Spring 5-15-2021

High-order Flexible Multirate Integrators for Multiphysics Applications

Rujeko Chinomona
Southern Methodist University, rchinomona@smu.edu

HIGH-ORDER FLEXIBLE MULTIRATE INTEGRATORS FOR MULTIPHYSICS APPLICATIONS

Approved by:

Dr. Daniel Reynolds,
Professor of Mathematics

Dr. Usama El Shamy,
Associate Professor of Civil Engineering

Dr. Thomas Hagstrom,
Professor of Mathematics

Dr. Johannes Tausch,
Professor of Mathematics

HIGH-ORDER FLEXIBLE MULTIRATE INTEGRATORS FOR MULTIPHYSICS APPLICATIONS

A Dissertation Presented to the Graduate Faculty of the

Dedman College: School of Humanities and Sciences

Southern Methodist University

in

Partial Fulfillment of the Requirements

for the degree of

Doctor of Philosophy

with a

Major in Computational and Applied Mathematics

by

Rujeko Chinomona

B.S., Georgia College and State University
M.A., Rice University

May 15, 2021

ACKNOWLEDGMENTS

This thesis is a great achievement, but it is even better because I can share this success with those that made a long and challenging journey bearable.

First, I extend my heartfelt appreciation to my thesis advisor, Professor Daniel Reynolds, who introduced me to the corner of applied mathematics that I have now made my own. Through the years he has made a great impact on the trajectory of my career through instruction, guidance, mentorship, encouragement, and especially through exposure to a variety of opportunities. Having such an attentive advisor has truly made all the difference.

My thesis committee Professor Usama El Shamy, Professor Thomas Hagstrom, and Professor Johannes Tausch were instrumental in the review and completion of this thesis and I am grateful for their insightful comments. I also had the pleasure of taking classes from both Professor Hagstrom and Professor Tausch in key areas affecting my research.

My graduate experiences were also enriched by my research collaborators. I have had fruitful collaborations with Dr. Vu Thai Luan that resulted in the expansion of the class of problems I worked on in this thesis. It is a privilege to have worked with Dr. David Gardner and Dr. Carol Woodward, whose unique set of experiences in applied mathematics edified my own experience, equipped me with versatile skills, and helped me navigate different aspects of being an applied mathematician.

Many thanks to the SMU community, specifically the Mathematics Department for providing an environment in which I could pursue my research interests and earn this achievement. I felt supported, celebrated, and encouraged, both academically and socially. As a testament to the financial support I received from SMU, I was a recipient of the University Ph.D. fellowship, the Dean's Dissertation Fellowship, and the Haberman fellowship. Support for my work also came from the Scientific Discovery through Advanced Comput-

ing (SciDAC) project "Frameworks, Algorithms and Scalable Technologies for Mathematics (FASTMath)," funded by the U.S. Department of Energy Office of Advanced Scientific Computing Research and National Nuclear Security Administration, under Lawrence Livermore National Laboratory subcontract B626484 and DOE award DE-SC0021354.

In Shona we say, "*munhu munhu nevanhu*" which directly translates to "*a person is a person among other people*" and speaks to the importance of community and relations. So I will be remiss if I do not extend gratitude to my family and friends who have shown up for me time and again to inspire, to prop me up, and help me believe in myself. Since I left Zimbabwe close to 11 years ago, my now late father Crabbie, my mother Petronella, and my siblings Anesu and Rutendo, as well as my extended family have never ceased to support me. I have also leaned on my partner Bradley Johnson and I am thankful for his selflessness, patience, and encouragement.

Thank you to everyone that has been on this never-ending journey of learning with me, even back to when it all started!

Chinomona , Rujeko

B.S., Georgia College and State University
M.A., Rice University

High-Order Flexible Multirate Integrators for Multiphysics Applications

Advisor: Professor Daniel Reynolds
Doctor of Philosophy degree conferred May 15, 2021
Dissertation completed April 16, 2021

Traditionally, time integration methods within multiphysics simulations have been chosen to cater to the most restrictive dynamics, sometimes at a great computational cost. Multirate integrators accurately and efficiently solve systems of ordinary differential equations that exhibit different time scales using two or more time steps. In this thesis, we explore three classes of time integrators that can be classified as one-step multi-stage multirate methods for which the slow dynamics are evolved using a traditional one step scheme and the fast dynamics are solved through a sequence of modified initial value problems. Practically, the fast dynamics are subcycled using a small time step and any time integration scheme of sufficient order. The overall contributions of this thesis fall into two main categories. First, we focus on the derivation of a novel class of integrators which we call implicit-explicit multirate infinitesimal generalized-structure additive Runge–Kutta (IMEX-MRI-GARK) methods. We present third and fourth order conditions for IMEX-MRI-GARK methods, consider their stability properties, and apply our derived methods to several test problems. In the second part, we discuss the numerical implementation of recently developed multirate exponential Runge–Kutta (MERK) and multirate exponential Rosenbrock (MERB) methods and their application to various test problems. MERK and MERB methods are to date some of the

highest order multirate methods, with orders of convergence up to fifth and sixth order respectively. We discuss our selection of test problems for exercising these methods, present ways to experimentally determine an optimal ratio between the slow and fast time scales, and compare the performance of several multirate methods.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

To my late father, Crabbie, who was my first inspiration and would have been ecstatic of this accomplishment, and to my mother, Petronella, who was there from the beginning and whose faith knows no bounds.

Chapter 1

Introduction

1.1. Multiscale nature of multiphysics problems

One of the defining features of multiphysics applications is the coupling of multiple physical phenomena. Examples of multiphysics processes are ubiquitous in the sciences for example in the atmospheric sciences, multiphase fluid flows, material sciences, or cosmological sciences. The simultaneous interaction of multiple physical processes often means multiphysics applications are multiscale in both space and time. The study of multiscale problems is very active with efforts focused on both creating multiscale models and accompanying numerical algorithms. A major driving force in the research of multiscale problems are the advances in high performance computing which have made it feasible for scientists to explore large-scale simulations and continue to add complexity to existing models. However, multiscale problems remain a challenge to many traditional numerical algorithms which were historically designed to cater to some aspects of the physics but can fail to capture the complete picture satisfactorily in terms of accuracy, stability, and computational efficiency.

1.2. Multiple temporal scales

In this thesis we focus on the challenge of dynamical systems with components of differing characteristic timescales. Differing temporal scales can occur as a result of actual differences in rates of evolution in a system of ordinary differential equations (ODEs), for example, in the simulation of cloud microphysics which are part of a larger system of climate processes, condensation (the autoconversion of cloud droplets to form rain) occurs at a slower pace than sedimentation (the falling of cloud particles relative to air) which can occur at rates up to

a hundred times faster. Uneven spatial discretizations can also result in a mixture of stable time steps leading to stiff terms - those that need a small time step for stable integration and non-stiff terms that can be evolved with a larger time step. Multirate time integration schemes are numerical algorithms that evolve a system of ODEs using two or more time step sizes. For many applications of multirate schemes, the main motivation for using a large time step for the slow dynamics is their computational cost. The less frequently we must evaluate the slow dynamics, the less the computational cost, additionally, within parallel implementations, the less the communication costs.

1.3. Motivation for high-order methods

Multirate schemes that are frequently used in multiphysics simulations today are operator splitting approaches that often have low accuracy (commonly up to order 2 convergence), limited coupling between different components leading to poor stability, and low computational efficiency. The case for high-order multirate schemes that have stronger coupling between components while also providing increased computational efficiency is therefore immediate. The need for these methods is also being propelled by the eminent migration of many multiphysics simulations from low-order spatial resolutions to high-order resolutions, this is particularly the case in climate simulations [47] .

1.4. Aim of thesis

The body of work that constitutes this thesis is focused on introducing new classes of efficient high-order multirate schemes, analysing their convergence and stability properties, investigating their numerical implementation, and devising ways to rigorously test them. The methods explored in this body of work can be classified as one-step multi-stage multirate methods for which the slow dynamics are evolved using some base integration scheme (implicit-explicit additive Runge–Kutta, exponential Runge–Kutta, exponential Rosenbrock) and the fast dynamics are assumed to be solved exactly through use of modified initial value

problems (IVPs). Practically, the fast dynamics are subcycled using any integration scheme of sufficient order. Subcycling with a much smaller time step defines the 'infinitesimal' nature of these methods. Throughout this thesis, we will therefore refer to these methods as multirate infinitesimal step (MIS)-type methods, named after some of the pioneering methods in the field which we further explore in Section 2.2 .

Remark 1.4.1. *The terms ordinary differential equation (ODE) and initial value problem (IVP) are used interchangeably throughout this thesis.*

1.5. Multirate splitting

Throughout this thesis we consider systems of ODEs of the form:

$$y'(t) = f(t,y) = \sum_{q=1}^{N} f^{\{q\}}(t,y), \quad y(t_0) = y_0, \quad t \geq t_0, \quad y \in \mathbb{R}^n, \tag{1.1}$$

where the right hand side can be split additively into N components, separated by stiffness, nonlinearity, rate of evolution, or computational cost. We focus on problems with $N = 2$ or $N = 3$. In the case of two components, we have a slow component $f^{\{S\}}$ and a fast component $f^{\{F\}}$

$$y'(t) = f^{\{S\}}(t,y) + f^{\{F\}}(t,y), \quad y(t_0) = y_0, \quad t \geq t_0, \quad y \in \mathbb{R}^n. \tag{1.2}$$

The slow dynamics are treated with a large time step H while the fast dynamics use a much smaller time step $h = H/m$, where $m \geq 1$ is the integer timescale separation factor also known as a subcycling factor. We note that an additive splitting also encompasses the case of component partitioned systems. A component partitioned system is defined as

$$\begin{bmatrix} y^{\{F\}} \\ y^{\{S\}} \end{bmatrix}' = \begin{bmatrix} f^{\{F\}}(t, y^{\{F\}}, y^{\{S\}}) \\ f^{\{S\}}(t, y^{\{F\}}, y^{\{S\}}) \end{bmatrix}, \quad \begin{bmatrix} y^{\{F\}}(t_0) \\ y^{\{S\}}(t_0) \end{bmatrix} = \begin{bmatrix} y_0^{\{F\}} \\ y_0^{\{S\}} \end{bmatrix} \tag{1.3}$$

which we can write equivalently in the additive form

$$\begin{bmatrix} y^{\{F\}} \\ y^{\{S\}} \end{bmatrix}' = \begin{bmatrix} 0 \\ f^{\{S\}}(t, y^{\{F\}}, y^{\{S\}}) \end{bmatrix} + \begin{bmatrix} f^{\{F\}}(t, y^{\{F\}}, y^{\{S\}}) \\ 0 \end{bmatrix}, \quad \begin{bmatrix} y^{\{F\}}(t_0) \\ y^{\{S\}}(t_0) \end{bmatrix} = \begin{bmatrix} y_0^{\{F\}} \\ y_0^{\{S\}} \end{bmatrix}. \quad (1.4)$$

When there are additional variations in stiffness of the slow component, we can further split the slow component into a slow-stiff component $f^{\{I\}}$ to be integrated implicitly and a slow-nonstiff component $f^{\{E\}}$ to be integrated explicitly:

$$y'(t) = f^{\{I\}}(t, y) + f^{\{E\}}(t, y) + f^{\{F\}}(t, y), \quad y(t_0) = y_0, \quad t \geq t_0, \quad y \in \mathbb{R}^n. \quad (1.5)$$

The three-way additively split problem provides extra flexibility in how the slow operators are treated which can be particularly useful for systems that have an advection-diffusion-reaction like structure. In this scenario, both advection and diffusion constitute the slow dynamics while quickly evolving reactions are the fast dynamics. We note that similar mixed implicit+explicit (IMEX) treatment of the fast time scale is immediately possible with any (MIS)-type method, since these techniques do not place any constraints on how the modified IVPs at the fast time scale are solved.

1.6. Review of multirate methods

In this section we give a general survey of multirate methods with the intention of furnishing the overall context for methods explored in this thesis. Although we list a variety of multirate methods, the overall goal of this section is to highlight examples of methods whose theories we extend, their orders of accuracy, and how flexible they are in treating the fast and slow time scales. Within the multirate community there has been different terminology for describing methods that apply two or more time steps to a system of ODEs, with different fields of application coining separate definitions. In this thesis we almost exclusively use the

term 'multirate methods' which is common in numerical analysis. However, we note that the literature of multirate methods also includes some other commonly used variations like multiple time stepping methods, variable time stepping methods, or subcycling algorithms.

1.6.1. Operator splitting approaches

Perhaps the most widely used in application sciences are operator splitting approaches to multirating. Starting with (1.2) (though the process can be generalized to (1.5) and (1.1)), the overall idea of operator splitting is to treat the original problem as different subproblems. In the case of multirate methods, these subproblems allow different time step sizes. Thus, (1.2) can be broken up into two equations

$$y'(t) = f^{\{S\}}(t, y), \quad \text{and} \quad y'(t) = f^{\{F\}}(t, y), \tag{1.6}$$

that are then solved sequentially, with the solution to the first subproblem serving as an initial condition in the second problem. The simplest operator splitting approach for (1.2) is Lie-Trotter, for example as presented in [76]. To advance from t_n to $t_n + H$, Lie-Trotter proceeds as follows:

$$y_{n+1}^{(1)} = y_n + H f^{\{S\}}(t_n, y_n), \tag{1.7}$$

$$\text{Solve} \begin{cases} v(0) = y_{n+1}^{(1)}, \\ v'(\theta) = f^{\{F\}}(t_n + \theta, v), \text{ for } \theta \in [0, H], \end{cases}$$

$$y_{n+1} = v(H).$$

Here $y_{n+1}^{(1)}$ is the solution to the slow subproblem, computed with a forward Euler step, that serves as the initial condition to the fast subproblem that can in turn be solved with another forward Euler step or otherwise, but with smaller time steps h. The Lie-Trotter approach is

at most first-order accurate. Another commonly used approach is the second-order accurate Strang-Marchuk [75, 103] method, e.g.

$$
\begin{aligned}
y_{n+1}^{(1)} &= y_n + \tfrac{H}{4} f^{\{S\}}\left(t_n, y_n\right) \\
&\quad + \tfrac{H}{4} f^{\{S\}}\left(t_n + \tfrac{H}{2}, y_n + \tfrac{H}{2} f^{\{S\}}\left(t_n, y_n\right)\right), \\
&\text{Solve} \begin{cases} v(0) = y_{n+1}^{(1)}, \\ v'(\theta) = f^{\{F\}}\left(t_n + \theta, v\right), \text{ for } \theta \in [0, H], \end{cases} \\
y_{n+1}^{(2)} &= v(H), \\
y_{n+1} &= y_{n+1}^{(2)} + \tfrac{H}{4} f^{\{S\}}\left(t_n + \tfrac{H}{2}, y_{n+1}^{(2)}\right) \\
&\quad + \tfrac{H}{4} f^{\{S\}}\left(t_{n+1}, y_{n+1}^{(2)} + \tfrac{H}{2} f^{\{S\}}\left(t_n + \tfrac{H}{2}, y_{n+1}^{(2)}\right)\right).
\end{aligned}
\tag{1.8}
$$

We note that here, the updates $y_n \to y_{n+1}^{(1)}$ and $y_{n+1}^{(2)} \to y_{n+1}$ correspond to using the explicit Heun method for a half time-step each. Though our presentation here is limited to the case where the slow dynamics are evolved by an explicit scheme, both approaches are quite flexible in how they treat both the fast and slow dynamics, allowing implicit, explicit, or IMEX treatment of both time scales. In fact, to our knowledge, Lie-Trotter and Strang-Marchuk are the only multirate methods prior to this thesis that have demonstrated application of IMEX splittings at the slow time scale. A major drawback of both approaches is the low-order of accuracy. Higher-order operator splitting methods, for example the fourth-order symplectic integrators by Yoshida [108] that target Hamiltonian systems, are possible. However, many higher-order operator splitting approaches involve backward in time integration which can lead to instabilities or inefficiencies in implementation [32].

1.6.2. General overview of multirate methods

One of the earliest multirate works based on Runge–Kutta methods dates back to Rice in 1960 [80]. Andrus also investigated multirate Runge–Kutta based methods starting with [2] and following up with investigations into stability [1]. Other authors that are often considered among the first to closely look at multirate methods are Gear and Wells, whose approaches are based off linear multistep methods. In their studies [35, 36] from 1974 and onwards, Gear and Wells gave detailed theoretical considerations for multirate methods that evolve the slow and fast time scale separately, but share information between the different dynamics through linear interpolation. Among these theoretical contributions were advances in error analysis, linear stability theory, and time adaptivity.

There have been several multirate methods based on traditional time integration techniques in the literature, including linear multistep [36, 89], extrapolation [7, 20, 28, 22], Runge–Kutta [64, 63, 41] and Rosenbrock approaches [43, 96]. With the exception of extrapolation methods that can theoretically be of arbitrary order, most of these approaches have been limited to third-order convergence, with explicit slow and fast components, and little to no flexibility for implicit or IMEX treatment at the slow time scale. Although extrapolation based methods can reach arbitrary accuracy, they can be cost prohibitive. In a recent preprint, Bartel and Günther [5] provide a unified order condition theory for interpolation and extrapolation multirate methods.

Several multirate methods [21, 90, 41] have also utilized partitioned Runge–Kutta theory applied to methods of the form (1.3), whose theory is outlined in [45]. The generalized-structure additive Runge–Kutta (GARK) framework by Sandu and Günther in [92] builds upon ideas from partitioned Runge–Kutta theory to form a more generalized class of additive Runge–Kutta methods. Subsequent work from Günther and Sandu in the derivation of multirate GARK (MRGARK) theory [42] has led to several new multirate methods of higher-order. The GARK formalism has allowed development of several multirate schemes with up to fourth-order convergence. Multirate GARK methods from Sarshar et al. [94] are among

the first of this kind to involve implicit solutions for both the slow and fast time scales and investigate time step control. Time adaptivity for multirate GARK schemes has also been investigated by Bremicker-Trübelhorn and Ortleb [8, 9].

A particularly successful recipe to constructing multirate methods is coupling fast and slow components through the infinitesimal step approach to form what we refer to as MIS-type methods. Knoth and Wolke [61] originally designed the infinitesimal step approach with an explicit treatment of the slow time scale and the 'exact' solution of the fast time scale through solving modified ODEs that bring in information from the slow dynamics. In implementation, the fast time scale can then be solved with any integrator: implicit, explicit or IMEX, and is typically subcycled using smaller steps than the slow time scale. Multiple third-order MIS-type methods were constructed using this idea [98, 99, 107], with the term MIS being coined by Wensch, Knoth, and Galant in [107]. Recently, since the introduction of GARK and MGARK order theory, several new multirate methods have used a combination of GARK theory and MIS structure to form fourth-order approaches. Sexton and Reynolds investigated some of the first fourth-order explicit at slow MIS-type methods in [102]. Sandu in his 2019 paper [88] introduced the first MIS-type fourth-order implicit at slow schemes called multirate infinitesimal GARK (MRI-GARK) methods. More investigations into implicit at slow MIS-type methods, including contributions to stability analysis followed from Roberts et al. [84, 81]. Bauer and Knoth also extended the original MIS methods to fourth-order [6]. By leveraging exponential Runge–Kutta theory, Luan, Chinomona, and Reynolds [68] (included in this thesis) developed the very first fifth-order MIS-type methods. Some of the most recent work on MIS-type methods is in a preprint by Chinomona and Reynolds [15] (included in this thesis) and extends MRI-GARK methods to have a more flexible IMEX treatment of the slow time scale. Preprints from Roberts et al. [82] and [40] also investigate new MIS-type multirate methods.

1.7. Thesis outline and summary of contributions

This dissertation is comprised of six chapters, in the structure: introduction (Chapter 1), background theory (Chapter 2), new results (Chapters 3-6), and conclusion (Chapter 7). The following is a brief summary of contributions:

Chapter 1 motivates the need for high-order multirate schemes, discusses the different multirate splittings, and gives a general overview of multirate methods.

Chapter 2 introduces some of the building blocks of the classes of methods explored in this thesis, namely the GARK framework, MIS methods, and multirate infinitesimal GARK methods.

Chapter 3 advances the theory of multirate methods through the introduction of a new class of multirate methods that leverage GARK theory and extend the ideas of MIS methods to three-way additively split ODEs. We develop methods called implicit-explicit multirate infinitesimal GARK (IMEX-MRI-GARK) that involve an IMEX splitting at the slow time scale. IMEX-MRI-GARK methods can be expressed within the GARK framework leading to the derivation of third- and fourth-order conditions. There is currently no consistent way of analyzing linear stability for multirate methods so we extend the concept of joint stability which was first applied to two-step Runge–Kutta methods by Zharovsky, Sandu, and Zhang [111] and apply it to IMEX-MRI-GARK methods. We construct two third-order methods (with one optimized for joint stability), and one fourth-order method. Numerical simulations of IMEX-MRI-GARK methods in MATLAB and C confirm convergence rates, provide insight into the parallel scalability of IMEX-MRI-GARK methods to larger problem sizes, and demonstrate efficiency and performance in comparison with legacy IMEX multirate approaches and implicit MRI-GARK methods. This work has been submitted for publication and is currently in the review process.

Chapter 4 takes a closer look at joint stability and the construction of fourth-order methods optimized for stability. We derive a new set of fourth-order coefficients, present stability plots that satisfy a relaxed definition of joint stability, and demonstrate comparable

orders of convergence and performance to IMEX-MRI-GARK methods in Chapter 3, but with improved stability at larger step sizes. This corresponds to some of the most recent work in this thesis and will hopefully form the backbone of another publication.

Chapter 5 introduces a new approach for constructing higher-order (MIS)-type multirate methods – multirate exponential Runge–Kutta (MERK) methods. MERK methods are built on a slow base of explicit exponential Runge–Kutta methods and follow the same structure as MIS methods of evolving the fast dynamics using modified ODEs. Modified ODEs replace the often costly matrix function evaluations associated with exponential Runge–Kutta methods. We provide the underlying theory of MERK methods, the MERK algorithm and convergence results. In addition, we present second-, third-, fourth-, and the very first fifth-order methods of MIS-type. This chapter contains our published work in the SIAM Journal on Scientific Computing [68] on this topic, and the contributions to note for the purposes of this thesis are in the numerical implementations of MERK methods. We investigate test problem choice for analyzing MERK methods or any multirate method, introduce ways to find the optimal time scale separation factors for each method and each test problem, and run verification tests crucial to the overall analysis of the methods.

Chapter 6 introduces another new approach for constructing even higher-order (MIS)-type multirate methods – multirate exponential Rosenbrock (MERB) methods. MERB methods are also of MIS-type and closely follow the same ideas in their derivation as MERK methods, however these start with an exponential Rosenbrock slow base method. We discuss the derivation of MERB methods and present methods up to sixth-order. As with the MERK chapter, the overall contributions to this thesis are in the numerical implementations of MERB methods ; we are currently drafting this manuscript for submission later this Spring. MERB methods are unlike other MIS-type methods because they dictate the splitting into fast and slow components through linearization of the right hand side at each time step (dynamic linearization). We present two test problems that best illustrate the performance of MERB methods using this somewhat restrictive splitting. We confirm convergence rates

for MERB methods and show their competitiveness compared to MERK and MRI-GARK methods. Our approach in comparing efficiency and performance results involves considering both dynamic linearization and other more common splittings for MERK and MRI-GARK methods. Another major highlight of our work in this chapter is our demonstration that MERK methods can be applied to nonlinear problems through the dynamic linearization approach.

Chapter 7 is the final chapter of this thesis and gives an overall conclusion and discussion of future works.

Chapter 2

Background Theory

In this chapter, we introduce some of the fundamental building blocks for the multirate methods we explore in this thesis, particularly those in Chapter 3. First, we describe generalized-structure additive Runge–Kutta (GARK) theory for representing families of general and additive Runge–Kutta methods. Our presentation of GARK methods is geared towards its use in Chapter 3. Next, we discuss multirate infinitesimal step (MIS) methods, presenting the unifying idea for all multirate methods explored in this thesis, and making connections with GARK theory. Lastly, a combination of GARK and MIS theory results in multirate infinitesimal GARK (MRI-GARK) methods that can handle implicitness at the slow time scale and can attain fourth-order convergence. We describe their order conditions, linear stability considerations, and construction. Our work in Chapter 3 directly extends MRI-GARK methods to allow an IMEX treatment of the slow time scale.

2.1. Generalized-structure additive Runge–Kutta theory

In this section we introduce the GARK framework by Sandu and Günther [92]. The GARK framework generalizes additive Runge–Kutta theory to allow flexibility in the stage values passed into different right hand side components. It was designed as a tool upon which time integration schemes for multiphysics processes like multirate methods can easily be built. In this thesis we use GARK theory to derive order conditions for the class of multirate methods we explore in Chapter 3. The GARK formulation applies to ODEs with additively split right hand sides (1.1). Here we give the GARK representation for the case when the right hand side is split into three components (1.5).

One step of a GARK scheme from t_n to $t_n + H$ is composed of stages $Y_i^{\{I\}}, Y_i^{\{E\}}, Y_i^{\{F\}}$ and solution update y_{n+1} represented by

$$Y_i^{\{I\}} = y_n + H\sum_{l=1}^{s^{\{I\}}} a_{i,l}^{\{I,I\}} f^{\{I\}}\left(Y_l^{\{I\}}\right) + H\sum_{l=1}^{s^{\{E\}}} a_{i,l}^{\{I,E\}} f^{\{E\}}\left(Y_l^{\{E\}}\right) + H\sum_{l=1}^{s^{\{F\}}} a_{i,l}^{\{I,F\}} f^{\{F\}}\left(Y_l^{\{F\}}\right), \tag{2.1a}$$

$$Y_j^{\{E\}} = y_n + H\sum_{l=1}^{s^{\{E\}}} a_{j,l}^{\{E,E\}} f^{\{E\}}\left(Y_l^{\{E\}}\right) + H\sum_{l=1}^{s^{\{I\}}} a_{j,l}^{\{E,I\}} f^{\{I\}}\left(Y_l^{\{I\}}\right) + H\sum_{l=1}^{s^{\{F\}}} a_{j,l}^{\{E,F\}} f^{\{F\}}\left(Y_l^{\{F\}}\right), \tag{2.1b}$$

$$Y_k^{\{F\}} = y_n + H\sum_{l=1}^{s^{\{F\}}} a_{k,l}^{\{F,F\}} f^{\{F\}}\left(Y_l^{\{F\}}\right) + H\sum_{l=1}^{s^{\{I\}}} a_{k,l}^{\{F,I\}} f^{\{I\}}\left(Y_l^{\{I\}}\right) + H\sum_{l=1}^{s^{\{E\}}} a_{k,l}^{\{F,E\}} f^{\{E\}}\left(Y_l^{\{E\}}\right), \tag{2.1c}$$

$$y_{n+1} = y_n + H\sum_{l=1}^{s^{\{I\}}} b_l^{\{I\}} f^{\{I\}}\left(Y_l^{\{I\}}\right) + H\sum_{l=1}^{s^{\{E\}}} b_l^{\{E\}} f^{\{E\}}\left(Y_l^{\{E\}}\right) + H\sum_{l=1}^{s^{\{F\}}} b_l^{\{F\}} f^{\{F\}}\left(Y_l^{\{F\}}\right), \tag{2.1d}$$

This form (2.1) does not assume the use of different time steps. Coefficients for a three component GARK method for (1.5) can be written in a Butcher tableau:

$$\begin{array}{ccc} \mathbf{A}^{\{I,I\}} & \mathbf{A}^{\{I,E\}} & \mathbf{A}^{\{I,F\}} \\ \mathbf{A}^{\{E,I\}} & \mathbf{A}^{\{E,E\}} & \mathbf{A}^{\{E,F\}} \\ \mathbf{A}^{\{F,I\}} & \mathbf{A}^{\{F,E\}} & \mathbf{A}^{\{F,F\}} \\ \hline \mathbf{b}^{\{I\}T} & \mathbf{b}^{\{E\}T} & \mathbf{b}^{\{F\}T} \end{array} \tag{2.2}$$

where $\left(\mathbf{A}^{\{q,q\}}, \mathbf{b}^{\{q\}}\right)$ represents an integration scheme applied to the component q and $\mathbf{A}^{\{q,p\}}, p \neq q$ are the coupling coefficients between integration schemes. Internal consistency

conditions within the GARK framework are

$$c_i^{\{q\}} = \sum_{j=1}^{s^{\{I\}}} a_{i,j}^{\{q,I\}} = \sum_{j=1}^{s^{\{E\}}} a_{i,j}^{\{q,E\}} = \sum_{j=1}^{s^{\{F\}}} a_{i,j}^{\{q,F\}}, \quad i = 1, \cdots, s^{\{q\}}, \quad q = I, E, F. \tag{2.3}$$

It is important to note that though we present GARK schemes in autonomous form, i.e. the right hand side depends only on the solution as in $f(y)$, we can easily transform to non-autonomous form $f(t, y)$ by using "$t = t_n + c_i H$ " that corresponds to $c_i^{\{q\}}$ in the internal consistency condition.

The order conditions for GARK methods are derived from ordinary Runge–Kutta order conditions and the use of N-tree theory from Araujo, Murua, and Sanz-Serna [3]. We reproduce the matrix-vector form of the GARK order conditions from [91]. The order conditions for GARK methods up to order 4 (assuming internal consistency) for σ, ν, λ varying over $\{I, E, F\}$ are as follows:

$$\mathbf{b}^{\{\sigma\}T} \mathbb{1}^{\{\sigma\}} = 1, \qquad \forall \sigma, \quad \text{(order 1)} \tag{2.4a}$$

$$\mathbf{b}^{\{\sigma\}T} \mathbf{c}^{\{\sigma\}} = \frac{1}{2}, \qquad \forall \sigma, \nu, \quad \text{(order 2)} \tag{2.4b}$$

$$\mathbf{b}^{\{\sigma\}T} \mathbf{c}^{\{\sigma\}\times 2} = \frac{1}{3}, \qquad \forall \sigma, \nu, \quad \text{(order 3)} \tag{2.4c}$$

$$\mathbf{b}^{\{\sigma\}T} \mathbf{A}^{\{\sigma,\nu\}} \mathbf{c}^{\{\nu\}} = \frac{1}{6}, \qquad \forall \sigma, \nu, \quad \text{(order 3)} \tag{2.4d}$$

$$\mathbf{b}^{\{\sigma\}T} \mathbf{c}^{\{\sigma\}\times 3} = \frac{1}{4}, \qquad \forall \sigma, \nu, \quad \text{(order 4)} \tag{2.4e}$$

$$\left(\mathbf{b}^{\{\sigma\}} \times \mathbf{c}^{\{\sigma\}}\right)^T \mathbf{A}^{\{\sigma,\nu\}} \mathbf{c}^{\{\nu\}} = \frac{1}{8}, \qquad \forall \sigma, \nu, \quad \text{(order 4)} \tag{2.4f}$$

$$\mathbf{b}^{\{\sigma\}T} \mathbf{A}^{\{\sigma,\nu\}} \mathbf{c}^{\{\nu\}\times 2} = \frac{1}{12}, \qquad \forall \sigma, \nu, \quad \text{(order 4)} \tag{2.4g}$$

$$\mathbf{b}^{\{\sigma\}T} \mathbf{A}^{\{\sigma,\nu\}} \mathbf{A}^{\{\nu,\lambda\}} \mathbf{c}^{\{\lambda\}} = \frac{1}{24}, \qquad \forall \sigma, \nu, \lambda, \quad \text{(order 4)} \tag{2.4h}$$

where $\mathbb{1}^{\{\sigma\}}$ is vector of ones in $\mathbb{R}^{s^{\{\sigma\}}\times 1}$, $a \times b$ denotes element-wise multiplication while $c^{\times k}$ denotes element-wise exponentiation.

Note that to create a second-order GARK method, the individual component tables $\mathbf{A}^{\{q,q\}}$ only need to satisfy second order Runge–Kutta conditions and there are no additional coupling conditions on $\mathbf{A}^{\{q,p\}}, p \neq q$. Standard Runge–Kutta methods and additive Runge–Kutta (ARK) methods can all be expressed as GARK methods with their complete set of order conditions represented in (2.4). Within the GARK framework, one step of an ARK method applied to (1.2) can be written as

$$Y_i = y_n + H \sum_{j=1}^{s} a_{i,j}^{\{S,S\}} f^{\{S\}}\left(Y_j\right) + H \sum_{j=1}^{s} a_{i,j}^{\{F,F\}} f^{\{F\}}\left(Y_j\right), \tag{2.5}$$

$$y_{n+1} = y_n + H \sum_{i=1}^{s} b^{\{S\}} f^{\{S\}}(Y_i) + H \sum_{i=1}^{s} b^{\{F\}} f^{\{F\}}(Y_i). \tag{2.6}$$

2.2. Multirate infinitesimal step methods

The idea behind multirate infinitesimal step methods (MIS) was originally featured in Knoth and Wolke's 1998 paper [61] and further developed by Knoth, Wolke and collaborators in [60, 98, 100, 101, 107]. Here for ease of comparison with other MIS-type methods, we use the notation for MIS methods used by Sandu [88]. MIS methods in their original form have an explicit slow base $s^{\{S\}}$-stage Runge–Kutta method with coefficients $(A^{\{S\}}, b^{\{S\}}, c^{\{S\}})$ and sorted $0 \leq c_1^{\{S\}} \leq c_2^{\{S\}} \leq \cdots \leq c_{s^{\{S\}}}^{\{S\}} \leq 1$. One step of an MIS method t_n to $t_n + H$ applied to the two-way additive problem (1.2) proceeds as follows:

$$\text{Let : } Y_1^{\{S\}} := y_n \tag{2.7a}$$

$$\text{For } i = 2, \ldots, s^{\{S\}} :$$

$$\begin{cases} \text{Let:} & v(0) := Y_{i-1}^{\{S\}}, \\ \text{Solve:} & v' = f^{\{F\}}(v) + \sum_{j=1}^{i-1} \frac{a_{i,j}^{\{S\}} - a_{i-1,j}^{\{S\}}}{c_i^{\{S\}} - c_{i-1}^{\{S\}}} f^{\{S\}}(Y_j^{\{S\}}), \text{ for } \tau \in [0, (c_i^{\{S\}} - c_{i-1}^{\{S\}})H], \\ \text{Let:} & Y_i^{\{S\}} := v((c_i^{\{S\}} - c_{i-1}^{s\{S\}})H), \end{cases} \tag{2.7b}$$

Compute step solution:

$$\begin{cases} \text{Let:} & v(0) = Y_{s\{S\}+1}^{\{S\}}, \\ \text{Solve:} & v' = f^{\{F\}}(v) + \sum_{j=1}^{s\{S\}} \frac{b_j^{\{S\}} - a_{s\{S\},j}^{\{S\}}}{1 - c_{s\{S\}}^{\{S\}}} f^{\{S\}}(Y_j^{\{S\}}), \quad \tau \in [0, (1 - c_{s\{S\}}^{\{S\}})H], \\ \text{Let:} & y_{n+1} := v((1 - c_{s\{S\}}^{\{S\}})H), \end{cases} \tag{2.7c}$$

Here $Y_i^{\{S\}}$ are the slow stages and y_{n+1} is the step solution. The MIS idea which carries on as a unifying theme in this work is the integration of a modified ODE (2.7b) with small time steps $h \ll H$ between each of the slow stages. This modified ODE is a combination of the fast process with a forcing term which is a linear combination of the slow processes evolved at previously computed slow stages. The coefficients in this linear combination are constant terms derived from the coefficients of the slow base Runge–Kutta method. These coefficients are sometimes referred to as "tendency terms" [107] or "slow tendency terms" [87]. In this format, the modified ODE can be solved with any method of qualifying order, typically the same order as the MIS method. The last modified ODE (2.7c) to get the step solution y_{n+1} is only necessary if the 'stiffly accurate' condition, $a_{s^{\{S\}},j}^{\{S\}} = b_j^{\{S\}}$ for all $j = 1, \ldots, s^{\{S\}}$, is not satisfied. For ease of analysis, methods such as multirate infinitesimal GARK methods which we discuss in the next section, pad the slow table with $b_j^{\{S\}}$'s in this case, allowing the entire fast modified ODE solution (2.7c) to be removed.

MIS methods satisfy a number of desirable properties. First, a second-order accurate MIS method only requires second-order accurate Runge–Kutta methods for the slow base and fast inner method; no additional conditions are imposed. Second, the work of Günther and Sandu

in [42] shows that MIS methods can be cast into the GARK framework, by assuming that (2.7b) is solved by a Runge–Kutta method $\left(A^{\{F\}}, b^{\{F\}}, c^{\{F\}}\right)$ with $s^{\{F\}}$ stages, and then determining the corresponding GARK coefficient matrices $\left(\mathbf{A}^{\{S,S\}}, \mathbf{b}^{\{S\}}\right)$, $\left(\mathbf{A}^{\{F,F\}}, \mathbf{b}^{\{F\}}\right)$, and coupling matrices $\left(\mathbf{A}^{\{F,S\}}, \mathbf{A}^{\{S,F\}}\right)$. Additionally, assuming we have third-order slow base and inner fast methods, Günther and Sandu in [42] identified a necessary and sufficient condition on the slow table $(A^{\{S\}}, b^{\{S\}}, c^{\{S\}})$ to guarantee that the MIS method is in fact third order accurate:

$$\sum_{i=2}^{s^{\{S\}}} \left(c_i - c_{i-1}\right)\left(e_i + e_{i-1}\right)^T A^{\{S\}} c^{\{S\}} + \left(1 - c^{\{S\}}_{s^{\{S\}}}\right)\left(\frac{1}{2} + e^T_{s^{\{S\}}} A^{\{S\}} c^{\{S\}}\right) = \frac{1}{3}. \tag{2.8}$$

It is therefore fairly easy to create MIS methods of up to third-order. Furthermore, MIS methods can use the same method for the slow base and the inner ODE solve, allowing for 'telescopic' multirate methods that nest multiple MIS methods inside one another to achieve an N-way multirate method. Finally, MIS method only require a single traversal of the time interval $[t_n, t_n + H]$ making them highly efficient.

Combining MIS methods and the GARK framework has proved to be a fruitful avenue for the creation of fourth-order multirate methods. Sexton and Reynolds [102] came up with their fourth-order relaxed MIS methods (RMIS) from using combinations of $f^{\{S\}}(Y_i^{\{S\}})$ and $f^{\{F\}}(Y_i^{\{S\}})$ in computing y_{n+1}. Sandu [88] came up with his multirate infinitesimal GARK (MRI-GARK) methods which introduce a time dependent forcing term to (2.7b) allowing for fourth-order implicit and explicit multirate methods. MRI-GARK methods form the backbone of the methods in chapter 3 and are introduced in the next section.

2.3. Multirate infinitesimal GARK methods

Multirate infinitesimal GARK methods were developed by Sandu in 2019 [88] for the two-way additive problem (1.2). They arise from leveraging MIS theory and GARK theory. Similar to MIS methods, MRI-GARK methods start with an internally consistent $\widehat{s^{\{S\}}}$ stage

slow Runge–Kutta base method $\left(\widehat{A^{\{S\}}}, \widehat{b^{\{S\}}}, \widehat{c^{\{S\}}}\right)$, but unlike MIS methods this base method can be explicit or diagonally implicit with explicit first stage (traditionally referred to as 'EDIRK'). The abscissae for MRI-GARK methods are also ordered to facilitate forward in time movement. For ease of notation we pad $\widehat{A^{\{S\}}}$ with the row $\widehat{b^{\{S\}T}}$ if the Runge–Kutta table does not meet the stiffly accurate condition. Once the original table is padded, we have an outer Runge–Kutta method with $s^{\{S\}}$ stages (equaling either $\widehat{s^{\{S\}}}$ or $\widehat{s^{\{S\}}}+1$) and coefficients $A^{\{S\}} \in \mathbb{R}^{s^{\{S\}} \times s^{\{S\}}}$ and $b^{\{S\}} \in \mathbb{R}^{s^{\{S\}} \times 1}$ with $c^{\{S\}}_{s^{\{S\}}} = 1$. We can define the increments between the abscissae by:

$$\Delta c_i^{\{S\}} := \begin{cases} 0, & i = 1, \\ c_i^{\{S\}} - c_{i-1}^{\{S\}} \geq 0, & i = 2, \ldots, s^{\{S\}}. \end{cases} \tag{2.9}$$

Unlike in Sandu's presentation [88], where $\Delta c_1^{\{S\}}$ is not necessarily equal to zero, we define it as zero here for an easily identifiable structure for implicit MRI-GARK methods. One step of an MRI-GARK method from t_n to $t_n + H$ when integrating (1.2) proceeds as follows:

$$\text{Let : } Y_1^{\{S\}} := y_n \tag{2.10a}$$

For $i = 2, \ldots, s^{\{S\}}$:

$$\begin{cases} \text{Let:} & v(0) := Y_{i-1}^{\{S\}} \text{ and } T_i = t_n + c_i^{\{S\}} H, \\ \text{Solve:} & v'(\theta) = \Delta c_i^{\{S\}} f^{\{F\}}\left(T_i + \Delta c_i^{\{S\}}\theta, v\right) + \sum_{j=1}^{i} \gamma_{i,j}(\frac{\theta}{H}) f^{\{S\}}(T_j, Y_j^{\{S\}}), \text{ for } \theta \in [0, H], \\ \text{Let:} & Y_i^{\{S\}} := v(H), \end{cases} \tag{2.10b}$$

$$y_{n+1} = Y_{s^{\{S\}}}^{\{S\}}. \tag{2.10c}$$

The slow tendency coefficients which dictate the coupling from the slow to the fast time scale are time dependent polynomials $\gamma_{i,j}(\tau)$ defined as

$$\gamma_{i,j}(\tau) \coloneqq \sum_{k\geq 0} \gamma_{i,j}^k \tau^k, \tag{2.11}$$

where typically $0 \leq k \leq 2$. Additionally, we define coefficients

$$\bar{\gamma}_{i,j} \coloneqq \sum_{k\geq 0} \frac{\gamma_{i,j}^k}{k+1}, \tag{2.12}$$

that are directly related to the slow base method $A^{\{S\}}$. The coefficients $\gamma_{i,j}(\tau)$ and $\bar{\gamma}_{i,j}$ can be populated into matrices of the form

$$\Gamma(\tau) = \sum_{k\geq 0} \Gamma^k \tau^k, \quad \overline{\Gamma} = \sum_{k\geq 0} \frac{1}{k+1} \Gamma^k.$$

We note that for our definition of MRI-GARK methods here, $\Gamma(\tau)$ is strictly lower triangular for explicit MRI-GARK methods and lower triangular for implicit MRI-GARK methods.

The major difference between MIS and MRI-GARK methods are the slow tendency terms. As we have seen in the last section, the coefficients of the linear combination of slow function evaluations used in (2.7b) are constants. The extension to time dependent polynomials provides more degrees of freedom to achieve higher-order methods. An MRI-GARK is defined when the coefficient matrices Γ^k and the abscissae $c^{\{S\}}$ are specified.

2.3.1. Order conditions

Similar to MIS methods, we can derive the MRI-GARK order conditions by applying to the modified ODEs (2.10b), a single step of an $s^{\{F\}}$-stage Runge–Kutta method $\left(A^{\{F\}}, b^{\{F\}}, c^{\{F\}}\right)$ that has at least the same order of accuracy as the MRI-GARK method. We can then identify the GARK slow table and slow-fast coupling coefficient matrices as

follows

$$\mathbf{A}^{\{S,S\}} := E\overline{\Gamma} = A^{\{S\}} \tag{2.13a}$$

$$\mathbf{b}^{\{S\}} := \mathbb{1}^{\{S\}T}\overline{\Gamma} = b^{\{S\}}, \tag{2.13b}$$

$$\mathbf{c}^{\{S\}} := E\overline{\Gamma}\mathbb{1}^{\{S\}} = A^{\{S\}}\mathbb{1}^{\{S\}} = c^{\{S\}}, \tag{2.13c}$$

$$\mathbf{A}^{\{S,F\}} := \left[\mathbf{A}^{\{S,F,1\}}, \quad \cdots, \quad \mathbf{A}^{\{S,F,s^{\{S\}}\}}\right] = \Delta C^{\{S\}} \otimes b^{\{F\}T} \in \mathbb{R}^{s^{\{S\}} \times s}. \tag{2.13d}$$

Here $s = s^{\{F\}}s^{\{S\}}$ is the total number of stages of the method, $\otimes$ is the Kronecker product, $\mathbb{1}^{\{S\}} \in \mathbb{R}^{s^{\{S\}}}$ is a column vector of all ones,

$$\mathbf{E} \in \mathbb{R}^{s^{\{S\}} \times s^{\{S\}}}, \quad \mathbf{E}_{i,j} = \begin{cases} 1, & i \geq j, \\ 0, & \text{otherwise}, \end{cases}, \quad \Delta C^{\{S\}} := \begin{bmatrix} \Delta c_1^{\{S\}} & 0^{\{F\}T} & \cdots & 0^{\{F\}T} \\ \Delta c_1^{\{S\}} & \Delta c_2^{\{S\}} & \cdots & 0^{\{F\}T} \\ \vdots & \vdots & \ddots & 0^{\{F\}T} \\ \Delta c_1^{\{S\}} & \Delta c_2^{\{S\}} & \cdots & \Delta c_{s^{\{S\}}}^{\{S\}} \end{bmatrix},$$

and $0^{\{F\}}$ is a column vector of all zeros in $\mathbb{R}^{s^{\{F\}}}$. We note that (2.13a) gives a consistency condition between the slow Runge–Kutta method and the MRI-GARK coefficients $\overline{\Gamma}$.

The GARK fast table and fast-slow coupling coefficients are as follows:

$$\mathbf{A}^{\{F,F\}} := \begin{bmatrix} \Delta c_1^{\{S\}} A^{\{F\}} & 0_{s^{\{F\}}\times s^{\{F\}}} & \cdots & 0_{s^{\{F\}}\times s^{\{F\}}} \\ \Delta c_1^{\{S\}} \mathbb{1}^{\{F\}} b^{\{F\}T} & \Delta c_2^{\{S\}} A^{\{F\}} & \cdots & \vdots \\ & \Delta c_2^{\{S\}} \mathbb{1}^{\{F\}} b^{\{F\}T} & \cdots & \\ \vdots & \vdots & \ddots & \vdots \\ \Delta c_1^{\{S\}} \mathbb{1}^{\{F\}} b^{\{F\}T} & \Delta c_2^{\{S\}} \mathbb{1}^{\{F\}} b^{\{F\}T} & \cdots & \Delta c_{s^{\{S\}}}^{\{S\}} A^{\{F\}} \end{bmatrix} \tag{2.14a}$$

$$= \operatorname{diag}\big(\Delta c^{\{S\}}\big) \otimes A^{\{F\}} + L\Delta C^{\{S\}} \otimes \mathbb{1}^{\{F\}} b^{\{F\}T} \in \mathbb{R}^{s\times s},$$

$$\mathbf{c}^{\{F\}} := \begin{bmatrix} \Delta c_1^{\{S\}} c^{\{F\}} \\ c_1^{\{S\}} \mathbb{1}^{\{F\}} + \Delta c_2^{\{S\}} c^{\{F\}} \\ \vdots \\ c_{s^{\{S\}}-1}^{\{S\}} \mathbb{1}^{\{F\}} + \Delta c_{s^{\{S\}}}^{\{S\}} c^{\{F\}} \end{bmatrix} = Lc^{\{S\}} \otimes \mathbb{1}^{\{F\}} + \Delta c^{\{S\}} \otimes c^{\{F\}} \in \mathbb{R}^s, \tag{2.14b}$$

$$\mathbf{b}^{\{F\}} := \begin{bmatrix} \Delta c_1^{\{S\}} b^{\{F\}} \\ \vdots \\ \Delta c_{s^{\{S\}}}^{\{S\}} b^{\{F\}} \end{bmatrix} = \Delta c^{\{S\}} \otimes b^{\{F\}} \in \mathbb{R}^s, \tag{2.14c}$$

$$\mathbf{A}^{\{F,S\}} := \begin{bmatrix} 0_{s^{\{F\}}\times s^{\{S\}}} \\ \mathbb{1}^{\{F\}}\big(e_1^T A^{\{S\}}\big) + \sum_{k\geq 0}\big(A^{\{F\}}c^{\{F\}\times k}\big)\big(e_2^T\Gamma^{\{k\}}\big) \\ \vdots \\ \mathbb{1}^{\{F\}}\big(e_{s^{\{S\}}-1}^T A^{\{S\}}\big) + \sum_{k\geq 0}\big(A^{\{F\}}c^{\{F\}\times k}\big)\big(e_{s^{\{S\}}}^T\Gamma^{\{k\}}\big) \end{bmatrix} \tag{2.14d}$$

$$= LA^{\{S\}} \otimes \mathbb{1}^{\{F\}} + \sum_{k\geq 0}\Gamma^{\{k\}} \otimes \big(A^{\{F\}}c^{\{F\}\times k}\big) \in \mathbb{R}^{s\times s^{\{S\}}},$$

where $\mathrm{diag}\big(\Delta c^{\{S\}}\big)$ is the diagonal matrix obtained by taking $\Delta c^{\{S\}}$ as its diagonal entries, and $L \in \mathbb{R}^{s^{\{S\}}\times s^{\{S\}}}$ has entries $L_{i,j} := \delta_{i,j+1}$. As expressed by Sandu in [88], once the GARK table coefficients have been determined, internal consistency conditions for MRI-GARK methods, $\mathbf{c}^{\{S\}} = \mathbf{c}^{\{S\}}$ and $\mathbf{c}^{\{F\}} = \mathbf{c}^{\{F,S\}}$ hold if and only if

$$\Gamma^0\mathbb{1}^{\{S\}} = \Delta c^{\{S\}} \quad \text{and} \quad \Gamma^k\mathbb{1}^{\{S\}} = 0 \quad \forall k \geq 1. \tag{2.15}$$

It then follows from GARK order conditions that if both the fast and slow methods have order at least two and satisfy internal consistency conditions, the resulting MRI-GARK method is second order. Furthermore, the fast and slow conditions for an MRI-GARK method of order q are satisfied by selecting slow and fast base Runge–Kutta methods that are of order q. Maintaining our notation in defining MRI-GARK methods, we list the relevant theorems from [88] on third and fourth order MRI-GARK methods.

