$(-2, -1, -0.5, 0, 1)$</td>
<td>$[0.894334, 1.11816]$</td>
</tr>
<tr>
<td>5</td>
<td>$(-3, -2, -1, 0, 1)$</td>
<td>$[0.849423, 1.17730]$</td>
</tr>
</tbody>
</table>

When $s = 4, 5$, we still have large $r$-interval for stability. The $r$-interval of some ESIRK methods are shown in Table 4.11. As with the cases $s = 2, 3$, the abscissae chosen with equal spacing in $[0, 1]$ have the best performance. When $c_i$ decreases (become negative), the $r$-intervals become smaller, and the choice of equal spacing still has better performance than when the abscissae are chosen closer together. For example, when $s = 4$, the $r$-interval $[0.680843, 1.46877]$ for $(-2, -1, 0, 1)$ is larger than the $r$-interval $[0.686243, 1.45721]$ for $(-1, -0.9, -0.8, 1)$, and when $s = 5$, the $r$-intervals for $(-2, -1, -0.5, 0, 1), (-3, -2, -1, 0, 1)$ are larger than the $r$-interval for $(-1, -0.9, -0.8, -0.7, 1)$. For the SIRK methods, they have the unbounded $r$-interval $(0, \infty)$, and for conciseness this case is omitted from the table.

In Table 4.12, when $s = 6, 8$, amongst the ESIRK methods listed, those with equally spaced abscissae in $[0, 1]$ and in $[-1, 1]$ still have larger $r$-intervals than...
Table 4.12: $r$-intervals for A-stability for order 6, 8 ESIRK methods

<table>
<thead>
<tr>
<th>$s$</th>
<th>Abscissae</th>
<th>$r$-interval</th>
</tr>
</thead>
<tbody>
<tr>
<td>6</td>
<td>$(0, \frac{1}{5}, \frac{2}{5}, \frac{3}{5}, \frac{4}{5}, 1)$</td>
<td>[0.642007, 1.55762]</td>
</tr>
<tr>
<td>6</td>
<td>$(-1, -\frac{2}{3}, -\frac{1}{3}, \frac{1}{3}, \frac{2}{3}, 1)$</td>
<td>[0.743688, 1.34465]</td>
</tr>
<tr>
<td>6</td>
<td>$(-1, -\frac{9}{10}, -\frac{8}{10}, -\frac{7}{10}, -\frac{6}{10}, 1)$</td>
<td>[0.858922, 1.16425]</td>
</tr>
<tr>
<td>6</td>
<td>$(-2, -\frac{3}{2}, -1, -\frac{1}{2}, 0, 1)$</td>
<td>[0.840240, 1.18901]</td>
</tr>
<tr>
<td>6</td>
<td>$(-4, -3, -2, -1, 0, 1)$</td>
<td>$r = 1$</td>
</tr>
<tr>
<td>8</td>
<td>$(0, \frac{1}{7}, \frac{2}{7}, \frac{3}{7}, \frac{4}{7}, \frac{5}{7}, \frac{6}{7}, 1)$</td>
<td>[0.824068, 1.21349]</td>
</tr>
<tr>
<td>8</td>
<td>$(-1, -\frac{3}{4}, -\frac{2}{4}, -\frac{1}{4}, \frac{1}{4}, \frac{2}{4}, \frac{3}{4}, 1)$</td>
<td>[0.886894, 1.12753]</td>
</tr>
<tr>
<td>8</td>
<td>$(-1, -\frac{9}{10}, -\frac{8}{10}, -\frac{7}{10}, -\frac{6}{10}, -\frac{5}{10}, -\frac{4}{10}, 1)$</td>
<td>[0.930437, 1.07456]</td>
</tr>
<tr>
<td>8</td>
<td>$(-2, -\frac{5}{3}, -\frac{4}{3}, -\frac{3}{3}, -\frac{2}{3}, -\frac{1}{3}, \frac{2}{3}, \frac{1}{3}, 1)$</td>
<td>$r = 1$</td>
</tr>
<tr>
<td>8</td>
<td>$(-3, -\frac{5}{2}, -2, -\frac{3}{2}, -1, -\frac{1}{2}, 0, 1)$</td>
<td>$r = 1$</td>
</tr>
</tbody>
</table>

the other choices. It is noted that when the abscissae are chosen to be smaller (negative), we cannot obtain an $r$-interval except the single point 1. This is due to the fact that their $E$-polynomials $\xi(y, r)$ oscillate between positive and negative values in very small $r$-intervals. In fact, in these cases, when $r$ is very close to 1, the negative values $\xi(y, r)$ are small (say smaller than $10^{-12}$) enough to be ignored.

Even though the classical SIRK methods allow wide variation in the stepsize without changing stability properties, the behaviour of ESIRK methods with alternative choices of the abscissae is still quite acceptable. Furthermore, the stability of the suggested abscissae in the previous study is satisfactory. The choice of equal spacing in $[0, 1]$ has good behaviour for both truncation error and stability. It does not seem to have any disadvantages when the abscissae are chosen in this way. If one wants to have more accurate ESIRK methods with a conservative code design (the bound for allowed stepsize change ratio is from
CHAPTER 4. THE EFFECTIVE ORDER OF SIRK METHODS

0.5 to 2), then, there are many options for choosing the abscissae which can be considered. We summarize these choices of the abscissae for ESIRK methods as follows.

(1) For \( s = 2 \), \( c_1 = \frac{7-4\sqrt{2}}{3} \).

(2) For \( s = 3 \), \( c_1 = -0.5, c_2 = 0 \).

(3) For \( s = 4 \), \( c_1 = -1, c_2 = -0.5, c_3 = 0.5 \).

(4) For \( s \geq 5 \), abscissae chosen with equal spacing in \([-1, 1]\).

These choices using equal spacing in \([0, 1]\) and the recommended abscissae together with their \( r \)-interval for stability for \( s = 2, 3, \ldots, 6, 8 \) are shown in Table 4.13. Of course, the abscissae shown in Table 4.13 are not the only good choice. An alternative such as \([-1, -\frac{1}{2}, 0, 1]\) is also a good option for \( s = 4 \).

**Experiment 4.5** In an attempt to verify the previous investigations, we propose to choose some second order and third order ESIRK methods to solve the stiff DETEST package ([37] 1975). We remark that the bound of stepsize changing ratio for selecting new stepsize is set to be \((0.5, 2)\). The Jacobian is updated for every step. The starting value for the Newton iteration is the output value of the previous step. The tolerance for stopping the iteration is set to be 0.1\textit{tol}, where \( \textit{tol} \) is the user-defined tolerance. For all chosen methods, we use the same strategy for controlling the local error and convergence. It is found that the numerical results obtained for all problems are consistent with our theoretical investigations. For illustration, we only show part of the results in Table 4.14 for the second-order methods. In Table 4.14, the expected accuracy tables for each problem solved by each method is produced by the DETEST package. In these experiments, the time taken by each method is approximately proportional to the number of function evaluations. Therefore we only show the number of function evaluations and number of steps in the tables. The ESIRK method \((\frac{7-4\sqrt{2}}{3}, 1)\) has overall better results compared with the other two methods, and except for high accuracy, the SIRK method seems to be better than the method \((0, 1)\) for most cases.

148
Table 4.13: Proper abscissae for ESIRK methods

<table>
<thead>
<tr>
<th>s</th>
<th>Abscissae</th>
<th>r-interval</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>(0, 1)</td>
<td>[0.355047, 2.81653]</td>
</tr>
<tr>
<td>3</td>
<td>(0, \frac{1}{2}, 1)</td>
<td>[0.444787, 2.24827]</td>
</tr>
<tr>
<td>4</td>
<td>(0, \frac{1}{3}, \frac{2}{3}, 1)</td>
<td>[0.368532, 2.71347]</td>
</tr>
<tr>
<td>5</td>
<td>(0, \frac{1}{4}, \frac{2}{4}, \frac{3}{4}, 1)</td>
<td>[0.707239, 1.413951]</td>
</tr>
<tr>
<td>6</td>
<td>(0, \frac{1}{5}, \frac{2}{5}, \frac{3}{5}, \frac{4}{5}, 1)</td>
<td>[0.642007, 1.55762]</td>
</tr>
<tr>
<td>8</td>
<td>(0, \frac{1}{7}, \frac{2}{7}, \frac{3}{7}, \frac{4}{7}, \frac{6}{7}, 1)</td>
<td>[0.824068, 1.21349]</td>
</tr>
</tbody>
</table>

In the case of order three, we have chosen the ESIRK methods \((-\frac{1}{2}, 0, 1), (0, \frac{1}{5}, 1)\) and the SIRK method for comparison. The numerical results (number of function evaluations and number of steps) for all DETEST problems are given in Tables 4.15, \ldots, 4.19. As a general impression, the two ESIRK methods perform better than the SIRK method except for some problems in which they have similar results (such as for \(D_3\)) or the SIRK method is slightly better than the two ESIRK methods (such as for \(B_1, B_5\) and \(D_2\)). For the ESIRK methods, the choice \((-\frac{1}{2}, 0, 1)\) seems to be better than \((0, \frac{1}{5}, 1)\) for most problems. However, we note that the choice \((0, \frac{1}{5}, 1)\) performs better than \((-\frac{1}{2}, 0, 1)\) for the E-group which has the non-linear problems with non-real eigenvalues. The reason for these outcomes probably can be explained by the fact that the SIRK method has better stability properties than either of the two ESIRK methods when variable
CHAPTER 4. THE EFFECTIVE ORDER OF SIRK METHODS

stepsize is used, and the ESIRK method \((0, \frac{1}{5}, 1)\) has better stability properties than the ESIRK method \((-\frac{1}{2}, 0, 1)\) for variable stepsize (cf. Table 4.10).

It is interesting to note that in the case of order three, the benefits of the ESIRK methods over the classical SIRK method is not only from variable stepsize but the fact that one of the abscissae of the SIRK method is greater than 1. This situation is more clear for nonlinear problems. The difference in the numerical results obtained from ESIRK and SIRK methods will obviously become greater in solving nonlinear problems, but this is not the situation for order two.

4.7 Error estimation for changing stepsize and order

Error estimation is necessary and crucial for practical purposes. The concept of "Richardson extrapolation" ([59] 1927) is that in an integration interval \((x_{n-1}, x_{n-1} + h)\), in addition to obtaining the numerical solution \(y_n\) at \(x_{n-1} + h\) with stepsize \(h\), using a different stepsize, say \(h/2\) is used, to integrate over two steps to obtain another numerical solution \(\tilde{y}_n\). The local truncation error of the latter is

\[2C_{s+1} \left(\frac{1}{2}\right)^{s+1} h^{s+1} y^{(s+1)}(x_n) + O(h^{s+2}).\]

Therefore from (4.43), the local error \(\delta\) can be estimated by

\[\delta = \frac{2^s(y_n - \tilde{y}_n)}{2^s - 1} + O(h^{s+2}).\]

Although using extrapolation makes it easy to estimate the local error, the disadvantages include the extra cost of the additional integrations and the inconvenience of designing a code. It is therefore not usually proposed for practical implementation. For Runge-Kutta methods, the scheme called "embedding" is more popular because it is easy to design in a code [50]. The error estimation of an embedding technique is to build up another method \((A, \bar{b}^T, c)\) of order one higher, say \(s + 1\), if the original method \((A, b^T, c)\) has order \(s\). If the numerical
4.7. ERROR ESTIMATION FOR CHANGING STEPSIZE AND ORDER

Table 4.14: Numerical results for stiff DETEST $A_1, A_2, A_3, D_1, D_2, D_3$ with order-2 SIRK, ESIRK methods

<table>
<thead>
<tr>
<th>Expected accuracy</th>
<th>ESIRK (0, 1)</th>
<th>SIRK</th>
<th>ESIRK $(\frac{7-4\sqrt{2}}{3}, 1)$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>nfun</td>
<td>nstep</td>
<td>nfcn</td>
</tr>
<tr>
<td>$A_1$</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$10^{-3}$</td>
<td>177</td>
<td>38</td>
<td>179</td>
</tr>
<tr>
<td>$10^{-4}$</td>
<td>453</td>
<td>109</td>
<td>480</td>
</tr>
<tr>
<td>$10^{-5}$</td>
<td>1421</td>
<td>351</td>
<td>1517</td>
</tr>
<tr>
<td>$A_2$</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$10^{-3}$</td>
<td>215</td>
<td>47</td>
<td>206</td>
</tr>
<tr>
<td>$10^{-4}$</td>
<td>536</td>
<td>130</td>
<td>543</td>
</tr>
<tr>
<td>$10^{-5}$</td>
<td>1666</td>
<td>413</td>
<td>1700</td>
</tr>
<tr>
<td>$A_3$</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$10^{-3}$</td>
<td>236</td>
<td>55</td>
<td>217</td>
</tr>
<tr>
<td>$10^{-4}$</td>
<td>621</td>
<td>151</td>
<td>606</td>
</tr>
<tr>
<td>$10^{-5}$</td>
<td>1966</td>
<td>488</td>
<td>1979</td>
</tr>
<tr>
<td>$D_1$</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$10^{-3}$</td>
<td>440</td>
<td>53</td>
<td>364</td>
</tr>
<tr>
<td>$10^{-4}$</td>
<td>674</td>
<td>103</td>
<td>578</td>
</tr>
<tr>
<td>$10^{-5}$</td>
<td>1302</td>
<td>215</td>
<td>1464</td>
</tr>
<tr>
<td>$D_2$</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$10^{-3}$</td>
<td>219</td>
<td>44</td>
<td>191</td>
</tr>
<tr>
<td>$10^{-4}$</td>
<td>535</td>
<td>105</td>
<td>515</td>
</tr>
<tr>
<td>$10^{-5}$</td>
<td>1679</td>
<td>332</td>
<td>1728</td>
</tr>
<tr>
<td>$D_3$</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$10^{-3}$</td>
<td>231</td>
<td>52</td>
<td>254</td>
</tr>
<tr>
<td>$10^{-4}$</td>
<td>567</td>
<td>139</td>
<td>621</td>
</tr>
<tr>
<td>$10^{-5}$</td>
<td>1720</td>
<td>428</td>
<td>1892</td>
</tr>
</tbody>
</table>
Table 4.15: Numerical results for A-group of stiff DETEST with order-3 SIRK, ESIRK methods

<table>
<thead>
<tr>
<th>Expected accuracy</th>
<th>ESIRK ((0, \frac{1}{5}, 1))</th>
<th>SIRK</th>
<th>ESIRK ((-\frac{1}{2}, 0, 1))</th>
</tr>
</thead>
<tbody>
<tr>
<td>(A_1)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(10^{-4})</td>
<td>271 41</td>
<td>312 50</td>
<td>273 41</td>
</tr>
<tr>
<td>(10^{-5})</td>
<td>589 95</td>
<td>643 105</td>
<td>553 89</td>
</tr>
<tr>
<td>(10^{-6})</td>
<td>1316 216</td>
<td>1316 217</td>
<td>1199 196</td>
</tr>
<tr>
<td>(A_2)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(10^{-4})</td>
<td>351 54</td>
<td>387 63</td>
<td>321 48</td>
</tr>
<tr>
<td>(10^{-5})</td>
<td>718 116</td>
<td>779 127</td>
<td>655 106</td>
</tr>
<tr>
<td>(10^{-6})</td>
<td>1599 263</td>
<td>1589 262</td>
<td>1570 258</td>
</tr>
<tr>
<td>(A_3)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(10^{-4})</td>
<td>389 61</td>
<td>409 66</td>
<td>393 62</td>
</tr>
<tr>
<td>(10^{-5})</td>
<td>815 132</td>
<td>848 139</td>
<td>801 130</td>
</tr>
<tr>
<td>(10^{-6})</td>
<td>1796 296</td>
<td>1786 295</td>
<td>1739 286</td>
</tr>
<tr>
<td>(A_4)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(10^{-4})</td>
<td>468 74</td>
<td>553 90</td>
<td>460 72</td>
</tr>
<tr>
<td>(10^{-5})</td>
<td>1057 173</td>
<td>1143 188</td>
<td>975 159</td>
</tr>
<tr>
<td>(10^{-6})</td>
<td>2454 406</td>
<td>2354 390</td>
<td>2232 369</td>
</tr>
</tbody>
</table>
Table 4.16: Numerical results for B-group of stiff DETEST with order-3 SIRK, ESIRK methods

<table>
<thead>
<tr>
<th>Expected accuracy</th>
<th>ESIRK (0, 1/3, 1)</th>
<th>SIRK</th>
<th>ESIRK (-1/3, 0, 1)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>nfunk</td>
<td>nstep</td>
<td>nfcn</td>
</tr>
<tr>
<td>$B_1$</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$10^{-3}$</td>
<td>1535</td>
<td>252</td>
<td>1473</td>
</tr>
<tr>
<td>$10^{-4}$</td>
<td>3354</td>
<td>556</td>
<td>3293</td>
</tr>
<tr>
<td>$10^{-5}$</td>
<td>7302</td>
<td>1214</td>
<td>7239</td>
</tr>
<tr>
<td>$B_2$</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$10^{-4}$</td>
<td>258</td>
<td>40</td>
<td>285</td>
</tr>
<tr>
<td>$10^{-5}$</td>
<td>581</td>
<td>93</td>
<td>619</td>
</tr>
<tr>
<td>$10^{-6}$</td>
<td>1315</td>
<td>216</td>
<td>1345</td>
</tr>
<tr>
<td>$B_3$</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$10^{-4}$</td>
<td>331</td>
<td>52</td>
<td>355</td>
</tr>
<tr>
<td>$10^{-5}$</td>
<td>723</td>
<td>117</td>
<td>745</td>
</tr>
<tr>
<td>$10^{-6}$</td>
<td>1656</td>
<td>273</td>
<td>1656</td>
</tr>
<tr>
<td>$B_4$</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$10^{-4}$</td>
<td>711</td>
<td>115</td>
<td>712</td>
</tr>
<tr>
<td>$10^{-5}$</td>
<td>1573</td>
<td>259</td>
<td>1593</td>
</tr>
<tr>
<td>$10^{-6}$</td>
<td>3473</td>
<td>575</td>
<td>3425</td>
</tr>
<tr>
<td>$B_5$</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$10^{-3}$</td>
<td>1513</td>
<td>249</td>
<td>1482</td>
</tr>
<tr>
<td>$10^{-4}$</td>
<td>3306</td>
<td>548</td>
<td>3273</td>
</tr>
<tr>
<td>$10^{-5}$</td>
<td>7240</td>
<td>1203</td>
<td>7202</td>
</tr>
</tbody>
</table>
Table 4.17: Results for C-group of stiff DETEST with order-3 SIRK, ESIRK methods

<table>
<thead>
<tr>
<th>Expected accuracy</th>
<th>ESIRK ( \left( 0, \frac{1}{5}, 1 \right) )</th>
<th>SIRK</th>
<th>ESIRK ( \left( -\frac{1}{2}, 0, 1 \right) )</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>nfun</td>
<td>nstep</td>
<td>nfcn</td>
</tr>
<tr>
<td>( C_1 )</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>( 10^{-4} )</td>
<td>408</td>
<td>62</td>
<td>459</td>
</tr>
<tr>
<td>( 10^{-5} )</td>
<td>880</td>
<td>138</td>
<td>934</td>
</tr>
<tr>
<td>( 10^{-6} )</td>
<td>1928</td>
<td>317</td>
<td>2002</td>
</tr>
<tr>
<td>( C_2 )</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>( 10^{-4} )</td>
<td>288</td>
<td>45</td>
<td>340</td>
</tr>
<tr>
<td>( 10^{-5} )</td>
<td>639</td>
<td>104</td>
<td>705</td>
</tr>
<tr>
<td>( 10^{-6} )</td>
<td>1448</td>
<td>238</td>
<td>1473</td>
</tr>
<tr>
<td>( C_3 )</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>( 10^{-4} )</td>
<td>288</td>
<td>40</td>
<td>373</td>
</tr>
<tr>
<td>( 10^{-5} )</td>
<td>640</td>
<td>93</td>
<td>814</td>
</tr>
<tr>
<td>( 10^{-6} )</td>
<td>1509</td>
<td>219</td>
<td>1647</td>
</tr>
<tr>
<td>( C_4 )</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>( 10^{-4} )</td>
<td>379</td>
<td>41</td>
<td>432</td>
</tr>
<tr>
<td>( 10^{-5} )</td>
<td>861</td>
<td>94</td>
<td>924</td>
</tr>
<tr>
<td>( 10^{-6} )</td>
<td>1956</td>
<td>231</td>
<td>2091</td>
</tr>
<tr>
<td>( C_5 )</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>( 10^{-4} )</td>
<td>388</td>
<td>40</td>
<td>449</td>
</tr>
<tr>
<td>( 10^{-5} )</td>
<td>848</td>
<td>91</td>
<td>929</td>
</tr>
<tr>
<td>( 10^{-6} )</td>
<td>1975</td>
<td>222</td>
<td>2139</td>
</tr>
</tbody>
</table>
4.7. ERROR ESTIMATION FOR CHANGING STEPSIZE AND ORDER

Table 4.18: Numerical results for D-group of stiff DETEST with order-3 SIRK, ESIRK methods

<table>
<thead>
<tr>
<th>Expected accuracy</th>
<th>ESIRK ((0, \frac{1}{5}, 1))</th>
<th>SIRK</th>
<th>ESIRK ((-\frac{1}{2}, 0, 1))</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>nfun</td>
<td>nstep</td>
<td>nfun</td>
</tr>
<tr>
<td>(D_1)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(10^{-3})</td>
<td>485</td>
<td>23</td>
<td>860</td>
</tr>
<tr>
<td>(10^{-4})</td>
<td>781</td>
<td>39</td>
<td>1011</td>
</tr>
<tr>
<td>(10^{-5})</td>
<td>945</td>
<td>71</td>
<td>1331</td>
</tr>
<tr>
<td>(D_2)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(10^{-4})</td>
<td>518</td>
<td>59</td>
<td>493</td>
</tr>
<tr>
<td>(10^{-5})</td>
<td>1017</td>
<td>112</td>
<td>989</td>
</tr>
<tr>
<td>(10^{-6})</td>
<td>2198</td>
<td>241</td>
<td>2209</td>
</tr>
<tr>
<td>(D_3)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(10^{-4})</td>
<td>584</td>
<td>72</td>
<td>568</td>
</tr>
<tr>
<td>(10^{-5})</td>
<td>1097</td>
<td>149</td>
<td>1120</td>
</tr>
<tr>
<td>(10^{-6})</td>
<td>2002</td>
<td>322</td>
<td>2283</td>
</tr>
<tr>
<td>(D_4)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(10^{-4})</td>
<td>307</td>
<td>41</td>
<td>284</td>
</tr>
<tr>
<td>(10^{-5})</td>
<td>502</td>
<td>68</td>
<td>496</td>
</tr>
<tr>
<td>(D_5)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(10^{-4})</td>
<td>620</td>
<td>52</td>
<td>560</td>
</tr>
<tr>
<td>(10^{-5})</td>
<td>828</td>
<td>74</td>
<td>964</td>
</tr>
<tr>
<td>(10^{-6})</td>
<td>–</td>
<td>–</td>
<td>1365</td>
</tr>
<tr>
<td>(D_6)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(10^{-4})</td>
<td>392</td>
<td>56</td>
<td>382</td>
</tr>
<tr>
<td>(10^{-5})</td>
<td>663</td>
<td>93</td>
<td>–</td>
</tr>
<tr>
<td>(10^{-6})</td>
<td>1308</td>
<td>180</td>
<td>–</td>
</tr>
</tbody>
</table>
Table 4.19: Numerical results for E-group of stiff DETEST with order-3 SIRK, ESIRK methods

<table>
<thead>
<tr>
<th>Expected accuracy</th>
<th>ESIRK ((0, \frac{1}{5}, 1))</th>
<th>SIRK</th>
<th>ESIRK ((-\frac{1}{5}, 0, 1))</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>nfum</td>
<td>nstep</td>
<td>nfcn</td>
</tr>
<tr>
<td>(E_1)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(10^{-4})</td>
<td>422</td>
<td>69</td>
<td>442</td>
</tr>
<tr>
<td>(10^{-5})</td>
<td>892</td>
<td>147</td>
<td>900</td>
</tr>
<tr>
<td>(10^{-6})</td>
<td>2019</td>
<td>335</td>
<td>2002</td>
</tr>
<tr>
<td>(E_2)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(10^{-2})</td>
<td>1237</td>
<td>92</td>
<td>1758</td>
</tr>
<tr>
<td>(10^{-3})</td>
<td>1728</td>
<td>198</td>
<td>2150</td>
</tr>
<tr>
<td>(10^{-4})</td>
<td>3779</td>
<td>486</td>
<td>4186</td>
</tr>
<tr>
<td>(E_3)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(10^{-4})</td>
<td>482</td>
<td>54</td>
<td>541</td>
</tr>
<tr>
<td>(10^{-5})</td>
<td>951</td>
<td>110</td>
<td>1009</td>
</tr>
<tr>
<td>(10^{-6})</td>
<td>2124</td>
<td>252</td>
<td>2219</td>
</tr>
<tr>
<td>(E_4)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(10^{-3})</td>
<td>983</td>
<td>119</td>
<td>1008</td>
</tr>
<tr>
<td>(10^{-4})</td>
<td>1806</td>
<td>247</td>
<td>1937</td>
</tr>
<tr>
<td>(10^{-5})</td>
<td>3603</td>
<td>534</td>
<td>4292</td>
</tr>
<tr>
<td>(E_5)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(10^{-4})</td>
<td>288</td>
<td>42</td>
<td>303</td>
</tr>
<tr>
<td>(10^{-5})</td>
<td>493</td>
<td>68</td>
<td>494</td>
</tr>
<tr>
<td>(10^{-6})</td>
<td>929</td>
<td>128</td>
<td>948</td>
</tr>
</tbody>
</table>
4.7. ERROR ESTIMATION FOR CHANGING STEPSIZE AND ORDER

solutions obtained by the order $s + 1$ and $s$ methods are $y_n$ and $\tilde{y}_n$ respectively, then we have

$$y(x_n) - y_n = \delta + O(h^{s+2}),$$
$$y(x_n) - \tilde{y}_n = O(h^{s+2}).$$

Therefore the local error $\delta$ can be estimated by

$$\delta = \tilde{y}_n - y_n + O(h^{s+2}) = h(b^T - b^T)F + O(h^{s+2}).$$

Because ESIRK methods have the advantage that their stage order is equal to the number of stages, we propose to use the Nordsieck vector mentioned in section 4.2 to estimate the local error. Suppose the grid point integrated is $x_{n-1}$. For an $s$-stage ESIRK method, our aim is to use the previous information to approximate the local error $Ch^{(s+1)}y^{(s+1)}$, where $C$ is the error constant of the method. The Nordsieck vector $\zeta^{[n-1]} = [h y'(x_{n-1}), h^2 y''(x_{n-1}), \ldots, h^s y^{(s)}(x_{n-1})]^T$, by the $C(s)$ condition, satisfies

$$hF^{[n-1]} = (W \otimes I_n)\zeta^{[n-1]},$$

where $W$ is the modified Vandermonde matrix and $hF^{[n-1]} = [hf(Y_1^{[n-1]}), \ldots, hf(Y_s^{[n-1]})]^T$. It follows that

$$h^s y^{(s)}(x_{n-1}) = e_s^T \zeta^{[n-1]} = e_s^T (W^{-1} \otimes I_n)hF^{[n-1]}, \quad (4.62)$$

where $e_s^T = [0, \ldots, 0, 1]^T$. Hence, for ESIRK methods, we can always approximate $h^s y^{(s)}(x_{n-1})$ by (4.62). In fact, using the $C(s)$ condition, we have

$$hF^{[n]} = (\bar{W} \otimes I_n)\tilde{\zeta}^{[n]},$$

where

$$\bar{W} = \begin{bmatrix}
1 & c_1 & \cdots & c_{s-1}^{(s-1)!} & \frac{c_s}{s!} \\
1 & c_2 & \cdots & c_{s-1}^{(s-1)!} & \frac{c_s}{s!} \\
1 & c_3 & \cdots & c_{s-1}^{(s-1)!} & \frac{c_s}{s!} \\
\vdots & \vdots & \ddots & \vdots & \vdots \\
1 & c_s & \cdots & c_{s-1}^{(s-1)!} & \frac{c_s}{s!} \\
\end{bmatrix}_{s \times (s+1)}$$

157
and
\[
\hat{\zeta}^{[n-1]} = [h y'(x_{n-1}), h^2 y''(x_{n-1}), \ldots, h^{s+1} y^{(s+1)}(x_{n-1})]^T.
\]
Therefore, we need another condition in order to approximate \(h^{s+1} y^{(s+1)}(x_{n-1})\).
One way to achieve this is to use the stepsizes of the previous step for this step, and use one of the stage approximations \(Y_i^{[n-2]}\) of the previous step. It follows that
\[
Y_i^{[n-2]} = y(x_{n-2} + c_i h) + O(h^{s+1})
= y(x_{n-1} + (c_i - 1)h) + O(h^{s+1}).
\]
By the Taylor expansion,
\[
h f(Y_i^{[n-2]}) = \sum_{k=0}^{s} \frac{(c_i - 1)^k}{k!} h^{k+1} y^{(k+1)}(x_{n-1}) + O(h^{s+2}).
\]
Hence, \(h^{s+1} y^{(s+1)}(x_{n-1})\) can be estimated by
\[
h^{s+1} y^{(s+1)}(x_{n-1}) = e_{s+1}^{T} \hat{\zeta}^{[n-1]} + O(h^{s+2}) = e_{s+1}^{T} (\hat{W}^{-1} \otimes I_N) h \hat{F}^{[n]} + O(h^{s+2}) \tag{4.63}
\]
where
\[
\hat{W} = \begin{bmatrix}
1 & c_1 - 1 & \frac{(c_1 - 1)^2}{2!} & \cdots & \frac{(c_1 - 1)^s}{s!} \\
1 & c_1 & \frac{c_1^2}{2} & \cdots & \frac{c_1^s}{s!} \\
1 & c_2 & \frac{c_2^2}{2} & \cdots & \frac{c_2^s}{s!} \\
\vdots & \vdots & \vdots & \ddots & \vdots \\
1 & c_s & \frac{c_s^2}{2} & \cdots & \frac{c_s^s}{s!}
\end{bmatrix},
\]
and \(h \hat{F}^{[n-1]} = [h f(Y_1^{[n-2]}), h f(Y_1^{[n-1]}), h f(Y_2^{[n-1]}), \ldots, h f(Y_s^{[n-1]})]\). So we can approximate \(h^{s+1} y^{(s+1)}(x_{n-1})\) as long as we can ensure that \(c_i - 1 \neq c_j, \forall j\). We present an example to illustrate how to find this error estimator.

**Example 4.17** In this example, we assume \(N = 1\). Consider an \(s = 2\) ESIRK method with \(\lambda = 1 - \frac{\sqrt{2}}{2}\) and with \(c = [0, 1]\). We then have the ESIRK method
\[
\begin{array}{c|cc}
0 & \frac{9-6\sqrt{2}}{4} & -3+2\sqrt{2} \\
1 & \frac{11-6\sqrt{2}}{4} & -1+2\sqrt{2} \\
\hline
\end{array}
\]
\[
\begin{array}{cc}
2-\sqrt{2} & -1+\sqrt{2}
\end{array}
\]
4.7. ERROR ESTIMATION FOR CHANGING STEPSIZE AND ORDER

In this case, $c_2 - c_1 = 1$, so we will use $hf(Y_1^{(n-2)})$ instead of $hf(Y_2^{(n-2)})$. From

$$h^3 y^{(3)}(x_{n-1}) = e^T_3 \hat{W}^{-1}h F^{[n-1]},$$

we can find the approximation $h^3 y^{(3)}(x_{n-1})$ by

$$[d_0, d_1, d_2] = e^T_3 \hat{W}^{-1}.$$

That is

$$\begin{bmatrix} d_0 & d_1 & d_2 \end{bmatrix} \begin{bmatrix} 1 & -1 & 1/2 \\ 1 & 0 & 0 \\ 1 & 1 & 1/2 \end{bmatrix} = [0 0 1].$$

This implies that $d_0 = 1$, $d_1 = -2$, $d_2 = 1$, and

$$Ch \left( f(Y_1^{(n-2)}) - 2f(Y_1^{(n-1)}) + f(Y_2^{(n-1)}) \right) = Ch^3 y^{(3)}(x_{n-1}) + O(h^4)$$

where $C$ is the error constant. Hence, the estimate is here of the form

$$h \left( \frac{-4 + 3\sqrt{2}}{6} f(Y_1^{(n-2)}) + \frac{4 - 3\sqrt{2}}{3} f(Y_1^{(n-1)}) + \frac{-4 + 3\sqrt{2}}{6} f(Y_2^{(n-1)}) \right).$$

Assuming that the stepsize has been constant over the last two steps, another alternative scheme to carry out the estimation is to use the difference

$$h^s y^{(s)}(x_n) - h^s y^{(s)}(x_{n-1}) = h^s y^{(s)}(x_n) - h^s y^{(s)}(x_{n-h})$$

$$= h^s y^{(s)}(x_n) - (h^s y^{(s)}(x_n) - h^{s+1} y^{(s+1)}(x_n)) + O(h^{s+2})$$

$$= h^{s+1} y^{(s+1)}(x_n) + O(h^{s+2}), \quad (4.64)$$

to approximate $h^{s+1} y^{(s+1)}(x_n)$. However, all schemes mentioned so far need the restriction of two consecutive constant steps. This is not convenient in practice. An estimator based on (4.64) can be modified in a suitable way to allow for variable stepsize. We give an example to illustrate how to achieve the estimation.

**Example 4.18** Suppose we want to estimate the local error at $x_n$, and let $x_{n-2} + \frac{h}{r} = x_{n-1}, x_{n-1} + h = x_n$. 

159
Consider an order 2 ESIRK method. We assume that $\delta_1, \delta_2$ are the local error approximations to $(h^2/r^2)y''(x_{n-2})$ and $h^2y''(x_{n-1})$ respectively, and suppose $d_1, d_2$ satisfy

$$
\delta_1 = d_1 \frac{h}{r} f(Y_1^{[n-2]}) + d_2 \frac{h}{r} f(Y_2^{[n-2]}),
$$

$$
\delta_2 = d_1 h f(Y_1^{[n-1]}) + d_2 h f(Y_2^{[n-1]}).
$$

By the $C(2)$ condition, we express $\delta_1, \delta_2$ in terms of the internal stage derivatives and use the Taylor series to expand them at $x_n$ as follows.

$$
\delta_1 = \left( \frac{h}{r} \right)^2 y''(x_{n-2}) + C_3 \left( \frac{h}{r} \right)^3 y'''(x_{n-2}) + O(h^4)
$$

$$
= \left( \frac{h}{r} \right)^2 y''(x_{n-2} + C_3 \left( \frac{h}{r} \right)) + O(h^4)
$$

$$
= \left( \frac{h}{r} \right)^2 y''(x_{n-1} - \frac{h}{r} + C_3 \left( \frac{h}{r} \right)) + O(h^4)
$$

$$
= \left( \frac{h}{r} \right)^2 y''(x_n - h + (C_3 - 1) \frac{h}{r}) + O(h^4)
$$

$$
= \left( \frac{h}{r} \right)^2 y''(x_n) + \left( -h + (C_3 - 1) \frac{h}{r} \right) \left( \frac{h}{r} \right)^2 y'''(x_n) + O(h^4)
$$

$$
= \left( \frac{h}{r} \right)^2 y''(x_n) + \left( -1 + \frac{C_3 - 1}{r^2} \right) h^3 y'''(x_n) + O(h^4),
$$

(4.65)

$$
\delta_2 = h^2 y''(x_{n-1}) + C_3 h^3 y'''(x_{n-1}) + O(h^4)
$$

$$
= h^2 y''(x_{n-1} + C_3 h) + O(h^4)
$$

$$
= h^2 y''(x_n + (C_3 - 1) h) + O(h^4)
$$

$$
= h^2 y''(x_n) + (C_3 - 1) h^3 y'''(x_n) + O(h^4),
$$

(4.66)
where $d_1, d_2$ satisfy
\[
\begin{cases}
d_1 + d_2 = 0, \\
d_1 c_1 + d_2 c_2 = 1, \\
\frac{1}{2}(d_1 c_1^2 + d_2 c_2^2) = C_3.
\end{cases}
\] (4.67)

Then, (4.66)–$r^2(4.65)$ yields
\[
\delta_2 - r^2 \delta_1 = (C_3 - 1) - \left(-1 + \frac{C_3 - 1}{r}\right) h^3 y'''(x_n) + O(h^4) \\
= \frac{(r-1)C_3 + 1}{r} h^3 y'''(x_n) + O(h^4).
\]

Therefore we can use \(\frac{r}{(r-1)c_3+1}(\delta_2 - r^2 \delta_1)\) to estimate $h^3 y'''(x_n)$. From (4.67), we find
\[
d_1 = \frac{1}{c_1 - c_2}, \quad d_2 = -\frac{1}{c_1 - c_2}, \quad C_3 = \frac{c_1 + c_2}{2},
\]
so the error estimator is given by
\[
\frac{2r}{2 + (r-1)(c_1 + c_2)}(\delta_2 - r^2 \delta_1),
\]
where
\[
\delta_1 = \frac{1}{c_2 - c_1} \left( \frac{h}{r} f(Y_2^{[n-2]}) - \frac{h}{r} f(Y_1^{[n-2]}) \right),
\]
\[
\delta_2 = \frac{1}{c_2 - c_1} \left( h f(Y_2^{[n-1]}) - h f(Y_1^{[n-1]}) \right).
\]

As we have seen, the error estimator derived in this way depends on the abscissae and the stepsize changing ratio $r$. For the special case $c_1 = 0, c_2 = 1$, the estimator will be of the form
\[
\frac{2r}{1 + r} \left( h f(Y_2^{[n-1]}) - h f(Y_1^{[n-1]}) - r^2 \left( \frac{h}{r} f(Y_2^{[n-2]}) - \frac{h}{r} f(Y_1^{[n-2]}) \right) \right).
\]

In general, we can have the following result for any $s$. 

