Lightly-Implicit Methods for the Time Integration of Large Applications

Paul Tranquilli

Dissertation submitted to the Faculty of the
Virginia Polytechnic Institute and State University
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy
in
Computer Science Application

Adrian Sandu, Chair
Mayya Tokman
Yang Cao
Eric de Sturler
Calvin J. Ribbens

June 23, 2016
Blacksburg, Virginia

Keywords: Time Integration, Numerical PDEs, Numerical ODEs

Lightly-Implicit Methods
for the Time Integration of Large Applications

Paul Tranquilli

(ABSTRACT)

Many scientific and engineering applications require the solution of large systems of initial value problems arising from method of lines discretization of partial differential equations. For systems with widely varying time scales, or with complex physical dynamics, implicit time integration schemes are preferred due to their superior stability properties. However, for very large systems accurate solution of the implicit terms can be impractical. For this reason approximations are widely used in the implementation of such methods.

The primary focus of this work is on the development of novel “lightly-implicit” time integration methodologies. These methods consider the time integration and the solution of the implicit terms as a single computational process. We propose several classes of lightly-implicit methods that can be constructed to allow for different, specific approximations.

Rosenbrock-Krylov and exponential-Krylov methods are designed to permit low accuracy Krylov based approximations of the implicit terms, while maintaining full order of convergence. These methods are matrix free, have low memory requirements, and are particularly well suited to parallel architectures. Linear stability analysis of K-methods is leveraged to construct implementation improvements for both Rosenbrock-Krylov and exponential-Krylov methods. Linearly-implicit Runge-Kutta-W methods are designed to permit arbitrary, time dependent, and stage varying approximations of the linear stiff dynamics of the initial value problem. The methods presented here are constructed with approximate matrix factorization in mind, though the framework is flexible and can be extended to many other approximations.

The flexibility of lightly-implicit methods, and their ability to leverage computationally favorable approximations makes them an ideal alternative to standard explicit and implicit schemes for large parallel applications.

Dedication

To my parents and brother whose love and support have made this work possible.

Acknowledgments

I would like to thank my advisor Dr. Adrian Sandu for his guidance, and high expectations. His impact on this work and my career is impossible to overstate. I would also like to extend my deepest gratitude to Dr. Mayya Tokman for her continued advice and support.

In addition I would like to thank the members of my committee for their advice and questions which helped to shape the focus of this work, as well as my labmates Ross Glandon and Mahesh Narayanamurthi for their effort and patience in proofreading and help in editing several of the chapters in this dissertation.

Finally, I would like to thank the faculty of the applied math department at the University of California, Merced in particular Drs. Arnold Kim, Mayya Tokman, Boaz Ilan, Francois Blanchette, and Michael Sprague who fostered and supported my interest in pursuing applied math and academic research.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The numerical solution of initial value problems is a critical component to the modeling and exploration of many systems of scientific and engineering interest. Ordinary differential equations (ODEs) model the evolution of chemical kinetic systems, electronic circuits, and the motion of planets. Other physical systems, such as the dynamics of the atmosphere and oceans, or the behavior of materials, require the solution of systems of partial differential equations (PDEs). A standard approach to solving PDEs known as the method of lines, is to discretize in space, reducing the problem to a large system of semi-discrete ODEs which are then integrated forward in time. ODE systems arising from a method of lines approach can be very large, and their size is only bounded by practical considerations such as available computational resources and the desired grid resolution for the problem under study.

The main contribution of this dissertation is to develop new time integration methodologies that are "lightly-implicit" and can take full advantage of modern computer architectures, while alleviating the limitations imposed by available computational resources. The end goal is to make the solution of larger, more computationally difficult, or previously intractable scientific and engineering problems possible.

1.1 Current Approaches to Solving ODEs

We consider systems of ODEs with general form

$$\frac{dy}{dt} = F(t,y), \quad t_0 \le t \le t_F, \quad y(t_0) = y_0; \quad y(t), F(t,y) \in \mathbb{R}^N. \tag{1.1}$$

Several different approaches are available to numerically approximate the solution of (1.1). These methods can be classified as either explicit, where future solutions are dependent only on information at the current and previous timesteps, or implicit, where future solutions are dependent on both current and future timesteps.

Explicit methods are relatively straightforward to analyze, construct, and implement. They depend exclusively on previously computed information, and so are extremely cost efficient per timestep. Additionally, they make exclusive use scalable operations, generally requiring only evaluation of the function $F(t, y)$ and vector addition, and so are well suited to parallelization. Unfortunately, for stable solutions the size of the timestep is restricted by the Courant-Friedrichs-Lewy (CFL) condition. This restriction can become particularly onerous if the system is stiff. Stiffness can arise out of system dynamics with widely varying natural timescales, or through increased spatial resolution in the semi-discretization of the PDE leading to (1.1). For such systems, the stability of explicit methods may require the use of extremely small timesteps, negating the benefit of their cheap per timestep cost, making them an unattractive choice.

Implicit methods, in contrast, are not constrained by the CFL condition, and so have significantly improved stability properties. The flipside of this improved stability, is their implicit dependence on the future solution leading to the need to solve one or more non-linear equations at each timestep. The cost of solving these non-linear systems can have a dramatic impact on the per timestep efficiency of implicit methods, and for parallel implementations may lead to poor scalability. Here, we arrive at a similar, though opposite, problem as explicit schemes: while relatively few timesteps are required for a stable solution, computing each timestep incurs a significantly greater cost.

A middle ground between explicit and implicit methods, are the so called "multimethods" [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]. These schemes operate on a split formulation of (1.1)

$$\frac{dy}{dt} = f(t, y) + g(t, y), \quad t_0 \leq t \leq t_F, \quad y(t_0) = y_0; \quad y(t), f(t, y), g(t, y) \in \mathbb{R}^N. \tag{1.2}$$

where $f(t, y)$ is nonstiff and $g(t, y)$ is stiff. Multi-methods treat $f(t, y)$ and $g(t, y)$ seperately, in an attempt to overcome the CFL restriction of explicit methods while maintaining a cheaper per timestep cost than implicit schemes.

Multirate methods approach the problem of stability by making use of different size timesteps for the different partitions. So that only the stiff term need be integrated with the most restrictive timestep, while the nonstiff term can make use of a much larger timestep. Alternatively, implicit-explicit (IMEX) methods treat the nonstiff term explicitly, and the stiff term implicitly. In this way they avoid the stability restriction of explicit methods, while avoiding the cost of solving non-linear systems dependent on the full right-hand-side vector $F(t, y)$.

The computational efficiency of both multirate and IMEX schemes depends on constructing favorable partitions $f(t, y)$ and $g(t, y)$. Constructing such partitions in general is difficult, and doing so automatically is an open question. Moreover, legacy code bases may not be amenable to easily splitting $F(t, y)$, so that the partitions $f(t, y)$ and $g(t, y)$ can be evaluated independently. IMEX schemes have the additional issue for legacy codes that a the implicit treatment of $g(t, y)$ requires Jacobian information for $g(t, y)$ only, and this may not be easily obtainable from a Jacobian for $F(t, y)$.

1.2 Objectives and Accomplishments of This Research

The focus of this work is on the development of new time integration schemes, called lightly-implicit methods, which have better stability than explicit schemes, while being considerably cheaper per timestep than implicit methods. Our philosophy is that such methods should be:

1. Built from the ground up to take maximal advantage of parallel infrastructures. This means making minimal use of operations requiring global communication, and allowing for scalable approximations wherever possible.

2. Minimally implicit, able to take larger timesteps than explicit methods while keeping the per-timestep cost low.

3. Easy to integrate with existing code bases. Minimizing implementation cost by reducing, or eliminating, the need to construct special subroutines outside the current norm for standard integration methods.

Standard implicit methods are constructed under the assumption that the non-linear systems in each timestep will be solved exactly. In practice, however, implementations of standard implicit schemes make use of approximate methods for solving these systems. Naive application of certain approximation methods may lead to unexpected outcomes, such as loss of order or degraded stability of the time integration process.

Lightly-implicit methods turn this process on its head. Instead of constructing a method based on the assumption of exact nonlinear solution, then applying an approximate solver and hoping that it preserves the expected properties of the integrator, we identify computationally favorable approximations from the get-go and construct an integrator which permits them. The two classes of lightly-implicit methods discussed in detail later, K-methods and LIRK-W schemes, are constructed to allow Krylov based approximate solutions and the use of approximate matrix factorization, respectively.

1.3 Dissertation Overview

The remainder of this dissertation is organized around distinct research accomplishments, as follows.

Chapter 2 introduces Rosenbrock-Krylov methods, an extension of Rosenbrock-W methods which make use of a specific Krylov based approximation of the linear system. Rosenbrock-Krylov methods treat the linear system solves and time integration scheme as a single computational process. These two ideas in tandem allow for a recoloring of the trees representing the order conditions for W-methods, so that Rosenbrock-Krylov schemes require significantly fewer order conditions than W-methods of the same order. Fourth order methods are derived and tested against several standard explicit and implicit schemes throughout the literature.

Chapter 3 extends the K-method framework to exponential methods. Exponential-Krylov methods are derived, which treat the time integration scheme and approximation of the $\varphi(z)$ functions as a single computational process. Once again, trees corresponding to order conditions for W-methods are recolored to obtain an order condition theory for the Exponential-Krylov methods, with significantly fewer order conditions. Additionally, B-series are used to show the impact of different choices for approximating the $\varphi(z)$ functions on the classical order of the method.

Chapter 4 presents a linear stability analysis for K-methods, with a specific focus on Rosenbrock-Krylov schemes, and derives stiff order conditions for Rosenbrock-Krylov schemes applied to index-1 differential algebraic equations. Two practical approaches for improving the stability of Rosenbrock-Krylov methods, by directly controlling the stage residuals, are presented.

Chapter 5 introduces a flexible W-type extension of linearly-implicit Runge-Kutta IMEX schemes, which allows for arbitrary, time-dependent, and stage varying approximations of the linear stiff dynamics of an initial value problem (1.1), called linearly-implicit Runge-Kutta-W (LIRK-W) methods. Two flavors of LIRK-W schemes are presented, with the specific aim of allowing for high-order methods making use of approximate matrix factorization in the linear system solves.

Finally, in Chapter 6 we provide a brief summation and some concluding remarks.

Chapter 2

Rosenbrock-Krylov Methods for Large Systems of Differential Equations

2.1 Introduction

This paper is concerned with the numerical solution of large initial value problems of the form

$$\frac{dy}{dt} = f(t,y)\,, \quad t_0 \le t \le t_F\,, \quad y(t_0) = y_0\,; \quad y(t), f(t,y) \in \mathbb{R}^N\,. \tag{2.1}$$

Ordinary differential equations (ODEs) model the evolution of chemical kinetic systems, electronic circuits, or the motion of planets. Other physical systems, such as dynamics of the atmosphere and oceans, or the behavior of materials, require the solution of a system of partial differential equations (PDEs). A standard approach for solving evolutionary PDEs is to discretize in space, reducing the problem to a large system of ODEs which are then integrated forward in time.

The equations of interest in many simulations are driven by multiple physical processes, e.g., associated with the simulation of fluid-structure interaction, sub-grid-scale physics, or chemical reaction terms. These processes have different dynamical characteristics, with some being slow (e.g., transport) and some fast (e.g., stiff chemistry). In addition, multiple spatial and temporal scales are associated with non-uniform meshes, with boundary layers, with fast waves, and with the structure of the system (e.g., the existence of jet fans). The existence of fast and slow dynamics poses considerable challenges to the solution of the semi-discrete PDEs (2.1) by explicit time stepping methods. Specifically, due to the Courant-Friedrichs-Lewy (CFL) stability condition, the largest allowable step sizes are bounded above by the shortest (fastest) time scale in the system.

To avoid stability restrictions on the step size, implicit time integration methods are becoming widely used in the simulation of large scale evolutionary PDEs (2.1). Implicit time stepping requires the solution of huge nonlinear systems of equations, coupling all variables in the model, at each time step. The associated computation and communication costs constitute a major scalability bottleneck at best, and can quickly become infeasible for large systems.

In this paper we examine, and extend, the Rosenbrock class of integration methods. These are linearly-implicit methods which enjoy the benefit of requiring a fixed number of linear solves, as opposed to non-linear solves in the case of implicit Runge-Kutta methods. A general s-stage Rosenbrock method reads [11, Section IV.7]

$$\begin{aligned} y_{n+1} &= y_n + \sum_{i=1}^{s} b_i\, k_i\,, && \text{(2.2a)} \\ (\mathbf{I}_{N\times N} - h\,\gamma_{i,i}\,\mathbf{A})\cdot k_i &= h\,f\left(t_n + \alpha_i h, y_n + \sum_{j=1}^{i-1}\alpha_{i,j}\,k_j\right) + h\,\mathbf{A}\sum_{j=1}^{i-1}\gamma_{i,j}\,k_j && \text{(2.2b)} \\ &\quad + h^2\,\gamma_i\,f_t(t_n, y_n)\,, \qquad i = 1,\ldots,s\,. \end{aligned}$$

Here $y_n \approx y(t_n)$ and $y_{n+1} \approx y(t_{n+1})$ are the numerical solutions computed by the method, and $h = t_{n+1} - t_n$ is the current time step. The term $f_t = \partial f/\partial t$ is the partial time derivative of f evaluated at the beginning of the current time step. The method coefficients $\alpha_{i,j}$, $\gamma_{i,j}$, and b_i are chosen such as to ensure the desired accuracy and stability properties. For convenience of notation one also defines

$$\alpha_i = \sum_{j=1}^{i-1}\alpha_{i,j}\,, \quad \gamma_i = \sum_{j=1}^{i}\gamma_{i,j}\,, \quad \beta_{i,j} = \alpha_{i,j} + \gamma_{i,j}\,.$$

In classical Rosenbrock methods the matrix $\mathbf{A} = \mathbf{J}_n \in \mathbb{R}^{N\times N}$, where

$$\mathbf{J}_n = f_y(t_n, y_n)$$

is the Jacobian of f evaluated at the beginning of the current time step. The vectors k_i are the intermediate stage values; each of them is computed by solving an $N \times N$ linear system. Typically all coefficients are chosen equal to each other, $\gamma_{i,i} = \gamma$ for $i = 1,\ldots,s$, such that all stages share the same LU decomposition. The order conditions of classical Rosenbrock methods rely on the assumptions that $\mathbf{A}$ is the exact ODE Jacobian, and that each of the linear systems (2.2b) is solved exactly.

For many systems the exact Jacobian $\mathbf{J}$ can be both costly and difficult to obtain, e.g., due to the size of the application and the use of complex spatial discretization schemes. The class of Rosenbrock-W methods [11, Section IV.7] has been derived for such situations. The have the same form (2.2), but the coefficients are selected such that the overall discretization order is preserved for any matrix $\mathbf{A}$, i.e., for arbitrary approximations of the Jacobian [12].

The role of the matrix is to ensure numerical stability, and its choice dictates the type and amount of implicitness used in (2.2).

Solving for the stage values k_i (2.2b) is the most expensive part of the integrator. For large-scale systems direct methods such as LU decomposition are not feasible, and iterative Krylov based methods, such as GMRES [13, 14], have been considered in the literature [15, 16, 17]. Krylov based methods use only Jacobian-vector products and do not require storage, or even knowledge of, the full Jacobian.

The use of Krylov(-Newton) methods is a natural approach to speed up the solutions of the linear/nonlinear systems of equations in implicit time integration [18, 19]. Such integration methods have been successfully employed in real applications [20, 21, 22]. Krylov space solvers have been used in the implementation of implicit Runge-Kutta [23, 24, 25], implicit linear multistep [26, 27, 22], and deferred correction [28] methods. Software for solving stiff ODEs and DAEs with this approach include LSODKR [29], LSODPK [30], VODPK [27], and DASPK [31]. In addition, Krylov space methods have been used for the exact integration of linear ODEs with source terms [32], and to improve stability [33]. Krylov space techniques have been used to accelerate convergence of deferred correction methods [34, 35].

Of particular interest in this work is the use of Krylov methods in the context of Rosenbrock time integrators (2.2). Classical Rosenbrock integrators are poor matrix free methods due to the explicit presence of the exact Jacobian matrix, and the approximate nature of Krylov based methods. Rosenbrock-W methods are better suited for coupling with Krylov based solvers. A family of matrix-free Rosenbrock methods, named Krylov-ROW, has been proposed in the 1990's [17, 36, 37, 38, 39]. The control of linear solution accuracy in each stage is discussed in [36], and preconditioning in [40]. Order results for Krylov-ROW methods are studied in [17, 15]. A multiple Arnoldi process is proposed, where the Krylov space is enriched at each stage, such that the information from previous stages is reused, and all the right hand side vectors belong to the subspace. The order of the underlying Rosenbrock method is preserved under modest requirements on the Krylov space size, and independent of the dimension of the ODE system. The implementation of methods of order four is done in the code ROWMAP [38], where the error estimation and step size selection strategies are inherited from the underlying Rosenbrock method. The application of Krylov-ROW methods to index-1 DAEs [16] reveals that the Krylov space dimension needs to exceed the number of algebraic variables. Krylov-ROW methods are therefore attractive in the case where the number of algebraic constraints is small compared to the number of differential equations; or, by extension, where the dimension of the stiff subspace is small. Novati [41] presents a class of W-methods where the Jacobian is approximated using quasi-Newton-like rank one updates, based on solution and function values in previous time steps. Periodic restarts are needed for stability, as the Jacobian approximation deteriorates with time. A related family of methods are exponential integrators [42], which use matrix exponentials of the Jacobian as part of the solution process, and evaluate matrix exponential times vector products via Krylov space methods [43, 44, 45, 46, 47].

In this paper we develop a new family of Rosenbrock-Krylov time stepping methods characterized by the lowest possible degree of implicitness that ensures stability. The new algorithms are implicit in only the stiff subspace, which is captured by a Krylov space. Moreover, they perform only scalable operations such as Jacobian-vector products, and solve only small linear systems at each step. A naive implementation of a matrix free Rosenbrock-W method requires construction of a Krylov subspace for each stage equation. The Rosenbrock-K methods proposed herein extend the framework of Rosenbrock-W methods, accounting for the Krylov subspace approximation of the linear system when constructing the order conditions of the integrator. In this way we substantially reduce the number of required order conditions for a given order, thereby reducing the number of necessary stages of the method. The Rosenbrock-K methods require the construction of only a single Krylov subspace for the solution of all stage values. The dimension of this subspace need only be as large as the desired order of the method to ensure accuracy.

The class of Rosenbrock-K methods differs from Krylov-ROW family [17, 15] in several important aspects. First, the Krylov space properties are an integral part of Rosenbrock-K order condition theory; this elegant approach ensures the desired orders of accuracy with much smaller subspaces than those required by the Krylov-ROW approach. Next, a single Krylov space is used across all stages; the implementation of Rosenbrock-K is much simpler, and considerably more scalable, than the implementation of Krylov-ROW, where the subspace needs to be extended at each subsequent stage.

The paper is laid out as follows: in Section 2.2 we present the framework of the proposed class of methods as well as the Krylov subspace approximation of the Jacobian used, in Section 2.3 we extend the theory of order trees for Rosenbrock-w methods to our new Rosenbrock-Krylov methods as well as give details of how to construct these trees and the method order conditions from them, in Section 2.4 we construct two new Rosenbrock-Krylov integrators and outline a method for the solution of the order conditions to derive specific method coefficients, and in Section 2.5 we present some numerical results.

2.2 Formulation of Rosenbrock-Krylov methods

Rosenbrock-Krylov methods have the same form as Rosenbrock-W methods (2.2), but use a particular approximation $\mathbf{A}$ of $\mathbf{J}_n$. We start the presentation with the case of autonomous systems.

Specifically, let $f_n = f(y_n)$ and consider the M-dimensional Krylov space

$$\begin{aligned} \mathcal{K}_M(\mathbf{J}_n\,, f_n) &= \operatorname{span}\left\{ f_n,\, \mathbf{J}_n\, f_n,\, \mathbf{J}_n^2\, f_n, \ldots, \mathbf{J}_n^{M-1}\, f_n \right\} \\ &= \operatorname{span}\left\{v_1, v_2, \ldots, v_M\right\}\,. \end{aligned} \tag{2.3}$$

An orthogonal basis $\{v_i\}_{i=1,\ldots,M}$ for $\mathcal{K}_M$ is constructed using a modified Arnoldi process [14]. The Arnoldi iteration returns two valuable pieces of information: a matrix $\mathbf{V}$ whose columns are the orthonormal basis vectors of $\mathcal{K}_M$, and an upper Hessenberg matrix $\mathbf{H}$, such that

$$\mathbf{V} = [v_1, \ldots v_M] \in \mathbb{R}^{N \times M}\,; \quad \mathbf{H} = \mathbf{V}^T \mathbf{J}_n \mathbf{V} \in \mathbb{R}^{M \times M}\,. \tag{2.4}$$

The Rosenbrock-Krylov matrix $\mathbf{A}$ *is the restriction of the full ODE Jacobian to the Krylov space:*

$$\mathbf{A} = \mathbf{V}\,\mathbf{H}\,\mathbf{V}^T = \mathbf{V}\,\mathbf{V}^T\,\mathbf{J}_n\,\mathbf{V}\,\mathbf{V}^T\,. \tag{2.5}$$

To obtain the stage vector k_i we decompose it in the component residing in $\mathcal{K}_M$, and the component orthogonal to $\mathcal{K}_M$

$$k_i = \underbrace{\mathbf{V}\,\lambda_i}_{\in \mathcal{K}_M} + \underbrace{\mu_i}_{\in \mathcal{K}_M^{\perp}}\,, \tag{2.6}$$

where the new vectors λ_i and μ_i are defined by

$$\lambda_i = \mathbf{V}^T\,k_i \in \mathbb{R}^M\,, \quad \mu_i = \left(\mathbf{I}_{N\times N} - \mathbf{V}\,\mathbf{V}^T\right)\,k_i \in \mathbb{R}^N\,.$$

We consider also the projections of the function values in (2.2b) onto the Krylov space

$$f_i = f\left(y_n + \sum_{j=1}^{i-1} \alpha_{i,j} k_j\right)\,, \quad \phi_i = \mathbf{V}^T\,f_i \in \mathbb{R}^M\,.$$

To construct a Rosenbrock-K method the Jacobian approximation (2.5) and the decomposition (2.6) are used in the stage formulation (2.2b) to obtain

$$\left(\mathbf{I}_{N\times N} - h\,\gamma\,\mathbf{V}\,\mathbf{H}\,\mathbf{V}^T\right)\cdot\left(\mathbf{V}\lambda_i + \mu_i\right) = h\,f_i + h\,\mathbf{V}\,\mathbf{H}\,\mathbf{V}^T \sum_{j=1}^{i-1} \gamma_{i,j}\,\left(\mathbf{V}\lambda_j + \mu_j\right)\,.$$

Using the facts that $\mathbf{V}^T\mathbf{V} = \mathbf{I}_{M\times M}$ and $\mathbf{V}^T\mu_i = 0$ the equation can be written as

$$\mu_i + \mathbf{V}\left(\mathbf{I}_{M\times M} - h\,\gamma\,\mathbf{H}\right)\,\lambda_i \;=\; h\,f_i + h\,\mathbf{V}\,\mathbf{H} \sum_{j=1}^{i-1} \gamma_{i,j}\,\lambda_j\,. \tag{2.7}$$

Multiplying both sides of (2.7) by $\mathbf{V}^T$ leads to the following reduced stage equation

$$\left(\mathbf{I}_{M\times M} - h\gamma\mathbf{H}\right)\lambda_i = h\phi_i + h\mathbf{H} \sum_{j=1}^{i-1} \gamma_{i,j}\lambda_j\,. \tag{2.8}$$

Multiplying both sides of (2.7) by $\mathbf{I}_{N\times N} - \mathbf{V}\mathbf{V}^T$ gives

$$\mu_i = h\,\left(f_i - \mathbf{V}\,\phi_i\right)\,. \tag{2.9}$$

Algorithm 1 One step of an autonomous Rosenbrock-K integrator

1: Compute $\mathbf{H}$ and $\mathbf{V}$ (2.4) using the N-dimensional Arnoldi process [14]
2: **for** $i = 1, \ldots, s$ **do** ▷ For each stage, in succession

$$\begin{aligned} f_i &= f\left(y_n + \sum_{j=1}^{i-1} \alpha_{i,j} k_j\right) \\ \phi_i &= \mathbf{V}^T f_i \\ \lambda_i &= \left(\mathbf{I}_{M\times M} - h\gamma\mathbf{H}\right)^{-1}\left(h\phi_i + h\mathbf{H}\sum_{j=1}^{i-1}\gamma_{i,j}\lambda_j\right) \\ k_i &= \mathbf{V}\,\lambda_i + h\left(f_i - \mathbf{V}\,\phi_i\right) \end{aligned}$$

3: **end for**
4: $y_{n+1} = y_n + \sum_{i=1}^{s} b_i k_i$

The system (2.8) is of size $M \times M$, where $M \ll N$, and can be inverted through the use of a direct method. The full stage values can now be recovered from (2.6), (2.8), and (2.9) as

$$k_i = \mathbf{V}\,\lambda_i + h\,\left(f_i - \mathbf{V}\phi_i\right)\,. \tag{2.10}$$

An autonomous Rosenbrock-K step from t_n to t_{n+1} is summarized in Algorithm 1.

We next consider the formulation of Rosenbrock-K methods for non-autonomous problems (2.1). With the extended state and function

$$\widehat{y}(t) = \begin{bmatrix} y(t) \\ t \end{bmatrix} \in \mathbb{R}^{N+1}\,, \quad \widehat{f}(\widehat{y}) = \begin{bmatrix} f(t,y) \\ 1 \end{bmatrix} \in \mathbb{R}^{N+1}\,, \tag{2.11}$$

the general ODE (2.1) can be formulated as autonomous system

$$\frac{d\widehat{y}}{dt} = \widehat{f}(\widehat{y})\,, \quad \widehat{y}(t_0) = \begin{bmatrix} y \\ t_0 \end{bmatrix}, \quad t_0 \le t \le t_F, \quad \widehat{y}(t)\,, \widehat{f}(\widehat{y})\,. \tag{2.12}$$

Non-autonomous Rosenbrock-K integrators are constructed using this technique. The Jacobian of the extended right hand side function is

$$\widehat{\mathbf{J}} = \begin{bmatrix} f_y & f_t \\ 0 & 0 \end{bmatrix}. \tag{2.13}$$

An extended Krylov space $\mathcal{K}_M(\widehat{\mathbf{J}}_n\,, \widehat{f}_n)$ is constructed using matrix-vector products of the form

$$\widehat{\mathbf{J}}_n \begin{bmatrix} \zeta \\ \xi \end{bmatrix} = \begin{bmatrix} \mathbf{J}_n\,\zeta + f_t(t_n,y_n)\,\xi \\ 0 \end{bmatrix}\,, \quad \zeta \in \mathbb{R}^N\,, \quad \xi \in \mathbb{R}\,. \tag{2.14}$$

The extended $(N+1)$-dimensional Arnoldi iterations produce the matrices $\widehat{\mathbf{V}}$ and $\mathbf{H}$ such that

$$\widehat{\mathbf{V}} = \begin{bmatrix} \mathbf{V} \\ w^T \end{bmatrix} \in \mathbb{R}^{(N+1)\times M}\,, \quad \mathbf{V} \in \mathbb{R}^{N\times M}\,, \quad w \in \mathbb{R}^{M}\,, \tag{2.15}$$

and

$$\mathbf{H} = \widehat{\mathbf{V}}^T\, \widehat{\mathbf{J}}_n\, \widehat{\mathbf{V}} = \mathbf{V}^T\, \mathbf{J}_n\, \mathbf{V} + \mathbf{V}^T\, f_t(t_n, y_n)\, w^T \in \mathbb{R}^{M\times M}\,.$$

This modified Arnoldi iteration proceeds as follows.

Algorithm 2 Modified Arnoldi iteration

$\beta = \left\| \begin{bmatrix} f_n \\ 1 \end{bmatrix} \right\|\,, \quad w_1 = 1/\beta, \quad v_1 = f_n/\beta\,.$
for $i = 1, \ldots, M$ **do**
  $\zeta = \mathbf{J}_n\, v_i + f_t(t_n, y_n)\, w_i$
  $\xi = 0$
  $\tau = \|\zeta\|$
  for $j = 1, \ldots, i$ **do**
    $H_{j,i} = \langle \zeta, v_j \rangle + \xi\, w_j$
    $\zeta = \zeta - H_{j,i}\, v_j$
    $\xi = \xi - H_{j,i}\, w_j$
  end for
  if $\left\| \begin{bmatrix} \zeta \\ \xi \end{bmatrix} \right\| / \tau \le \kappa$ **then**
    for $j = 1, \ldots, i$ **do**
      $\rho = \langle \zeta, v_j \rangle + \xi\, w_j$
      $\zeta = \zeta - \rho\, v_j$
      $\xi = \xi - \rho\, w_j$
      $H_{j,i} = H_{j,i} + \rho$
    end for
  end if
  $H_{i+1,i} = \left\| \begin{bmatrix} \zeta \\ \xi \end{bmatrix} \right\|$
  $v_{i+1} = \zeta / H_{i+1,i}$
  $w_{i+1} = \xi / H_{i+1,i}$
end for

The non-autonomous Rosenbrock-K integrator is obtained by applying the autonomous step (Algorithm 1) to the extended system (2.12), and decoupling the state and time variables. The procedure is summarized in Algorithm 3.

Algorithm 3 One step of a non-autonomous Rosenbrock-K integrator

1: Compute $\mathbf{H}$, $\mathbf{V}$, and w (2.15) using the $(N+1)$-dimensional Arnoldi process.

2: **for** $i = 1, \ldots, s$ **do** ▷ For each stage, in succession

$$\begin{aligned} f_i &= f\left(t_n + \alpha_i h \,,\, y_n + \sum_{j=1}^{i-1} \alpha_{i,j} k_j\right) \\ \phi_i &= \mathbf{V}^T f_i + w \\ \lambda_i &= \left(\mathbf{I}_{M\times M} - h\gamma\mathbf{H}\right)^{-1} \left(h\phi_i + h\mathbf{H}\sum_{j=1}^{i-1} \gamma_{i,j}\lambda_j\right) \\ k_i &= \mathbf{V}\,\lambda_i + h\left(f_i - \mathbf{V}\,\phi_i\right) \end{aligned}$$

3: **end for**

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

2.3 Order Conditions

The accuracy theory is based on matching the Taylor series of the numerical solution and of the exact solution, up to some specified order. Butcher-trees [48, Section IV.7] are a well accepted method of representing individual terms in the Taylor series expansions. The derivation of order conditions for Rosenbrock-K methods is an extension of the framework used to derive order conditions for Rosenbrock-W methods. The existing theory for Rosenbrock-W methods is based on the use of TW-trees, a subclass of P-trees [11, Section IV.7]. P-trees themselves are an extension of the set T of Butcher-trees that allow for two different colors of the nodes. We have the following definition [11, Section IV.7]:

$$TW = \left\{ \begin{array}{ll} P\text{-trees:} & \text{end vertices are meagre, and} \\ & \text{fat vertices are singly branched} \end{array} \right\}$$

In the context of Rosenbrock-K and Rosenbrock-W methods, a meagre node represents an actual derivative of f coming from the first term on the right of equation (2.2b), while a meagre node is an appearance of the approximate Jacobian $\mathbf{A}$ coming from the second term on the right of equation (2.2b). Each tree represents a single elementary differential in the Taylor series of either the exact or numerical solutions of the ODE.

Figure 2.1 shows all TW-trees and Rosenbrock-W conditions for up to order three [11, Section IV.7]. The correspondence between the TW-trees and elementary differentials, and Rosenbrock-W order conditions, is summarized next:

- For the elementary differentials in Figure 2.1 superscripts represent component indices, and subscripts represent indices of variables with respect to which partial derivatives are taken. For example, f^J is the J-th component of f, and $f^J_K f^K = \sum_K \partial f^J / \partial y^K \cdot f^K$.

- A meagre node represents a derivative of f.
- The order of the f derivative equals the number of children the meagre node has.
- A fat node represents the appearance of the approximate Jacobian matrix, $\mathbf{A}$, in the elementary differential.

The correspondence between the TW-trees and Rosenbrock-W order conditions, is as follows:

- For the order conditions in Figure 2.1 the summations apply to all repeated indices in the expression.
- For Rosenbrock-W methods:
  - an edge connecting meagre node j to a fat node k gives $\alpha_{j,k}$,
  - an edge connecting meagre node j to a meagre node k gives $\gamma_{j,k}$.
- For classical Rosenbrock methods
  - an edge connecting meagre node j, having multiple children, to a meagre node k gives $\alpha_{j,k}$,
  - an edge connecting meagre node j, having a single child, to a meagre node k gives $\beta_{j,k}$.

The exact solution is represented by trees containing only meagre nodes, since the approximate Jacobian matrix never appears in its series expansion. For this reason Rosenbrock-W methods have two sets of order conditions: those arising from trees containing only meagre needs, and those arising from trees containing at least one fat node. Trees containing fat nodes do not correspond to any trees in the exact solution and, as seen in Figure 2.1, the corresponding coefficients are set to zero [11].

In order to build the relevant trees for Rosenbrock-K methods we need to have a closer look at the properties of the Jacobian approximation (2.5).

Lemma 1 (Property of the Rosenbrock-Krylov approximate Jacobian (2.5))**.** *For any* $0 \leq k \leq M-1$ *it holds that*

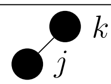
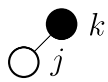
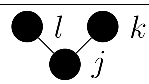
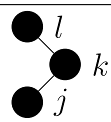
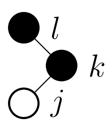
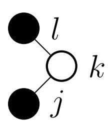
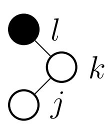
$$\mathbf{A}^k \, f_n = \mathbf{J}_n^k \, f_n \,,$$

where $M = dim(\mathcal{K}_M)$*.*

Proof. Recall that $\mathbf{V}\,\mathbf{V}^T$ is the orthogonal projector onto $\mathcal{K}_M$. If a vector u is in the Krylov space $\mathcal{K}_M$, its orthogonal projection onto $\mathcal{K}_M$ is the vector itself:

$$u \in \mathcal{K}_M \quad \Rightarrow \quad \mathbf{V}\,\mathbf{V}^T\, u = u \,.$$

Figure 2.1: TW-trees and Rosenbrock-W conditions up to order three [11, Section IV.7].

a		f^J	$\sum b_j$	$=$	1
b_1		$f^J_K f^K$	$\sum b_j \alpha_{j,k}$	$=$	$1/2$
b_2		$\mathbf{A}_{JK} f^K$	$\sum b_j \gamma_{j,k}$	$=$	0
c		$f^J_{KL} f^K f^L$	$\sum b_j \alpha_{j,k} \alpha_{j,l}$	$=$	$1/3$
d_1		$f^J_K f^K_L f^L$	$\sum b_j \alpha_{j,k} \alpha_{k,l}$	$=$	$1/6$
d_2		$\mathbf{A}_{JK} f^K_L f^L$	$\sum b_j \gamma_{j,k} \alpha_{k,l}$	$=$	0
d_3		$f^J_K \mathbf{A}_{KL} f^L$	$\sum b_j \alpha_{j,k} \gamma_{k,l}$	$=$	0
d_4		$A_{JK} \mathbf{A}_{KL} f^L$	$\sum b_j \gamma_{j,k} \gamma_{k,l}$	$=$	0

The proof of the Lemma is by finite induction. As the base case we have that

$$\mathbf{A}^0\, f_n = \mathbf{J}_n^0\, f_n = f_n\,.$$

Next we assume that $\mathbf{A}^{i-1}\, f_n = \mathbf{J}_n^{i-1}\, f_n$ for some $i \le M-1$ and will show that $\mathbf{A}^i\, f_n = \mathbf{J}_n^i\, f_n$. By the definition of the approximate Jacobian (2.5) and our assumption it holds that

$$\mathbf{A}^i\, f_n = \mathbf{A}\cdot\mathbf{A}^{i-1}\, f_n = \mathbf{A}\cdot\mathbf{J}_n^{i-1}\, f_n = \mathbf{V}\,\mathbf{V}^T\,\mathbf{J}_n\,\mathbf{V}\,\mathbf{V}^T\cdot\mathbf{J}_n^{i-1}\, f_n\,.$$

Since $M \ge i$ we have that $\mathbf{J}_n^{i-1} f_n \in \mathcal{K}_M$, and

$$\mathbf{V}\,\mathbf{V}^T\,\mathbf{J}_n^{i-1} f_n = \mathbf{J}_n^{i-1} f_n \quad\Rightarrow\quad \mathbf{A}^i\, f_n = \mathbf{V}\,\mathbf{V}^T\,\mathbf{J}_n\cdot\mathbf{J}_n^{i-1} f_n = \mathbf{V}\,\mathbf{V}^T\,\mathbf{J}_n^i f_n\,.$$

Since $M \ge i+1$ we have that $\mathbf{J}_n^i f_n \in \mathcal{K}_M$ and

$$\mathbf{V}\,\mathbf{V}^T\,\mathbf{J}_n^i f_n = \mathbf{J}_n^i f_n \quad\Rightarrow\quad \mathbf{A}^i\, f_n = \mathbf{J}_n^i f_n\,.$$

□

Lemma 2 (Property of elementary differentials using the approximation (2.5))**.** *When the Rosenbrock-Krylov matrix approximation* (2.5) *is used, all linear TW-trees of order $k \le M$ correspond to a single elementary differential, regardless of the color of their nodes.*

Proof. In a linear tree each node has only one child. A linear TW-tree with a fat root can be described by the sequence of its nodes, starting from the root. For example, the structure of a linear tree where the first ν_1 nodes from the root are fat, followed by μ_1 meagre nodes, etc. is described by the sequence $(\circ^{\nu_1}\,\bullet^{\mu_1}\cdots\circ^{\nu_p}\,\bullet^{\mu_p})$ with $\mu_p > 1$ (since the leaf is a meagre node).

Consider now a tree of order $k = \nu_1+\mu_1+\cdots+\nu_p+\mu_p \le M$. The corresponding elementary differential has the form $\mathbf{A}^{\nu_1}\mathbf{J}_n^{\mu_1}\ldots\mathbf{A}^{\nu_p}\mathbf{J_n}^{\mu_p-1} f_n$. Repeated applications of Lemma 1 reveal that

$$\mathbf{A}^{\nu_1}\mathbf{J}_n^{\mu_1}\ldots\mathbf{A}^{\nu_p}\underbrace{\mathbf{J_n}^{\mu_p-1} f_n}_{=\mathbf{A}^{\mu_p-1} f_n} \;=\; \mathbf{A}^{\nu_1}\mathbf{J}_n^{\mu_1}\ldots\underbrace{\mathbf{A}^{\nu_p+\mu_p-1} f_n}_{=\mathbf{J}_n^{\nu_p+\mu_p-1} f_n} = \cdots = \mathbf{J_n}^{k-1} f_n\,.$$

Consequently, any linear TW-tree of order $k \le M$ has the same differential as the linear tree with only meagre nodes ($\bullet^k$). □

An important consequence of Lemma 2 is that if a linear TW sub-tree with $k \le M$ nodes has a fat root, the corresponding differential is the same as for the linear tree with only meagre nodes ($\bullet^k$). This observation allows us to essentially "recolor" linear TW sub-trees with a fat root (i.e., group them in classes of equivalence). This leads to the following.

Definition 1 (TK-trees)**.**

$$
\begin{aligned}
TK &= \{\textit{TW-trees: no linear sub-tree has a fat root}\}\ ; \\
TK(k) &= \{\textit{TW-treesp: no linear sub-tree of order} \\
&\qquad\qquad \textit{smaller than or equal to } k \textit{ has a fat root}\}\ .
\end{aligned}
$$

Fot $t \in TK$ let $\rho(t)$ define the number of vertices. We denote by $t = \bullet[t_1, \ldots, t_m]$ a TK-tree tree with a meagre root linking the subtrees $t_1, \ldots, t_m$, and by $t = \circ[t_1]$ a TK-tree with a fat root to which the subtree t_1 is connected. A special case is the single node tree $\bullet[\,]$. The elementary differentials associated with TK-trees are the same as those of TW-trees, [11, Definition 7.5]. The TK-tree coefficients are constructed recursively as follows.