Theorem 2.3.1. *Third-order coupling condition.*
An internally consistent MRI-GARK method has order three if and only if the slow base method has order at least three and the following third order coupling condition holds:

$$\Delta c^{\{S\}T}\mathcal{A}^{\{I,\zeta\}}c^{\{S\}} = \frac{1}{6}, \tag{2.16}$$

where

$$\mathcal{A}^{\{S,\zeta\}} = LA^{\{S\}} + \sum_{k\geq 0} \zeta_k \Gamma^{\{k\}} \quad \textit{and} \quad \zeta_k = b^{\{F\}T} A^{\{F\}} c^{\{F\}\times k}. \tag{2.17}$$

Theorem 2.3.2. *Fourth-order coupling conditions.*

An internally consistent MRI-GARK method has order four if and only if the slow base method has order at least four, satisfies all the third order conditions and the following coupling conditions hold:

$$\left(\Delta c^{\{S\}} \times L c^{\{S\}}\right)^T \mathcal{A}^{\{S,\zeta\}} c^{\{S\}} + \left(\Delta c^{\{S\}\times 2}\right)^T \mathcal{A}^{\{S,\beta\}} c^{\{S\}} = \frac{1}{8}, \tag{2.18a}$$

$$\Delta c^{\{S\}T} \mathcal{A}^{\{S,\zeta\}} c^{\{S\}\times 2} = \frac{1}{12}, \tag{2.18b}$$

$$\left(\Delta c^{\{S\}} \times (D b^{\{S\}})\right)^T \mathcal{A}^{\{\nu,\zeta\}} c^{\{S\}} = \frac{1}{24}, \tag{2.18c}$$

$$\left(\Delta c^{\{S\}\times 2}\right)^T \mathcal{A}^{\{S,\xi\}} c^{\{S\}} + \Delta c^{\{S\}T} L \Delta C^{\{S\}} \mathcal{A}^{\{S,\zeta\}} c^{\{S\}} = \frac{1}{24}, \quad \textit{and} \tag{2.18d}$$

$$\Delta c^{\{S\}T} \mathcal{A}^{\{S,\zeta\}} A^{\{S\}} c^{\{S\}} = \frac{1}{24}, \tag{2.18e}$$

where

$$\mathcal{A}^{\{S,\beta\}} := \frac{1}{2} LA^{\{S\}} + \sum_{k\geq 0} \beta_k \Gamma^{\{k\}}, \qquad \mathcal{A}^{\{S,\xi\}} := \frac{1}{2} LA^{\{S\}} + \sum_{k\geq 0} \xi_k \Gamma^{\{k\}}, \tag{2.19}$$

$$L_{i,j} = \delta_{i,j+1}, \qquad D_{i,j} = \begin{cases} 1, & j \geq i, \\ 0, & \textit{otherwise}, \end{cases}$$

$$\beta_k := (b^{\{F\}} \times c^{\{F\}})^T A^{\{F\}} c^{\{F\}\times k}, \ \textit{and} \qquad \xi_k := b^{\{F\}T} A^{\{F\}} A^{\{F\}} c^{\{F\}\times k}. \tag{2.20}$$

2.3.2. Linear stability

Linear stability for regular Runge–Kutta methods is concerned with the behavior of the numerical solution as $t \to \infty$ for a fixed step size H. To investigate linear stability the test problem

$$y'(t) = \lambda y(t), \quad y(t_0) = y_0, \quad \lambda \in \mathbb{C}, \tag{2.21}$$

with exact solution $y(t) = y_0 e^{\lambda t}$ is considered. Since the stability of (2.21) is guaranteed for $\mathrm{Re}(\lambda) < 0$, the idea is to check if the numerical method is also stable for $\mathrm{Re}(\lambda) < 0$. Applying a Runge–Kutta method to (2.21) leads to an iteration $y_{n+1} = R(H\lambda)y_n$, for some stability function $R(H\lambda)$. The stability region associated with the Runge–Kutta method is therefore the set of all $z = H\lambda \in \mathbb{C}$ such that $|R(z)| \leq 1$.

Similar notions of linear stability can be formulated for multirate methods, however, there is currently no standardized approach for assessing linear stability. To assess the linear stability of MRI-GARK methods, Sandu considers scalar stability analysis and matrix stability analysis [88]. The scalar stability analysis closely resembles the stability analysis we describe above for Runge–Kutta methods and considers the scalar test problem

$$y' = \lambda^{\{F\}} y + \lambda^{\{S\}} y, \quad y(t_0) = y_0, \quad \lambda^{\{F\}}, \lambda^{\{S\}} \in \mathbb{C}^-. \tag{2.22}$$

Here we can define $z^{\{F\}} := H\lambda^{\{F\}}$ and $z^{\{S\}} := H\lambda^{\{S\}}$, where H is the slow time step. An application of the MRI-GARK method to (2.22), including an analytical solve of the modified ODE (2.10b), leads to the relation $y_{n+1} = R\left(z^{\{F\}}, z^{\{S\}}\right) y_n$. Sandu then defines the scalar slow stability region as

$$S^{1D}_{\rho,\alpha} = \left\{ z^{\{S\}} \in \mathbb{C} \middle| \left| R\left(z^{\{F\}}, z^{\{S\}}\right) \right| \leq 1, \quad \forall z^{\{F\}} \in \mathbb{C}^- : |z^{\{F\}}| \leq \rho, \quad |\arg(z^{\{F\}} - \pi| \leq \alpha \right\}. \tag{2.23}$$

Stability regions of form (2.23) show the slow stability region only given the fast $z^{\{F\}}$ is in some wedge of α-degrees that stretches out to ρ in the left complex-plane. For MRI-GARK

methods, such stability regions are usually smaller than the stability region of the slow base method and tend to shrink in size as the wedge becomes wider.

Scalar stability analysis has the advantage of being easier to implement and assess given its parallels to stability analysis of regular Runge–Kutta methods. However, it still relies on the assumption that the Jacobians of the fast and slow processes are simultaneously diagonalizable (a property that guarantees that the choice of basis for a system of linear ODEs does not affect the numerical method's stability), but does not necessarily hold for multirate schemes applied to additive systems [36].

Matrix stability analysis relies on the ideas first presented by Gear [35], refined by Kværnø [63], and later applied to multirate methods in [97, 23, 95, 55, 88, 81] and is applied to a model problem in component partitioned form

$$\begin{bmatrix} y^{\{F\}} \\ y^{\{S\}} \end{bmatrix}' = \begin{bmatrix} \lambda^{\{F\}} & \eta^{\{S\}} \\ \eta^{\{F\}} & \lambda^{\{S\}} \end{bmatrix} \begin{bmatrix} y^{\{F\}} \\ y^{\{S\}} \end{bmatrix} = \underbrace{\begin{bmatrix} \lambda^{\{F\}} & \frac{1-\xi}{\alpha}\left(\lambda^{\{F\}} - \lambda^{\{S\}}\right) \\ -\alpha\xi\left(\lambda^{\{F\}} - \lambda^{\{S\}}\right) & \lambda^{\{S\}} \end{bmatrix}}_{\mathbf{\Omega}} \begin{bmatrix} y^{\{F\}} \\ y^{\{S\}} \end{bmatrix}. \tag{2.24}$$

By changing variables to form $\mathbf{\Omega}$, the eigenvalues of $\mathbf{\Omega}$ can be expressed as linear combinations of $\lambda^{\{F\}}$ and $\lambda^{\{S\}}$ i.e. the two eigenvalues of $\mathbf{\Omega}$ are $\xi\lambda^{\{F\}}+(1-\xi)\,\lambda^{\{S\}}$ and $(1-\xi)\,\lambda^{\{F\}}+\xi\lambda^{\{S\}}$. The benefits of studying this test problem are in the parameter ξ which controls the strength of the coupling between the fast and slow variables. In particular, for $|\xi| \ll 1$, the fast weakly impacts the slow and for $|1-\xi| \ll 1$, the slow weakly impacts the fast. Because the system (2.24) is complex in general, it can often lead to complicated analysis. In analyzing MRI-GARK methods, Sandu makes the assumptions that $\alpha = 1$ and $\xi \in [0, 1]$. After defining $z^{\{F\}} = H\lambda^{\{F\}}$, $z^{\{S\}} = H\lambda^{\{S\}}$, $\omega^{\{F\}} = H\eta^{\{F\}}$, and $\omega^{\{S\}} = H\eta^{\{S\}}$, an application of the

MRI-GARK method to (2.24) leads to the recurrence relation

$$\begin{bmatrix} y_{n+1}^{\{F\}} \\ \\ y_{n+1}^{\{S\}} \end{bmatrix} = \mathbf{M}\left(z^{\{F\}}, z^{\{S\}}, \omega^{\{F\}}, \omega^{\{S\}}\right) \begin{bmatrix} y_{n+1}^{\{F\}} \\ \\ y_{n+1}^{\{S\}} \end{bmatrix} \tag{2.25}$$

for some error propagation matrix $\mathbf{M}$. The matrix stability region is therefore defined as

$$\begin{aligned} S_{\rho,\beta}^{2D} = \{z^{\{S\}} \in \mathbb{C} \quad | \quad & \max \mid \text{eig } \mathbf{M}\left(z^{\{F\}}, z^{\{S\}}\right) \mid < 1, \\ & \forall z^{\{F\}} \in \mathbb{C}^{-} : |z^{\{F\}}| \leq \rho, \quad |\arg(z^{\{F\}}) - \pi| \leq \beta\}. \end{aligned} \tag{2.26}$$

Studying the matrix stability region is certainly useful in determining the impact the strength of the coupling between fast and slow has on stability. However, for ease of analysis, we have only considered the scalar stability analysis that is quite similar to Sandu's for our methods in Chapter 3. Furthermore, our numerical results have shown scalar stability analysis to have some predictive power on the behavior of our methods on non-trivial test problems.

2.3.3. Construction of MRI-GARK methods

In order to generate an MRI-GARK method, the first step is to pick a slow base method that satisfies the required conditions on the abscissae and then figure out which Γ^k matrices satisfy the GARK order conditions. In [88], Sandu derives several methods up to fourth-order with some methods allowing implicitness at the slow time scale (decoupled implicit MRI-GARK). Decoupled/solve-decoupled implicit MRI-GARK methods have implicitness in either the fast or slow processes but do not involve nonlinear solves that couple both fast and slow processes. Practically, this means (2.10b) is either a fast evolution solve or a regular DIRK stage i.e. a solve-decoupled MRI-GARK scheme computes two consecutive stages of

(2.10) as follows:

For $\Delta c_i^{\{S\}} > 0$, solve modified ODE:

$$\begin{cases} \text{Let:} & v(0) := Y_{i-1}^{\{S\}} \text{ and } T_i = t_n + c_i^{\{S\}} H, \\ \text{Solve:} & v'(\theta) = \Delta c_i^{\{S\}} f^{\{F\}} \left(T_i + \Delta c_i^{\{S\}} \theta, v \right) + \sum_{j=1}^{i} \gamma_{i,j}(\frac{\theta}{H}) f^{\{S\}}(T_j, Y_j^{\{S\}}), \text{ for } \theta \in [0, H], \\ \text{Let:} & Y_i^{\{S\}} := v(H), \end{cases} \tag{2.27a}$$

For $\Delta c_i^{\{S\}} = 0$, solve DIRK stage:

$$Y_{i+1}^{\{S\}} = Y_i^{\{S\}} + H \sum_{j=1}^{i} \overline{\gamma}_{i,j} f^{\{S\}} \left(Y_j^{\{S\}} \right). \tag{2.27b}$$

Solve-decoupled implicit MRI-GARK methods are easier to implement than coupled methods but have limited stability [88, 81].

Chapter 3

Implicit-explicit multirate infinitesimal GARK methods

The contents of this chapter have been submitted for publication under the title "Implicit-explicit multirate infinitesimal GARK methods" in collaboration with Daniel R. Reynolds [15]

3.1. Introduction

In recent years, there has been a renewed interest in time integration methods, most notably those that allow both high accuracy and increased flexibility with regard to how various components of the problem are treated. These methods range from those that apply a uniform time step size for all components of a problem but vary the algorithms used on individual terms, to 'multirate' methods that evolve separate solution components using different step sizes.

Methods in the former category have been introduced primarily to handle problems that couple stiff and nonstiff processes. Here, instead of applying a fully implicit or fully explicit treatment, that would be ideally suited to only the stiff or nonstiff components of the problem, respectively, these approaches allow more robust implicit solvers to be applied to the stiff components, leaving the remaining nonstiff (and frequently nonlinear) components to be treated explicitly. Various techniques within this category include mixed implicit-explicit (IMEX) additive Runge–Kutta methods [4, 24, 25, 57, 58, 92], exponential Runge–Kutta (ExpRK) and exponential Rosenbrock (ExpRB) methods [50, 70, 71, 78, 105, 104] and general linear methods (GLM) [12, 11, 85, 110, 109].

Multirate methods, on the other hand, evolve separate solution components or dynamical processes using entirely different time step sizes. These frequently arise due to 'multiphysics'

problems wherein separate physical processes evolve on disparate time scales. Either due to stability or accuracy considerations the 'fast' processes must be evolved with small step sizes, but due to their computational cost the 'slow' processes are evolved using sometimes much larger time steps. While simplistic low-order 'subcycling' approaches have been employed in computational simulations for decades, research into higher-order approaches has seen dramatic recent advances [6, 23, 36, 42, 68, 84, 81, 88, 98, 101, 102, 107].

In this paper we introduce a hybrid of two of the above techniques: IMEX Runge–Kutta and multirate methods. While the large majority of recent research on multirate methods has focused on the two-way, additive initial-value problem (IVP) combining a fast $\{F\}$ and a slow $\{S\}$ process,

$$y' = f(t,y) = f^{\{F\}}(t,y) + f^{\{S\}}(t,y), \quad y(t_0) = y_0,\ t \in [t_0, t_f], \tag{3.1}$$

we focus on problems that further break down the slow portion into stiff $\{I\}$ and nonstiff $\{E\}$ components. Thus we consider the three-way additive IVP:

$$y' = f^{\{I\}}(t,y) + f^{\{E\}}(t,y) + f^{\{F\}}(t,y), \quad y(t_0) = y_0,\ t \in [t_0, t_f]. \tag{3.2}$$

Of the various approaches for multirate integration, we focus on those that are agnostic as to the precise methods applied to the fast dynamics. These are based on 'infinitesimal' formulations, including the seminal work on multirate infinitesimal step (MIS) methods [98, 107] and their more recent extensions to higher temporal order [6, 68, 84, 88, 102]. In such formulations, the fast dynamics are assumed to be solved 'exactly', typically through evolution of a sequence of modified fast IVPs,

$$v'(\theta) = f^{\{F\}}(\theta, v) + g(\theta), \quad v(\theta_0) = v_0,\ \theta \in [\theta_0, \theta_f],$$

where the forcing function $g(\theta)$ is determined by the multirate method to incorporate information from $f^{\{S\}}$. In practice, these fast IVPs are solved using another numerical method with smaller step size, which in turn could employ further decompositions via an IMEX, ExpRK, ExpRB, GLM, or multirate approach.

To our knowledge, there exist only two multirate schemes that simultaneously allow IMEX treatment of the slow dynamics and infinitesimal treatment of the fast dynamics, both of which have low accuracy and have been shown to demonstrate poor stability [29, 86]. The first of these is the standard first-order "Lie-Trotter" splitting that performs the time step $y_n \to y_{n+1}$ (here $y_n \approx y(t_n)$ and $t_{n+1} - t_n = H$) [76] via the algorithm:

$$
\begin{aligned}
y_{n+1}^{(1)} &= y_n + H f^{\{E\}}(t_n, y_n), \\
y_{n+1}^{(2)} &= y_{n+1}^{(1)} + H f^{\{I\}}(t_{n+1}, y_{n+1}^{(2)}), \\
&\text{Solve} \begin{cases} v(0) = y_{n+1}^{(2)}, \\ v'(\theta) = f^{\{F\}}(t_n + \theta, v), \text{ for } \theta \in [0, H], \end{cases} \\
y_{n+1} &= v(H).
\end{aligned}
\tag{3.3}
$$

The second is a variant of the second-order "Strang" (or "Strang-Marchuk") splitting formulation [75, 103] ,

$$
\begin{aligned}
y_{n+1}^{(1)} &= y_n + \tfrac{H}{4} f^{\{E\}}\left(t_n, y_n\right) \\
&\quad + \tfrac{H}{4} f^{\{E\}}\left(t_n + \tfrac{H}{2}, y_n + \tfrac{H}{2} f^{\{E\}}\left(t_n, y_n\right)\right), \\
y_{n+1}^{(2)} &= y_{n+1}^{(1)} + \tfrac{H}{4} f^{\{I\}}\left(t_n, y_{n+1}^{(1)}\right) + \tfrac{H}{4} f^{\{I\}}\left(t_n + \tfrac{H}{2}, y_{n+1}^{(2)}\right),
\end{aligned}
\tag{3.4}
$$

$$
\begin{aligned}
&\text{Solve} \begin{cases} v(0) = y_{n+1}^{(2)}, \\ v'(\theta) = f^{\{F\}}\left(t_n + \theta, v\right), \text{ for } \theta \in [0, H], \end{cases} \\
&y_{n+1}^{(3)} = v(H), \\
&y_{n+1}^{(4)} = y_{n+1}^{(3)} + \tfrac{H}{4} f^{\{I\}}\left(t_n + \tfrac{H}{2}, y_{n+1}^{(3)}\right) + \tfrac{H}{4} f^{\{I\}}\left(t_{n+1}, y_{n+1}^{(4)}\right), \\
&y_{n+1} = y_{n+1}^{(4)} + \tfrac{H}{4} f^{\{E\}}\left(t_n + \tfrac{H}{2}, y_{n+1}^{(4)}\right) \\
&\qquad + \tfrac{H}{4} f^{\{E\}}\left(t_{n+1}, y_{n+1}^{(4)} + \tfrac{H}{2} f^{\{E\}}\left(t_n + \tfrac{H}{2}, y_{n+1}^{(4)}\right)\right).
\end{aligned}
$$

We note that here, the updates $y_n \to y_{n+1}^{(1)}$ and $y_{n+1}^{(4)} \to y_{n+1}$ correspond to using the explicit Heun method for a half time-step each, while the updates $y_{n+1}^{(1)} \to y_{n+1}^{(2)}$ and $y_{n+1}^{(3)} \to y_{n+1}^{(4)}$ correspond to using the implicit trapezoid rule for a half time-step each. However to our knowledge, there do not exist multirate methods allowing IMEX treatment of the slow time scale that have order of accuracy three or higher. The purpose of this paper is to address this need, through proposal of a new class of *implicit-explicit multirate infinitesimal generalized-structure additive Runge–Kutta* (IMEX-MRI-GARK) methods for problems of the form (3.2), including derivation of order conditions up to fourth-order, and numerical tests to demonstrate the benefit of such methods over the legacy approaches (3.3) and (3.4), as well as to provide comparisons against recent third and fourth-order implicit MRI-GARK methods.

3.2. Implicit-explicit multirate infinitesimal GARK methods

We base our proposed methods off of the MRI-GARK class of two-component multirate methods [88]. Just as those methods begin with an explicit or diagonally-implicit Runge–Kutta method for the slow time scale, we start with an IMEX additive Runge-Kutta scheme

(IMEX-ARK) of order q and having $\tilde{s}^{\{S\}}$ stages. These methods are characterized by a pair of Butcher tables:

$$\begin{array}{c|c} c^{\{E\}} & A^{\{E\}} \\ \hline 1 & b^{\{E\}T} \end{array} \qquad \begin{array}{c|c} c^{\{I\}} & A^{\{I\}} \\ \hline 1 & b^{\{I\}T} \end{array}$$

Before constructing IMEX-MRI-GARK methods, we place three additional restrictions on the base IMEX-ARK method: (a) the tables are "internally consistent," (i.e., $c^{\{E\}} = c^{\{I\}} := c^{\{S\}}$), (b) the tables have explicit slow first stage (i.e., $c_1^{\{S\}} = 0$), and (c) the tables have non-decreasing abscissae (i.e., $c_1^{\{S\}} \leq c_2^{\{S\}} \leq \cdots \leq c_{\tilde{s}^{\{S\}}}^{\{S\}}$). To reduce complexity in our analyses we follow [84, 88] and write the base IMEX-ARK method in stiffly accurate form, i.e., the last row of $A^{\{E\}}$ and $A^{\{I\}}$ equal $b^{\{E\}T}$ and $b^{\{I\}T}$, respectively. We note that for methods that do not satisfy this requirement in simplest form, they may easily be converted to the stiffly accurate form by padding c and A with 1 and b^T, respectively:

$$\begin{array}{c|c|c} c^{\{S\}} & A^{\{E\}} & A^{\{I\}} \\ \hline 1 & b^{\{E\}T} & b^{\{I\}T} \end{array} \rightarrow \begin{array}{c|cc|cc} c^{\{S\}} & A^{\{E\}} & 0^{\{S\}} & A^{\{I\}} & 0^{\{S\}} \\ 1 & b^{\{E\}T} & 0 & b^{\{I\}T} & 0 \\ \hline 1 & b^{\{E\}T} & 0 & b^{\{I\}T} & 0 \end{array}$$

where $0^{\{S\}} \in \mathbb{R}^{\tilde{s}^{\{S\}}}$. Thus for the remainder of this paper, we let $A^{\{E,E\}}, A^{\{I,I\}} \in \mathbb{R}^{s^{\{S\}} \times s^{\{S\}}}$ be the stiffly-accurate versions of the IMEX-ARK Butcher tables $A^{\{E\}}$ and $A^{\{I\}}$, respectively. We note that this extension of the tables to include the row of b coefficients does not affect the order conditions of the original IMEX-ARK table, and thus all order conditions satisfied by the original IMEX-ARK tables remain unchanged. Based on the above assumptions on the abscissae, the increments between consecutive stages are given by

$$\Delta c_i^{\{S\}} := \begin{cases} 0, & i = 1, \\ c_i^{\{S\}} - c_{i-1}^{\{S\}} \geq 0, & i = 2, \ldots, s^{\{S\}}. \end{cases} \tag{3.5}$$

Definition 3.2.1 (IMEX-MRI-GARK methods for additive systems)**.** *The following algorithm defines one step from t_n to $t_{n+1} = t_n + H$ of an IMEX-MRI-GARK scheme for the problem* (3.2)*:*

$$\text{Let: } Y_1^{\{S\}} := y_n \tag{3.6a}$$

$$\text{For } i = 2, \ldots, s^{\{S\}} : \tag{3.6b}$$

$$\begin{cases} \text{Let:} & v(0) := Y_{i-1}^{\{S\}} \quad \text{and} \quad T_{i-1} := t_n + c_{i-1}^{\{S\}} H, \\ \text{Solve:} & v'(\theta) = \Delta c_i^{\{S\}} f^{\{F\}}\left(T_{i-1} + \Delta c_i^{\{S\}}\theta,\, v(\theta)\right) \\ & \qquad + \sum_{j=1}^{i} \gamma_{i,j}\left(\frac{\theta}{H}\right)\, f_j^{\{I\}} + \sum_{j=1}^{i-1} \omega_{i,j}\left(\frac{\theta}{H}\right)\, f_j^{\{E\}}, \ \text{for } \theta \in [0, H], \\ \text{Let:} & Y_i^{\{S\}} := v(H), \end{cases} \tag{3.6c}$$

$$y_{n+1} = Y_{s^{\{S\}}}^{\{S\}}. \tag{3.6d}$$

where $f_j^{\{I\}} := f^{\{I\}}\left(t_n + c_j^{\{S\}} H, Y_j^{\{S\}}\right)$ and $f_j^{\{E\}} := f^{\{E\}}\left(t_n + c_j^{\{S\}} H, Y_j^{\{S\}}\right)$.

Definition 3.2.2 (Slow tendency coefficients)**.** *The functions $\gamma_{i,j}$ and $\omega_{i,j}$ from equation* (3.6c) *are polynomials in time that dictate the couplings from the slow to the fast time scale. As in [88], $\gamma_{i,j}$ are defined using coefficients $\left\{\gamma_{i,j}^{\{k\}}\right\}$ as:*

$$\gamma_{i,j}(\tau) := \sum_{k \geq 0} \gamma_{i,j}^{\{k\}} \tau^k, \tag{3.7}$$

and the polynomials $\omega_{i,j}(\tau)$ are defined similarly. Thus the coefficients $\left\{\gamma_{i,j}^{\{k\}}\right\}$, $\left\{\omega_{i,j}^{\{k\}}\right\}$ and $c^{\{S\}}$ uniquely define an IMEX-MRI-GARK method. We note that these sums over $k \geq 0$ are not infinite, and only involve as many terms as there exist nonzero coefficients (typically $0 \leq k \leq 2$).

We group these coefficients into matrices $\Gamma^{\{k\}}, \Omega^{\{k\}}, \overline{\Gamma}$ *and* $\overline{\Omega}$*, that respectively contain* $\left\{\gamma_{i,j}^{\{k\}}\right\}$, $\left\{\omega_{i,j}^{\{k\}}\right\}$, $\{\overline{\gamma}_{i,j}\}$ *and* $\{\overline{\omega}_{i,j}\}$*, where*

$$\overline{\gamma}_{i,j} \coloneqq \sum_{k\geq 0} \gamma_{i,j}^{\{k\}} \frac{1}{k+1} \quad \textit{and} \quad \overline{\omega}_{i,j} \coloneqq \sum_{k\geq 0} \omega_{i,j}^{\{k\}} \frac{1}{k+1}. \tag{3.8}$$

We note that Definitions 3.2.1 and 3.2.2 differ slightly from those in [88], in that we consider these tendency coefficients to be organized into $s^{\{S\}} \times s^{\{S\}}$ matrices having first row identically zero.

3.2.1. Order Conditions

We derive order conditions for the slow tendency coefficients by first expressing IMEX-MRI-GARK methods in GARK form, following similar derivations for other infinitesimal methods [6, 84, 88, 102]. To express IMEX-MRI-GARK methods in GARK form, we must identify GARK tables $\mathbf{A}^{\{\sigma,\nu\}}$, $\mathbf{b}^{\sigma}$ and $\mathbf{c}^{\sigma}$ for $\sigma, \nu \in \{I, E, F\}$. To this end, we consider the inner fast modified IVP (3.6c) to be evolved using a single step of an arbitrary $s^{\{F\}}$-stage Runge–Kutta method with Butcher table $(A^{\{F,F\}}, b^{\{F\}}, c^{\{F\}})$, having order of accuracy q at least as accurate as the IMEX-MRI-GARK method. Thus the k^{th} fast stage ($k = 1, \ldots, s^{\{F\}}$) within the i^{th} slow stage ($i = 2, \ldots, s^{\{S\}}$) is given by:

$$\begin{aligned} Y_k^{\{F,i\}} = Y_{i-1}^{\{S\}} &+ H\Delta c_i^{\{S\}} \sum_{l=1}^{s^{\{F\}}} a_{k,l}^{\{F,F\}} f_l^{\{F,i\}} \\ &+ H \sum_{j=1}^{i} \left(\sum_{l=1}^{s^{\{F\}}} a_{k,l}^{\{F,F\}} \gamma_{i,j}\left(c_l^{\{F\}}\right) \right) f_j^{\{I\}} \\ &+ H \sum_{j=1}^{i-1} \left(\sum_{l=1}^{s^{\{F\}}} a_{k,l}^{\{F,F\}} \omega_{i,j}\left(c_l^{\{F\}}\right) \right) f_j^{\{E\}}, \end{aligned} \tag{3.9}$$

where $f_l^{\{F,i\}} := f^{\{F\}}\Big(T_{i-1} + c_l^{\{F\}}\Delta c_i^{\{S\}}H, Y_l^{\{F,i\}}\Big)$. Similarly, the slow stages in this scenario become:

$$\begin{aligned} Y_i^{\{S\}} &= Y_{i-1}^{\{S\}} + H\sum_{j=1}^{i}\left(\sum_{l=1}^{s^{\{F\}}} b_l^{\{F\}}\gamma_{i,j}\Big(c_l^{\{F\}}\Big)\right) f_j^{\{I\}} \\ &\quad + H\sum_{j=1}^{i-1}\left(\sum_{l=1}^{s^{\{F\}}} b_l^{\{F\}}\omega_{i,j}\Big(c_l^{\{F\}}\Big)\right) f_j^{\{E\}} \\ &\quad + H\Delta c_i^{\{S\}}\sum_{l=1}^{s^{\{F\}}} b_l^{\{F\}} f_l^{\{F,i\}}. \\ &= y_n + H\sum_{\lambda=1}^{i}\sum_{j=1}^{\lambda}\left(\sum_{l=1}^{s^{\{F\}}}\sum_{k\geq 0}\gamma_{\lambda,j}^{\{k\}} b_l^{\{F\}} c_l^{\{F\}\times k}\right) f_j^{\{I\}} \\ &\quad + H\sum_{\lambda=1}^{i}\sum_{j=1}^{\lambda-1}\left(\sum_{l=1}^{s^{\{F\}}}\sum_{k\geq 0}\omega_{\lambda,j}^{\{k\}} b_l^{\{F\}} c_l^{\{F\}\times k}\right) f_j^{\{E\}}, \\ &\quad + H\sum_{\lambda=1}^{i}\Delta c_\lambda^{\{S\}}\sum_{l=1}^{s^{\{F\}}} b_l^{\{F\}} f_l^{\{F,\lambda\}}, \end{aligned} \tag{3.10}$$

due to (3.7), and where we use the notation $c^{\times k}$ to indicate element-wise exponentiation. Then using (3.8) and our assumption that the fast Runge–Kutta method satisfies $b^{\{F\}T}c^{\{F\}\times k} = 1/(k+1)$ for $k = 1, \ldots, q$, we simplify (3.10) to obtain:

$$\begin{aligned} Y_i^{\{S\}} &= y_n + H\sum_{j=1}^{i}\sum_{\lambda=j}^{i}\overline{\gamma}_{\lambda,j} f_j^{\{I\}} + H\sum_{j=1}^{i-1}\sum_{\lambda=j}^{i}\overline{\omega}_{\lambda,j} f_j^{\{E\}} \\ &\quad + H\sum_{\lambda=1}^{i}\sum_{l=1}^{s^{\{F\}}}\Delta c_\lambda^{\{S\}} b_l^{\{F\}} f_l^{\{F,\lambda\}}. \end{aligned} \tag{3.11}$$

Recalling that the original IMEX-ARK method had an explicit first stage, (3.11) is equivalent to the standard GARK formulation,

$$Y_i^{\{S\}} = y_n + H\sum_{j=1}^{i} a_{i,j}^{\{I,I\}} f_j^{\{I\}} + H\sum_{j=1}^{i-1} a_{i,j}^{\{E,E\}} f_j^{\{E\}} + H\sum_{\lambda=1}^{i}\sum_{j=1}^{s^{\{F\}}} a_{i,j}^{\{S,F,\lambda\}} f_j^{\{F,\lambda\}}, \tag{3.12}$$

for slow stages $i = 1, \ldots, s^{\{S\}}$, where we identify the slow-implicit, slow-explicit and slow-fast coupling coefficients as:

$$a_{i,j}^{\{I,I\}} := \sum_{\lambda=j}^{i} \overline{\gamma}_{\lambda,j}, \quad a_{i,j}^{\{E,E\}} := \sum_{\lambda=j}^{i} \overline{\omega}_{\lambda,j}, \quad a_{i,j}^{\{S,F,\lambda\}} := \Delta c_\lambda^{\{S\}} b_j^{\{F\}}. \tag{3.13}$$

The first two of these may be represented as the GARK tables

$$\mathbf{A}^{\{I,I\}} := E\overline{\Gamma} = A^{\{I,I\}} \quad \text{and} \quad \mathbf{A}^{\{E,E\}} := E\overline{\Omega} = A^{\{E,E\}}, \tag{3.14}$$

where

$$E \in \mathbb{R}^{s^{\{S\}}\times s^{\{S\}}}, \quad E_{i,j} := \begin{cases} 1, & i \geq j, \\ 0, & \text{otherwise.} \end{cases}$$

We note that due to our assumptions on the underlying IMEX-ARK tables, $\overline{\Gamma}$ is lower-triangular and $\overline{\Omega}$ is strictly lower-triangular, with both having zero first row. Additionally, we note that the conditions $E\overline{\Gamma} = A^{\{I,I\}}$ and $E\overline{\Omega} = A^{\{E,E\}}$ in (3.14) also ensure consistency between the IMEX-MRI-GARK method (3.6) and the underlying IMEX-ARK method in the non-multirate case where $f^{\{F\}} = 0$.

Furthermore, since the GARK formulation of standard IMEX-ARK methods satisfies $A^{\{I,E\}} = A^{\{E,E\}}$ and $A^{\{E,I\}} = A^{\{I,I\}}$ (see [92]), the GARK formulation of our IMEX-MRI-GARK method results in the slow-explicit and slow-implicit portions having *shared* slow-fast coupling matrix $\mathbf{A}^{\{E,F\}} = \mathbf{A}^{\{I,F\}} := \mathbf{A}^{\{S,F\}} \in \mathbb{R}^{s^{\{S\}}\times s}$ with $s = s^{\{F\}}s^{\{S\}}$. From (3.13), we

have the sub-matrices

$$\mathbf{A}^{\{S,F,\lambda\}} := \Delta c_\lambda^{\{S\}}\, \mathbf{g}_\lambda\, b^{\{F\}T}, \quad \text{for} \quad \lambda = 1, \ldots, s^{\{S\}}, \tag{3.15}$$

where $\mathbf{g}_\lambda \in \mathbb{R}^{s^{\{S\}}}$ with

$$\Big(\mathbf{g}_\lambda\Big)_i := \begin{cases} 1, & i \geq \lambda, \\ 0, & \text{otherwise.} \end{cases}$$

Combining these into an overall slow-fast coupling matrix, we have

$$\mathbf{A}^{\{S,F\}} := \begin{bmatrix} \mathbf{A}^{\{S,F,1\}}, & \cdots, & \mathbf{A}^{\{S,F,s^{\{S\}}\}} \end{bmatrix} = \Delta C^{\{S\}} \otimes b^{\{F\}T}, \tag{3.16}$$

where

$$\Delta C^{\{S\}} := \begin{bmatrix} \Delta c_1^{\{S\}} & 0^{\{F\}T} & \cdots & 0^{\{F\}T} \\ \Delta c_1^{\{S\}} & \Delta c_2^{\{S\}} & \cdots & 0^{\{F\}T} \\ \vdots & \vdots & \ddots & 0^{\{F\}T} \\ \Delta c_1^{\{S\}} & \Delta c_2^{\{S\}} & \cdots & \Delta c_{s^{\{S\}}}^{\{S\}} \end{bmatrix},$$

and $0^{\{F\}}$ is a column vector of all zeros in $\mathbb{R}^{s^{\{F\}}}$.

For completeness, we note the corresponding GARK slow-implicit and slow-explicit coefficients [88],

$$\mathbf{b}^{\{I\}} := \mathbb{1}^{\{S\}T}\overline{\Gamma} = b^{\{I\}}, \tag{3.17}$$

$$\mathbf{c}^{\{I\}} := E\overline{\Gamma}\mathbb{1}^{\{S\}} = A^{\{I,I\}}\mathbb{1}^{\{S\}} = c^{\{S\}}, \tag{3.18}$$

$$\mathbf{b}^{\{E\}} := \mathbb{1}^{\{S\}T}\overline{\Omega} = b^{\{E\}}, \tag{3.19}$$

$$\mathbf{c}^{\{E\}} := E\overline{\Omega}\mathbb{1}^{\{S\}} = A^{\{E,E\}}\mathbb{1}^{\{S\}} = c^{\{S\}}, \tag{3.20}$$

where $\mathbb{1}^{\{S\}} \in \mathbb{R}^{s^{\{S\}}}$ is a column vector of all ones, and we have relied on our assumption of internal consistency in the underlying IMEX-ARK method. From enforcing the row-sum conditions on $\mathbf{A}^{\{S,F\}}$, we have

$$\mathbf{c}^{\{S,F\}} := \sum_{\lambda=1}^{s^{\{S\}}} \mathbf{A}^{\{S,F,\lambda\}}\mathbb{1}^{\{F\}} = \sum_{\lambda=1}^{s^{\{S\}}} \Delta c_\lambda \mathbf{g}_\lambda \tag{3.21}$$

$$\Rightarrow$$

$$\mathbf{c}_i^{\{S,F\}} = \sum_{\lambda=1}^{s^{\{S\}}} (c_\lambda^{\{S\}} - c_{\lambda-1}^{\{S\}})(\mathbf{g}_\lambda)_i = \sum_{\lambda=1}^{i} (c_\lambda^{\{S\}} - c_{\lambda-1}^{\{S\}}) = c_i^{\{S\}},$$

which ensures internal consistency between each partition of the GARK table (i.e., $\mathbf{c}^{\{I,I\}} = \mathbf{c}^{\{E,E\}} = \mathbf{c}^{\{S,F\}} = c^{\{S\}}$).

To reveal the GARK coefficients for the fast method and fast-slow couplings, we insert (3.11) into (3.9) to write the k^{th} fast stage ($k = 1, \ldots, s^{\{F\}}$) within the i^{th} slow stage ($i = 2, \ldots, s^{\{S\}}$) as:

$$\begin{aligned} Y_k^{\{F,i\}} = y_n &+ H\sum_{\lambda=1}^{i-1}\sum_{l=1}^{s^{\{F\}}} \Delta c_\lambda^{\{S\}} b_l^{\{F\}} f_l^{\{F,\lambda\}} + H\Delta c_i^{\{S\}} \sum_{l=1}^{s^{\{F\}}} a_{k,l}^{\{F,F\}} f_l^{\{F,i\}} \\ &+ H\sum_{j=1}^{i-1} a_{i-1,j}^{\{I,I\}} f_j^{\{I\}} + H\sum_{j=1}^{i}\left(\sum_{l=1}^{s^{\{F\}}} a_{k,l}^{\{F,F\}} \gamma_{i,j}\left(c_l^{\{F\}}\right) f_j^{\{I\}}\right) \\ &+ H\sum_{j=1}^{i-2} a_{i-1,j}^{\{E,E\}} f_j^{\{E\}} + H\sum_{j=1}^{i-1}\left(\sum_{l=1}^{s^{\{F\}}} a_{k,l}^{\{F,F\}} \omega_{i,j}\left(c_l^{\{F\}}\right) f_j^{\{E\}}\right). \end{aligned} \tag{3.22}$$

The fast method coefficients are therefore:

$$\mathbf{A}^{\{F,F\}} := \begin{bmatrix} \Delta c_1^{\{S\}} A^{\{F,F\}} & 0_{s^{\{F\}}\times s^{\{F\}}} & \cdots & 0_{s^{\{F\}}\times s^{\{F\}}} \\ \Delta c_1^{\{S\}} \mathbb{1}^{\{F\}} b^{\{F\}T} & \Delta c_2^{\{S\}} A^{\{F,F\}} & \cdots & \vdots \\ & \Delta c_2^{\{S\}} \mathbb{1}^{\{F\}} b^{\{F\}T} & \cdots & \\ \vdots & \vdots & \ddots & \vdots \\ \Delta c_1^{\{S\}} \mathbb{1}^{\{F\}} b^{\{F\}T} & \Delta c_2^{\{S\}} \mathbb{1}^{\{F\}} b^{\{F\}T} & \cdots & \Delta c_{s^{\{S\}}}^{\{S\}} A^{\{F,F\}} \end{bmatrix} \tag{3.23}$$

$$= \operatorname{diag}\big(\Delta c^{\{S\}}\big) \otimes A^{\{F,F\}} + L\Delta C^{\{S\}} \otimes \mathbb{1}^{\{F\}} b^{\{F\}T} \in \mathbb{R}^{s\times s},$$

where $\operatorname{diag}\big(\Delta c^{\{S\}}\big)$ is the diagonal matrix obtained by taking $\Delta c^{\{S\}}$ as its diagonal entries, and where $L \in \mathbb{R}^{s^{\{S\}}\times s^{\{S\}}}$ has entries $L_{i,j} := \delta_{i,j+1}$; similarly,

$$\mathbf{c}^{\{F\}} := \begin{bmatrix} \Delta c_1^{\{S\}} c^{\{F\}} \\ c_1^{\{S\}} \mathbb{1}^{\{F\}} + \Delta c_2^{\{S\}} c^{\{F\}} \\ \vdots \\ c_{s^{\{S\}}-1}^{\{S\}} \mathbb{1}^{\{F\}} + \Delta c_{s^{\{S\}}}^{\{S\}} c^{\{F\}} \end{bmatrix} = Lc^{\{S\}} \otimes \mathbb{1}^{\{F\}} + \Delta c^{\{S\}} \otimes c^{\{F\}} \in \mathbb{R}^s \tag{3.24}$$

and

$$\mathbf{b}^{\{F\}} := \begin{bmatrix} \Delta c_1^{\{S\}} b^{\{F\}} \\ \vdots \\ \Delta c_{s^{\{S\}}}^{\{S\}} b^{\{F\}} \end{bmatrix} = \Delta c^{\{S\}} \otimes b^{\{F\}} \in \mathbb{R}^s. \tag{3.25}$$

Finally, the fast-implicit and fast-explicit coupling coefficients are

$$\mathbf{A}^{\{F,I\}} := \begin{bmatrix} 0_{s^{\{F\}} \times s^{\{S\}}} \\ \mathbb{1}^{\{F\}} \big(e_1^T A^{\{I,I\}}\big) + \sum_{k \geq 0} \big(A^{\{F,F\}} c^{\{F\} \times k}\big) \big(e_2^T \Gamma^{\{k\}}\big) \\ \vdots \\ \mathbb{1}^{\{F\}} \big(e_{s^{\{S\}}-1}^T A^{\{I,I\}}\big) + \sum_{k \geq 0} \big(A^{\{F,F\}} c^{\{F\} \times k}\big) \big(e_{s^{\{S\}}}^T \Gamma^{\{k\}}\big) \end{bmatrix} \tag{3.26}$$

$$= LA^{\{I,I\}} \otimes \mathbb{1}^{\{F\}} + \sum_{k \geq 0} \Gamma^{\{k\}} \otimes \big(A^{\{F,F\}} c^{\{F\} \times k}\big) \in \mathbb{R}^{s \times s^{\{S\}}}$$

and

$$\mathbf{A}^{\{F,E\}} := \begin{bmatrix} 0_{s^{\{F\}} \times s^{\{S\}}} \\ \mathbb{1}^{\{F\}} \big(e_1^T A^{\{E,E\}}\big) + \sum_{k \geq 0} \big(A^{\{F,F\}} c^{\{F\} \times k}\big) \big(e_2^T \Omega^{\{k\}}\big) \\ \vdots \\ \mathbb{1}^{\{F\}} \big(e_{s^{\{S\}}-1}^T A^{\{E,E\}}\big) + \sum_{k \geq 0} \big(A^{\{F,F\}} c^{\{F\} \times k}\big) \big(e_{s^{\{S\}}}^T \Omega^{\{k\}}\big) \end{bmatrix} \tag{3.27}$$

$$= LA^{\{E,E\}} \otimes \mathbb{1}^{\{F\}} + \sum_{k \geq 0} \Omega^{\{k\}} \otimes \big(A^{\{F,F\}} c^{\{F\} \times k}\big) \in \mathbb{R}^{s \times s^{\{S\}}},$$

where we have leveraged the fact that $\Gamma^{\{k\}}$ and $\Omega^{\{k\}}$ have zero first row. These give rise to

$$\mathbf{c}^{\{F,I\}} := Lc^{\{S\}} \otimes \mathbb{1}^{\{F\}} + \sum_{k \geq 0} \Gamma^{\{k\}} \mathbb{1}^{\{S\}} \otimes (A^{\{F,F\}} c^{\{F\} \times k}), \quad \text{and} \tag{3.28}$$

$$\mathbf{c}^{\{F,E\}} := Lc^{\{S\}} \otimes \mathbb{1}^{\{F\}} + \sum_{k \geq 0} \Omega^{\{k\}} \mathbb{1}^{\{S\}} \otimes (A^{\{F,F\}} c^{\{F\} \times k}). \tag{3.29}$$

Theorem 3.2.1 (Internal consistency conditions)**.** *IMEX-MRI-GARK methods fulfill the "internal consistency" conditions:*

$$\mathbf{c}^{\{I,F\}} = \mathbf{c}^{\{E,F\}} = \mathbf{c}^{\{S,F\}} = \mathbf{c}^{\{S\}} \equiv c^{\{S\}}, \quad \textit{and} \tag{3.30}$$

$$\mathbf{c}^{\{F,I\}} = \mathbf{c}^{\{F,E\}} = \mathbf{c}^{\{F\}}, \tag{3.31}$$

for any fast method if and only if the following conditions hold:

$$\Gamma^{\{0\}}\mathbb{1}^{\{S\}} = \Omega^{\{0\}}\mathbb{1}^{\{S\}} = \Delta c^{\{S\}} \quad \textit{and} \quad \Gamma^{\{k\}}\mathbb{1}^{\{S\}} = \Omega^{\{k\}}\mathbb{1}^{\{S\}} = 0 \quad \forall k \geq 1. \tag{3.32}$$

Proof. From the definition of $\mathbf{c}^{\{S,F\}}$ in equation (3.21), we have already shown that (3.30) is satisfied. Now

$$\mathbf{c}^{\{F,I\}} = \mathbf{c}^{\{F\}} \Leftrightarrow$$

$$Lc^{\{S\}} \otimes \mathbb{1}^{\{F\}} + \sum_{k\geq 0} \Gamma^{\{k\}}\mathbb{1}^{\{S\}} \otimes (A^{\{F,F\}}c^{\{F\}\times k}) = Lc^{\{S\}} \otimes \mathbb{1}^{\{F\}} + \Delta c^{\{S\}} \otimes c^{\{F\}},$$

and similarly

$$\mathbf{c}^{\{F,E\}} = \mathbf{c}^{\{F\}} \Leftrightarrow$$

$$Lc^{\{S\}} \otimes \mathbb{1}^{\{F\}} + \sum_{k\geq 0} \Omega^{\{k\}}\mathbb{1}^{\{S\}} \otimes (A^{\{F,F\}}c^{\{F\}\times k}) = Lc^{\{S\}} \otimes \mathbb{1}^{\{F\}} + \Delta c^{\{S\}} \otimes c^{\{F\}},$$

which are equivalent to the conditions (3.32). □

3.2.1.1. IMEX-MRI-GARK Order Conditions

Due to the structure of the IMEX-MRI-GARK method (3.6), many of the GARK order conditions are automatically satisfied. As discussed in [88], since $\mathbf{A}^{\{I,I\}} = A^{\{I,I\}}$, $\mathbf{A}^{\{E,E\}} =$

$A^{\{E,E\}}$, $\mathbf{b}^{\{I\}} = b^{\{I\}}$, $\mathbf{b}^{\{E\}} = b^{\{E\}}$, $\mathbf{c}^{\{I\}} = c^{\{S\}}$, and $\mathbf{c}^{\{E\}} = c^{\{S\}}$ from (3.14) and (3.17)-(3.20), and since our base IMEX-ARK method has order q, then all of the GARK order conditions up to order q corresponding to only the "slow" components (and their couplings) will be satisfied. Similarly, since 'infinitesimal' methods assume that the fast component is solved exactly (or at least using an approximation of order $\geq q$), then the "fast" GARK order q conditions will similarly be satisfied. Additionally as discussed in [92], if all component tables have order at least two, then an IMEX-MRI-GARK method (3.6) that satisfies the internal consistency conditions from Theorem 3.2.1 will be at least second-order accurate. Therefore, in this section we focus on only the remaining coupling conditions between the fast and slow components (both implicit and explicit) for orders three and four.