161
CHAPTER 4. THE EFFECTIVE ORDER OF SIRK METHODS

Theorem 4.23 For an s-stage ESIRK method with distinct abscissae \( c_1, c_2, \ldots, c_s \), suppose \( \delta_1, \delta_2 \) satisfy
\[
\delta_1 = \left( \frac{h}{r} \right)^s y^{(s)}(x_{n-2}) + O(h^{s+1}), \quad \delta_2 = h^s y^{(s)}(x_{n-1}) + O(h^{s+1}),
\]
where \( r \) is defined as the same as for Example 4.18. Then the principal local truncation error is given by
\[
C h^{s+1} y^{(s+1)}(x_n) = C \frac{sr}{s + (r - 1) \sum_{i=1}^{s} C_i} (\delta_2 - r^s \delta_1) + O(h^{s+2}),
\]
where \( C \) is the associated error constant.

Proof: From the derivations (4.65), (4.66) in Example 4.18, it is easy to see that
\[
\delta_1 = \left( \frac{h}{r} \right)^s y^{(s)}(x_n) + \left( -1 + \frac{C_{s+1} - 1}{r^s} \right) h^{s+1} y^{(s+1)}(x_n) + O(h^{s+2}),
\]
\[
\delta_2 = h^s y^{(s)}(x_n) + (C_{s+1} - 1) h^{s+1} y^{(s+1)}(x_n) + O(h^{s+2}),
\]
and
\[
\frac{r}{(r - 1) C_{s+1} + 1} (\delta_2 - r^s \delta_1) = h^{s+1} y^{(s+1)}(x_n) + O(h^{s+2}). \tag{4.68}
\]
If we assume that
\[
\delta_1 = \sum_{i=1}^{s} d_i \frac{h}{r} f(Y_i^{[n-2]}), \quad \delta_2 = \sum_{i=1}^{s} d_i h f(Y_i^{[n-1]}),
\]
by the Taylor expansion, it can then be shown that
\[
\begin{align*}
\sum_{i=1}^{s} d_i c_i^k &= 0, & k &= 0, 1, \ldots, s - 2, \\
\sum_{i=1}^{s} d_i c_i^{s-1} &= 1, \\
\sum_{i=1}^{s} d_i c_i^s &= C_{s+1}.
\end{align*}
\]
This yields
\[
C_{s+1} = \frac{1}{s!} \sum_{i=1}^{s} d_i c_i^s = \frac{1}{s!} \sum_{i=1}^{s} d_i \left( \prod_{j=1}^{s} (c_i - c_j) + c_i^{s-1} \sum_{j=1}^{s} c_j \right) = \frac{1}{s!} \sum_{i=1}^{s} d_i c_i^{s-1} \sum_{j=1}^{s} c_j = \frac{(s - 1)!}{s!} \sum_{j=1}^{s} c_j. \tag{4.69}
\]

162
4.7. ERROR ESTIMATION FOR CHANGING STEPSIZE AND ORDER

Substituting (4.69) into (4.68), the result follows.

We also note that Theorem 4.23 is valid for collocation methods. In practice, we can use (4.62) to find \( d_4 \), then form \( \delta_1 \) and \( \delta_2 \) after every successful Newton iteration.

**Experiment 4.6** We use the second-order ESIRK method with the error estimator in Theorem 4.23 to solve the Kaps problem (3.9). The exact solution for the Kaps problems is known to be \([\exp(-2x), \exp(-x)]\). Hence, we can plot the exact local error together with the estimated error obtained by the code as a function of \( h \). In Figure 4.26, \( \circ \) and \( * \) stand for the exact error and estimated error respectively. We find that the error estimator gives a very accurate estimation from the first few steps, and become slightly over-estimated when \( h \) has become larger.

![Figure 4.26: Error estimation of order-2 ESIRK for the Kaps problem](image)

For ESIRK methods, it is possible to implement the code with variable order. If we use the standard strategy to select different order for a code [5], [34], it
CHAPTER 4. THE EFFECTIVE ORDER OF SIRK METHODS

is necessary to approximate \( h^{s+2}y^{(s+2)}(x_n) \) in order that the code will be able to decide to increase or decrease the order for the next step. We can achieve this aim using several constant steps to get the approximation to \( h^{s+2}y^{(s+2)}(x_n) \).

The number of constant steps depend on which scheme is used to estimate the local error. For example, if we adopt (4.63) to approximate the local error and use one of the previous stage derivatives \( Y^{[n-1]}_i \) combined with the current stage \( Y^{[n]}_i, i = 1, 2, \ldots, s \) to form \( \hat{F} \), then these two steps have to be constant. In order to approximate \( h^{s+2}y^{(s+2)}(x_n) \) at the end of the next step, the stepsize of the next step must be the same as in the previous two steps. In other words, it is necessary to have three constant steps in order to achieve this goal. In this case, we have the following result

**Theorem 4.24** Let \( (c_1, c_2, \ldots, c_s) \) be the distinct abscissae for an \( s \)-stage ESIRK method. Assume for each \( i, c_i \neq c_j - 1 \). Suppose \( E_n \) satisfies

\[
E_n = h^{s+1}y^{(s+1)}(x_n) = h\tilde{b}_0 f(Y^{[n-1]}_i) + h \sum_{j=1}^{s} \tilde{b}_j f(Y^{[n]}_j) + O(h^{s+2}),
\]

for some \( i = 1, 2, \ldots, s \). If \( \nabla E_n = E_n - E_{n-1} \), where the stepsize \( h \) is constant over three steps, then

\[
\nabla E_n = h^{s+2}y^{(s+2)}(x_n) + O(h^{s+3}).
\]

**Proof:** Let \( g_i \) be such that

\[
Y_i = y(x_{n-1} + c_i h) + g_i h^{s+1} + O(h^{s+2}),
\]

and let \( C_1, C_2 \) be such that

\[
y_{n-2} - y(x_{n-2}) = C_1 h^{s+1} + O(h^{s+2}),
\]

and

\[
y_{n-1} - y(x_{n-1}) = C_2 h^{s+1} + O(h^{s+2}),
\]

and also

\[
y_{n-3} = y(x_{n-3}).
\]
4.7. ERROR ESTIMATION FOR CHANGING STEPSIZE AND ORDER

We then have \( C_2 = 2C_1 \) and for arbitrary \( 1 \leq i \leq s \), we have

\[
\nabla E_n = h\tilde{b}_0 f(Y_i^{[n-1]}) + h \sum_{j=1}^{s} \tilde{b}_j f(Y_j^{[n]})
\]

\[
- h \tilde{b}_0 f(Y_i^{[n-2]}) - h \sum_{j=1}^{s} \tilde{b}_j f(Y_j^{[n-1]}) + O(h^{s+2})
\]

\[
= h\tilde{b}_0 f(y(x_{n-2} + c_i h) + (g_i + C_1) h^{s+1})
\]

\[
+ h \sum_{j=1}^{s} \tilde{b}_j f(y(x_{n-2} + (c_j + 1) h) + (g_j + C_2) h^{s+1})
\]

\[
- h\tilde{b}_0 f(y(x_{n-3} + c_i h + g_i h^{s+1})) - h \sum_{j=1}^{s} \tilde{b}_j f(y(x_{n-3} + (c_j + 1) h)
\]

\[
+ (g_j + C_1) h^{s+1}) + O(h^{s+3})
\]

\[
= h\tilde{b}_0 f(y(x_{n-2} + c_i h)) + h \sum_{j=1}^{s} \tilde{b}_j f(y(x_{n-2} + (c_j + 1) h))
\]

\[
- h\tilde{b}_0 f(y(x_{n-3} + c_i h)) - h \sum_{j=1}^{s} \tilde{b}_j f(y(x_{n-3} + (c_j + 1) h))
\]

\[
+ h \sum_{j=1}^{s} \tilde{b}_j f(y(x_{n-2} + (c_j + 1) h)) (g_j + C_2) h^{s+1}
\]

\[
- h\tilde{b}_0 f(y(x_{n-3} + c_0 h)) g_i h^{s+1}
\]

\[
- h \sum_{j=1}^{s} \tilde{b}_j f(y(x_{n-3} + (c_j + 1) h)) (g_i + C_1) h^{s+1} + O(h^{s+3})
\]

\[
= h\tilde{b}_0 f(y(x_{n-2} + c_i h)) + h \sum_{j=1}^{s} \tilde{b}_j f(y(x_{n-2} + (c_j + 1) h))
\]

\[
- h\tilde{b}_0 f(y(x_{n-3} + c_i h)) - h \sum_{j=1}^{s} \tilde{b}_j f(y(x_{n-3} + (c_j + 1) h))
\]

\[
+ h \sum_{j=1}^{s} \tilde{b}_j f'(y_n)(g_i + C_1) h^{s+1} + h \sum_{j=1}^{s} \tilde{b}_j f'(y_n) h^{s+1}(g_j + C_2) h^{s+1}
\]

\[
- h\tilde{b}_0 f'(y_n)g_i h^{s+1} - h \sum_{j=1}^{s} \tilde{b}_j f'(y_n) h^{s+1}(g_j + C_1) h^{s+1}
\]

\[
+ h \sum_{j=0}^{s} \tilde{b}_j f(y(x_{n-2} + c_j h)) - h \sum_{j=0}^{s} \tilde{b}_j f(y(x_{n-3} + c_j h))
\]
CHAPTER 4. THE EFFECTIVE ORDER OF SIRK METHODS

\[ + h \sum_{j=0}^{s} b_j f'(y_n) h^{s+1} C_1 h^{s+1} \]

\[ = h \sum_{j=0}^{s} b_j (y'(x_{n-2} + c_j h) - y'(x_{n-3} + c_j h)) \]

\[ = h \sum_{j=0}^{s} b_j \int_{x_{n-3}}^{x_{n-2}} y''(x + c_j h) dx \]

\[ = \int_{x_{n-3}}^{x_{n-2}} h^{s+1} y^{(s+2)}(x) dx \]

\[ = (x_{n-1} - x_{n-2}) h^{s+1} y^{(s+2)}(x_n) + O(h^{s+3}). \]

\[ \square \]

If we use Theorem 4.23 to estimate the local error, in a similar way, we can obtain the approximation \( h^{s+2} y^{(s+2)}(x_n) \) at the end of two steps with constant stepsize \( h \). We give the result below without proof.

**Lemma 4.25** For an \( s \)-stage ESIRK method, suppose \( E_n \) is obtained by

\[ E_n = h^{s+1} y^{(s+1)}(x_n) = \frac{sr}{s + (r - 1) \sum_{i=1}^{s} c_i} (\delta_2 - r \delta_1) + O(h^{s+1}), \]

where \( \delta_1, \delta_2 \) are defined as in Theorem 4.23, and the stepsize \( h \) is constant over two steps \( (x_{n-3} + \frac{h}{r} = x_{n-2}, x_{n-2} + h = x_{n-1}, x_{n-1} + h = x_n) \), then

\[ \nabla E_n = h^{s+2} y^{(s+2)}(x_n) + O(h^{s+3}). \]

### 4.8 Implementation and some numerical results

If \( N \) is the system dimension, and \( s \) is the order of the method, it is well known that solving the \( sN \) nonlinear algebraic equations is a crucial part in the implementation of implicit Runge-Kutta formulae. The main purpose of using singly-implicit methods is to reduce the overall integration costs by using old information repeatedly under the assumption that suitable conditions are satisfied.

In this section, we will first discuss some implementations of convergence control, such as how to arrange those conditions in order to achieve maximum savings.
Then we discuss some criterion about solving the $sN$ nonlinear algebraic equations. Because the stepsize changing scheme for ESIRK is quite different from that for usual methods, we also want to discuss the implementation of variable stepsize for ESIRK methods. In the final part, we give some numerical results obtained using the $A$-stable ESIRK methods and their associated $A(\alpha)$-stable methods to examine whether there are any advantages resulting from the smaller error constants. Some numerical comparisons using the existing IRK codes $SDIRK$ and $RADAU5$ together with the ESIRK methods of the same order are also given.

As we have seen in section 3.1, using the modified Newton method to solve the stage values of a singly-implicit Runge-Kutta method is equivalent to solving with the iteration matrix

$$M = I_N - h\lambda J,$$

where $\lambda$ is the single eigenvalue of the method and $J = f_y(y_0)$ is the Jacobian matrix. For ESIRK methods, we use (3.8) to implement the Newton iterations, so we want to solve the system with the iteration matrix (4.70). Ideally, we may want to use $M$ and $J$ unchanged repeatedly as far as possible. Since $M$ is determined by $h$ and $J$, if $h$ or $J$ has been changed, then $M$ should be updated. For $J$, if either the convergence rate (we discuss this later) of the last step is too slow or we have tried refactorizing $M$ and it still fails to converge, then we must update $J$. But we must be careful to avoid recomputing $J$ if we have just already updated it within this step. Of course, it is possible that after we update the Jacobian and the iteration matrix, it still diverges. If so, we then need to lower the stepsize. Therefore, the algorithm for updating $M, J, h$ for convergence control can be written as follows.

1. **Algorithm: Jacobian and iteration matrix update policy**

   if

   1.1 We have tried refactorizing
   1.2 We have tried recomputing $J$
   1.3 OR we have tried lowering $h$ already
   1.4 It still fails to converge
CHAPTER 4. THE EFFECTIVE ORDER OF SIRK METHODS

2.2
Lower the value of $h$

endif

if
2.1 We have tried refactorizing
2.2 It still fails to converge
2.3 BUT do not recompute $J$ if we have already lowered $h$
2.4 OR the convergence rate is too slow

3 Evaluate Jacobian

endif

if
3.1 We have just changed $J$
3.2 OR We have just changed $h$

4 Factorize $I - h\lambda J$

endif

Sometimes, even if a new stepsize $h$ is used but with a moderate change only, we may want to use the old $M$ when an old $J$ is used. We have to decide how much change in the value of $h$ to allow for us to use the old copy. In the DESI code, 20\% change of $h$ is the upper bound for keeping the old $M$ [34]. But this is very dependent on the problems we are solving. For more details, one can refer to [42], [34].

In the solution of the internal stages, some criteria we want to discuss are the starting values for the Newton iterations, when to stop the iterations, and the evaluation of the derivative stage values. For the predictor, it is natural to use the previous output approximation $y_n$ as the starting value for iterations. Alternatively, we can use some conditions to construct an interpolation polynomial which can be used to predict the starting values. From the idea in [57], for an order-$p$ method, the interpolation formulae can be found using the $B(p)$ condition. Supposing the ESIRK method to be of order $p$, we write

$$y_n(\theta) = y(x_{n-1} + \theta h) + O(h^{p+1}), \quad (4.71)$$

168
where
\[ y_n(\theta) = y_{n-1} + h \sum_{i=1}^{p} b_i(\theta) f(Y_i^{[n-1]}). \]  
(4.72)

Let \( x_{n+1} = x_n + rh \). By the \( C(p) \) condition, the internal stages \( Y_i \) can be written as
\[
Y_i^{[n]} = y(x_n + c_i rh) + O(h^{p+1}) \\
= y(x_{n-1} + (1 + c_i r) h) + O(h^{p+1}), \quad i = 1, 2, \ldots, p. 
\]  
(4.73)

From (4.71), the prediction for \( Y_i \) is given by setting \( \theta = 1 + rc_i \). To illustrate this, we give an example below.

**Example 4.19** Consider the second-order ESIRK method (4.42) with abscissae \( c_1 = 0, c_2 = 1 \). By \( B(2) \) and (4.72), we have
\[
b_1(\theta) + b_2(\theta) = \theta, \quad b_1(\theta)c_1 + b_2(\theta)c_2 = \frac{\theta^2}{2}.
\]

Solving the above equations yields
\[
b_1(\theta) = \theta - \frac{\theta^2}{2}, \quad b_2(\theta) = \frac{\theta^2}{2}.
\]

Here \( \theta \) is 1 and \( 1 + r \) for \( Y_1^{[n]} \) and \( Y_2^{[n]} \) respectively. From (4.71), the predictor is given by
\[
\begin{align*}
Y_1^{[n]} & \approx y_{n-1} + \frac{h}{2} \left( f(Y_1^{[n-1]}) + f(Y_2^{[n-1]}) \right), \\
Y_2^{[n]} & \approx y_{n-1} + h \left( 1 + r - \frac{(1 + r)^2}{2} \right) f(Y_1^{[n-1]}) + \frac{(1 + r)^2}{2} f(Y_2^{[n-1]}) \bigg) .
\end{align*}
\]

Because ESIRK methods have stage order equal to the number of stages, another approach to constructing an interpolation polynomial for predicting the starting value for the Newton iterations is to let \( Y_i^{[n]} \) be the \( i \)-th stage value of the \( n \)-th step and \( x_n = x_{n-1} + h \). Also assume the stepsize at the \( n \)-th step is \( rh \). Then from the \( C(p) \) condition and the Taylor series, we have
\[
Y_i^{[n]} = y(x_n + c_i rh) + O(h^{p+1}) \\
= y(x_n) + c_i rh y'(x_n) + \frac{c_i^2 rh^2}{2} y''(x_n) + \cdots + \frac{c_i^p r^p}{p!} h^p y^{(p)}(x_n) + O(h^{p+1}).
\]
The question now is how we use the previous derivatives \( hf(Y_i^{[n-1]}) \) to approximate \( Y_i^{[n]} \). This can be done by the following argument. By \( C(p) \) again, we have

\[
Y_i^{[n-1]} = y(x_{n-1} + c_i h) + O(h^{p+1})
\]

\[
= y(x_n + (c_i - 1) h) + O(h^{p+1}),
\]

hence, for \( i = 1, 2, \ldots, s \), it follows that

\[
hf(Y_i^{[n-1]}) = hy'(x_n + (c_i - 1) h) + O(h^{p+2})
\]

\[
= hy'(x_n) + (c_i - 1) h^2 y''(x_n) + \cdots + \frac{(c_i - 1)^{p-1}}{(p-1)!} h^p y^{(p)}(x_n) + O(h^{p+1}).
\]

Therefore we can write \( \eta^{[n]} = [hy'(x_n), h^2 y''(x_n), \ldots, h^p y^{(p)}(x_n)]^T \), in terms of \( hf(Y_i^{[n-1]}) \). If we let

\[
h_F^{[n-1]} = \begin{bmatrix}
    hf(Y_1^{(n-1)}) \\
    hf(Y_2^{(n-1)}) \\
    \vdots \\
    hf(Y_p^{(n-1)})
\end{bmatrix}, \quad W = \begin{bmatrix}
    1 & c_1 - 1 & \frac{(c_1 - 1)^2}{2} & \cdots & \frac{(c_1 - 1)^{p-1}}{(p-1)!} \\
    1 & c_2 - 1 & \frac{(c_2 - 1)^2}{2} & \cdots & \frac{(c_2 - 1)^{p-1}}{(p-1)!} \\
    \vdots & \vdots & \vdots & \ddots & \vdots \\
    1 & c_p - 1 & \frac{(c_p - 1)^2}{2} & \cdots & \frac{(c_p - 1)^{p-1}}{(p-1)!}
\end{bmatrix},
\]

and \( T_i = [c_i r, \frac{c_i^2 r^2}{2}, \ldots, \frac{c_i^p r^p}{p!}] \), then the predictor is given by

\[
Y_i^{[n]} = y(x_n) + c_i r hy'(x_n) + \frac{c_i^2 r^2}{2} h^2 y''(x_n) + \cdots + \frac{c_i^p r^p}{p!} h^p y^{(p)}(x_n) + O(h^{p+1})
\]

\[
= y(x_n + T_i \eta^{[n]}) + O(h^{p+1})
\]

\[
= y(x_n) + (T_i W^{-1} \otimes I_N) h F_i^{[n-1]} + O(h^{p+1}). \tag{4.74}
\]

We give another example to show how to find the predictor using (4.74).

**Example 4.20** We also use the second-order ESIRK method (4.42) with \( c_1 = 0, c_2 = 1 \). From \( T_i = [c_i r, \frac{c_i^2 r^2}{2}] \), \( i = 1, 2 \), we have

\[
T_1 = [0, 0], \quad T_2 = [r, \frac{r^2}{2}].
\]

Now

\[
W = \begin{bmatrix}
    1 & c_1 - 1 \\
    1 & c_2 - 1
\end{bmatrix} \Rightarrow W^{-1} = \begin{bmatrix}
    0 & 1 \\
    -1 & 1
\end{bmatrix},
\]

170
and by (4.74), the predictor is given by

\[
\begin{align*}
Y_1^{[n]} &\approx y(x_n), \\
Y_2^{[n]} &\approx y(x_n) - \frac{r^2}{2} hf(Y_1^{[n-1]}) + \left( r + \frac{r^2}{2} \right) hf(Y_2^{[n-1]}).
\end{align*}
\]

**Experiment 4.7** In Tables 4.20 and 4.21, comparisons using different predictors for order 2, 3 ESIRK methods are given. The problems tested are the Kaps problem, the Robertson kinetic problems and Van der Pol’s equation. We compare the predictors (4.73), (4.74) together with the one using the previous solution as the starting values. It is clear to see that the predictor (4.74) is the overall best way to obtain starting values. The numerical results show that the reduction of the number of function evaluations is mainly due to the lower average iteration numbers.

In solving the nonlinear algebraic equations for implicit Runge-Kutta methods another issue arises: when to stop the Newton iterations and to accept the approximations from the iterations. The way this is done greatly affects the efficiency of a code. It is hoped to obtain an expected accurate solution from the iterations at the lowest computational cost. The criteria for stopping the iterations have been addressed by many authors, see for example [61], [48], [42] and [34]. The main idea is that if \( Y^{[n]} \) denotes the \( n \)-th Newton update during the iteration in a certain step, then the quantity

\[
\tau^{[n]} = \frac{||Y^{[n+1]} - Y^{[n]}||}{||Y^{[n]} - Y^{[n-1]}||},
\]

is the convergence rate. It is clear that if some \( \tau^{[n]} > 1 \), the iteration is not contracting from the starting value \( Y^{[0]} \) and the iteration should be terminated. Furthermore, it can be shown that [34],

\[
||Y - Y^{[n+1]}|| \leq \frac{\tau}{1 - \tau} ||Y^{[n+1]} - Y^{[n]}||.
\]

Hence, if we assume \( \tau \) is the expected error bound (closely related to the accuracy \( \varepsilon \) desired for the solution of the differential equation), then the test for stopping
CHAPTER 4. THE EFFECTIVE ORDER OF SIRK METHODS

Table 4.20: Results when using previous solution value (A), predictor (4.73) (B) and predictor (4.74) (C) with order-2 ESIRK method

<table>
<thead>
<tr>
<th>Predictor</th>
<th>nstep</th>
<th>nrej(lte/newt)</th>
<th>nfcn</th>
<th>nfolps</th>
<th>gberr</th>
<th>niter</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Kaps problem</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$Tol = 10^{-2}$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A</td>
<td>19</td>
<td>3/0</td>
<td>64</td>
<td>12910</td>
<td>5.32 x 10^{-2}</td>
<td>1.41</td>
</tr>
<tr>
<td>B</td>
<td>29</td>
<td>5/0</td>
<td>86</td>
<td>21344</td>
<td>1.46 x 10^{-1}</td>
<td>1.24</td>
</tr>
<tr>
<td>C</td>
<td>15</td>
<td>0/0</td>
<td>52</td>
<td>9948</td>
<td>2.03 x 10^{-3}</td>
<td>1.67</td>
</tr>
<tr>
<td>$Tol = 10^{-4}$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A</td>
<td>71</td>
<td>21/0</td>
<td>274</td>
<td>52526</td>
<td>2.36 x 10^{-3}</td>
<td>1.48</td>
</tr>
<tr>
<td>B</td>
<td>138</td>
<td>43/0</td>
<td>474</td>
<td>112352</td>
<td>8.07 x 10^{-3}</td>
<td>1.30</td>
</tr>
<tr>
<td>C</td>
<td>40</td>
<td>0/0</td>
<td>144</td>
<td>25973</td>
<td>1.60 x 10^{-4}</td>
<td>1.77</td>
</tr>
<tr>
<td>Robertson kinetics</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$Atol = 10^{-2}$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A</td>
<td>93</td>
<td>1/0</td>
<td>346</td>
<td>91388</td>
<td>1.40 x 10^{-8}</td>
<td>1.84</td>
</tr>
<tr>
<td>B</td>
<td>95</td>
<td>1/1</td>
<td>500</td>
<td>119508</td>
<td>3.26 x 10^{-8}</td>
<td>2.60</td>
</tr>
<tr>
<td>C</td>
<td>94</td>
<td>1/0</td>
<td>362</td>
<td>96764</td>
<td>4.41 x 10^{-9}</td>
<td>1.90</td>
</tr>
<tr>
<td>$Atol = 10^{-4}$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A</td>
<td>305</td>
<td>1/0</td>
<td>1206</td>
<td>302846</td>
<td>9.78 x 10^{-10}</td>
<td>1.97</td>
</tr>
<tr>
<td>B</td>
<td>306</td>
<td>2/0</td>
<td>1240</td>
<td>336475</td>
<td>1.46 x 10^{-9}</td>
<td>2.01</td>
</tr>
<tr>
<td>C</td>
<td>305</td>
<td>1/0</td>
<td>800</td>
<td>265160</td>
<td>3.28 x 10^{-10}</td>
<td>1.31</td>
</tr>
<tr>
<td>Van der Pol</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$Atol = 10^{-2}$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A</td>
<td>200</td>
<td>10/19</td>
<td>1248</td>
<td>166749</td>
<td>1.58 x 10^{-2}</td>
<td>2.97</td>
</tr>
<tr>
<td>B</td>
<td>221</td>
<td>16/20</td>
<td>1412</td>
<td>209533</td>
<td>4.04 x 10^{-2}</td>
<td>2.97</td>
</tr>
<tr>
<td>C</td>
<td>197</td>
<td>11/16</td>
<td>1092</td>
<td>159630</td>
<td>6.35 x 10^{-2}</td>
<td>2.62</td>
</tr>
<tr>
<td>$Atol = 10^{-4}$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A</td>
<td>741</td>
<td>12/0</td>
<td>3064</td>
<td>479313</td>
<td>1.84 x 10^{-3}</td>
<td>2.03</td>
</tr>
<tr>
<td>B</td>
<td>744</td>
<td>12/0</td>
<td>3330</td>
<td>563377</td>
<td>2.03 x 10^{-3}</td>
<td>2.20</td>
</tr>
<tr>
<td>C</td>
<td>735</td>
<td>8/0</td>
<td>2164</td>
<td>428389</td>
<td>2.20 x 10^{-3}</td>
<td>1.45</td>
</tr>
</tbody>
</table>
### 4.8. IMPLEMENTATION AND SOME NUMERICAL RESULTS

Table 4.21: Results when using previous solution value (A), predictor (4.73) (B) and predictor (4.74) (C) with order-3 ESIRK method

<table>
<thead>
<tr>
<th>Predictor</th>
<th>nstep</th>
<th>nrej(lte/newt)</th>
<th>nfcn</th>
<th>nfolps</th>
<th>glberr</th>
<th>niter</th>
</tr>
</thead>
<tbody>
<tr>
<td>Kaps problem</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Tol = 10^{-4}</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A</td>
<td>32</td>
<td>1/0</td>
<td>144</td>
<td>32913</td>
<td>2.68e-04</td>
<td>1.39</td>
</tr>
<tr>
<td>B</td>
<td>69</td>
<td>10/0</td>
<td>378</td>
<td>89855</td>
<td>1.35e-03</td>
<td>1.57</td>
</tr>
<tr>
<td>C</td>
<td>30</td>
<td>0/0</td>
<td>129</td>
<td>30119</td>
<td>3.68e-05</td>
<td>1.37</td>
</tr>
<tr>
<td><strong>Tol = 10^{-6}</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A</td>
<td>92</td>
<td>6/0</td>
<td>678</td>
<td>108615</td>
<td>5.94e-05</td>
<td>2.29</td>
</tr>
<tr>
<td>B</td>
<td>106</td>
<td>6/0</td>
<td>768</td>
<td>146410</td>
<td>2.51e-05</td>
<td>2.27</td>
</tr>
<tr>
<td>C</td>
<td>80</td>
<td>0/0</td>
<td>288</td>
<td>72916</td>
<td>1.99e-06</td>
<td>1.18</td>
</tr>
<tr>
<td>Robertson kinetics</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Atol = 10^{-4}</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A</td>
<td>227</td>
<td>0/0</td>
<td>1353</td>
<td>363485</td>
<td>1.84e-09</td>
<td>1.98</td>
</tr>
<tr>
<td>B</td>
<td>231</td>
<td>2/0</td>
<td>1713</td>
<td>468195</td>
<td>1.89e-09</td>
<td>2.45</td>
</tr>
<tr>
<td>C</td>
<td>228</td>
<td>1/0</td>
<td>807</td>
<td>318402</td>
<td>1.76e-09</td>
<td>1.17</td>
</tr>
<tr>
<td><strong>Atol = 10^{-6}</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A</td>
<td>628</td>
<td>1/0</td>
<td>3762</td>
<td>1000145</td>
<td>1.01e-10</td>
<td>1.99</td>
</tr>
<tr>
<td>B</td>
<td>629</td>
<td>4/0</td>
<td>4326</td>
<td>1221739</td>
<td>1.09e-10</td>
<td>2.28</td>
</tr>
<tr>
<td>C</td>
<td>628</td>
<td>2/0</td>
<td>2073</td>
<td>848757</td>
<td>1.12e-10</td>
<td>1.09</td>
</tr>
<tr>
<td>Van der Pol</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Atol = 10^{-4}</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A</td>
<td>511</td>
<td>19/0</td>
<td>3501</td>
<td>563942</td>
<td>2.49e-04</td>
<td>2.20</td>
</tr>
<tr>
<td>B</td>
<td>500</td>
<td>17/0</td>
<td>4227</td>
<td>727837</td>
<td>3.39e-04</td>
<td>2.72</td>
</tr>
<tr>
<td>C</td>
<td>520</td>
<td>21/0</td>
<td>2148</td>
<td>503057</td>
<td>2.52e-04</td>
<td>1.32</td>
</tr>
<tr>
<td><strong>Atol = 10^{-6}</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A</td>
<td>1469</td>
<td>13/0</td>
<td>5885</td>
<td>1571723</td>
<td>1.37e-05</td>
<td>2.22</td>
</tr>
<tr>
<td>B</td>
<td>1468</td>
<td>9/0</td>
<td>11595</td>
<td>2023805</td>
<td>1.38e-05</td>
<td>2.62</td>
</tr>
<tr>
<td>C</td>
<td>1470</td>
<td>9/0</td>
<td>5511</td>
<td>1332078</td>
<td>1.38e-05</td>
<td>1.24</td>
</tr>
</tbody>
</table>
the iterations is given by

\[
\frac{r}{1 - r} ||Y^{n+1} - Y^n|| \leq \tau. \tag{4.75}
\]

In general, \( \tau \) is chosen smaller than \( \varepsilon \). Shampine [61] pointed out that when \( \tau \) is a great deal smaller than \( \varepsilon \), it does not improve the solution of the differential equation and the smaller \( \tau \) is made, the more it costs to compute the internal stage values. In general, if the user's defined tolerance is \( tol = \varepsilon \), then \( \tau = 0.1\varepsilon \) is suggested. This is also very dependent on problems and methods. For example, for linear problems, \( \tau \) can be chosen larger than for non-linear problems. Furthermore, if the methods used take a large step, such as DESI, then a smaller \( \tau \) is suggested. From [48], we also notice that there must be a consistent choice between the type of error estimator and the vector used for the test of convergence. When solving some special stiff problems, such as the Robertson reaction kinetics, it is best to use the absolute tolerance, \( atol \), and the relative tolerance, \( rtol \), that is to use the error [42],

\[ LTE \leq atol + \max_i(||y_i||)rtol, \]

where \( LTE \) is the local error estimate and \( y_i \) denotes the \( i \)-th component of the numerical solution. Hence, the test (4.75) can be written as

\[
\frac{r}{1 - r} ||Y^{n+1} - Y^n|| \leq 0.1(atol + \max_i(||y_i||)rtol).
\]

Once the computed stage value \( Y_i \) has been accepted, we may want to export the derivatives \( f(Y_i) \) for the purpose of estimating error. We avoid a final evaluation of \( f(Y_i) \), not simply because we wish to save the cost of this evaluation, but also because stiffness would affect the accuracy of such a procedure [61]. Instead, we compute the \( f(Y_i) \) using

\[ hF = (A^{-1} \otimes I_N) (Y - (e \otimes I_N)y_n). \]

This corresponds to the \( P(EC)^n \) scheme for linear multistep methods (cf. [45]). In addition to the signal of divergence or convergence, the iterations should be terminated when the number of iterations exceeds the maximum allowed. For codes based on BDF, the iteration numbers are less than for IRK codes. Generally, 3 – 4 iterations are the maximum allowed for BDF codes. A reasonable
number for the codes based on IRK methods is 5. Another approach is to compute the number of iterations that is needed for convergence [34]. The algorithm for solving the internal stages is given as follows.

1. Algorithm: control of iteration

while it is not yet convergent AND it is not yet divergent AND number of iterations does not exceed the maximum allowed

1.1 Compute the internal stages
1.2 Form convergence rate

if it is divergent
1.3 Stop the iteration
else
1.4 Proceed with the convergence test

if it is convergent
1.41 Stop the iteration
1.42 AND update the internal stage value
1.43 AND compute the derivatives
endif
endif
endwhile

Unlike most Runge-Kutta methods, changing stepsize for ESIRK methods is more complicated due to the perturbed initial value for each step. In the case of fixed order, the perturbed initial values of the first step come from the starting methods. After that, if the code decides to change stepsize in the next step, the output approximation of this step has to be perturbed to suit the use of different stepsizes. These are achieved using the formula

\[ y_n = y_{n-1} + (b(r) \otimes I_N)hF, \]  

\[ (4.76) \]
where \( r \) is the stepsize changing ratio (refer to (4.46)). In practice, we have to be careful in dealing with the initial value of every step. The accuracy and convergence rate of the solutions are very sensitive to the initial input. Incorrect input will cause lower accuracy than we expect and it will be more difficult to attain convergence. In particular, when the present step is rejected (because the truncation error in the error estimate is too large or the iterations are divergent), we have to reproduce an input for the next step because of the new stepsize chosen. As a consequence, the storage of the previous initial value \( y_{n-1} \) and the derivative \( hF \) is a crucial aspect of the implementation of ESIRK methods. The key point is to store the stepsize \( h \), the perturbed input numerical solution \( y_{n-1} \) and the derivative \( hF \) for every accepted step. Recall the Jacobian and iteration matrix update algorithm; if we need to lower the value of \( h \), then we have to reproduce an input value using (4.76) with the new stepsize ratio \( r \). Let \( y_p, h_p, F_p \) denote the old copies of \( y, h, F \) of the previously accepted step. Below, we give the algorithm for the changing stepsize part of the error control.

1. **Algorithm: stepsize changing**

   if error is too large

1.1 Lower the value of \( h \)

1.2 Compute new stepsize ratio \( r = \frac{h}{h_p} \)

1.3 Compute a new input value \( y \) using \( y_p, F_p \) with stepsize ratio \( r \)

else

1.4 Update the integrated point \( x \)

1.5 Store the present stepsize \( h_p = h \)

1.6 Store the perturbed numerical solution \( y_p = y \)

1.7 Store the derivative \( F_p = F \)

1.8 Decide to use the same stepsize or a different stepsize for the next step

   if decide to use the same stepsize

1.81 Set stepsize ratio \( r = 1 \)

1.82 Compute a new input value \( y \) using \( y_p, F_p \) with \( r = 1 \)
1.83 else
1.84     Update stepsize \( h \)
1.85 Compute new stepsize ratio \( r = \frac{h}{h_p} \)
1.86 Compute a new input value \( y \) using \( y_p, F_p \) with stepsize ratio \( r \)
1.87 endif
1.88 endif

In the above algorithm, when the change in the stepsize ratio is moderate, we may want to use the same stepsize for the next step because it will then not be necessary to refactorize the iteration matrix. In our code, in addition to the first two steps (for the purpose of estimating error), when \( 0.9 \leq r \leq 1.2 \) or the previous step is rejected, the stepsize is kept the same.