Definition 2 (Coefficients of TK-trees)**.** *For $t \in TK$*

$$
\phi_j(t) = \begin{cases}
1 & \textit{if} \quad t = \bullet[\,] \\
\displaystyle\sum_{k_1,\ldots,k_m} \alpha_{j,k_m}\, \phi_{k_1}(t_1)\ldots\phi_{k_m}(t_m) & \textit{if} \quad t = \bullet[t_1,\ldots,t_m],\ m \geq 2 \\
\displaystyle\sum_{k} \beta_{j,k}\, \phi_k(t_1) & \textit{if} \quad t = \bullet[t_1] \\
\displaystyle\sum_{k} \gamma_{j,k}\, \phi_k(t_1) & \textit{if} \quad t = \circ[t_1]
\end{cases}
$$

2.3.1 Rosenbrock-K methods of type 1

Definition 3. *A Rosenbrock-K method of type 1 is given by Algorithm 1 (or Algorithm 3) and uses an underlying Krylov subspace given by* (2.3)*.*

Theorem 1 (Order conditions for Rosenbrock-K methods)**.** *A Rosenbrock-K method of type 1 has order p iff the underlying Krylov space* (2.3) *has dimension $M \geq p$, and the following order conditions hold:*

$$
\sum_j b_j\, \phi_j(t) = \frac{1}{\gamma(t)} \quad \forall\, t \in T \quad \textit{with } \rho(t) \leq p\,, \tag{2.16a}
$$

$$
\sum_j b_j\, \phi_j(t) = 0 \quad \forall\, t \in TK \backslash T \quad \textit{with } \rho(t) \leq p\,. \tag{2.16b}
$$

Here $\rho(t)$ is the number of vertices of the tree t, and $\gamma(t)$ is the "product of $\rho(t)$ and all orders of the trees which appear, if the roots, one after another, are removed from t" [48, Section II.2].

Proof. The proof follows from our discussion and from the order conditions of Rosenbrock-W methods [11, Theorem 7.7]. □

The following result follows immediately.

Theorem 2 (Order conditions for Rosenbrock-K methods with smaller Krylov space)**.** *A Rosenbrock-K method of type 1 with an underlying Krylov space* (2.3) *of dimension $M < p$ has order p iff condition* (2.16a) *holds, and, in addition:*

$$\sum_j b_j \, \phi_j(t) = 0 \quad \forall \, t \in TK(M) \backslash T \quad \textit{with } \rho(t) \leq p \, . \tag{2.16c}$$

Remark 1. *The number of required order conditions for Rosenbrock-K methods is substantially smaller than the number of order conditions for Rosenbrock-W methods.*

Figure 2.1 reveals that all TW-trees up to order three containing a fat root are linear, and so every tree containing a fat node can be recolored to contain only meagre nodes. Thus the order conditions for Rosenbrock-K methods are the same as those for classical Rosenbrock methods for up to order three (while Rosenbrock-W methods need four additional conditions). Figure 2.3.1 shows the TK-trees and order conditions for up to order four; Rosenbrock-K methods require only a single extra order condition for order four (while Rosenbrock-W methods require seventeen additional conditions). Finally, Figure 2.3.1 shows the four additional TK-trees and Rosenbrock-K conditions needed for order five.

2.3.2 Rosenbrock-K methods of type 2

Definition 4. *A Rosenbrock-K method of type 2 is given by Algorithm 1 (or Algorithm 3) and uses an enriched underlying Krylov subspace, where additional basis vectors are added to those in* (2.3)*. The additional basis vectors are chosen such that different elementary differentials associated with trees in $TK \backslash T$ are equal to those of similar trees in T. Consequently, the order conditions of a type 2 Rosenbrock-K method are the same as those of classical Rosenbrock methods.*

For example, consider the tree g_2 in Table 2.3.1. The corresponding term in the Taylor series of the solution is $\mathbf{A} \cdot f_{y,y}(f_n, f_n)$. The application of the second derivative tensor $f_{y,y}$ to a pair of function values results in the vector

$$u^k_{g_2} = \left(f_{y,y}(f_n, f_n)\right)^k = \sum_{\ell,m=1}^{N} \left. \frac{\partial^2 f^k}{\partial y^\ell \, \partial y^m} \right|_{y=y_n} f^\ell(y_n) \, f^m(y_n) \, , \quad k = 1, \ldots, N \, . \tag{2.17}$$

To obtain a type 2 Rosenbrock-K method of order four the Krylov space (2.3) is extended as follows:

$$\begin{aligned} \mathcal{K}_{M+2}(\mathbf{J}_n \, , f_n) &= \operatorname{span} \left\{ \, f_n, \, \mathbf{J}_n \, f_n, \, \ldots, \mathbf{J}_n^{M-1} \, f_n \, , u_{g_2} \, , \mathbf{J}_n \, u_{g_2} \, \right\} \\ &= \operatorname{span} \left\{ v_1, v_2, \ldots, v_{M+2} \right\} \, . \end{aligned} \tag{2.18}$$

Figure 2.2: TK-trees and Rosenbrock-Krylov conditions up to order four.

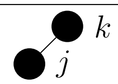
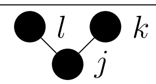
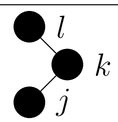
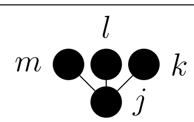
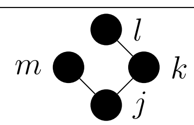
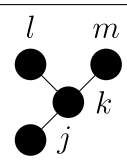
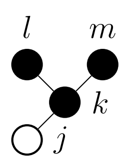
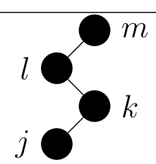
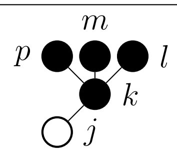
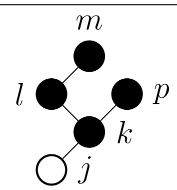
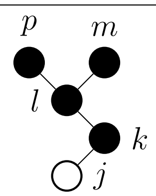
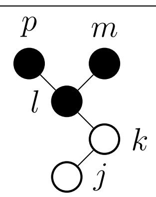
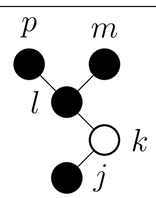
a		f^J	$\sum b_j$	$=$	1
b		$f^J_K f^K$	$\sum b_j \beta_{j,k}$	$=$	$1/2$
c		$f^J_{KL} f^K f^L$	$\sum b_j \alpha_{j,k} \alpha_{j,l}$	$=$	$1/3$
d		$f^J_K f^K_L f^L$	$\sum b_j \beta_{j,k} \beta_{k,l}$	$=$	$1/6$
e		$f^J_{KLM} f^K f^L f^M$	$\sum b_j \alpha_{j,k} \alpha_{j,l} \alpha_{jm}$	$=$	$1/4$
f		$f^J_{KM} f^K_L f^L f^M$	$\sum b_j \alpha_{j,k} \beta_{k,l} \alpha_{j,m}$	$=$	$1/8$
g_1		$f^J_K f^K_{LM} f^L f^M$	$\sum b_j \alpha_{j,k} \alpha_{k,m} \alpha_{k,l}$	$=$	$1/12$
g_2		$\mathbf{A}_{JK} f^K_{LM} f^L f^M$	$\sum b_j \gamma_{j,k} \alpha_{k,m} \alpha_{k,l}$	$=$	0
h		$f^J_K f^K_L f^L_M f^M$	$\sum b_j \beta_{j,k} \beta_{k,l} \beta_{l,m}$	$=$	$1/24$

Figure 2.3: Additional TK-trees and Rosenbrock-Krylov conditions for order five.

	$\mathbf{A}_{JK} f^K_{LMP} f^L f^M f^P$	$\sum \gamma_{j,k}\alpha_{k,l}\alpha_{k,m}\alpha_{k,p} \quad = \quad 0$
	$\mathbf{A}_{JK} f^K_{LP} f^L_M f^M f^P$	$\sum \gamma_{j,k}\alpha_{k,p}\alpha_{k,l}\beta_{l,m} \quad = \quad 0$
	$\mathbf{A}_{JK} f^K_L f^L_{MP} f^P f^M$	$\sum \gamma_{j,k}\alpha_{k,l}\alpha_{l,m}\alpha_{l,p} \quad = \quad 0$
	$\mathbf{A}_{JK}\mathbf{A}_{KL} f^L_{MP} f^P f^M$	$\sum \gamma_{j,k}\gamma_{k,l}\alpha_{l,m}\alpha_{l,p} \quad = \quad 0$
	$f^J_K \mathbf{A}_{KL} f^L_{MP} f^P f^M$	$\sum \alpha_{j,k}\gamma_{k,l}\alpha_{l,m}\alpha_{l,p} \quad = \quad 0$

The Jacobian approximation is $\mathbf{A} = \mathbf{V}\,\mathbf{H}\,\mathbf{V}^T$ where $\mathbf{V} \in \mathbb{R}^{N\times(M+2)}$, and $\mathbf{H} = \mathbf{V}^T\,\mathbf{J}_n\,\mathbf{V} \in \mathbb{R}^{(M+2)\times(M+2)}$ is no longer upper Hessenberg.

The construction (2.18) ensures that the elementary differential of the tree $g_2 \in TW\backslash T$ coincides with the elementary differential of a regular Butcher tree:

$$\mathbf{A}\, f_{y,y}(f_n, f_n) = \mathbf{A}\, u_{g_2} = \mathbf{V}\,\mathbf{V}^T\,\mathbf{J}_n \underbrace{\mathbf{V}\,\mathbf{V}^T\, u_{g_2}}_{=u_{g_2}} = \mathbf{V}\,\mathbf{V}^T\,\mathbf{J}_n\, u_{g_2} = \mathbf{J}_n\, u_{g_2} = \mathbf{J}_n\, f_{y,y}(f_n, f_n)\,.$$

We have the following interesting consequences.

Remark 2. *Any classical Rosenbrock method of order four (or higher) becomes a type 2 Rosenbrock-K method of order four when the Jacobian approximation* (2.5) *uses a six-dimensional extended Krylov space (more exactly, when the underlying Krylov space is* (2.18) *with $M \geq 4$).*

Remark 3. *General Rosenbrock-K methods of any order can be obtained by combining the type 2 and type 1 approaches. Specifically, some of the trees in $TK\backslash T$ are recolored (to obtain the similar trees in T) by extending the underlying Krylov space, i.e., by using a type 2 approach. The elementary differentials corresponding to the remaining trees in $TK\backslash T$ are then cancelled by imposing additional type 1 order conditions.*

2.3.3 Finite difference approximations

The Arnoldi iteration requires only Jacobian-vector products. These can be obtained by automatic differentiation. Alternatively, Jacobian-vector products can be approximated by finite differences of the form [19]

$$\mathbf{J}_n\, u \approx \frac{f(t_n, y_n + \delta\, u) - f(t_n, y_n)}{\delta}\,. \tag{2.19}$$

The increment δ is related to machine precision. Equation (2.19) is sometimes referred to as a "matrix-free" approximation. For example, "Jacobian-free Newton-Krylov" methods [19] employ the approximation (2.19) within linear Krylov space solvers in the context of Newton iterations for nonlinear systems.

Similar finite differences can also be used to approximate higher order derivatives. For example, the second derivative term (2.17) can approximated by finite differences of Jacobian-vector products, as follows:

$$f_{y,y}(u,u) \;\approx\; \frac{f_y(t_n, y_n + \delta\, u)\cdot u - \mathbf{J}_n \cdot u}{\delta} \approx \frac{f(t_n, y_n + 2\delta\, u) - 2\, f(t_n, y_n + \delta\, u) + f(t_n, y_n)}{\delta^2}\,.$$

An analysis of matrix-free Newton-Krylov methods is provided in [49, Theorem 2.3]. Assume that the Arnoldi process with the exact Jacobian-vector products produces $\mathbf{H}$, $\mathbf{V}$, while the

Arnoldi process using finite difference approximations (2.19) produces $\tilde{\mathbf{H}}$ and $\tilde{\mathbf{V}}$. The errors in the finite difference approximations (2.19) during each Arnoldi iteration are

$$e_i \;=\; \frac{f(t_n, y_n + \delta_i\, \tilde{v}_i) - f(t_n, y_n)}{\delta_i} - \mathbf{J}_n \cdot \tilde{v}_i, \quad i = 1, \ldots, M-1\,.$$

Collect these error vectors, together with $e_0 = 0$ (the error in computing $\tilde{v}_1 = f_n / \|f_n\|$), in the matrix

$$\mathbf{E} = [e_0, e_1, \ldots, e_{M-1}] \in \mathbb{R}^{N \times M}\,.$$

According to [49, Theorem 2.3], the matrices $\tilde{\mathbf{H}}$ and $\tilde{\mathbf{V}}$ can be obtained by an application of the *exact* Arnoldi process (i.e., with exact Jacobian-vector products) to obtain a basis of the modified space

$$\mathcal{K}_M\left(\tilde{\mathbf{J}}_n, f_n\right) \quad \text{with} \quad \tilde{\mathbf{J}}_n := \mathbf{J}_n + \mathbf{E}\,\tilde{\mathbf{V}}^T\,.$$

According to Lemma 1, the matrix approximation $\tilde{\mathbf{A}} = \tilde{\mathbf{V}}\tilde{\mathbf{H}}\tilde{\mathbf{V}}^T$ has the following property for any $0 \le k \le M-1$

$$\begin{aligned}
\tilde{\mathbf{A}}^k f_n &= \tilde{\mathbf{J}}_n^{\,k} f_n \\
&= \left(\mathbf{J}_n + \mathbf{E}\,\tilde{\mathbf{V}}^T\right)^k f_n \\
&= \mathbf{J}_n^k f_n + k\,\mathbf{J}_n^{k-1}\,\mathbf{E}\,\tilde{\mathbf{V}}^T f_n + \ldots \\
&= \mathbf{J}_n^k f_n + \mathcal{O}(\|\mathbf{E}\|)\,,
\end{aligned}$$

where we have made the assumption that the Jacobian powers are uniformly bounded.

When exact Jacobian-vector products are used no additional type 1 Rosenbrock-K order conditions are imposed for trees in $TW \backslash TK$. Similarly, when higher derivatives are computed exactly no additional order conditions are needed for type 2 methods. When finite difference approximations are used, however, the elementary differentials of trees in $TW \backslash TK$ appear in the expansion of the numerical solution with nonzero coefficients of size $\mathcal{O}(\|\mathbf{E}\|)$, i.e., of the size of the *absolute* errors incurred in the finite difference approximations. If the finite difference approximations are not sufficiently accurate the order of the Rosenbrock-K methods may be lost.

For example, consider the tree $d_2 \in TW \backslash TK$ in Figure 2.1. When finite differences are used, it contributes the following $\mathcal{O}\left(\|\mathbf{E}\|\, h^2\right)$ term to the local error

$$h^2 \left(\sum_{j=1}^{s} b_j \gamma_{j,k}\right) (\mathbf{E}\,\tilde{v}_1)\, \|f_n\|\,.$$

In order to ensure that the Rosenbrock-K method preserves its order p, a sufficient condition is that the finite difference errors are bounded by

$$\|\mathbf{E}\| \le C\, h^{p-1}\,.$$

When the exact Jacobian-vector products are unavailable, and when the finite difference approximations cannot be computed this accurately, it may be advantageous to choose Rosenbrock-K methods whose coefficients satisfy the full set of Rosenbrock-W conditions.

2.4 Construction of Rosenbrock-K methods of order four

We now construct practical type 1 Rosenbrock-K methods of order four. We consider the case with $\gamma_{i,i} = \gamma$ for all i, and denote

$$\beta_i = \sum_{j=1}^{i} \beta_{i,j} = \alpha_i + \gamma_i \,, \quad \beta_i' = \sum_{j=1}^{i-1} \beta_{i,j} \,.$$

We examine numerically the linear stability properties of the resulting methods. Rosenbrock-K methods share the same stability function with classical Rosenbrock methods

$$R(z) = 1 + z b^T \left(\mathbf{I}_{s\times s} - z\beta\right)^{-1} \mathbb{1} \,, \tag{2.20}$$

where $\mathbb{1} \in \mathbb{R}^s$ is a vector of ones.

2.4.1 ROK4a: a four stages, fourth order, L-stable Rosenbrock-K method

We start with constructing a four stages, fourth order Rosenbrock-K method. The order conditions are as follows:

$$\begin{array}{rlcl}
(a) & b_1 + b_2 + b_3 + b_4 & = & 1 \\
(b) & b_2\beta_2' + b_3\beta_3' + b_4\beta_4' & = & \frac{1}{2} - \gamma = p_{21}(\gamma) \\
(c) & b_2\alpha_2^2 + b_3\alpha_3^2 + b_4\alpha_4^2 & = & \frac{1}{3} \\
(d) & b_3(\beta_{3,2}\beta_2') + b_4(\beta_{4,2}\beta_2' + \beta_{4,3}\beta_3') & = & \frac{1}{6} - \gamma + \gamma^2 = p_{3,2}(\gamma) \\
(e) & b_2\alpha_2^3 + b_3\alpha_3^3 + b_4\alpha_4^3 & = & \frac{1}{4} \\
(f) & b_3\alpha_{3,2}\beta_2' + b_4(\alpha_{4,2}\beta_2' + \alpha_{4,3}\beta_3) & = & \frac{1}{8} - \frac{1}{3}\gamma = p_{4,2}(\gamma) \\
(g_1) & b_3\alpha_{3,2}\alpha_2^2 + b_4(\alpha_{4,2}\alpha_2^2 + \alpha_{4,3}\alpha_3^2) & = & \frac{1}{12} \\
(g_2) & b_3\gamma_{3,2}\alpha_2^2 + b_4(\gamma_{4,2}\alpha_2^2 + \gamma_{4,3}\alpha_3^2) & = & -\frac{1}{3}\gamma \\
(h) & b_4\beta_{4,3}\beta_{3,2}\beta_2' & = & \frac{1}{24} - \frac{1}{2}\gamma + \frac{3}{2}\gamma^2 - \gamma^3 = p_{4,4}(\gamma)
\end{array} \tag{2.21}$$

To solve (2.21) we follow the solution process outlined in [11, Section IV.7]. First we choose $\gamma = 0.572816062482135$ so that $R(\infty) = 0$, where $R(z)$ is the stability function of the method. We then treat equations (2.21.a), (2.21.c), and (2.21.e) separately, as a linear system in b_i's. We make the arbitrary choices

$$b_3 = 0, \quad \alpha_2 = \frac{1}{2}, \quad \alpha_3 = 1, \quad \alpha_4 = 1\,,$$

and the solve the system

$$\begin{bmatrix} 1 & 1 & 1 \\ 0 & \alpha_2^2 & \alpha_4^2 \\ 0 & \alpha_2^3 & \alpha_4^3 \end{bmatrix} \begin{bmatrix} b_1 \\ b_2 \\ b_4 \end{bmatrix} = \begin{bmatrix} 1 \\ \frac{1}{3} \\ \frac{1}{4} \end{bmatrix}$$

to obtain b_1, b_2, and b_4.

In order to allow for the existence of an embedded method of order three we require that the third order conditions are not satisfied uniquely. The following equation guarantees that by setting the determinant of the system of third order conditions to zero:

$$(\beta_2'\alpha_4^2 - \beta_4'\alpha_2)\,\beta_{3,2}\,\beta_2' = (\beta_2'\alpha_3^2 - \beta_3'\alpha_2^2)\sum_{j=2}^{3}\beta_{4,j}\beta_j'\,. \tag{2.22}$$

We now take $\beta_{4,3}$ as a free parameter and compute $\beta_{3,2}\beta_2'$ from (2.21.h) and $(\beta_{4,2}\beta_2' + \beta_{4,3}\beta_3')$ from (2.21.d). Inserting these expressions into (2.22) yields a relation between β_2', β_3', β_4'. Eliminating $(b_4\beta_{4,2} + b_3\beta_{3,2})$ from (2.21.d), and from $\{(2.21.g_1) + (2.21.g_2)\}$, yields a second relation. A third relation is obtained from (2.21.b), and this leads to the following system for β_2', β_3', β_4':

$$\begin{bmatrix} \alpha_4^2\frac{p_{4,4}}{b_4\beta_{4,3}} - \alpha_3^2\frac{p_{3,2}}{b_4} & \alpha_2^2\frac{p_{3,2}}{b_4} & -\alpha_2^2\frac{p_{4,4}}{b_4\beta_{4,3}} \\ b_2 & b_3 & b_4 \\ b_4\beta_{4,3}\alpha_3^2 - p_{4,3} & -b_4\beta_{4,3}\alpha_2^2 & 0 \end{bmatrix} \begin{bmatrix} \beta_2' \\ \beta_3' \\ \beta_4' \end{bmatrix} = \begin{bmatrix} 0 \\ p_{2,1} \\ -\alpha_2^2 p_{3,2} \end{bmatrix}.$$

Here we make the arbitrary choice

$$\beta_{43} = -\frac{1}{4}$$

and compute $\beta_{3,2}$ and $\beta_{4,2}$ from

$$\beta_{3,2} = \frac{p_{4,4}}{b_4\beta_{4,3}\beta_2'}, \quad \beta_{4,2} = \frac{p_{3,2} - b_4\beta_{4,3}\beta_3'}{b_4\beta_2'}\,.$$

Next we impose directly equations (2.21.f), $(2.21.g_1)$, and $(2.21.g_2)$ along with the definition of $\beta_{i,j}$:

$$\begin{aligned} b_3\alpha_{3,2}\beta_2' + b_4(\alpha_{4,2}\beta_2' + \alpha_{4,3}\beta_3) &= p_{4,2} \\ b_3\alpha_{3,2}\alpha_2^2 + b_4(\alpha_{4,2}\alpha_2^2 + \alpha_{4,3}\alpha_3^2) &= \tfrac{1}{12} \\ b_3\gamma_{3,2}\alpha_2^2 + b_4(\gamma_{4,2}\alpha_2^2 + \gamma_{4,3}\alpha_3^2) &= -\tfrac{1}{3}\gamma \\ \gamma_{3,2} + \alpha_{3,2} &= \beta_{3,2} \\ \gamma_{4,2} + \alpha_{4,2} &= \beta_{4,2} \\ \gamma_{4,3} + \alpha_{4,3} &= \beta_{4,3} \end{aligned}$$

$\gamma = 0.572816062482135$					
$\alpha_{2,1}$	=	1	$\gamma_{2,1}$	=	-1.91153192976055097824
$\alpha_{3,1}$	=	0.10845300169319391758	$\gamma_{3,1}$	=	0.32881824061153522156
$\alpha_{3,2}$	=	0.39154699830680608241	$\gamma_{3,2}$	=	0.0
$\alpha_{4,1}$	=	0.43453047756004477624	$\gamma_{4,1}$	=	0.03303644239795811290
$\alpha_{4,2}$	=	0.14484349252001492541	$\gamma_{4,2}$	=	-0.24375152376108235312
$\alpha_{4,3}$	=	-0.07937397008005970166	$\gamma_{4,3}$	=	-0.17062602991994029834
b_1	=	0.16666666666666666667	$\widehat{b}_1$	=	0.50269322573684235345
b_2	=	0.16666666666666666667	$\widehat{b}_2$	=	0.27867551969005856226
b_3	=	0.0	$\widehat{b}_3$	=	0.21863125457309908428
b_4	=	0.66666666666666666667	$\widehat{b}_4$	=	0.0

Table 2.1: Coefficients of ROK4a, a fourth order, L-stable, type 1 Rosenbrock-K method.

Finally $\alpha_{i,1}$ and $\beta_{i,1}$ follow immediately from the definition of α_i and β'_i respectively. The coefficient values for this method, named ROK4a, are given in Table 2.1.

The choice of γ ensures that for the main method $R(\infty) = 0$. The embedded method has $\widehat{R}(\infty) = -0.55$ Figure 2.4 shows the stability function values for both the main and embedded methods along the imaginary axis. We see that the absolute function values are below one, which implies that the main Rok4a method is L-stable, and the embedded method is strongly A-stable.

It is important to note that the stability results presented here apply to the case where a full Jacobian is used, and does not account for the impact of Krylov approximation. The impact of the Krylov approximation on the stability will be the subject of future work.

Exact stability requirements when making use of the Krylov approximation of the Jacobian are as yet undetermined, though a result by Wensch in [16] gives reason to believe that the size of the Krylov subspace must be as large as the number of stiff variables in the underlying problem.

2.4.2 ROK4b: a six stages, fourth order stiffly accurate Rosenbrock-K method

Stiff accuracy is a desirable property when solving very stiff systems or index-1 differential algebraic equations. A stiffly accurate Rosenbrock method [11, Section IV.4] is characterized by the property

$$b_i = \beta_{s,i}\,, \quad i = 1, \ldots, s\,.$$

We have derived a six-stage, stiffly accurate, fourth-order Rosenbrock-K method, named ROK4b. For brevity we do not show here the order conditions, nor we present the solution

$\gamma = 0.31$					
$\alpha_{2,1}$	=	1.0	$\gamma_{2,1}$	=	-22.824608269858540
$\alpha_{3,1}$	=	0.530633333333333	$\gamma_{3,1}$	=	-69.343635255712726
$\alpha_{3,2}$	=	-0.030633333333333	$\gamma_{3,2}$	=	-0.030633333333333
$\alpha_{4,1}$	=	0.894444444444444	$\gamma_{4,1}$	=	404.7106882480958
$\alpha_{4,2}$	=	0.055555555555556	$\gamma_{4,2}$	=	0.055555555555556
$\alpha_{4,3}$	=	0.05	$\gamma_{4,3}$	=	0.05
$\alpha_{5,1}$	=	0.738333333333333	$\gamma_{5,1}$	=	-0.571666666666667
$\alpha_{5,2}$	=	-0.121666666666667	$\gamma_{5,2}$	=	-0.121666666666667
$\alpha_{5,3}$	=	0.333333333333333	$\gamma_{5,3}$	=	0.333333333333333
$\alpha_{5,4}$	=	0.05	$\gamma_{5,4}$	=	0.05
$\alpha_{6,1}$	=	-0.096929102825711	$\gamma_{6,1}$	=	0.263595769492377
$\alpha_{6,2}$	=	-0.121666666666667	$\gamma_{6,2}$	=	-0.121666666666667
$\alpha_{6,3}$	=	1.045582889789120	$\gamma_{6,3}$	=	-0.378916223122453
$\alpha_{6,4}$	=	0.173012879703258	$\gamma_{6,4}$	=	-0.073012879703258
$\alpha_{6,5}$	=	0.0	$\gamma_{6,5}$	=	0
b_1	=	0.166666666666667	$\widehat{b}_1$	=	0.166666666666667
b_2	=	-0.243333333333333	$\widehat{b}_2$	=	-0.243333333333333
b_3	=	0.666666666666667	$\widehat{b}_3$	=	0.666666666666667
b_4	=	0.100000000000000	$\widehat{b}_4$	=	0.1
b_5	=	0.0	$\widehat{b}_5$	=	0.31
b_6	=	0.31	$\widehat{b}_6$	=	0.0

Table 2.2: Coefficients of ROK4b, a fourth order, stiffly accurate, type 1 Rosenbrock-K method.

method. The coefficients have been obtained through a process similar to that outlined in Section 2.4.3. The Rok4b method coefficients are shown in Table 2.2. ROK4b has the additional benefit of both the main and embedded methods are L-stable. Figure 2.4 shows the stability functions of the main and embedded methods of Rok4b evaluated along the imaginary axis.

2.4.3 ROK4p: a five stages, fourth order, parabolic Rosenbrock-K method

Due to their low stage order Rosenbrock methods can be marred by order reduction when solving initial value problems arising from the semi-discretization of PDEs. The following set of additional conditions guarantees the full order of convergence for Rosenbrock methods applied to semi-discrete *parabolic* PDEs [50, 41]:

$$b^T \beta^j \left(2\beta^2 \mathbb{1} - \alpha^2\right) = 0 \quad \text{for} \;\; p-2 \le j \le s-1 \;\; \text{and} \;\; p \ge 3\,. \tag{2.23}$$

Here $b = (b_i)_{i=1,\dots,s}$, $\alpha = (\alpha_i)_{i=1,\dots,s},$, $\beta = (\beta_i)_{i=1,\dots,s}$, p is the order of the method, and s is the number of stages. Multiplications are understood component-wise. We will call a Rosenbrock method *parabolic* if it satisfies (2.23).

The order conditions for a five-stage, fourth-order, parabolic Rosenbrock-K are:

$$\begin{array}{rlcl}
(a) & b_1 + b_2 + b_3 + b_4 + b_5 & = & 1 \\
(b) & b_2\beta_2' + b_3\beta_3' + b_4\beta_4' + b_5\beta_5' & = & p_{2,1}(\gamma) \\
(c) & b_2\alpha_2^2 + b_3\alpha_3^2 + b_4\alpha_4^2 + b_5\alpha_5^2 & = & \frac{1}{3} \\
(d) & b_3\beta_{3,2}\beta_2' + b_4(\beta_{4,2}\beta_2' + \beta_{4,3}\beta_3') & & \\
 & \quad + b_5(\beta_{5,2}\beta_2' + \beta_{5,3}\beta_3' + \beta_{5,4}\beta_4') & = & p_{3,2}(\gamma) \\
(e) & b_2\alpha_2^3 + b_3\alpha_3^3 + b_4\alpha_4^3 + b_5\alpha_5^3 & = & \frac{1}{4} \\
(f) & b_3\alpha_3\alpha_{3,2}\beta_2' + b_4\alpha_4(\alpha_{4,2}\beta_2' + \alpha_{4,3}\beta_3') & & \\
 & \quad + b_5\alpha_5(\alpha_{5,2}\beta_2' + \alpha_{5,3}\beta_3' + \alpha_{5,4}\beta_4') & = & p_{4,2}(\gamma) \\
(g_1) & b_3\alpha_{3,2}\alpha_2^2 + b_4(\alpha_{4,2}\alpha_2^2 + \alpha_{4,3}\alpha_3^2) & & \\
 & \quad + b_5(\alpha_{5,2}\alpha_2^2 + \alpha_{5,3}\alpha_3^2 + \alpha_{5,4}\alpha_4^2) & = & \frac{1}{12} \\
(g_2) & b_3\gamma_{3,2}\alpha_2^2 + b_4(\gamma_{4,2}\alpha_2^2 + \gamma_{4,3}\alpha_3^2) & & \\
 & \quad + b_5(\gamma_{5,2}\alpha_2^2 + \gamma_{5,3}\alpha_3^2 + \gamma_{5,4}\alpha_4^2) & = & -\frac{1}{3}\gamma \\
(h) & b_4\beta_{4,3}\beta_{3,2}\beta_2' + b_5(\beta_{5,3}\beta_{3,2}\beta_2' + \beta_{5,4}\beta_{4,2}\beta_2' + \beta_{5,4}\beta_{4,3}\beta_3' & = & p_{4,4}(\gamma) \\
(i) & 2b_5\beta_{5,4}\beta_{4,3}\beta_{3,2}\beta_2' - b_4\beta_{4,3}\beta_{3,2}\alpha_2^2 - b_5\beta_{5,3}\beta_{3,2}\alpha_2^2 & & \\
 & \quad - b_5\beta_{5,4}\beta_{4,2}\alpha_2^2 - b_5\beta_{5,4}\beta_{4,3}\alpha_3^2 & = & \pi_1(\gamma) \\
(j) & b_5\beta_{5,4}\beta_{4,3}\beta_{3,2}\alpha_2^2 & = & \pi_2(\gamma) \\
(k) & 0 & = & \pi_3(\gamma)
\end{array} \tag{2.24}$$

where the polynomials $p_{i,j}(\gamma)$ are defined in (2.21), and

$$\begin{aligned}
\pi_1(\gamma) &= 2\gamma p_{4,3} - 8\gamma p_{4,4}(\gamma) + \frac{1}{3}\gamma^2 - 12\gamma^2 p_{3,2}(\gamma) - 8\gamma^3 p_{2,1}(\gamma) - 2\gamma^4\,, \\
\pi_2(\gamma) &= 3\gamma\pi_1(\gamma) - 3\gamma^2 p_{4,3}(\gamma) + 20\gamma^2 p_{4,4}(\gamma) - \frac{1}{3}\gamma^3 + 20\gamma^3 p_{3,2}(\gamma) + 10\gamma^4 p_{2,1}(\gamma) + 2\gamma^5\,, \\
\pi_3(\gamma) &= -4\gamma\pi_2(\gamma) + 6\gamma^2\pi_1(\gamma) - 4\gamma^3 p_{4,3}(\gamma) + 40\gamma^3 p_{4,4}(\gamma) - \frac{1}{3}\gamma^4 + 30\gamma^4 p_{3,2}(\gamma) + 12\gamma^5 p_{2,1}(\gamma) + 2\gamma^6\,.
\end{aligned}$$

$\gamma = 0.572816062482135$					
$\alpha_{2,1}$	=	0.757900000000000	$\gamma_{2,1}$	=	-0.757900000000000
$\alpha_{3,1}$	=	0.170400000000000	$\gamma_{3,1}$	=	-0.295086678808293
$\alpha_{3,2}$	=	0.821100000000000	$\gamma_{3,2}$	=	0.178900000000000
$\alpha_{4,1}$	=	1.196218621274069	$\gamma_{4,1}$	=	-1.836333117783808
$\alpha_{4,2}$	=	0.297700000000000	$\gamma_{4,2}$	=	-0.247700000000000
$\alpha_{4,3}$	=	-1.433618621274069	$\gamma_{4,3}$	=	1.681409044712106
$\alpha_{5,1}$	=	-0.010650410785863	$\gamma_{5,1}$	=	-0.197089800872483
$\alpha_{5,2}$	=	0.142100000000000	$\gamma_{5,2}$	=	-0.684644029868020
$\alpha_{5,3}$	=	-0.129349589214137	$\gamma_{5,3}$	=	0.166330242942910
$\alpha_{5,4}$	=	0.392800000000000	$\gamma_{5,4}$	=	0.000000000000000
b_1	=	0.056000000000000	$\widehat{b}_1$	=	-0.186875355621256
b_2	=	0.116601238130482	$\widehat{b}_2$	=	-0.250433793031115
b_3	=	0.160300000000000	$\widehat{b}_3$	=	0.326360736478684
b_4	=	-0.031109354304222	$\widehat{b}_4$	=	0.110948412173687
b_5	=	0.698208116173739	$\widehat{b}_5$	=	1.000000000000000

Table 2.3: Coefficients of ROK4p, a fourth order, parabolic, type 1 Rosenbrock-K method.

The approach to solve the system of equations (2.24) is similar to that used for (2.21). A sequence of linear systems is constructed, and for each system arbitrary choices are made for the values of some parameters. A numerical genetic optimization algorithm is employed to select free parameter values which lead to method coefficients of acceptable magnitudes. The coefficients of the resulting method, named ROK4p, are given in Table 2.3.

The choice of γ ensures that for the main method $R(\infty) = 0$. The embedded method has $\widehat{R}(\infty) = 0.24$. Figure 2.4 shows the stability function values for both the main and embedded methods along the imaginary axis. We see that the absolute function values are below one, which implies that the main ROK4p method is L-stable, and the embedded method is strongly A-stable.

2.5 Numerical Results

Here we present some results from numerical experiments verifying the properties of the methods discussed above, as well as comparing performance of Rosenbrock-Krylov methods with several standard classical Rosenbrock and Rosenbrock-W methods. RANG3 is a third order Rosenbrock-W method [12], RODAS4 is a fourth order, stiffly accurate classical Rosenbrock method [11, Section IV.10], and ROS4 is a fourth or10.0der, L-stable, classical Rosenbrock method [11, Section IV.10].

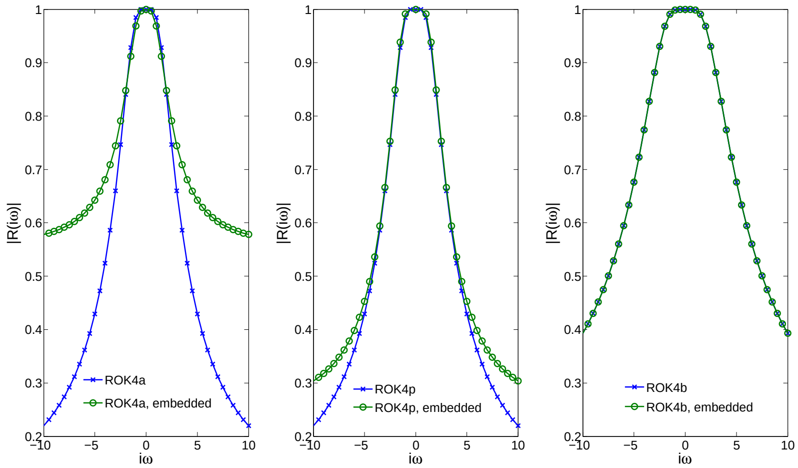


Figure 2.4: Stability functions for both the main and embedded methods of Rok4a, Rok4p, and Rok4b

2.5.1 Lorenz 96

The nonlinear test is carried out with the Lorenz-96 model [51]. This chaotic model has $N = 40$ states, periodic boundary conditions, and is described by the following equations:

$$\begin{aligned} \frac{dy_j}{dt} &= -y_{j-1}\ (y_{j-2} - y_{j+1}) - y_j + F\ , \quad j = 1, \ldots, N\ , \\ y_{-1} &= y_{N-1}\ , \quad y_0 = y_N\ , \quad y_{N+1} = y_1\ . \end{aligned} \tag{2.25}$$

The forcing term is $F = 8.0$, with $t \in [0,\ 0.3]$.

	Rang3	Ros4	Rodas4	Rok4a	Rok4p	Rok4b
M = N	2.99	4.01	3.99	4.01	3.99	3.99
M = 4	2.99	3.03	3.05	4.01	3.98	3.99

Table 2.4: Convergence Rates on Lorenz 96

Table 2.4 shows the convergence orders of all methods applied to the Lorenz-96 system, using both the full Jacobian as well as a four dimensional Krylov approximation of the Jacobian. Figure 2.5 verifies numerically the theoretical order results for all methods using the full Jacobian.

Recall that all methods satisfying the classical Rosenbrock order conditions are also Rosenbrock-K methods of at least order three. Table 2.4 shows this property, where the third order method Rang3 maintains its order and both fourth order methods, Ros4 and Rodas4, reduce to third order while using the approximate Jacobian.

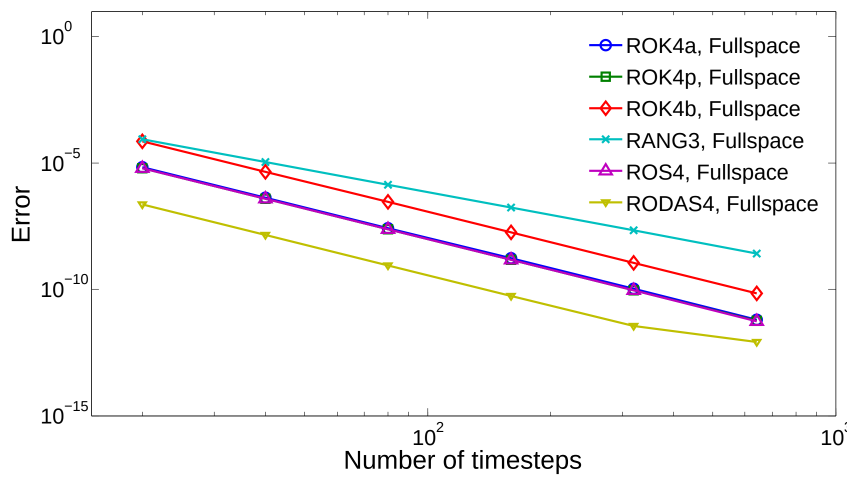


Figure 2.5: Precision diagram for Lorenz 96, showing convergence order of methods using a full Jacobian

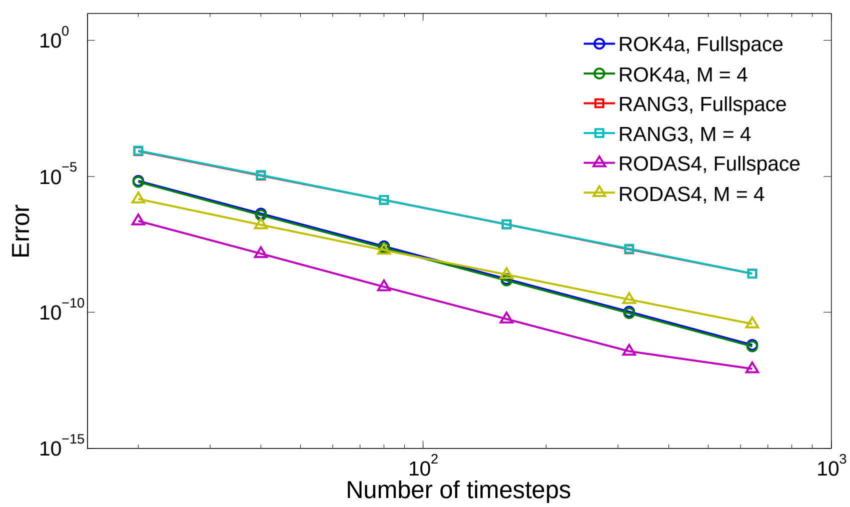


Figure 2.6: Precision diagram showing the convergence order of ROK4a, RANG3, and RODAS4 using the full and Krylov approximated Jacobian.