We make use of the following simplifying conditions as listed in Lemma 3.8 of [88], reproduced here in matrix form, taking into account the structure of our slow base IMEX-ARK method:

$$\mathbf{A}^{\{S,F\}}\mathbf{c}^{\{F\}} = \frac{1}{2}c^{\{S\}\times 2}, \tag{3.33}$$

$$\mathbf{b}^{\{I\}T}\mathbf{A}^{\{S,F\}} = \left((\Delta c^{\{S\}} \times (Db^{\{I\}})) \otimes b^{\{F\}}\right)^T, \tag{3.34}$$

$$\mathbf{b}^{\{E\}T}\mathbf{A}^{\{S,F\}} = \left((\Delta c^{\{S\}} \times (Db^{\{E\}})) \otimes b^{\{F\}}\right)^T, \tag{3.35}$$

$$\mathbf{b}^{\{F\}T}\mathbf{A}^{\{F,I\}} = \Delta c^{\{S\}T}\mathcal{A}^{\{I,\zeta\}}, \tag{3.36}$$

$$\mathbf{b}^{\{F\}T}\mathbf{A}^{\{F,E\}} = \Delta c^{\{S\}T}\mathcal{A}^{\{E,\zeta\}}, \tag{3.37}$$

$$\mathbf{A}^{\{F,I\}}\mathbf{c}^{\{S\}} = \left((LA^{\{I,I\}}) \otimes \mathbb{1}^{\{F\}} + \sum_{k\geq 0} \Gamma^{\{k\}} \otimes \left(A^{\{F,F\}}c^{\{F\}\times k}\right)\right)c^{\{S\}}, \tag{3.38}$$

$$\mathbf{A}^{\{F,E\}}\mathbf{c}^{\{S\}} = \left((LA^{\{E,E\}}) \otimes \mathbb{1}^{\{F\}} + \sum_{k\geq 0} \Omega^{\{k\}} \otimes \left(A^{\{F,F\}}c^{\{F\}\times k}\right)\right)c^{\{S\}}, \tag{3.39}$$

and

$$\mathbf{A}^{\{F,F\}}\mathbf{c}^{\{F\}} = \frac{1}{2}(Lc^{\{S\}})^{\times 2} \otimes \mathbb{1}^{\{F\}} + \Big((Lc^{\{S\}}) \times \Delta c^{\{S\}}\Big) \otimes c^{\{F\}} \tag{3.40}$$

$$+ \Delta c^{\{S\}\times 2} \otimes \Big(A^{\{F,F\}}c^{\{F\}}\Big),$$

where we use the notation $a \times b$ to indicate element-wise multiplication of two vectors, and where we define

$$\mathcal{A}^{\{I,\zeta\}} = LA^{\{I,I\}} + \sum_{k\geq 0}\zeta_k\Gamma^{\{k\}}, \qquad \mathcal{A}^{\{E,\zeta\}} = LA^{\{E,E\}} + \sum_{k\geq 0}\zeta_k\Omega^{\{k\}}, \tag{3.41}$$

$$L_{i,j} = \delta_{i,j+1}, \qquad D_{i,j} = \begin{cases} 1, & j \geq i, \\ 0, & \text{otherwise}, \end{cases}$$

and

$$\zeta_k = b^{\{F\}T}A^{\{F,F\}}c^{\{F\}\times k}. \tag{3.42}$$

Theorem 3.2.2 (Third-order conditions)**.** *An internally consistent IMEX-MRI-GARK method* (3.6) *has order three iff the base IMEX-ARK method has order at least three, and the coupling conditions*

$$\Delta c^{\{S\}T}\mathcal{A}^{\{I,\zeta\}}c^{\{S\}} = \frac{1}{6} \qquad \textit{and} \qquad \Delta c^{\{S\}T}\mathcal{A}^{\{E,\zeta\}}c^{\{S\}} = \frac{1}{6} \tag{3.43}$$

hold, where $\mathcal{A}^{\{I,\zeta\}}$ *and* $\mathcal{A}^{\{E,\zeta\}}$ *are defined in equation* (3.41)*.*

Proof. Using (3.33), we have that

$$\mathbf{b}^{\{\sigma\}T}\mathbf{A}^{\{S,F\}}\mathbf{c}^{\{F\}} = \frac{1}{2}b^{\{S\}T}c^{\{S\}\times 2} = \frac{1}{2}\Big(\frac{1}{3}\Big)$$

for $\sigma \in \{I, E\}$, and thus two of the third-order GARK conditions are automatically satisfied. Similarly, from (3.36) and (3.37) we have

$$\mathbf{b}^{\{F\}T}\mathbf{A}^{\{F,\sigma\}}\mathbf{c}^{\{S\}} = \Delta c^{\{S\}T}\mathcal{A}^{\{\sigma,\zeta\}}c^{\{S\}},$$

which result in the conditions (3.43). □

Theorem 3.2.3 (Fourth-order conditions)**.** *An IMEX-MRI-GARK method* (3.6) *that satisfies Theorem 3.2.2 has order four iff the base IMEX-ARK method has order at least four, and the following coupling conditions hold for* $\sigma, \nu \in \{I, E\}$*:*

$$\left(\Delta c^{\{S\}} \times Lc^{\{S\}}\right)^T \mathcal{A}^{\{\sigma,\zeta\}}c^{\{S\}} + \left(\Delta c^{\{S\}\times 2}\right)^T \mathcal{A}^{\{\sigma,\beta\}}c^{\{S\}} = \frac{1}{8}, \tag{3.44a}$$

$$\Delta c^{\{S\}T}\mathcal{A}^{\{\sigma,\zeta\}}c^{\{S\}\times 2} = \frac{1}{12}, \tag{3.44b}$$

$$\left(\Delta c^{\{S\}} \times (Db^{\{\sigma\}})\right)^T \mathcal{A}^{\{\nu,\zeta\}}c^{\{S\}} = \frac{1}{24}, \tag{3.44c}$$

$$\left(\Delta c^{\{S\}\times 2}\right)^T \mathcal{A}^{\{\sigma,\xi\}}c^{\{S\}} + \Delta c^{\{S\}T}L\Delta C^{\{S\}}\mathcal{A}^{\{\sigma,\zeta\}}c^{\{S\}} = \frac{1}{24}, \quad \textit{and} \tag{3.44d}$$

$$\Delta c^{\{S\}T}\mathcal{A}^{\{\sigma,\zeta\}}A^{\{\nu,\nu\}}c^{\{S\}} = \frac{1}{24}, \tag{3.44e}$$

where we have defined the auxiliary variables

$$\mathcal{A}^{\{I,\beta\}} := \frac{1}{2}LA^{\{I,I\}} + \sum_{k\geq 0}\beta_k\Gamma^{\{k\}}, \tag{3.45}$$

$$\mathcal{A}^{\{E,\beta\}} := \frac{1}{2}LA^{\{E,E\}} + \sum_{k\geq 0}\beta_k\Omega^{\{k\}}, \tag{3.46}$$

$$\mathcal{A}^{\{I,\xi\}} := \frac{1}{2}LA^{\{I,I\}} + \sum_{k\geq 0}\xi_k\Gamma^{\{k\}}, \tag{3.47}$$

$$\mathcal{A}^{\{E,\xi\}} := \frac{1}{2} L A^{\{E,E\}} + \sum_{k\geq 0} \xi_k \Omega^{\{k\}}, \tag{3.48}$$

$$\beta_k := (b^{\{F\}} \times c^{\{F\}})^T A^{\{F,F\}} c^{\{F\}\times k}, \quad \textit{and} \tag{3.49}$$

$$\xi_k := b^{\{F\}T} A^{\{F,F\}} A^{\{F,F\}} c^{\{F\}\times k}. \tag{3.50}$$

Proof. Since the GARK representation of our IMEX-MRI-GARK method is internally consistent, there are 26 coupling conditions of order 4. Of these, ten are automatically satisfied due the IMEX-MRI-GARK method structure and our assumed accuracy of the base IMEX-ARK method: for $\sigma, \nu \in \{I, E\}$,

$$\left(\mathbf{b}^{\{\sigma\}} \times \mathbf{c}^{\{S\}}\right)^T \mathbf{A}^{\{S,F\}} \mathbf{c}^{\{F\}} = \frac{1}{8}, \tag{3.51a}$$

$$\mathbf{b}^{\{\sigma\}T} \mathbf{A}^{\{\nu,\nu\}} \mathbf{A}^{\{S,F\}} \mathbf{c}^{\{F\}} = \frac{1}{24}, \tag{3.51b}$$

$$\mathbf{b}^{\{\sigma\}T} \mathbf{A}^{\{S,F\}} \mathbf{c}^{\{F\}\times 2} = \frac{1}{12}, \quad \text{and} \tag{3.51c}$$

$$\mathbf{b}^{\{\sigma\}T} \mathbf{A}^{\{S,F\}} \mathbf{A}^{\{F,F\}} \mathbf{c}^{\{F\}} = \frac{1}{24}. \tag{3.51d}$$

The remaining 16 coupling conditions are

$$\left(\mathbf{b}^{\{F\}} \times \mathbf{c}^{\{F\}}\right)^T \mathbf{A}^{\{F,\sigma\}} \mathbf{c}^{\{S\}} = \frac{1}{8}, \tag{3.52a}$$

$$\mathbf{b}^{\{F\}T} \mathbf{A}^{\{F,\sigma\}} c^{\{S\}\times 2} = \frac{1}{12}, \tag{3.52b}$$

$$\mathbf{b}^{\{\sigma\}T} \mathbf{A}^{\{S,F\}} \mathbf{A}^{\{F,\nu\}} \mathbf{c}^{\{S\}} = \frac{1}{24}, \tag{3.52c}$$

$$\mathbf{b}^{\{F\}T} \mathbf{A}^{\{F,F\}} \mathbf{A}^{\{F,\sigma\}} \mathbf{c}^{\{S\}} = \frac{1}{24}, \tag{3.52d}$$

$$\mathbf{b}^{\{F\}T} \mathbf{A}^{\{F,\sigma\}} \mathbf{A}^{\{\nu,\nu\}} \mathbf{c}^{\{S\}} = \frac{1}{24}, \quad \text{and} \tag{3.52e}$$

$$\mathbf{b}^{\{F\}T}\mathbf{A}^{\{F,\sigma\}}\mathbf{A}^{\{S,F\}}\mathbf{c}^{\{F\}} = \frac{1}{24}, \tag{3.52f}$$

where again $\sigma, \nu \in \{I, E\}$.

We first prove the automatically-satisfied conditions (3.51). Using (3.33) and our assumption that the base IMEX-ARK method is order four,

$$\left(\mathbf{b}^{\{\sigma\}} \times \mathbf{c}^{\{S\}}\right)^T \mathbf{A}^{\{S,F\}}\mathbf{c}^{\{F\}} = \frac{1}{2}b^{\{\sigma\}T}c^{\{S\}\times 3} = \frac{1}{2}\left(\frac{1}{4}\right)$$

and

$$\mathbf{b}^{\{\sigma\}T}\mathbf{A}^{\{\nu,\nu\}}\mathbf{A}^{\{S,F\}}\mathbf{c}^{\{F\}} = \frac{1}{2}b^{\{\sigma\}T}A^{\{\nu,\nu\}}c^{\{S\}\times 2} = \frac{1}{2}\left(\frac{1}{12}\right),$$

for $\sigma, \nu \in \{I, E\}$, and hence (3.51a) and (3.51b) are satisfied. Using the definition of $\mathbf{c}^{\{F\}}$ from (3.24), the simplifying formulas (3.34)-(3.35), and our assumptions that $c_1^{\{S\}} = 0$, the fast method is at least third-order, and the IMEX-ARK method is at least fourth-order, we have for $\sigma \in \{I, E\}$:

$$\begin{aligned}
&\mathbf{b}^{\{\sigma\}T}\mathbf{A}^{\{S,F\}}\mathbf{c}^{\{F\}\times 2} \\
&= \left((\Delta c^{\{S\}} \times (Db^{\{\sigma\}})) \otimes b^{\{F\}}\right)^T \left(Lc^{\{S\}} \otimes \mathbb{1}^{\{F\}} + \Delta c^{\{S\}} \otimes c^{\{F\}}\right)^{\times 2} \\
&= \left(\Delta c^{\{S\}} \times (Db^{\{\sigma\}})\right)^T \left((Lc^{\{S\}})^{\times 2} + (Lc^{\{S\}} \times \Delta c^{\{S\}}) + \frac{1}{3}\Delta c^{\{S\}\times 2}\right) \\
&= \left(Db^{\{\sigma\}}\right)^T \left((Lc^{\{S\}})^{\times 2} \times \Delta c^{\{S\}} + Lc^{\{S\}} \times \Delta c^{\{S\}\times 2} + \frac{1}{3}\Delta c^{\{S\}\times 3}\right) \\
&= \frac{1}{3}\left(Db^{\{\sigma\}}\right)^T \left(c^{\{S\}\times 3} - (Lc^{\{S\}})^{\times 3}\right) \\
&= \frac{1}{3}\sum_{i=2}^{s^{\{S\}}} \left(\sum_{l=i}^{s^{\{S\}}} b_l^{\{\sigma\}}\right) \left(c_i^{\{S\}\times 3} - c_{i-1}^{\{S\}\times 3}\right) = \frac{1}{3}b^{\{\sigma\}T}c^{\{S\}\times 3} = \frac{1}{3}\left(\frac{1}{4}\right),
\end{aligned}$$

which proves the coupling conditions (3.51c). Using the simplifying formulas (3.34), (3.35) and (3.40), and the same assumptions as in the previous step, for $\sigma \in \{I, E\}$ we have

$$
\begin{aligned}
&\mathbf{b}^{\{\sigma\}T}\mathbf{A}^{\{S,F\}}\mathbf{A}^{\{F,F\}}\mathbf{c}^{\{F\}} \\
&= \left((\Delta c^{\{S\}} \times (Db^{\{\sigma\}})) \otimes b^{\{F\}}\right)^T \left(\frac{1}{2}(Lc^{\{S\}})^{\times 2} \otimes \mathbb{1}^{\{F\}}\right. \\
&\quad \left. + \left((Lc^{\{S\}}) \times \Delta c^{\{S\}}\right) \otimes c^{\{F\}} + \Delta c^{\{S\}\times 2} \otimes \left(A^{\{F,F\}}c^{\{F\}}\right)\right) \\
&= \left(\Delta c^{\{S\}} \times (Db^{\{\sigma\})}\right)^T \left(\frac{1}{2}(Lc^{\{S\}})^{\times 2} + \frac{1}{2}(Lc^{\{S\}}) \times \Delta c^{\{S\}} + \frac{1}{6}\Delta c^{\{S\}\times 2}\right) \\
&= \frac{1}{6}\left(Db^{\{\sigma\}}\right)^T \left(c^{\{S\}\times 3} - (Lc^{\{S\}})^{\times 3}\right) = \frac{1}{6}b^{\{\sigma\}T}c^{\{S\}\times 3} = \frac{1}{6}\left(\frac{1}{4}\right),
\end{aligned}
$$

and thus the coupling conditions (3.51d) are automatically satisfied as well.

We now examine the 16 remaining fourth-order GARK conditions (3.52). Starting with (3.52a), we use the definitions (3.25) and (3.24), the simplifying formulas (3.38)-(3.39), and that the fast method is at least second-order to obtain:

$$
\begin{aligned}
\frac{1}{8} &= \left(\mathbf{b}^{\{F\}} \times \mathbf{c}^{\{F\}}\right)^T \mathbf{A}^{\{F,I\}}\mathbf{c}^{\{S\}} \\
&= \left(\left(\Delta c^{\{S\}} \otimes b^{\{F\}}\right) \times \left(Lc^{\{S\}} \otimes \mathbb{1}^{\{F\}} + \Delta c^{\{S\}} \otimes c^{\{F\}}\right)\right)^T \\
&\quad \left(LA^{\{I,I\}} \otimes \mathbb{1}^{\{F\}} + \sum_{k\geq 0} \Gamma^{\{k\}} \otimes \left(A^{\{F,F\}}c^{\{F\}\times k}\right)\right) c^{\{S\}} \\
&= \left(\Delta c^{\{S\}} \times Lc^{\{S\}}\right)^T \mathcal{A}^{\{I,\zeta\}}c^{\{S\}} + \left(\Delta c^{\{S\}\times 2}\right)^T \mathcal{A}^{\{I,\beta\}}c^{\{S\}}.
\end{aligned}
$$

A similar argument gives

$$\frac{1}{8} = \left(\Delta c^{\{S\}} \times L c^{\{S\}}\right)^T \mathcal{A}^{\{E,\zeta\}} c^{\{S\}} + \left(\Delta c^{\{S\}\times 2}\right)^T \mathcal{A}^{\{E,\beta\}} c^{\{S\}},$$

which establishes the conditions (3.44a). Using the simplifying formulas (3.36)-(3.37), the order conditions (3.52b) become

$$\frac{1}{12} = \mathbf{b}^{\{F\}T} \mathbf{A}^{\{F,\sigma\}} \mathbf{c}^{\{S\}\times 2} = \Delta c^{\{S\}T} \mathcal{A}^{\{\sigma,\zeta\}} c^{\{S\}\times 2}$$

for $\sigma \in \{I, E\}$, which are equivalent to the conditions (3.44b). For the order conditions (3.52c), we use simplifying formulas (3.34)-(3.35) and (3.38) to obtain for $\sigma \in \{I, E\}$:

$$\begin{aligned}
\frac{1}{24} &= \mathbf{b}^{\{\sigma\}T} \mathbf{A}^{\{S,F\}} \mathbf{A}^{\{F,I\}} \mathbf{c}^{\{S\}} \\
&= \left((\Delta c^{\{S\}} \times (D b^{\{\sigma\}})) \otimes b^{\{F\}}\right)^T \left(L A^{\{I,I\}} \otimes \mathbb{1}^{\{F\}} + \sum_{k\geq 0} \Gamma^{\{k\}} \otimes (A^{\{F,F\}} c^{\{F\}\times k}\right) c^{\{S\}} \\
&= (\Delta c^{\{S\}} \times (D b^{\{\sigma\}}))^T \mathcal{A}^{\{I,\zeta\}} c^{\{S\}}.
\end{aligned}$$

Similarly using the simplifying formulas (3.34)-(3.35) and (3.39), we have

$$\frac{1}{24} = (\Delta c^{\{S\}} \times (D b^{\{\sigma\}}))^T \mathcal{A}^{\{E,\zeta\}} c^{\{S\}},$$

resulting in the conditions (3.44c). We use the definitions (3.25) and (3.23), and the simplifying formula (3.38) to convert the order condition (3.52d) for $\sigma = I$:

$$\begin{aligned}
\frac{1}{24} &= \mathbf{b}^{\{F\}T} \mathbf{A}^{\{F,F\}} \mathbf{A}^{\{F,I\}} \mathbf{c}^{\{S\}} \\
&= \left(\Delta c^{\{S\}} \otimes b^{\{F\}}\right)^T \left(\operatorname{diag}\left(\Delta c^{\{S\}}\right) \otimes A^{\{F,F\}} + L \Delta C^{\{S\}} \otimes \mathbb{1}^{\{F\}} b^{\{F\}T}\right)
\end{aligned}$$

$$\left(LA^{\{I,I\}} \otimes \mathbb{1}^{\{F\}} + \sum_{k\geq 0} \Gamma^{\{k\}} \otimes \left(A^{\{F,F\}} c^{\{F\}\times k} \right) \right) c^{\{S\}}$$

$$= \left(\left(\Delta c^{\{S\}\times 2} \right)^T \otimes \left(b^{\{F\}T} A^{\{F,F\}} \right) + \Delta c^{\{S\}T} L \Delta C^{\{S\}} \otimes b^{\{F\}T} \right)$$

$$\left(LA^{\{I,I\}} \otimes \mathbb{1}^{\{F\}} + \sum_{k\geq 0} \Gamma^{\{k\}} \otimes \left(A^{\{F,F\}} c^{\{F\}\times k} \right) \right) c^{\{S\}}$$

$$= \left(\Delta c^{\{S\}\times 2} \right)^T \mathcal{A}^{\{I,\xi\}} c^{\{S\}} + \Delta c^{\{S\}T} L \Delta C^{\{S\}} \mathcal{A}^{\{I,\zeta\}} c^{\{S\}}.$$

Similarly, the simplifying formula (3.39) converts (3.52d) for $\sigma = E$ to

$$\frac{1}{24} = \left(\Delta c^{\{S\}\times 2} \right)^T \mathcal{A}^{\{E,\xi\}} c^{\{S\}} + \Delta c^{\{S\}T} L \Delta C^{\{S\}} \mathcal{A}^{\{E,\zeta\}} c^{\{S\}},$$

which establishes the conditions (3.44d). Using the simplifying formulas (3.36) and (3.37), the order conditions (3.52e) become for $\sigma, \nu \in \{I, E\}$:

$$\frac{1}{24} = \mathbf{b}^{\{F\}T} \mathbf{A}^{\{F,\sigma\}} \mathbf{A}^{\{\nu,\nu\}} \mathbf{c}^{\{S\}} = \Delta c^{\{S\}T} \mathcal{A}^{\{\sigma,\zeta\}} A^{\{\nu,\nu\}} c^{\{S\}},$$

which are the coupling conditions (3.44e). The final order conditions, (3.52f), may be simplified using formulas (3.33) and (3.36)-(3.37) for $\sigma \in \{I, E\}$:

$$\frac{1}{24} = \mathbf{b}^{\{F\}T} \mathbf{A}^{\{F,\sigma\}} \mathbf{A}^{\{S,F\}} \mathbf{c}^{\{F\}} = \frac{1}{2} \Delta c^{\{S\}} \mathcal{A}^{\{\sigma,\zeta\}} c^{\{S\}\times 2},$$

which are equivalent to the coupling conditions (3.44b). □

Remark 3.2.1. *For many IMEX-ARK methods the coefficients are chosen so that* $\mathbf{b}^{\{E\}} = \mathbf{b}^{\{I\}}$ *to reduce the number of order conditions that must be satisfied. Similarly, when* $\mathbf{b}^{\{E\}} = \mathbf{b}^{\{I\}}$ *many of the 3-component GARK order conditions (on which IMEX-MRI-GARK methods rely) are duplicated. One could then wonder whether the assumption* $\mathbf{b}^{\{E\}} = \mathbf{b}^{\{I\}}$ *would significantly reduce the number of order conditions required to derive IMEX-MRI-GARK*

methods. This is not in fact the case, since the large majority of these duplicated GARK order conditions are already automatically satisfied in (3.51) *due to the IMEX-MRI-GARK structure and our assumptions on the order of the underlying IMEX-ARK method. Of the remaining 16 GARK order conditions in* (3.52) *that are not automatically satisfied, only the conditions* (3.52c) *(that correspond with the IMEX-MRI-GARK condition* (3.44c)*) benefit from an assumption that* $\mathbf{b}^{\{E\}} = \mathbf{b}^{\{I\}}$*, causing those 4 conditions to simplify to 2. Thus although all of the IMEX-MRI-GARK methods presented later in Section 3.4 are derived from IMEX-ARK methods satisfying* $\mathbf{b}^{\{E\}} = \mathbf{b}^{\{I\}}$*, this should by no means be considered as a requirement when deriving new IMEX-MRI-GARK methods.*

3.3. Linear stability

There is no standard theoretical framework for analyzing linear stability of methods for additive problems (of *either* form (3.1) or (3.2)). Thus although it relies on an assumption that the Jacobians with respect to y of $f^{\{I\}}$, $f^{\{E\}}$ and $f^{\{F\}}$ are simultaneously diagonalizable, similarly to [88] we analyze linear stability on an additive scalar test problem:

$$y' = \lambda^{\{F\}} y + \lambda^{\{E\}} y + \lambda^{\{I\}} y \tag{3.53}$$

where each of $\lambda^{\{F\}}, \lambda^{\{E\}}, \lambda^{\{I\}} \in \mathbb{C}^-$, and we define $z^{\{F\}} := H\lambda^{\{F\}}$, $z^{\{E\}} := H\lambda^{\{E\}}$, and $z^{\{I\}} := H\lambda^{\{I\}}$. Applying the IMEX-MRI-GARK method (3.6) to the scalar model problem (3.53), the modified fast IVP for each slow stage $i = 2, \ldots, s^{\{S\}}$ becomes:

$$\begin{aligned} v' &= \Delta c_i^{\{S\}} \lambda^{\{F\}} v + \lambda^{\{E\}} \sum_{j=1}^{i-1} \omega_{i,j}\left(\frac{\theta}{H}\right) Y_j^{\{S\}} + \lambda^{\{I\}} \sum_{j=1}^{i} \gamma_{i,j}\left(\frac{\theta}{H}\right) Y_j^{\{S\}} \\ &= \Delta c_i^{\{S\}} \lambda^{\{F\}} v + \lambda^{\{E\}} \sum_{j=1}^{i-1} \sum_{k\geq 0} \omega_{i,j}^{\{k\}} \frac{\theta^k}{H^k} Y_j^{\{S\}} + \lambda^{\{I\}} \sum_{j=1}^{i} \sum_{k\geq 0} \gamma_{i,j}^{\{k\}} \frac{\theta^k}{H^k} Y_j^{\{S\}}, \end{aligned}$$

for $\theta \in [0, H]$, with initial condition $v(0) = Y_{i-1}^{\{S\}}$. We solve for the updated slow stage $Y_i^{\{S\}} := v(H)$ analytically using the variation of constants formula:

$$Y_i^{\{S\}} = e^{\Delta c_i^{\{S\}} z^{\{F\}}} Y_{i-1}^{\{S\}} + z^{\{E\}} \sum_{j=1}^{i-1} \sum_{k\geq 0} \omega_{i,j}^{\{k\}} \left(\int_0^1 e^{\Delta c_i^{\{S\}} z^{\{F\}}(1-t)} t^k \mathrm{d}t \right) Y_j^{\{S\}} \tag{3.54}$$

$$+ z^{\{I\}} \sum_{j=1}^{i} \sum_{k\geq 0} \gamma_{i,j}^{\{k\}} \left(\int_0^1 e^{\Delta c_i^{\{S\}} z^{\{F\}}(1-t)} t^k \mathrm{d}t \right) Y_j^{\{S\}}$$

$$= \varphi_0\Big(\Delta c_i^{\{S\}} z^{\{F\}}\Big) Y_{i-1}^{\{S\}} + z^{\{E\}} \sum_{j=1}^{i-1} \eta_{i,j}(z^{\{F\}}) Y_j^{\{S\}} + z^{\{I\}} \sum_{j=1}^{i} \mu_{i,j}(z^{\{F\}}) Y_j^{\{S\}},$$

where η and μ depend on the fast variable:

$$\eta_{i,j}\big(z^{\{F\}}\big) = \sum_{k\geq 0} \omega_{i,j}^{\{k\}} \varphi_{k+1}\Big(\Delta c_i^{\{S\}} z^{\{F\}}\Big)$$

$$\mu_{i,j}\big(z^{\{F\}}\big) = \sum_{k\geq 0} \gamma_{i,j}^{\{k\}} \varphi_{k+1}\Big(\Delta c_i^{\{S\}} z^{\{F\}}\Big),$$

and the family of analytical functions $\{\varphi_k\}$ are defined as in [88],

$$\varphi_0(z) = e^z, \quad \varphi_k(z) = \int_0^1 e^{z(1-t)} t^{k-1} \mathrm{d}t, \quad k \geq 1,$$

or recursively as

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

Concatenating $Y = \begin{bmatrix} Y_1^{\{S\}T} & \dots & Y_{s^{\{S\}}}^{\{S\}T} \end{bmatrix}^T$, we can write (3.54) in matrix form as

$$Y = \mathrm{diag}\Big(\varphi_0\big(\Delta c^{\{S\}} z^{\{F\}}\big)\Big) LY + \varphi_0\big(\Delta c_1 z^{\{F\}}\big) y_n e_1$$

$$+ z^{\{E\}} \eta\big(z^{\{F\}}\big) Y + z^{\{I\}} \mu\big(z^{\{F\}}\big) Y$$

$$= \left(I - \operatorname{diag}\Big(\varphi_0\big(\Delta c^{\{S\}} z^{\{F\}}\big)\Big) L - z^{\{E\}} \eta\big(z^{\{F\}}\big) - z^{\{I\}} \mu\big(z^{\{F\}}\big) \right)^{-1} y_n e_1,$$

where

$$\eta\big(z^{\{F\}}\big) = \sum_{k \geq 0} \operatorname{diag}\Big(\varphi_{k+1}\big(\Delta c^{\{S\}} z^{\{F\}}\big)\Big) \Omega^{\{k\}} \quad \text{and}$$

$$\mu\big(z^{\{F\}}\big) = \sum_{k \geq 0} \operatorname{diag}\Big(\varphi_{k+1}\big(\Delta c^{\{S\}} z^{\{F\}}\big)\Big) \Gamma^{\{k\}}.$$

Thus the linear stability function for IMEX-MRI-GARK on the problem (3.53) becomes

$$R\big(z^{\{F\}}, z^{\{E\}}, z^{\{I\}}\big) \tag{3.55}$$

$$:= e_{s^{\{S\}}}^T \left(I - \operatorname{diag}\Big(\varphi_0\big(\Delta c^{\{S\}} z^{\{F\}}\big)\Big) L - z^{\{E\}} \eta\big(z^{\{F\}}\big) - z^{\{I\}} \mu\big(z^{\{F\}}\big) \right)^{-1} e_1.$$

Following a similar definition as in [111], we define the joint stability for the slow, nonstiff region as:

$$\mathcal{J}_{\alpha,\beta} := \left\{ z^{\{E\}} \in \mathbb{C}^- \ : \ |R(z^{\{F\}}, z^{\{E\}}, z^{\{I\}})| \leq 1, \ \forall z^{\{F\}} \in \mathcal{S}_\alpha^{\{F\}}, \ \forall z^{\{I\}} \in \mathcal{S}_\beta^{\{I\}} \right\}$$

where $\mathcal{S}_\alpha^\sigma := \{z^\sigma \in \mathbb{C}^- \ : \ |\arg(z^\sigma) - \pi| \leq \alpha\}$. Since such stability regions are not widespread in the literature, we highlight the role of each component, before plotting these for candidate IMEX-MRI-GARK methods in the next section. $\mathcal{J}_{\alpha,\beta}$ provides a plot of the stability region for the slow explicit component only, under assumptions that (a) $z^{\{I\}}$ can range throughout an entire infinitely-long sector $\mathcal{S}_\alpha^{\{I\}}$ in the complex left-half plane, and (b) $z^{\{F\}}$ can range throughout another [infinite] sector $\mathcal{S}_\beta^{\{F\}}$ in $\mathbb{C}^-$. These sectors both include the entire negative real axis, as well as a swath of values with angle at most α or β above and below this axis, respectively. As such, one should expect the joint stability region $\mathcal{J}_{\alpha,\beta}$ to be significantly smaller than the standard stability region for just the slow explicit table $(A^{\{E\}}, b^{\{E\}}, c^{\{E\}})$,

and to shrink in size as both α, β increase. Furthermore, we note that this notion of a joint stability region is artificially restrictive, since in practice the functions $f^{\{I\}}$ and $f^{\{F\}}$ will not be *infinitely* stiffer than $f^{\{E\}}$.

3.4. Example IMEX-MRI-GARK methods

While our focus in this paper is on the underlying theory regarding IMEX-MRI-GARK methods of the form (3.2.1), in this section we discuss how IMEX-MRI-GARK methods may be constructed, and provide methods of orders 3 and 4 to use in demonstrating our numerical results in Section 3.5.

3.4.1. Third-order Methods

We create two third-order IMEX-MRI-GARK methods, both based on the '(3,4,3)' IMEX-ARK method from Section 2.7 of [4],

$$\begin{array}{c|cccc|cccc}
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\eta & \eta & 0 & 0 & 0 & 0 & \eta & 0 & 0 \\
\frac{1+\eta}{2} & a_{3,1} & a_{3,2} & 0 & 0 & 0 & \frac{1-\eta}{2} & \eta & 0 \\
1 & 1-2\alpha & \alpha & \alpha & 0 & 0 & b_2 & b_3 & \eta \\
\hline
1 & 0 & b_2 & b_3 & \eta & 0 & b_2 & b_3 & \eta
\end{array}$$

where

$$\eta = 0.435866521508458999416019451193556842529 3,$$

$$\alpha = 0.552929148035939819361188729738592476494 9,$$

$$a_{3,2} = \left(-\frac{15}{4} + 15\eta - \frac{21}{4}\eta^2\right)\alpha + 4 - \frac{25}{2}\eta + \frac{9}{2}\eta^2,$$

$$a_{3,1} = \left(\frac{15}{4} - 15\eta + \frac{21}{4}\eta^2\right)\alpha - \frac{7}{2} + 13\eta - \frac{9}{2}\eta^2,$$

$$b_2 = -\frac{3}{2}\eta^2 + 4\eta - \frac{1}{4},$$

$$b_3 = \frac{3}{2}\eta^2 - 5\eta + \frac{5}{4}.$$

As the explicit portion of this pair is not 'stiffly accurate' we pad the tables as discussed in Section 3.2. We then convert this to 'solve-decoupled' form [88] by inserting additional rows and columns into the tables to ensure that any stage with a nonzero diagonal value in the slow implicit table is associated with $\Delta c_i = 0$,

0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
η	η	0	0	0	0	0	0	0	η	0	0	0	0	0	0	0
η	η	0	0	0	0	0	0	0	0	0	η	0	0	0	0	0
$\frac{1+\eta}{2}$	□	0	□	0	0	0	0	0	□	0	□	0	0	0	0	0
$\frac{1+\eta}{2}$	$a_{3,1}$	0	$a_{3,2}$	0	0	0	0	0	0	0	$\frac{1-\eta}{2}$	0	η	0	0	0
1	□	0	□	0	□	0	0	0	□	0	□	0	□	0	0	0
1	$1-2\alpha$	0	α	0	α	0	0	0	0	0	b_2	0	b_3	0	η	0
1	0	0	b_2	0	b_3	0	η	0	0	0	b_2	0	b_3	0	η	0
1	0	0	b_2	0	b_3	0	η	0	0	0	b_2	0	b_3	0	η	0

where each entry in $A^{\{E,E\}}$ and $A^{\{I,I\}}$ above labeled with □ need only be chosen to satisfy internal consistency for the ARK table. We note that although the proposed IMEX-MRI-GARK methods (3.6) do not *require* that the implicit portion of the IMEX-ARK table have this 'solve-decoupled' pattern, we create tables with this structure due to their ease of implementation. Specifically, if the corresponding IMEX-MRI-GARK method included a 'solve-coupled' stage i (i.e., both $\overline{\gamma}_{i,i} \neq 0$ and $\Delta c_i \neq 0$), then the stage solution $Y_i^{\{S\}}$ must *both* define the fast IVP right-hand side (3.6c),

$$v'(\theta) = \Delta c_i^{\{S\}} f^{\{F\}}\Big(T_{i-1} + \Delta c_i^{\{S\}}\theta,\, v(\theta)\Big) + \sum_{j=1}^{i-1}\left(\gamma_{i,j}\left(\tfrac{\theta}{H}\right) f_j^{\{I\}} + \omega_{i,j}\left(\tfrac{\theta}{H}\right) f_j^{\{E\}}\right)$$

$$+ \gamma_{i,i}\left(\tfrac{\theta}{H}\right) f^{\{I\}}\left(t_n + c_i^{\{S\}} H, Y_i^{\{S\}}\right), \quad \theta \in [0, H],$$

and be the solution to this fast IVP, $Y_i^{\{S\}} = v(H)$. Solve-decoupled methods, on the other hand, may be performed by alternating between standard implicit solves for each implicit stage, followed by fast evolution for non-implicit stages. However, as noted in [81, 84], while the solve-decoupled approach makes for easier implementation of MRI methods, it also results in methods with diminished stability.

The first IMEX-MRI-GARK that we built from the table above is "IMEX-MRI-GARK3a". We simultaneously found the 10 $\square$ values to complete the IMEX-ARK table, the 24 unknown $\Gamma^{\{0\}}$ coefficients and the 20 unknown $\Omega^{\{0\}}$ coefficients by solving the ARK consistency conditions (3.14), the internal consistency conditions (3.32), and the third-order conditions (3.56). Since this only constitutes 50 unique conditions that depend linearly on 54 unknown entries, the corresponding linear system of equations was under-determined. For IMEX-MRI-GARK3a we used the particular solution returned by MATLAB (a basic least-squares solution). The resulting nonzero coefficients $\mathbf{c}^{\{S\}}$, $\Gamma^{\{0\}}$ and $\Omega^{\{0\}}$ are provided in Appendix A.1.

Our second IMEX-MRI-GARK constructed from this same base IMEX-ARK table is "IMEX-MRI-GARK3b". Here, beginning with the IMEX-MRI-GARK3a particular solution above, we then used the four remaining free variables to maximize the extent of the joint stability region along the negative real-axis. The nonzero coefficients $\mathbf{c}^{\{S\}}$, $\Gamma^{\{0\}}$ and $\Omega^{\{0\}}$ for the resulting method are given in Appendix A.2.

Remark 3.4.1. *An alternative approach for creating solve-decoupled third-order IMEX-MRI-GARK methods is to take advantage of the free $\square$ variables within the extended IMEX-ARK table, plus assumptions that $\overline{\Gamma} = \Gamma^{\{0\}}$ and $\overline{\Omega} = \Omega^{\{0\}}$. Here, one may select the $\square$ values to ensure that the IMEX-ARK is internally consistent and satisfies*

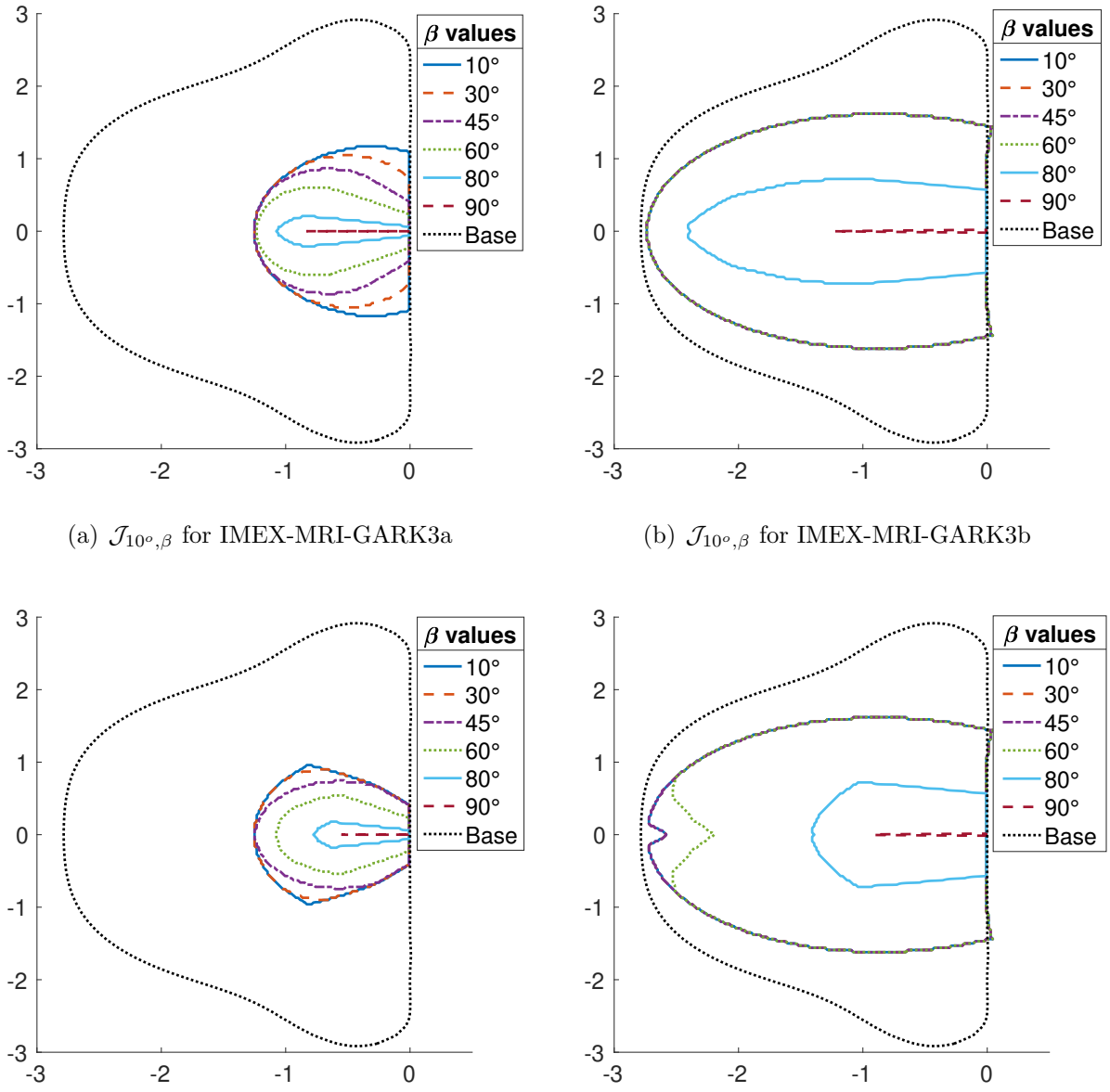
$$\Delta c^{\{S\}T}\left(L + \tfrac{1}{2}E^{-1}\right)A^{\{E,E\}}c^{\{S\}} = \frac{1}{6}, \quad \Delta c^{\{S\}T}\left(L + \tfrac{1}{2}E^{-1}\right)A^{\{I,I\}}c^{\{S\}} = \frac{1}{6}, \tag{3.56}$$

as these are equivalent to the third-order coupling conditions (3.43)*, with equation* (3.14) *providing one-to-one correspondences between* $A^{\{I,I\}}$ *and* $\Gamma^{\{0\}}$*, and between* $A^{\{E,E\}}$ *and* $\Omega^{\{0\}}$*. We note that the conditions* (3.56) *each correspond to the previously-discovered third-order condition for MIS methods introduced in [61].*

In Figure 3.1 we plot the joint stability regions $\mathcal{J}_{\alpha,\beta}$ for both the IMEX-MRI-GARK3a and IMEX-MRI-GARK3b methods, for the fast time scale sectors $\mathcal{S}_{\alpha}^{\{F\}}$, $\alpha \in \{10^o, 45^o\}$ and for the slow implicit sectors $\mathcal{S}_{\beta}^{\{I\}}$, $\beta \in \{10^o, 30^o, 45^o, 60^o, 80^o, 90^o\}$. In these figures we also plot the joint stability region for the slow base IMEX-ARK method, taken using the implicit slow wedge $\mathcal{S}_{90^o}^{\{I\}}$ (black dotted line). These results indicate that the joint stability regions for IMEX-MRI-GARK3a at each fast and implicit sector angle is significantly smaller than the base IMEX-ARK stability region. Furthermore, these stability regions shrink considerably as the implicit sector angle β grows from 10^o to 80^o. In contrast, the joint stability regions for IMEX-MRI-GARK3b are much larger, encompassing the majority of the base IMEX-ARK stability region for both fast sector angles $\alpha = 10^o$ and 45^o, and for implicit sector angles $\beta \leq 60^o$, including a significant extent along the imaginary axis. We therefore anticipate that this method should provide increased stability for IMEX multirate problems wherein advection comprises the slow explicit portion, as the corresponding Jacobian eigenvalues typically reside on the imaginary axis.

3.4.2. Fourth-order method

We also constructed a fourth-order IMEX-MRI-GARK method using a base IMEX-ARK method of our own design (since we knew of no existing fourth-order method that satisfied our 'sorted abscissae' requirement, $0 \leq c_1^{\{S\}} \leq \cdots \leq c_{s^{\{S\}}}^{\{S\}} \leq 1$). To obtain IMEX-MRI-GARK4 we first converted our IMEX-ARK table to solve-decoupled form and then obtained the missing coefficients by satisfying internal consistency of the IMEX-ARK method. We then found the unknowns in $\Gamma^{\{0\}}$, $\Gamma^{\{1\}}$, $\Omega^{\{0\}}$ and $\Omega^{\{1\}}$ by solving the linear system resulting from (3.14), (3.32), (3.43) and (3.44) in MATLAB. The nonzero coefficients $\mathbf{c}^{\{S\}}$, $\Gamma^{\{0\}}$, $\Gamma^{\{1\}}$,



(a) $\mathcal{J}_{10^o,\beta}$ for IMEX-MRI-GARK3a

(b) $\mathcal{J}_{10^o,\beta}$ for IMEX-MRI-GARK3b

(c) $\mathcal{J}_{45^o,\beta}$ for IMEX-MRI-GARK3a

(d) $\mathcal{J}_{45^o,\beta}$ for IMEX-MRI-GARK3b

Figure 3.1: Joint stability regions $\mathcal{J}_{\alpha,\beta}$ for both IMEX-MRI-GARK3a (left) and IMEX-MRI-GARK3b (right), at fast sector angles $\alpha = 10^o$ (top) and $\alpha = 45^o$ (bottom), for a variety of implicit sector angles β. Each plot includes the joint stability region for the base IMEX-ARK table (shown as "Base"). The benefits of simultaneously optimizing the IMEX-MRI-GARK coefficients $\Gamma^{\{0\}}$ and $\Omega^{\{0\}}$ are clear, as $\mathcal{J}_{\alpha,\beta}$ for IMEX-MRI-GARK3b are significantly larger than those for IMEX-MRI-GARK3a.

$\Omega^{\{0\}}$ and $\Omega^{\{1\}}$ for this method, again accurate to 36 decimal digits, are given in Appendix A.3.

While this method indeed satisfies the full set of ARK consistency conditions (3.14), internal consistency conditions (3.32), third-order conditions (3.43), and fourth-order conditions (3.44), we have not yet been successful at optimizing its joint stability region $\mathcal{J}_{\alpha,\beta}$. In fact, even when ignoring the slow-explicit portion by setting $z^{\{E\}} = 0$ in our stability function (3.55), the implicit+fast joint stability region is very small, rendering the full joint stability regions $\mathcal{J}_{\alpha,\beta}$ empty. While we have already noted that this definition of joint stability is overly restrictive, and thus there may indeed be applications in which IMEX-MRI-GARK4 is suitable, we do not promote its widespread use, but include it here to demonstrate the predicted fourth-order convergence in our multirate example problems.

3.5. Numerical results

In this section we demonstrate the expected rates of convergence for the IMEX-MRI-GARK methods from Section 3.4. Additionally, we compare the efficiency of the proposed methods against the legacy Lie–Trotter and Strang–Marchuk splittings (3.3) and (3.4), as well as against two implicit MRI-GARK schemes from [88] of orders 3 and 4, respectively: MRI-GARK-ESDIRK34a and MRI-GARK-ESDIRK46a. We consider two test problems: in Section 3.5.1 we use a small Kværno-Prothero-Robinson (KPR) test problem to demonstrate the convergence of our methods, and in Section 3.5.2 we use a more challenging stiff 'brusselator' test problem to investigate computational efficiency. Computations for the KPR problem were carried out in MATLAB while computations for the brusselator test were carried out in C using infrastructure from ARKODE, an ODE integration package within the SUNDIALS suite which provides explicit, implicit, and IMEX Runge–Kutta methods as well as MRI-GARK methods [34]. MATLAB implementations of both test problems are available in the public GitHub repository [16].

3.5.1. Kværno-Prothero-Robinson (KPR) Test

We first consider the KPR test problem adapted from Sandu [88],

$$\begin{bmatrix} u \\ v \end{bmatrix}' = \mathbf{\Lambda} \begin{bmatrix} \frac{-3+u^2-\cos(\beta t)}{2u} \\ \frac{-2+v^2-\cos(t)}{2v} \end{bmatrix} - \begin{bmatrix} \frac{\beta \sin(\beta t)}{2u} \\ \frac{\sin(t)}{2v} \end{bmatrix}, \quad t \in \left[0, \tfrac{5\pi}{2}\right],$$

where

$$\mathbf{\Lambda} = \begin{bmatrix} \lambda^{\{F\}} & \frac{1-\varepsilon}{\alpha}(\lambda^{\{F\}} - \lambda^{\{S\}}) \\ -\alpha\varepsilon(\lambda^{\{F\}} - \lambda^{\{S\}}) & \lambda^{\{S\}} \end{bmatrix},$$

and with initial conditions $u(0) = 2$, $v(0) = \sqrt{3}$, corresponding to the exact solutions $u(t) = \sqrt{3 + \cos(\beta t)}$ and $v(t) = \sqrt{2 + \cos(t)}$. Here, u and v correspond to the "fast" and "slow" solution variables, respectively. We use the parameters $\lambda^{\{F\}} = -10$, $\lambda^{\{S\}} = -1$, $\varepsilon = 0.1$, $\alpha = 1$, $\beta = 20$. While this problem does not inherently require IMEX methods at the slow time scale, it is both nonlinear and non-autonomous, and has an analytical solution. Thus this serves as an excellent problem to assess the convergence rates for the proposed IMEX-MRI-GARK methods.

We split this problem into the form (3.2) by setting each portion of the right hand side to be

$$f^{\{E\}} = \begin{bmatrix} 0 \\ \frac{\sin(t)}{2v} \end{bmatrix}, \qquad f^{\{I\}} = \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix} \mathbf{\Lambda} \begin{bmatrix} \frac{-3+u^2-\cos(\beta t)}{2u} \\ \frac{-2+v^2-\cos(t)}{2v} \end{bmatrix}, \qquad \text{and}$$

$$f^{\{F\}} = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} \mathbf{\Lambda} \begin{bmatrix} \frac{-3+u^2-\cos\beta t}{2u} \\ \frac{-2+v^2-\cos(t)}{2v} \end{bmatrix} - \begin{bmatrix} \frac{\beta \sin(\beta t)}{2u} \\ 0 \end{bmatrix}.$$

The slow component for implicit MRI-GARK methods is the sum of $f^{\{E\}}$ and $f^{\{I\}}$ which is then treated implicitly.

