UC Merced
UC Merced Electronic Theses and Dissertations

Title
Development, analysis, and implementation of exponential integrators

Permalink
https://escholarship.org/uc/item/4p84g9t3

Author
Rainwater, Greg

Publication Date
2017

Peer reviewed|Thesis/dissertation

eScholarship.org Powered by the California Digital Library
University of California

Development, analysis, and implementation of exponential integrators

by

Greg Rainwater

A dissertation submitted in partial satisfaction of the
requirements for the degree of
Doctor of Philosophy

in

Applied Mathematics

in the

Graduate Division

of the

University of California, Merced

Committee in charge:

Professor Mayya Tokman, Chair
Professor François Blanchette
Professor Arnold Kim

Summer 2017

The dissertation of Greg Rainwater, titled Development, analysis, and implementation of exponential integrators, is approved:

Chair _____ Date _____
Professor Mayya Tokman

_____ Date _____
Professor François Blanchette

_____ Date _____
Professor Arnold Kim

University of California, Merced

Development, analysis, and implementation of exponential integrators

Chapter 2 © 2014 Elsevier
Chapter 3 © 2017 Elsevier
Chapter 4 © 2016 Elsevier
All other chapters © 2017 Greg Rainwater

Abstract

Development, analysis, and implementation of exponential integrators

by

Greg Rainwater

Doctor of Philosophy in Applied Mathematics

University of California, Merced

Professor Mayya Tokman, Chair

Exponential propagation iterative methods of Runge-Kutta type (EPIRK) provide a flexible framework to derive efficient exponential integrators for different types of ODE systems. Here we develop new classes of EPIRK methods depending on the properties of the equations to be solved. We derive the order conditions for the new classes of integrators and present new approaches to solving the order conditions where we optimize coefficients to construct more efficient schemes. Performance comparisons are conducted on a set of test problems to demonstrate the computational advantages of the new schemes themselves and the overall improvement in computational cost due to the recent additions to the EPIRK family.

Contents

Abstract **1**

Contents **ii**

List of Figures **iv**

List of Tables **vi**

Curriculum Vitae **1**

1 Introduction **3**

2 A new class of split EPIRK methods (sEPIRK) for semilinear systems of ODEs **6**
2.1 Introduction 6
2.2 Construction of a new class of split EPIRK (sEPIRK) methods 6
2.3 Construction of third-order schemes 9
2.4 Numerical results 21

3 Preconditioned Implicit-Exponential (IMEXP) Time Integrators for Stiff Differential Equations **33**
3.1 Introduction 33
3.2 Construction and analytical framework for analysis of the IMEXP methods. 35
3.3 IMEXP Runge–Kutta methods 38
3.4 Hybrid IMEXP 45
3.5 Numerical Experiments 51

4 A new approach to constructing efficient stiffly accurate EPIRK methods **59**
4.1 Introduction 59
4.2 Stiffly accurate EPIRK methods 60
4.3 Construction of new schemes 70
4.4 Numerical Experiments 82

5 Conclusion **94**

A Appendix **96**

Bibliography **99**

List of Figures

1.1 EPIRK Family Tree 5

2.1 Notation for the bicolored trees 9
2.2 Operations on bicolored trees 14
2.3 Choice of parameters and g_{ij} coefficients 18
2.4 Log-log plots of the error versus the time step size. For convenience the lines with slopes equal to three (dashed), four (dash-dotted), and five (dotted) are shown. 24
2.5 Precision diagrams. Note that the axes scales are different in each graph. To generate the graphs we used the step sizes $h = 0.02, 0.01, 0.005, 2.5e^{-3}, 1.25e^{-3}$ for Brusselator, & Gray-Scott problems, $h = 0.01, 0.005, 2.5e^{-3}, 1.25e^{-3}, 6.25e^{-4}$ for ADR & Burgers problems, and $h = 0.1, 0.05, 0.025, 0.0125, 6.25e^{-3}$ for Allen-Cahn problem. 27
2.6 Precision diagrams. Note that the axes scales are different in each graph. To generate the graphs we used the step sizes $h = 0.02, 0.01, 0.005, 2.5e^{-3}, 1.25e^{-3}$ for Brusselator, & Gray-Scott problems, $h = 0.01, 0.005, 2.5e^{-3}, 1.25e^{-3}, 6.25e^{-4}$ for ADR & Burgers problems, and $h = 0.1, 0.05, 0.025, 0.0125, 6.25e^{-3}$ for Allen-Cahn problem. 28
2.7 Precison diagrams and spectra 29
2.8 Precision diagrams. Note that the axes scales are different in each graph. To generate the graphs we used the step sizes $h = 0.02, 0.01, 0.005, 2.5e^{-3}, 1.25e^{-3}$ for the Brusselator, & Gray-Scott problems, $h = 0.01, 0.005, 2.5e^{-3}, 1.25e^{-3}, 6.25e^{-4}$ for the ADR & Burgers problems, and $h = 0.1, 0.05, 0.025, 0.0125, 6.25e^{-3}$ for the Allen-Cahn problem. 30
2.9 Precision diagrams. Note that the axes scales are different in each graph. To generate the graphs we used the step sizes $h = 0.02, 0.01, 0.005, 2.5e^{-3}, 1.25e^{-3}$ for the Brusselator, & Gray-Scott problems, $h = 0.01, 0.005, 2.5e^{-3}, 1.25e^{-3}, 6.25e^{-4}$ for the ADR & Burgers problems, and $h = 0.1, 0.05, 0.025, 0.0125, 6.25e^{-3}$ for the Allen-Cahn problem. 31
2.10 Precision diagrams. Note that the axes scales are different in each graph. To generate the graphs we used the step sizes $h = 0.02, 0.01, 0.005, 2.5e^{-3}, 1.25e^{-3}$ for the Brusselator, & Gray-Scott problems, $h = 0.01, 0.005, 2.5e^{-3}, 1.25e^{-3}, 6.25e^{-4}$ for the ADR & Burgers problems, and $h = 0.1, 0.05, 0.025, 0.0125, 6.25e^{-3}$ for the Allen-Cahn problem. 32

3.1 Plots of the error vs. time step size. For convenience a line with slope two (dotted) is shown. 54

3.2 2D Schnakenberg problem ($n_x = n_y = 128$): CPU execution time versus error for constant time step 57
3.3 2D Allen-Cahn problem ($n_x = n_y = 150$): CPU execution time versus error for constant time step 58

4.1 g_{21} and g_{31} 80
4.2 Log-log plots of error vs. time step size. For convenience the lines with slopes equal to three (dashed), four (dash-dotted), and five (dotted) are shown. 85
4.3 CPU execution time versus error for constant time step experiments 87
4.4 Log-log plots of error vs. time step size for problems with non-homogeneous boundary conditions. For convenience the lines with slopes equal to three (dashed), four (dash-dotted), and five (dotted) are shown. 90
4.5 CPU execution time versus error for variable time step experiments 93

List of Tables

2.1 Trees in $\mathcal{T}$, corresponding order of trees $|\tau|$, symmetry $\sigma(\tau)$, density $\gamma(\tau)$ and elementary differentials . . . 11
2.2 Expansion terms and coefficients according to Corollary 2 . . . 15
2.3 Order conditions . . . 16
2.4 Explicit formulas for α, β, γ, and a_{22} . . . 17
2.5 Coefficients for sEPIRK3. For the implementation the free parameter g_{22} is chosen to be equal to g_{32} . . . 19
2.6 Coefficients for sEPIRK3b. For the implementation the free parameter g_{22} is chosen to be equal to g_{32} . . . 19
2.7 Coefficients for EPIRK methods . . . 23
2.8 Approximate values of g_{ij} coefficients for the EPIRK methods being compared. . . . 29

3.1 Functions $R_1(Z)$ and $R_{1i}(Z)$ and the corresponding exponential Runge-Kutta stiff order conditions for the methods of orders 1 and 2. . . . 43
3.2 Largest time step sizes taken for each of the problems and each of the methods. . . . 53
3.3 Approximated CPU time for `HimExp2J` as a percentage of the CPU time for `2-sBDF` given a prescribed accuracy. . . . 56

4.1 Stiff order conditions for EPIRK methods. Note that Z and K are arbitrary square matrices and Ψ_i is given by (4.27) . . . 70
4.2 Simplified stiff order conditions. Note that Z and K are arbitrary square matrices, Ψ_i is given by (4.27),and $\psi_{3,i}(Z) = \sum_{j=2}^{i-1} a_{ij}(Z)\frac{c_j^2}{2} - c_i^3\varphi_3(c_iZ)$ . . . 77
4.3 Stiff order conditions for EPIRK methods and EXPRB. Note that Z and K are arbitrary square matrices, Ψ_i is given by (4.27),and $\psi_{3i}(Z) = \sum_{j=2}^{i-1} a_{ij}(Z)\frac{c_j^2}{2} - c_i^3\varphi_3(c_iZ)$ . . . 79
4.4 *EXPRB53s3* and grouping of terms for mixed exponential-adaptive-Krylov . . . 81
4.5 Comparison of CPU times for *EPIRK4s3A*-Vert with *EPIRK4s3A*-Horz and *EXPRB53s3*-Vert with *EPIRK5s3*-Horz (Cost of VertMethod/Cost of HorzMethod) . . . 86
4.6 Cost of horizontal and vertical implementations in comparison to mixed implementation 88
4.7 Approximate CPU times for a given accuracy . . . 91
4.8 Approximate order of methods for problems with non-homogeneous boundary conditions 92

Acknowledgments

The work in Chapters 2, 3 and 4 were supported by a grant from the National Science Foundation, Computational Mathematics Program, under Grant No. 1115978.

The material in Chapter 2 is based on Journal of Computational Physics 269 (2014) 40–60 with permission to be reprinted. The co-author listed in this publication directed and supervised research which forms the basis for the dissertation.

The material in Chapter 3 is based on Journal of Computational Physics 335 (2017) 846–864 with permission to be reprinted, co-authored with Vu Thai Luan and Mayya Tokman. Mayya Tokman directed and supervised research which forms the basis for the dissertation.

The material in Chapter 4 is based on Journal of Computational Physics 323 (2016) 283–309 with permission to be reprinted. The co-author listed in this publication directed and supervised research which forms the basis for the dissertation.

Greg Rainwater

grainwater@outlook.com • 925-209-3021

Research Interests

Numerical and functional analysis; scientific computing; exponential integrators; numerical methods for evolution equations, climate modeling

Education

2011-Present **Ph.D.**, Applied Mathematics, **University of California, Merced**, Merced, California
- Thesis: "Development, analysis, and implementation of exponential integrators"
- Advisor: Professor Mayya Tokman

2008-2011 **M.S.**, Mathematics, **Portland State University**, Portland, Oregon
- Thesis: "Poincaré inequality and its applications to partial differential equations"
- Advisor: Professor Dacian Daescu

2006-2007 **B.S.**, Mathematics, **California State University-Monterey Bay**, Seaside, California

Publications

- S. Gaudreault, G. Rainwater, M. Tokman, J. Pudykiewicz, C. Girard. *A scalable adaptive-Krylov exponential solver for the shallow-water equations in the GEM model.* In preparation.
- M. Tokman, G. Rainwater. *Designing efficient exponential integrators with EPIRK framework.* Submitted to ICNAAMs Proceedings (http://icnaam.org)
- V. Luan, M. Tokman, G. Rainwater. *Preconditioned Implicit-Exponential Integrators (IMEXP) for Stiff PDEs.* Journal of Computational Physics. 335:846–864, April 2017.
- G. Rainwater and M. Tokman. *A new approach to constructing efficient stiffly accurate exponential propagation iterative methods of Runge-Kutta type (EPIRK).* Journal of Computational Physics. 323:283–309, October 2016.
- M. Tokman and G. Rainwater. *Four classes of exponential propagation iterative methods of Runge-Kutta type (EPIRK).* Oberwolfach Reports. Nonlinear Evolution Equations: Analysis and Numerics, (14), 855–858, 2014.
- G. Rainwater and M. Tokman. *A new class of split exponential propagation iterative methods of Runge-Kutta type (sEPIRK) for semilinear systems of ODEs.* Journal of Computational Physics, 269:40–60, July 2014.

Academic Experience

2011-Present **Research Assistant**, **University of California, Merced**
- ◦ Developed new numerical algorithms for simulating time dynamics of complex systems
- ◦ Optimized, analyzed, and proved convergence of new methods
- ◦ Implemented new algorithms and demonstrated their performance on a suite of test problems
- ◦ Applied new method to problems in weather prediction and climate modelling

2011-2012 **Teaching Assistant**, **University of California, Merced**
- ◦ Taught discussion sections of integral and vector calculus courses

2008-2011 **Lecturer / Teaching Assistant**, **Portland State University**
- ◦ Taught remedial algebra and precalculus courses

2007-2007 **Supplemental Instructor**, **California State University-Monterey Bay**
- ◦ Statistics for natural and biological sciences course

2007-2007 **Supplemental Instructor**, **Hartnell College**
- Integral calculus course

2006-2007 **Instructional Student Assistant**, **California State University-Monterey Bay**
- Introductory statistics courses

Presentations / Talks

Jul. 2016 *Improving the efficiency of stiffly accurate exponential propagation iterative methods of Runge-Kutta type (EPIRK).* 2016 SIAM Annual Meeting, Boston, Massachusetts, USA

Aug. 2015 *A new approach to constructing efficient stiffly accurate EPIRK methods.* The 8th International Congress on Industrial and Applied Mathematics, Beijing, China

Oct. 2014 *On the similarities and differences between classes of exponential integrators.* 8th NAI Workshop Numerical Analysis of Evolution Equations, Innsbruck, Austria

Jul. 2013 *A new class of split exponential propagation iterative methods of Runge-Kutta type (sEPIRK) for semilinear systems of ODEs.* 2013 SIAM Annual Meeting, San Diego, California, USA

Awards

- Graduate Dean's Dissertation Fellowship (Spring 2017)
- School of Natural Sciences Dean's Distinguished Scholars Fellowship (Spring 2016)
- Applied Mathematics Summer Travel Award (2015, 2016)
- Applied Mathematics Summer Research Fellowship (2013, 2015)
- Graduate Division General Fellowship (Spring 2012)
- Applied Mathematics Enhancement Award (Fall 2011)

Computer Skills

- Operating Systems
  - Unix, Microsoft Windows
- Programming / Software / Databasing
  - Proficient in MATLAB, LaTeX, Mathematica
  - Basic knowledge of C++, Fortran, Java, SQL, PHP, MySQL

Chapter 1

Introduction

Over the past several decades exponential time integration emerged as an alternative technique to explicit and implicit methods for solving evolution equations. Given the autonomous initial value problem for a system of ODEs

$$y' = f(y), \quad y(t_0) = y_0 \tag{1.1}$$

where y, $f(y) \in \mathbf{R}^N$, exponential integrators are constructed based on the idea that numerical solution will involve computation of an exponential of either the full Jacobian $f'(y) \in \mathbf{R}^{N\times N}$ or a portion of it if a meaningful partitioning of $f(y)$ exists (e.g. if $f(y) = Ly + N(y)$ where L and N represent the linear and nonlinear part of $f(y)$ respectively). Original exponential methods date as far back as 1960's [10, 57, 36, 51]. However, these methods were deemed computationally infeasible for large scale applications and did not receive widespread attention until more efficient algorithms for approximating an exponential or another function of a large matrix were introduced. In [49, 55] Krylov projection-based methods were used to compute the exponential of a matrix and exponentially propagate solution of a Hamiltonian problem from chemical physics and the Schrodinger equation respectively. After Van der Vorst [70] showed that Krylov projection algorithm can be used to approximate a product of an arbitrary function of a matrix and a vector, a renewed interest in exponential integration ensued and a number of exponential propagation-based integrators have been proposed (e.g. [14, 15, 16, 24, 12, 66, 27, 33, 35], see also review [28]).

EPIRK framework has been developed to allow derivation of efficient integrators particularly for large scale stiff equations that arise from large ODE systems or spatial discretizations of PDEs. The general formulation of an s-stage EPIRK method for the initial value problem (1.1) is given by

$$\begin{aligned} Y_i &= y_n + a_{i1}\psi_{i1}(g_{i1}hA_{i1})hf(y_n) + \textstyle\sum_{j=2}^{i-1} a_{ij}\psi_{ij}(g_{ij}hA_{ij})h\Delta^{(j-1)}r(y_n), \quad i = 2, ..., s \\ y_{n+1} &= y_n + b_1\psi_{(s+1)1}(g_{(s+1)1}hA_{(s+1)1})hf(y_n) + \textstyle\sum_{j=2}^{s} b_j\psi_{(s+1)j}(g_{(s+1)j}hA_{ij})h\Delta^{(j-1)}r(y_n), \end{aligned} \tag{1.2}$$

where y_n is an approximation to the solution at some time t_n, $h = t_{n+1} - t_n$ is the time step and Δ denotes the forward difference operator constructed on stage-values $y_n, Y_2, ..., Y_s$. Different choices for functions $r(y)$, ψ_{ij} and matrices A_{ij} result in different classes of EPIRK methods [67] . For example, suppose $r(y) = f(y) - f(y_n) - f'(y_n)(y - y_n)$ is the remainder of the first order Taylor

expansion of $f(y)$, all matrices $A_{ij} = f'(y_n)$ are equal to the Jacobian evaluated at y_n and functions $\psi_{ij}(z) = \sum_{k=1}^{P} p_{ijk}\varphi_k(z)$ are linear combinations of exponential-like functions

$$\varphi_0(z) = e^z, \quad \varphi_k(z) = \int_0^1 e^{(1-\theta)z} \frac{\theta^{k-1}}{(k-1)!}\, d\theta, \quad k \geq 1. \tag{1.3}$$

These choices result in a class of general (unpartitioned) EPIRK methods which are appropriate for cases where no meaningful partitioning exists for the operator $f(y)$ and the problem's stiffness comes from the full Jacobian matrix $f'(y_n)$. Specific schemes can be constructed by deriving order conditions and solving them to determine coefficients a_{ij}, b_{ij} and p_{ijk}.

The most efficient general EPIRK schemes result from coupling construction of particular integrators with the choice of an algorithm to approximate the matrix functions vector products. Krylov projection based algorithms are the methods of choice if no useful information can be obtained on the structure or the spectrum of the matrix and if a matrix-free implementation is necessary due to the scale of the problem which does not permit explicit storage of the matrix so that only matrix-vector products are available. A particularly efficient adaptive Krylov algorithm has been developed in [50]. This method allows cost efficient computation of linear combinations of type

$$\varphi_0(A)b_0 + \varphi_1(A)b_1 + ... + \varphi_K(A)b_K \tag{1.4}$$

for a matrix A and vectors b_i, $i = 0, ..., K$. Flexibility of the EPIRK framework allows construction of exponential integrators that take advantage of the efficiency of the adaptive Krylov method and reduce both the number and the cost of the matrix function vector products evaluations.

In this thesis, we extend the family of EPIRK methods to four classes for different types of problems (Figure 1.1) and offer new approaches to constructing computationally advantageous schemes. We begin with split (or partitioned) EPIRK methods in Chapter 2. Split EPIRK methods (or sEPIRK) targets problems where $f(y)$ can be split into a stiff linear and a nonlinear parts and defined by choosing $A_{ij} = L$, $r(y) = N(y)$, and $\psi_{ij}(z) = \sum_{k=1}^{P} p_{ijk}\varphi_k(z)$. We demonstrate how bicolored trees-based B-series can be used to derive the order conditions for the new integrators. An algorithm is developed to solve the order conditions up to order three and several new schemes with desirable properties are proposed. While numerical experiments confirm computational advantages of split EPIRK compared to unpartitioned EPIRK schemes, sEPIRK methods cannot have an order higher than its number of stages. To overcome this order limitation we constructed schemes where $A_{i1} = L$ and $A_{ij} = L + \frac{DN}{dy}$ for $j > 1$. Methods of this type belong to a class we call hybrid EPIRK methods.

While these classes of EPIRK methods have been shown to offer computational advantages for different types of problems, neither of these categories address the following limitations of all exponential integrators compared to implicit methods. For many problems the main computational advantage of an implicit method comes from the availability of a very efficient preconditioner for solving the linear systems inside the integrator. No efficient preconditioning strategies currently exist for exponential integrators. Additionally, for some split problems the stiffness is not limited to the linear part Ly and the nonlinearity $N(y)$ can become stiff at least for a portion of time integration interval. Often implicit-explicit (IMEX) methods are used to solve the split problems [3, 60].

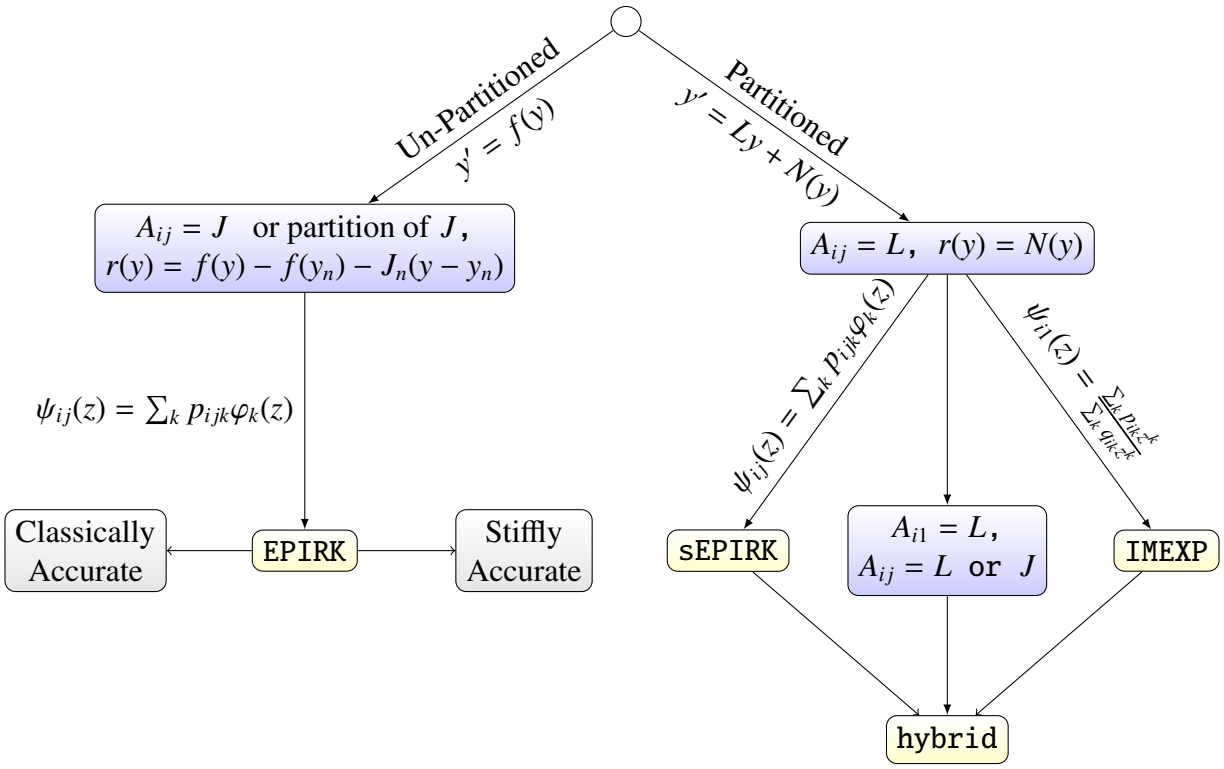


Figure 1.1: EPIRK Family Tree

These techniques treat the linear part implicitly and the nonlinearity explicitly. This approach will obviously encounter stability limitations for the problems where the nonlinear part can become stiff. In Chapter 3, we introduce a new category of the EPIRK methods, the implicit-exponential (IMEXP) integrators, designed to address these two challenges.

The unpartitioned EPIRK methods have been shown to perform well on stiff problems, even though all of the schemes proposed up to now have been derived using classical order conditions. In Chapter 4 we extend the stiff order conditions and the convergence theory developed for the exponential Rosenbrock methods to the EPIRK integrators. We derive stiff order conditions for the EPIRK methods and develop algorithms to solve them to obtain specific schemes. Moreover, we propose a new approach to constructing particularly efficient EPIRK integrators that are optimized to work with an adaptive Krylov algorithm. We use a set of numerical examples to illustrate the computational advantages that the newly constructed EPIRK methods offer compared to previously proposed exponential integrators.

A summary of the results of the thesis is given in the final chapter.

Chapter 2

A new class of split EPIRK methods (sEPIRK) for semilinear systems of ODEs

2.1 Introduction

In this chapter we propose a general framework to derive split EPIRK (sEPIRK) integrators and demonstrate that specific schemes can be constructed to achieve better performance for a given problem.

Previously the classical order conditions were proposed [4, 26] for a large class of exponential integrators for problems of type (2.11). However, the sEPIRK methods constitute a more general class of integrators and the classical order conditions derived in this chapter is a superset of the order conditions constructed previously. The methods proposed in [4, 25] are a subset of the EPIRK class of integrators and can be formulated using the EPIRK framework with restrictive conditions on the coefficients.

The chapter is organized as follows. In Section 2.2 we present the general construction of the split EPIRK methods. We demonstrate how B-series can be used to construct order conditions and develop an algorithm to solve order conditions for methods up to order three in Section 2.3. Section 2.4 contains numerical tests and comparisons that illustrate the performance of the new methods and show how the structure of the schemes can be optimized to improve the computational efficiency.

2.2 Construction of a new class of split EPIRK (sEPIRK) methods

Recall that the non-split EPIRK methods [65] are derived from the following integral form of the system (2.12)

$$y(t_0 + h) = y(t_0) + \varphi_1(hf'(y(t_0)))hf(y(t_0)) + h\int_0^1 e^{hf'(y(t_0))(1-\theta)} r(y(t_0 + h\theta))d\theta,$$

where $f'(y(t_0)) \in \mathbb{R}^{n\times n}$ is the Jacobian and $r(y) = f(y) - f(y(t_0)) - f'(y(t_0))(y - y(t_0))$ is the nonlinear remainder of the first-order Taylor expansion of $f(y)$. We can use the same methodology to derive split methods but consider a right-hand-side function $f(y(t))$ which can be written as the sum $f(y(t)) = Ly(t) + N(y(t))$. Here N is a nonlinear operator and L is a linear operator which is responsible for the stiffness of the system. By using the integrating factor e^{-tL}, we can derive an integral form of (2.11) as

$$y(t_0 + h) = e^{Lh}y(t_0) + \int_{t_0}^{t_0+h} e^{L(t_0+h-t)}N(y(t))dt. \tag{2.1}$$

Changing the integration variable to $\theta = (t - t_0)/h$ and making the substitution $\varphi_1(z) = (e^z - 1)/z = \int_0^1 e^{z(1-\theta)}d\theta$ where $\varphi_1(z)$ is an analytic function (and its matrix-valued form $\varphi_1(hL)$ is defined via a Taylor series expansion) we reformulate (2.1) as

$$y(t_0 + h) = y(t_0) + \varphi_1(hL)hL(y(t_0)) + h\int_0^1 e^{hL(1-\theta)}N(y(t_0 + h\theta))d\theta. \tag{2.2}$$

A split EPIRK method is constructed by developing an approximation to the integral term in (2.2). As in [65] we create a general ansatz that will allow us to derive particular schemes by solving the order conditions. Analogous to non-split EPIRK methods we use the Runge-Kutta framework and propose methods of type

$$\begin{aligned} Y_i &= y_0 + a_{i1}\psi_{i1}(g_{i1}hL)h(Ly_0 + N(y_0)) + \sum_{j=2}^{i} a_{ij}\psi_{ij}(g_{ij}hL)h\Delta^{(j-1)}N(y_0), \quad i = 1, \dots, s-1, \\ y_1 &= y_0 + b_1\psi_{s1}(g_{s1}hL)h(Ly_0 + N(y_0)) + \sum_{j=2}^{s} b_j\psi_{sj}(g_{sj}hL)\Delta^{(j-1)}N(y_0) \end{aligned} \tag{2.3}$$

where $y_0 \approx y(t_0)$, $y_1 \approx y(t_0 + h)$, the forward differences $\Delta^{(j-1)}N(y_0)$ are computed on the nodes $y_0, Y_1, \dots, Y_{s-1}$, and $\psi_{ij}(z) = \sum_{k=1}^{j} p_{ijk}\varphi_k(z)$ are linear combinations of the functions

$$\varphi_k(z) = \int_0^1 e^{z(1-\theta)}\frac{\theta^{k-1}}{(k-1)!}d\theta. \tag{2.4}$$

Note, the term $\psi_{i1}(g_{i1}hL)N(y_0)$ is part of the approximation to the integral in (2.2) and for $N \equiv 0$ the method is exact if $\psi_{s1}(z)$ is defined as $\psi_{s1}(z) = \varphi_1(z)$. The coefficients a_{ij}, g_{ij}, b_j and p_{ijk} are chosen based on the order conditions. The wide range of coefficients in anzats (2.3) ensures that the order conditions have enough flexibility to build a variety of EPIRK schemes. As demonstrated below, this flexibility allows construction of particularly efficient integrators for a specific problem.

Ansatz (2.3) is advantageous because it offers flexibility in the order conditions that allows construction of a multitude of efficient methods. The most computationally intensive part of approximating the solution at each time step is the calculation of terms like $\psi_{ij}(cL)v$ where c is a

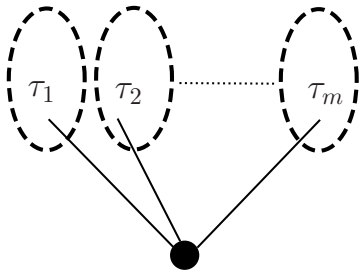
constant coefficient, $L \in R^{n\times n}$ and $v \in R^n$. For a general matrix L the method of choice to perform these calculations is the Krylov projection algorithm [1, 70]. Recall that the core of the Krylov projection method is the Arnoldi iteration that gives us an orthonormal basis in the Krylov subspace $S_k = \{v, Lv, L^2v, ..., L^{m-1}v\}$. The Arnoldi algorithm is scale invariant in a sense that we need only one iteration to approximate several terms of type $\psi_{ij}(cL)v$ with different coefficients c and functions $\psi_{ij}(z)$ but the same matrix L and vector v. Thus all of the terms $\psi_{i1}(g_{i1}hL)(Ly_0 + N(y_0))$ for $i = 1, \ldots, s$ can be computed using a single Krylov subspace projection. Similarly for a fixed j, $\psi_{ij}(g_{ij}hL)\Delta^{(j-1)}N(y_0)$ can be computed using a single Krylov subspace projection. Therefore the number of Krylov projections needed to compute a single time step is equal to the number of stages s in the method. Note also that the smaller the coefficient c is the fewer Krylov vectors will be required to reach the desired accuracy in each Krylov subspace projection, thus improving its computational efficiency. The flexibility in the order conditions offered by the EPIRK ansatz enables construction of high order schemes with both a low number of stages and a choice of small coefficients g_{ij}. Furthermore, as was shown in [65, 69] it is possible to further improve computational efficiency by using an adaptive Krylov projection algorithm to approximate $\psi_{ij}(cL)v$ terms. The adaptive EPIRK methods can be derived by setting $\psi_{ij}(z) = \varphi_{k_j}(z)$ for $i = 1, ..., s$ and $j = 1, ..., s-1$ [69]. In Section 2.3 we derive efficient split EPIRK methods of order three by minimizing coefficients g_{ij} and allowing for Krylov adaptivity.

A closely related class of exponential integrators for (2.11) are the exponential Runge-Kutta methods for semi-linear parabolic problems. These methods were first proposed in [36] and analyzed in [25]. Unlike the exponential Runge-Kutta methods [26], the split EPIRK do not impose any restrictions on the coefficients g_{ij}. In addition, the following simplifying assumptions are used for the exponential Runge-Kutta methods

$$\sum_{j=1}^{s} b_j\psi_{sj}(z) = \varphi_1(z), \qquad \sum_{j=1}^{s} a_{ij}\psi_{ij}(z) = c_i\varphi_1(c_iz), \quad i = 1, \ldots, s.$$

As was demonstrated in [65] more flexibility in the coefficients of the non-split EPIRK methods allows to derive a wider range of efficient schemes. Here we extend this advantage of the EPIRK framework to the split methods.

As demonstrated by the numerical experiments below, the computational advantages of split vs. non-split EPIRK methods depend on the properties of the specific problem we are trying to solve. Since the main difference in the computational complexity of the methods is the evaluation of $\varphi(\gamma L)v$ vs. $\varphi(\tau f'(y_0))v$, the comparative computational cost will depend on which of these evaluations is more efficient. If, for example, the spectral radius of the L operator is smaller than $f'(y_0)$ and the Krylov method is used to approximate the matrix functions-vectors products, we can expect that the split method will require fewer Krylov iterations and will be faster. However, as shown in Section 2.3 the order conditions for the non-split methods constitute a subset of the order conditions for the split methods. Thus if the order conditions make it possible to derive a method with significantly smaller τ values compared to the γ values, then the non-split method might have some computational advantages. Additionally, since L does not change from time step to time step, it is possible that a more efficient algorithm which re-uses information from previous time steps



(a) $\tau = [\tau_1 \cdots \tau_m]$

Figure 2.1: Notation for the bicolored trees

can be found for approximating $\varphi(\gamma L)v$ (e.g. in some cases FFT can be used to diagonalize L and compute the matrix function directly). If the development of efficient algorithms for non-changing L terms is possible, it can also lead to computational savings for the split compared to the non-split methods. To summarize, the split and non-split EPIRK framework offers enough flexibility so the most efficient method can be developed based on the properties of the specific problem.

In the following section we describe the derivation of the classical order conditions and solve them to construct specific schemes.

2.3 Construction of third-order schemes

Classical order conditions

In this section we use Butcher's order conditions theory to construct split EPIRK methods. In [26] the order conditions were derived by reducing the exponential method to an ordinary Runge-Kutta method. We use a different method based on B-series which allows for the derivation of the classical order conditions in a simplified way that can be easily automated. Note that in [4] the classical order conditions are also derived with the use of B-series.

We modify the approach in [65] by extending the standard B-series to account for the bicolored trees needed for the split methods. The elementary differentials in terms of f will now be split into differentials which involve both N and L (e.g. $f'f = L^2 + LN + N'L + N'N$). So rather than having single black nodes representing the elementary differentials of f we will have two different types of nodes: black nodes (filled circles) which represent $N(y(t))$ in an elementary differential, and white nodes (open circles) which represent the operator L. We will use $\widetilde{\mathcal{T}}$ to denote the set of all bicolored trees. If $\{\tau_i\}_{i=1}^m$ is the set of m trees from $\widetilde{\mathcal{T}}$ then using Butcher's notation [6, 7] we define $\tau = [\tau_1 \cdots \tau_m]$ to be a tree obtained by attaching trees τ_i as branches of a black root. We will use $\tau = [\tau_i]_\circ$ to denote a tree that is obtained by adding a white vertex to $\tau_i \in \widetilde{\mathcal{T}}$ as its root (Fig. 2.1).

The order of a tree $|\tau|$, symmetry of a tree $\sigma(\tau)$, and density of a tree $\gamma(\tau)$ (for $\tau \in \widetilde{\mathcal{T}}$) are defined in the same way as for the single colored trees [6, 7, 65]. The order $|\tau|$ is defined as the number of vertices in τ and the density, $\gamma(\tau)$, of tree τ is defined as the product over all vertices of the order of the subtree rooted at that vertex. For the definition of the symmetry of a tree $sigma(\tau)$ we

refer to [7]. A recursive definition for the symmetry can be obtained as follows. Suppose that $\tau = [\tau_1 \cdots \tau_m]$, where some τ_i $(i = 1, \ldots, m)$ are identical. We can then write $\tau = [\tau_1^{n_1} \tau_2^{n_2} \cdots \tau_k^{n_k}]$ where τ_i $(i = 1, \ldots, k)$ are distinct and n_i denote the number of copies of τ_i with $m = n_1 + n_2 + \cdots + n_k$. Given that the symmetry of a single vertex tree is 1 we obtain a recursive definition for the symmetry of any $\tau = [\tau_1^{n_1} \cdots \tau_k^{n_k}]$ as $\sigma(\tau) = \Pi_{i=1}^{k} n_i! \sigma(\tau_i)^{n_i}$. Note that for any $\tau = [\tau_i]_\circ$, $\sigma(\tau) = \sigma(\tau_i)$.

For our purposes there is a simple algorithm to compute the symmetry $\sigma(\tau)$. Assign a number n_i to each vertex $(i = 1, \ldots, |\tau|)$ so that n_i denotes the number of identical subtrees branching from each vertex. Then $\sigma(\tau)$ is given as the product of factorials of these numbers, i.e. $\sigma(\tau) = n_1! n_2! \cdots n_{|\tau|}!$. The values of $|\tau|$, $\sigma(\tau)$ and $\gamma(\tau)$ are shown in Table 2.1 for all trees up to order three. Let $\{\emptyset\}$ denote a set containing only the empty tree. Now by considering a mapping $x : \widetilde{\mathcal{T}} \cup \{\emptyset\} \to \mathbb{R}$ we can express expansions in terms of elementary differentials as a B-series

$$B(x, y_0) = x(\emptyset) y_0 + \sum_{\tau \in \widetilde{\mathcal{T}}} x(\tau) \frac{h^{|\tau|}}{\sigma(\tau)} \delta(\tau)(y_0),$$

where $\delta(\tau)(y_0)$ is used to denote the elementary differential (evaluated at y_0) corresponding to a tree $\tau \in \widetilde{\mathcal{T}}$.

Compared to single colored trees, the number of the bicolored trees is much larger. For example, if we consider all trees up to fourth-order then there are 51 bicolored trees and only 8 single colored trees. With this in mind, constructing the general formulas for the B-series composition as was done in [65] can be quite cumbersome. Instead we will take the direct approach that yields the desired B-series and can be easily automated. We will then illustrate our methodology by deriving three-stage methods of order three.

Generally functions $\psi_{ij}(z)$ can be chosen to be independent of the stage index i and the term number j. Here we assume $\psi_{ij}(z) = \psi_j(z) = \sum_{k=1}^{j} p_{jk} \varphi_k(z)$ in order to simplify the order conditions and the implementation of the adaptive schemes. The general form of the three-stage methods with this simplification is

$$\begin{aligned} Y_1 &= y_0 + a_{11} \psi_1(g_{11} hL) h(L y_0 + N(y_0)) \\ Y_2 &= y_0 + a_{21} \psi_1(g_{21} hL) h(L y_0 + N(y_0)) + a_{22} \psi_2(g_{22} hL) h \Delta N(y_0) \\ y_1 &= y_0 + b_1 \psi_1(g_{31} hL) h(L y_0 + N(y_0)) + b_2 \psi_2(g_{32} hL) h \Delta N(y_0) + b_3 \psi_3(g_{33} hL) h \Delta^2 N(y_0) \end{aligned} \quad , \qquad (2.5)$$

where $\psi_j(z) = \sum_{k=1}^{j} p_{jk} \varphi_k(z)$, $\Delta N(y_0) = N(Y_1) - N(y_0)$ and $\Delta^2 N(y_0) = N(Y_2) - 2N(Y_1) + N(y_0)$. Later in the section we prove that a third-order method with fewer than three stages cannot be derived.

We consider $\mathcal{T} \cup \{\emptyset\} \subset \widetilde{\mathcal{T}}$ where $\mathcal{T}$ consists of all 17 bicolored trees up to order three (See Table 2.1). As we seek to compute the B-series representation of the exponential integrator (2.5), we need to compute the B-series coefficients for the terms of type $N(Y_i)$ and the products $\psi(hL) h N(Y_i)$ (for $i = 1, 2$). A starting point is to define a generic truncated B-series $B(x, y_0) = x(\emptyset) y_0 + \sum_{i=1}^{17} x(\tau_i) \frac{h^{|\tau_i|}}{\sigma(\tau_i)} \delta(\tau_i)(y_0)$ whose coefficients are given by $\{x(\emptyset), x(\tau_1), x(\tau_2), \ldots, x(\tau_{17})\} = \{1, x_1, x_2, \ldots, x_{17}\}$. We now find the B-series coefficients of $N(B(x, y_0))$ using Butcher's formulation of Taylor's theorem (Theorem 306A in [6]). In order to state this theorem, we need to introduce the following notation. Let I denote a sequence of integers from the set $\{1, 2, \ldots, m\}$ and

$\mathcal{I}_m$ the set of all such sequences. If I contains k_i occurrences of i, for each $i = 1, \ldots, m$, then the symmetry of I is defined by $\widehat{\sigma}(I) = k_1!k_2! \ldots k_m!$.

Elem. Diff., $\delta(\tau)(y_0)$	N	L	$N'N$	$N'L$	LN'	L^2
Trees	τ_1	τ_2	τ_3	τ_4	τ_5	τ_6
Order, $\lvert\tau_i\rvert$	1	1	2	2	2	2
Symmetry, $\sigma(\tau_i)$	1	1	1	1	1	1
Density, $\gamma(\tau_i)$	1	1	2	2	2	2

Elem. Diff., $\delta(\tau)(y_0)$	$N''(N,N)$	$N''(L,N)$	$N''(L,L)$	$N'N'N$	$N'LN$
Trees	τ_7	τ_8	τ_9	τ_{10}	τ_{11}
Order, $\lvert\tau_i\rvert$	3	3	3	3	3
Symmetry, $\sigma(\tau_i)$	2	1	2	1	1
Density, $\gamma(\tau_i)$	3	3	3	6	6

Elem. Diff., $\delta(\tau)(y_0)$	$N'N'L$	$N'L^2$	$LN'N$	L^2N	L^3	$LN'L$
Trees	τ_{12}	τ_{13}	τ_{14}	τ_{15}	τ_{16}	τ_{17}
Order, $\lvert\tau_i\rvert$	3	3	3	3	3	3
Symmetry, $\sigma(\tau_i)$	1	1	1	1	1	1
Density, $\gamma(\tau_i)$	6	6	6	6	6	6

Table 2.1: Trees in $\mathcal{T}$, corresponding order of trees $|\tau|$, symmetry $\sigma(\tau)$, density $\gamma(\tau)$ and elementary differentials

Theorem 1. *Let $N(y) : \mathbb{R}^n \to \mathbb{R}^n$ and take $\mathcal{I}_m$ to be the set of all finite sequences from the set of integers $\{1, \ldots, m\}$. Let $y_0, \xi_i \in \mathbb{R}^n$ $(i = 1, 2, \ldots, m)$ then the Taylor expansion of $N\left(y_0 + \sum_{i=1}^m \xi_i\right)$ is given by*

$$N\left(y_0 + \sum_{i=1}^{m} \xi_i\right) = \sum_{I \in \mathcal{I}_m} \frac{1}{\widehat{\sigma}(I)} N^{(\#I)}(y_0)\xi_I, \tag{2.6}$$

where I is any sequence in $\mathcal{I}_m$, $\widehat{\sigma}(I)$ is the symmetry of the sequence, $\#I$ is the cardinality of the sequence, and the operands, ξ_I for the multilinear operators $N^{(\#I)}(y_0)$ are defined for each $I = (i_1, i_2, \ldots, i_k) \in \mathcal{I}_m$ as $(\xi_{i_1}, \xi_{i_2}, \ldots, \xi_{i_k})$.

The following corollary is an extension of Butcher's result for single colored trees to the bicolored trees.