2.5.2 Dissipative Burger's Equation

We apply the newly derived methods to an ODE system coming from a semi-discretization of a partial differential equation using the method of lines. The dissipative Burger's equation is a one-dimensional PDE described by

$$\frac{du}{dt} + \frac{d}{dx}\left(\frac{1}{2}u^2\right) = \varepsilon \frac{d^2u}{dx^2}, \quad x \in [0, 10], \quad t \in [0,\ 0.5], \quad \varepsilon = .001\,, \tag{2.26}$$

with homogeneous boundary conditions, and initial condition

$$u(x, t = 0) = \frac{1}{6}\sin^2\left(\frac{1}{5}\pi x\right)\left(1 - x^2\right), \quad \varepsilon = .001$$

The spatial discretization is a Nodal Discontinuous Galerkin method using equispaced fourth-order elements, making use of the code base provided for [52].

Figures 2.7 and 2.8 show a performance comparison of the newly proposed methods with both Ros4 and Rodas4. The figures show that for problems of even modest size, Rosenbrock-Krylov methods have comparable efficiency with previously existing methods. The increase in relative efficiency between Rosenbrock-K methods and the classical Rosenbrock methods as the problem size increases is a good indicator that Rosenbrock-K methods are likely to be much more efficient than fullspace methods as problem size increases, and the benefits of solving a reduced system become more pronounced.

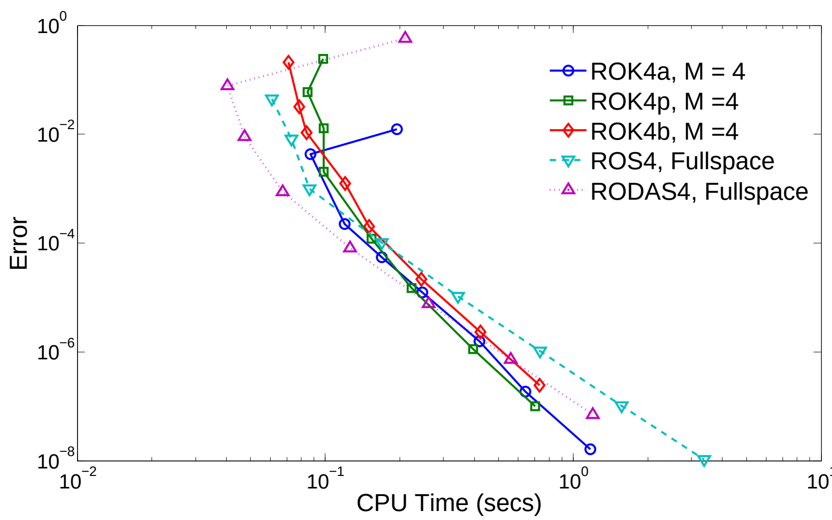


Figure 2.7: Precision Diagram for Burger's Equation using 10 fourth order elements.

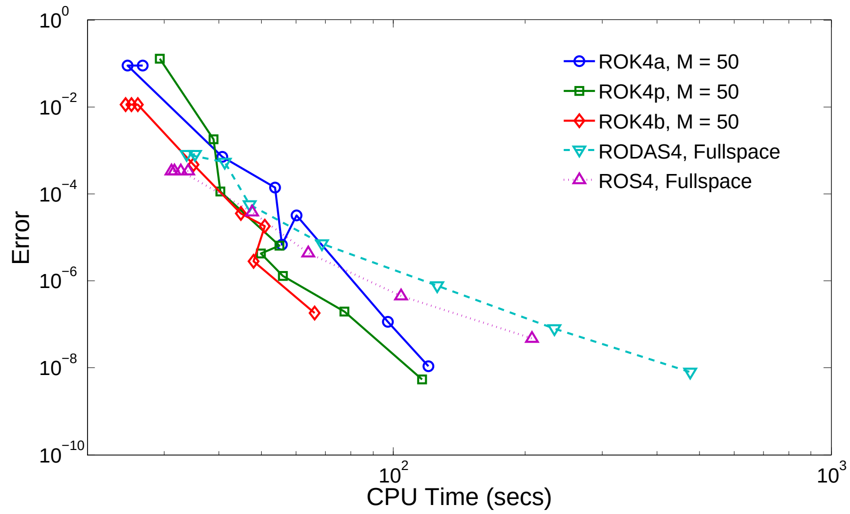


Figure 2.8: Precision Diagram for Burger's Equation using 400 fourth order elements.

2.5.3 CBM-IV

Here we give some results for ROK methods applied to a stiff system of ODEs coming from a KPP MATLAB implementation of the CBM-IV model [53]. This problem is based on the Carbon Bond Mechanism IV (CBM-IV), consisting of 32 chemical species involved in 70 thermal and 11 photolytic reactions [54].

While CBM-IV is a perfect example of a problem for which Rosenbrock-K methods are a poor choice, due to its small size and relatively large number of stiff variables, it does allow us to illustrate numerically the relationship between stability and choice of Krylov basis size. Figure 2.9 shows the number of timesteps, normalized to a fullspace solution of the respective method, required to obtain a reasonable solution in a single day simulation of the CBM-IV model. The number of timesteps required for a fullspace solution are given in Table 2.5.

ROK4a	ROK4p	ROK4b
1301	10270	261

Table 2.5: Number of required timesteps in a fullspace solution with tolerances of 10^{-2}.

We see from this figure that for small Krylov basis sizes, Rosenbrock-K methods are unstable. However, as the size of the Krylov space nears the size of the fullspace the behavior of the Rosenbrock-K methods approaches that of the fullspace method.

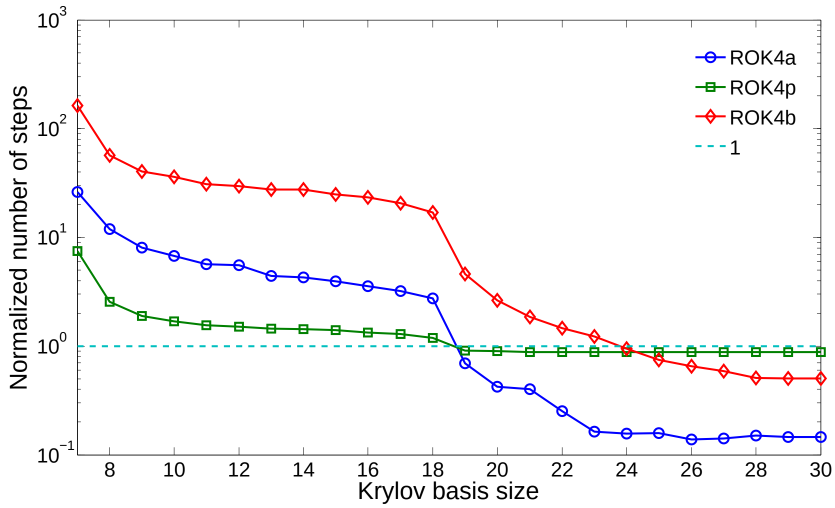


Figure 2.9: Number of accepted steps in a fullspace solution with tolerances of 10^{-2} for CBM-IV.

2.5.4 Shallow water equations

We examine relative performance of the methods on the shallow water equations [55].

$$\frac{\partial}{\partial t}h + \frac{\partial}{\partial x}(uh) + \frac{\partial}{\partial y}(vh) = 0 \tag{2.27a}$$

$$\frac{\partial}{\partial t}(uh) + \frac{\partial}{\partial x}\left(u^2h + \frac{1}{2}gh^2\right) + \frac{\partial}{\partial y}(uvh) = 0 \tag{2.27b}$$

$$\frac{\partial}{\partial t}(vh) + \frac{\partial}{\partial x}(uvh) + \frac{\partial}{\partial y}\left(v^2h + \frac{1}{2}gh^2\right) = 0. \tag{2.27c}$$

with state variables

$$u = u(x, y, t), \quad v = v(x, y, t), \quad h = h(x, y, t).$$

The spatial discretization is a centered finite difference, using a 32×32 grid so that after transformation to the standard ODE form in equation (2.1), we have that

$$y = [u\ v\ h]^T \in \mathbb{R}^N, \quad f_y(t, y) = \mathbf{J} \in \mathbb{R}^{N\times N}, N = 3072.$$

Figure 2.10 gives an efficiency comparsion of the Rosenbrock-K and classical Rosenbrock methods, all methods make use of a sparse Jacobian matrix. This problem illustrates the scalability of the Rosenbrock-K methods, when the stiff subspace of the problem is kept relatively small. Here there are 3072 state variables, but the Rosenbrock-K methods require

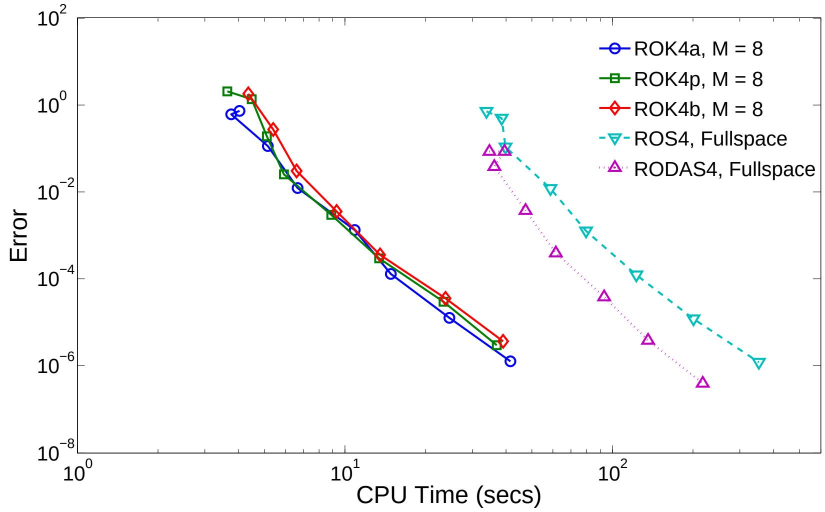


Figure 2.10: Precision diagram for the shallow water equations.

only eight basis vectors for stability, and so the cost of computing the Krylov space and solving the small system is much cheaper than solving the linear system in the full space.

With the small basis size requirements in mind we explore in Figure 2.11 the relative difference in cost, measured by the number of right hand side evaluations, between an explicit Runge-Kutta method and matrix-free Rosenbrock-K methods. Figure 2.11 shows the number of function evaluations on the x-axis and the Error of the resulting solution on the y-axis. The number of function evaluations for ROK4a includes those required to compute the matrix-free Jacobian-vector products in the Arnoldi iteration. For the shallow water equations we see that Rosenbrock-K methods perform well against the explicit Runge-Kutta method, when low accuracy is desired and the CFL condition begins to constrain the explicit method.

2.6 Conclusions and future work

In this work we have developed a new class of Rosenbrock like integrators, along with a corresponding order condition theory. We consider the ODE integrator and linear solver as a single computational process to develop methods with the least possible amount of implicitness.

The Rosenbrock-K order conditions remove the requirement for accurate solution of the linear systems which constrain the use of approximate methods in classical Rosenbrock integrators. For accuracy of the integration process, the size of the Krylov approximation of the Jacobian need be only as large as the desired order of the method. Stability considerations give

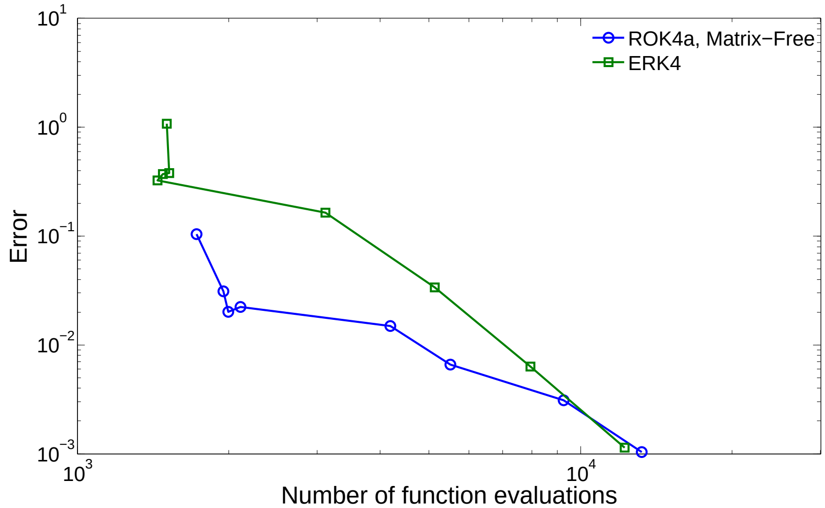


Figure 2.11: Comparison of ROKa and ERK4 on the shallow water equations.

stricter requirements on the size of the Krylov basis used, though the exact nature of these requirements is not yet entirely understood and will be the focus of future work. Some numerical investigation, and a result by Wensch [16], give reason to believe that the required size of the Krylov subspace is related to the number of stiff variables in the underlying problem.

The Rosenbrock-K methods devloped here have many favorable properties over similar integrators. Rosenbrock-K methods have substantially fewer order conditions than Rosenbrock-W methods, requiring only a single extra order condition for order four methods as opposed to four extra conditions for order three methods in the case of Rosenbrock-W. The reduced number of order conditions allows for methods of higher order, we have given conditions up to order five, or for methods with fewer stages. Further, the structure of Rosenbrock-K methods allows for the computation of a single Krylov subspace at each timestep without the requirement of enriching this space for each internal stage, as is the case for Krylov-ROW methods.

The efficiency of Rosenbrock-K integrators applied to a specific problem is dependent on the stability requirements, and so the stiffness of the underlying problem. For this reason Rosenbrock-K methods are best suited to very large problems in which there is a relatively small number of stiff variables, though they are expected to perform at least as well as Krylov-ROW methods in all cases.

Acknowledgements

This work has been supported in part by NSF through awards NSF OCI–8670904397, NSF CCF–0916493, NSF DMS–0915047, NSF CMMI–1130667, NSF CCF–1218454, AFOSR FA9550–12–1–0293–DEF, AFOSR 12-2640-06, and by the Computational Science Laboratory at Virginia Tech.

Chapter 3

Exponential-Krylov Methods for Ordinary Differential Equations

3.1 Introduction

Many methods exist to numerically approximate the solution of initial value problems

$$\frac{dy}{dt} = f(t, y), \quad t_0 \leq t \leq t_F, \quad y(t_0) = y_n; \quad y(t), f(t, y) \in \mathbb{R}^N. \tag{3.1}$$

Multistep methods make use of the solution at several previous timesteps to compute the solution at t_{n+1}, while Runge-Kutta methods interpolate the solution at several points between the current solution, t_n, and the future solution, t_{n+1}. In both cases implicit methods require the solution of (non-)linear system of equations at each time step. Much work has been done towards the acceleration of the solutions to these systems. Iterative Krylov-based linear algebra solvers are the typical choice for large-scale applications (3.1). The generalized minimal residual (GMRES) method [13] is the standard approach for constructing efficient solutions to linear systems arising throughout the integration of ODEs. Jacobian-free Newton-Krylov (JFNK) methods [18, 19] make use of a GMRES like solver within a Newton iteration to solve the nonlinear equations arising from Runge-Kutta and multistep methods.

Rosenbrock methods [11], a class of integrators coming from a linearization of Runge-Kutta methods, require only the solution of a linear system at each stage and are characterized by the explicit appearance of the Jacobian matrix

$$\mathbf{J}_n = \left. \frac{\partial f}{\partial y} \right|_{t=t_n, y=y_n}$$

in the method itself. Due to the approximate nature of solutions coming from iterative methods, the explicit appearance of $\mathbf{J}_n$ causes order reduction unless the system solution is very accurate. For this reason Rosenbrock-W methods [11, 12, 41], an extension of Rosenbrock methods allowing for arbitrary approximations of the matrix $\mathbf{J}_n$, have been developed.

Krylov-ROW methods [15, 36, 38, 37] couple Rosenbrock methods with Krylov based solvers for the linear systems arising therein. A multiple Arnoldi process is used to enrich the Krylov space at each stage, and the order of the underlying Rosenbrock method is preserved with modest requirements on the Krylov space size, independent of the dimension of the ODE system under consideration.

The authors have recently developed Rosenbrock-K methods [56] to pursue a similar goal. Krylov-ROW methods ensure the order results with standard Rosenbrock-W discretizations by adding requirements to the underlying Krylov space. In contradistinction, Rosenbrock-K methods guarantee the accuracy order through the use of a specific Krylov-based approximation of the Jacobian and the construction of new order conditions which take this approximation into account. Rosenbrock-K methods have substantially fewer order conditions than Rosenbrock-W methods allowing for the construction of schemes of higher order with fewer stages. More importantly, Rosenbrock-K methods give a strict lower bound on the number of Krylov basis vectors required for accuracy that depends only on the order of the method, and is completely independent of the dimension of the ODE system under consideration.

Exponential integrators [46, 57, 45] replace the need to construct solutions to a linear system, or equivalently approximate the rational matrix function times vector product $\left(\mathbf{I}_N - hA\right)^{-1} v$, with the similar, hopefully cheaper, requirement to approximate the exponential matrix times vector product $\exp(hA)v$. Like the solution of large linear systems, approximations of the matrix exponential times vectors are typically obtained using Krylov based methods.

In this paper we extend the ideas of the Rosenbrock-K methods presented in [56] to the particular set of exponential integrators discussed in Hochbruck, Lubich, and Selhofer [46] and introduce the new family of exponential-Krylov (exponential-K) methods. The new schemes require the construction of only a single Kyrlov basis at each timestep, as opposed to each stage in the case of standard exponential methods. Moreover, the required dimension of the subspace to guarantee the desired order of accuracy is independent of the system (3.1) under consideration.

The remainder of the paper is organized as follows. Section 3.2 presents the exponential-K framework and the Krylov approximation of the Jacobian used. Section 3.3 develops the order condition theory for the new exponential-K methods using both Butcher trees and B-series. Section 3.4 constructs a practical four stage, fourth order exponential-K method. Section 3.5 discusses alternative implementations of existing exponential methods, and Section 3.6 presents numerical results. Conclusions are drawn in Section 3.7.

3.2 Formulation of exponential-Krylov methods

3.2.1 Exponential-W integrators

The starting point of our investigation is following class of exponential-W integrators proposed in [46]

$$\begin{aligned}
k_i &= \varphi(h\gamma\mathbf{A}_n)\left(h\,F_i + h\mathbf{A}_n\sum_{j=1}^{i-1}\gamma_{i,j}k_j\right),\\
F_i &= f\left(y_n + \sum_{j=1}^{i-1}\alpha_{i,j}k_j\right),\\
y_{n+1} &= y_n + \sum_{j=1}^{s} b_i k_i,
\end{aligned} \tag{3.2}$$

where $\mathbf{A}_n$ is either the matrix $\mathbf{J}_n$ or an approximation of it. Equation (3.2) formalizes explicit Runge-Kutta methods when $\varphi(z) = 1$, Rosenbrock methods when $\varphi(z) = 1/(1-z)$, and exponential methods when $\varphi(z) = (e^z - 1)/z$. Note that similar to the Rosenbrock methods discussed before, equation (3.2) makes explicit use of the matrix $\mathbf{J}_n$, and so it is natural to explore conditions allowing for arbitrary approximations as in the case of Rosenbrock-W methods. A discussion of these methods and their order conditions is given in [46].

3.2.2 Exponential-K integrators

The new exponential-K methods proposed in this work have the same general form as exponential-W methods (3.2), but use a specific, Krylov based-approximation $\mathbf{A}_n$ of the Jacobian. To begin we construct the M-dimensional Krylov space $\mathcal{K}_M$ where $M \ll N$ and

$$\begin{aligned}
\mathcal{K}_M &= \operatorname{span}\left\{f_n, \mathbf{J}_n f_n, \mathbf{J}_n^2 f_n, \dots, \mathbf{J}_n^{M-1} f_n\right\}\\
&= \operatorname{span}\left\{v_1, v_2, \dots, v_M\right\}
\end{aligned} \tag{3.3}$$

using a modified Arnoldi iteration [14]. The Arnoldi iteration returns the matrix

$$\mathbf{V} = [v_1, v_2, \dots, v_M] \in \mathbb{R}^{N\times M}$$

whose columns form an orthonormal basis of $\mathcal{K}_M$, and the upper Hessenberg matrix

$$\mathbf{H} = \mathbf{V}^T\,\mathbf{J}_n\,\mathbf{V} \in \mathbb{R}^{M\times M}. \tag{3.4}$$

From these two matrices we construct the following Krylov-based approximation of the Jacobian

$$\mathbf{A}_n = \mathbf{V}\,\mathbf{H}\,\mathbf{V}^T = \mathbf{V}\,\mathbf{V}^T\,\mathbf{J}_n\,\mathbf{V}\,\mathbf{V}^T. \tag{3.5}$$

The powers of this matrix have the following property.

Lemma 3 (Powers of $\mathbf{A}_n$)**.** *For any* $k = 0, 1, 2, \dots$

$$\mathbf{A}_n^k = \mathbf{V}\,\mathbf{H}^k\,\mathbf{V}^T.$$

Proof. We give the proof of the Lemma by induction. As the base case we have that

$$\mathbf{A}_n^1 = \mathbf{V}\,\mathbf{H}\,\mathbf{V}^T = \mathbf{V}\,\mathbf{H}^1\,\mathbf{V}^T$$

next we assume that $\mathbf{A}_n^{k-1} = \mathbf{V}\,\mathbf{H}^{k-1}\,\mathbf{V}^T$ and show that $\mathbf{A}_n^k = \mathbf{V}\,\mathbf{H}^k\,\mathbf{V}^T$.

$$\mathbf{A}_n^k = \mathbf{A}_n\mathbf{A}_n^{k-1} = \mathbf{V}\,\mathbf{H}\,\mathbf{V}^T\left(\mathbf{V}\,\mathbf{H}^{k-1}\,\mathbf{V}^T\right) = \mathbf{V}\,\mathbf{H}\,\left(\mathbf{V}^T\,\mathbf{V}\right)\mathbf{H}^{k-1}\,\mathbf{V}^T$$

because V is an orthonormal matrix $\mathbf{V}^T\,\mathbf{V} = \mathbf{I}_N$, and so

$$\mathbf{A}_n^k = \mathbf{V}\,\mathbf{H}^k\,\mathbf{V}^T$$

□

The construction of exponential integrators uses matrix functions of the form $\varphi_k(h\gamma\mathbf{A}_n)$, where the functions are defined by

$$\varphi_k(z) = \int_0^1 e^{z(1-\theta)}\,\frac{\theta^{k-1}}{(k-1)!}\,d\theta, \quad k = 0, 1, 2, \dots$$

and satisfy the recurrence relation

$$\varphi_{k+1}(z) = \frac{\varphi_k(z) - 1/k!}{z}, \quad \varphi_k(0) = \frac{1}{k!}.$$

The matrix functions have the following property.

Lemma 4 (Matrix functions of the approximate Jacobian)**.**

$$\varphi_k(h\gamma\mathbf{A}_n) = \frac{1}{k!}(\mathbf{I}_N - \mathbf{V}\,\mathbf{V}^T) + \mathbf{V}\,\varphi_k(h\gamma\mathbf{H})\,\mathbf{V}^T \quad k = 1, 2, \dots$$

Proof. It is possible to expand $\varphi_k(z)$ as a Taylor series [58]:

$$\varphi_k(z) = \sum_{i=0}^{\infty} c_i\,\frac{z^i}{(k+i)!} \tag{3.6}$$

We have that

$$\varphi_k(h\gamma\mathbf{A}_n) = \frac{1}{k!}\mathbf{I}_N + \sum_{i=1}^{\infty}\frac{(h\gamma)^i}{(k+i)!}\mathbf{A}_n^i$$

after applying Lemma 3 we obtain

$$\varphi_k(h\gamma\mathbf{A}_n) = \frac{1}{k!}\mathbf{I}_N + \sum_{i=1}^{\infty}\frac{(h\gamma)^i}{(k+i)!}\mathbf{V}\mathbf{H}^i\mathbf{V}^T.$$

Similarly we can expand $\mathbf{V}\varphi_k(h\gamma H)\mathbf{V}^T$ as

$$\mathbf{V}\varphi_k(h\gamma\mathbf{H})\mathbf{V}^T = \frac{1}{k!}\mathbf{V}\mathbf{V}^T + \sum_{i=1}^{\infty}\frac{(h\gamma)^i}{(k+i)!}\mathbf{V}\mathbf{H}^i\mathbf{V}^T,$$

taking the difference we see that

$$\varphi_k(h\gamma\mathbf{A}_n) - \mathbf{V}\varphi_k(h\gamma\mathbf{H})\mathbf{V}^T = \frac{1}{k!}\left(\mathbf{I}_N - \mathbf{V}\mathbf{V}^T\right).$$

Finally we move $\mathbf{V}\varphi_k(h\gamma H)\mathbf{V}^T$ across the equality to obtain

$$\varphi_k(h\gamma\mathbf{A}_n) = \frac{1}{k!}\left(\mathbf{I}_N - \mathbf{V}\mathbf{V}^T\right) + \mathbf{V}\varphi_k(h\gamma\mathbf{H})\mathbf{V}^T \tag{3.7}$$

□

To finish the derivation of a reduced form for the exponential-Krylov integrator (3.2) we introduce the following notation

$$k_i = \underbrace{\mathbf{V}\lambda_i}_{\in\mathcal{K}_M} + \underbrace{\mu_i}_{\in\mathcal{K}_M^{\perp}}, \quad F_i = \underbrace{\mathbf{V}\psi_i}_{\in\mathcal{K}_M} + \underbrace{\delta_i}_{\in\mathcal{K}_M^{\perp}}$$

where $\mathbf{V}\lambda_i$ and $\mathbf{V}\psi_i$ represent the components of k_i and F_i which reside in the Krylov subspace $\mathcal{K}_M$, and similarly μ_i and δ_i are the components residing in the space orthogonal to $\mathcal{K}_M$. Insert equation (3.7) and the split forms of k_i and F_i into the general method formulation (3.2) to obtain

$$\mathbf{V}\,\lambda_i + \mu_i = \mathbf{V}\left(\varphi(h\gamma\mathbf{H})\psi_i + h\mathbf{H}\sum_{j=1}^{i-1}\gamma_{i,j}\lambda_j\right) + \delta_i\,.$$

This leads to the following equation for the reduced stage vector

$$\lambda_i = \varphi(h\gamma\mathbf{H})\left(\psi_i + h\mathbf{H}\sum_{j=1}^{i-1}\gamma_{i,j}\lambda_j\right). \tag{3.8}$$

The full stage values can be recovered as

$$k_i = \mathbf{V}\,\lambda_i + (F_i - \mathbf{V}\psi_i). \tag{3.9}$$

A single step of an autonomous exponential-K method is given in Algorithm 4.

Algorithm 4 One step of an autonomous exponential-K integrator

Compute $\mathbf{H}$ and $\mathbf{V}$ using the N-dimensional Arnoldi process [14]

for $i = 1, \ldots, s$ **do** ▷ For each stage, in succession

$$\begin{aligned} F_i &= f\left(y_n + \sum_{j=1}^{i-1} \alpha_{i,j} k_j\right) \\ \psi_i &= \mathbf{V}^T F_i \\ \lambda_i &= \varphi\left(h\gamma\mathbf{H}\right)\left(h\psi_i + h\mathbf{H}\sum_{j=1}^{i-1}\gamma_{i,j}\lambda_j\right) \\ k_i &= \mathbf{V}\,\lambda_i + h\left(F_i - \mathbf{V}\,\psi_i\right) \end{aligned}$$

end for

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

Remark 4. *Because the matrix* $\mathbf{H}$ *has dimension* $M \times M$ *direct methods can be used to compute the matrix function* $\varphi(h\gamma\mathbf{H})$. *In the case of Rosenbrock methods where* $\varphi(z) = 1/(1-z)$ *a direct LU-decomposition can be used. For exponential methods where* $\varphi(z) = (e^z - 1)/z$ *a Pade approximation [59] can be utilized. Furthermore, a single matrix function needs to be evaluated at each step when the matrices* $\mathbf{H}$ *and* $\mathbf{V}$ *are constructed.*

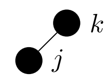
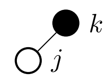
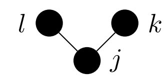
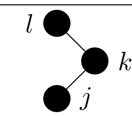
Remark 5. *We have only given here the autonomous form of an exponential-K method. A non-autonomous form is possible by constructing an extended ODE system and the corresponding Jacobian. This construction follows closely the treatment given in [56], where an* $N+1$ *dimensional Arnoldi iteration is discussed.*

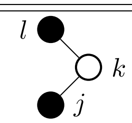
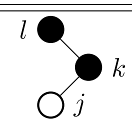
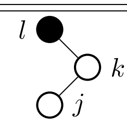
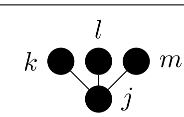
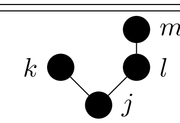
3.3 Order conditions for exponential-K methods

We construct classical order conditions for exponential-K methods. To this end we match the Taylor series expansion of the numerical and exact solutions up to a specified order. Butcher-trees[48] are an established method of representing terms in the Taylor series expansions of Runge-Kutta like methods. The derivation of order conditions for K-methods is an extension of the framework developed for W-methods. The theory for W methods is constructed using TW-trees, a subclass of P-trees, which are themselves an extension of T-trees that allow for two different colored nodes.

$$TW = \left\{ \begin{array}{ll} P\text{-trees:} & \text{end vertices are meagre, and} \\ & \text{fat vertices are singly branched} \end{array} \right\}$$

In the context of TW(and TK)-trees a meagre, or solid, node represents an appearance of the exact Jacobian matrix $\mathbf{J}_n$, while a fat, or empty, node represents the appearance of the approximate Jacobian matrix $\mathbf{A}_n$. Each tree represents a single elementary differential in the Taylor series of either the exact or numerical solutions of the ODE.

τ					
$F(\tau)$	f^J	$f^J_K f^K$	$\mathbf{A}_{JK} f^K$	$f^J_{KL} f^K f^L$	$f^J_K f^K_L f^L$
$\mathsf{a}(\tau)$	x_1	x_2	x_3	x_4	x_5
$B^{\#}\left(hf(B(\mathsf{a},y))\right)$	1	x_1	0	x_1^2	x_2
$B^{\#}\left(h\mathbf{A}B(\mathsf{a},y)\right)$	0	0	x_1	0	0
$B^{\#}\left(\varphi(h\gamma\mathbf{A})B(\mathsf{a},y)\right)$	x_1	x_2	$x_3 + c_1 x_1$	x_4	x_5

τ					
$F(\tau)$	$f^J_K \mathbf{A}_{KL} f^L$	$\mathbf{A}_{JK} f^K_L f^L$	$\mathbf{A}_{JK}\mathbf{A}_{KL} f^L$	$f^J_{KLM} f^K f^L f^M$	$f^J_{KL} f^L_M f^M f^K$
$\mathsf{a}(\tau)$	x_6	x_7	x_8	x_9	x_{10}
$B^{\#}\left(hf(B(\mathsf{a},y))\right)$	x_3	0	0	x_1^3	$x_1 x_2$
$B^{\#}\left(h\mathbf{A}B(\mathsf{a},y)\right)$	0	x_2	x_3	0	0
$B^{\#}\left(\varphi(h\gamma\mathbf{A})B(\mathsf{a},y)\right)$	x_6	$x_7 + c_1 x_2$	$x_8 + c_1 x_3 + c_2 x_1$	x_9	x_{10}

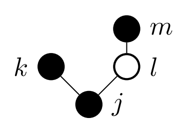
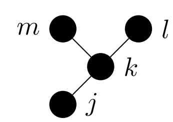
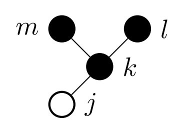
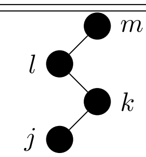
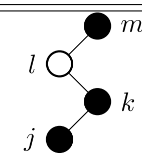
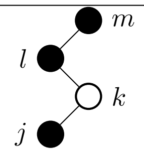
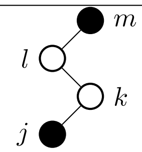
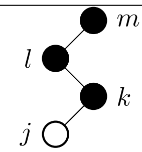
τ					
$F(\tau)$	$f^J_{KL}\mathbf{A}_{LM} f^M f^K$	$f^J_K f^K_{LM} f^M f^L$	$\mathbf{A}_{JK} f^K_{LM} f^L f^M$	$f^J_K f^K_L f^L_M f^M$	$f^J_K f^K_L \mathbf{A}_{LM} f^M$
$\mathsf{a}(\tau)$	x_{11}	x_{12}	x_{13}	x_{14}	x_{15}
$hf(B(\mathsf{a},y))$	$x_1 x_3$	x_4	0	x_5	x_6
$B^{\#}\left(h\mathbf{A}B(\mathsf{a},y)\right)$	0	0	x_4	0	0
$B^{\#}\left(\varphi(h\gamma\mathbf{A})B(\mathsf{a},y)\right)$	x_{11}	x_{12}	$x_{13} + c_1 x_4$	x_{14}	x_{15}

Figure 3.1: TW-trees up to order four (part one of two).

A fundamental component of our derivation of order conditions for exponential-K methods are B-series, a way of representing an expansion in trees, or elementary differentials, as a

τ			
$F(\tau)$	$f^J_K \mathbf{A}_{KL} f^L_M f^M$	$f^J_K \mathbf{A}_{KL} \mathbf{A}_{LM} f^M$	$\mathbf{A}_{JK} f^K_L f^L_M f^M$
$\mathsf{a}(\tau)$	x_{16}	x_{17}	x_{18}
$B^{\#}\left(hf(B(\mathsf{a},y))\right)$	x_7	x_8	0
$B^{\#}\left(h\mathbf{A}B(\mathsf{a},y)\right)$	0	0	x_5
$B^{\#}\left(\varphi(h\gamma\mathbf{A})B(\mathsf{a},y)\right)$	x_{16}	x_{17}	$x_{18} + c_1 x_5$

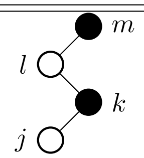
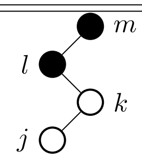
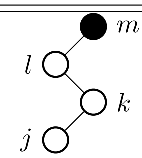
τ			
$F(\tau)$	$\mathbf{A}_{JK} f^K_L \mathbf{A}_{LM} f^M$	$\mathbf{A}_{JK} \mathbf{A}_{KL} f^L_M f^M$	$\mathbf{A}_{JK} \mathbf{A}_{KL} \mathbf{A}_{LM} f^M$
$\mathsf{a}(\tau)$	x_{19}	x_{20}	x_{21}
$B^{\#}\left(hf(B(\mathsf{a},y))\right)$	0	0	0
$B^{\#}\left(h\mathbf{A}B(\mathsf{a},y)\right)$	x_6	x_7	x_8
$B^{\#}\left(\varphi(h\gamma\mathbf{A})B(\mathsf{a},y)\right)$	$x_{19} + c_1 x_6$	$x_{20} + c_1 x_7 + c_2 x_2$	$x_{21} + c_1 x_8 + c_2 x_3 + c_3 x_1$

Figure 3.2: TW-trees up to order four (part two of two).

sequence of real numbers. A mapping $\mathsf{a} : TW \cup \{\emptyset\} \to \mathbb{R}$ represents the series

$$B(\mathsf{a}, y) = \mathsf{a}(\emptyset) y + \sum_{\tau \in TW} a(\tau) \frac{h^{|\tau|}}{\sigma(\tau)} F(\tau)(y).$$

Here τ are TW-trees; the order $|\tau|$, and the symmetry $\sigma(\tau)$ of a tree are defined in the same way as for single colored trees [48, 57], and $F(\tau)$ is the elementary differential belonging to tree τ as in figure 3.1. Similarly we introduce the operator $B^{\#}(f)$, which takes as input a function (which can be represented by a series) and returns the B-series coefficients

$$B^{\#}\left(B(\mathsf{a}, y)\right) = \mathsf{a}.$$

Figure 3.1 shows all TW-trees to order four, the coefficients of a generic B-series $B(\mathsf{a}, y)$, the result of composing $B(\mathsf{a}, y)$ with the function $f(y)$, the result of a multiplication of $B(\mathsf{a}, y)$ with the Krylov approximation matrix $\mathbf{A}_n$, and the result of a multiplication of $B(\mathsf{a}, y)$ by $\varphi(h\gamma\mathbf{A}_n)$. The full details of the composition of B-series can be found in [60], while details of products with the Jacobian and φ-functions can be found in [61].

Because K-methods are an extension of W-methods we first construct order conditions for the W-methods, then prove two lemmas that allow us to obtain the specific exponential-K order conditions. We follow a similar derivation procedure to that outlined in [62]. Throughout the derivation we track the progress of several truncated B-series which include all terms up to order four. These series have 21 terms corresponding to the TW-trees shown in Figure 3.1.

We begin the construction of order conditions for the W-method with a truncated B-series for y_n

$$B^{\#}\left(y(t_0)\right) = \mathsf{a}_0 = \left\{\mathsf{a}_0(\emptyset) = 1, x_i = 0 \ \ \forall i = 1, \ldots, 21\right\},$$

and then progress through individual stages of the W-method in equation (3.2), making use of the formulas from Figure 3.1 to construct the resultant B-series of the composition and multiplication operations. Algorithm 5 gives a method of constructing the B-series of the numerical solution y_n that approximates the exact solution $y(t_0 + h)$. Note that the sum of two B-series is another B-series with coefficients equal to the sum of individual coefficients of the series being combined. Similarly, the product of a B-series with a scalar is a new series with each coefficient multiplied by the scalar.

Algorithm 5 Construction of B-series of the numerical solution of a fourth-order, s stage exponential-W-method (3.2)

for $i = 1, \ldots, s$ **do**
  $\mathsf{u} = \mathsf{a}_0$
  for $j = 1, \ldots, i-1$ **do**
    $\mathsf{u} = \mathsf{u} + \alpha_{i,j} \cdot \mathsf{k}_i$
  end for
  $\mathsf{q} = B^{\#}\left(hf\left(B(\mathsf{u}, y)\right)\right)$
  for $j = 1, \ldots, i-1$ **do**
    $\mathsf{q} = \mathsf{q} + \gamma_{i,j} \cdot B^{\#}\left(h\mathbf{A}_n B(\mathsf{k}_i, y)\right)$
  end for
  $\mathsf{k}_i = B^{\#}\left(\varphi(h\gamma\mathbf{A}_n) B(\mathsf{q}, y)\right)$
end for
$\mathsf{a}_n = \mathsf{a}_0$
for $i = 1, \ldots, s$ **do**
  $\mathsf{a}_n = \mathsf{a}_n + b_i \cdot \mathsf{k}_i$
end for

The order conditions of the W-methods are obtained by matching the B-series coefficients of the exact solution $B^{\#}\left(y(t_n + h)\right)$ with those of the numerical solution $B^{\#}\left(\mathsf{y}_{n+1}\right)$ up to a specified order. Keeping in mind that we do not ultimately seek order conditions for a W-method itself, that they are simply a means to an end, we look now at the process for obtaining order conditions of the K-method from this result.

The extension of the theory of TW-trees to TK-trees is done in [56]. This extension allows us to "recolor" all linear sub-trees (possessing only singly branched nodes) of the TW-trees and to substantially reduce the number of required conditions. This is done using Lemmas 5 and 6, taken from [56], and repeated here without proof.

Lemma 5 (Property of the Krylov approximate Jacobian (3.5) [56])**.** *For any* $0 \leq k \leq M-1$ *it holds that*

$$\mathbf{A}_n^k\, f_n = \mathbf{J}_n^k\, f_n\,,$$

where $M = dim(\mathcal{K}_M)$*.*

Lemma 6 (Property of elementary differentials using the approximation (3.5) [56])**.** *When the Krylov approximation matrix* (3.5) *is used in equation (3.2), all linear TW-trees of order* $k \leq M$ *correspond to a single elementary differential, regardless of the color of their nodes.*

TK-trees are the result of an application of Lemmas 5 and 6 to reduce the set of TW-trees that need to be considered in the order conditions when the Krylov approximation matrix (3.5) is used [56].