Table 4.22: \( \frac{1}{\lambda} \) of \( A(\alpha)\)-stable ESIRK methods with \( \alpha \geq 1.45 \), the value of \( \alpha \) is given in parentheses

<table>
<thead>
<tr>
<th>order ( p )</th>
<th>( p = 1 )</th>
<th>( p = 2 )</th>
<th>( p = 3 )</th>
<th>( p = 4 )</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>( 1.000(\pi/2) )</td>
<td>( 0.586(\pi/2) )</td>
<td>( 0.416(1.56) )</td>
<td>( 0.323(1.54) )</td>
</tr>
<tr>
<td></td>
<td>( - )</td>
<td>( 3.414(\pi/2) )</td>
<td>( 2.294(\pi/2) )</td>
<td>( 1.746(\pi/2) )</td>
</tr>
<tr>
<td></td>
<td>( - )</td>
<td>( - )</td>
<td>( - )</td>
<td>( 4.537(1.56) )</td>
</tr>
<tr>
<td>order ( p )</td>
<td>( p = 5 )</td>
<td>( p = 6 )</td>
<td>( p = 7 )</td>
<td>( p = 8 )</td>
</tr>
<tr>
<td>-------------</td>
<td>----------------</td>
<td>----------------</td>
<td>----------------</td>
<td>----------------</td>
</tr>
<tr>
<td></td>
<td>( 0.264(1.52) )</td>
<td>( 0.223(1.51) )</td>
<td>( 0.193(1.50) )</td>
<td>( 0.170(1.49) )</td>
</tr>
<tr>
<td></td>
<td>( 1.413(1.57) )</td>
<td>( 1.189(1.57) )</td>
<td>( 1.027(1.57) )</td>
<td>( 0.904(1.56) )</td>
</tr>
<tr>
<td></td>
<td>( 3.596(\pi/2) )</td>
<td>( 2.993(\pi/2) )</td>
<td>( 2.568(1.57) )</td>
<td>( 2.251(1.57) )</td>
</tr>
<tr>
<td></td>
<td>( - )</td>
<td>( 5.775(1.49) )</td>
<td>( 4.900(1.57) )</td>
<td>( 4.267(\pi/2) )</td>
</tr>
</tbody>
</table>

As with SIRK methods, except for order-7, A-stable ESIRK methods are available. However, these A-stable methods may not be efficient enough because of their larger error constants. In section 3.4, the error constants of SIRK and DESI
methods have been shown in Table 3.4. The error constants for SIRK method of order 4, 6 and 7 in that table are $A(\alpha)$-stable. From [6], we list the zeros of the $p$-th order Laguerre polynomial in Table 4.22. The value in parentheses is the value of $\alpha$. $A$-stable method has $\alpha = \frac{\pi}{2}$. When the eigenvalue is chosen to be smaller (the zero of $L_\alpha$ is larger), the error constant is smaller. It is clear that all $A$-stable methods can have smaller error constants except for order 4 and 6.

**Experiment 4.8** We compare the order 4 and 6 $A$-stable methods with their corresponding $A(\alpha)$-stable methods for some stiff problems. The problems tested are Prothero-Robinson (2.5), Kaps(3.9), Robertson (2.6) and Van der Pol (2.7). The stiffness parameter for Van der Pol is chosen to be $10^6$. The work/precision diagrams for order 4 and 6 methods are shown in Figure 4.27 and Figure 4.28 respectively. The general impression is that for order 4 methods, except for the Prothero-Robinson problem, $A(\alpha)$-stable methods are more efficient than $A$-stable methods and perform better for higher precision. For order-6 methods, for all problems, except when using lower precision, $A(\alpha)$-stable methods also have better performance. It is not surprising to see that the $A(\alpha)$-stable methods are more efficient for higher precision because of their much smaller error constants (approximately one-quarter of the corresponding $A$-stable methods). We also note that for order-4, $A(\alpha)$-stable methods can compete with $A$-stable methods for the lower precisions, whereas it is not the case for order 6. The value of $\alpha$ for order 4 is 1.56 and for order 6 is 1.49 ($\frac{\pi}{2} \approx 1.57$) which probably explains the result.

**Experiment 4.9** Using the same test problems, some other numerical results we report are the comparisons between order 4, 5 and 6 $A$-stable methods. In Figure 4.29, we show the work/precision diagrams for each problem. As we expected, for all problems, the lower-order methods perform better than the higher-order methods in low precision. Because the error constants for order-5 and order-6 methods are quite similar, when the precision is tightened, the order-6 method seems not to have any advantage over the order-5 method. On the contrary, the order-5 method is more efficient overall than the order 6 method. From Figure
4.8. IMPLEMENTATION AND SOME NUMERICAL RESULTS

Figure 4.27: Work/Precision diagrams for order-4 ESIRK methods, A-stable: --, A(α = 1.56)-stable: ---

In 4.28, we note that the order-6 A(α)-stable method is more efficient than the order-5 method in high precision for these test problems.

Among the existing codes based on implicit Runge-Kutta method, the RADAU5 is one of the outstanding and reliable solvers for stiff problems. For stiff systems with small or large dimensions, the numerical results obtained by RADAU5 seem to be satisfactory. In particular, RADAU5 is competitive with codes based on BDF formulae for many problems [45], [52]. Below, we show some numerical results for (1) the Robertson problem (2.6) with integration interval [0, 10^{10}], (2) the Van der Pol problem (2.7) with the stiff parameter θ = 10^6, and (3) the Oreg-onator problem (4.77), using the IRK codes SDIRK and RADAU5 implemented by Hairer and Wanner [45] and the BDF code LSODE implemented by Hindmarsh [47] together with the A-stable and A(α)-stable order-4 ESIRK methods, A-stable order-5 method. The code SDIRK is based on the 5-stage fourth order
CHAPTER 4. THE EFFECTIVE ORDER OF SIRK METHODS

Figure 4.28: Work/Precision diagrams for order-6 ESIRK methods, A-stable: --, 
A(\( \alpha = 1.49 \))-stable: ---

SDIRK method (3.1).

**Experiment 4.10** For order-4 methods (SDIRK, ESIRK4, A(\( \alpha \))-stable ESIRK4), the numerical results for the Van der Pol oscillator are shown in Table 4.23. When the tolerance is large, the SDIRK performs better than the A(\( \alpha \))-stable ESIRK4 method, the two ESIRK4 methods are more efficient than SDIRK. The A(\( \alpha \))-stable method seems to be more efficient than the others in higher accuracy. The work/precision (fcns/error) diagram is shown in Figure 4.30. We note that the number of function evaluations for SDIRK increases rapidly when the tolerance becomes smaller. The additional fifth stage for SDIRK is part of the reason. Furthermore, the stage order for SDIRK methods is only one making the overall performance less satisfactory. 

**Experiment 4.11** The numerical results obtained using RADAU5, LSODE and
Figure 4.29: Work/Precision diagrams for order 4, 5, 6 ESIRK methods, order 4: +, order 5: o, order 6: x

ESIRK5 are shown in Table 4.24, Table 4.25 and Table 4.26. For the Robertson problem (2.6) and the Oregonator problem (4.77), we set $atol = 10^{-4}rtol$ for the ESIRK method ($atol = 10^{-6}rtol$ for RADAU5 and LSODE). Numerical results show that for the Robertson and the Van der Pol problem, RADAU5 performs better overall than ESIRK5 and LSODE, and the two IRK methods perform better than LSODE. It is not surprising that LSODE is faster than the others. We note that for the Robertson problem, when the tolerance is smaller than $10^{-4}$, the end point error obtained using LSODE does not decrease because of the round-off errors and this also happens in testing RADAU5 when the tolerance is less than $10^{-7}$. For the Oregonator problem, LSODE use fewer function evaluations, and RADAU5 performs better in high precision.

181
CHAPTER 4. THE EFFECTIVE ORDER OF SIRK METHODS

Figure 4.30: Work/precision diagram of ESIRK4 and $SDIRK_4$

Oregonator

$$
\begin{align*}
    y'_1 &= 77.27 \left( y_2 + y_1 \left( 1 - 8.375 \times 10^{-6} y_1 - y_2 \right) \right), & y_1(0) &= 1 \\
    y'_2 &= \frac{1}{77.27} \left( y_3 - (1 + y_1) y_2 \right), & y_2(0) &= 2 \\
    y'_3 &= 0.161 (y_1 - y_3), & y_2(0) &= 3 \\
    x &\in [0, 30] 
\end{align*}
$$

(4.77)

The work/precision diagrams are shown in Figure 4.31, Figure 4.32 and Figure 4.33. For lower precision, the over-estimation of the error makes $RADAU5$ more successful than ESIRK5. It is noted that when the tolerance decreases, increase
in the number of function evaluations for ESIRK5 is less than for RADAU5. For ESIRK5, the average iteration number per step is less than 1.5 for the Robertson problem and less than 1.8 for the Van der Pol problem and the Oregonator problem when $\text{tol} < 10^{-5}$. The reliable and consistent numerical performance ensures that ESIRK methods are reasonable candidates for stiff problems.

Although ESIRK methods are quite suitable for stiff problems, we still have at least the following three items which need improving.

(1) The lower efficiency in lower precision.

(2) The larger error constants for high order methods.
CHAPTER 4. THE EFFECTIVE ORDER OF SIRK METHODS

Figure 4.32: Work/precision diagram of ESIRK5, LSODE and RADAU5

(3) The inconvenience in estimating local truncation error.

The first item seems to be easy to overcome by using variable order. However, the third item gives another difficulty which is closely related to order changing for ESIRK methods. We realize that ESIRK methods need starting methods because of the perturbed initial values. Once the code shows a preference for a different order method, the perturbed initial value must be supplied by the information found in the previous step. For example, if $\alpha_1, \alpha_2, \alpha_3$ are the perturbations of the third-order method, then to change an order-2 method to an order-3 method at the integrating point $x_n$, the initial value

$$y_n = y(x_n) + \alpha_1 h y'(x_n) + \alpha_2 h^2 y''(x_n) + \alpha_3 h^3 y'''(x_n) + O(h^4),$$
must be produced for the new method. However, $h^3y'''(x_n)$ can be approximated using the local error estimation in the last step. Therefore a successful change of order is very dependent on the accuracy of the local error estimate. Furthermore, the first local error estimate for the new method also relies on the previous estimate because the way of estimating error is to use the difference of $h^py^{(p)}(x_n)$ between two consecutive steps (refer to Theorem 4.24 and Lemma 4.25). The DESI formulae have extra stages making the error estimate easier (for both step-size changing and order changing) and have smaller error constants. In the next chapter, we propose to apply effective order to DESI formulae and analyze the influence caused by effective order.
Table 4.23: Numerical results for Van der Pol (2.7) by testing order-4 L-stable, A(\(\alpha\))-stable ESIRKs and order-4 L-stable SDIRK

<table>
<thead>
<tr>
<th>tol</th>
<th>nsteps</th>
<th>rej</th>
<th>nfcn</th>
<th>njac</th>
<th>lu</th>
<th>endglberr</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>L-stable ESIRK4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10^{-2}</td>
<td>256</td>
<td>46</td>
<td>2576</td>
<td>181</td>
<td>280</td>
<td>4.54 \times 10^{-3}</td>
</tr>
<tr>
<td>10^{-3}</td>
<td>333</td>
<td>25</td>
<td>2216</td>
<td>232</td>
<td>318</td>
<td>3.19 \times 10^{-4}</td>
</tr>
<tr>
<td>10^{-4}</td>
<td>483</td>
<td>18</td>
<td>2640</td>
<td>319</td>
<td>424</td>
<td>2.79 \times 10^{-4}</td>
</tr>
<tr>
<td>10^{-5}</td>
<td>712</td>
<td>14</td>
<td>3636</td>
<td>400</td>
<td>559</td>
<td>2.53 \times 10^{-5}</td>
</tr>
<tr>
<td>10^{-6}</td>
<td>1083</td>
<td>12</td>
<td>5084</td>
<td>545</td>
<td>750</td>
<td>5.91 \times 10^{-6}</td>
</tr>
<tr>
<td>10^{-7}</td>
<td>1663</td>
<td>10</td>
<td>7504</td>
<td>811</td>
<td>1040</td>
<td>1.24 \times 10^{-6}</td>
</tr>
<tr>
<td>10^{-8}</td>
<td>2595</td>
<td>7</td>
<td>11324</td>
<td>1152</td>
<td>1414</td>
<td>1.81 \times 10^{-7}</td>
</tr>
<tr>
<td>10^{-9}</td>
<td>4082</td>
<td>5</td>
<td>17528</td>
<td>1645</td>
<td>2001</td>
<td>2.82 \times 10^{-8}</td>
</tr>
<tr>
<td>A((\alpha))-stable ESIRK4, (\alpha = 1.56)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10^{-2}</td>
<td>177</td>
<td>30</td>
<td>2304</td>
<td>155</td>
<td>202</td>
<td>1.26 \times 10^{-2}</td>
</tr>
<tr>
<td>10^{-3}</td>
<td>221</td>
<td>43</td>
<td>2924</td>
<td>190</td>
<td>248</td>
<td>2.82 \times 10^{-4}</td>
</tr>
<tr>
<td>10^{-4}</td>
<td>295</td>
<td>37</td>
<td>2920</td>
<td>255</td>
<td>311</td>
<td>3.88 \times 10^{-5}</td>
</tr>
<tr>
<td>10^{-5}</td>
<td>412</td>
<td>24</td>
<td>3240</td>
<td>350</td>
<td>402</td>
<td>1.65 \times 10^{-5}</td>
</tr>
<tr>
<td>10^{-6}</td>
<td>611</td>
<td>27</td>
<td>4372</td>
<td>514</td>
<td>571</td>
<td>2.99 \times 10^{-6}</td>
</tr>
<tr>
<td>10^{-7}</td>
<td>896</td>
<td>13</td>
<td>5616</td>
<td>742</td>
<td>795</td>
<td>4.16 \times 10^{-7}</td>
</tr>
<tr>
<td>10^{-8}</td>
<td>1373</td>
<td>18</td>
<td>7720</td>
<td>1126</td>
<td>1182</td>
<td>7.26 \times 10^{-8}</td>
</tr>
<tr>
<td>10^{-9}</td>
<td>2124</td>
<td>7</td>
<td>11128</td>
<td>1711</td>
<td>1768</td>
<td>1.78 \times 10^{-8}</td>
</tr>
<tr>
<td>SDIRK</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10^{-2}</td>
<td>137</td>
<td>20</td>
<td>1663</td>
<td>91</td>
<td>180</td>
<td>1.40 \times 10^{-3}</td>
</tr>
<tr>
<td>10^{-3}</td>
<td>194</td>
<td>32</td>
<td>2481</td>
<td>114</td>
<td>220</td>
<td>2.92 \times 10^{-4}</td>
</tr>
<tr>
<td>10^{-4}</td>
<td>325</td>
<td>33</td>
<td>3777</td>
<td>180</td>
<td>293</td>
<td>3.08 \times 10^{-5}</td>
</tr>
<tr>
<td>10^{-5}</td>
<td>640</td>
<td>22</td>
<td>6755</td>
<td>305</td>
<td>459</td>
<td>2.54 \times 10^{-5}</td>
</tr>
<tr>
<td>10^{-6}</td>
<td>1432</td>
<td>24</td>
<td>13515</td>
<td>595</td>
<td>789</td>
<td>1.63 \times 10^{-6}</td>
</tr>
<tr>
<td>10^{-7}</td>
<td>3542</td>
<td>12</td>
<td>31043</td>
<td>1296</td>
<td>1482</td>
<td>2.26 \times 10^{-7}</td>
</tr>
<tr>
<td>10^{-8}</td>
<td>9268</td>
<td>7</td>
<td>78201</td>
<td>2157</td>
<td>3405</td>
<td>2.96 \times 10^{-7}</td>
</tr>
<tr>
<td>10^{-9}</td>
<td>–</td>
<td>8</td>
<td>–</td>
<td>3302</td>
<td>8003</td>
<td>3.14 \times 10^{-7}</td>
</tr>
</tbody>
</table>

186
4.8. IMPLEMENTATION AND SOME NUMERICAL RESULTS

Table 4.24: Numerical results for Van der Pol (2.7) problem by testing LSODE, RADAU5 and order-5 L-stable ESIRK

<table>
<thead>
<tr>
<th>tol</th>
<th>nsteps</th>
<th>rej</th>
<th>nfcn</th>
<th>njac</th>
<th>lu</th>
<th>endglberr</th>
</tr>
</thead>
<tbody>
<tr>
<td>(10^{-2})</td>
<td>969</td>
<td>-</td>
<td>1433</td>
<td>154</td>
<td>-</td>
<td>(7.80 \times 10^{-2})</td>
</tr>
<tr>
<td>(10^{-3})</td>
<td>1338</td>
<td>-</td>
<td>1860</td>
<td>159</td>
<td>-</td>
<td>(5.69 \times 10^{-3})</td>
</tr>
<tr>
<td>(10^{-4})</td>
<td>1939</td>
<td>-</td>
<td>2566</td>
<td>204</td>
<td>-</td>
<td>(1.17 \times 10^{-3})</td>
</tr>
<tr>
<td>(10^{-5})</td>
<td>2794</td>
<td>-</td>
<td>3464</td>
<td>272</td>
<td>-</td>
<td>(8.86 \times 10^{-5})</td>
</tr>
<tr>
<td>(10^{-6})</td>
<td>4023</td>
<td>-</td>
<td>4830</td>
<td>342</td>
<td>-</td>
<td>(1.53 \times 10^{-5})</td>
</tr>
<tr>
<td>(10^{-7})</td>
<td>5739</td>
<td>-</td>
<td>6742</td>
<td>424</td>
<td>-</td>
<td>(1.43 \times 10^{-6})</td>
</tr>
<tr>
<td>(10^{-8})</td>
<td>8413</td>
<td>-</td>
<td>9881</td>
<td>631</td>
<td>-</td>
<td>(1.48 \times 10^{-7})</td>
</tr>
<tr>
<td>(10^{-9})</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>859</td>
<td>-</td>
<td>(2.98 \times 10^{-8})</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>tol</th>
<th>nsteps</th>
<th>rej</th>
<th>nfcn</th>
<th>njac</th>
<th>lu</th>
<th>endglberr</th>
</tr>
</thead>
<tbody>
<tr>
<td>(10^{-2})</td>
<td>206</td>
<td>40</td>
<td>2730</td>
<td>191</td>
<td>226</td>
<td>(1.81 \times 10^{-3})</td>
</tr>
<tr>
<td>(10^{-3})</td>
<td>242</td>
<td>45</td>
<td>3280</td>
<td>207</td>
<td>273</td>
<td>(1.84 \times 10^{-4})</td>
</tr>
<tr>
<td>(10^{-4})</td>
<td>281</td>
<td>30</td>
<td>3080</td>
<td>240</td>
<td>302</td>
<td>(7.63 \times 10^{-6})</td>
</tr>
<tr>
<td>(10^{-5})</td>
<td>381</td>
<td>20</td>
<td>3525</td>
<td>322</td>
<td>379</td>
<td>(2.96 \times 10^{-6})</td>
</tr>
<tr>
<td>(10^{-6})</td>
<td>501</td>
<td>16</td>
<td>4305</td>
<td>426</td>
<td>486</td>
<td>(3.72 \times 10^{-7})</td>
</tr>
<tr>
<td>(10^{-7})</td>
<td>699</td>
<td>14</td>
<td>5345</td>
<td>589</td>
<td>643</td>
<td>(3.25 \times 10^{-7})</td>
</tr>
<tr>
<td>(10^{-8})</td>
<td>986</td>
<td>10</td>
<td>6745</td>
<td>817</td>
<td>872</td>
<td>(4.25 \times 10^{-8})</td>
</tr>
<tr>
<td>(10^{-9})</td>
<td>1415</td>
<td>10</td>
<td>9325</td>
<td>1180</td>
<td>1276</td>
<td>(6.71 \times 10^{-9})</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>tol</th>
<th>nsteps</th>
<th>rej</th>
<th>nfcn</th>
<th>njac</th>
<th>lu</th>
<th>endglberr</th>
</tr>
</thead>
<tbody>
<tr>
<td>(10^{-2})</td>
<td>140</td>
<td>6</td>
<td>1348</td>
<td>95</td>
<td>161</td>
<td>(3.22 \times 10^{-4})</td>
</tr>
<tr>
<td>(10^{-3})</td>
<td>183</td>
<td>7</td>
<td>1663</td>
<td>122</td>
<td>205</td>
<td>(1.29 \times 10^{-4})</td>
</tr>
<tr>
<td>(10^{-4})</td>
<td>246</td>
<td>7</td>
<td>2242</td>
<td>163</td>
<td>252</td>
<td>(8.34 \times 10^{-6})</td>
</tr>
<tr>
<td>(10^{-5})</td>
<td>340</td>
<td>8</td>
<td>2954</td>
<td>221</td>
<td>310</td>
<td>(2.13 \times 10^{-6})</td>
</tr>
<tr>
<td>(10^{-6})</td>
<td>488</td>
<td>8</td>
<td>3985</td>
<td>306</td>
<td>411</td>
<td>(2.94 \times 10^{-7})</td>
</tr>
<tr>
<td>(10^{-7})</td>
<td>712</td>
<td>7</td>
<td>5751</td>
<td>435</td>
<td>587</td>
<td>(1.82 \times 10^{-8})</td>
</tr>
<tr>
<td>(10^{-8})</td>
<td>1046</td>
<td>6</td>
<td>8281</td>
<td>483</td>
<td>842</td>
<td>(2.26 \times 10^{-9})</td>
</tr>
<tr>
<td>(10^{-9})</td>
<td>1541</td>
<td>4</td>
<td>12019</td>
<td>530</td>
<td>1194</td>
<td>(3.66 \times 10^{-10})</td>
</tr>
</tbody>
</table>

187
Table 4.25: Numerical results for Robertson problem (2.6) by testing LSODE, RADAU5 and order-5 L-stable ESIRK

<table>
<thead>
<tr>
<th>$tol$</th>
<th>nsteps</th>
<th>rej</th>
<th>nfcn</th>
<th>njac</th>
<th>lu</th>
<th>endglberr</th>
</tr>
</thead>
<tbody>
<tr>
<td>$10^{-1}$</td>
<td>81</td>
<td>–</td>
<td>119</td>
<td>40</td>
<td>–</td>
<td>$1.13 \times 10^{-3}$</td>
</tr>
<tr>
<td>$10^{-2}$</td>
<td>133</td>
<td>–</td>
<td>176</td>
<td>41</td>
<td>–</td>
<td>$6.12 \times 10^{-5}$</td>
</tr>
<tr>
<td>$10^{-3}$</td>
<td>223</td>
<td>–</td>
<td>296</td>
<td>60</td>
<td>–</td>
<td>$4.50 \times 10^{-7}$</td>
</tr>
<tr>
<td>$10^{-4}$</td>
<td>328</td>
<td>–</td>
<td>402</td>
<td>68</td>
<td>–</td>
<td>$1.84 \times 10^{-6}$</td>
</tr>
<tr>
<td>$10^{-5}$</td>
<td>496</td>
<td>–</td>
<td>673</td>
<td>88</td>
<td>–</td>
<td>$1.91 \times 10^{-6}$</td>
</tr>
<tr>
<td>$10^{-6}$</td>
<td>736</td>
<td>–</td>
<td>965</td>
<td>118</td>
<td>–</td>
<td>$1.86 \times 10^{-6}$</td>
</tr>
<tr>
<td>$10^{-7}$</td>
<td>1082</td>
<td>–</td>
<td>1468</td>
<td>172</td>
<td>–</td>
<td>$1.88 \times 10^{-6}$</td>
</tr>
<tr>
<td>$10^{-2}$</td>
<td>129</td>
<td>6</td>
<td>1355</td>
<td>128</td>
<td>134</td>
<td>$5.64 \times 10^{-10}$</td>
</tr>
<tr>
<td>$10^{-3}$</td>
<td>122</td>
<td>3</td>
<td>1065</td>
<td>118</td>
<td>123</td>
<td>$2.63 \times 10^{-10}$</td>
</tr>
<tr>
<td>$10^{-4}$</td>
<td>143</td>
<td>1</td>
<td>1165</td>
<td>136</td>
<td>143</td>
<td>$1.33 \times 10^{-10}$</td>
</tr>
<tr>
<td>$10^{-5}$</td>
<td>182</td>
<td>4</td>
<td>1385</td>
<td>176</td>
<td>184</td>
<td>$8.02 \times 10^{-11}$</td>
</tr>
<tr>
<td>$10^{-6}$</td>
<td>244</td>
<td>1</td>
<td>1730</td>
<td>234</td>
<td>244</td>
<td>$1.74 \times 10^{-11}$</td>
</tr>
<tr>
<td>$10^{-7}$</td>
<td>331</td>
<td>0</td>
<td>2175</td>
<td>319</td>
<td>328</td>
<td>$3.77 \times 10^{-12}$</td>
</tr>
<tr>
<td>$10^{-8}$</td>
<td>463</td>
<td>2</td>
<td>2835</td>
<td>410</td>
<td>429</td>
<td>$6.61 \times 10^{-13}$</td>
</tr>
<tr>
<td>$10^{-9}$</td>
<td>656</td>
<td>1</td>
<td>3805</td>
<td>526</td>
<td>548</td>
<td>$1.12 \times 10^{-13}$</td>
</tr>
<tr>
<td>$10^{-2}$</td>
<td>82</td>
<td>1</td>
<td>574</td>
<td>74</td>
<td>82</td>
<td>$1.41 \times 10^{-10}$</td>
</tr>
<tr>
<td>$10^{-3}$</td>
<td>104</td>
<td>1</td>
<td>722</td>
<td>99</td>
<td>104</td>
<td>$4.70 \times 10^{-11}$</td>
</tr>
<tr>
<td>$10^{-4}$</td>
<td>136</td>
<td>2</td>
<td>985</td>
<td>128</td>
<td>138</td>
<td>$3.82 \times 10^{-12}$</td>
</tr>
<tr>
<td>$10^{-5}$</td>
<td>184</td>
<td>1</td>
<td>1348</td>
<td>175</td>
<td>185</td>
<td>$2.82 \times 10^{-13}$</td>
</tr>
<tr>
<td>$10^{-6}$</td>
<td>253</td>
<td>1</td>
<td>1909</td>
<td>240</td>
<td>253</td>
<td>$2.46 \times 10^{-14}$</td>
</tr>
<tr>
<td>$10^{-7}$</td>
<td>358</td>
<td>2</td>
<td>2728</td>
<td>334</td>
<td>347</td>
<td>$2.00 \times 10^{-15}$</td>
</tr>
<tr>
<td>$10^{-8}$</td>
<td>512</td>
<td>2</td>
<td>3896</td>
<td>368</td>
<td>390</td>
<td>$1.33 \times 10^{-15}$</td>
</tr>
<tr>
<td>$10^{-9}$</td>
<td>744</td>
<td>1</td>
<td>5655</td>
<td>395</td>
<td>428</td>
<td>$1.22 \times 10^{-15}$</td>
</tr>
</tbody>
</table>
### 4.8. IMPLEMENTATION AND SOME NUMERICAL RESULTS

Table 4.26: Numerical results for Oregonator (4.77) by testing *LSODE*, *RADAU5* and order-5 L-stable ESIRK

<table>
<thead>
<tr>
<th>tol</th>
<th>nsteps</th>
<th>rej</th>
<th>nfcn</th>
<th>njac</th>
<th>lu</th>
<th>endglberr</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$10^{-3}$</td>
<td>271</td>
<td>-</td>
<td>461</td>
<td>63</td>
<td>-</td>
<td>$4.24 \times 10^1$</td>
</tr>
<tr>
<td>$10^{-4}$</td>
<td>426</td>
<td>-</td>
<td>656</td>
<td>76</td>
<td>-</td>
<td>$7.33 \times 10^0$</td>
</tr>
<tr>
<td>$10^{-5}$</td>
<td>588</td>
<td>-</td>
<td>882</td>
<td>104</td>
<td>-</td>
<td>$4.61 \times 10^{-1}$</td>
</tr>
<tr>
<td>$10^{-6}$</td>
<td>805</td>
<td>-</td>
<td>1106</td>
<td>114</td>
<td>-</td>
<td>$7.86 \times 10^{-2}$</td>
</tr>
<tr>
<td>$10^{-7}$</td>
<td>1029</td>
<td>-</td>
<td>1380</td>
<td>120</td>
<td>-</td>
<td>$9.28 \times 10^{-3}$</td>
</tr>
<tr>
<td>$10^{-8}$</td>
<td>1511</td>
<td>-</td>
<td>1942</td>
<td>148</td>
<td>-</td>
<td>$9.64 \times 10^{-4}$</td>
</tr>
<tr>
<td>$10^{-9}$</td>
<td>2099</td>
<td>-</td>
<td>2636</td>
<td>212</td>
<td>-</td>
<td>$1.74 \times 10^{-4}$</td>
</tr>
<tr>
<td>$10^{-10}$</td>
<td>3052</td>
<td>-</td>
<td>3770</td>
<td>293</td>
<td>-</td>
<td>$1.17 \times 10^{-5}$</td>
</tr>
</tbody>
</table>

### LSODE

| $10^{-3}$ | 117    | 13  | 1675 | 97   | 131 | $7.82 \times 10^{-1}$ |
| $10^{-4}$ | 149    | 19  | 1960 | 128  | 164 | $3.74 \times 10^{-1}$ |
| $10^{-5}$ | 196    | 14  | 2125 | 172  | 203 | $8.18 \times 10^{-2}$ |
| $10^{-6}$ | 264    | 14  | 2575 | 237  | 266 | $1.38 \times 10^{-2}$ |
| $10^{-7}$ | 364    | 6   | 3050 | 328  | 351 | $2.43 \times 10^{-3}$ |
| $10^{-8}$ | 523    | 6   | 4000 | 451  | 493 | $3.06 \times 10^{-4}$ |
| $10^{-9}$ | 759    | 6   | 5405 | 607  | 692 | $3.87 \times 10^{-5}$ |
| $10^{-10}$ | 1101  | 5   | 7455 | 806  | 972 | $5.20 \times 10^{-6}$ |

### ESIRK5

| $10^{-3}$ | 98     | 4   | 1052 | 86   | 116 | $2.11 \times 10^{-2}$ |
| $10^{-4}$ | 132    | 6   | 1368 | 118  | 149 | $6.06 \times 10^{-2}$ |
| $10^{-5}$ | 178    | 4   | 1726 | 162  | 192 | $1.38 \times 10^{-2}$ |
| $10^{-6}$ | 251    | 5   | 2369 | 229  | 262 | $2.86 \times 10^{-3}$ |
| $10^{-7}$ | 359    | 4   | 3266 | 304  | 356 | $5.04 \times 10^{-5}$ |
| $10^{-8}$ | 520    | 3   | 4537 | 381  | 491 | $1.14 \times 10^{-5}$ |
| $10^{-9}$ | 759    | 3   | 6375 | 456  | 676 | $1.41 \times 10^{-6}$ |
| $10^{-10}$ | 1112   | 2   | 9035 | 551  | 935 | $2.02 \times 10^{-7}$ |

| $10^{-3}$ | 98     | 4   | 1052 | 86   | 116 | $2.11 \times 10^{-2}$ |
| $10^{-4}$ | 132    | 6   | 1368 | 118  | 149 | $6.06 \times 10^{-2}$ |
| $10^{-5}$ | 178    | 4   | 1726 | 162  | 192 | $1.38 \times 10^{-2}$ |
| $10^{-6}$ | 251    | 5   | 2369 | 229  | 262 | $2.86 \times 10^{-3}$ |
| $10^{-7}$ | 359    | 4   | 3266 | 304  | 356 | $5.04 \times 10^{-5}$ |
| $10^{-8}$ | 520    | 3   | 4537 | 381  | 491 | $1.14 \times 10^{-5}$ |
| $10^{-9}$ | 759    | 3   | 6375 | 456  | 676 | $1.41 \times 10^{-6}$ |
| $10^{-10}$ | 1112   | 2   | 9035 | 551  | 935 | $2.02 \times 10^{-7}$ |
Chapter 5

The design of an EDESI integrator

There are at least two reasons which make ESIRK methods a good generalization of the classical SIRK methods. The first one is that we can choose any $s$ distinct abscissae for SIRK methods. The second one is that the local truncation error can be reduced by applying a variable stepsize scheme to ESIRK methods. However, due to the stability requirement, the error constants for ESIRK (SIRK) methods are not small enough to make them efficient. It is desirable to look for alternative methods which preserve the one point spectrum feature and have a smaller error constant. As shown in section 3.4, the DESI formulae are good extensions of the SIRK methods because of the smaller error constants which allow larger stepsizes to be taken. L-stable DESI methods are available for every order. Besides, DESI also has advantages in estimating the error.

In this chapter, these two generalizations are combined into a unified scheme. It is hoped that they may compensate for each other's weakness by using the strengths of these two types of methods. We call these new methods: "EDESI" (Effective order Diagonally Extended Singly Implicit Runge-Kutta methods).

In the first section, we discuss how to construct EDESI methods and give some examples. Variable stepsize is a crucial part in effective order methods. We will focus on a variable stepsize scheme for EDESI methods in the second section.
Some implementations including an error estimation, and the starting value prediction for Newton iterations will be the main content of the third section. In the final section, we intend to make comparisons between some members of the singly-implicit family. In order to see the influence caused by effective order, we will test some stiff problems with SIRK, DESI, ESIRK, EDESI methods.

5.1 Construction of EDESI methods

While free parameters of ESIRK methods come from their perturbed input approximations, DESI methods have extra parameters because of their additional diagonal stages. It can be expected that we will have more freedom in choosing the coefficients for EDESI methods. The question is how to make the best use of these extra parameters. Before discussing details about deriving the coefficients, some features of EDESI methods are shown below.

(1) New methods have the same structures as DESI methods (including the singly-implicit part and the diagonally-implicit part).

(2) New methods have the singly-implicitness property.

(3) The singly-implicit part of EDESI is an ESIRK method.

(4) The additional diagonal stages retain the same stage order as the singly-implicit part.

(5) EDESIs have the same stability property as their associated DESIs.

The corresponding tableau for EDESI methods is the same as DESI, refer to (3.23) in section 3.4. The coefficients for the singly ESIRK block can be derived in the same way as the ESIRK methods. In other words, given abscissae $c_1, c_2, \ldots, c_p$ and an eigenvalue $\lambda$ for stability, the coefficient matrix of the singly ESIRK block is obtained by the modified stage order conditions (4.17). Having the same stability as DESI, EDESI methods have the same eigenvalue as the corresponding DESI methods. Hence, $\lambda$ can be chosen from [23] or [34] to fulfill this requirement.
5.1. CONSTRUCTION OF EDESI METHODS

If we consider the p-th order EDESI methods with number of stages \( s = p + 1 \) only, the additional diagonal stage can be determined immediately once \( c_{p+1} \) is assigned because in this case the stage order is \( p \) and the number of the order conditions is \( p \). We have only \( a_{p+1,1}, a_{p+1,2}, \ldots, a_{p+1,p} \) to specify (note that \( a_{p+1,p+1} = \lambda \)). From the weight vector \( b^T \), we have \( p + 1 \) unknowns to be specified. As with ESIRK methods, we have to modify \( b^T \) so as to produce the perturbed initial value. From (4.21) in Theorem 4.10, there are \( p \) conditions to be satisfied,

\[
\sum_{i=1}^{p+1} b_i \frac{c_k^{k-1}}{(k-1)!} = \sum_{j=1}^{k} \frac{1}{(k-j+1)!} \alpha_{j-1} \quad \alpha_0 = 1, \quad k = 1, 2, \ldots, p.
\]

Another condition comes from the stability constraint. We note that the EDESI methods are of order \( p \), but we have \( p + 1 \) stages, so the numerator of the stability function satisfying the order conditions is just up to the term \( z^p \). To obtain L-stability, we need to ensure the coefficient of \( z^{p+1} \) is 0. Due to this constraint, we can determine the weights \( b^T \) uniquely. We now summarize the procedures for constructing order-\( p \) EDESI methods with \( s = p + 1 \) stages.

1. When \( c_1, c_2, \ldots, c_p, c_s = 1 \) are distinct, and \( s = p + 1 \), choose the eigenvalue \( \lambda \) for stability from [23].

2. Use (4.35) and (4.36) in Theorem 4.15 to find the perturbations \( \alpha_i, i = 1, \ldots, p \).

3. Use the modified order conditions \( C(p) \).

\[
A \frac{c^{k-1}}{(k-1)!} + \alpha_k e = \frac{c^k}{k!}, \quad k = 1, 2, \ldots, p,
\]

to find the coefficient matrix \( A \);

4. The weights \( b^T \) will be obtained by

\[
[b_1, b_2, \ldots, b_s]W = \left[ 1, \frac{1}{2} + \alpha_1, \cdots, \frac{1}{p!} + \frac{1}{(p-1)!} \alpha_1 + \cdots, \alpha_{p-1} \right], \quad (5.1)
\]

where \( W \) is the modified Vandermonde matrix. The other condition comes from the \( P(z) \) of the stability function, \( R(z) = \frac{P(z)}{Q(z)} \). Hence, the coefficient of \( z^{p+1} \) must be zero, and \( Q(z) = (1 - \lambda z)^a \).
We give an example for \( s = p + 1 \) below.