For the fast time scale of each method we use a step of size $h = \frac{H}{20}$, where we match the order of the inner solver with the overall method order: IMEX-MRI-GARK3 (a, b) and MRI-GARK-ESDIRK34a use the third-order explicit "RK32" from equation (233f) of [10], IMEX-MRI-GARK4 and MRI-GARK-ESDIRK46a use the popular fourth-order explicit "RK4" method from [62], Strang–Marchuk uses the second-order explicit Heun method, and Lie–Trotter uses the explicit forward Euler method. For the implicit slow components of each method we use a standard Newton-Raphson nonlinear solver with dense Jacobian matrix and linear solver.

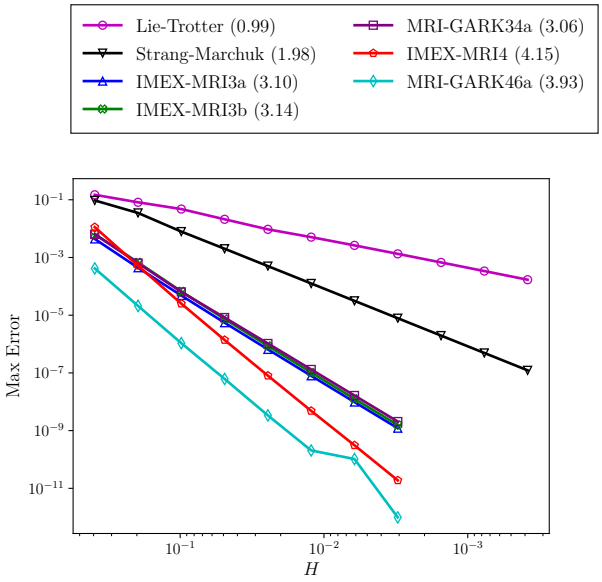


Figure 3.2: Convergence for the KPR test problem from Section 3.5.1. The measured convergence rates (given in parentheses) for each method match their theoretical predictions.

In Figure 3.2 we plot the maximum solution error over a set of 20 evenly-spaced temporal outputs in $[0, 5\pi/2]$ for each method, at each of the slow step sizes $H = \pi/2^k$, for $k = 3, \ldots, 10$ with IMEX-MRI-GARK and MRI-GARK methods and $k = 3, \ldots, 13$ for the legacy methods. In the legend parentheses we show the overall estimated convergence rate, computed using a

least-squares best fit of the log(Max Error) versus $\log(H)$ results for each method. For each method the theoretical order of convergence is reproduced.

3.5.2. Brusselator Test

Our second, and more strenuous, test problem focuses on an advection-diffusion-reaction system of partial differential equations, as these are pervasive in computational physics and are typically solved using one of the two legacy methods (3.3) or (3.4). Here, both advection and diffusion may be evolved at the slow time scale, but due to their differential structure advection is typically treated explicitly, while diffusion is implicit. Chemical reactions, however, frequently evolve on much faster time scales than advection and diffusion, and due to their nonlinearity and bound constraints (typically these are mass densities that must be non-negative), often require subcycling for both accuracy and stability.

We therefore consider the following example which is a stiff variation of the standard "brusselator" test problem [45, 46]:

$$u_t = \alpha_u \nabla^2 u + \rho_u \nabla u + a - (w+1)u + u^2 v,$$

$$v_t = \alpha_v \nabla^2 v + \rho_v \nabla v + wu - u^2 v,$$

$$w_t = \alpha_w \nabla^2 w + \rho_w \nabla w + \frac{b-w}{\varepsilon} - wu,$$

solved on $t \in [0, 3]$ and $x \in [0, 1]$, using stationary boundary conditions,

$$u_t(t,0) = u_t(t,1) = v_t(t,0) = v_t(t,1) = w_t(t,0) = w_t(t,1) = 0,$$

and initial values,

$$u(0,x) = a + 0.1\sin(\pi x),$$

$$v(0,x) = b/a + 0.1\sin(\pi x),$$

$$w(0,x) = b + 0.1\sin(\pi x),$$

with parameters $\alpha_j = 10^{-2}$, $\rho_j = 10^{-3}$, $a = 0.6$, $b = 2$, and $\varepsilon = 10^{-2}$. We discretize these in space using a second-order accurate centered difference approximation with 201 or 801 grid points. As we do not have an analytical solution to this problem, we compute error by comparing against a reference solution generated using the same spatial grid, but that uses ARKODE's default fifth-order diagonally implicit method with a time step of $H = 10^{-7}$.

We split this problem into the form (3.2) by setting each portion of the right hand side to be the spatially-discretized versions of the operators

$$f^{\{E\}} = \begin{bmatrix} \rho_u \nabla u \\ \rho_v \nabla v \\ \rho_w \nabla w \end{bmatrix}, \quad f^{\{I\}} = \begin{bmatrix} \alpha_u \nabla^2 u \\ \alpha_v \nabla^2 v \\ \alpha_w \nabla^2 w \end{bmatrix}, \quad \text{and} \quad f^{\{F\}} = \begin{bmatrix} a - (w+1)u + u^2 v \\ wu - u^2 v \\ \frac{b-w}{\varepsilon} - wu \end{bmatrix}.$$

The slow component for implicit MRI-GARK methods is the sum of $f^{\{E\}}$ and $f^{\{I\}}$ which is then treated implicitly.

We note that although this test problem indeed exhibits the same differential structure as large-scale advection-diffusion-reaction PDE models, a significant majority of those models are based on the compressible Navier–Stokes equations, wherein the 'slow explicit' operator $f^{\{E\}}$ would be nonlinear, would dominate the transport of reactants throughout the domain, and would be treated using a shock-capturing or essentially non-oscillatory spatial discretization. Thus our results which follow should serve as only a simplified test problem for such scenarios, since in reality one would instead expect $f^{\{E\}}$ to require a significantly larger share of the overall computational effort. As a result, our subsequent results show only a 'best case' scenario for implicit MRI-GARK methods, as implicit treatment of $f^{\{E\}}$

in such large-scale applications is typically avoided due to its extreme cost and potential for nonlinear solver convergence issues.

For the subcycling portions of each method, we use a fast time step of $h = H/5$. With the exception of Lie–Trotter we use fast implicit methods having accuracy equal to their corresponding multirate method: IMEX-MRI-GARK3 (a, b) and MRI-GARK-ESDIRK34a use the diagonally-implicit method from Section 3.2.3 of [18] with $\beta = (3 + \sqrt{3})/6$, IMEX-MRI-GARK4 and MRI-GARK-ESDIRK46a use the diagonally-implicit (5,3,4) method from [13], while Lie-Trotter and Strang–Marchuk use an implicit second-order method given by the Butcher table $\begin{array}{c|cc} 1 & 1 & 0 \\ 0 & -1 & 1 \\ \hline & 1/2 & 1/2 \end{array}$. For both the implicit slow stages and the implicit fast stages we use a standard Newton-Raphson nonlinear solver with a banded direct linear solver.

For each spatial grid size in Figure 3.3, we plot the runtimes and maximum solution error over a set of 10 evenly-spaced temporal outputs in $[0, 3]$ for each method, at each of the slow step sizes $H = 0.1 \cdot 2^{-k}$ for $k = 0, \ldots, 10$. We compute least squares fit convergence rates only on points within the asymptotic convergence regime, discarding points at larger H values with higher than expected errors and points at smaller H values where errors have already reached our reference solution accuracy. We first note that as expected when applying Runge–Kutta methods to stiff applications, the measured convergence rates are slightly deteriorated from their theoretical peaks. In addition to the challenges presented by stiffness, the reduced convergence for IMEX-MRI-GARK4 and MRI-GARK-ESDIRK46a is likely due to the limited reference solution accuracy of around 10^{-11}. Additionally, the higher-order methods experience order reduction when we increase the spatial grid size from 201 points to 801 points.

Furthermore, we point out that this problem highlights the reduced joint stability region for both the IMEX-MRI-GARK4 and MRI-GARK-ESDIRK46a methods, as the IMEX

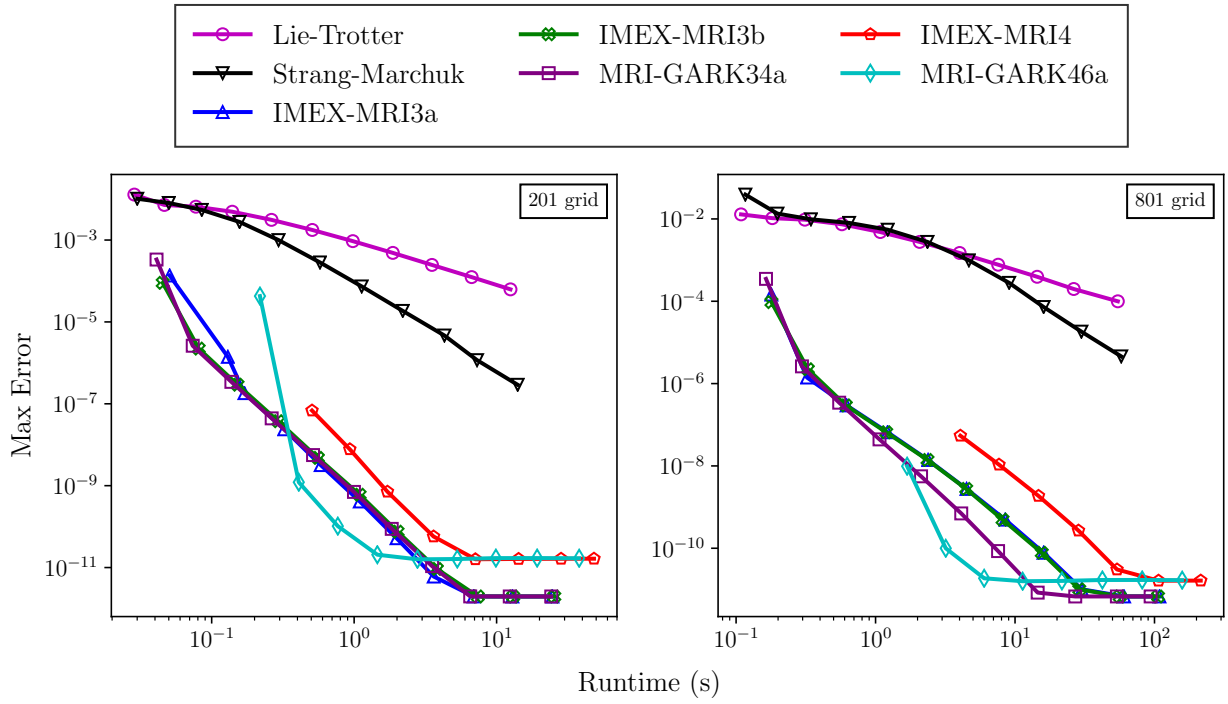


Figure 3.3: Efficiency for the stiff brusselator test problem from Section 3.5.2, using 201 grid points (left) and 801 grid points (right). Best fit convergence rates on the 201 grid are $0.91, 1.92, 2.86, 2.92, 2.94, 3.12, 2.94$ for (Lie–Trotter, Strang–Marchuk, IMEX-MRI3a, IMEX-MRI3b, MRI-GARK34a, IMEX-MRI4, and MRI-GARK46a, resp.) and $0.90, 1.87, 2.41, 2.47, 3.02, 2.69, 2.42$ for the 801 grid. MRI-GARK46a and IMEX-MRI4 have limited stability on this test problem, with their curves missing for $H > 1/40$ and $H > 1/80$ respectively on the 201 grid, and $H > 1/80$ and $H > 1/160$ on the 801 grid.

method was unstable for time step sizes larger than $H = 1/80$ for 201 spatial grid points and larger than $H = 1/160$ for the 801 spatial grid, while the implicit method was unstable for step sizes larger than $H = 1/40$ and $H = 1/80$ for 201 and 801 spatial grids respectively. All of the other methods were stable (if inaccurate) at even the largest step sizes tested.

Focusing our discussion on efficiency, at all accuracy levels shown in Figure 3.3, IMEX-MRI-GARK and implicit MRI-GARK schemes are more efficient for this application than legacy approaches. This is hardly surprising, due to their increased convergence rates and tighter coupling between the operators at the fast and slow time scales. Comparing the third and fourth-order IMEX-MRI-GARK methods, the third-order methods are clearly more efficient for this test, which we believe results from three primary factors. First, the third-order methods require fewer slow-implicit solves per step (3 vs 5). Second, the fast-scale implicit Runge–Kutta methods used for both schemes have significantly different costs, with the third and fourth-order fast methods requiring 2 and 5 implicit stages per step, respectively. Both of these cost differences should be expected due to their differing method order; however the IMEX-MRI-GARK4 also experienced more severe order reduction for this problem, precluding those increased costs from being balanced by a significantly higher achievable convergence rate.

Expanding our consideration to include the full range of higher-order MRI-GARK approaches, MRI-GARK-ESDIRK46a is the most efficient at achieving tight desired accuracies (below 10^{-8}), while all of the third-order methods were comparably efficient for larger accuracy levels. For the 201 grid, there is no discernible difference in runtime between our IMEX-MRI-GARK3 a/b methods and MRI-GARK-ESDIRK34a; however MRI-GARK-ESDIRK34a achieves better efficiency for the 801 grid. We recall, however, that due to the simple linear advection model in this problem, the results shown here represent a *best case* scenario for implicit MRI-GARK methods, whereas the IMEX-MRI-GARK results should more accurately reflect their expected performance on large-scale reactive flow problems. We thus anticipate that when applied to the targeted large-scale applications, the IMEX-MRI-

GARK3 a/b methods will prove to be significantly more efficient, due to their combination of excellent convergence and flexibility in allowing explicit treatment of $f^{\{E\}}$.

We finally note that of the methods that allow the originally-desired IMEX + multirate treatment of this problem (i.e., *not* including the implicit MRI-GARK methods), the proposed IMEX-MRI-GARK methods enable accuracies that would otherwise be intractable with legacy splitting approaches.

3.6. Conclusions

In this paper we have introduced a new class of multirate integration methods that support implicit-explicit treatment of the slow time scale. These IMEX-MRI-GARK methods are highly-flexible: in addition to supporting IMEX treatment of the slow time scale, the fast time scale is only assumed to be solved using another sufficiently-accurate approximation, thereby allowing for the fast time scale to be further decomposed into a mix of implicit and explicit components, or even into a multirate method itself. As with their related non-IMEX MRI-GARK counterparts [88], the coupling from slow to fast time scale occurs through modification of the fast time-scale function $f^{\{F\}}(t,y)$ to include a polynomial forcing term, $g(t)$, that incorporates slow time scale tendencies into the fast time scale dynamics.

In addition to defining IMEX-MRI-GARK methods, we have provided rigorous derivation of conditions on their coefficients to guarantee orders three and four. Furthermore, we have provided the corresponding linear stability function for IMEX-MRI-GARK methods, and extended the definition of "joint stability" from Zharovsky et al. [111] to accommodate a three-component additive splitting.

With these theoretical foundations, we have presented three specific IMEX-MRI-GARK methods, two third-order methods derived from Ascher, Ruuth and Spiteri's '(3,4,3)' ARK method [4], and one fourth-order method of our own design.

We then provided asymptotic convergence results for the three proposed methods, using the standard Kværno-Prothero-Robinson (KPR) multirate test problem, where each method

exhibited its expected convergence rate. To assess method efficiency, we utilized a more challenging stiff brusselator PDE test problem, which showed that the proposed methods were uniformly more efficient than the legacy Lie–Trotter and Strang–Marchuk methods at all accuracy levels tested. Moreover, although such methods cannot allow for IMEX treatment of the slow time scale (and thus efficiency comparisons are somewhat artificial), we also compared against recently-proposed implicit MRI-GARK methods [88]. Here, our third-order IMEX-MRI-GARK methods proved competitive, and the higher cost per step of our fourth-order IMEX-MRI-GARK method rendered it the least efficient of the group.

We note that much work remains. For starters, we plan to derive a new fourth-order IMEX-MRI-GARK method with an optimal linear stability region and with a decreased cost per step. We anticipate that this will require simultaneous derivation of both the base IMEX-ARK method *and* its IMEX-MRI-GARK extension, due to the tight interplay between these methods and their joint stability. An obvious (yet tedious) extension of this work would be to derive the order conditions for fifth-order IMEX-MRI-GARK methods, and to construct tables to implement such approaches. Additionally, we would like to create new IMEX-MRI-GARK methods that include embeddings, thereby allowing for robust temporal adaptivity at both the slow and fast time scales. While extension of the IMEX-MRI-GARK algorithm to include an alternate set of IMEX-ARK embedding coefficients is straightforward, creation of optimal embedded multirate methods and fast/slow temporal adaptivity controllers have barely been touched in the literature. Finally, we anticipate the creation of 'solve-coupled' IMEX-MRI-GARK and MRI-GARK methods, and the accompanying work on efficient nonlinear solvers, to allow a tighter coupling between implicit and fast processes in these multirate approaches.

Chapter 4

Stability optimized fourth order methods

In this chapter we further investigate linear stability of IMEX-MRI-GARK methods with the intention of constructing a fourth-order method with better stability properties compared to the one introduced in Chapter 3. We achieve this task by relaxing our definition of joint stability from Chapter 3, after which we create a new stability optimized fourth-order method. We plot stability regions for our IMEX-MRI-GARK methods under the relaxed definition of joint stability. Numerical experiments on the brusselator test problem from Chapter 3 demonstrate significant improvement in stability for the new fourth order method.

4.1. Relaxed definition of joint stability

Our discussion is a continuation of Section 3.3 in Chapter 3. Starting with the expression of the stability function

$$\begin{aligned} & R\big(z^{\{F\}}, z^{\{E\}}, z^{\{I\}}\big) \qquad (4.1)\\ & := e_{s\{S\}}^T \left(I - \operatorname{diag}\Big(\varphi_0\big(\Delta c^{\{S\}} z^{\{F\}}\big)\Big) L - z^{\{E\}} \eta\big(z^{\{F\}}\big) - z^{\{I\}} \mu\big(z^{\{F\}}\big) \right)^{-1} e_1. \end{aligned}$$

we defined the joint stability region following Zharovsky and collaborators in [111], to be

$$\mathcal{J}_{\alpha,\beta} := \left\{ z^{\{E\}} \in \mathbb{C}^- \ : \ \left| R(z^{\{F\}}, z^{\{E\}}, z^{\{I\}}) \right| \leq 1, \quad \forall z^{\{F\}} \in \mathcal{S}_\alpha^{\{F\}}, \quad \forall z^{\{I\}} \in \mathcal{S}_\beta^{\{I\}} \right\} \qquad (4.2)$$

where $\mathcal{S}_\alpha^\sigma := \{z^\sigma \in \mathbb{C}^- \ : \ |\arg(z^\sigma) - \pi| \leq \alpha\}$. This gives the stability region for the slow-explicit component only, assuming $z^{\{I\}}$ and $z^{\{F\}}$ can range throughout entire infinitely-long

sectors $\mathcal{S}_\alpha^{\{I\}}$ and $\mathcal{S}_\beta^{\{F\}}$ in $\mathbb{C}^-$. Under this definition of joint stability, we can plot regions for our third order methods IMEX-MRI-GARK3a and IMEX-MRI-GARK3b. However, this requirement of stability over infinitely-long sectors resulted in our IMEX-MRI-GARK4 method having no region of joint stability, making it difficult for optimization utilities to improve.

Considering both $z^{\{I\}}$ and $z^{\{F\}}$ to be infinitely-long sectors in the left complex plane is rather restrictive. For the purposes of maximizing the joint stability region, we can loosen this restriction by considering $z^{\{F\}}$ to be in a smaller sector, akin to assuming the fast time scale to be non-stiff. We therefore refine our definition of the joint stability region (4.2) to be

$$\mathcal{J}_{\alpha,\beta}^{\rho} \coloneqq \left\{ z^{\{E\}} \in \mathbb{C}^- \ : \ \left| R(z^{\{F\}}, z^{\{E\}}, z^{\{I\}}) \right| \leq 1, \quad \forall z^{\{F\}} \in \mathcal{S}_{\rho,\alpha}^{\{F\}}, \quad \forall z^{\{I\}} \in \mathcal{S}_\beta^{\{I\}} \right\} \tag{4.3}$$

with $\mathcal{S}_\beta^{\{I\}}$ defined as before and

$$\mathcal{S}_{\rho,\alpha}^{\{F\}} \coloneqq \left\{ z^{\{F\}} \in \mathbb{C}^- \ : \ |z^{\{F\}}| \leq \rho, \quad |\arg(z^{\{F\}}) - \pi| \leq \alpha \right\} \tag{4.4}$$

We note that this in fact matches Sandu's definitions of scalar stability regions in [88]. Following his lead, we thus consider the fast sector to have extent $\rho = 1$ in (4.4), which allows us to create a new fourth order method that demonstrates better stability properties.

4.2. Fourth-order method with improved joint stability

We create a fourth-order IMEX-MRI-GARK method based on an IMEX-ARK of our design starting with a 6-stage stiffly accurate ESDIRK method $\left(A^{\{I\}}, b^{\{I\}}, c^{\{I\}}\right)$ from Kennedy and Carpenter [56],

0	0	0	0	0	0	0
2γ	γ	γ	0	0	0	0
$c_3^{\{S\}}$	$(c_3^{\{S\}} - a_{32}^{\{I\}} - \gamma)$	$a_{32}^{\{I\}}$	γ	0	0	0
$c_4^{\{S\}}$	$(c_4^{\{S\}} - a_{42}^{\{I\}} - a_{43}^{\{I\}} - \gamma)$	$a_{42}^{\{I\}}$	$a_{43}^{\{I\}}$	γ	0	0
$c_5^{\{S\}}$	$(c_5^{\{S\}} - a_{52}^{\{I\}} - a_{53}^{\{I\}} - a_{54}^{\{I\}} - \gamma)$	$a_{52}^{\{I\}}$	$a_{53}^{\{I\}}$	$a_{54}^{\{I\}}$	γ	0
1	$(1 - b_2^{\{S\}} - b_3^{\{S\}} - b_4^{\{S\}} - b_5^{\{S\}} - \gamma)$	$b_2^{\{S\}}$	$b_3^{\{S\}}$	$b_4^{\{S\}}$	$b_5^{\{S\}}$	γ
1	$(1 - b_2^{\{S\}} - b_3^{\{S\}} - b_4^{\{S\}} - b_5^{\{S\}} - \gamma)$	$b_2^{\{S\}}$	$b_3^{\{S\}}$	$b_4^{\{S\}}$	$b_5^{\{S\}}$	γ

and build a corresponding explicit Runge–Kutta method $\left(A^{\{E\}}, b^{\{E\}}, c^{\{E\}}\right)$ with $c^{\{I\}} = c^{\{E\}} \equiv c^{\{S\}}$ and $b^{\{I\}} = b^{\{E\}} \equiv b^{\{S\}}$. We note that this is the same table we started with when constructing our fourth-order IMEX-MRI-GARK method in Chapter 3. Leaving the derivation of embedding coefficients for future work, there are 14 degrees of freedom.

We use symbolic MATLAB to find the free variables in the following several steps:
To satisfy L-stability for the ESDIRK table, we choose $\gamma = \frac{1}{4}$, and we set the remaining $c^{\{S\}}$ coefficients so that they are nondecreasing and equidistant from each other. Next, we choose $b_2^{\{S\}}, b_3^{\{S\}}, b_4^{\{S\}}$ to satisfy the conditions

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

For the ESDIRK table we solve

$$b^{\{S\}T} A^{\{I\}} c^{\{S\}} = \frac{1}{6}, \qquad \left(b^{\{S\}} \times c^{\{S\}}\right)^T A^{\{I\}} c^{\{S\}} = \frac{1}{8},$$

$$b^{\{S\}T} A^{\{I\}} c^{\{S\}\times 2} = \frac{1}{12}, \qquad b^{\{S\}T} A^{\{I\}} A^{\{I\}} c^{\{S\}} = \frac{1}{24},$$

to obtain $a_{43}^{\{I\}}, a_{52}^{\{I\}}, a_{53}^{\{I\}}, a_{54}^{\{I\}}$. We then satisfy $b_5^{\{S\}}, a_{42}^{\{E\}}, a_{52}^{\{E\}}, a_{62}^{\{E\}}$ from conditions

$$b^{\{S\}T} A^{\{E\}} c^{\{S\}} = \frac{1}{6}, \qquad \left(b^{\{S\}} \times c^{\{S\}}\right)^T A^{\{E\}} c^{\{S\}} = \frac{1}{8},$$

$$b^{\{S\}T} A^{\{E\}} c^{\{S\}\times 2} = \frac{1}{12}, \qquad b^{\{S\}T} A^{\{E\}} A^{\{E\}} c^{\{S\}} = \frac{1}{24}.$$

The coupling conditions

$$b^{\{S\}T} A^{\{I\}} A^{\{E\}} c^{\{S\}} = \frac{1}{24}, \qquad b^{\{S\}T} A^{\{E\}} A^{\{I\}} c^{\{S\}} = \frac{1}{24},$$

lead to values for $a_{63}^{\{E\}}, a_{64}^{\{E\}}$. Finally $a_{42}^{\{I\}}$ is chosen to satisfy L-stability.

Upon solving each of the above conditions using the stated variables, we are guaranteed that the IMEX-ARK method satisfies fourth-order conditions, and we are left with 6 free variables: $a_{32}^{\{I\}}, a_{32}^{\{E\}}, a_{43}^{\{E\}}, a_{53}^{\{E\}}, a_{54}^{\{E\}}$, and $a_{65}^{\{E\}}$. We then transform the resulting IMEX-ARK table based on these 6 free variables into solve-decoupled form as described in Section 3.4. The new variables introduced by rewriting IMEX-ARK tables in solve-decoupled form are inconsequential to the coefficients in the Γ^k and Ω^k matrices, as they can be uniquely determined by enforcing internal consistency of the IMEX-ARK table. This makes it so that all of the IMEX-MRI-GARK order conditions that remain to be satisfied are linear in the unknown variables from $\Gamma^0, \Gamma^1, \Omega^0, \Omega^1$ from the IMEX-MRI-GARK table, and our remaining free variables $a_{32}^{\{I\}}, a_{32}^{\{E\}}, a_{43}^{\{E\}}, a_{53}^{\{E\}}, a_{54}^{\{E\}}$ and $a_{65}^{\{E\}}$ from the IMEX-ARK table. This linear system of equations is under-determined, leaving 51 free variables to modify in an attempt to maximize the joint stability region. We choose our objective function to be the largest real-valued $z^{\{E\}}$ that is on the boundary of the joint stability region, i.e. given guesses of the free variables, we solve the root-finding problem

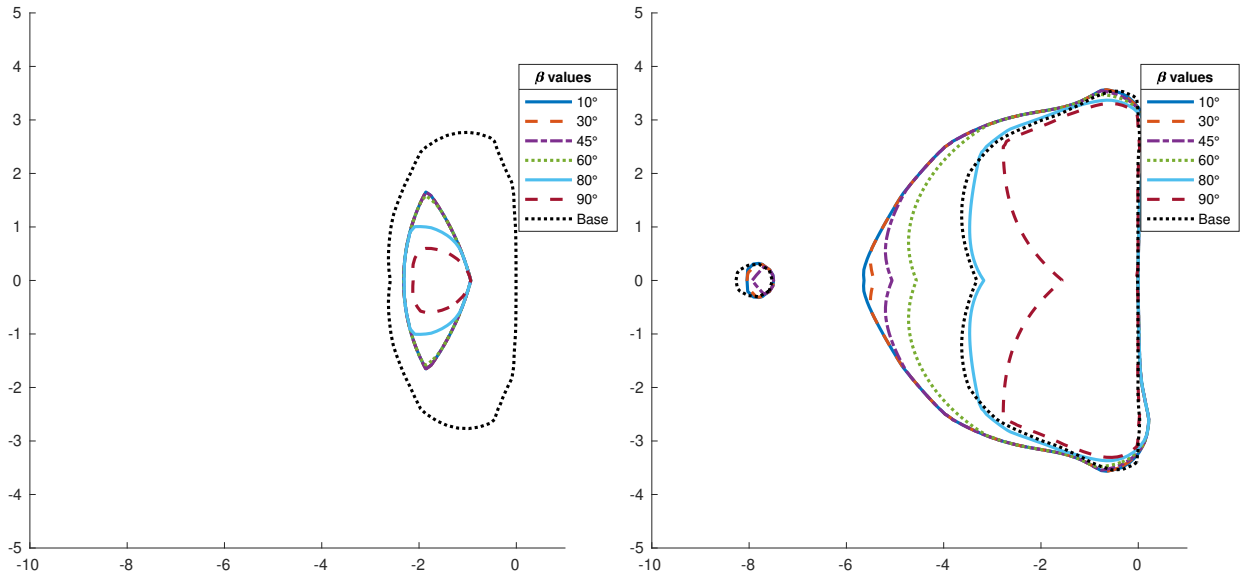
$$f\left(z^{\{E\}}\right) = 1 - \max_{\substack{z^{\{F\}}\in(-1,0),\\ z^{\{I\}}\in(-10^8,0)}} \left|R\left(z^{\{F\}}, z^{\{I\}}, z^{\{E\}}\right)\right| = 0$$

with the bisection method, and return $z^{\{E\}}$ as the value of the objective function. We use MATLAB's `fmincon` algorithm to carry out the optimization process of minimizing this objective function. We name the resulting fourth-order IMEX-MRI-GARK method to be IMEX-MRI-GARK4s, and its coefficients are provided in Appendix A.4.

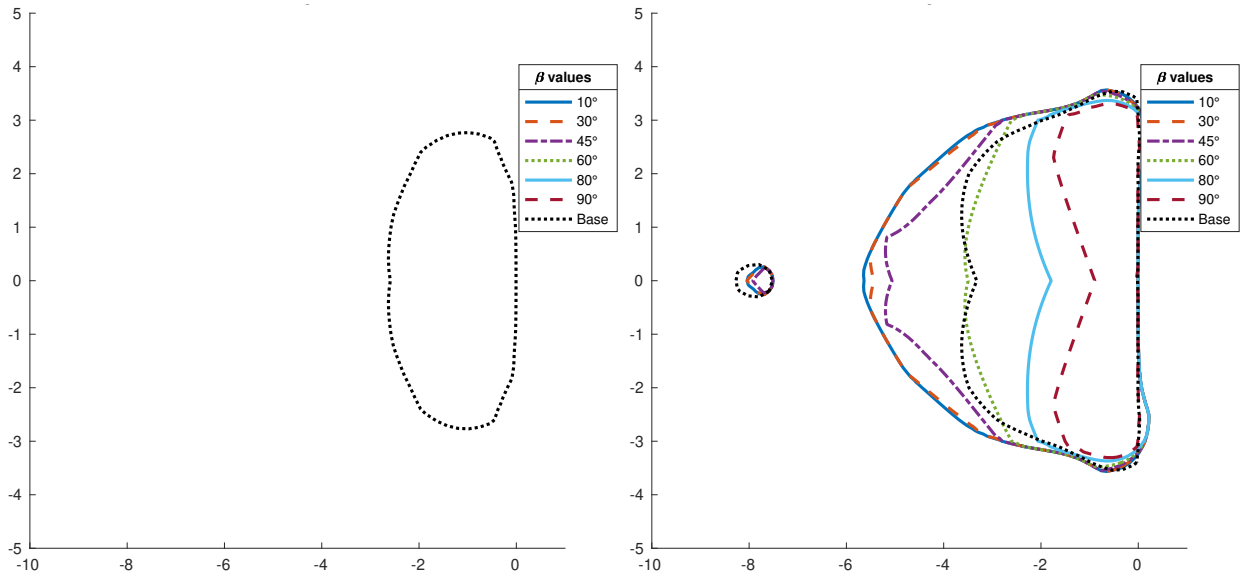
Figure 4.1 shows the joint stability regions $\mathcal{J}^1_{\alpha,\beta}$ for the IMEX-MRI-GARK4 method from Section 3.4 in Chapter 3 and IMEX-MRI-GARK4s for fast wedges $\mathcal{S}^{\{F\}}_{\rho=1,\alpha}$ with $\alpha \in \{10°, 45°\}$ and for slow implicit wedges $\mathcal{S}^{\{I\}}_{\beta}$ with $\beta \in \{10°, 30°, 45°, 60°, 80°, 90°\}$. Included in our plots is the joint stability region for IMEX-ARK given the slow implicit wedge $\mathcal{S}^{\{I\}}_{90°}$ shown by the black dotted line. We note that for IMEX-MRI-GARK4, $\mathcal{J}^1_{\alpha,\beta}$ deteriorates with increasing α and β from the base. In fact, there is no joint stability region for sectors with fast $\alpha = 45°$ at all β values. For IMEX-MRI-GARK4s, the joint stability region is broken into two, a larger region closer to the imaginary axis and a much smaller region centered around -8 on the real-axis. Having disjointed stability regions is not ideal since a numerical method can potentially be stable in two independent regions, with a region of instability between them, making it hard to capture the threshold values for stable time steps. For the regions centered around -8, the base IMEX-ARK joint stability region is the largest and IMEX-MRI-GARK4s regions shrink with increasing α and β. However for the regions closest to the imaginary axis, the base IMEX-ARK joint stability region is smaller than IMEX-MRI-GARK4s regions with α-β pairs: 10°-60° and 45°-45°. Overall, the benefits of optimizing are still clear as IMEX-MRI-GARK4s has larger $\mathcal{J}^1_{\alpha,\beta}$ than IMEX-MRI-GARK4 at all values of α and β. Additionally, we note that IMEX-MRI-GARK4s includes significantly more of the imaginary axis within $\mathcal{J}^1_{\alpha,\beta}$.

4.3. Numerical results

In this section we extend our results of the brusselator test problem of Section 3.5.2 from Chapter 3 which introduces IMEX-MRI-GARK methods. Everything from that problem remains the same including comparisons with legacy approaches and MRI-GARK methods,



(a) $\mathcal{J}^1_{10^o,\beta}$ for IMEX-MRI-GARK4 (b) $\mathcal{J}^1_{10^o,\beta}$ for IMEX-MRI-GARK4s



(c) $\mathcal{J}^1_{45^o,\beta}$ for IMEX-MRI-GARK4 (d) $\mathcal{J}^1_{45^o,\beta}$ for IMEX-MRI-GARK4s

Figure 4.1: Joint stability regions $\mathcal{J}^1_{\alpha,\beta}$ for both IMEX-MRI-GARK4 (left) and IMEX-MRI-GARK4s (right), at fast sector angles $\alpha = 10^o$ (top) and $\alpha = 45^o$ (bottom), for a variety of implicit sector angles β. Each plot includes the joint stability region for the base IMEX-ARK table (shown as "Base"). Stability optimized IMEX-MRI-GARK4s has larger $\mathcal{J}^1_{\alpha,\beta}$ for all α and β.

we only add results for IMEX-MRI-GARK4s and present error versus H plots. IMEX-MRI-GARK4s is shortened to IMEX-MRI4s in the plot legends.

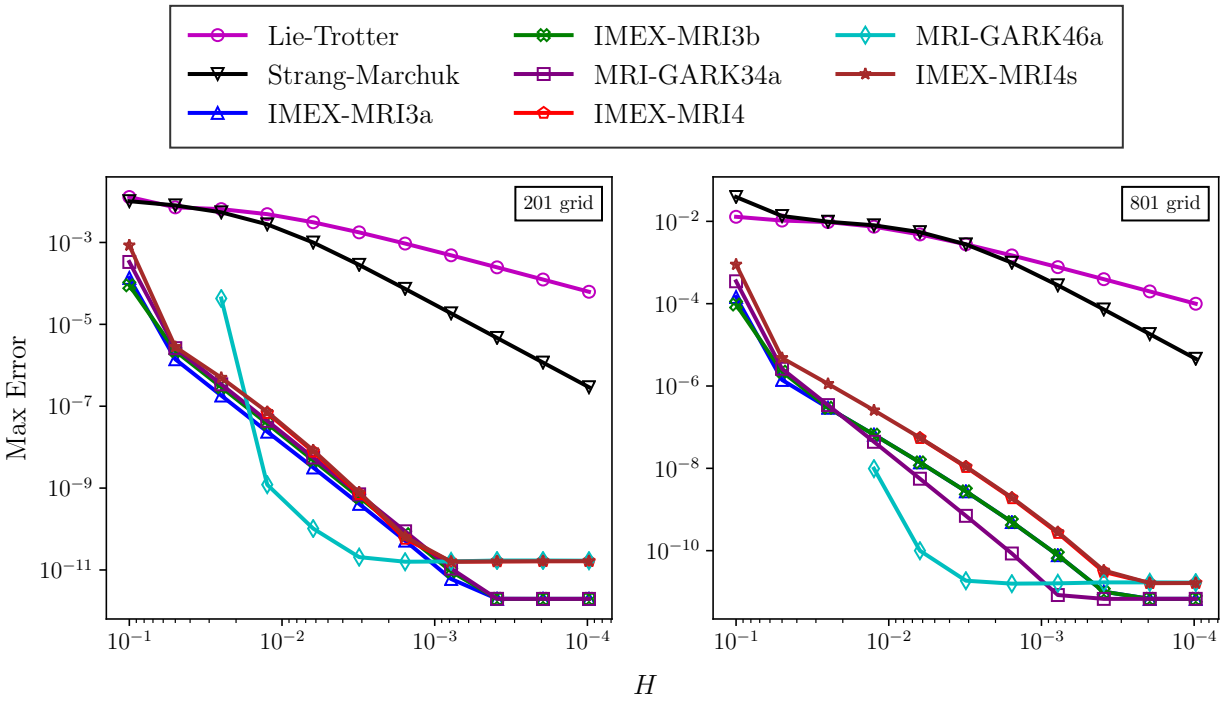


Figure 4.2: Convergence of methods applied to the brusselator problem of Chapter 3, Section 3.5.2. MRI-GARK46a and IMEX-MRI4 have limited stability on this test problem, their curves are missing for $H > \frac{1}{40}$ and $H > \frac{1}{80}$ respectively on the 201 grid, and $H > \frac{1}{80}$ and $H > \frac{1}{160}$ on the 801 grid. The stability optimized IMEX-MRI4s and the rest of the methods are stable at largest value of H tested.

In Figure 4.2 we plot the convergence results for legacy approaches Lie-Trotter and Strang–Marchuk, IMEX-MRI-GARK3a/3b, IMEX-MRI-GARK4, MRI-GARK-ESDIRK34a/46a, and the new IMEX-MRI-GARK4s method. We extend our discussion of the brusselator test problem from Section 3.5.2, focusing on IMEX-MRI-GARK4s. We note that IMEX-MRI-GARK4s is stable for the largest slow time step $H = \frac{1}{10}$ on both the 201 and 801 grid; in contrast to IMEX-MRI-GARK4 which is only stable for $H > \frac{1}{80}$ and $H > \frac{1}{160}$ on the 201 and 801 grids respectively. We also point out that the new IMEX-MRI-GARK4s is even more stable than the MRI-GARK-ESDIRK46a method by Sandu at large step sizes, which is even more notable due to the fact that MRI-GARK-ESDIRK46a treats the entire slow

time scale implicitly. These results confirm the the larger joint stability region we observed for IMEX-MRI-GARK4s. The accuracy of IMEX-MRI-GARK4s is however equivalent to IMEX-MRI-GARK4 for the step sizes that IMEX-MRI-GARK4 is stable for.

Despite being rather simplistic, there are some merits to studying the scalar test problem (3.53) from Chapter 3 and plotting joint stability plots for IMEX-MRI-GARK methods. Limiting our discussion to the refined joint stability in this chapter, IMEX-MRI-GARK4 has a smaller joint stability region than the stability optimized IMEX-MRI-GARK4s. This seemingly predicts the behavior of both methods on the brusselator test problem. Ideally, we need fourth-order methods with decent joint stability for infinite slow implicit and fast wedges, however, as demonstrated here, relaxed definitions for the joint stability can be a useful tool in constructing methods with better stability properties.

Chapter 5

Multirate exponential Runge–Kutta methods

The contents of this chapter have been published in SIAM J. Sci. Comput., 42(2), pp. A1245–A1268 under the title "A new class of high-order methods for multirate differential equations" in collaboration with Vu Thai Luan and Daniel R. Reynolds [68].

5.1. Introduction

In this paper, we focus on the construction, analysis, and implementation of efficient, highly accurate, multirate time stepping algorithms, based on explicit exponential Runge–Kutta methods. These algorithms may be applied to initial value problems (IVPs) of the form

$$u'(t) = F(t, u(t)) = \mathcal{L}u(t) + \mathcal{N}(t, u(t)), \quad u(t_0) = u_0, \tag{5.1}$$

on the interval $t_0 < t \leq T$, where the vector field $F(t, u(t))$ can be decomposed into a linear part $\mathcal{L}u(t)$ comprising the "fast" time scale, and a nonlinear part $\mathcal{N}(t, u(t))$ comprising the "slow" time scale. Such systems frequently result from so-called "multi-physics" simulations that couple separate physical processes together, or from the spatial semi-discretization of time-dependent partial differential equations (PDEs). Our primary interest in this paper lies in the case where the fast component is much less costly to compute than the slow component, thereby opening the door for methods that evolve each component with different time step sizes – so-called multirate (or multiple time-stepping, MTS) methods. This case is common in practice when using a non-uniform grid for the spatial semi-discretization of PDEs, or in parallel computations where the fast component is comprised of spatially-localized processes but the slow component requires communication across the parallel network.

In recent years, there has been renewed interest in the construction of multirate time integration methods for systems of ODEs. Generally, these efforts have focused on techniques to achieve orders of accuracy greater than two, since second-order methods may be obtained through simple interpolation between time scales. These recent approaches broadly fit into two categories: methods that attain higher-order through deferred correction or extrapolation of low-order methods [7, 19, 23], and methods that directly satisfy order conditions for partitioned and/or additive Runge–Kutta methods [31, 41, 42, 59, 60, 63, 87, 93, 98, 100, 101, 102, 107]. Of these, the latter category promises increased efficiency due to the need to traverse the time interval only once. However, only very recently have methods of this type been constructed that can achieve full fourth-order accuracy [87, 102, 83], and we know of no previous methods having order five or higher.

Among numerical methods that use the same time step for all components of (5.1), exponential integrators have shown great promise in recent years [26, 48, 49, 50, 51, 52, 69, 70, 71, 73, 66, 74]. Most such methods require the approximation of products of matrix functions with vectors, i.e., $\phi(\mathcal{L})\, v$, for $\mathcal{L} \in \mathbb{R}^{d\times d}$ and $v \in \mathbb{R}^d$.

Inspired by recent results on local-time stepping methods for problems related to (5.1) [33, 37, 38, 39], and motivated by the idea in [53, Sect. 5.3] that establishes a multirate procedure for exponential *multistep* methods of Adams-type, here we derive multirate procedures for exponential *one-step* methods. Starting from an s-stage explicit exponential Runge–Kutta (ExpRK) method applied to (5.1), we employ the idea of backward error analysis to define $s-1$ modified differential equations whose exact solutions coincide with the ExpRK internal stages. These modified differential equations may then be evolved using standard ODE solvers at the fast time scale. We name the resulting methods as *Multirate Exponential Runge–Kutta* (MERK) methods.

The ability to construct modified ODEs for each slow ExpRK stage is dependent on the form of the ExpRK method itself, and we identify these restrictions within this manuscript. Using this approach, we derive a general multirate algorithm (Algorithm 1) that can be

interpreted as a particular implementation (without matrix functions) of explicit exponential Rung–Kutta methods. With this algorithm in hand, we perform a rigorous convergence analysis for the proposed MERK methods. We additionally construct MERK schemes with orders of accuracy two through five, based on some well-known ExpRK methods from the literature.

We note that the resulting methods show strong similarities to the *MIS* methods in [59, 60, 98, 100, 101, 107] and the follow-on *RMIS* methods [102] and *MRI-GARK* methods [87, 83], in that the MERK algorithm requires the construction of a set of modified "fast" initial-value problems that must be solved to proceed between slow stages, and where these modifications take the form of polynomials based on "slow" function data. While these approaches indeed result in similar algorithmic structure, (R)MIS and MRI-GARK methods are based on partitioned and generalized-structure additive Runge–Kutta theory [92], and as such their derivation requires satisfaction of many more order conditions than MERK methods, particularly as the desired method order increases, to the end that no MIS method of order greater than three, and no RMIS or MRI-GARK methods of order greater than four, have ever been proposed. Additionally, to obtain an overall order p method, all fast IVPs for (R)MIS and MRI-GARK methods must be solved to order p, whereas the internal stages in MERK methods may use an order $p-1$ solver. Finally, both (R)MIS and the MRI-GARK methods from [87] require sorted abcissae $c_1 \leq c_2 \leq \cdots \leq c_s$, a requirement that is not present for MERK methods or for the `SPC-MRI-GARK` method in [83].

The outline of this paper is as follows: in Section 5.2, we derive the general class of exponential Runge–Kutta methods in a way that facilitates construction of MERK procedures. In Section 5.3, we then derive the general MERK algorithm for exponential Runge–Kutta methods, and provide a rigorous convergence analysis for these schemes. In Section 5.4, we derive specific MERK methods based on existing exponential Runge–Kutta methods. We present a variety of numerical examples in Section 5.5 to illustrate the efficiency of the new MERK schemes with order of accuracy up to five. The main contributions of this paper are

Algorithm 1, convergence analysis for MERK methods (Theorem 5.3.2), and the proposed MERK schemes with order of accuracy up to five.

5.2. Motivation

We begin with a general derivation of exponential Runge–Kutta methods [50, 52], which motivates a multirate procedure for solving (5.1).

5.2.1. Exponential Runge–Kutta methods

When deriving ExpRK methods, it is crucial to represent the exact solution of (5.1) at time $t_{n+1} = t_n + H$ using the variation-of-constants formula,

$$u(t_{n+1}) = u(t_n + H) = \mathrm{e}^{H\mathcal{L}} u(t_n) + \int_0^H \mathrm{e}^{(H-\tau)\mathcal{L}} \mathcal{N}(t_n + \tau, u(t_n + \tau)) \mathrm{d}\tau. \tag{5.2}$$

The integral in (5.2) is then approximated using a quadrature rule having nodes c_i and weights $b_i(H\mathcal{L})$ $(i = 1, \ldots, s)$, which yields

$$u(t_{n+1}) \approx \mathrm{e}^{H\mathcal{L}} u(t_n) + H \sum_{i=1}^{s} b_i(H\mathcal{L}) \mathcal{N}(t_n + c_i H, u(t_n + c_i H)). \tag{5.3}$$

By applying (5.2) (with $c_i H$ in place of H), the unknown intermediate values $u(t_n + c_i H)$ in (5.3) can be represented as

$$u(t_n + c_i H) = \mathrm{e}^{c_i H\mathcal{L}} u(t_n) + \int_0^{c_i H} \mathrm{e}^{(c_i H-\tau)\mathcal{L}} \mathcal{N}(t_n + \tau, u(t_n + \tau)) \mathrm{d}\tau. \tag{5.4}$$

Again, one can use another quadrature rule with the same nodes c_i as before (to avoid the generation of new unknowns) and new weights $a_{ij}(H\mathcal{L})$ to approximate the integrals in (5.4). This gives

$$u(t_n + c_i H) \approx \mathrm{e}^{c_i H\mathcal{L}} u(t_n) + H \sum_{j=1}^{s} a_{ij}(H\mathcal{L}) \mathcal{N}(t_n + c_j H, u(t_n + c_j H)). \tag{5.5}$$

Now, denoting the approximations $u_n \approx u(t_n)$ and $U_{n,i} \approx u(t_n + c_i H)$, then from (5.3) and (5.5) one may obtain the so-called *exponential Runge–Kutta methods*

$$U_{n,i} = \mathrm{e}^{c_i H\mathcal{L}} u_n + H \sum_{j=1}^{s} a_{ij}(H\mathcal{L})\,\mathcal{N}(t_n + c_j H, U_{n,j}), \quad i = 1, \ldots, s, \tag{5.6a}$$

$$u_{n+1} = \mathrm{e}^{H\mathcal{L}} u_n + H \sum_{i=1}^{s} b_i(H\mathcal{L})\,\mathcal{N}(t_n + c_i H, U_{n,i}). \tag{5.6b}$$

The formula (5.6) is considered *explicit* when $a_{ij}(H\mathcal{L}) = 0$ for all $i \leq j$ (thus $c_1 = 0$ and consequentially $U_{n,1} = u_n$). Throughout this paper we restrict our attention to explicit exponential Runge–Kutta methods, which can be reformulated as (see [70, 72]):

$$U_{n,i} = u_n + c_i H \varphi_1(c_i H\mathcal{L}) F(t_n, u_n) + H \sum_{j=2}^{i-1} a_{ij}(H\mathcal{L}) D_{n,j}, \ i = 2, \ldots, s, \tag{5.7a}$$

$$u_{n+1} = u_n + H\varphi_1(H\mathcal{L}) F(t_n, u_n) + H \sum_{i=2}^{s} b_i(H\mathcal{L}) D_{n,i}, \tag{5.7b}$$

where

$$D_{n,i} = \mathcal{N}(t_n + c_i H, U_{n,i}) - \mathcal{N}(t_n, u_n), \qquad i = 2, \ldots, s. \tag{5.8}$$

Here, the coefficients $a_{ij}(H\mathcal{L})$ and $b_i(H\mathcal{L})$ are often linear combinations of the functions $\varphi_k(c_i H\mathcal{L})$ and $\varphi_k(H\mathcal{L})$, respectively, wherein $\varphi_k(z)$ are given by

$$\varphi_k(z) = \int_0^1 \mathrm{e}^{(1-\theta)z} \frac{\theta^{k-1}}{(k-1)!} \mathrm{d}\theta, \quad k \geq 1, \tag{5.9}$$

and satisfy the recurrence relations

$$\varphi_k(z) = \frac{\varphi_{k-1}(z) - \varphi_{k-1}(0)}{z}, \quad \varphi_0(z) = \mathrm{e}^z. \tag{5.10}$$

5.2.2. Adopting the idea of backward error analysis

Motivated by the idea of [53, Sect. 5.3], and recalling the equations (5.2) and (5.4), we note that $u(t_{n+1})$ and $u(t_n + c_i H)$ are the exact solutions of the differential equation

$$v'(\tau) = \mathcal{L}v(\tau) + \mathcal{N}(t_n + \tau, u(t_n + \tau)), \quad v(0) = u(t_n), \tag{5.11}$$

evaluated at $\tau = H$ and $\tau = c_i H$, respectively. In other words, solving (5.11) exactly (by means of using the variation-of-constants formula) on the time intervals $[0, H]$ and $[0, c_i H]$ shows that $v(H) = u(t_{n+1})$ and $v(c_i H) = u(t_n + c_i H)$. Unfortunately, explicit representations of these analytical solutions are generally impossible to find, since $u(t_n)$ and $u(t_n + \tau)$ are unknown values. This observation, however, suggests the use of backward error analysis (see, for instance [44, Chap. IX]).