Definition 5 (TK-trees [56])**.**

$$\begin{aligned} TK &= \{TW\text{-trees: no linear sub-tree has a fat root}\} \\ TK(k) &= \{TW\text{-trees: no linear sub-tree of order} \\ &\qquad\qquad \text{smaller than or equal to } k \text{ has a fat root}\}\,. \end{aligned}$$

Figure 3.3 shows all TK-trees to order four and the corresponding exponential-K order conditions. Note that there are only nine TK-trees as opposed to the original twenty TW-trees. There is only one additional order condition compared to methods which make use of the exact Jacobian, and this condition corresponds to a tree which has a doubly-branched node occurring as a descendant of a fat node.

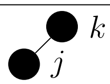
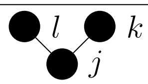
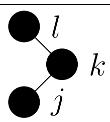
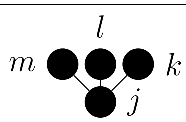
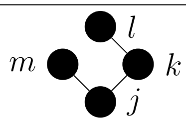
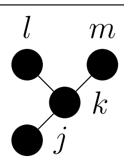
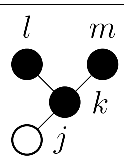
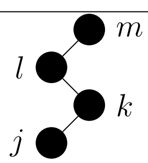
Remark 6. *The order conditions given here are only for the exponential-K methods, but the same process can be used to rederive the Rosenbrock-K conditions given in [56] through the use of different* c_n *in the Taylor expansion of* $\varphi(h\gamma\mathbf{A}_n)$ *in equation (3.6).*

Theorem 3 (Order conditions for exponential-K methods)**.** *An exponential-K method has order p iff the underlying Krylov space* (3.3) *has dimension* $M \geq p$*, and the following order conditions hold:*

$$\sum_j b_j\, \Phi_j(\tau) = P_\tau(\tau) \quad \forall\, \tau \in TK \quad \text{with } |\tau| \leq p\,. \tag{3.10}$$

Here $|\tau|$ *is the order, or number of vertices of the tree* τ*, and* $\Phi_j(\tau)$ *and* $P_\tau(\tau)$ *are computed using Algorithm 5; they are shown in Figure 3.3 for* $p \leq 4$*.*

Figure 3.3: TK-trees and exponential-K conditions up to order four.

τ	$F(\tau)$	$\Phi(\tau)$	$P_\tau(\gamma)$
	f^J	1	1
	$f^J_K f^K$	$\sum \beta_{j,k}$	$1/2\,(1-\gamma)$
	$f^J_{KL} f^K f^L$	$\sum \alpha_{j,k}\alpha_{j,l}$	$1/3$
	$f^J_K f^K_L f^L$	$\sum \beta_{j,k}\beta_{k,l}$	$1/3(1/2-\gamma)(1-\gamma)$
	$f^J_{KLM} f^K f^L f^M$	$\sum \alpha_{j,k}\alpha_{j,l}\alpha_{jm}$	$1/4$
	$f^J_{KM} f^K_L f^L f^M$	$\sum \alpha_{j,k}\beta_{k,l}\alpha_{j,m}$	$1/8-\gamma/6$
	$f^J_K f^K_{LM} f^L f^M$	$\sum \alpha_{j,k}\alpha_{k,m}\alpha_{k,l}$	$1/12$
	$\mathbf{A}_{JK} f^K_{LM} f^L f^M$	$\sum \gamma_{j,k}\alpha_{k,m}\alpha_{k,l}$	$-\gamma/6$
	$f^J_K f^K_L f^L_M f^M$	$\sum \beta_{j,k}\beta_{k,l}\beta_{l,m}$	$1/4(1/3-\gamma)(1/2-\gamma)(1-\gamma)$

Proof. The proof follows from our discussion, the near equivalence of order conditions for exponential-W and Rosenbrock-W methods [46], and from the order conditions of Rosenbrock-W methods [11, Theorem 7.7]. □

Remark 7 (Stiff order conditions.). *This section has developed classical order conditions that explain the accuracy of the methods on non-stiff problems. The behavior of the methods when applied to very stiff problems may be different, e.g., true to order reduction. A stiff order conditions theory for exponential methods has been proposed by Luan and Ostermann [63]. The development of stiff order conditions for exponential-K methods falls outside the scope of this paper.*

Remark 8 (Stability considerations.). *The numerical stability of exponential-K solutions depends on the choice of Krylov space. Intuitively, the size of the K space should be large enough to cover the stiff subspace of the system.Note that traditional exponential methods focus on accurate computations of matrix-exp-vector products (e.g., by monitoring residuals), but do not account explicitly for the impact of Krylov approximations on stability. The large number of basis vectors required to achieve accurate matrix function vector products favors stability. In our case a small K dimension ensures accuracy, so the stability needs to be considered separately. An automatic procedure to select dimension such as to achieve stability is important, but falls outside the scope here.*

3.4 An exponential-K method of order four

We now construct an exponential-K method of order four. As before we consider the case where $\gamma_{i,i} = \gamma$ for all stages i and denote

$$\beta_{i,j} = \alpha_{i,j} + \gamma_{i,j}, \quad \beta'_i = \sum_{j=1}^{i-1} \beta_{i,j}.$$

The following nine non-linear equations arise from the order conditions of a four stage, fourth order exponential-K method

$$
\begin{array}{llcl}
(a) & b_1 + b_2 + b_3 + b_4 & = & 1 \\
(b) & b_2\beta_2' + b_3\beta_3' + b_4\beta_4' & = & \frac{1}{2}(1-\gamma) \quad = \quad p_{21}(\gamma) \\
(c) & b_2\alpha_2^2 + b_3\alpha_3^2 + b_4\alpha_4^2 & = & \frac{1}{3} \\
(d) & b_3(\beta_{3,2}\beta_2') + b_4(\beta_{4,2}\beta_2' + \beta_{4,3}\beta_3') & = & \frac{1}{3}(\frac{1}{2}-\gamma)(1-\gamma) \quad = \quad p_{3,2}(\gamma) \\
(e) & b_2\alpha_2^3 + b_3\alpha_3^3 + b_4\alpha_4^3 & = & \frac{1}{4} \\
(f) & b_3\alpha_{3,2}\beta_2' + b_4(\alpha_{4,2}\beta_2' + \alpha_{4,3}\beta_3) & = & \frac{1}{8} - \frac{1}{6}\gamma \quad = \quad p_{4,2}(\gamma) \\
(g_1) & b_3\alpha_{3,2}\alpha_2^2 + b_4(\alpha_{4,2}\alpha_2^2 + \alpha_{4,3}\alpha_3^2) & = & \frac{1}{12} \\
(g_2) & b_3\gamma_{3,2}\alpha_2^2 + b_4(\gamma_{4,2}\alpha_2^2 + \gamma_{4,3}\alpha_3^2) & = & -\frac{1}{6}\gamma \\
(h) & b_4\beta_{4,3}\beta_{3,2}\beta_2' & = & \frac{1}{4}(\frac{1}{3}-\gamma)(\frac{1}{2}-\gamma)(1-\gamma) \quad = \quad p_{4,4}(\gamma)
\end{array}
\tag{3.11}
$$

If we now set

$$p_{4,3}(\gamma) = \frac{1}{12} - \frac{1}{6}\gamma,$$

we can follow exactly the solution procedure given in [56] for obtaining the ROK4a method, where we make use of the $p_{i,j}$ given above, and as suggested in [46] to guarantee exact solutions for linear ODEs choose γ as the reciprocal of an integer. For the EXPK method given in Table 3.1 we make the arbitrary choices

$$\gamma = \frac{1}{4}, \quad b_3 = 0, \quad \alpha_2 = 1, \quad \alpha_3 = \alpha_4 = \frac{1}{2}, \quad \beta_{4,3} = -\frac{1}{4}.$$

3.5 Alternative implementations of existing exponential methods

We now consider alternative implementations of previously derived methods EXP4 [46] and EROW4 [43]. These reformulations make use of only a single Krylov subspace projection per time step and exploit the B-series analysis of Section 3.3.

$\gamma = \frac{1}{4}$					
$\alpha_{2,1}$	$=$	1	$\gamma_{2,1}$	$=$	$\frac{7}{8}$
$\alpha_{3,1}$	$=$	$\frac{41}{80}$	$\gamma_{3,1}$	$=$	$\frac{1}{16}$
$\alpha_{3,2}$	$=$	$\frac{1}{80}$	$\gamma_{3,2}$	$=$	0
$\alpha_{4,1}$	$=$	$\frac{1}{4}$	$\gamma_{4,1}$	$=$	$-\frac{1}{32}$
$\alpha_{4,2}$	$=$	$\frac{1}{12}$	$\gamma_{4,2}$	$=$	$\frac{1}{24}$
$\alpha_{4,3}$	$=$	$\frac{1}{6}$	$\gamma_{4,3}$	$=$	$-\frac{5}{12}$
b_1	$=$	$\frac{1}{6}$	$\widehat{b}_1$	$=$	$\frac{8}{3}$
b_2	$=$	$\frac{1}{6}$	$\widehat{b}_2$	$=$	1
b_3	$=$	0	$\widehat{b}_3$	$=$	$-\frac{8}{3}$
b_4	$=$	$\frac{2}{3}$	$\widehat{b}_4$	$=$	0

Table 3.1: Coefficients of EXPK, a fourth order exponential-K method.

The method EXP4 [46] has the alternative formulation:

$$k_1 = \varphi_1\left(\frac{1}{3}h\mathbf{A}_n\right)f(y_n), \quad k_2 = \varphi_1(\frac{2}{3}h\mathbf{A}_n)f(y_n), \quad k_3 = \varphi_1(h\mathbf{A}_n)f(y_n), \tag{3.12a}$$

$$w_4 = \frac{-7}{300}k_1 + \frac{97}{150}k_2 - \frac{37}{300}k_3, \tag{3.12b}$$

$$u_4 = y_n + hw_4, \quad d_4 = f(u_4) - f(y_n) - h\mathbf{A}_n w_4, \tag{3.12c}$$

$$k_4 = \varphi_1\left(\frac{1}{3}h\mathbf{A}_n\right)d_4, \quad k_5 = \varphi_1(\frac{2}{3}h\mathbf{A}_n)d_4, \quad k_6 = \varphi_1(h\mathbf{A}_n)d_4, \tag{3.12d}$$

$$w_7 = \frac{59}{300}k_1 - \frac{7}{75}k_2 + \frac{269}{300}k_3 + \frac{2}{3}\left(k_4 + k_5 + k_6\right), \tag{3.12e}$$

$$u_7 = y_n + hw_7, \quad d_7 = f(u_7) - f(y_n) - h\mathbf{A}_n w_7, \tag{3.12f}$$

$$k_7 = \varphi_1\left(\frac{1}{3}h\mathbf{A}_n\right)d_7, \tag{3.12g}$$

$$y_1 = y_n + h\left(k_3 + k_4 - \frac{4}{3}k_5 + k_6 + \frac{1}{6}k_7\right). \tag{3.12h}$$

The method EROW4 [43] has the alternative formulation

$$k_1 = \varphi_1(\frac{1}{2}h\mathbf{A}_n)f(y_n), \tag{3.13a}$$

$$w_2 = \frac{1}{2}k_1, \tag{3.13b}$$

$$u_2 = y_n + hw_2, \quad d_2 = f(u_2) - f(y_n) - h\mathbf{A}_n w_2, \tag{3.13c}$$

$$k_2 = \varphi_1(h\mathbf{A}_n)f(y_n), \quad k_3 = \varphi_1(h\mathbf{A}_n)d_2, \tag{3.13d}$$

$$w_4 = k_2 + k_3, \tag{3.13e}$$

$$u_4 = y_n + hw_4, \quad d_4 = f(u_4) - f(y_n) - h\mathbf{A}_n w_4, \tag{3.13f}$$

$$k_4 = \varphi_3(h\mathbf{A}_n)d_2, \quad k_5 = \varphi_4(h\mathbf{A}_n)d_2, \quad k_6 = \varphi_3(h\mathbf{A}_n)d_4, \quad k_7 = \varphi_4(h\mathbf{A}_n)d_4, \tag{3.13g}$$

$$y_{n+1} = y_n + h\left(k_2 + 16k_4 - 48k_5 - 2k_6 + 12k_7\right) \tag{3.13h}$$

We implement these methods in three different forms: first, in the standard way outlined in the literature [57, 45, 46]; second, entirely in the reduced space such as given in (3.12), (3.13), and in [56]; and finally, using only a single Krylov projection to approximate the φ functions. These implementations are discussed below.

3.5.1 Standard implementation

The primary feature of a standard implementation of an exponential method is the approximation of φ functions using Krylov subspaces. For a term of the form $\varphi(h\mathbf{A}_n)b$ this is done by projecting $\varphi(h\mathbf{A}_n)$ and b onto the space $\mathcal{K}_M(\mathbf{A}_n, b) = \text{span}\left\{b, \mathbf{A}_n b, \mathbf{A}_n^2 b, \ldots, \mathbf{A}_n^{M-1}b\right\}$ as follows

$$\varphi(h\mathbf{A}_n)b \approx \mathbf{V}\mathbf{V}^T\varphi(h\mathbf{A}_n)\mathbf{V}\mathbf{V}^T b. \tag{3.14}$$

Note that $V^T b = \|b\|_2 e_1$, where e_1 is the first canonical basis vector. Making use of equation (3.4) we obtain the final Krylov subspace approximation

$$\varphi(h\mathbf{A}_n)b \approx \|b\|_2 V\varphi(h\mathbf{H})e_1\,. \tag{3.15}$$

This approximation is computed as in [64], in which the exponential of an augmented matrix $\widetilde{\mathbf{H}}$ is constructed [59] and (3.15) is read off from this result.

Remark 9. *The standard implementation requires the construction of a new Krylov space for each vector b operated on by a φ function, as well as the evaluation of a small matrix exponential to compute each $\varphi(h\gamma\mathbf{H})$ function. Both* EXP4 *and* EROW4 *require the construction of three Krylov subspaces and the evaluation of seven small matrix exponentials.*

3.5.2 K-type implementation

K-type implementations of EXP4 and EROW4 follow the style of [56] and Section 1. For each $k_i \in \mathbb{R}^N$ we create a corresponding $\lambda_i = \mathbf{V}^T k_i \in \mathbb{R}^M$, similarly $\sigma_i = \mathbf{V}^T w_i \in \mathbb{R}^M$, and evaluate all linear algebra operations, including Jacobian-vector products, in the reduced space.

Further, we construct only a single Krylov subspace and perform full matrix computations of three φ function evaluations in the case of EXP4, and four in the case of EROW4.

The K-type implementation of EXP4, called EXP4K, is:

$$\psi_0 = \mathbf{V}^T f(y_n), \quad f_0^{\perp} = f - \mathbf{V}\psi_0,$$

$$\lambda_1 = \varphi_1(\frac{1}{3}h\mathbf{H})\psi_0, \quad \lambda_2 = \varphi_1(\frac{2}{3}h\mathbf{H})\psi_0, \quad \lambda_3 = \varphi_1(h\mathbf{H})\psi_0,$$

$$k_1 = \mathbf{V}\lambda_1 + f_0^{\perp}, \quad k_2 = \mathbf{V}\lambda_2 + f_0^{\perp}, \quad k_3 = \mathbf{V}\lambda_3 + f_0^{\perp}, \tag{3.16a}$$

$$w_4 = \frac{-7}{300}k_1 + \frac{97}{150}k_2 - \frac{37}{300}k_3, \quad \sigma_4 = \frac{-7}{300}\lambda_1 + \frac{97}{150}\lambda_2 - \frac{37}{300}\lambda_3, \tag{3.16b}$$

$$u_4 = y_n + hw_4, \quad \psi_4 = \mathbf{V}^T f(u_4), \quad f_4^{\perp} = f(u_4) - \mathbf{V}\psi_4, \quad \delta_4 = \psi_4 - \psi_0 - h\mathbf{H}\sigma_4. \tag{3.16c}$$

$$\lambda_4 = \varphi_1(\frac{1}{3}h\mathbf{H})\delta_4, \quad \lambda_4 = \varphi_1(\frac{1}{3}h\mathbf{H})\delta_4, \quad \lambda_4 = \varphi_1(\frac{1}{3}h\mathbf{H})\delta_4,$$

$$k_4 = \mathbf{V}\lambda_4 + f_4^{\perp} - f_0^{\perp}, \quad k_5 = \mathbf{V}\lambda_5 + f_4^{\perp} - f_0^{\perp}, \quad k_6 = \mathbf{V}\lambda_6 + f_4^{\perp} - f_0^{\perp}, \tag{3.16d}$$

$$w_7 = \frac{59}{300}k_1 - \frac{7}{75}k_2 + \frac{269}{300}k_3 + \frac{2}{3}\left(k_4 + k_5 + k_6\right),$$

$$\sigma_7 = \frac{59}{300}\lambda_1 - \frac{7}{75}\lambda_2 + \frac{269}{300}\lambda_3 + \frac{2}{3}\left(\lambda_4 + \lambda_5 + \lambda_6\right), \tag{3.16e}$$

$$u_7 = y_n + hw_7, \quad \psi_7 = \mathbf{V}^T f(u_7), \quad f_7^{\perp} = f(u_7) - V\psi_7, \quad \delta_7 = \psi_7 - \psi_0 - h\mathbf{H}\sigma_7. \tag{3.16f}$$

$$\lambda_7 = \varphi_1(\frac{1}{3}h\mathbf{H})\delta_7, \quad k_7 = \mathbf{V}^T\lambda_7 + f_7^{\perp} - f_0^{\perp}, \tag{3.16g}$$

$$y_1 = y_n + h\left(k_3 + k_4 - \frac{4}{3}k_5 + k_6 + \frac{1}{6}k_7\right). \tag{3.16h}$$

The K-type implementation of EROW4, called EROW4K, is:

$$\psi_0 = \mathbf{V}^T f(y_n), \quad f_0^{\perp} = f(y_n) - \mathbf{V}\psi_0,$$

$$\lambda_1 = \varphi_1(\frac{1}{2}h\mathbf{H})\psi_0, \quad k_1 = \mathbf{V}\lambda_1 + f_0^{\perp}, \tag{3.17a}$$

$$w_2 = \frac{1}{2}k_1, \quad \sigma_2 = \frac{1}{2}\lambda_1, \tag{3.17b}$$

$$u_2 = y_n + hw_2, \quad \psi_2 = \mathbf{V}^T f(u_2), \quad f_2^{\perp} = f(u_2) - \mathbf{V}\psi_2, \quad \delta_2 = \psi_2 - \psi_0 - h\mathbf{H}\sigma_2, \tag{3.17c}$$

$$\lambda_2 = \varphi_1(h\mathbf{H})\psi_0, \quad k_2 = \mathbf{V}\lambda_2 + f_0^{\perp},$$

$$\lambda_3 = \varphi_1(h\mathbf{H})\delta_2, \quad k_3 = \mathbf{V}\lambda_3 + f_2^{\perp} - f_0^{\perp}, \tag{3.17d}$$

$$w_4 = k_2 + k_3, \quad \sigma_4 = \lambda_2 + \lambda_3, \tag{3.17e}$$

$$u_4 = y_n + hw_4, \quad \psi_4 = \mathbf{V}^T f(u_4), \quad f_4^{\perp} = f(u_4) - V\psi_4, \quad \delta_4 = \psi_4 - \psi_0 - h\mathbf{H}\sigma_4, \tag{3.17f}$$

$$\lambda_4 = \varphi_3(h\mathbf{H})\delta_2, \quad k_4 = \mathbf{V}\lambda_4 + \frac{1}{3!}\left(f_2^{\perp} - f_0^{\perp}\right),$$

$$\lambda_5 = \varphi_4(h\mathbf{H})\delta_2,, \quad k_5 = \mathbf{V}\lambda_5 + \frac{1}{4!}\left(f_2^{\perp} - f_0^{\perp}\right),$$

$$\lambda_6 = \varphi_3(h\mathbf{H})\delta_4, \quad k_6 = \mathbf{V}\lambda_6 + \frac{1}{3!}\left(f_4^{\perp} - f_0^{\perp}\right),$$

$$\lambda_7 = \varphi_4(h\mathbf{H})\delta_4, \quad k_7 = \mathbf{V}\lambda_7 + \frac{1}{4!}\left(f_4^{\perp} - f_0^{\perp}\right), \tag{3.17g}$$

$$y_{n+1} = y_n + h\left(k_2 + 16k_4 - 48k_5 - 2k_6 + 12k_7\right) \tag{3.17h}$$

3.5.3 Single projection implementation

The results of Section 3.3 imply that a single Krylov subspace need to be computed per time step guarantee the order of accuracy. In contradistinction the standard implementation constructs several Krylov spaces, primarily due to the use of residuals indicating how accurately the matrix function times vector products have been approximated. In the single projection implementation we construct only the one subspace, and similarly to K-type implementation compute the full matrix result of the φ functions. The implementation differs from the K-type implementation in that the linear algebra operations, including Jacobian-vector products, are computed in the full space.

The single projection implementation of EXP4, called EXP4SP, is:

$$\psi_0 = \mathbf{V}^T f(y_n), \quad f_0^{\perp} = f(y_n) - \mathbf{V}\psi_0,$$
$$\lambda_1 = \varphi_1(\frac{1}{3}h\mathbf{H})\psi_0, \quad \lambda_2 = \varphi_1(\frac{2}{3}h\mathbf{H})\psi_0, \quad \lambda_3 = \varphi_1(h\mathbf{H})\psi_0,$$
$$k_1 = \mathbf{V}\lambda_1 + f_0^{\perp}, \quad k_2 = \mathbf{V}\lambda_2 + f_0^{\perp}, \quad k_3 = \mathbf{V}\lambda_3 + f_0^{\perp}, \tag{3.18a}$$
$$w_4 = \frac{-7}{300}k_1 + \frac{97}{150}k_2 - \frac{37}{300}k_3, \tag{3.18b}$$
$$u_4 = y_n + hw_4, \quad d_4 = f(u_4) - f(y_n) - h\mathbf{J}_n w_4, \tag{3.18c}$$
$$\psi_4 = \mathbf{V}^T d_4, \quad d_4^{\perp} = d_4 - \mathbf{V}\psi_4,$$
$$\lambda_4 = \varphi_1(\frac{1}{3}h\mathbf{H})\psi_4, \quad \lambda_5 = \varphi_1(\frac{2}{3}h\mathbf{H})\psi_4, \quad \lambda_6 = \varphi_1(h\mathbf{H})\psi_4,$$
$$k_4 = \mathbf{V}\lambda_4 + d_4^{\perp}, \quad k_5 = \mathbf{V}\lambda_5 + d_4^{\perp}, \quad k_6 = \mathbf{V}\lambda_6 + d_4^{\perp}, \tag{3.18d}$$
$$w_7 = \frac{59}{300}k_1 - \frac{7}{75}k_2 + \frac{269}{300}k_3 + \frac{2}{3}\left(k_4 + k_5 + k_6\right), \tag{3.18e}$$
$$u_7 = y_n + hw_7, \quad d_7 = f(u_7) - f(y_n) - h\mathbf{J}_n w_7, \tag{3.18f}$$
$$\psi_7 = \mathbf{V}^T d_7, \quad d_7^{\perp} = d_7 - \mathbf{V}\psi_7,$$
$$\lambda_7 = \varphi_1(\frac{1}{3}h\mathbf{H})\psi_7, \quad k_7 = \mathbf{V}\lambda_7 + d_7^{\perp}, \tag{3.18g}$$
$$y_1 = y_n + h\left(k_3 + k_4 - \frac{4}{3}k_5 + k_6 + \frac{1}{6}k_7\right). \tag{3.18h}$$

The single projection implementation of EROW4, called EROW4SP, is:

$$\psi_0 = \mathbf{V}^T f(y_n), \quad f_0^{\perp} = f(y_n) - \mathbf{V}\psi_0,$$
$$\lambda_1 = \varphi_1(\frac{1}{2}h\mathbf{H})f(y_n), \quad k_1 = \mathbf{V}\lambda_1 + f_0^{\perp}, \tag{3.19a}$$
$$w_2 = \frac{1}{2}k_1, \tag{3.19b}$$
$$u_2 = y_n + hw_2, \quad d_2 = f(u_2) - f(y_n) - h\mathbf{J}_n w_2, \tag{3.19c}$$
$$\psi_2 = V^T d_2, \quad d_2^{\perp} = d_2 - \mathbf{V}\psi_2,$$
$$\lambda_2 = \varphi_1(h\mathbf{H})\psi_0, \quad k_2 = \mathbf{V}\lambda_2 + f_0^{\perp},$$
$$\lambda_3 = \varphi_1(h\mathbf{H})\psi_2, \quad k_3 = \mathbf{V}\lambda_3 + d_2^{\perp}, \tag{3.19d}$$
$$w_4 = k_2 + k_3, \tag{3.19e}$$
$$u_4 = y_n + hw_4, \quad d_4 = f(u_4) - f(y_n) - h\mathbf{J}_n w_4, \tag{3.19f}$$
$$\psi_4 = V^T d_4, \quad d_4^{\perp} = d_4 - \mathbf{V}\psi_4,$$
$$\lambda_4 = \varphi_3(h\mathbf{H})d_2, \quad k_4 = \mathbf{V}\lambda_4 + \frac{1}{3!}d_2^{\perp}, \quad \lambda_5 = \varphi_4(h\mathbf{H})d_2, \quad k_5 = \mathbf{V}\lambda_5 + \frac{1}{4!}d_2^{\perp},$$
$$\lambda_6 = \varphi_3(h\mathbf{H})d_4, \quad k_6 = \mathbf{V}\lambda_6 + \frac{1}{3!}d_4^{\perp}, \quad \lambda_7 = \varphi_4(h\mathbf{H})d_4, \quad k_7 = \mathbf{V}\lambda_7 + \frac{1}{4!}d_4^{\perp}, \tag{3.19g}$$
$$y_{n+1} = y_n + h\left(k_2 + 16k_4 - 48k_5 - 2k_6 + 12k_7\right). \tag{3.19h}$$

3.5.4 Accuracy analysis of alternative implementations

Using the approach described by Algorithm 5 we construct B-series representations of the numerical solutions produced by EXP4K, EXP4SP, EROW4K, and EROW4SP. Table 3.2 shows the B-series coefficients for up to fourth order. Note that coefficients associated to various trees change not only for different methods but also for different formulations of the same method.

The critical coefficient is that belonging to τ_{13}, i.e., corresponding to the K order condition. Table 3.2 reveals that EXP4K is fourth order, while both EXP4SP, EROW4K, and EROW4SP are only third order. These analytical results are confirmed experimentally in the next section.

3.6 Numerical Results

We perform numerical tests using a nonlinear ODE model, the two-dimensional shallow water equations, and the two-dimensional Allen-Cahn problem. While we have constructed both K- and SP- type implementations of EXP4 and EROW4, we present performance comparisons of only the fourth order methods EXPK, EXP4, EXP4K, and EROW4, since the third order methods perform the same amount of work as fourth order methods but yield lower accuracy.

3.6.1 Lorenz-96 model

The chaotic Lorenz-96 model [51] has $N = 40$ states, periodic boundary conditions, and is described by the following equations:

$$\begin{aligned} \frac{dy_j}{dt} &= -y_{j-1}\,(y_{j-2} - y_{j+1}) - y_j + F\,, \quad j = 1, \ldots, N\,, \\ y_{-1} &= y_{N-1}\,, \quad y_0 = y_N\,, \quad y_{N+1} = y_1\,. \end{aligned} \tag{3.20}$$

The forcing term is $F = 8.0$, with $t \in [0,\ 0.3]$ (time units).

Figure 3.4 shows the precision diagrams for EXPK with $M = 5$, and for the standard implementations of EXP4 and EROW4. All methods show the theoretical convergence order four. The performance of different implementations of EXP4 and EROW4 are shown in Figures 3.5(a) and 3.5(b), respectively. The results confirm the lower orders of alternative implementations predicted by the B-series analysis presented in table 3.2. The convergence rates for all methods applied to the Lorenz-96 model are summarized in Table 3.3 .

<table>
<tr><th>i</th><th>$F(\tau_i)$</th><th>EXP4K</th><th>EXP4SP</th><th>EROW4K</th><th>EROW4SP</th><th>Exact Solution</th></tr>
<tr><td>1</td><td>f^J</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td></tr>
<tr><td>2</td><td>$f^J_K f^K$</td><td>$\frac{1}{2}$</td><td>0</td><td>$\frac{1}{2}$</td><td>0</td><td rowspan="2">$\frac{1}{2}$</td></tr>
<tr><td>3</td><td>$\mathbf{A}_{JK} f^K$</td><td>0</td><td>$\frac{1}{2}$</td><td>0</td><td>$\frac{1}{2}$</td></tr>
<tr><td>4</td><td>$f^J_{KL} f^K f^L$</td><td>$\frac{1}{3}$</td><td>$\frac{1}{3}$</td><td>$\frac{1}{3}$</td><td>$\frac{1}{3}$</td><td>$\frac{1}{3}$</td></tr>
<tr><td>5</td><td>$f^J_K f^K_L f^L$</td><td>$\frac{1}{6}$</td><td>0</td><td>$\frac{1}{12}$</td><td>0</td><td rowspan="4">$\frac{1}{6}$</td></tr>
<tr><td>6</td><td>$f^J_K \mathbf{A}_{KL} f^L$</td><td>$\frac{1}{120}$</td><td>0</td><td>$\frac{1}{12}$</td><td>0</td></tr>
<tr><td>7</td><td>$\mathbf{A}_{JK} f^K_L f^L$</td><td>$-\frac{1}{36}$</td><td>0</td><td>$\frac{1}{15}$</td><td>0</td></tr>
<tr><td>8</td><td>$\mathbf{A}_{JK} \mathbf{A}_{KL} f^L$</td><td>$\frac{7}{360}$</td><td>$\frac{1}{6}$</td><td>$-\frac{1}{15}$</td><td>$\frac{1}{6}$</td></tr>
<tr><td>9</td><td>$f^J_{KLM} f^K f^L f^M$</td><td>$\frac{1}{4}$</td><td>$\frac{1}{4}$</td><td>$\frac{1}{4}$</td><td>$\frac{1}{4}$</td><td>$\frac{1}{4}$</td></tr>
<tr><td>10</td><td>$f^J_{KL} f^L_M f^M f^K$</td><td>$\frac{1}{6}$</td><td>0</td><td>$\frac{1}{12}$</td><td>0</td><td rowspan="2">$\frac{1}{8}$</td></tr>
<tr><td>11</td><td>$f^J_{KL} \mathbf{A}_{LM} f^M f^K$</td><td>$-\frac{1}{24}$</td><td>$\frac{1}{8}$</td><td>$\frac{1}{24}$</td><td>$\frac{1}{8}$</td></tr>
<tr><td>12</td><td>$f^J_K f^K_{LM} f^M f^L$</td><td>$\frac{1}{12}$</td><td>0</td><td>$\frac{1}{24}$</td><td>0</td><td>$\frac{1}{12}$</td></tr>
<tr><td>13</td><td>$\mathbf{A}_{JK} f^K_{LM} f^M f^L$</td><td>0</td><td>$\frac{1}{12}$</td><td>$\frac{1}{24}$</td><td>$\frac{1}{12}$</td><td>0</td></tr>
<tr><td>14</td><td>$f^J_K f^K_L f^L_M f^M$</td><td>0</td><td>0</td><td>0</td><td>0</td><td rowspan="8">$\frac{1}{24}$</td></tr>
<tr><td>15</td><td>$f^J_K f^K_L \mathbf{A}_{LM} f^M$</td><td>$\frac{1}{20}$</td><td>0</td><td>$\frac{1}{48}$</td><td>0</td></tr>
<tr><td>16</td><td>$f^J_K \mathbf{A}_{KL} f^L_M f^M$</td><td>$\frac{1}{18}$</td><td>0</td><td>$\frac{1}{24}$</td><td>0</td></tr>
<tr><td>17</td><td>$f^J_K \mathbf{A}_{KL} \mathbf{A}_{LM} f^M$</td><td>$-\frac{1537}{24300}$</td><td>0</td><td>$-\frac{1}{48}$</td><td>0</td></tr>
<tr><td>18</td><td>$\mathbf{A}_{JK} f^K_L f^L_M f^M$</td><td>$\frac{1}{36}$</td><td>0</td><td>$\frac{1}{120}$</td><td>0</td></tr>
<tr><td>19</td><td>$\mathbf{A}_{JK} f^K_L \mathbf{A}_{LM} f^M$</td><td>$-\frac{23}{720}$</td><td>0</td><td>$\frac{1}{80}$</td><td>0</td></tr>
<tr><td>20</td><td>$\mathbf{A}_{JK} \mathbf{A}_{KL} f^L_M f^M$</td><td>$-\frac{1}{27}$</td><td>0</td><td>$-\frac{1}{60}$</td><td>0</td></tr>
<tr><td>21</td><td>$\mathbf{A}_{JK} \mathbf{A}_{KL} \mathbf{A}_{LM} f^M$</td><td>$\frac{3943}{97200}$</td><td>$\frac{1}{24}$</td><td>$-\frac{1}{240}$</td><td>$\frac{1}{24}$</td></tr>
</table>

Table 3.2: B-series expansion of the numerical solution for different exponential methods and implementations.

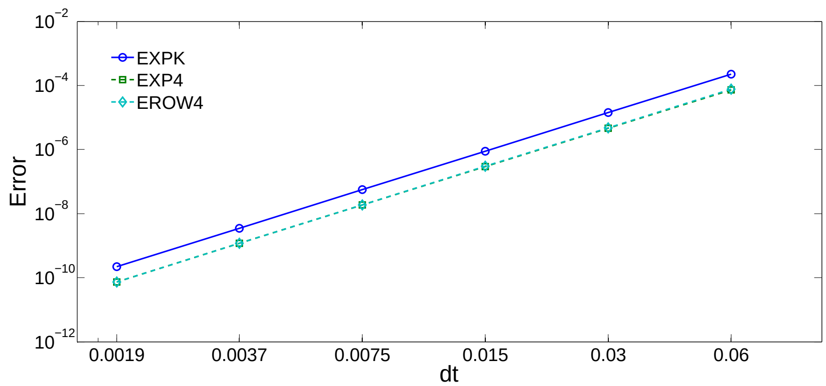
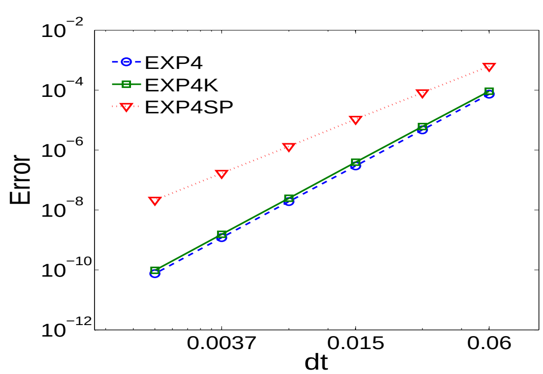


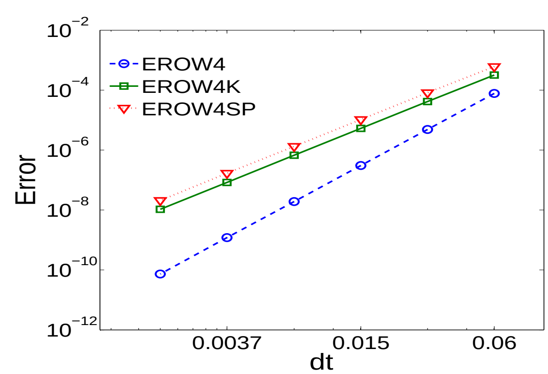
Figure 3.4: Precision diagrams for different exponential methods in standard implementation applied to the Lorenz-96 test problem. EXPK use a Krylov space of dimension $M = 5$.

	Standard	K-type	SP-type
EXPK	–	3.99	–
EXP4	3.98	3.97	2.97
EROW4	4.00	2.97	2.96

Table 3.3: Convergence rates for all methods and implementations applied to the Lorenz-96 model.



(a) EXP4



(b) EROW4

Figure 3.5: Work-precision diagrams for different implementations of traditional exponential integrators applied to the Lorenz-96 test problem (3.20).

3.6.2 Shallow water equations

We examine the relative performance of the methods on the two-dimensional shallow water equations [55], a hyperbolic system of partial differential equations

$$\frac{\partial}{\partial t}h + \frac{\partial}{\partial x}(uh) + \frac{\partial}{\partial y}(vh) = 0, \tag{3.21a}$$

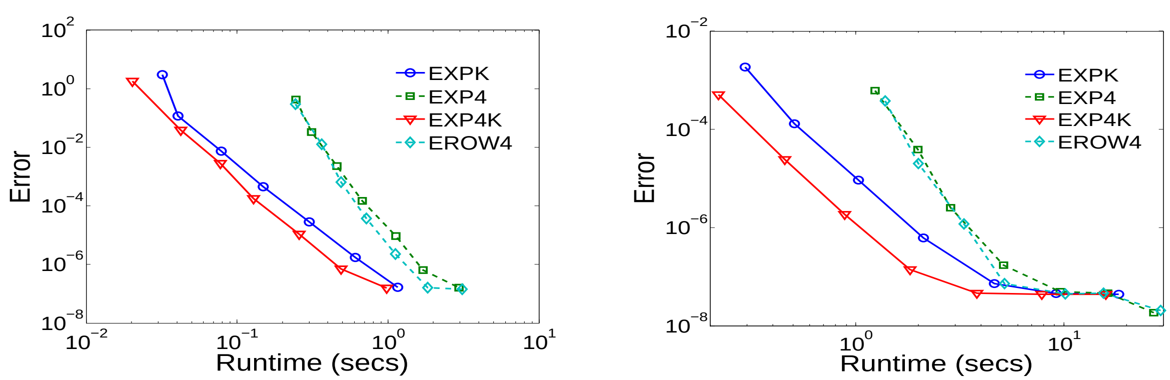
$$\frac{\partial}{\partial t}(uh) + \frac{\partial}{\partial x}\left(u^2 h + \frac{1}{2}gh^2\right) + \frac{\partial}{\partial y}(uvh) = 0, \tag{3.21b}$$

$$\frac{\partial}{\partial t}(vh) + \frac{\partial}{\partial x}(uvh) + \frac{\partial}{\partial y}\left(v^2 h + \frac{1}{2}gh^2\right) = 0, \tag{3.21c}$$

where $u(x,y,t)$, $v(x,y,t)$ are the flow velocity components and $h(x,y,t)$ is the fluid height. After spatial discretization using centered finite differences (3.21) is brought to the standard ODE form (3.1) with

$$y = [u\ v\ h]^T \in \mathbb{R}^N, \quad f_y(t,y) = \mathbf{J} \in \mathbb{R}^{N\times N}.$$

The standard exponential integrators compute the product $\varphi(h\mathbf{A}_n)b$ with an adaptive basis size to guarantee accuracy and the comparisons include the cost of the extra residual computations required to do so, while the K- type implementations use a constant basis size chosen empirically for stability. Automatic selection of Krylov basis size for stability is an open problem, and is the subject of future work. A special subroutine was implemented to compute exact Jacobian-vector products using a matrix-free approach.



(a) $N = 3 \times 32 \times 32$ (b) $N = 3 \times 128 \times 128$

Figure 3.6: Work-precision diagrams for exponential integrators applied to the shallow water test problem (3.21). Different problem sizes results from different spatial resolutions.

Figures 3.6(a) and 3.6(b) show a performance comparison of the standard and K-type implementations of EXPK, EXP4, and EROW4. Two grid sizes of 32×32 and 128×128 points are considered. In both cases the K-type implementations are more efficient for lower error tolerances, while the adaptivity of the standard implementations allows them to ‘catch up’ in performance as the errors decrease.