**Example 5.1** Let \( p = 2 \). In this case, the related Butcher tableau is of the form

\[
\begin{array}{c|ccc}
 c_1 & a_{11} & a_{12} & 0 \\
 c_2 & a_{21} & a_{22} & 0 \\
 1 & a_{31} & a_{32} & \lambda \\
 \hline
 b_1 & b_2 & b_3
\end{array}
\]

(5.2)

If we choose the eigenvalue \( \lambda = \frac{2}{11} \) (an approximation to \( \frac{1}{5.5424597568} \)) from [29] and the abscissae \( c_1 = 0, c_2 = \frac{1}{2}, c_3 = 1 \), the perturbations \( \alpha_1, \alpha_2 \) can be derived using (4.35) and (4.36),

\[
\alpha_1 = \frac{5}{44}, \quad \alpha_2 = \frac{9}{1936}.
\]

From the modified stage order conditions,

\[
\begin{align*}
 a_{11} + a_{12} + \alpha_1 &= c_1, & a_{11}c_1 + a_{12}c_2 + \alpha_2 &= \frac{c_1^2}{2}, \\
 a_{21} + a_{22} + \alpha_1 &= c_2, & a_{21}c_1 + a_{22}c_2 + \alpha_2 &= \frac{c_2^2}{2}, \\
 a_{31} + a_{32} + \alpha_1 &= c_3, & a_{31}c_1 + a_{32}c_2 + \lambda c_3 + \alpha_2 &= \frac{c_3^2}{2},
\end{align*}
\]

we obtain

\[
\begin{align*}
 a_{11} &= \frac{119}{968}, & a_{12} &= \frac{-9}{968}, & a_{21} &= \frac{361}{968}, \\
 a_{22} &= \frac{233}{968}, & a_{31} &= \frac{295}{968}, & a_{32} &= \frac{607}{968}.
\end{align*}
\]

The weights \( b_1, b_2, b_3 \) are obtained using

\[
\begin{align*}
 b_1 + b_2 + b_3 &= 1, \\
 b_1c_1 + b_2c_2 + b_3c_3 &= \frac{1}{2} + \alpha_1,
\end{align*}
\]

(5.3) \hspace{1cm} (5.4)

and

\[
1 + zb^T(I_3 - zA)^{-1}e = \frac{P(z)}{(1 - \lambda z)^3},
\]

(5.5)
where
\[ P(z) = 1 + \gamma_1 z + \gamma_2 z^2 + \gamma_3 z^3, \]
and \( \gamma_3 = 0 \). It follows that
\[ b_1 = \frac{933}{2662}, \quad b_2 = \frac{1401}{2662}, \quad b_3 = \frac{164}{1331}. \]
In this case, the error constant is
\[ C_3 = (-1)^3 \lambda^3 L_3 \left( \frac{1}{\lambda} \right) = -0.01289757074881042. \]

Applying effective order to a \( p \)-th order DESI method with \( p + 1 \) stages, the immediate advantage is that the abscissae \( c_1, c_2, \ldots, c_p \) can be chosen as we wish (we need only that \( c_1, c_2, \ldots, c_p \) are distinct). Furthermore, by using the one additional diagonal stage, an EDESI method has the advantage over an ESIRK method because it is easier to construct a local error estimator. Therefore, it is natural to follow this idea and to look for the case of \( s = p + 2 \). In this case, we use a similar approach to \( s = p + 1 \). The first \( p + 1 \) rows of the coefficient matrix can easily be obtained by stage order conditions once the eigenvalue \( \lambda \) is assigned. For the last row, the \( p + 1 \) parameters that need to be specified are
\[ a_{p+2,1}, \quad a_{p+2,2}, \quad \ldots, \quad a_{p+2,p+1}. \]
There are only \( p \) independent order conditions for \( Y_{p+2} \) to be of order \( p \) because of the modified \( C(p) \) condition. We have one free parameter left because we do not need to use this parameter to fulfill the stability constraint. The reason is that unlike the corresponding DESI formulae, the weight vector \( b^T \) is the last row of \( A \) for an EDESI method. In addition to satisfying the order conditions, there are still some free parameters left for \( b^T \) to satisfy the stability requirement. In practice, we propose to use this free parameter to estimate the error for a variable order implementation. Therefore, we will show how to use this parameter, say \( a_{p+2,p+1} \), to estimate the error \( h^{p+2} y^{(p+2)}(x_n) \) in the following.
We assume that $d_i$, for all $i = 1, \ldots, s$ satisfies
\[
\sum_{i=1}^{s} d_i h f(Y_i) = h^{p+2} y^{(p+2)}(x_n) + O(h^{p+3}),
\] (5.6)
and if $\delta_i$ is the error constant of the $i$-th stage, $i = 1, \ldots, s$, then
\[
Y_i = y(x_{n-1} + c_i h) + \delta_i h^{p+1} y^{(p+1)}(x_{n-1}) + O(h^{p+2}), \quad i = 1, \ldots, s.
\] (5.7)
If $y_{n-1}$ is the input value for the $n$-th step, then we have
\[
Y_i = y_{n-1} + h \sum_{j=1}^{s} a_{ij} f(Y_j)
\]
\[
= y_{n-1} + \sum_{j=1}^{s} a_{ij} h y'(x_{n-1} + c_j h) + O(h^{p+2})
\]
\[
= y_{n-1} + \sum_{j=1}^{s} a_{ij} \sum_{k=1}^{p} \frac{c_j^{k-1}}{(k-1)!} h^k y^{(k)}(x_{n-1}) + O(h^{p+2})
\]
\[
= y_{n-1} + \sum_{j=1}^{s} a_{ij} \left( \sum_{k=1}^{p} \frac{c_j^{k-1}}{(k-1)!} h^k y^{(k)}(x_{n-1}) + \frac{c_j^p}{p!} h^{p+1} y^{(p+1)}(x_{n-1}) \right) + O(h^{p+2})
\]
\[
= y_{n-1} + \sum_{k=1}^{p} \frac{c_j^k}{k!} h^k y^{(k)}(x_{n-1}) + \sum_{j=1}^{s} a_{ij} \frac{c_j^p}{p!} h^{p+1} y^{(p+1)}(x_{n-1}) + O(h^{p+2}),
\]
and
\[
y(x_{n-1} + c_i h) = y_{n-1} + \sum_{k=1}^{p} \frac{c_i^k}{k!} h^k y^{(k)}(x_{n-1})
\]
\[
+ \frac{c_i^{p+1}}{(p+1)!} h^{(p+1)} y^{(p+1)}(x_{n-1}) + O(h^{p+2}).
\] (5.8)
It follows that
\[
\delta_i = \sum_{j=1}^{s} a_{ij} \frac{c_j^p}{p!} - \frac{c_i^{p+1}}{(p+1)!}.
\] (5.9)
Also from (5.6), (5.7), and (5.9), it follows that
\[
h f(Y_i) = h y'(x_{n-1} + c_i h) + h \frac{\partial f(y_{n-1})}{\partial y} \delta_i h^{p+1} y^{(p+1)}(x_{n-1}) + O(h^{p+3}).
\]
Therefore we have
\[\sum_{i=1}^{s} d_i h f(Y_i) = \sum_{i=1}^{s} d_i h y'(x_{n-1} + c_i h) + h \sum_{i=1}^{s} d_i \frac{\partial f(y_{n-1})}{\partial y}.
\]
\[= \sum_{i=1}^{s} d_i \sum_{k=1}^{p+2} \frac{c_k^{k-1}}{(k-1)!} h^k y(k)(x_{n-1}) + \frac{\partial f(y_{n-1})}{\partial y} \sum_{i=1}^{s} d_i.
\]
\[\cdot \left(y(x_{n-1} + c_i h) + \delta_i h^{p+1} y^{(p+1)}(x_{n-1}) - y_{n-2} - h \sum_{j=1}^{s} a_{ij} y'(x_{n-1} + c_j h)\right) + O(h^{p+3}).
\]
\[= \sum_{i=1}^{s} \sum_{k=1}^{p+2} \frac{c_k^{k-1}}{(k-1)!} h^k y(k)(x_{n-1}) + \frac{\partial f(y_{n-1})}{\partial y} \sum_{i=1}^{s} d_i.
\]
\[\cdot \left(h y(x_{n-1} + c_i h) + \delta_i h^{p+2} y^{(p+1)}(x_{n-1}) - h y_{n-2} - h^2 \sum_{j=1}^{p+2} a_{ij} y'(x_{n-1} + c_j h)\right) + O(h^{p+3}).
\]

Our aim is to make every term on the right hand side of the above equations vanish except for \(h^{p+2} y^{(p+2)}(x_{n-1}) + O(h^{p+3})\). This can be done by assuming
\[
\begin{align*}
\sum_{i=1}^{s} \frac{d_i}{k!} c_k^{k-1} &= 0, \\
\sum_{i=1}^{s} \frac{d_i}{(p+1)!} c_{p+1}^{p+1} &= 0,
\end{align*}
\]
which is equivalent to
\[
\begin{align*}
d^T c^{k-1} &= 0, \\
&\quad k = 1, \ldots, p + 1, \\
d^T c^{p+1} &= (p + 1)!,
\end{align*}
\]
where \(d^T = [d_1, \ldots, d_{p+2}]^T\).

Therefore, we have
\[\sum_{i=1}^{s} d_i h f(Y_i) = h^{p+2} y^{(p+2)}(x_{n-1}) + \frac{\partial f(y_{n-1})}{\partial y} \sum_{i=1}^{s} d_i.
\]
\[\cdot \left(h y(x_{n-1} + c_i h) + \delta_i h^{p+2} y^{(p+1)}(x_{n-1}) - \sum_{j=1}^{p+2} a_{ij} h^2 y'(x_{n-1} + c_j h)\right) + O(h^{p+3})
\]

197
CHAPTER 5. THE DESIGN OF AN EDESJ INTEGRATOR

\[ h^{p+2}y^{(p+2)}(x_{n-1}) + \frac{\partial f(y_{n-1})}{\partial y} = \]

\[ \sum_{i=1}^{p+2} \frac{d_i c_i^{p+1}}{(p + 1)!} h^{p+2}y^{(p+1)}(x_{n-1}) + \sum_{i=1}^{p+2} d_i h^{p+2}y^{(p+1)}(x_{n-1}) \]

\[ - \sum_{i=1}^{p+2} d_i \sum_{j=1}^{p+1} a_{ij} \sum_{k=1}^{k-1} \frac{c_j^{k-1}}{(k-1)!} h^{k+1}y^{(k)}(x_{n-1}) + O(h^{p+3}) \]

\[ = h^{p+2}y^{(p+2)}(x_{n-1}) + \left( \sum_{i=1}^{p+2} \frac{d_i c_i^{p+1}}{(p + 1)!} + \sum_{i=1}^{p+2} d_i \delta_i - \sum_{i=1}^{p+2} \sum_{j=1}^{p+1} d_i a_{ij} \frac{c_j^p}{p!} \right) \cdot h^{p+2} \frac{\partial f(y_{n-1})}{\partial y} \]

\[ = h^{p+2}y^{(p+2)}(x_{n-1}) + O(h^{p+3}) \]

From (5.8) and the order conditions, we have

\[ \begin{cases} 
\sum_{i=1}^{s} d_i \sum_{j=1}^{s} a_{ij} \sum_{k=1}^{s} c_i^{k-1} = 0, & k = 1, \ldots, p, \\
\sum_{i=1}^{s} d_i \sum_{j=1}^{s} a_{ij} c_i^p = 1, &
\end{cases} \]

(5.12)

or

\[ \begin{cases} 
d^T A c^{k-1} = 0, & k = 1, \ldots, p, \\
d^T A c^p = p!. &
\end{cases} \]

(5.13)

Hence, by assuming (5.10), we can estimate \( h^{p+2}y^{(p+2)}(x_{n-1}) \) using \( \sum_{i=1}^{s} d_i h f(Y_i) \).

Our next goal is to find \( a_{p+2,p+1} \). At first, \( d_i \) can be specified from (5.10) (or (5.11)). Then, by (5.12) (or (5.13)), we can find \( a_{p+2,p+1} \). Let \( c = [c_1, c_2, \ldots, c_{p+2}]^T \).

From (5.11) and (5.13), it follows that

\[ d^T A \Pi_{i=1}^{p} (c - c_i e) = p!. \]

We also denote

\[ \pi_{p+1} = \Pi_{i=1}^{p} (c_{p+1} - c_i), \]

\[ \pi_{p+2} = \Pi_{i=1}^{p} (c_{p+2} - c_i). \]

Because the coefficient matrix \( A \) has a singly-implicit block and a diagonal block, \( a_{p+2,p+1} \) can be found using

\[ \lambda d_{p+1} \pi_{p+1} + d_{p+2} a_{p+2,p+1} \pi_{p+1} + \lambda d_{p+2} \pi_{p+2} = p!. \]

(5.14)

198
5.1. CONSTRUCTION OF EDESI METHODS

We can also simplify (5.14) as follows. If we define

$$\hat{\pi}_i = \prod_{j \neq i, j=1}^{p+2} (c_i - c_j),$$

then from (5.10) and (5.11), we have

$$d^T \prod_{j=1}^{p+1} (c - c_j e) = (p + 1)!,$$
$$d^T \prod_{j \neq p+1, j=1}^{p+2} (c - c_j e) = (p + 1)!. $$

These are equivalent to

$$d_{p+2} \hat{\pi}_{p+2} = (p + 1)!,$$
$$d_{p+1} \hat{\pi}_{p+1} = (p + 1)!. $$

Thus, we have

$$\hat{\pi}_{p+1} = (c_{p+1} - c_{p+2}) \pi_{p+1},$$
$$\hat{\pi}_{p+2} = (c_{p+2} - c_{p+1}) \pi_{p+2}. $$

Therefore, (5.14) can be simplified as

$$d_{p+2} a_{p+2, p+1} \pi_{p+1} = p!. $$

It follows that

$$a_{p+2, p+1} = \frac{p!}{\pi_{p+1} d_{p+2}} = \frac{\hat{\pi}_{p+2}}{(p + 1) \pi_{p+1}} = \frac{(c_{p+2} - c_{p+1}) \pi_{p+2}}{(p + 1) \pi_{p+1}} = \frac{1}{p + 1} \frac{\prod_{i=1}^{p+1} (c_{p+1} - c_i)}{\prod_{i=1}^{p} (c_{p+1} - c_i)}. $$

(5.15)

Consequently, we have another advantage over the classical DESI methods when effective order is applied to the $p$-th order DESI methods with $s = p + 2$ stages. That is, we can construct an error estimator for the purpose of changing order whereas the classical DESI methods cannot achieve this conveniently unless $s \geq 3$.

Finally, the weights $b^T$ can be found using (5.1) together with the stability requirement. We give another example to demonstrate the derivation of a $p$-th order EDESI method with $s = p + 2$ stages.
CHAPTER 5. THE DESIGN OF AN EDESI INTEGRATOR

Example 5.2 We use $s = 4, p = 2$ to illustrate how to obtain this new method. Let the abscissae be chosen to be $c = [0, \frac{1}{3}, \frac{2}{3}, 1]^T$. Suppose the stability function of the method satisfies

$$R(z) = \frac{1 + \gamma_1 z + \gamma_2 z^2 + \gamma_3 z^3 + \gamma_4 z^4}{(1 - \lambda z)^4}.$$ 

According to [23], the intervals of $\lambda$ for L-stability are $[0.129945766, 3.284267796]$, and it is clear that $\gamma_4 = 0$. We choose $\lambda = \frac{3}{23}$ (an approximation to 0.129945766). Because the method is of order 2, from

$$R(z) = \exp(z) + O(z^3),$$

it is easy to see that

$$\gamma_1 = \frac{11}{23}, \quad \gamma_2 = \frac{85}{1058}.$$ 

By use of the E-polynomial, we have

$$E(y) = \left(1 + \frac{11}{23}iy - \frac{85}{1058}y^2 - \gamma_3 iy^3\right)\left(1 - \frac{11}{23}iy - \frac{85}{1058}y^2 + \gamma_3 iy^3\right) - \left(1 + \left(\frac{3}{23}\right)^2 y^2\right)^4$$

$$= \left(-\frac{5281}{1119364} + \frac{22}{23}\gamma_3\right)y^4 + \left(\frac{2916}{148035889} - \gamma_3^2\right)y^6 + \left(\frac{6561}{78310985281}\right)y^8.$$ 

If $E(y)$ is divided by $y^4$, and we let $t = y^2$, then $E(y)$ can be written as a second order polynomial $P(t)$. It is noted that the method is A-stable if and only if

$$P(t) = p_0 + p_1 t + p_2 t^2 \geq 0, \quad \forall t > 0.$$ 

Since $p_2 > 0$, if we can ensure that

(i) $p_0 > 0, p_1 > 0, \text{ or}$

(ii) $p_1^2 - 4p_0p_2 < 0,$

then the method is A-stable. In this case, $\gamma_3$ is not available for A-stability using (i). From (ii), we have

$$\frac{9(11 - \sqrt{-49 + 36\sqrt{2}})}{12167\sqrt{2}} < \gamma_3 < \frac{9(11 + \sqrt{-49 + 36\sqrt{2}})}{12167\sqrt{2}}.$$ 

200
5.1. CONSTRUCTION OF EDESI METHODS

We can choose \( \gamma_3 \) from this interval to obtain the smallest possible error constant. Note that the error constant of this method is given by

\[
C_3 = (-1)^4 L_4' \left( \frac{1}{\lambda} \right) \lambda^3 - \gamma_3.
\]

It is easy to verify that the lower bound of \( \gamma_3 \) makes the error constant smallest. For simplicity, we choose \( \gamma_3 = \frac{63}{12167} \), then \( C_3 = 6.178 \times 10^{-3} \). We use three procedures to derive the coefficients for the method.

(1) Find the perturbations \( \alpha_1, \alpha_2 \).

Since \( c_1 = 0, c_2 = \frac{1}{3} \) and \( \lambda = \frac{3}{23} \), from (4.35),

\[
(1 - c_1 z)(1 - c_2 z) = 1 + 2\beta_1 z + 2\beta_2 z^2,
\]

and we find

\[
\beta_1 = -\frac{1}{6}, \quad \beta_2 = 0.
\]

From (4.36), we have

\[
1 + \alpha_1 z + \alpha_2 z^2 = \frac{1}{1 - \lambda z} (1 + \beta_1 z + \beta_2 z^2)^{-1} + O(z^3).
\]

It follows that

\[
\alpha_1 = -\frac{13}{138}, \quad \alpha_2 = \frac{25}{19044}.
\]

(2) Find the coefficient matrix \( A \).

The first two rows of \( A \) can be specified by the order conditions.

\[
\sum_{i=1}^{2} a_{ij} \frac{c_j^{k-1}}{(k-1)!} + \alpha_k = \frac{c_k}{k!}, \quad k = 1, 2.
\]  

(5.16)

From (5.10) or (5.11), we have \( d = [-27, 81, -81, 27] \). Using (5.15) and the following two conditions,

\[
\pi_3 = (c_3 - c_1)(c_3 - c_2) = \frac{2}{9},
\]

\[
\pi_4 = (c_4 - c_1)(c_4 - c_2) = \frac{2}{3}.
\]

we have \( a_{43} = \frac{1}{3} \). The remaining \( a_{41}, a_{42} \) can be found using (5.16).
(3) Find the weight vector $b^T$.

Once we have the coefficient matrix $A$, $b^T$ can be found using

$$b^T[e, c] = [1, \frac{1}{2} + \alpha_1],$$

and the stability function

$$1 + z b^T (I_4 - zA)^{-1} e = \frac{1 + \gamma_1 z + \gamma_2 z^2 + \gamma_3 z^3}{(1 - \lambda z)^4},$$

with $\gamma_3 = \frac{63}{12167}$. Hence, the resulting Butcher tableau is

$$\begin{array}{cccc}
0 & 623 & -25 & 0 & 0 \\
\frac{1}{3} & 1681 & 1033 & 0 & 0 \\
\frac{2}{3} & 1451 & 2551 & \frac{3}{23} & 0 \\
1 & 407 & 927 & \frac{1}{3} & \frac{3}{23} \\
\end{array}$$

(5.17)

In a similar way, we can derive EDESI methods with $s = p + 3, p + 4, ...$, but for simplicity of implementation in the case of $s = p + 2$ it is enough to estimate the error for variable order, we will not go further to discuss the case of $s > p + 2$. EDESI methods inherit the stability properties from their associated DESI methods. This can be shown in the following theorem.

**Theorem 5.1** The stability functions for the $p$-th order EDESI methods with stage $p + 1$ (or $p + 2$) and their corresponding DESI methods (same $\lambda$ and $\gamma_{p+1}$ (for $p + 2$)) are identical.

**Proof:** Let $R(z), \overline{R}(z)$ be the stability functions for EDESI and DESI methods respectively. It is clear that

$$\overline{R}(z) = \exp(z) + O(z^{p+1}).$$

Let $\alpha_1, \alpha_2, \ldots, \alpha_p$ be the perturbations, and let $\phi(z) = 1 + \alpha_1 z + \alpha_2 z^2 + \ldots + \alpha_p z^p$. We apply the EDESI method to $y' = \lambda y$ and write $z = h\lambda$. The input
5.1. CONSTRUCTION OF EDESI METHODS

perturbation for an EDESI method is 
\[ y(x_0)\phi(z) + O(h^{p+1}) = y(x_0)\phi(z)\exp(z) + O(z^{p+1}) \].

Therefore, \( R(z) \) satisfies

\[ \phi(z)R(z) = \phi(z)\exp(z) + O(z^{p+1}) \]

Hence, the result follows. \( \square \)

It is also easy to see that the error constant for an EDESI method is the same as its associated DESI method. By Lemma 3.1, (3.12) and (3.13), we can derive the error constants for EDESI methods. The error constants of EDESI methods with \( s = p + 1 \) and \( s = p + 2 \) are given in Table 3.4 (cf. section 3.4). Note that the error constants shown in Table 3.4 are the optimum choices. In other words, the eigenvalue is chosen to minimize the error constant. In the case of \( s = p + 1 \), the stability function \( R(z) \) is

\[ R(z) = \frac{(-1)^s \sum_{i=0}^{p} L_{s}^{(s-i)} \left( \frac{1}{\lambda} \right) \lambda^i z^i}{(1 - \lambda z)^s} \]

with the error term

\[ \exp(z) - R(z) = (-1)^s L_{s}^{(s-p-1)} \left( \frac{1}{\lambda} \right) \lambda^{p+1} + O(z^{p+2}) \]

For \( s = p + 2 \), the degree of \( P(z) \) becomes \( p + 1 \). In addition to specifying the coefficients of \( z^i, i = 0, \ldots, p \), the coefficient of \( z^{p+1} \) in \( P(z) \) should be chosen for L-stability. It is found that

\[ R(z) = \frac{(-1)^s \sum_{i=0}^{p} L_{s}^{(s-i)} \left( \frac{1}{\lambda} \right) \lambda^i z^i + \gamma_{p+1} z^{p+1}}{(1 - \lambda z)^s} \]

with the error constant

\[ C = (-1)^s L_{s}^{(s-p-1)} \left( \frac{1}{\lambda} \right) \lambda^{p+1} - \gamma_{p+1} \]

Here \( \gamma_{p+1} \) is found to satisfy L-stability.

As with the ESIRK methods, it is necessary to use a starting procedure for EDESI methods. We will use the classical \( p \)-th order SIRK method as the starting method because the singly-implicit block of EDESI methods is an ESIRK method. Hence, all that we modify are the weights of an SIRK method in order that the initial value for an EDESI method is of the form

\[ y_0 = y(x_0) + \alpha_1 h y'(x_0) + \alpha_2 h^2 y''(x_0) + \cdots + \alpha_p h^p y''(x_0) \]
where \( a_1, a_2, \ldots, a_p \) are the corresponding perturbations for the method. For example, the weight vectors \( b^T \) of the EDESI methods mentioned in Example 5.1, 5.2 are

\[
b^T = \begin{bmatrix}
\frac{1945 + 1945\sqrt{2}}{7852} & \frac{55162 - 21395\sqrt{2}}{86372}
\end{bmatrix},
\]

\[
b^T = \begin{bmatrix}
\frac{9497 + 9497\sqrt{2}}{38088} & \frac{2503 - 9497\sqrt{2}}{38088}
\end{bmatrix},
\]

respectively.

Figure 5.1: Error vs. stepsize for order-2 singly-implicit methods with constant stepsize

Figure 5.2: Flops vs. error for order-2 singly-implicit methods with constant stepsize

**Experiment 5.1** In accordance with the study of the error constants for EDESI methods, we intend to give some numerical tests on the Kaps problem (3.9). We test order 2 SIRK(3.21), DESI\((s = p + 2)(3.25)\), ESIRK\((c_1 = \frac{7 - 4\sqrt{2}}{3})\), \( s = 3, p = 2 \) EDESI(5.2), \( s = 4, p = 2 \) EDESI(5.17) methods with constant stepsize. As shown in Figure 5.1. The SIRK, ESIRK methods have the same global error because they have the same error constant \( 4.044 \times 10^{-2} \). The EDESI \((s = p + 2)\) method and its associated DESI method also have similar global errors (error constant is \( 6.4203 \times 10^{-3} \)), but the error for the DESI method is slightly smaller than that for the EDESI method when \( h \) is not small enough. We note that the eigenvalue...
chosen ($\lambda = 0.129945766$) for the DESI method is the optimal choice for which the error constant is smallest, whereas the eigenvalue for the EDESI($s = p + 2$) method chosen as $\frac{3}{25}$ is just an approximation to the optimal choice. The global error produced by the $s = 3, p = 2$ EDESI method is between that for the SIRK and DESI ($s = p + 2$) methods because the error constant is $1.2185 \times 10^{-2}$ (which is between the corresponding error constants).

Table 5.1: Numerical results for the Kaps problem using order 2 SIRK, DESI, ESIRK and EDESI ($s = p + 1, s = p + 2$) with constant stepsize

<table>
<thead>
<tr>
<th>method</th>
<th>SIRK</th>
<th>DESI</th>
<th>ESIRK</th>
<th>$s = p + 1$</th>
<th>$s = p + 2$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>h = 0.5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>glberr</td>
<td>$4.6513 \times 10^{-6}$</td>
<td>$7.4402 \times 10^{-7}$</td>
<td>$4.6947 \times 10^{-6}$</td>
<td>$1.5396 \times 10^{-6}$</td>
<td>$8.4532 \times 10^{-7}$</td>
</tr>
<tr>
<td>averit</td>
<td>2.7500</td>
<td>2.6375</td>
<td>2.7500</td>
<td>2.6333</td>
<td>2.5125</td>
</tr>
<tr>
<td>flops</td>
<td>32549</td>
<td>39141</td>
<td>32404</td>
<td>35820</td>
<td>38474</td>
</tr>
<tr>
<td>h = 0.25</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>glberr</td>
<td>$1.1652 \times 10^{-6}$</td>
<td>$1.8422 \times 10^{-7}$</td>
<td>$1.1699 \times 10^{-6}$</td>
<td>$3.7710 \times 10^{-7}$</td>
<td>$1.9353 \times 10^{-7}$</td>
</tr>
<tr>
<td>averit</td>
<td>2.6250</td>
<td>2.5000</td>
<td>2.6250</td>
<td>2.5167</td>
<td>2.4438</td>
</tr>
<tr>
<td>flops</td>
<td>62531</td>
<td>74857</td>
<td>62294</td>
<td>68117</td>
<td>74781</td>
</tr>
<tr>
<td>h = 0.125</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>glberr</td>
<td>$2.8975 \times 10^{-7}$</td>
<td>$4.5819 \times 10^{-8}$</td>
<td>$2.9030 \times 10^{-7}$</td>
<td>$9.3005 \times 10^{-8}$</td>
<td>$4.6130 \times 10^{-8}$</td>
</tr>
<tr>
<td>averit</td>
<td>2.4875</td>
<td>2.4188</td>
<td>2.4875</td>
<td>2.4375</td>
<td>2.3813</td>
</tr>
<tr>
<td>flops</td>
<td>119481</td>
<td>145674</td>
<td>118912</td>
<td>132400</td>
<td>145671</td>
</tr>
<tr>
<td>h = 0.0625</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>glberr</td>
<td>$7.2134 \times 10^{-8}$</td>
<td>$1.1424 \times 10^{-8}$</td>
<td>$7.2201 \times 10^{-8}$</td>
<td>$2.3075 \times 10^{-8}$</td>
<td>$1.1249 \times 10^{-8}$</td>
</tr>
<tr>
<td>averit</td>
<td>2.4125</td>
<td>2.3500</td>
<td>2.4125</td>
<td>2.3688</td>
<td>2.3141</td>
</tr>
<tr>
<td>flops</td>
<td>232900</td>
<td>284183</td>
<td>231647</td>
<td>258159</td>
<td>283630</td>
</tr>
</tbody>
</table>

It is also interesting to examine whether the introduction of effective order affects the behaviour of convergence. In this experiment, we update the Jacobian once per step and the starting value used is the output value of the previous step. The tolerance for stopping the Newton iteration is chosen as $10^{-7}$. The average number of iterations for different stepsizes for all methods are shown in Table 5.1. For the SIRK and ESIRK methods, the average iteration numbers are the same. For the DESI and EDESI($s = p + 2$) methods, the equal spacing of the EDESI
method shows its slight advantage in convergence behaviour. We also plot the
total flops against the maximum global error in Figure 5.2. It is clear that the
DESI and EDESI \((s = p + 2)\) methods seem to be more efficient than the other
methods in this experiment.

5.2 Variable step-size for EDESI methods

Because the singly-implicit block of an EDESI method is a ESIRK method, this
suggest using the stepsize changing scheme introduced in Lemma 4.21 for EDESI
methods. Let \(N\) denote the dimension of the system. For an \(s\)-stage ESIRK
method, if the input value is \(B_{s,h}(\alpha, y(x_{n-1}))\) and the stepsize ratio is \(r\) (suppose
the stepsize taken is from \(h\) to \(rh\)), then from (4.46) the output value is given
approximately by

\[
B_{s, rh}(\alpha, y(x_{n-1} + h)) \approx B_{s,h}(\alpha, y(x_{n-1})) + h \left( (\tau^T - \alpha^T + \alpha^T RE)W^{-1} \otimes I_N \right) F.
\]

We note that this output approximation depends not only on the stepsize chang-
ing ratio \(r\), but on the perturbations \(\alpha^T\). Furthermore, \(\alpha^T\) is determined by
the choice of abscissae. Therefore, the behaviours of local error and stability for
EDESI methods with variable stepsize are also closely related to the choice of the
abscissae. In this section, we will invoke this same strategy to achieve variable
stepsize for EDESI methods.

The difference between an EDESI and an ESIRK method are those additional di-
agonal stages of the EDESI methods. Hence, EDESI is not a collocation method.
If stage order is equal to stage number, then the modified Vandermonde matrix
\(W\) defined as in section 4.2 can be written as a square matrix. Hence, for ESIRK
methods of order \(s\), the Nordsieck vector \(\zeta^{[n-1]}\) can be written as

\[
\zeta^{[n-1]} = (W^{-1} \otimes I_N) hF + O(h^{s+1}).
\]

For \(s\)-stage EDESI methods of effective order \(p\), we can certainly use the modified
Vandermonde matrix of the first \(p\) stages which is invertible to express \(\zeta^{[n-1]}\).
However, if we intend to use all stage derivatives to achieve a changing stepsize,
5.2. VARIABLE STEP-SIZE FOR EDESI METHODS

then there are some free parameters left. From (4.46), we write the stepsize changing formulae as

\[ B_{p,rh}(\alpha, y(x_{n-1} + h)) \approx B_{p,h}(\alpha, y(x_{n-1})) + ((\tau^T - \alpha^T + \alpha^T R E) \otimes I_N) \zeta^{[n-1]}. \]

Here \( \tau^T, \alpha^T \) are \( 1 \times p \) vectors and \( R, E \) are \( p \times p \) matrices. In order to use the derivatives \( hF = [hf(Y_1), hf(Y_2), \ldots, hf(Y_s)] \), we need to express \( \zeta^{[n-1]} \) in terms of \( hF \). In other words, we have to find a matrix, say \( D = [d_{ij}] \), satisfying

\[ (D \otimes I_N)hF = \zeta^{[n-1]} + O(h^{p+1}). \]  (5.18)

Obviously, \( D \) has dimension \( p \times s \). If we assume \( N = 1 \), then (5.18) can be written as

\[ h \sum_{j=1}^{s} d_{ij}f(Y_j) = h^iy^{(i)}(x_{n-1}) + O(h^{p+1}), \quad i = 1, 2, \ldots, p. \]

Using the Taylor expansions in the above equations, we have

\[ \sum_{j=1}^{s} d_{ij} \frac{c_{ij}^{k-1}}{(k-1)!} = \delta_{ik}, \quad i = 1, 2, \ldots, p; \quad k = 1, 2, \ldots, p, \]  (5.19)

where \( \delta_{ik} \) is the Kronecker delta. It is clear that (5.19) gives \( p^2 \) equations and we have \( sp \) parameters \( d_{ij} \) to specify. From Theorem 4.10 and Lemma 4.21, we note that if the matrix \( R \) is an identity matrix, then the weights are equivalent to the weights in \( b^T \) for constant stepsize. But, for EDESI methods, the weights come from not only the modified order conditions, but also the stability requirements (for \( s = p + 1 \), we have 1 constraint, for \( s = p + 2 \) we have 2 constraints). Consequently, in addition to satisfying (5.19), we also need to ensure that

\[ (\tau^T - \alpha^T + \alpha^T E)D = b^T, \]  (5.20)

where \( b^T \) is the weight vector of the method with constant stepsize. In summary, the total number of free parameters that need to be specified is

\[ sp - p^2 - (s - p) = (s - p)(p - 1). \]

We give an example to demonstrate how to achieve variable stepsize for EDESI methods.
Example 5.3 We take the $s = 4, p = 2$ method mentioned in Example 5.2. In this case, $\alpha_1 = -\frac{13}{138}, \alpha_2 = \frac{25}{19044}, c = [0, \frac{1}{3}, \frac{2}{3}, 1]^T$ and the weight vector $b^T$ is given by
\[
b^T = \begin{bmatrix} 126379 & 236527 & 145503 & 51273 \ 559682 & 559682 & 559682 & 559682 \ \end{bmatrix}^T.
\]
Hence, we have $(4 - 2)(2 - 1) = 2$ free parameters in the matrix $D = [d_{ij}]_{2 \times 4}$. (5.19) gives
\[
d_{11} + d_{12} + d_{13} + d_{14} = 1,

d_{21} + d_{22} + d_{23} + d_{24} = 0,

d_{11}c_1 + d_{12}c_2 + d_{13}c_3 + d_{14}c_4 = 0,

d_{21}c_1 + d_{22}c_2 + d_{23}c_3 + d_{24}c_4 = 1.
\]
If we solve the above equations for $d_{11}, d_{21}, d_{12}, d_{22}$, then we find
\[
d_{11} = 1 + d_{13} + 2d_{14},

d_{21} = d_{23} + 3(-1 + d_{24}) - d_{24},

d_{12} = -2d_{13} - 3d_{14},

d_{22} = -2d_{23} - 3(-1 + d_{24}).
\]
We also need to satisfy (5.20), and it follows that
\[
(\tau^T - \alpha^T + \alpha^T E)D = \begin{bmatrix} 1 & 28 \\ 69 \end{bmatrix} = b^T.
\]
Solving the above equations, we find
\[
d_{13} = \frac{436509 - 681352d_{23}}{1679046}, \quad d_{14} = \frac{153819 - 681352d_{24}}{1679046}.
\]
If we choose $d_{23} = 0, d_{24} = 0$, then we have
\[
D = \begin{bmatrix} 807731 & -444825 & 145503 & 51273 \\ 559682 & 559682 & 559682 & 559682 \\ -3 & 3 & 0 & 0 \ \end{bmatrix}.
\]
The weights for variable stepsize can be derived from
\[
b^T(\tau) = (\tau^T - \alpha^T + \alpha^T RE)D.
\]

208
Therefore, we have the stepsize changing formulae for this method as follows.

\[ B_{p,rh}(\alpha, y(x_{n-1} + h)) = B_{p,h}(\alpha, y(x_{n-1})) + h(b_1(r) f(Y_1) + b_2(r) f(Y_2) + b_3(r) f(Y_3) + b_4(r) f(Y_4)) + O(h^{p+1}), \]

where

\[
\begin{align*}
  b_1(r) &= \frac{6417382 + 11327095r - 304175r^2}{77236116}, \\
  b_2(r) &= \frac{48381424 - 16044873r + 304175r^2}{77236116}, \\
  b_3(r) &= \frac{48501(151 - 13r)}{25745372}, \\
  b_4(r) &= \frac{17091(151 - 13r)}{25745372}.
\end{align*}
\]

For EDESI methods with variable stepsize, we may now want to ask these questions: Can the local error also be reduced? Is the \( r \)-interval still reasonable for the stability? Using the same scheme mentioned in Section 4.6, we can derive the normalized error constant for these methods. For example, \( s = 3, p = 2 \), if the eigenvalue is chosen as \( \frac{2}{11} \), then the error constant of the method is \( 1.2898 \times 10^{-2} \).

At first, the error constant for the two systematic steps \((h, rh)\) is given by

\[
\widehat{R}_2(z, r) \exp(-(1 + r)h) = 1 + \widehat{C}_2(r) z^3 + O(z^4).
\]

where \( \widehat{R}_2 \) is the two-step stability function mentioned in Section 4.6. The normalized error constant is

\[
\widehat{C}(r) = \frac{\widehat{C}_2(r)}{1 + r^3} = \frac{671 + 2625r + 2625r^2 + 671r^3}{255552(1 + r^3)},
\]

which has a local maximum at \( r = 1 \). For \( s = 4, p = 2 \), if we use the method and the weights \( b_1(r), \ldots, b_4(r) \) mentioned in Example 5.2, then the normalized error constant is

\[
\widehat{C}(r) = \frac{-1120135 + 1493563r + 1493563r^2 - 1120135r^3}{60445656(1 + r^3)},
\]

which also has a local maximum at \( r = 1 \) and \( \widehat{C}(r) < 6.17791 \times 10^{-3} \) (error constant for constant stepsize), when \( r \in (0.39, 2.61) \). Hence, the local errors
are reduced for both methods using the stepsize changing scheme. For stability, under the stepsize changing pattern assumption \((h, rh, h, rh, \ldots)\), the value \(r\) which makes a method A-stable is shown to be \([0.8379, 1.19346]\) for \(s = 3\) and 1 for \(s = 4\). Therefore, like ESIRK methods, EDESI methods seem to benefit from variable stepsize as well.

Figure 5.3: Work/Precision diagram (flops/maximum error) for EDESI methods solving the Kaps problem, \(s = p + 1: - -, s = p + 2: - -\)

**Experiment 5.2** In attempting to examine which of the cases \(s = p + 1\) or \(s = p + 2\) is a better option, we compare \(s = p + 1\) and \(s = p + 2\) EDESI methods for \(p = 1, 2, 3, 4\). In Figure 5.3, we show the work/precision diagrams for the different order-\(p\) methods for solving the Kaps problem (3.9). Although
the case \( s = p + 2 \) has one more stage than the case \( s = p + 1 \), except \( p = 3 \) and \( p = 1 \) in low accuracy, the case \( s = p + 2 \) seems to be more efficient than \( s = p + 1 \). The different error constants of the methods seem able to offer an explanation. Refering to Table 3.4, the third order method with 4 stages has a smaller error constant than the method with 5 stages. This is also the case for the first order method. For variable order, methods with \( s = p + 2 \) are good candidates for implementation. Therefore, in the following context of this thesis, the word “EDESI” will indicate the case \( s = p + 2 \) only.