Corollary 2. *Consider the truncated B-series* $B(x, y_0) = y_0 + \sum_{i=1}^{m} x(\tau_i)\frac{h^{|\tau_i|}}{\sigma(\tau_i)}\delta(\tau_i)(y_0)$ *where* $\{\tau_i\}_{i=1}^{m} \subset \widetilde{\mathcal{T}}$ *consists of all bicolored trees up to order s. Under the assumptions of Theorem 1 we have the following B-series representation of* $hN(B(x, y_0))$*:*

$$hN(B(x, y_0)) = \widetilde{x}(\emptyset)y_0 + \sum_{\tau\in\widetilde{\mathcal{T}}} \widetilde{x}(\tau)\frac{h^{|\tau|}}{\sigma(\tau)}\delta(\tau)(y_0) \tag{2.7}$$

where the coefficients $\widetilde{x}(\tau)$ *are given by*

$$\widetilde{x}(\tau) = \begin{cases} 0, & \tau = \emptyset, \\ 1, & \tau = \tau_1, \\ x(\tau_{i_1})\cdots x(\tau_{i_k}), & \tau = [\tau_{i_1}\cdots\tau_{i_k}] \text{ for } I = (i_1, \ldots, i_k) \in \mathcal{I}_m \\ 0, & \text{for all other trees (including all trees with a white root)} \end{cases} . \tag{2.8}$$

Proof. For ease of notation let us use $\delta_i = \delta(\tau_i)(y_0)$ to denote an elementary differential (evaluated at y_0) corresponding to a tree $\tau_i \in \mathcal{T}$. By taking $\xi_i = \frac{x(\tau_i)h^{|\tau_i|}}{\sigma(\tau_i)}\delta_i$, applying Theorem 1 and using the linearity and/or multi-linearity of the operator $N^{(\#I)}$ we arrive at

$$N\left(y_0 + \sum_{i=1}^{m} x(\tau_i)\frac{h^{|\tau_i|}}{\sigma(\tau_i)}\delta_i\right) = \sum_{I\in\mathcal{I}_m} \frac{1}{\widehat{\sigma}(I)}N^{(\#I)}(y_0)\xi_I, \tag{2.9}$$

where for each $I = (i_1, i_2, \ldots, i_k) \in \mathcal{I}_m$

$$\xi_I = \frac{x(\tau_{i_1})\cdots x(\tau_{i_k})\cdot h^{|\tau_{i_1}|+\cdots+|\tau_{i_k}|}}{\sigma(\tau_{i_1})\cdots\sigma(\tau_{i_1})}(\delta_{i_1}, \ldots, \delta_{i_k}). \tag{2.10}$$

Note that none of the terms in the expansion (2.9) include the elementary differentials which have a leading L. Thus all trees which have a white root will have a zero coefficient.

For $I = (i_1, i_2, \ldots, i_k) \in \mathcal{I}_m$, the term $h\left(\frac{1}{\widehat{\sigma}(I)}N^{(\#I)}(y_0)\xi_I\right)$ corresponds to the tree $\tau = [\tau_{i_1}\cdots\tau_{i_k}] \in \widetilde{\mathcal{T}}$. By definition we have $|\tau| = \left|\tau_{i_1}\right| + \cdots + \left|\tau_{i_k}\right| + 1$. Let l_j denote the number of subtrees serving as branches of the root vertex of a tree τ_j. It follows that $\sigma(\tau) = (l_1!\cdots l_m!)\,\sigma(\tau_{i_1})\cdots\sigma(\tau_{i_k})$. Since $\widehat{\sigma}(I) = l_1!\cdots l_m!$, we have $\sigma(\tau) = \widehat{\sigma}(I)\sigma(\tau_{i_1})\cdots\sigma(\tau_{i_k})$. Therefore with the use of (2.10), $h\left(\frac{1}{\widehat{\sigma}(I)}N^{(\#I)}(y_0)\xi_I\right)$ can be written as $\frac{\widetilde{x}(\tau)h^{|\tau|}}{\sigma(\tau)}\delta(\tau)(y_0)$:

$$\begin{aligned} h\left(\frac{1}{\widehat{\sigma}(I)}N^{(\#I)}(y_0)\xi_I\right) &= \frac{x(\tau_{i_1})\cdots x(\tau_{i_k})\cdot h^{|\tau_{i_1}|+\cdots+|\tau_{i_k}|+1}}{\widehat{\sigma}(I)\sigma(\tau_{i_1})\cdots\sigma(\tau_{i_1})}N^{(\#I)}(\delta_{i_1}, \ldots, \delta_{i_k}) \\ &= \frac{\widetilde{x}(\tau)h^{|\tau|}}{\sigma(\tau)}\delta(\tau)(y_0). \end{aligned}$$

□

The original exponential integrators were constructed for the first-order ODE systems where the right-hand-side could be split into the linear and nonlinear parts as follows

$$y'(t) = Ly(t) + N(y(t)), \quad y(t_0) = y_0. \tag{2.11}$$

Here the linear part $L \in \mathbb{R}^{n\times n}$ is assumed to contain the stiffness of the problem and $N(y(t)) : \mathbb{R}^n \to \mathbb{R}^n$ is the nonlinear remainder. Later exponential methods were extended to more general non-split systems of the form

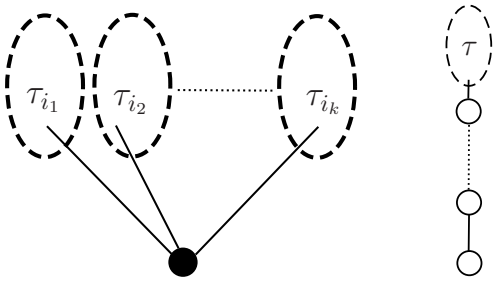
$$y'(t) = f(y(t)), \quad y(t_0) = y_0, \tag{2.12}$$

where $f(y(t)) : \mathbb{R}^n \to \mathbb{R}^n$ [24, 27, 66]. The key to constructing an exponential integrator is in expressing an approximate solution in terms of an exponential or exponential-like functions (see eqn. (2.4)) of the linear operator L for the problems of type (2.11) or a Jacobian Df/Dy for the systems like (2.12). This approach alleviates severe stability restrictions on the step size characteristic of the explicit methods. To be competitive with implicit methods an exponential integrator must also be very efficient per integration step. Different techniques have been used to reduce the per-step computational complexity of exponential methods [24, 29, 66]. Since evaluation of the matrix exponential functions constitutes the main computational cost of an exponential integrator [38], development of an efficient integrator is tied to an algorithm chosen to approximate these functions. One of the most general and widely used ways to compute the matrix exponential functions is the Krylov projection algorithm [1, 70, 23]. The exponential propagation iterative methods of Runge-Kutta type (EPIRK) [66, 65] are designed to reduce the computational cost per-step. This class of exponential integrators possesses several desirable features that allow construction of very efficient schemes. The generality and flexibility of EPIRK-type methods have been shown to offer advantages over other exponential and implicit methods [38, 66, 65, 69]. In this chapter we extend the ideas behind the construction of the EPIRK methods to split problems of type (2.11).

Now we describe the procedure of finding the B-series coefficients of $hN(B(x, y_0))$ by using Corollary 2. For generality, let us consider $\{\tau_i\}_{i=1}^m \subset \widetilde{\mathcal{T}}$ which consists of all bicolored trees of order up to s. We begin the procedure by identifying trees which have zero coefficients (or equivalently, trees that will not appear in the expansion (2.9)). According to Corollary 2 the coefficient for any tree with a white root will be zero. Hence we set the B-series coefficients of trees with a white root to zero.

The next step is to collect all the terms in (2.9) that correspond to $\#I = 1$. Each of these terms must correspond to a tree with a black root and a single branch. Therefore we simply identify the trees which have a black root with a single branch and write down the coefficients according to Corollary 2. For example, suppose τ_j is a tree created by adding a black root to a tree τ_i (i.e. $\tau_j = [\tau_i]$). Then the B-series coefficient $x(\tau_i)$ becomes the coefficient $\widetilde{x}(\tau_j)$ of the differential corresponding to the tree τ_j in the B-series expansion of $hN(B(x, y - 0))$ given in (2.7).

We can now obtain the remaining coefficients by following the process described below starting with $\#I = 2$ then increasing the cardinality incrementally to $\#I = s - 1$. For each value $\#I = k$ ($k = 2, \ldots, (s - 1)$) and the corresponding set of trees, we look for all trees whose root has k branches. If τ is such a tree (i.e. $\tau = [\tau_{i_1} \cdots \tau_{i_k}]$) with branches τ_{i_j} attached to the root vertex (Fig. 2.2), then the B-series coefficient $\widetilde{x}(\tau)$ can be directly found from (2.8). For example, considering



(a) $N^{(k)}(\delta_{i_1}, \delta_{i_2}, \ldots, \delta_{i_k})$ (b) $(hL)^n\delta(\tau)(y_0)$

Figure 2.2: Operations on bicolored trees

$\mathcal{T}$ and $\#I = 2$ we have three such trees τ_7, τ_8, and τ_9 (Table 2.1). For τ_7 we have τ_1 is attached to both branches and so the B-series coefficient $\widetilde{x}(\tau_7) = x(\tau_1)x(\tau_1) = x(\tau_1)^2$. The coefficient of τ_9 is obtained in a similar way, except that both branches consist of τ_2 and so the B-series coefficient is given by $\widetilde{x}(\tau_9) = x(\tau_2)^2$. For τ_8 we have that τ_2 is attached to one branch while τ_1 is attached to the other which results in the coefficient $\widetilde{x}(\tau_8) = x(\tau_2)x(\tau_1)$.

By applying the above procedure for $\mathcal{T} \subset \widetilde{\mathcal{T}}$ with $m = 17$ we obtain $hN\left(y_0 + \sum_{i=1}^{17} x_i \frac{h^{|\tau_i|}}{\sigma(\tau_i)}\delta_i\right)$ corresponds to the following B-series $B(\widetilde{x}, y_0) = \widetilde{x}(\emptyset)y_0 + \sum_{\tau\in\mathcal{T}} \widetilde{x}(\tau)\frac{h^{|\tau|}}{\sigma(\tau)}\delta(\tau)(y_0)$ with coefficients $\widetilde{x}(\tau) = \left\{0, 1, 0, x_1, x_2, 0, 0, x_1^2, x_1x_2, x_2^2, x_3, x_5, x_4, x_6, 0, 0, 0, 0\right\}$ (see also Table 2.2).

Now we also need to compute B-series coefficients of the product terms like $\psi_j(hL)h(Ly_0 + N(y_0))$, $\psi_j(hL)h\Delta N(y_0)$ and $\psi_j(hL)h\Delta^2 N(y_0)$ where $\psi_j(z)$ can be expanded in a Taylor series. For B-series consisting of single colored trees, Butcher observed [7] that if such B-series is multiplied by hf' on the left, then the coefficient of each tree τ in the resulting B-series is equal to the coefficient in the original series corresponding to a tree that produces τ if a black root is added to it. Similarly, for the bicolored trees B-series, if a B-series gets multiplied by hL on the left, the coefficients of the resulting B-series are simply the shifted coefficients of the original series, with a coefficient of each tree in the original B-series mapped to a coefficient of a tree obtained by adding a white root to the original tree in the B-series of the product. For example, $\tau_{15} = [\tau_5]_\circ$. For repeated multiplication of hL, say $(hL)^n$, we simply get a stack of n white nodes (with each having at most one branch (or child)) added to the existing tree τ (see Figure 2.2). Thus we can easily compute the B-series coefficients corresponding to the terms $(hL)^n v$ and hence the B-series coefficients for $\psi_j(hL)v$.

Using the above results we can now find the B-series expansions of the stage values Y_i and of the approximate solution $y_1 \approx y(t_0 + h)$. Let $B(a, y_0) = a(\emptyset)y_0 + \sum_{\tau\in\mathcal{T}} a(\tau)\frac{h^{|\tau|}}{\sigma(\tau)}\delta(\tau)(y_0)$ be the

Elem. Diff., $\delta(\tau)(y_0)$	δ_1	δ_2	δ_3	δ_4	δ_5	δ_6
Trees	τ_1	τ_2	τ_3	τ_4	τ_5	τ_6
Expansion Term			$N'\delta_1$	$N'\delta_2$		
Expansion Coefficient, $\widetilde{x}(\tau)$	1	0	x_1	x_2	0	0

Elem. Diff., $\delta(\tau)(y_0)$	δ_7	δ_8	δ_9	δ_{10}	δ_{11}
Trees	τ_7	τ_8	τ_9	τ_{10}	τ_{11}
					
Expansion Term	$N''(\delta_1,\delta_1)$	$N''(\delta_2,\delta_1)$	$N''(\delta_2,\delta_2)$	$N'\delta_3$	$N'\delta_5$
Expansion Coefficient, $\widetilde{x}(\tau)$	x_1^2	x_1x_2	x_2^2	x_3	x_5

Elem. Diff., $\delta(\tau)(y_0)$	$N'N'L$	$N'L^2$	$LN'N$	L^2N	L^3	$LN'L$
Trees	τ_{12}	τ_{13}	τ_{14}	τ_{15}	τ_{16}	τ_{17}
Expansion Term	$N'\delta_4$	$N'\delta_6$				
Expansion Coefficient, $\widetilde{x}(\tau)$	x_4	x_6	0	0	0	0

Table 2.2: Expansion terms and coefficients according to Corollary 2

B-series for our approximate solution. Let us Taylor expand the exact solution $y(t_0 + h)$ as

$$\begin{aligned} y(t_0+h) &= y_0 + h(Ly_0 + N(y_0)) + \frac{h^2}{2}(L^2 + LN + N'L + N'N) \\ &\quad + \frac{h^3}{3!}\Big(L^3 + L^2N + LN'L + LN'N + N''(L,L) + 2N''(L,N) \\ &\quad + N''(N,N) + N'L^2 + N'LN + N'N'L + N'N'N\Big) + \dots \qquad (2.13) \\ &= y_0 + \sum_{\tau\in\mathcal{T}} \frac{h^{|\tau|}}{\gamma(\tau)\sigma(\tau)}\delta(\tau)(y_0). \end{aligned}$$

By matching the coefficients of the approximate and exact solutions we set $a(\tau) = \frac{1}{\gamma(\tau)}$ for each $\tau \in \mathcal{T}$ and obtain the desired order conditions. We developed a simple Mathematica script to compute the order conditions up to order three which are listed in Table 2.3.

Solving conditions up to order three.

In this section we propose an algorithm to solve the order conditions (Table 2.3) to construct schemes of order three. We first note that (C1), (C3), and (C8) yield $b_1 = 1/p_{11}$ and $g_{31} = 1$.

	Cor. Trees	Order Condition
(C1)	τ_1 & τ_2	$b_1 p_{11} = 1$
(C2)	τ_3 & τ_4	$6a_{11}b_2p_{11}p_{21} + 3a_{11}b_2p_{11}p_{22} - 12a_{11}b_3p_{11}p_{31} + 6a_{21}b_3p_{11}p_{31}$ $-6a_{11}b_3p_{11}p_{32} + 3a_{21}b_3p_{11}p_{32} - 2a_{11}b_3p_{11}p_{33} + a_{21}b_3p_{11}p_{33} = 3$
(C3)	τ_5 & τ_6	$b_1 g_{31} p_{11} = 1$
(C4)	τ_7, τ_8 & τ_{10}	$6a_{11}^2b_2p_{11}^2p_{21} + 3a_{11}^2b_2p_{11}^2p_{22} - 12a_{11}^2b_3p_{11}^2p_{31} + 6a_{21}^2b_3p_{11}^2p_{31}$ $-6a_{11}^2b_3p_{11}^2p_{32} + 3a_{21}^2b_3p_{11}^2p_{32} - 2a_{11}^2b_3p_{11}^2p_{33} + a_{21}^2b_3p_{11}^2p_{33} = 2$
(C5)	τ_{10} & τ_{12}	$12a_{11}a_{22}b_3p_{11}p_{21}p_{31} + 6a_{11}a_{22}b_3p_{11}p_{22}p_{31} + 6a_{11}a_{22}b_3p_{11}p_{21}p_{32}$ $+3a_{11}a_{22}b_3p_{11}p_{22}p_{32} + 2a_{11}a_{22}b_3p_{11}p_{21}p_{33}$ $+a_{11}a_{22}b_3p_{11}p_{22}p_{33} = 2$
(C6)	τ_{11} & τ_{13}	$6a_{11}b_2g_{11}p_{11}p_{21} + 3a_{11}b_2g_{11}p_{11}p_{22} - 12a_{11}b_3g_{11}p_{11}p_{31}$ $+6a_{21}b_3g_{21}p_{11}p_{31} - 6a_{11}b_3g_{11}p_{11}p_{32} + 3a_{21}b_3g_{21}p_{11}p_{32}$ $-2a_{11}b_3g_{11}p_{11}p_{33} + a_{21}b_3g_{21}p_{11}p_{33} = 2$
(C7)	τ_{14} & τ_{17}	$12a_{11}b_2g_{32}p_{11}p_{21} + 4a_{11}b_2g_{32}p_{11}p_{22} - 24a_{11}b_3g_{33}p_{11}p_{31}$ $+12a_{21}b_3g_{33}p_{11}p_{31} - 8a_{11}b_3g_{33}p_{11}p_{32} + 4a_{21}b_3g_{33}p_{11}p_{32}$ $-2a_{11}b_3g_{33}p_{11}p_{33} + a_{21}b_3g_{33}p_{11}p_{33} = 4$
(C8)	τ_{15} & τ_{16}	$b_1 g_{31}^2 p_{11} = 1$

Table 2.3: Order conditions

Define $\alpha = a_{11}b_2p_{11}$, $\beta = a_{11}b_3p_{11}$, and $\gamma = a_{21}b_3p_{11}$ then conditions (C2), (C6), and (C7) form a linear system (2.14) for α, β, and γ.

$$\begin{aligned} 6\alpha p_{21} + 3\alpha p_{22} + \gamma(6p_{31} + 3p_{32} + p_{33}) - 12\beta(p_{31} - 6p_{32} - 2p_{33}) &= 3 \\ \gamma g_{21}(6p_{31} + 3p_{32} + p_{33}) + g_{11}(3\alpha(2p_{21} + p_{22}) - 2\beta(6p_{31} + 3p_{32} + p_{33})) &= 2 \\ 4\alpha g_{32}(3p_{21} + p_{22}) - (2\beta - \gamma)g_{33}(12p_{31} + 4p_{32} + p_{33}) &= 4 \end{aligned} \tag{2.14}$$

The linear system (2.14) can be solved for α, β, and γ in terms of the coefficients p_{ij}'s and g_{ij}'s (see Table 2.4). Then solving (C5) yields an expression for a_{22} which is also in terms of the coefficients p_{ij}'s and g_{ij}'s (see Table 2.4).

Note that we have eight order conditions with 18 parameters and the solution to this system (Table 2.3) is non-unique. We will use computational efficiency of the resulting method as a guiding principle for deriving specific schemes. As was mentioned in Section 2.2 we seek a method that can be coupled with the adaptive Krylov algorithm and we want to minimize the g_{ij} coefficients. There is a lot of flexibility in deriving an adaptive Krylov-friendly method with the requirement that the ψ functions consist of a single function φ_k. Following [69] we construct such schemes with $p_{22} = p_{31} = p_{32} = 0$. Since $g_{31} = 1$ must hold and the order conditions are independent of g_{22}, we will focus on minimizing g_{32} and g_{33}.

We substitute the expressions given in Table 2.4 for α, β, and γ into (C4) and solve for g_{21} to obtain

$$g_{21} = \frac{2a_{11}p_{11} - 2a_{21}p_{11} + g_{11}(-2 + 3a_{21}p_{11})}{-2 + 3a_{11}p_{11}}. \tag{2.15}$$

Now by making another substitution: $A = a_{11}p_{11}$ and $B = a_{21}p_{11}$ we have $a_{11} = A/p_{11}$, $a_{21} = B/p_{11}$, $b_2 = \alpha/A$ and $g_{21} = (2A - 2B + (-2 + 3B)g_{11})/(-2 + 3A)$. From $b_3 = \gamma/B = \beta/A$ and

$\alpha = \dfrac{12\,(-2+3g_{33})\,p_{31} + 12\,(-1+g_{33})\,p_{32} + (-4+3g_{33})\,p_{33}}{-4g_{32}\,(3p_{21}+p_{22})\,(6p_{31}+3p_{32}+p_{33}) + 3g_{33}\,(2p_{21}+p_{22})\,(12p_{31}+4p_{32}+p_{33})},$
$\beta = \dfrac{G_1}{G_2},$ $G_1 = 12\,(g_{11}-g_{21})\,(2p_{21}+p_{22})\,(6p_{31}+3p_{32}+p_{33})$ $+4\,(3g_{21}-2)\,g_{32}\,(3p_{21}+p_{22})\,(6p_{31}+3p_{32}+p_{33})$ $-3\,(3g_{11}-2)\,g_{33}\,(2p_{21}+p_{22})\,(12p_{31}+4p_{32}+p_{33})$ $G_2 = 8\,(g_{11}-g_{21})\,g_{32}\,(3p_{21}+p_{22})\,(6p_{31}+3p_{32}+p_{33})^2$ $-6\,(g_{11}-g_{21})\,g_{33}\,(2p_{21}+p_{22})\,(6p_{31}+3p_{32}+p_{33})\,(12p_{31}+4p_{32}+p_{33})$
$\gamma = \dfrac{-2+3g_{11}}{(g_{11}-g_{21})\,(6p_{31}+3p_{32}+p_{33})},$
$a_{22} = \dfrac{G_1}{G_2},$ $G_1 = 16\,(g_{21}g_{32}-g_{11}g_{32})\,(3p_{21}+p_{22})\,(6p_{31}+3p_{32}+p_{33})$ $-12\,(g_{21}g_{33}-g_{11}g_{33})\,(2p_{21}+p_{22})\,(12p_{31}+4p_{32}+p_{33})$ $G_2 = 12\,(g_{21}-g_{11})\,(2p_{21}+p_{22})^2\,(6p_{31}+3p_{32}+p_{33})$ $-4\,(3g_{21}-2)\,g_{32}\,(2p_{21}+p_{22})\,(3p_{21}+p_{22})\,(6p_{31}+3p_{32}+p_{33})$ $+3\,(3g_{11}-2)\,g_{33}\,(2p_{21}+p_{22})^2\,(12p_{31}+4p_{32}+p_{33})$

Table 2.4: Explicit formulas for α, β, γ, and a_{22}

substituting our expressions for β, γ, and g_{21} into $\gamma/B = \beta/A$ we obtain an equation which can be solved for g_{32}:

$$g_{32} = \frac{4AB - 4B^2 - 4Ag_{33} + 6A^2g_{33} + 2Bg_{33} - 3ABg_{33}}{-8A + 12A^2 + 2(2-3B)B}. \tag{2.16}$$

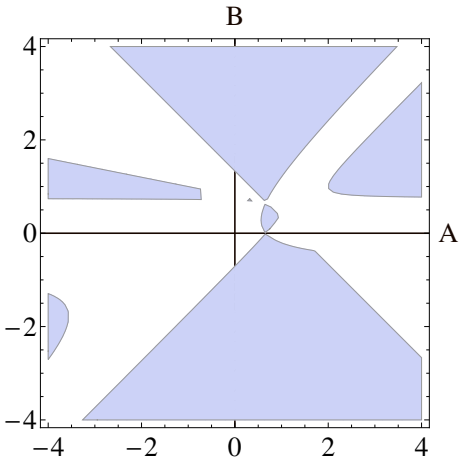
Now we can vary A and B to minimize the g_{ij} coefficients. Following this approach can result in a_{ij} and b_i coefficients becoming very large, so we put restrictions on our parameters A and B. The growth of A and B can be controlled by setting

$$|p_{11}a_{11}| \le M, \quad |p_{21}a_{22}| \le M, \quad |p_{21}b_2| \le M, \quad |p_{33}b_3| \le M, \tag{2.17}$$

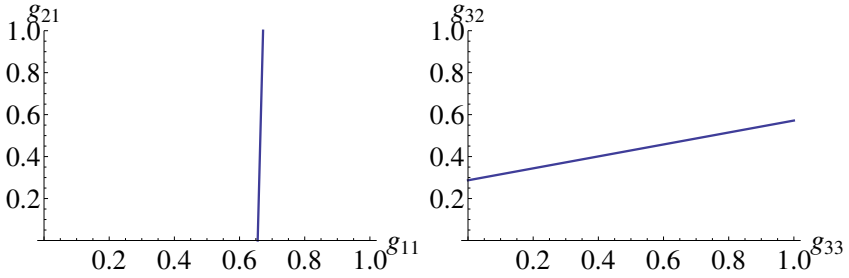
for some $M > 0$. Upon simplifying expressions for a_{22}, b_2, and b_3 we have

$$\begin{aligned} a_{22} &= \frac{2(A-B)(2g_{32}-g_{33})}{(-4A+4B+(4-6B)g_{32}+(-2+3A)g_{33})p_{21}}, \\ b_2 &= \frac{-4+3g_{33}}{6A(-2g_{32}+g_{33})p_{21}}, \quad b_3 = \frac{-2+3A}{(A-B)Bp_{33}}. \end{aligned} \tag{2.18}$$

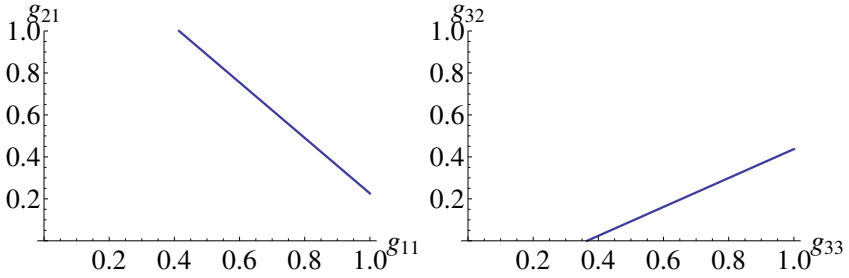
Using a Mathematica script we can identify a region (Fig. 2.3(a)) of values of A and B such that (2.17) and (2.18) are satisfied. Then we plot linear in g_{ij} expressions (2.15) and (2.16) for different values of A and B and select desirable values for g_{ij}'s. For example by taking $M = 4$ in (2.17) we find two such values of A and B, the first of which enables us to minimize g_{33} and the second to minimize g_{32}: (i) $A = \frac{7741}{11812}$ and $B = \frac{116109 - \sqrt{14906535801}}{354360}$ (ii) $A = \frac{19576}{13335}$ and $B = -\frac{11}{28}$.



(a) Admissible values for parameters A and B



(b) $A = \frac{7741}{11812}$ and $B = \frac{116109 - \sqrt{14906535801}}{354360}$



(c) $A = \frac{19576}{13335}$ and $B = -\frac{11}{28}$

Figure 2.3: Choice of parameters and g_{ij} coefficients

Choosing $A = \frac{7741}{11812}$, $B = \frac{116109-\sqrt{14906535801}}{354360}$, $p_{11} = p_{21} = 1$, $p_{33} = -1$, $g_{11} = 84/125$, and $g_{33} = 1/50$ we obtain a method within this new class which we will call *sEPIRK3*. The coefficients of this scheme are given in Table 2.5. Another method, which we will call *sEPIRK3b* is obtained by choosing $A = \frac{19576}{13335}$ and $B = -\frac{11}{28}$ and setting $p_{11} = p_{21} = 1$, $p_{33} = -1$, $g_{11} = 1$, and $g_{33} = 2/5$ (see Table 2.6).

$$\begin{bmatrix} a_{11} & & \\ a_{21} & a_{22} & \\ b_1 & b_2 & b_3 \end{bmatrix} = \begin{bmatrix} 0.6553504910260752 & & \\ -0.01688487669656697 & 0.5101814783323579 & \\ 1 & 1.772759951134867 & -2.990898555087077 \end{bmatrix}$$

$$\begin{bmatrix} g_{11} & & \\ g_{21} & g_{22} & \\ g_{31} & g_{32} & g_{33} \end{bmatrix} = \begin{bmatrix} 0.672 & & \\ 0.988825702285879 & g_{22} & \\ 1 & 0.2926125311398587 & 0.02 \end{bmatrix}$$

$$\begin{bmatrix} p_{11} & & \\ p_{21} & p_{22} & \\ p_{31} & p_{32} & p_{33} \end{bmatrix} = \begin{bmatrix} 1 & & \\ 1 & 0 & \\ 0 & 0 & -1 \end{bmatrix}$$

Table 2.5: Coefficients for sEPIRK3. For the implementation the free parameter g_{22} is chosen to be equal to g_{32}

$$\begin{bmatrix} a_{11} & & \\ a_{21} & a_{22} & \\ b_1 & b_2 & b_3 \end{bmatrix} = \begin{bmatrix} 1.468016497937758 & & \\ -0.3928571428571428 & -0.2071463079588416 & \\ 1 & -0.90222697401869 & 3.288454722218915 \end{bmatrix}$$

$$\begin{bmatrix} g_{11} & & \\ g_{21} & g_{22} & \\ g_{31} & g_{32} & g_{33} \end{bmatrix} = \begin{bmatrix} 1 & & \\ 0.225942042547882 & g_{22} & \\ 1 & 0.02383077579162529 & 0.4 \end{bmatrix}$$

$$\begin{bmatrix} p_{11} & & \\ p_{21} & p_{22} & \\ p_{31} & p_{32} & p_{33} \end{bmatrix} = \begin{bmatrix} 1 & & \\ 1 & 0 & \\ 0 & 0 & -1 \end{bmatrix}$$

Table 2.6: Coefficients for sEPIRK3b. For the implementation the free parameter g_{22} is chosen to be equal to g_{32}

Note that the order conditions for a non-split EPIRK method are a subset of the order conditions for a split EPIRK scheme. Thus the coefficients derived for a split EPIRK method automatically give a non-split EPIRK integrator. For example, let us consider a three-stage EPIRK method [65] of the form

$$\begin{aligned} Y_1 &= y_0 + a_{11}\psi_1(g_{11}hf'(y_0))hf(y_0) \\ Y_2 &= y_0 + a_{21}\psi_1(g_{21}hf'(y_0))hf(y_0) + a_{22}\psi_2(g_{22}hf'(y_0))h\Delta r(y_0) \\ y_1 &= y_0 + b_1\psi_1(g_{31}hf'(y_0))hf(y_0) + b_2\psi_2(g_{32}hf'(y_0))h\Delta r(y_0) + b_3\psi_3(g_{33}hf'(y_0))h\Delta^2 r(y_0) \end{aligned}, \tag{2.19}$$

where $f'(y_0) \in \mathbb{R}^{n\times n}$ is the Jacobian and $r(y) = f(y) - f(y_0) - f'(y_0)(y - y_0)$ is the nonlinear remainder of the first-order Taylor expansion of $f(y)$. Then the coefficients for our third-order sEPIRK3 and

sEPIRK3b methods (Tables 2.5 & 2.6) yield two third-order three-stage EPIRK schemes which will be referred to as EPIRK3s3 and EPIRK3s3b. We will use the scheme EPIRK3s3 for numerical comparisons in Section 2.4.

We now turn back to the question of what is the minimum number of stages needed to derive a split EPIRK method of a given order. We use the results obtained above to prove that it is not possible to construct a method of order three with fewer than three stages and, in general, we need at least s stages to construct a method of order s.

Lemma 3. *A s-stage sEPIRK-type method cannot achieve order higher than s.*

Proof. To prove this lemma it is enough to show that the order conditions corresponding to a method of order $s+1$ cannot be solved. We begin by showing that it is not possible to construct a two-stage sEPIRK method of order three due to the lack of an elementary differential $N'N'N$ in the expansion of the numerical solution. In the Taylor expansion of the exact solution (2.13) the third-order term $N'N'N$ arises with the coefficient $1/6$. Recall that a two-stage sEPIRK method will have the form

$$\begin{aligned} Y_1 &= y_0 + a_{11}\psi_1(g_{11}hL)h(Ly_0 + N(y_0)) \\ y_1 &= y_0 + b_1\psi_1(g_{21}hL)h(Ly_0 + N(y_0)) + b_2\psi_2(g_{22}hL)h\Delta N(y_0) \end{aligned}, \tag{2.20}$$

where $\Delta N(y_0) = N(Y_1) - N(y_0)$. Expanding the approximate solution in a bicolored B-series we find that the elementary differential with the highest number of black nodes comes from the term $\psi_2(g_{22}hL)h\Delta N(y_0)$. However, this term consists of a stack of only two black nodes. Thus the expansion of the numerical solution lacks the third-order term $N'N'N$ and the order condition corresponding to this term in the exact solution cannot be solved.

Now let us assume the expansion of a $(s-1)$-stage sEPIRK type method does not contain a term involving the elementary differential $\underbrace{N'\cdots N'}_{s-1}N$. In other words, this method lacks the purely black tree τ consisting of a stack of s black nodes. By adding an additional stage and Taylor expanding the resulting numerical solution, we observe that the elementary differentials expansion of the $(s-1)$-stage method gets multiplied by only one additional term N' which comes from the term $\psi_{(s-1)}(g_{(s-1)(s-1)}hL)\Delta^{(s-1)}N(y_0)$. Since the only way to obtain the purely black tree consisting of a stack of $s+1$ black nodes is by multiplying τ by N' we have that the expansion of an s-stage sEPIRK type method will lack the term involving the elementary differential $\underbrace{N'\cdots N'}_{s}N$. □

Note that if we replace the last term of the last stage in (2.20) with

$$\psi_2(g_{22}hf')\Delta N(y_0) = \psi_2(g_{22}h(L+N'))\Delta N(y_0),$$

we are able to construct a "hybrid" EPIRK-type method (i.e. a method which has both: the ψ-functions with an argument dependent on the linear operator L as well as the ψ-functions with the full Jacobian f' in the argument) for which we can have order greater than two. For instance,

$$\begin{aligned} Y_1 &= y_0 + a_{11}\psi_1(g_{11}hL)h(Ly_0 + N(y_0)) \\ y_1 &= y_0 + b_1\psi_1(g_{21}hL)h(Ly_0 + N(y_0)) + b_2\psi_2(g_{22}h(L+N'))h\Delta N(y_0) \end{aligned}, \tag{2.21}$$

with coefficients

$$\begin{bmatrix} a_{11} & \\ b_1 & b_2 \end{bmatrix} = \begin{bmatrix} \frac{2}{3p_{11}} & \\ \frac{1}{p_{11}} & \frac{3}{2(2p_{21}+p_{22})} \end{bmatrix}, \begin{bmatrix} g_{11} & \\ g_{21} & g_{22} \end{bmatrix} = \begin{bmatrix} \frac{2}{3} & \\ 1 & \frac{2p_{21}+p_{22}}{3p_{21}+p_{22}} \end{bmatrix}, \begin{bmatrix} p_{11} & \\ p_{21} & p_{22} \end{bmatrix} = \begin{bmatrix} p_{11} & \\ p_{21} & p_{22} \end{bmatrix}$$

is a third-order two-stage method.

2.4 Numerical results

In this section we address the performance of the new third-order split EPIRK methods using the suite of test problems from [38] which we describe in Section 2.4 for the reader's convenience. In addition to verifying the theoretically expected order of the newly derived schemes we also demonstrate the following important features of their computational performance. First, we address the question of comparative performance between the split and the non-split EPIRK methods and find that the most efficient scheme can be either a split or a non-split EPIRK integrator depending on the properties of the spectrum of the Jacobian and its linear part. Second, we illustrate how optimizing g-coefficients for a given problem can bring computational savings and help develop a scheme that performs particularly well for this problem. Overall, the numerical experiments show that the EPIRK framework provides many options for designing tailored efficient EPIRK schemes for a specific problem.

Test problems

For the numerical experiments we use the same set of test problems as in [38]. Our computations are performed at two different sizes of spatial discretization (except for Burgers equation which we instead solve for different diffusion coefficients). In all of the problems presented below the ∇^2 term is discretized using the standard second order finite differences.

Allen-Cahn 2D. Two-dimensional Allen-Cahn equation [54]:

$$u_t = \alpha\nabla^2 u + u - u^3, \ x, y \in [-1, 1], t \in [0, 1.0]$$

with $\alpha = 0.1$, using no-flow boundary conditions and initial conditions given by $u = 0.1 + 0.1\cos(2\pi x)\cos(2\pi y)$.

Brusselator 2D. Two-dimensional Brusselator problem [37, 20]:

$$\begin{aligned} u_t &= 1 + uv^2 - 4u + \alpha\nabla^2 u, \ x, y \in [0, 1], t \in [0, 0.1], \\ v_t &= 3u - u^2 v + \alpha\nabla^2 v, \end{aligned}$$

with $\alpha = 0.2$. We used Dirichlet boundary conditions with initial and boundary values given by

$$\begin{aligned} u &= 1 + \sin(2\pi x)\sin(2\pi y), \\ v &= 3. \end{aligned}$$

Burgers. One-dimensional Burgers equation:

$$u_t + uu_x = \nu u_{xx},\ x \in [0,1], t = [0,1]$$

with $\nu = 3e-6$, $\nu = 0.03$, $\nu = 0.3$ and with Dirichlet boundary conditions and initial and boundary values given by $u = (\sin(3\pi x))^3(1-x)^{3/2}$. The uu_x term was discretized as

$$uu_x = \frac{u_{i+1}^2 - u_{i-1}^2}{4\Delta x},\ i = 1, ..., N$$

where N is the number of spatial grid points chosen for the problem.

Gray-Scott 2D. Two-dimensional Gray-Scott problem [18]:

$$\begin{aligned} u_t &= d_u\nabla^2 u - uv^2 + a(1-u),\ x, y \in [0,1], t \in [0, 0.1], \\ v_t &= d_v\nabla^2 v + uv^2 - (a+b)v, \end{aligned}$$

with $d_u = 0.2$, $d_v = 0.1$, $a = 0.04$, and $b = 0.06$. Periodic boundary conditions were used and the initial conditions were given by

$$\begin{aligned} u &= 1 - e^{-150(x-\frac{1}{2})^2+(y-\frac{1}{2})^2}, \\ v &= e^{-150(x-\frac{1}{2})^2+2(y-\frac{1}{2})^2}. \end{aligned}$$

ADR 2D. Two-dimensional advection-diffusion-reaction equation [8]:

$$u_t = \epsilon(u_{xx} + u_{yy}) - \alpha(u_x + u_y) + \gamma u(u - \tfrac{1}{2})(1-u),\ x, y \in [0,1], t \in [0, 0.1],$$

where $\epsilon = 1/100$, $\alpha = -10$, and $\gamma = 100$. Homogeneous Neumann boundary conditions were used and the initial conditions were given by $u = 256(xy(1-x)(1-y))^2 + 0.3$.

Verification of accuracy

We verified the accuracy of the newly constructed schemes as follows. To provide a comparative baseline we also include the following methods: (i) the three-stage fourth-order EPIRK4 [65], (ii) the three-stage fifth-order EPIRK5-P1 [69], (iii) the fourth-order Exp4 [24], and (iv) the fourth-order ERow4 [29]. For convenience the coefficients for EPIRK4 and EPIRK5-P1 are given in Table 2.7. Note that while most of the higher order exponential methods (i)-(iii) were constructed using classical order conditions, the scheme ERow4 was derived from the stiff order conditions.

The implementation of the exponential integrators was done in MATLAB. For all integrators, we use the Krylov subspace projection method coupled with the Padé approximation algorithm of Higham [22] to compute the products of the matrix φ-functions and vectors as described in [38]. To verify the order of the schemes we have to choose integration parameters so that the Krylov approximation error does not dominate the time stepping error. To meet this requirement we had

$$\textbf{EPIRK4}\ \begin{bmatrix} p_{11} & & \\ p_{21} & p_{22} & \\ p_{31} & p_{32} & p_{33} \end{bmatrix} = \begin{bmatrix} 1 & & \\ 0 & 1 & \\ 0 & -1 & 6 \end{bmatrix}$$

$$\begin{bmatrix} a_{11} & & \\ a_{21} & a_{22} & \\ b_1 & b_2 & b_3 \end{bmatrix} = \begin{bmatrix} \frac{27(s^2+18)}{12(54-3s^2+2s^3)} & 0 & \\ \frac{18s(s^2+18)}{24(54-3s^2+2s^3)} & \frac{96(54-s^2)(54-3s^2+2s^3)^2}{729*s^2+18)^3} & \frac{384(54-2s^2+2s^3)^2}{162(s^2+18)^3} \\ 1 & & \end{bmatrix}, \quad s = \sqrt{30}$$

$$\begin{bmatrix} g_{11} & & \\ g_{21} & g_{22} & \\ g_{31} & g_{32} & g_{33} \end{bmatrix} = \begin{bmatrix} 1/3 & & \\ 2/3 & 2/3 & \\ 1 & 1 & 1 \end{bmatrix}$$

$$\textbf{EPIRK5-P1}\ \begin{bmatrix} p_{11} & & \\ p_{21} & p_{22} & \\ p_{31} & p_{32} & p_{33} \end{bmatrix} = \begin{bmatrix} 1 & & \\ 1 & 0 & \\ 0 & 0 & 1 \end{bmatrix}$$

$$\begin{bmatrix} a_{11} & & \\ a_{21} & a_{22} & \\ b_1 & b_2 & b_3 \end{bmatrix} = \begin{bmatrix} 0.3512959269505819 & & \\ 0.8440547201165712 & 1.6905891609568963 & \\ 1 & 1.27271273173568923 & 2.271459926542262227 \end{bmatrix}$$

$$\begin{bmatrix} g_{11} & & \\ g_{21} & g_{22} & \\ g_{31} & g_{32} & g_{33} \end{bmatrix} = \begin{bmatrix} 0.3512959269505819 & & \\ 0.8440547201165712 & 1 & \\ 1 & 0.71111095364366870 & 0.62378111953371494 \end{bmatrix}$$

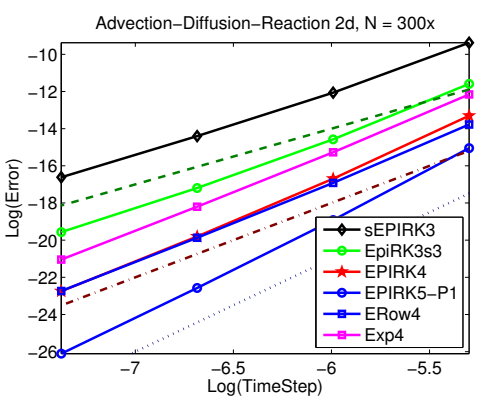
Table 2.7: Coefficients for EPIRK methods

to set the tolerance for all except the Brusselator problems at 10^{-12} and for the Brusselator problem to 10^{-16}. A reference solution was computed using MATLAB's *ode15s* integrator with absolute and relative tolerances set to 10^{-14} and the error was defined as the discrete infinity (maximum) norm of the difference between the computed solution and this approximation.

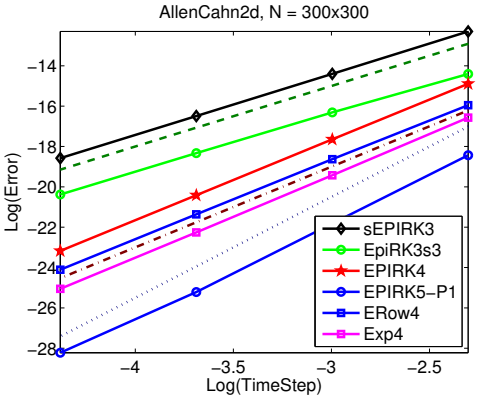
Figure 2.4 shows the achieved order for all of the compared integrators for all of the test problems. As we can see from Figures 2.4(a-b) order three is, in fact, attained by the schemes. As one would expect, for stiffer problems an order reduction phenomenon is observed (Fig. 2.4(c-e)). The order reduction is slight for Burgers and Gray-Scott problems (Fig. 2.4c,d) and more severe for the Brusselator system (Fig 2.4e). Note that even the stiffly accurate ERow4 scheme is experiencing some order reduction particularly for the Brusselator problem. Overall the comparisons show that the new EPIRK methods perform as expected from the perspective of accuracy.