Given an exponential Runge–Kutta method (5.6), we therefore search for modified differential equations of the form (5.11), such that their exact solutions at $\tau = c_i H$ and $\tau = H$ coincide with the ExpRK approximations $U_{n,i}$ $(i = 2, \ldots, s)$ and u_{n+1}, respectively. We may then approximate solutions to these modified equations to compute our overall approximation of (5.1).

5.3. Multirate exponential Runge–Kutta methods

In this section, we construct a new multirate procedure based on approximation of ExpRK schemes; we call the resulting algorithms *Multirate Exponential Runge-Kutta* (MERK) methods. Following this derivation, we present a rigorous convergence analysis.

5.3.1. Construction of modified differential equations

We begin with the construction of MERK methods, through definition of modified differential equations corresponding with the ExpRK stages $U_{n,i}$ $(i = 2, \ldots, s)$ and solution u_{n+1}.

Theorem 5.3.1. *Assuming that the coefficients $a_{ij}(H\mathcal{L})$ and $b_i(H\mathcal{L})$ of an explicit exponential Runge–Kutta method* (5.7) *may be written as linear combinations*

$$a_{ij}(H\mathcal{L}) = \sum_{k=1}^{\ell_{ij}} \alpha_{ij}^{(k)} \varphi_k(c_i H\mathcal{L}), \quad b_i(H\mathcal{L}) = \sum_{k=1}^{m_i} \beta_i^{(k)} \varphi_k(H\mathcal{L}) \tag{5.12}$$

for some positive integers ℓ_{ij} and m_i, and where the functions $\varphi_k(c_i H\mathcal{L})$ and $\varphi_k(H\mathcal{L})$ are given in (5.9)*, then $U_{n,i}$ and u_{n+1} are the exact solutions of the linear differential equations*

$$v_{n,i}'(\tau) = \mathcal{L} v_{n,i}(\tau) + p_{n,i}(\tau), \qquad v_n(0) = u_n, \qquad i = 2, \dots, s, \tag{5.13a}$$

$$v_n'(\tau) = \mathcal{L} v_n(\tau) + q_n(\tau), \qquad v_n(0) = u_n \tag{5.13b}$$

at the times $\tau = c_i H$ and $\tau = H$, respectively. Here $p_{n,i}(\tau)$ and $q_n(\tau)$ are polynomials in τ given by

$$p_{n,i}(\tau) = \mathcal{N}(t_n, u_n) + \sum_{j=2}^{i-1} \Big(\sum_{k=1}^{\ell_{ij}} \frac{\alpha_{ij}^{(k)}}{c_i^k H^{k-1}(k-1)!} \tau^{k-1} \Big) D_{n,j}, \tag{5.14a}$$

$$q_n(\tau) = \mathcal{N}(t_n, u_n) + \sum_{i=2}^{s} \Big(\sum_{k=1}^{m_i} \frac{\beta_i^{(k)}}{H^{k-1}(k-1)!} \tau^{k-1} \Big) D_{n,i}. \tag{5.14b}$$

Proof. By changing the integration variable to $\tau = H\theta$ in (5.9), we obtain

$$\varphi_k(z) = \frac{1}{H^k} \int_0^H \mathrm{e}^{(H-\tau)\frac{z}{H}} \frac{\tau^{k-1}}{(k-1)!} \mathrm{d}\tau, \quad k \geq 1. \tag{5.15}$$

Substituting $z = c_i H\mathcal{L}$ and $z = H\mathcal{L}$ into (5.15) and inserting the obtained results for $\varphi_k(c_i H\mathcal{L})$ and $\varphi_k(H\mathcal{L})$ into (5.12) shows that

$$a_{ij}(H\mathcal{L}) = \int_0^{c_i H} \mathrm{e}^{(c_i H - \tau)\mathcal{L}} \sum_{k=1}^{\ell_{ij}} \frac{\alpha_{ij}^{(k)}}{(c_i H)^k (k-1)!} \tau^{k-1} \mathrm{d}\tau, \tag{5.16a}$$

$$b_i(H\mathcal{L}) = \int_0^H \mathrm{e}^{(H-\tau)\mathcal{L}} \sum_{k=1}^{m_i} \frac{\beta_i^{(k)}}{H^k(k-1)!}\tau^{k-1}\mathrm{d}\tau. \tag{5.16b}$$

Using the fact that $F(t_n, u_n) = \mathcal{L}u_n + \mathcal{N}(t_n, u_n)$ and

$$\varphi_1(Z) = \frac{1}{H}\int_0^H \mathrm{e}^{(H-\tau)\frac{Z}{H}}\mathrm{d}\tau = (\mathrm{e}^Z - I)Z^{-1}. \tag{5.17}$$

we can write (5.7) in an equivalent form,

$$U_{n,i} = \mathrm{e}^{c_iH\mathcal{L}}u_n + c_iH\varphi_1(c_iH\mathcal{L})\mathcal{N}(t_n, u_n) + H\sum_{j=2}^{i-1} a_{ij}(H\mathcal{L})D_{n,j}, \tag{5.18a}$$

$$u_{n+1} = \mathrm{e}^{H\mathcal{L}}u_n + H\varphi_1(H\mathcal{L})\mathcal{N}(t_n, u_n) + H\sum_{i=2}^{s} b_i(H\mathcal{L})D_{n,i}, \tag{5.18b}$$

for $i = 2, \ldots, s$. We now insert the integral form of $\varphi_1(Z)$ in (5.17) (with $Z = c_iH\mathcal{L}$ and $Z = H\mathcal{L}$) and (5.16) into (5.18) to get

$$U_{n,i} = \mathrm{e}^{c_iH\mathcal{L}}u_n + \int_0^{c_iH} \mathrm{e}^{(c_iH-\tau)\mathcal{L}}p_{n,i}(\tau)\mathrm{d}\tau, \quad i = 2, \ldots, s, \tag{5.19a}$$

$$u_{n+1} = \mathrm{e}^{H\mathcal{L}}u_n + \int_0^H \mathrm{e}^{(h-\tau)\mathcal{L}}q_n(\tau)\mathrm{d}\tau \tag{5.19b}$$

with $p_{n,i}(\tau)$ and $q_n(\tau)$ as shown in (5.14). Clearly, these representations (variation-of-constant formulas) show the conclusion of Theorem 5.3.1. In particular, $U_{n,i} = v_{n,i}(c_iH)$ and $u_{n+1} = v_n(H)$. Thus one can consider (5.13) as modified differential equations with identical solutions as the ExpRK approximations to (5.11). □

We note that the idea of using an ODE to represent a linear combination of matrix-vector $\varphi_k(A)v_k$ was also used in [77].

5.3.2. MERK methods and a multirate algorithm

We now present an algorithm to solve the modified ODEs (5.13a) and (5.13b) numerically. For later use, we first denote the numerical solutions of (5.13a) on $[0, c_iH]$ and (5.13b) on $[0, H]$ by $\widehat{U}_{n,i}$ and $\hat{u}_{n+1}$, respectively. This means that $\widehat{U}_{n,i} \approx v_{n,i}(c_iH) = U_{n,i}$ and $\hat{u}_{n+1} \approx v_n(H) = u_{n+1}$.

Clearly, the polynomials $p_{n,i}(\tau)$ and $q_n(\tau)$ in (5.14) are not given analytically since $D_{n,i}$ are unknowns; however, these polynomials can be numerically determined as follows. For simplicity, we illustrate our procedure by starting with $n = 0$ and $i = s = 2$. In this case we know $u_0 = u(t_0)$ and $p_{0,2}(\tau) = \mathcal{N}(t_0, u_0)$, so one can solve the ODE (5.13a) on $[0, c_2H]$ to get an approximation to $U_{0,2}$, $\widehat{U}_{0,2} \approx U_{0,2} = v_{0,2}(c_2H)$. Then replacing the unknown $U_{0,2}$ in (5.14b) by $\widehat{U}_{0,2}$, we have

$$\hat{q}_0(\tau) = \mathcal{N}(t_0, u_0) + \sum_{k=1}^{m_i} \frac{\beta_2^{(k)}}{H^{k-1}(k-1)!}\tau^{k-1}\widehat{D}_{0,2},$$

where $\widehat{D}_{0,2} = \mathcal{N}(t_0 + c_2H, \widehat{U}_{0,2}) - \mathcal{N}(t_0, u_0)$. Since $\hat{q}_0(\tau) \approx q_0(\tau)$, we may then solve the ODE (5.13b) on $[0, H]$ with $\hat{q}_0(\tau)$ in place of $q_0(\tau)$ to obtain an approximation $\hat{u}_1 \approx u_1 = v_0(H)$.

This general process may be extended to larger numbers of stages $s \geq 2$ and for subsequent time steps $n \geq 0$. Approximating $\hat{u}_n \approx u_n$ (with $\hat{u}_0 = u_0$), then for $i = 2, \ldots, s$, we define the following perturbed linear ODEs over $\tau \in [0, c_iH]$:

$$y'_{n,i}(\tau) = \mathcal{L}y_{n,i}(\tau) + \hat{p}_{n,i}(\tau), \quad y_{n,i}(0) = \hat{u}_n, \tag{5.20}$$

with

$$\hat{p}_{n,i}(\tau) = \mathcal{N}(t_n, \hat{u}_n) + \sum_{j=2}^{i-1} \Big(\sum_{k=1}^{\ell_{ij}} \frac{\alpha_{ij}^{(k)}}{c_i^k H^{k-1}(k-1)!}\tau^{k-1} \Big) \widehat{D}_{n,j}, \tag{5.21}$$

$$\widehat{D}_{n,i} = \mathcal{N}(t_n + c_iH, \widehat{U}_{n,i}) - \mathcal{N}(t_n, \widehat{u}_n), \tag{5.22}$$

that provide the approximations

$$\widehat{U}_{n,i} \approx y_{n,i}(c_i H) \approx v_{n,i}(c_i H) = U_{n,i}.$$

With these in place, we then solve the linear ODE

$$y_n'(\tau) = \mathcal{L} y_n(\tau) + \hat{q}_n(\tau), \quad y_n(0) = \hat{u}_n \tag{5.23}$$

over $\tau \in [0, H]$, with

$$\hat{q}_n(\tau) = \mathcal{N}(t_n, \hat{u}_n) + \sum_{i=2}^{s} \Big(\sum_{k=1}^{m_i} \frac{\beta_i^{(k)}}{H^{k-1}(k-1)!} \tau^{k-1} \Big) \widehat{D}_{n,i}, \tag{5.24}$$

to obtain the approximate time-step solutions,

$$\hat{u}_{n+1} \approx y_n(H) \approx v_n(H) = u_{n+1}.$$

Since the above procedure uses a "macro" time step H to integrate the slow process, and a "micro" time step h to integrate the fast process (via solving the ODEs (5.20) and (5.23)), we call the resulting methods (5.20)-(5.22) *Multirate Exponential Runge–Kutta (MERK)* methods. By construction, these MERK methods offer several interesting features. They reduce the solution of nonlinear problems (5.1) to the solution of a sequence of linear differential equations (5.20) and (5.23), using very few evaluations of the nonlinear operator $\mathcal{N}$. Thus they can be more efficient for problems where the linear part is much less costly to compute than the nonlinear part. Additionally, they do not require the computation of matrix functions, as is the case with ExpRK methods. Moreover, these methods do not require a starting value procedure as in multirate algorithms for exponential multistep methods [27, 53].

We provide the following Algorithm 1 to give a succinct overview of the implementation of our MERK methods.

- **Input:** $\mathcal{L}$; $\mathcal{N}(t,u)$; t_0; u_0; s; c_i $(i=1,\ldots,s)$; H
- **Initialization:** Set $n=0$; $\hat{u}_n = u_0$.
  While $t_n < T$
  1. Set $\widehat{U}_{n,1} = \hat{u}_n$.
  2. For $i = 2,\ldots,s$ do
     (a) Find $\hat{p}_{n,i}(\tau)$ as in (5.21).
     (b) Solve (5.20) on $[0, c_i H]$ to obtain $\widehat{U}_{n,i} \approx y_{n,i}(c_i H)$.
  3. Find $\hat{q}_{n,s}(\tau)$ as in (5.24)
  4. Solve (5.23) on $[0,H]$ to get $\hat{u}_{n+1} \approx y_n(H)$.
  5. Update $t_{n+1} := t_n + H$, $n := n+1$.
- **Output:** Approximate values $\hat{u}_n \approx u_n, n = 1,2,\ldots$ (where u_n is the numerical solution at time t_n obtained by an ExpRK method).

Algorithm 1: MERK method

5.3.3. Convergence analysis

Since MERK methods are constructed to approximate ExpRK methods, we perform their error analysis in the framework of analytic semigroups on a Banach space X, under the following assumptions (see e.g., [51, 70]).

Assumption 1. The linear operator $\mathcal{L}$ is the infinitesimal generator of an analytic semigroup $\mathrm{e}^{t\mathcal{L}}$ on X. This implies that

$$\|\mathrm{e}^{t\mathcal{L}}\|_{X\leftarrow X} \leq C, \quad t \geq 0 \tag{5.25}$$

and consequently $\varphi_k(H\mathcal{L})$, $a_{ij}(H\mathcal{L})$ and $b_i(H\mathcal{L})$ are bounded operators.

Assumption 2 (for high-order methods). The solution $u : [t_0, T] \to X$ of (5.1) is sufficiently smooth with derivatives in X, and $\mathcal{N} : [t_0, T] \times X \to X$ is sufficiently Fréchet differentiable in a strip along the exact solution. All derivatives occurring in the remainder of this section are therefore assumed to be uniformly bounded.

We analyze the error in MERK methods starting with the local error of ExpRK methods. Therefore, we first consider (5.6) (in its explicit form) with exact initial value $u(t_n)$:

$$\breve{U}_{n,i} = \mathrm{e}^{c_i H\mathcal{L}} u(t_n) + H \sum_{j=1}^{i-1} a_{ij}(H\mathcal{L})\,\mathcal{N}(t_n + c_j H, \breve{U}_{n,j}), \quad i = 2, \dots, s, \tag{5.26a}$$

$$\breve{u}_{n+1} = \mathrm{e}^{H\mathcal{L}} u(t_n) + H \sum_{i=1}^{s} b_i(H\mathcal{L})\,\mathcal{N}(t_n + c_i H, \breve{U}_{n,i}) \tag{5.26b}$$

and thus the MERK methods (5.13)–(5.14) are considered with polynomials

$$\breve{p}_{n,i}(\tau) = \mathcal{N}(t_n, u(t_n)) + \sum_{j=2}^{i-1} \Big(\sum_{k=1}^{\ell_{ij}} \frac{\alpha_{ij}^{(k)}}{c_i^k H^{k-1}(k-1)!} \tau^{k-1} \Big) \breve{D}_{n,j}, \tag{5.27a}$$

$$\breve{q}_{n}(\tau) = \mathcal{N}(t_n, u(t_n)) + \sum_{i=2}^{s} \Big(\sum_{k=1}^{m_i} \frac{\beta_{i}^{(k)}}{H^{k-1}(k-1)!} \tau^{k-1} \Big) \breve{D}_{n,i}, \tag{5.27b}$$

where $\breve{D}_{n,i} = \mathcal{N}(t_n + c_i H, \breve{U}_{n,i}) - \mathcal{N}(t_n, u(t_n))$.

Error notation. Since MERK methods consist of approximations to approximations, we must clearly isolate the errors induced at each approximation level. To this end, we let $\hat{e}_{n+1} = \hat{u}_{n+1} - u(t_{n+1})$ denote the global error at time t_{n+1} of a MERK method (5.20)-(5.22). Let $\breve{e}_{n+1} = \breve{u}_{n+1} - u(t_{n+1})$ denote the local error at t_{n+1} of the base ExpRK method. Let $\hat{\varepsilon}_{n,i} = \widehat{U}_{n,i} - y_{n,i}(c_i H)$ and $\hat{\varepsilon}_{n+1} = \hat{u}_{n+1} - y_n(H)$ denote the (global) errors of the ODE solvers when integrating (5.20) on $[0, c_i H]$ and (5.23) on $[0, H]$ (note that $\hat{\varepsilon}_{n,1} = \hat{u}_n - y_{n,i}(0) = 0$ since $c_1 = 0$).

First, we may write total error as the sum of the errors in each approximation,

$$\hat{e}_{n+1} = \hat{\varepsilon}_{n+1} + (y_n(H) - \breve{u}_{n+1}) + \breve{e}_{n+1}. \tag{5.28}$$

Applying the variation-of-constants formula to (5.24) and using (5.16b), we then write

$$y_n(H) = \mathrm{e}^{H\mathcal{L}}\hat{u}_n + H\sum_{i=1}^{s} b_i(H\mathcal{L})\mathcal{N}(t_n + c_iH, \widehat{U}_{n,i}). \tag{5.29}$$

Inserting $y_n(H)$ and $\breve{u}_{n+1}$ from (5.29) and (5.26b) into (5.28) gives

$$\hat{e}_{n+1} = \mathrm{e}^{H\mathcal{L}}\hat{e}_n + \hat{\varepsilon}_{n+1} + H\mathcal{S}_{n,s} + \breve{e}_{n+1}, \tag{5.30}$$

where

$$\mathcal{S}_{n,s} = \sum_{i=1}^{s} b_i(H\mathcal{L})\big(\mathcal{N}(t_n + c_iH, \widehat{U}_{n,i}) - \mathcal{N}(t_n + c_iH, \breve{U}_{n,i})\big). \tag{5.31}$$

Next, we prove some preliminary results.

Lemma 5.3.1. *Denoting* $\widehat{E}_{n,i} = \widehat{U}_{n,i} - \breve{U}_{n,i}$ *and* $\breve{N}_{n,i} = \frac{\partial \mathcal{N}}{\partial u}(t_n + c_iH, \breve{U}_{n,i})$*, we have*

$$\widehat{E}_{n,i} = \hat{\varepsilon}_{n,i} + \mathrm{e}^{c_iH\mathcal{L}}\hat{e}_n + H\sum_{j=1}^{i-1} a_{ij}(H\mathcal{L})(\breve{N}_{n,j}\widehat{E}_{n,j} + \widehat{R}_{n,j}) \tag{5.32}$$

with

$$\widehat{R}_{n,j} = \int_0^1 (1-\theta)\frac{\partial^2 \mathcal{N}}{\partial u^2}(t_n + c_jH, \breve{U}_{n,j} + \theta\widehat{E}_{n,j})(\widehat{E}_{n,j}, \widehat{E}_{n,j})\mathrm{d}\theta. \tag{5.33}$$

Furthermore, under Assumption 2, the bound

$$\|\widehat{R}_{n,j}\| \leqslant C\|\widehat{E}_{n,j}\|^2, \ \textit{i.e.}, \ \widehat{R}_{n,j} = \mathcal{O}(\|\widehat{E}_{n,j}\|^2) \tag{5.34}$$

is held as long as $\widehat{E}_{n,j}$ *remains in a sufficiently small neighborhood of* 0*.*

Proof. We first rewrite

$$\widehat{E}_{n,i} = \widehat{U}_{n,i} - y_{n,i}(c_i H) + (y_{n,i}(c_i H) - \breve{U}_{n,i}) = \hat{\varepsilon}_{n,i} + (y_{n,i}(c_i H) - \breve{U}_{n,i}). \tag{5.35}$$

Here $y_{n,i}(c_i H)$ is the exact solution of (5.20), which can be represented by the variation-of-constants formula and then rewritten by using (5.16a) and (5.21) as:

$$y_{n,i}(c_i H) = \mathrm{e}^{c_i H \mathcal{L}} \hat{u}_n + H \sum_{j=1}^{i-1} a_{ij}(H\mathcal{L}) \mathcal{N}(t_n + c_j H, \widehat{U}_{n,j}). \tag{5.36}$$

Subtracting (5.26a) from (5.36) and inserting the obtained result into (5.35) gives

$$\widehat{E}_{n,i} = \hat{\varepsilon}_{n,i} + \mathrm{e}^{c_i H \mathcal{L}} \hat{e}_n + H \sum_{j=1}^{i-1} a_{ij}(H\mathcal{L}) \big(\mathcal{N}(t_n + c_j H, \widehat{U}_{n,j}) - \mathcal{N}(t_n + c_j H, \breve{U}_{n,j}) \big). \tag{5.37}$$

Using the Taylor series expansion of $\mathcal{N}(t, u)$ at $(t_n + c_j H, \breve{U}_{n,j})$, we get

$$\mathcal{N}(t_n + c_j H, \widehat{U}_{n,j}) - \mathcal{N}(t_n + c_j H, \breve{U}_{n,j}) = \breve{N}_{n,j} \widehat{E}_{n,j} + \widehat{R}_{n,j} \tag{5.38}$$

with the remainder $\widehat{R}_{n,j}$ given in (5.33), which clearly satisfies (5.34) due to Assumption 2. Inserting (5.38) into (5.37) shows (5.32). □

Lemma 5.3.2. *Under Assumptions 1 and 2, there exist bounded operators $\mathcal{T}_{n,i}(\hat{\varepsilon}_{n,i})$ and $\mathcal{B}_n(\hat{e}_n)$ on X such that*

$$\mathcal{S}_{n,s} = \sum_{i=2}^{s} \big(b_i(H\mathcal{L}) \breve{N}_{n,i} + H \mathcal{T}_{n,i}(\hat{\varepsilon}_{n,i}) \big) \hat{\varepsilon}_{n,i} + \mathcal{B}_n(\hat{e}_n) \hat{e}_n. \tag{5.39}$$

Note that $\mathcal{T}_{n,i}$ also depends on H, $\hat{\varepsilon}_{n,j}$, $a_{ij}(H\mathcal{L})$, $\breve{N}_{n,j}$ $(j = 2, \ldots, i-1)$, and $\hat{e}_n$; and $\mathcal{B}_n$ also depends on H, $b_i(H\mathcal{L}), a_{ij}(H\mathcal{L})$, c_i, and $\breve{N}_{n,i}$.

Proof. Inserting (5.38) (with i in place of j) into (5.31) gives

$$\mathcal{S}_{n,s} = \sum_{i=1}^{s} b_i(H\mathcal{L})\big(\breve{N}_{n,i}\widehat{E}_{n,i} + \widehat{R}_{n,i}\big). \tag{5.40}$$

Using the recursion (5.32) from Lemma 5.3.1, we further expand $\widehat{E}_{n,i}$ as

$$\begin{aligned}\widehat{E}_{n,i} = \hat{\varepsilon}_{n,i} &+ H\sum_{j=1}^{i-1} a_{ij}(H\mathcal{L})\breve{N}_{n,j}\hat{\varepsilon}_{n,j} + H^2\sum_{j=1}^{i-1} a_{ij}(H\mathcal{L})\breve{N}_{n,j}\sum_{k=1}^{j-1} a_{jk}(H\mathcal{L})\breve{N}_{n,k}\widehat{E}_{n,k} \\ &+ H\sum_{j=1}^{i-1} a_{ij}(H\mathcal{L})\widehat{R}_{n,j} + H^2\sum_{j=1}^{i-1} a_{ij}(H\mathcal{L})\breve{N}_{n,j}\sum_{k=1}^{j-1} a_{jk}(H\mathcal{L})\widehat{R}_{n,k} \\ &+ \Big(\mathrm{e}^{c_i H\mathcal{L}} + H\sum_{j=1}^{i-1} a_{ij}(H\mathcal{L})\breve{N}_{n,j}\mathrm{e}^{c_j H\mathcal{L}}\Big)\hat{e}_n.\end{aligned} \tag{5.41}$$

Using (5.32) and (5.34) ($\widehat{R}_{n,i} = \mathcal{O}(\|\widehat{E}_{n,i}\|^2)$ and proceeding by induction, one can complete the recursion (5.41) for $\widehat{E}_{n,i}$. Inserting this recursion into (5.40) (and noting that $\hat{\varepsilon}_{n,1} = 0$) yields (5.39). Based on the structure of (5.41) and (5.40), under the given assumptions it is clear that the boundedness of $\mathcal{T}_{n,i}(\hat{\varepsilon}_{n,i})$ and $\mathcal{B}_n(\hat{e}_n)$ follow from the boundedness of $a_{ij}(H\mathcal{L}), b_i(H\mathcal{L})$, $\breve{N}_{n,i}$, and $\widehat{R}_{n,i}$. □

We now present the main convergence result for MERK methods.

Theorem 5.3.2. *Let the initial value problem* (5.1) *satisfy Assumptions 1–2. Consider for its numerical solution a MERK method* (5.20)–(5.22) *that is constructed from an ExpRK method of global order p. We further assume that the "fast" ODEs* (5.20) *and* (5.23) *associated with the MERK method are integrated with micro time step $h = H/m$ by using ODE solvers that have global order of convergence q and r, respectively, and where m is the number*

of fast steps per slow step. Then, the MERK method is convergent, and has error bound

$$\|u_n - u(t_n)\| \leq C_1 H^p + C_2 H h^q + C_3 h^r \tag{5.42}$$

on compact time intervals $t_0 \leq t_n = t_0 + nH \leq T$. *Here, the constant* C_1 *depends on* $T - t_0$, *but is independent of* n *and* H; *and the constants* C_2 *and* C_3 *also depend on the error constants of the choosen ODE solvers.*

Proof. We first note that since we only employ the fast ODE solvers on time intervals $[0, c_i H]$ and $[0, H]$, then our assumption regarding their accuracies of order q and r is typically equivalent to [45, Thm. 3.6]

$$\hat{\varepsilon}_{n,i} = \frac{\tilde{c}_2}{\Lambda_i} h^q \left(\mathrm{e}^{\Lambda_i c_i H} - 1 \right) = \tilde{c}_2 c_i \varphi_1(\Lambda_i c_i H) H h^q = c_2 H h^q, \tag{5.43a}$$

$$\hat{\varepsilon}_{n} = \frac{\tilde{c}_3}{\Lambda} h^r \left(\mathrm{e}^{\Lambda H} - 1 \right) = \tilde{c}_3 \varphi_1(\Lambda H) H h^r = c_3 H h^r, \tag{5.43b}$$

(due to (5.17)) where Λ_i, Λ are the Lipschitz constants for the increment functions of the ODE solvers applied to the problems (5.20) and (5.23), respectively.

For simplicity of notation, we denote $B_{n,i} = b_i(H\mathcal{L})\breve{N}_{n,i} + H\mathcal{T}_{n,i}(\hat{\varepsilon}_{n,i})$.
Clearly, $B_{n,i}$ is a bounded operator and thus (5.39) becomes

$$\mathcal{S}_{n,s} = \sum_{i=2}^{s} B_{n,i} \hat{\varepsilon}_{n,i} + \mathcal{B}_n(\hat{e}_n)\hat{e}_n. \tag{5.44}$$

Inserting this into (5.30) gives

$$\hat{e}_{n+1} = \mathrm{e}^{H\mathcal{L}} \hat{e}_n + H\mathcal{B}_n(\hat{e}_n)\hat{e}_n + \breve{e}_{n+1} + H \left(\sum_{i=2}^{s} B_{n,i} \hat{\varepsilon}_{n,i} + \hat{\varepsilon}_{n+1} \right). \tag{5.45}$$

Solving recursion (5.45) and using $\hat{e}_0 = 0$ (since $\hat{u}_0 = u_0 = u(t_0)$) finally yields

$$\hat{e}_n = H \sum_{j=0}^{n-1} \mathrm{e}^{(n-1-j)H\mathcal{L}} \mathcal{B}_j(\hat{e}_j)\hat{e}_j + \sum_{j=0}^{n-1} \mathrm{e}^{jH\mathcal{L}} \Big(\breve{e}_{n-j} + H \sum_{i=2}^{s} B_{n-1-j,i} \hat{\varepsilon}_{n-1-j,i} + \hat{\varepsilon}_{n-j} \Big) \tag{5.46}$$

Since the ExpRK method has global order p, we have the local error $\breve{e}_{n-j} = \mathcal{O}(H^{p+1})$, and from (5.43) we have $\hat{\varepsilon}_{n-1-j,i} = \mathcal{O}(Hh^q)$, $\hat{\varepsilon}_{n-j} = \mathcal{O}(Hh^r)$. Using (5.25) we derive from (5.46) that

$$\|\hat{e}_n\| \leq H \sum_{j=0}^{n-1} C\|\hat{e}_j\| + \sum_{j=0}^{n-1} C\big(c_1 H^{p+1} + c_2 H^2 h^q + c_3 H h^r\big). \tag{5.47}$$

An application of a discrete Gronwall lemma to (5.47) results in the bound (5.42). □

Remark 5.3.1. *Under the assumption of a fixed time-scale separation factor, $m = H/h$, Theorem 5.3.2 implies that for a MERK method* (5.20)–(5.22) *to converge with order p, the inner ODE solvers for* (5.20) *and* (5.23) *must have orders $q \geq p-1$ and $r \geq p$, respectively.*

Remark 5.3.2. *Alternately, Theorem 5.3.2 shows that MERK methods may in fact use a method of* any *order to integrate the modified fast ODEs, as long as m is adjusted accordingly. Specifically, if methods of order q and r are used to solve the modified "fast" ODEs* (5.20) *and* (5.23)*, then an overall order p MERK method may be retained by use of $m \geq C_1 H^{(q-p+1)/q}$ for* (5.20) *and $m \geq C_2 H^{(r-p)/r}$ for* (5.23)*, for appropriate constants C_1 and C_2.*

5.4. Derivation of MERK methods

Based on the theory presented in Section 5.3, we now derive MERK schemes up to order 5, relying heavily on ExpRK schemes that fit the assumption of Theorem 5.3.1. As we are interested in problems with significant time scale separation $H \gg h$, we primarily focus on stiffly-accurate ExpRK schemes. Since MERK methods involve linear ODEs (5.20) and (5.23) with a fixed coefficient matrix $\mathcal{L}$ for the fast portion, they are characterized by the polynomials defined in (5.21) and (5.24). Therefore, when deriving MERK schemes we display only their corresponding polynomials $\hat{p}_{n,i}(\tau)$ and $\hat{q}_n(\tau)$.

5.4.1. Second-order methods

When searching for stiffly-accurate second-order ExpRK methods, we find the following scheme that uses $s = 2$ stages (see [50, Sect. 5.1]) and satisfies Theorem 5.3.1:

$$\begin{aligned} U_{n,2} &= u_n + c_2 H \varphi_1(c_2 H\mathcal{L}) F(t_n, u_n) \\ u_{n+1} &= u_n + H\varphi_1(H\mathcal{L})F(t_n, u_n) + h\tfrac{1}{c_2}\varphi_2(H\mathcal{L})D_{n,2}. \end{aligned} \tag{5.48}$$

From this, using the conclusion of Theorem 5.3.1, we derive the corresponding family of second-order MERK methods, which we call `MERK2`:

$$\begin{aligned} \hat{p}_{n,2}(\tau) &= \mathcal{N}(t_n, \hat{u}_n), \qquad\qquad \tau \in [0, c_2 H] \\ \hat{q}_n(\tau) &= \mathcal{N}(t_n, \hat{u}_n) + \tfrac{\tau}{c_2 H}\widehat{D}_{n,2}, \qquad \tau \in [0, H]. \end{aligned} \tag{5.49}$$

Since we do not use this scheme in our numerical experiments, we do not specify a value for c_2. We note that for these methods, the fast time scale must be evolved a duration of $(1 + c_2)H$ for each slow time step.

5.4.2. Third-order methods

Also from [50, Sect. 5.2] we consider the following family of third-order, three-stage, ExpRK methods that satisfy Theorem 5.3.1:

$$\begin{aligned} U_{n,2} &= u_n + c_2 H \varphi_1(c_2 H\mathcal{L}) F(t_n, u_n) \\ U_{n,3} &= u_n + \tfrac{2}{3} H \varphi_1(\tfrac{2}{3} H\mathcal{L}) F(t_n, u_n) + \tfrac{4}{9c_2}\varphi_2(\tfrac{2}{3}H\mathcal{L})D_{n,2}, \\ u_{n+1} &= u_n + H\varphi_1(H\mathcal{L})F(t_n, u_n) + h\tfrac{3}{2}\varphi_2(H\mathcal{L})D_{n,3}. \end{aligned} \tag{5.50}$$

From these, we construct the following third-order `MERK3` scheme:

$$\begin{aligned}
\hat{p}_{n2}(\tau) &= \mathcal{N}(t_n, \hat{u}_n), && \tau \in [0, c_2 H] \\
\hat{p}_{n3}(\tau) &= \mathcal{N}(t_n, \hat{u}_n) + \tfrac{\tau}{c_2 H}\widehat{D}_{n,2}, && \tau \in [0, \tfrac{2}{3}H] \\
\hat{q}_{n}(\tau) &= \mathcal{N}(t_n, \hat{u}_n) + \tfrac{3\tau}{2H}\widehat{D}_{n,3}, && \tau \in [0, H].
\end{aligned} \tag{5.51}$$

In our numerical experiments with this scheme, we choose $c_2 = \frac{1}{2}$. Hence, the fast time scale must be evolved a duration of $\frac{13}{6}H$ for each slow time step.

5.4.3. Fourth-order methods

To the best of our knowledge, the only 5 stage, stiffly-accurate ExpRK method of order four was given in [50, Sect. 5.3]. However, this scheme does not satisfy Theorem 5.3.1 due to the coefficient

$$a_{52}(H\mathcal{L}) = \tfrac{1}{2}\varphi_2(c_5 H\mathcal{L}) - \varphi_3(c_4 H\mathcal{L}) + \tfrac{1}{4}\varphi_2(c_4 H\mathcal{L}) - \tfrac{1}{2}\varphi_3(c_5 H\mathcal{L}),$$

which is not a linear combination of $\{\varphi_k(c_5 H\mathcal{L})\}_{k=1}^{5}$. Therefore, we cannot use it to derive a fourth-order MERK scheme. However, in a very recent submitted paper [67], we have derived a family of fourth-order, 6-stage, stiffly-accurate ExpRK methods (named `expRK4s6`), that

additionally fulfill Theorem 5.3.1:

$$
\begin{aligned}
U_{n,2} &= u_n + \varphi_1(c_2 H\mathcal{L}) c_2 H F(t_n, u_n), \\
U_{n,k} &= u_n + \varphi_1(c_k H\mathcal{L}) c_k H F(t_n, u_n) + \varphi_2(c_k H\mathcal{L}) \tfrac{c_k^2}{c_2} H D_{n,2}, && k = 3,4 \\
U_{n,j} &= u_n + \varphi_1(c_j H\mathcal{L}) c_j h F(t_n, u_n) + \varphi_2(c_j H\mathcal{L}) \tfrac{c_j^2}{c_3 - c_4} H\big(\tfrac{-c_4}{c_3} D_{n,3} + \tfrac{c_3}{c_4} D_{n,4}\big) \\
&\quad + \varphi_3(c_j H\mathcal{L}) \tfrac{2c_j^3}{c_3 - c_4} H\big(\tfrac{1}{c_3} D_{n,3} - \tfrac{1}{c_4} D_{n,4}\big), && j = 5,6 \\
u_{n+1} &= u_n + \varphi_1(H\mathcal{L}) h F(t_n, u_n) + \varphi_2(H\mathcal{L}) \tfrac{1}{c_5 - c_6} H\big(\tfrac{-c_6}{c_5} D_{n,5} + \tfrac{c_5}{c_6} D_{n,6}\big) \\
&\quad + \varphi_3(H\mathcal{L}) \tfrac{2}{c_5 - c_6} H\big(\tfrac{1}{c_5} D_{n,5} - \tfrac{1}{c_6} D_{n,6}\big).
\end{aligned}
\tag{5.52}
$$

Since the pairs of internal stages $\{U_{n,3}, U_{n,4}\}$ and $\{U_{n,5}, U_{n,6}\}$ are independent of one other (they can be computed simultaneously) and have the same format, this scheme behaves like a 4-stage method. Hence, instead of using 6 polynomials we need only 4 to derive the following family of fourth-order MERK schemes, which we call `MERK4`:

$$
\begin{aligned}
\hat{p}_{n,2}(\tau) &= \mathcal{N}(t_n, \hat{u}_n), && \tau \in [0, c_2 H] \\
\hat{p}_{n,3}(\tau) &= \hat{p}_{n,4}(\tau) = \mathcal{N}(t_n, \hat{u}_n) + \tfrac{\tau}{c_2 H} \widehat{D}_{n,2}, && \tau \in [0, c_3 H] \\
\hat{p}_{n,5}(\tau) &= \hat{p}_{n,6}(\tau) = \mathcal{N}(t_n, \hat{u}_n) + \tfrac{\tau}{H}\big(\tfrac{-c_4}{c_3(c_3 - c_4)} \widehat{D}_{n,3} + \tfrac{c_3}{c_4(c_3 - c_4)} \widehat{D}_{n,4}\big) \\
&\quad + \tfrac{\tau^2}{H^2}\big(\tfrac{1}{c_3(c_3 - c_4)} \widehat{D}_{n,3} - \tfrac{1}{c_4(c_3 - c_4)} \widehat{D}_{n,4}\big), && \tau \in [0, c_5 H] \\
\hat{q}_n(\tau) &= \mathcal{N}(t_n, \hat{u}_n) + \tfrac{\tau}{H}\big(\tfrac{-c_6}{c_5(c_5 - c_6)} \widehat{D}_{n,5} + \tfrac{c_5}{c_6(c_5 - c_6)} \widehat{D}_{n,6}\big) \\
&\quad + \tfrac{\tau^2}{H^2}\big(\tfrac{1}{c_5(c_5 - c_6)} \widehat{D}_{n,5} - \tfrac{1}{c_6(c_5 - c_6)} \widehat{D}_{n,6}\big), && \tau \in [0, H].
\end{aligned}
\tag{5.53}
$$

For our numerical experiments, we choose the coefficients $c_2 = c_3 = \frac{1}{2}$, $c_4 = c_6 = \frac{1}{3}$, and $c_5 = \frac{5}{6}$. With this choice, we may then solve the linear ODE (5.20) using the polynomial $\hat{p}_{n,3}(\tau)$ on $[0, c_3 H]$ to get both $\widehat{U}_{n,3} \approx U_{n,3} = v_{n,3}(c_3 H)$ and $\widehat{U}_{n,4} \approx U_{n,4}$ (since $c_4 < c_3$) without solving an additional fast differential equation on $[0, c_4 H]$. Similarly, we may solve the linear ODE (5.20) with the polynomial $\hat{p}_{n,5}(\tau)$ on $[0, c_5 H]$ to obtain both $\widehat{U}_{n,5} \approx U_{n,5}$ and $\widehat{U}_{n,6} \approx U_{n,6}$. As a result, the fast time scale must only be evolved for a total duration of $\frac{17}{6} H$ for each slow time step.

5.4.4. Fifth-order methods

Similar to fourth-order ExpRK methods, there are no stiffly-accurate fifth-order methods available in the literature that fulfill Theorem 5.3.1. In particular, the only existing fifth-order scheme (`expRK5s8`, that requires 8 stages) was constructed in [70]. However, its coefficients $a_{75}(H\mathcal{L})$, $a_{76}(H\mathcal{L})$, $a_{85}(H\mathcal{L})$, $a_{86}(H\mathcal{L})$ and $a_{87}(H\mathcal{L})$ involve several different linear combinations of $\varphi_k(c_i H\mathcal{L})$ with different scalings c_6, c_7, c_8, and may not be used to create a MERK method. Again, in [67], we have constructed a new family of efficient, fifth-order, 10-stage, stiffly-accurate ExpRK methods (called `expRK5s10`) that fulfills Theorem 5.3.1:

$$
\begin{aligned}
U_{n,2} &= u_n + \varphi_1(c_2 H\mathcal{L}) c_2 H F(t_n, u_n), \\
U_{n,k} &= u_n + \varphi_1(c_k H\mathcal{L}) c_k H F(t_n, u_n) + \varphi_2(c_k H\mathcal{L}) \tfrac{c_k^2}{c_2} H D_{n,2}, \qquad k = 3, 4 \\
U_{n,j} &= u_n + \varphi_1(c_j H\mathcal{L}) c_j H F(t_n, u_n) + \varphi_2(c_j H\mathcal{L}) c_j^2 H \big(\alpha_3 D_{n,3} + \alpha_4 D_{n,4}\big) \\
&\quad + \varphi_3(c_j H\mathcal{L}) c_j^3 H \big(\beta_3 D_{n,3} - \beta_4 D_{n,4}\big), \qquad j = 5, 6, 7
\end{aligned}
\tag{5.54a}
$$

$$
\begin{aligned}
U_{n,m} &= u_n + \varphi_1(c_m H\mathcal{L}) c_m H F(t_n, u_n) \\
&\quad + \varphi_2(c_m H\mathcal{L}) c_m^2 H\big(\alpha_5 D_{n,5} + \alpha_6 D_{n,6} + \alpha_7 D_{n,7}\big) \\
&\quad + \varphi_3(c_m H\mathcal{L}) c_m^3 H\big(\beta_5 D_{n,5} - \beta_6 D_{n,6} - \beta_7 D_{n,7}\big) \\
&\quad + \varphi_4(c_m H\mathcal{L}) c_m^4 H\big(\gamma_5 D_{n,5} + \gamma_6 D_{n,6} + \gamma_7 D_{n,7}\big), \qquad m = 8, 9, 10 \qquad (5.54b) \\
u_{n+1} &= u_n + \varphi_1(H\mathcal{L}) H F(t_n, u_n) + \varphi_2(H\mathcal{L}) H\big(\alpha_8 D_{n,8} + \alpha_9 D_{n,9} + \alpha_{10} D_{n,10}\big) \\
&\quad - \varphi_3(H\mathcal{L}) H\big(\beta_8 D_{n,8} + \beta_9 D_{n,9} + \beta_{10} D_{n,10}\big) \\
&\quad + \varphi_4(H\mathcal{L}) H\big(\gamma_8 D_{n,8} + \gamma_9 D_{n,9} + \gamma_{10} D_{n,10}\big)
\end{aligned}
$$

with coefficients given by

$$
\begin{aligned}
&\alpha_3 = \tfrac{c_4}{c_3(c_4-c_3)},\ \alpha_4 = \tfrac{c_3}{c_4(c_3-c_4)}, \\
&\alpha_5 = \tfrac{c_6 c_7}{c_5(c_5-c_6)(c_5-c_7)},\ \alpha_6 = \tfrac{c_5 c_7}{c_6(c_6-c_5)(c_6-c_7)},\ \alpha_7 = \tfrac{c_5 c_6}{c_7(c_7-c_5)(c_7-c_6)}, \\
&\alpha_8 = \tfrac{c_9 c_{10}}{c_8(c_8-c_9)(c_8-c_{10})},\ \alpha_9 = \tfrac{c_8 c_{10}}{c_9(c_9-c_8)(c_9-c_{10})},\ \alpha_{10} = \tfrac{c_8 c_9}{c_{10}(c_{10}-c_8)(c_{10}-c_9)} \\
&\beta_3 = \tfrac{2}{c_3(c_3-c_4)}, \beta_4 = \tfrac{2}{c_4(c_3-c_4)}, \\
&\beta_5 = \tfrac{2(c_6+c_7)}{c_5(c_5-c_6)(c_5-c_7)},\ \beta_6 = \tfrac{2(c_5+c_7)}{c_6(c_6-c_5)(c_6-c_7)},\ \beta_7 = \tfrac{2(c_5+c_6)}{c_7(c_7-c_5)(c_7-c_6)}, \\
&\beta_8 = \tfrac{2(c_9+c_{10})}{c_8(c_8-c_9)(c_8-c_{10})},\ \beta_9 = \tfrac{2(c_8+c_{10})}{c_9(c_9-c_8)(c_9-c_{10})},\ \beta_{10} = \tfrac{2(c_8+c_9)}{c_{10}(c_{10}-c_8)(c_{10}-c_9)} \\
&\gamma_5 = \tfrac{6}{c_5(c_5-c_6)(c_5-c_7)},\ \gamma_6 = \tfrac{6}{c_6(c_6-c_5)(c_6-c_7)},\ \gamma_7 = \tfrac{6}{c_7(c_7-c_5)(c_7-c_6)}, \\
&\gamma_8 = \tfrac{6}{c_8(c_8-c_9)(c_8-c_{10})},\ \gamma_9 = \tfrac{6}{c_9(c_9-c_8)(c_9-c_{10})},\ \gamma_{10} = \tfrac{6}{c_{10}(c_{10}-c_8)(c_{10}-c_9)}.
\end{aligned}
\qquad (5.54c)
$$

Although this scheme has 10 stages, again its structure facilitates an efficient implementation. Specifically, we note that there are multiple stages $U_{n,i}$ which share the same format (same matrix functions with different inputs c_i), and are independent of one another (namely, $\{U_{n,3}, U_{n,4}\}$, $\{U_{n,5}, U_{n,6}, U_{n,7}\}$, and $\{U_{n,8}, U_{n,9}, U_{n,10}\}$). These groups of stages can again be computed simultaneously, allowing the scheme to behave like a 5-stage method. We therefore propose the corresponding fifth-order MERK methods that use only 5 polynomials, which we name `MERK5`:

$$
\begin{aligned}
\hat{p}_{n,2}(\tau) &= \mathcal{N}(t_n, \hat{u}_n), && \tau \in [0, c_2 H] \\
\hat{p}_{n,3}(\tau) = \hat{p}_{n,4}(\tau) &= \mathcal{N}(t_n, \hat{u}_n) + \tfrac{\tau}{c_2 H}\widehat{D}_{n,2}, && \tau \in [0, c_3 H] \\
\hat{p}_{n,5}(\tau) = \hat{p}_{n,6}(\tau) = \hat{p}_{n,7}(\tau) &= \mathcal{N}(t_n, \hat{u}_n) + \tfrac{\tau}{H}\big(\alpha_3 \widehat{D}_{n,3} + \alpha_4 \widehat{D}_{n,4}\big) \\
&\quad + \tfrac{\tau^2}{2H^2}\big(\beta_3 \widehat{D}_{n,3} - \beta_3 \widehat{D}_{n,4}\big), && \tau \in [0, c_5 H] \\
\hat{p}_{n,8}(\tau) = \hat{p}_{n,9}(\tau) = \hat{p}_{n,10}(\tau) &= \mathcal{N}(t_n, \hat{u}_n) + \tfrac{\tau}{H}\big(\alpha_5 \widehat{D}_{n,5} + \alpha_6 \widehat{D}_{n,6} + \alpha_7 \widehat{D}_{n,7}\big) \\
&\quad - \tfrac{\tau^2}{2H^2}\big(\beta_5 \widehat{D}_{n,5} + \beta_6 \widehat{D}_{n,6} + \beta_7 \widehat{D}_{n,7}\big) \\
&\quad + \tfrac{\tau^3}{6H^3}\big(\gamma_5 \widehat{D}_{n,5} + \gamma_6 \widehat{D}_{n,6} + \gamma_7 \widehat{D}_{n,7}\big), && \tau \in [0, c_8 H] \\
\hat{q}_n(\tau) &= \mathcal{N}(t_n, \hat{u}_n) + \tfrac{\tau}{H}(\alpha_8 \widehat{D}_{n,8} + \alpha_9 \widehat{D}_{n,9} + \alpha_{10} \widehat{D}_{n,10}) \\
&\quad - \tfrac{\tau^2}{2H^2}(\beta_8 \widehat{D}_{n,8} + \beta_9 \widehat{D}_{n,9} + \beta_{10} \widehat{D}_{n,10}) \\
&\quad + \tfrac{\tau^3}{6H^3}\big(\gamma_8 \widehat{D}_{n,8} + \gamma_9 \widehat{D}_{n,9} + \gamma_{10} \widehat{D}_{n,10}\big), && \tau \in [0, H].
\end{aligned}
\tag{5.55}
$$

For our numerical experiments, we choose $c_2 = c_3 = c_5 = c_9 = \frac{1}{2}$, $c_4 = c_6 = \frac{1}{3}$, $c_7 = \frac{1}{4}$, $c_8 = \frac{7}{10}$, and $c_{10} = \frac{2}{3}$. Again, since $c_4 < c_3$, when solving the fast time-scale problem (5.20) with polynomial $\hat{p}_{n,3}(\tau)$ on $[0, c_3 H]$ gives $\widehat{U}_{n,3} \approx U_{n,3} = v_{n,3}(c_3 h)$ and $\widehat{U}_{n,4} \approx U_{n,4} = v_{n,3}(c_4 h)$.