### 5.3 Implementation

Because EDESI methods are inherited from DESI and ESIRK methods, implementing these methods become more complicated. Extra diagonal stages cause more iterations, and the factor of effective order increases the difficulty in changing stepsize. But EDESI methods also have some advantages over ESIRK methods because of their additional stages. For example, it is easier to obtain an asymptotic error estimator and to construct an interpolation formula within each step.

Unlike SIRK or ESIRK methods, EDESI methods have enough stages to form the estimator. Suppose \( Y_1, Y_2, \ldots, Y_{p+2} \) are those stage values of the previous accepted step. From the argument in section 4.7, it is easy to see that we only need \( p + 1 \) values of \( Y_i \) so as to form a local error estimator. Therefore, for EDESI methods, we have many options in constructing the estimator. For example, we can choose the first \( p + 1 \) values of \( Y_i \) to estimate the error. By using the modified Vandermonde matrix \( W \), and letting \( hF = [hf(Y_1), hf(Y_2), \ldots, hf(Y_{p+1})] \), we have

\[
h^{p+1}y^{(p+1)}(x_n) = (e_{p+1}^T W \otimes I_N)hF + O(h^{p+2}),
\]

where the \( i \)-th component of the \( j \)-th row of \( W \) is given by \( \frac{j-1}{(i-1)!} \), for \( 1 \leq i \leq p+1 \), \( 1 \leq j \leq p+1 \). It is also natural to use the average of some different estimates to \( h^{p+1}y^{(p+1)}(x_n) \) to estimate the error. In [26], the average of using the first \( p + 1 \) and the last \( p + 1 \) values of \( Y_i \) is suggested.
In addition to estimating the error, a big difference between EDESI and ESIRK methods is caused by using the modified Newton method. EDESI has singly-implicit stages and diagonally-implicit stages. Let \( B_{p,h}(\alpha, y(x_{n-1})) \) be the Butcher solution of a \( p \)-th order method, then the internal stages can be written as

\[
Y_i = B_{p,h}(\alpha, y(x_{n-1})) + h \sum_{j=1}^{p} a_{ij} f(Y_j), \quad i = 1, 2, \ldots, p, \tag{5.21}
\]

\[
Y_i = B_{p,h}(\alpha, y(x_{n-1})) + h \sum_{j=1}^{i-1} a_{ij} f(Y_j) + h \lambda f(Y_j), \quad i = p + 1, p + 2. \tag{5.22}
\]

The integration of (5.21) is the same as for the ESIRK method. After the convergence of the Newton iteration for (5.21), we proceed by the Newton method to solve (5.22). The computational cost for the latter is much cheaper than for the former. For solving the nonlinear algebraic system (5.21), the scheme (3.8) mentioned in section 3.1 is proposed. The other implementation criteria, such as when to stop the iterations, calculation of the derivatives, how to change stepsize and tolerance setting, are similar to that for an ESIRK method.

Because of the small error constant, the stepsize taken by EDESI methods are much larger than for usual methods, and a good predictor is desirable and indeed necessary. There are many ways to find the starting value for the modified Newton iterations. For example, we can construct an interpolation polynomial based on the \( B(p) \) conditions or, alternatively, use Lagrange's interpolation formula to form the interpolation polynomial. Furthermore, the numerical experiments for DESI code have shown that from the information on \( h f(Y_i) \) of the previous step, using the \( C(p) \) conditions to derive a quadrature formula for predicting the starting values for singly-implicit stages, and making use of the recently evaluated singly-implicit stages to form the quadrature formulae for predicting the starting values for diagonally-implicit stages is a better predictor [34].

In the singly-implicit part, if we choose \( p \) of the \( s \) previous stages \( Y_i \) to form the predictor, this will be equivalent to using (4.74) mentioned in section 4.8. The
5.3. IMPLEMENTATION

The predictor is given by

\[ Y_t^{[n]} = y(x_n) + (T_t W^{-1} \otimes I_N) h F^{[n-1]} + O(h^{p+1}). \]

\( T_t W^{-1} \) in (4.74) is a \( r \)-polynomial of degree \( p \) because \( p \) quadrature points are used. Since EDESI methods have \( p + 2 \) stages, we can use all or some of the abscissae to form a different \( r \)-polynomial from that using the first \( p \) abscissae. Numerical testing shows that generally the better choice is to choose the last \( p \) points. For example, in the case of \( p = 3 \) (\( s = 5 \)), we can select \( \frac{3}{4}, \frac{3}{4}, 1 \) amongst the abscissae \( (0, \frac{1}{4}, \frac{2}{4}, \frac{3}{4}, 1) \) to construct the interpolation formula. In the part of the diagonal stages, there are two stage values we need to predict, \( Y_{p+1} \) and \( Y_{p+2} \). For \( Y_{p+1} \), the derivatives of the singly-implicit stages of the present step can be used. For the last stage \( Y_{p+2} \), the values \( F(Y_1), F(Y_2), \ldots, F(Y_{p+1}) \) are available for prediction. We now give an example to show how to construct the predictor for EDESI methods.

**Example 5.4** We take the second order EDESI method (5.17) with \( c_1 = 0, c_2 = \frac{1}{2}, c_3 = \frac{1}{3}, c_4 = 1 \). From \( T_i = [c_i r, (\frac{c_i}{2}) r^2], i = 1, 2, \) we have

\[ T_1 = [0, 0], \quad T_2 = [\frac{r}{3}, \frac{r^2}{18}]. \]

If we choose \( h f(Y_3^{[n-1]}), h f(Y_4^{[n-1]}) \) to form the predictor, then from

\[ W = \begin{bmatrix} 1 & c_3 - 1 \\ 1 & c_4 - 1 \end{bmatrix} \rightarrow W^{-1} = \begin{bmatrix} 0 & 1 \\ -\frac{3}{2} & \frac{3}{2} \end{bmatrix}, \]

the predictor for the singly-implicit part is given by

\[ Y_1^{[n]} \approx y(x_n), \]

\[ Y_2^{[n]} \approx y(x_n) - \frac{r^2}{12} h f(Y_1^{[n-1]}) + \left( \frac{r}{3} + \frac{r^2}{12} \right) h f(Y_2^{[n-1]}). \]

For the diagonal-implicit part, using the Taylor expansion, we have

\[ Y_3 \approx y(x_n) + \left[ c_3, \frac{c_3^2}{2} \right] \begin{bmatrix} 1 & c_1 \\ 1 & c_2 \end{bmatrix}^{-1} \begin{pmatrix} h f(Y_1) \\ h f(Y_2) \end{pmatrix}. \]
Experiment 5.3 To compare the predictor using the previous solution \( y_n \) as the starting value with predictor (4.74), we use the second order and third order EDESI methods to solve the Kaps, the Robertson and Van der Pol problems and give the number of steps taken, number of function evaluations and the end point global error (maximum error for Kaps). From Table 5.2, we conclude that using the predictor (4.74) improves all the numerical performances except the case for solving the Van der Pol problem when \( rtol = atol = 10^{-2} \). But, in this case, these two predictors still give similar results. When high accuracy is required, the positive influence of using the predictor (4.74) becomes greater.

5.4 Numerical results

In an attempt to analyze the outcomes due to the application of effective order, in this section, we compare the numerical results between the members of singly-implicit methods including SIRK, DESI, ESIRK and EDESI methods. In general, the efficiency of a L-stable method for solving stiff problems is determined by the following two main factors:

(1) global error (or accuracy);

(2) number of operations.

It is noted that the error constants for SIRK and ESIRK methods are identical while DESI and EDESI methods also have the same error constants (much smaller
### 5.4. NUMERICAL RESULTS

Table 5.2: Results when using predictor (4.74) (A) and previous solution value (B) with order 2, 3 EDESI

<table>
<thead>
<tr>
<th>Tolerance</th>
<th>nstep(A)</th>
<th>nstep(B)</th>
<th>nfcn(A)</th>
<th>nfcn(B)</th>
<th>glberr(A)</th>
<th>glberr(B)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Order 2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Rtol=Atol=Tol</td>
<td>Kaps problem</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$10^{-2}$</td>
<td>10</td>
<td>12</td>
<td>62</td>
<td>76</td>
<td>$4.93 \times 10^{-4}$</td>
<td>$4.01 \times 10^{-4}$</td>
</tr>
<tr>
<td>$10^{-4}$</td>
<td>25</td>
<td>60</td>
<td>143</td>
<td>517</td>
<td>$3.68 \times 10^{-5}$</td>
<td>$1.11 \times 10^{-4}$</td>
</tr>
<tr>
<td>Rtol=10^{-4}Atol</td>
<td>Robertson kinetics</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$10^{-2}$</td>
<td>56</td>
<td>57</td>
<td>389</td>
<td>411</td>
<td>$3.35 \times 10^{-8}$</td>
<td>$3.39 \times 10^{-8}$</td>
</tr>
<tr>
<td>$10^{-4}$</td>
<td>167</td>
<td>167</td>
<td>915</td>
<td>1312</td>
<td>$9.92 \times 10^{-8}$</td>
<td>$3.81 \times 10^{-9}$</td>
</tr>
<tr>
<td>Rtol=Atol=Tol</td>
<td>Van der Pol</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$10^{-2}$</td>
<td>132</td>
<td>127</td>
<td>1744</td>
<td>1585</td>
<td>$4.35 \times 10^{-3}$</td>
<td>$4.62 \times 10^{-3}$</td>
</tr>
<tr>
<td>$10^{-4}$</td>
<td>427</td>
<td>427</td>
<td>2721</td>
<td>3607</td>
<td>$1.92 \times 10^{-4}$</td>
<td>$7.45 \times 10^{-5}$</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Tolerance</th>
<th>nstep(A)</th>
<th>nstep(B)</th>
<th>nfcn(A)</th>
<th>nfcn(B)</th>
<th>glberr(A)</th>
<th>glberr(B)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Order 3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Rtol=Atol=Tol</td>
<td>Kaps problem</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$10^{-6}$</td>
<td>22</td>
<td>27</td>
<td>208</td>
<td>363</td>
<td>$1.66 \times 10^{-5}$</td>
<td>$1.66 \times 10^{-5}$</td>
</tr>
<tr>
<td>$10^{-8}$</td>
<td>58</td>
<td>58</td>
<td>507</td>
<td>834</td>
<td>$2.23 \times 10^{-7}$</td>
<td>$2.23 \times 10^{-7}$</td>
</tr>
<tr>
<td>Rtol=10^{-4}Atol</td>
<td>Robertson kinetics</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$10^{-6}$</td>
<td>158</td>
<td>158</td>
<td>1477</td>
<td>1958</td>
<td>$4.44 \times 10^{-11}$</td>
<td>$1.30 \times 10^{-11}$</td>
</tr>
<tr>
<td>$10^{-8}$</td>
<td>449</td>
<td>451</td>
<td>3172</td>
<td>5492</td>
<td>$3.78 \times 10^{-12}$</td>
<td>$1.40 \times 10^{-12}$</td>
</tr>
<tr>
<td>Rtol=Atol=Tol</td>
<td>Van der Pol</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$10^{-6}$</td>
<td>369</td>
<td>370</td>
<td>3904</td>
<td>5654</td>
<td>$7.96 \times 10^{-7}$</td>
<td>$1.71 \times 10^{-6}$</td>
</tr>
<tr>
<td>$10^{-8}$</td>
<td>1063</td>
<td>1061</td>
<td>8190</td>
<td>14815</td>
<td>$6.30 \times 10^{-8}$</td>
<td>$3.93 \times 10^{-9}$</td>
</tr>
</tbody>
</table>
CHAPTER 5. THE DESIGN OF AN EDESI INTEGRATOR

than SIRK). In the count of the total operations, in addition to considering the cost for LU factorization and back substitutions, we also need to take the transformation costs into account. Approximately, for an \( s \) stage, \( p \)-th order method solving a system of dimension \( N \), the total operation number can be summarized as follows.

<table>
<thead>
<tr>
<th>method</th>
<th>operation count (( p ): order, ( s ): stage no., ( N ): system dimension)</th>
</tr>
</thead>
<tbody>
<tr>
<td>SDIRK</td>
<td>( \frac{N^3}{3} )\text{(factorization)}+sN^2\text{(back substitution)}</td>
</tr>
<tr>
<td>SIRK, ESIRK</td>
<td>( \frac{N^3}{3} + pN^2 + 3Np^2 )\text{(transformation)}</td>
</tr>
<tr>
<td>DESI, EDESI</td>
<td>( \frac{N^3}{3} + pN^2 + 3Np^2 + (s - p)N^2 )\text{(diagonal part)}</td>
</tr>
</tbody>
</table>

The underlining in the above table indicates the extra computation costs shown in parentheses. We note that the cost of back substitution for SDIRK is the square of system dimension \( N^2 \) multiplied by the stage number \( s \). In [1], R. Alexander shows that when \( s \leq 3 \), we can have L-stable SDIRK method with order \( p = s \), but it is not possible to attain order 4 in 4 stages. In fact, we need at least 5 stages so as to obtain 4-th order L-stable SDIRK methods (see [45]). For DESI and EDESI, the number of additional added diagonal stages is \( s - p \). In the following discussions, we will use the case \( s - p = 2 \) for DESI and EDESI. As shown in section 3.4, DESI (EDESI) methods can compensate for their larger number of operations by the choice of a much smaller error constant. If we suppose there is only one iteration required per step and use the length of the stepsize taken per flop at each step as the measurement of efficiency for the order-\( p \) methods, then the magnitude of efficiency will be proportional to

\[
E = \frac{1}{N_{\text{opn}}C^\frac{1}{p+1}},
\]

where \( N_{\text{opn}} \) is the number of operations and \( C \) is the error constant of the related method. Therefore, the values \( N = 2, 5, 50 \) give the efficiency table 5.3.

In table 5.3, in the case of the SDIRK methods, we use \( s = 7 \) and \( s = 9 \) for \( p = 5 \) and \( p = 6 \) respectively. It can be seen that when we are dealing with
Table 5.3: Efficiency measurement of some singly-implicit methods

<table>
<thead>
<tr>
<th>N = 2</th>
<th>p = 1</th>
<th>p = 2</th>
<th>p = 3</th>
<th>p = 4</th>
<th>p = 5</th>
<th>p = 6</th>
</tr>
</thead>
<tbody>
<tr>
<td>SDIRK</td>
<td>6.00e-1</td>
<td>1.15e+1</td>
<td>8.89e+0</td>
<td>2.60e+2</td>
<td>2.78e+2</td>
<td>1.04e+4</td>
</tr>
<tr>
<td>SIRK, ESIRK</td>
<td>3.15e-1</td>
<td>3.54e+0</td>
<td>1.90e+0</td>
<td>4.23e+1</td>
<td>4.93e+1</td>
<td>1.65e+3</td>
</tr>
<tr>
<td>DESI, EDESI</td>
<td>3.04e+3</td>
<td>4.55e+1</td>
<td>5.09e+2</td>
<td>4.39e+2</td>
<td>2.82e+3</td>
<td>1.78e+5</td>
</tr>
</tbody>
</table>

| N = 5 |
|-------|-------|-------|-------|-------|-------|-------|
| SDIRK | 6.00e-2 | 1.34e+0 | 1.11e+0 | 3.42e+1 | 3.93e+1 | 1.50e+3 |
| SIRK, ESIRK | 4.89e-2 | 8.10e-1 | 5.18e-1 | 1.27e+1 | 1.57e+1 | 5.49e+2 |
| DESI, EDESI | 4.77e+2 | 9.63e+0 | 1.29e+2 | 1.24e+2 | 8.64e+2 | 5.73e+4 |

| N = 50 |
|-------|-------|-------|-------|-------|-------|-------|
| SDIRK | 9.05e-5 | 2.63e-3 | 2.65e-3 | 9.40e-2 | 1.44e-1 | 6.26e+0 |
| SIRK, ESIRK | 9.02e-5 | 2.60e-3 | 2.58e-3 | 8.98e-2 | 1.47e-1 | 6.47e+0 |
| DESI, EDESI | 1.27e+0 | 3.71e-2 | 7.04e-1 | 9.13e-1 | 8.12e+0 | 6.68e+2 |

small dimension systems, SDIRK and DESI(EDESI) methods are more efficient than SIRK(ESIRK) methods, and DESI(EDESI) methods are most efficient. For large systems, SDIRK and SIRK(ESIRK) methods have similar efficiency, while DESI(EDESI) methods still has best performance. We also need to note that the SDIRK method has stage order equal to one only, and some of the abscissae for higher order SIRK and DESI methods ($p \geq 3$, for DESI, unless $s = 6$ for $p = 3$, $s = 7$ for $p = 4$, $s = 11$ for $p = 5$) will be greater than 1.

**Experiment 5.4** In our experiments on test problems, we choose (1) the Prothero-Robinson problem (2.5) with $\lambda = 1000$, the smooth function $g(x) = \sin(x)$ and integration interval $[0, \pi]$, (2) the Curtis problem (5.23), (3) the Kaps problem (3.9), (4) the Oregonator (4.77), (5) the Robertson kinetic problem (2.6) with integration interval $[0, 10^{10}]$ and (6) the Van der Pol oscillator (2.7) with $\theta = 10^6$. 

217
and integration interval \([0, 2]\). For the Robertson problem and the Oregonator problem, \(atol = 10^{-4}rtol\). For the rest of the problems, \(atol = rtol\).

The Curtis problem

\[
\begin{align*}
  y' &= f = Ay + du - Au, \quad y = [y_1, y_2]^T, \quad y_1(0) = 1 \\
  u &= [\cos(x), \sin(x)]^T, \quad du = [-\sin(x), \cos(x)]^T, \quad y_2(0) = 0 \\
  A(1, 1) &= -1 - \lambda \cos(\theta x)^2, \quad \theta = \frac{1}{3} \\
  A(2, 2) &= -1 - \lambda \sin(\theta x)^2, \quad \lambda = 1000 \\
  A(1, 2) &= A(2, 1) = \lambda \cos(\theta x) \sin(\theta x), \quad x \in [0, 10\pi]
\end{align*}
\]

For methods, we have chosen order 2,3,4 L-stable SIRK, DESI, ESIRK and EDESI methods. We use the stage number \(s = p + 3\) for implementing the order-\(p\) DESI methods and \(s = p + 2\) for the order-\(p\) EDESI methods. Therefore, for the same order DESI has the smallest error constant, followed by EDESI, then SIRK and ESIRK methods (cf. Table 3.4). Besides, for effective order methods, the abscissae used are equally spaced in \([0, 1]\).

In an attempt to make reasonable comparisons, in addition to having a good predictor and an error estimator for every method, we use the same scheme (3.8) mentioned in section 3.1 for the Newton iterations. We have also carefully chosen some crucial parameters for each method in order to have better numerical results. These parameters, such as the safety factor for steps size prediction, convergent rate for updating the Jacobian matrix, the relation parameter for stopping the iterations and the stepsize bound for constant stepsize, all affect the numerical performance directly. For example, for large tolerances, the stepsize taken is larger, the Newton iterations will converge slower, and we may lower the convergence rate for updating the Jacobian matrix so as to reduce the number of iterations. The parameters chosen usually are very dependent on the test problems, such as for a mildly stiff problem. The stepsize bound for constant stepsize is advisedly chosen to be smaller than the one for a strongly stiff problem.

For each problem and for each fixed-order method, some numerical details are given in table (cf. Table 5.6,...,Table 5.23) in order that we can analyze the
5.4. NUMERICAL RESULTS

influence caused by effective order. The details include: the accepted steps, the rejected steps (caused by both error and convergence), the function evaluations, the numbers of Jacobian evaluations, the LU factorizations, the total flops, the maximum global error (end point for Robertson, Oregonator and Van der Pol) and the average number of iterations per step.

In the case of order-2 methods, the ESIRK method does not benefit from effective order; its performance is very similar to the SIRK method but it has a slightly greater average number of iterations and is less efficient than the SIRK method. When using variable stepsize, the normalized error constant for the ESIRK method with abscissae in [0, 1] is larger than the classical SIRK method (cf. section 4.6.1). For the DESI and EDESI methods, except with the Prothero-Robinson problem, the latter seems to have advantages over the former because of effective order. This situation is clearer for strictly stiff problems. Roughly speaking, the EDESI method is more efficient than the others (cf. Figure 5.4).

For order-3 and order-4 methods, the positive influences of effective order are apparent. ESIRK methods are overall more efficient than SIRK methods. Some of the abscissae are greater than 1 for SIRK methods and the ESIRK methods have the advantage over the SIRK methods when using variable stepsize (cf. section 4.6.1) are good explanations for the results. We note that for order 3, 4 SIRK methods, we have to lower the parameter of convergence rate for updating the Jacobian from \(10^{-2}\) to \(10^{-4}\) (\(10^{-2}\) is for order 2) in order to obtain better performance. For DESI and EDESI methods, for the linear Prothero-Robinson problem and the Kaps problem with mildly stiff parameter, DESI methods are slightly more efficient than EDESI methods. We are not surprised at these results because DESI methods have smaller error constants than EDESI methods and the abscissae for order 3, 4 DESI methods still remain inside [0, 1]. The variable stepsize does not give EDESI methods enough benefit to beat DESI methods for these two problems. But it is not the case for the rest of the problems (for DESI methods, we fail to have the end-point error for the oregonator problem). The stiffness make the influences of effective order more "effective". EDESI methods almost perform better than DESI methods when low accuracy is required. We also note that because of the much smaller error constants for the DESI and EDESI
CHAPTER 5. THE DESIGN OF AN EDESI INTEGRATOR

Figure 5.4: Work/precision diagrams of order-2 singly-implicit methods, SIRK: +, DESI: *, ESIRK: o, EDESI: ×
methods, when the tolerance is not small enough, the larger stepsize taken makes it difficult for the Newton iterations to converge. Therefore, the parameter for stopping the iterations and for updating the Jacobian should be chosen strictly, especially for the DESI method. Generally, when demanding more accuracy, the DESI and EDESI methods are more efficient than SIRK and ESIRK methods. ESIRK methods converge faster and perform well for all problems and for a wider range of tolerance. DESI methods can be very robust for linear stiff problems and perform much better in high precision. (cf. Figure 5.5 and Figure 5.6).

Generally, it can be concluded that

(1) The second order SIRK method does not have abscissae greater than 1. Using "additional diagonal stages" or "effective order" does not improves the numerical performance.

(2) When order $\geq 3$, the adoption of effective order successfully improve the numerical results by moving all the abscissae of SIRK methods inside the integration interval.

(3) One way to increase the efficiency for SIRK (ESIRK) is to use the $A(\alpha)$ stable methods. An alternative without destroying the A-stability is to add some additional diagonal stages for these methods. That is to use the DESI or EDESI type methods.

(4) For DESI or EDESI type methods, because of the much smaller error constants, they are more efficient in high precision. Methods with order $\geq 3$ are regarded as the "high order" methods.

(5) Because the stepsizes taken by DESI or EDESI methods are much larger than the corresponding SIRK (ESIRK) methods, a good starting value and a severe stopping criterion for Newton iterations are necessary. Furthermore, some implementation strategies need more conservative parameters.

For completeness, we summarize some characteristics for these singly-implicit methods in Table 5.4 and Table 5.5.
Figure 5.5: Work/precision diagrams of order-3 singly-implicit methods, SIRK: +, DESI: *, ESIRK: ◦, EDESI: ×
5.4. NUMERICAL RESULTS

Figure 5.6: Work/precision diagrams of order-4 singly-implicit methods, SIRK: +, DESI: *, ESIRK: o, EDESI: x
Table 5.4: Summary of SIRK and DESI

<table>
<thead>
<tr>
<th>Method</th>
<th>Characteristic</th>
</tr>
</thead>
</table>
| **SIRK** | - The coefficient matrix has one point spectrum property.  
- Good stability and accuracy for stiff problems, A-stable (L-stable) methods exist from order-1 to 8 except order-7.  
- Collocation methods and stage order is equal to the order of the methods, satisfying $B(s), C(s)$ and stiff accuracy.  
- The abscissae $c_1, c_2, \ldots, c_p$ of an $s$-stage method are obtained from $c_i = \lambda \xi_i$, where $\xi_i$ is the zero of the $s$-th degree Laguerre polynomial $L_s(x)$.  
- Some abscissae are larger than 1 when order $\geq 3$, hence it has difficulty in solving some nonlinear problems. |
| **DESI** | - The coefficient matrix has singly-implicit and diagonal-implicit parts, satisfying one point spectrum property.  
- The abscissae of the singly-implicit part are obtained from $c_i = \lambda \xi_i$.  
- The abscissae of the diagonal-implicit part can be chosen for convenience.  
- Some abscissae are greater than 1 for higher order methods (say $\geq 5$ if $s = p + 3$).  
- Additional diagonal stages lower the overall computation cost, and make it easier to construct an error estimator for variable order.  
- Stage order is equal to the quadrature order.  
- L-stable methods exist for any order methods.  
- Error constants can be very small (depending on how many additional stages are added), therefore they are very efficient for linear problems.  
- For variable order purpose, $s = p + 3$ is suggested.  
- Stepsizes taken are very large, for some nonlinear problems, Newton iterations have difficulty converging.  
- Ideal for high precision. |
Table 5.5: Summary of ESIRK and EDESI

<table>
<thead>
<tr>
<th>Method</th>
<th>Characteristic</th>
</tr>
</thead>
</table>
| **ESIRK** | ◊ Same stability as their corresponding SIRK method.  
◊ Starting methods are necessary except for order-1 method.  
◊ Abscissae can be chosen for convenience as long as they are distinct,  
and there is no difficulty in solving nonlinear problems.  
◊ Collocation methods and stage order are equal to the order of the methods,  
satisfying $B(s), C(s)$ and stiff accuracy.  
◊ Using variable stepsize aids convergence.  
◊ Order-2 method has the optimal choice $\left[\frac{7-4\sqrt{2}}{3}, 1\right]$.  
◊ More computational cost because the order-$s$ method has $s$-stage  
singly-implicit part. |
| **EDESI** | ◊ Same structure of the coefficient matrix and stability as the corresponding  
DESI method.  
◊ Starting methods are necessary except for order-1 method.  
◊ Abscissae can be chosen for convenience only if they are distinct,  
they have no difficulty in solving nonlinear problems.  
◊ Additional diagonal stages lower the overall computation cost,  
and make it easier to construct an error estimator for variable order,  
$s = p + 2$ is suggested for the purpose of variable order.  
◊ Stage order is equal to the quadrature order.  
◊ L-stable methods exist for any order methods.  
◊ Error constants are small, stepsize taken is larger than for ESIRK method,  
more efficient for high precision requirement.  
◊ Using variable stepsize aids convergence, hence Newton iterations converge  
more easily than for DESI method, especially for nonlinear problems and strictly  
stiff problems.  
◊ Ideal for high precision. |
Table 5.6: Numerical results for Curtis (5.23) by testing order-2 singly-implicit methods

<table>
<thead>
<tr>
<th>tol</th>
<th>nsteps</th>
<th>rej(err/newt)</th>
<th>nfcn</th>
<th>njac</th>
<th>lu</th>
<th>flops</th>
<th>maxerr</th>
<th>niter</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SIRK</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10^{-3}</td>
<td>474</td>
<td>95/206</td>
<td>2832</td>
<td>474</td>
<td>775</td>
<td>505827</td>
<td>1.88e-03</td>
<td>2.49</td>
</tr>
<tr>
<td>10^{-4}</td>
<td>316</td>
<td>35/24</td>
<td>1862</td>
<td>316</td>
<td>375</td>
<td>316712</td>
<td>5.53e-04</td>
<td>2.65</td>
</tr>
<tr>
<td>10^{-5}</td>
<td>541</td>
<td>1/0</td>
<td>2828</td>
<td>541</td>
<td>542</td>
<td>481920</td>
<td>7.92e-05</td>
<td>2.61</td>
</tr>
<tr>
<td>10^{-6}</td>
<td>1138</td>
<td>0/0</td>
<td>4386</td>
<td>1138</td>
<td>1138</td>
<td>844924</td>
<td>2.05e-05</td>
<td>1.93</td>
</tr>
<tr>
<td>10^{-7}</td>
<td>2456</td>
<td>0/0</td>
<td>7500</td>
<td>2456</td>
<td>2456</td>
<td>1611981</td>
<td>4.51e-06</td>
<td>1.53</td>
</tr>
<tr>
<td>DESI</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10^{-3}</td>
<td>196</td>
<td>0/205</td>
<td>4668</td>
<td>196</td>
<td>401</td>
<td>844730</td>
<td>1.85e-04</td>
<td>4.76</td>
</tr>
<tr>
<td>10^{-4}</td>
<td>165</td>
<td>0/50</td>
<td>2875</td>
<td>165</td>
<td>215</td>
<td>581391</td>
<td>1.47e-04</td>
<td>3.48</td>
</tr>
<tr>
<td>10^{-5}</td>
<td>242</td>
<td>0/0</td>
<td>3157</td>
<td>242</td>
<td>242</td>
<td>649377</td>
<td>4.63e-05</td>
<td>2.61</td>
</tr>
<tr>
<td>10^{-6}</td>
<td>513</td>
<td>0/0</td>
<td>4650</td>
<td>513</td>
<td>513</td>
<td>1068513</td>
<td>9.33e-06</td>
<td>1.81</td>
</tr>
<tr>
<td>10^{-7}</td>
<td>1105</td>
<td>0/0</td>
<td>8893</td>
<td>1105</td>
<td>1105</td>
<td>2077418</td>
<td>2.01e-06</td>
<td>1.61</td>
</tr>
<tr>
<td>ESIRK</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10^{-3}</td>
<td>509</td>
<td>112/226</td>
<td>3110</td>
<td>508</td>
<td>846</td>
<td>599724</td>
<td>1.08e-03</td>
<td>2.50</td>
</tr>
<tr>
<td>10^{-4}</td>
<td>362</td>
<td>48/42</td>
<td>2166</td>
<td>361</td>
<td>451</td>
<td>393780</td>
<td>7.05e-04</td>
<td>2.64</td>
</tr>
<tr>
<td>10^{-5}</td>
<td>552</td>
<td>0/0</td>
<td>3040</td>
<td>551</td>
<td>551</td>
<td>535852</td>
<td>7.80e-05</td>
<td>2.75</td>
</tr>
<tr>
<td>10^{-6}</td>
<td>1179</td>
<td>0/0</td>
<td>4608</td>
<td>1178</td>
<td>1178</td>
<td>941861</td>
<td>1.88e-05</td>
<td>1.95</td>
</tr>
<tr>
<td>10^{-7}</td>
<td>2543</td>
<td>0/0</td>
<td>8384</td>
<td>2542</td>
<td>2542</td>
<td>1863432</td>
<td>4.02e-06</td>
<td>1.65</td>
</tr>
<tr>
<td>EDESI</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10^{-3}</td>
<td>202</td>
<td>0/96</td>
<td>2097</td>
<td>201</td>
<td>297</td>
<td>471415</td>
<td>1.19e-02</td>
<td>2.59</td>
</tr>
<tr>
<td>10^{-4}</td>
<td>179</td>
<td>0/16</td>
<td>1743</td>
<td>178</td>
<td>194</td>
<td>366640</td>
<td>7.78e-04</td>
<td>2.43</td>
</tr>
<tr>
<td>10^{-5}</td>
<td>306</td>
<td>0/0</td>
<td>2212</td>
<td>305</td>
<td>305</td>
<td>515489</td>
<td>7.28e-05</td>
<td>1.81</td>
</tr>
<tr>
<td>10^{-6}</td>
<td>642</td>
<td>0/0</td>
<td>3662</td>
<td>641</td>
<td>641</td>
<td>949683</td>
<td>9.25e-06</td>
<td>1.43</td>
</tr>
<tr>
<td>10^{-7}</td>
<td>1350</td>
<td>0/0</td>
<td>6754</td>
<td>1349</td>
<td>1349</td>
<td>1893279</td>
<td>2.09e-06</td>
<td>1.25</td>
</tr>
</tbody>
</table>
Table 5.7: Numerical results for Prothero-Robinson (2.5) by testing order-2 singly-implicit methods

<table>
<thead>
<tr>
<th>tol</th>
<th>nsteps</th>
<th>rej(err/newt)</th>
<th>nfcn</th>
<th>njac</th>
<th>lu</th>
<th>flops</th>
<th>maxerr</th>
<th>niter</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SIRK</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$10^{-2}$</td>
<td>14</td>
<td>2/0</td>
<td>34</td>
<td>1</td>
<td>14</td>
<td>4988</td>
<td>$6.68e-05$</td>
<td>1.06</td>
</tr>
<tr>
<td>$10^{-3}$</td>
<td>24</td>
<td>2/0</td>
<td>54</td>
<td>1</td>
<td>18</td>
<td>7968</td>
<td>$2.17e-05$</td>
<td>1.04</td>
</tr>
<tr>
<td>$10^{-4}$</td>
<td>47</td>
<td>3/0</td>
<td>102</td>
<td>1</td>
<td>32</td>
<td>15139</td>
<td>$3.51e-06$</td>
<td>1.02</td>
</tr>
<tr>
<td>$10^{-5}$</td>
<td>95</td>
<td>2/0</td>
<td>198</td>
<td>1</td>
<td>50</td>
<td>29248</td>
<td>$4.74e-07$</td>
<td>1.02</td>
</tr>
<tr>
<td>$10^{-6}$</td>
<td>196</td>
<td>2/0</td>
<td>404</td>
<td>1</td>
<td>101</td>
<td>59606</td>
<td>$1.09e-07$</td>
<td>1.02</td>
</tr>
<tr>
<td>DESI</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$10^{-2}$</td>
<td>8</td>
<td>0/0</td>
<td>50</td>
<td>1</td>
<td>7</td>
<td>6314</td>
<td>$1.08e-04$</td>
<td>1.25</td>
</tr>
<tr>
<td>$10^{-3}$</td>
<td>14</td>
<td>2/0</td>
<td>98</td>
<td>1</td>
<td>11</td>
<td>12485</td>
<td>$8.73e-06$</td>
<td>1.23</td>
</tr>
<tr>
<td>$10^{-4}$</td>
<td>24</td>
<td>2/0</td>
<td>158</td>
<td>1</td>
<td>16</td>
<td>20210</td>
<td>$1.93e-06$</td>
<td>1.22</td>
</tr>
<tr>
<td>$10^{-5}$</td>
<td>44</td>
<td>2/0</td>
<td>278</td>
<td>1</td>
<td>20</td>
<td>35630</td>
<td>$4.22e-07$</td>
<td>1.21</td>
</tr>
<tr>
<td>$10^{-6}$</td>
<td>89</td>
<td>2/0</td>
<td>548</td>
<td>1</td>
<td>24</td>
<td>70300</td>
<td>$7.13e-08$</td>
<td>1.20</td>
</tr>
<tr>
<td>ESIRK</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$10^{-2}$</td>
<td>14</td>
<td>2/0</td>
<td>36</td>
<td>1</td>
<td>11</td>
<td>5329</td>
<td>$2.83e-04$</td>
<td>1.06</td>
</tr>
<tr>
<td>$10^{-3}$</td>
<td>24</td>
<td>2/0</td>
<td>56</td>
<td>1</td>
<td>18</td>
<td>8450</td>
<td>$4.31e-05$</td>
<td>1.04</td>
</tr>
<tr>
<td>$10^{-4}$</td>
<td>47</td>
<td>3/0</td>
<td>104</td>
<td>1</td>
<td>23</td>
<td>15783</td>
<td>$8.63e-06$</td>
<td>1.02</td>
</tr>
<tr>
<td>$10^{-5}$</td>
<td>94</td>
<td>3/0</td>
<td>200</td>
<td>1</td>
<td>32</td>
<td>30235</td>
<td>$1.47e-06$</td>
<td>1.02</td>
</tr>
<tr>
<td>$10^{-6}$</td>
<td>195</td>
<td>3/0</td>
<td>406</td>
<td>1</td>
<td>42</td>
<td>61156</td>
<td>$2.46e-07$</td>
<td>1.02</td>
</tr>
<tr>
<td>EDESI</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$10^{-2}$</td>
<td>10</td>
<td>2/0</td>
<td>61</td>
<td>1</td>
<td>9</td>
<td>7518</td>
<td>$2.04e-04$</td>
<td>1.23</td>
</tr>
<tr>
<td>$10^{-3}$</td>
<td>17</td>
<td>2/0</td>
<td>96</td>
<td>1</td>
<td>13</td>
<td>11750</td>
<td>$5.44e-05$</td>
<td>1.24</td>
</tr>
<tr>
<td>$10^{-4}$</td>
<td>30</td>
<td>2/0</td>
<td>161</td>
<td>1</td>
<td>17</td>
<td>19219</td>
<td>$7.85e-06$</td>
<td>1.24</td>
</tr>
<tr>
<td>$10^{-5}$</td>
<td>57</td>
<td>2/0</td>
<td>296</td>
<td>1</td>
<td>20</td>
<td>34442</td>
<td>$1.67e-06$</td>
<td>1.25</td>
</tr>
<tr>
<td>$10^{-6}$</td>
<td>111</td>
<td>2/0</td>
<td>566</td>
<td>1</td>
<td>28</td>
<td>65147</td>
<td>$1.93e-07$</td>
<td>1.25</td>
</tr>
</tbody>
</table>
CHAPTER 5. THE DESIGN OF AN EDES/INTEGRATOR

Table 5.8: Numerical results for Kaps (3.9) by testing order-2 singly-implicit methods