Comparative performance of split vs. non-split EPIRK schemes.

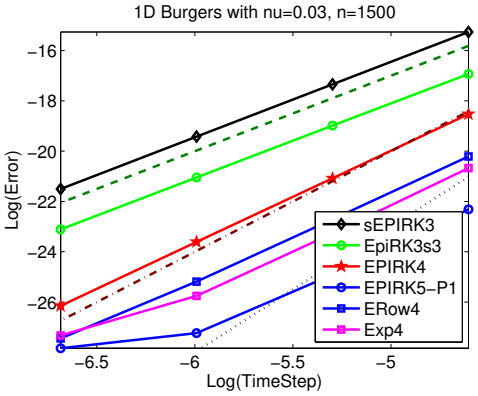
The following numerical experiments are designed to evaluate the performance of the new split methods and compare them with the non-split EPIRK schemes. We have performed simulations



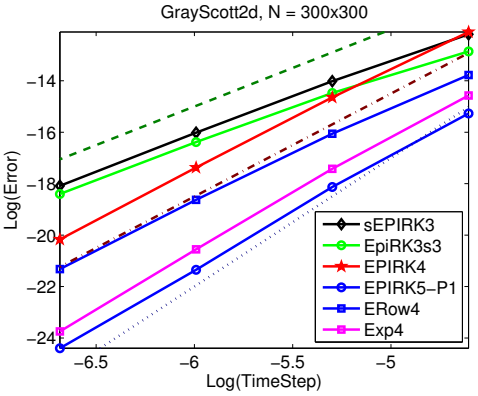
(a) ADR, $N = 300^2$



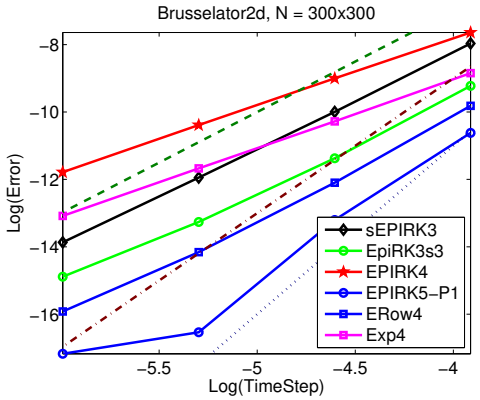
(b) Allen-Cahn, $N = 300^2$



(c) Burgers, $N = 1500$, $\nu = 0.03$



(d) GrayScott, $N = 300^2$



(e) Brusselator, $N = 300^2$

Figure 2.4: Log-log plots of the error versus the time step size. For convenience the lines with slopes equal to three (dashed), four (dash-dotted), and five (dotted) are shown.

of all test problems using sEPIRK3 and EPIRK3s3 with the following parameter values for the spatial discretization and the time step size h:

- *ADR:* $N = 150^2$ & $N = 300^2$ with $h = 0.01, 0.005, 2.5e^{-3}, 1.25e^{-3}, 6.25e^{-4}$
- *Allen-Cahn:* $N = 150^2$ & $N = 300^2$ with $h = 0.1, 0.05, 0.025, 0.0125, 6.25e^{-3}$
- *Brusselator:* $N = 150^2$ & $N = 300^2$ with $h = 0.02, 0.01, 0.005, 2.5e^{-3}, 1.25e^{-3}$
- *Gray-Scott:* $N = 150^2$ & $N = 300^2$ with $h = 0.02, 0.01, 0.005, 2.5e^{-3}, 1.25e^{-3}$
- *Burgers:* $N = 1500$: $\nu = 0.3$ & $\nu = 0.03$ with $h = 0.01, 0.005, 2.5e^{-3}, 1.25e^{-3}, 6.25e^{-4}$ & $\nu = 3e-6$ with $h = 5e^{-4}, 2.5e^{-4}, 1.25e^{-4}, 6.25e^{-5}, 3.125e^{-5}$.

Figures 2.5 and 2.6 show the precision diagrams for all simulations (note that the axes scales are different in each of the graphs and the comparative performance varies in terms of relative scales). As we can see for the majority of the tests the performance of the two schemes is fairly comparable with the split method performing slightly better for the Gray-Scott problem (over all time step sizes the minimum advantage of the split method is at 13% of performance and the maximum advantage is at 36%), the Burgers equation with $\nu = 3e-6$ (minimum $\sim$ 18%, maximum $\sim$ 39%), ADR (minimum $\sim$ 0%, maximum $\sim$ 6%), and Allen-Cahn equation (minimum $\sim$ 0%, maximum $\sim$ 11%). The non-split scheme exhibits a performance advantage in the Brusselator problem (minimum advantage of the non-split scheme is at 1%, and maximum is at 10%), the Burgers equation with $\nu = 0.3$ (minimum $\sim$ 4%, maximum $\sim$ 20%), and $\nu = 0.03$ (minimum $\sim$ 0%, maximum $\sim$ 21%). Note that some of the graphs have a "C" shape, this is due to a quadratic in the number of Krylov vectors complexity of the Krylov projection algorithm. When the Krylov projection is used to approximate $\psi(hA)v$, the complexity of the Krylov subspace projection grows as h increases and sometimes can overshadow the savings achieved by taking a larger time step h. This phenomenon is also characteristic of the implicit Newton-Krylov methods and in the context of exponential integrators has been extensively studied in [38].

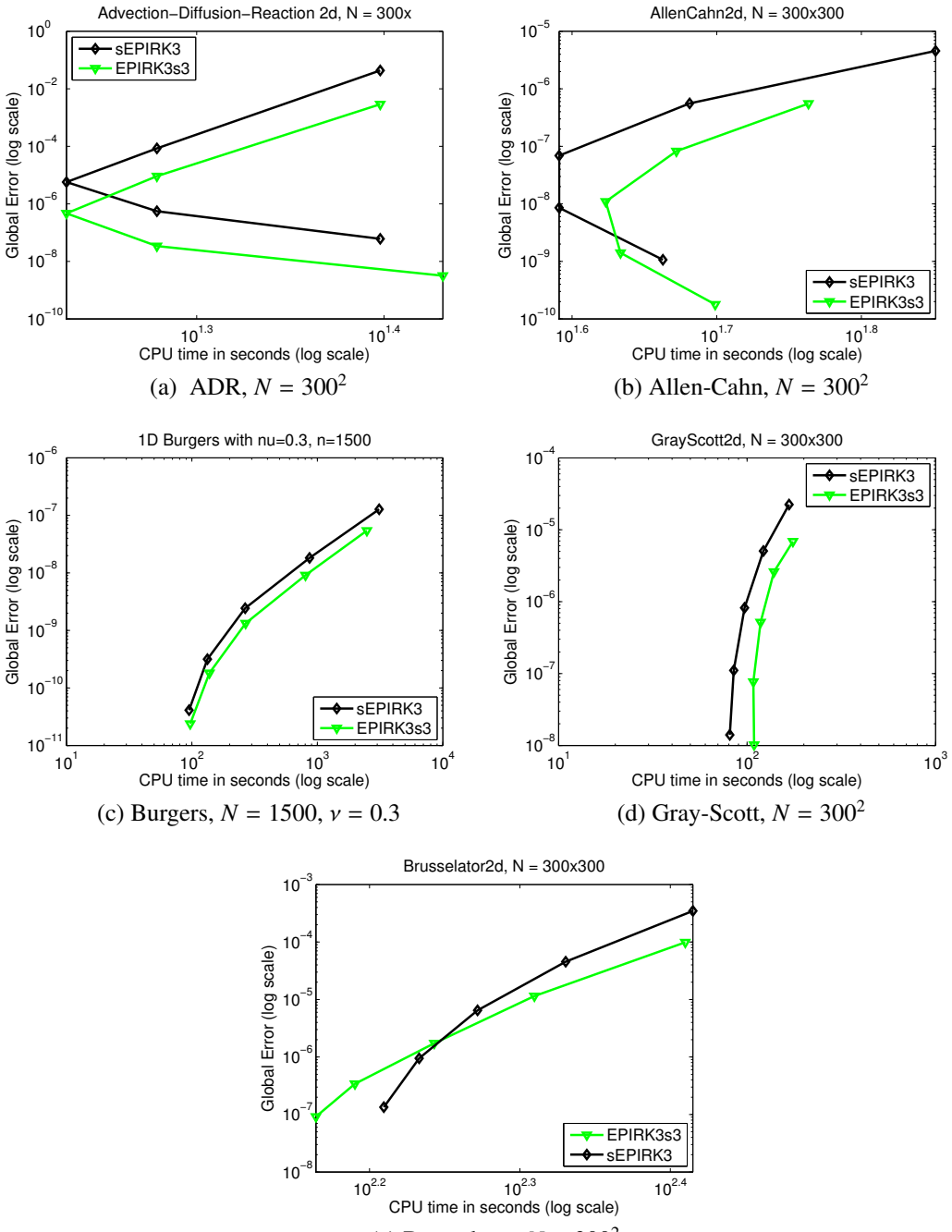
Since the main computational cost of an exponential integrator lies in the evaluations of the matrix φ-functions and vectors products, i.e. in our case the Krylov projections used to estimate $\varphi(hA)v$, we expect the computational efficiency to be primarily controlled by the number of Krylov vectors needed to estimate these products for $A = L$ for split methods and $A = f'(y(t))$ for the non-split schemes. Thus the differences between the spectra of these matrices should mostly account for the differences in the performance of a corresponding method. Computing the spectra of all of the matrices, we find that for most of the problems the Jacobian matrix has a very similar spectrum to the linear part L. Accordingly, we see that the performances of the split and non-split methods are close with the non-split methods yielding a more accurate solution. The exception is the Burgers equation ($\nu = 3e-6$) where the Jacobian has eigenvalues with significant non-zero imaginary component, and thus its spectrum is different from the eigenvalues of the linear part which are pure real negative numbers. While this difference does not cause the non-split method to perform significantly slower for a given size of a time step, it still wins in accuracy. Therefore overall the non-split method performs better for the Burgers equation. Note that for the Gray-Scott problem

we see, albeit less significant, but the reverse comparative performance, i.e. the presence of the purely imaginary eigenvalues in the spectrum of the Jacobian causes enough degradation of the computational efficiency of the Krylov projection for the non-split method to yield the performance advantage to the split scheme. Our results confirm expectations that while non-split methods offer more accuracy in the approximate solution, it is the details of the differences in spectra between L and $f'(y(t))$ that make the split or non-split methods advantageous overall. In other words, the choice between a split or a non-split exponential method completely depends on a given problem.

The effect of the choice of g coefficients on the performance

In Section 2.3 we derived two third-order split EPIRK schemes with different coefficients g_{ij}. One of these methods, the sEPIRK3b, was obtained by minimizing g_{32} coefficient, while another, sEPIRK3, was derived by minimizing g_{33} (Table 2.8). Figures 2.8 and 2.9 show the performance comparison of the two methods. It can be seen that sEPIRK3 performs better for the Brusselator, Burgers, and Gray-Scott problems while sEPIRK3b shows a slight performance advantage for the ADR and Allen-Cahn problems. These results demonstrate that just by adapting the g_{ij} coefficients we can achieve significantly better efficiency for a given problem (e.g. note two orders of magnitude improvement in accuracy for the Brusselator problem for the same computational cost - Fig. 2.8e).

We want to note that this idea of optimizing g_{ij} coefficients can be applicable to other classes of exponential integrators. In particular, since we showed above that each split EPIRK scheme has an associated non-split EPIRK method (with the same coefficients satisfying order conditions for both schemes), we could illustrate the improvements in performance achieved by these optimized non-split EPIRK integrators of third order. To do so, we compare the third-order optimized EPIRK methods with previously derived non-split integrators of higher order [65, 69]. The precision diagrams in Figure 2.10 indicate that for certain problems (ADR, Brusselator, and Gray-Scott) the third-order EPIRK methods may, in fact, be preferable, if we want to achieve a given accuracy for the lowest computational cost. Since it is the case that all EPIRK methods with three stages must have $g_{31} = 1$, then optimizing the g_{ij} coefficients does not effect the computational cost of the first Krylov projection. Thus it is expected that the computational advantage of minimizing g_{ij} coefficients can be realized only for problems where the first Krylov projection does not account for the majority of the overall cost and the subsequent Krylov projections are costly. For these problems the third-order EPIRK methods with minimized g_{ij} coefficients can be more efficient then the higher order methods. In Figure 2.10 for the ADR, Gray-Scott, and Brusselator problems we can see this advantage of the third-order methods. For the Burgers and Allen-Cahn problems the first projection does indeed dominate the computational cost and therefore the higher order methods with the same number of stages retain a computational advantage since they offer more accuracy for the same computational cost. These comparisons illustrate that the idea of optimizing g_{ij} coefficients can be applied to other classes of exponential integrators as well as to specific problems.



(a) ADR, $N = 300^2$

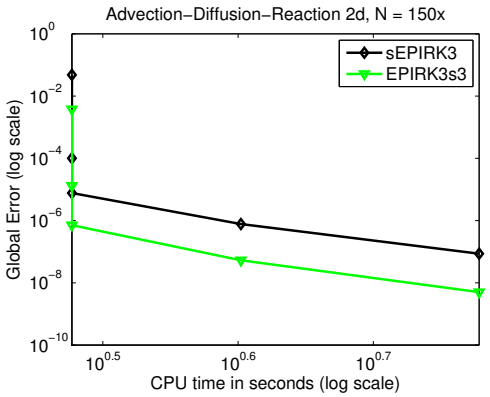
(b) Allen-Cahn, $N = 300^2$

(c) Burgers, $N = 1500$, $\nu = 0.3$

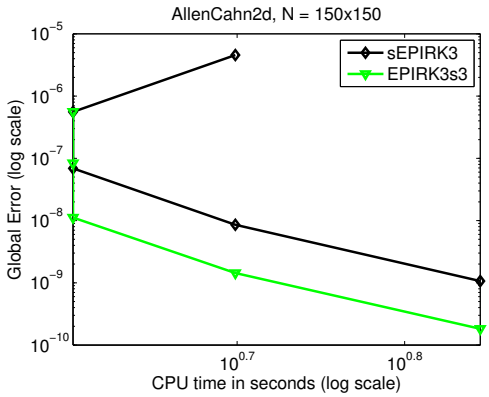
(d) Gray-Scott, $N = 300^2$

(e) Brusselator, $N = 300^2$

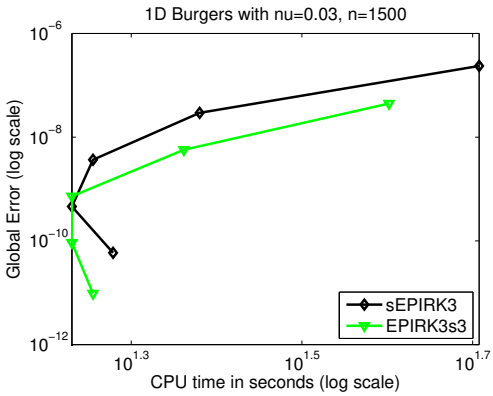
Figure 2.5: Precision diagrams. Note that the axes scales are different in each graph. To generate the graphs we used the step sizes $h = 0.02, 0.01, 0.005, 2.5e^{-3}, 1.25e^{-3}$ for Brusselator, & Gray-Scott problems, $h = 0.01, 0.005, 2.5e^{-3}, 1.25e^{-3}, 6.25e^{-4}$ for ADR & Burgers problems, and $h = 0.1, 0.05, 0.025, 0.0125, 6.25e^{-3}$ for Allen-Cahn problem.



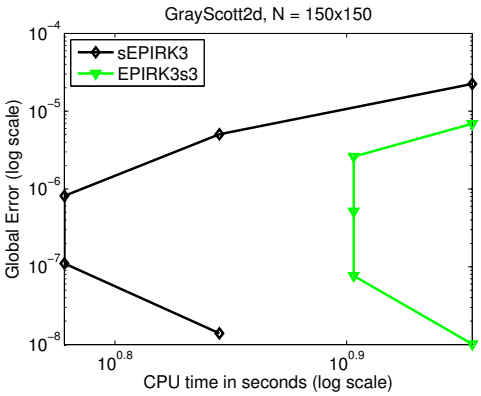
(a) ADR, $N = 150^2$



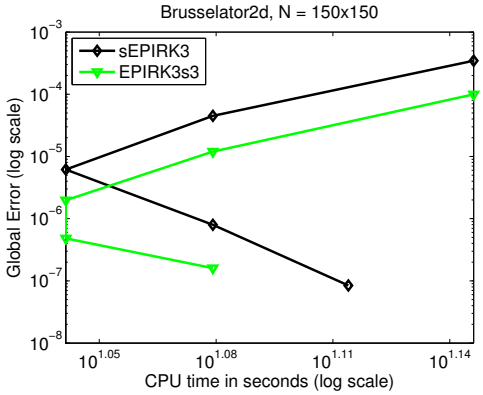
(b) Allen-Cahn, $N = 150^2$



(c) Burgers, $N = 1500$, $\nu = 0.03$

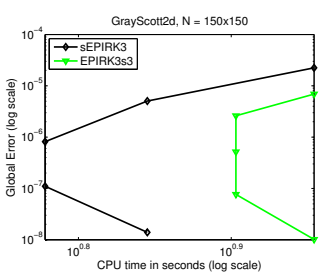


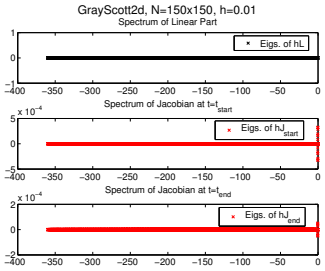
(d) Gray-Scott, $N = 150^2$



(e) Brusselator, $N = 150^2$

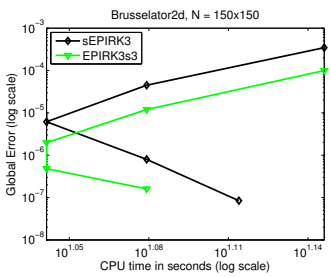
Figure 2.6: Precision diagrams. Note that the axes scales are different in each graph. To generate the graphs we used the step sizes $h = 0.02, 0.01, 0.005, 2.5e^{-3}, 1.25e^{-3}$ for Brusselator, & Gray-Scott problems, $h = 0.01, 0.005, 2.5e^{-3}, 1.25e^{-3}, 6.25e^{-4}$ for ADR & Burgers problems, and $h = 0.1, 0.05, 0.025, 0.0125, 6.25e^{-3}$ for Allen-Cahn problem.

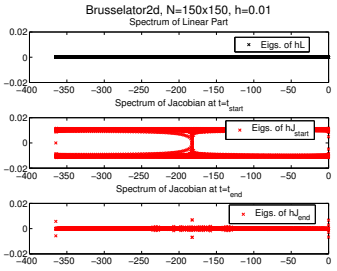




	Avg. Eig. Dist.	Spectral Radius
L	94.0574101	360
J_{start}	94.0572890	360
J_{end}	94.0573161	360
J_{avg}	94.0573025	360

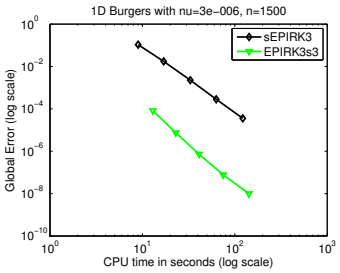
(a) Gray-Scott $N = 150^2$

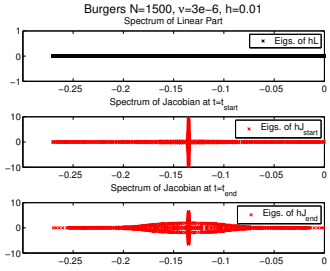




	Avg. Eig. Dist.	Spectral Radius
L	104.1107382	364.7770460
J_{start}	104.1107427	364.7400522
J_{end}	104.1107362	364.7378953
J_{avg}	104.1107395	364.7389737

(b) Brusselator $N = 150^2$





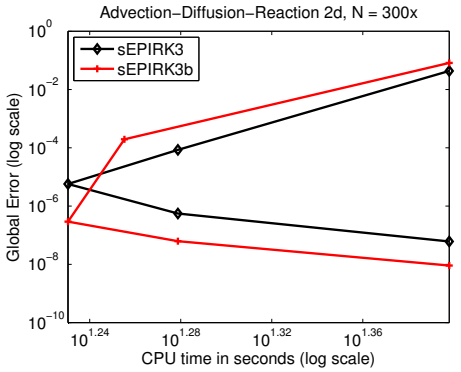
	Avg. Eig. Dist.	Spectral Radius
L	0.1095387	0.2703595
J_{start}	3.0759742	19.9181463
J_{end}	2.8712787	14.8195093
J_{avg}	2.9736265	17.3688278

(c) Burgers, $N = 1500$, $\nu = 3e-6$

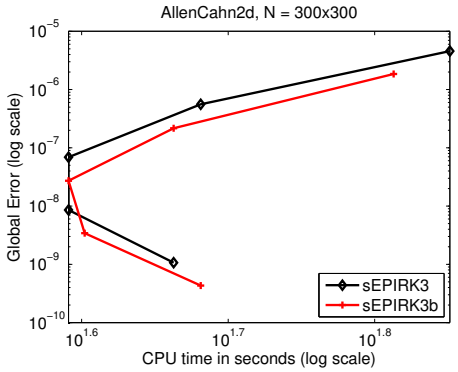
Figure 2.7: Precison diagrams and spectra

	Krylov Proj. 1			Krylov Proj. 2		Krylov Proj. 3
	g_{11}	g_{21}	g_{31}	g_{22}	g_{32}	g_{33}
sEPIRK3 / EPIRK3s3	0.672	0.989	1.000	0.293	0.293	0.020
sEPIRK3b / EPIRK3s3b	1.000	0.226	1.000	0.024	0.024	0.400
EPIRK4	0.333	0.667	1.000	0.667	1.000	1.000
EPIRK5-P1	0.351	0.844	1.000	1.000	0.711	0.624

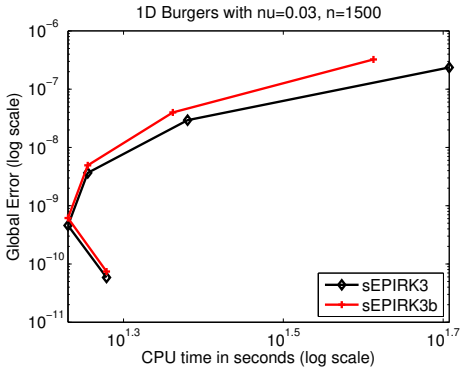
Table 2.8: Approximate values of g_{ij} coefficients for the EPIRK methods being compared.



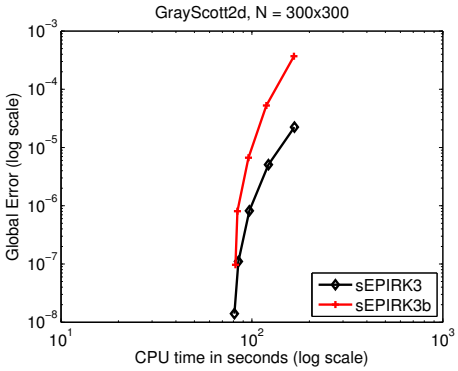
(a) ADR 2d, $N = 300^2$



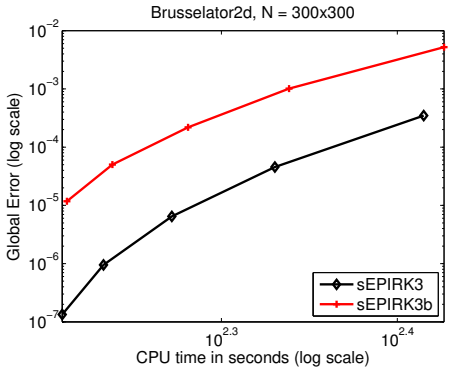
(b) Allen-Cahn, $N = 300^2$



(c) Burgers 1d, $N = 1500$, $\nu = 0.03$

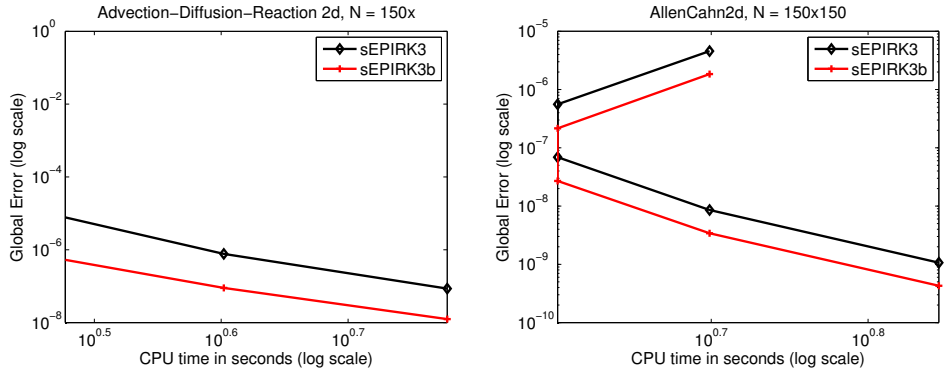


(d) Gray-Scott 2d, $N = 300^2$



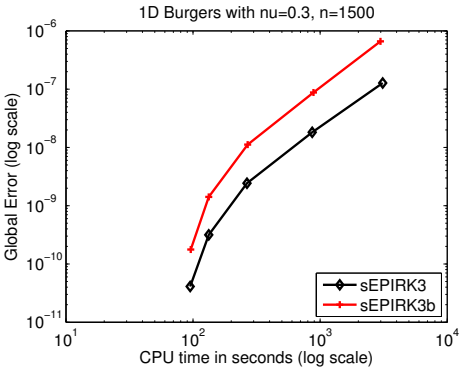
(e) Brusselator 2d, $N = 300^2$

Figure 2.8: Precision diagrams. Note that the axes scales are different in each graph. To generate the graphs we used the step sizes $h = 0.02, 0.01, 0.005, 2.5e^{-3}, 1.25e^{-3}$ for the Brusselator, & Gray-Scott problems, $h = 0.01, 0.005, 2.5e^{-3}, 1.25e^{-3}, 6.25e^{-4}$ for the ADR & Burgers problems, and $h = 0.1, 0.05, 0.025, 0.0125, 6.25e^{-3}$ for the Allen-Cahn problem.

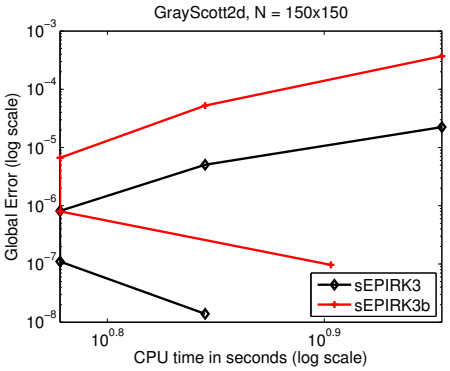


(a) ADR 2d, $N = 150^2$

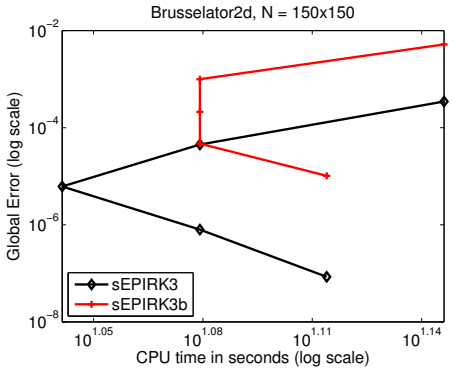
(b) Allen-Cahn, $N = 150^2$



(c) Burgers 1d, $N = 1500$, $\nu = 0.3$

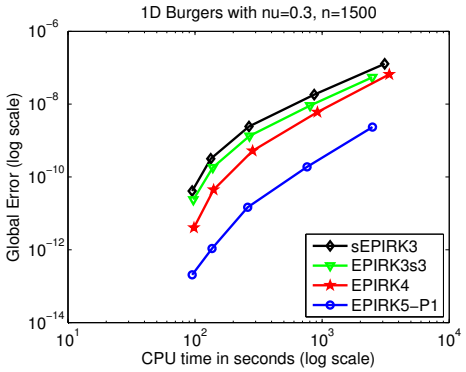


(d) Gray-Scott 2d, $N = 150^2$

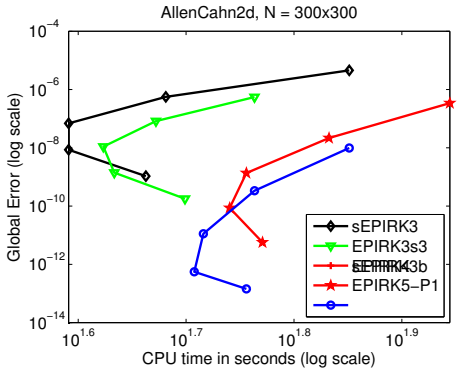


(e) Brusselator 2d, $N = 150^2$

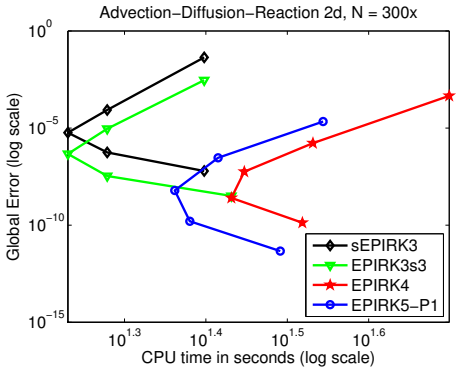
Figure 2.9: Precision diagrams. Note that the axes scales are different in each graph. To generate the graphs we used the step sizes $h = 0.02, 0.01, 0.005, 2.5e^{-3}, 1.25e^{-3}$ for the Brusselator, & Gray-Scott problems, $h = 0.01, 0.005, 2.5e^{-3}, 1.25e^{-3}, 6.25e^{-4}$ for the ADR & Burgers problems, and $h = 0.1, 0.05, 0.025, 0.0125, 6.25e^{-3}$ for the Allen-Cahn problem.



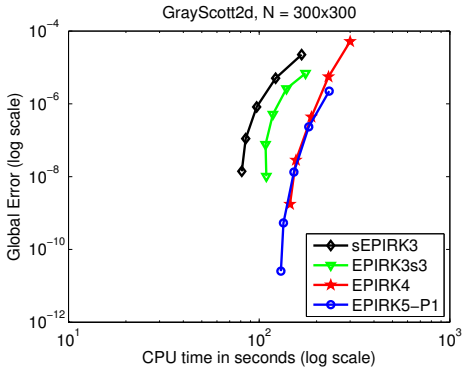
(a) Burgers 1d, $N = 1500$, $\nu = 0.3$



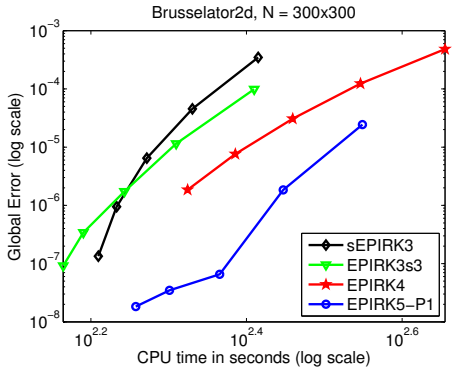
(b) Allen-Cahn, $N = 300^2$



(c) ADR 2d, $N = 300^2$



(d) Gray-Scott 2d, $N = 300^2$



(e) Brusselator 2d, $N = 300^2$

Figure 2.10: Precision diagrams. Note that the axes scales are different in each graph. To generate the graphs we used the step sizes $h = 0.02, 0.01, 0.005, 2.5e^{-3}, 1.25e^{-3}$ for the Brusselator, & Gray-Scott problems, $h = 0.01, 0.005, 2.5e^{-3}, 1.25e^{-3}, 6.25e^{-4}$ for the ADR & Burgers problems, and $h = 0.1, 0.05, 0.025, 0.0125, 6.25e^{-3}$ for the Allen-Cahn problem.

Chapter 3

Preconditioned Implicit-Exponential (IMEXP) Time Integrators for Stiff Differential Equations

3.1 Introduction

In this chapter we address a class of initial value problems which can be written in the form

$$u'(t) = F(u(t)) = L(u(t)) + N(u(t)), \qquad u(t_0) = u_0, \tag{3.1}$$

where both differential operators L and N can be stiff. Often L is a linear operator that represents, for instance diffusion. If N is not a stiff operator, equations of type (3.1) are usually solved using implicit-explicit (IMEX) integrators. IMEX schemes have been widely used in a variety of fields with some of the earlier applications coming from fluid dynamics in conjunction with spectral methods [9, 34]. An example of one of the most widely used, particularly in the context of large-scale applications, IMEX schemes, is the second order BDF-type method (we will call it here 2-sBDF) which was proposed in [3], one of the first publications where IMEX methods were systematically analyzed and derived. Over the past decades a range of IMEX schemes have been introduced such as, for example, linear multistep [3, 32], Runge-Kutta [2] and extrapolated [11] IMEX methods. Such schemes have proven to be very efficient for problems such as advection-diffusion equations or reaction-diffusion equations where advection or reaction are slow and diffusion is occurring on a fast time scale. Diffusion in this case is treated implicitly while advection or reaction terms are treated with an explicit method. The IMEX methods are particularly efficient if a good preconditioner is available to speed up convergence of an iterative method used in the implicit solver. Construction of an efficient preconditioner is the topic of extensive research and software development; frequently the majority of time spent on development and implementation of an IMEX scheme for a large scale problem goes to creating a preconditioner [71]. The complexity of a differential operator that has to be preconditioned is directly related to the difficulty in constructing an efficient preconditioner. For example, a large number of preconditioners have been developed for the Laplacian operator which models linear diffusion process.

The IMEX schemes work well if L is a stiff operator and N is not. However in many applications both of these terms introduce stiffness. Such problems arise from a wide range of fields, from electrochemistry [31] to combustion [5] and plasma physics [68]. A reaction-diffusion system describing chemical kinetic mechanisms involved in ignition of different chemical mixtures can involve thousands of reactions occurring over a wide range of time scales comparable to those of the diffusive processes in the system [5]. Similar structure can be encountered in models of electrochemical material growth where for certain parameter regimes the reactive terms can be as stiff as the diffusive operators in the equations [31]. In magnetohydrodynamic (MHD) equations describing the large scale plasma behavior, stiffness arises from the presence of a wide range of waves encoded in the complex nonlinear terms of the system [17]. While the IMEX or closely-related semi-implicit integrators are typically used for these problems, their performance suffers. The stiffness of the nonlinear operator N which is treated explicitly imposes prohibitive stability restrictions on the time step size. Abandoning the IMEX approach in this case and using a method that treats $N(u(t))$ implicitly as well, also poses a computational challenge. The operator N can be very complex and development of an efficient preconditioner to enable implicit treatment of this term might be difficult or even impossible.

It has been shown that exponential time integrators can offer significant computational savings in comparison to implicit Newton-Krylov methods [24, 66, 30, 28, 69, 46, 44, 40]. However, such comparisons are valid for problems where no efficient preconditioner is available for the implicit Newton-Krylov integrators. A shortcoming of the exponential integrators is that, at present, there are no algorithms that can utilize preconditioners in a way that makes them clearly competitive with the preconditioned implicit Newton-Krylov methods. Here we present a new class of implicit-exponential (IMEXP) methods which can both take advantage of efficient preconditioners developed for given operators and improve computational efficiency for problems where both operators L and N in (3.1) are stiff. The idea of combining an implicit treatment of operator L and an exponential approach to integrate the term with N was first proposed in [67] where a classically accurate second order IMEXP method was constructed. Here we expand this approach to derive several types of IMEXP integrators and provide derivation of stiffly accurate schemes along with the convergence theory for these methods. While we propose two main classes of integrators – IMEXP Runge–Kutta and hybrid IMEXP schemes – the ideas behind these methods can be used to construct many additional schemes that would address particular structure of the problem (3.1).

The chapter is organized as follows. Section 3.2 outlines the main ideas behind construction of IMEXP schemes and presents the analytical framework that enables us to derive the stiff order conditions and to prove stability and convergence of the schemes. Construction and analysis of IMEXP Runge-Kutta methods is presented in Section 3.3 and development of hybrid IMEXP schemes is the focus of Section 3.4. Section 3.5 contains numerical experiments that validate theoretical results and illustrate computational savings that IMEXP methods can bring compared to IMEX schemes for problems with stiff N.

3.2 Construction and analytical framework for analysis of the IMEXP methods.

We begin the construction of IMEXP methods by considering the general EPIRK framework introduced in [64]. The exponential propagation iterative methods of Runge-Kutta type (EPIRK) to solve (3.1) can be written as

$$\begin{aligned} U_{ni} &= u_n + \alpha_{i1}\psi_{i1}(g_{i1}h_nA_{i1})h_nF(u_n) + h_n\sum_{j=2}^{i-1}\alpha_{ij}\psi_{ij}(g_{ij}h_nA_{ij})D_{nj}, \quad i = 2,\dots,s, \\ u_{n+1} &= u_n + \beta_1\psi_{s+1,1}(g_{s+1,1}h_nA_{s+1,1})h_nF(u_n) + h_n\sum_{j=2}^{s}\beta_j\psi_{s+1\,j}(g_{s+1,j}h_nA_{s+1,j})D_{n,s+1}, \end{aligned} \tag{3.2}$$

where u_n is an approximation to the solution of (3.1) at time $t_n = t_0 + \sum_{i=1}^{n} h_i$ and vectors D_{nj} typically depend on the stage values U_{ni}. As explained in [64, 67, 58], different choices for functions ψ_{ij}, matrices A_{ij} and vectors D_{nj} result in different classes of EPIRK methods. To construct IMEXP methods we can use the flexibility of EPIRK framework and choose ψ_{ij}, A_{ij} and D_{nj} to address the structure of the problem (3.1). Namely, we construct methods of two types – IMEXP Runge-Kutta and hybrid IMEXP schemes. The main idea behind both classes of methods is to choose some of the functions ψ_{ij} to be rational functions, similar to implicit or IMEX methods. The remaining ψ_{ij} are then set as linear combination of the exponential-like functions $\varphi_k(z)$ defined by

$$\varphi_0(z) = \mathrm{e}^z, \quad \varphi_k(z) = \int_0^1 \mathrm{e}^{(1-\theta)z}\frac{\theta^{k-1}}{(k-1)!}\,\mathrm{d}\theta, \quad k \geq 1. \tag{3.3}$$

The arguments A_{ij} of these functions can be chosen as either the full Jacobian $J_n = DF(u_n) = F'(u_n) = L'(u_n) + N'(u_n) = L'_n + N'_n$ or its components L' or N'. As noted in [64, 59, 58] the vectors D_{nj} could be node values, differences or forward differences Δ^j constructed using the remainder functions such as $F(u) - F(u_n) - F'(u_n)(u - u_n)$ or its component $N(u)$. In order to simplify the analysis we will restrict our choice to D_{nj} being the difference constructed using the nonlinearity $N(u)$ as explained below. Below we present three classes of methods that are defined by these choices of functions ψ_{ij} and their arguments A_{ij}.

Construction of IMEX Runge-Kutta methods

The IMEXP Runge-Kutta methods are designed for problems of type

$$u'(t) = F(u(t)) = Lu(t) + N(u(t)), \qquad u(t_0) = u_0, \tag{3.4}$$

where L is a stiff linear operator and the nonlinear operator $N(u(t))$ is either nonstiff or mildly stiff compared to L. In this case we choose

$$\psi_{i1}(z) = R_{i1}(z) = \frac{\sum_{k=0}^{m_i} p_{ik} z^k}{\sum_{k=0}^{n_i} q_{ik} z^k}, \quad i = 1, ..., s \tag{3.5}$$

$$\psi_{s+1,1}(z) = R_1(z) = \frac{\sum_{k=0}^{m} p_k z^k}{\sum_{k=0}^{n} q_k z^k}, \tag{3.6}$$

i.e. where $R_{i1}(z)$ and $R_1(z)$ are rational functions. The remaining functions $\psi_{ij}(z)$, $j \geq 2$ are chosen to be linear combinations of $\varphi_k(z)$ functions defined in (3.3). To simplify the notation, and in anticipation of our analysis approach described below, we denote

$$\begin{aligned} a_{ij}(z) &= \alpha_{ij}\psi_{ij}(z), \quad j = 2, ..., i-1, \quad i = 1, ..., s, \\ b_j(z) &= \beta_j \psi_{s+1,j}(z), \quad j = 2, ..., s. \end{aligned} \tag{3.7}$$

Since L is the main source of stiffness in this type of problems we use it as the argument of all functions $\psi_{ij}(z)$. The remainder, or defect, vectors D_{ni} are chosen as

$$D_{ni} = N(U_{ni}) - N(u_n). \tag{3.8}$$

Here we consider the constant time step version of the method and set $h_n = h$. The resulting general form of the IMEXP Runge-Kutta schemes is then given by

$$U_{ni} = u_n + c_i h R_{1i}(c_i hL) F(u_n) + h \sum_{j=2}^{i-1} a_{ij}(hL) D_{nj}, \; 2 \leq i \leq s, \tag{3.9a}$$

$$u_{n+1} = u_n + hR_1(hL)F(u_n) + h \sum_{i=2}^{s} b_i(hL) D_{ni}, \tag{3.9b}$$

$$D_{ni} = N(U_{ni}) - N(u_n). \tag{3.9c}$$

where we have also used the simplifying assumptions

$$\alpha_{i1} = g_{i1} = c_i \tag{3.10a}$$

$$\beta_1 = g_{s+1,1} = 1. \tag{3.10b}$$

(3.10b) is motivated by Lemma 4 in [58] which shows that this requirement is a necessary condition to satisfy the stiff order conditions. As shown in [58], assumption (3.10a) can potentially be relaxed to derive more methods, but we choose it for simplicity of the subsequent analysis.

Construction of hybrid IMEXP methods

We construct the hybrid IMEXP methods to address problems (3.4) where both the operator L and the nonlinearity $N(u(t))$ are stiff. Since the operator N is stiff, it is desirable to treat $N'(u)$ in an

implicit or exponential way. We assume that implicit treatment is difficult in this case due to the lack of a readily available efficient preconditioner which is crucial to making the implicit method sufficiently fast. Therefore we will use the exponential-type approach in treating $N(u(t))$. Thus, we note that for stability reasons the product of the Lipschitz constant of N and the time step must be sufficiently small. To improve the stability of an integrator, one can use the idea of linearizing $F(u)$ continuously along the numerical solution, which can make coefficients of leading error terms become smaller in each integration step. This motivates two possibilities– either we use the full Jacobian

$$J_n = J(u_n) = F'(u_n) = L + N'(u_n) = L + N'_n \tag{3.11}$$

or just its nonlinear part $N'(u_n)$ as an argument for the functions $a_{ij}(z)$ and $b_j(z)$ in (3.7). As before we assume that a rational function of the stiff operator L can be computed efficiently, e.g. due to the availability of an efficient preconditioner. Thus we get a class of hybrid IMEXP methods that can be written as

$$U_{ni} = u_n + c_i hR_{1i}(c_i hL)F(u_n) + h\sum_{j=2}^{i-1} a_{ij}(hJ_n)D_{nj},\ 2 \le i \le s, \tag{3.12a}$$

$$u_{n+1} = u_n + hR_1(hL)F(u_n) + h\sum_{i=2}^{s} b_i(hJ_n)D_{ni} \tag{3.12b}$$

with

$$D_{ni} = N(U_{ni}) - N(u_n), \qquad 2 \le i \le s. \tag{3.12c}$$

If the nonlinear portion of the Jacobian $N'(u)$ is used, we get a modified hybrid IMEXP method

$$U_{ni} = u_n + c_i hR_{1i}(c_i hL)F(u_n) + h\sum_{j=2}^{i-1} a_{ij}(hN'_n)D_{nj},\ 2 \le i \le s, \tag{3.13a}$$

$$u_{n+1} = u_n + hR_1(hL)F(u_n) + h\sum_{i=2}^{s} b_i(hN'_n)D_{ni} \tag{3.13b}$$

with

$$D_{ni} = N(U_{ni}) - N(u_n), \qquad 2 \le i \le s. \tag{3.13c}$$

It is possible that scheme (3.13) can be beneficial in cases where evaluation of $N'(u)v$ for a vector v is significantly less computationally expensive than computing a product of a full Jacobian with a vector $J(u)v$. However, as we will see from the numerical experiments this has to be counterbalanced by some loss of accuracy compared to scheme (3.12)

Analytical framework for derivation of the stiff order conditions and convergence analysis

As for exponential Runge–Kutta methods, the convergence analysis of IMEXP schemes (3.9), (3.12) and (3.13) can be carried out in the abstract framework of strongly continuous semigroups

in a Banach space X with norm $\|\cdot\|$ (for instance, see [13, 56]). As usual in exponential integrators, this framework allows us to handle stiff problems. In particular, throughout this chapter we will make use of the following main assumption.