Similarly, since $c_7 < c_6 < c_5$, $\widehat{U}_{n,5}, \widehat{U}_{n,6}$, and $\widehat{U}_{n,7}$ can be obtained by solving a single fast time-scale problem with polynomial $\hat{p}_{n,5}(\tau)$ on $[0, c_5 H]$. Finally since $c_9 < c_{10} < c_8$, one can compute $\widehat{U}_{n,8}$, $\widehat{U}_{n,9}$, and $\widehat{U}_{n,10}$ by solving a single fast time-scale problem with polynomial $\hat{p}_{n,8}(\tau)$ on $[0, c_8 H]$. The sum total of these solves corresponds to evolving the fast time scale for an overall duration of $\frac{16}{5}H$ for each slow time step.

5.5. Numerical experiments

In this section we present results from a variety of numerical tests to examine the performance of the proposed `MERK3`, `MERK4` and `MERK5` methods. These tests are designed to confirm the theoretical convergence rates from Section 5.3.3, and compare efficiency against the Multirate Infinitesimal Step method `MIS-KW3`, which uses a similar approach of evolving the fast component using modified systems of differential equations [61, 107, 98, 101]. Unless otherwise noted, we run these methods with inner explicit Runge–Kutta ODE solvers of the same order of convergence as the MERK method, p:

- Third-order `MIS-KW3` uses the `Knoth-Wolke-ERK` inner method [61];
- Third-order `MERK3` uses the `ERK-3-3` inner method,

$$\begin{array}{c|ccc} 0 & & & \\ 1/2 & 1/2 & & \\ 1 & -1 & 2 & \\ \hline & 1/6 & 2/3 & 1/6 \end{array} \quad ;$$

- Fourth-order `MERK4` uses the `ERK-4-4` inner method [45, Table 1.2, left];
- Fifth-order `MERK5` uses the `Cash-Karp-ERK` inner method [14].

We note that although Theorem 5.3.2 guarantees that when using a MERK method of order p, the internal stage solutions (5.20) can be computed with a solver of order $q = p - 1$ and the step solution (5.23) can use a solver of order $r = p$, for simplicity we have used $r = q = p$

in the majority of our tests. However, we more closely investigate these inner solver order requirements in Section 5.5.3 below.

Not all of our test problems have convenient analytical solutions; for these tests, we compute a reference solution using an 8th-order explicit or a 12th-order implicit Runge–Kutta method with a time step smaller than the smallest micro time step h. When computing solution error, we report the maximum absolute error over all time steps and solution components. From these, we compute convergence rates using a linear least-squares fit of the log-error versus log-macro time step H. For each test we present three types of plots: one convergence plot (error vs H) and two efficiency plots. Generally, efficiency plots present error versus the computational cost. However in the multirate context, fast and slow function costs can differ dramatically. As such, we separately consider efficiency using total function calls and slow function calls. Since the dominant number of total calls are from the fast function, the "total" plots represent the method efficiency for simulations with comparable fast/slow function cost, whereas the "slow-only" plots represent the method efficiency for simulations in which the slow function calls are significantly more expensive (as explained in Section 5.1 as our original motivation for multirate methods). Individual applications will obviously lie somewhere between these extremes, but we assume that they are typically closer to the "slow-only" results.

Applications scientists traditionally use multirate solvers for one of two reasons. The first category are concerned with simulations of stiff systems, but where they choose to use a subcycled explicit method instead an implicit one for the stiff portion of the problem. Generally, these applications are primarily concerned with selecting h to satisfy stability of the fast time scale (instead of accuracy). The second category consider simulations wherein it is essential to capture the coupling between the slow and fast times scales accurately, since temporal errors at the fast time scale can significantly deteriorate the slow time scale solution; here h is chosen based on accuracy considerations. We therefore separately explore test problems in both of these categories in the Sections 5.5.1 and 5.5.2 below.

To facilitate reproducibility of the results in this section, we have provided an open-source MATLAB implementation of the `MERK3`, `MERK4`, `MERK5` and `MIS-KW3` methods, along with scripts to perform all tests from this section [17].

5.5.1. Category I

As this category of problems is concerned with stability at the fast time scale, we choose a fixed, linearly stable micro time step h, and vary the macro time step H (and similarly, $m = H/h$). To this end, we focus on two stiff applications: a reaction diffusion problem and a stiff version of the brusselator problem.

5.5.1.1. Reaction Diffusion

We consider a reaction diffusion problem with a traveling wave solution similar to the one considered by Savcenco et al. [97],

$$u_t = \frac{1}{100}u_{xx} + u^2(1-u), \qquad 0 < x < 5, \quad 0 < t \leq 3,$$

$$u_x(0,t) = u_x(5,t) = 0, \qquad u(x,0) = (1+e^{\lambda(x-1)})^{-1},$$

where $\lambda = 5\sqrt{2}$. We discretize in space using a second-order accurate central finite difference scheme using 1000 spatial points. This gives us a system for which we take $\mathcal{L}$ and $\mathcal{N}(t,u(t))$ to be the discretized versions of $\frac{1}{100}u_{xx}$ and $u^2(1-u)$ respectively. The micro time step is chosen to satisfy the Courant-Friedrichs-Lewy (CFL) linear stability condition, $h = 10^{-3}$.

In the left of Figure 5.1 we plot the method convergence as H is varied, which shows slighty convergence rates that are better than predicted for all methods tested. As this behavior is not consistently observed for the remaining test problems, we believe that this is an artifact of this particular test problem. Here, we compute the best-fit rates using only the error values larger than $\sim 10^{-13}$, where the error stagnates due to the accuracy of the reference solution.

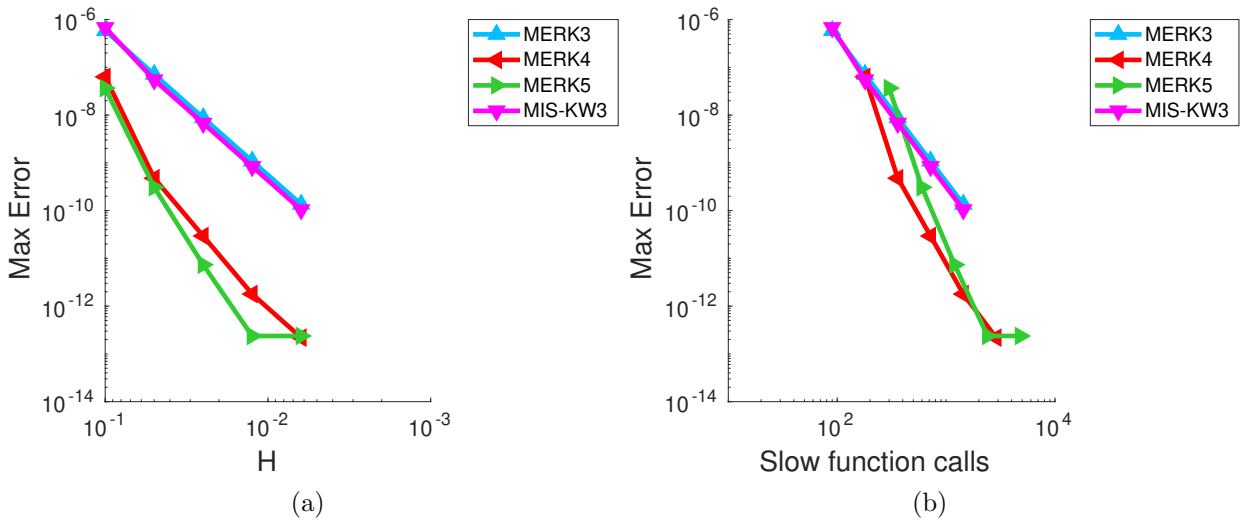


Figure 5.1: Reaction diffusion convergence (left) and "slow-only" efficiency (right). The best fit convergence rates are 3.03, 4.93, 5.71, 3.20 (`MERK3`, `MERK4`, `MERK5`, and `MIS-KW3`, resp.). The most "efficient" methods at a given error are to the left of their less efficient counterparts.

The efficiency plots for both test problems in this category are very similar, so we present the "slow-only" efficiency plot for this problem in the right of Figure 5.1, saving the "total" efficiency plot for the next test. Here, we note that for tolerances larger than 10^{-7}, `MERK3` and `MIS-KW3` are the most efficient, but for tighter tolerances `MERK4` is the best. Although `MERK5` has a higher rate of convergence, the increased cost per step causes it to lag behind until it reaches the reference solution accuracy, where it begins to overtake `MERK4`.

5.5.1.2. Brusselator

The brusselator is an oscillating chemical reaction problem for which one of the reaction products acts as a catalyst. A two-component version of this problem is widely used as a test for ODE solvers [45, (16.12) of Sec. I.16]. We examine a stiff version of this problem

that has been used to test IMEX and multirate methods [102, 79]:

$$\begin{bmatrix} u \\ v \\ w \end{bmatrix}' = \begin{bmatrix} a-(w+1)u+u^2v \\ wu-u^2v \\ \frac{b-w}{\epsilon}-uw \end{bmatrix}, \qquad \mathbf{u}(0) = \begin{bmatrix} 1.2 \\ 3.1 \\ 3 \end{bmatrix},$$

over the interval $t \in (0, 2]$, with parameters $a = 1, b = 3.5$ and $\frac{1}{\epsilon} = 100$. We convert this to have the multirate form (5.1) by defining

$$\mathcal{L} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & \frac{-1}{\epsilon} \end{bmatrix}, \qquad \mathcal{N}(t, \mathbf{u}(t)) = \begin{bmatrix} a-(w+1)u+u^2v \\ wu-u^2v \\ \frac{b}{\epsilon}-uw \end{bmatrix}.$$

In the left of Figure 5.2 we plot the error versus H, and list the corresponding best-fit convergence rates. We observe that all the tested methods perform slightly worse than their predicted convergence rates, which we attribute to order reduction due to the stiffness of the problem; however, the relative convergence rates of each method compare as expected against one another.

For this test problem, we plot the efficiency based on total function calls in the right of Figure 5.2. We note that each curve is almost vertical since the micro time step h is held constant for these tests, and is significantly smaller than H. Here, `MIS-KW3` takes the least amount of total function calls since its structure ensures that it only traverses the time step interval $[t_n, t_n + H]$ once when evaluating the modified ODEs, whereas `MERK3`, `MERK4` and `MERK5` require approximately 2, 3 and 3 traversals, respectively. We note that although these additional traversals of the time step interval $[t_n, t_n + H]$ result in significant increases in the

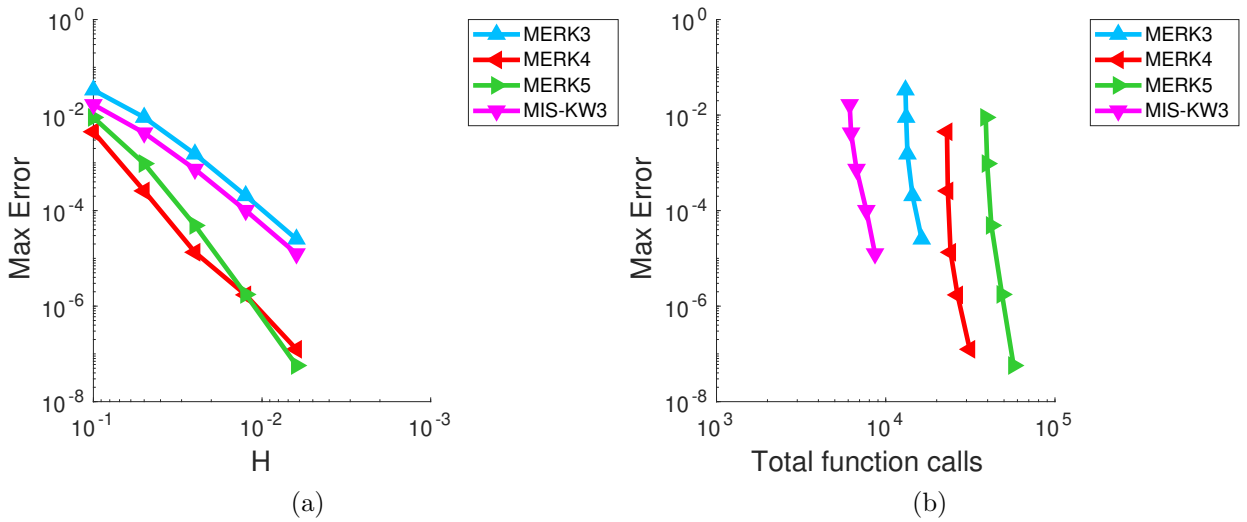


Figure 5.2: Brusselator convergence (left) and "total" efficiency (right). The best fit convergence rates are 2.62, 3.75, 4.36 and 2.61 (`MERK3`, `MERK4`, `MERK5`, and `MIS-KW3`, respectively). Note the near-vertical lines in the efficiency plots, indicating the dominance of "fast" function calls in the estimate of total cost.

number of fast function calls, the number of potentially more costly slow function calls for all methods is equal to the number of slow stages.

5.5.2. Category II

Recalling that our second category of multirate applications focuses on accurately coupling the fast and slow processes, for these test problems we choose a fixed time scale separation factor m for each method/test, and vary H (and proportionally, $h = H/m$). For this group of tests we consider a linear multirate problem from Estep et al. [30] for which the fast variables are coupled into the slow equation (one-directional coupling) and a linear multirate problem of our own design where both the fast and slow variables are coupled (bi-directional coupling). Since the "optimal" value of m for each multirate algorithm is problem-dependent, we describe our approach for determining this m value in Section 5.5.2.1 below.

5.5.2.1. One-directional coupling

We consider a linear system of ODEs [30]:

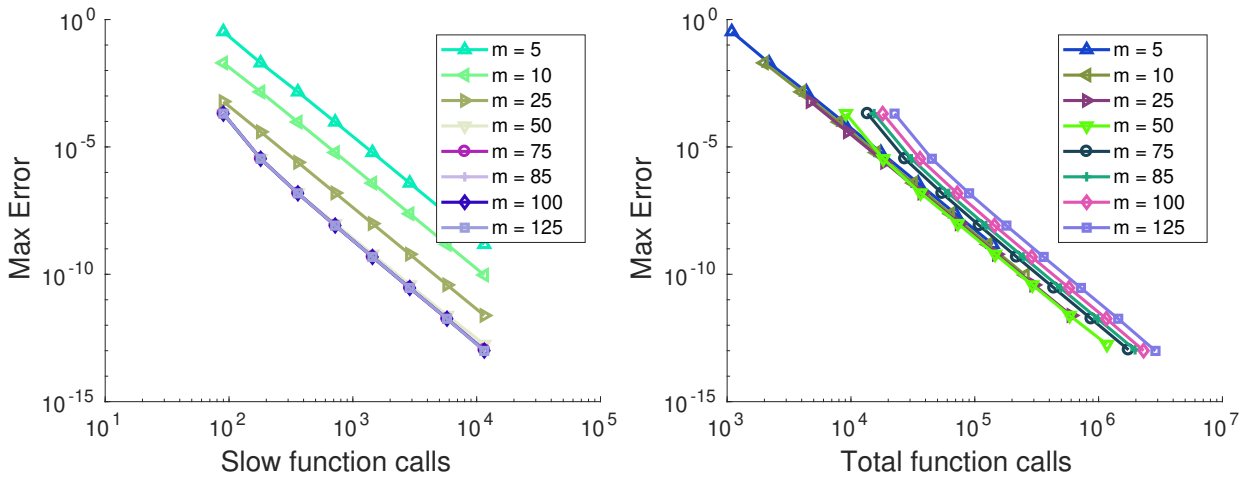
$$\begin{bmatrix} u \\ v \\ w \end{bmatrix}' = \begin{bmatrix} 0 & -50 & 0 \\ 50 & 0 & 0 \\ 1 & 1 & -1 \end{bmatrix} \begin{bmatrix} u \\ v \\ w \end{bmatrix}, \qquad \mathbf{u}(0) = \begin{bmatrix} 1 \\ 0 \\ 2 \end{bmatrix}, \tag{5.56}$$

over the interval $t \in (0, 1]$. This has analytical solution $u(t) = \cos(50t)$, $v(t) = \sin(50t)$, and $w(t) = \frac{5051}{2501}e^{-t} - \frac{49}{2501}\cos(50t) + \frac{51}{2501}\sin(50t)$. We convert this problem to multirate form (5.1) by decomposing it as:

$$\mathcal{L} = \begin{bmatrix} 0 & -50 & 0 \\ 50 & 0 & 0 \\ 1 & 1 & 0 \end{bmatrix}, \qquad \mathcal{N}(t, \mathbf{u}(t)) = \begin{bmatrix} 0 \\ 0 \\ -w \end{bmatrix}.$$

We first discuss our approach in determining the "optimal" time-scale separation factor m. For illustration, we consider `MERK4` on this problem; however, we apply this approach to all methods for both this test and the following bi-directional coupling test in Section 5.5.2.2. We begin by repeatedly solving the problem (5.56) using the multirate method with different factors $m = \{5, 10, 25, 50, 75, 85, 100, 125\}$. For each value of m, we vary H (and hence $h = H/m$). We then analyze the resulting "total" and "slow-only" efficiency plots for each fixed m value, as shown in Figure 5.3.

We first note that both plots show a group of m values with identical efficiency, along with other less efficient results. In Figure 5.3(a), the more efficient group is comprised of *larger* m values, whereas in Figure 5.3(b) the more efficient group has *smaller* m values.



(a) (b)

Figure 5.3: Efficiency plots for `MERK4` applied to the one-directional coupling test, resulting from various m factors.

This is unsurprising, since increases in m for a fixed H correspond to decreases in h, leading to accuracy improvements at the fast time scale alone. While this will results in increased total function calls, the number of slow function calls will remain fixed. We therefore define the "optimal" m as the value where the fast and slow solution errors are balanced. Hence, in Figure 5.3(b) this corresponds to the largest m that remains in the more efficient group, and in Figure 5.3(a) this corresponds to the smallest m that remains in the more efficient group. Inspecting both plots in Figure 5.3, the optimal value for `MERK4` on this problem is $m = 50$. Carrying out a similar process for the other methods on this problem, `MERK3` has an optimal value of $m = 75$, `MERK5` $m = 25$, and `MIS-KW3` has an optimal value of $m = 75$.

Using these m values, In Figure 5.4 we plot the convergence results for the four methods on this problem, confirming the analytical orders of convergence, with errors stagnating around 10^{-13} due to accumulation of floating-point roundoff. While we find slightly better-than-expected convergence rates for the MERK methods, and only the expected rate for `MIS-KW3`, we do not draw conclusions regarding this behavior.

Similarly, in Figure 5.5 we plot both the "total" and "slow-only" efficiency of each method on the one-directional test problem. When measuring only slow function calls, both `MIS-KW3`

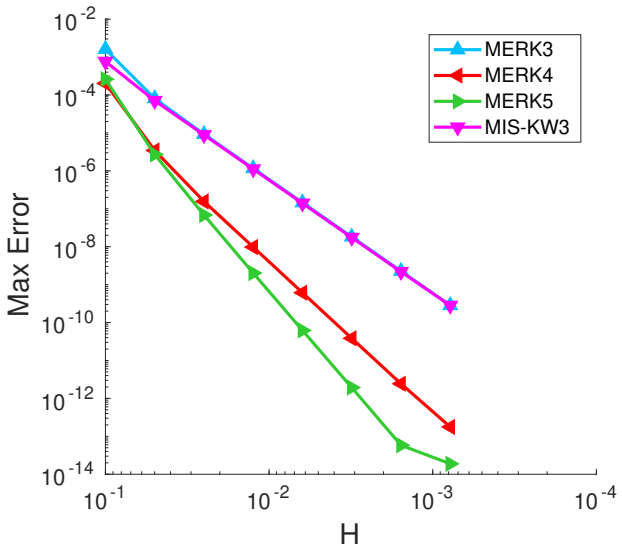


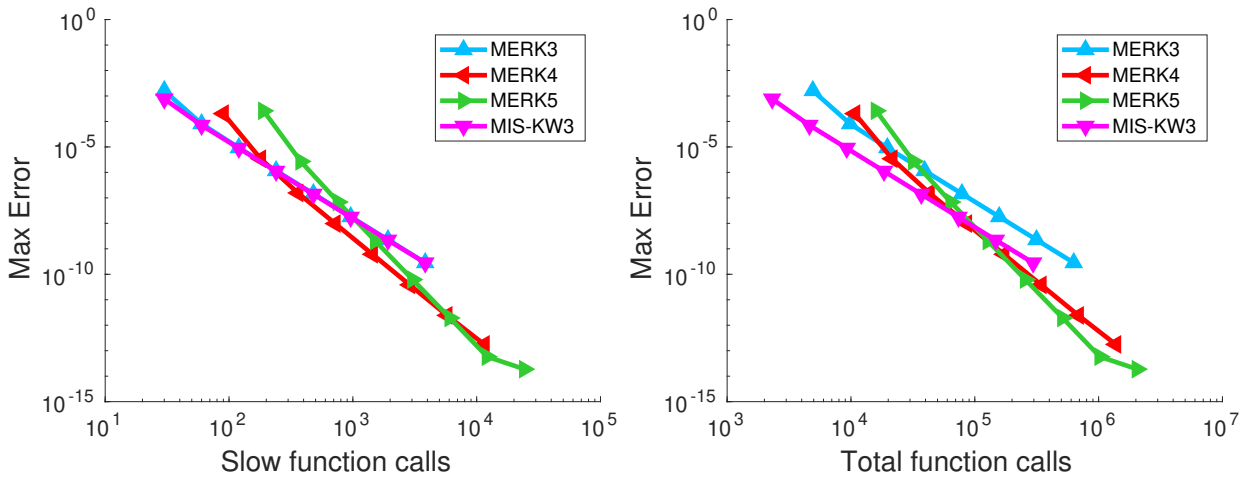
Figure 5.4: One-directional coupling convergence. Best fit convergence rates are 3.16, 4.28, 5.26 and 3.04 (`MERK3`, `MERK4`, `MERK5` and `MIS-KW3`, respectively).

and `MERK3` tie for errors larger than 10^{-6}, `MERK4` is the most efficient for errors between 10^{-6} and 10^{-12} and `MERK5` is the most efficient at the tightest error values. When the fast function calls are given equal weight as the slow, however, `MIS-KW3` is the most efficient at errors larger than 10^{-8}, while `MERK5` is the most efficient at tighter error thresholds.

5.5.2.2. Bi-directional coupling

Taking inspiration from the preceding one-directional test, we designed a problem with coupling between both the fast and slow components to further demonstrate the flexibility and robustness of MERK methods. To this end, we consider the following test problem

$$\begin{bmatrix} u \\ v \\ w \end{bmatrix}' = \begin{bmatrix} 0 & 100 & 1 \\ -100 & 0 & 0 \\ 1 & 0 & -1 \end{bmatrix} \begin{bmatrix} u \\ v \\ w \end{bmatrix}, \qquad \mathbf{u}(0) = \begin{bmatrix} 9001/10001 \\ -100000/10001 \\ 1000 \end{bmatrix}, \tag{5.57}$$



(a) (b)

Figure 5.5: One-directional coupling efficiency. The most efficient method depends on how "cost" is measured, as well as on the desired accuracy.

over $t \in (0, 2]$. Converting to multirate form (5.1), we set $\mathcal{L}$ and $\mathcal{N}(t, \mathbf{u}(t))$ as:

$$\mathcal{L} = \begin{bmatrix} 0 & 100 & 0 \\ -100 & 0 & 0 \\ 1 & 0 & 0 \end{bmatrix}, \qquad \mathcal{N}(t, \mathbf{u}(t)) = \begin{bmatrix} w \\ 0 \\ -w \end{bmatrix}.$$

While this is a linear test problem that may be solved using the matrix exponential, this solution is difficult to represent in closed-form, and so we use a reference solution for convenience. Using the previously-described approach for determining the optimal time-scale separation factor m for each method on this problem, we have $m = 50$ for `MERK3` and `MERK4`, $m = 10$ for `MERK5` and $m = 25$ for `MIS-KW3`.

In Figure 5.6 we plot the convergence rates of each method on this test problem, again confirming the analytical orders of convergence, with errors stagnating around 10^{-12} due to the reference solution accuracy.

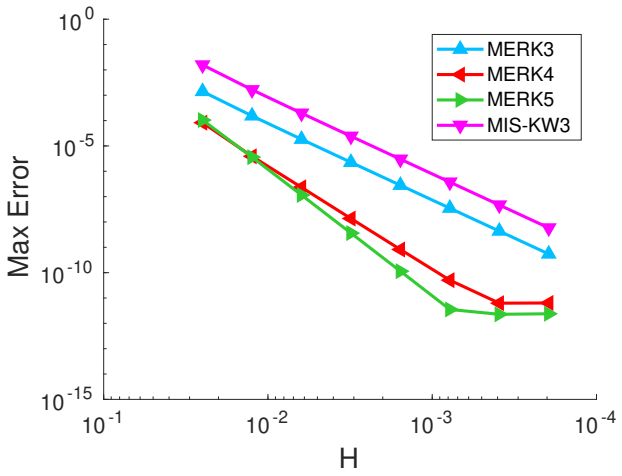


Figure 5.6: Bi-directional coupling convergence. Best fit convergence rates are 3.03, 3.99, 4.97 and 3.06 (`MERK3`, `MERK4`, `MERK5`, `MIS-KW3`, respectively).

Similarly, in Figure 5.7 we plot both the "slow-only" and "total" efficiency plots for this problem. Here, when measuring only the slow function calls, the most efficient method is `MERK3` at error thresholds above 10^{-5}, and `MERK5` for smaller error values. Strikingly, when considering the total number of function calls, the `MERK5` is the most efficient at nearly all error thresholds. We note, however, that the optimal time-scale separation factor for `MERK5` is $m = 10$ for this problem, which results in reduced fast function calls per slow step, and hence an overal reduction in total function calls.

5.5.3. Variations in the fast method

We finish by demonstrating the effects of using inner methods with differing orders of accuracy. Here, we consider only the `MERK` methods, applied to the bi-directional coupling problem (5.57). Here, we vary the order of method applied for computing both the internal stage solutions (5.20), q, and the step solution (5.23), r. Recalling the convergence theory presented in Theorem 5.3.2, a MERK method of order p should use inner methods of orders $q \geq p-1$ and $r \geq p$. However, in these tests we apply other variations on orders to ascertain whether (a) the inner methods could have even lower order and still obtain overall order p,

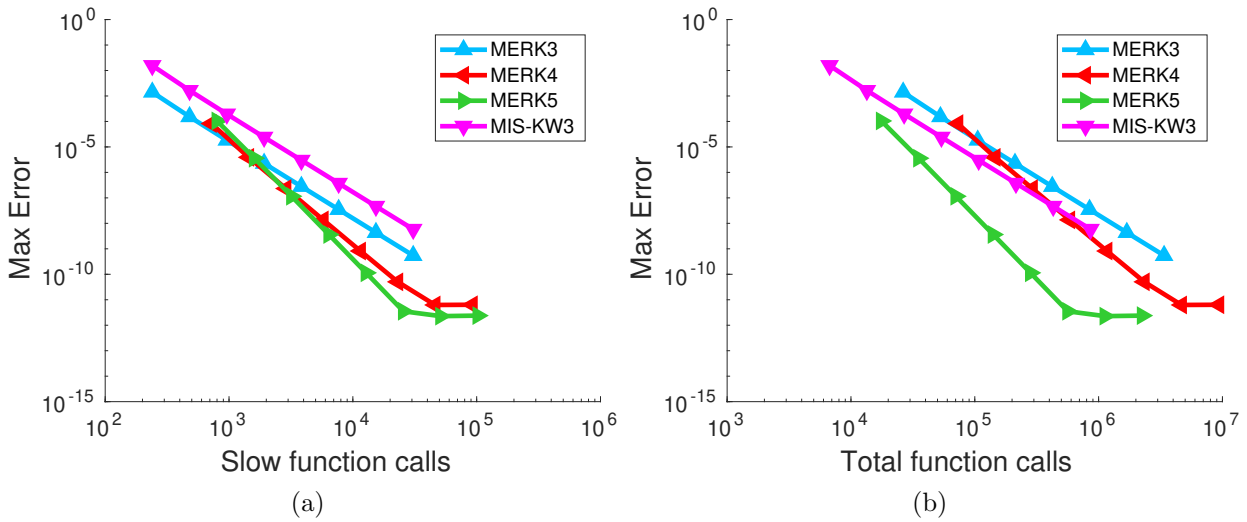


Figure 5.7: Bi-directional coupling efficiency. Again, the most efficient method depends on how "cost" is measured, as well as on the desired accuracy, however `MERK5` demonstrates the best overall performance.

or (b) use of higher-order inner methods can result in overall convergence higher than p. We present the best-fit convergence rates for this ensemble of tests in Table 5.1.

These numerical results show that in fact the inner method order requirements presented in Theorem 5.3.2 are both necessary and sufficient, i.e., the least-expensive combination for attaining a MERK method of order p is to compute stage solutions (5.20) using an inner method of order $p-1$, and the time step solution (5.23) using an inner method of order p. Furthermore, use of higher-order inner methods with orders $q = r > p$ *does not* result in overall order higher than p, due to the first term $C_1 H^p$ in Theorem 5.3.2, that corresponds to the coupling between the fast and slow processes.

5.6. Conclusion

We propose a novel class of multirate methods constructed from explicit exponential Runge–Kutta methods, wherein the action of the matrix exponential is approximated via solution of "fast" initial value problems for each ExpRK stage. Algorithmically, these methods offer a number of desirable properties. Since these are created through defining a set of

MERK3($p=3$)			MERK4($p=4$)			MERK5($p=5$)		
q	r	Observed order	q	r	Observed order	q	r	Observed order
2	2	2.00	3	3	3.01	4	4	4.00
3	2	2.00	4	3	3.01	5	4	4.00
2	3	3.03	3	4	3.99	4	5	4.97
3	3	3.03	4	4	3.99	5	5	4.97
4	4	3.03	5	5	3.99	6	6	4.96

Table 5.1: Convergence rate dependence on inner ODE solvers.

modified IVPs (like (R)MIS and MRI-GARK methods), MERK implementations have near complete freedom in evolving the problem at the fast time scale; however, unlike (R)MIS and MRI-GARK, MERK methods may utilize inner solvers of reduced accuracy for the internal stages. Additionally, since the MERK structure follows directly from ExpRK methods satisfying Theorem 5.3.1, derivation of high-order MERK methods, including versions supporting embeddings for temporal adaptivity, is much simpler than for alternate multirate frameworks. As a result, MERK methods constitute the first multirate algorithms of order five, without requiring deferred correction or extrapolation techniques. Furthermore, the proposed approach may be similarly applied to exponential Rosenbrock methods, allowing for problems where the fast time scale is nonlinear, although such methods are not considered in this work.

In addition to proposing the MERK class of multirate methods and providing rigorous analysis of their convergence, we provide numerical comparisons of the performance of multiple MERK and MIS methods on a variety of multirate test problems. Based on these experiments, we find that the MERK methods indeed exhibit their theoretical orders of convergence, including tests that clearly demonstrate our primary convergence result in Theorem 5.3.2. Furthermore, the proposed methods compare favorably against standard

MIS multirate methods, particularly when increased accuracy is desired and for problems wherein the "slow" right-hand side function is significantly more costly than the "fast."

This work may be extended in numerous ways. As alluded to above, extensions of these approaches to explicit exponential Rosenbrock methods are straightforward, and are already under investigation. Additionally, extensions to higher order will follow from related developments of higher-order exponential methods. Finally, we plan to investigate the use of embeddings at both the fast and slow time scales to perform temporal adaptivity in both H and h for efficient, tolerance-based calculations.

Chapter 6

Multirate exponential Rosenbrock methods

This chapter introduces a newly developed class of MIS-type multirate methods whose structure closely follows that of MERK methods. Starting from exponential Rosenbrock (ExpRB) methods, we derive a new multirating procedure and call these new schemes multirate exponential Rosenbrock (MERB) methods. The outline of this chapter is as follows: first, we give a brief background of ExpRB schemes, present the MERB algorithm and example MERB methods with orders up to six, finally we provide detailed numerical results for MERB methods, comparing them with other MIS-type methods. This chapter is part of a manuscript currently under preparation in collaboration with Vu Thai Luan and Daniel R. Reynolds. Convergence proofs for MERB methods will appear in this manuscript.

6.1. Exponential Rosenbrock schemes

We consider a system of ODEs represented by

$$u'(t) = F(t, u(t)) = F_f(t, u) + F_s(t, u), \quad u(t_0) = u_0, \tag{6.1}$$

where $F(t, u(t))$ represents some vector field that can be split additively into a fast component $F_f(t, u)$ and slow component $F_s(t, u)$ through linearizing the right hand side or otherwise. First, we note that multirate exponential Runge–Kutta (MERK) methods for solving (6.1) were derived in our recent work [68] for the case that the fast time scale involves in a linear operator, i.e., $F_f(t, u) = Lu(t)$ (e.g. diffusion-reaction systems). This case can arise from a prior linearization of (6.1) (e.g., at the initial state u_0), leading to a semi-linear system $u'(t) = Lu(t) + N(t, u(t))$, where $L = \frac{\partial F}{\partial u}(t_0, u_0)$ and $N(t, u) = F(t, u) - Lu = F_s(t, u)$. In

many scenarios, however, (6.1) has a strong nonlinearity even after performing the prior linearization (e.g., $N(t,u)$ is still large). In this situation, the dynamic linearization approach is more appropriate for improving the stability after each integration step. Namely, linearizing the vector field $F(t,u)$ within each time step $[t_n, t_{n+1}]$ around the current numerical solution (t_n, u_n) gives

$$u'(t) = F(t,u(t)) = J_n u(t) + V_n t + N_n(t,u(t)) \tag{6.2}$$

with

$$J_n = \frac{\partial F}{\partial u}(t_n,u_n), \quad V_n = \frac{\partial F}{\partial t}(t_n,u_n), \quad N_n(t,u) = F(t,u) - J_n u - V_n t. \tag{6.3}$$

Now one can represent the exact solution at time $t_{n+1} = t_n + H$ by applying the variation-of-constants formula (or the Duhamel's principle) to (6.2) (see [65]),

$$\begin{aligned} u(t_{n+1}) &= \mathrm{e}^{HJ_n}u(t_n) + \int_0^H \mathrm{e}^{(H-\tau)J_n}\Big(V_n.(t_n+\tau) + N_n(t_n+\tau, u(t_n+\tau))\Big)\mathrm{d}\tau \\ &= \mathrm{e}^{HJ_n}u(t_n) + H\varphi_1(HJ_n)V_n t_n + H^2\varphi_2(HJ_n)V_n \\ &\quad + \int_0^H \mathrm{e}^{(H-\tau)J_n} N_n(t_n+\tau, u(t_n+\tau))\mathrm{d}\tau, \end{aligned} \tag{6.4}$$

where $\varphi_1(Z)$ and $\varphi_2(Z)$ ($Z = HJ_n$) belong to the family of φ-functions given by

$$\varphi_k(Z) = \frac{1}{H^k}\int_0^H \mathrm{e}^{(H-\tau)\frac{Z}{H}}\frac{\tau^{k-1}}{(k-1)!}\mathrm{d}\tau, \quad k \geq 1. \tag{6.5}$$

Next, approximating the integral in (6.4) by using some quadrature rule with nodes c_i in $[0,1]$ and denoting $u_n \approx u(t_n)$ and $U_{n,i} \approx u(t_n + c_i h)$ leads to a general class of linearized exponential integrators, the so-called *exponential Rosenbrock* (ExpRB) methods (see [54, 65, 71]),

$$U_{n,i} = \mathrm{e}^{c_i HJ_n}u_n + c_i H\varphi_1(c_i HJ_n)V_n t_n + c_i^2 H^2\varphi_2(c_i HJ_n)V_n \tag{6.6a}$$

$$+ H \sum_{j=1}^{i-1} a_{ij}(HJ_n) N_n(t_n + c_j H, U_{nj}), \quad i = 1, \ldots, s,$$

$$u_{n+1} = \mathrm{e}^{HJ_n} u_n + H\varphi_1(HJ_n) V_n t_n + H^2 \varphi_2(HJ_n) V_n + H \sum_{i=1}^{s} b_i(HJ_n) N_n(t_n + c_i H, U_{n,i}). \tag{6.6b}$$

Since this is an explicit scheme, one can take $c_1 = 0$ and thus $U_{n1} = u_n$. Here, the weights $a_{ij}(HJ_n)$ and $b_i(HJ_n)$ are usually chosen (by construction) as linear combinations of $\varphi_k(c_i HJ_n)$ and $\varphi_k(HJ_n)$ functions given in (6.5), respectively (this can be also justified by the fact that if one expands $N_n(t_n + \tau, u(t_n + \tau))$ in a Taylor series at $(t_n, u(t_n))$ then the exact solution $u(t_{n+1})$ in (6.4) can be represented as a linear combination of the product of φ_k-functions with vectors). These unknown matrix functions can be determined by solving a system of order conditions, depending on the required order of accuracy (see below).

For an efficient implementation of (6.6), one can reformulate it (see [54]) by using (6.3) and introducing

$$D_{n,i} = N_n(t_n + c_i H, U_{n,i}) - N_n(t_n, u_n), \qquad i = 2, \ldots, s \tag{6.7}$$

(note $D_{n1} = 0$) to obtain an equivalent form

$$\begin{aligned} U_{n,i} &= u_n + c_i H\varphi_1(c_i HJ_n) F(t_n, u_n) + c_i^2 H^2 \varphi_2(c_i HJ_n) V_n + H \sum_{j=2}^{i-1} a_{ij}(HJ_n) D_{nj}, \\ u_{n+1} &= u_n + H\varphi_1(HJ_n) F(t_n, u_n) + H^2 \varphi_2(HJ_n) V_n + H \sum_{i=2}^{s} b_i(HJ_n) D_{n,i}. \end{aligned} \tag{6.8}$$

Remark 6.1.1. *(Order conditions) From [69] the stiff order conditions for ExpRB methods up to order 6 (see Table 6.1). One can see that it requires only 7 conditions for methods of order 6, which is much less than 36 conditions needed for classical or exponential Runge–Kutta methods of the same order. This is the advantage of the dynamic linearization approach*

and can be explained by observing from (6.3) *that*

$$\frac{\partial N_n}{\partial u}(t_n, u_n) = \frac{\partial N_n}{\partial t}(t_n, u_n) = 0, \tag{6.9}$$

which significantly simplifies the number of order conditions (and in turn higher-order schemes can be achieved by using only a few stages). As a direct consequence of (6.9)*, we have from* (6.7) *that* $D_{n,i} = \mathcal{O}(H^2)$*, meaning that ExpRB methods are at least of order 2.*

Table 6.1: Stiff order conditions for exponential Rosenbrock methods up to order 6. Here Z, K, and M denote arbitrary square matrices.

No.	*Order condition*	*Order*
1	$\sum_{i=2}^{s} b_i(Z)c_i^2 = 2\varphi_3(Z)$	*3*
2	$\sum_{i=2}^{s} b_i(Z)c_i^3 = 6\varphi_4(Z)$	*4*
3	$\sum_{i=2}^{s} b_i(Z)c_i^4 = 24\varphi_5(Z)$	*5*
4	$\sum_{i=2}^{s} b_i(Z)c_i K\big(\sum_{k=2}^{i-1} a_{ik}(Z)\frac{c_k^2}{2!} - c_i^3\varphi_3(c_iZ)\big) = 0$	*5*
5	$\sum_{i=2}^{s} b_i(Z)c_i^5 = 120\varphi_6(Z)$	*6*
6	$\sum_{i=2}^{s} b_i(Z)c_i^2 M\big(\sum_{k=2}^{i-1} a_{ik}(Z)\frac{c_k^2}{2!} - c_i^3\varphi_3(c_iZ)\big) = 0$	*6*
7	$\sum_{i=2}^{s} b_i(Z)c_i K\big(\sum_{k=2}^{i-1} a_{ik}(Z)\frac{c_k^3}{3!} - c_i^4\varphi_4(c_iZ)\big) = 0$	*6*

Remark 6.1.2. *(ExpRB methods for autonomous problems). We note that* (6.6) *(and thus* (6.8)*) can be easily applied to autonomous versions of* (6.1)*, i.e.,* $u'(t) = F(u(t))$*, by setting* $V_n = 0$*. In this case,* $N_n(t, u)$ *becomes* $N_n(u) = F(u) - J_n u$*.*

6.2. A multirate procedure for ExpRB methods

Inspired by our recent work [68], we now show that ExpRB schemes can be interpreted as a class of multirate infinitesimal step-type methods (MIS). Namely, we will construct modified differential equations whose exact solutions correspond to the ExpRB internal stages

$U_{n,i}$ $(i = 2, \ldots, s)$, and the final stage u_{n+1}. This can be done by adapting the result of [68, Theorem 3.1] (where we used the idea of backward error analysis [44, Chap. IX] to build modified ODEs for exponential Runge–Kutta schemes) to ExpRB schemes (6.8). In particular, one can directly obtain the following result.

Lemma 6.2.1. *Consider an explicit ExpRB scheme* (6.8)*. Assume that the weights* $a_{ij}(HJ_n)$ *and* $b_i(HJ_n)$ *can be written as linear combinations of* φ_k *functions (given in* (6.5)*), i.e.,*

$$a_{ij}(HJ_n) = \sum_{k=1}^{\ell_{ij}} \alpha_{ij}^{(k)} \varphi_k(c_i HJ_n), \quad b_i(HJ_n) = \sum_{k=1}^{m_i} \beta_i^{(k)} \varphi_k(HJ_n), \tag{6.10}$$

where ℓ_{ij} *and* m_i *are some positive integers. Then,* $U_{n,i}$ *and* u_{n+1} *are the exact solutions of the following (linear) modified differential equations of* (6.4)

$$v'_{n,i}(\tau) = J_n v_{n,i}(\tau) + p_{n,i}(\tau), \qquad v_{n,i}(0) = u_n, \qquad i = 2, \ldots, s, \tag{6.11a}$$

$$v'_n(\tau) = J_n v_n(\tau) + q_n(\tau), \qquad v_n(0) = u_n \tag{6.11b}$$

at the times $\tau = c_i H$ *and* $\tau = H$*, respectively. Here* $p_{n,i}(\tau)$ *and* $q_n(\tau)$ *are polynomials in* τ *given by*

$$p_{n,i}(\tau) = N_n(t_n, u_n) + (t_n + \tau)V_n + \sum_{j=2}^{i-1} \Big(\sum_{k=1}^{\ell_{ij}} \frac{\alpha_{ij}^{(k)}}{c_i^k H^{k-1}(k-1)!} \tau^{k-1} \Big) D_{nj}, \tag{6.12a}$$

$$q_n(\tau) = N_n(t_n, u_n) + (t_n + \tau)V_n + \sum_{i=2}^{s} \Big(\sum_{k=1}^{m_i} \frac{\beta_i^{(k)}}{H^{k-1}(k-1)!} \tau^{k-1} \Big) D_{n,i}. \tag{6.12b}$$

Proof. The proof can be carried out in a very similar way as done in [68, Theorem 3.1]. Here, we only sketch the main idea. First, we insert the φ_k functions in (6.5) into (6.10) to

get the integral representations of $a_{ij}(HJ_n)$ and $b_i(HJ_n)$:

$$a_{ij}(HJ_n) = \int_0^{c_iH} \mathrm{e}^{(c_iH-\tau)J_n} \sum_{k=1}^{\ell_{ij}} \frac{\alpha_{ij}^{(k)}}{(c_iH)^k(k-1)!}\tau^{k-1}\mathrm{d}\tau, \tag{6.13a}$$

$$b_i(HJ_n) = \int_0^{H} \mathrm{e}^{(H-\tau)J_n} \sum_{k=1}^{m_i} \frac{\beta_{i}^{(k)}}{H^k(k-1)!}\tau^{k-1}\mathrm{d}\tau. \tag{6.13b}$$

Then inserting these into (6.8) shows that

$$U_{n,i} = \mathrm{e}^{c_iHJ_n}u_n + \int_0^{c_iH} \mathrm{e}^{(c_iH-\tau)J_n}p_{n,i}(\tau)\mathrm{d}\tau, \quad i = 2,\ldots,s, \tag{6.14a}$$

$$u_{n+1} = \mathrm{e}^{HJ_n}u_n + \int_0^{H} \mathrm{e}^{(H-\tau)J_n}q_n(\tau)\mathrm{d}\tau, \tag{6.14b}$$

which clearly show that $U_{n,i} = v_{n,i}(c_iH)$ and $u_{n+1} = v_n(H)$ by means of the variation-of-constants formula applied to (6.11a) and (6.11b), respectively.

□

MERB methods. Starting from the initial value $u_0 = u(t_0)$, Lemma 6.2.1 suggests a multirate procedure to approximate the numerical solution u_{n+1} $(n = 0, 1, 2, \ldots)$ obtained by ExpRB methods. Specifically, one may integrate the slow process using a macro time step H and integrate the fast process using a micro time step $h = H/m$ (where $m > 1$ is an integer representing the time-scale separation factor) via solving the 'fast' ODEs (6.11a) on $[0, c_iH]$ and (6.11b) on $[0, H]$. By denoting the corresponding numerical solutions of these ODEs by $\widehat{U}_{n,i}$ $(\approx v_{n,i}(c_iH) = U_{n,i})$ and $\hat{u}_{n+1}$ $(\approx v_n(H) = u_{n+1})$, one can practically formulate this multirate procedure in each step by solving (6.11)–(6.12) with the initial value $\hat{u}_n$ $(\hat{u}_0 = u_0)$ (and thus all the corresponding $J_n, V_n, N_n(t,u)$ in (6.3) and $D_{n,i}$ in (6.7) must be evaluated at $(t_n, \hat{u}_n)$ to update the polynomials in (6.12)).

Namely, starting with $\hat{u}_0 = u_0$, we solve the following perturbed linear ODEs for $i = 2, \dots, s$:

$$y'_{n,i}(\tau) = \hat{J}_n y_{n,i}(\tau) + \hat{p}_{n,i}(\tau), \quad y_{n,i}(0) = \hat{u}_n, \text{ on } [0, c_i H] \tag{6.15}$$

with

$$\hat{p}_{n,i}(\tau) = N_n(t_n, \hat{u}_n) + (t_n + \tau)\hat{V}_n + \sum_{j=2}^{i-1} \Big(\sum_{k=1}^{\ell_{ij}} \frac{\alpha_{ij}^{(k)}}{c_i^k H^{k-1}(k-1)!} \tau^{k-1} \Big) \widehat{D}_{n,j} \tag{6.16}$$

$$\left(\text{here } \hat{J}_n = \frac{\partial F}{\partial u}(t_n, \hat{u}_n),\ \hat{V}_n = \frac{\partial F}{\partial t}(t_n, \hat{u}_n),\ \widehat{D}_{n,j} = \hat{N}_n(t_n + c_j H, \widehat{U}_{nj}) - \hat{N}_n(t_n, \hat{u}_n) \right)$$

to obtain

$$\widehat{U}_{n,i} \approx y_{n,i}(c_i H) \approx v_{n,i}(c_i H) = U_{n,i}.$$

Then, using these approximations, we find

$$\hat{q}_n(\tau) = N_n(t_n, \hat{u}_n) + (t_n + \tau)\hat{V}_n + \sum_{i=2}^{s} \Big(\sum_{k=1}^{m_i} \frac{\beta_i^{(k)}}{H^{k-1}(k-1)!} \tau^{k-1} \Big) \widehat{D}_{n,i} \tag{6.17}$$

and solve one additional linear ODE

$$y'_n(\tau) = \hat{J}_n y_n(\tau) + \hat{q}_n(\tau), \quad y_n(0) = \hat{u}_n \quad \text{on } [0, H] \tag{6.18}$$

to obtain the update

$$\hat{u}_{n+1} \approx y_n(H) \approx v_n(H) = u_{n+1}.$$

Since this process can be derived from ExpRB schemes satisfying (6.10), the resulting methods (6.15)–(6.18) will be henceforth called *Multirate Exponential Rosenbrock (MERB)* methods.