<table>
<thead>
<tr>
<th>tol</th>
<th>nsteps</th>
<th>rej(err/newt)</th>
<th>nfcn</th>
<th>njac</th>
<th>lu</th>
<th>flops</th>
<th>maxerr</th>
<th>niter</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>SIRK</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$10^{-2}$</td>
<td>14</td>
<td>0/0</td>
<td>48</td>
<td>11</td>
<td>14</td>
<td>8237</td>
<td>1.84e-03</td>
<td>1.71</td>
</tr>
<tr>
<td>$10^{-3}$</td>
<td>22</td>
<td>0/0</td>
<td>78</td>
<td>18</td>
<td>22</td>
<td>13080</td>
<td>6.05e-04</td>
<td>1.77</td>
</tr>
<tr>
<td>$10^{-4}$</td>
<td>40</td>
<td>0/0</td>
<td>138</td>
<td>33</td>
<td>39</td>
<td>23378</td>
<td>1.55e-04</td>
<td>1.73</td>
</tr>
<tr>
<td>$10^{-5}$</td>
<td>78</td>
<td>0/0</td>
<td>228</td>
<td>53</td>
<td>60</td>
<td>42031</td>
<td>3.60e-05</td>
<td>1.46</td>
</tr>
<tr>
<td>$10^{-6}$</td>
<td>160</td>
<td>0/0</td>
<td>392</td>
<td>63</td>
<td>80</td>
<td>79478</td>
<td>8.23e-06</td>
<td>1.23</td>
</tr>
<tr>
<td><strong>DESI</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$10^{-2}$</td>
<td>8</td>
<td>0/0</td>
<td>62</td>
<td>8</td>
<td>8</td>
<td>12836</td>
<td>5.63e-04</td>
<td>1.55</td>
</tr>
<tr>
<td>$10^{-3}$</td>
<td>12</td>
<td>0/0</td>
<td>94</td>
<td>12</td>
<td>12</td>
<td>19807</td>
<td>2.01e-04</td>
<td>1.57</td>
</tr>
<tr>
<td>$10^{-4}$</td>
<td>20</td>
<td>0/0</td>
<td>150</td>
<td>20</td>
<td>20</td>
<td>32285</td>
<td>6.43e-05</td>
<td>1.50</td>
</tr>
<tr>
<td>$10^{-5}$</td>
<td>37</td>
<td>0/0</td>
<td>260</td>
<td>37</td>
<td>37</td>
<td>57679</td>
<td>1.57e-05</td>
<td>1.41</td>
</tr>
<tr>
<td>$10^{-6}$</td>
<td>74</td>
<td>0/0</td>
<td>480</td>
<td>74</td>
<td>74</td>
<td>111712</td>
<td>3.62e-06</td>
<td>1.30</td>
</tr>
<tr>
<td><strong>ESIRK</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$10^{-2}$</td>
<td>15</td>
<td>0/0</td>
<td>52</td>
<td>13</td>
<td>14</td>
<td>9944</td>
<td>2.03e-03</td>
<td>1.67</td>
</tr>
<tr>
<td>$10^{-3}$</td>
<td>23</td>
<td>0/0</td>
<td>82</td>
<td>20</td>
<td>22</td>
<td>15115</td>
<td>6.18e-04</td>
<td>1.74</td>
</tr>
<tr>
<td>$10^{-4}$</td>
<td>41</td>
<td>0/0</td>
<td>146</td>
<td>32</td>
<td>36</td>
<td>26407</td>
<td>1.60e-04</td>
<td>1.76</td>
</tr>
<tr>
<td>$10^{-5}$</td>
<td>78</td>
<td>0/0</td>
<td>240</td>
<td>38</td>
<td>48</td>
<td>46397</td>
<td>3.79e-05</td>
<td>1.52</td>
</tr>
<tr>
<td>$10^{-6}$</td>
<td>161</td>
<td>0/0</td>
<td>420</td>
<td>49</td>
<td>62</td>
<td>88544</td>
<td>8.34e-06</td>
<td>1.30</td>
</tr>
<tr>
<td><strong>EDES/INTEGRATOR</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$10^{-2}$</td>
<td>10</td>
<td>0/0</td>
<td>62</td>
<td>9</td>
<td>9</td>
<td>13792</td>
<td>6.77e-04</td>
<td>1.50</td>
</tr>
<tr>
<td>$10^{-3}$</td>
<td>15</td>
<td>0/0</td>
<td>92</td>
<td>14</td>
<td>14</td>
<td>20422</td>
<td>1.23e-04</td>
<td>1.50</td>
</tr>
<tr>
<td>$10^{-4}$</td>
<td>25</td>
<td>0/0</td>
<td>143</td>
<td>24</td>
<td>24</td>
<td>32892</td>
<td>3.63e-05</td>
<td>1.41</td>
</tr>
<tr>
<td>$10^{-5}$</td>
<td>45</td>
<td>0/0</td>
<td>231</td>
<td>44</td>
<td>44</td>
<td>56223</td>
<td>1.18e-05</td>
<td>1.27</td>
</tr>
<tr>
<td>$10^{-6}$</td>
<td>89</td>
<td>0/0</td>
<td>450</td>
<td>87</td>
<td>88</td>
<td>109617</td>
<td>3.47e-06</td>
<td>1.26</td>
</tr>
</tbody>
</table>
Table 5.9: Numerical results for Oregonator (4.77) by testing order-2 singly-implicit methods

<table>
<thead>
<tr>
<th>tol</th>
<th>nsteps</th>
<th>rej(err/newt)</th>
<th>nfcn</th>
<th>njac</th>
<th>lu</th>
<th>flops</th>
<th>maxerr</th>
<th>niter</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SIRK</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$10^{-3}$</td>
<td>164</td>
<td>10/2</td>
<td>718</td>
<td>156</td>
<td>174</td>
<td>161125</td>
<td>$2.80e + 00$</td>
<td>2.06</td>
</tr>
<tr>
<td>$10^{-4}$</td>
<td>330</td>
<td>2/0</td>
<td>994</td>
<td>305</td>
<td>320</td>
<td>265114</td>
<td>$4.45e + 00$</td>
<td>1.50</td>
</tr>
<tr>
<td>$10^{-5}$</td>
<td>705</td>
<td>1/0</td>
<td>1840</td>
<td>668</td>
<td>681</td>
<td>533478</td>
<td>$1.78e + 00$</td>
<td>1.30</td>
</tr>
<tr>
<td>$10^{-6}$</td>
<td>1508</td>
<td>1/0</td>
<td>3594</td>
<td>1358</td>
<td>1374</td>
<td>1098930</td>
<td>$3.75e - 01$</td>
<td>1.19</td>
</tr>
<tr>
<td>$10^{-7}$</td>
<td>3239</td>
<td>1/0</td>
<td>7252</td>
<td>2613</td>
<td>2642</td>
<td>2294010</td>
<td>$7.82e - 02$</td>
<td>1.12</td>
</tr>
<tr>
<td>DESI</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$10^{-3}$</td>
<td>96</td>
<td>17/17</td>
<td>1595</td>
<td>89</td>
<td>130</td>
<td>462032</td>
<td>–</td>
<td>2.82</td>
</tr>
<tr>
<td>$10^{-4}$</td>
<td>165</td>
<td>9/5</td>
<td>1619</td>
<td>156</td>
<td>177</td>
<td>531971</td>
<td>–</td>
<td>1.86</td>
</tr>
<tr>
<td>$10^{-5}$</td>
<td>325</td>
<td>3/0</td>
<td>2387</td>
<td>308</td>
<td>320</td>
<td>858648</td>
<td>–</td>
<td>1.46</td>
</tr>
<tr>
<td>$10^{-6}$</td>
<td>685</td>
<td>2/0</td>
<td>4590</td>
<td>646</td>
<td>662</td>
<td>1714815</td>
<td>–</td>
<td>1.34</td>
</tr>
<tr>
<td>$10^{-7}$</td>
<td>1463</td>
<td>2/0</td>
<td>9304</td>
<td>1286</td>
<td>1304</td>
<td>3574980</td>
<td>–</td>
<td>1.27</td>
</tr>
<tr>
<td>ESIRK</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$10^{-3}$</td>
<td>187</td>
<td>8/1</td>
<td>862</td>
<td>174</td>
<td>190</td>
<td>207055</td>
<td>$3.41e + 00$</td>
<td>2.21</td>
</tr>
<tr>
<td>$10^{-4}$</td>
<td>387</td>
<td>4/0</td>
<td>1234</td>
<td>348</td>
<td>372</td>
<td>358395</td>
<td>$2.01e + 00$</td>
<td>1.58</td>
</tr>
<tr>
<td>$10^{-5}$</td>
<td>826</td>
<td>1/0</td>
<td>2262</td>
<td>657</td>
<td>761</td>
<td>716117</td>
<td>$1.34e + 00$</td>
<td>1.37</td>
</tr>
<tr>
<td>$10^{-6}$</td>
<td>1778</td>
<td>1/0</td>
<td>4402</td>
<td>1210</td>
<td>1583</td>
<td>1482963</td>
<td>$3.11e - 01$</td>
<td>1.24</td>
</tr>
<tr>
<td>$10^{-7}$</td>
<td>3830</td>
<td>1/0</td>
<td>8716</td>
<td>2211</td>
<td>3299</td>
<td>3098499</td>
<td>$6.74e - 02$</td>
<td>1.14</td>
</tr>
<tr>
<td>EDESI</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$10^{-3}$</td>
<td>114</td>
<td>11/3</td>
<td>901</td>
<td>103</td>
<td>127</td>
<td>312234</td>
<td>$8.95e + 00$</td>
<td>1.80</td>
</tr>
<tr>
<td>$10^{-4}$</td>
<td>218</td>
<td>7/0</td>
<td>1268</td>
<td>201</td>
<td>219</td>
<td>491233</td>
<td>$3.36e + 00$</td>
<td>1.41</td>
</tr>
<tr>
<td>$10^{-5}$</td>
<td>447</td>
<td>2/0</td>
<td>2378</td>
<td>415</td>
<td>431</td>
<td>955254</td>
<td>$8.10e - 01$</td>
<td>1.32</td>
</tr>
<tr>
<td>$10^{-6}$</td>
<td>956</td>
<td>3/0</td>
<td>4859</td>
<td>726</td>
<td>867</td>
<td>2002125</td>
<td>$1.67e - 01$</td>
<td>1.27</td>
</tr>
<tr>
<td>$10^{-7}$</td>
<td>2051</td>
<td>1/0</td>
<td>10339</td>
<td>1056</td>
<td>1725</td>
<td>4260441</td>
<td>$3.66e - 02$</td>
<td>1.26</td>
</tr>
</tbody>
</table>
Table 5.10: Numerical results for Robertson (2.6) by testing order-2 singly-implicit methods

<table>
<thead>
<tr>
<th>tol</th>
<th>nsteps</th>
<th>rej(err/newt)</th>
<th>nfcn</th>
<th>njac</th>
<th>lu</th>
<th>flops</th>
<th>maxerr</th>
<th>niter</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>SIRK</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$10^{-2}$</td>
<td>89</td>
<td>1/0</td>
<td>334</td>
<td>88</td>
<td>90</td>
<td>81495</td>
<td>$1.61e-08$</td>
<td>1.86</td>
</tr>
<tr>
<td>$10^{-3}$</td>
<td>157</td>
<td>1/0</td>
<td>490</td>
<td>150</td>
<td>157</td>
<td>131592</td>
<td>$7.27e-10$</td>
<td>1.55</td>
</tr>
<tr>
<td>$10^{-4}$</td>
<td>303</td>
<td>1/1</td>
<td>776</td>
<td>123</td>
<td>190</td>
<td>227293</td>
<td>$2.05e-09$</td>
<td>1.28</td>
</tr>
<tr>
<td>$10^{-5}$</td>
<td>609</td>
<td>1/0</td>
<td>1392</td>
<td>167</td>
<td>268</td>
<td>431386</td>
<td>$4.96e-10$</td>
<td>1.14</td>
</tr>
<tr>
<td>$10^{-6}$</td>
<td>1271</td>
<td>1/0</td>
<td>2786</td>
<td>246</td>
<td>364</td>
<td>877595</td>
<td>$4.90e-11$</td>
<td>1.10</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>DESI</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$10^{-2}$</td>
<td>55</td>
<td>0/6</td>
<td>507</td>
<td>53</td>
<td>61</td>
<td>165284</td>
<td>$5.71e-09$</td>
<td>1.84</td>
</tr>
<tr>
<td>$10^{-3}$</td>
<td>80</td>
<td>0/0</td>
<td>645</td>
<td>78</td>
<td>80</td>
<td>215971</td>
<td>$4.93e-10$</td>
<td>1.61</td>
</tr>
<tr>
<td>$10^{-4}$</td>
<td>144</td>
<td>0/0</td>
<td>1042</td>
<td>142</td>
<td>144</td>
<td>371859</td>
<td>$3.88e-10$</td>
<td>1.45</td>
</tr>
<tr>
<td>$10^{-5}$</td>
<td>283</td>
<td>0/0</td>
<td>1868</td>
<td>281</td>
<td>283</td>
<td>709697</td>
<td>$2.74e-10$</td>
<td>1.32</td>
</tr>
<tr>
<td>$10^{-6}$</td>
<td>582</td>
<td>1/0</td>
<td>3702</td>
<td>579</td>
<td>583</td>
<td>1445344</td>
<td>$9.33e-11$</td>
<td>1.27</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>ESIRK</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$10^{-2}$</td>
<td>94</td>
<td>1/0</td>
<td>362</td>
<td>93</td>
<td>94</td>
<td>96764</td>
<td>$4.41e-09$</td>
<td>1.90</td>
</tr>
<tr>
<td>$10^{-3}$</td>
<td>160</td>
<td>0/0</td>
<td>534</td>
<td>144</td>
<td>157</td>
<td>152769</td>
<td>$7.33e-09$</td>
<td>1.67</td>
</tr>
<tr>
<td>$10^{-4}$</td>
<td>305</td>
<td>1/0</td>
<td>794</td>
<td>147</td>
<td>200</td>
<td>258562</td>
<td>$1.26e-09$</td>
<td>1.30</td>
</tr>
<tr>
<td>$10^{-5}$</td>
<td>612</td>
<td>1/0</td>
<td>1420</td>
<td>150</td>
<td>254</td>
<td>488807</td>
<td>$3.75e-10$</td>
<td>1.16</td>
</tr>
<tr>
<td>$10^{-6}$</td>
<td>1275</td>
<td>1/0</td>
<td>2824</td>
<td>224</td>
<td>334</td>
<td>991917</td>
<td>$3.61e-11$</td>
<td>1.11</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>EDES1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$10^{-2}$</td>
<td>57</td>
<td>0/0</td>
<td>344</td>
<td>54</td>
<td>56</td>
<td>129628</td>
<td>$3.53e-09$</td>
<td>1.50</td>
</tr>
<tr>
<td>$10^{-3}$</td>
<td>90</td>
<td>0/0</td>
<td>459</td>
<td>87</td>
<td>89</td>
<td>193874</td>
<td>$2.06e-09$</td>
<td>1.27</td>
</tr>
<tr>
<td>$10^{-4}$</td>
<td>167</td>
<td>1/0</td>
<td>843</td>
<td>164</td>
<td>167</td>
<td>356158</td>
<td>$5.16e-11$</td>
<td>1.25</td>
</tr>
<tr>
<td>$10^{-5}$</td>
<td>331</td>
<td>1/0</td>
<td>1659</td>
<td>326</td>
<td>331</td>
<td>695079</td>
<td>$6.12e-11$</td>
<td>1.25</td>
</tr>
<tr>
<td>$10^{-6}$</td>
<td>686</td>
<td>1/0</td>
<td>3436</td>
<td>669</td>
<td>682</td>
<td>1427801</td>
<td>$8.29e-11$</td>
<td>1.25</td>
</tr>
</tbody>
</table>
### 5.4. NUMERICAL RESULTS

Table 5.11: Numerical results for Van der Pol (2.7) by testing order-2 singly-implicit methods

<table>
<thead>
<tr>
<th>tol</th>
<th>nsteps</th>
<th>rej(err/newt)</th>
<th>nfcn</th>
<th>njac</th>
<th>lu</th>
<th>flops</th>
<th>maxerr</th>
<th>niter</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SIRK</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$10^{-2}$</td>
<td>180</td>
<td>17/14</td>
<td>1046</td>
<td>112</td>
<td>172</td>
<td>141340</td>
<td>2.72e-02</td>
<td>2.65</td>
</tr>
<tr>
<td>$10^{-3}$</td>
<td>358</td>
<td>29/0</td>
<td>1366</td>
<td>203</td>
<td>276</td>
<td>222772</td>
<td>5.19e-03</td>
<td>1.76</td>
</tr>
<tr>
<td>$10^{-4}$</td>
<td>736</td>
<td>17/0</td>
<td>2128</td>
<td>263</td>
<td>376</td>
<td>390155</td>
<td>8.84e-04</td>
<td>1.41</td>
</tr>
<tr>
<td>$10^{-5}$</td>
<td>1574</td>
<td>6/0</td>
<td>4040</td>
<td>413</td>
<td>596</td>
<td>782601</td>
<td>2.11e-04</td>
<td>1.28</td>
</tr>
<tr>
<td>$10^{-6}$</td>
<td>3419</td>
<td>7/0</td>
<td>8378</td>
<td>723</td>
<td>953</td>
<td>1661846</td>
<td>3.64e-05</td>
<td>1.22</td>
</tr>
<tr>
<td>DESI</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$10^{-2}$</td>
<td>112</td>
<td>18/39</td>
<td>2322</td>
<td>89</td>
<td>161</td>
<td>378403</td>
<td>8.76e-03</td>
<td>3.57</td>
</tr>
<tr>
<td>$10^{-3}$</td>
<td>195</td>
<td>33/7</td>
<td>2375</td>
<td>148</td>
<td>211</td>
<td>455215</td>
<td>1.17e-03</td>
<td>2.08</td>
</tr>
<tr>
<td>$10^{-4}$</td>
<td>358</td>
<td>19/0</td>
<td>2954</td>
<td>264</td>
<td>313</td>
<td>619634</td>
<td>2.33e-04</td>
<td>1.57</td>
</tr>
<tr>
<td>$10^{-5}$</td>
<td>738</td>
<td>11/0</td>
<td>5137</td>
<td>522</td>
<td>576</td>
<td>1153302</td>
<td>9.31e-05</td>
<td>1.37</td>
</tr>
<tr>
<td>$10^{-6}$</td>
<td>1562</td>
<td>5/0</td>
<td>10364</td>
<td>1052</td>
<td>1112</td>
<td>2364481</td>
<td>1.87e-05</td>
<td>1.32</td>
</tr>
<tr>
<td>ESIRK</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$10^{-2}$</td>
<td>197</td>
<td>10/16</td>
<td>1052</td>
<td>113</td>
<td>182</td>
<td>155412</td>
<td>9.81e-03</td>
<td>2.54</td>
</tr>
<tr>
<td>$10^{-3}$</td>
<td>363</td>
<td>21/0</td>
<td>1416</td>
<td>204</td>
<td>280</td>
<td>242886</td>
<td>6.88e-03</td>
<td>1.84</td>
</tr>
<tr>
<td>$10^{-4}$</td>
<td>741</td>
<td>11/0</td>
<td>2256</td>
<td>265</td>
<td>392</td>
<td>431992</td>
<td>1.16e-03</td>
<td>1.50</td>
</tr>
<tr>
<td>$10^{-5}$</td>
<td>1577</td>
<td>6/0</td>
<td>4110</td>
<td>407</td>
<td>604</td>
<td>853812</td>
<td>1.92e-04</td>
<td>1.30</td>
</tr>
<tr>
<td>$10^{-6}$</td>
<td>3404</td>
<td>5/0</td>
<td>8446</td>
<td>713</td>
<td>961</td>
<td>1798506</td>
<td>3.72e-05</td>
<td>1.24</td>
</tr>
<tr>
<td>EDESI</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$10^{-2}$</td>
<td>135</td>
<td>30/14</td>
<td>1442</td>
<td>102</td>
<td>164</td>
<td>279154</td>
<td>1.20e-03</td>
<td>2.18</td>
</tr>
<tr>
<td>$10^{-3}$</td>
<td>234</td>
<td>31/3</td>
<td>1647</td>
<td>168</td>
<td>231</td>
<td>368440</td>
<td>4.09e-04</td>
<td>1.55</td>
</tr>
<tr>
<td>$10^{-4}$</td>
<td>442</td>
<td>15/0</td>
<td>2499</td>
<td>308</td>
<td>358</td>
<td>591431</td>
<td>2.04e-04</td>
<td>1.37</td>
</tr>
<tr>
<td>$10^{-5}$</td>
<td>913</td>
<td>9/0</td>
<td>4885</td>
<td>610</td>
<td>762</td>
<td>1183891</td>
<td>5.50e-05</td>
<td>1.32</td>
</tr>
<tr>
<td>$10^{-6}$</td>
<td>1941</td>
<td>5/0</td>
<td>10135</td>
<td>1128</td>
<td>1560</td>
<td>2479793</td>
<td>1.27e-05</td>
<td>1.30</td>
</tr>
</tbody>
</table>
Table 5.12: Numerical results for Prothero-Robinson by testing order-3 methods

<table>
<thead>
<tr>
<th>tol</th>
<th>nsteps</th>
<th>rej(err/newt)</th>
<th>nfcn</th>
<th>njac</th>
<th>lu</th>
<th>flops</th>
<th>maxerr</th>
<th>niter</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10^{-3}</td>
<td>18</td>
<td>1/0</td>
<td>60</td>
<td>1</td>
<td>11</td>
<td>9760</td>
<td>7.63e-06</td>
<td>1.05</td>
</tr>
<tr>
<td>10^{-4}</td>
<td>28</td>
<td>1/0</td>
<td>90</td>
<td>1</td>
<td>13</td>
<td>14740</td>
<td>1.15e-06</td>
<td>1.03</td>
</tr>
<tr>
<td>10^{-5}</td>
<td>45</td>
<td>2/0</td>
<td>144</td>
<td>1</td>
<td>12</td>
<td>23678</td>
<td>1.69e-07</td>
<td>1.02</td>
</tr>
<tr>
<td>10^{-6}</td>
<td>77</td>
<td>1/0</td>
<td>240</td>
<td>1</td>
<td>14</td>
<td>39275</td>
<td>3.26e-08</td>
<td>1.03</td>
</tr>
<tr>
<td>10^{-7}</td>
<td>128</td>
<td>1/0</td>
<td>396</td>
<td>1</td>
<td>14</td>
<td>64802</td>
<td>6.20e-09</td>
<td>1.02</td>
</tr>
<tr>
<td>10^{-8}</td>
<td>224</td>
<td>1/0</td>
<td>693</td>
<td>1</td>
<td>19</td>
<td>113072</td>
<td>1.05e-09</td>
<td>1.03</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10^{-3}</td>
<td>9</td>
<td>0/0</td>
<td>66</td>
<td>1</td>
<td>7</td>
<td>9924</td>
<td>1.16e-05</td>
<td>1.22</td>
</tr>
<tr>
<td>10^{-4}</td>
<td>14</td>
<td>0/0</td>
<td>101</td>
<td>1</td>
<td>7</td>
<td>15319</td>
<td>1.95e-06</td>
<td>1.20</td>
</tr>
<tr>
<td>10^{-5}</td>
<td>23</td>
<td>1/0</td>
<td>171</td>
<td>1</td>
<td>10</td>
<td>26120</td>
<td>2.87e-07</td>
<td>1.19</td>
</tr>
<tr>
<td>10^{-6}</td>
<td>39</td>
<td>2/0</td>
<td>290</td>
<td>1</td>
<td>12</td>
<td>44469</td>
<td>4.22e-08</td>
<td>1.18</td>
</tr>
<tr>
<td>10^{-7}</td>
<td>63</td>
<td>1/0</td>
<td>451</td>
<td>1</td>
<td>13</td>
<td>69295</td>
<td>1.47e-08</td>
<td>1.17</td>
</tr>
<tr>
<td>10^{-8}</td>
<td>114</td>
<td>2/0</td>
<td>815</td>
<td>1</td>
<td>17</td>
<td>125418</td>
<td>8.53e-09</td>
<td>1.17</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10^{-3}</td>
<td>19</td>
<td>1/0</td>
<td>66</td>
<td>1</td>
<td>13</td>
<td>10851</td>
<td>1.81e-06</td>
<td>1.03</td>
</tr>
<tr>
<td>10^{-4}</td>
<td>28</td>
<td>1/0</td>
<td>93</td>
<td>1</td>
<td>11</td>
<td>15310</td>
<td>1.91e-07</td>
<td>1.02</td>
</tr>
<tr>
<td>10^{-5}</td>
<td>46</td>
<td>1/0</td>
<td>147</td>
<td>1</td>
<td>13</td>
<td>24378</td>
<td>4.70e-08</td>
<td>1.01</td>
</tr>
<tr>
<td>10^{-6}</td>
<td>79</td>
<td>1/0</td>
<td>249</td>
<td>1</td>
<td>18</td>
<td>38604</td>
<td>5.33e-09</td>
<td>1.02</td>
</tr>
<tr>
<td>10^{-7}</td>
<td>135</td>
<td>1/0</td>
<td>420</td>
<td>1</td>
<td>20</td>
<td>66408</td>
<td>5.70e-10</td>
<td>1.02</td>
</tr>
<tr>
<td>10^{-8}</td>
<td>229</td>
<td>1/0</td>
<td>708</td>
<td>1</td>
<td>30</td>
<td>116988</td>
<td>1.02e-10</td>
<td>1.02</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10^{-3}</td>
<td>8</td>
<td>0/0</td>
<td>51</td>
<td>1</td>
<td>6</td>
<td>7238</td>
<td>9.95e-05</td>
<td>1.18</td>
</tr>
<tr>
<td>10^{-4}</td>
<td>10</td>
<td>1/0</td>
<td>69</td>
<td>1</td>
<td>9</td>
<td>9890</td>
<td>3.08e-05</td>
<td>1.18</td>
</tr>
<tr>
<td>10^{-5}</td>
<td>14</td>
<td>1/0</td>
<td>93</td>
<td>1</td>
<td>11</td>
<td>13435</td>
<td>4.36e-06</td>
<td>1.19</td>
</tr>
<tr>
<td>10^{-6}</td>
<td>22</td>
<td>1/0</td>
<td>141</td>
<td>1</td>
<td>13</td>
<td>20361</td>
<td>6.40e-07</td>
<td>1.19</td>
</tr>
<tr>
<td>10^{-7}</td>
<td>35</td>
<td>1/0</td>
<td>219</td>
<td>1</td>
<td>18</td>
<td>31691</td>
<td>1.01e-07</td>
<td>1.19</td>
</tr>
<tr>
<td>10^{-8}</td>
<td>58</td>
<td>1/0</td>
<td>357</td>
<td>1</td>
<td>20</td>
<td>51442</td>
<td>1.97e-08</td>
<td>1.20</td>
</tr>
</tbody>
</table>

232
Table 5.13: Numerical results for Curtis problem by testing order-3 methods

<table>
<thead>
<tr>
<th>tol</th>
<th>nsteps</th>
<th>rej(err/newt)</th>
<th>nfcn</th>
<th>njac</th>
<th>lu</th>
<th>flops</th>
<th>maxerr</th>
<th>niter</th>
</tr>
</thead>
<tbody>
<tr>
<td>SIRK</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10^{-4}</td>
<td>705</td>
<td>36/346</td>
<td>7941</td>
<td>705</td>
<td>1087</td>
<td>1400058</td>
<td>1.39e-05</td>
<td>3.57</td>
</tr>
<tr>
<td>10^{-5}</td>
<td>772</td>
<td>338/353</td>
<td>17193</td>
<td>772</td>
<td>1463</td>
<td>2690710</td>
<td>1.70e-05</td>
<td>5.16</td>
</tr>
<tr>
<td>10^{-6}</td>
<td>418</td>
<td>0/0</td>
<td>7875</td>
<td>418</td>
<td>418</td>
<td>1156869</td>
<td>1.17e-05</td>
<td>6.28</td>
</tr>
<tr>
<td>10^{-7}</td>
<td>677</td>
<td>0/0</td>
<td>8385</td>
<td>677</td>
<td>677</td>
<td>1356561</td>
<td>1.91e-06</td>
<td>4.13</td>
</tr>
<tr>
<td>10^{-8}</td>
<td>1194</td>
<td>0/0</td>
<td>10548</td>
<td>1194</td>
<td>1194</td>
<td>1890662</td>
<td>3.81e-07</td>
<td>2.94</td>
</tr>
<tr>
<td>10^{-9}</td>
<td>2073</td>
<td>0/0</td>
<td>18219</td>
<td>2073</td>
<td>2073</td>
<td>3271299</td>
<td>6.56e-08</td>
<td>2.93</td>
</tr>
<tr>
<td>DESI</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10^{-4}</td>
<td>231</td>
<td>0/229</td>
<td>4311</td>
<td>231</td>
<td>460</td>
<td>884114</td>
<td>9.85e-07</td>
<td>3.11</td>
</tr>
<tr>
<td>10^{-5}</td>
<td>227</td>
<td>0/184</td>
<td>6939</td>
<td>227</td>
<td>411</td>
<td>1251926</td>
<td>1.37e-05</td>
<td>5.09</td>
</tr>
<tr>
<td>10^{-6}</td>
<td>200</td>
<td>5/17</td>
<td>4869</td>
<td>200</td>
<td>222</td>
<td>938966</td>
<td>2.78e-06</td>
<td>3.96</td>
</tr>
<tr>
<td>10^{-7}</td>
<td>311</td>
<td>12/0</td>
<td>5466</td>
<td>311</td>
<td>323</td>
<td>1170708</td>
<td>2.76e-07</td>
<td>2.82</td>
</tr>
<tr>
<td>10^{-8}</td>
<td>550</td>
<td>7/0</td>
<td>8329</td>
<td>550</td>
<td>557</td>
<td>1846421</td>
<td>2.95e-08</td>
<td>2.49</td>
</tr>
<tr>
<td>10^{-9}</td>
<td>1126</td>
<td>18/0</td>
<td>16439</td>
<td>1126</td>
<td>1144</td>
<td>3667528</td>
<td>8.19e-09</td>
<td>2.39</td>
</tr>
<tr>
<td>ESIRK</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10^{-4}</td>
<td>244</td>
<td>44/50</td>
<td>1983</td>
<td>243</td>
<td>337</td>
<td>423092</td>
<td>5.04e-04</td>
<td>2.29</td>
</tr>
<tr>
<td>10^{-5}</td>
<td>261</td>
<td>10/6</td>
<td>1881</td>
<td>260</td>
<td>276</td>
<td>388665</td>
<td>4.64e-05</td>
<td>2.30</td>
</tr>
<tr>
<td>10^{-6}</td>
<td>434</td>
<td>0/0</td>
<td>2454</td>
<td>433</td>
<td>433</td>
<td>550509</td>
<td>6.85e-06</td>
<td>1.88</td>
</tr>
<tr>
<td>10^{-7}</td>
<td>714</td>
<td>0/0</td>
<td>3951</td>
<td>713</td>
<td>713</td>
<td>894618</td>
<td>1.42e-06</td>
<td>1.84</td>
</tr>
<tr>
<td>10^{-8}</td>
<td>1261</td>
<td>0/0</td>
<td>5955</td>
<td>1260</td>
<td>1260</td>
<td>1458050</td>
<td>2.56e-07</td>
<td>1.57</td>
</tr>
<tr>
<td>10^{-9}</td>
<td>2431</td>
<td>0/0</td>
<td>8589</td>
<td>2430</td>
<td>2430</td>
<td>2466689</td>
<td>3.59e-08</td>
<td>1.18</td>
</tr>
<tr>
<td>EDESI</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10^{-4}</td>
<td>179</td>
<td>0/108</td>
<td>3170</td>
<td>178</td>
<td>286</td>
<td>653697</td>
<td>3.52e-04</td>
<td>3.54</td>
</tr>
<tr>
<td>10^{-5}</td>
<td>239</td>
<td>0/140</td>
<td>4022</td>
<td>238</td>
<td>378</td>
<td>847432</td>
<td>7.06e-05</td>
<td>3.36</td>
</tr>
<tr>
<td>10^{-6}</td>
<td>288</td>
<td>0/128</td>
<td>4538</td>
<td>287</td>
<td>415</td>
<td>1006197</td>
<td>2.55e-05</td>
<td>3.15</td>
</tr>
<tr>
<td>10^{-7}</td>
<td>241</td>
<td>0/41</td>
<td>3658</td>
<td>240</td>
<td>281</td>
<td>793577</td>
<td>4.64e-06</td>
<td>3.03</td>
</tr>
<tr>
<td>10^{-8}</td>
<td>295</td>
<td>0/0</td>
<td>4228</td>
<td>294</td>
<td>294</td>
<td>900136</td>
<td>5.24e-07</td>
<td>2.86</td>
</tr>
<tr>
<td>10^{-9}</td>
<td>504</td>
<td>0/0</td>
<td>6437</td>
<td>503</td>
<td>503</td>
<td>1406499</td>
<td>5.68e-08</td>
<td>2.55</td>
</tr>
</tbody>
</table>
Table 5.14: Numerical results for Kaps by testing order-3 methods

<table>
<thead>
<tr>
<th>tol</th>
<th>nsteps</th>
<th>rej(err/newt)</th>
<th>nfcn</th>
<th>njac</th>
<th>lu</th>
<th>flops</th>
<th>maxerr</th>
<th>niter</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SIRK</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$10^{-3}$</td>
<td>19</td>
<td>0/0</td>
<td>96</td>
<td>19</td>
<td>19</td>
<td>18576</td>
<td>2.25e-04</td>
<td>1.68</td>
</tr>
<tr>
<td>$10^{-4}$</td>
<td>29</td>
<td>0/0</td>
<td>165</td>
<td>29</td>
<td>29</td>
<td>29923</td>
<td>5.25e-05</td>
<td>1.90</td>
</tr>
<tr>
<td>$10^{-5}$</td>
<td>48</td>
<td>0/0</td>
<td>264</td>
<td>46</td>
<td>48</td>
<td>48638</td>
<td>1.12e-05</td>
<td>1.83</td>
</tr>
<tr>
<td>$10^{-6}$</td>
<td>83</td>
<td>1/0</td>
<td>447</td>
<td>80</td>
<td>83</td>
<td>83679</td>
<td>3.62e-06</td>
<td>1.77</td>
</tr>
<tr>
<td>$10^{-7}$</td>
<td>134</td>
<td>0/0</td>
<td>762</td>
<td>130</td>
<td>133</td>
<td>137754</td>
<td>4.13e-07</td>
<td>1.90</td>
</tr>
<tr>
<td>$10^{-8}$</td>
<td>236</td>
<td>0/0</td>
<td>1401</td>
<td>205</td>
<td>208</td>
<td>247290</td>
<td>7.49e-08</td>
<td>1.98</td>
</tr>
<tr>
<td>DESI</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$10^{-3}$</td>
<td>10</td>
<td>0/0</td>
<td>102</td>
<td>10</td>
<td>10</td>
<td>22970</td>
<td>2.27e-05</td>
<td>1.70</td>
</tr>
<tr>
<td>$10^{-4}$</td>
<td>14</td>
<td>0/0</td>
<td>142</td>
<td>14</td>
<td>14</td>
<td>32028</td>
<td>6.02e-06</td>
<td>1.69</td>
</tr>
<tr>
<td>$10^{-5}$</td>
<td>21</td>
<td>0/0</td>
<td>203</td>
<td>21</td>
<td>21</td>
<td>47084</td>
<td>1.24e-06</td>
<td>1.61</td>
</tr>
<tr>
<td>$10^{-6}$</td>
<td>32</td>
<td>0/0</td>
<td>301</td>
<td>31</td>
<td>31</td>
<td>71180</td>
<td>2.76e-07</td>
<td>1.57</td>
</tr>
<tr>
<td>$10^{-7}$</td>
<td>53</td>
<td>0/0</td>
<td>451</td>
<td>51</td>
<td>52</td>
<td>113418</td>
<td>4.35e-08</td>
<td>1.42</td>
</tr>
<tr>
<td>$10^{-8}$</td>
<td>90</td>
<td>0/0</td>
<td>710</td>
<td>82</td>
<td>83</td>
<td>187281</td>
<td>6.73e-09</td>
<td>1.31</td>
</tr>
<tr>
<td>ESIRK</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$10^{-3}$</td>
<td>20</td>
<td>0/0</td>
<td>84</td>
<td>18</td>
<td>19</td>
<td>20257</td>
<td>1.39e-04</td>
<td>1.33</td>
</tr>
<tr>
<td>$10^{-4}$</td>
<td>30</td>
<td>0/0</td>
<td>129</td>
<td>26</td>
<td>29</td>
<td>30107</td>
<td>3.68e-05</td>
<td>1.37</td>
</tr>
<tr>
<td>$10^{-5}$</td>
<td>48</td>
<td>0/0</td>
<td>183</td>
<td>31</td>
<td>38</td>
<td>45058</td>
<td>9.16e-06</td>
<td>1.23</td>
</tr>
<tr>
<td>$10^{-6}$</td>
<td>80</td>
<td>0/0</td>
<td>288</td>
<td>51</td>
<td>58</td>
<td>72550</td>
<td>1.95e-06</td>
<td>1.18</td>
</tr>
<tr>
<td>$10^{-7}$</td>
<td>136</td>
<td>0/0</td>
<td>474</td>
<td>68</td>
<td>76</td>
<td>120551</td>
<td>3.84e-07</td>
<td>1.15</td>
</tr>
<tr>
<td>$10^{-8}$</td>
<td>238</td>
<td>0/0</td>
<td>801</td>
<td>99</td>
<td>113</td>
<td>206957</td>
<td>7.24e-08</td>
<td>1.11</td>
</tr>
<tr>
<td>EDESI</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$10^{-3}$</td>
<td>9</td>
<td>0/0</td>
<td>78</td>
<td>7</td>
<td>8</td>
<td>18235</td>
<td>7.86e-03</td>
<td>1.64</td>
</tr>
<tr>
<td>$10^{-4}$</td>
<td>11</td>
<td>0/0</td>
<td>94</td>
<td>10</td>
<td>10</td>
<td>22053</td>
<td>3.29e-04</td>
<td>1.64</td>
</tr>
<tr>
<td>$10^{-5}$</td>
<td>15</td>
<td>0/0</td>
<td>135</td>
<td>14</td>
<td>14</td>
<td>30430</td>
<td>9.38e-05</td>
<td>1.72</td>
</tr>
<tr>
<td>$10^{-6}$</td>
<td>22</td>
<td>0/0</td>
<td>201</td>
<td>20</td>
<td>21</td>
<td>44545</td>
<td>1.58e-05</td>
<td>1.77</td>
</tr>
<tr>
<td>$10^{-7}$</td>
<td>35</td>
<td>0/0</td>
<td>314</td>
<td>33</td>
<td>34</td>
<td>69568</td>
<td>1.99e-06</td>
<td>1.76</td>
</tr>
<tr>
<td>$10^{-8}$</td>
<td>58</td>
<td>0/0</td>
<td>474</td>
<td>56</td>
<td>57</td>
<td>110130</td>
<td>2.22e-07</td>
<td>1.61</td>
</tr>
</tbody>
</table>
### 5.4. Numerical Results