Assumption 1. The linear operator L is the generator of a strongly continuous semigroup e^{tL} in X.

This assumption implies that there exist constants C and ω such that

$$\| \mathrm{e}^{tL}\|_{X\leftarrow X} \leq C\, \mathrm{e}^{\omega t}, \quad t \geq 0 \tag{3.14}$$

holds uniformly, leading to the boundedness of coefficients $a_{ij}(hL)$ and $b_i(hL)$.

In the subsequent analysis we will investigate questions on the consistency and stability of the proposed IMEXP methods and derive specific schemes.

3.3 IMEXP Runge–Kutta methods

Through out this section the nonlinearity $N(u)$ is supposed to be *nonstiff* with a moderate Lipschitz constant. We thus can make use of the following additional regularity assumption (for instance, see [42]) in order to study the local error of the proposed ansatz (3.9).

Assumption 2. We suppose that (3.1) *possesses a sufficiently smooth solution $u : [0, T] \to X$, with derivatives in X and that $N : X \to X$ is sufficiently often Fréchet differentiable in a strip along the exact solution. All occurring derivatives are assumed to be uniformly bounded in time.*

Assumption 2 implies that N is locally Lipschitz in a strip along the exact solution. Typical examples are certain semilinear reaction-diffusion-advection equations (e.g. in a Banach space of continuous functions), see [21].

Local error analysis

Let $\tilde{u}_n$ denote the exact solution of (3.1) at time t_n, i.e., $\tilde{u}_n = u(t_n)$. Let

$$\bar{e}_{n+1} = \bar{u}_{n+1} - \tilde{u}_{n+1} \tag{3.15}$$

denote the local error of (3.9) at t_{n+1}. To analyze $\bar{e}_{n+1}$, we consider one-step of the integration starting with the initial value $\tilde{u}_n$ being the exact solution.

$$\overline{U}_{ni} = \tilde{u}_n + c_i hR_{1i}(c_i hL)F(\tilde{u}_n) + h\sum_{j=2}^{i-1} a_{ij}(hL)\overline{D}_{nj}, \tag{3.16a}$$

$$\bar{u}_{n+1} = \tilde{u}_n + hR_1(hL)F(\tilde{u}_n) + h\sum_{i=2}^{s} b_i(hL)\overline{D}_{ni} \tag{3.16b}$$

with

$$\overline{D}_{ni} = N(\overline{U}_{ni}) - N(\tilde{u}_n). \tag{3.16c}$$

Taking a closer look at (3.16), one can consider it as a perturbation scheme of the exponential Runge–Kutta scheme

$$\breve{U}_{ni} = \tilde{u}_n + c_i h\varphi_1(c_i hL)F(\tilde{u}_n) + h\sum_{j=2}^{i-1} a_{ij}(hL)\breve{D}_{nj}, \tag{3.17a}$$

$$\breve{u}_{n+1} = \tilde{u}_n + h\varphi_1(hL)F(\tilde{u}_n) + h\sum_{i=2}^{s} b_i(hL)\breve{D}_{ni} \tag{3.17b}$$

with

$$\breve{D}_{ni} = N(\breve{U}_{ni}) - N(\tilde{u}_n). \tag{3.17c}$$

This suggests to us to employ the result on local errors of exponential Runge–Kutta methods (see [47, 43]) for studying the local errors $\bar{e}_{n+1}$ of scheme (3.9).

Denoting $\breve{N}'_{ni} = N'(\breve{U}_{ni})$, $\breve{E}_{ni} = \overline{U}_{ni} - \breve{U}_{ni}$ and using the Taylor series expansion of $N(u)$, we get

$$\overline{D}_{ni} - \breve{D}_{ni} = N(\overline{U}_{ni}) - N(\breve{U}_{ni}) = \breve{N}'_{ni}\breve{E}_{ni} + \breve{R}_{ni} \tag{3.18}$$

with remainder

$$\breve{R}_{ni} = \int_0^1 (1-\theta)N''(\breve{U}_{ni} + \theta\breve{E}_{ni})(\breve{E}_{ni}, \breve{E}_{ni})\mathrm{d}\theta. \tag{3.19}$$

Employing Assumption 2 shows that

$$\|\breve{R}_{ni}\| \le C\|\breve{E}_{ni}\|^2, \tag{3.20}$$

as long as $\breve{E}_{ni}$ remains in a sufficiently small neighborhood of 0.
By subtracting (3.17a) from (3.16a) and denoting $\breve{K}_{ni} = \overline{D}_{ni} - \breve{D}_{ni}$, we get

$$\breve{E}_{ni} = c_i h(R_{1i}(c_i hL) - \varphi_1(c_i hL))F(\tilde{u}_n) + h\sum_{j=2}^{i-1} a_{ij}(hL)\breve{K}_{nj}. \tag{3.21}$$

Inserting (3.21) into (3.18) gives

$$\breve{K}_{ni} = c_i h\breve{N}'_{ni}(R_{1i}(c_i hL) - \varphi_1(c_i hL))F(\tilde{u}_n) + h\breve{N}'_{ni}\sum_{j=2}^{i-1} a_{ij}(hL)\breve{K}_{nj} + \breve{R}_{ni}. \tag{3.22}$$

With this preparation at hand, we are ready to state the following result concerning the stiff order conditions for IMEXP Runge–Kutta methods

Theorem 3.1. *Under Assumptions 1 and 2, an IMEXP Runge–Kutta method* (3.9) *has order of consistency p if its coefficients $a_{ij}(hL), b_i(hL)$ satisfy the stiff order conditions of the exponential Runge–Kutta methods up to some order p and if the coefficients $R_1(hL)$ and $R_{1i}(c_i hL)$ stated in* (3.6) *are chosen in such a way that*

$$(R_{1i}(c_i hL) - \varphi_1(c_i hL))F(\tilde{u}_n) = O(h^{p-1}), \tag{3.23a}$$

$$(R_1(hL) - \varphi_1(hL))F(\tilde{u}_n) = O(h^p). \tag{3.23b}$$

Proof. First, we note that the difference $\check{u}_{n+1} - \tilde{u}_{n+1}$ is the local errors of the exponential Runge–Kutta methods. It is proved in [43] that if the coefficients $a_{ij}(hL), b_i(hL)$ satisfy the stiff order conditions of the exponential Runge–Kutta methods of order p then

$$\check{u}_{n+1} - \tilde{u}_{n+1} = O(h^{p+1}). \tag{3.24}$$

We then express the local errors $\bar{e}_{n+1}$ given by (3.15) as

$$\bar{e}_{n+1} = (\bar{u}_{n+1} - \check{u}_{n+1}) + (\check{u}_{n+1} - \tilde{u}_{n+1}) = \bar{u}_{n+1} - \check{u}_{n+1} + O(h^{p+1}). \tag{3.25}$$

Subtracting (3.17b) from (3.16b) and using (3.18), (3.22) gives us

$$\begin{aligned} \bar{u}_{n+1} - \check{u}_{n+1} &= h(R_1(hL) - \varphi_1(hL))F(\tilde{u}_n) \\ &\quad + h^2 \sum_{i=2}^{s} c_i b_i(hL) \breve{N}'_{ni}(R_{1i}(c_i hL) - \varphi_1(c_i hL))F(\tilde{u}_n) \\ &\quad + h^2 \sum_{i=2}^{s} b_i(hL)(\breve{N}'_{ni} \sum_{j=2}^{i-1} a_{ij}(hL)\breve{K}_{nj} + \breve{R}_{ni}). \end{aligned} \tag{3.26}$$

Under the assumptions in (3.23) and using (3.19)–(3.22), it is straightforward to derive $\breve{K}_{ni} = O(h^p)$, $\breve{R}_{ni} = O(h^{2p})$ and thus we deduce from (3.26) that

$$\bar{u}_{n+1} - \check{u}_{n+1} = O(h^{p+1}). \tag{3.27}$$

From (3.25) and (3.27), it shows at once $\bar{e}_{n+1} = O(h^{p+1})$.

□

Stability and convergence results

Let $e_{n+1} = u_{n+1} - u(t_{n+1}) = u_{n+1} - \tilde{u}_{n+1}$ denote the global error of scheme (3.9) at time t_{n+1}, and let $\overline{E}_{ni} = U_{ni} - \overline{U}_{ni}$. It is easy to see that

$$e_{n+1} = u_{n+1} - \bar{u}_{n+1} + \bar{u}_{n+1} - \tilde{u}_{n+1} = u_{n+1} - \bar{u}_{n+1} + \bar{e}_{n+1}. \tag{3.28}$$

Subtracting (3.16b) from (3.9b) and using $F(u) = Lu + N(u)$ shows that

$$u_{n+1} - \bar{u}_{n+1} = (I + R_1(hL)hL)e_n + hR_1(hL)(N(u_n) - N(\tilde{u}_n)) + h \sum_{i=2}^{s} b_i(hL)\overline{K}_{ni} \tag{3.29}$$

with

$$\overline{K}_{ni} = D_{ni} - \overline{D}_{ni} = (N(U_{ni}) - N(\overline{U}_{ni})) - (N(u_n) - N(\tilde{u}_n)). \tag{3.30}$$

Inserting (3.29) into (3.28) gives

$$e_{n+1} = (I + R_1(hL)hL)e_n + hS_n + \bar{e}_{n+1}, \tag{3.31}$$

where

$$S_n = R_1(hL)(N(u_n) - N(\tilde{u}_n)) + \sum_{i=2}^{s} b_i(hL)\overline{K}_{ni}. \tag{3.32}$$

Next, we consider S_n. Let $\tilde{N}'_n = N'(\tilde{u}_n)$ and $\overline{N}'_{ni} = N'(\overline{U}_{ni})$. Again, we use the Taylor series expansion of $N(u)$ to get

$$N(u_n) - N(\tilde{u}_n) = \tilde{N}'_n e_n + \tilde{r}_n, \tag{3.33a}$$

$$N(U_{ni}) - N(\overline{U}_{ni}) = \overline{N}'_{ni}\overline{E}_{ni} + \overline{R}_{ni} \tag{3.33b}$$

with remainders $\tilde{r}_n$ and $\overline{R}_{ni}$

$$\tilde{r}_n = \int_0^1 (1-\theta)N''(\tilde{u}_n + \theta e_n)(e_n, e_n)\mathrm{d}\theta, \tag{3.34a}$$

$$\overline{R}_{ni} = \int_0^1 (1-\theta)N''(\overline{U}_{ni} + \theta\overline{E}_{ni})(\overline{E}_{ni}, \overline{E}_{ni})\mathrm{d}\theta. \tag{3.34b}$$

Due to Assumption 2 we have

$$\|\tilde{r}_n\| \le C\|e_n\|^2, \quad \|\overline{R}_{ni}\| \le C\|\overline{E}_{ni}\|^2, \tag{3.35}$$

as long as e_n and $\overline{E}_{ni}$ remain in a sufficiently small neighborhood of 0.
Inserting (3.30) into (3.32) and using (3.33), we obtain

$$S_n = \big(R_1(hL) - \sum_{i=2}^{s} b_i(hL)\big)(\tilde{N}'_n e_n + \tilde{r}_n) + \sum_{i=2}^{s} b_i(hL)(\overline{N}'_{ni}\overline{E}_{ni} + \overline{R}_{ni}). \tag{3.36}$$

Lemma 3.3.1. *If Assumptions 1 and 2 hold and functions $R_1(hL)$ and $R_{1i}(c_i hL)$ $(i = 2, \dots, s)$ are chosen in such a way that the following bounds*

$$\|R_1(hL)\|_{X\leftarrow X} \le C, \quad \|R_{1i}(c_i hL)\|_{X\leftarrow X} \le C,$$
$$\|I + R_{1i}(c_i hL)c_i hL\|_{X\leftarrow X} \le C, \tag{3.37}$$

hold, then there exist bounded operators $\mathcal{B}_n(e_n)$ on X such that

$$S_n = \mathcal{B}_n(e_n)e_n. \tag{3.38}$$

Proof. First we derive a recursion for $\overline{E}_{ni}$ by subtracting (3.16a) from (3.9a) and employing (3.33)

$$\begin{aligned}\overline{E}_{ni} &= (I + R_{1i}(c_i hL)c_i hL)e_n + h\big(c_i R_{1i}(c_i hL) - \sum_{j=2}^{i-1} a_{ij}(hL)\big)(\tilde{N}'_n e_n + \tilde{r}_n) \\ &+ h\sum_{j=2}^{i-1} a_{ij}(hL)(\overline{N}'_{nj}\overline{E}_{nj} + \overline{R}_{nj}).\end{aligned} \tag{3.39}$$

Solving recursion (3.39) by induction on the index j with the help of (3.34) and inserting the obtained result into (3.36) yields (3.38). The boundedness of $\mathcal{B}_n(e_n)$ follows from the assumptions of Lemma 3.3.1 and the bounds in (3.35). □

We are now ready to provide sufficient conditions for convergence of IMEXP Runge–Kutta methods (3.9).

Theorem 3.2. *Let the initial value problem* (3.1) *satisfy Assumptions 1–2. Consider for its numerical solution an IMEXP Runge–Kutta method* (3.9) *that fulfills the order conditions of an exponential Runge–Kutta method up to some order p. If coefficients $R_1(hL)$ and $R_{1i}(c_i hL)$ are chosen such that the order conditions in* (3.23) *are fulfilled and the bounds in* (3.37) *and the stability bound*

$$\|(I + R_1(hL)hL\Big)^{n-j}\|_{X\leftarrow X} \leq C_S, \;\; j = 0, \ldots, n-1 \tag{3.40}$$

hold uniformly, then the numerical solution u_n satisfies the error bound

$$\|u_n - u(t_n)\| \leq Ch^p \tag{3.41}$$

uniformly on $t_0 \leq t_n = t_0 + nh \leq T$ with a constant C that depends on $T - t_0$, but is independent of n and h.

Proof. In view of (3.31) and (3.38), we get

$$e_{n+1} = (I + R_1(hL)hL)e_n + h\mathcal{B}_n(e_n)e_n + \bar{e}_{n+1}. \tag{3.42}$$

Solving recursion (3.42) and using $e_0 = 0$ finally yields

$$e_n = h\sum_{j=0}^{n-1}(I + R_1(hL)hL)^{n-j}(\mathcal{B}_j(e_j)e_j + \frac{1}{h}\bar{e}_{j+1}\Big). \tag{3.43}$$

Under the assumptions of Theorem 3.2, we have $\|\mathcal{B}_j(e_j)e_j\| \leq C\|e_j\|$ and $\bar{e}_{j+1} = \mathcal{O}(h^{p+1})$. The proof is completed with the help of the stability bound (3.40) and an application of a discrete Gronwall lemma to (3.43). □

On the choice of rational functions $R_1(Z)$ and $R_{1i}(Z)$

The result of Theorem 3.2 shows the sufficient conditions for which the coefficients $R_1(hL)$ and $R_{1i}(c_i hL)$ need to fulfill for an implicit exponential Runge–Kutta method of order p. In particular, the order conditions (3.23) and the bounds (3.37), (3.40) are the constraints for choosing such coefficients.

First, we focus on conditions (3.23). Our idea is to use Padé approximation (for instance, see [19, Chap. IV]) for finding a rational function $R(z)$ which approximates to $\varphi_1(z)$. Given the fact that $\varphi_1(z) = \dfrac{e^z - 1}{z}$, we employ Padé approximations to the exponential function to derive the following results

$$\varphi_1(z) = (1 - z)^{-1} + \mathcal{O}(z), \tag{3.44a}$$

$$\varphi_1(z) = (1 - \frac{1}{2}z)^{-1} + \mathcal{O}(z^2). \tag{3.44b}$$

Table 3.1: Functions $R_1(Z)$ and $R_{1i}(Z)$ and the corresponding exponential Runge-Kutta stiff order conditions for the methods of orders 1 and 2.

Order p	$R_1(Z)$ and $R_{1i}(Z)$	Stiff order conditions for exponential Runge–Kutta methods
1	$R_1(Z) = (1 - Z)^{-1}$	
2	$R_{1i}(Z) = (1 - Z)^{-1}$ $R_1(Z) = (1 - \frac{1}{2}Z)^{-1}$	$\sum_{i=2}^{s} b_i(Z)c_i = \varphi_2(Z)$

This suggests at once the sought-after functions $R_1(Z)$ and $R_{1i}(Z)$ for methods of possible orders 1 and 2, see Table 3.1. Indeed, we can prove the following result for the Banach space X with the standard inner product denoted by $(\cdot, \cdot)$.

Lemma 3.3.2. *Suppose that the matrix L satisfies*

$$(u, Lu) \leq 0 \text{ for all } u \in X. \tag{3.45}$$

Under Assumptions 1 and 2, we have

$$\|((I - hL)^{-1} - \varphi_1(hL))F(\tilde{u}_n)\| \leq Ch. \tag{3.46}$$

Further assume that $L\dfrac{d}{dt}N(u(t))\Big|_{t=t_n}$ *is uniformly bounded on X. Then, we have*

$$\|((I - \frac{1}{2}hL)^{-1} - \varphi_1(hL))F(\tilde{u}_n)\| \leq Ch^2. \tag{3.47}$$

The constants in (3.46) *and* (3.47) *can be chosen uniformly on* $[t_0, T]$*. Moreover, they are independent of n and h (i.e. independent of* $\|L\|$*).*

Proof. Let $\tilde{v}_n = (I - hL)^{-1}F(\tilde{u}_n)$ and $\tilde{w}_n = (I - \frac{1}{2}hL)^{-1}F(\tilde{u}_n)$. Under condition (3.45), it follows from Theorem 11.2 in [19, Chap. IV] that

$$\|(I - hL)^{-1}\| \leq \sup_{z\leq 0} |\tfrac{1}{1-z}| \leq 1, \ \|(I - \frac{1}{2}hL)^{-1}\| \leq \sup_{z\leq 0} |\tfrac{1}{1-z/2}| \leq 1. \tag{3.48}$$

This reflects the fact that the rational functions $\frac{1}{1-z}$ and $\frac{1}{1-z/2}$ are A-stable. From this it is easy to see that $\tilde{v}_n, \tilde{w}_n$ are uniformly bounded. Using $\tilde{u}_n' = u'(t_n) = F(\tilde{u}_n)$, one gets

$$\tilde{u}_n' = \tilde{v}_n - hL\tilde{v}_n, \tag{3.49a}$$

$$\tilde{u}_n' = \tilde{w}_n - \frac{1}{2}hL\tilde{w}_n, \quad \tilde{u}_n'' = L\tilde{w}_n - \frac{1}{2}hL^2\tilde{w}_n + \frac{\mathrm{d}}{\mathrm{d}t}N(u(t))\Big|_{t=t_n}. \tag{3.49b}$$

We infer from (3.49) that $L\tilde{v}_n, L\tilde{w}_n, L^2\tilde{w}_n$ are uniformly bounded under assumption 2. We now use the following expansion of $\varphi_1(hL)F(\tilde{u}_n)$ (see [47])

$$\varphi_1(hL)F(\tilde{u}_n) = \tilde{u}'_n + \frac{h}{2}(\tilde{u}''_n - 2\varphi_2(hL)\frac{\mathrm{d}}{\mathrm{d}t}N(u(t))\Big|_{t=t_n}) + O(h^2), \tag{3.50}$$

where the remainder terms behind the Landau notation (mutiplying by h^2) are uniformly bounded by the assumptions 1 and 2. Inserting (3.49a) into (3.50) one obtains

$$\varphi_1(hL)F(\tilde{u}_n) = \tilde{v}_n - h(L\tilde{v}_n - \frac{1}{2}\tilde{u}''_n + \varphi_2(hL)\frac{\mathrm{d}}{\mathrm{d}t}N(u(t))\Big|_{t=t_n}) + O(h^2). \tag{3.51}$$

Inserting (3.49b) into (3.50) and using the fact that $\varphi_2(hL) = \frac{1}{2}I + h\varphi_3(hL)L$, we end up with

$$\varphi_1(hL)F(\tilde{u}_n) = \tilde{w}_n - h^2(\frac{1}{4}L^2\tilde{w}_n + \varphi_3(hL)L\frac{\mathrm{d}}{\mathrm{d}t}N(u(t))\Big|_{t=t_n}) + O(h^2). \tag{3.52}$$

It is clear that the coefficients of h in (3.51) and h^2 in (3.52) are uniformly bounded under the conditions of Lemma 3.3.2. Therefore, one derives at once (3.46) and (3.47). □

Remark. Condition (3.45) is often fulfilled for the matrix L, which is resulted from the spatial discretization of a second-order strongly elliptic differential operator (e.g. the Laplacian or the gradient). The assumption of uniform boundedness of $L\frac{\mathrm{d}}{\mathrm{d}t}N(u(t))\Big|_{t=t_n}$ is valid for many semilinear parabolic PDEs such as reaction-diffusion equations, the Allen-Cahn equation (see, e.g [72]) and the Chafee-Infante problem [21, Chap. 5].

Next, we verify whether the chosen rational functions in Table 3.1 satisfy the stability bounds (3.37) and (3.40). Clearly, under condition (3.45) the first two bounds in (3.37) are fulfilled due to (3.48). Similarly, one can see that the third bound (with $R_{1i}(c_ihL) = (I - c_ihL)^{-1}$) and the bound (3.40) (with $R_1(hL) = (I-hL)^{-1}$ or $R_1(hL) = (I-\frac{1}{2}hL)^{-1}$) are also fulfilled because of the following observations

$$\|I + (I - c_ihL)^{-1}c_ihL\| = \|(I - c_ihL)^{-1}\| \le 1, \tag{3.53a}$$

$$\|(I + (I - hL)^{-1}hL)^{n-j}\| \le \|(I - hL)^{-1}\|^{n-j} \le 1, \tag{3.53b}$$

$$\|(I + (I - \frac{1}{2}hL)^{-1}hL\Big)^{n-j}\| \le \|(I - \frac{1}{2}hL)^{-1}(I + \frac{1}{2}hL)\|^{n-j} \le 1. \tag{3.53c}$$

The last inequality in (3.53) holds since $\sup_{z\le 0}|\frac{1+z/2}{1-z/2}| \le 1$ (the rational function $\frac{1+z/2}{1-z/2}$ is A-stable).

Derivation of the first- and second-order methods

Based on the results of Theorem 3.2 and Section 3.3, it is straightforward to derive methods of orders 1 and 2 with the chosen rational functions in Table 3.1 (the convergence of these methods follows directly from Theorem 3.2). In particular, we obtain the following 1-stage IMEXP Runge–Kutta method of order 1 which we will call `ImExpRK1`:

$$u_{n+1} = u_n + h(I - hL)^{-1}F(u_n). \tag{3.54}$$

Using $F(u_n) = Lu_n + N(u_n)$ and the equality $I + (I - hL)^{-1}hL = (I - hL)^{-1}$, one realizes that the method `ImExpRK1` coincides with the implicit-explicit Rung–Kutta method of order 1: $u_{n+1} = u_n + hLu_{n+1} + hN(u_n)$.

As $s = 2$ we take $c_2 = \frac{1}{2}$ and choose from Table 3.1 coefficients $R_{12}(hL) = R_1(hL) = (I - \frac{1}{2}hL)^{-1}$, $b_2(hL) = 2\varphi_2(hL)$. This leads to the following second-order method which we will call `ImExpRK2`:

$$\begin{aligned} U_{n2} &= u_n + \frac{1}{2}h(I - \frac{1}{2}hL)^{-1}F(u_n), \\ u_{n+1} &= u_n + h(I - \frac{1}{2}hL)^{-1}F(u_n) + 2h\varphi_2(hL)(N(U_{n2}) - N(u_n)). \end{aligned} \tag{3.55}$$

A discussion of higher-order methods

We now discuss whether it is possible to derive IMEXP Runge–Kutta methods of higher orders. Clearly, due to (3.23) such methods require higher-order rational approximations of $\varphi_1(hL)F(u_n)$, which also need to satisfy the bounds in (3.37) and the stability bound (3.40). Again, we make use of the Padé approximations to the exponential function to get, for instance,

$$\varphi_1(z) = (1 + \frac{1}{6}z)(1 - \frac{1}{3}z)^{-1} + O(z^3), \tag{3.56a}$$

$$\varphi_1(z) = (1 - \frac{1}{2}z + \frac{1}{12}z^2)^{-1} + O(z^4). \tag{3.56b}$$

This offers, for example, the choice of $R_1(hL) = (1 + \frac{1}{6}hL)(1 - \frac{1}{3}hL)^{-1}$ for methods of order 3. However, we can show that such a method requires much stronger regularity assumptions in order to fulfill (3.23). Construction of higher order methods will be the subject of future publications.

3.4 Hybrid IMEXP

In this section, we consider the case where the stiffness of problem (3.1) comes from both the linear L and nonlinear N operators. Motivated by the results of Section 3.3, we propose the following two perturbation schemes of (3.55)

$$U_{n2} = u_n + \frac{1}{2}h(I - \frac{1}{2}hL)^{-1}F(u_n), \tag{3.57a}$$

$$u_{n+1} = u_n + h(I - \frac{1}{2}hL)^{-1}F(u_n) + 2h\varphi_2(hJ_n)(N(U_{n2}) - N(u_n)) \tag{3.57b}$$

and

$$\begin{aligned} U_{n2} &= u_n + \frac{1}{2}h(I - \frac{1}{2}hL)^{-1}F(u_n), \\ u_{n+1} &= u_n + h(I - \frac{1}{2}hL)^{-1}F(u_n) + 2h\varphi_2(hN'_n)(N(U_{n2}) - N(u_n)), \end{aligned} \tag{3.58}$$

which will be called `HImExp2J` and `HImExp2N`, respectively. In the following we will show that these two schemes are of order 2 as well. First, note that since we are working with the nonlinearity

$N(u)$ which is assumed to have some stiffness, the norms of the Jacobian $N'(u)$ and its higher derivatives could be large so that Assumption 2 is not feasible. Thus, in order to give a convergence result for those two schemes, we will make use of the following weaker, but reasonable, assumption on the solution and the nonlinearity.

Assumption 3. Suppose that (3.1) *possesses a solution that is thrice differentiable with (uniformly) bounded derivatives in X, and that $N : X \to X$ is twice continuously Fréchet differentiable in a strip along the exact solution.*

Under this assumption, $N(u)$ satisfies local Lipschitz continuity in a strip along the exact solution. Moreover, we have the following property of the second-order Fréchet derivative

$$N(u + \Delta u) = N(u) + N'(u)\Delta u + \int_0^1 (1 - s)N''(u + s\Delta u)(\Delta u, \Delta u)\mathrm{d}s. \tag{3.59}$$

Since $N'_n = N'(u_n)$ is a linear bounded operator on X, the Jacobians $J_n = L + N'_n$ and N'_n generate strongly continuous semigroups e^{tJ_n} and $\mathrm{e}^{tN'_n}$(see [56, Chap. 3.1]). Thus one infers from Assumption 1 that e^{hJ_n} and $\mathrm{e}^{hN'_n}$ are uniformly bounded in a neighborhood of the exact solution, and thus so are $\varphi_2(hJ_n)$ and $\varphi_2(hN'_n)$.

Expansion of the exact solution

Using Assumptions 1 and 3, we will derive an expansion of the exact solution of (3.1) at time t_{n+1}, i.e., $u(t_{n+1})$. Again, let $\tilde{u}_n = u(t_n)$. Expressing $u(t_{n+1})$ by the variation-of-constants formula gives

$$\tilde{u}_{n+1} = u(t_{n+1}) = \mathrm{e}^{hL}\tilde{u}_n + h\int_0^1 \mathrm{e}^{(1-\theta)hL}N(u(t_n + \theta h))\,\mathrm{d}\theta. \tag{3.60}$$

Now let $\mathcal{U}_n = u(t_n + \theta h) - \tilde{u}_n$ and let $\tilde{u}'_n, \tilde{u}''_n, \tilde{u}'''_n$ denote the first, second, and third derivative of the exact solution $u(t)$ of (3.1) evaluated at time t_n. Under Assumption 3 we can expand $u(t_n + \theta h)$ in a Taylor series at t_n to get

$$\mathcal{U}_n = \theta h\tilde{u}'_n + \theta^2 h^2 \int_0^1 (1 - s)u''(t_n + \theta h s)\mathrm{d}s. \tag{3.61}$$

Using (3.59) with $u = \tilde{u}_n, \Delta u = \mathcal{U}_n$, one derives

$$N(u(t_n + \theta h)) = N(\tilde{u}_n) + N'(\tilde{u}_n)\mathcal{U}_n + \int_0^1 (1 - s)N''(\tilde{u}_n + s\mathcal{U}_n)(\mathcal{U}_n, \mathcal{U}_n)\mathrm{d}s. \tag{3.62}$$

With the help of (3.61), inserting (3.62) into (3.60) and using (3.3) (with $k = 1, 2$), we eventually obtain an expansion of the exact solution

$$\tilde{u}_{n+1} = \tilde{u}_n + h\varphi_1(hL)F(\tilde{u}_n) + h^2\varphi_2(hL)\tfrac{\mathrm{d}}{\mathrm{d}t}N(u(t))\Big|_{t=t_n} + h^3\mathcal{R}_n \tag{3.63}$$

with

$$\mathcal{R}_n = \int_0^1 \mathrm{e}^{(1-\theta)hL}\int_0^1 \theta^2(1 - s)\big(N'(\tilde{u}_n)u''(t_n + \theta h) + N''(\tilde{u}_n + s\mathcal{U}_n)(\mathcal{V}_n, \mathcal{V}_n)\big)\mathrm{d}s\mathrm{d}\theta, \tag{3.64}$$

where $\mathcal{V}_n = \tilde{u}_n' + \theta h \int_0^1 (1-s)u''(t_n + \theta h s)\mathrm{d}s$. Note that this remainder is not the same as the one in the expansion of $u(t_{n+1})$ given in [42], which requires Assumption 2. It is clear that $\|\mathcal{R}_n\| \leq C$ (uniformly) due to Assumptions 1 and 3.

Local error analysis of schemes `HImExp2J` and `HImExp2N`

Using the expansion of the exact solution in (3.63), we prove the following result concerning the local errors of schemes (3.57) and (3.58) at time $t = t_{n+1}$.

Lemma 3.4.1. *Let the initial value problem* (3.1) *satisfy Assumptions 1 and 3. Further assume that L satisfies the condition* (3.45) *and that* $L\frac{d}{dt}N(u(t))\big|_{t=t_n}$ *is uniformly bounded on X. Then, both schemes* (3.57) *and* (3.58) *have order of consistency 3.*

Proof. First, we study the local error of scheme (3.57), i.e., `HImExp2J`. For that we consider one step with initial value $\tilde{u}_n$, i.e.

$$\overline{U}_{n2} = \tilde{u}_n + \frac{1}{2}h(I - \frac{1}{2}hL)^{-1}F(\tilde{u}_n), \tag{3.65a}$$

$$\bar{u}_{n+1} = \tilde{u}_n + h(I - \frac{1}{2}hL)^{-1}F(\tilde{u}_n) + 2h\varphi_2(h\tilde{J}_n)(N(\overline{U}_{n2}) - N(\tilde{u}_n)). \tag{3.65b}$$

Therefore, the local error of (3.57) at time t_{n+1} is given by

$$\bar{e}_{n+1,\texttt{HImExp2J}} = \bar{u}_{n+1} - \tilde{u}_{n+1}. \tag{3.66}$$

Proceeding in the same manner as in Lemma 1 of [47], we get

$$\begin{aligned} \varphi_1(hL)F(\tilde{u}_n) &= \tilde{u}_n' + \frac{h}{2}(\tilde{u}_n'' - 2\varphi_2(hL)\tfrac{\mathrm{d}}{\mathrm{d}t}N(u(t))\big|_{t=t_n}) \\ &+ h^2\varphi_3(hL)(\tilde{u}_n''' - \tfrac{\mathrm{d}^2}{\mathrm{d}t^2}N(u(t))\big|_{t=t_n}). \end{aligned} \tag{3.67}$$

Clearly, due to Assumptions 1 and 3, we get back (3.50) from (3.67). This shows that the result of Lemma 3.3.2 holds. Using the same notation $\tilde{w}_n = (I - \frac{1}{2}hL)^{-1}F(\tilde{u}_n)$, (3.65) can be rewritten as

$$\bar{u}_{n+1} = \tilde{u}_n + h\tilde{w}_n + 2h\varphi_2(h\tilde{J}_n)(N(\tilde{u}_n + \frac{1}{2}h\tilde{w}_n) - N(\tilde{u}_n)). \tag{3.68}$$

Employing (3.52) gives

$$\tilde{w}_n = \varphi_1(hL)F(\tilde{u}_n) + h^2(\frac{1}{4}L^2\tilde{w}_n + \varphi_3(hL)L\frac{\mathrm{d}}{\mathrm{d}t}N(u(t))\big|_{t=t_n}) + O(h^2).$$

Inserting this into (3.68), applying (3.59) to $N(\tilde{u}_n + \frac{1}{2}h\tilde{w}_n) - N(\tilde{u}_n)$, and noting that $\tilde{w}_n = \tilde{u}_n' + \frac{1}{2}hL\tilde{w}_n$ (see (3.49b)) one finally gets

$$\bar{u}_{n+1} = \tilde{u}_n + h\varphi_1(hL)F(\tilde{u}_n) + h^2\varphi_2(h\tilde{J}_n)\tfrac{\mathrm{d}}{\mathrm{d}t}N(u(t))\big|_{t=t_n} + h^3\overline{\mathcal{R}}_n \tag{3.69}$$

with

$$\begin{aligned}\bar{\mathcal{R}}_n = &\frac{1}{4}L^2\tilde{w}_n + \varphi_3(hL)L\frac{\mathrm{d}}{\mathrm{d}t}N(u(t))\Big|_{t=t_n} \\ &+ \frac{1}{2}\varphi_2(h\tilde{J}_n)\Big(N'(\tilde{u}_n)L\tilde{w}_n + \int_0^1 (1-s)N''(\tilde{u}_n + \frac{1}{2}sh\tilde{w}_n)(\tilde{w}_n,\tilde{w}_n)\mathrm{d}s\Big) + O(h^2),\end{aligned} \tag{3.70}$$

which is uniformly bounded due to the assumptions of Lemma 3.3. Inserting (3.63) and (3.69) into (3.66) gives

$$\bar{e}_{n+1,\texttt{HImExp2J}} = h^2(\varphi_2(h\tilde{J}_n) - \varphi_2(hL))\frac{\mathrm{d}}{\mathrm{d}t}N(u(t))\Big|_{t=t_n} + h^3(\bar{\mathcal{R}}_n - \mathcal{R}_n). \tag{3.71}$$

Using (3.3), we have

$$\varphi_2(h\tilde{J}_n) - \varphi_2(hL) = \int_0^1 (\,\mathrm{e}^{(1-\theta)h\tilde{J}_n} - \,\mathrm{e}^{(1-\theta)hL})\theta\mathrm{d}\theta. \tag{3.72}$$

Next, we use $\tilde{J}_n = L + \tilde{N}'_n$ (with $\tilde{N}'_n = N'(\tilde{u}_n)$) and apply the variation-of-constants formula to the differential equation

$$v'(t) = \tilde{J}_n v(t) = Lv(t) + \tilde{N}'_n v(t),\ v(0) = I$$

(with the exact solution $v(t) = \,\mathrm{e}^{t\tilde{J}_n}$) to get

$$\begin{aligned}\,\mathrm{e}^{t\tilde{J}_n} &= \,\mathrm{e}^{tL} + \int_0^t \,\mathrm{e}^{(t-\xi)L}\tilde{N}'_n v(\xi)\mathrm{d}\xi \\ &= \,\mathrm{e}^{tL} + t\int_0^1 \,\mathrm{e}^{(1-s)tL}\tilde{N}'_n\,\mathrm{e}^{stJ_n}\mathrm{d}s.\end{aligned} \tag{3.73}$$

The last equality in (3.73) is due to the change of variable $\xi = ts$. Inserting (3.73) with $t = (1-\theta)h$ into (3.72) shows that

$$\varphi_2(h\tilde{J}_n) - \varphi_2(hL) = h\phi(hL, h\tilde{N}'_n), \tag{3.74}$$

where

$$\phi(hL, h\tilde{N}'_n) = \int_0^1\int_0^1 \,\mathrm{e}^{(1-s)(1-\theta)hL}\theta(1-\theta)\tilde{N}'_n\,\mathrm{e}^{s(1-\theta)h\tilde{J}_n}\mathrm{d}s\mathrm{d}\theta \tag{3.75}$$

which is a bounded operator. We now insert (3.74) into (3.71) to obtain

$$\bar{e}_{n+1,\texttt{HImExp2J}} = h^3(\phi(hL, h\tilde{N}'_n)\frac{\mathrm{d}}{\mathrm{d}t}N(u(t))\Big|_{t=t_n} + \bar{\mathcal{R}}_n - \mathcal{R}_n) = O(h^3). \tag{3.76}$$

Next, we consider the local error of scheme (3.58), i.e., `HImExp2N`. Let denote the local error of it at time t_{n+1} by $\bar{e}_{n+1,\texttt{HImExp2N}}$. It is easy to see that this local error can be analyzed in a very similar way as done for the scheme `HImExp2J`. Thus we only focus on the following new aspects. Instead of (3.71), we now get

$$\bar{e}_{n+1,\texttt{HImExp2N}} = h^2(\varphi_2(h\tilde{N}'_n) - \varphi_2(hL))\frac{\mathrm{d}}{\mathrm{d}t}N(u(t))\Big|_{t=t_n} + h^3(\bar{\mathcal{R}}_n - \mathcal{R}_n) \tag{3.77}$$

with $\tilde{N}_n'$ in place of $\tilde{J}_n$ appearing in (3.71) as well as in (3.70) (for $\overline{\mathcal{R}}_n$).
First, we observe that

$$\begin{aligned}\varphi_2(h\tilde{N}_n') - \varphi_2(hL) &= (\varphi_2(h\tilde{N}_n') - \varphi_2(h\tilde{J}_n)) + (\varphi_2(h\tilde{J}_n) - \varphi_2(hL)),\\ &= (\varphi_2(h\tilde{N}_n') - \varphi_2(h\tilde{J}_n)) + \phi(hL, h\tilde{N}_n').\end{aligned} \tag{3.78}$$

Here, $\phi(hL, h\tilde{N}_n')$ is given in (3.75). To analyze the term $\varphi_2(h\tilde{N}_n') - \varphi_2(h\tilde{J}_n)$, we perform the similar procedure as done in (3.72)-(3.73). In particular, by again using (3.3), applying the variation-of-constants formula to the differential equation $y'(t) = \tilde{N}_n' y(t) = \tilde{J}_n y(t) - Ly(t)$, $y(0) = I$, and employing the fact that $\mathrm{e}^z = 1 + z\varphi_1(z)$, we finally end up with

$$\varphi_2(h\tilde{N}_n') - \varphi_2(hL) = h\Big(\psi_1(h\tilde{J}_n)L - h\psi_2(h\tilde{J}_n, h\tilde{N}_n') + \phi(hL, h\tilde{N}_n')\Big), \tag{3.79}$$

where

$$\psi_1(h\tilde{J}_n) = \int_0^1 \int_0^1 \mathrm{e}^{(1-\theta)h\tilde{J}_n}\theta(\theta-1)\mathrm{d}s\mathrm{d}\theta \tag{3.80}$$

which is also a bounded operator, and

$$\psi_2(h\tilde{J}_n, h\tilde{N}_n') = \int_0^1 \int_0^1 \mathrm{e}^{(1-s)(1-\theta)h\tilde{J}_n} s\theta(1-\theta)^2 L\tilde{N}_n'\varphi_1(s(1-\theta)h\tilde{N}_n')\mathrm{d}s\mathrm{d}\theta \tag{3.81}$$

is bounded as well (due to the assumption $L\frac{\mathrm{d}}{\mathrm{d}t}N(u(t))\Big|_{t=t_n} = L\tilde{N}_n' u'(t_n)$ is uniformly bounded).

Inserting (3.79) into (3.77) clearly shows that

$$\bar{e}_{n+1,\texttt{HImExp2N}} = h^3(\psi_1(h\tilde{J}_n)L\tfrac{\mathrm{d}}{\mathrm{d}t}N(u(t))\Big|_{t=t_n}) + O(h^3) = O(h^3). \tag{3.82}$$

□

We are now ready to state the main result of this section.

Theorem 3.3. *Let the initial value problem* (3.1) *satisfy the conditions of Lemma 3.4.1. Then, the numerical solution u_n of the hybrid implicit-explicit exponential methods* `HImExp2J` *or* `HImExp2N` *satisfies the error bound*

$$\|u_n - u(t_n)\| \le Ch^2 \tag{3.83}$$

uniformly on $t_0 \le t_n = t_0 + nh \le T$ with a constant C that depends on $T - t_0$, but is independent of n and h.

Proof. First, we prove the convergence result for scheme `HImExp2J`. Let $e_{n+1} = u_{n+1} - u(t_{n+1}) = u_{n+1} - \tilde{u}_{n+1}$ denote the global error of scheme (3.57) at time t_{n+1}. We have

$$e_{n+1} = u_{n+1} - \bar{u}_{n+1} + \bar{e}_{n+1,\texttt{HImExp2J}}. \tag{3.84}$$

Subtracting (3.65b) from (3.57b), inserting the obtained result into (3.84), and using $F(u) = Lu + N(u)$ shows that

$$e_{n+1} = R(hL)e_n + hT_n + \bar{e}_{n+1,\texttt{HImExp2J}} \tag{3.85}$$

with $R(hA) = I + (I - \frac{1}{2}hL)^{-1}hL = (I - \frac{1}{2}hL)^{-1}(I + \frac{1}{2}hL)$ and

$$\begin{aligned} T_n =&((I - \frac{1}{2}hL)^{-1} - 2\varphi_2(h\tilde{J}_n))(N(u_n) - N(\tilde{u}_n)) + 2\varphi_2(hJ_n)(N(U_{n2}) - N(\overline{U}_{n2})) \\ &+2(\varphi_2(hJ_n) - \varphi_2(h\tilde{J}_n))(N(\overline{U}_{n2}) - N(u_n)). \end{aligned} \tag{3.86}$$

Subtracting (3.65a) from (3.57a) and using the identity $I + \frac{1}{2}hL(I - \frac{1}{2}hL)^{-1} = (I - \frac{1}{2}hL)^{-1}$ gives

$$U_{n2} - \overline{U}_{n2} = (I - \frac{1}{2}hL)^{-1}e_n + \frac{1}{2}h(I - \frac{1}{2}hL)^{-1}(N(u_n) - N(\tilde{u}_n)). \tag{3.87}$$

Using the Lipschitz property of $N(u)$ and employing the bound $\|\varphi_2(hJ_n) - \varphi_2(h\tilde{J}_n)\| \leq Ch\|e_n\|$ (as a consequence of Lemma 4.3 in [46]), we derive at once $\|T_n\| \leq C\|e_n\|$ as long as the global errors e_n remain in a sufficiently small neighborhood of 0. Solving recursion (3.85) and using $e_0 = 0$ finally yields

$$e_n = h\sum_{j=0}^{n-1}(R(hL))^{n-j}\Big(T_j + \frac{1}{h}\bar{e}_{n+1,\texttt{HImExp2J}}\Big). \tag{3.88}$$

Now using (3.76) and the bound (3.53c) ($\|R(hL))^{n-j}\| \leq 1$) we can estimate

$$\|e_n\| \leq Ch\sum_{j=0}^{n-1}(\|e_j\| + h^2). \tag{3.89}$$

Again, an application of a discrete Gronwall lemma to (3.89) shows the desired bound (3.83).