Remark 6.2.1. *(A comparison with MERK methods). In view of MERB methods* (6.15)–(6.18)*, one can see that, in each step, they have similar structure as MERK methods [68]. Hence, they can retain MERK's interesting features (such as using very few evaluations of*

the slow components and thus be more efficient when the slow components are more expensive to compute than fast components; also they do not require computing matrix functions as ExpRB methods do). The main difference, however, is that, in order to proceed with the next integration step, MERB methods require updating the coefficient matrix of the linear part (i.e., the Jacobian $\hat{J}_n$) and thus their corresponding polynomials result from the dynamic linearization approach. An advantage of MERB methods over MERK methods due to the property (6.9)*, we expect that the number of fast ODEs needed for constructing MERB methods will be less than for MERK methods of the same order (see Section 6.3).*

6.2.1. MERB algorithm

In Algorithm 2 we provide a precise description of the MERB algorithm. We note that

- **Input:** F; $J_n(t,u)$; $V_n(t,u)$; t_0; u_0; s; c_i $(i = 1, \ldots, s)$; H
- **Initialization:** Set $n = 0$; $\hat{u}_n = u_0$.
  While $t_n < T$
  1. Set $\widehat{U}_{n,1} = \hat{u}_n$.
  2. For $i = 2, \ldots, s$ do
     (a) Find $\hat{p}_{n,i}(\tau)$ as in (6.16).
     (b) Solve (6.15) on $[0, c_i H]$ to obtain $\widehat{U}_{n,i} \approx y_{n,i}(c_i H)$.
  3. Find $\hat{q}_{n,s}(\tau)$ as in (6.17)
  4. Solve (6.18) on $[0, H]$ to get $\hat{u}_{n+1} \approx y_n(H)$.
  5. Update $t_{n+1} := t_n + H$, $n := n + 1$.
- **Output:** Approximate values $\hat{u}_n \approx u_n, n = 1, 2, \ldots$ (where u_n is the numerical solution at time t_n obtained by an ExpRB method).

Algorithm 2: MERB method

in our implementations of MERB methods, we have found it beneficial to include formulas for $N_n(t,u)$ and $D_{n,i}(t,u)$ as additional inputs to the algorithm (provided they can be pre-computed) for use in equations (6.16) and (6.17) to avoid floating-point cancellation errors.

6.3. Construction of specific MERB methods

We derive MERB methods up to order 6. Here we only provide the polynomials $\hat{p}_{n,i}(\tau)$ and $\hat{q}_n(\tau)$ that characterize each MERB method in nonautonomous form. In general, MERB methods result in fewer modified ODEs to be solved per slow time step compared with MERK methods.

6.3.1. Second-order methods

A second-order MERB method only requires the solution of one modified ODE, with

$$\hat{q}_n(\tau) = N_n(t_n, \hat{u}_n) + (t_n + \tau)V_n, \quad \tau \in [0, H]. \tag{6.19}$$

We do not include this method in any of our numerical experiments.

6.3.2. Third-order methods

Our third-order method is called MERB3 and involves the solution of two modified ODEs per slow time step, with polynomials

$$\hat{p}_{n,2}(\tau) = N_n(t_n, \hat{u}_n) + (t_n + \tau)V_n, \quad \tau \in [0, c_2 H], \tag{6.20}$$

$$\hat{q}_n(\tau) = N_n(t_n, \hat{u}_n) + (t_n + \tau)V_n + \frac{\tau^2}{c_2^2 H^2}\widehat{D}_{n2}, \quad \tau \in [0, H]. \tag{6.21}$$

In our numerical experiments we chose $c_2 = \frac{1}{2}$. The total fast time step traversal for MERB3 is $\frac{3}{2}H$, i.e., we must traverse the fast time scale for a total effective interval that is 50% larger than the overall slow step size.

6.3.3. Fourth-order method

MERB4 is our fourth-order method, that also involves only two modified ODEs per slow time step:

$$\hat{p}_{n,2}(\tau) = N_n(t_n, \hat{u}_n) + (t_n + \tau)V_n, \quad \tau \in [0, \frac{3}{4}H] \tag{6.22}$$

$$\hat{q}_n(\tau) = N_n(t_n, \hat{u}_n) + (t_n + \tau)V_n + \frac{16}{9}\frac{\tau^2}{H^2}\widehat{D}_{n2}, \quad \tau \in [0, H]. \tag{6.23}$$

The total fast traversal time for MERB4 is $\frac{7}{4}H$.

6.3.4. Fifth-order methods

We only require 3 modified ODEs to define our fifth-order method, MERB5:

$$\hat{p}_{n,2}(\tau) = N_n(t_n, \hat{u}_n) + (t_n + \tau)V_n; \quad \tau \in [0, c_2 H] \tag{6.24}$$

$$\hat{p}_{n,3}(\tau) = \hat{p}_{n,4}(\tau) = N_n(t_n, \hat{u}_n) + (t_n + \tau)V_n + \Big(\frac{\tau}{c_2 H}\Big)^2 \widehat{D}_{n2}; \quad \tau \in [0, c_3 H] \tag{6.25}$$

$$\hat{q}_n(\tau) = N_n(t_n, \hat{u}_n) + (t_n + \tau)V_n + \frac{\tau^2}{H^2}\Big(\frac{c_4}{c_3^2(c_4 - c_3)}\widehat{D}_{n3} + \frac{c_3}{c_4^2(c_3 - c_4)}\widehat{D}_{n4}\Big) + \tag{6.26}$$

$$\frac{\tau^3}{H^3}\Big(\frac{-1}{c_3^2(c_4 - c_3)}\widehat{D}_{n3} - \frac{1}{c_4^2(c_3 - c_4)}\widehat{D}_{n4}\Big); \quad \tau \in [0, H].$$

Here we have the condition that $c_4 < c_3$, with $c_4 = \frac{15c_3 - 12}{20c_3 - 15}$. In our experiments we pick $c_2 = c_4 = \frac{1}{4}$ and $c_3 = \frac{33}{40}$. Because $p_{n,3} = p_{n,4}$, a computationally efficient way of implementing MERB5 is to solve for both $\widehat{U}_{n,3}$ and $\widehat{U}_{n,4}$ simultaneously on the interval $[0, c_3 H]$, without solving an additional fast ODE on the shorter interval $[0, c_4 H]$. With this strategy, the total fast traversal for MERB5 is $(1 + c_2 + c_3)H = \frac{83}{40}$.

6.3.5. Sixth-order methods

Here we present the first-ever sixth-order MIS-type multirate method, MERB6, that requires only 3 modified ODEs per each slow time step, which makes it incredibly efficient. The corresponding polynomials are

$$\hat{p}_{n,2}(\tau) = \hat{p}_{n,3}(\tau) = N_n(t_n, \hat{u}_n) + (t_n + \tau)V_n; \quad \tau \in [0, c_2 H] \tag{6.27}$$

$$\hat{p}_{n,4}(\tau) \equiv \hat{p}_{n,5}(\tau) = \hat{p}_{n,6}(\tau) = \hat{p}_{n,7}(\tau) =$$

$$N_n(t_n, \hat{u}_n) + (t_n + \tau)V_n + \frac{\tau^2}{H^2}\Big(\frac{c_3}{c_2^2(c_3 - c_2)}\widehat{D}_{n2} + \frac{c_2}{c_3^2(c_2 - c_3)}\widehat{D}_{n3}\Big)$$

$$+ \frac{\tau^3}{H^3}\Big(\frac{-1}{c_2^2(c_3 - c_2)}\widehat{D}_{n2} - \frac{1}{c_3^2(c_2 - c_3)}\widehat{D}_{n3}\Big); \quad \tau \in [0, c_4 H] \tag{6.28}$$

$$\hat{q}_n(\tau) = N_n(t_n, \hat{u}_n) + (t_n + \tau)V_n - \frac{\tau^2}{H^2}\big(\sum_{i=4}^{7}\hat{\alpha}_i\widehat{D}_{n,i}\big) + \frac{\tau^3}{H^3}\big(\sum_{i=4}^{7}\hat{\eta}_i\widehat{D}_{n,i}\big) - \frac{\tau^4}{H^4}\big(\sum_{i=4}^{7}\hat{\beta}_i\widehat{D}_{n,i}\big)$$

$$+ \frac{\tau^5}{H^5}\big(\sum_{i=4}^{7}\hat{\gamma}_i\widehat{D}_{n,i}\big), \quad \tau \in [0, H] \tag{6.29}$$

where $\hat{\gamma}_i,\ \hat{\alpha}_i,\ \hat{\beta}_i,\ \hat{\eta}_i$ $(i = 4, 5, 6, 7)$ are given by

$$\hat{\gamma}_i = \frac{1}{c_i^2(c_i - c_k)(c_i - c_l)(c_i - c_m)} \tag{6.30}$$

$$\hat{\alpha}_i = c_k c_l c_m \hat{\gamma}_i \tag{6.31}$$

$$\hat{\beta}_i = (c_k + c_l + c_m)\hat{\gamma}_i \tag{6.32}$$

$$\hat{\eta}_i = (c_k c_l + c_l c_m + c_k c_m)\hat{\gamma}_i. \tag{6.33}$$

(note that $i, k, l, m \in \{4, 5, 6, 7\}$ are distinct indices and that c_i, c_k, c_l, c_m are distinct (positive) nodes). Here we have the conditions $c_3 < c_2$ and $c_5, c_6, c_7 < c_4$. We choose $c_3 = c_5 =$

$\frac{1}{10} < c_2 = c_6 = \frac{1}{9} < c_7 = \frac{1}{8} < c_4 = \frac{1}{7}$ for our numerical experiments. This gives a total fast traversal time of $(1 + c_2 + c_4)H = \frac{79}{63}H$.

6.4. Numerical Experiments

In this section we implement MERB methods on multirate test problems to demonstrate convergence rates and computational costs. We first discuss choices we make for the inner fast integrators, fast-slow splitting, optimal time scale separation factors, and give a general overview of how we perform error and efficiency analysis. We then present numerical experiments for two multirate ODE systems. The first ODE system arises from a spatial discretization of a reaction-diffusion problem and the second is a semi-linear nonautonomous system with coupling between the fast and slow variables, we call this the bidirectional coupling problem. For each test problem we implement MERB3, MERB4, MERB5, and MERB6, then we compare them with implementations of other recently developed multirate methods that treat the slow time scale explicitly: MERK3, MERK4, and MERK5 from [68], plus MRI-GARK-ERK33a and MRI-GARK-ERK45a from [88], written here in short form as MRI-GARK33a and MRI-GARK45a.

6.4.1. Choice of inner integrators

For uniformity, in our implementations of MERB, MERK, and MRI-GARK methods of the same order, we use the same fast integrators for the internal stages and step solution. Third-order methods use a 3 stage explicit third-order method from equation (233f) of [10], fourth-order methods use a 4 stage explicit fourth-order method commonly known as "RK4" from [62], fifth-order methods use an 8 stage fifth-order method which is the explicit part of ARK5(4)8L[2]SA from [57], while the sixth-order method uses an 8 stage explicit sixth-order method based on the 8,5(6) procedure of [106]. We note that like with MERK methods, for a MERB method of order q, we only require that the internal stages are computed with an order $q-1$ integrator, along with an integrator of order q to compute the step solution.

6.4.2. Fast-slow splitting

The splitting into fast and slow components for MERB methods is dictated by the dynamic linearization process at each time step. This brings about interesting questions concerning the comparison process with MERK and MRI-GARK methods that do not necessarily require dynamic linearization. MERK methods require a linear fast component but MRI-GARK methods do not have constraints on what constitutes the fast and the slow time scale. With these multirate splittings in mind, we endeavor to present results that provide a fuller picture of the competitiveness of MERB methods in comparison with MERK and MRI-GARK. We consider two separate fast-slow splittings, each having its own merits. The first is dynamic linearization which involves the linearization of the right hand side, therefore requiring a Jacobian computation at each time step. The fast component then becomes the linear portion and the slow component is the remainder. The dynamic linearization process can place more of the problem at the fast time scale than other fixed multirate splittings, and depending on how much the fast problem is subcycled, can lead to better overall accuracy. Using MERK methods with the dynamic linearization approach demonstrates their applicability to nonlinear systems.

Our other fast-slow splitting defines a fixed linear portion of the test problem as the fast component and the rest as the slow component, in the ensuing results we call this 'fixed linearization'. Though the motivation for this splitting is from the MERK requirements of a linear fast component, we also apply it to MRI-GARK methods. We note that other fixed splittings that can offer different accuracy and efficiency insights on multirate methods are possible, especially for our bidirectional coupling problem. We however only focus on one fixed splitting for each test problem. Methods that use fixed linearization are denoted with an asterisk in our results, for example MERK3* uses a fixed linearization while MERK3 uses dynamic linearization.

6.4.3. Optimal time scale separation

In order to compare methods at their peak performance, we strive to determine an optimal time scale separation factor for each multirate method on each test problem. The optimal time scale separation factor $m = H/h$ is the integer ratio between the slow and fast time step sizes which is most efficient. This value is determined experimentally by comparing efficiency in terms of slow-only evaluations and total (slow+fast) evaluations for several different values of m. Keeping the values of H constant, as we increase m, h becomes smaller and the fast ODE is solved more accurately. The optimal m is the value where we stop seeing improvement in the overall solution error due to more accurate fast evolution. Thus any larger m would result in the same errors as the optimal value, but at a greater cost in terms of fast function evaluations, hence total function evaluations. A more in depth illustration and discussion of this process can be found in [68].

Method	Reaction-diffusion m		Bidirectional coupling m	
	Dynamic	Fixed	Dynamic	Fixed
MERB3	10		80	
MERK3	20	10	80	10
MRI-GARK33a	20	5	80	10
MERB4	10		40	
MERK4	20	10	40	10
MRI-GARK45a	10	1	40	1
MERB5	5		10	
MERK5	5	5	10	10
MERB6	5		5	

Table 6.2: Optimal time scale separation factor for each test problem, each splitting, and each method.

Table 6.2 presents the optimal m values for each method and each test problem depending on splitting type. A trend emerges among MERK and MRI-GARK methods that use both dynamic and fixed linearization; for these methods, dynamic linearization almost exclusively results in the need for a larger m value than the one required by fixed linearization. This is likely because in both test problems, dynamic linearization results in a fast-slow splitting where more of the problem is included within the fast dynamics, so there is need for a larger m value to resolve the dynamics. We also note that for the fixed linearization, both MRI-GARK methods have smaller optimal values of m, suggesting an efficient solve for the MRI-GARK fast problem with larger values of h.

6.4.4. Presentation of results

For each test problem we group our numerical results by order of convergence. We have 3 groups corresponding to $\mathcal{O}(H^3)$ methods, $\mathcal{O}(H^4)$ methods, and $\mathcal{O}(H^5)$ combined with $\mathcal{O}(H^6)$ methods. In each group we provide four kinds of "log-log" plots, one convergence plot (solution error versus slow time step size H) and three efficiency plots featuring solution error versus each of slow function calls, total function calls, and MATLAB runtimes. Errors given are maximum absolute errors over all spatial grid points and time outputs measured against an analytical solution or highly accurate numerically determined reference solution. We also provide rates of convergence computed from a least squares error fit on the error versus H data, neglecting points at the reference error floor. Each of our three efficiency measurements tells a different story. First, we consider slow function calls to illustrate the costs of each multirate method especially when dealing with systems with expensive slow components. Second, we consider total function calls to capture the fast evaluations and highlight properties of methods related to their total traversal times. Lastly, though MATLAB runtimes are often not the best representatives for runtimes on HPC applications, we use them here to possibly capture the costs added by Jacobian computations in dynamic linearization and measure how they affect efficiency.

6.4.5. Reaction-diffusion

From Savcenco et al.[97], we consider the reaction-diffusion equation:

$$u_t = \epsilon u_{xx} + \gamma u^2(1-u), \quad 0 < x < 5, \quad 0 < t \leq 5, \tag{6.34}$$

where $\lambda = \frac{1}{2}\sqrt{2\gamma/\epsilon}$. We use initial conditions $u_x(0,t) = u_x(5,t) = 0$, and boundary conditions $u(x,0) = (1+\exp(\lambda(x-1)))^{-1}$. Multiple combinations of γ and ϵ are possible but we specifically chose $\gamma = 0.1$ and $\epsilon = 0.01$ so the problem exhibits an optimal $m > 1$ for MERB methods. We use a second-order centered finite difference scheme with 101 spatial grid points to discretize the diffusion term. In addition to dynamic linearization, MERK and MRI-GARK methods also use a fixed splitting where the fast component is $F_f(t,u) = \epsilon u_{xx}$ and the slow component is $F_s(t,u) = \gamma u^2(1-u)$. We consider the solution at 10 different points within the interval and slow time steps $H = 0.5 \times 2^{-k}$, for $k = 0, \ldots, 6$. Since we do not have an analytical solution for this test problem, we compare numerical solutions against a reference solution obtained using MATLAB `ode15s` with relative tolerance 10^{-13} and absolute tolerance 10^{-14}.

Figures 6.1 - 6.3 show accuracy and efficiency results for the reaction-diffusion problem. For our convergence discussion on third-order methods, we focus on Figure 6.1 (top-left) and the legend. For the rates of convergence listed in the legend of Figure 6.1, each third-order method attains the expected order of convergence on this test problem. The observed errors for the dynamic linearization approach on all methods are less than for fixed linearization. This can be attributed to a larger fast portion in the case of dynamic linearization which results in higher optimal time scale separation factors (as shown in Table 6.2) and lower errors. Among the methods that apply dynamic linearization, MERK3 and MRI-GARK33a have almost identical errors that are lower than those for MERB3 which uses an m two times smaller. MERK3* and MRI-GARK33a* have the largest errors on this test problem.

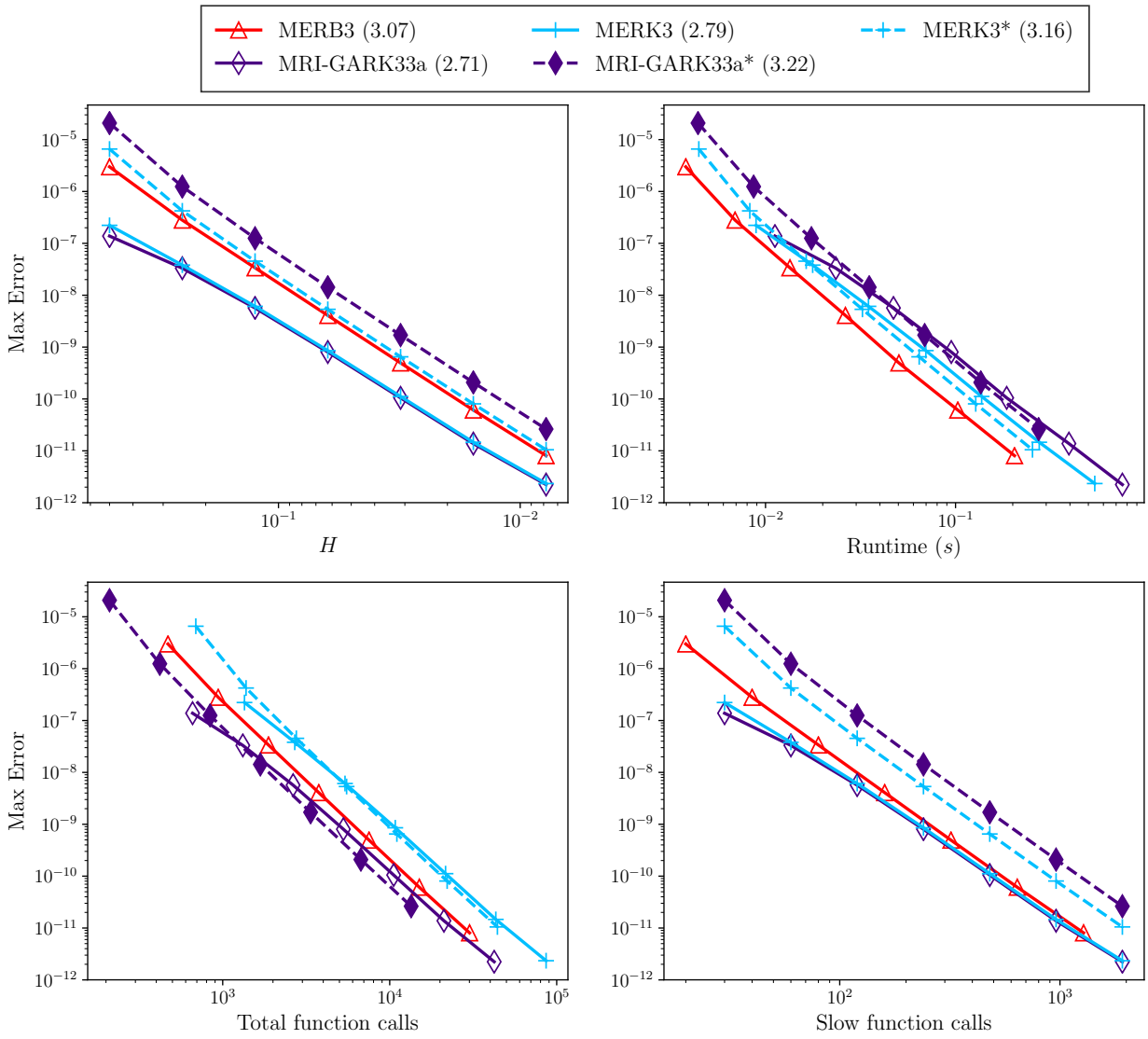


Figure 6.1: Convergence (top-left) and efficiency (top-right, bottom) for $\mathcal{O}(H^3)$ methods on the reaction-diffusion problem of section 6.4.5. MERK3* and MRI-GARK33a* use fixed linearization while the rest of the methods use dynamic linearization. The legend shows the slopes of the least-squares error fit of max error versus H data. It is clear that the most efficient method for this problem depends heavily one the definition of 'cost': MERB3 has the best runtime efficiency, MRI-GARK33a has the best total function call efficiency, and all of three methods utilizing dynamic linearization have the best slow function call efficiency.

Next we discuss efficiency focusing on Figure 6.1 (top-right and bottom) plots. The most efficient methods in each of these plots are closest to the bottom left corner. For our MATLAB implementations, MERB3 has an obvious advantage in terms of runtimes, while both MRI-GARK33a and MRI-GARK33a* have the least efficient implementation. Taking into account only MERK and MRI-GARK methods, there is not much of a difference in runtimes between the dynamic linearization approach and the fixed linearization approach, though the fixed linearization does correspond to slightly less runtime. When looking at total function calls, both MRI-GARK33a and MRI-GARK33a* are the most efficient of the group, largely owing to the property that they only traverse the time step H once, while MERB3 has a total traversal time of $1.5H$, and MERK3 has a total traversal time of $2.166H$ and therefore has the most total function calls. The behavior of multirate methods in terms of slow function calls, Figure 6.1 (bottom-right) is closely aligned with the convergence behavior Figure 6.1 (top-left). At large values of H, MERK3 and MRI-GARK33a are most efficient, but MERB3 is just as efficient as MERK3 and MRI-GARK33a at smaller H values.

For fourth-order methods, we observe in Figure 6.2 (top-left) and legend that all methods reach at least their expected order of convergence, with MERK4* and MRI-GARK45a* performing better than expected. MERK4 has the least errors, but also uses an m value that is two times greater than other fourth-order methods on this test problem as shown in Table 6.2. MERK4* starts off with larger errors than MERB4 and MRI-GARK45a, but because it converges at fifth-order for this test problem, its errors quickly drop below those for MERB4 and MRI-GARK45a. MRI-GARK45a* has an $m = 1$ which seemingly puts it at a disadvantage when comparing accuracy with other methods, however, larger values of m only lead to more total function evaluations and not reduction in errors.

To compare efficiency, we focus on Figure 6.2 (top-right and bottom) plots. From the runtime plot, MERB4 is more efficient at larger values of H, but MERK4 is eventually the most efficient at smaller values of H. Meanwhile, the lessons we learned from third-order methods are largely repeated in the plot for total function calls - Figure 6.2 (bottom-left).

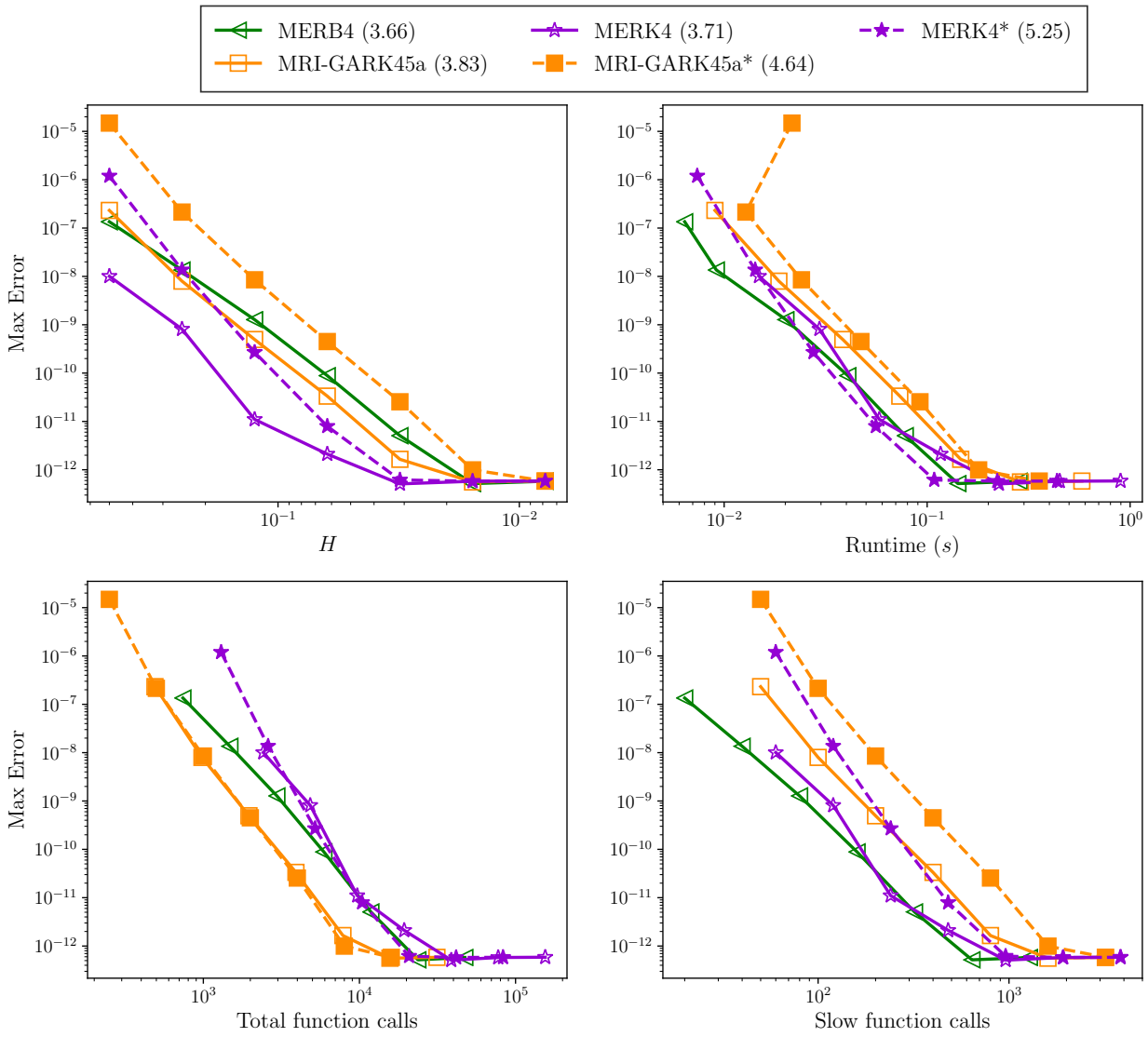


Figure 6.2: Convergence (top-left) and efficiency (top-right, bottom) for $\mathcal{O}(H^4)$ methods on reaction-diffusion problem of section 6.4.5. MERK4* and MRI-GARK45a* use fixed linearization while the rest of the methods use dynamic linearization. The legend shows the slopes of the least-squares error fit of max error versus H data. MERB and MERK methods are have the best efficiency in terms of runtime and slow function calls. MRI-GARK methods have the best efficiency for total function calls.

MRI-GARK45a and MRI-GARK45a* are the most efficient in terms of total function calls and closely line up, MERB4 which has a total traversal time of $1.75H$ performs better than MERK4 and MERK4* with total traversal times of $2.833H$. Finally, comparing slow function calls in Figure 6.2 (bottom-right), MERB4 is the most efficient. This is expected since MERB4 only has 2 slow stages, compared with 6 for MERK4 and 5 for MRI-GARK45a.

The first thing to note in the discussion of fifth and sixth-order methods is that they all use the same $m = 5$ (Table 6.2). All methods converge at their expected rates as shown in the legend for Figure 6.3. The fifth-order methods: MERB5, MERK5, and MERK5*, all cluster around similar error values in Figure 6.3 (top-left), but MERB6 has the largest errors. As with previous observations above, MERK5 which does dynamic linearization has slightly less error than MERK5* which does fixed linearization.

In all efficiency plots, Figure 6.3 (top-right and bottom), MERB5 is the most efficient. Looking at the total function calls plot, MERB5 has a total traversal time of $2.075H$ compared to $3.2H$ for MERK5 and $1.253H$ for MERB6 (though we barely get to see advantages of this since MERB6 has the largest errors). When it comes to slow function calls, Figure 6.3 (bottom-right), MERB5 has 4 slow stages which is much lower than the 10 stages for MERK5, and 7 stages for MERB6. Combining the merits of MERB5 from total function calls and slow function calls explains the runtimes in Figure 6.3 (top-right).

6.4.6. Bidirectional coupling system

In this section we discuss a test problem of our own making that is inspired by the numerical example in Section 5.1 of Estep et al. [30]. We consider the following system on $0 < t \leq 1$:

$$u' = \sigma v - w - \beta t, \tag{6.35}$$

$$v' = -\sigma u,$$

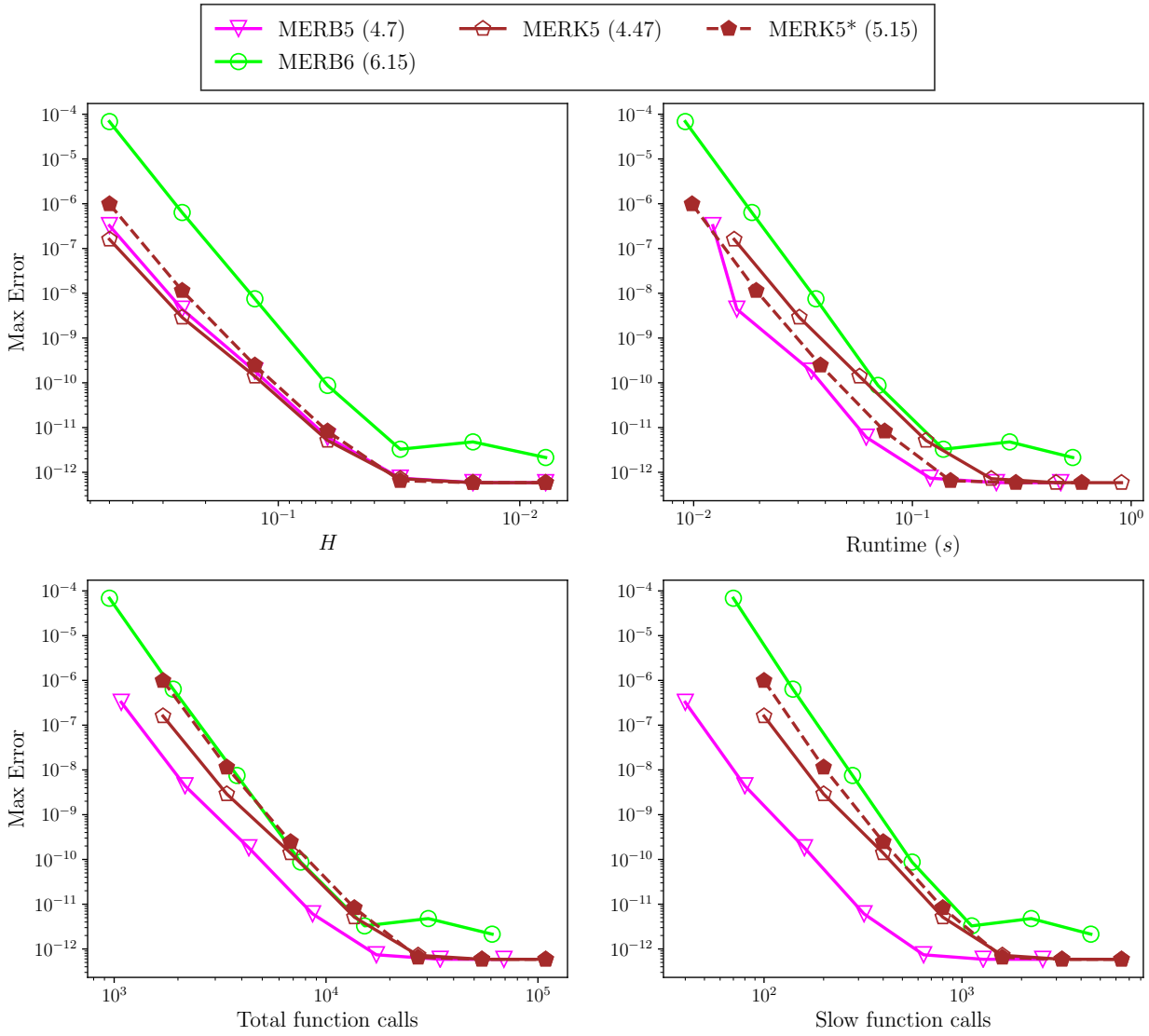


Figure 6.3: Convergence (top-left) and efficiency (top-right, bottom) of $\mathcal{O}(H^5)$ and $\mathcal{O}(H^6)$ methods on reaction-diffusion problem of section 6.4.5. MERK5* uses fixed linearization, all the other methods use dynamic linearization. The legend shows slopes of the least-squares error fit of the max error versus H data. MERB5 has the best efficiency of all $\mathcal{O}(H^5)$ and $\mathcal{O}(H^6)$ methods.

$$w' = -\lambda(w+\beta t) - \beta\left(u - \frac{a(w+\beta t)}{a\lambda + b\sigma}\right)^2 - \beta\left(v - \frac{b(w+\beta t)}{a\lambda + b\sigma}\right)^2,$$

with exact solution $u(t) = \cos(\sigma t) + ae^{-\lambda t}$, $\quad v(t) = -\sin(\sigma t) + be^{-\lambda t}$, $\quad$ and $\quad w(t) = (a\lambda + b\sigma)e^{-\lambda t} - \beta t$. This problem features linear coupling from fast to slow time scales through the equation for $u'(t)$, and nonlinear coupling from slow to fast time scales through the equation for $w'(t)$. In addition, the problem is nonautonomous and includes tunable parameters $\{a, b, \beta, \lambda, \sigma\}$ taken here to be $\{1, 20, 0.01, 5, 100\}$, with $a\sigma = b\lambda$. The parameter σ determines the frequency of the fast time scale and β controls the strength of the nonlinearity. In the case of dynamic linearization, a small value of β corresponding to weak nonlinearity results in high values of the optimal time scale separation factor m. While there are various possible splittings into a fixed linear component and remainder, we chose the most natural splitting into fast variables and slow variables informed by the exact solution:

$$F_f(t, \mathbf{u}) = \begin{bmatrix} 0 & \sigma & -1 \\ -\sigma & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} u \\ v \\ w \end{bmatrix}, \quad F_s(t, \mathbf{u}) = \begin{bmatrix} -\beta t \\ 0 \\ -\lambda w - \lambda\beta t - \beta\left(u - \frac{a(w-\beta t)}{a\lambda+b\sigma}\right)^2 - \beta\left(v - \frac{b(w-\beta t)}{a\lambda+b\sigma}\right)^2 \end{bmatrix}$$

We assess error at 20 equally spaced points within the time interval and consider slow time steps $H = 0.05 \times 2^{-k}$ for integers $k = 0, 1, \ldots, 7$.

Accuracy and efficiency plots for the bidirectional problem are shown in Figures 6.4-6.6. Starting with third-order methods, in Figure 6.4 (top-left) all methods incorporating dynamic linearization have similar errors, coinciding with their uniform time scale separation factor of $m = 80$. Similarly, fixed linearization approaches MERK3* and MRI-GARK33a* have the same $m = 10$, leading to comparable errors. As with the results in Section 6.4.5, dynamic linearization leads to lower errors than fixed linearization, here the difference in errors for the same H is up to 10^3.

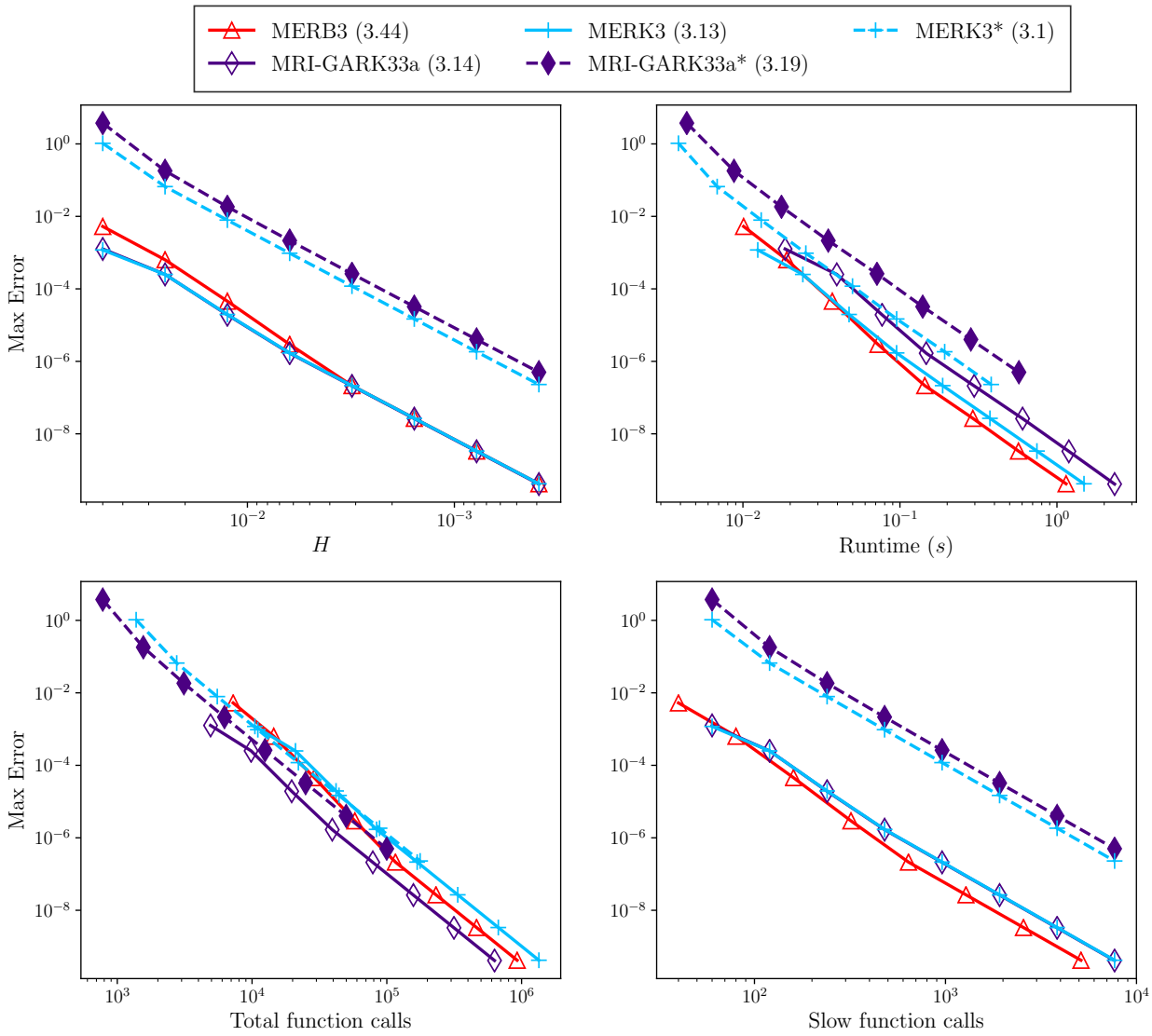


Figure 6.4: Convergence (top-left) and efficiency (top-right, bottom) of $\mathcal{O}(H^3)$ methods on the bidirectional coupling problem of section 6.4.6. MERK3* and MRI-GARK33a* use fixed linearization, all the other methods use dynamic linearization. The legend shows slopes of the least-squares fit of the max error versus H data. Methods implemented with dynamic linearization have lower errors than those implemented with fixed linearization. MERB3 has the best runtime and slow function call efficiency. MRI-GARK33a has the best total function call efficiency.

The lessons in efficiency from the reaction-diffusion problem of Section 6.4.5 are repeated here. Our implementation of MERB3 is the most efficient in MATLAB run times and slow function evaluations, while MRI-GARK33a is most efficient in total function evaluations. Overall, our new MERB3 method is competitive with other explicit MIS-type multirate methods.

Results for fourth-order methods are plotted in Figure 6.5. Like with the third-order methods, we use the same m for dynamic linearization methods, but here there is slightly more variation in errors, with MERK4 being the most accurate of the group. Given MERB4 and MERK4, MERB4 is the best choice when considering efficiency and comes very close to matching the accuracy of MERK4.

Finally, the performance of fifth and sixth-order methods on the bidirectional coupling problem is illustrated in Figure 6.6. The accuracy of all the fifth and sixth-order methods is almost identical on this test problem, so we focus on the efficiency comparisons. Both of our new MERB methods are the most competitive for this test problem. We observe that MERB5 is the most efficient in terms of run time at large values of H but as the H values become smaller, MERB6 is the most efficient. MERB6 is also the most efficient in total function calls followed by MERB5, because of fewer total traversal times compared with those for MERK5. The small number of stages for MERB5 make it the most efficient when it comes to slow function calls.

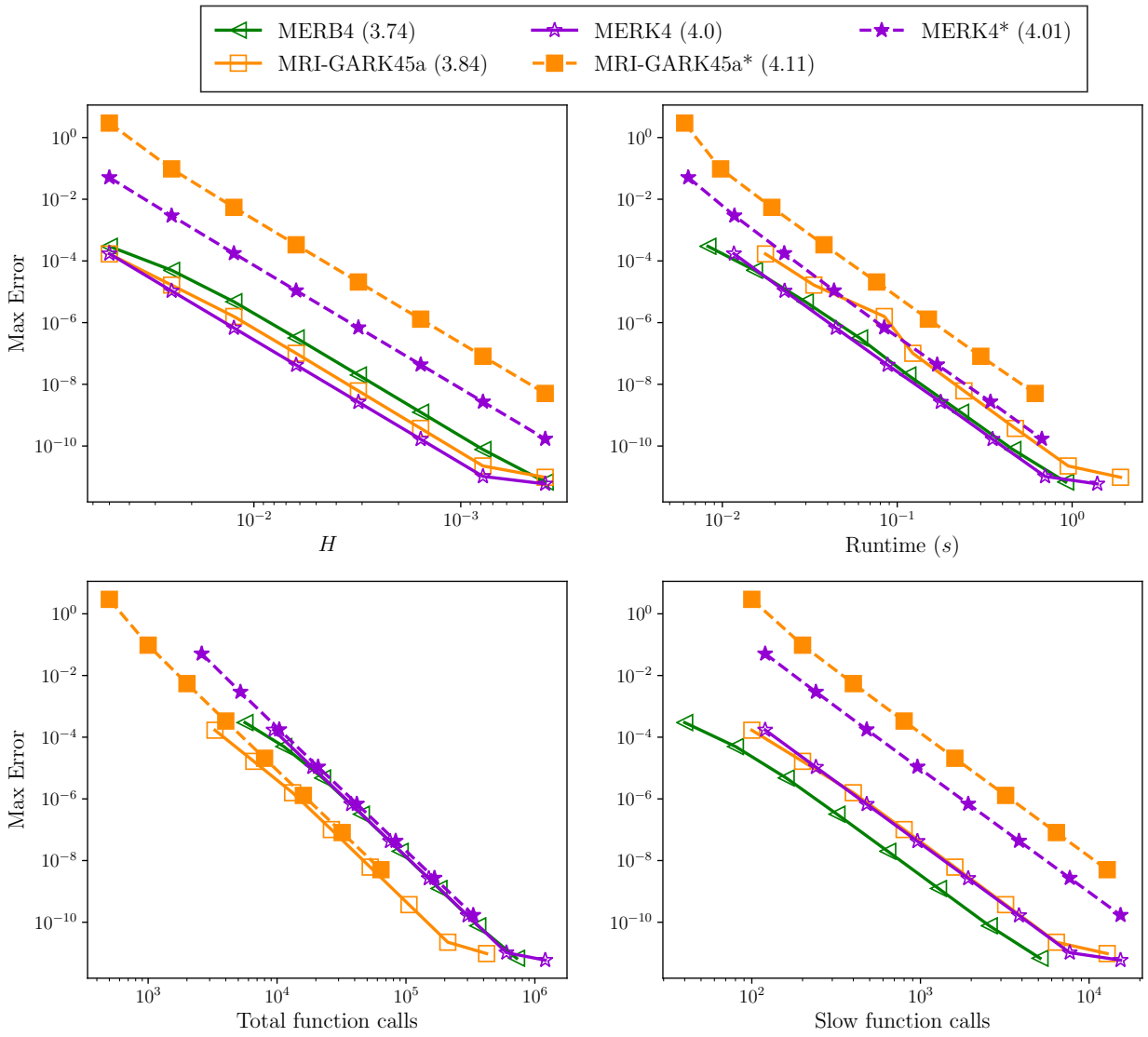


Figure 6.5: Convergence (top-left) and efficiency (top-right, bottom) of $\mathcal{O}(H^4)$ methods on the bidirectional coupling problem of section 6.4.6. MERK4* and MRI-GARK45a* use fixed linearization, all the other methods use dynamic linearization. The legend shows slopes of the least-squares fit of the max error versus H data. MERB4 and MERK4 have the most efficient runtimes. MRI-GARK methods have the most efficient total function calls, while MERB4 excels in slow function call efficiency.

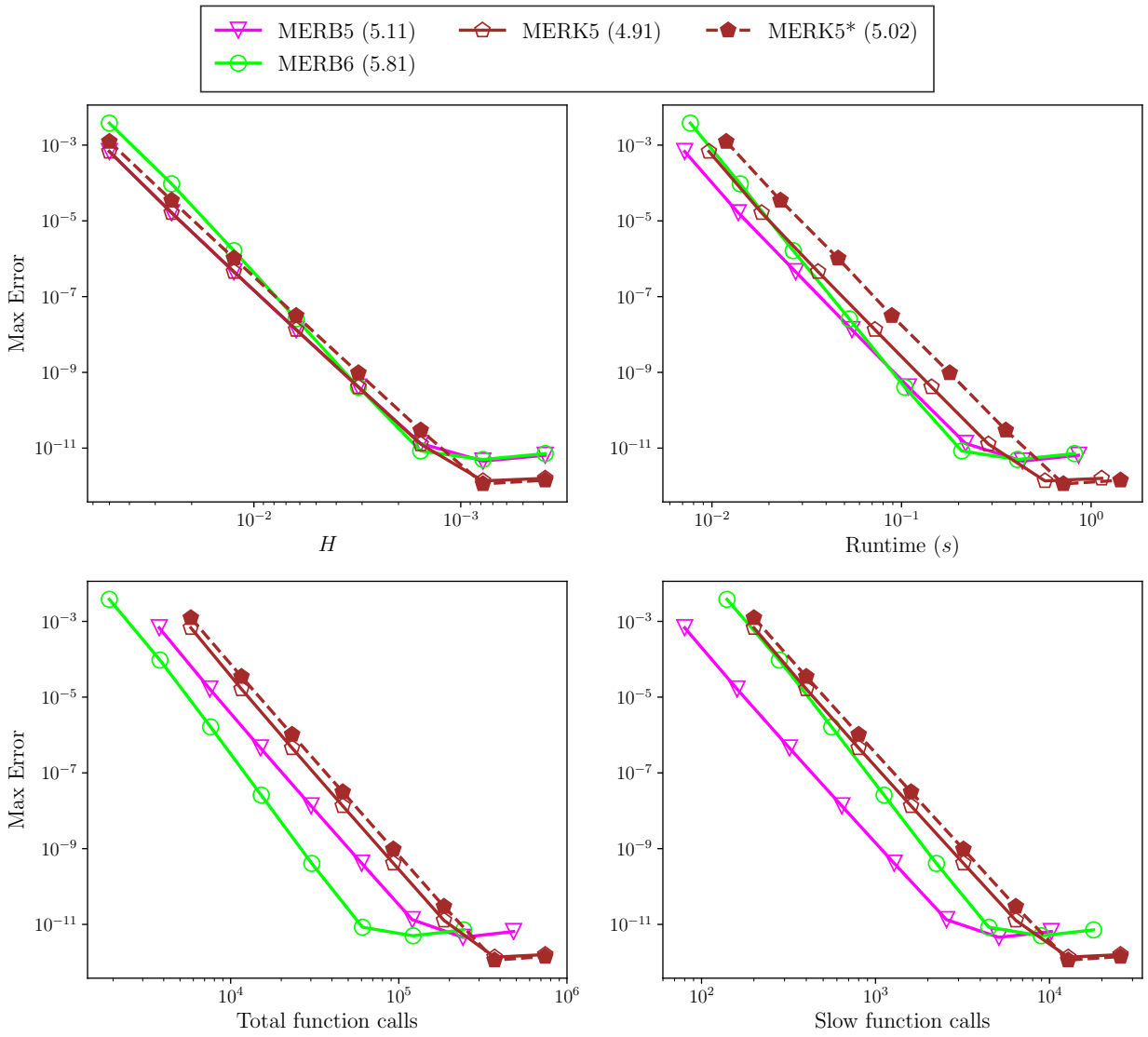


Figure 6.6: Convergence (top-left) and efficiency (top-right, bottom) of $\mathcal{O}(H^5)$ and $\mathcal{O}(H^6)$ methods on the bidirectional coupling problem of section 6.4.6. MERK5* uses fixed linearization, all the other methods use dynamic linearization. The legend shows slopes of the least-squares fit of the max error versus H data. MERB5 and MERB6 have the best runtime efficiency. Individually, MERB6 has the best total function call efficiency and MERB5 has the best slow function call efficiency.