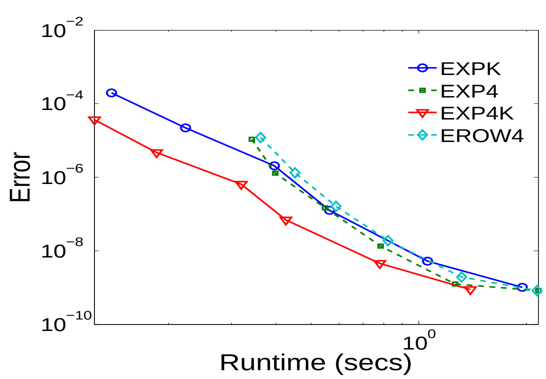
3.6.3 The Allen-Cahn problem

For further performance comparison we consider the two-dimensional Allen-Cahn system, a parabolic partial differential equation

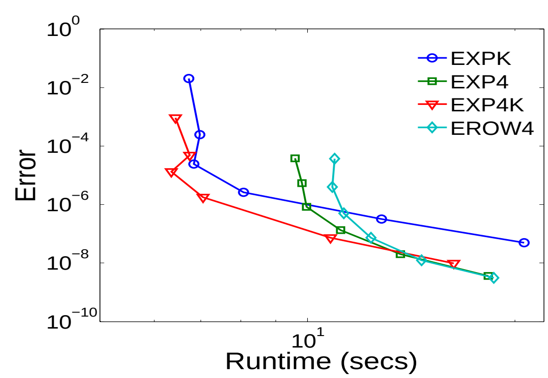
$$\frac{\partial}{\partial t}u = \alpha\nabla^2 u + \gamma\left(u - u^3\right), \quad (x, y) \in [0, 1] \times [0, 1], \quad t \in [0, 0.2], \tag{3.22}$$

with $\alpha = 0.1$ and $\gamma = 1.0$. The problem has homogeneous Neumann boundary conditions and the initial solution $u(t = 0) = 0.4 + 0.1(x + y) + 0.1 \sin(10x) \sin(20y)$. Unlike the shallow water equations, the reaction-diffusion problem (3.22) is stiff.

The standard implementations of various methods make use of an adaptive Krylov basis size while the K-type implementations use empirically selected basis sizes. Figures 3.7(a) and 3.7(b) compares the performance of the standard and K-type implementations for different problem sizes. We once again see a better efficiency of the K-type methods for lower error values, but with a much earlier break-even point for efficiency. This is due primarily to the spectrum of the diffusion operator, and the difference in stability and accuracy requirements between the shallow water and Allen-Cahn equations. In the case of Allen-Cahn the stability requirements are more strict than the accuracy considerations and so the multiple smaller projections of the standard implementation become more efficient than the single larger projection in the K-type method, even though the latter uses fewer overall basis vectors.



(a) $N = 50 \times 50$



(b) $N = 150 \times 150$

Figure 3.7: Work-precision diagrams for exponential integrators applied to the Allen-Cahn test problem (3.22). Different problem sizes result from different spatial resolutions.

3.7 Conclusions

This work extends the K-method approach proposed in [56] to exponential integrators and develops the new family of exponential-K schemes. A rigorous framework for order conditions analysis is developed that accounts for both temporal truncation errors and Krylov approximation errors. We construct an exponential-K method based on the general form given in [46], and reformulate existing exponential methods in order to take advantage of the reduced workload permitted by the new analysis.

Numerical experiments are carried out with three test problems, an ordinary differential equation and hyperbolic and parabolic partial differential equations. The results indicate that the new K-type exponential methods have the potential to be more efficient than their classical counterparts. While the new K- method EXPK derived here does not appear to be more efficient than the previously existing methods, primarily due to a less efficient general form, it validates the order conditions theory of exponential-K methods. We have shown that the traditional EXP4 method satisfies the additional order four K-condition when reformulated as a K-method, and that the resulting EXP4K scheme is more efficient than previous methods for the test problems presented here.

Future work will focus on developing a methodology to automatically select the Krylov subspace size in order to guarantee numerical stability, as on the construction of new exponential methods that can take full advantage of the inherent benefits present in the K-type formulation.

Acknowledgements

This work has been supported in part by NSF through awards NSF CMMI–1130667, NSF CCF–1218454, NSF CCF–0916493, AFOSR FA9550–12–1–0293–DEF, AFOSR 12-2640-06, and by the Computational Science Laboratory at Virginia Tech.

Chapter 4

Advances in K-Type Methods

4.1 Introduction

We seek a scheme to efficiently approximate solutions to the initial value problem

$$\frac{dy}{dt} = f(t,y)\,, \quad t_0 \le t \le t_F\,, \quad y(t_0) = y_n\,; \quad y(t), f(t,y) \in \mathbb{R}^N\,. \tag{4.1}$$

Specifically, we start our discussion with the initial general form (4.2) shared by several familes of Rosenbrock and exponential-Rosenbrock schemes

$$\begin{aligned} k_i &= \varphi(h\gamma\mathbf{A}_n)\left(h\,F_i + h\mathbf{A}_n\sum_{j=1}^{i-1}\gamma_{i,j}k_j\right), && \text{(4.2a)}\\ F_i &= f\left(y_n + \sum_{j=1}^{i-1}\alpha_{i,j}k_j\right), && \text{(4.2b)}\\ y_{n+1} &= y_n + \sum_{j=1}^{s} b_i k_i. && \text{(4.2c)} \end{aligned}$$

where $\mathbf{A}_n$ is either the exact Jacobian matrix coming from the right-hand-side vector $f(t,y)$

$$\mathbf{A}_n = \left.\frac{\partial f}{\partial y}\right|_{t=t_n}, \tag{4.3}$$

or an approximation of it. The general form (4.2) encapsulates explicit Runge-Kutta methods when $\varphi(z) = 1$, Rosenbrock methods when $\varphi(z) = 1/(1-z)$, and exponential-Rosenbrock methods when $\varphi(z) = (e^z - 1)/z$.

Interestingly, (4.2) also forms the basis of subfamilies of both Rosenbrock and exponential-Rosenbrock schemes. When the coefficients $\alpha_{i,j}$, $\gamma_{i,j}$, and b_i are chosen so that $\mathbf{A}_n$ must equal the Jacobian matrix $\mathbf{J}_n$, the scheme (4.2) is known as either a classical Rosenbrock or exponential-Rosenbrock method depending on the choice of $\varphi(z)$. Alternatively, the coefficients can be chosen so that the matrix $\mathbf{A}_n$ can be any arbitrary approximation of the Jacobian $\mathbf{J}_n$, and in this case the scheme (4.2) is known as either a Rosenbrock-W or exponential-Rosenbrock-W method depending once again on the choice of $\varphi(z)$.

The use of W-methods is convenient when the exact jacobian $\mathbf{J}_n$ is difficult, or particularly expensive, to compute. The cost of such an approach is a dramatic increase in the number of required order conditions imposed on the selection of the coefficients $\alpha_{i,j}$, $\gamma_{i,j}$, and b_i. The effect is so pronounced, in fact, that methods of order greater than three do not exist throughout the literature. For this reason, among others, we focus our attention in this manuscript on an alternative formulation of (4.2) called Rosenbrock-Krylov [56] or exponential-Krylov [65] methods, in which the coefficients are chosen to permit a specific, Krylov based approximation, $\mathbf{A}_n$, of the Jacobian matrix $\mathbf{J}_n$.

Classifying stability for W-methods with arbitrary matrices is an open question. Stability questions for the case when $\|f'(y_n) - \mathbf{A}_n\|$ is small are considered in [11, Section IV.11]. Similarly, exponential schemes are trivially A-stable when the $\varphi(\mathbf{J}_n)$ functions are evaluated exactly, however when these functions are evaluated using approximate methods this property is lost. Quantifying the impact of the approximation of the $\varphi(\mathbf{J}_n)$ on the stability of exponential methods, is closely related to the stability questions for W-methods and also remains open.

K-methods are similarly plagued with stability questions, though have the benefit of making use of a very specific formulation for the approximate Jacobian $\mathbf{A}_n$. In [66], stability for W-methods making use of the Krylov based approximation, similar to that used for K-methods, is examined through the lense of contractivity for semilinear problems. Here we will examine the linear stability of K-methods, and present two practical approaches for improving stability of these methods. Additionally, we derive stiff order conditions for application of K-methods to index-1 differential algebraic equations.

The rest of the paper is laid out as follows: in section 4.2 we give a brief overview of K-methods, in section 4.3 we discuss the issues inherent in analyzing stability for methods making use of approximate evaluation of $\varphi(\mathbf{J}_n)$ functions as well as an initial linear stability analysis for K-methods, in section 4.4 we present two strategies for improving the stability of K-methods through direct control of stage residuals, in section 4.5 we derive stiff order conditions for K-methods applied to index-1 differential algebraic equations, some numerical results are presented in section 4.6, and finally in section 4.7 we give concluding remarks.

4.2 An overview of K-type methods

A full presentation of the order condition theory for can be found in [56, 65], here we present an overview of these schemes.

Rosenbrock-Krylov methods have the same form as Rosenbrock-W methods (4.2), but use a particular approximation $\mathbf{A}$ of $\mathbf{J}_n$. We restrict our presentation to the case of autonomous systems.

Specifically, let $f_n = f(y_n)$ and consider the M-dimensional Krylov space

$$\begin{aligned} \mathcal{K}_M(\mathbf{J}_n\,, f_n) &= \operatorname{span}\left\{\, f_n,\, \mathbf{J}_n\, f_n,\, \mathbf{J}_n^2\, f_n, \ldots, \mathbf{J}_n^{M-1}\, f_n \,\right\} \\ &= \operatorname{span}\left\{v_1, v_2, \ldots, v_M\right\}\,. \end{aligned} \tag{4.4}$$

An orthogonal basis $\{v_i\}_{i=1,\ldots,M}$ for $\mathcal{K}_M$ is constructed using a modified Arnoldi process [14]. The Arnoldi iteration returns two valuable pieces of information: a matrix $\mathbf{V}$ whose columns are the orthonormal basis vectors of $\mathcal{K}_M$, and an upper Hessenberg matrix $\mathbf{H}$, such that

$$\mathbf{V} = [v_1, \ldots v_M] \in \mathbb{R}^{N\times M}\,; \quad \mathbf{H} = \mathbf{V}^T\, \mathbf{J}_n\, \mathbf{V} \in \mathbb{R}^{M\times M}\,. \tag{4.5}$$

The Krylov approximation matrix $\mathbf{A}$ *is the restriction of the full ODE Jacobian to the Krylov space:*

$$\mathbf{A} = \mathbf{V}\,\mathbf{H}\,\mathbf{V}^T = \mathbf{V}\,\mathbf{V}^T\,\mathbf{J}_n\,\mathbf{V}\,\mathbf{V}^T\,. \tag{4.6}$$

Algorithm 6 One step of an autonomous K-type integrator

1: Compute $\mathbf{H}$ and $\mathbf{V}$ using the N-dimensional Arnoldi process [14]

2: **for** $i = 1, \ldots, s$ **do** ▷ For each stage, in succession

$$\begin{aligned} F_i &= f\left(y_n + \sum_{j=1}^{i-1} \alpha_{i,j} k_j\right) \\ \psi_i &= \mathbf{V}^T\, F_i \\ \lambda_i &= \varphi\left(h\gamma\mathbf{H}\right)\left(h\psi_i + h\mathbf{H}\sum_{j=1}^{i-1}\gamma_{i,j}\lambda_j\right) \\ k_i &= \mathbf{V}\,\lambda_i + h\left(F_i - \mathbf{V}\,\psi_i\right) \end{aligned}$$

3: **end for**

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

4.3 Stability issues

The standard approach for investigating the linear stability of a time integration scheme is through application of the method to the linear test problem, or Dahlquist test equation

$$y' = \lambda y, \quad y_0 = 1, \quad z = h\lambda, \tag{4.7}$$

so that a local recurrence relation for the solution at t_{n+1} in terms of the solution at t_n can be constructed as

$$y_{n+1} = R(h\lambda)y_n$$

and the global solution to (4.7) is

$$y_{n+1} = R(h\lambda)^n y_0.$$

The set

$$S = \{z \in \mathbb{C}; |R(z)| \leq 1\}$$

is called the stability region of the method [11].

Unfortunately, this approach is not sufficient for examining the stability of Rosenbrock methods where $\mathbf{A}_n \neq f'(y_n)$, such as W-methods and K-methods, or for exponential schemes where the $\varphi(\mathbf{J}_n)$ are computed approximately. For these schemes, since in general $\mathbf{J}_n$ and $\mathbf{A}_n$ are not simultaneously diagonalizable, it is necessary instead to consider the vector-valued linear test problem

$$y' = \mathbf{J}y, \quad y_0 = \mathbb{1}. \tag{4.8}$$

Application of a W-, or K-, method to equation (4.8) will lead to the local recurrence

$$y_{n+1} = R(h\mathbf{J}, h\mathbf{A})y_n$$

with global solution

$$y_{n+1} = R(h\mathbf{J}, h\mathbf{A})^n y_0,$$

so that the stability region is determined by bounding $\|R(h\mathbf{J}, h\mathbf{A})\| \leq 1$.

4.3.1 Stability Analysis of Rosenbrock-Krylov Methods

Take the s-stage Rosenbrock method (4.2), applied to the linear test problem (4.8)

$$k_i = h\mathbf{J}\left(y_0 + \sum_{j=1}^{i-1} \alpha_{ij}k_j\right) + \left(h\mathbf{A}\sum_{j=1}^{i} \gamma_{i,j}k_j\right) \tag{4.9a}$$

$$y_{n+1} = y_n + \sum_{j=1}^{i} b_i k_i \tag{4.9b}$$

We define the supervectors

$$\underbrace{\mathbf{K} = \begin{bmatrix} k_1 \\ \vdots \\ k_s \end{bmatrix}, \quad \mathbf{Y}_n = \begin{bmatrix} y_n \\ \vdots \\ y_n \end{bmatrix} = (\mathbb{1} \otimes \mathbf{I})\, y_n, \quad \mathbf{R}_n = \begin{bmatrix} r_{n;1} \\ \vdots \\ r_{n;s} \end{bmatrix}}_{\in \mathbb{R}^{(N \cdot s)}}$$

and coefficient matrices

$$\alpha = \begin{bmatrix} 0 & \cdots & \cdots & 0 \\ \alpha_{2,1} & 0 & \cdots & \vdots \\ \vdots & \ddots & \ddots & \vdots \\ \alpha_{s,1} & \cdots & \alpha_{s,s-1} & 0 \end{bmatrix}, \quad \gamma = \begin{bmatrix} \gamma & 0 & \cdots & 0 \\ \gamma_{2,1} & \gamma & 0 & \vdots \\ \vdots & \ddots & \ddots & 0 \\ \gamma_{s,1} & \cdots & \gamma_{s,s-1} & \gamma \end{bmatrix}, \quad \beta = \alpha + \gamma$$

then (4.9) becomes

$$\begin{aligned} \mathbf{K} &= h\left(\mathbf{I} \otimes \mathbf{J}\right)\mathbf{Y}_n + h\left[\left(\alpha \otimes \mathbf{J}\right) + \left(\gamma \otimes \mathbf{A}\right)\right]\mathbf{K} && \text{(4.10a)} \\ y_{n+1} &= y_n + \left(b^T \otimes \mathbf{I}\right)\mathbf{K} && \text{(4.10b)} \end{aligned}$$

or when $\mathbf{A} = \mathbf{J}$

$$\begin{aligned} \mathbf{K} &= h\left(\mathbf{I} \otimes \mathbf{J}\right)\mathbf{Y}_n + h\left(\beta \otimes \mathbf{J}\right)\mathbf{K} + \mathbf{R}_n && \text{(4.11a)} \\ y_{n+1} &= y_n + \left(b^T \otimes \mathbf{I}\right)\mathbf{K} && \text{(4.11b)} \end{aligned}$$

$$\begin{aligned} \mathbf{K} &= h\left[\mathbf{I} - \left(\alpha \otimes h\mathbf{J}\right) - \left(\gamma \otimes h\mathbf{A}\right)\right]^{-1}\left(\mathbf{I} \otimes \mathbf{J}\right)\mathbf{Y}_n \\ &= h\left(\mathbf{I} - h\beta \otimes \mathbf{J}\right)^{-1}\left(\mathbf{I} \otimes \mathbf{J}\right)\mathbf{Y}_n + \left(\mathbf{I} - h\beta \otimes \mathbf{J}\right)^{-1}\mathbf{R}_n \\ y_{n+1} &= y_n + \left(b^T \otimes \mathbf{I}\right)\mathbf{K} \end{aligned}$$

From here we have that

$$\begin{aligned} \mathbf{R}_n &= \left(\mathbf{I} - h\beta \otimes \mathbf{J}\right)\left[\mathbf{I} - \left(\alpha \otimes \mathbf{J}\right) - \left(\gamma \otimes \mathbf{A}\right)\right]^{-1}\left(\mathbf{I} \otimes h\mathbf{J}\right)\mathbf{Y}_n - \left(\mathbf{I} \otimes h\mathbf{J}\right)\mathbf{Y}_n \\ \left(\mathbf{I} - \beta \otimes h\mathbf{J}\right)^{-1}\mathbf{R}_n &= \left(\left[\mathbf{I} - \left(\alpha \otimes \mathbf{J}\right) - \left(\gamma \otimes \mathbf{A}\right)\right]^{-1} - \left(\mathbf{I} - h\beta \otimes \mathbf{J}\right)^{-1}\right)\left(\mathbf{I} \otimes h\mathbf{J}\right)\mathbf{Y}_n \end{aligned}$$

The stability of the ROK method is given by:

$$\begin{aligned} y_{n+1} &= \left(\mathbf{I} + \left(b^T \otimes \mathbf{I}\right)\left(\mathbf{I} - \beta \otimes h\mathbf{J}\right)^{-1}\left(\mathbb{1} \otimes h\mathbf{J}\right)\right)y_n + \left(b^T \otimes \mathbf{I}\right)\left(\mathbf{I} - \beta \otimes h\mathbf{J}\right)^{-1}\mathbf{R}_n \\ &= R\left(h\mathbf{J}\right)y_n + \left(b^T \otimes \mathbf{I}\right)\left(\left[\mathbf{I} - \left(\alpha \otimes \mathbf{J}\right) - \left(\gamma \otimes \mathbf{A}\right)\right]^{-1} - \left(\mathbf{I} - h\beta \otimes \mathbf{J}\right)^{-1}\right)\left(\mathbb{1} \otimes h\mathbf{J}\right)y_n \\ &= R\left(h\mathbf{J}\right)y_n + S\left(h\mathbf{J}, h\mathbf{A}\right)y_n \\ &= \widetilde{R}\left(h\mathbf{J}, h\mathbf{A}\right)y_n \end{aligned}$$

where the effective stability function of the ROK method is not the ROS stability function R, but is given by the modification, $\widetilde{R}$ which depends on both $\mathbf{J}$ and $\mathbf{A}$.

The linear stability of the underlying Rosenbrock method

$$\left\|R\left(h\mathbf{J}\right)\right\| < 1$$

does not guarantee the stability of the Rosenbrock-Krylov method

$$\left\| \widetilde{R}\left(h\mathbf{J}, h\mathbf{A}\right)\right\| \leq 1,$$

unless the stage stability function $S(h\mathbf{J}, h\mathbf{A})$ is small in some sense. A practical way to ensure this is to control the size of the residual

$$\mathbf{R}_n = \left(b^T \otimes \mathbf{I}\right)\left(\mathbf{I} - \beta \otimes h\mathbf{J}\right) S(h\mathbf{J}, h\mathbf{A}) \cdot y_n.$$

4.4 Directly Controlling the Residual

The most obvious strategy for guaranteeing, or improving, the stability of a Rosenbrock-Krylov method is by directly controlling the residual at each stage of the time integration scheme.

Theorem 4 (Residual of the i'th stage of a Rosenbrock-Krylov method)**.** *When the time integration scheme* (4.2) *is implemented as in algorithm 6, then the residual of the i'th stage is given by:*

$$r_{m;i} = -\sum_{j=1}^{i} h^2 \gamma_{ij} \mathbf{J}\left(F_j - \mathbf{V}_m \psi_j\right) - h\mathbf{h}_{m+1,m} v_{m+1} e_m^T \sum_{j=1}^{i} \gamma_{ij} \lambda_j \tag{4.12}$$

where $m = dim(\mathcal{K}_m)$ is the dimension of the underlying Krylov subspace.

Proof. We begin with the formulation of the method as

$$k_i = hF_i + h\mathbf{J}\sum_{j=1}^{i} \gamma_{i,j} k_j + r_{m;i} \tag{4.13a}$$

$$\lambda_i = h\psi_i + h\mathbf{H}\sum_{j=1}^{i} \gamma_{i,j} \lambda_j \tag{4.13b}$$

$$\psi_i = \mathbf{V}^T F_i \tag{4.13c}$$

$$k_i = \mathbf{V}\lambda_i + h\left(F_i - \mathbf{V}\psi_i\right) \tag{4.13d}$$

Inserting (4.13d) into (4.13a) and expanding

$$r_{m;i} = \mathbf{V}\lambda_i - h\mathbf{V}\psi_i - h^2\mathbf{J}\sum_{j=1}^{i} \gamma_{i,j}\left(hF_j - h\mathbf{V}\psi_j\right) - h\mathbf{J}\,\mathbf{V}\sum_{j=1}^{i} \gamma_{i,j}\lambda_j$$

We then apply the Arnoldi relationship

$$\mathbf{J}\mathbf{V}_m = \mathbf{V}_m \mathbf{H}_m + \mathbf{h}_{m+1,m} v_{m+1} e_m^T, \tag{4.14}$$

to terms containing λ_j's

$$r_{m;i} = \mathbf{V}\lambda - h\mathbf{V}\psi_i - h^2\mathbf{J}\sum_{j=1}^{i}\gamma_{i,j}\left(F_j - \mathbf{V}\psi_j\right) - h\mathbf{V}\,\mathbf{H}\sum_{j=1}^{i}\gamma_{i,j}\lambda_j - \mathbf{h}_{m+1,m}v_{m+1}e_m^T\sum_{j=1}^{i}\gamma_{i,j}\lambda_j$$

Collecting terms with (4.13b) in mind, we obtain

$$r_{m;i} = \underbrace{\mathbf{V}\left(\lambda - h\psi - h\mathbf{H}\sum_{j=1}^{i}\lambda_{i,j}\lambda_j\right)}_{=0} - h^2\mathbf{J}\sum_{j=1}^{i}\gamma_{i,j}\left(F_j - \mathbf{V}\psi_j\right) - \mathbf{h}_{m+1,m}v_{m+1}e_m^T\sum_{j=1}^{i}\gamma_{i,j}\lambda_j,$$

and arrive at the final result (4.12). □

Unfortunately, the residuals depend not only on the reduced stage solutions, λ_j's, but also on the fullspace matrix-vector products, $\mathbf{J}\left(F_j - \mathbf{V}\psi_j\right)$. The dependence on λ_j's means that verifying the size of the residual in the final stage requires essentially computing the full timestep, and so any strategy attempting to bound this residual will be computationally infeasible. An alternative approach is to bound the residuals only in the first stage, so that only λ_1 is required, this has the additional computational benefit that $(F_1 - \mathbf{V}\psi_1) = 0$ since $F_1 \in \mathcal{K}_m$, and so no fullspace matrix-vector products are required.

Corollary 1 (Residual of the first stage of a Rosenbrock-Krylov method)**.** *When the time integration scheme* (4.2) *is implemented as in algorithm 6, then the residual of the i'th stage is given by:*

$$r_{m;1} = -h\gamma\mathbf{h}_{m+1,m}v_{m+1}e_m^T\lambda_1$$

with norm

$$\|r_{m;1}\| = \left|h\gamma\mathbf{h}_{m+1,m}\right|\left|e_m^T\lambda_1\right|.$$

It is possible then to implement a Rosenbrock-Krylov method which automatically determines the dimension, m, of the Krylov subspace to control the size of the residual in the first stage through the use of an adaptive Arnoldi process as in algorithm 7. It is possible to reduce the computational impact of computing the residuals, by only doing so for some subset of iteration numbers. For example it was proposed in [46] to compute the residual only for the test indices, $i \in \{1, 2, 3, 4, 6, 8, 11, 15, 20, 27, 36, 48\}$.

4.4.1 Adding vectors from outside the Krylov space

An alternative to simply increasing the size of the Krylov subspace, is the idea of extending the subspace with vectors from other sources. In general we can extend the Krylov basis with arbitrary vectors $b_1, \dots, b_r$, by first computing the Arnoldi matrices $\mathbf{V}_m$ and $\mathbf{H}_m$, comprised exclusively of vectors coming from the Krylov space $\mathcal{K}_m$, as usual. We then introduce the

Algorithm 7 Modified Arnoldi iteration [13]

$\beta = \|f_n\|$, $\mathbf{V}_1 = f_n/\beta$.
for $i = 1, \ldots, M$ **do**
  $\zeta = \mathbf{J}_n v_i$
  for $j = 1, \ldots, i$ **do**
    $\mathbf{h}_{j,i} = \langle \zeta, \mathbf{V}_j \rangle$
    $\zeta = \zeta - \mathbf{h}_{j,i} \mathbf{V}_j$
  end for
  $\mathbf{h}_{i+1,i} = \|\zeta\|$
  $\mathbf{V}_{i+1} = \zeta / \mathbf{h}_{i+1,i}$
  if $i \in$ testIndices **then**
    $\lambda_1 = (\mathbf{I} - h\gamma\mathbf{H}_i)^{-1} f_n$
    if $|h\gamma\mathbf{h}_{i+1,i}| \left| e_m^T \lambda_1 \right| \leq TOL$ **then**
      return
    end if
  end if
end for

new vectors $b_1, \ldots, b_r$ into $\mathbf{V}_{m+r}$ through the same orthonormalization procedure present in the Arnoldi iteration, excluding the computation of components of the columns of $\mathbf{H}_m$, so that

$$\mathbf{V}_{m+r} = \left[v_1, v_2, \ldots, v_m, \bar{v}_1, \ldots, \bar{v}_r \right].$$

We then construct the columns of the extended matrix, $\mathbf{H}_{m+r}$, to maintain the relationship that $\mathbf{H}_{m+r} = \mathbf{V}_{m+r}^T \mathbf{J} \mathbf{V}_{m+r}$ by computing products of the form:

$$h_{m+i} = \mathbf{V}^T \left(\mathbf{J} \bar{v}_i \right), \quad \text{for } i = 1, \ldots, r$$

and the matrix $\mathbf{H}_{m+r} \in \mathbb{R}^{(m+r)\times(m+r)}$ can then be constructed as

$$\mathbf{H}_{m+r} = \begin{bmatrix} \mathbf{H}_m & \\ 0 & h_{m+1}, \ldots, h_{m+r} \end{bmatrix}.$$

Remark 10. *In previous sections, the Arnoldi relationship* (4.14) *is written in terms of* v_{m+1}*, the next vector which would be added to the Krylov basis* $\mathbf{V}_m$*, and* $\mathbf{h}_{m+1,m}$ *a coefficient corresponding to* v_{m+1} *which would be added to the upper-Hessenberg matrix* $\mathbf{H}_m$*. This notation is no longer optimal, because the* $m+1th$ *vector in an extended basis* $\mathbf{V}_{m+r}$ *does not correspond to the same vector* v_{m+1}*. So, we expand the vector* $\mathbf{h}_{m+1,m} v_{m+1}$ *using definitions of the Arnoldi process, and find that*

$$\mathbf{h}_{m+1,m} v_{m+1} = \left(\mathbf{I} - \mathbf{V}_m \mathbf{V}_m^T \right) \mathbf{J} v_m,$$

where v_m *is the last column in the Krylov basis matrix* $\mathbf{V}_m$*. With this, we then rewrite the Arnoldi relationship as*

$$\mathbf{J}\mathbf{V}_m = \mathbf{V}_m \mathbf{H}_m + \left(\mathbf{I} - \mathbf{V}_m \mathbf{V}_m^T \right) \mathbf{J} v_m e_m^T. \tag{4.15}$$

Lemma 7 (Arnoldi-like relationship for extended basis)**.** *When the Krylov basis matrix* $\mathbf{V}_m$ *is extended with vectors* $\bar{v}_1$, $\bar{v}_2$, ..., $\bar{v}_r$, *the Arnoldi relationship* (4.15)

$$\mathbf{J}\,\mathbf{V}_m = \mathbf{V}_m\mathbf{H}_m + \left(\mathbf{I} - \mathbf{V}_m\,\mathbf{V}_m^T\right)\mathbf{J}v_m e_m^T$$

can be extended to the new basis as

$$\mathbf{J}\,\mathbf{V}_{m+r} = \mathbf{V}_{m+r}\mathbf{H}_{m+r} + \left(\mathbf{I} - \mathbf{V}_m\,\mathbf{V}_m^T\right)\mathbf{J}v_m e_m^T + \left(\mathbf{I} - \mathbf{V}_{m+r}\mathbf{V}_{m+r}^T\right)\mathbf{J}\sum_{i=1}^{r}\bar{v}_i e_{m+i}^T \tag{4.16}$$

where e_j *is the* j*'th canonical basis vector of* $\mathbb{R}^{(m+r)}$.

Proof. Recall that $\mathbf{V}_{m+r}$ and $\mathbf{H}_{m+r}$ have the following structure:

$$\mathbf{V}_{m+r} = \left[\begin{array}{c|c} \mathbf{V}_m & \overline{\mathbf{V}} \end{array}\right] \in \mathbb{R}^{N\times(m+r)}, \quad \mathbf{H}_{m+r} = \left[\begin{array}{c|c} \mathbf{H}_m & \multirow{2}{*}{$\overline{\mathbf{H}}$} \\ 0 & \end{array}\right] \in \mathbb{R}^{(m+r)\times(m+r)},$$

where $\overline{\mathbf{V}} = \left[\bar{v}_1, \ldots, \bar{v}_r\right]$ and $\overline{\mathbf{H}} = \mathbf{V}_{m+r}^T\left(\mathbf{J}\overline{\mathbf{V}}\right)$. Computing $\mathbf{J}\mathbf{V}_{m+r}$ gives

$$\mathbf{J}\mathbf{V}_{m+r} = \left[\begin{array}{c|c} \mathbf{J}\mathbf{V}_m & \mathbf{J}\overline{\mathbf{V}} \end{array}\right] = \left[\begin{array}{c|c} \mathbf{V}_m\mathbf{H}_m + \left(\mathbf{I} - \mathbf{V}_m\,\mathbf{V}_m^T\right)\mathbf{J}v_m e_m^T & \mathbf{J}\overline{\mathbf{V}} \end{array}\right],$$

which follows from the Arnoldi relationship (with $e_m \in \mathbb{R}^m$). Similarly, computing $\mathbf{V}_{m+r}\mathbf{H}_{m+r}$ gives

$$\mathbf{V}_{m+r}\mathbf{H}_{m+r} = \left[\begin{array}{c|c} \mathbf{V}_m & \overline{\mathbf{V}} \end{array}\right]\left[\begin{array}{c|c} \mathbf{H}_m & \multirow{2}{*}{$\overline{\mathbf{H}}$} \\ 0 & \end{array}\right] = \left[\begin{array}{c|c} \mathbf{V}_m\mathbf{H}_m & \mathbf{V}_{m+r}\overline{\mathbf{H}} \end{array}\right].$$

Subtracting gives

$$\mathbf{J}\mathbf{V}_{m+r} - \mathbf{V}_{m+r}\mathbf{H}_{m+r} = \left[\begin{array}{c|c} \left(\mathbf{I} - \mathbf{V}_m\,\mathbf{V}_m^T\right)\mathbf{J}v_m e_m^T & \mathbf{J}\overline{\mathbf{V}} - \mathbf{V}_{m+r}\overline{\mathbf{H}} \end{array}\right].$$

Substituting in the definition of $\overline{\mathbf{H}}$ and rearranging, we have

$$\mathbf{J}\mathbf{V}_{m+r} = \mathbf{V}_{m+r}\mathbf{H}_{m+r} + \left[\begin{array}{c|c} \left(\mathbf{I} - \mathbf{V}_m\,\mathbf{V}_m^T\right)\mathbf{J}v_m e_m^T & \left(\mathbf{I} - \mathbf{V}_{m+r}\mathbf{V}_{m+r}^T\right)\mathbf{J}\overline{\mathbf{V}} \end{array}\right].$$

Finally, we can rewrite the last matrix as a sum of rank 1 matrices using canonical basis vectors $e_j \in \mathbb{R}^{(m+r)}$:

$$\left[\begin{array}{c|c} \left(\mathbf{I} - \mathbf{V}_m\,\mathbf{V}_m^T\right)\mathbf{J}v_m e_m^T & \left(\mathbf{I} - \mathbf{V}_{m+r}\mathbf{V}_{m+r}^T\right)\mathbf{J}\overline{\mathbf{V}} \end{array}\right] = \left(\mathbf{I} - \mathbf{V}_m\,\mathbf{V}_m^T\right)\mathbf{J}v_m e_m^T + \sum_{i=1}^{r}\left(\mathbf{I} - \mathbf{V}_{m+r}\mathbf{V}_{m+r}^T\right)\mathbf{J}\bar{v}_i e_{m+i}^T.$$

Making this substitution and with some rearranging, we get equation (4.16),

$$\mathbf{J}\,\mathbf{V}_{m+r} = \mathbf{V}_{m+r}\mathbf{H}_{m+r} + \left(\mathbf{I} - \mathbf{V}_m\,\mathbf{V}_m^T\right)\mathbf{J}v_m e_m^T + \left(\mathbf{I} - \mathbf{V}_{m+r}\,\mathbf{V}_{m+r}^T\right)\mathbf{J}\sum_{i=1}^{r}\bar{v}_i e_{m+i}^T.$$

□

With the ability to extend the basis $\mathbf{V}$ with arbitrary vectors, the question now remains: what vectors provide improved benefits in controlling the residuals of later stages of the Rosenbrock-Krylov method?

4.4.2 Exploiting the form of the residual

We see from equation (4.12) in theorem 4 that residuals for stages beyond the first contain terms for which it is difficult to guarantee boundedness, even when the first stage residuals are controlled. We attempt to overcome this fact, by extending the basis $\mathbf{V}_m$ at the i'th stage to include the intermediate RHS values F_j for $j = 2, \ldots, i$ so that

$$(F_i - \mathbf{V}_{m:i}\psi_i) = 0, \quad \text{for } i = 1, \ldots, s.$$

The matrices $\mathbf{V}_{m;i}$ and $\mathbf{H}_{m;i}$ can be constructed iteratively, extending the basis $\mathbf{V}_{m;i-1}$ with F_i at each stage to form $\mathbf{V}_{m;i}$ and $\mathbf{H}_{m;i}$ as outlined in the previous section. One difficulty of extending the basis in this way, is that the $\mathbf{LU}$-decomposition of the left-hand-side of (4.2a), $(\mathbf{I} - h\gamma\mathbf{H})$ cannot be reused. This problem can be overcome through the use of an update to the $\mathbf{L}$ and $\mathbf{U}$ matrices. The $\mathbf{LU}$-decomposition is constructed to solve

$$\begin{aligned} (\mathbf{I} - h\gamma\mathbf{H})\, x &= b \\ \mathbf{P}\,(\mathbf{I} - h\gamma\mathbf{H})\, x &= \mathbf{P}b \\ \mathbf{LU}x &= \mathbf{P}b \end{aligned}$$

so that

$$\begin{aligned} \mathbf{P}\,(\mathbf{I} - h\gamma\mathbf{H}) &= \mathbf{LU} \\ \mathbf{U} &= \mathbf{L}^{-1}\mathbf{P}\,(\mathbf{I} - h\gamma\mathbf{H}) \end{aligned}$$

and so exploiting the upper-Hessenberg structure of $\mathbf{H}$, the LU-decomposition can be updated as

$$\begin{bmatrix} \mathbf{u}_{1,m+1} \\ \vdots \\ \mathbf{u}_{m,m+1} \end{bmatrix} = -h\gamma\mathbf{L}_m^{-1}\mathbf{P}_m \begin{bmatrix} \mathbf{h}_{1,m+1} \\ \vdots \\ \mathbf{h}_{m,m+1} \end{bmatrix}, \quad \mathbf{u}_{m+1,m+1} = (1 - h\gamma\mathbf{h}_{m+1,m+1})\,, \quad \mathbf{l}_{m+1,m+1} = 1$$

Theorem 5 (Residual of the ith stage with extended basis)**.** *When the time integration scheme* (4.2) *is implemented with the basis extended (via the process described in section 4.4.1) at each stage to include the current right-hand-side vector, F_i, then the residual of the ith stage when using an m-dimensional Krylov space is given by:*

$$r_{m;i} = -h\left(\mathbf{I} - \mathbf{V}_m\,\mathbf{V}_m^T\right)\mathbf{J}v_m e_m^T \sum_{j=1}^{i} \gamma_{ij}\widehat{\lambda}_j - h\left(\mathbf{I} - \mathbf{V}_{m;i}\mathbf{V}_{m;i}^T\right)\mathbf{J}\sum_{k=1}^{i-1} v_{m+k}e_{m+k}^T \sum_{j=1}^{i} \gamma_{ij}\widehat{\lambda}_j, \tag{4.17}$$

where $e_j \in \mathbb{R}^{(m+i-1)}$ are canonical basis vectors, $\widehat{\lambda}_j = \left[\lambda_j^T, 0, \ldots, 0\right]^T \in \mathbb{R}^{(m+i-1)}$, and $\mathbf{V}_{m;1} = \mathbf{V}_m$.

Proof. The general stage equations for the method are:

$$\begin{aligned}
k_i &= hF_i + h\mathbf{J}\sum_{j=1}^{i}\gamma_{ij}k_j + r_{m;i}\\
\widehat{\lambda}_i &= h\psi_i + h\mathbf{H}_{m;i}\sum_{j=1}^{i-1}\gamma_{ij}\widehat{\lambda}_j\\
\psi_i &= \mathbf{V}_{m;i}^T F_i\\
k_i &= \mathbf{V}_{m;i}\widehat{\lambda}_i + h\left(F_i - \mathbf{V}_{m;i}\psi_i\right) = \mathbf{V}_{m;i}\widehat{\lambda}_i
\end{aligned}$$

Additionally, we define $\widehat{\lambda}_j \in \mathbb{R}^{(m+i-1)}$ as follows:

$$\widehat{\lambda}_j = \begin{cases} \left[\lambda_j^T, 0, \ldots, 0\right]^T, & j < i,\\ \lambda_j, & j = i,\end{cases}$$

where reduced space solutions $\lambda_j \in \mathbb{R}^{(m+j-1)}$ from previous stages are extended with zeros to match the dimension of the current stage system.

Starting from the full space equation, we make substitutions for F_i and the k's:

$$r_{m;i} = \mathbf{V}_{m;i}\widehat{\lambda}_i - h\mathbf{V}_{m;i}\psi_i - h\mathbf{J}\mathbf{V}_{m;i}\sum_{j=1}^{i}\gamma_{ij}\widehat{\lambda}_j,$$

Making use of lemma 7, we make substitutions for appearances of $\mathbf{J}\mathbf{V}_{m;i}$:

$$\begin{aligned}
r_{m;i} = \mathbf{V}_{m;i}\widehat{\lambda}_i - h\mathbf{V}_{m;i}\psi_i - h\mathbf{V}_{m;i}\mathbf{H}_{m;i}\sum_{j=1}^{i}\gamma_{i,j}\widehat{\lambda}_j - \left(\mathbf{I} - \mathbf{V}_m\mathbf{V}_m^T\right)\mathbf{J}v_m e_m^T\sum_{j=1}^{i}\gamma_{i,j}\widehat{\lambda}_j\\
- \left(\mathbf{I} - \mathbf{V}_{m;i}\mathbf{V}_{m;i}^T\right)\mathbf{J}\sum_{k=1}^{i-1}v_{m+k}e_{m+k}^T\sum_{j=1}^{i}\gamma_{i,j}\widehat{\lambda}_j
\end{aligned}$$

Collecting terms and rearranging allows us to find and cancel the terms from the reduced space equation:

$$\begin{aligned}
r_{m;i} = \mathbf{V}_{m;i}\underbrace{\left[\widehat{\lambda}_i - h\psi_i - h\mathbf{H}_{m;i}\sum_{j=1}^{i}\gamma_{i,j}\widehat{\lambda}_j\right]}_{=0} - \left(\mathbf{I} - \mathbf{V}_m\mathbf{V}_m^T\right)\mathbf{J}v_m e_m^T\sum_{j=1}^{i}\gamma_{i,j}\widehat{\lambda}_j\\
- \left(\mathbf{I} - \mathbf{V}_{m;i}\mathbf{V}_{m;i}^T\right)\mathbf{J}\sum_{k=1}^{i-1}v_{m+k}e_{m+k}^T\sum_{j=1}^{i}\gamma_{i,j}\widehat{\lambda}_j
\end{aligned}$$

$$r_{m;i} = -h\left(\mathbf{I} - \mathbf{V}_m\,\mathbf{V}_m^T\right)\mathbf{J}v_m e_m^T\sum_{j=1}^{i}\gamma_{ij}\widehat{\lambda}_j - h\left(\mathbf{I} - \mathbf{V}_{m;i}\mathbf{V}_{m;i}^T\right)\mathbf{J}\sum_{k=1}^{i-1}v_{m+k}e_{m+k}^T\sum_{j=1}^{i}\gamma_{ij}\widehat{\lambda}_j.$$

□

The new residual coming from the extended basis (4.17) appears to be a substantial improvement over the residuals from the vanilla method. Since here the residuals consist exclusively of terms projected out of the span of $\mathbf{V}_m$ and $\mathbf{V}_{m;s}$.