Next, because of the similarity between the structure of the two schemes `HImExp2N` and `HImExp2J`, the convergence proof for `HImExp2N` method can be carried out in a very similar way. We thus obtain a representation of the global error of scheme (3.58) at time t_n as

$$e_n = h\sum_{j=0}^{n-1}(R(hL))^{n-j}\Big(\hat{T}_j + \frac{1}{h}\bar{e}_{n+1,\texttt{HImExp2N}}\Big), \tag{3.90}$$

where $\hat{T}_j$ is given by

$$\begin{aligned} \hat{T}_j =&((I - \frac{1}{2}hL)^{-1} - 2\varphi_2(h\tilde{N}_j))(N(u_j) - N(\tilde{u}_j)) + 2\varphi_2(hN_j)(N(U_{j2}) - N(\overline{U}_{j2})) \\ &+2(\varphi_2(hN_j) - \varphi_2(h\tilde{N}_j))(N(\overline{U}_{j2}) - N(u_j)). \end{aligned} \tag{3.91}$$

In this case one can show that $\|\hat{T}_j\| \leq C\|e_j\| + Ch^2$ (by using the boundedness of $(I - \frac{1}{2}hL)^{-1} - 2\varphi_2(h\tilde{N}_j)$, $\varphi_2(hN_j)$, the Lipschitz property of $N(u)$, and the estimate $\|\varphi_2(hN_j) - \varphi_2(h\tilde{N}_j\| \leq Ch\|N_j - \tilde{N}_j\| \leq Ch$, which can be easily proved by applying Lemma 4.2 given in [46]). Using (3.82) (which implies $\|\bar{e}_{n+1,\texttt{HImExp2N}}\| \leq Ch^3$) and the bound (3.53c), i.e. $\|R(hL))^{n-j}\| \leq 1$, we again get the same bound for the global error of scheme `HImExp2N` as in (3.89) which proves the bound (3.83).

□

3.5 Numerical Experiments

Integrators and implementation

In this section we verify theoretical predictions regarding the accuracy and efficiency of the newly constructed IMEXP schemes `ImExpRK2` (3.55), `HImExp2J` (3.57), and `HImExp2N` (3.58). Several IMEX schemes were compared in [60] and it was shown that the second-order semi-implicit backwards differentiation formula-based IMEX scheme `2-sBDF` has the least stringent stability restrictions on the time step and consequently the best computational efficiency for reaction-diffusion problems arising in pattern formation. The same conclusion was reached in [31]. We thus choose to compare the new IMEXP schemes with the widely used `2-sBDF` integrator [60]:

$$3U_{n+1} - 4U_n - U_{n-1} = 2h(LU_{n+1} + 2N(U_n) - N(U_{n-1}). \tag{3.92}$$

Clearly since the operator N is treated explicitly, there are stability restrictions on the time step of `2-sBDF` associated with the degree of stiffness in N. Since the stability constraints on the time step of the new IMEXP schemes are less restrictive, we expect to be able to take a larger time step compared to `2-sBDF`. However, treating N exponentially imposes additional computational cost as compared with `2-sBDF`. Below we present numerical experiments that demonstrate that as the stiffness of N is increased the ability to take a significantly larger time step outweighs the additional computational cost per time step and the IMEXP schemes can outperform `2-sBDF`.

All integrators were implemented in MATLAB. The adaptive Krylov algorithm as described in [69] was used to compute products of matrix φ-functions and vectors. It is known that for a given matrix A and a vector v a Krylov projection-based method to approximate a matrix function-vector product of type $f(A)v$ is typically more efficient for functions $f(z) = \varphi_k(z)$ with higher values of k [23, 41]. Also evaluating $f(A)v$ for a rational function $f(z) = (1 - z)^{-1}$ is expected to be less efficient than computing $\varphi_k(A)v$. Exponential methods in general and EPIRK methods, in particular, are designed to take advantage this property of the Krylov projection [24, 66, 64]. Note that the IMEXP methods proposed here employ $\varphi_2(z)$ in the exponential step of the second stage. Using $\varphi_2(z)$ introduces computational savings compared to estimating $\varphi_0(z)$, $\varphi_1(z)$ or a rational function of the Jacobian as required by fully implicit methods.

The implicit terms are computed in two ways. Since in our examples L is symmetric we can calculate the implicit terms using conjugate-gradient (CG) method. The IMEXP schemes, however, are equally valid for nonsymmetric L and to give a better feel for the performance of these schemes on more general problems we also include comparisons with implicit terms treated with a generalized minimal residual (GMRES) method. MATLAB's built-in functions `pcg` and `gmres` were used to compute the implicit terms using CG and GMRES both with a tolerance set at 10^{-12} for all problems. For demonstration and comparison purposes, we chose to use a sparse incomplete Cholesky factorization (`ichol`) as the preconditioner for both linear solvers. The default parameter values were used except in cases where a given tolerance was desired. All deviations from the default parameter values are detailed in descriptions of the individual problems below. Note that while by taking advantage of symmetry of L CG somewhat reduces the computational time per each Krylov iteration compared to GMRES, the main computational cost of using either of these

iterative methods is the number of Krylov vectors it takes for each Krylov projection. Since both CG and GMRES require the same number of Krylov vectors for each projection, it is not expected that computational savings of CG will be significant compared to GMRES. Figures 3.2 and 3.3 illustrate this point as explained later in the text.

For each problem, the same set of parameters were used for all integrators. All simulations were performed with a constant time step starting with h_1 and the time step sizes were halved $h_i = h_1/2^{i-1}$, $i = 2, \ldots, 5$ to generate the graphs. Table 3.2 shows the largest time step size taken for each of the problems and each of the methods.

Test problems and verification of accuracy

We choose the following three test problems that satisfy Assumptions 1-3 from the previous section. Note that the first problem, the 1D semilinear parabolic equations from [26], was originally designed to demonstrate the order reduction non-stiffly accurate methods can experience when a problem is very stiff. However, this problem is not very computationally intensive and thus we only include it to verify the accuracy of our method and do not use it for performance analysis.

1D Semilinear parabolic. One-dimensional semilinear parabolic problem [26]

$$\frac{\partial u}{\partial t}(x,t) - \frac{\partial^2 u}{\partial x^2}(x,t) = \int_0^1 u(x,t)dx + \Phi(x,t), \quad x \in [0,1], \quad t \in [0,1]$$

with homogeneous Dirichlet boundary conditions. The source function Φ is chosen so that $u(x,t) = x(1-x)e^t$ is the exact solution. To achieve the desired tolerance we increased the default maximum number of GMRES iterations in MATLAB routines to 500.

Allen-Cahn 2D. Two-dimensional stiff Allen-Cahn equation with periodic boundary conditions [72]:

$$\begin{aligned} &\frac{\partial u}{\partial t} = \Delta u - \frac{1}{\epsilon^2}(u^3 - u), \quad x \in [-0.5, 0.5]^2, \quad t \in [0, 0.075] \\ &u(0,x) = \tanh\left(\frac{R_0 - \|x\|_2}{\sqrt{2}\epsilon}\right), \end{aligned} \quad ,$$

where $R_0 = 0.4$ and ϵ specified to be $0.01, 0.02$, and 0.005.

Schnakenberg 2D. Two-dimensional Schnakenberg system [61, 31]

$$\begin{aligned} \frac{\partial u}{\partial t} &= \gamma(a - u + u^2 v) + \Delta u, \\ \frac{\partial v}{\partial t} &= \gamma(b - u^2 v) + d\Delta v, \quad (x,y) \in [0,1]^2 \end{aligned}$$

with homogeneous Neumann boundary conditions, $a = 0.1$, $b = 0.9$, $d = 10$, $t_{end} = 0.1$ for $\gamma = 1000$ & $\gamma = 10000$, and $t_{end} = 0.01$ for $\gamma = 10^5$. The initial conditions were chosen to be perturbations of the equilibrium $(\bar{u}, \bar{v}) = (a + b, b/(a + b)^2)$ as in [48, 31]. For this problem the default restart parameter for the MATLAB's GMRES routine was changed from 10 to 20.

The Laplacian term Δ in all problems was discretized using the standard second order finite differences. For 2D Allen-Cahn equation we use $n_x = n_y = 150$ spatial discretization points

in each spatial dimension, while for the 2D Schnackenberg system $n_x = n_y = 128$. Except for the semilinear parabolic problem where an exact solution is provided, a reference solution was computed for the remaining problems using MATLAB's `ode15s` integrator with absolute and relative tolerances set to 10^{-14}. The error is computed in the discrete infinity (maximum) norm of the difference between the computed and the reference solutions. Figure 4.2 shows the order attained by all methods for each of the problems. For convenience we included a line of slope two (dotted) in the graphs. As can be seen from this plot all methods achieve the second order of accuracy as predicted by the theory.

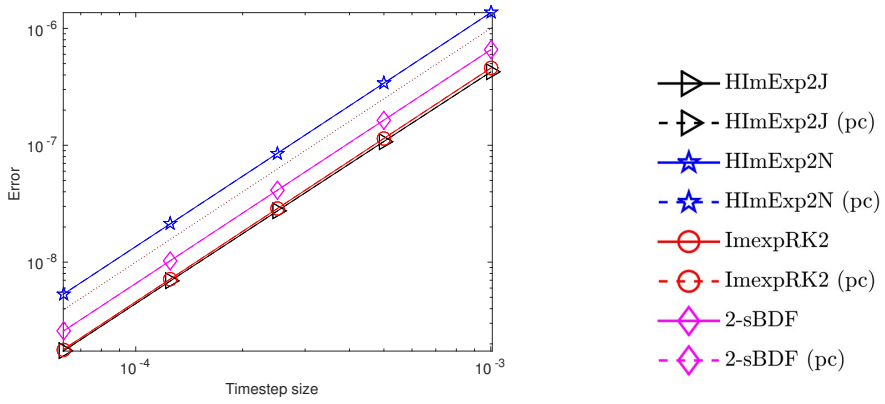
Table 3.2: Largest time step sizes taken for each of the problems and each of the methods.

	2D Schnakenberg $n_x \times n_y = 128^2$			2D Allen-Cahn $n_x \times n_y = 150^2$		
	$\gamma = 1000$	$\gamma = 10000$	$\gamma = 50000$	$\epsilon = 0.02$	$\epsilon = 0.01$	$\epsilon = 0.005$
`HImExp2J`	$1 \cdot 10^{-2}$	$1 \cdot 10^{-3}$	$5 \cdot 10^{-5}$	$5 \cdot 10^{-4}$	$2 \cdot 10^{-4}$	$5 \cdot 10^{-6}$
`HImExp2N`	$1 \cdot 10^{-3}$	$1 \cdot 10^{-4}$	$5 \cdot 10^{-6}$	$1 \cdot 10^{-4}$	$5 \cdot 10^{-5}$	$1 \cdot 10^{-6}$
`ImExpRK2`	$1 \cdot 10^{-3}$	$1 \cdot 10^{-4}$	$1 \cdot 10^{-6}$	$2 \cdot 10^{-4}$	$5 \cdot 10^{-5}$	$1 \cdot 10^{-5}$
`2-sBDF`	$5 \cdot 10^{-4}$	$1 \cdot 10^{-5}$	$5 \cdot 10^{-6}$	$2 \cdot 10^{-4}$	$5 \cdot 10^{-5}$	$5 \cdot 10^{-7}$

Performance evaluation

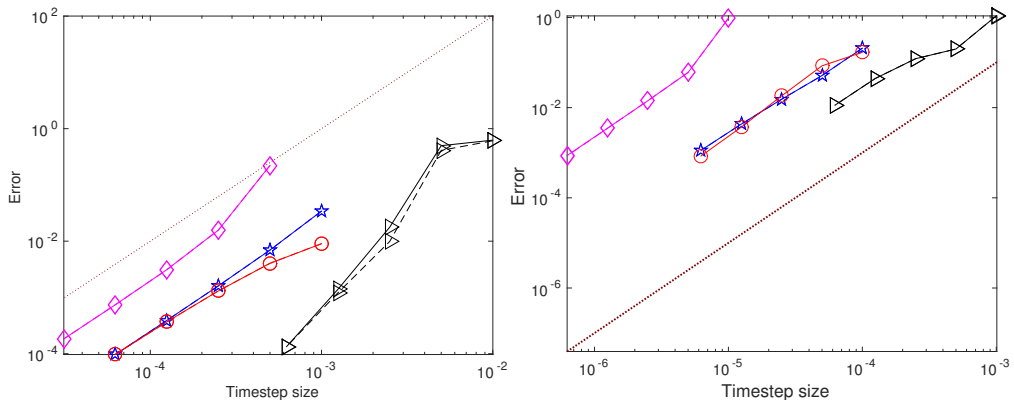
Figures 3.3 and 3.2 show the precision diagrams for all four methods for the 2D Allen-Cahn equation and the 2D Schnackenberg model respectively. For the Allen-Cahn equation the stiffness of the nonlinear operator N is increased with the decreasing value of the ϵ parameter. We choose $\epsilon = 0.02$, 0.01 and 0.005, with the most stiff problem corresponding to $\epsilon = 0.005$. The nonlinear operator of the 2D Schnakenberg's system becomes more stiff as the parameter γ is increased. We set $\gamma = 10^3$, 10^4 and $5 \cdot 10^4$. The graphs show both the nonpreconditioned (solid lines) as well as the preconditioned (dashed lines) version of each of the algorithms. As the Figures 3.3 and 3.2 demonstrate, the method `HImExp2J` is consistently the best performing scheme for all of the simulations for both problems. For Allen-Cahn equation methods `HImExp2N` and `ImExpRK2` are slower than `2-sBDF` for the parameters chosen here due to the higher computational cost per time step, but the largest time step that can be taken with these IMEXP schemes is an order of magnitude larger than the `2-sBDF` method (Table 3.2). It is feasible to imagine that there exist problems for which this will be advantageous. In fact for the most stiff version of the 2D Schnakenberg system (Figure 3.2 (c,d)), `HImExp2N` and `ImExpRK2` begin to outperform `2-sBDF`. `HImExp2J`, however, remains the best performing scheme for Schnakenberg's equation as well. As discussed in section 3.2 this is anticipated due to the fact that `HImExp2J` can offer better stability when integrating stiff nonlinearities.

Figures 3.2 and 3.3 illustrate that computing the implicit terms with CG vs. GMRES does not change our conclusions. As explained in section 3.5 the main computational cost of both CG



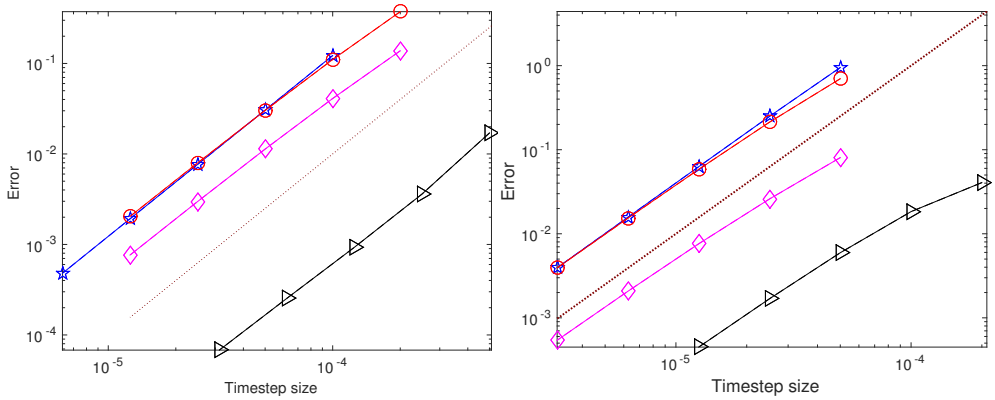
(a) Semilinear parabolic $n_x = 500$

2D Schnakenberg $n_x = n_y = 128$



(b) $\gamma = 10^3$ (c) $\gamma = 10^4$

Allen-Cahn $n_x = n_y = 150$



(d) $\epsilon = 0.02$ (e) $\epsilon = 0.01$

Figure 3.1: Plots of the error vs. time step size. For convenience a line with slope two (dotted) is shown.

and GMRES depends on the number of Krylov vector required for a given accuracy. While CG involves fewer calculations per Krylov vector, the overall number of Krylov vectors needed is the close to GMRES. Thus relative performance of IMEX methods compared to IMEXP schemes does not change.

To detail the computational savings offered by the `HImExp2J` method compared to `2-sBDF` we present Table 4.7 where the CPU times required by the `HImExp2J` scheme are shown as a percentage of the CPU execution time taken by `2-sBDF` for several values of the corresponding error in the solution. When exact CPU time is not available for a given accuracy we interpolate its value from the corresponding precision graphs. The data in the table clearly shows that the computational savings offered by `HImExp2J` grow as the stiffness of the nonlinear operator N is increased. The savings as even more pronounced for the preconditioned versions of the algorithms since the computational cost per time step is decreased and consequently the ability to take a larger time step becomes more important. Note that without the loss of generality in our conclusions we can vary the length of the integration interval. Since the total computational time to achieve accuracy of 10^{-3} with `2-sBDF` for the long time interval for the Schnakenberg system became too long (see Figure 3.2c), we decreased the total integration interval to be able to get CPU times ratio for the stiffest version of the problem with $\gamma = 5 \cdot 10^4$.

The results of our numerical simulations clearly verify the theoretical predictions of the performance of the new IMEXP schemes and present these methods as a promising alternative to the IMEX integrators for problems with stiff nonlinearity N. In this chapter we focused on problems for which efficient preconditioner is available for the linear part, and therefore general exponential integrators are at a disadvantage compared to semi-implicit methods. More research is needed to develop qualitative guidelines that help to determine whether fully implicit, semi-implicit, fully exponential or semi-exponential integrator will be most efficient for a given problem. We plan to address these issues in our future investigations.

We stress that ideas presented in this chapter can be easily extended to construct many additional methods. For example, one could use similar techniques to address problems where the operator L is nonlinear but a good preconditioner is available. Other combinations of the uses of operators L', N' and J are possible. We defer these developments to our future publications along with the construction of higher order IMEXP methods.

Table 3.3: Approximated CPU time for `HimExp2J` as a percentage of the CPU time for `2-sBDF` given a prescribed accuracy.

(a) Allen-Cahn $n_x = n_y = 150$

Accuracy		$\epsilon = 0.02$	$\epsilon = 0.01$	$\epsilon = 0.005$
10^{-2}	Non-Preconditioned	55%	56%	44%
	Preconditioned	51%	56%	44%
10^{-3}	Non-Preconditioned	43%	54%	45%
	Preconditioned	35%	54%	42%

(b) Schnakenberg $n_x = n_y = 128$

Accuracy		$\gamma = 1000$	$\gamma = 10000$	$\gamma = 50000$
10^{-1}	Non-Preconditioned	55 %	33%	33%
	Preconditioned	171%	31%	24%
10^{-2}	Non-Preconditioned	44%	32%	27%
	Preconditioned	120%	22%	20%
10^{-3}	Non-Preconditioned	48%	19%	24%
	Preconditioned	70%	13%	20%

Figure 3.2: 2D Schnakenberg problem ($n_x = n_y = 128$): CPU execution time versus error for constant time step

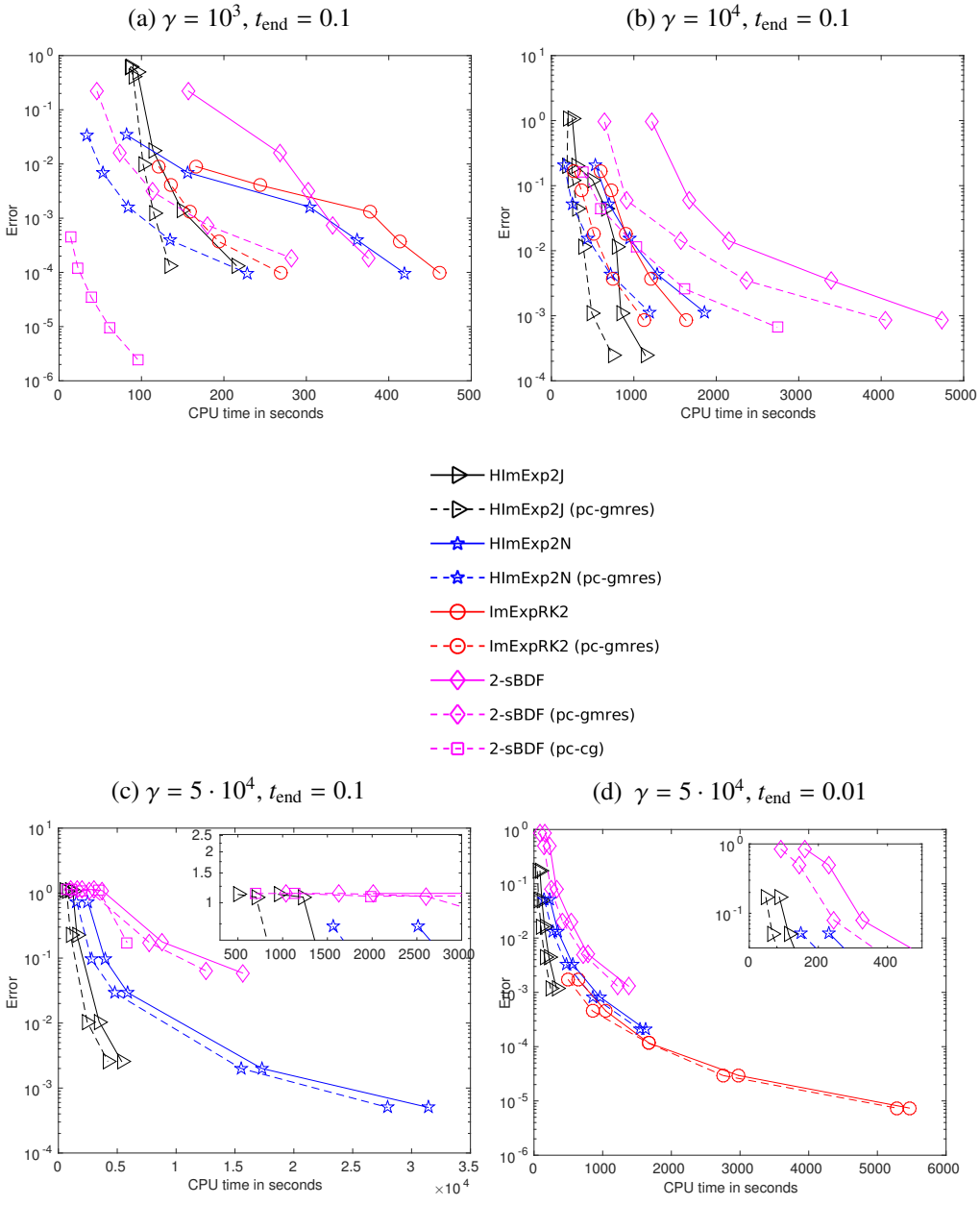
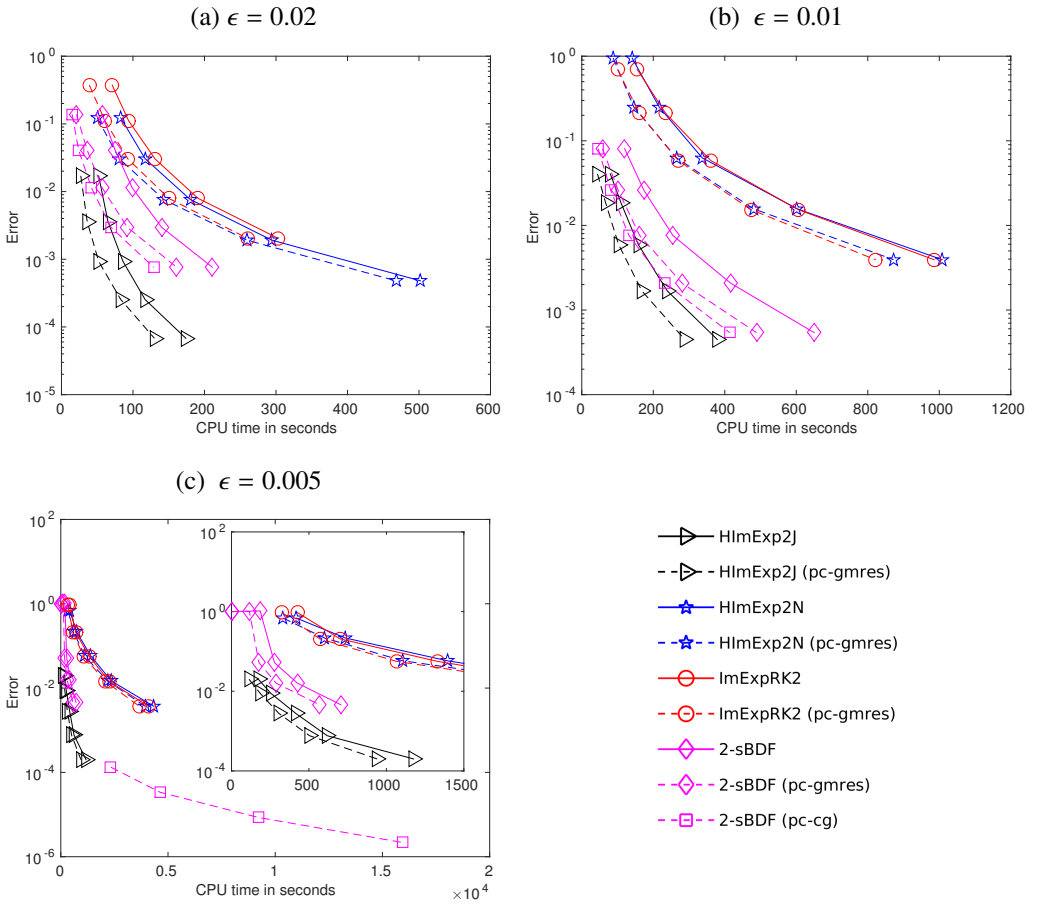


Figure 3.3: 2D Allen-Cahn problem ($n_x = n_y = 150$): CPU execution time versus error for constant time step



Chapter 4

A new approach to constructing efficient stiffly accurate EPIRK methods

4.1 Introduction

A number of numerical studies showed that the EPIRK methods performed well on stiff problems [66, 65, 59]. However, the derivation of the EPIRK schemes and the general convergence theory were based on classical rather than stiff order conditions in previous publications. Methods constructed using stiff order conditions are a subset of classically accurate schemes that do not suffer from order reduction for certain classes of problems. In this chapter we demonstrate that the stiff order conditions and convergence theory developed in [26, 45, 47] can be extended to the EPIRK methods. We derive stiff order conditions for the EPIRK methods of nonsplit (or unpartitioned) type, i.e. the most general version of the EPIRK integrators with $A_{ij} = J_n$ and $r(u) = f(u) - f(u_n) - f'(u_n)(u - u_n)$. We present a systematic way to solve the resulting stiff order conditions and show how the flexibility of the EPIRK framework can be utilized to construct particularly efficient schemes.

The chapter is organized as follows. Section 4.2 describes how the stiff order conditions and the convergence theory from [46] can be extended to include the EPIRK methods. This section also includes an explanation of the differences between the EPIRK framework and the exponential Rosenbrock methods for which the theory was originally developed. In Section 4.3 we propose a new optimization approach and procedure to solve the stiff order conditions for EPIRK methods to derive a range of efficient fourth- and fifth-order schemes. In particular, we construct EPIRK methods that can be particularly efficient when used together with an adaptive Krylov-projection algorithm, currently the most general and efficient way to estimate the exponential matrix function vector products. Finally, Section 4.4 contains numerical tests that validate the performance of the newly derived methods and demonstrate the relative efficiency of these techniques compared to previously proposed schemes.

4.2 Stiffly accurate EPIRK methods

EPIRK and exponential Rosenbrock methods

In this chapter we focus on the nonsplit, or unpartitioned, [65, 59] EPIRK schemes for the general problem (1.1). The unpartitioned EPIRK methods are constructed from (1.2) by setting $A_{ij} = J_n$ to obtain:

$$\begin{aligned} U_{ni} &= u_0 + \alpha_{i1}\psi_{i1}(g_{i1}h_nJ_n)h_nf(u_n) + h_n\sum_{j=2}^{i-1}\alpha_{ij}\psi_{ij}(g_{ij}h_nJ_n)\Delta^{(j-1)}r(u_n), \quad i = 2,\dots,s, \\ u_{n+1} &= u_n + \beta_1\psi_{s+1\,1}(g_{s+1\,1}h_nJ_n)h_nf(u_n) + h_n\sum_{j=2}^{s}\beta_j\psi_{s+1\,j}(g_{s+1,j}h_nJ_n)\Delta^{(j-1)}r(u_n) \end{aligned} \tag{4.1}$$

where $r(u) = f(u) - f(u_n) - J_n(u - u_n)$. Classical order conditions were derived for EPIRK schemes in [65] and these methods were shown to be efficient for stiff problems [38, 69]. The extension of the theory to stiff order conditions presented below will enable us to construct stiffly accurate EPIRK schemes that can be proved to be convergent even for unbounded operators J_n.

The stiff order conditions and the corresponding convergence theory has been developed in [27, 46, 45] for the exponential Rosenbrock methods. While the original formulation of the exponential Rosenbrock methods was first proposed in [57], the full development of this class of integrators, including derivation of the classical and stiff order conditions along with the convergence theory, have not been done until the resurgence of interest in exponential methods over the past several decades [26, 27, 46]. Due to efficiency of implementation and theoretical considerations, in [27] the original formulation of the exponential Rosenbrock methods have been recast in the following form

$$\begin{aligned} U_{ni} &= u_n + c_ih_n\varphi_1(c_ih_nJ_n)f(u_n) + h_n\sum_{j=2}^{i-1}a_{ij}(h_nJ_n)D_{nj} \\ u_{n+1} &= u_n + h_n\varphi_1(h_nJ_n)f(u_n) + h_n\sum_{j=2}^{s}b_i(h_nJ_n)D_{ni} \end{aligned} \tag{4.2}$$

where $N_n(u) = f(u) - J_nu$, $D_{nj} = N_n(U_{nj}) - N_n(u_n)$ and coefficients $a_{ij}(z)$ and $b_{ij}(z)$ are functions comprised of linear combinations $\varphi_k(z)$. The structural difference between (4.1) and (4.2) is in the use of g_{ij} coefficients in (4.1) and in allowing the second term of the right-hand-side in each of the stages to have a more general function $\psi_{i1}(z)$ rather than restricting it to $\psi_{i1}(z) = \varphi_1(z)$ as in (4.2). Note that $D_{nj} = r(U_{nj})$. Any exponential Rosenbrock method can be written in EPIRK form. Any EPIRK method can be written in an extended exponential Rosenbrock form if the differences mentioned above are taken into account. To make it more straightforward to apply the stiff order conditions theory developed for exponential Rosenbrock methods to EPIRK integrators we re-write (4.1) in the extended exponential Rosenbrock form using the expansion

$$\Delta^{(n)}r(u_n) = \sum_{i=2}^{n+1}\begin{pmatrix} n-1 \\ i-1 \end{pmatrix}(-1)^{n-i-2}r(U_{ni}) = \sum_{i=2}^{n}\begin{pmatrix} n-1 \\ i-1 \end{pmatrix}(-1)^{n-i-2}r(U_{ni}), \tag{4.3}$$

and collecting the terms corresponding to each $r(U_{ni})$ in every stage. Then (4.1) can be expressed as

$$\begin{aligned} U_{ni} &= u_n + \alpha_{i1}\psi_{i1}(g_{i1}h_nJ_n)h_nf(u_n) + h_n\sum_{j=2}^{i-1} a_{ij}(h_nJ_n)r(U_{nj}), \quad i = 2,\dots,s, \\ u_{n+1} &= u_n + \beta_1\psi_{s+11}(g_{s+11}h_nJ_n)h_nf(u_n) + h_n\sum_{j=2}^{s} b_j(h_nJ_n)r(U_{nj}) \end{aligned} \tag{4.4}$$

where

$$a_{ij}(z) = \sum_{k=j}^{i-1}\begin{pmatrix} k-1 \\ i-1 \end{pmatrix}(-1)^{k-i-2}\alpha_{ik}\psi_{ik}(g_{ik}z) \quad \text{and} \quad b_j(z) = \sum_{k=j}^{s}\begin{pmatrix} k-1 \\ s \end{pmatrix}(-1)^{k-s-3}\beta_k\psi_{s+1k}(g_{s+1k}z) \tag{4.5}$$

We additionally define $\psi_{i1}(z) = \sum_{k=1}^{s} p_{i1k}\varphi_k(z)$. We now incorporated the g_{ij} coefficients into the definitions of $a_{ij}(z)$ and $b_{ij}(z)$ and extended the second term of the right-hand-sides in stages to general $\psi_{ij}(z)$ functions. Later we will show how these generalizations of the exponential Rosenbrock methods to EPIRK framework offer added flexibility that allows for derivation of more efficient methods. To illustrate this reformulation we consider a simple example of a three-stage EPIRK method

$$\begin{aligned} U_{n2} &= u_n + \alpha_{21}\psi_{21}(g_{21}h_nJ_n)h_nf(u_n) \\ U_{n3} &= u_n + \alpha_{31}\psi_{31}(g_{31}h_nJ_n)h_nf(u_n) + h_n\alpha_{32}\psi_{32}(g_{32}h_nJ_n)\underbrace{\Delta r(u_n)}_{r(U_{n2})} \\ u_{n+1} &= u_n + \beta_1\psi_{41}(g_{41}h_nJ_n)h_nf(u_n) + h_n\beta_2\psi_{42}(g_{42}h_nJ_n)\underbrace{\Delta r(u_n)}_{r(U_{n2})} + \beta_3\psi_{43}(g_{43}h_nJ_n)\underbrace{\Delta^2 r(u_n)}_{r(U_{n3})-2r(U_{n2})}. \end{aligned} \tag{4.6}$$

This EPIRK method can be re-written in an extended exponential Rosenbrock way as

$$\begin{aligned} U_{n2} &= u_n + \alpha_{21}\psi_{21}(g_{21}h_nJ_n)h_nf(u_n) \\ U_{n3} &= u_n + \alpha_{31}\psi_{31}(g_{31}h_nJ_n)h_nf(u_n) + h_n\underbrace{\alpha_{32}\psi_{32}(g_{32}h_nJ_n)}_{a_{32}(h_nJ_n)}r(U_{n2}) \\ u_{n+1} &= u_n + \beta_1\psi_{41}(g_{41}h_nJ_n)h_nf(u_n) + h_n\underbrace{(\beta_3\psi_{42}(g_{42}h_nJ_n) - 2\beta_3\psi_{43}(g_{43}h_nJ_n))}_{b_2(h_nJ_n)}r(U_{n2}). \\ &\quad + h_n\underbrace{\beta_3\psi_{43}(g_{43}h_nJ_n)}_{b_3(h_nJ_n)}r(U_{n3}) \end{aligned} \tag{4.7}$$

Due to the close relationship between EPIRK and exponential Rosenbrock methods outlined above, most of the theory from [46] applies to EPIRK directly. But this generalization has to be handled with care since additional results are needed to account for the more general form of EPIRK schemes. Below we outline the theory for the reader's convenience and present more detail in places where the differences between EPIRK and exponential Rosenbrock methods result in distinct expressions and, ultimately, modified stiff order conditions.

Analytical framework.

As in [46] the analysis is based on the theory of strongly continuous semigroups in a Banach space $\mathcal{X}$ with norm $\|\cdot\|$. For the reader's convenience we state the assumptions from [46] that form the base for the stiff order conditions and the convergence theory; we also outline the main ideas of the theory. For our analysis we consider (1.1) written in a linearized form

$$u'(t) = f(u(t)) = Lu(t) + N(u(t)), \qquad u(t_0) = u_0, \tag{4.8}$$

with Jacobian

$$J = J(u) = \frac{\partial f}{\partial u}(u) = L + \frac{\partial N}{\partial u}(u). \tag{4.9}$$

The following two assumptions are then made about operators L and $N(u)$:
Assumption 1 ([46]). *The linear operator L is the generator of a strongly continuous semigroup e^{tL} in $\mathcal{X}$.*

Assumption 2 ([46]). *We assume that (1.1) possesses a sufficiently smooth solution $u : [0, T] \to \mathcal{X}$ with derivatives in $\mathcal{X}$ and that the nonlinearity $N : \mathcal{X} \to \mathcal{X}$ is sufficiently often Fréchet differentiable in a strip along the exact solution.*

Given these assumptions it can be shown that the Jacobian J in (4.9) is the generator of a strongly continuous semigroup ([56],Chap. 3.1). This implies that there exist constants C and ω such that

$$\left\| e^{tJ} \right\|_{\mathcal{X}\leftarrow\mathcal{X}} \leq Ce^{\omega t}, \quad t \geq 0 \tag{4.10}$$

holds uniformly in a neighborhood of the exact solution. Furthermore, it can be concluded from this result that $\varphi_k(h_nJ)$ and subsequently their linear combinations $a_{ij}(h_nJ)$ & $b_i(h_nJ)$, are bounded operators. Assumption 2 also implies that the Jacobian (4.9) satisfies the Lipschitz condition

$$\| J(u) - J(v) \|_{\mathcal{X}\leftarrow\mathcal{X}} = \left\| \frac{\partial N}{\partial u}(u) - \frac{\partial N}{\partial u}(v) \right\|_{\mathcal{X}\leftarrow\mathcal{X}} \leq L \| u - v \| \tag{4.11}$$

in a neighborhood of the exact solution.

Note that problems with homogeneous or no-flow boundary conditions satisfy Assumptions 1 and 2. In general, non-homogeneous boundary conditions (including time-dependent) do not necessarily satisfy Assumption 1. As a simple example, consider a semigroup $T(t) = e^{tA}$ defined over the Banach space $\mathcal{X} = L^2(\mathbb{R})$ of square-integrable functions. Assumption 1 requires the semigroup to be strongly continuous. By the Hille-Yoshida theorem the necessary condition for the semigroup to be strongly continuous is that the domain $D(A)$ of the infinitesimal generator of the semigroup A is dense in $\mathcal{X}$. Thus, it is necessary to restrict the domain $D(A)$ in order to ensure that the semigroup e^{tA} is strongly continuous. If we assume that the solutions $w(t)$ of $w'(t) = Aw$ belong to a subspace $C_0^\infty(\mathbb{R}) \in \mathcal{X}$ it is possible to prove that the semigroup e^{tA} is strongly continuous [56] . This is not necessarily the case for the subspace of functions $w(t)$ with non-homogeneous boundary conditions. The analysis quantifying how much order reduction one can expect for the problems with non-homogeneous boundary conditions has been done for standard Runge-Kutta and Rosenbrock methods in [53, 52]. Developing similar fractional order convergence theory for exponential

methods or developing exponential schemes which do not exhibit order reduction for problems with non-homogeneous boundary conditions are non-trivial tasks which we plan to address in our future research. In this chapter we address this issue by using numerical examples to illustrate how much order reduction one can expect if the boundary conditions are non-homogeneous.

The derivation of the stiff order conditions and the convergence theory for problems which satisfy Assumptions 1 and 2 then proceeds as follows.

Local error and stiff order conditions for EPIRK methods.

Derivation of the stiff order conditions and proof of convergence require analysis of the local error and construction of expressions for the approximate and exact solutions. To accommodate the difference between EPIRK and exponential Rosenbrock methods derivation of expressions for the numerical solution in [46] has to be adjusted. Thus, we choose to present this derivation in more detail and simply restate other results from [46] that are used without alternations. The key idea in the convergence theory is to express the error in terms of operators that are bounded. While the Jacobian operator J_n can potentially be unbounded, expressions involving only derivatives of the solution $u_n^{(k)}$ and/or the nonlinearity $\partial^k N/\partial^k u$ are bounded given Assumptions 1 and 2. The following formulas that connect these two groups of operators help make these transitions. Consider linearization of (1.1) along the exact solution $\tilde{u}_n = u(t_n)$ to get

$$u'(t) = \tilde{J}_n u(t) + \tilde{N}_n(u(t)) \tag{4.12}$$

where

$$\tilde{J}_n = \frac{\partial f}{\partial u}(\tilde{u}_n) = L + \frac{\partial N}{\partial u}(\tilde{u}_n), \quad \tilde{N}_n(u) = f(u) - \tilde{J}_n u. \tag{4.13}$$

From (4.13), we obtain

$$\tilde{N}_n(u) = N(u) - \frac{\partial N}{\partial u}(\tilde{u}_n)u \quad \text{and} \quad \frac{\partial \tilde{N}_n}{\partial u}(\tilde{u}_n) = 0. \tag{4.14}$$

Using these identities along with the repeated differentiation of (4.12) we get:

$$\tilde{J}_n \tilde{u}_n' = \tilde{u}_n'' \qquad \& \qquad \tilde{J}_n \tilde{u}_n'' = \tilde{u}_n^{(3)} - \frac{\partial^2 \tilde{N}_n}{\partial u^2}(\tilde{u}_n)(\tilde{u}_n', \tilde{u}_n'), \tag{4.15}$$

where $\tilde{u}_n^{(k)}$ denotes the kth derivative of the exact solution of (4.12). More generally, we have

$$\tilde{J}_n u^{(k)}(t) = u^{(k+1)}(t) - \frac{d^k}{dt^k}\tilde{N}_n(u(t)) \tag{4.16}$$

which shows that $\tilde{J}_n u^{(k)}$ is bounded (due to Assumption 2). This result and identities (4.14) are key to deriving the stiff order conditions.