Chapter 7

Conclusion

In this chapter we give concluding remarks for this thesis and briefly discuss future directions.

7.1. Overall contributions

In this thesis we have introduced three classes of high-order, flexible multirate methods namely, IMEX-MRI-GARK, MERK, and MERB. These methods are geared towards multiphysics problems that exhibit multiple time scales. Their unifying property is that of a fast time scale that can be integrated almost arbitrarily through the definition of modified ODE problems, an idea that was first introduced by Knoth and Wolke [61] and further developed to create multirate infinitesimal step methods [107]. The overall contributions to each class of methods are as follows:

In Chapter 3 we presented a newly developed class of IMEX-MRI-GARK methods that extend Sandu's MRI-GARK methods [88] to include an IMEX treatment of the slow time scale, making them the first class of methods of MIS-type to have this property. This flexibility in the integration of the slow dynamics is in addition to the flexibility at the fast time scale which can be treated explicitly, implicitly, or in an IMEX fashion. We derived order conditions for IMEX-MRI-GARK methods up to fourth-order, taking advantage of the structure of our IMEX-ARK slow base. Because of their ease of implementation and following [88], we constructed solve-decoupled methods that alternate between integrating fast modified ODEs and standard diagonally-implicit Runge–Kutta solves. Combining Sandu's scalar stability analysis and Zharovsky and collaborators' joint stability analysis, we defined our own joint stability region to assess linear stability. Using this definition of joint stability we were able to create a third-order method optimized to extend the region of stability on the negative real-axis. Numerical experiments confirmed the order of convergence for our third and fourth-order methods. Comparisons with legacy Lie–Trotter, Strang–Marchuk, and implicit MRI-GARK methods show that IMEX-MRI-GARK are just as accurate and efficient as

implicit MRI-GARK methods on select test problems, and more accurate and efficient than legacy approaches.

In Chapter 4 we further investigated the linear stability of IMEX-MRI-GARK methods with the specific goal of constructing a fourth-order IMEX-MRI-GARK method with better stability properties. Relaxing our definition of joint stability allowed us to find viable directions for maximizing the joint stability region on the negative real axis. Numerical experiments of the newly constructed fourth-order method show improved stability properties compared to the fourth-order method of Chapter 3 on an advection-diffusion-reaction test problem.

In Chapter 5 we introduced MERK methods which expand on MIS theory, starting with exponential Runge–Kutta methods. MERK methods allow the solution of exponential Runge–Kutta methods without the evaluation of matrix functions, instead, we solve modified ODEs to advance from slow stage to slow stage. We discussed the theory of MERK methods including convergence results and showed that the modified ODEs corresponding to internal stages can be solved with a fast integrator of degree one less than the overall MERK method. We derived methods up to fifth-order, including the very first fifth-order method of MIS-type. In numerical experiments we identified two main categories for practitioners of multirate methods: those that have stability limited systems of ODEs and those with systems of ODEs with differing time scales. This distinction enabled us to pick different sets of test problems in each category. For the second category, we presented an experimental technique for determining the optimal ratio between the fast and slow time scales. Such optimal ratios enable an efficient solution of test problems when using fixed time steps for both the slow and fast dynamics, in the absence of time adaptivity techniques. Our numerical experiments confirmed orders of convergence and provided efficiency results.

In Chapter 6 we presented MERB methods, another addition to the family of MIS-type methods. The theory of MERB methods closely follows that of MERK methods, with a lot of properties carrying over from MERK to MERB. MERB methods involve the least number of order conditions of all MIS-type methods owing to their exponential Rosenbrock base, in addition, high-order methods are easily attainable. We derived MERB methods up to the very first sixth-order method of MIS-type. Similar to our treatment of MERK methods, we provided in-depth numerical experiments. Our results involved comparisons with MERK and MRI-GARK methods, which led to

the investigation of both fixed linearization and dynamic linearization approaches to splitting the right hand side. In addition to confirming convergence of MERB methods, we demonstrated their competitiveness, particularly in efficiency.

7.2. Future directions

There are several immediate research directions in extension of this thesis work. An extension to all methods discussed is the derivation of embedding coefficients for time adaptivity. Procedures to effectively use these embedding coefficients for the control of two time step sizes are also needed. Proper error control for multirate methods will promote their widespread adoption.

Another topic of extension is the stability analysis for additive multirate systems. Commonly used tools in this area and their implications are not yet fully realized due to added complexity in multirate schemes. Furthermore, there is currently no standard way of assessing stability of multirate methods. All the methods we have explored in this thesis can benefit from rigorous stability analysis that investigates dependence on fast-slow splitting, time scale separation factors, and coupling strength between fast and slow components.

There is also need for more test problems that can rigorously test various aspects of multirate methods before their application to large-scale systems.

Finally, in relation to IMEX-MRI-GARK methods, solve-coupled approaches are predicted to provide more stability, opening up new questions about how to effectively implement them.

Appendix A

IMEX-MRI-GARK appendix

A.1. IMEX-MRI-GARK3a

The nonzero coefficients for IMEX-MRI-GARK3a (accurate to 36 decimal digits) are:

$$c_1^{\{S\}} = 0,$$

$$c_2^{\{S\}} = c_3^{\{S\}} = 0.435866521508458999416019451193556842 5,$$

$$c_4^{\{S\}} = c_5^{\{S\}} = 0.717933260754229499708009725596778421 3,$$

$$c_6^{\{S\}} = c_7^{\{S\}} = c_8^{\{S\}} = 1,$$

$$\gamma_{2,1}^{\{0\}} = -\gamma_{3,1}^{\{0\}} = \gamma_{3,3}^{\{0\}} = \gamma_{5,5}^{\{0\}} = \gamma_{6,1}^{\{0\}} = -\gamma_{7,1}^{\{0\}} = \gamma_{7,7}^{\{0\}}$$

$$= 0.435866521508458999416019451193556842 5,$$

$$\gamma_{4,1}^{\{0\}} = -\gamma_{5,1}^{\{0\}} = -0.410333696228852501459951372016107893 7,$$

$$\gamma_{4,3}^{\{0\}} = 0.692400435474623001751941646419329472 4,$$

$$\gamma_{5,3}^{\{0\}} = -0.846200217737311500875970823209664736 2,$$

$$\gamma_{6,3}^{\{0\}} = 0.926429909930239570044487409660101532 8,$$

$$\gamma_{6,5}^{\{0\}} = -1.080229692192928069168516586450436797,$$

$$\omega_{2,1}^{\{0\}} = \omega_{8,7}^{\{0\}} = 0.435866521508458999416019451193556842 5,$$

$$\omega_{4,1}^{\{0\}} = -0.568871580123440092846503292531793202 1,$$

$$\omega_{4,3}^{\{0\}} = 0.850938319369210593138493566935014780 9,$$

$$\omega_{5,1}^{\{0\}} = -\omega_{5,3}^{\{0\}} = 0.454283944643608855878770886900124654,$$

$$\omega_{6,1}^{\{0\}} = -0.427137182100507401170664505039073247 4,$$

$$\omega_{6,3}^{\{0\}} = 0.156274773310338082101466049703702349 6,$$

$$\omega_{6,5}^{\{0\}} = 0.552929148035939819361188729738592476 5,$$

$$\omega_{8,1}^{\{0\}} = 0.105858296071879638722377459477184953,$$

$$\omega_{8,3}^{\{0\}} = 0.655567501140070250975288954324730635,$$

$$\omega_{8,5}^{\{0\}} = -1.197292318720408889113685864995472431.$$

We note that these coefficients (and all of those that follow) are available electronically in [16].

A.2. IMEX-MRI-GARK3b

The nonzero coefficients for IMEX-MRI-GARK3b (accurate to 36 decimal digits) are:

$$c_1^{\{S\}} = 0,$$

$$c_2^{\{S\}} = c_3^{\{S\}} = 0.435866521508458999416019451193556842 5,$$

$$c_4^{\{S\}} = c_5^{\{S\}} = 0.717933260754229499708009725596778421 3,$$

$$c_6^{\{S\}} = c_7^{\{S\}} = c_8^{\{S\}} = 1,$$

$$\gamma_{2,1}^{\{0\}} = -\gamma_{3,1}^{\{0\}} = \gamma_{3,3}^{\{0\}} = \gamma_{5,5}^{\{0\}} = \gamma_{7,7}^{\{0\}}$$

$$= 0.435866521508458999416019451193556842 5,$$

$$\gamma_{4,1}^{\{0\}} = -\gamma_{5,1}^{\{0\}} = 0.041427375356441483715379923027827 5639,$$

$$\gamma_{4,3}^{\{0\}} = 0.240639363889329016576610351375394 0148$$

$$\gamma_{5,3}^{\{0\}} = -0.394439146152017515700639528165729 2786$$

$$\gamma_{6,1}^{\{0\}} = -\gamma_{7,1}^{\{0\}} = 0.112337314300604780263354341688960 5123$$

$$\gamma_{6,3}^{\{0\}} = 1.051807513648115027700693049638099167$$

$$\gamma_{6,5}^{\{0\}} = -0.882078088702949307672057116923838 1009$$

$$\gamma_{7,3}^{\{0\}} = -0.125377603717875457656205639977997 6346$$

$$\gamma_{7,5}^{\{0\}} = -0.198151603489978761496459469526598 6957$$

$$\omega_{2,1}^{\{0\}} = \omega_{8,7}^{\{0\}} = 0.435866521508458999416019451193556842 5,$$

$$\omega_{4,1}^{\{0\}} = -0.175014528557046759061067000001874 9059,$$

$$\omega_{4,3}^{\{0\}} = 0.457081267802817259353057274405096484 6,$$

$$\omega_{5,1}^{\{0\}} = -\omega_{5,3}^{\{0\}} = 0.060426893077215522093334594370206357 74,$$

$$\omega_{6,1}^{\{0\}} = 0.119521395942545444003878603402793686 9,$$

$$\omega_{6,3}^{\{0\}} = -1.843725226689661917898533950296297 65,$$

$$\omega_{6,5}^{\{0\}} = 2.00627056999288697418664562129672554 2,$$

$$\omega_{7,1}^{\{0\}} = -0.54665857804305284517454310844186693 43,$$

$$\omega_{7,3}^{\{0\}} = 2,$$

$$\omega_{7,5}^{\{0\}} = -1.45334142195694715482545689155813306 6,$$

$$\omega_{8,1}^{\{0\}} = 0.10585829607187963872237745947718495 3,$$

$$\omega_{8,3}^{\{0\}} = 0.65556750114007025097528895432473063 5,$$

$$\omega_{8,5}^{\{0\}} = -1.19729231872040888911368586499547243 1.$$

A.3. IMEX-MRI-GARK4

The nonzero coefficients for IMEX-MRI-GARK4 (accurate to 36 decimal digits) are:

$$\mathbf{c}^{\{S\}} = \begin{bmatrix} 0 & \frac{1}{2} & \frac{1}{2} & \frac{5}{8} & \frac{5}{8} & \frac{3}{4} & \frac{3}{4} & \frac{7}{8} & \frac{7}{8} & 1 & 1 & 1 \end{bmatrix},$$

$$\gamma_{2,1}^{\{0\}} = \tfrac{1}{2},$$

$$\gamma_{3,1}^{\{0\}} = -\gamma_{3,3}^{\{0\}} = -\gamma_{5,5}^{\{0\}} = -\gamma_{7,7}^{\{0\}} = -\gamma_{9,9}^{\{0\}} = -\gamma_{11,11}^{\{0\}} = -\tfrac{1}{4},$$

$$\gamma_{4,1}^{\{0\}} = -3.97728124810848818306703385146227889,$$

$$\gamma_{4,3}^{\{0\}} = 4.10228124810848818306703385146227889,$$

$$\gamma_{5,1}^{\{0\}} = -0.06905388741401691232724147084809374 06,$$

$$\gamma_{5,3}^{\{0\}} = -0.18094611258598308767275852915190625 9,$$

$$\gamma_{6,1}^{\{0\}} = -1.76176766375792052886337896482241241,$$

$$\gamma_{6,3}^{\{0\}} = 2.69452469837729861015533815079146138,$$

$$\gamma_{6,5}^{\{0\}} = -0.80775703461937808129195918596904897 8,$$

$$\gamma_{7,1}^{\{0\}} = 0.55587217915539694873050810095880849 6,$$

$$\gamma_{7,3}^{\{0\}} = -0.67991405015799950139585015278834869 5,$$

$$\gamma_{7,5}^{\{0\}} = -\gamma_{8,5}^{\{0\}} = -0.12595812899739744733465794817045980 1,$$

$$\gamma_{8,1}^{\{0\}} = -5.84017602872495595444642665754106511,$$

$$\gamma_{8,3}^{\{0\}} = 8.17445668429191508919127080571071637,$$

$$\gamma_{8,7}^{\{0\}} = -2.33523878456435658207950209634011106,$$

$$\gamma_{9,1}^{\{0\}} = -1.90679264516781180809475930503605 23,$$

$$\gamma_{9,3}^{\{0\}} = -\gamma_{10,3}^{\{0\}} = -1.54705781138512393363298457924938844$$

$$\gamma_{9,5}^{\{0\}} = -\gamma_{10,5}^{\{0\}} = 4.12988801314935030595449173802031322,$$

$$\gamma_{9,7}^{\{0\}} = -\gamma_{10,7}^{\{0\}} = -0.92603755659641456422674785373487247 7,$$

$$\gamma_{10,1}^{\{0\}} = 3.33702815168872605455765278252966252,$$

$$\gamma_{10,9}^{\{0\}} = -1.55523550652091424646289347749361021,$$

$$\gamma_{11,1}^{\{0\}} = -0.82129362922100761872052411231244675 2,$$

$$\gamma_{11,3}^{\{0\}} = 0.32861035606859998855167726426896964 6,$$

$$\gamma_{11,5}^{\{0\}} = 0.67800181210202669414264123242113951 6,$$

$$\gamma_{11,7}^{\{0\}} = -0.342779287862800022896645471462060708,$$

$$\gamma_{11,9}^{\{0\}} = -0.092539251086819041077148912915601702 5,$$

$$\gamma_{4,1}^{\{1\}} = -\gamma_{4,3}^{\{1\}} = 8.704562496216976366134067702924557 78,$$

$$\gamma_{6,1}^{\{1\}} = 3.911643102343874882382412408713410122 9,$$

$$\gamma_{6,3}^{\{1\}} = -5.027157171582631044965159243279110 25,$$

$$\gamma_{6,5}^{\{1\}} = 1.115514069238756162583918371938097 96,$$

$$\gamma_{8,1}^{\{1\}} = 10.818607699139118011431837113164513 2,$$

$$\gamma_{8,3}^{\{1\}} = -14.989085268267831175590841305844735 4,$$

$$\gamma_{8,7}^{\{1\}} = 4.170477569128713164159004192680222 13,$$

$$\gamma_{10,1}^{\{1\}} = -2.610471013041828492925786954987220 43,$$

$$\gamma_{10,9}^{\{1\}} = 2.610471013041828492925786954987220 43,$$

$$\omega_{2,1}^{\{0\}} = \tfrac{1}{2},$$

$$\omega_{4,1}^{\{0\}} = -1.917165343636628688781722160649469 05,$$

$$\omega_{4,3}^{\{0\}} = 2.042165343636628688781722160649469 05,$$

$$\omega_{5,1}^{\{0\}} = -\omega_{5,3}^{\{0\}} = -0.404751031801105942697915907046990469,$$

$$\omega_{6,1}^{\{0\}} = 11.451466022492216366656980286026317 3,$$

$$\omega_{6,3}^{\{0\}} = -30.210757475265042714406478155739506 1,$$

$$\omega_{6,5}^{\{0\}} = 18.884291452772826347749497869713188 8,$$

$$\omega_{7,1}^{\{0\}} = -0.709033564760261450684711672946330144,$$

$$\omega_{7,3}^{\{0\}} = 1.030307208587518766526161908840047 18,$$

$$\omega_{7,5}^{\{0\}} = -\omega_{8,5}^{\{0\}} = -0.321273643827257315841450235893717036,$$

$$\omega_{8,1}^{\{0\}} = -29.995487164558284398409106849441992 7,$$

$$\omega_{8,3}^{\{0\}} = 37.605982774991801805364896856243857,$$

$$\omega_{8,7}^{\{0\}} = -7.806769254260774722797240242695581 29,$$

$$\omega_{9,1}^{\{0\}} = 3.104665054272962116338769391849124 22,$$

$$\omega_{9,3}^{\{0\}} = -\omega_{10,3}^{\{0\}} = -2.430325019757162297132065927415566 36,$$

$$\omega_{9,5}^{\{0\}} = -\omega_{10,5}^{\{0\}} = -1.905479301151524635219201659483842 13,$$

$$\omega_{9,7}^{\{0\}} = -\omega_{10,7}^{\{0\}} = 1.231139266635724816012498195050284 27,$$

$$\omega_{10,1}^{\{0\}} = -2.424429547752047869875875914355514 01,$$

$$\omega_{10,9}^{\{0\}} = -0.555235506520914246462893477493610215,$$

$$\omega_{11,1}^{\{0\}} = -0.010441350444797485902945189451653542,$$

$$\omega_{11,3}^{\{0\}} = 0.072603036146550745051521045054881416 1,$$

$$\omega_{11,5}^{\{0\}} = -0.12882759516772609522394540985764 2431,$$

$$\omega_{11,7}^{\{0\}} = 0.11293553500938235661394401071221 5408,$$

$$\omega_{11,9}^{\{0\}} = \omega_{12,9}^{\{0\}} = -0.04626962554340952053857445645780 08512,$$

$$\omega_{12,1}^{\{0\}} = -0.81085227877621013281757892286079321,$$

$$\omega_{12,3}^{\{0\}} = 0.25600731992204924350015621921408823,$$

$$\omega_{12,5}^{\{0\}} = 0.80682940726975278936658664227878 1947,$$

$$\omega_{12,7}^{\{0\}} = -0.45571482287218237951058948217427 6116,$$

$$\omega_{12,11}^{\{0\}} = \tfrac{1}{4}$$

$$\omega_{4,1}^{\{1\}} = -\omega_{4,3}^{\{1\}} = 4.08433068727325737756344432129893 81,$$

$$\omega_{6,1}^{\{1\}} = -21.84342998138222084791812875795865 36,$$

$$\omega_{6,3}^{\{1\}} = 59.61201288692787354341712449738503 12,$$

$$\omega_{6,5}^{\{1\}} = -37.76858290554565269549899573942637 76,$$

$$\omega_{8,1}^{\{1\}} = 61.65904145863709169818763704477664 58,$$

$$\omega_{8,3}^{\{1\}} = -77.27257996715864114378211753016780 84,$$

$$\omega_{8,7}^{\{1\}} = 15.61353850852154944559448048539116 26,$$

$$\omega_{10,1}^{\{1\}} = -\omega_{10,9}^{\{1\}} = -1.11047101304182849292578695498722043.$$

A.4. IMEX-MRI-GARK4s

The nonzero coefficients for IMEX-MRI-GARK4s (accurate to 34 decimal digits) are:

$$\mathbf{c}^{\{S\}} = \begin{bmatrix} 0 & \frac{1}{2} & \frac{1}{2} & \frac{5}{8} & \frac{5}{8} & \frac{3}{4} & \frac{3}{4} & \frac{7}{8} & \frac{7}{8} & 1 & 1 & 1 \end{bmatrix},$$

$$\gamma^0_{2,1} = \tfrac{1}{2}$$

$$\gamma^0_{3,1} = -\gamma^0_{3,3} = -\gamma^0_{5,5} = -\gamma^0_{7,7} = -\gamma^0_{11,11} = -\tfrac{1}{4}$$

$$\gamma^0_{4,1} = -2.81783149465147021718400110363652932 8$$

$$\gamma^0_{4,3} = 2.94283149465147021718400110363652932 8$$

$$\gamma^0_{5,1} = -0.19526646111176367004436116303622839 04$$

$$\gamma^0_{5,3} = -0.05473353888823632995563883696377160959$$

$$\gamma^0_{6,1} = 1.57338666468518023113209548564270494 9$$

$$\gamma^0_{6,3} = -2.84811532116455498139585953787534516 8$$

$$\gamma^0_{6,5} = 1.39972865647937475026376405223264021 9$$

$$\gamma^0_{7,1} = -0.01856566842724184119671364205880345497$$

$$\gamma^0_{7,3} = 0.17467195432700476598037030375682450 17$$

$$\gamma^0_{7,5} = -0.40610628589976292478365666169802104 67$$

$$\gamma^0_{8,1} = -13.17145919027832704094939131031560574$$

$$\gamma^0_{8,3} = 18.04050552525041321233472357853754826$$

$$\gamma^0_{8,5} = -0.19256888926787043071601784618442500 39$$

$$\gamma^0_{8,7} = -4.55147744570421574066931442203751 7517$$

$$\gamma^0_{9,1} = -0.72929649460200298745530827432804423 62$$

$$\gamma^0_{9,3} = -3.95187540946422062673182831975848093 3$$

$$\gamma^0_{9,5} = 5.40721608002002010072915897172519337 7$$

$$\gamma^0_{9,7} = -0.97589054587441980712724790732726818 48$$

$$\gamma^0_{9,9} = 0.24984636992062332058522552968859997 69$$

$$\gamma^0_{10,1} = 4.27436986605952044736828671850893082$$

$$\gamma^0_{10,3} = 3.97456554307070271561749140681073886 8$$

$$\gamma^0_{10,5} = -9.82031238529149473626447877990242816 7$$

$$\gamma^0_{10,7} = 3.23670228737408545348042252683674222 6$$

$$\gamma^0_{10,9} = -1.54032531121281388020172187225398374 7$$

$$\gamma^0_{11,1} = -0.82116937759344011566834768226878319 79$$

$$\gamma^0_{11,3} = 0.32426154910373738172550221274074525 96$$

$$\gamma^0_{11,5} = 0.68843894881769695052546135608887804 36$$

$$\gamma^0_{11,7} = -0.35147690179252523654899557451850948 08$$

$$\gamma^0_{11,9} = -0.09005421853546898003362031204233062454$$

$$\gamma^1_{4,1} = 6.385662989302940434368002207273058657$$

$$\gamma^1_{4,3} = -6.385662989302940434368002207273058657$$

$$\gamma^1_{6,1} = -2.506240407146833122175468645212953117$$

$$\gamma^1_{6,3} = 5.805697720105582622702996749678233555$$

$$\gamma^1_{6,5} = -3.299457312958749500527528104465280438$$

$$\gamma^1_{8,1} = 26.630049717411137764292209904748 8184$$

$$\gamma^1_{8,3} = -36.430354959154835956630187764588 74553$$

$$\gamma^1_{8,5} = 1.197350350335266710999349015764892101$$

$$\gamma^1_{8,7} = 8.602954891408431481338628844075035034$$

$$\gamma^1_{9,1} = -0.001040736867997488831073110704394935055$$

$$\gamma^1_{9,3} = 0.000118956391737380250041983265063549 8794$$

$$\gamma^1_{9,5} = 0.000307260158753411196513871672664208 7451$$

$$\gamma^1_{9,7} = 0.000307260158753338554968315143867130 2458$$

$$\gamma^1_{9,9} = 0.000307260158753358829548940622800046 1851$$

$$\gamma^1_{10,1} = -6.839106006047037430994883777657378232$$

$$\gamma^1_{10,3} = -0.045499223604701558021368157369579421 36$$

$$\gamma^1_{10,5} = 8.825885350384195859874125744681805372$$

$$\gamma^1_{10,7} = -4.52193074315808463126131755416281521 3$$

$$\gamma^1_{10,9} = 2.58065062242562776040344374450796749 4$$

$$\omega^0_{2,1} = \tfrac{1}{2}$$

$$\omega^0_{4,1} = -3.80247472635965817702473197159983100 5$$

$$\omega^0_{4,3} = 3.92747472635965817702473197159983100 5$$

$$\omega^0_{5,1} = -0.21056322319013590083081055581715190 78$$

$$\omega^0_{5,3} = -\omega^0_{5,1}$$

$$\omega^0_{6,1} = 1.95332786216781413282585895549956496 3$$

$$\omega^0_{6,3} = -2.72651833567709877539843994773647302 2$$

$$\omega^0_{6,5} = 0.89819047350928464257258099223690805 96$$

$$\omega^0_{7,1} = 0.07522455689146829287472230399977197 606$$

$$\omega^0_{7,3} = 0.11794649914631932114894850845779510 48$$

$$\omega^0_{7,5} = -0.19317105603778761402367081245756708 09$$

$$\omega^0_{8,1} = -13.58095856011764416959932319624351 428$$

$$\omega^0_{8,3} = 6.23437088884622582352425709924284226 8$$

$$\omega^0_{8,5} = 9.78290996863593098620270893661654554 3$$

$$\omega^0_{8,7} = -2.31132229736451264012764283961587352 7$$

$$\omega_{9,1}^{0} = 1.677015123971293108584232945561385364$$

$$\omega_{9,3}^{0} = -0.404991753645633067537288375008361995 6$$

$$\omega_{9,5}^{0} = -2.813322851301324867279163299826905131$$

$$\omega_{9,7}^{0} = 1.541299480975664826232218729273881763$$

$$\omega_{10,1}^{0} = -5.652730530866139340795071311533834901$$

$$\omega_{10,3}^{0} = 0.652878024197623318491621225605345912 7$$

$$\omega_{10,5}^{0} = 13.736776265929779405894350929884240 03$$

$$\omega_{10,7}^{0} = -8.071598448048449503389178971701767296$$

$$\omega_{10,9}^{0} = -0.540325311212813880201721872253983747 2$$

$$\omega_{11,1}^{0} = 0.005010108382277661249247933502754606596$$

$$\omega_{11,3}^{0} = 0.088620418914167213014488297506349348 11$$

$$\omega_{11,5}^{0} = -0.252328597389734883541274352050391648 1$$

$$\omega_{11,7}^{0} = 0.203725179361024499294348277062453005 6$$

$$\omega_{11,9}^{0} = -0.045027109267734490016810156021165312 27$$

$$\omega_{12,1}^{0} = -0.826179485975717776917595615771537804 5$$

$$\omega_{12,3}^{0} = 0.235641130189570168711013915234395911 5$$

$$\omega_{12,5}^{0} = 0.940767546207431834066735708139269691 7$$

$$\omega^0_{12,7} = -0.55520208115354973584334385158096248 64$$

$$\omega^0_{12,9} = -0.04502710926773449001681015602116531227$$

$$\omega^0_{12,11} = \tfrac{1}{4}$$

$$\omega^1_{4,1} = 7.85494945271931635404946394319966 2009$$

$$\omega^1_{4,3} = -\omega^1_{4,1}$$

$$\omega^1_{6,1} = -3.23552927795535646399009679936482611$$

$$\omega^1_{6,3} = 5.03191022497392574913525878383864 2229$$

$$\omega^1_{6,5} = -1.79638094701856928514516198447381 6119$$

$$\omega^1_{8,1} = 27.26146800645235175344920178448748462$$

$$\omega^1_{8,3} = -12.70463477598509028934641121540127474$$

$$\omega^1_{8,5} = -19.17947782519628674435807624831795692$$

$$\omega^1_{8,7} = 4.62264459472902528025528567923174 7054$$

$$\omega^1_{10,1} = 8.20143081378969246442167673194489 9075$$

$$\omega^1_{10,3} = -0.49577254110398050190866570119396783 41$$

$$\omega^1_{10,5} = -21.84690682925690907723037526011466 98$$

$$\omega^1_{10,7} = 13.06059793414556935431392048485577106$$

$$\omega^1_{10,9} = 1.08065062242562776040344374450796749 4$$

REFERENCES

[1] Andrus, J. Stability of a multi-rate method for numerical integration of ODE's. *Computers & Mathematics with Applications 25*, 2 (1993), 3–14.

[2] Andrus, J. F. Numerical solution of systems of ordinary differential equations separated into subsystems. *SIAM Journal on Numerical Analysis 16*, 4 (1979), 605–611.

[3] Araujo, A. L., Murua, A., and Sanz-Serna, J. M. Symplectic methods based on decompositions. *SIAM Journal on Numerical Analysis 34*, 5 (1997), 1926–1947.

[4] Ascher, U. M., Ruuth, S. J., and Spiteri, R. J. Implicit-explicit Runge–Kutta methods for time-dependent partial differential equations. *Applied Numerical Mathematics 25*, 2-3 (Nov. 1997), 151–167.

[5] Bartel, A., and Günther, M. Inter/extrapolation-based multirate schemes-a dynamic-iteration perspective, 2020.

[6] Bauer, T. P., and Knoth, O. Extended multirate infinitesimal step methods: Derivation of order conditions. *Journal of Computational and Applied Mathematics* (2019), 112541.

[7] Bouzarth, E. L., and Minion, M. L. A multirate time integrator for regularized stokeslets. *Journal of Computational Physics 229*, 11 (2010), 4208–4224.

[8] Bremicker-Trübelhorn, S., and Ortleb, S. Time-adaptive multirate generalized-structure additively partitioned Runge–Kutta schemes for mechanical fluid-structure interaction. *PAMM 16* (10 2016), 869–870.

[9] Bremicker-Trübelhorn, S., and Ortleb, S. On multirate GARK schemes with adaptive micro step sizes for fluid–structure interaction: Order conditions and preservation of the geometric conservation law. *Aerospace 4* (2 2017), 8.

[10] Butcher, J. C. *Numerical Methods for Ordinary Differential Equations*. John Wiley & Sons, Apr. 2008.

[11] Cardone, A., Jackiewicz, Z., Sandu, A., and Zhang, H. Extrapolation-based implicit-explicit general linear methods. *Numer. Algor. 65*, 3 (Mar. 2014), 377–399.

[12] Cardone, A., Jackiewicz, Z., Sandu, A., and Zhang, H. Construction of highly stable implicit-explicit general linear methods. *Conference Publications*, special (2015), 185.

[13] Cash, J. R. Diagonally implicit Runge–Kutta formulae with error estimates. *IMA J Appl Math 24*, 3 (Nov. 1979), 293–301.

[14] Cash, J. R., and Karp, A. H. A variable order Runge-Kutta method for initial value problems with rapidly varying right-hand sides. *ACM Trans. Math. Softw. 16*, 3 (Sept. 1990), 201–222.

[15] Chinomona, R., and Reynolds, D. R. Implicit–explicit multirate infinitesimal GARK methods. *arXiv:2007.09776 [math.NA]* (July 2020).

[16] Chinomona, R., and Reynolds, D. R. Implicit-explicit multirate infinitesimal GARK (IMEX-MRI-GARK) methods. `https://github.com/rujekoc/imexmri`, 2020.

[17] Chinomona, R., Reynolds, D. R., and Luan, V. T. Multirate exponential Runge–Kutta methods (MERK). `https://github.com/rujekoc/merk_v1`, 2019.

[18] Conde, S., Gottlieb, S., Grant, Z. J., and Shadid, J. N. Implicit and implicit–explicit strong stability preserving Runge–Kutta methods with high linear order. *J. Sci. Comput. 73*, 2-3 (Dec. 2017), 667–690.

[19] Constantinescu, E., and Sandu, A. Extrapolated implicit-explicit time stepping. *SIAM J. Sci. Comput. 31*, 6 (2010), 4452–4477.

[20] Constantinescu, E., and Sandu, A. Extrapolated multirate methods for differential equations with multiple time scales. *J. Sci. Comput. 56* (12 2012).

[21] Constantinescu, E. M., and Sandu, A. Multirate timestepping methods for hyperbolic conservation laws. *J Sci Comput 33*, 3 (Oct. 2007), 239–278.

[22] Constantinescu, E. M., and Sandu, A. *On Extrapolated Multirate Methods*. Springer Berlin Heidelberg, Berlin, Heidelberg, 2010, pp. 341–347.

[23] Constantinescu, E. M., and Sandu, A. Extrapolated multirate methods for differential equations with multiple time scales. *J. Sci. Comput. 56*, 1 (July 2013), 28–44.

[24] Cooper, G. J., and Sayfy, A. Additive methods for the numerical solution of ordinary differential equations. *Math. Comp. 35* (1980), 1159–1172.

[25] Cooper, G. J., and Sayfy, A. Additive Runge–Kutta methods for stiff ordinary differential equations. *Math. Comp. 40*, 161 (Jan. 1983), 207–207.

[26] Cox, S. M., and Matthews, P. C. Exponential time differencing for stiff systems. *Journal of Computational Physics 176*, 2 (2002), 430–455.

[27] Demirel, A., Niegemann, J., Busch, K., and Hochbruck, M. Efficient multiple time-stepping algorithms of higher order. *Journal of Computational Physics 285* (2015), 133–148.

[28] Engstler, C., and Lubich, C. Mur8: A multirate extension of the eighth-order dormand-prince method. *Appl. Numer. Math. 25*, 2-3 (Nov. 1997), 185–192.

[29] Estep, D., Ginting, V., Ropp, D., Shadid, J. N., and Tavener, S. An a posteriori–a priori analysis of multiscale operator splitting. *SIAM J. Numer. Anal. 46*, 3 (Jan. 2008), 1116–1146.

[30] Estep, D., Ginting, V., and Tavener, S. A posteriori analysis of a multirate numerical method for ordinary differential equations. *Computer Methods in Applied Mechanics and Engineering 223–224* (06 2012), 10–27.

[31] Fok, P. W. A linearly fourth order multirate Runge-Kutta method with error control. *J. Sci. Comput. 66* (2016), 177–195.

[32] Forest, E., and Ruth, R. D. Fourth-order symplectic integration. *Physica D: Nonlinear Phenomena 43* (5 1990), 105–117.

[33] Gander, M. J., and Halpern, L. Techniques for locally adaptive time stepping developed over the last two decades. In *Domain decomposition methods in science and engineering XX*. Springer, 2013, pp. 377–385.

[34] Gardner, D. J., Reynolds, D. R., Woodward, C. S., and Balos, C. J. Enabling new flexibility in the SUNDIALS suite of nonlinear and differential/algebraic equation solvers. *arXiv:2011.10073 [cs.MS]* (Nov. 2020).

[35] Gear, C. Multirate methods for ordinary differential equations. Tech. Rep. UIUCDCS-F-74-880, Department of Computer Sciences, University of Illinois, 1974.

[36] Gear, C. W., and Wells, D. R. Multirate linear multistep methods. *BIT 24*, 4 (Dec. 1984), 484–502.

[37] Grote, M., and Mitkova, T. Explicit local time-stepping methods for time-dependent wave propagation. In *Direct and Inverse Problems in Wave Propagation and Applications* (2013), I. Graham, U. Langer, J.-M. Melenk, and M. Sini, Eds., de Gruyter, pp. 187–218.

[38] Grote, M. J., and Mitkova, T. Explicit local time-stepping methods for Maxwell's equations. *Journal of Computational and Applied Mathematics 234*, 12 (2010), 3283–3302.

[39] Grote, M. J., and Mitkova, T. High-order explicit local time-stepping methods for damped wave equations. *Journal of Computational and Applied Mathematics 239* (2013), 270–289.

[40] Günther, and Sandu, A. Multirate linearly-implicit gark schemes. *arXiv:2102.10203v1 [na, math]* (2021).

[41] Günther, M., Kværnø, A., and Rentrop, P. Multirate partitioned Runge-Kutta Methods. *BIT Numerical Mathematics 41*, 3 (June 2001), 504–514.

[42] Günther, M., and Sandu, A. Multirate generalized additive Runge–Kutta methods. *Numer. Math. 133*, 3 (July 2016), 497–524.

[43] Günther, M., and Rentrop, P. Multirate ROW methods and latency of electric circuits. *Applied Numerical Mathematics 13*, 1 (1993), 83–102.

[44] Hairer, E., Lubich, C., and Wanner, G. *Geometric numerical integration: structure-preserving algorithms for ordinary differential equations*, vol. 31. Springer Science & Business Media, 2006.

[45] Hairer, E., Nørsett, S. P., and Wanner, G. *Solving Ordinary Differential Equations I: Nonstiff Problems*, second ed. Springer Series in Computational Mathematics. Springer-Verlag, Berlin Heidelberg, 1993.

[46] Hairer, E., and Wanner, G. *Solving Ordinary Differential Equations II: Stiff and Differential-Algebraic Problems*, second ed. Springer Series in Computational Mathematics. Springer-Verlag, Berlin Heidelberg, 1996.

[47] Hewitt, H. T., Bell, M. J., Chassignet, E. P., Czaja, A., Ferreira, D., Griffies, S. M., Hyder, P., McClean, J. L., New, A. L., and Roberts, M. J. Will high-resolution global ocean models benefit coupled predictions on short-range to climate timescales? *Ocean Modelling 120* (2017), 120–136.

[48] Hochbruck, M., and Lubich, C. On Krylov subspace approximations to the matrix exponential operator. *SIAM J. Numer. Anal. 34* (1997), 1911–1925.

[49] Hochbruck, M., Lubich, C., and Selhofer, H. Exponential integrators for large systems of differential equations. *SIAM J. Sci. Comput. 19* (1998), 1552–1574.

[50] Hochbruck, M., and Ostermann, A. Explicit exponential Runge–Kutta methods for semilinear parabolic problems. *SIAM J. Numer. Anal. 43*, 3 (Jan. 2005), 1069–1090.

[51] Hochbruck, M., and Ostermann, A. Exponential Runge-Kutta methods for parabolic problems. *Applied Numerical Mathematics 53*, 2-4 (2005), 323–339.

[52] Hochbruck, M., and Ostermann, A. Exponential integrators. *Acta Numerica 19* (2010), 209–286.

[53] Hochbruck, M., and Ostermann, A. Exponential multistep methods of Adams-type. *BIT Numerical Mathematics 51*, 4 (2011), 889–908.

[54] Hochbruck, M., Ostermann, A., and Schweitzer, J. Exponential Rosenbrock-type methods. *SIAM J. Numer. Anal. 47* (2009), 786–803.

[55] Hundsdorfer, W., and Savcenco, V. Analysis of a multirate theta-method for stiff odes. *Applied Numerical Mathematics 59*, 3 (2009), 693–706. Selected Papers from NUMDIFF-11.

[56] Kennedy, C. A., and Carpenter, M. Diagonally implicit Runge–Kutta methods for ordinary differential equations. a review, 2016.

[57] Kennedy, C. A., and Carpenter, M. H. Additive Runge–Kutta schemes for convection–diffusion–reaction equations. *Applied Numerical Mathematics 44*, 1-2 (Jan. 2003), 139–181.

[58] Kennedy, C. A., and Carpenter, M. H. Higher-order additive Runge–Kutta schemes for ordinary differential equations. *Applied Numerical Mathematics 136* (Feb. 2019), 183–205.

[59] Knoth, O., and Arnold, M. Numerical solution of multiscale problems in atmospheric modeling. *Applied numerical mathematics 62* (2012).

[60] Knoth, O., and Wensch, J. Generalized split-explicit Runge-Kutta methods for the compressible Euler equations. *Monthly Weather Review 142*, 5 (2014), 2067.

[61] Knoth, O., and Wolke, R. Implicit-explicit Runge–Kutta methods for computing atmospheric reactive flows. *Applied Numerical Mathematics 28*, 2 (1998), 327 – 341.

[62] Kutta, W. Beitrag zur näherungsweisen integration totaler differentialgleichungen. *Zeitschrift für Math. u. Phys. 46* (1901), 435–453.

[63] Kværnø, A. Stability of multirate Runge-Kutta schemes. *International Journal of Differential Equations and Applications 1* (2000), 97–105.

[64] Kværnø, A., and Rentrop, P. Low order multirate Runge–Kutta methods in electric circuit simulation, 1999.

[65] Luan, V. T. *High-order exponential integrators.* PhD thesis, University of Innsbruck, 2014.

[66] Luan, V. T. Fourth-order two-stage explicit exponential integrators for time-dependent PDEs. *Applied Numerical Mathematics 112* (2017), 91–103.

[67] Luan, V. T. Derivation of new efficient exponential Runge–Kutta methods. *BIT, submitted* (2019).

[68] Luan, V. T., Chinomona, R., and Reynolds, D. R. A new class of high-order methods for multirate differential equations. *SIAM Journal on Scientific Computing 42*, 2 (Jan. 2020), A1245–A1268.

[69] Luan, V. T., and Ostermann, A. Exponential B-series: The stiff case. *SIAM J. Numer. Anal. 51* (2013), 3431–3445.

[70] Luan, V. T., and Ostermann, A. Explicit exponential Runge–Kutta methods of high order for parabolic problems. *Journal of Computational and Applied Mathematics 256* (Jan. 2014), 168–179.

[71] Luan, V. T., and Ostermann, A. Exponential Rosenbrock methods of order five — construction, analysis and numerical comparisons. *Journal of Computational and Applied Mathematics 255* (Jan. 2014), 417–431.

[72] Luan, V. T., and Ostermann, A. Stiff order conditions for exponential Runge–Kutta methods of order five. In *Modeling, Simulation and Optimization of Complex Processes - HPSC 2012* (2014), H. B. et al., Ed., Springer, pp. 133–143.

[73] Luan, V. T., and Ostermann, A. Parallel exponential Rosenbrock methods. *Comput. Math. Appl. 71* (2016), 1137–1150.

[74] Luan, V. T., Pudykiewicz, J. A., and Reynolds, D. R. Further development of efficient and accurate time integration schemes for meteorological models. *Journal of Computational Physics 376* (January 2019), 817–837.

[75] Marchuk, G. I. Some application of splitting-up methods to the solution of mathematical physics problems. *Aplikace Matematiky 13*, 2 (1968), 103–132.

[76] McLachlan, R. I., and Quispel, G. R. W. Splitting methods. *Acta Numerica 11* (Jan. 2002), 341–434.

[77] Niesen, J., and Wright, W. M. Algorithm 919: A Krylov subspace algorithm for evaluating the φ-functions appearing in exponential integrators. *ACM Transactions on Mathematical Software (TOMS) 38*, 3 (2012), 22.

[78] Rainwater, G., and Tokman, M. A new class of split exponential propagation iterative methods of Runge–Kutta type (sEPIRK) for semilinear systems of ODEs. *Journal of Computational Physics 269* (July 2014), 40–60.

[79] Reynolds, D. R. Example programs for ARKode. Tech. Rep. LLNL-SM-668082, Lawrence Livermore National Laboratory, 2016.

[80] Rice, J. R. Split Runge-Kutta method for simultaneous equations. *J. RES. NATL. BUR. STAN. SECT. B. MATH. MATH. PHYS. 64B*, 3 (July 1960), 151.

[81] Roberts, S., Loffeld, J., Sarshar, A., Woodward, C. S., and Sandu, A. Implicit multirate GARK methods. *arXiv:1910.14079 [cs, math]* (Oct. 2019).

[82] Roberts, S., Popov, A. A., Sarshar, A., and Sandu, A. "a fast time-stepping strategy for dynamical systems equipped with a surrogate model. *arxiv:2011.03688v2 [math.NA]* (2020).

[83] Roberts, S., Sarshar, A., and Sandu, A. Coupled multirate infinitesimal GARK schemes for stiff systems with multiple time scales. *arxiv:1812.00808 [math.NA]* (2018).

[84] Roberts, S., Sarshar, A., and Sandu, A. Coupled multirate infinitesimal GARK schemes for stiff systems with multiple time scales. *SIAM Journal on Scientific Computing 42*, 3 (2020), A1609–A1638.

[85] Roberts, S., Sarshar, A., and Sandu, A. Parallel implicit-explicit general linear methods. *arXiv:2002.00868 [cs, math]* (Apr. 2020).

[86] Ropp, D. L., and Shadid, J. N. Stability of operator splitting methods for systems with indefinite operators: Reaction-diffusion systems. *Journal of Computational Physics 203*, 2 (Mar. 2005), 449–466.

[87] Sandu, A. A class of multirate infinitesimal GARK methods. *arxiv:1808.02759 [cs.NA]* (2018).

[88] Sandu, A. A class of multirate infinitesimal GARK methods. *SIAM Journal on Numerical Analysis 57*, 5 (Jan. 2019), 2300–2327.

[89] Sandu, A., and Constantinescu, E. M. Multirate explicit Adams methods for time integration of conservation laws. *J Sci Comput 38*, 2 (Feb. 2009), 229–249.

[90] Sandu, A., and Constantinescu, E. M. Multirate time discretizations for hyperbolic partial differential equations. *AIP Conference Proceedings 1168*, 1 (2009), 1411–1414.

[91] Sandu, A., and Günther, M. A class of generalized additive Runge–Kutta methods. *arxiv:1310.5573 [math.NA]* (2013).

[92] Sandu, A., and Günther, M. A generalized-structure approach to additive Runge–Kutta methods. *SIAM J. Numer. Anal. 53*, 1 (Jan. 2015), 17–42.

[93] Sarshar, A., Roberts, S., and Sandu, A. Design of high-order decoupled multirate GARK schemes. *arxiv:1804.07716 [cs.NA]* (2018).

[94] Sarshar, A., Roberts, S., and Sandu, A. Design of high-order decoupled multirate GARK schemes. *SIAM Journal on Scientific Computing 41*, 2 (2019), A816–A847.

[95] Savcenco, V. Comparison of the asymptotic stability properties for two multirate strategies. *Journal of Computational and Applied Mathematics 220*, 1 (2008), 508–524.

[96] Savcenco, V. Construction of a multirate RODAS method for stiff ODEs. *Journal of Computational and Applied Mathematics 225*, 2 (2009), 323–337.

[97] Savcenco, V., Hundsdorfer, W., and G. Verwer, J. A multirate time stepping strategy for stiff ordinary differential equations. *BIT 47* (03 2007), 137–155.

[98] Schlegel, M., Knoth, O., Arnold, M., and Wolke, R. Multirate Runge–Kutta schemes for advection equations. *Journal of Computational and Applied Mathematics 226*, 2 (Apr. 2009), 345–357.

[99] Schlegel, M., Knoth, O., Arnold, M., and Wolke, R. Multirate implicit-explicit time integration schemes in atmospheric modelling. *AIP Conference Proceedings 1281*, 1 (2010), 1831–1834.

[100] Schlegel, M., Knoth, O., Arnold, M., and Wolke, R. Implementation of multirate time integration methods for air pollution modelling. *Geoscientific Model Development 5* (2012), 1395–1405.

[101] Schlegel, M., Knoth, O., Arnold, M., and Wolke, R. Numerical solution of multiscale problems in atmospheric modeling. *Appl. Numer. Math. 62*, 10 (Oct. 2012), 1531–1543.

[102] Sexton, J. M., and Reynolds, D. R. Relaxed multirate infinitesimal step methods. *arxiv:1808.03718 [math.NA]* (2018).

[103] Strang, G. On the construction and comparison of difference schemes. *SIAM J. Numer. Anal. 5*, 3 (Sept. 1968), 506–517.

[104] Tokman, M. A new class of exponential propagation iterative methods of Runge–Kutta type (EPIRK). *Journal of Computational Physics 230*, 24 (Oct. 2011), 8762–8778.

[105] Tokman, M., Loffeld, J., and Tranquilli, P. New adaptive exponential propagation iterative methods of Runge–Kutta type. *SIAM J. Sci. Comput. 34*, 5 (Jan. 2012), A2650–A2669.

[106] Verner, J. H. Explicit runge-kutta methods with estimates of the local truncation error. *SIAM Journal on Numerical Analysis 15*, 4 (1978), 772–790.

[107] Wensch, J., Knoth, O., and Galant, A. Multirate infinitesimal step methods for atmospheric flow simulation. *BIT Numer. Math. 49*, 2 (June 2009), 449–473.

[108] Yoshida, H. Construction of higher order symplectic integrators. *Physics Letters A 150* (11 1990), 262–268.

[109] Zhang, H., Sandu, A., and Blaise, S. Partitioned and implicit–explicit general linear methods for ordinary differential equations. *J. Sci. Comput. 61*, 1 (Oct. 2014), 119–144.

[110] Zhang, H., Sandu, A., and Blaise, S. High order implicit-explicit general linear methods with optimized stability regions. *SIAM J. Sci. Comput. 38*, 3 (Jan. 2016), A1430–A1453.

[111] Zharovsky, E., Sandu, A., and Zhang, H. A class of implicit-explicit two-step Runge–Kutta methods. *SIAM Journal on Numerical Analysis 53*, 1 (2015), 321–341.