4.5 Rosenbrock-Krylov methods for Differential Algebraic Equations

An alternative path to stability is to examine the effect of a relatively small number of "infinitely stiff" modes in the initial value problem (4.1). To do so we will examine systems coming from a singular perturbation of the differential algebraic equation

$$f(y, z, \dot{y}, t) = 0\,, \quad y(t_0) = 0\,, \quad t \in \mathbb{R}\,,\ y \in \mathbb{R}^N\,,\ z \in \mathbb{R}^L. \tag{4.18}$$

We consider in this paper only the seperable and autonomous form of (4.18)

$$\begin{aligned} \dot{y} &= f(y, z), \quad y(t_0) = y_0 && (4.19\text{a}) \\ 0 &= g(y, z), \quad z(t_0) = z_0. && (4.19\text{b}) \end{aligned}$$

Where a singular perturbation of (4.19) takes the form:

$$\begin{bmatrix} I & 0 \\ 0 & \varepsilon I \end{bmatrix} \dot{X} = \begin{bmatrix} \dot{y} \\ \varepsilon \dot{z} \end{bmatrix} = \begin{bmatrix} f(y, z) \\ g(y, z) \end{bmatrix} = F(X), \tag{4.20}$$

so that in the limit as $\varepsilon \to 0$ equation (4.20) is exactly (4.19). It is clear from equation (4.20) that components corresponding to the algebraic variable z are extremely stiff, and this stiffness only increases as $\varepsilon \to 0$.

A Rosenbrock scheme applied to (4.20) has general form

$$\begin{bmatrix} k_i \\ l_i \end{bmatrix} = h \begin{bmatrix} f(v_i, w_i) \\ \varepsilon^{-1} g(v_i, w_i) \end{bmatrix} + h \begin{bmatrix} f_y & f_z \\ \varepsilon^{-1} g_y & \varepsilon^{-1} g_z \end{bmatrix} (y_0, z_0) \sum_{j=1}^{i} \gamma_{i,j} \begin{bmatrix} k_j \\ l_j \end{bmatrix} \tag{4.21a}$$

$$\begin{bmatrix} v_i \\ w_i \end{bmatrix} = \begin{bmatrix} y_n \\ z_n \end{bmatrix} + \sum_{j=1}^{i-1} \alpha_{i,j} \begin{bmatrix} k_j \\ l_j \end{bmatrix}, \quad \begin{bmatrix} y_{n+1} \\ z_{n+1} \end{bmatrix} = \begin{bmatrix} y_n \\ z_n \end{bmatrix} + \sum_{i=1}^{s} \begin{bmatrix} k_i \\ l_i \end{bmatrix}. \tag{4.21b}$$

Alternatively, multiplying the second line of (4.21a) by ε and then letting $\varepsilon = 0$ we obtain

$$\begin{bmatrix} k_i \\ 0 \end{bmatrix} = h \begin{bmatrix} f(v_i, w_i) \\ g(v_i, w_i) \end{bmatrix} + h \begin{bmatrix} f_y & f_z \\ g_y & g_z \end{bmatrix} (y_0, z_0) \sum_{j=1}^{i} \gamma_{i,j} \begin{bmatrix} k_j \\ l_j \end{bmatrix} \tag{4.22a}$$

$$\begin{bmatrix} v_i \\ w_i \end{bmatrix} = \begin{bmatrix} y_n \\ z_n \end{bmatrix} + \sum_{j=1}^{i-1} \alpha_{i,j} \begin{bmatrix} k_j \\ l_j \end{bmatrix}, \quad \begin{bmatrix} y_{n+1} \\ z_{n+1} \end{bmatrix} = \begin{bmatrix} y_n \\ z_n \end{bmatrix} + \sum_{i=1}^{s} \begin{bmatrix} k_i \\ l_i \end{bmatrix}. \tag{4.22b}$$

4.5.1 A Krylov based approach

We seek to construct order conditions for the scheme (4.22) using a K-type framework, similar to that derived for ODE systems for Rosenbrock methods in [56] and exponential methods in [65]. In this framework we make use of a specific, Krylov based approximation of the Jacobian matrix.

We then construct the m-dimensional Krylov space $\mathcal{K}_m(\mathbf{J}, F) = span\,\{F, \mathbf{J}F, \ldots, \mathbf{J}^{m-1}F\}$, with $m > L$, via a modified Arnoldi process to generate the orthonormal basis matrix $\mathbf{V}$ and upper-Hessenberg matrix $\mathbf{H}$ which satisfy

$$\mathbf{V}^T\,\mathbf{V} = \mathbf{I}, \quad span\,\{\mathbf{V}\} = \mathcal{K}_m(J, F), \quad \mathbf{H} = \mathbf{V}^T\,\mathbf{J}\,\mathbf{V}$$

with Jacobian matrix

$$\mathbf{J} = \begin{bmatrix} f_y & f_z \\ g_y & g_z \end{bmatrix}. \tag{4.23}$$

Finally, we build the Krylov approximation matrix, $\mathbf{A}$, in the usual way, so that

$$\mathbf{A} = \mathbf{V}\,\mathbf{H}\,\mathbf{V}^T = \mathbf{V}\,\mathbf{V}^T\,\mathbf{J}\,\mathbf{V}\,\mathbf{V}^T \tag{4.24}$$

Wensch has shown constructively in [16], with only minor modification to the problem statement, that building $\mathcal{K}_m(J, F)$ in this way leads to a basis matrix $\mathbf{V}$ of the form

$$\mathbf{V} = \begin{bmatrix} 0 & \mathbf{V}_f \\ \mathbf{V}_g & 0 \end{bmatrix} \tag{4.25}$$

where

$$\mathbf{V}_f \in \mathbb{R}^{N\times(m-L)}, \quad \mathbf{V}_g \in \mathbb{R}^{L\times L}, \quad \text{and } span\,\{V_g\} = \mathbb{R}^L$$

so that the entire algebraic space is covered. Inserting equation (4.25) into (4.24) leads to

$$\mathbf{A} = \begin{bmatrix} \mathbf{V}_f\,\mathbf{V}_f^T & 0 \\ 0 & \mathbf{V}_g\,\mathbf{V}_g^T \end{bmatrix} \begin{bmatrix} f_y & f_z \\ g_y & g_z \end{bmatrix} \begin{bmatrix} \mathbf{V}_f\,\mathbf{V}_f^T & 0 \\ 0 & \mathbf{V}_g\,\mathbf{V}_g^T \end{bmatrix}.$$

So that the final form of the Krylov approximation is given as:

$$\mathbf{A} = \begin{bmatrix} \mathbf{V}_f\,\mathbf{V}_f^T\,f_y\,\mathbf{V}_f\,\mathbf{V}_f^T & \mathbf{V}_f\,\mathbf{V}_f^T\,f_z\,\mathbf{V}_g\,\mathbf{V}_g^T \\ \mathbf{V}_g\,\mathbf{V}_g^T\,g_y\,\mathbf{V}_f\,\mathbf{V}_f^T & \mathbf{V}_g\,\mathbf{V}_g^T\,g_z\,\mathbf{V}_g\,\mathbf{V}_g^T \end{bmatrix} = \begin{bmatrix} \mathbf{V}_f\,\mathbf{V}_f^T\,f_y\,\mathbf{V}_f\,\mathbf{V}_f^T & \mathbf{V}_f\,\mathbf{V}_f^T\,f_z \\ g_y\,\mathbf{V}_f\,\mathbf{V}_f^T & g_z \end{bmatrix} \tag{4.26}$$

4.5.2 Order conditions for ROK-DAE method.

The numerical solution computed using the classical Rosenbrock method (4.22) can be expanded in a Taylor series [11], as

$$k_i^{(q)} = q \sum_{m+n\geq 2} \frac{\partial^{m+n} f(y_0,z_0)}{\partial y^m \partial z^n} \left(\sum_{j=1}^{i-1} \alpha_{ij} k_j^{(\mu_1)}, \ldots, \sum_{j=1}^{i-1} \alpha_{ij} l_j^{(\nu_1)}, \ldots \right) + q\,(f_y)_0 \sum_{j=1}^{i} \beta_{ij} k_j^{(q-1)} + q\,(f_z)_0 \sum_{j=1}^{i} \beta_{ij} l_j^{(q-1)} \quad (4.27a)$$

$$l_i^{(q)} = (-g_z)_0^{-1} \sum_{j=1}^{i} \omega_{ij} \sum_{m+n\geq 2} \frac{\partial^{m+n} g(y_0,z_0)}{\partial y^m \partial z^n} \left(\sum_{k=1}^{j-1} \alpha_{jk} k_k^{(\mu_1)}, \ldots, \sum_{k=1}^{j-1} \alpha_{jk} l_k^{(\nu_1)}, \ldots \right) + \left(\left(-g_z^{-1}\right) g_y \right)_0 k_i^{(q)}. \quad (4.27b)$$

In this formulation the linear term makes use of the exact Jacobian (4.23). If we instead make use of a W-type strategy, where the Jacobian is approximated using a form similar to the Krylov approximation matrix, so that

$$\mathbf{A} = \begin{bmatrix} \mathbf{A}_1 & \mathbf{A}_2 \\ \mathbf{A}_3 & g_z \end{bmatrix}. \quad (4.28)$$

we can expand the solution as

$$k_i^{(q)} = q \sum_{m+n\geq 2} \frac{\partial^{m+n} f(y_0,z_0)}{\partial y^m \partial z^n} \left(\sum_{j=1}^{i-1} \alpha_{ij} k_j^{(\mu_1)}, \ldots, \sum_{j=1}^{i-1} \alpha_{ij} l_j^{(\nu_1)}, \ldots \right) + q\,(f_y)_0 \sum_{j=1}^{i} \alpha_{ij} k_j^{(q-1)} + q\,(\mathbf{A}_1)_0 \sum_{j=1}^{i} \gamma_{ij} k_j^{(q-1)} + q\,(f_z)_0 \sum_{j=1}^{i} \alpha_{ij} l_j^{(q-1)} + q\,(\mathbf{A}_2)_0 \sum_{j=1}^{i} \gamma_{ij} l_j^{(q-1)} \quad (4.29a)$$

$$l_i^{(q)} = (-g_z)_0^{-1} \sum_{j=1}^{i} \omega_{ij} \sum_{m+n\geq 2} \frac{\partial^{m+n} g(y_0,z_0)}{\partial y^m \partial z^n} \left(\sum_{k=1}^{j-1} \alpha_{jk} k_k^{(\mu_1)}, \ldots, \sum_{k=1}^{j-1} \alpha_{jk} l_k^{(\nu_1)}, \ldots \right) + \left(\left(-g_z^{-1}\right) g_y \right)_0 \sum_{j=1}^{i} \omega_{ij} \sum_{k=1}^{j-1} \alpha_{jk} k_i^{(q)} + (-g_z)_0^{-1} \mathbf{A}_3 \sum_{j=1}^{i} \omega_{ij} \sum_{k=1}^{j} \gamma_{jk} k_i^{(q)}. \quad (4.29b)$$

Equation (4.29) looks very similar to (4.27), with the exception that the terms coming from the linear piece of (4.22) are now split into contributions coming from actual derivatives of the DAE right-hand-sides, and contributions coming from the appearance of the Krylov approximation matrix, $\mathbf{A}$.

Butcher trees for the numerical solution expansion (4.29) can be constructed from a simple extension of the Butcher trees for (4.27), which are derived and presented in [11, Definition 4.1]. The Butcher trees for the classical Rosenbrock scheme (4.27) are comprised of two node colors, meagre and fat, corresponding to $f(y,z)$ and $g(y,z)$ and their derivatives, respectively. The Butcher trees for the K-type scheme (4.29) are instead comprised of four node colors, where square nodes (both meagre and fat) are used to represent an appearance of the submatrices of the approximate Jacobian matrix (4.28).

Definition 6 (Structure of butcher trees for a Rosenbrock-W-DAE scheme.). *Let $DATW = DATW_y \cup DATW_z$ denote the set of trees defined recursively by*

a) $\tau_y \in DATW_y$, $\tau_z \in DATW_z$*;*

b) $[t_1, \dots, t_m, u_1, \dots, u_n]_{\bullet} \in DATW_y$
if $t_1, \dots t_m, \in DATW_y$, *and* $u_1, \dots, u_n \in DATW_z$*;*

c) $[t_1, \dots, t_m, u_1, \dots, u_n]_{\circ} \in DATW_z$
if $t_1, \dots t_m, \in DATW_y$, $u_1, \dots, u_n \in DATW_z$, *and* $(m,n) \neq (0,1)$*;*

d) $[t_1]_{\blacksquare} \in DATW_y$, *and* $[u_1]_{\blacksquare} \in DATW_y$
if $t_1 \in DATW_y$, *and* $u_1 \in DATW_z$*;*

e) $[t_1]_{\square} \in DATW_z$
if $t_1 \in DATW_y$.

Where a visual representation of an $DATW$ tree can be constructed in the usual way using the recursive bracket notation, so that $[t_1, \dots t_m, u_1 \dots u_n]_{\bullet}$ can be constructed by connecting the roots of the trees $t_1, \dots t_m, u_1, \dots u_n$ by $m+n$ branches to a new root, a meagre circle. Similarly $[t_1, \dots t_m, u_1 \dots u_n]_{\circ}$ connects the trees $t_1, \dots t_m, u_1, \dots u_n$ to a new root, a fat circle, and $[\cdot]_{\blacksquare}$ and $[\cdot]_{\square}$ are formed by attaching the argument to a new root, a meagre or fat square respectively. Figure 4.1 shows an example of this notation, and gives a definition for the trees τ_y and τ_z.

Definition 7 (Elementary differentials corresponding to $DATW$ trees.). *The elementary differential corresponding to $DATW$ trees for* (4.29) *are*

a) $F(\tau_y) = f$, $F(\tau_z) = (-g_z^{-1})\, g_y f$*;*

b) $F(t) = \frac{\partial^{m+n} f}{\partial y^m \partial z^n} \left(F(t_1), \dots F(t_m), F(u_1) \dots F(u_n)\right)$
if $t = [t_1, \dots, t_m, u_1, \dots, u_n]_{\bullet} \in DATW_y$*;*

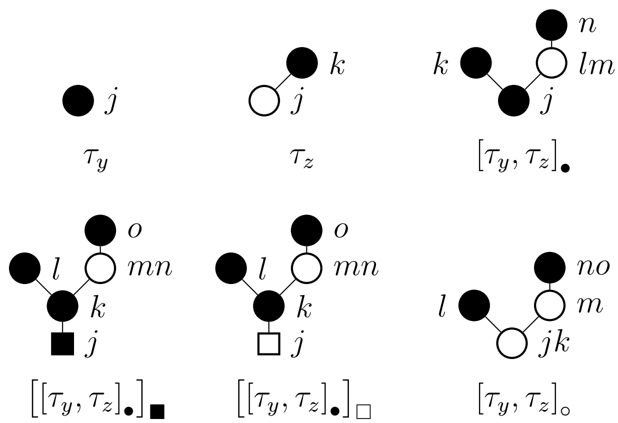


Figure 4.1: Illustration of the recursive definition of $DATW$-trees.

c) $F(u) = (-g_z)^{-1} \frac{\partial^{m+n} g}{\partial y^m \partial z^n} \left(F(t_1), \ldots F(t_m), F(u_1) \ldots F(u_n)\right)$
if $u = [t_1, \ldots, t_m, u_1, \ldots, u_n]_\circ \in DATW_z$*;*

d) $F(t) = \mathbf{A}_1 F(t_1)$
if $t = [t_1]_\blacksquare \in DATW_y$*;*

e) $F(t) = \mathbf{A}_2 F(u_1)$
if $t = [u_1]_\blacksquare \in DATW_z$*;*

f) $F(u) = (-g_z)^{-1} \mathbf{A}_3 F(t_1)$
if $u = [t_1]_\square \in DATW_y$.

The visual representation of a $DATW$ tree can be mapped to the left hand side of its corresponding order condition, in a manner very similar to that for DAT trees in [11], by attaching to each meagre vertex one summation index, and two summation indices to each fat vertex and summing over all indices of a product with factors

b_i if "i" is the index of the root (the lower index if the root is fat);
$\alpha_{i,j}$ if "j" lies directly above "i" and "i" is multiply branched;
$\alpha_{i,j}$ if "j" lies directly above "i" and "i" is singly branched and corresponds to a meagre vertex;
$\gamma_{i,j}$ if "j" lies directly above "i" and "i" is singly branched and corresponds to a fat vertex;
$\omega_{i,j}$ if "i, j" are the two indices of a fat vertex ("i" below "j"),

with

$$\beta_{i,j} = \alpha_{i,j} + \gamma_{i,j}, \quad \text{and} \quad (\omega_{i,j}) = (\beta_{i,j})^{-1}.$$

Then the order conditions for a ROW-DAE method with approximate Jacobian of the form (4.28), can be formed using the idea of W-methods, in which contributions to the error coming from trees in which the approximate Jacobian appears are set to zero. So that the left hand side of the order condition is formed as above, while the right hand side is formed exactly as in [11, Theorem 4.8] for $t \in DAT_y$ or $u \in DAT_z$, or is identically zero for $t \in \{DATW_y \setminus DAT_y\}$ or $u \in \{DATW_z \setminus DAT_z\}$.

4.5.3 Reduction of order conditions

To begin, we reiterate that the column space of $\mathbf{V}_g$ spans $\mathbb{R}^L$ as long as the generated Krylov space, $\mathcal{K}_m(\mathbf{J}, F)$, has dimension greater than L due to the assumption that the problem is index one, that g_z is invertible, and the results of Wensch in [16]. Moreover if we introduce a new, non-orthogonal, $(m-L)$-dimensional basis

$$\mathbf{U} \in \mathbb{R}^{N \times (m-L)}$$

with

$$u_n = f - f_z g_z^{-1} g \qquad \text{for } n = 1 \tag{4.30a}$$
$$u_n = \left(f_y - f_z g_z^{-1} g_y\right) u_{n-1} \qquad \text{for } n = 2, \dots, m-L \tag{4.30b}$$

then $span\,\{\mathbf{V}_f\} = span\,\{\mathbf{U}\}$ according to [16]. Additionally, if the initial value of the DAE system (4.18) is consistent, then $g = 0$ and equation (4.30) reduces to

$$u_n = f \qquad \text{for } n = 1 \tag{4.31a}$$
$$u_n = \left(f_y - f_z g_z^{-1} g_y\right) u_{n-1} \qquad \text{for } n = 2, \dots, m-L. \tag{4.31b}$$

The first three basis vectors of the differential space can be given as:

$$\begin{aligned} u_1 &= f \\ u_2 &= f_y f - f_z g_z^{-1} g_y f \\ u_3 &= f_y f_y f - f_y f_z g_z^{-1} g_y f - f_z g_z^{-1} g_y f_y f + f_z g_z^{-1} g_y f_z g_z^{-1} g_y f \end{aligned}$$

We extend the framework for K-type order conditions developed in [56] making use of equation (4.31) to reduce the order conditions for the W-method given in equation (4.22). To do so we note that since $\mathbf{V}_f$ is an orthonormal basis matrix, then $\mathbf{V}_f\,\mathbf{V}_f^T$ is an orthogonal projector onto the column space of $\mathbf{V}_f$. We again note that

$$span\,\{\mathbf{V}_f\} = span\,\{\mathbf{U}\}\,, \tag{4.33}$$

so that products containing the submatrices of the Krylov approximation matrix (4.26) can be reduced. For example, consider the elementary differential coming from the tree τ_1 in figure 4.2

$$\mathbf{A}_1 f = \mathbf{V}_f\,\mathbf{V}_f^T f_y \mathbf{V}_f\,\mathbf{V}_f^T f = \mathbf{V}_f\,\mathbf{V}_f^T f_y f$$

since $f \in span\,\{\mathbf{V}_f\}$. Additionally, by identifying a compatible tree, specifically τ_2 in figure 4.2, and taking a sum of the elementary differential represented by the two trees we see that

$$\mathbf{V}_f\,\mathbf{V}_f^T \left(f_y f - f_z g_z^{-1} g_y f\right) = f_y f + f_z g_z^{-1} g_y f,$$

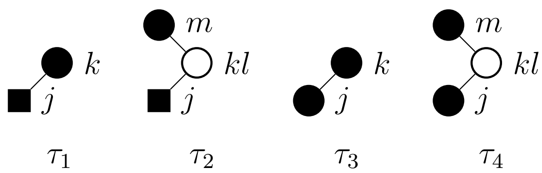


Figure 4.2: Illustration of tree recombination procedure

is exactly the sum of the elementary differentials represented by the trees τ_3 and τ_4 in figure 4.2. In this way, what once required four terms in the series expansion of the numerical solution, can now be represented by only two.

Repeated application of this procedure, making use of the additional third order trees provided by u_3 makes recoloring from W- to K- type order conditions possible. Moreover, when these trees are recolored terms containing $\beta_{ij} = \alpha_{ij} + \gamma_{ij}$ appear, which can be further exploited to reduce the number of order conditions dramatically as in [11].

4.6 Numerical Results

4.6.1 Non-stiff case: shallow water equations

We examine relative performance of the methods on the shallow water equations.The shallow water equations in spherical coordinates are

$$\begin{aligned}
\frac{\partial u}{\partial t} + \frac{1}{a\cos\theta}\left(u\frac{\partial u}{\partial\lambda} + v\cos\theta\frac{\partial u}{\partial\theta}\right) - \left(f + \frac{u\tan\theta}{a}\right)a + \frac{g}{a\cos\theta}\frac{\partial h}{\partial\lambda} &= 0 \qquad (4.34a)\\
\frac{\partial v}{\partial t} + \frac{1}{a\cos\theta}\left(u\frac{\partial v}{\partial\lambda} + v\cos\theta\frac{\partial v}{\partial\theta}\right) + \left(f + \frac{u\tan\theta}{a}\right)u + \frac{g}{a}\frac{\partial h}{\partial\theta} &= 0 \qquad (4.34b)\\
\frac{\partial h}{\partial t} + \frac{1}{a\cos\theta}\left(\frac{\partial(hu)}{\partial\lambda} + \frac{\partial(hv\cos\theta)}{\partial\theta}\right) &= 0. \qquad (4.34c)
\end{aligned}$$

Where $f = 2\Omega\sin\theta$, h is the height of the atmosphere, u is the zonal wind component, v is the meridional wind component, θ and λ are the latitudinal and longitudinal directions, a is the radius of the earth, Ω is the rotational velocity of the earth, and g is the gravitational constant. The space discretization is performed using the unstaggered Turkel-Zwas scheme [67, 68], with 72 nodes in the longitudinal direction and 36 nodes in the latitudinal direction.

Figure 4.3 shows the relative performance of a fourth order ROK method with varying basis sizes, applied to the shallow water equations. Figure 4.3(a) compares methods making use of a fixed basis size at each timestep, and illustrates the inefficiency of adding vectors to the basis for non stiff problems. Here, four basis vectors is enough to resolve the stiff modes of the system and additional vectors only add to the cost of each timestep without providing

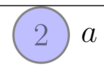
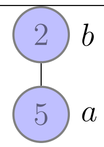
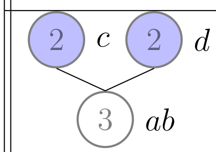
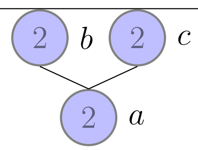
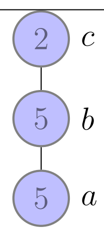
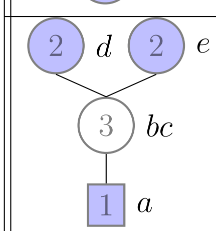
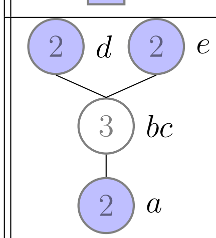
	f	$\sum b_a$
	$f_y f$	$\sum b_a \beta_{ab}$
	$(-g_z)^{-1} g_{yy}(f,f)$	$\sum b_a \omega_{ab} \alpha_{bc} \alpha_{bd}$
	$f_{yy}(f,f)$	$\sum b_a \alpha_{ab} \alpha_{ac}$
	$f_y f_y f$	$\sum b_a \beta_{ab} \beta_{bc}$
	$A_2 (-g_z)^{-1} g_{yy}(f,f)$	$\sum b_a \gamma_{ab} \omega_{bc} \alpha_{cd} \alpha_{ce}$
	$f_z (-g_z)^{-1} g_{yy}(f,f)$	$\sum b_a \alpha_{ab} \omega_{bc} \alpha_{cd} \alpha_{ce}$

Table 4.1: Trees, elementary differentials, and left hand side of the order condition for a third order ROK-DAE method.

additional benefit. Similarly, figure 4.3(b) compares methods making use of an adaptive basis size, where the basis is chosen to control the residual of the first stage. We see that the method making use of low tolerances is most efficient, since not many basis vectors are required to resolve the stiff modes. In both cases extending the basis to include the right-hand-side vector at each stage does not provide a substantial benefit, once again due to the lack of stiffness for this problem.

4.6.2 Stiff case: CBM-IV

Here we give some results for ROK methods applied to a stiff system of ODEs coming from a KPP MATLAB implementation of the CBM-IV model [53]. This problem is based on the Carbon Bond Mechanism IV (CBM-IV), consisting of 32 chemical species involved in 70 thermal and 11 photolytic reactions [54].

Figure 4.4 shows the relative performance of a fourth order ROK method with varying tolerances for the first stage residual applied to the CBM-IV test model. We present here only methods using adaptive basis sizes, since methods using a fixed strategy do not converge for most tolerance settings. There is clearly some benefit to the extension of the basis to include the right-hand-side vector at each stage, since these implementations are able to resolve solutions with lower accuracy, where the methods without extension fail to converge for these integration tolerances.

4.6.3 Varying stiffness: the Allen-Cahn problem

For further performance comparison we consider the two-dimensional Allen-Cahn system, a parabolic partial differential equation

$$\frac{\partial}{\partial t}u = \alpha \nabla^2 u + \gamma \left(u - u^3\right), \quad (x,y) \in [0,1] \times [0,1], \quad t \in [0, 0.2], \tag{4.35}$$

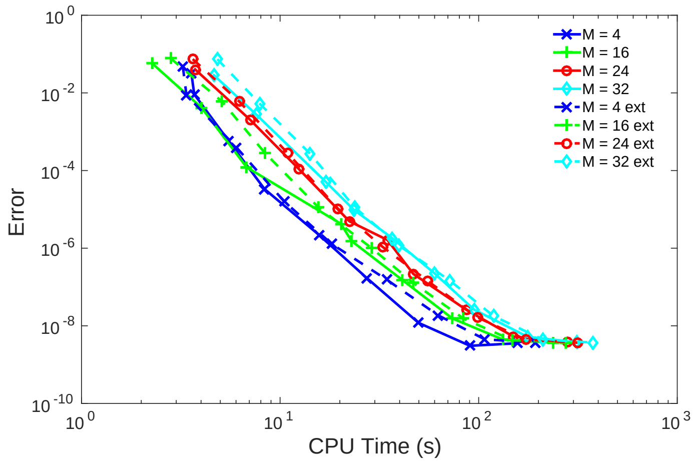
with α set to 0.1 or 1.0 and $\gamma = 1.0$. The problem has homogeneous Neumann boundary conditions and the initial solution $u(t=0) = 0.4 + 0.1(x+y) + 0.1\sin(10x)\sin(20y)$.

Figure 4.5 compares the relative performance of a fourth order ROK method with varying tolerances for the residual of the first stage applied to the Allen-Cahn problem. Here we have varied the problem from relatively nonstiff in figure 4.5(a) to stiff in 4.5(c) by varying both the spatial resolution and size of the diffusion parameter α. What we see is that there is a decided impact on the efficiency of the integrator, when using larger basis sizes, and a particularly strong impact from the extension of the basis at each stage, especially as the stiffness is increased.

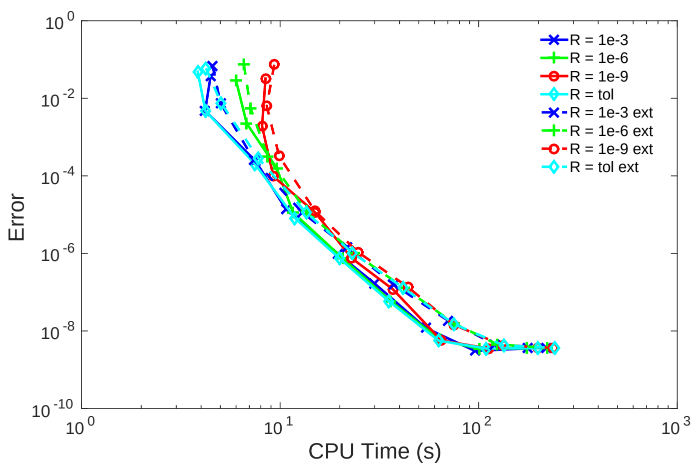
4.7 Conclusions

We have presented an initial exploration of the complicated question of stability for Rosenbrock-Krylov methods. Classification of stability for these schemes is closely tied to similar questions for Rosenbrock-W and exponential schemes in which approximate methods are used. The linear stability analysis of Rosenbrock-Krylov methods lead us to two practical methods of improving the stability of these schemes, through a direct control of the stage residuals. Emprirical results, show significant improvement in the efficiencey of Rosenbrock-Krylov methods making use of an extended basis. Additionally, we have presented a framework for constructing stiff order conditions for Rosenbrock-Krylov methods applied to index-1 differential algebraic equations.

The linear stability analysis presented is suitable for both Rosenbrock-W and Rosenbrock-Krylov schemes. Future work that attempts to exploit the special structure of the Krylov approximation matrix and the K-type order condition theory in the context of stability may be a valuable step torwards more efficient schemes.



(a) Fixed basis size



(b) Adaptive basis size

Figure 4.3: Work-precision diagram for a Rosenbrock-Krylov method applied to the shallow water equations (4.34a).

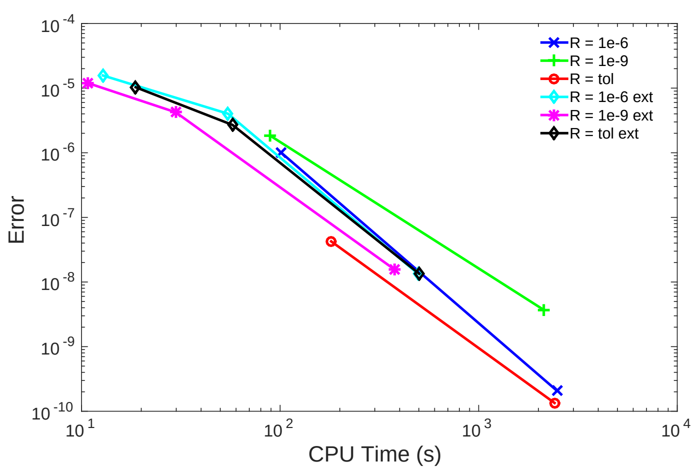
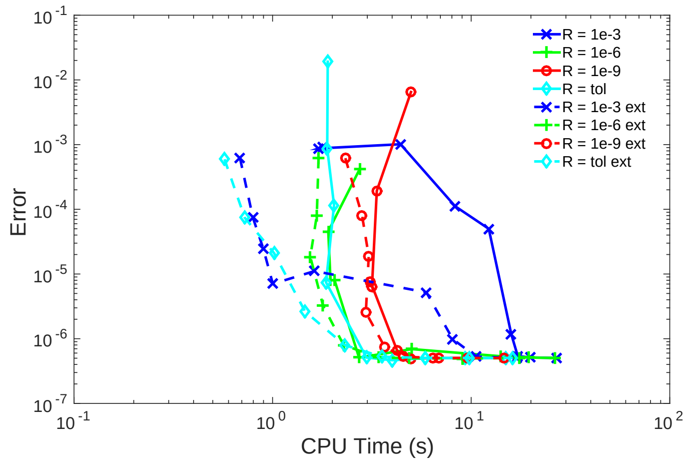
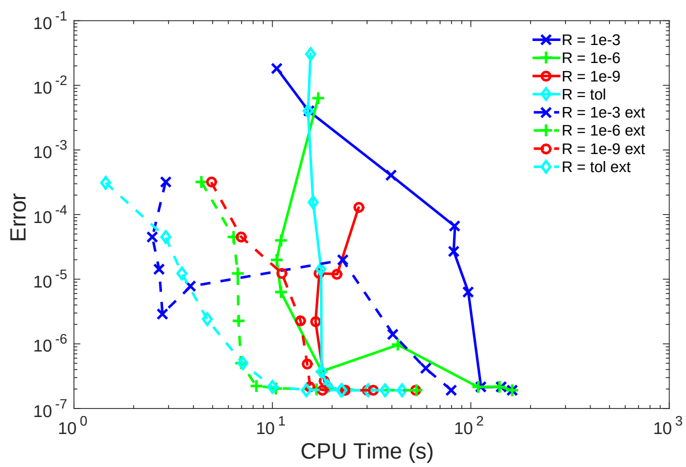


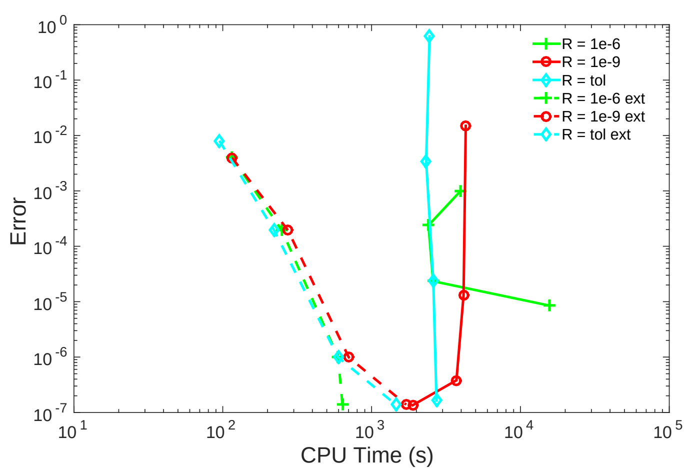
Figure 4.4: Precision Diagram for CBM-IV test problem, using adaptive basis size.



(a) $N = 64 \times 64$, $\alpha = 0.1$



(b) $N = 64 \times 64$, $\alpha = 1.0$



(c) $N = 256 \times 256$, $\alpha = 0.1$

Figure 4.5: Work-precision diagram for a Rosenbrock-Krylov method applied to the Allen-Cahn test problem (4.35).

Chapter 5

Linearly-Implicit Runge-Kutta-W Methods for Large Systems of Differential Equations

5.1 Introduction

A standard approach to the solution of systems of partial differential equations is the method of lines technique, in which a discretization method such as finite differences, finite volumes, or finite elements is used to approximate derivatives in space to arrive at the semi-discrete initial value problem (5.1)

$$\frac{dy}{dt} = F(y)\,, \quad t_0 \le t \le t_F\,, \quad y(t_0) = y_0\,; \qquad y(t), \quad F(y) \in \mathbb{R}^N\,. \tag{5.1}$$

The system (5.1) can be evolved through time using a time integration scheme to approximate solutions at discrete times $t_n < t_i < t_F$. For problems with stiff dynamics, or where mesh refinement leads to unfortunate Courant-Friedrichs-Lewy (CFL) numbers, explicit methods may not be suitable.

Implicit methods, such as Backwards Differentation or Runge-Kutta methods improve upon the stability of explicit methods, at the cost of requiring one or more non-linear solves at each timestep. The approximation of solutions of these nonlinear systems represents the bulk of the computational cost of the time integration process.

Linearly implicit methods, such as Rosenbrock [11, section IV.7] or linearly-implicit Runge-Kutta [69, 70, 71, 72], schemes replace the need to solve a nonlinear system with the solution of more computationally efficient linear systems. Once again, however, the cost of approximating the solution of these linear systems represents a disproportionately large percentage of the overall method cost. Rosenbrock-W [11, section IV.7] [12] methods attempt to alleviate this burden by allowing for the use of arbitrary approximations, and so permit relatively cheap, and inaccurate, solutions of the linear system while maintaining full order of convergence. Unfortunately, while these methods permit arbitrary approximations, the order condition theory on which they are built does not account for variations in the method of approximation between stages of the method.

For this reason, many approximation techniques may not be available in the context of Rosenbrock-W methods. Primary among them is the use of Krylov based iterative solvers, such as GMRES [13], which computes the solution of several different, nearby linear systems when the iteration procedure is truncated early [73]. Similarly, making use of an approximate matrix factorization (AMF) to accelerate the solution of the linear systems also leads to a reduction in the order of the Rosenbrock-W scheme. Several families of time integration schemes (ROWMAP [38] and Rosenbrock-Krylov [56]) have been constructed to couple Krylov solvers and Rosenbrock-W methods to alleviate order reduction in the case of GMRES like approximations.

Here we present a new family of time integration schemes, called linearly-implicit Runge-Kutta-W (LIRK-W) methods, based on an implicit-explicit (IMEX) [74] approach, which allow for arbitrary, time dependent, and stage varying approximations of the linear systems appearing in the method. The rest of the paper is laid out as follows: In section 5.2 we motivate, and present, the general form of a LIRK-W method; in section 5.2.3 we review the use of AMF to accelerate the solution of linear systems in the context of PDEs; in section 5.3 we present the order condition theory for a LIRK-W method; in section 5.4 we give linear stability results for a LIRK-W method; in section 5.5 we derive a specific LIRK-W method; and finally in section 5.6 we present numerical results.

5.2 Linearly-Implicit Runge-Kutta-W (LIRK-W) Methods

5.2.1 Implicit-explicit Runge Kutta methods

Consider a splitting of the right hand side $F(y)$ of the initial value problem (5.1)

$$\frac{dy}{dt} = F(y) = f(y) + g(y)\,, \quad t_0 \le t \le t_F\,, \tag{5.2}$$

such that $f(y)$ represents slow dynamics, unlikely to impact the stability of the numerical integration, and $g(y)$ contains the fast dynamics. An implicit-explicit Runge-Kutta (IMEX-RK) method applies different discretizations to the the two terms, and integrates the non-stiff component $f(y)$ explicitly and stiff component $g(y)$ implicitly [2]:

$$Y_i = y_n + h \sum_{j=1}^{i-1} a_{i,j}\, f(Y_j) + h \sum_{j=1}^{i} \widehat{a}_{i,j}\, g(Y_j), \quad i = 1, \ldots, s, \tag{5.3a}$$

$$y_{n+1} = y_n + h \sum_{j=1}^{s} b_j\, f(Y_j) + h \sum_{j=1}^{s} \widehat{b}_j\, g(Y_j). \tag{5.3b}$$

The fact that only $g(y)$ is treated implicitly has the benefit of reducing the per-timestep cost as compared to implicit Runge-Kutta methods, since one only solves non-linear systems containing $g(y)$ instead of the entire of $F(y)$. In the same time (5.3) has considerably better stability properties than explicit Runge-Kutta schemes since the stiff dynamics is integrated implicitly.

Unfortunately, the application of (5.3) requires to first partition the system $F(t, y)$ into non-stiff and stiff parts (5.2), and to provide the Jacobian operator corresponding to the stiff term. These steps can be difficult to achieve for systems implemented in large legacy codes.