Table 5.15: Numerical results for Oregonator by testing order-3 methods

<table>
<thead>
<tr>
<th>tol</th>
<th>nsteps</th>
<th>rej(err/newt)</th>
<th>nfcn</th>
<th>njac</th>
<th>lu</th>
<th>flops</th>
<th>maxerr</th>
<th>niter</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SIRK</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10^-3</td>
<td>179</td>
<td>6/31</td>
<td>1977</td>
<td>162</td>
<td>214</td>
<td>399415</td>
<td>2.00e+01</td>
<td>3.56</td>
</tr>
<tr>
<td>10^-4</td>
<td>245</td>
<td>8/13</td>
<td>2292</td>
<td>225</td>
<td>260</td>
<td>488355</td>
<td>4.23e+00</td>
<td>3.02</td>
</tr>
<tr>
<td>10^-5</td>
<td>396</td>
<td>8/3</td>
<td>2742</td>
<td>372</td>
<td>393</td>
<td>661425</td>
<td>4.85e-01</td>
<td>2.26</td>
</tr>
<tr>
<td>10^-6</td>
<td>677</td>
<td>3/0</td>
<td>3861</td>
<td>611</td>
<td>634</td>
<td>1017256</td>
<td>8.20e-02</td>
<td>1.89</td>
</tr>
<tr>
<td>10^-7</td>
<td>1190</td>
<td>1/0</td>
<td>6870</td>
<td>823</td>
<td>862</td>
<td>1783692</td>
<td>1.10e-02</td>
<td>1.92</td>
</tr>
<tr>
<td>10^-8</td>
<td>2115</td>
<td>7/0</td>
<td>12654</td>
<td>1190</td>
<td>1305</td>
<td>3218324</td>
<td>1.63e-03</td>
<td>1.99</td>
</tr>
<tr>
<td>DESI</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10^-3</td>
<td>76</td>
<td>24/18</td>
<td>2506</td>
<td>70</td>
<td>118</td>
<td>611702</td>
<td>4.18</td>
<td></td>
</tr>
<tr>
<td>10^-5</td>
<td>133</td>
<td>12/0</td>
<td>2193</td>
<td>126</td>
<td>145</td>
<td>660236</td>
<td>2.52</td>
<td></td>
</tr>
<tr>
<td>10^-6</td>
<td>205</td>
<td>9/0</td>
<td>2902</td>
<td>197</td>
<td>212</td>
<td>911803</td>
<td>2.26</td>
<td></td>
</tr>
<tr>
<td>10^-9</td>
<td>824</td>
<td>4/0</td>
<td>8549</td>
<td>784</td>
<td>799</td>
<td>3054473</td>
<td>1.72</td>
<td></td>
</tr>
<tr>
<td>ESIRK</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10^-3</td>
<td>155</td>
<td>13/0</td>
<td>924</td>
<td>134</td>
<td>163</td>
<td>273821</td>
<td>9.71e+00</td>
<td>1.83</td>
</tr>
<tr>
<td>10^-4</td>
<td>257</td>
<td>7/0</td>
<td>1206</td>
<td>228</td>
<td>253</td>
<td>397899</td>
<td>1.30e+00</td>
<td>1.52</td>
</tr>
<tr>
<td>10^-5</td>
<td>445</td>
<td>4/0</td>
<td>1824</td>
<td>379</td>
<td>424</td>
<td>645542</td>
<td>1.34e-01</td>
<td>1.35</td>
</tr>
<tr>
<td>10^-6</td>
<td>781</td>
<td>2/0</td>
<td>2979</td>
<td>609</td>
<td>712</td>
<td>1096426</td>
<td>2.48e-02</td>
<td>1.27</td>
</tr>
<tr>
<td>10^-7</td>
<td>1383</td>
<td>2/0</td>
<td>4812</td>
<td>970</td>
<td>1225</td>
<td>1877121</td>
<td>2.81e-03</td>
<td>1.16</td>
</tr>
<tr>
<td>10^-8</td>
<td>2455</td>
<td>2/0</td>
<td>8115</td>
<td>1560</td>
<td>2135</td>
<td>3270921</td>
<td>9.01e-04</td>
<td>1.10</td>
</tr>
<tr>
<td>EDESI</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10^-3</td>
<td>63</td>
<td>5/21</td>
<td>1467</td>
<td>55</td>
<td>88</td>
<td>409352</td>
<td>1.04e+01</td>
<td>4.30</td>
</tr>
<tr>
<td>10^-4</td>
<td>84</td>
<td>15/15</td>
<td>1736</td>
<td>77</td>
<td>113</td>
<td>505826</td>
<td>9.47e-02</td>
<td>3.50</td>
</tr>
<tr>
<td>10^-5</td>
<td>119</td>
<td>19/0</td>
<td>1720</td>
<td>109</td>
<td>135</td>
<td>549513</td>
<td>1.99e-01</td>
<td>2.49</td>
</tr>
<tr>
<td>10^-6</td>
<td>189</td>
<td>9/0</td>
<td>2080</td>
<td>175</td>
<td>193</td>
<td>703350</td>
<td>1.41e-01</td>
<td>2.10</td>
</tr>
<tr>
<td>10^-7</td>
<td>325</td>
<td>6/0</td>
<td>2929</td>
<td>300</td>
<td>319</td>
<td>1079028</td>
<td>2.52e-03</td>
<td>1.77</td>
</tr>
<tr>
<td>10^-8</td>
<td>565</td>
<td>3/0</td>
<td>4423</td>
<td>510</td>
<td>540</td>
<td>1752430</td>
<td>7.41e-05</td>
<td>1.56</td>
</tr>
</tbody>
</table>
Table 5.16: Numerical results for Robertson by testing order-3 methods

<table>
<thead>
<tr>
<th>tol</th>
<th>nsteps</th>
<th>rej(err/newt)</th>
<th>nfcn</th>
<th>njac</th>
<th>lu</th>
<th>flops</th>
<th>maxerr</th>
<th>niter</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SIRK</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$10^{-3}$</td>
<td>125</td>
<td>0/0</td>
<td>891</td>
<td>122</td>
<td>125</td>
<td>215691</td>
<td>4.65e-09</td>
<td>2.38</td>
</tr>
<tr>
<td>$10^{-4}$</td>
<td>207</td>
<td>1/0</td>
<td>1269</td>
<td>201</td>
<td>208</td>
<td>330891</td>
<td>1.92e-10</td>
<td>2.03</td>
</tr>
<tr>
<td>$10^{-5}$</td>
<td>349</td>
<td>1/0</td>
<td>1911</td>
<td>336</td>
<td>349</td>
<td>527332</td>
<td>1.37e-10</td>
<td>1.82</td>
</tr>
<tr>
<td>$10^{-6}$</td>
<td>607</td>
<td>1/0</td>
<td>3555</td>
<td>571</td>
<td>592</td>
<td>945858</td>
<td>8.56e-11</td>
<td>1.95</td>
</tr>
<tr>
<td>$10^{-7}$</td>
<td>1061</td>
<td>0/0</td>
<td>6309</td>
<td>685</td>
<td>735</td>
<td>1649285</td>
<td>2.08e-11</td>
<td>1.98</td>
</tr>
<tr>
<td>$10^{-8}$</td>
<td>1876</td>
<td>1/0</td>
<td>11256</td>
<td>1033</td>
<td>1129</td>
<td>2921360</td>
<td>3.84e-12</td>
<td>2.00</td>
</tr>
<tr>
<td>DESI</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$10^{-4}$</td>
<td>72</td>
<td>0/6</td>
<td>1096</td>
<td>70</td>
<td>78</td>
<td>337457</td>
<td>2.47e-10</td>
<td>2.54</td>
</tr>
<tr>
<td>$10^{-5}$</td>
<td>98</td>
<td>0/4</td>
<td>1402</td>
<td>96</td>
<td>102</td>
<td>437621</td>
<td>8.48e-09</td>
<td>2.38</td>
</tr>
<tr>
<td>$10^{-6}$</td>
<td>148</td>
<td>0/9</td>
<td>2077</td>
<td>146</td>
<td>157</td>
<td>660219</td>
<td>6.17e-09</td>
<td>2.34</td>
</tr>
<tr>
<td>$10^{-7}$</td>
<td>219</td>
<td>1/0</td>
<td>2709</td>
<td>217</td>
<td>220</td>
<td>902926</td>
<td>6.76e-10</td>
<td>2.05</td>
</tr>
<tr>
<td>$10^{-8}$</td>
<td>351</td>
<td>1/0</td>
<td>3801</td>
<td>347</td>
<td>352</td>
<td>1363778</td>
<td>7.72e-13</td>
<td>1.80</td>
</tr>
<tr>
<td>$10^{-9}$</td>
<td>574</td>
<td>1/0</td>
<td>5976</td>
<td>567</td>
<td>572</td>
<td>2176799</td>
<td>2.83e-13</td>
<td>1.73</td>
</tr>
<tr>
<td>ESIrK</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$10^{-3}$</td>
<td>147</td>
<td>2/0</td>
<td>564</td>
<td>112</td>
<td>135</td>
<td>212867</td>
<td>5.09e-09</td>
<td>1.26</td>
</tr>
<tr>
<td>$10^{-4}$</td>
<td>228</td>
<td>1/0</td>
<td>807</td>
<td>129</td>
<td>172</td>
<td>315001</td>
<td>1.64e-09</td>
<td>1.17</td>
</tr>
<tr>
<td>$10^{-5}$</td>
<td>370</td>
<td>0/0</td>
<td>1254</td>
<td>168</td>
<td>232</td>
<td>497820</td>
<td>4.30e-10</td>
<td>1.13</td>
</tr>
<tr>
<td>$10^{-6}$</td>
<td>628</td>
<td>2/0</td>
<td>2076</td>
<td>235</td>
<td>314</td>
<td>832622</td>
<td>1.09e-10</td>
<td>1.10</td>
</tr>
<tr>
<td>$10^{-7}$</td>
<td>1083</td>
<td>1/0</td>
<td>3510</td>
<td>318</td>
<td>422</td>
<td>1415001</td>
<td>2.33e-11</td>
<td>1.08</td>
</tr>
<tr>
<td>$10^{-8}$</td>
<td>1897</td>
<td>1/0</td>
<td>6063</td>
<td>498</td>
<td>623</td>
<td>2457442</td>
<td>4.43e-12</td>
<td>1.06</td>
</tr>
<tr>
<td>EDesi</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$10^{-3}$</td>
<td>53</td>
<td>0/2</td>
<td>541</td>
<td>50</td>
<td>54</td>
<td>184581</td>
<td>3.06e-09</td>
<td>2.03</td>
</tr>
<tr>
<td>$10^{-4}$</td>
<td>67</td>
<td>0/0</td>
<td>649</td>
<td>64</td>
<td>66</td>
<td>227558</td>
<td>1.82e-09</td>
<td>1.93</td>
</tr>
<tr>
<td>$10^{-5}$</td>
<td>98</td>
<td>0/0</td>
<td>875</td>
<td>95</td>
<td>97</td>
<td>323498</td>
<td>3.34e-10</td>
<td>1.78</td>
</tr>
<tr>
<td>$10^{-6}$</td>
<td>158</td>
<td>1/0</td>
<td>1408</td>
<td>154</td>
<td>158</td>
<td>518063</td>
<td>8.28e-11</td>
<td>1.77</td>
</tr>
<tr>
<td>$10^{-7}$</td>
<td>263</td>
<td>1/0</td>
<td>2010</td>
<td>257</td>
<td>263</td>
<td>813004</td>
<td>1.34e-11</td>
<td>1.52</td>
</tr>
<tr>
<td>$10^{-8}$</td>
<td>448</td>
<td>1/0</td>
<td>3013</td>
<td>432</td>
<td>441</td>
<td>1323958</td>
<td>3.91e-12</td>
<td>1.34</td>
</tr>
</tbody>
</table>
### Table 5.17: Numerical results for Van der Pol by testing order-3 methods

<table>
<thead>
<tr>
<th>tol</th>
<th>nsteps</th>
<th>rej(err/newt)</th>
<th>nfcn</th>
<th>njac</th>
<th>lu</th>
<th>flops</th>
<th>maxerr</th>
<th>niter</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SIRK</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$10^{-3}$</td>
<td>310</td>
<td>18/17</td>
<td>3387</td>
<td>277</td>
<td>324</td>
<td>467644</td>
<td>5.89e-04</td>
<td>3.44</td>
</tr>
<tr>
<td>$10^{-4}$</td>
<td>486</td>
<td>16/0</td>
<td>3546</td>
<td>412</td>
<td>442</td>
<td>569033</td>
<td>4.61e-04</td>
<td>2.35</td>
</tr>
<tr>
<td>$10^{-5}$</td>
<td>837</td>
<td>17/0</td>
<td>5088</td>
<td>641</td>
<td>686</td>
<td>884954</td>
<td>6.61e-05</td>
<td>1.99</td>
</tr>
<tr>
<td>$10^{-6}$</td>
<td>1472</td>
<td>11/0</td>
<td>8163</td>
<td>1037</td>
<td>1091</td>
<td>1476563</td>
<td>1.36e-05</td>
<td>1.83</td>
</tr>
<tr>
<td>$10^{-7}$</td>
<td>2586</td>
<td>7/0</td>
<td>14838</td>
<td>1409</td>
<td>1503</td>
<td>2621753</td>
<td>2.62e-06</td>
<td>1.91</td>
</tr>
<tr>
<td>$10^{-8}$</td>
<td>4564</td>
<td>6/0</td>
<td>28191</td>
<td>2180</td>
<td>2320</td>
<td>4789765</td>
<td>2.44e-07</td>
<td>2.06</td>
</tr>
<tr>
<td>DESI</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$10^{-3}$</td>
<td>135</td>
<td>46/25</td>
<td>3711</td>
<td>111</td>
<td>199</td>
<td>636859</td>
<td>9.41e-04</td>
<td>3.42</td>
</tr>
<tr>
<td>$10^{-4}$</td>
<td>184</td>
<td>50/5</td>
<td>4008</td>
<td>148</td>
<td>226</td>
<td>725490</td>
<td>1.09e-04</td>
<td>2.85</td>
</tr>
<tr>
<td>$10^{-5}$</td>
<td>270</td>
<td>41/0</td>
<td>4361</td>
<td>213</td>
<td>282</td>
<td>849132</td>
<td>2.48e-07</td>
<td>2.34</td>
</tr>
<tr>
<td>$10^{-6}$</td>
<td>406</td>
<td>22/0</td>
<td>5313</td>
<td>317</td>
<td>372</td>
<td>1090620</td>
<td>5.92e-07</td>
<td>2.07</td>
</tr>
<tr>
<td>$10^{-7}$</td>
<td>638</td>
<td>11/0</td>
<td>7588</td>
<td>495</td>
<td>543</td>
<td>1596217</td>
<td>3.90e-08</td>
<td>1.95</td>
</tr>
<tr>
<td>$10^{-8}$</td>
<td>1037</td>
<td>13/0</td>
<td>11551</td>
<td>807</td>
<td>863</td>
<td>2491992</td>
<td>9.06e-09</td>
<td>1.83</td>
</tr>
<tr>
<td>ESIRK</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$10^{-3}$</td>
<td>317</td>
<td>25/0</td>
<td>1626</td>
<td>172</td>
<td>275</td>
<td>341005</td>
<td>2.03e-03</td>
<td>1.58</td>
</tr>
<tr>
<td>$10^{-4}$</td>
<td>518</td>
<td>20/0</td>
<td>2172</td>
<td>225</td>
<td>364</td>
<td>499879</td>
<td>2.44e-04</td>
<td>1.34</td>
</tr>
<tr>
<td>$10^{-5}$</td>
<td>853</td>
<td>18/0</td>
<td>3360</td>
<td>618</td>
<td>690</td>
<td>795337</td>
<td>6.37e-05</td>
<td>1.28</td>
</tr>
<tr>
<td>$10^{-6}$</td>
<td>1462</td>
<td>8/0</td>
<td>5481</td>
<td>490</td>
<td>688</td>
<td>1310768</td>
<td>1.40e-05</td>
<td>1.24</td>
</tr>
<tr>
<td>$10^{-7}$</td>
<td>2571</td>
<td>8/0</td>
<td>9108</td>
<td>757</td>
<td>998</td>
<td>2246127</td>
<td>2.12e-06</td>
<td>1.18</td>
</tr>
<tr>
<td>$10^{-8}$</td>
<td>4551</td>
<td>5/0</td>
<td>15141</td>
<td>1073</td>
<td>1365</td>
<td>3871936</td>
<td>3.83e-07</td>
<td>1.11</td>
</tr>
<tr>
<td>EDES1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$10^{-3}$</td>
<td>111</td>
<td>15/40</td>
<td>2554</td>
<td>87</td>
<td>158</td>
<td>442945</td>
<td>7.94e-04</td>
<td>4.05</td>
</tr>
<tr>
<td>$10^{-4}$</td>
<td>152</td>
<td>26/32</td>
<td>3006</td>
<td>115</td>
<td>197</td>
<td>552768</td>
<td>8.48e-05</td>
<td>3.37</td>
</tr>
<tr>
<td>$10^{-5}$</td>
<td>239</td>
<td>34/4</td>
<td>2878</td>
<td>177</td>
<td>244</td>
<td>602615</td>
<td>1.90e-05</td>
<td>2.11</td>
</tr>
<tr>
<td>$10^{-6}$</td>
<td>374</td>
<td>21/0</td>
<td>3566</td>
<td>269</td>
<td>326</td>
<td>790712</td>
<td>1.61e-06</td>
<td>1.80</td>
</tr>
<tr>
<td>$10^{-7}$</td>
<td>625</td>
<td>9/0</td>
<td>5012</td>
<td>438</td>
<td>494</td>
<td>1194759</td>
<td>3.02e-07</td>
<td>1.58</td>
</tr>
<tr>
<td>$10^{-8}$</td>
<td>1103</td>
<td>11/0</td>
<td>7879</td>
<td>761</td>
<td>924</td>
<td>2006080</td>
<td>6.01e-08</td>
<td>1.41</td>
</tr>
</tbody>
</table>
Table 5.18: Numerical results for Prothero-Robinson by testing order-4 methods

<table>
<thead>
<tr>
<th>tol</th>
<th>nsteps</th>
<th>rej(err/newt)</th>
<th>nfcn</th>
<th>njac</th>
<th>lu</th>
<th>flops</th>
<th>maxerr</th>
<th>niter</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>SIRK</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$10^{-3}$</td>
<td>17</td>
<td>2/0</td>
<td>80</td>
<td>1</td>
<td>14</td>
<td>16507</td>
<td>3.78e−06</td>
<td>1.05</td>
</tr>
<tr>
<td>$10^{-4}$</td>
<td>24</td>
<td>2/0</td>
<td>108</td>
<td>1</td>
<td>13</td>
<td>22431</td>
<td>5.41e−07</td>
<td>1.04</td>
</tr>
<tr>
<td>$10^{-5}$</td>
<td>34</td>
<td>1/0</td>
<td>148</td>
<td>1</td>
<td>14</td>
<td>30333</td>
<td>1.55e−07</td>
<td>1.06</td>
</tr>
<tr>
<td>$10^{-6}$</td>
<td>52</td>
<td>2/0</td>
<td>224</td>
<td>1</td>
<td>16</td>
<td>46433</td>
<td>2.45e−08</td>
<td>1.04</td>
</tr>
<tr>
<td>$10^{-7}$</td>
<td>78</td>
<td>2/0</td>
<td>332</td>
<td>1</td>
<td>18</td>
<td>68736</td>
<td>2.44e−09</td>
<td>1.04</td>
</tr>
<tr>
<td>$10^{-8}$</td>
<td>120</td>
<td>2/0</td>
<td>500</td>
<td>1</td>
<td>19</td>
<td>104315</td>
<td>3.66e−10</td>
<td>1.02</td>
</tr>
<tr>
<td><strong>DESI</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$10^{-3}$</td>
<td>7</td>
<td>0/0</td>
<td>67</td>
<td>1</td>
<td>5</td>
<td>10323</td>
<td>2.76e−06</td>
<td>1.37</td>
</tr>
<tr>
<td>$10^{-4}$</td>
<td>10</td>
<td>2/0</td>
<td>112</td>
<td>1</td>
<td>10</td>
<td>17410</td>
<td>3.34e−07</td>
<td>1.33</td>
</tr>
<tr>
<td>$10^{-5}$</td>
<td>13</td>
<td>2/0</td>
<td>143</td>
<td>1</td>
<td>12</td>
<td>21941</td>
<td>8.47e−08</td>
<td>1.36</td>
</tr>
<tr>
<td>$10^{-6}$</td>
<td>17</td>
<td>2/0</td>
<td>179</td>
<td>1</td>
<td>12</td>
<td>27597</td>
<td>1.39e−08</td>
<td>1.35</td>
</tr>
<tr>
<td>$10^{-7}$</td>
<td>24</td>
<td>2/0</td>
<td>242</td>
<td>1</td>
<td>14</td>
<td>37506</td>
<td>1.94e−09</td>
<td>1.33</td>
</tr>
<tr>
<td>$10^{-8}$</td>
<td>36</td>
<td>2/0</td>
<td>354</td>
<td>1</td>
<td>16</td>
<td>54762</td>
<td>2.48e−10</td>
<td>1.33</td>
</tr>
<tr>
<td><strong>ESIRK</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$10^{-3}$</td>
<td>18</td>
<td>2/0</td>
<td>88</td>
<td>1</td>
<td>13</td>
<td>17949</td>
<td>3.33e−08</td>
<td>1.02</td>
</tr>
<tr>
<td>$10^{-4}$</td>
<td>25</td>
<td>2/0</td>
<td>116</td>
<td>1</td>
<td>14</td>
<td>23574</td>
<td>9.81e−09</td>
<td>1.02</td>
</tr>
<tr>
<td>$10^{-5}$</td>
<td>36</td>
<td>2/0</td>
<td>160</td>
<td>1</td>
<td>17</td>
<td>32662</td>
<td>1.35e−09</td>
<td>1.01</td>
</tr>
<tr>
<td>$10^{-6}$</td>
<td>54</td>
<td>2/0</td>
<td>232</td>
<td>1</td>
<td>18</td>
<td>47119</td>
<td>4.13e−10</td>
<td>1.01</td>
</tr>
<tr>
<td>$10^{-7}$</td>
<td>80</td>
<td>2/0</td>
<td>340</td>
<td>1</td>
<td>25</td>
<td>68459</td>
<td>2.82e−11</td>
<td>1.02</td>
</tr>
<tr>
<td>$10^{-8}$</td>
<td>123</td>
<td>2/0</td>
<td>516</td>
<td>1</td>
<td>54</td>
<td>103456</td>
<td>3.59e−12</td>
<td>1.02</td>
</tr>
<tr>
<td><strong>EDESI</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$10^{-3}$</td>
<td>11</td>
<td>1/0</td>
<td>89</td>
<td>1</td>
<td>9</td>
<td>15573</td>
<td>1.16e−05</td>
<td>1.15</td>
</tr>
<tr>
<td>$10^{-4}$</td>
<td>14</td>
<td>2/0</td>
<td>117</td>
<td>1</td>
<td>11</td>
<td>20532</td>
<td>2.68e−06</td>
<td>1.16</td>
</tr>
<tr>
<td>$10^{-5}$</td>
<td>19</td>
<td>1/0</td>
<td>145</td>
<td>1</td>
<td>13</td>
<td>25497</td>
<td>3.79e−07</td>
<td>1.16</td>
</tr>
<tr>
<td>$10^{-6}$</td>
<td>27</td>
<td>2/0</td>
<td>208</td>
<td>1</td>
<td>19</td>
<td>36810</td>
<td>8.42e−08</td>
<td>1.16</td>
</tr>
<tr>
<td>$10^{-7}$</td>
<td>40</td>
<td>2/0</td>
<td>299</td>
<td>1</td>
<td>26</td>
<td>52890</td>
<td>7.49e−09</td>
<td>1.16</td>
</tr>
<tr>
<td>$10^{-8}$</td>
<td>58</td>
<td>2/0</td>
<td>425</td>
<td>1</td>
<td>31</td>
<td>74815</td>
<td>1.38e−09</td>
<td>1.16</td>
</tr>
</tbody>
</table>
Table 5.19: Numerical results for Curtis problem by testing order-4 methods

<table>
<thead>
<tr>
<th>tol</th>
<th>nsteps</th>
<th>rej(err/newt)</th>
<th>nfcn</th>
<th>njac</th>
<th>lu</th>
<th>flops</th>
<th>maxerr</th>
<th>niter</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>SIRK</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$10^{-5}$</td>
<td>873</td>
<td>0/437</td>
<td>7920</td>
<td>873</td>
<td>1310</td>
<td>1991128</td>
<td>4.85e−08</td>
<td>2.27</td>
</tr>
<tr>
<td>$10^{-6}$</td>
<td>1437</td>
<td>1/717</td>
<td>11772</td>
<td>1437</td>
<td>2155</td>
<td>3113525</td>
<td>6.25e−09</td>
<td>2.05</td>
</tr>
<tr>
<td>$10^{-7}$</td>
<td>1716</td>
<td>222/842</td>
<td>23912</td>
<td>1716</td>
<td>2780</td>
<td>5187288</td>
<td>5.82e−08</td>
<td>3.08</td>
</tr>
<tr>
<td>$10^{-8}$</td>
<td>1278</td>
<td>236/317</td>
<td>23616</td>
<td>1278</td>
<td>1831</td>
<td>4553967</td>
<td>5.77e−08</td>
<td>3.90</td>
</tr>
<tr>
<td>$10^{-9}$</td>
<td>1107</td>
<td>8/8</td>
<td>18280</td>
<td>1107</td>
<td>1123</td>
<td>3381835</td>
<td>1.18e−08</td>
<td>4.10</td>
</tr>
<tr>
<td>$10^{-10}$</td>
<td>1684</td>
<td>0/0</td>
<td>19928</td>
<td>1684</td>
<td>1684</td>
<td>4106407</td>
<td>2.11e−09</td>
<td>2.96</td>
</tr>
<tr>
<td><strong>DESI</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$10^{-5}$</td>
<td>336</td>
<td>0/325</td>
<td>11733</td>
<td>336</td>
<td>661</td>
<td>2210468</td>
<td>1.06e−05</td>
<td>4.99</td>
</tr>
<tr>
<td>$10^{-6}$</td>
<td>328</td>
<td>0/289</td>
<td>12889</td>
<td>328</td>
<td>617</td>
<td>2337135</td>
<td>1.47e−07</td>
<td>5.61</td>
</tr>
<tr>
<td>$10^{-7}$</td>
<td>341</td>
<td>0/280</td>
<td>14840</td>
<td>341</td>
<td>621</td>
<td>2611850</td>
<td>2.97e−08</td>
<td>6.22</td>
</tr>
<tr>
<td>$10^{-8}$</td>
<td>358</td>
<td>0/229</td>
<td>18445</td>
<td>358</td>
<td>587</td>
<td>3101506</td>
<td>1.79e−08</td>
<td>7.36</td>
</tr>
<tr>
<td>$10^{-9}$</td>
<td>312</td>
<td>0/6</td>
<td>9308</td>
<td>312</td>
<td>318</td>
<td>1807109</td>
<td>6.13e−09</td>
<td>4.26</td>
</tr>
<tr>
<td>$10^{-10}$</td>
<td>472</td>
<td>0/0</td>
<td>12060</td>
<td>472</td>
<td>472</td>
<td>2426821</td>
<td>1.10e−09</td>
<td>3.65</td>
</tr>
<tr>
<td><strong>ESIRK</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$10^{-5}$</td>
<td>203</td>
<td>13/5</td>
<td>1992</td>
<td>202</td>
<td>220</td>
<td>472926</td>
<td>5.33e−05</td>
<td>2.29</td>
</tr>
<tr>
<td>$10^{-6}$</td>
<td>288</td>
<td>6/1</td>
<td>2092</td>
<td>287</td>
<td>294</td>
<td>557001</td>
<td>5.45e−06</td>
<td>1.77</td>
</tr>
<tr>
<td>$10^{-7}$</td>
<td>438</td>
<td>0/0</td>
<td>2596</td>
<td>437</td>
<td>437</td>
<td>755389</td>
<td>5.22e−07</td>
<td>1.47</td>
</tr>
<tr>
<td>$10^{-8}$</td>
<td>693</td>
<td>0/0</td>
<td>3584</td>
<td>692</td>
<td>692</td>
<td>1121239</td>
<td>8.17e−08</td>
<td>1.29</td>
</tr>
<tr>
<td>$10^{-9}$</td>
<td>1035</td>
<td>0/0</td>
<td>5020</td>
<td>1034</td>
<td>1034</td>
<td>1611679</td>
<td>1.39e−08</td>
<td>1.21</td>
</tr>
<tr>
<td>$10^{-10}$</td>
<td>1637</td>
<td>0/0</td>
<td>7540</td>
<td>1636</td>
<td>1636</td>
<td>2495835</td>
<td>2.27e−09</td>
<td>1.15</td>
</tr>
<tr>
<td><strong>EDES1</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$10^{-5}$</td>
<td>245</td>
<td>2/42</td>
<td>3827</td>
<td>244</td>
<td>288</td>
<td>969536</td>
<td>3.31e−05</td>
<td>2.58</td>
</tr>
<tr>
<td>$10^{-6}$</td>
<td>250</td>
<td>2/42</td>
<td>3860</td>
<td>249</td>
<td>293</td>
<td>979827</td>
<td>2.39e−06</td>
<td>2.55</td>
</tr>
<tr>
<td>$10^{-7}$</td>
<td>228</td>
<td>7/14</td>
<td>3584</td>
<td>227</td>
<td>248</td>
<td>887090</td>
<td>6.35e−07</td>
<td>2.54</td>
</tr>
<tr>
<td>$10^{-8}$</td>
<td>266</td>
<td>0/1</td>
<td>4081</td>
<td>265</td>
<td>266</td>
<td>987642</td>
<td>9.20e−08</td>
<td>2.55</td>
</tr>
<tr>
<td>$10^{-9}$</td>
<td>413</td>
<td>0/0</td>
<td>5333</td>
<td>412</td>
<td>412</td>
<td>1354435</td>
<td>5.58e−09</td>
<td>2.15</td>
</tr>
<tr>
<td>$10^{-10}$</td>
<td>656</td>
<td>0/0</td>
<td>7222</td>
<td>655</td>
<td>655</td>
<td>1979387</td>
<td>7.47e−10</td>
<td>1.83</td>
</tr>
</tbody>
</table>
Table 5.20: Numerical results for Kaps by testing order-4 methods

<table>
<thead>
<tr>
<th>tol</th>
<th>nsteps</th>
<th>rej(err/newt)</th>
<th>nfcn</th>
<th>njac</th>
<th>lu</th>
<th>flops</th>
<th>maxerr</th>
<th>niter</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SIRK</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$10^{-3}$</td>
<td>17</td>
<td>0/0</td>
<td>120</td>
<td>17</td>
<td>17</td>
<td>27206</td>
<td>1.12e-04</td>
<td>1.76</td>
</tr>
<tr>
<td>$10^{-4}$</td>
<td>25</td>
<td>0/0</td>
<td>156</td>
<td>24</td>
<td>24</td>
<td>37901</td>
<td>2.45e-05</td>
<td>1.56</td>
</tr>
<tr>
<td>$10^{-5}$</td>
<td>37</td>
<td>0/0</td>
<td>240</td>
<td>36</td>
<td>36</td>
<td>56962</td>
<td>5.03e-06</td>
<td>1.62</td>
</tr>
<tr>
<td>$10^{-6}$</td>
<td>55</td>
<td>0/0</td>
<td>380</td>
<td>51</td>
<td>51</td>
<td>86896</td>
<td>9.75e-07</td>
<td>1.73</td>
</tr>
<tr>
<td>$10^{-7}$</td>
<td>86</td>
<td>0/0</td>
<td>656</td>
<td>81</td>
<td>83</td>
<td>142026</td>
<td>1.68e-07</td>
<td>1.91</td>
</tr>
<tr>
<td>$10^{-8}$</td>
<td>135</td>
<td>0/0</td>
<td>1020</td>
<td>128</td>
<td>129</td>
<td>221898</td>
<td>2.83e-08</td>
<td>1.89</td>
</tr>
<tr>
<td>DESI</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$10^{-3}$</td>
<td>8</td>
<td>0/0</td>
<td>133</td>
<td>8</td>
<td>8</td>
<td>27816</td>
<td>2.41e-05</td>
<td>2.38</td>
</tr>
<tr>
<td>$10^{-4}$</td>
<td>11</td>
<td>0/0</td>
<td>176</td>
<td>11</td>
<td>11</td>
<td>37488</td>
<td>6.99e-06</td>
<td>2.29</td>
</tr>
<tr>
<td>$10^{-5}$</td>
<td>14</td>
<td>1/0</td>
<td>240</td>
<td>14</td>
<td>15</td>
<td>50757</td>
<td>1.65e-06</td>
<td>2.29</td>
</tr>
<tr>
<td>$10^{-6}$</td>
<td>19</td>
<td>0/0</td>
<td>310</td>
<td>18</td>
<td>19</td>
<td>65424</td>
<td>3.49e-07</td>
<td>2.33</td>
</tr>
<tr>
<td>$10^{-7}$</td>
<td>27</td>
<td>0/0</td>
<td>443</td>
<td>26</td>
<td>27</td>
<td>93116</td>
<td>6.54e-08</td>
<td>2.34</td>
</tr>
<tr>
<td>$10^{-8}$</td>
<td>40</td>
<td>0/0</td>
<td>648</td>
<td>39</td>
<td>40</td>
<td>136878</td>
<td>1.15e-08</td>
<td>2.31</td>
</tr>
<tr>
<td>ESIRK</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$10^{-3}$</td>
<td>20</td>
<td>0/0</td>
<td>104</td>
<td>16</td>
<td>19</td>
<td>31587</td>
<td>6.34e-05</td>
<td>1.19</td>
</tr>
<tr>
<td>$10^{-4}$</td>
<td>28</td>
<td>0/0</td>
<td>140</td>
<td>23</td>
<td>26</td>
<td>42555</td>
<td>1.54e-05</td>
<td>1.17</td>
</tr>
<tr>
<td>$10^{-5}$</td>
<td>40</td>
<td>0/0</td>
<td>196</td>
<td>34</td>
<td>37</td>
<td>58972</td>
<td>3.36e-06</td>
<td>1.17</td>
</tr>
<tr>
<td>$10^{-6}$</td>
<td>60</td>
<td>0/0</td>
<td>280</td>
<td>41</td>
<td>46</td>
<td>85317</td>
<td>7.17e-07</td>
<td>1.13</td>
</tr>
<tr>
<td>$10^{-7}$</td>
<td>91</td>
<td>0/0</td>
<td>416</td>
<td>63</td>
<td>69</td>
<td>126848</td>
<td>1.40e-07</td>
<td>1.12</td>
</tr>
<tr>
<td>$10^{-8}$</td>
<td>140</td>
<td>0/0</td>
<td>624</td>
<td>85</td>
<td>92</td>
<td>191446</td>
<td>2.48e-08</td>
<td>1.10</td>
</tr>
<tr>
<td>EDESI</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$10^{-4}$</td>
<td>18</td>
<td>0/0</td>
<td>132</td>
<td>16</td>
<td>16</td>
<td>43818</td>
<td>2.43e-06</td>
<td>1.17</td>
</tr>
<tr>
<td>$10^{-5}$</td>
<td>23</td>
<td>0/0</td>
<td>171</td>
<td>21</td>
<td>21</td>
<td>55817</td>
<td>8.17e-07</td>
<td>1.17</td>
</tr>
<tr>
<td>$10^{-6}$</td>
<td>31</td>
<td>1/0</td>
<td>246</td>
<td>30</td>
<td>31</td>
<td>78395</td>
<td>2.07e-07</td>
<td>1.23</td>
</tr>
<tr>
<td>$10^{-7}$</td>
<td>43</td>
<td>0/0</td>
<td>317</td>
<td>39</td>
<td>41</td>
<td>102605</td>
<td>2.08e-08</td>
<td>1.19</td>
</tr>
<tr>
<td>$10^{-8}$</td>
<td>63</td>
<td>0/0</td>
<td>461</td>
<td>42</td>
<td>47</td>
<td>148288</td>
<td>3.52e-09</td>
<td>1.20</td>
</tr>
<tr>
<td>$10^{-9}$</td>
<td>95</td>
<td>0/0</td>
<td>681</td>
<td>41</td>
<td>49</td>
<td>220002</td>
<td>7.91e-10</td>
<td>1.18</td>
</tr>
</tbody>
</table>
### 5.4. Numerical Results