First following the procedure in [46] we carry out one integration step with (4.4) with exact solution $\tilde{u}_n$ used for the initial value to express the numerical solution as

$$\begin{aligned} \widehat{U}_{ni} &= \tilde{u}_n + \alpha_{i1}\psi_{i1}(g_{i1}h_n\tilde{J}_n)h_n f(\tilde{u}_n) + h_n\sum_{j=2}^{i-1} a_{ij}(h_n\tilde{J}_n)\widehat{r}_{nj}, \quad i = 2,\dots,s, \\ \widehat{u}_{n+1} &= \tilde{u}_n + \beta_1\psi_{s+1\,1}(g_{s+1\,1}h_n\tilde{J}_n)h_n f(\tilde{u}_n) + h_n\sum_{i=2}^{s} b_i(h_n\tilde{J}_n)\widehat{r}_{ni} \end{aligned}, \tag{4.17}$$

with

$$\widehat{r}_{ni} = \tilde{N}_n(\widehat{U}_{ni}) - \tilde{N}_n(\tilde{u}_n). \tag{4.18}$$

We now begin computing the Taylor expansion of (4.17) by first calculating $\widehat{r}_{ni}$ as a Taylor series around $\tilde{u}_n$. Using (4.14) we obtain

$$\widehat{r}_{ni} = \sum_{q=2}^{k}\frac{h_n^q}{q!}\frac{\partial^q\tilde{N}_n}{\partial u^q}(\tilde{u}_n)\underbrace{(V_i,\dots,V_i)}_{q\text{ times}} + \mathcal{R}_{ki} \tag{4.19}$$

with

$$V_i = \frac{1}{h_n}\left(\widehat{U}_{ni} - \tilde{u}_n\right) = \alpha_{i1}\psi_{i1}(g_{i1}h_n\tilde{J}_n) + \sum_{j=2}^{i-1} a_{ij}(h_n\tilde{J}_n)\widehat{r}_{nj} \tag{4.20}$$

and the remainder

$$\mathcal{R}_{ki} = h_n^{k+1}\int_0^1 \frac{(1-\theta)^k}{k!}\frac{\partial^{k+1}\tilde{N}_n}{\partial u^{k+1}}(\tilde{u}_n + \theta h_n V_i)\underbrace{(V_i,\dots,V_i)}_{k+1\text{ times}}d\theta$$

which is bounded and of order $\mathcal{R}_{ki} = \mathcal{O}(h_n^{k+1})$ by Assumptions 1 and 2. Note the expression (4.20) corresponds to the formula (3.7) in [46] with functional coefficients ψ_{i1}, a_{ij} generalized. By substituting (4.19) into (4.17) we obtain

$$\hat{u}_{n+1} = \tilde{u}_n + \beta_1\psi_{s+1\,1}(g_{s+1\,1}h_n\tilde{J}_n)h_n f(\tilde{u}_n) + \sum_{i=2}^{s} b_i(h_n\tilde{J}_n)\sum_{q=2}^{k}\frac{h_n^{q+1}}{q!}\frac{\partial^q\tilde{N}_n}{\partial u^q}\underbrace{(V_i,\dots,V_i)}_{q\text{ times}} + \mathcal{O}(h_n^{k+2}) \tag{4.21}$$

The following lemmas represent analogues of lemmas 3.1 and 3.2 in [46]. These results allow us to obtain the expansion of (4.21) avoiding terms containing powers of the possibly unbounded operator $\tilde{J}_n$.

Lemma 4. *Under Assumptions 1 and 2, we have for all $t \geq 0$*

$$\begin{aligned} \varphi_k(th_n\tilde{J}_n)f(\tilde{u}_n) =& \frac{1}{k!}\tilde{u}_n' + h_n\frac{t}{(k+1)!}\tilde{u}_n'' + \\ &+ t^2h_n^2\frac{1}{(k+2)!}\left(\tilde{u}_n^{(3)} - (k+2)!\varphi_{k+2}(th_n\tilde{J}_n)\frac{\partial^2\tilde{N}_n}{\partial u^2}(\tilde{u}_n)(\tilde{u}_n',\tilde{u}_n')\right) + \mathcal{O}(t^3h_n^3), \end{aligned} \tag{4.22}$$

and furthermore,

$$\psi_{i1}(t\tilde{J}_n)f(\tilde{u}_n) = \mathbb{P}_{i1}\tilde{u}_n' + t\mathbb{P}_{i2}\tilde{u}_n'' + t^2\left(\mathbb{P}_{i3}\tilde{u}_n^{(3)} - \left(\sum_{k=1}^{s} p_{i1k}\varphi_{k+2}(t\tilde{J}_n)\right)\frac{\partial^2\tilde{N}_n}{\partial u^2}(\tilde{u}_n)(\tilde{u}_n', \tilde{u}_n')\right) + O(t^3) \quad (4.23)$$

where

$$\mathbb{P}_{i1} = \sum_{k=1}^{s}\frac{p_{i1k}}{k!}, \quad \mathbb{P}_{i2} = \sum_{k=1}^{s}\frac{p_{i1k}}{(k+1)!}, \quad \mathbb{P}_{i3} = \sum_{k=1}^{s}\frac{p_{i1k}}{(k+2)!} \quad (4.24)$$

Proof. By using $f(\tilde{u}_n) = \tilde{u}_n'$, the recurrence relation

$$\varphi_k(z) = z\varphi_{k+1}(z) + 1/k! \quad (4.25)$$

and formulas (4.15) we have

$$\begin{aligned}
\varphi_k(th_n\tilde{J}_n)\tilde{u}_n' &= \left(\frac{1}{k!} + th_n\tilde{J}_n\varphi_{k+1}(th_n\tilde{J}_n)\right)\tilde{u}_n' \\
&= \frac{1}{k!}\tilde{u}_n' + th_n\varphi_{k+1}(th_n\tilde{J}_n)\tilde{J}_n\tilde{u}_n' \\
&= \frac{1}{k!}\tilde{u}_n' + th_n\left(\frac{1}{(k+1)!} + th_n\tilde{J}_n\varphi_{k+2}(th_n\tilde{J}_n)\right)\tilde{u}_n'' \\
&= \frac{1}{k!}\tilde{u}_n' + h_n\frac{t}{(k+1)!}\tilde{u}_n'' + t^2h_n^2\varphi_{k+2}(th_n\tilde{J}_n)\left(\tilde{u}_n^{(3)} - \frac{\partial^2\tilde{N}_n}{\partial u^2}(\tilde{u}_n)(\tilde{u}_n', \tilde{u}_n')\right) \\
&= \frac{1}{k!}\tilde{u}_n' + h_n\frac{t}{(k+1)!}\tilde{u}_n'' + t^2h_n^2\left(\frac{1}{(k+2)!} + th_n\tilde{J}_n\varphi_{k+3}(th_n\tilde{J}_n)\right)\tilde{u}_n^{(3)} + \\
&\qquad -t^2h_n^2\varphi_{k+2}(th_n\tilde{J}_n)\frac{\partial^2\tilde{N}_n}{\partial u^2}(\tilde{u}_n)(\tilde{u}_n', \tilde{u}_n') \\
&= \frac{1}{k!}\tilde{u}_n' + h_n\frac{t}{(k+1)!}\tilde{u}_n'' + t^2h_n^2\frac{1}{(k+2)!}\tilde{u}_n^{(3)} + t^3h_n^3\varphi_{k+3}(th_n\tilde{J}_n)\tilde{J}_n\tilde{u}_n^{(3)} + \\
&\qquad -t^2h_n^2\varphi_{k+2}(th_n\tilde{J}_n)\frac{\partial^2\tilde{N}_n}{\partial u^2}(\tilde{u}_n)(\tilde{u}_n', \tilde{u}_n') \\
&= \frac{1}{k!}\tilde{u}_n' + h_n\frac{t}{(k+1)!}\tilde{u}_n'' + \\
&\qquad +t^2h_n^2\frac{1}{(k+2)!}\left(\tilde{u}_n^{(3)} - (k+2)!\varphi_{k+2}(th_n\tilde{J}_n)\frac{\partial^2\tilde{N}_n}{\partial u^2}(\tilde{u}_n)(\tilde{u}_n', \tilde{u}_n')\right) + O(t^3h_n^3),
\end{aligned}$$

where the last equality holds due to the boundedness of $\tilde{J}_n\tilde{u}_n^{(3)}$. Equation (4.23) follows directly from using these expansions for $\varphi_k(z)$ in $\psi_{i1}(z) = \sum_{k=1}^{s} p_{i1k}\varphi_k(z)$. □

Lemma 5. *Under Assumptions 1 and 2, we have*

$$V_i = \alpha_{i1}\mathbb{P}_{i1}\tilde{u}_n' + h_n\alpha_{i1}g_{i1}\mathbb{P}_{i2}\tilde{u}_n'' + h_n^2\alpha_{i1}g_{i1}^2\mathbb{P}_{i3}\tilde{u}_n^{(3)} + h_n^2\Psi_i(h_n\tilde{J}_n)\frac{\partial^2\tilde{N}_n}{\partial y^2}(\tilde{u}_n)(\tilde{u}_n', \tilde{u}_n') + O(h_n^3) \quad (4.26)$$

where

$$\Psi_i(z) = \frac{1}{2!}\sum_{j=2}^{i-1}\alpha_{j1}^2\mathbb{P}_{j1}^2 a_{ij}(z) - g_{i1}^2\left(\sum_{k=1}^{s} p_{i1k}\varphi_{k+2}(g_{i1}z)\right) \tag{4.27}$$

Proof. Inserting (4.20) into (4.19) for $k = 2$ and using Lemma 4 with $t = g_{1j}h_n$ gives

$$\begin{aligned}
\widehat{r}_{nj} &= \frac{h_n^2}{2!}\frac{\partial^2\tilde{N}_n}{\partial u^2}(\tilde{u}_n)(V_j, V_j) + O(h_n^3) \\
&= \frac{h_n^2}{2!}\frac{\partial^2\tilde{N}_n}{\partial u^2}(\tilde{u}_n)\left(\alpha_{j1}\varphi_1(g_{j1}h_n\tilde{J}_n)f(\tilde{u}_n), \alpha_{j1}\varphi_1(g_{j1}h_n\tilde{J}_n)f(\tilde{u}_n)\right) + O(h_n^3) \\
&= \frac{h_n^2}{2!}\frac{\partial^2\tilde{N}_n}{\partial u^2}(\tilde{u}_n)\left(\alpha_{j1}\mathbb{P}_{j1}\tilde{u}_n', \alpha_{j1}\mathbb{P}_{j1}\tilde{u}_n'\right) + O(h_n^3) \\
&= \frac{h_n^2}{2!}\alpha_{j1}^2\mathbb{P}_{j1}^2\frac{\partial^2\tilde{N}_n}{\partial u^2}(\tilde{u}_n)\left(\tilde{u}_n', \tilde{u}_n'\right) + O(h_n^3)
\end{aligned}$$

Substituting this into (4.20) we obtain

$$V_i = \alpha_{i1}\psi_{i1}(g_{i1}h_n\tilde{J}_n) + \frac{h_n^2}{2!}\sum_{j=2}^{i-1} a_{ij}(h_n\tilde{J}_n)\alpha_{j1}^2\mathbb{P}_{j1}^2\frac{\partial^2\tilde{N}_n}{\partial u^2}(\tilde{u}_n)\left(\tilde{u}_n', \tilde{u}_n'\right)$$

which yields the desired result if Lemma 4 is used:

$$\begin{aligned}
V_i &= \alpha_{i1}\mathbb{P}_{i1}\tilde{u}_n' + \alpha_{i1}h_n g_{i1}\mathbb{P}_{i2}\tilde{u}_n'' + \alpha_{i1}h_n^2 g_{i1}^2\left(\mathbb{P}_{i3}\tilde{u}_n^{(3)} - \left(\sum_{k=1}^{s} p_{i1k}\varphi_{k+2}(t\tilde{J}_n)\right)\frac{\partial^2\tilde{N}_n}{\partial u^2}(\tilde{u}_n)(\tilde{u}_n', \tilde{u}_n')\right) \\
&\quad + \frac{h_n^2}{2!}\sum_{j=2}^{i-1} a_{ij}(h_n\tilde{J}_n)\alpha_{j1}^2\mathbb{P}_{j1}^2\frac{\partial^2\tilde{N}_n}{\partial u^2}(\tilde{u}_n)\left(\tilde{u}_n', \tilde{u}_n'\right) + O(h_n^3) \\
&= \alpha_{i1}\mathbb{P}_{i1}\tilde{u}_n' + h_n\alpha_{i1}g_{i1}\mathbb{P}_{i2}\tilde{u}_n'' + h_n^2\alpha_{i1}g_{i1}^2\mathbb{P}_{i3}\tilde{u}_n^{(3)} \\
&\quad + h_n^2\left(-g_{i1}^2\sum_{k=1}^{s}\alpha_{i1}p_{i1k}\varphi_{k+2}(t\tilde{J}_n) + \frac{1}{2!}\sum_{j=2}^{i-1} a_{ij}(h_n\tilde{J}_n)\alpha_{j1}^2\mathbb{P}_{j1}^2\right)\frac{\partial^2\tilde{N}_n}{\partial u^2}(\tilde{u}_n)\left(\tilde{u}_n', \tilde{u}_n'\right) + O(h_n^3).
\end{aligned}$$

□

We now use these expansions of V_i to obtain

$$\begin{aligned}
\frac{\partial^4 \tilde{N}_n}{\partial u^4}(\tilde{u}_n)(V_i,V_i,V_i,V_i) &= \alpha_{i1}^4 \mathbb{P}_{i1}^4 \frac{\partial^4 \tilde{N}_n}{\partial u^4}(\tilde{u}_n)(\tilde{u}_n',\tilde{u}_n',\tilde{u}_n',\tilde{u}_n') + O(h_n) \\
\frac{\partial^3 \tilde{N}_n}{\partial u^3}(\tilde{u}_n)(V_i,V_i,V_i) &= \alpha_{i1}^3 \mathbb{P}_{i1}^3 \frac{\partial^3 \tilde{N}_n}{\partial u^3}(\tilde{u}_n)(\tilde{u}_n',\tilde{u}_n',\tilde{u}_n') + \\
&\quad +3h_n\alpha_{i1}^3 g_{i1}\mathbb{P}_{i1}^2\mathbb{P}_{i2}\frac{\partial^3 \tilde{N}_n}{\partial u^3}(\tilde{u}_n)(\tilde{u}_n',\tilde{u}_n',\tilde{u}_n'') + O(h_n^2) \\
\frac{\partial^2 \tilde{N}_n}{\partial u^2}(\tilde{u}_n)(V_i,V_i) &= \alpha_{i1}^2 \mathbb{P}_{i1}^2 \frac{\partial^2 \tilde{N}_n}{\partial u^2}(\tilde{u}_n)(\tilde{u}_n',\tilde{u}_n') + 2h_n\alpha_{i1}^2 g_{i1}\mathbb{P}_{i1}\mathbb{P}_{i2}\frac{\partial^2 \tilde{N}_n}{\partial u^2}(\tilde{u}_n)(\tilde{u}_n',\tilde{u}_n'') \\
&\quad +2h_n^2\alpha_{i1}^2 g_{i1}^2\mathbb{P}_{i1}\mathbb{P}_{i3}\frac{\partial^2 \tilde{N}_n}{\partial u^2}(\tilde{u}_n)(\tilde{u}_n',\tilde{u}_n^{(3)}) + h_n^2\alpha_{i1}^2 g_{i1}^2\mathbb{P}_{i2}^2\frac{\partial^2 \tilde{N}_n}{\partial u^2}(\tilde{u}_n)(\tilde{u}_n'',\tilde{u}_n'') \\
&\quad +2h_n^2\alpha_{i1}^2\mathbb{P}_{i1}\frac{\partial^2 \tilde{N}_n}{\partial u^2}(\tilde{u}_n)\left(\tilde{u}_n',\Psi_i(h_n\tilde{J}_n)\frac{\partial^2 \tilde{N}_n}{\partial u^2}(\tilde{u}_n)(\tilde{u}_n',\tilde{u}_n')\right) + O(h_n^3)
\end{aligned}$$

and insert these expressions into (4.21) with $k = 4$ to get

$$\hat{u}_{n+1} = \tilde{u}_n + \beta_1\psi_{s+1\,1}(g_{s+1\,1}h_n\tilde{J}_n)h_n f(\tilde{u}_n) + \tag{4.28}$$

$$+\sum_{i=2}^{s} b_i(h_n\tilde{J}_n)\left(\frac{h_n^3}{2}\frac{\partial^2 \tilde{N}_n}{\partial u^2}(\tilde{u}_n)(V_i,V_i) + \frac{h_n^4}{3!}\frac{\partial^3 \tilde{N}_n}{\partial u^3}(\tilde{u}_n)(V_i,V_i,V_i) + \tag{4.29}$$

$$+\frac{h_n^5}{4!}\frac{\partial^4 \tilde{N}_n}{\partial u^4}(\tilde{u}_n)(V_i,V_i,V_i,V_i)\right) + O(h_n^6)$$

and ultimately

$$\begin{aligned}
\hat{u}_{n+1} \;=\; & \tilde{u}_n + \beta_1 \psi_{s+1\,1}(g_{s+1\,1} h_n \tilde{J}_n) h_n f(\tilde{u}_n) + h_n^3 \left(\sum_{i=2}^{s} b_i(h_n \tilde{J}_n) \frac{\alpha_{i1}^2 \mathbb{P}_{i1}^2}{2!} \right) \frac{\partial^2 \tilde{N}_n}{\partial y^2}(\tilde{u}_n)(\tilde{u}_n', \tilde{u}_n') \\
& + h_n^4 \left(\sum_{i=2}^{s} b_i(h_n \tilde{J}_n) \frac{2}{2!} \alpha_{i1}^2 g_{i1} \mathbb{P}_{i1} \mathbb{P}_{i2} \right) \frac{\partial^2 \tilde{N}_n}{\partial y^2}(\tilde{u}_n)(\tilde{u}_n', \tilde{u}_n'') \\
& + h_n^4 \left(\sum_{i=2}^{s} b_i(h_n \tilde{J}_n) \frac{1}{3!} \alpha_{i1}^3 \mathbb{P}_{i1}^3 \right) \frac{\partial^3 \tilde{N}_n}{\partial u^3}(\tilde{u}_n)(\tilde{u}_n', \tilde{u}_n', \tilde{u}_n') \\
& + h_n^5 \left[\left(\sum_{i=2}^{s} b_i(h_n \tilde{J}_n) \frac{2}{2!} g_{i1}^2 \alpha_{i1}^2 \mathbb{P}_{i1} \mathbb{P}_{i3} \right) \frac{\partial^2 \tilde{N}_n}{\partial y^2}(\tilde{u}_n)(\tilde{u}_n', \tilde{u}_n^{(3)}) \right. \\
& + \left(\sum_{i=2}^{s} b_i(h_n \tilde{J}_n) \frac{1}{2!} \alpha_{i1}^2 g_{i1}^2 \mathbb{P}_{i2}^2 \right) \frac{\partial^2 \tilde{N}_n}{\partial y^2}(\tilde{u}_n)(\tilde{u}_n'', \tilde{u}_n'') \\
& + \left(\sum_{i=2}^{s} b_i(h_n \tilde{J}_n) \frac{3}{3!} g_{i1} \alpha_{i1}^3 \mathbb{P}_{i1}^2 \mathbb{P}_{i2} \right) \frac{\partial^3 \tilde{N}_n}{\partial u^3}(\tilde{u}_n)(\tilde{u}_n', \tilde{u}_n', \tilde{u}_n'') \\
& \left. + \left(\sum_{i=2}^{s} b_i(h_n \tilde{J}_n) \frac{1}{4!} \alpha_{i1}^4 \mathbb{P}_{i1}^4 \right) \frac{\partial^4 \tilde{N}_n}{\partial u^4}(\tilde{u}_n)(\tilde{u}_n', \tilde{u}_n', \tilde{u}_n', \tilde{u}_n') \right] \\
& + h_n^5 \left(\sum_{i=2}^{s} b_i(h_n \tilde{J}_n) \frac{2}{2!} \alpha_{i1}^2 \mathbb{P}_{i1} \right) \frac{\partial^2 \tilde{N}_n}{\partial u^2}(\tilde{u}_n) \left(\tilde{u}_n', \Psi_i(h_n \tilde{J}_n) \frac{\partial^2 \tilde{N}_n}{\partial u^2}(\tilde{u}_n)(\tilde{u}_n', \tilde{u}_n') \right). \qquad (4.30)
\end{aligned}$$

The expression for the Taylor expansion of the exact solution we borrow directly from [46]:

$$\begin{aligned}
\tilde{u}_{n+1} = \; & \tilde{u}_n + h_n \varphi_1(h_n \tilde{J}_n) f(\tilde{u}_n) + h_n^3 \varphi_3(h_n \tilde{J}_n) \frac{\partial^2 \tilde{N}_n}{\partial u^2}(\tilde{u}_n)(\tilde{u}_n', \tilde{u}_n') \\
& + h_n^4 \varphi_4(h_n \tilde{J}_n) \left[3 \frac{\partial^2 \tilde{N}_n}{\partial u^2}(\tilde{u}_n)(\tilde{u}_n', \tilde{u}_n'') + \frac{\partial^3 \tilde{N}_n}{\partial u^3}(\tilde{u}_n)(\tilde{u}_n', \tilde{u}_n', \tilde{u}_n') \right] \\
& + h_n^5 \varphi_5(h_n \tilde{J}_n) \left[4 \frac{\partial^2 \tilde{N}_n}{\partial u^2}(\tilde{u}_n)(\tilde{u}_n', \tilde{u}_n^{(3)}) + 3 \frac{\partial^2 \tilde{N}_n}{\partial u^2}(\tilde{u}_n)(\tilde{u}_n'', \tilde{u}_n'') \right. \\
& \left. + 6 \frac{\partial^3 \tilde{N}_n}{\partial u^3}(\tilde{u}_n)(\tilde{u}_n', \tilde{u}_n', \tilde{u}_n'') + \frac{\partial^4 \tilde{N}_n}{\partial u^4}(\tilde{u}_n)(\tilde{u}_n', \tilde{u}_n', \tilde{u}_n', \tilde{u}_n') \right] + O(h_n^6)
\end{aligned} \qquad (4.31)$$

Now subtracting (4.31) from (4.30) we obtain the expression for the local error $\tilde{e}_{n+1} = \hat{u}_{n+1} - \tilde{u}_{n+1}$

in the form:

$$
\begin{aligned}
\tilde{e}_{n+1} \quad &= h_n \left(\beta_1 \psi_{s+1\,1}(g_{s+11} h_n \tilde{J}_n) - \varphi_1(h_n \tilde{J}_n)\right) f(\tilde{u}_n) \\
&+ h_n^3 \left(\sum_{i=2}^{s} b_i(h_n \tilde{J}_n) \frac{\alpha_{i1}^2 \mathbb{P}_{i1}^2}{2!} - \varphi_3(h_n \tilde{J}_n) \right) \frac{\partial^2 \tilde{N}_n}{\partial y^2}(\tilde{u}_n)(\tilde{u}_n', \tilde{u}_n') \\
&+ h_n^4 \left(\sum_{i=2}^{s} b_i(h_n \tilde{J}_n) \alpha_{i1}^2 g_{i1} \mathbb{P}_{i1} \mathbb{P}_{i2} - 3\varphi_4(h_n \tilde{J}_n) \right) \frac{\partial^2 \tilde{N}_n}{\partial y^2}(\tilde{u}_n)(\tilde{u}_n', \tilde{u}_n'') \\
&+ h_n^4 \left(\sum_{i=2}^{s} \frac{1}{3!} b_i(h_n \tilde{J}_n) \alpha_{i1}^3 \mathbb{P}_{i1}^3 - \varphi_4(h_n \tilde{J}_n) \right) \frac{\partial^3 \tilde{N}_n}{\partial u^3}(\tilde{u}_n)(\tilde{u}_n', \tilde{u}_n', \tilde{u}_n') \\
&+ h_n^5 \left(\sum_{i=2}^{s} b_i(h_n \tilde{J}_n) \frac{2}{2!} g_{i1}^2 \alpha_{i1}^2 \mathbb{P}_{i1} \mathbb{P}_{i3} - 4\varphi_5(h_n \tilde{J}_n) \right) \frac{\partial^2 \tilde{N}_n}{\partial y^2}(\tilde{u}_n)(\tilde{u}_n', \tilde{u}_n^{(3)}) \\
&+ h_n^5 \left(\sum_{i=2}^{s} b_i(h_n \tilde{J}_n) \frac{1}{2!} \alpha_{i1}^2 g_{i1}^2 \mathbb{P}_{i2}^2 - 3\varphi_5(h_n \tilde{J}_n) \right) \frac{\partial^2 \tilde{N}_n}{\partial y^2}(\tilde{u}_n)(\tilde{u}_n'', \tilde{u}_n'') \\
&+ h_n^5 \left(\sum_{i=2}^{s} b_i(h_n \tilde{J}_n) \frac{1}{2!} g_{i1} \alpha_{i1}^3 \mathbb{P}_{i1}^2 \mathbb{P}_{i2} - 6\varphi_5(h_n \tilde{J}_n) \right) \frac{\partial^3 \tilde{N}_n}{\partial u^3}(\tilde{u}_n)(\tilde{u}_n', \tilde{u}_n', \tilde{u}_n'') \\
&+ h_n^5 \left(\sum_{i=2}^{s} b_i(h_n \tilde{J}_n) \frac{1}{4!} \alpha_{i1}^4 \mathbb{P}_{i1}^4 - \varphi_5(h_n \tilde{J}_n) \right) \frac{\partial^4 \tilde{N}_n}{\partial u^4}(\tilde{u}_n)(\tilde{u}_n', \tilde{u}_n', \tilde{u}_n', \tilde{u}_n') \\
&+ h_n^5 \left(\sum_{i=2}^{s} b_i(h_n \tilde{J}_n) \frac{2}{2!} \alpha_{i1}^2 \mathbb{P}_{i1} \right) \frac{\partial^2 \tilde{N}_n}{\partial u^2}(\tilde{u}_n) \left(\tilde{u}_n', \Psi_i(h_n \tilde{J}_n) \frac{\partial^2 \tilde{N}_n}{\partial u^2}(\tilde{u}_n)(\tilde{u}_n', \tilde{u}_n') \right),
\end{aligned}
\tag{4.32}
$$

with $\Psi_i(z)$ given by (4.27). From (4.32) we can easily read off the stiff order conditions by ensuring the terms for a given order (three,four, or five) vanish. These conditions are given in Table 4.1. To eliminate the first order term in our error we must have $\beta_1 \psi_{s+1\,1}(g_{s+11} z) = \varphi_1(z)$. If this condition is satisfied, the resulting method will be of stiff order two. Throughout the rest of the chapter we will set $\beta_1 = g_{s+1} = 1$ and $\psi_{s+1\,1} = \varphi_1$. In Section 4.3 we will show how more flexibility in the stiff order conditions for EPIRK compared to the exponential Rosenbrock methods leads to construction of more efficient techniques.

Convergence.

The majority of the convergence proof presented in [46] for the exponential Rosenbrock (EXPRB) methods can be applied directly to the EPIRK schemes. The only exception is Lemma 4.5 in [46]. In order to obtain the same result as in this lemma we need to use additional assumption on the coefficients. Assumption 3 given below allows for less restrictive choice of the coefficients compared to EXPRB methods but enables us to proceed with the convergence proof in the same way as in [46]:

Ref. label	Order condition	Order
C1	$\sum_{i=2}^{s} b_i(Z)\alpha_{i1}^2\mathbb{P}_{i1}^2 = 2\varphi_3(Z)$	3
C2	$\sum_{i=2}^{s} b_i(Z)\alpha_{i1}^2 g_{i1}\mathbb{P}_{i1}\mathbb{P}_{i2} = 3\varphi_4(Z)$	4
C3	$\sum_{i=2}^{s} b_i(Z)\alpha_{i1}^3\mathbb{P}_{i1}^3 = 3!\varphi_4(Z)$	4
C4	$\sum_{i=2}^{s} b_i(Z)g_{i1}^2\alpha_{i1}^2\mathbb{P}_{i1}\mathbb{P}_{i3} = 4\varphi_5(Z)$	5
C5	$\sum_{i=2}^{s} b_i(Z)\alpha_{i1}^2 g_{i1}^2\mathbb{P}_{i2}^2 = 3!\varphi_5(Z)$	5
C6	$\sum_{i=2}^{s} b_i(Z)g_{i1}\alpha_{i1}^3\mathbb{P}_{i1}^2\mathbb{P}_{i2} = 4!\varphi_5(Z)$	5
C7	$\sum_{i=2}^{s} b_i(Z)\alpha_{i1}^4\mathbb{P}_{i1}^4 = 4!\varphi_5(Z)$	5
C8	$\sum_{i=2}^{s} b_i(Z)\alpha_{i1}^2\mathbb{P}_{i1}K\Psi_i(Z) = 0$	5

Table 4.1: Stiff order conditions for EPIRK methods. Note that Z and K are arbitrary square matrices and Ψ_i is given by (4.27)

Assumption 3. *Suppose the coefficients of an EPIRK scheme satisfy one of the following for each i and all k*

$$\alpha_{i1}p_{i1k} = g_{i1} \quad or \quad \alpha_{i1} = g_{i1} \quad or \quad p_{i1k} = g_{i1}. \tag{4.33}$$

Given this assumption we then need to modify Lemma 4.5 and its proof as described in the Appendix. With this modification the convergence is proved exactly as in [46] and results in the following Theorem 6 (see Theorem 4.1 in [46]).

Theorem 6. *Let the initial value problem* (1.1) *satisfy Assumptions 1 and 2. Consider for its numerical solution an explicit exponential propagation iterative method of Runge-Kutta type (4.4) that fullfills Assumption 3 and the order conditions of Table 4.1 up to order p for some $3 \leq p \leq 5$. Then, under the stability assumption (4.16 [46]), the method converges with order p. In particular, the numerical solution satisfies the error bound*

$$\| u_n - u(t_n) \| \leq C \sum_{\nu=0}^{n-1} h_\nu^{p+1} \tag{4.34}$$

uniformly on $t_0 \leq t_n \leq T$. The constant C is independent of the chosen step size sequence.

Using Theorem 6 we now construct specific stiffly accurate methods of order four and five.

4.3 Construction of new schemes

In this section we demonstrate how the flexibility of coefficients in the EPIRK framework can be used to construct efficient stiffly accurate schemes. The improvement of the computational cost comes from constructing a method particularly optimized given an algorithm for evaluating the exponential matrix function-vector products. Evaluation of exponential-like matrix functions and

vector products $\psi(A)v$ constitutes the largest computational cost of an exponential integrator. The EPIRK methods have been originally introduced to minimize the number of such estimates required per time step as well as reduce the cost of each of these evaluations by carefully selecting the exponential functions in the products [65, 38, 40, 59]. In [69] the authors derived particularly efficient EPIRK methods that use the adaptive Krylov algorithm to approximate products $\psi(A)v$. While there have been several techniques introduced to evaluate $\psi(A)v$, the adaptive Krylov algorithm remains one of the most general cost efficient way to estimate these terms if no a priori information is available about the spectrum of A. Thus we adopt using this algorithm in constructing the new stiffly accurate EPIRK methods here.

We begin with a description of our method of choice, the adaptive Krylov algorithm, and discuss the structural requirements application of this method imposes on a time integrator. A systematic approach to solving the order conditions is then offered to derive appropriate three-stage methods of order four and five. Specific EPIRK schemes are constructed following this technique.

Adaptive Krylov algorithm and its implementations

The computational cost of the standard Krylov-projection based algorithm to approximate terms of type $\psi_{ij}(g_{ij}h_nJ_n)v$ scales as $O(m^2)$ where m is the size of the Krylov basis required to achieve a prescribed accuracy. The value m, in turn, depends on the spectrum of the matrix and it is expected that the computational cost of the Krylov projection will increase with the time step size h_n. Thus, for a given error tolerance it might actually be more efficient to integrate with a smaller time step rather than encounter large Krylov bases. An alternative and more efficient approach was proposed in [63, 50]. The adaptive Krylov algorithm seeks to evaluate linear combinations of type

$$\varphi_0(A)b_0 + \varphi_1(A)b_1 + \varphi_2(A)b_2 + \cdots + \varphi_p(A)b_p \tag{4.35}$$

where $A \in \mathbb{R}^{N\times N}$ and $b_i \in \mathbb{R}^N$ for $i = 0, \cdots, p$. The idea is to replace computing one large Krylov subspace of size m with a finite number of smaller Krylov subspaces of sizes m_1, m_2, ..., m_K. In [50] it was observed that expressions like (4.35) can be computed in a way that replaces the evaluation of terms like $\varphi_p(A)b_P$ with something that requires fewer Krylov vectors, like $\varphi_p(\tau_k A)b_p$ with $0 < \tau_k < 1$. For example, consider the following discretization of the interval $[0, 1]$, $0 = t_0 < t_1 < \cdots < t_k < t_k + \tau_k < \cdots < t_K = t_{end} = 1$. Then we would have to compute K Krylov projections at computational cost proportional to $O(m_1^2) + O(m_2^2) + ... + O(m_K^2)$ which can be more efficient than computing $\varphi_p(A)b_p$ that has computational complexity of $O(m^2)$. Obviously if K gets too large, the total cost of computing K smaller Krylov subspaces may exceed the cost of computing only one large Krylov basis. Therefore the efficiency of the algorithm is dependent on the choice of step sizes τ_k. As the optimal choice varies, an algorithm was developed in [50] to choose these step sizes adaptively. Further details and information can be found in [50, 63, 69, 40].

Vertical exponential-Krylov methods.

In [69], specific EPIRK methods were designed to efficiently employ the adaptive Krylov algorithm. By enforcing the requirement $\psi_{ij}(z) = \varphi_k(z)$ for fixed j and $i = 2, \ldots, s-1$, we can construct an s-stage adaptive Krylov based EPIRK scheme that requires only s Krylov projections per time step[69, 40]. This requirement is necessary since if a linear combination (4.35) consists of a single term $\varphi_k(tJ)b_k$, it can be computed as $\varphi_k(tJ)b_k = u(t)/t^k$ for any real value t. Hence we can compute $\varphi_k(g_{ij}hJ)b_k = u(g_{ij})/g_{ij}^k$ for fixed j with just one Krylov evaluation (as long as g_{ij} are included in the set of times $\{t_k\}_{k=0}^K$). We refer to this implementation as "vertical exponential-adaptive-Krylov" or "vertical exponential-Krylov" due to computing the "columns" or the terms of each stage with shared vectors b_k with one Krylov subspace. Note that the last term of the last stage is not restricted to a single φ-function but rather can be any linear combination of φ-functions since it does not share a vector with any terms of the previous stages.

The flexibility and choice of g_{ij} coefficients in EPIRK methods allows for further improvement in overall computational cost. In particular, by taking these coefficients to be smaller than 1 we would effectively reduce the size of the Krylov basis since $t_{end} < 1$. When implementing vertical adaptive Krylov we have for each fixed $j = 1, \ldots, s$, $t_{end} = \max_{i\in[2,j+1]} g_{ij}$. Therefore an efficient vertical Krylov EPIRK scheme has $\max_i g_{ij} < 1$ for each $j = 2, \ldots, s$. The classical order conditions offer enough freedom to choose small g-coefficients. However, the same approach is difficult to use to construct the stiffly accurate methods since the stiff order conditions require that $\max_i g_{ij} = g_{(s+1)j} = 1$ as shown in Lemma 7.

Lemma 7. *An s-stage EPIRK method satisfying the stiff order conditions must have $g_{(s+1)k} = 1$ for $k = 2, \ldots, s$.*

Proof. It must be the case that there is at least one $b_j(Z)$ such that $b_j \not\equiv 0$ for some $j = 2, \ldots, s$, otherwise the method can only be of order two. We can then solve conditions (C1)-(C3) for the non-zero $b_j(Z)$ functions. The resulting $b_j(Z)$ are simply linear combinations of $\varphi_3(Z)$, $\varphi_4(Z)$ functions. By substituting these solution(s) for b_j's into (4.5) we find that

$$\psi_{(s+1)k}(g_{(s+1)k}Z) = A_3\varphi_3(Z) + A_4\varphi_4(Z), \qquad \text{for some } A_i \in \mathbb{R}$$

Since this must holds for all $Z \in \mathbb{R}^{n\times n}$, we can conclude that $g_{(s+1)k} = 1$. □

Even though the stiff order conditions are restrictive with respect to the g-coefficients in the last stage, we still have flexibility with the g's in the internal stages and will use them to reduce the computational cost. Our approach is to modify the implementation of the adaptive Krylov algorithm from computing "vertically" to computing "horizontally" or in a "mixed" way as described below.

Horizontal exponential-adaptive-Krylov.

The "horizontal" exponential-adaptive-Krylov, or exponential-Krylov, algorithm is intended to compute all terms in each stage with one Krylov evaluation. In contrast to the vertical Krylov,

here we compute along the "rows" (i.e. compute $\psi_{i_0 j}(z)b_j$ for fixed i_0 and $j = 1, \ldots, i_0 - 1$). Since each term will have a different vector b_j, the only way for a s-stage method to require only s Krylov evaluations per time-step is to enforce the condition that any non-zero exponential terms in a given stage must share the same g_{ij}-value. As an example let us consider the following internal stage of a five-stage EPIRK method

$$\begin{aligned} U_{n5} = u_n + \alpha_{51}\psi_{51}(g_{51}h_nJ_n)h_nf(u_n) + \alpha_{52}\psi_{52}(g_{52}h_nJ_n)h_n\Delta r(u_n) + 0\cdot\psi_{53}(g_{53}h_nJ_n)\Delta^2 r(u_n) + \\ + \alpha_{54}\psi_{54}(g_{54}h_nJ_n)\Delta^3 r(u_n) \end{aligned} \tag{4.36}$$

where $\psi_{51}(z) = \varphi_1(z), \psi_{52}(z) = \varphi_2(z) + \varphi_3(z)$, and $\psi_{54}(z) = \varphi_3(z)$. Using the recurrence relation (4.25) we can express (4.36) as

$$\begin{aligned} U_{n5} &= u_n + \alpha_{51}\Big(\Big(\varphi_3(g_{51}h_nJ_n)g_{51}h_nJ_n + \tfrac{1}{2}\Big)g_{51}h_nJ_n + 1\Big)h_nf(u_n) \\ &\quad + \alpha_{52}\left((\varphi_3(g_{52}h_nJ_n)g_{52}h_nJ_n + \tfrac{1}{2}) + \varphi_3(g_{52}h_nJ_n)\right)h_n\Delta r(u_n) \\ &\quad + 0\cdot h_n\Delta^2 r(u_n) + \alpha_{55}\varphi_3(g_{54}h_nJ_n)h_n\Delta^3 r(u_n) \\ &= u_n + b_0 + \varphi_3(g_{51}h_nJ_n)b_1 + \varphi_3(g_{52}h_nJ_n)b_2 + 0 + \varphi_3(g_{54}h_nJ_n)b_4 \end{aligned} . \tag{4.37}$$

Since the vectors b_1, b_2, b_4 are not the same we must have $g_{51} = g_{52} = g_{54}$ so (4.37) can be written in the form

$$U_{n5} = u_n + \varphi_3(g_{51}h_nJ_n)(b_1 + b_2 + b_4). \tag{4.38}$$

Then the adaptive Krylov algorithm can be used to compute (4.38) with the possibility of taking $g_{51} < 1$. Note that due to Lemma 7 all projections in the vertical method require adaptive Krylov algorithm to integrate over the interval $[0, 1]$. Thus the savings associated with smaller g coefficients which are equivalent to reducing the integration interval in adaptive Krylov algorithm to $[0, g]$ are not possible for the vertical methods. Thus the horizontal version of the method with coefficients $g_{i1} < 1$ carries computational savings comparable to the vertical method.

Mixed exponential-adaptive-Krylov.

We can also use a combination of both vertical and horizontal Krylov adaptation to develop an EPIRK scheme. The idea is to compute the last stage horizontally and the internal stages vertically or with a combination of vertical and horizontal approaches depending on what yields the most optimized scheme. This alleviates the drawbacks of strictly implementing vertical Krylov or horizontal Krylov for stiffly accurate EPIRK schemes and opens more possibilities for customization of methods as well as improving the efficiency of a particular scheme. After solving the order conditions below, we will construct a specific mixed exponential-adaptive-Krylov, or exponential-Krylov, method that will serve as an illustrative example of this idea.

Note that any EPIRK method can be implemented in a vertical, horizontal or mixed way, however, the schemes can be constructed to be particularly optimized for a given implementation. Thus, for example, we will call an EPIRK scheme optimized for a mixed implementation a mixed exponential-Krylov method but in the numerical tests section we test such method with either vertical, horizontal or mixed implementation and demonstrate that the integrator optimized for the mixed implementation and applied in this way is the most efficient.

Solving the order conditions

Our primary objective is to construct new and more efficient stiffly accurate EPIRK schemes. We focus on constructing three-stage methods. Below we solve the order conditions given in Table 4.1 to obtain new general classes of three-stage fourth and fifth-order EPIRK methods. For each of these classes, the remaining free-parameters are then chosen to obtain specific schemes each targeted for a specific version (vertical, horizontal or mixed) of the adaptive Krylov algorithm. We begin by considering a three-stage EPIRK method in EXPRB form (4.7)

$$\begin{aligned} U_{n2} &= u_n + \alpha_{21}\psi_{21}(z)(g_{21}h_nJ_n)h_nf(u_n) \\ U_{n3} &= u_n + \alpha_{31}\psi_{31}(z)(g_{31}h_nJ_n)h_nf(u_n) + \underbrace{\alpha_{32}\psi_{32}(g_{32}h_nJ_n)}_{a_{32}(h_nJ_n)}h_nr(U_{n2}) \\ u_{n+1} &= u_n + \varphi_1(z)(h_nJ_n)h_nf(u_n) + \underbrace{(\beta_2\psi_{42}(g_{42}h_nJ_n) - 2\beta_3\psi_{43}(g_{43}h_nJ))}_{b_2(h_nJ_n)}h_nr(U_{n2}) + , \\ &\quad + \underbrace{\beta_3\psi_{43}(h_ng_{43})}_{b_3(h_nJ_n)}h_nr(U_{n3}) \end{aligned} \tag{4.39}$$

where $\psi_{i1}(z) = p_{i11}\varphi_1(z) + p_{i12}\varphi_2(z) + p_{i13}\varphi_3(z)$ for $i = 2, 3$.