5.2.2 Arbitrary linear approximations of the stiff term

Here we propose the new class of Linearly-Implicit Runge-Kutta-W (LIRK-W) time integrators that make use of an alternative partitioning, based on a linear/non-linear splitting of the right hand side operator:

$$\frac{dy}{dt} = \mathbf{L}\, y + (F(y) - \mathbf{L}y)\ , \quad t_0 \le t \le t_F\,, \quad y(t_0) = y_n\,; \quad y(t), F(y) \in \mathbb{R}^N\,, \quad \mathbf{L} \in \mathbb{R}^{N\times N}. \tag{5.4}$$

where $\mathbf{L}y$ ideally captures the stiff dynamics of $F(y)$. In this way we will seek to treat implicitly the linear terms $\mathbf{L}$ that capture the stiffness of the system, and to treat explicitly the remaining nonlinear part. The IMEX-RK method (5.3) applied to (5.4), after some rearranging of terms, reads:

$$\left(\mathbf{I} - h\,\widehat{a}_{i,i}\,\mathbf{L}\right)\, Y_i = y_n + h \sum_{j=1}^{i-1} a_{i,j}\, F(Y_j) + h\,\mathbf{L} \sum_{j=1}^{i-1} \left(\widehat{a}_{i,j} - a_{i,j}\right)\ Y_j, \tag{5.5a}$$

$$y_{n+1} = y_n + h \sum_{j=1}^{s} b_j\, F(Y_j) + h\,\mathbf{L} \sum_{j=1}^{s} \left(\widehat{b}_j - b_j\right)\ Y_j. \tag{5.5b}$$

Using the notation

$$\gamma_{i,j} = \left(\widehat{a}_{i,j} - a_{i,j}\right), \quad \text{and} \quad g_i = \left(\widehat{b}_j - b_j\right)$$

in equation (5.5) leads to the standard form of a Linearly-Implicit Runge-Kutta-W (LIRK-W) method:

$$\begin{aligned}
\left(\mathbf{I} - h\,\gamma_{i,i}\,\mathbf{L}\right)\,Y_i &= y_n + h\sum_{j=1}^{i-1} a_{i,j}\,F(Y_j) + h\,\mathbf{L}\sum_{j=1}^{i-1}\gamma_{i,j}\;Y_j, && (5.6a)\\
y_{n+1} &= y_n + h\sum_{j=1}^{s} b_j\,F(Y_j) + h\,\mathbf{L}\sum_{j=1}^{s} g_j\,Y_j. && (5.6b)
\end{aligned}$$

It is clear from equation (5.4) that the linear operator $\mathbf{L}$ can be arbitrary, since collecting terms leads back to the original form of the initial value problem (5.1). Additionally, (5.6) makes exclusive use of the entire right-hand-side vector F(y), there is no splitting necessary. There are several benefits of this framework. First, the stiff terms that are integrated implicitly are linear, so no solutions of nonlinear systems are required. Next, LIRK-W methods will be designed to preserve accuracy for any matrix $\mathbf{L}$. The selected $\mathbf{L}$ is only used to ensure numerical stability; its structure is arbitrary and can be chosen to ensure computational efficiency on the hardware at hand. With this in mind, an ideal choice of the linear operator is one which approximates the stiff dynamics of the system, $\mathbf{L} \approx \partial g(t,y)/\partial y$, in order to improve stability of the numerical integration, and where solutions of linear systems containing $\mathbf{L}$ can be computed efficiently.

LIRK-W methods are similar to Rosenbrock-W schemes, in that they depend only on linear solves and permit arbitrary matrices. However, as will be discussed more thoroughly in section 5.2.5, there are several advantages to the LIRK-W framework.

5.2.3 Approximate Matrix Factorization

Approximate Matrix Factorization (AMF) is often employed to speed up the solution of linear systems required when computing stage vectors (5.3a) or (5.6a) of implicit methods:

$$\left(\mathbf{I} - h\gamma_{i,i}\mathbf{L}\right)\,k_i = d_i. \tag{5.7}$$

AMF splits the matrix $\mathbf{L}$, usually the Jacobian of the right hand side vector in (5.1), into a sum of parts

$$\mathbf{L} = \sum_{r=1}^{R}\mathbf{L}^{\{r\}}, \tag{5.8}$$

and approximates the solution to (5.7) by replacing the matrix as follows:

$$\begin{aligned}
\left(\mathbf{I} - h\,\gamma_{i,i}\,\mathbf{L}\right) \approx \left(\mathbf{I} - h\,\gamma_{i,i}\,\widetilde{\mathbf{L}}\right) &= \prod_{r=1}^{R}\left(\mathbf{I} - h\,\gamma_{i,i}\,\mathbf{L}^{\{r\}}\right),\\
k_i = \left(\mathbf{I} - h\gamma_{i,i}\mathbf{L}\right)^{-1}\,d_i &\approx \prod_{r=1}^{R}\left(\mathbf{I} - h\,\gamma_{i,i}\,\mathbf{L}^{\{r\}}\right)^{-1}\,d_i.
\end{aligned} \tag{5.9}$$

The solution of one large linear system with matrix $\mathbf{L}$ is replaced by solving in succession R small linear systems with matrices $\mathbf{L}^{\{r\}}$, potentially leading to considerable computational savings.

The approximation (5.9) implicitly defines the matrix $\widetilde{\mathbf{L}}$ as

$$\widetilde{\mathbf{L}} = \mathbf{L} + \sum_{k=2}^{R} (-h\gamma_{i,i})^{k-1} \sum_{1\le i_1<i_2<\cdots<i_k\le R} \mathbf{L}_{i_1}\,\mathbf{L}_{i_2}\ldots\mathbf{L}_{i_k}. \tag{5.10}$$

For example, if $\mathbf{L}$ is the discrete two-dimensional Laplacian, then in (5.8) the parts $\mathbf{L}^{\{1\}}$ and $\mathbf{L}^{\{2\}}$ can correspond to derivatives along the x_1 and x_2 directions respectively. In this case the AMF approximation corresponds to the alternating directions factorization [75, 76]

$$\mathbf{I} - h\gamma\widetilde{\mathbf{L}} := \left(\mathbf{I} - h\gamma\mathbf{L}^{\{1\}}\right)\,\left(\mathbf{I} - h\gamma\mathbf{L}^{\{2\}}\right), \qquad \widetilde{\mathbf{L}} = \mathbf{L} - h\gamma\mathbf{L}^{\{1\}}\mathbf{L}^{\{2\}}.$$

In this way it is possible to solve individually much simpler one-dimensional systems corresponding to each individual $\mathbf{L}_i$. An alternating directions factorization is not the only choice possible, but it does motivate many of our design decisions for LIRK-W methods.

5.2.4 Alternative formulations of LIRK-W methods

We note from (5.10) that the approximate matrix $\widetilde{\mathbf{L}}(h)$ depends on the step size, i.e., depends on time. The general form (5.6) making use of a time dependent, stage varying matrix $\mathbf{L}$ can be interpreted in several ways. In the most straightforward variation, we can let there be one $\mathbf{L}_i$ per stage, and make use of these matrices everywhere that $\mathbf{L}$ appears in (5.6), we will call this a type 1 LIRK-W method

$$Y_i = y_n + h\sum_{j=1}^{i-1} a_{i,j}\,F(Y_j) + h\sum_{j=1}^{i} \gamma_{i,j}\,\mathbf{L}_j\,Y_j, \tag{5.11a}$$

$$y_{n+1} = y_n + h\sum_{i=1}^{s} b_i\,F(Y_i) + h\sum_{i=1}^{s} g_i\,\mathbf{L}_i\,Y_i, \tag{5.11b}$$

with $\mathbf{L}_i = \widetilde{\mathbf{L}}(t_n + h\gamma_{i,i})$.

Alternatively, we can attempt to interpret (5.6) to account for an AMF form of $\mathbf{L}$, where there exists two forms of $\mathbf{L}$: equation (5.8), which has clear advantages for a multiplication by $\mathbf{L}$; and equation (5.9), which has clear advantages when used in the inversion of the left hand side of equation (5.6a). A LIRK-W method of type 2 exploits the natural features of an AMF form of $\mathbf{L}$ and has the general form

$$Y_i = y_n + h \sum_{j=1}^{i-1} a_{i,j}\, F(Y_j) + h\, \mathbf{L} \sum_{j=1}^{i-1} \gamma_{i,j}\; Y_j + h\gamma_{i,i} \mathbf{L}_i Y_i, \tag{5.12a}$$

$$y_{n+1} = y_n + h \sum_{j=1}^{s} b_j\, F(Y_j) + h\, \mathbf{L} \sum_{j=1}^{s} g_i\, Y_j, \tag{5.12b}$$

where $\mathbf{L}_i = \mathbf{L}(t_n + h\gamma_{i,i})$.

More variations of (5.6) are possible, and different choices for constructing $\mathbf{L}_i$ may motivate alternatives to type 1 or type 2 LIRK-W schemes.

5.2.5 LIRK-W methods are not Rosenbrock-W methods

As was discussed briefly above, LIRK-W methods share many common traits with a Rosenbrock, or Rosenbrock-W, approach. Primary among them is that LIRK-W methods require only the solution of linear systems, and permit arbitrary matrices $\mathbf{L}$. The defining characteristic of LIRK-W schemes, in contrast to Rosenbrock-W methods, is the allowance for the use of several different approximations in a single timestep. Additionally, the arbitrary linear term, $\mathbf{L}_i$, may depend on time so that it is possible to directly approximate $(\mathbf{I} - h\gamma_{i,i}\mathbf{L}_i)^{-1}$ without violating assumptions inherent in the order condition framework.

One example of this, is the use of a GMRES like approximation to the linear systems $(\mathbf{I} - h\gamma_{i,i}\mathbf{J})\, x = b_i$. Early truncation of the GMRES procedure may lead to loss of order for Rosenbrock-W schemes since each linear system is solved using a different subspace, and is essentially the same as solving s systems with s different left-hand-sides. It is possible to formulate a LIRK-W method for exactly this scenario, one possible approach is:

$$Y_i = y_n + h \sum_{j=1}^{i-1} a_{i,j}\, F(Y_j) + h\, \mathbf{L}_i \sum_{j=1}^{i} \gamma_{i,j}\; Y_j \tag{5.13a}$$

$$y_{n+1} = y_n + h \sum_{j=1}^{s} b_j\, F(Y_j) + h\, \mathbf{L} \sum_{j=1}^{s} g_i\, Y_j, \tag{5.13b}$$

5.3 Order Conditions for Linearly-Implicit Runge-Kutta-W Methods

We construct classical order conditions for both types of LIRK-W method by matching the Taylor series expansion of the exact solution of equation (5.1) with the Taylor series expansion of the numerical solutions. To do so we make use of Butcher trees [48, section

II.2], with suitable modifications to handle the particular aspects of LIRK-W methods. These modifications lead to the new family of $LW-$trees, which we discuss next.

5.3.1 LW-trees

These trees contain three different colored vertices: meagre vertices are filled and represent an appearance of $F(y)$ and its derivatives, fat vertices are empty and represent an appearance of the linear term $\mathbf{L}$, and finally square vertices which represent a differentiation in time of the fat node they terminate in. We refer to the set of these trees as LW_i-trees, where i denotes the type of LIRK-W method being discussed.

For both LIRK-W methods of type 1 and type 2, we make use of the standard notation that the tree $[\tau_1, \ldots, \tau_m]_\bullet$ is constructed by attaching the subtrees $\tau_1, \ldots, \tau_m$ to a meagre node as its root. Similarly, the $[\tau]_\circ$ is constructed by attaching the subtree τ to a fat node, and new to this manuscript is the notation that $[\tau]_{\theta_p}$ is constructed by attaching the subtree τ to a fat node prepended by p square nodes. Figure 5.1 shows how trees can be constructed from sub-trees.

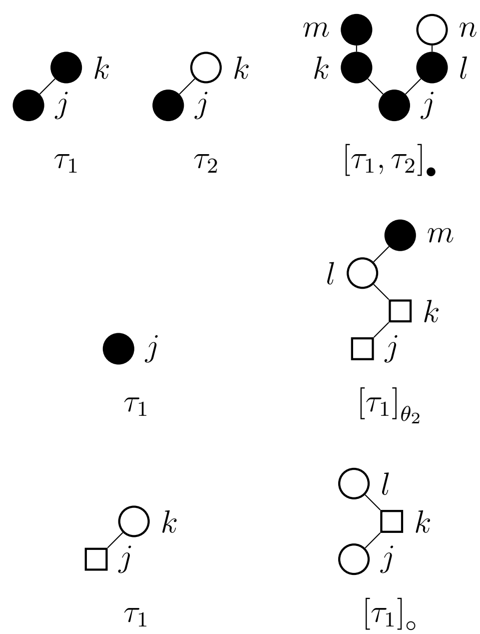


Figure 5.1: Illustration of the recursive definition of LW-trees.

To derive the expansion of the numerical solution for type 1 and 2 methods we make use of Faà di Bruno's formula [48, section II.2] to compute general high-order derivatives of the initial value problem right-hand-side vector $F(y)$

$$\left(F(Y_j)\right)^{(q-1)}\Big|_{h=0} = \sum_{m>=1} \frac{\partial^m f}{\partial y^m}\left(Y_j^{(\mu_1)},\ldots,Y_j^{(\mu_m)}\right), \quad \mu_1+\cdots+\mu_m = q-1. \tag{5.14}$$

5.3.2 Order conditions for LIRK-W methods of type 1

Here we derive order conditions for LIRK-W methods of type 1 (5.11).

We can make use of the expansion (5.14) to construct derivatives of the type 1 LIRK-W scheme to arrive at the result that

$$(Y_i)^{(q)}\Big|_{h=0} = q\sum_{j=1}^{i-1} a_{i,j}\sum_{m\geq 1}\frac{\partial^m F}{\partial y^m}\left(Y_j^{(\mu_1)},\ldots,Y_j^{(\mu_m)}\right) + q\sum_{j=1}^{i}\gamma_{i,j}\sum_{k=0}^{q-1}\binom{q-1}{k}\gamma_{j,j}^k\left(\frac{d^k}{dt^k}\mathbf{L}\right)Y_j^{(q-1-k)} \tag{5.15a}$$

$$(y_{n+1})^{(q)}\Big|_{h=0} = q\sum_{i=1}^{s} b_i\sum_{m\geq 1}\frac{\partial^m F}{\partial y^m}\left(Y_i^{(\mu_1)},\ldots,Y_i^{(\mu_m)}\right) + q\sum_{i=1}^{s} g_i\sum_{k=0}^{q-1}\binom{q-1}{k}\gamma_{i,i}^k\left(\frac{d^k}{dt^k}\mathbf{L}\right)Y_i^{(q-1-k)} \tag{5.15b}$$

Leading to the definition for the LW_1 trees

$$LW_1 = \left\{\begin{array}{ll} N_3\text{-trees:} & \text{fat vertices are singly branched, and} \\ & \text{square vertices are singly branched} \\ & \text{with square or fat children}\end{array}\right\}$$

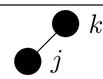
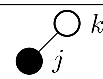
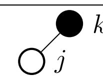
Figures 5.2 and 5.3, give the LW_1-trees, τ_i, representing the elementary differentials, $f(\tau_i)$, present in the Taylor expansion of the exact and numerical solutions, as well as the coefficients of these terms, $\Phi^1(\tau_i)$ and $P(\tau_i)$ respectively.

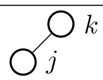
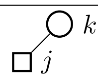
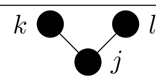
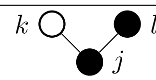
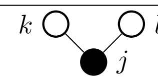
The recursive formulas in equation (5.16) map the visual representation of a given LW_1-tree to the left hand side of the order condition, $\Phi_j(\tau)$, corresponding to that tree.

$$\Phi_j^1(\tau) = \left\{\begin{array}{lccl} b_j\bar{\Phi}_j^1(\tau_1)\cdots\bar{\Phi}_j^1(\tau_m) & \text{if}\ \tau & = & [\tau_1,\ldots,\tau_m]_\bullet\,, \\ g_j\bar{\Phi}_j^1(\tau_1) & \text{if}\ \tau & = & [\tau_1]_\circ\,, \\ g_j\gamma_{j,j}^p\bar{\Phi}_j^1(\tau_1) & \text{if}\ \tau & = & [\tau_1]_{\theta_p}\,, \\ b_j & \text{if}\ \tau & = & [\,]_\bullet\,, \\ g_j & \text{if}\ \tau & = & [\,]_\circ\,, \end{array}\right. \tag{5.16a}$$

$$\bar{\Phi}_j^1(\tau) = \left\{\begin{array}{lccl} a_{j,k}\bar{\Phi}_k^1(\tau_1)\cdots\bar{\Phi}_k^1(\tau_m) & \text{if}\ \tau & = & [\tau_1,\ldots,\tau_m]_\bullet\,, \\ \gamma_{j,k}\bar{\Phi}_k^1(\tau_1) & \text{if}\ \tau & = & [\tau_1]_\circ\,, \\ \gamma_{j,k}\gamma_{k,k}^p\bar{\Phi}_j^1(\tau_1) & \text{if}\ \tau & = & [\tau_1]_{\theta_p}\,, \\ a_{j,k} & \text{if}\ \tau & = & [\,]_\bullet\,, \\ \gamma_{j,k} & \text{if}\ \tau & = & [\,]_\circ\,, \end{array}\right. \tag{5.16b}$$

(5.16c)

i	1	2	3	4	5
τ_i					
$f(\tau_i)$	f^J	$\mathbf{L}_{JK}y^K$	$f^J_K f^K$	$f^J_K \mathbf{L}_{KL} y^L$	$\mathbf{L}_{JK} f^K$
$\Phi^1(\tau_i)$	b_j	g_j	$b_j a_{j,k}$	$b_j \gamma_{j,k}$	$g_j a_{j,k}$
$\Phi^2(\tau_i)$	b_j	g_j	$b_j a_{j,k}$	$b_j \gamma_{j,k}$	$g_j a_{j,k}$
$P(\tau_i)$	1	0	1/2	0	0

i	6	7	8	9	10
τ_i					
$f(\tau_i)$	$\mathbf{L}_{JK}\mathbf{L}_{KL}y^L$	$\mathbf{L}'_{JK}y^K$	$f^J_{KL} f^K f^L$	$f^J_{KL}\mathbf{L}_{KM}y^M f^L$	$f^J_{KL}\mathbf{L}_{KM}y^M \mathbf{L}_{LN}y^N$
$\Phi^1(\tau_i)$	$g_j \gamma_{j,k}$	$g_k \gamma_{k,k}$	$b_j a_{j,k} a_{j,l}$	$b_j \gamma_{j,k} a_{j,l}$	$b_j \gamma_{j,k}\gamma_{j,l}$
$\Phi^2(\tau_i)$	$g_j \gamma_{j,k}$	N/A	$b_j a_{j,k} a_{j,l}$	$b_j \gamma_{j,k} a_{j,l}$	$b_j \gamma_{j,k}\gamma_{j,l}$
$P(\tau_i)$	0	0	1/3	0	0

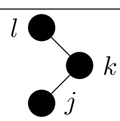
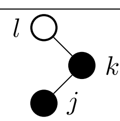
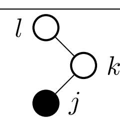
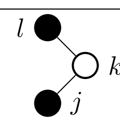
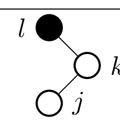
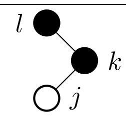
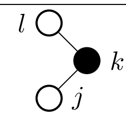
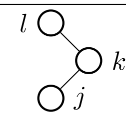
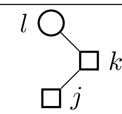
i	11	12	13	14	15
τ_i					
$f(\tau_i)$	$f^J_K f^K_L f^L$	$f^J_K f^K_L \mathbf{L}_{LM} y^M$	$f^J_K \mathbf{L}_{KL}\mathbf{L}_{LM}y^M$	$f^J_K \mathbf{L}_{KL} f^L$	$\mathbf{L}_{JK}\mathbf{L}_{KL} f^L$
$\Phi^1(\tau_i)$	$b_j a_{j,k} a_{k,l}$	$b_j a_{j,k}\gamma_{k,l}$	$b_j \gamma_{j,k}\gamma_{k,l}$	$b_j \gamma_{j,k} a_{k,l}$	$g_j \gamma_{j,k} a_{k,l}$
$\Phi^2(\tau_i)$	$b_j a_{j,k} a_{k,l}$	$b_j a_{j,k}\gamma_{k,l}$	$b_j \gamma_{j,k}\gamma_{k,l}$	$b_j \gamma_{j,k} a_{k,l}$	$g_j \gamma_{j,k} a_{k,l}$
$P(\tau_i)$	1/6	0	0	0	0

Figure 5.2: Trees and order conditions for LIRK-W methods up to order three.

i	16	17	18	19
τ_i				
$f(\tau_i)$	$\mathbf{L}_{JK} f^K_L f^L$	$\mathbf{L}_{JK} f^K_L \mathbf{L}_{LM} y^M$	$\mathbf{L}_{JK} \mathbf{L}_{KL} \mathbf{L}_{LM} y^M$	$\mathbf{L}''_{JM} y^M$
$\Phi^1(\tau_i)$	$g_j a_{j,k} a_{k,l}$	$g_j a_{j,k} \gamma_{k,l}$	$g_j \gamma_{j,k} \gamma_{k,l}$	$g_l \gamma_{l,l} \gamma_{l,l}$
$\Phi^2(\tau_i)$	$g_j a_{j,k} a_{k,l}$	$g_j a_{j,k} \gamma_{k,l}$	$g_j \gamma_{j,k} \gamma_{k,l}$	N/A
$P(\tau_i)$	0	0	0	0

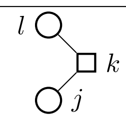
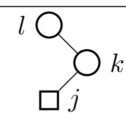
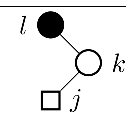
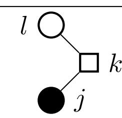
i	20	21	22	23
τ_i				
$f(\tau_i)$	$\mathbf{L}_{JK} \mathbf{L}'_{KM} y^M$	$\mathbf{L}'_{JL} \mathbf{L}_{LM} y^M$	$\mathbf{L}'_{JL} f^L$	$f^J_K \mathbf{L}'_{KM} y^M$
$\Phi^1(\tau_i)$	$g_j \gamma_{j,l} \gamma_{l,l}$	$g_k \gamma_{k,k} \gamma_{k,l}$	$g_k \gamma_{k,k} a_{k,l}$	$b_j \gamma_{j,l} \gamma_{l,l}$
$\Phi^2(\tau_i)$	$g_j \gamma_{j,j} \gamma_{j,j}$	N/A	N/A	$b_j \gamma_{j,j} \gamma_{j,j}$
$P(\tau_i)$	0	0	0	0

Figure 5.3: Trees and order conditions for LIRK-W methods up to order three (Continued).

Similarly, the recursive formula in equation (5.17) maps the visual representation of an LW_1-tree to its corresponding elementary differential in the Taylor series expansion of the numerical solution computed using a LIRK-W scheme (5.11).

$$F^J(\tau)(y) = \begin{cases} \sum\limits_{K_1,\ldots,K_m} f^J_{K_1,\ldots,K_M}(y)\cdot\left(F^{K_1}(\tau_1)\cdots F^{K_m}(\tau_m)\right)(y) & \text{if} \quad \tau = [\tau_1,\ldots,\tau_m]_\bullet \\ \sum\limits_K \mathbf{L}_{JK}F^K(\tau_1)(y) & \text{if} \quad \tau = [\tau_1]_\circ \\ \sum\limits_K \left(\frac{d^p}{dt^p}\mathbf{L}_{JK}\right)F^K(\tau_1)(y) & \text{if} \quad \tau = [\tau_1]_{\theta_p} \\ \sum\limits_K \mathbf{L}_{JK}y^K & \text{if} \quad \tau = []_\circ \\ \sum\limits_K \left(\frac{d^p}{dt^p}\mathbf{L}_{JK}\right)y^K & \text{if} \quad \tau = []_{\theta_p} \end{cases} \tag{5.17}$$

Remark 11. *We note that the Taylor series expansion of the exact solution will be comprised of only terms containing the function $F(t,y)$ and its derivatives, and so these terms will be represented by trees containing only meagre nodes. Alternatively, the Taylor series expansion of the numerical solution will be comprised of terms containing the function $F(t,y)$ and its derivatives, as well as terms containing $\mathbf{L}$ and its derivatives. For this reason, the order conditions corresponding to trees containing fat and square nodes will be set to zero, to eliminate any contribution of these terms to the error of the LIRK-W method.*

Theorem 6 (Order conditions for LIRK-W methods of type 1)**.** *A LIRK-W method of type 1 has order p iff the following order conditions hold:*

$$\sum_j \Phi^1_j(\tau) = \frac{1}{\gamma(\tau)} \quad \forall\, \tau \in T \quad \textit{with } \rho(\tau) \le p\,, \tag{5.18}$$

$$\sum_j \Phi^1_j(\tau) = 0 \quad \forall\, \tau \in LW\backslash T \quad \textit{with } \rho(\tau) \le p\,. \tag{5.19}$$

Here $\rho(t)$ is the number of vertices of the tree t, and $\gamma(t)$ is the "product of $\rho(t)$ and all orders of the trees which appear, if the roots, one after another, are removed from t" [48, Section II.2].

Proof. The proof follows from our discussion, equation (5.15) and the order conditions of Rosenbrock-W methods [11, Section IV.7]. □

5.3.3 Order conditions for LIRK-W methods of type 2

Here we derive order conditions for LIRK-W methods of type 2 (5.12).

We once again make use of the expansion (5.14) to construct derivatives of the type 2 LIRK-W scheme to arrive at the result that

$$\left.(Y_i)^{(q)}\right|_{h=0} = q\sum_{j=1}^{i-1} a_{i,j} \sum_{m>=1} \frac{\partial^m f}{\partial y^m}\left(Y_j^{(\mu_1)},\ldots,Y_j^{(\mu_m)}\right) + q\mathbf{L}\sum_{j=1}^{s}(Y_j)^{(q-1)} + q\sum_{k=0}^{q-1}\binom{q-1}{k}\gamma_{i,i}^{k+1}\left(\frac{d^k}{dt^k}\mathbf{L}\right)(Y_i)^{(q-1-k)} \quad (5.20a)$$

$$\left.(y_{n+1})^{(q)}\right|_{h=0} = q\sum_{j=1}^{s} b_j \sum_{m>=1} \frac{\partial^m f}{\partial y^m}\left(Y_j^{(\mu_1)},\ldots,Y_j^{(\mu_m)}\right) + q\mathbf{L}\sum_{j=1}^{s} g_j\,(Y_j)^{(q-1)} \quad (5.20b)$$

Leading to the definition of LW_2-trees

$$LW_2 = \left\{ \begin{array}{ll} N_3\text{-trees:} & \text{fat vertices are singly branched, and} \\ & \text{no tree has a square root, and} \\ & \text{square vertices are singly branched} \\ & \quad\text{with square or fat children.} \end{array} \right\}$$

Figures 5.2 and 5.3, give the LW_2-trees, τ_i, representing the elementary differentials, $f(\tau_i)$, present in the Taylor expansion of the exact and numerical solutions, as well as the coefficients of these terms, $\Phi^2(\tau_i)$ and $P(\tau_i)$ respectively.

Remark 12. *LW_2-trees are a subset of LW_1-trees, and so figures 5.2 and 5.3 contains trees which are not present in an expansion of the numerical solution of a LIRK-W method of type 2. These trees can be identified by the presence of a square root, and the corresponding entry for $\Phi^2(\tau_i)$ being listed as "N/A". In addition, $\Phi^1(\tau_i)$ and $\Phi^2(\tau_i)$ are identical, except in the case where τ_i contains a square node.*

The recursive formulas in equation (5.21) map the visual representation of a given LW-tree to the left hand side of the order condition, $\Phi_j(\tau)$, corresponding to that tree.

$$\Phi_j^2(\tau) = \left\{ \begin{array}{llcl} b_j\bar{\Phi}_j^2(\tau_1)\cdots\bar{\Phi}_j^2(\tau_m) & \text{if } \tau & = & [\tau_1,\ldots,\tau_m]_\bullet\,, \\ g_j\bar{\Phi}_j^2(\tau_1) & \text{if } \tau & = & [\tau_1]_\circ\,, \\ g_j\gamma_{j,j}^p\bar{\Phi}_j^2(\tau_1) & \text{if } \tau & = & [\tau_1]_{\theta_p}\,, \\ b_j & \text{if } \tau & = & [\,]_\bullet\,, \\ g_j & \text{if } \tau & = & [\,]_\circ\,, \end{array} \right. \quad (5.21a)$$

$$\bar{\Phi}_j^2(\tau) = \left\{ \begin{array}{llcl} a_{j,k}\bar{\Phi}_k^2(\tau_1)\cdots\bar{\Phi}_k^2(\tau_m) & \text{if } \tau & = & [\tau_1,\ldots,\tau_m]_\bullet\,, \\ \gamma_{j,k}\bar{\Phi}_k^2(\tau_1) & \text{if } \tau & = & [\tau_1]_\circ\,, \\ \gamma_{jj}^{p+1}\bar{\Phi}_j^2(\tau_1) & \text{if } \tau & = & [\tau_1]_{\theta_p}\,, \\ a_{j,k} & \text{if } \tau & = & [\,]_\bullet\,, \\ \gamma_{j,k} & \text{if } \tau & = & [\,]_\circ\,, \end{array} \right. \quad (5.21b)$$

(5.21c)

LW_1 and LW_2 map from trees to elementary differentials in the same way, so that once again equation (5.17) maps the visual representation of an LW_2-tree to its corresponding elementary differential in the Taylor series expansion of the numerical solution using the LIRK-W scheme of type 2 in equation (5.12).

Making use of the same logic as for LIRK-W methods of type-1 we have that for LIRK-W methods of type 2

Theorem 7 (Order conditions for LIRK-W methods of type 2)**.** *A LIRK-W method of type 1 has order p iff the following order conditions hold:*

$$\sum_j \Phi_j^2(\tau) = \frac{1}{\gamma(\tau)} \quad \forall\, \tau \in T \quad \textit{with } \rho(\tau) \le p\,, \tag{5.22}$$

$$\sum_j \Phi_j^2(\tau) = 0 \quad \forall\, \tau \in LW \backslash T \quad \textit{with } \rho(\tau) \le p\,. \tag{5.23}$$

Here $\rho(t)$ is the number of vertices of the tree t, and $\gamma(t)$ is the "product of $\rho(t)$ and all orders of the trees which appear, if the roots, one after another, are removed from t" [48, Section II.2].

Proof. The proof follows from our discussion, equation (5.20) and the order conditions of Rosenbrock-W methods [11, Section IV.7]. □

5.4 Linear stability of LIRK-W methods

To examine the linear stability of the proposed method we solve the linear test problem

$$\frac{dy}{dt} = \mathbf{L}y + (\mathbf{J} - \mathbf{L})\, y\,, \quad t_0 \le t \le t_F\,, \quad y(t_0) = y_n\,; \quad y(t) \in \mathbb{R}^N \quad \mathbf{J}, \mathbf{L} \in \mathbb{R}^{N\times N}. \tag{5.24}$$

and apply the method (5.5) in compact form

$$\mathbf{Y} = \mathbb{1} \otimes y_n + \left[\mathbf{A} \otimes (h\mathbf{J} - h\mathbf{L}(h)) + \widehat{\mathbf{A}} \otimes h\mathbf{L}(h)\right] \mathbf{Y}, \tag{5.25a}$$

$$y_{n+1} = y_n + \left[\mathbf{b}^T \otimes (h\mathbf{J} - h\mathbf{L}(h)) + \widehat{\mathbf{b}}^T \otimes h\mathbf{L}(h)\right] \mathbf{Y}, \tag{5.25b}$$

where

$$\mathbf{A} = \begin{bmatrix} a_{1,1} & \dots & a_{1,s} \\ \vdots & \ddots & \vdots \\ a_{s,1} & \dots & a_{s,s} \end{bmatrix}, \quad \widehat{\mathbf{A}} = \begin{bmatrix} \widehat{a}_{1,1} & \dots & \widehat{a}_{1,s} \\ \vdots & \ddots & \vdots \\ \widehat{a}_{s,1} & \dots & \widehat{a}_{s,s} \end{bmatrix}, \quad \mathbf{b} = \begin{bmatrix} b_1 \\ \vdots \\ b_s \end{bmatrix}, \quad \widehat{\mathbf{b}} = \begin{bmatrix} \widehat{b}_1 \\ \vdots \\ \widehat{b}_s \end{bmatrix}$$

and

$$\mathbf{Y} = \left[\; Y_1^T \quad \dots \quad Y_s^T \;\right]^T.$$

5.4.1 Type 1 methods

Application of a type 1 method (5.11) to (5.24) gives:

$$\begin{aligned} Y_i &= y_n + h \sum_{j=1}^{i-1} a_{i,j} \, (\mathbf{J} - \mathbf{L}_j) \, Y_j + h \sum_{j=1}^{i} \widehat{a}_{i,j} \, \mathbf{L}_j \, Y_j, \\ y_{n+1} &= y_n + h \sum_{i=1}^{s} b_i \, (\mathbf{J} - \mathbf{L}_i) \, Y_i + h \sum_{i=1}^{s} \widehat{b}_i \, \mathbf{L}_i \, Y_i. \end{aligned}$$

In compact notation we have:

$$\begin{aligned} \mathbf{Y} &= \mathbb{1} \otimes y_n + \left[(\mathbf{A} \otimes \mathbf{I}_N) \operatorname*{blkdiag}_{i=1,\dots,s} (h\mathbf{J} - h\mathbf{L}_i) + (\widehat{\mathbf{A}} \otimes \mathbf{I}_N) \operatorname*{blkdiag}_{i=1,\dots,s} (h\mathbf{L}_i) \right] \mathbf{Y}, \\ y_{n+1} &= y_n + \left[(\mathbf{b}^T \otimes \mathbf{I}_N) \operatorname*{blkdiag}_{i=1,\dots,s} (h\mathbf{J} - h\mathbf{L}_i) + (\widehat{\mathbf{b}}^T \otimes \mathbf{I}_N) \operatorname*{blkdiag}_{i=1,\dots,s} (h\mathbf{L}_i) \right] \mathbf{Y}. \end{aligned}$$

Solving for y_{n+1} in terms of y_n leads to the transfer matrix:

$$\begin{aligned} y_{n+1} &= R(h\mathbf{J}, h\mathbf{L}_1, \dots, h\mathbf{L}_s) \, y_n, \\ R(h\mathbf{J}, h\mathbf{L}_1, \dots, h\mathbf{L}_s) &= \mathbf{I} + \left[(\mathbf{b}^T \otimes \mathbf{I}_N) \operatorname*{blkdiag}_{i=1,\dots,s} (h\mathbf{J} - h\mathbf{L}_i) + (\widehat{\mathbf{b}}^T \otimes \mathbf{I}_N) \operatorname*{blkdiag}_{i=1,\dots,s} (h\mathbf{L}_i) \right] \cdot \\ & \quad \left[\mathbf{I} - (\mathbf{A} \otimes \mathbf{I}_N) \operatorname*{blkdiag}_{i=1,\dots,s} (h\mathbf{J} - h\mathbf{L}_i) - (\widehat{\mathbf{A}} \otimes \mathbf{I}_N) \operatorname*{blkdiag}_{i=1,\dots,s} (h\mathbf{L}_i) \right]^{-1} (\mathbb{1} \otimes \mathbf{I}). \end{aligned}$$

For a stiffly accurate method with

$$\mathbf{b}^T = \mathbf{e}_s^T \mathbf{A} \qquad \text{and} \qquad \widehat{\mathbf{b}}^T = \mathbf{e}_s^T \widehat{\mathbf{A}}, \tag{5.26}$$

standard calculations give:

$$R(h\mathbf{J}, h\mathbf{L}_1, \dots, h\mathbf{L}_s) = (\mathbf{e}_s^T \otimes \mathbf{I}) \cdot \left[\mathbf{I} - (\mathbf{A} \otimes \mathbf{I}_N) \operatorname*{blkdiag}_{i=1,\dots,s} (h\mathbf{J} - h\mathbf{L}_i) - (\widehat{\mathbf{A}} \otimes \mathbf{I}_N) \operatorname*{blkdiag}_{i=1,\dots,s} (h\mathbf{L}_i) \right]^{-1} (\mathbb{1} \otimes \mathbf{I}).$$

This matrix goes to zero for stiff linear parts:

$$\|h\mathbf{L}_i\| \to \infty, \quad i = 1, \dots, s \quad \Rightarrow \quad R(h\mathbf{J}, h\mathbf{L}_1, \dots, h\mathbf{L}_s) \to 0.$$

5.4.2 Type 2 methods

Application of a type 2 method (5.12) to (5.24) gives:

$$\begin{aligned} Y_i &= y_n + h \sum_{j=1}^{i-1} a_{i,j} \, F(Y_j) + h \, \mathbf{L} \sum_{j=1}^{i-1} \gamma_{i,j} \, Y_j + h\gamma_{i,i} \mathbf{L}_i Y_i, \\ y_{n+1} &= y_n + h \sum_{i=1}^{s} b_i \, F(Y_i) + h \, \mathbf{L} \sum_{i=1}^{s} g_i \, Y_i. \end{aligned}$$

In compact form:

$$\begin{aligned}
\mathbf{Y} &= \mathbb{1} \otimes y_n + \left[\mathbf{A} \otimes (h\mathbf{J} - h\mathbf{L}) + \widehat{\mathbf{A}} \otimes h\mathbf{L} + \underset{i=1,\dots,s}{\mathrm{blkdiag}}\big(\gamma_{i,i}(h\mathbf{L}_i - h\mathbf{L})\big) \right] \mathbf{Y}, \\
y_{n+1} &= y_n + \left[\mathbf{b}^T \otimes (h\mathbf{J} - h\mathbf{L}) + \widehat{\mathbf{b}}^T \otimes h\mathbf{L} \right] \mathbf{Y},
\end{aligned}$$

where blkdiag denotes a block-diagonal matrix with the blocks given by its arguments; the block indices should be clear from the context. The transfer matrix reads:

$$\begin{aligned}
R(h\mathbf{J}, h\mathbf{L}_1, \dots, h\mathbf{L}_s) &= \mathbf{I} + \left[(\mathbf{b}^T \otimes \mathbf{I}_N) \big(\mathbf{I}_s \otimes (h\mathbf{J} - h\mathbf{L})\big) + (\widehat{\mathbf{b}}^T \otimes \mathbf{I}_N) \big(\mathbf{I}_s \otimes h\mathbf{L}\big) \right] \cdot \\
&\quad \Big[\mathbf{I} - (\mathbf{A} \otimes \mathbf{I}_N) \big(\mathbf{I}_s \otimes (h\mathbf{J} - h\mathbf{L})\big) - (\widehat{\mathbf{A}} \otimes \mathbf{I}_N) \big(\mathbf{I}_s \otimes h\mathbf{L}\big) \\
&\quad - \underset{i=1,\dots,s}{\mathrm{blkdiag}}\big(\gamma_{i,i}(h\mathbf{L}_i - h\mathbf{L})\big) \Big]^{-1} (\mathbb{1} \otimes \mathbf{I}).
\end{aligned}$$

For a stiffly accurate method (5.26) standard calculations give:

$$\begin{aligned}
R(h\mathbf{J}, h\mathbf{L}_1, \dots, h\mathbf{L}_s) &= (\mathbf{e}_s^T \otimes \mathbf{I}) \cdot \left[\mathbf{I} - \underset{i=1,\dots,s}{\mathrm{blkdiag}}\big(\gamma_{i,i}(h\mathbf{L}_i - h\mathbf{L})\big) \right] \\
&\quad \cdot \Big[\mathbf{I} - (\mathbf{A} \otimes \mathbf{I}_N) \big(\mathbf{I}_s \otimes (h\mathbf{J} - h\mathbf{L})\big) - (\widehat{\mathbf{A}} \otimes \mathbf{I}_N) \big(\mathbf{I}_s \otimes h\mathbf{L}\big) \\
&\quad - \underset{i=1,\dots,s}{\mathrm{blkdiag}}\big(\gamma_{i,i}(h\mathbf{L}_i - h\mathbf{L})\big) \Big]^{-1} (\mathbb{1} \otimes \mathbf{I}) \\
&= (\mathbf{e}_s^T \otimes \mathbf{I}) \cdot \left[\mathbf{I} - \underset{i=1,\dots,s}{\mathrm{blkdiag}}\big(\gamma_{i,i}(h\mathbf{L}_i - h\mathbf{L})\big) \right] \\
&\quad \cdot \left[\mathbf{I} - \mathbf{A} \otimes (h\mathbf{J}) - \mathrm{tril}(\mathbf{\Gamma}) \otimes (h\mathbf{L}) - \underset{i=1,\dots,s}{\mathrm{blkdiag}}\big(\gamma_{i,i}(h\mathbf{L}_i - h\mathbf{L})\big) \right]^{-1} (\mathbb{1} \otimes \mathbf{I}) \\
&= (\mathbf{e}_s^T \otimes \mathbf{I}) \cdot \\
&\quad \cdot \left[\mathbf{I} - \Big(\mathbf{A} \otimes (h\mathbf{J}) + \mathrm{tril}(\mathbf{\Gamma}) \otimes (h\mathbf{L}) \Big) \left[\mathbf{I} - \underset{\substack{i=1,\dots,s \\ i}}{\mathrm{blkdiag}}\big(\gamma_{i,i}(h\mathbf{L}_i - h\mathbf{L})\big) \right]^{-1} \right]^{-1} (\mathbb{1} \otimes \mathbf{I}),
\end{aligned}$$

where we denote

$$\mathrm{tril}(\mathbf{\Gamma}) = \mathbf{\Gamma} - \underset{1=1,\dots,s}{\mathrm{diag}} \big(\gamma_{i,i}\big).$$

The stability matrix goes to zero when $\mathbf{L}_i \approx \mathbf{L}$ in the sense that the difference increases slower than the matrices with increasing stiffness:

$$\frac{\|h\mathbf{L}\|}{\|h\mathbf{L} - h\mathbf{L}_i\|} \to \infty, \quad i = 1, \dots, s \quad \Rightarrow \quad R(h\mathbf{J}, h\mathbf{L}_1, \dots, h\mathbf{L}_s) \to 0.$$

or

$$\frac{\|h\mathbf{J}\|}{\|h\mathbf{L} - h\mathbf{L}_i\|} \to \infty, \quad i = 1, \dots, s \quad \Rightarrow \quad R(h\mathbf{J}, h\mathbf{L}_1, \dots, h\mathbf{L}_s) \to 0.$$

Alternatively, the stability matrix goes to one when the time dependent linear term $h\mathbf{L}_i$ grows more quickly than either $h\mathbf{J}$ or $h\mathbf{L}$ with increasing stiffness, so that

$$\frac{\|h\mathbf{L}\| + \|h\mathbf{J}\|}{\|h\mathbf{L}_i\|} \to 0, \quad i = 1, \ldots, s \quad \Rightarrow \quad R(h\mathbf{J}, h\mathbf{L}_1, \ldots, h\mathbf{L}_s) \to 1.$$

From this it is clear that the stability of the method is dependent on the choice of both $\mathbf{L}$ and $\mathbf{L}_i$.