Table 5.21: Numerical results for Oregonator by testing order-4 methods

<table>
<thead>
<tr>
<th>tol</th>
<th>nsteps</th>
<th>rej(err/newt)</th>
<th>nfcn</th>
<th>njac</th>
<th>lu</th>
<th>flops</th>
<th>maxerr</th>
<th>niter</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SIRK</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$10^{-3}$</td>
<td>295</td>
<td>4/106</td>
<td>3932</td>
<td>280</td>
<td>404</td>
<td>985939</td>
<td>3.14e + 00</td>
<td>3.29</td>
</tr>
<tr>
<td>$10^{-4}$</td>
<td>297</td>
<td>7/63</td>
<td>5188</td>
<td>280</td>
<td>362</td>
<td>1146815</td>
<td>2.04e + 00</td>
<td>4.27</td>
</tr>
<tr>
<td>$10^{-5}$</td>
<td>366</td>
<td>9/35</td>
<td>6444</td>
<td>347</td>
<td>400</td>
<td>1400929</td>
<td>3.93e - 01</td>
<td>4.30</td>
</tr>
<tr>
<td>$10^{-6}$</td>
<td>519</td>
<td>8/22</td>
<td>7076</td>
<td>498</td>
<td>537</td>
<td>1681171</td>
<td>6.17e - 02</td>
<td>3.36</td>
</tr>
<tr>
<td>$10^{-7}$</td>
<td>770</td>
<td>2/2</td>
<td>8088</td>
<td>744</td>
<td>756</td>
<td>2132321</td>
<td>1.19e - 02</td>
<td>2.62</td>
</tr>
<tr>
<td>$10^{-8}$</td>
<td>1202</td>
<td>1/0</td>
<td>10496</td>
<td>1160</td>
<td>1169</td>
<td>3028526</td>
<td>1.81e - 03</td>
<td>2.18</td>
</tr>
<tr>
<td>DESI</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$10^{-3}$</td>
<td>74</td>
<td>8/29</td>
<td>2905</td>
<td>67</td>
<td>111</td>
<td>682804</td>
<td>–</td>
<td>5.06</td>
</tr>
<tr>
<td>$10^{-4}$</td>
<td>94</td>
<td>13/41</td>
<td>4170</td>
<td>87</td>
<td>148</td>
<td>950717</td>
<td>–</td>
<td>5.57</td>
</tr>
<tr>
<td>$10^{-5}$</td>
<td>121</td>
<td>14/17</td>
<td>3842</td>
<td>112</td>
<td>152</td>
<td>995167</td>
<td>–</td>
<td>4.07</td>
</tr>
<tr>
<td>$10^{-8}$</td>
<td>376</td>
<td>9/0</td>
<td>6984</td>
<td>354</td>
<td>373</td>
<td>2204349</td>
<td>–</td>
<td>2.59</td>
</tr>
<tr>
<td>ESIRK</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$10^{-3}$</td>
<td>170</td>
<td>15/0</td>
<td>1232</td>
<td>146</td>
<td>180</td>
<td>450306</td>
<td>1.03e + 01</td>
<td>1.66</td>
</tr>
<tr>
<td>$10^{-4}$</td>
<td>240</td>
<td>10/0</td>
<td>1424</td>
<td>211</td>
<td>240</td>
<td>572281</td>
<td>1.79e + 00</td>
<td>1.42</td>
</tr>
<tr>
<td>$10^{-5}$</td>
<td>360</td>
<td>6/0</td>
<td>1912</td>
<td>325</td>
<td>346</td>
<td>808170</td>
<td>3.72e - 01</td>
<td>1.30</td>
</tr>
<tr>
<td>$10^{-6}$</td>
<td>561</td>
<td>5/0</td>
<td>2688</td>
<td>502</td>
<td>531</td>
<td>1207472</td>
<td>4.69e - 02</td>
<td>1.18</td>
</tr>
<tr>
<td>$10^{-7}$</td>
<td>883</td>
<td>5/0</td>
<td>4000</td>
<td>741</td>
<td>821</td>
<td>1856783</td>
<td>7.88e - 03</td>
<td>1.12</td>
</tr>
<tr>
<td>$10^{-8}$</td>
<td>1389</td>
<td>3/0</td>
<td>6096</td>
<td>1092</td>
<td>1262</td>
<td>2881278</td>
<td>1.26e - 03</td>
<td>1.09</td>
</tr>
<tr>
<td>EDESI</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$10^{-3}$</td>
<td>218</td>
<td>65/0</td>
<td>2854</td>
<td>206</td>
<td>282</td>
<td>1219691</td>
<td>8.67e - 01</td>
<td>1.68</td>
</tr>
<tr>
<td>$10^{-4}$</td>
<td>151</td>
<td>18/0</td>
<td>1889</td>
<td>141</td>
<td>167</td>
<td>752492</td>
<td>1.59e - 01</td>
<td>1.86</td>
</tr>
<tr>
<td>$10^{-5}$</td>
<td>186</td>
<td>16/0</td>
<td>2132</td>
<td>174</td>
<td>199</td>
<td>880133</td>
<td>4.19e - 02</td>
<td>1.75</td>
</tr>
<tr>
<td>$10^{-6}$</td>
<td>251</td>
<td>13/0</td>
<td>2480</td>
<td>233</td>
<td>257</td>
<td>1099023</td>
<td>8.89e - 03</td>
<td>1.56</td>
</tr>
<tr>
<td>$10^{-7}$</td>
<td>351</td>
<td>8/0</td>
<td>3219</td>
<td>324</td>
<td>345</td>
<td>1470832</td>
<td>1.94e - 03</td>
<td>1.49</td>
</tr>
<tr>
<td>$10^{-8}$</td>
<td>558</td>
<td>21/0</td>
<td>4911</td>
<td>515</td>
<td>552</td>
<td>2324093</td>
<td>4.50e - 04</td>
<td>1.41</td>
</tr>
</tbody>
</table>

241


Table 5.22: Numerical results for Robertson by testing order-4 methods

<table>
<thead>
<tr>
<th>tol</th>
<th>nsteps</th>
<th>rej(err/newt)</th>
<th>nfcn</th>
<th>njac</th>
<th>lu</th>
<th>flops</th>
<th>maxerr</th>
<th>niter</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SIRK</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10^{-3}</td>
<td>130</td>
<td>0/10</td>
<td>1544</td>
<td>126</td>
<td>140</td>
<td>398590</td>
<td>2.77e-10</td>
<td>2.97</td>
</tr>
<tr>
<td>10^{-4}</td>
<td>176</td>
<td>0/0</td>
<td>1984</td>
<td>172</td>
<td>176</td>
<td>517918</td>
<td>8.57e-10</td>
<td>2.82</td>
</tr>
<tr>
<td>10^{-5}</td>
<td>276</td>
<td>0/0</td>
<td>2672</td>
<td>267</td>
<td>273</td>
<td>748337</td>
<td>7.02e-11</td>
<td>2.42</td>
</tr>
<tr>
<td>10^{-6}</td>
<td>438</td>
<td>1/0</td>
<td>3728</td>
<td>428</td>
<td>437</td>
<td>1114730</td>
<td>2.90e-11</td>
<td>2.12</td>
</tr>
<tr>
<td>10^{-7}</td>
<td>694</td>
<td>1/0</td>
<td>5280</td>
<td>678</td>
<td>688</td>
<td>1674494</td>
<td>1.17e-11</td>
<td>1.90</td>
</tr>
<tr>
<td>10^{-8}</td>
<td>1101</td>
<td>2/0</td>
<td>7392</td>
<td>1062</td>
<td>1080</td>
<td>2513779</td>
<td>1.72e-12</td>
<td>1.68</td>
</tr>
<tr>
<td>DESI</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10^{-5}</td>
<td>14</td>
<td>2/0</td>
<td>2166</td>
<td>112</td>
<td>116</td>
<td>681725</td>
<td>–</td>
<td>2.67</td>
</tr>
<tr>
<td>10^{-6}</td>
<td>163</td>
<td>0/0</td>
<td>2796</td>
<td>161</td>
<td>162</td>
<td>921812</td>
<td>–</td>
<td>2.45</td>
</tr>
<tr>
<td>10^{-7}</td>
<td>247</td>
<td>1/0</td>
<td>3936</td>
<td>245</td>
<td>248</td>
<td>1357555</td>
<td>–</td>
<td>2.27</td>
</tr>
<tr>
<td>10^{-8}</td>
<td>375</td>
<td>2/0</td>
<td>5697</td>
<td>371</td>
<td>376</td>
<td>2022189</td>
<td>–</td>
<td>2.16</td>
</tr>
<tr>
<td>10^{-9}</td>
<td>576</td>
<td>2/0</td>
<td>8419</td>
<td>568</td>
<td>576</td>
<td>3055321</td>
<td>–</td>
<td>2.08</td>
</tr>
<tr>
<td>10^{-10}</td>
<td>896</td>
<td>1/0</td>
<td>12650</td>
<td>880</td>
<td>888</td>
<td>4681604</td>
<td>–</td>
<td>2.01</td>
</tr>
<tr>
<td>ESIRK</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10^{-3}</td>
<td>156</td>
<td>1/0</td>
<td>752</td>
<td>152</td>
<td>155</td>
<td>343949</td>
<td>3.08e-09</td>
<td>1.19</td>
</tr>
<tr>
<td>10^{-4}</td>
<td>220</td>
<td>1/0</td>
<td>1012</td>
<td>159</td>
<td>194</td>
<td>470783</td>
<td>1.11e-09</td>
<td>1.14</td>
</tr>
<tr>
<td>10^{-5}</td>
<td>319</td>
<td>0/0</td>
<td>1428</td>
<td>192</td>
<td>239</td>
<td>667322</td>
<td>2.86e-10</td>
<td>1.11</td>
</tr>
<tr>
<td>10^{-6}</td>
<td>481</td>
<td>0/0</td>
<td>2116</td>
<td>253</td>
<td>315</td>
<td>991362</td>
<td>6.05e-11</td>
<td>1.10</td>
</tr>
<tr>
<td>10^{-7}</td>
<td>741</td>
<td>2/0</td>
<td>3216</td>
<td>341</td>
<td>414</td>
<td>1511945</td>
<td>1.20e-11</td>
<td>1.08</td>
</tr>
<tr>
<td>10^{-8}</td>
<td>1148</td>
<td>2/0</td>
<td>4928</td>
<td>450</td>
<td>541</td>
<td>2319046</td>
<td>2.38e-12</td>
<td>1.07</td>
</tr>
<tr>
<td>EDESI</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10^{-5}</td>
<td>177</td>
<td>2/1</td>
<td>2158</td>
<td>159</td>
<td>174</td>
<td>796219</td>
<td>6.11e-10</td>
<td>2.01</td>
</tr>
<tr>
<td>10^{-6}</td>
<td>199</td>
<td>3/0</td>
<td>1864</td>
<td>193</td>
<td>201</td>
<td>845171</td>
<td>6.68e-11</td>
<td>1.53</td>
</tr>
<tr>
<td>10^{-7}</td>
<td>289</td>
<td>2/0</td>
<td>2350</td>
<td>277</td>
<td>288</td>
<td>1160511</td>
<td>7.69e-12</td>
<td>1.34</td>
</tr>
<tr>
<td>10^{-8}</td>
<td>439</td>
<td>2/0</td>
<td>3380</td>
<td>419</td>
<td>432</td>
<td>1724404</td>
<td>1.27e-12</td>
<td>1.28</td>
</tr>
<tr>
<td>10^{-9}</td>
<td>676</td>
<td>2/0</td>
<td>4975</td>
<td>625</td>
<td>648</td>
<td>2609413</td>
<td>1.82e-13</td>
<td>1.22</td>
</tr>
<tr>
<td>10^{-10}</td>
<td>1052</td>
<td>2/0</td>
<td>7552</td>
<td>865</td>
<td>898</td>
<td>4014523</td>
<td>3.04e-14</td>
<td>1.19</td>
</tr>
</tbody>
</table>
### Table 5.23: Numerical results for Van der Pol by testing order-4 methods

<table>
<thead>
<tr>
<th>tol</th>
<th>nsteps</th>
<th>rej(err/newt)</th>
<th>nfcn</th>
<th>njac</th>
<th>lu</th>
<th>flops</th>
<th>maxerr</th>
<th>niter</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>SIRK</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$10^{-3}$</td>
<td>481</td>
<td>15/131</td>
<td>6968</td>
<td>428</td>
<td>598</td>
<td>1174894</td>
<td>6.70e-04</td>
<td>3.51</td>
</tr>
<tr>
<td>$10^{-4}$</td>
<td>499</td>
<td>16/53</td>
<td>9388</td>
<td>411</td>
<td>510</td>
<td>1396930</td>
<td>3.30e-05</td>
<td>4.56</td>
</tr>
<tr>
<td>$10^{-5}$</td>
<td>676</td>
<td>16/75</td>
<td>12036</td>
<td>530</td>
<td>658</td>
<td>1821405</td>
<td>5.22e-05</td>
<td>4.35</td>
</tr>
<tr>
<td>$10^{-6}$</td>
<td>1043</td>
<td>13/0</td>
<td>10304</td>
<td>801</td>
<td>858</td>
<td>1943895</td>
<td>9.46e-06</td>
<td>2.44</td>
</tr>
<tr>
<td>$10^{-7}$</td>
<td>1602</td>
<td>10/0</td>
<td>13708</td>
<td>1192</td>
<td>1254</td>
<td>2768651</td>
<td>1.49e-06</td>
<td>2.13</td>
</tr>
<tr>
<td>$10^{-8}$</td>
<td>2516</td>
<td>8/0</td>
<td>19460</td>
<td>1823</td>
<td>1896</td>
<td>4137767</td>
<td>1.98e-07</td>
<td>1.93</td>
</tr>
<tr>
<td><strong>DESI</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$10^{-3}$</td>
<td>132</td>
<td>22/32</td>
<td>3517</td>
<td>109</td>
<td>180</td>
<td>641599</td>
<td>-</td>
<td>3.26</td>
</tr>
<tr>
<td>$10^{-4}$</td>
<td>160</td>
<td>15/29</td>
<td>4654</td>
<td>124</td>
<td>192</td>
<td>795349</td>
<td>-</td>
<td>3.80</td>
</tr>
<tr>
<td>$10^{-5}$</td>
<td>239</td>
<td>29/7</td>
<td>5839</td>
<td>185</td>
<td>248</td>
<td>1081825</td>
<td>-</td>
<td>3.11</td>
</tr>
<tr>
<td>$10^{-6}$</td>
<td>748</td>
<td>279/2</td>
<td>12996</td>
<td>262</td>
<td>877</td>
<td>3131390</td>
<td>-</td>
<td>1.81</td>
</tr>
<tr>
<td>$10^{-7}$</td>
<td>606</td>
<td>66/1</td>
<td>11010</td>
<td>383</td>
<td>542</td>
<td>2302226</td>
<td>-</td>
<td>2.34</td>
</tr>
<tr>
<td>$10^{-8}$</td>
<td>810</td>
<td>27/0</td>
<td>13699</td>
<td>572</td>
<td>660</td>
<td>2861847</td>
<td>-</td>
<td>2.34</td>
</tr>
<tr>
<td><strong>ESIRK</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$10^{-3}$</td>
<td>330</td>
<td>22/0</td>
<td>2140</td>
<td>264</td>
<td>328</td>
<td>552346</td>
<td>8.98e-04</td>
<td>1.52</td>
</tr>
<tr>
<td>$10^{-4}$</td>
<td>480</td>
<td>17/0</td>
<td>2632</td>
<td>307</td>
<td>437</td>
<td>738976</td>
<td>2.66e-04</td>
<td>1.32</td>
</tr>
<tr>
<td>$10^{-5}$</td>
<td>713</td>
<td>11/0</td>
<td>3624</td>
<td>569</td>
<td>629</td>
<td>1051341</td>
<td>3.87e-05</td>
<td>1.25</td>
</tr>
<tr>
<td>$10^{-6}$</td>
<td>1092</td>
<td>12/0</td>
<td>5112</td>
<td>849</td>
<td>954</td>
<td>1562180</td>
<td>4.40e-06</td>
<td>1.16</td>
</tr>
<tr>
<td>$10^{-7}$</td>
<td>1693</td>
<td>7/0</td>
<td>7580</td>
<td>1203</td>
<td>1467</td>
<td>2378763</td>
<td>1.11e-06</td>
<td>1.11</td>
</tr>
<tr>
<td>$10^{-8}$</td>
<td>2645</td>
<td>6/0</td>
<td>11508</td>
<td>1731</td>
<td>2262</td>
<td>3676047</td>
<td>1.73e-07</td>
<td>1.08</td>
</tr>
<tr>
<td><strong>EDESI</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$10^{-3}$</td>
<td>260</td>
<td>39/6</td>
<td>3292</td>
<td>207</td>
<td>291</td>
<td>832908</td>
<td>1.28e-04</td>
<td>1.83</td>
</tr>
<tr>
<td>$10^{-4}$</td>
<td>330</td>
<td>37/4</td>
<td>3807</td>
<td>256</td>
<td>348</td>
<td>989276</td>
<td>5.48e-06</td>
<td>1.73</td>
</tr>
<tr>
<td>$10^{-5}$</td>
<td>387</td>
<td>26/0</td>
<td>3783</td>
<td>290</td>
<td>369</td>
<td>1050314</td>
<td>1.14e-06</td>
<td>1.52</td>
</tr>
<tr>
<td>$10^{-6}$</td>
<td>524</td>
<td>20/0</td>
<td>4577</td>
<td>387</td>
<td>466</td>
<td>1337397</td>
<td>3.91e-07</td>
<td>1.40</td>
</tr>
<tr>
<td>$10^{-7}$</td>
<td>747</td>
<td>12/0</td>
<td>6198</td>
<td>543</td>
<td>619</td>
<td>1845495</td>
<td>5.61e-08</td>
<td>1.36</td>
</tr>
<tr>
<td>$10^{-8}$</td>
<td>1093</td>
<td>10/0</td>
<td>8883</td>
<td>774</td>
<td>905</td>
<td>2671743</td>
<td>1.60e-08</td>
<td>1.34</td>
</tr>
</tbody>
</table>
Appendix A

Mathematica and Matlab programs

As we mentioned in the body of this thesis, some investigations have been carried out using Mathematica and Matlab programs. In this appendix, we present three sections consisting of some of these programs:

(1) Transformation matrix $T$ and the factorization $T = UL$;

(2) The study of stability properties;

(3) The study of local truncation error.

After the subroutines themselves, we discuss the applications of the programs.

A.1 Transformation matrix $T = WV$ for ESIRK methods

The transformation matrices $T$ for ESIRK methods are based on the modified Vandermonde matrices $W$ mentioned in section 4.2 and matrices $V$ mentioned in Theorem 4.16. The Mathematica subroutines "Verd" is used to find the modified
Vandermonde matrix for given abscissae and "Twv" is used to find the transformation matrix $T$ for given abscissae and the eigenvalue $\lambda$.

Input[22]: ** subroutine for modified Vandermonde matrix  
Verd[c_] :=  
Module[{s,W},  
s=Length[c];  
W= {};  
Do[W=Join[W,{Insert[Table[c[[i]]~k/k!,{k,1,s-1}],1,1}]],{i,1,s}];  
Return[W];];

Input[23]:  
Verd[{0,1/2,1}]  
Output[23]:

\[
\{{1,0,0},\{1,\frac{1}{2},\frac{1}{8}\},\{1,1,\frac{1}{2}\}\}
\]

Input[24]: ** subroutine for the transformation matrix $T$  
Twv[c_,lam_] :=  
Module[{s,mcz,prod,liss,facts,tt,beta,  
lis,arr,lon,res,beliss,t,fot,Voo,W,V,T},  
s=Length[c];  
mcz=z-c;  
prod=1;  
Do[prod=prod mcz[[i]],{i,1,s}];  
prod=Expand[prod];  
liss=CoefficientList[prod,z];  
facts=Table[z~j((s-j)!)/(s!){j,0,s}];  
 tt=Reverse[liss].facts;  
 beta=CoefficientList[tt,z];  ** form beta  
lis=Rest[beta];  
arr={};lon={};res={};
A.1. TRANSFORMATION MATRIX $T = W V$ FOR ESIRK METHODS

Do[arr=Join[arr,{Table[lam^-i,{i,j,0,-1]}]},{j,1,s-1}];
beliss=Insert[lis,1,1];
t=Length[beliss];
Do[lon=Join[lon,{Take[beliss,j]}]},{j,2,t}];
Do[res=Join[res,{arr[[k]].lon[[k]]]}]},{k,1,s-1}];
fot=Append[Reverse[res],1];
Voo={};W={};
Do[Voo=Join[Voo,1/k! D[fot,{lam,k}]]]},{k,s-1,1,-1}];
hes=Join[Voo,fot];
V=Transpose[hes]; ** form matrix V
W=verd[c]; ** call subroutine verd
T=Simplify[W.V]; ** form T
Return[T];
]

Input[25]:
Tmatrix=Twv[{0,1/2,1},lam]
Output[25]:

$$\begin{align*}
\{ & \frac{1}{2} + 2 \text{lam}, \frac{1}{12} - \frac{\text{lam}}{2} + \text{lam}^2 \\
& 1, 2 \text{lam}, -\frac{1}{24} + \text{lam}^2 \\
& 1, \frac{1}{2} + 2 \text{lam}, \frac{1}{12} + \frac{\text{lam}}{2} + \text{lam}^2 \} \\
\end{align*}$$

Input[26]:
lam=1grt[3];
Tmatrix
Output[26.]:

$$\begin{align*}
\{ & 1, 0.3717330430169179988, 0.0553796971509877852 \\
& 1, 0.8717330430169179988, 0.1483129579052172849 \\
& 1, 1.3717330430169179988, 0.4912462186594467846 \}
\end{align*}$$
Once we have the transformation matrix $T$, it is proposed to use (3.8) in section 3.1 to do the Newton iteration. We need to factorize $T$ to be the multiplication of a unit upper triangular $U$ and a lower triangular matrix $L$. Below the matlab program "ul" is used to factorize a matrix "A" to be a unit upper triangular matrix "u" and a lower triangular matrix "l". We can derive $M = \lambda(U^{-1}AU)^{-1}$ once the transformation matrix $U$ is obtained.

%% This is a program to factorize a square matrix A into
%% A=ul, u is a unit upper triangular matrix, l is a lower
%% triangular matrix

function [u,l]=ul(A)
p=[];q=[];r=[];t=[];
[s,s]=size(A);
for i=1:s,
    u(i,i)=1;
end;
l(s,:)=A(s,:);
for i=1:s-1,
    u(i,s)=A(i,s)/l(s,s);
end;

for i=s-1:-1:2,
    for k=s:-1:i+1,
        p=[p,l(k,i)*u(i,k)];
    end;
l(i,i)=A(i,i)-sum(p);
end;

for j=i-1:-1:1
    for k=s:-1:i+1,
        q=[q,l(k,j)*u(i,k)];
        r=[r,l(k,i)*u(j,k)];
end;
end;

248
A.2 Stability function $R(z, r)$

In section 4.6, a systematic changing stepsize pattern $(h, rh, h, rh, \ldots)$ has been studied. It turns out that the stability function for this two steps depends on the choice of the abscissae and the stepsize changing ratio $r$. We give some Mathematica programs which we used to study the stability properties for ESIRK methods. The subroutine "lgrt" gives the eigenvalues $\lambda$ of the order $2, 3, \ldots, 6$ and $8$ A-stable ESIRK methods.

Input[1]: ** subroutine of the eigenvalues for order
lgrt[s_] :=
Module[{ans, x},
ans = NSolve[LaguerreL[s, x] == 0, x, 20]; ** 2 - 6 and 8 A-stable
If[s == 1 || s == 2, lam = 1/ans[[s, 1, 2]], ** ESIRK methods
If[s == 3 || s == 4, lam = 1/ans[[2, 1, 2]],
If[s == 5, lam = 1/ans[[3, 1, 2]],
lam = 1/ans[[s - 3, 1, 2]]]]];
Return[lam];
]

Input[2]: ** input the eigenvalue of order 4 method
lgrt[4]
The subroutine "alphas" produces the perturbations $\alpha_0 = 1, \alpha_1, \alpha_2, \ldots, \alpha_s$ for given abscissae. The subroutine "stabfun" can yield the numerator and denominator of the stability function $R(r, z)$ for given abscissae and stepsize changing ratio.

Input[3]: ** subroutine for finding the perturbations
alphas[c_] :=
  Module[{s, lam, betseries, x, poly, clist, facts, alphseries},
    s = Length[c];
    lam = lgrt[s]; ** call subroutine lgrt
    poly = 1;
    Do[poly = poly(x - c[[i]])/i, {i, s}];
    poly = Expand[poly];
    clist = CoefficientList[poly, x];
    facts = Table[x^(s-i) Factorial[i], {i, 0, s}];
    betseries = Expand[clist.facts];
    alphseries = Normal[Series[(1 - x lam)^s/betseries, {x, 0, s}]];
    Return[CoefficientList[alphseries, x]]
  ];

Input[4]:
alphas[{0, 1/2, 1}]
Output[4]:

\{1, -0.8075995645253769982, 0.0828057581196300222, 0.0258970846506330721\}

Input[5]: ** Subroutine for finding the stability function $R(z, r)$
A.2. STABILITY FUNCTION \( R(Z, R) \)

\[
\text{stabfun}[c_-, r_-] := \\
\text{Module}[\{\text{alphseries}, \text{alfas}, \text{num}, \text{den}, \text{zpowers}, \text{alphaR}, s, \text{lam}\}, \\
s = \text{Length}[c]; \\
\text{alfas} = \text{alphas}[c]; \quad ** \text{call subroutine alphas} \\
\text{lam} = \text{lgrt}[c]; \quad ** \text{call subroutine lgrt} \\
\text{zpowers} = \text{Table}[z^i, \{i, 0, s\}]; \\
\text{alphseries} = \text{Expand}[\text{zpowers.alfas}]; \\
\text{alphaR} = \text{alphseries}/(r^z); \\
\text{den} = (1 - \text{lam} z)^s; \\
\text{num} = \text{Normal}[\text{Series}[\text{den alphaR Exp[z]/alphseries}, \{z, 0, s\}]]; \\
\text{Return}[\{\text{num}, \text{den}\}];]
\]

**Input [6]:**

\( \text{stabfun}[\{0, 1/2, 1\}, r] \)

**Output [6]:**

\[
\{1 + (0.5 - 0.8075995645253769982r)z + (0.08333333333333333 - 0.4037997822626884991r + 0.0828057581196300222r^2)z^2 \\
+ (-0.067299963710448083r + 0.0414028790598150111r^2 + 0.0258970846506330721r^3)z^3, (1 - 0.435866521505084589994z)^3\}
\]

From the subroutine "twostep", we can obtain the two-step stability function \( \hat{R}_2(z, r) = \hat{R}(z, r)\hat{R}(rz, \frac{1}{r}) \), then using the E-polynomial scheme, we will be able to investigate the stability properties for ESIRK methods under the systematic stepsize changing pattern mentioned in 4.6. See Input[9]. We note that, from the Input[9], the ESIRK method \( (0, \frac{1}{2}, 1) \) has the E-polynomial with degree 12. Even for the case order 3, the expansion of the E-polynomial is very large (see Output[9]). Because we need to study the coefficients of the E-polynomial in order to decide the \( r \)-interval for A-stability, one way for simplifying the tasks is to divide the polynomial by \( r^{4n}, n = 1, 2, \ldots \) and to examine the sign of the coefficients.
**APPENDIX A. MATHEMATICA AND MATLAB PROGRAMS**

Input[7]: ** subroutine for two-step (h, rh) stability
twostep[c_,r_]:=
       Module[{lam,first,second}, ** function
          lam=lgtr[c];
          first=stabfun[c,r]; ** call subroutine stabfun
          second=stabfun[c,1/r]/.z->(z r); ** form R(rz, 1/r)
          Return[first second];];

Input[8]:
sta=twostep[{0,1/2,1},r];

Input[9]:
top=sta[[1]]/.z->(I y); bot=sta[[2]]/.z->(I y);
topc=sta[[1]]/.z->(-I y); botc=sta[[2]]/.z->(-I y);
epoly=Expand[botc bot -topc top] ** form E-polynomial

Output[9]:

\[
\begin{align*}
0.012096124964028068 y^4 + 0.02589708465063307 r y^4 + \\
0.02760191937321001 r^2 y^4 + 0.02589708465063307 r^3 y^4 + \\
0.01209612496402807 r^4 y^4 + \\
0.001250215736831654 y^6 + 0.00001366265996885 r y^6 + \\
0.01943836778356521 r^2 y^6 + 0.03634897877443039 r^3 y^6 + \\
0.01943836778356521 r^4 y^6 + 0.00001366265996885 r^5 y^6 - \\
0.00125021573683165 r^6 y^6 + \\
0.00039398368105033001 y^8 + 1.138554997404210^{-6} r y^8 + \\
0.000037180751064019 r^2 y^8 + 0.00408623181851926 r^3 y^8 + \\
0.0088888346607671281 r^4 y^8 + 0.00408623181851926 r^5 y^8 + \\
0.000037180751064019 r^6 y^8 + 1.138554997404210^{-6} r^7 y^8 - \\
0.00039398368105033001 r^8 y^8 + \\
4.6573541208476046 10^{-6} y^{10} - 0.00001489185921932677 r y^{10} - 
\end{align*}
\]

252
A.3 LOCAL TRUNCATION ERROR

0.00022614898940470748 r^2 y^{10} + 0.0003873182866630209 r^3 y^{10} +
0.0006020220852719495 r^4 y^{10} - 2.4213713945876 \times 10^{-6} r^5 y^{10} +
0.0006020220852719495 r^6 y^{10} + 0.0003873182866630209 r^7 y^{10} -
0.00022614898940470748 r^8 y^{10} - 0.000014891859219332677 r^9 y^{10} -
4.6573541208476046 \times 10^{-6} r^{10} y^{10} -
3.0376057963436044 \times 10^{-6} r^2 y^{12} + 5.975259707035863 \times 10^{-6} r^3 y^{12} +
0.000021162466463939749 r^4 y^{12} + 0.000017729164249061437 r^5 y^{12} -
0.000012741912251141142 r^6 y^{12} + 0.000017729164249061437 r^7 y^{12} +
0.000021162466463939749 r^8 y^{12} - 5.975259707035863 \times 10^{-6} r^9 y^{12} -
3.0376057963436044 \times 10^{-6} r^{10} y^{12}

A.3 Local truncation error

In section 4.6.1, we have used two approaches to investigate the local truncation error for ESIRK methods with variable stepsize. In this section, we propose to list the programs used for studying the truncation error and their applications.

A.3.1 Error ratio $\varepsilon(r, c)$

The error ratio $\varepsilon(r, c)$ of variable stepsize against constant stepsize can be obtained by using subroutines "errorate". Some of the plots $\varepsilon(r)$ for various choices of the abscissae have been given in section 4.6.1.

Input[10]:** subroutine modified from alphas
alphas1[c_]:= 
Module[{s,lam,betseries,poly,clist,facts,alphseries},
s=Length[c];
lam=lgrt[s];
poly=1;
Do[poly=poly(z-c[[i]])/i,{i,s}];
poly=Expand[poly];
clist=CoefficientList[poly,z];
facts=Table[z^(-s-i) Factorial[i],{i,0,s}];
betseries=Expand[clist.facts];
alphseries=Normal[Series[(1-z lam)^s/betseries,{x,0,s}]];
Return[alphseries]
];

Input[11]:** subroutine for the error ratio C_2(r)/(C(1)*(1/2+r/2)^*(s+1))
errorate[c_]:= 
Module[{lambda, alpha, Ralpha, coalpha, Ppoly, RQpoly,
coPoly, Qpoly, RQpoly, coQpoly, Bstab, coBstab, Const, coConst, ratio},
s=Length[c];
lambda=lgFt[s]; ** call subroutine lgFt
alpha=alphasl[c]; ** call subroutine alphasl
Ralpha=alpha/.z->(r z);
coalpha=alpha/.z->((1+r)/2 z);
Ppoly=(1-lambda z)^s;
RQpoly=(1-lambda r z)^s;
coPoly=(1-lambda (1+r)/2 z)^s;
Qpoly=Normal[Series[Ppoly Ralpha Exp[z]/alpha,{z,0,s}]];
RQpoly=Normal[Series[RQpoly alpha Exp[r z]/Ralpha,
{z,0,s}]];
coQpoly=Normal[Series[coPoly coalpha Exp[(1+r)/2 z]/coalpha,
{z,0,s}]];
Bstab=Normal[Series[(Qpoly/Ppoly) (RQpoly/RQpoly)*
Exp[-z-r z],{z,0,s+1}]];
coBstab=Normal[Series[(coPoly/coPoly) (coQpoly/coPoly)*
Exp[-z-r z],{z,0,s+1}]];
Const=Simplify[Part[CoefficientList[Bstab,z],s+2]];
coConst=Simplify[Part[CoefficientList[coBstab,z],s+2]];
ratio=Const/coConst;
Return[ratio];
A.3. LOCAL TRUNCATION ERROR

];

Input[12]:
errorate[\{0,1/2,1\}]

Output[12]:

\[-0.025897084650633072 - 0.025897084650633072 r^4 \]
\[-0.003237136 - 0.012948542 r - 0.019422813 r^2 - 0.012948542 r^3 - 0.003237136 r^4 \]

A.3.2 Normalized error constant $\hat{C}(r, c)$

In subsection 4.6.1, another approach for studying the local error for ESIRK methods with variable stepsize is to use the normalized error constant $\hat{C}(r, c)$. The subroutine "nec" is to obtain $\hat{C}(r, c)$ for given abscissae. We note that when $r = 1$, the error constant obtained is the same as the corresponding SIRK method, it is independent of the abscissae. See Input[15] and Input[16]. The subroutines "Secder" and "mhash3" are used to find the maximum (or minimum) of the second derivative $\frac{d^2}{dx^2} \varepsilon(r, c_1)|_{r=1}$ mentioned in subsection 4.6.1.

Input[13]: ** subroutine for normalized error constant

nec[c_, r_] :=

Module[{lambda, alpha, Ralpha, Ppoly, RPpoly, Qpoly, RQpoly, Bstab, Const, Norconst},
    s = Length[c];
    lambda = lgrt[s];
    alpha = alphas1[c];
    Ralpha = alpha/.z->(r z);
    Ppoly = Normal[Series[(1-lambda z)^s,\{z,0,s\}]];
    RPpoly = Normal[Series[(1-lambda r z)^s,\{2,0,s\}]];
    Qpoly = Normal[Series[Ppoly Ralpha Exp[z]/alpha,\{z,0,s\}]];
    RQpoly = Normal[Series[RPpoly alpha Exp[r z]/Ralpha,\{z,0,s\}]];
    255


APPENDIX A. MATHEMATICA AND MATLAB PROGRAMS

Bstab=Normal[Series[(Qpoly/Ppoly) (RQpoly/RPpoly)*
Exp[-z-r z]},{z,0,s+2}];
Const=Simplify[Part[CoefficientList[Bstab,z],s+2]];
Norconst=Const/(1+r^4); ** Norconst is the normalized
Return[Norconst];] ; ** error constant for two steps

Input[14]:
nec[{0,1/2,1,1},r]

Output[14]:

\[
\frac{-0.0258970846506330721 - 0.02589708465063307 r^4}{1 + r^4}
\]

Input[15]: ** error constant for order 3 ESIRK method
nec[{0,1/2,1,1},1]
Output[15]: ** error constant is independent of abscissae

\[-0.02589708465063307\]

Input[16]:
nec[{0,1/5,1,1},1]
Output[16]:

\[-0.02589708465063307\]

Input[17]: ** subroutine for second derivative of the normalized error
Secder[c_] :=
    Module[{result}, **constant
        result=D[nec[c,r],{r,2}]/.r->1; ** call subroutine nec
        Return[result];] ; ** derivative d^2/dr^2 nec[c,r] | r=1

Input[18]: ** subroutine for searching the maximum of the second derivative

256
A.3. LOCAL TRUNCATION ERROR

\texttt{minsort3[c_] := \texttt{** for order 3 methods}}

\texttt{Module[\{n, lambda, var, m, fir, sml, ans\},}

\texttt{n = Length[c];}

\texttt{lambda = lgrt[n];}

\texttt{var = \{};

\texttt{Do[Do[var = Join[var, \{\{c[[i]], c[[j]], 1\}\}],}

\texttt{\{j, i + 1, n\}], \{i, 1, n\};}

\texttt{m = Length[var];}

\texttt{fir = Table[Secder[c], \{i, 1, m\}];}

\texttt{sml = Max[fir]; ** find the maximum since the error constant}

\texttt{Do[If[Secder[c] == sml, ** for s=3 is negative}

\texttt{ans = var[[k]], \{k, 1, m\};}

\texttt{Return[ans];]}

\texttt{Input[19]: ** form an array \([-1, 0.9]\) with stepsize 0.1}

\texttt{arr = Table[x, \{x, -1, 0.9, 0.1\}];}

\texttt{Input[20]:}

\texttt{minsort3[arr]}

\texttt{Output[19]:}

\texttt{\{-1, -0.9\}}
Bibliography


[65] A.Wolfram, A study of Rosenbrock processes with respective to order conditions and stiff stability, Department of Computer Science, Chalmers University of Technology, Göteborg.