Fourth-order methods

From the conditions given in Table 4.3, (C1) and (C2) yield solutions for $b_2(z)$ and $b_3(z)$:

$$\begin{aligned} b_2(z) &= \frac{2g_{31}\mathbb{P}_{32}}{\alpha_{21}^2\mathbb{P}_{21}(g_{31}\mathbb{P}_{21}\mathbb{P}_{32}-g_{21}\mathbb{P}_{22}\mathbb{P}_{31})}\varphi_3(z) - \frac{3\mathbb{P}_{31}}{\alpha_{21}^2\mathbb{P}_{21}(g_{31}\mathbb{P}_{21}\mathbb{P}_{32}-g_{21}\mathbb{P}_{22}\mathbb{P}_{31})}\varphi_4(z) \\ b_3(z) &= \frac{2g_{21}\mathbb{P}_{22}}{\alpha_{31}^2\mathbb{P}_{31}(g_{21}\mathbb{P}_{22}\mathbb{P}_{31}-g_{31}\mathbb{P}_{21}\mathbb{P}_{32})}\varphi_3(z) - \frac{3\mathbb{P}_{21}}{\alpha_{31}^2\mathbb{P}_{31}(g_{21}\mathbb{P}_{22}\mathbb{P}_{31}-g_{31}\mathbb{P}_{21}\mathbb{P}_{32})}\varphi_4(z) \end{aligned}, \tag{4.40}$$

with $\mathbb{P}_{ij}$'s defined by (4.24). Upon substituting (4.40) into (C3) we obtain

$$\frac{2(\alpha_{31}g_{21}\mathbb{P}_{22}\mathbb{P}_{31}^2 - \alpha_{21}g_{31}\mathbb{P}_{21}^2\mathbb{P}_{32})\varphi_3(z) + (3\mathbb{P}_{21}(2g_{31}\mathbb{P}_{32} - \alpha_{31}\mathbb{P}_{31}^2 + \alpha_{21}\mathbb{P}_{21}\mathbb{P}_{31}) - 6g_{21}\mathbb{P}_{22}\mathbb{P}_{31})\varphi_4(z)}{g_{21}\mathbb{P}_{22}\mathbb{P}_{31} - g_{31}\mathbb{P}_{21}\mathbb{P}_{32}} = 0.$$

This condition is satisfied by enforcing the coefficients of φ_3 and φ_4 to be zero. The resulting equations can then be used to solve for

$$\alpha_{21} = \frac{2g_{21}\mathbb{P}_{22}}{\mathbb{P}_{21}^2} \qquad \text{and} \qquad \alpha_{31} = \frac{2g_{31}\mathbb{P}_{32}}{\mathbb{P}_{31}^2}. \tag{4.41}$$

With (4.40) and (4.41) satisfied, methods of the form (4.39) define a new class of stiffly accurate fourth-order three-stage methods. The flexibility with the remaining parameters makes these methods appealing. For example, the choice $a_{32}(Z) \equiv 0$ and $\psi_{21}(Z) = \psi_{31}(Z) = \varphi_1(Z)$ yields the structure necessary in order to construct a vertical, horizontal, or a mixed exponential-Krylov method. These methods will require three Krylov projections per time-step when implementing the vertical or horizontal whereas it is possible to construct a mixed exponential-Krylov scheme

that only requires two projections. The removal of a whole projection each time-step can significantly reduce the overall cost. Another computation saving feature of the fourth-order schemes is the ability to choose both g_{21} and g_{31}. The choice $g_{21} = \frac{1}{2}$ and $g_{31} = \frac{2}{3}$ leads to the following three-stage fourth-order method *EPIRK4s3A*:

$$\begin{aligned} U_{n2} &= u_n + \frac{1}{2}\varphi_1(\frac{1}{2}h_nJ_n)h_nf(u_n) \\ U_{n3} &= u_n + \frac{2}{3}\varphi_1(\frac{2}{3}h_nJ_n)h_nf(u_n) \\ u_{n+1} &= u_n + \varphi_1(h_nJ_n)h_nf(u_n) + (32\varphi_3(h_nJ_n) - 144\varphi_4(h_nJ_n))\,h_nr(U_{n2}) + \\ &\quad + \left(-\frac{27}{2}\varphi_3(h_nJ_n) + 81\varphi_4(h_nJ_n)\right)h_nr(U_{n3}) \end{aligned} . \tag{4.42}$$

Another fourth-order method can be obtained similarly by taking $\psi_{21}(Z) = \psi_{31}(Z) = \varphi_2(Z)$. Different g-coefficients were specified but were chosen so that they are comparable to those above to obtain the *EPIRK4s3B* method

$$\begin{aligned} U_{n2} &= u_n + \frac{2}{3}\varphi_2(\frac{1}{2}h_nJ_n)h_nf(u_n) \\ U_{n3} &= u_n + \varphi_2(\frac{3}{4}h_nJ_n)h_nf(u_n) \\ u_{n+1} &= u_n + \varphi_1(h_nJ_n)h_nf(u_n) + (54\varphi_3(h_nJ_n) - 324\varphi_4(h_nJ_n))\,h_nr(U_{n2}) + \\ &\quad + (-16\varphi_3(h_nJ_n) + 144\varphi_4(h_nJ_n))\,h_nr(U_{n3}). \end{aligned} \tag{4.43}$$

Note that method *EPIRK4s3B* lies outside of the set of exponential Rosebrock methods because it is using $\varphi_2(z)$ function in the internal stages. In the numerical experiments section we will show the performance of *EPIRK4s3B* is very similar with *EPIRK4s3A*. This illustrates that the EPIRK form allows for more flexibility in constructing the methods.

In Section 4.4 it will be shown that (4.42) performs particularly well when implemented in a horizontal or a mixed exponential-Krylov way. The flexibility in constructing this fourth order method allows building a scheme that can even be computationally favorable compared to three-stage fifth order methods as shown below.

Fifth-order methods

Our construction of methods of order five are built upon the solutions obtained above which satisfy conditions (C1)-(C3). A fifth-order method must additionally satisfy (C4)-(C8); upon inserting (4.40) it is found that there is no solution which is able to satisfy these conditions for all $Z \in \mathbb{R}^{n\times n}$. However, as noted in [46] for EXPRB methods, with additional regularity assumptions the convergence results hold under weaker assumptions on the coefficients of the method. Similar results can be obtained for the general EPIRK form by considering a simplified set of conditions (C4*)-(C8*) given in Table 4.2 that replace conditions (C4)-(C8). The regularity assumption on operators in (4.9) needed to prove convergence in this case is the same as for EXPRB methods but stated in a

less compact form as follows.

Assumption 4. *The operator L and the nonlinearity N from* (4.8) *are such that*

$$L(\tfrac{\partial^2 \tilde{N}_n}{\partial y^2}(\tilde{u}_n)(\tilde{u}_n', \tilde{u}_n^{(3)})), \quad L(\tfrac{\partial^2 \tilde{N}_n}{\partial y^2}(\tilde{u}_n)(\tilde{u}_n'', \tilde{u}_n'')), \quad L(\tfrac{\partial^3 \tilde{N}_n}{\partial u^3}(\tilde{u}_n)(\tilde{u}_n', \tilde{u}_n', \tilde{u}_n'')), \quad L\tfrac{\partial^4 \tilde{N}_n}{\partial u^4}(\tilde{u}_n)(\tilde{u}_n', \tilde{u}_n', \tilde{u}_n', \tilde{u}_n')$$
$$L(\tfrac{\partial^2 \tilde{N}_n}{\partial u^2}(\tilde{u}_n)\left(\tilde{u}_n', \Psi_i(h\tilde{J}_n)\tfrac{\partial^2 \tilde{N}_n}{\partial u^2}(\tilde{u}_n)(\tilde{u}_n', \tilde{u}_n')\right)) \tag{4.44}$$

are uniformly bounded on $\mathcal{X}$ for all $2 \le i \le s$.

Given Assumption 4, we can now prove convergence of methods that satisfy conditions (C4*)-(C8*), if the stability requirement 4.16 in [46] holds. The major aspects of the proof are exactly the same as in [46] but several modifications are needed as we describe below.

Theorem 8 (Theorem 4.2 [46])**.** *Let the initial value problem (1.1) satisfy Assumptions 1,2, and 4. Consider for its numerical solution an EPIRK method (4.4) which satisfies Assumption 3, the order conditions (C1)-(C3) of Table 4.1 and (C4*)-(C8*) of Table 4.2. Then, under the stability assumption (4.16 [46]), the method is convergent of order 5. In particular, the numerical solution u_n satisfies the error bound*

$$\| u_n - u(t_n) \| \le C \sum_{\nu=0}^{n-1} h_\nu^6 \tag{4.45}$$

uniformly on $t_0 \le t_n \le T$. The constant C is independent of the chosen step size sequence.

Proof. We begin by defining terms of $\mathcal{O}(h_n^5)$ in (4.32) by

$$\begin{aligned}
\xi_{5,1}(z) &= \sum_{i=2}^{s} b_i(z) g_{i1}^2 \alpha_{i1}^2 \mathbb{P}_{i1}\mathbb{P}_{i3} - 4\varphi_5(z) \\
\xi_{5,2}(z) &= \sum_{i=2}^{s} b_i(z) \frac{1}{2!} \alpha_{i1}^2 g_{i1}^2 \mathbb{P}_{i2}^2 - 3\varphi_5(z) \\
\xi_{5,3}(z) &= \sum_{i=2}^{s} b_i(z) \frac{1}{2!} g_{i1} \alpha_{i1}^3 \mathbb{P}_{i1}^2 \mathbb{P}_{i2} - 6\varphi_5(z) \\
\xi_{5,4}(z) &= \sum_{i=2}^{s} b_i(z) \frac{1}{4} \alpha_{i1}^4 \mathbb{P}_{i1}^4 - \varphi_5(z),
\end{aligned} \tag{4.46}$$

each of which can be written in the following form

$$\xi_{5,j}(z) = \sum_{i=2}^{s} b_i(z) C_{1,j} - C_{2,j}\varphi_5(z)$$

Ref. label	Order Condition	Order
(C4*)	$\sum_{i=2}^{s} b_i(0) g_{i1}^2 \alpha_{i1}^2 \mathbb{P}_{i1}\mathbb{P}_{i3} = 4\varphi_5(0)$	5
(C5*)	$\sum_{i=2}^{s} b_i(0) \alpha_{i1}^2 g_{i1}^2 \mathbb{P}_{i2}^2 = 3!\varphi_5(0)$	5
(C6*)	$\sum_{i=2}^{s} b_i(0) g_{i1} \alpha_{i1}^3 \mathbb{P}_{i1}^2 \mathbb{P}_{i2} = 12\varphi_5(0)$	5
(C7*)	$\sum_{i=2}^{s} b_i(0) \alpha_{i1}^4 \mathbb{P}_{i1}^4 = 4!\varphi_5(0)$	5
(C8*)	$\sum_{i=2}^{s} b_i(0) \alpha_{i1}^2 \mathbb{P}_{i1} K\Psi_i(Z) = 0$	5

Table 4.2: Simplified stiff order conditions. Note that Z and K are arbitrary square matrices, Ψ_i is given by (4.27),and $\psi_{3,i}(Z) = \sum_{j=2}^{i-1} a_{ij}(Z)\frac{c_j^2}{2} - c_i^3\varphi_3(c_i Z)$

where $C_{1,j}, C_{2,j} \in \mathbb{R}$ and $j = 1, \ldots, 4$. Using the recurrence relation (4.25) we have for each $j = 1, \ldots, 5$

$$\begin{aligned}
\xi_{5,j}(z) - \xi_{5,j}(0) &= \sum_{i=2}^{s} \left(b_i(z) - b_i(0)\right) C_{1,j} + \left(\varphi_5(z) - \varphi_5(0)\right) C_{2,j} \\
&= \sum_{i=2}^{s} z\widehat{b_i}(z) C_{1,j} + z\varphi_6(z) C_{2,j} \qquad (4.47) \\
&= z\left(\sum_{i=2}^{s} \widehat{b_i}(z) C_{1,j} + \varphi_6(z) C_{2,j}\right) \\
&= z\widehat{\xi_{5,j}}(z) \qquad (4.48)
\end{aligned}$$

where $\widehat{\xi_{5,j}}$ is a bounded operator. Since the simplified conditions (C4*)-(C8*) are satisfied, we have $\xi_{5,j}(h_n\tilde{J}_n) = 0 + h_n\tilde{J}_n\widehat{\xi_{5,j}}(h_n\tilde{J}_n)$. Substituting this back into (4.32) and using Assumption 4 yields each term to be of order $O(h_n^6)$. Therefore our error $\tilde{e}_{n+1} = O(h_n^6)$. □

As a result of Theorem 8, a stiffly accurate three-stage fifth-order EPIRK scheme can be constructed. Let us consider the solutions obtained for the fourth-order schemes and (C4*)-(C8*). We first note that the simplified conditions (C6*) and (C7*) become equivalent and are used to solve for g_{21}:

$$g_{21} = \frac{3\mathbb{P}_{21}\left(2\mathbb{P}_{31} - 5g_{31}\mathbb{P}_{32}\right)}{5\mathbb{P}_{22}\left(3\mathbb{P}_{31} - 8g_{31}\mathbb{P}_{32}\right)}. \qquad (4.49)$$

Substituting (4.49) into simplified conditions (C4*) & (C5*) and using Mathematica software we found that there is only one solution that yields desirable coefficients (i.e. $g_{ij} \in (0, 1]$ and $\mathbb{P}_{ij} \in \mathbb{R}$) and is able to satisfy the remaining conditions

$$\begin{aligned}
&\mathbb{P}_{21} = 2\mathbb{P}_{22},\ \mathbb{P}_{32} = -\frac{\mathbb{P}_{31}}{2}, \\
&\mathbb{P}_{33} = \frac{\mathbb{P}_{31}\left(80g_{31}^2\mathbb{P}_{22} - 225g_{31}^2\mathbb{P}_{23} + 120g_{31}\mathbb{P}_{22} - 360g_{31}\mathbb{P}_{23} + 48\mathbb{P}_{22} - 144\mathbb{P}_{23}\right)}{30g_{31}^2\mathbb{P}_{22}}
\end{aligned} \qquad (4.50)$$

Expressions for p_{ijk} coefficients are then found by replacing $\mathbb{P}_{ij}$'s in (4.50) with (4.24), resulting in two possibilities

$$p_{211} = -\frac{p_{212}\left(225g_{31}^2p_{311}+40g_{31}^2p_{312}-360g_{31}p_{311}-60g_{31}p_{312}+144p_{311}+24p_{312}\right)}{15g_{31}^2p_{312}}, p_{213} = -2p_{212}, p_{313} = 2p_{312} \quad (4.51)$$

or

$$p_{212} = 0, p_{213} = 0, p_{312} = 0, p_{313} = 0. \quad (4.52)$$

Expressions in (4.51) give rise to the use of general ψ_{i1} functions and the ability to construct horizontal exponential-Krylov methods. The coefficients given by (4.52) simplify (4.39) by setting $\psi_{i1}(z) = \varphi_1(z)$. This simplification condition provides these methods with the structure necessary for construction of a mixed exponential-Krylov scheme. For this reason we consider each case separately and construct methods specifically designed for the two different approaches - horizontal and mixed.

Beginning with (4.51), the remaining condition (C8*) is

$$b_2(0)\alpha_{21}^2\mathbb{P}_{21}\cdot\Psi_2(Z) + b_3(0)\alpha_{31}^2\mathbb{P}_{31}\cdot\Psi_3(Z) = 0, \quad (4.53)$$

which is satisfied by

$$\begin{aligned} a_{32} = {} & \frac{2g_{21}g_{31}p_{211}P_{32}\left(8g_{31}P_{32}-3P_{31}\right)\phi_{3,g_{21}}}{\alpha_{21}P_{22}P_{31}^2\left(3P_{21}-8g_{21}P_{22}\right)} + \\ & + \frac{2g_{31}\phi_{3,g_{31}}\left(3\alpha_{31}g_{31}p_{311}P_{21}P_{22}P_{31}^2 - 8\alpha_{31}g_{21}g_{31}p_{311}P_{22}^2P_{31}^2\right)}{\alpha_{21}^2P_{21}^2P_{22}P_{31}^2\left(3P_{21}-8g_{21}P_{22}\right)} \end{aligned} \quad (4.54)$$

Horizontal exponential-Krylov methods.

After substituting (4.50) and (4.51) into (4.54), we have

$$\begin{aligned} a_{32}(Z) = {} & \frac{100\left(3-4g_{31}\right)^2g_{31}^3p_{311}}{3\left(4-5g_{31}\right)^2\left(6p_{311}+p_{312}\right)}\varphi_3(g_{31}Z) + \\ & + \frac{20\left(3-4g_{31}\right)^2g_{31}\left(5g_{31}^2\left(45p_{311}+8p_{312}\right)-60g_{31}\left(6p_{311}+p_{312}\right)+24\left(6p_{311}+p_{312}\right)\right)}{9\left(4-5g_{31}\right)^2\left(6p_{311}+p_{312}\right)}\varphi_3(g_{21}Z) \end{aligned}. \quad (4.55)$$

Since $a_{32}(Z)$ is a function of both g_{31} and g_{21}, a horizontal exponential-Krylov method is not possible unless one of the terms vanish (since $g_{21} \neq g_{31}$). We thus introduce a new condition by setting one of these terms to zero. The first term in the second internal stage already uses g_{31} in the evaluation of $\varphi_1(Z)$ and therefore we seek removing the term associated with $\varphi_3(g_{21}Z)$ in (4.55). With the use of Mathematica, only one solution was found which did not violate any of the conditions or specifications,

$$p_{312} = -\frac{9\left(5g_{31}-4\right)^2p_{311}}{4\left(10g_{31}^2-15g_{31}+6\right)}.$$

Ref. label	Order Condition	Order
C1	$\sum_{i=2}^{s} b_i(Z)\alpha_{i1}^2\mathbb{P}_{i1}^2 = 2\varphi_3(Z)$	3
C2	$\sum_{i=2}^{s} b_i(Z)\alpha_{i1}^2 g_{i1}\mathbb{P}_{i1}\mathbb{P}_{i2} = 3\varphi_4(Z)$	4
C3	$\sum_{i=2}^{s} b_i(Z)\alpha_{i1}^2\mathbb{P}_{i1}^3 = 3!\varphi_4(Z)$	4
C4	$\sum_{i=2}^{s} b_i(Z)g_{i1}^2\alpha_{i1}^2\mathbb{P}_{i1}\mathbb{P}_{i3} = 4\varphi_5(Z)$	5
C5	$\sum_{i=2}^{s} b_i(Z)\alpha_{i1}^2 g_{i1}^2\mathbb{P}_{i2}^2 = 3!\varphi_5(Z)$	5
C6	$\sum_{i=2}^{s} b_i(Z)g_{i1}\alpha_{i1}^3\mathbb{P}_{i1}^2\mathbb{P}_{i2} = 12\varphi_5(Z)$	5
C7	$\sum_{i=2}^{s} b_i(Z)\alpha_{i1}^4\mathbb{P}_{i1}^4 = 4!\varphi_5(Z)$	5
C8	$\sum_{i=2}^{s} b_i(Z)\alpha_{i1}^2\mathbb{P}_{i1}K\Psi_i(Z) = 0$	5

Ref. label	Order Condition	Order
C1′	$\sum_{i=2}^{s} b_i(Z)c_i^2 = 2!\varphi_3(Z)$	3
C2′	$\sum_{i=2}^{s} b_i(Z)c_i^3 = 3!\varphi_4(Z)$	4
C3′	$\sum_{i=2}^{s} b_i(Z)c_i^4 = 4!\varphi_5(Z)$	5
C4′	$\sum_{i=2}^{s} c_i b_i(Z)K\psi_{3,i}(Z) = 0$	5

Table 4.3: Stiff order conditions for EPIRK methods and EXPRB. Note that Z and K are arbitrary square matrices, Ψ_i is given by (4.27),and $\psi_{3i}(Z) = \sum_{j=2}^{i-1} a_{ij}(Z)\frac{c_j^2}{2} - c_i^3\varphi_3(c_iZ)$

With all conditions satisfied, the remaining parameters g_{31}, p_{212}, and p_{311} can be chosen freely. In order to optimize our g-coefficients, we use (4.49) to help choose g_{31}. The relationship (4.49) can be reduced under the conditions for a fifth-order method to

$$g_{21} = \frac{3\,(5g_{31} - 4)}{5\,(4g_{31} - 3)}.$$

From the plot of this relationship in Figure 4.1 it can be seen that minimizing either g_{21} or g_{31} results in the other coefficient approaching eight tenths. Furthermore, Figure 4.1 shows that minimizing one of these coefficients is the best choice computationally. Thus let us specify $g_{31} = 4/9$, $p_{212} = 1$

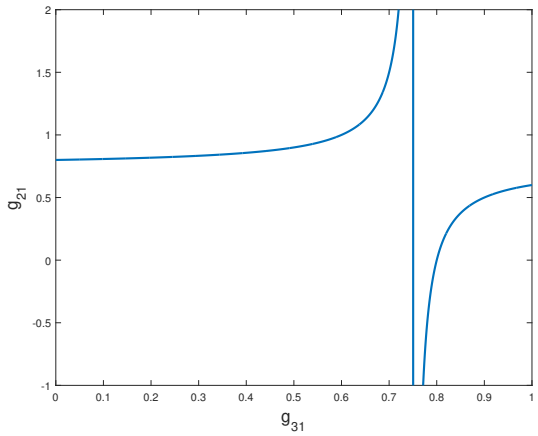


Figure 4.1: g_{21} and g_{31}

and $p_{311} = 1$ to obtain our first horizontal exponential-Krylov friendly method *EPIRK5s3*-Horz:

$$
\begin{aligned}
U_{n2} &= u_n + \frac{288}{55}\left(\varphi_2(\frac{48}{55}h_nJ_n) - 2\varphi_3(\frac{48}{55}h_nJ_n)\right)h_nf(u_n) \\
U_{n3} &= u_n + \frac{212}{45}\left(\varphi_1(\frac{4}{9}h_nJ_n) - \frac{288}{53}\varphi_2(\frac{4}{9}h_nJ_n) + \frac{576}{53}\varphi_3(\frac{4}{9}h_nJ_n)\right)h_nf(u_n) + \\
&\quad + \frac{32065}{13122}\varphi_3(\frac{4}{9}h_nJ_n)h_nr(U_{n2}) \qquad , \qquad (4.56) \\
u_{n+1} &= u_n + \varphi_1(h_nJ_n)h_nf(u_n) + \left(-\frac{166375}{61056}\varphi_3(h_nJ_n) + \frac{499125}{27136}\varphi_4(h_nJ_n)\right)h_nr(U_{n2}) \\
&\quad + \left(\frac{2187}{106}\varphi_3(h_nJ_n) - \frac{2187}{106}\varphi_4(h_nJ_n)\right)h_nr(U_{n3})
\end{aligned}
$$

Mixed exponential-Krylov methods.

For a mixed exponential-Krylov method, we propose using vertical exponential-Krylov approach to compute the internal stages and horizontal exponential-Krylov method for the last stage (see Table 4.4). For a three-stage method, this type of a mixed exponential-Krylov method requires $\Psi_{21}(z) = \Psi_{31}(z) = \varphi_k(z)$ for some fixed $k \in \mathbb{N}$. The simplification from coefficients (4.52) satisfies this requirement with $k = 1$. Therefore we obtain a three-stage fifth-order mixed exponential-Krylov method by additionally satisfying

$$
a_{32}(Z) = \frac{2\alpha_{31}g_{31}^2p_{311}\varphi_3(g_{31}Z)}{\alpha_{21}^2p_{211}^2} + \frac{2\left(20g_{31}^2 - 31g_{31} + 12\right)g_{31}\varphi_3(g_{21}Z)}{\alpha_{21}p_{211}}.
$$

A further result from this simplification is that a stiffly accurate fifth-order three-stage EPIRK method is also a fifth-order three-stage EXPRB method. Thus any three-stage EXPRB scheme is

of a mixed exponential-Krylov type. As an example and for our numerical experiments we will consider a fifth-order three-stage method from [46], *EXPRB53s3* (Table 4.4).

$$
\begin{aligned}
U_{n2} &= u_n + \boxed{\tfrac{1}{2}\varphi_1(\tfrac{1}{2}h_nJ_n)h_nf(u_n)} \\
U_{n3} &= u_n + \boxed{\tfrac{9}{10}\varphi_1(\tfrac{9}{10}h_nJ_n)h_nf(u_n)} + \boxed{\left(\tfrac{27}{25}\varphi_3(\tfrac{1}{2}h_nJ_n) + \tfrac{729}{125}\varphi_3(\tfrac{9}{10}h_nJ_n)\right)h_nr(U_{n2})} \\
u_{n+1} &= u_n + \boxed{\varphi_1(h_nJ_n)h_nf(u_n) + (18\varphi_3(h_nJ_n) - 60\varphi_4(h_nJ_n))\,h_nr(U_{n2}) + \left(-\tfrac{250}{81}\varphi_3(h_nJ_n) + \tfrac{500}{27}\varphi_4(h_nJ_n)\right)h_nr(U_{n3})}
\end{aligned}
$$

Table 4.4: *EXPRB53s3* and grouping of terms for mixed exponential-adaptive-Krylov

Variable time-stepping

Variable time-stepping has been used with both the vertical implementation of EPIRK and EXPRB methods in [66] and [29] respectively. For both of these classes of methods the approach to implementing an efficient variable step-size mechanism was to embed a lower-order error estimator into a higher-order method in such a way that both rely on the same internal stages and do not require additional Krylov projections per time step. While this is possible for vertical Krylov implementation, the horizontal and mixed implementations are limited by computing the final stage horizontally where one Krylov projection is used for its approximation and accounts for the specific coefficients of that stage. Thus, the implementation of variable time-stepping in this manner will require an extra Krylov projection each time-step in order to calculate the error estimator. To further reduce computational cost of the horizontal and mixed implementations with variable time-stepping the adaptive Krylov algorithm has to be modified. This is the goal of our current research but in this thesis we restrict our attention to the existing adaptive Krylov algorithm as in [50].

Since the horizontal and vertical implementations require the same number of projections per time-step, the cost of an additional projection can offset any computational gains from optimized g-coefficients in the horizontal implementation. However, the mixed implementation of methods like (4.42) requires fewer projections each time-step than the vertical implementation of a method with the same number of stages. Therefore the extra Krylov evaluation for the mixed implementation would still be competitive with the vertical implementation due to now having the same number of projections each time-step. As an example we can embed the following third-order method into *EPIRK4s3*

$$\widehat{u}_{n+1} = u_n + \varphi_1(h_nJ_n)h_nf(u_n) + 8\varphi_3(h_nJ_n)h_nr(U_{n2}) \tag{4.57}$$

and use it as our error estimator for both vertical and mixed implementations.

To efficiently approximate (4.57), the vertical method needs to use the same Krylov bases that are computed each time-step for (4.42). In [69] methods were restricted to using single φ-functions for terms who shared the same vector. By modifying the implementation we can account

for multiple φ-functions and approximate the terms

$$(32\varphi_3(h_nJ_n) - 144\varphi_4(h_nJ_n))\, h_nr(U_{n2}) \quad \text{and} \quad 8\varphi_3(h_nJ_n)h_nr(U_{n2})$$

in (4.42) and (4.57) using the same Krylov basis. After computation of the Krylov basis for $\varphi_4(h_nJ_n)h_nr(U_{n2})$, an approximation of $\varphi_3(h_nJ_n)h_nr(U_{n2})$ can then be obtained by using the recurrence relation $\varphi_k(z) = z\varphi_{k+1}(z) + 1/k!$.

4.4 Numerical Experiments

The numerical experiments presented below are designed to address several objectives. First, we want to demonstrate the performance of the new stiffly accurate EPIRK schemes. Second, we will confirm our claim that implementing the horizontal and/or mixed exponential-adaptive-Krylov algorithm for stiffly accurate methods can offer significant computational savings. Third, we examine the accuracy of the integrators on problems which do not satisfy Assumptions 1 and 2. Finally, we conclude the section with tests that illustrate the performance of the variable time stepping version of the methods.

The integrators employed in our numerical experiments are: *EPIRK4s3A* (4.42), *EPIRK4s3B* (4.43), *EPIRK5s3* (4.56), *EXPRB53s3* (Table 4.4) and one classically (non-stiff) derived method *EPIRK5P1* from [69]. Each of the stiffly accurate methods will be implemented in its vertical, horizontal and mixed exponential-Krylov versions as follows:

- *EPIRK4s3A* - vertical, horizontal, and mixed; these three implementations of the same fourth-order method demonstrate the advantages of mixed or horizontal forms;
- *EPIRK4s3B* - mixed; this is an EPIRK method that cannot be written in the exponential Rosenbrock form;
- *EPIRK5s3* - horizontal; this fifth-order method has been derived specifically to take advantage of the horizontal form;
- *EXPRB5s3* - vertical and mixed; this fifth-order method has been designed to take advantage of a mixed form.

The classically derived *EPIRK5P1* method has been included to illustrate the difference in performance compared to the stiffly accurate and particular implementation adapted schemes.

We begin by describing the test problems that we will be using and verifying the theoretically predicted order of all the integrators used in our experiments. The simulations and results are then detailed.

Test problems

Our numerical experiments are conducted on a select subset of the test problems used in [38]. For each of these test problems, numerical comparisons between previously derived exponential

schemes not included here can be found in [65, 38, 26, 46]. In all of the problems presented below the ∇^2 term is discretized using the standard second order finite differences.

Allen-Cahn 2D. Two-dimensional Allen-Cahn equation [54]:

$$u_t = \alpha\nabla^2 u + u - u^3,\ x, y \in [-1, 1], t \in [0, 1.0]$$

with $\alpha = 0.1$, using no-flow boundary conditions and initial conditions given by $u = 0.1 + 0.1\cos(2\pi x)\cos(2\pi y)$.

ADR 2D. Two-dimensional advection-diffusion-reaction equation [8]:

$$u_t = \epsilon(u_{xx} + u_{yy}) - \alpha(u_x + u_y) + \gamma u(u - \tfrac{1}{2})(1 - u),\ x, y \in [0, 1], t \in [0, 0.1],$$

where $\epsilon = 1/100$, $\alpha = -10$, and $\gamma = 100$. Homogeneous Neumann boundary conditions were used and the initial conditions were given by $u = 256(xy(1 - x)(1 - y))^2 + 0.3$.

Brusselator 2D. Two-dimensional Brusselator [20, 37]

$$\begin{aligned} u_t &= 1 + u^2 v - 4u + \alpha\nabla^2 u,\ x, y \in [0, 1] \\ v_t &= 3u - u^2 v + \alpha\nabla^2 v \\ \alpha &= 0.02 \end{aligned}$$

with homogeneous Neumann boundary conditions, $t \in [0, 1]$, and initial values

$$\begin{aligned} u &= 2 + 0.25y \\ v &= 1 + 0.8x \end{aligned}$$

Gray-Scott 2D. Two-dimensional Gray-Scott [18] with periodic boundary conditions:

$$\begin{aligned} u_t &= d_u\nabla^2 u - uv^2 + a(1 - u),\ x, y \in [0, 1] \\ v_t &= d_v\nabla^2 v + uv^2 - (a + b)v, \end{aligned}$$

and $d_u = 0.2, d_v = 0.1, a = 0.04, b = 0.06$. Initial conditions given by

$$\begin{aligned} u &= 1 - e^{-150[(x-\frac{1}{2})^2 + (y-\frac{1}{2})^2]}, \\ v &= e^{-150[(x-\frac{1}{2})^2 + 2(y-\frac{1}{2})^2]}. \end{aligned}$$

1D Semilinear parabolic. One-dimensional semilinear parabolic problem [26]:

$$\frac{\partial U}{\partial t}(x, t) - \frac{\partial^2 U}{\partial x^2}(x, t) = \int_0^1 U(x, t)dx + \Phi(x, t)$$

with homogeneous Dirichlet boundary conditions and for $x \in [0, 1]$ and $t \in [0, 1]$. The source function Φ is chosen such that $U(x, t) = x(1 - x)e^t$ is the exact solution.

Verification of accuracy

The implementation of the exponential integrators was done in MATLAB. For all the integrators, the adaptive Krylov algorithm as described in [69] was used to compute products of matrix φ-functions and vectors. To verify the order of the schemes, we set the tolerance to 10^{-14} so that the Krylov approximation error has a minimal effect on the time stepping error. A reference solution was computed using MATLAB's *ode15s* integrator with absolute and relative tolerances set to 10^{-14} and the error was defined as the discrete infinity (maximum) norm of the difference between the computed solution and this approximation.

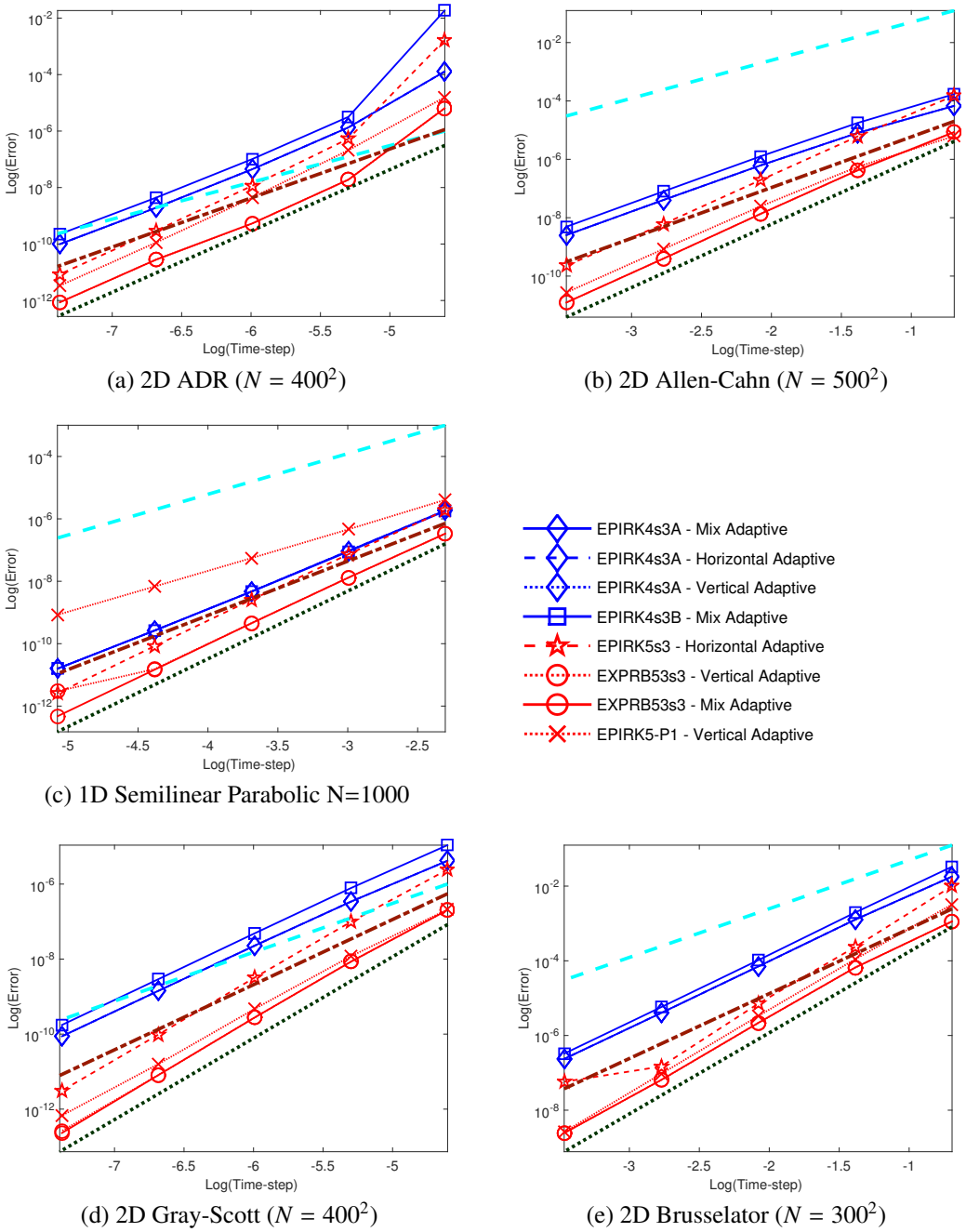
Figure 4.2 shows the achieved order of all the methods we are considering. For convenience we included lines (without markers) of slope three (dashed), four (dash-dotted), and five (dotted). It can be seen from Figure 4.2 that the predicted order of each stiffly accurate method is attained whereas the classically derived EPIRK5P1 achieves fifth-order for all but one exception, semilinear parabolic problem (Figure 4.2(c)) where a significant order reduction is observed. Note that there is not guarantee of order reduction if a method is derived classically and does not satisfy the stiff order conditions. Whether order reduction is observed depends on the specifics of the differential operator of a given problem. As Figure 4.2 illustrates it is possible that a classical method does not exhibit order reduction even if a problem is stiff. Figure 4.2 also confirms that the same numerical solution is obtained for each version of the exponential-Krylov implementation.

Comparative performance

The results of our numerical experiments are presented and analyzed to address the comparative performance of the different exponential-Krylov implementations and the new stiffly accurate EPIRK schemes themselves. Our comparisons are based on the analysis of precision diagrams for the following simulations:

- ADR: $N = 400^2$ with $h = 0.01, 0.005, 0.0025, 0.00125, 6.25e-4$,
- Allen-Cahn: $N = 500^2$ with $h = 0.5, 0.25, 0.1250, 0.0625, 0.03125$,
- SemilinearParabolic: $N = 1000$ with $h = 0.1, 0.05, 0.0250, 0.0125, 0.00625$,
- Gray-Scott: $N = 400^2$ with $h = 0.01, 0.005, 0.0025, 0.00125, 6.25e-4$,
- Brusselator: $N = 300^2$ with $h = 0.5, 0.25, 0.1250, 0.0625, 0.03125$,

where N and h correspond to the spatial discretization and time-step sizes respectively. The precision diagrams are given in Figure 4.3. Previously published performance comparisons such as [38] addressed computational issue characteristic of the EPIRK methods in general such as, for example, the C-shape of the precision graphs which is induced by the computational complexity scaling of the Krylov algorithm with respect to the size of the time step. Here we concentrate on the numerical experiments demonstrating the properties of the stiffly accurate and optimized with respect of a particular implementation EPIRK methods.



(a) 2D ADR ($N = 400^2$)

(b) 2D Allen-Cahn ($N = 500^2$)

(c) 1D Semilinear Parabolic N=1000

(d) 2D Gray-Scott ($N = 400^2$)

(e) 2D Brusselator ($N = 300^2$)

Figure 4.2: Log-log plots of error vs. time step size. For convenience the lines with slopes equal to three (dashed), four (dash-dotted), and five (dotted) are shown.

Overall, the figures verify that the performance of each method highly depends on the number of Krylov evaluations and the size of the interval $[0, g]$ (which depends on the chosen g-coefficients) that the adaptive Krylov method has to traverse. For example, consider the horizontal (dashed) and vertical (dotted) implementation of the fourth-order *EPIRK4s3A* (diamond). The same number of adaptive-Krylov evaluations per time-step were taken (three) but Figure 4.3 shows a considerable difference in the overall computational cost. By comparing the CPU times for each time-step we can easily see how much computational savings are obtained by using the horizontal adaptive-Krylov algorithm. Table 4.5(a) displays the maximum, minimum, and average of the cost of *EPIRK4s3A*-Vert compared to cost of *EPIRK4s3A*-Horz over all time-steps. Considering all the test problems, the vertical implementation of *EPIRK4s3A* costs on average 129% of the cost of the horizontal implementation. Similar results are also found when comparing the fifth-order *EXPRB53s3*-Vert with the specifically constructed *EPIRK5s3*-Horz (Table 4.6 (b)). As we predicted, these savings come from the horizontal implementations ability to make use of $g_{ij} < 1$ coefficients by reducing the Krylov basis size. While this advantage should be observed for the vertical implementation of any closely related method of the same order and same number of stages, the amount of savings will depend on the coefficients of the method.

Table 4.5: Comparison of CPU times for *EPIRK4s3A*-Vert with *EPIRK4s3A*-Horz and *EXPRB53s3*-Vert with *EPIRK5s3*-Horz (Cost of VertMethod/Cost of HorzMethod)

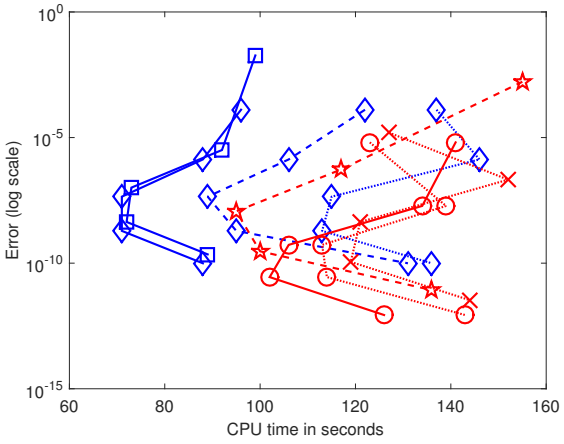
	Max	Min	Avg
ADR:	137 %	114 %	120 %
Allen-Cahn:	127 %	104%	120 %
Brusselator:	133%	98 %	120 %
Gray-Scott:	131%	116%	126%
Semilinear Parabolic:	157 %	143 %	152%

(a)CPU times: *EPIRK4s3A*-Vert/ *EPIRK4s3A*-Horz

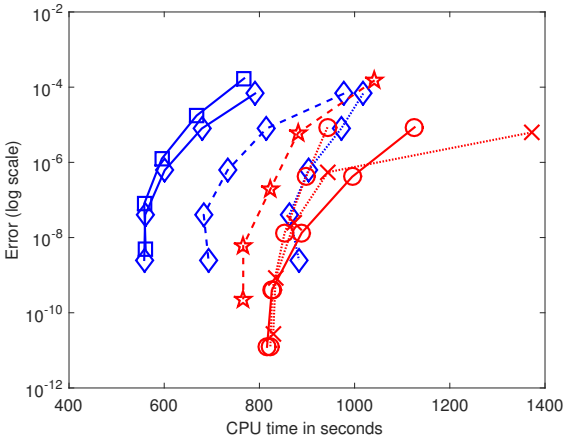
	Max	Min	Avg
ADR:	119%	79%	107%
Allen-Cahn:	108%	90%	102%
Brusselator:	107%	94%	101%
Gray-Scott:	109%	102%	106%
Semilinear Parabolic:	138%	107%	124%

(b) CPU times: *EXPRB53s3*-Vert / *EPIRK5s3*-Horz

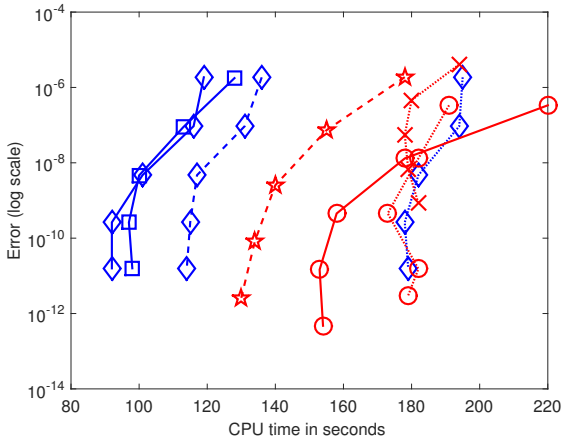
The vertical and horizontal implementation of *EPIRK4s3A* require three Krylov evaluations each time-step. The mixed implementation of *EPIRK4s3A* only requires two Krylov projections and therefore it is expected this method will further increase the savings compared to the vertically implemented *EPIRK4s3A*. Our numerical experiments confirm *EPIRK4s3A*-Mix has a clear advantage over both its horziontal and vertical implementations and can offer up to 50% savings (compared to its vertical implementation). The maximum/minimum/average of the per time-step comparisons are given in Table 4.6 and plots of CPU execution time versus error in Figure 4.3. In the case where the number of Krylov evaluations are the same (i.e. fifth-order three-stage methods), the mixed implementation can still offer computational savings over the vertical implementation but is really dependent on the g-coefficients. For example, the difference in performance between the mixed and vertical implementations of *E*XPRB53s3 is much smaller due to $g_{31} = 9/10$. For this value the resulting intervals $[0, g]$ are nearly the same and no significant savings are obtained. We will pursue further optimization of coefficients strategies for horizontal and mixed implementations in our future research.