5.5 Construction of Practical LIRK-W Methods

5.5.1 A third-order LIRK-W method of type 1

For stability considerations we seek a method which is stiffly accurate, which is to say that

$$b_i = a_{s,i}, \quad \widehat{b}_i = \widehat{a}_{s,i}. \tag{5.27}$$

Moreover we wish to make use of an approximate matrix factorization, specifically the alternating directions approximation. Our choice of general form (5.11) and the structure of $\widetilde{\mathbf{L}}_i$ in equation (5.10) naturally leads to the choice that

$$\widehat{a}_{i,i} = \sum_{j=1}^{i-1} a_{i,j}. \tag{5.28}$$

Finally, for convenience we require that

$$\sum_{j=1}^{i-1} a_{i,j} = \sum_{j=1}^{i} \widehat{a}_{i,j} \tag{5.29}$$

The requirements imposed by (5.27), (5.28), and (5.29) taken together imply that a method with general form (5.6) must have the additional following properties:

$$\sum_{j=1}^{i} \gamma_{i,j} = 0, \quad \gamma_{i,i} = \sum_{j=1}^{i-1} a_{i,j}, \quad \sum_{j=1}^{i-1} \gamma_{i,j} = -\gamma_{i,i}, \quad \sum_{i=1}^{s} g_i = 0, \quad g_i = \gamma_{s,i}. \tag{5.30}$$

Interestingly, the first of these conditions, implied by (5.29), automatically satisfies all order conditions coming from trees in figures 5.2 and 5.3 which end in a fat node not directly preceded by a square node. Additionally, the simplifications in equations (5.27), (5.28), (5.29), and (5.30) means that several order conditions reduce to the same as those coming from other trees, so that:

$$\Phi(\tau_5) = \Phi(\tau_7), \quad \Phi(\tau_{14}) = \Phi(\tau_{23}), \quad \Phi(\tau_{15}) = \Phi(\tau_{20}), \quad \Phi(\tau_{22}) = \Phi(\tau_{19}).$$

In this way number of order conditions required to construct such a method of order three reduces from the original twenty-three to a much more reasonable nine. Figure 5.4 shows these trees and the order conditions associated with them.

These conditions, with significant simplification coming from equations (5.27), (5.28), (5.29), and (5.30) as well as some algebraic manipulation are given for a five-stage method as:

$$\begin{aligned}
a_{5,1} + a_{5,2} + a_{5,3} + a_{5,4} &= 1 && (5.31a)\\
a_{5,2}a_2 + a_{5,3}a_3 + a_{5,4}a_4 &= \frac{1}{2} && (5.31b)\\
\gamma_{5,2}a_2 + \gamma_{5,3}a_3 + \gamma_{5,4}a_4 &= -1 && (5.31c)\\
a_{5,2}a_2^2 + a_{5,3}a_3^2 + a_{5,4}a_4^2 &= \frac{1}{3} && (5.31d)\\
a_{5,3}a_{3,2}a_2 + a_{5,4}a_{4,3}a_3 + a_{5,4}a_{4,2}a_2 &= \frac{1}{6} && (5.31e)\\
a_{5,3}\gamma_{3,2}a_2 + a_{5,4}\gamma_{4,3}a_3 + a_{5,4}\gamma_{4,2}a_2 &= -\frac{1}{3} && (5.31f)\\
\gamma_{5,3}\gamma_{3,2}a_2 + \gamma_{5,4}\gamma_{4,3}a_3 + \gamma_{5,4}\gamma_{4,2}a_2 &= 1 && (5.31g)\\
\gamma_{5,3}a_{3,2}a_2 + \gamma_{5,4}a_{4,3}a_3 + \gamma_{5,4}a_{4,2}a_2 &= -\frac{1}{2} && (5.31h)\\
\gamma_{5,2}a_2^2 + \gamma_{5,3}a_3^2 + \gamma_{5,4}a_4^2 &= -1 && (5.31i)
\end{aligned}$$

Table 5.2 gives the coefficients for a LIRK-W method of third order.

5.5.2 Third-order LIRK-W methods of type 2

For stability considersations we again seek a method which is stiffly accurate, and so satisfies (5.27), as well as the condition that (5.29) to reduce the number of required order conditions. Due to the difference in general forms we no longer impose the condition (5.28), which provides the freedom to impose the condition that

$$\gamma_{i,i} = \begin{cases} 0 & \text{for} \quad i = 1 \\ \gamma & \text{for} \quad i = 2, \ldots, s \end{cases} \tag{5.32}$$

Taking requirements imposed by (5.27), (5.29), and (5.32) taken together imply that a method with general form (5.12) must have the additional following properties:

$$\sum_{j=1}^{i} \gamma_{i,j} = 0, \quad \sum_{j=1}^{i-1} \gamma_{i,j} = -\gamma_{i,i}, \quad \sum_{i=1}^{s} g_i = 0, \quad g_i = \gamma_{s,i}. \tag{5.33}$$

Once again the first of these conditions, implied by (5.29), automatically satisfies all order conditions coming from trees in figures 5.2 and 5.3 which end in a fat node not directly

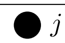
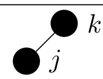
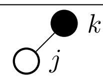
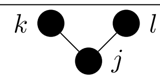
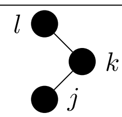
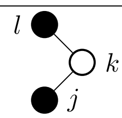
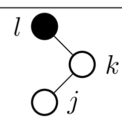
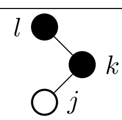
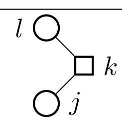
τ_1		f^J	$\sum b_j \quad = \quad 1$
τ_3		$f^J_K f^K$	$\sum b_j a_{j,k} \quad = \quad 1/2$
τ_5		$\mathbf{L}_{JK} f^K$	$\sum g_j a_{j,k} \quad = \quad 0$
τ_8		$f^J_{KL} f^K f^L$	$\sum b_j a_{j,k} a_{j,l} \; = \; 1/3$
τ_{11}		$f^J_K f^K_L f^L$	$\sum b_j a_{j,k} a_{k,l} \; = \; 1/6$
τ_{14}		$f^J_K \mathbf{L}_{KL} f^L$	$\sum b_j \gamma_{j,k} a_{k,l} \; = \; 0$
τ_{15}		$\mathbf{L}_{JK} \mathbf{L}_{KL} f^L$	$\sum g_j \gamma_{j,k} a_{k,l} \; = \; 0$
τ_{16}		$\mathbf{L}_{JK} f^K_L f^L$	$\sum g_j a_{j,k} a_{k,l} \; = \; 0$
τ_{22}		$\mathbf{L}_{JK} \mathbf{L}'_{KL} y^L$	$\sum g_j \gamma^2_{j,j} \quad = \quad 0$

Figure 5.4: Butcher trees and LIRK-W conditions up to order three for a type 1 method.

$$a = \begin{bmatrix} 0 & & & & \\ 0.520300000000000 & 0 & & & \\ 0.026500000000000 & 0.938000000000000 & 0 & & \\ 0.122175553766880 & 0.105600000000000 & 0.018300000000000 & 0 & \\ -0.033950868284890 & 0.218016324016351 & 0.258600000000000 & 0.557334544268539 & 0 \end{bmatrix}$$

$$\gamma = \begin{bmatrix} 0 & & & & \\ -0.520300000000000 & 0.520300000000000 & & & \\ 0.911500000000000 & -1.876000000000000 & 0.964500000000000 & & \\ -0.401069249711528 & 0.663393695944647 & -0.508400000000000 & 0.246075553766880 & \\ -0.155925222099085 & -0.084089256959580 & -1.070724285228281 & 0.310738764286946 & 1 \end{bmatrix}$$

$$b = \begin{bmatrix} -0.033950868284890 & 0.218016324016351 & 0.258600000000000 & 0.557334544268539 & 0 \end{bmatrix}$$

$$g = \begin{bmatrix} -0.155925222099085 & -0.084089256959580 & -1.070724285228281 & 0.310738764286946 & 1 \end{bmatrix}$$

Table 5.1: Method coefficients for a third order LIRK-W method of type 1.

preceded by a square node. In this way the number of required order conditions required to construct such a method of order three reduces from the original nineteen to a much more reasonable ten. Figure 5.5 shows these trees and the order conditions associated with them.

Examining the order conditions coming from the first and last two trees of figure 5.5 leads to the additional strategic choice that

$$a_{5,1} = 1, \quad \gamma_{5,1} = 0 \tag{5.34}$$

so that the order conditions coming from τ_{18} and τ_{19} are automatically satisfied.

These conditions, with significant simplification coming from equations (5.27), (5.29), (5.33), and (5.34) as well as some algebraic manipulation are given for a five-stage method as:

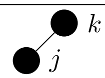
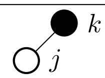
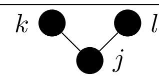
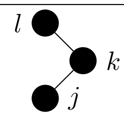
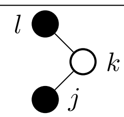
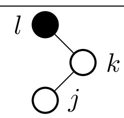
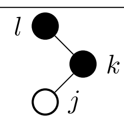
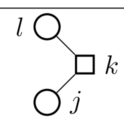
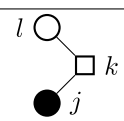
τ_1		f^J	$\sum b_j \quad = \quad 1$
τ_3		$f^J_K f^K$	$\sum b_j a_{j,k} \quad = \quad 1/2$
τ_5		$\mathbf{L}_{JK} f^K$	$\sum g_j a_{j,k} \quad = \quad 0$
τ_8		$f^J_{KL} f^K f^L$	$\sum b_j a_{j,k} a_{j,l} \quad = \quad 1/3$
τ_{11}		$f^J_K f^K_L f^L$	$\sum b_j a_{j,k} a_{k,l} \quad = \quad 1/6$
τ_{14}		$f^J_K \mathbf{L}_{KL} f^L$	$\sum b_j \gamma_{j,k} a_{k,l} \quad = \quad 0$
τ_{15}		$\mathbf{L}_{JK} \mathbf{L}_{KL} f^L$	$\sum g_j \gamma_{j,k} a_{k,l} \quad = \quad 0$
τ_{16}		$\mathbf{L}_{JK} f^K_L f^L$	$\sum g_j a_{j,k} a_{k,l} \quad = \quad 0$
τ_{18}		$\mathbf{L}_{JK} \mathbf{L}'_{KL} y^L$	$g_j \gamma^2_{j,j} \quad = \quad 0$
τ_{19}		$f^J_K \mathbf{L}'_{KL} y^L$	$b_j \gamma^2_{j,j} \quad = \quad 0$

Figure 5.5: Butcher trees and LIRK-W conditions up to order three for a type 2 method.

$$
\begin{aligned}
a &= \begin{bmatrix}
0 & & & & \\
\frac{1}{6} & 0 & & & \\
\frac{1}{3} - \frac{9\gamma+2\gamma_{5,4}}{3(5\gamma+2\gamma_{5,4})} & \frac{9\gamma+2\gamma_{5,4}}{3(5\gamma+2\gamma_{5,4})} & 0 & & \\
\frac{1}{2} - \frac{2}{3}(3a_{4,3}-1) - a_{4,3} & \frac{2}{3}(3a_{4,3}-1) & a_{4,3} & 0 & \\
1 & -\frac{3}{2} & 0 & \frac{3}{2} & 0
\end{bmatrix} \\
\gamma &= \begin{bmatrix}
0 & & & & \\
-\gamma & \gamma & & & \\
\frac{2\left(3\gamma^2+\gamma\gamma_{5,4}\right)}{5\gamma+2\gamma_{5,4}} - \gamma & -\frac{2\left(3\gamma^2+\gamma\gamma_{5,4}\right)}{5\gamma+2\gamma_{5,4}} & \gamma & & \\
2(\gamma+\gamma_{4,3}) - \gamma - \gamma_{4,3} & -2(\gamma+\gamma_{4,3}) & \gamma_{4,3} & \gamma & \\
0 & 4\gamma+\gamma_{5,4} & -5\gamma-2\gamma_{5,4}, & \gamma_{5,4} & \gamma
\end{bmatrix} \\
b_i &= \begin{bmatrix} 1 & -\frac{3}{2} & 0 & \frac{3}{2} & 0 \end{bmatrix} \\
g_i &= \begin{bmatrix} 0 & 4\gamma+\gamma_{5,4} & -5\gamma-2\gamma_{5,4}, & \gamma_{5,4} & \gamma \end{bmatrix}
\end{aligned}
$$

Table 5.2: Method coefficients for a third order LIRK-W method of type 2.

$$
\begin{aligned}
a_{5,2} + a_{5,3} + a_{5,4} &= 0 && \text{(5.35a)} \\
\gamma_{5,2} + \gamma_{5,3} + \gamma_{5,4} &= -\gamma && \text{(5.35b)} \\
a_{5,2}a_2 + a_{5,3}a_3 + a_{5,4}a_4 &= \frac{1}{2} && \text{(5.35c)} \\
\gamma_{5,2}a_2 + \gamma_{5,3}a_3 + \gamma_{5,4}a_4 &= -\gamma && \text{(5.35d)} \\
a_{5,2}a_2^2 + a_{5,3}a_3^2 + a_{5,4}a_4^2 &= \frac{1}{3} && \text{(5.35e)} \\
a_{5,3}a_{3,2}a_2 + a_{5,4}a_{4,2}a_2 + a_{5,4}a_{4,3}a_3 &= \frac{1}{6} && \text{(5.35f)} \\
a_{5,3}\gamma_{3,2}a_2 + a_{5,4}\gamma_{4,2}a_2 + a_{5,4}\gamma_{4,3}a_3 &= -\frac{1}{2}\gamma && \text{(5.35g)} \\
\gamma_{5,3}\gamma_{3,2}a_2 + \gamma_{5,4}\gamma_{4,2}a_2 + \gamma_{5,4}\gamma_{4,3}a_3 &= \gamma^2 && \text{(5.35h)} \\
\gamma_{5,3}a_{3,2}a_2 + \gamma_{5,4}a_{4,2}a_2 + \gamma_{5,4}a_{4,3}a_3 &= -\frac{1}{2}\gamma && \text{(5.35i)}
\end{aligned}
$$

5.6 Numerical Experiments

We show here the convergence of both type 1 and type 2 LIRK-W schemes when applied to the shallow water equations on a sphere. The shallow water equations in spherical coordinates are

$$\begin{aligned}
\frac{\partial u}{\partial t} + \frac{1}{a\cos\theta}\left(u\frac{\partial u}{\partial \lambda} + v\cos\theta\frac{\partial u}{\partial \theta}\right) - \left(f + \frac{u\tan\theta}{a}\right)a + \frac{g}{a\cos\theta}\frac{\partial h}{\partial \lambda} &= 0 && (5.36a)\\
\frac{\partial v}{\partial t} + \frac{1}{a\cos\theta}\left(u\frac{\partial v}{\partial \lambda} + v\cos\theta\frac{\partial v}{\partial \theta}\right) + \left(f + \frac{u\tan\theta}{a}\right)u + \frac{g}{a}\frac{\partial h}{\partial \theta} &= 0 && (5.36b)\\
\frac{\partial h}{\partial t} + \frac{1}{a\cos\theta}\left(\frac{\partial (hu)}{\partial \lambda} + \frac{\partial (hv\cos\theta)}{\partial \theta}\right) &= 0. && (5.36c)
\end{aligned}$$

Where $f = 2\Omega\sin\theta$, h is the height of the atmosphere, u is the zonal wind component, v is the meridional wind component, θ and λ are the latitudinal and longitudinal directions, a is the radius of the earth, Ω is the rotational velocity of the earth, and g is the gravitational constant. The space discretization is performed using the unstaggered Turkel-Zwas scheme [67, 68], with 72 nodes in the longitudinal direction and 36 nodes in the latitudinal direction. Figure 5.6 confirms the third order convergence of type 1 and type 2 LIRK-W schemes applied to the spherical shallow water equations, when $\mathbf{L} = \mathbf{J}$.

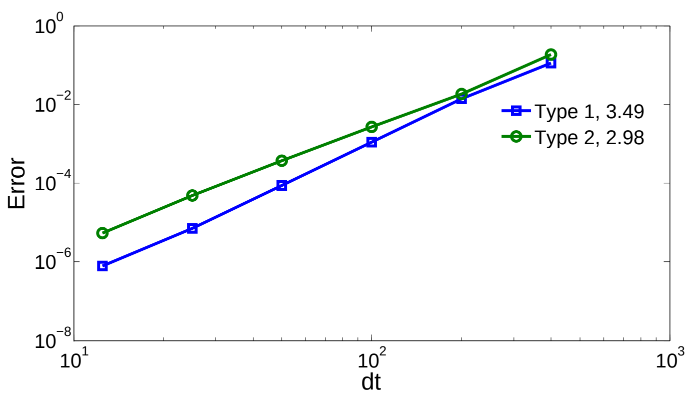


Figure 5.6: Convergence diagram for LIRK-W methods applied to spherical shallow water equations.

The numerical exploration of these schemes is ongoing, and work is currently being done to apply these methods to large computational fluid dynamics models. This exploration will include the application of several different formulations of the time dependent, and stage varying linear term $\mathbf{L}_i y$.

5.7 Conclusions

We have presented a new class of time-integration schemes, which permit an arbitrary, time dependent, and stage varying linear approximation of the stiff system dynamics, called

Linearly-Implicit Runge-Kutta-W methods. These schemes maintain high-order when making use of an approximate matrix factorization, and can be reformulated to permit other time dependent approximations. Future work that explores new formulations of the method and order conditions to permit new, or different, approximations may lead to more efficient integrators.

Chapter 6

Conclusions and Future Research Directions

In this work we have developed several novel lightly implicit time integration methodologies. A lightly implicit method considers the time integration and approximation of the (non)linear systems arising therein as a single computational process. The structure and analytical framework of these schemes is designed to permit computationally favorable approximations.

The first family, K-methods, offer an alternative to the standard application of Krylov based methods in the approximation of the solution to linear systems, in the case of Rosenbrock methods in chapter 2, and the computation of the matrix exponential in chapter 3, in the case of exponential schemes. These methods are designed to admit very low accuracy Krylov approximations, while maintaining full order of accuracy. K-methods, both Rosenbrock-Krylov and exponential-Krylov, are well suited for implementation on parallel architectures due to their exclusive use of scalable operations, relatively low memory requirements, and use of only matrix vector products so that matrix-free implementations are possible.

The Linearly-implicit Runge-Kutta-W schemes presented in Chapter 5, in contrast, are specifically designed to maintain full order of accuracy when making use of approximate matrix factorization solutions of the linear systems. This family of methods provides a flexible framework for the use of arbitrary, time dependent, and stage varying approximations of stiff linear dynamics of the initial value problem. The flexibility of the linearly-implicit Runge-Kutta-W framework makes it possible to leverage many different approximations, and tailor the computational approach to the specific problem under study.

In addition to presenting several new classes of lightly implicit time integration methods, we have examined several stability questions for time integration schemes that employ approximate solutions of linear systems, or approximate evaluations of matrix functions. In Chapter 4 we leveraged this stability analysis to propose two adaptive implementations of K-methods that automatically select the Krylov basis size and improve the efficiency of the time integration schemes.

Future work in the identification of approximations well suited to specific classes of problems, or the coupling of approximations to particular spatial discretization schemes, may lead to the development of new lightly-implicit time integration schemes specifically designed to allow for such approximations. Aside from the construction of new lightly-implicit methods, additional work is needed on the stability analysis of these schemes. While the order condition theories for lightly-implicit methods account for special approximations of their implicit terms, the current stability analysis does not. A focus on incorporating the special structure of the approximations in to the stability analysis may prove fruitful in the pursuit of more efficient schemes.

In conclusion, lightly-implicit methods are a strong alternative to both standard explicit and implicit time integration schemes. Their ability to leverage computationally favorable approximations makes them an excellent candidate for solving very large problems on parallel architectures. This dissertation research provides the groundwork for the next generation of scalable and efficient lightly-implicit time integration schemes.

Bibliography

[1] M. Günther, C. Hachtel, and A. Sandu. Multirate GARK schemes for multiphysics problems. In *10th International Conference on Scientific Computing in Electrical Engineering*, 2014.

[2] A. Sandu and M. Günther. A generalized-structure approach to additive Runge-Kutta methods. *SIAM Journal on Numerical Analysis*, 53(1):17–42, 2015.

[3] M. Günther and A. Sandu. Multirate generalized additive Runge-Kutta methods. *Numerische Mathematik*, pages 1–28, 2015.

[4] A. Cardone, Z. Jackiewicz, A. Sandu, and H. Zhang. Extrapolation-based implicit-explicit general linear methods. *Numerical Algorithms*, In print, 2013.

[5] H. Zhang, A. Sandu, and S. Blaise. High order implicit–explicit general linear methods with optimized stability regions. *Submitted.*, 2014.

[6] H. Zhang, A. Sandu, and S. Blaise. Partitioned and implicit-explicit general linear methods for ordinary differential equations. *Journal of Scientific Computing*, 61(1):119–144, 2014.

[7] E.M. Constantinescu and A. Sandu. Extrapolated implicit-explicit time stepping. *SIAM Journal on Scientific Computing*, 31(6):4452–4477, 2010.

[8] A. Sandu and E. Constantinescu. Multirate time discretizations for hyperbolic partial differential equations. In *International Conference of Numerical Analysis and Applied Mathematics (ICNAAM 2009)*, volume 1168-1 of *American Institute of Physics (AIP) Conference Proceedings*, pages 1411–1414, 2009.

[9] A. Sandu and E.M. Constantinescu. Multirate Adams methods for hyperbolic equations. *Journal of Scientific Computing*, 38(2):229–249, 2009.

[10] E.M. Constantinescu and A. Sandu. Extrapolated multirate methods for differential equations with multiple time scales. *Journal of Scientific Computing*, 56(1):28–44, 2013.

[11] E. Hairer and G. Wanner. *Solving Ordinary Differential Equations II: Stiff and Differential-Algebraic Problems.* Springer, 2002.

[12] J. Rang and L. Angermann. New Rosenbrock W-methods of order 3 for partial differential algebraic equations of index 1. *BIT Numerical Mathematics*, 45(4):761–787, 2005.

[13] Y. Saad. *Iterative methods for sparse linear systems.* PWS Pub. Co., Boston, 1996.

[14] Henk A. van der Vorst. *Iterative Methods for Large Linear Systems.* Cambridge University Press, 2003.

[15] R. Weiner and B.A. Schmitt. Order results for Krylov-W methods. *Computing*, 61:69–89, 1998.

[16] J. Wensch. Krylov-ROW methods for DAEs of index 1 with applications to viscoelasticity. *Applied Numerical Mathematics*, 53:527–541, 2005.

[17] R. Weiner and B.A. Schmitt. Consistency of Krylov-W-methods in initial value problems. Technical Report 14, Fachbereich Mathematik und Informatik, Martin-Luther-Universitat Halle-Wittenberg, 1995.

[18] C.T. Kelley, I.G. Kevrekidis, and L. Qiao. Newton-Krylov solvers for time-steppers. arXiv:math/0404374v1, 2004.

[19] D.A. Knoll and D.E. Keyes. Jacobian-free Newton–Krylov methods: a survey of approaches and applications. *Journal of Computational Physics*, 193:357–397, 2004.

[20] I.M.G. Grotowskya and J. Ballmanna. Efficient time integration of Navier–Stokes equations. *Computers & Fluids*, 28(2):243–263, 1996.

[21] J. C. Schulze, P. J. Schmid, and J. L. Sesterhenn. Exponential time integration using Krylov subspaces. *International Journal for Numerical Methods in Fluids*, 60(6):561–609, 2008.

[22] K.J. Evans and D.A. Knoll. Temporal accuracy analysis of phase change convection simulations using the JFNK-SIMPLE algorithm. *International Journal for Numerical Methods in Fluids*, 55(637–653), 2007.

[23] L.O. Jay. A parallelizable preconditioner for the iterative solution of implicit Runge–Kutta-type methods. *Journal of Computational and Applied Mathematics*, 111(1-1):63–76, 1999.

[24] K. Dekker. Partitioned Krylov subspace iteration in implicit Runge–Kutta methods. *Linear Algebra and its Applications*, 431:488–494, 2009.

[25] A. Bouhamidi and K. Jbilou. Stein implicit Runge–Kutta methods with high stage order for large-scale ordinary differential equations. *Applied Numerical Mathematics*, 61:149–159, 2011.

[26] S.M. Hosseini and G. Hojjati. Matrix free MEBDF method for the solution of stiff systems of ODEs. *Mathematical and Computer Modelling*, 29(4):67–77, 1999.

[27] A.C. Hindmarsh, P.N. Brown, and G.D. Byrne. VODPK: Variable-coefficient ordinary differential equation solver with the Preconditioned Krylov method GMRES for the solution of linear systems. http://www.netlib.org/ode/vodpk.f, 2002.

[28] J. Huang, J. Jia, and M. Minion. Arbitrary order Krylov deferred correction methods for differential algebraic equations. *Journal of Computational Physics*, 221(2):739–760, 2007.

[29] A.C. Hindmarsh. LSODKR, stiff ordinary differential equations (ODE) system solver with Krylov iteration with rootfinding, 1991.

[30] A.C. Hindmarsh and P.N. Brown. LSODPK, ordinary differential equations solver for stiff and nonstiff system with Krylov corrector iteration, 1991.

[31] P. N. Brown, A. C. Hindmarsh, and L. R. Petzold. Using Krylov methods in the solution of large-scale differential-algebraic systems. *SIAM Journal on Scientific Computing*, 15:1467–1488, 1994.

[32] M.A. Botchev. Block Krylov subspace exact time integration of linear ODE systems. part 1: algorithm description. http://arxiv.org/abs/1109.5100, 2011.

[33] M.A. Botchev, G.L.G. Sleijpen, and H.A. van der Vors. Low-dimensional Krylov subspace iterations for enhancing stability of time-step integration schemes. Technical report, University Utrecht, 2001.

[34] J. Huang, J. Jia, and M.L. Minion. Accelerating the convergence of spectral deffered correction methods. *Journal of Computational Physics*, 214:633–656, 2006.

[35] J. Huang, J. Jia, and M.L. Minion. Arbitrary order Krylov deffered corrction methods for differential algebraic equations. *Journal of Computational Physics*, 221:739–760, 2007.

[36] B.A. Schmitt and R. Weiner. Matrix free W-methods using a multiple Arnoldi iteration. *Applied Numerical Mathematics*, 18:307–320, 1995.

[37] H. Podhaisky, R. Weiner, and B.A. Schmitt. Numerical experiments with Krylov integrators. *Applied Numerical Mathematics*, 28:413–425, 1997.

[38] R.Weiner, B.A. Schmitt, and H. Podhaisky. ROWMAP–a ROW-code with Krylov techniques for large stiff ODEs. *Applied Numerical Mathematics*, 25:303–319, 1997.

[39] K. Strehmel, R. Weiner, and M. Büttner. Order results for Rosenbrock type methods on classes of stiff equations. *Numerische Mathematik*, 59(7):723–737, 1991.

[40] B.A. Schmitt and R. Weiner. Polynomial preconditioning in Krylov-ROW-methods. *Applied Numerical Mathematics*, 28(2-4):427–437, 1998.

[41] P. Novati. Some secant approximations for Rosenbrock W-methods. *Applied Numerical Mathematics*, 58(3):195–211, 2008.

[42] M. Hochbruck and C. Lubich. On Krylov subspace approximations to the matrix exponential operator. *SIAM Journal on Numerical Analysis*, 34(5):1911–1925, 1997.

[43] M. Hochbruck, A. Ostermann, and J. Schweitzer. Exponential Rosenbrock-type methods. *SIAM Journal on Numerical Analysis*, 47:786–803, 2009.

[44] M. Caliaria and A. Ostermann. Implementation of exponential Rosenbrock-type integrators. *Applied Numerical Mathematics*, 3–4:568–581, 2009.

[45] M. Tokman. Efficient integration of large stiff systems of ODEs with exponential propagation iterative (EPI) methods. *Journal of Computational Physics*, 213(2):748–776, 2006.

[46] M. Hochbruck, C. Lubich, and H. Selhofer. Exponential integrators for large systems of differential equations. *SIAM Journal of Scientific Computing*, 19(5):1552—1574, 1998.

[47] M. Hochbruck and A. Ostermann. Exponential integrators. *Acta Numerica*, 19:209–286, 2012.

[48] E. Hairer, S.P. Norsett, and G. Wanner. *Solving Ordinary Differential Equations I: Nonstiff Problems*. Springer, 2008.

[49] P.N. Brown, H.F. Walker, R. Wasyk, and C.S. Woodward. On using approximate finite differences in matrix-free Newton-Krylov methods. *SIAM Journal on Numerical Analysis*, 46(4):1892–1911, 2008.

[50] C. Lubich and A. Ostermann. Linearly implicit time discretization of nonlinear parabolic equations. *IMA Journal on Numerical Mathematics*, 15:555–583, 1995.

[51] Edward N.. Lorenz. Predictability – a problem partly solved. In *Predictability of Weather and Climate*. Cambridge University Press, 2006.

[52] Jan S. Hesthaven and Tim Warburton. *Nodal Discontinuous Galerkin Methods*. Texts in Applied Mathematics. Springer, 2008.

[53] Michael W. Gery, Gary Z. Whitten, James P. Killus, and Marcia C. Dodge. A photochemical kinetics mechanism for urban and regional scale computer modeling. *Journal of Geophysical Research: Atmospheres*, 94(D10):12925–12956, 1989.

[54] A. Sandu, J.G. Verwer, M. Van Loon, G.R. Carmichael, F.A. Potra, D. Dabdub, and J.H. Seinfeld. Benchmarking stiff ode solvers for atmospheric chemistry problems-i. implicit vs explicit. *Atmospheric Environment*, 31(19):3151 – 3166, 1997. ¡ce:title¿EUMAC: European Modelling of Atmospheric Constituents.

[55] Richard Liska and Burton Wendroff. Composite schemes for conservation laws, 1997.

[56] Paul Tranquilli and Adrian Sandu. Rosenbrock-krylov methods for large systems of differential equations. *SIAM J. Scientific Computing*, 36(3), 2014.

[57] M. Tokman. A new class of exponential propagation iterative methods of Runge–Kutta type (EPIRK). *Journal of Computational Physics*, 230:8762—8778, 2011.

[58] J. Loffeld and M. Tokman. Comparative performance of exponential, implicit, and explicit integrators for stiff systems of ODEs. *Journal of Computational and Applied Mathematics*, 241(0):45 – 67, 2013.

[59] N. Higham. The scaling and squaring method for the matrix exponential revisited. *SIAM Journal on Matrix Analysis and Applications*, 26(4):1179–1193, 2005.

[60] Philippe Chartier, Ernst Hairer, and Gilles Vilmart. Algebraic structures of B-series. *Found. Comput. Math.*, 10(4):407–427, August 2010.

[61] J.C. Butcher. Trees, b-series and exponential integrators. *IMA Journal of Numerical Analysis*, 30:131—140, 2010.

[62] G. Rainwater and M. Tokman. A new class of split exponential propagation iterative methods of rungekutta type (sEPIRK) for semilinear systems of ODEs. *Journal of Computational Physics*, 269(0):40 – 60, 2014.

[63] Vu Thai Luan and Alexander Ostermann. Explicit exponential Runge Kutta methods of high order for parabolic problems. *Journal of Computational and Applied Mathematics*, 256(0):168 – 179, 2014.

[64] Roger B. Sidje. Expokit: a software package for computing matrix exponentials. *ACM Trans. Math. Softw.*, 24(1):130–156, March 1998.

[65] Paul Tranquilli and Adrian Sandu. Exponential-Krylov methods for ordinary differential equations. *Journal of Computational Physics*, 278:31 – 46, 2014.

[66] Mario Bttner, Bernhard A. Schmitt, and Rdiger Weiner. W-methods with automatic partitioning by krylov techniques for large stiff systems. *SIAM Journal on Numerical Analysis*, 32(1):260–284, 1995.

[67] I. M. Navon and R. de Villiers. The application of the turkelzwas explicit large timestep scheme to a hemispheric barotropic model with constraint restoration. *Monthly Weather Review*, 115(5):1036–1052, 1987.

[68] I.M. Navon and Jian Yu. Exshall: A turkel-zwas explicit large time-step fortran program for solving the shallow-water equations in spherical coordinates. *Computers & Geosciences*, 17(9):1311 – 1343, 1991.

[69] I. Grooms and K. Julien. Linearly implicit methods for nonlinear PDEs with linear dispersion and dissipation. *Journal of Computational Physics*, 230(9):3630 – 3650, 2011.

[70] M.P. Calvo, J. de Frutos, and J. Novo. Linearly implicit Runge-Kutta methods for advection-reaction-diffusion equations. *Applied Numerical Mathematics*, 37(4):535–549, 2001.

[71] G. Akrivis, O. Karakashian, and F. Karakatsani. Linearly implicit methods for nonlinear evolution equations. *Numerische Mathematik*, 94:403–418, 2003.

[72] G. Akrivis and M. Crouzeix. Linearly implicit methods for nonlinear parabolic equations. *Mathematics of Computation*, 73(246):pp. 613–635, 2004.

[73] Paul Tranquilli, S. Ross Glandon, Arash Sarshar, and Adrian Sandu. Analytical jacobian-vector products for the matrix-free time integration of partial differential equations. *Journal of Computational and Applied Mathematics*, pages –, 2016.

[74] U.M. Ascher, S.J. Ruuth, and R.J. Spiteri. Implicit-explicit Runge-Kutta methods for time-dependent partial differential equations. *Applied Numerical Mathematics*, 25:151–167, 1997.

[75] J. Douglas Jr. Alternating direction methods for three space variables. *Numerische Mathematik*, 4(1):41–63, 1962.

[76] D. Peaceman and H. Rachford Jr. The numerical solution of parabolic and elliptic differential equations. *Journal of the Society for Industrial and Applied Mathematics*, 3(1):28–41, 1955.

Appendix A

A.1 Order conditions for exponential-W methods.

Theorem 8 (Order conditions for exponential-W methods)**.** *An exponential-W method with general form (3.2) has order p iff the following order conditions hold:*

$$\sum_j b_j \, \Phi_j(\tau) = P_\tau(\tau) \quad \forall \, \tau \in TW \quad \textit{with} \ |\tau| \leq p \, . \tag{A.1}$$

Here $|\tau|$ is the order, or number of vertices of the tree τ, and $\Phi_j(\tau)$ and $P_\tau(\tau)$ can be computed using Algorithm 5; they are shown in Figure A.1 for $p \leq 4$.

Proof. The proof follows from our discussion in section 3.3, the near equivalence of order conditions for exponential-W and Rosenbrock-W methods [46], and from the order conditions of Rosenbrock-W methods [11, Theorem 7.7]. $\square$

i	$F(\tau_i)$	$\Phi(\tau)$	$P_\tau(\gamma)$
1	f^J	1	1
2	$f^J_K f^K$	$\sum \alpha_{jk}$	$\frac{1}{2}$
3	$\mathbf{A}_{JK} f^K$	$\sum \gamma_{jk}$	$\frac{-\gamma}{2}$
4	$f^J_{KL} f^K f^L$	$\sum \alpha_{jk}\alpha_{jl}$	$\frac{1}{3}$
5	$f^J_K f^K_L f^L$	$\sum \alpha_{jk}\alpha_{kl}$	$\frac{1}{6}$
6	$f^J_K \mathbf{A}_{KL} f^L$	$\sum \alpha_{jk}\gamma_{kl}$	$\frac{-\gamma}{4}$
7	$\mathbf{A}_{JK} f^K_L f^L$	$\sum \gamma_{jk}\alpha_{kl}$	$\frac{-\gamma}{4}$
8	$\mathbf{A}_{JK}\mathbf{A}_{KL} f^L$	$\sum \gamma_{jk}\gamma_{kl}$	$\frac{\gamma^2}{3}$
9	$f^J_{KLM} f^K f^L f^M$	$\sum \alpha_{jk}\alpha_{jl}\alpha_{jm}$	$\frac{1}{4}$
10	$f^J_{KL} f^L_M f^M f^K$	$\sum \alpha_{jk}\alpha_{jl}\alpha_{jm}$	$\frac{1}{8}$
11	$f^J_{KL}\mathbf{A}_{LM} f^M f^K$	$\sum \alpha_{jk}\alpha_{jl}\gamma_{lm}$	$\frac{-\gamma}{6}$
12	$f^J_K f^K_{LM} f^M f^L$	$\sum \alpha_{jk}\alpha_{kl}\alpha_{km}$	$\frac{1}{12}$
13	$\mathbf{A}_{JK} f^K_{LM} f^M f^L$	$\sum \gamma_{jk}\alpha_{kl}\alpha_{km}$	$\frac{-\gamma}{6}$
14	$f^J_K f^K_L f^L_M f^M$	$\sum \alpha_{jk}\alpha_{kl}\alpha_{lm}$	$\frac{1}{24}$
15	$f^J_K f^K_L A_{LM} f^M$	$\sum \alpha_{jk}\alpha_{kl}\gamma_{lm}$	$\frac{-\gamma}{12}$
16	$f^J_K \mathbf{A}_{KL} f^L_M f^M$	$\sum \alpha_{jk}\gamma_{kl}\alpha_{lm}$	$\frac{-\gamma}{12}$
17	$f^J_K \mathbf{A}_{KL}\mathbf{A}_{LM} f^M$	$\sum \alpha_{jk}\gamma_{kl}\gamma_{lm}$	$\frac{\gamma^2}{6}$
18	$\mathbf{A}_{JK} f^K_L f^L_M f^M$	$\sum \gamma_{jk}\alpha_{kl}\alpha_{lm}$	$\frac{-\gamma}{12}$
19	$\mathbf{A}_{JK} f^K_L \mathbf{A}_{LM} f^M$	$\sum \gamma_{jk}\alpha_{kl}\gamma_{lm}$	$\frac{\gamma^2}{8}$
20	$\mathbf{A}_{JK}\mathbf{A}_{KL} f^L_M f^M$	$\sum \gamma_{jk}\gamma_{kl}\alpha_{lm}$	$\frac{\gamma^2}{6}$
21	$\mathbf{A}_{JK}\mathbf{A}_{KL}\mathbf{A}_{LM} f^M$	$\sum \gamma_{jk}\gamma_{kl}\gamma_{lm}$	$\frac{-\gamma^3}{4}$

Table A.1: Order conditions for exponential-W methods with general form (3.2) and $p \leq 4$.