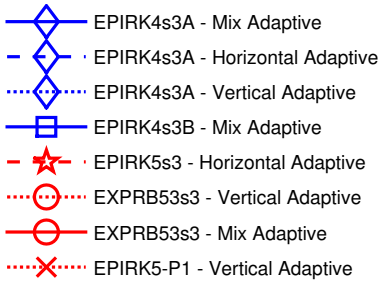
(a) 2D ADR ($N = 400^2$)

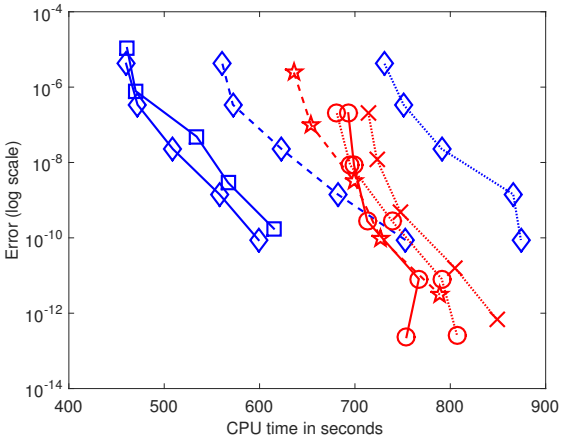


(b) 2D Allen-Cahn ($N = 500^2$)

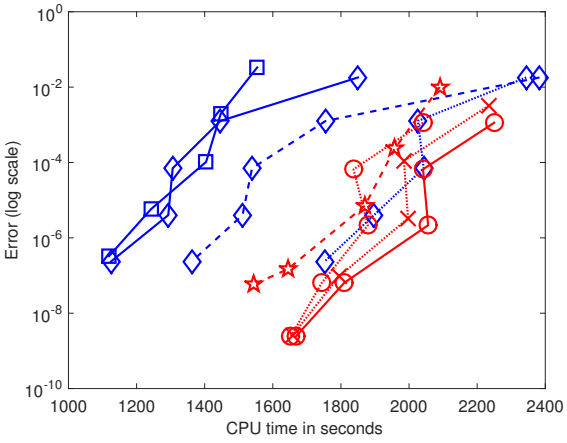


(c) 1D Semilinear Parabolic N=1000





(d) 2D Gray-Scott ($N = 400^2$)



(e) 2D Brusselator ($N = 300^2$)

Figure 4.3: CPU execution time versus error for constant time step experiments

Table 4.6: Cost of horizontal and vertical implementations in comparison to mixed implementation

		Comparisons to EPIRK4s3A-Mixed				Comparisons to EXPRB5s3-Mixed		
		Max	Min	Avg		Max	Min	Avg
ADR:	EPIRK4s3A-Horz	149%	120%	131%	EPIRK5s3-Horz	110%	87%	99%
	EPIRK4s3A-Vert	166%	143%	157%	EXPRB53s3-Vert	113%	87%	105%
Allen-Cahn:	EPIRK4s3A-Horz	124%	120%	122%	EPIRK5s3-Horz	94%	89%	92%
	EPIRK4s3A-Vert	158%	129%	147%	EXPRB53s3-Vert	101%	84%	94%
Brusselator:	EPIRK4s3A-Horz	129%	117%	121%	EPIRK5s3-Horz	96%	91%	93%
	EPIRK4s3A-Vert	156%	127%	145%	EXPRB53s3-Vert	99%	90%	94%
Gray-Scott:	EPIRK4s3A-Horz	126%	121%	123%	EPIRK5s3-Horz	105%	92%	97%
	EPIRK4s3A-Vert	159%	146%	155%	EXPRB53s3-Vert	107%	98%	102%
Semilinear Parabolic:	EPIRK4s3A-Horz	125%	113%	118%	EPIRK5s3-Horz	89%	81%	86%
	EPIRK4s3A-Vert	195%	164%	180%	EXPRB53s3-Vert	119%	87%	107%

We now turn to comparing the performance of the schemes themselves. While there is no clear dominate fifth-order method in regards to computational cost we do see that *EXPRB53s3* is slightly more accurate for all problems. The more interesting comparison is that of the fourth-order method with the fifth-order schemes. For a prescribed accuracy, the fourth-order mixed (and horizontal) *EPIRK4s3A* can offer significant (up to 64%) savings in comparison to the fifth order methods (of any implementation). In Table 4.7 we list the CPU execution times for each method and each test problem for various tolerances. For any set tolerance we see that the mixed implementation of *EPIRK4s3A* can achieve this level of accuracy at a fraction of the cost of any of the fifth-order methods. A simple justification is that conditions for a stiffly accurate fifth-order method are far more restrictive than for a fourth-order scheme. Thus the additional flexibility of stiffly accurate fourth-order schemes allows for more customization and design of methods which optimize the efficiency.

Non-homogeneous boundary conditions

As mentioned in Section 4.2 problems with non-homogeneous boundary conditions do not necessarily satisfy the assumptions of our framework and therefore the stiff order is not guaranteed. The purpose of this section is to show that order reduction occurs and identify how much of a reduction to expect for these problems. We perform simulations with the following test problems:

- Allen-Cahn 2d: Neumann boundary conditions with initial and boundary values given by

$$u = 0.4 + 0.1(x + y) + 0.1 \sin\left(\frac{3}{2}\pi x\right) \sin\left(\frac{5}{2}\pi y\right).$$

- Brusselator 2d: Dirichlet boundary conditions with initial and boundary values given by

$$u = 1 + \sin(2\pi x)\sin(2\pi y)$$
$$v = 3$$

- 1D Degenerate nonlinear diffusion [62]:

$$\frac{\partial u}{\partial t} = \frac{\partial}{\partial x}\left[u\frac{\partial u}{\partial x}\right] + u(1-u), \qquad x \in (-23, 50),\ t \in [0, 50]$$

with Dirichlet boundary conditions $u(-23, t) = 1$ and $u(50, t) = 0$, and initial conditions

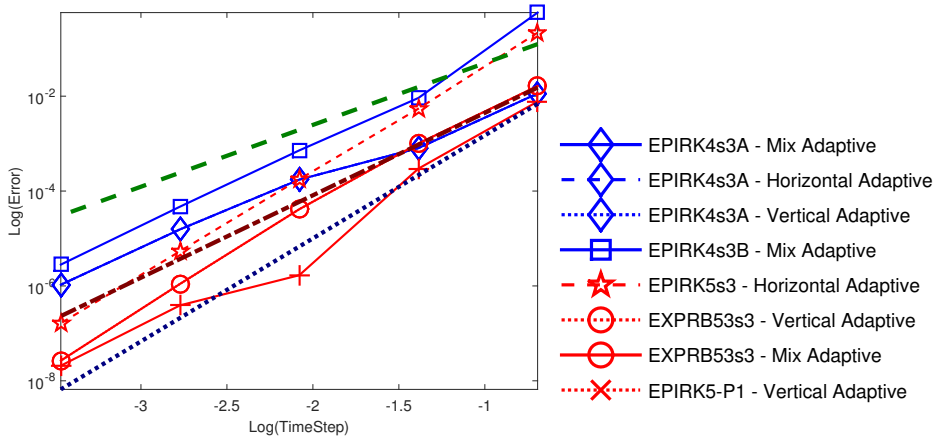
$$u(x, 0) = \begin{cases} 1 & \text{if } x < 0 \\ e^{-1.3x} & \text{if } x > 0. \end{cases}$$

The same spatial discretization and time-step sizes were used for the Brusselator problem as in the previous section. The Allen-Cahn and degenerate nonlinear diffusion problem were conducted with time-step sizes $h = 0.05, 0.0250, 0.0125, 0.00625, 0.003125$ and respective discretization sizes of $N = 500^2$ and $N = 1000$.

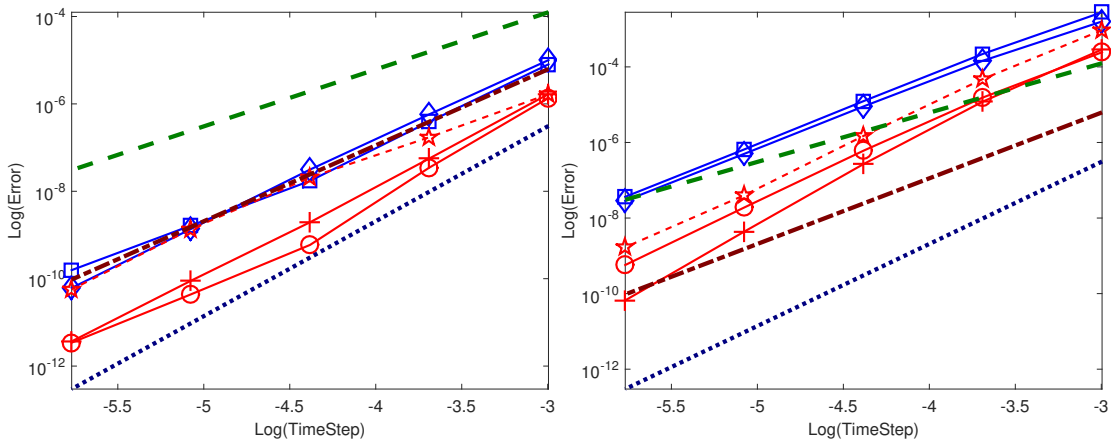
Figure 4.4 displays the log-log plots of time-step size versus error and Table 4.8 has the approximate order exhibited by each of the method for every test problem. While some of the methods achieve full order for some problems, generally the results illustrate that a reduction of order is possible even if the method is stiffly accurate. The extent of the order reduction ranges from 0.03 to 1.34. Such reduction is expected since a similar phenomenon occurs for implicit methods. A theory presented in [53, 52] allows to quantify the extent of order reduction for Rosenbrock methods. We plan to pursue development of a similar theory for exponential integrators applied to nonhomogeneous problems in our future research.

Variable time-step comparisons

We present here the results of our variable time-step experiments on tests problems described in Section 4.4. In addition to the stiffly accurate schemes from Section 4.3 we will also consider the fifth-order classical (non-stiff) *EPIRK5-P1* method with a fourth-order error estimator [39]. We used the same configuration for our experiments as in [38]. For each problem, five runs were made with the following absolute and relative tolerances $Atol = Rtol = 10^{-2}, 10^{-3}, 10^{-4}, 10^{-5}, 10^{-6}$. The resulting diagrams of CPU execution time versus error are displayed in Figure 4.5. The classically derived *EPIRK5-P1* shows to be the most efficient method across all the problems but has the potential drawback of suffering from a reduction of order as seen with the semilinear parabolic problem. This further confirms the need for more efficient stiffly accurate methods as well as illustrates the need for a more refined theory that predicts how much order reduction can be expected for a given problem and a chosen integrator.



(a) 2D Brusselator ($N = 300^2$)



(c) 1D Degenerate Nonlinear Diffusion ($N = 1000$)

(d) 2D Allen-Cahn ($N = 500^2$)

Figure 4.4: Log-log plots of error vs. time step size for problems with non-homogeneous boundary conditions. For convenience the lines with slopes equal to three (dashed), four (dash-dotted), and five (dotted) are shown.

Table 4.7: Approximate CPU times for a given accuracy

Accuracy	Approx. CPU times (seconds)			
	EPIRK4s3A-mixed	*EPIRK5s3*-horz	*EXPRB53s3*-vert	*EXPRB53s3*-mixed
10^{-6}	616	851	911	1034
10^{-7}	575	811	879	950
10^{-8}	558	775	849	885

(a) Allen-Cahn $N = 500^2$

Accuracy	Approx. CPU times (seconds)			
	EPIRK4s3A-mixed	*EPIRK5s3*-horz	*EXPRB53s3*-vert	*EXPRB53s3*-mixed
10^{-7}	74	107	134	136
10^{-8}	71	95	134	128
10^{-9}	75	98	117	110

(b) ADR $N = 400^2$

Accuracy	Approx. CPU times (seconds)			
	EPIRK4s3A-mixed	*EPIRK5s3*-horz	*EXPRB53s3*-vert	*EXPRB53s3*-mixed
10^{-7}	116	156	187	204
10^{-8}	104	145	181	176
10^{-9}	96	138	175	162

(c) Semilinear Parabolic $N = 1000$

Accuracy	Approx. CPU times (seconds)			
	EPIRK4s3A-mixed	*EPIRK5s3*-horz	*EXPRB53s3*-vert	*EXPRB53s3*-mixed
10^{-7}	488	655	683	695
10^{-8}	523	684	694	698
10^{-9}	562	708	722	708

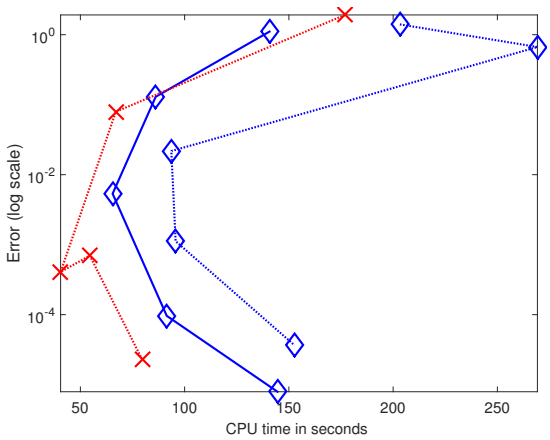
(d) Gray-Scott $N = 400^2$

Accuracy	Approx. CPU times (seconds)			
	EPIRK4s3A-mixed	*EPIRK5s3*-horz	*EXPRB53s3*-vert	*EXPRB53s3*-mixed
10^{-4}	1321	1937	1861	2076
10^{-5}	1295	1877	1859	2050
10^{-6}	1204	1754	1847	1999

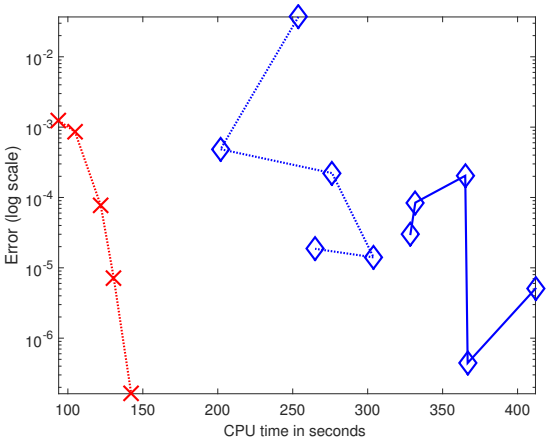
(d) Brusselator $N = 300^2$

Table 4.8: Approximate order of methods for problems with non-homogeneous boundary conditions

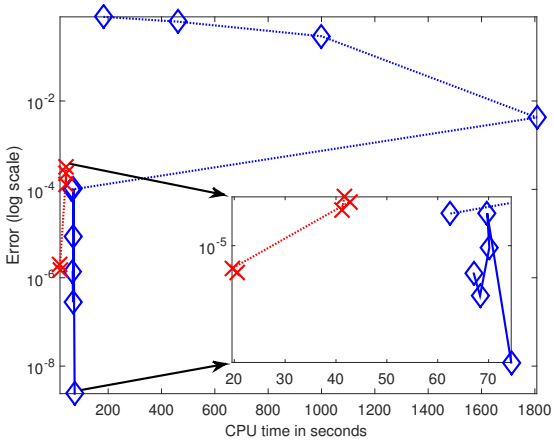
	EPIRK4s3A	*EPIRK4s3B*	*EPIRK5s3*	*EXPRB53s3*	*EPIRK5-P1*
Allen-Cahn:	3.97	4.08	4.84	4.71	5.56
Brusselator:	3.25	4.29	5.06	4.83	4.66
Deg NL Diff:	4.34	3.92	3.66	4.68	4.69



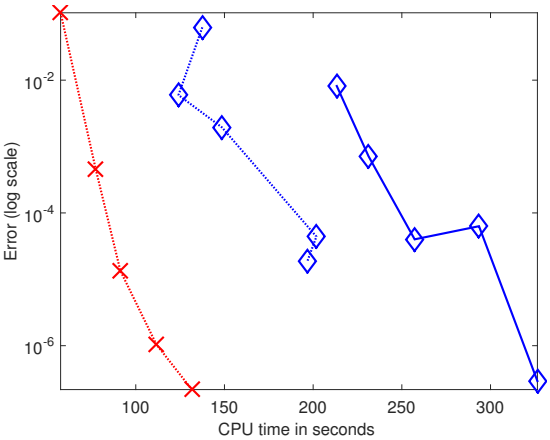
(a) 2D ADR ($N = 400^2$)



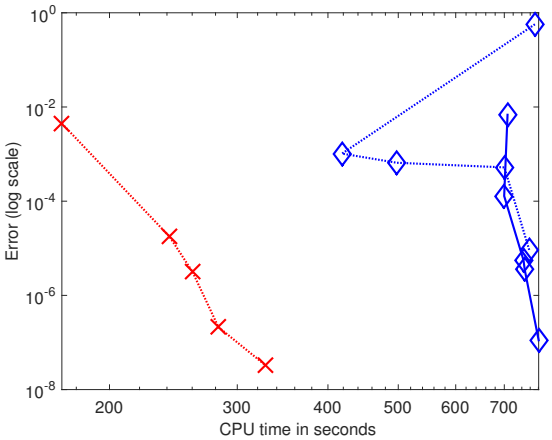
(b) 2D Allen-Cahn ($N = 500^2$)



(c) 1D Semilinear Parabolic N=1000



(c) 2D Gray-Scott ($N = 400^2$)



(d) 2D Brusselator ($N = 300^2$)

Figure 4.5: CPU execution time versus error for variable time step experiments

Chapter 5

Conclusion

In this thesis we extended previous work on exponential integrators in a number of new directions. As the complexity and scale of scientific and engineering problems increases, there is a pressing demand for more efficient numerical integrators. This translates into the need for better understanding of how to exploit the structure of a problem to gain efficiency and requires analytical results to guarantee convergence of the new integrators. In our work we made advances in both of these areas of research. We introduced new classes of methods and designed new algorithms for constructing efficient schemes that can take advantage known properties of a given problem to increase efficiency. We also demonstrated computational advantages of these new methods. In addition we extended convergence theory of major classes of exponential integrators for stiff problems. We made advances in the following particular areas.

We have introduced a new class of exponential time integrators for the split problems of type (2.11), the sEPIRK methods. We extended the standard B-series to account for bicolored trees required to derive the order conditions for the sEPIRK schemes. An algorithm was developed to solve the order conditions for the sEPIRK methods up to order three and several schemes were constructed with coefficients optimized for efficiency. It was also shown that sEPIRK methods can only achieve the order equal to the number of stages. The numerical results presented in this thesis illustrate the performance of the new methods and demonstrate how the flexibility in the order conditions can be used to derive schemes with improved computational efficiency for a given problem. For example, it was shown that an optimized third-order method can be more efficient then a higher order scheme for some problems. In addition, we demonstrated that the comparative efficiency of a split versus a non-split EPIRK method is highly problem dependent. This work opens several new directions of research including further study of "hybrid"-type EPIRK methods and derivation of stiffly accurate sEPIRK schemes. We are currently preparing a public release of an exponential integrators software package. The structure and performance of this software will be addressed in our next publication.

A new class of methods we developed helps overcome an important disadvantage of exponential integrators. For many problems, a portion of the forcing operator can be efficiently preconditioned in the context of implicit methods. A number of efficient preconditioning techniques have been developed over the past years. Exponential integrators however are not able to take advantage

of such preconditioners. We developed a new class of implicit-exponential (IMEXP) methods that over come this limitation. Such methods are appropriate for problems where the forcing operator can be partitioned into a term L for which an efficient preconditioner is available and nonlinear stiff term N which cannot be easily preconditioned. Typically such problems are addressed by implicit-explicit (IMEX) methods which advance L implicitly and N explicitly. However, if N is stiff the explicit treatment results in stability restriction on the time step size. Our new IMEXP methods still allow preconditioned implicit treatment of L while treating N exponentially. Thus IMEXP methods can take advantage of preconditioning while avoiding the stability limitations on the time step imposed by the stiff nonlinear term N if it is treated explicitly. Such methods can be beneficial for arising in many fields such as reaction-diffusion equations in combustion, pattern formation problems or biochemical models.

Finally, we have extended the stiff order conditions and convergence theory in [46] for EXPRB methods to EPIRK-type methods. We offered a different approach to solving the stiff order conditions that allows construction of efficient schemes of several types particularly when these methods are used in conjunction with the adaptive Krylov algorithm. Using the generality of the EPIRK framework we constructed new stiffly accurate fourth and fifth-order schemes and numerically confirmed they achieved their full predicted order of accuracy on a set of test problems. Our numerical experiments further showed that the new technique of deriving horizontal or mixed EPIRK schemes does offer improved computational savings compared to previously derived (EPIRK & EXPRB) methods. For a given exponential method, however, the most efficient implementation will depend on its coefficients and the structure of the problem under consideration. We are currently working on a modified adaptive Krylov algorithm that provides more computational savings for horizontal and mixed optimized EPIRK schemes. Development of better guidelines in constructing/choosing the most efficient integrator for a given problem is a goal of our future investigations. We also plan to extend/develop the stiff order conditions theory for partitioned (or split) EPIRK, implicit-exponential type-methods. Finally, more research is needed to investigate whether stiffly accurate exponential integrators that do not suffer from order reduction can be developed for problems with non-homogeneous boundary conditions.

Appendix A

Appendix

In this appendix we present the details of the necessary modifications to the theory in [46] to prove convergence of the stiffly accurate EPIRK methods. The convergence proof in [46] proceeds by expressing the global error $e_{n+1} = u_{n+1} - u(t_{n+1}) = u_{n+1} - \tilde{u}_{n+1}$ in the following form

$$e_{n+1} = e^{h_n \tilde{J}_n} e_n + h_n P_n + \tilde{e}_{n+1}, \quad e_0 = 0, \tag{A.1}$$

where

$$P_n = q_n + \sum_{i=2}^{s} Q_{ni} \tag{A.2}$$

with

$$q_n = \varphi_1(h_n J_n)(N_n(u_n) - N_n(\tilde{u}_n)) + (\varphi_1(h_n J_n) - \varphi_1(h_n \tilde{J}_n)) f(\tilde{u}_n), \tag{A.3}$$

$$Q_{ni} = (b_i(h_n J_n) - b_i(h_n \tilde{J}_n))\widehat{r}_{ni} + b_i(h_n J_n)(r(U_{ni}) - \widehat{r}_{ni}). \tag{A.4}$$

Lemmas 4.1 through 4.5 in [46] provide bounds to the different terms in this expression. All of these lemmas are directly applicable to the EPIRK methods except for Lemma 4.5. Here we present a modified proof of Lemma 4.5 that accounts for the fact that EPIRK methods employ the general ψ-function rather than the φ_1-function as in the exponential Rosenbrock methods. To motivate the lemma we begin by applying Lemma 4.4 in [46] to (A.2) and obtain the preliminary estimate

$$\| P_n \| \leq C h_n \| e_n \| + C \| e_n \|^2 + \sum_{i=2}^{s} C \| e_n \| + C \| e_n \|^2 + C \left(h_n + \| e_n \| + \left\| \widehat{E}_{ni} \right\| \right) \left\| \widehat{E}_{ni} \right\|, \tag{A.5}$$

where $\widehat{E}_{ni} = U_{ni} - \widehat{U}_{ni}$ is the difference between the numerical solutions obtained from (4.4) and (4.17). Our desired estimate for P_n is obtained by bounding $\left\| \widehat{E}_{ni} \right\|$ in terms of $\| e_n \|$. The bound found in [46] for EXPRB methods only holds for methods whose internal stages strictly use φ_1-function in the first term. With the additional assumption that the method satisfies Assumption 3, we prove the same bound holds for any linear combination of φ-functions.

Lemma 9. *Under Assumptions 1-3, for all i, we have*

$$\left\| \widehat{E}_{ni} \right\| \leq C \| e_n \| + Ch_n \| e_n \|^2 + Ch_n^5 \tag{A.6}$$

$$\| P_n \| \leq C \| e_n \| + C \| e_n \|^2 + Ch_n^6 \tag{A.7}$$

as long as the global errors e_n remain in a sufficiently small neighborhood of 0 *and $h_n \leq C_H$.*

Proof. Without loss of generality and for sake of presentation, we will assume the method satisfies $p_{i1k} = g_{i1}$ of Assumption 3 for each i and all k. We begin by proving the estimate

$$\left\| \widehat{E}_{ni} \right\| \leq C \| e_n \| + Ch_n \| e_n \|^2 + h_n \sum_{j=2}^{i-1} C(h_n + \| e_n \| + \left\| \widehat{E}_{nj} \right\|) \left\| \widehat{E}_{nj} \right\|. \tag{A.8}$$

By adding and subtracting $\alpha_{i1} h_n p_{i1k} \varphi_k(g_{i1} h_n J_n) f(\tilde{u}_n)$ for each $k = 1, \ldots, s$ to $\widehat{E}_{ni}$ we can then write $\widehat{E}_{ni}$ as

$$\begin{aligned} \widehat{E}_{ni} = e_n &+ \alpha_{i1} h_n \sum_{k=1}^{s} p_{i1k} \varphi_k(g_{i1} h_n J_n) \left(f(u_n) - f(\tilde{u}_n) \right) + \\ &+ \alpha_{i1} h_n \sum_{k=1}^{s} p_{i1k} \left(\varphi_k(g_{i1} h_n J_n) - \varphi_k(g_{i1} h_n \tilde{J}_n) \right) f(\tilde{u}_n) + \\ &+ h_n \sum_{j=2}^{i-1} a_{ij}(h_n J_n)(r_{nj} - \hat{r}_{nj}) + h_n \sum_{j=2}^{i-1} (a_{ij}(h_n J_n) - a_{ij}(h_n \tilde{J}_n)) \hat{r}_{nj}. \end{aligned} \tag{A.9}$$

Using the identity $f(u) - f(\tilde{u}_n) = J_n e_n + N_n(u_n) - g_n(\tilde{u}_n)$, (4.33) and recurrence relation (4.25), (A.9) can be expressed as

$$\begin{aligned} \widehat{E}_{ni} &= e_n + \alpha_{i1} \sum_{k=1}^{s} \left[h_n g_{i1} \varphi_k(g_{i1} h_n J_n) J_n e_n + h_n g_{i1} \varphi_k(g_{i1} h_n J_n)(N_n(u_n) - N_n(\tilde{u}_n)) \right] + \\ &\quad + \alpha_{i1} g_{i1} h_n \sum_{k=1}^{s} (\varphi_k(g_{i1} h_n J_n) - \varphi_k(g_{i1} h_n \tilde{J}_n)) f(\tilde{u}_n) + h_n \sum_{j=2}^{i-1} a_{ij}(h_n J_n)(r_{nj} - \hat{r}_{nj}) + \\ &\quad + h_n \sum_{j=2}^{i-1} (a_{ij}(h_n J_n) - a_{ij}(h_n \tilde{J}_n)) \hat{r}_{nj} \\ &= e_n + \alpha_{i1} \sum_{k=1}^{s} \left[(\varphi_{k-1}(g_{i1} h_n J_n) - 1/k!) e_n + \alpha_{i1} g_{i1} h_n \varphi_k(g_{i1} h_n J_n)(N_n(u_n) - N_n(\tilde{u}_n)) \right] + \\ &\quad + \alpha_{i1} g_{i1} h_n \sum_{k=1}^{s} (\varphi_k(g_{i1} h_n J_n) - \varphi_k(g_{i1} h_n \tilde{J}_n)) f(\tilde{u}_n) + h_n \sum_{j=2}^{i-1} a_{ij}(h_n J_n)(r_{nj} - \hat{r}_{nj}) + \\ &\quad + h_n \sum_{j=2}^{i-1} a_{ij}(h_n J_n) - a_{ij}(h_n \tilde{J}_n)) \hat{r}_{nj} \end{aligned} \tag{A.10}$$

The estimate (A.8) then follows from the positive-scalability and sub-additivity of the norm, the estimates of Lemmas 4.1 & 4.3 in [46], boundedness of $f(\tilde{u}_n) = \tilde{u}'_n$ and $\varphi_k(h_nJ)$ (and $a_{ij}(h_nJ_n)$). Now we can prove (A.6). Since e_n is assumed to remain in a sufficiently small neighborhood of 0, there exists $0 < \delta < 1$ such that $\| e_n \| < \delta$ for all n. This implies that for each n, $\| e_n \|^2 \leq \| e_n \|$ and furthermore shows that

$$\left\| \widehat{E}_{n2} \right\| \leq C \| e_n \| + Ch_n \| e_n \|^2 \leq C \| e_n \| + Ch_n \| e_n \| \tag{A.11}$$

by using (A.8) with $i = 2$. Assuming $\left\| \widehat{E}_{ni-1} \right\| \leq C_1 \| e_n \| + C_2 h_n \| e_n \|^2$ we obtain

$$\begin{aligned} \left\| \widehat{E}_{ni} \right\| &\leq C \| e_n \| + Ch_n \| e_n \|^2 + h_n \sum_{j=2}^{i-1} C(h_n + \| e_n \| + \left\| \hat{E}_{nj} \right\|) \left\| \hat{E}_{nj} \right\| \qquad (A.12) \\ &\leq C \| e_n \| + Ch_n \| e_n \|^2 + h_n \sum_{j=2}^{i-1} C(h_n + \| e_n \| + \\ &\qquad +(C_1 \| e_n \| + C_2 h_n \| e_n \|^2))(C_1 \| e_n \| + C_2 h_n \| e_n \|^2). \end{aligned}$$

By expanding the terms and using the assumption that $\| e_n \| < \delta$ we arrive at

$$\begin{aligned} \left\| \widehat{E}_{ni} \right\| &\leq (C + h_n^2 C_1) \| e_n \| + \\ &\qquad +(C + C_2 h_n^2 + C_1 + C_1^2 + C_2 h_n + 2C_2 C_1 h_n^2 + C_2^2 h_n^2) h_n \| e_n \|^2 \qquad (A.13) \\ &\leq (C + C_H^2 C_1) \| e_n \| + \\ &\qquad +(C + C_2 C_M^2 + C_1 + C_1^2 + C_2 C_M + 2C_2 C_1 C_M^2 + C_2^2 C_M^2) h_n \| e_n \|^2 \qquad (A.14) \\ &\leq \mathbf{C} \| e_n \| + \mathbf{C} h_n \| e_n \|^2 \qquad (A.15) \end{aligned}$$

where $C_M = \max(C_H, 1)$ and $\mathbf{C} = \max((C + C_H^2 C_1), (C + C_2 C_M^2 + C_1 + C_1^2 + C_2 C_M + 2C_2 C_1 C_M^2 + C_2^2 C_M^2), 1)$. The estimate (A.7) now follows from (A.5) and (A.6). □

Bibliography

[1] W.E. Arnoldi. “The principle of minimized iteration in the solution of the matrix eigenvalue problem”. In: *Quart. Appl. Math.* 9 (1951), pp. 17–29.

[2] U. M. Ascher, S. J. Ruuth, and R.J. Spiteri. “Implicit-explicit Runge–Kutta methods for time-dependent partial differential equations”. In: *Appl. Numer. Math.* 25 (1997), pp. 151–167.

[3] U.M. Ascher, S.J. Ruuth, and B.T.R. Wetton. “Implicit-explicit methods for time-dependent PDEs”. In: *SIAM J. Numer. Anal.* 32.3 (1997), pp. 797–823.

[4] H. Berland, B. Owren, and B. Skaflestad. “B-series and order conditions for exponential integrators”. In: *SIAM J. Numer. Anal.* 43 (4 2005), pp. 1715–1727.

[5] F. Bisetti. “Integration of large chemical kinetic mechanisms via exponential methods with Krylov approximations to Jacobian matrix functions”. In: *Combust. Theor. Model.* 16.3 (2012), pp. 387–418.

[6] J. C. Butcher. *Numerical Methods for Ordinary Differential Equations*. 2nd ed. Wiley, 2008.

[7] J. C. Butcher. “Trees, B-series and exponential integrators”. In: *IMA J.Numer. Anal.* 30 (2009), pp. 131–140.

[8] M. Caliari and A. Ostermann. “Implementation of exponential Rosenbrock-type integrators”. In: *Appl. Numer. Math.* 59 (2009), pp. 568–581.

[9] C. Canuto et al. *Spectral methods in fluid dynamics*. New York: Springer, 1987.

[10] J. Certaine. *The solution of ordinary differential equations with large time constants*. Wiley, 1967.

[11] E.M. Constantinescu and A. Sandu. “Extrapolated implicit-explicit time stepping”. In: *SIAM J. Sci. Comput.* 31.6 (2010), pp. 4452–4477.

[12] S.M. Cox and P.C. Matthews. “Exponential time differencing for stiff systems”. In: *J. Comput. Phys.* 176 (2002), pp. 430–455.

[13] K.J. Engel and R. Nagel. *One-parameter semigroups for linear evolution equations*. New York: Springer, 2000.

[14] R. A. Friesner et al. “A method for exponential propagation of large systems of stiff nonlinear differential equations”. In: *J. Sci. Comput.* 4 (1989), pp. 327–354.

[15] J.M. Keiser G. Beylkin and L. Vozovoi. "A new class of time discretization schemes for the solution of nonlinear PDEs". In: *J. Comput. Phys.* 147 (1998), pp. 362–387.

[16] E. Gallopoulos and Y. Saad. "Efficient solution of parabolic equations by Krylov approximation methods". In: *SIAM J. Sci. Stat. Comp* 13 (1992), pp. 1236–126429.

[17] H. Goedbloed and S. Poedts. *Principles of Magnetohydrodynamics with Applications to Laboratory and Astrophysical Plasmas.* Cambridge University Press, 2004.

[18] P. Gray and S.K. Scott. "Autocatalytic reaction in the isothermal continuous stirred tank reactor: oscillations and instabilities in the system $A + 2B \rightarrow 3B$, $B \rightarrow C$". In: *Chem. Eng. Sci.* 39 (1984), pp. 1087–1097.

[19] E. Hairer and G. Wanner. *Solving Ordinary Differential Equations II: Stiff and Differential-Algebraic Problems*. Springer, New York, 1996.

[20] E. Hairer and G. Wanner. *Solving Ordinary Differential Equations II: Stiff and Differential Algebraic Problems*. 2nd. Springer, 2004.

[21] D. Henry. *Geometric theory of semilinear parabolic equations*. Vol. 840. Lecture Notes in Mathematics. Springer Berlin Heidelberg, 1981.

[22] N. Higham. "The scaling and squaring method for matrix exponential revisited". In: *SIAM J. Matrix Anal. Appl.* 26 (2005), pp. 1179–1193.

[23] M. Hochbruck and Ch. Lubich. "On Krylov subspace approximations to the matrix exponential operator". In: *SIAM J. Numer. Anal.* 34 (1997), pp. 1911–1925.

[24] M. Hochbruck, Ch. Lubich, and H. Selhofer. "Exponential integrators for large systems of differential equations". In: *SIAM J. Sci. Comput.* 19 (1998), pp. 1552–1574.

[25] M. Hochbruck and A. Ostermann. "Explicit Exponential Runge-Kutta Methods for Semilinear Parabolic Problems". In: *SIAM J. Numer. Anal.* 43.3 (2005), pp. 1069–1090.

[26] M. Hochbruck and A. Ostermann. "Exponential Runge-Kutta methods for parabolic problems". In: *Appl. Numer. Math.* 53 (2005), pp. 323–339.

[27] M. Hochbruck and A. Ostermann. "Exponential integrators of Rosenbrock-type". In: *Oberwolfach Reports* 3 (2006), pp. 1107–1110.

[28] M. Hochbruck and A. Ostermann. "Exponential integrators". In: *Acta Numer.* (2010), pp. 209–286.

[29] M. Hochbruck, A. Ostermann, and J. Schweitzer. "Exponential Rosenbrock-type methods". In: *SIAM J. Numer. Anal.* 47 (2009), pp. 786–803.

[30] M. Hochbruck, A. Ostermann, and J. Schweitzer. "Exponential Rosenbrock-type methods". In: *SIAM J. Numer. Anal.* 47 (2009), pp. 786–803.

[31] B. Bozzini D. Lacitignola I. Sgura. "Numerical approximation of Turing patterns in electrodeposition by ADI methods". In: *J. Comp. Appl. Math.* 236 (16 2012), pp. 4132–4147.

[32] W. Hundsorfer J.G. Verwer J. Frank. "On the stability of implicit-explicit linear multistep methods". In: *Applied Numer. Math.* 25 (1997), pp. 193–205.

[33] A.K. Kassam and L.N. Trefethen. "Fourth-order time stepping for stiff PDEs." In: *SIAM J. Sci. Comput.* 26.4 (2005), pp. 1214–1233.

[34] J. Kim and P. Moin. "Application of a fractional-step method to incompressible Navier-Stokes equations". In: *J. Comput. Phys.* 59 (1985), pp. 308–323.

[35] S. Krogstad. "Generalized integrating factor methods for stiff PDEs". In: *J. Comp. Phys.* 203 (2005), pp. 72–88.

[36] J.D. Lawson. "Generalized Runge-Kutta processes for stable systems with large Lipschitz constants". In: *SIAM J. Numer. Anal.* 4 (1967), pp. 372–380.

[37] R. Lefever and G. Nicolis. "Chemical instabilities and sustained oscillations". In: *J. Theor. Biol.* 3 (1971), pp. 267–284.

[38] J. Loffeld and M. Tokman. "Comparative performance of exponential, implicit, and explicit integrators for stiff systems of ODEs". In: *Journal of Computational and Applied Mathematics* 214 (2013), pp. 45–67.

[39] J. Loffeld and M. Tokman. "Comparative performance of exponential, implicit, and explicit integrators for stiff systems of ODEs". In: *Journal of Computational and Applied Mathematics* 214 (2013), pp. 45–67.

[40] J. Loffeld and M. Tokman. "Implementation of Parallel Adaptive-Krylov exponential solvers for stiff problems". In: *SIAM J. Sci. Comput.* 36.5 (2014), pp. C591–C616.

[41] John Loffeld and Mayya Tokman. "Comparative performance of exponential, implicit, and explicit integrators for stiff systems of ODEs". In: *Journal of Computational and Applied Mathematics* 241 (2013), pp. 45–67.

[42] V. T. Luan and A. Ostermann. "Explicit exponential Runge–Kutta methods of high order for parabolic problems". In: *J. Comput. Appl. Math.* 256 (2014), pp. 168–179.

[43] V. T. Luan and A. Ostermann. "Exponential B-series: The stiff case". In: *SIAM J. Numer. Anal.* 51 (2013), pp. 3431–3445.

[44] V. T. Luan and A. Ostermann. "Parallel exponential Rosenbrock methods". In: *Comput. Math. Appl.* 71 (2016), pp. 1137–1150.

[45] V.T. Luan and A. Ostermann. "Exponential B-Series: The Stiff Case". In: *SIAM J. Numer. Anal.* 51 (6 (2013)), pp. 3431–3445.

[46] V.T. Luan and A. Ostermann. "Exponential Rosenbrock methods of order five — construction, analysis and numerical comparisons". In: *Journal of Computational and Applied Mathematics* (2014), pp. 417–431. ISSN: 0377-0427.

[47] V.T. Luan and A. Ostermann. "Stiff Order Conditions for Exponential Runge–Kutta Methods of Order Five". In: *Modeling, Simulation and Optimization of Complex Processes - HPSC 2012*. Ed. by Hans Georg Bock et al. Springer International Publishing, 2014, pp. 133–143. ISBN: 978-3-319-09062-7.

[48] A. Madzvamuse and P.K. Main. "Velocity-induced numerical solutions of reaction-diffusion systems on continuously growing domains". In: *J. Comput. Phys.* 225 (2007), pp. 100–119.

[49] A. Nauts and R. E. Wyatt. "New approach to many-state quantum dynamics: the recursive-residue-generation method". In: *Phys. Rev. Lett.* 51 (1983), pp. 2238–2241.

[50] Jitse Niesen and Will M. Wright. "Algorithm 919: A Krylov Subspace Algorithm for Evaluating the ϕ-Functions Appearing in Exponential Integrators". In: *ACM Trans. Math. Softw.* 38.3 (Apr. 2012), 22:1–22:19. ISSN: 0098-3500. DOI: 10.1145/2168773.2168781. URL: http://doi.acm.org/10.1145/2168773.2168781.

[51] S. P. Nørsett. "An A-stable modification of the Adams-Bashforth methods". In: *Conference on the Numerical Solution of Differential Equations, Lecture Notes in Mathematics*. Vol. 109. Springer, 1969, pp. 214–219.

[52] A. Ostermann and M. Roche. "Rosenbrock methods for partial differential equations and fractional orders of convergence". In: *SIAM J. Numer. Anal.* 30.4 (1993), pp. 1084–1098.

[53] A. Ostermann and M. Roche. "Runge-Kutta methods for partial differential equations and fractional orders of convergence". In: *Math Comp.* 59 (1992), pp. 403–420.

[54] S. Brown J. Han P.W. Bates. "Numerical analysis for a Nonlocal Allen-Cahn equation". In: *Int. J. Numer. Anal. Mod.* 6.1 (2009), pp. 33–49.

[55] T. J. Park and J. C. Light. "Unitary quantum time evolution by iterative Lanczos reduction". In: *J. Chem. Phys.* 85 (1986), pp. 5870–5876.

[56] A. Pazy. *Semigroups of Linear Operators and Applications to Partial Differential Equations*. 1st ed. Springer-Verlag, 1983.

[57] D. A. Pope. "An exponential method of numerical integration of ordinary differential equations". In: *Comm. ACM* 6 (1963), pp. 491–493.

[58] G. Rainwater and M. Tokman. "A new approach to constructing efficient stiffly accurate EPIRK methods". In: *Journal of Computational Physics* 323 (2016), pp. 283–309. ISSN: 0021-9991. DOI: 10.1016/j.jcp.2016.07.026. URL: http://www.sciencedirect.com/science/article/pii/S0021999116303217.

[59] G. Rainwater and M. Tokman. "A new class of split exponential propagation iterative methods of Runge–Kutta type (sEPIRK) for semilinear systems of ODEs". In: *J. Comput. Phys.* 269 (2014), pp. 40–60.

[60] S. Ruuth. "Implicit-explicit methods for reaction-diffusion problems in pattern formation". In: *J. Mathematical Biology* 34 (1995), pp. 148–176.

[61] J. Schnakenberg. "Simple chemical reaction systems with limit cycle behaviour". In: *J. Theoretical Biology* 81 (1979), pp. 389–400.

[62] J.A. Sherratt. "On the form of smooth-front traveling waves in a reaction-diffusion equation with degenerate nonlinear diffusion". In: *Math. Model. Nat. Phenom.* 5 (5 2010), pp. 63–78.

[63] R.B. Sidje. "Expokit: A software package for computing matrix exponentials". In: *ACM Trans. Math. Softw.* 24 (1 1998), pp. 130–156.

[64] M. Tokman. "A new class of exponential propagation iterative methods of Runge–Kutta type (EPIRK)". In: *J. Comput. Phys.* 230 (2011), pp. 8762–8778.

[65] M. Tokman. "A new class of exponential propagation iterative methods of Runge-Kutta type (EPIRK)". In: *Journal of Computational Physics* 230.24 (2011), pp. 8762–8778.

[66] M. Tokman. "Efficient integration of large stiff systems of ODEs with exponential propagation iterative (EPI) methods". In: *J. Comp. Phys.* 213 (2006), pp. 748–776.

[67] M. Tokman. "Four classes of exponential EPIRK integrators". In: *Oberwolfach Reports* 14 (2014), pp. 855–858.

[68] M. Tokman. "Magnetohydrodynamic modeling of solar magnetic arcades using exponential propagation methods". PhD thesis. California Institute of Technology, 2001.

[69] M. Tokman, J. Loffeld, and P. Tranquilli. "New adaptive exponential propagation iterative methods of Runge-Kutta type (EPIRK)". In: *SIAM J. Sci. Comput.* 34.5 (2012), A2650–A2669.

[70] H. A. Van der Vorst. "An iterative solution method for solving $f(A) = b$ using Krylov subspace information obtained for the symmetric positive definite matrix A". In: *J. Comput. Appl. Math.* 18 (1987), pp. 249–263.

[71] A.J. Wathen. "Preconditioning". In: *Acta Numer.* 24 (2015), pp. 329–376.

[72] Liyong Zhu, Lili Ju, and Weidong Zhao. "Fast High-Order Compact Exponential Time Differencing Runge–Kutta Methods for Second-Order Semilinear Parabolic Equations". In: *Journal of Scientific Computing* (2015), pp. 1–23. ISSN: 1573-7691. DOI: 10.1007/s10915-015-0117-1. URL: http://dx.doi.org/10.1007/s10915-015-0117-1.