UNIVERSITY OF CALIFORNIA, MERCED

Construction, analysis, and application of novel exponential time integrators for stiff problems

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

in

Applied Mathematics

by

Valentin Dallerit

Committee in charge:

Mayya Tokman, Chair

Stéphane Gaudreault

John Loffeld

Noemi Petra

Maxime Theillard

2022

The dissertation of Valentin Dallerit is approved, and it is acceptable in quality and form for publication on microfilm and electronically:

(Stéphane Gaudreault)

(John Loffeld)

(Noemi Petra)

(Maxime Theillard)

(Mayya Tokman, Chair)

University of California, Merced

2022

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGEMENTS

First and foremost, I would like to express my sincere gratitude to my advisor, Mayya Tokman, for her assistance and support at every stage of the thesis. Her invaluable advice made a huge difference in my work. I have a lot of appreciation for the guidance and feedback she gave me throughout the years. She gave me the freedom to explore some of my ideas while still being there to guide me when I needed it the most. I also thank my committee members Stéphane Gaudreault, John Loffeld, Noemi Petra, and Maxime Theillard for their feedback during the course of this work, and for taking the time to serve on my committee. My appreciation also goes to my family and friends for their continuous encouragement and support. They made this thesis possible and much more enjoyable.

The work in Chapter 2 was supported in part by the grant #1115978 from the National Science Foundation, Computational Mathematics Program

The work in Chapter 3 was partly supported by the NSF grants DMS-1720495 and DMS-2012875 and through the summer internship at Lawrence Livermore National Laboratory.

The material of Chapter 2 is reprinted with permission as it appears in Gaudreault, S., Charron, M., Dallerit, V., & Tokman, M. (2022). High-order numerical solutions to the shallow-water equations on the rotated cubed-sphere grid. *Journal of Computational Physics*, *449*, 110792.

The material of Chapter 3 is reprinted with permission of the material in Dallerit, V., Buvoli, T., Tokman, M., & Gaudreault, S. (2022). Second-order rosenbrock-exponential (ROSEXP) methods for partitioned differential equations. *to be submitted to SIAM Journal on Scientific Computing*.

The material of Chapter 5 is reprinted with permission as it appears in Dallerit, V., Tokman, M., & Joseph, I. (2022). Exponential integrators for non-linear diffusion. https://doi.org/10.2172/1860647.

Mayya Tokman directed and supervised research which forms the basis for the dissertation.

Curriculum Vitae

Valentin Dallerit

Ph.D. Candidate
Applied Mathematics
University of California, Merced
vdallerit@ucmerced.edu

RESEARCH INTERESTS

- Numerical Analysis
- Scientific Computing
- High-performance Computing

EDUCATION

University of California, Merced *August 2017 - present (expected August 2022)*
Ph.D. candidate in Applied Mathematics – Advisor: Mayya Tokman

Université Lyon 1 - École centrale de Lyon *September 2014 - September 2015*
Master's degree in Applied Mathematics with specialization in geosciences and the environnement

Polytech Nantes and Polytech Lyon *September 2010 - September 2015*
Engineering diploma and Master's degree in Applied Mathematics and Modeling

EXPERIENCE

University of California, Merced August 2017 - persent
Graduate Research Assistant / Teaching Assistant *Merced, CA*

Research:

- Introduced and analyzed new numerical methods for time integration of large scale complex stiff systems
- Implemented and tested numerical models in Matlab, Python and C++/ MPI for serial and parallel architectures
- Applied novel numerical techniques to study scientific problems in such fields as climate and weather prediction, plasma physics, and fluid dynamics

Teaching Assistant:

- Calculs III, Fall 2017 & Spring 2018
- Numerical Methods for Scientists and Engineers, Fall 2019 & Spring 2020

Lawrence Livermore National Laboratory June 2019 - August 2019
Intern *Livermore, CA*

· Studied stability properties of numerical methods for partitioned system of differential equations
· Developed A-stable Exponential-Implicit method for nonlinear-nonlinear partitioned problem

Lawrence Livermore National Laboratory June 2018 - August 2018
Intern *Livermore, CA*

· Studied multi-rate time integration in the context of Adaptive Mesh Refinement grids
· Implemented advection-diffusion PDE solver using AMReX framework

IFP Energies Nouvelles October 2015 - May 2017
Research Engineer *Solaize, France*

· Developed algorithms and software for simulation of the oil refinement process
· Created models and tools to assess uncertainties of refinement
· Worked to improve experimental design

Canadian Meteorological Centre, Dept. of Environment April 2015 - September 2015
Physical Scientist *Montréal, Canada*

· Developed numerical models for weather and climate prediction
· Designed and implemented high-order spatial discretization schemes (Discontinuous Galerkin and CPR) for shallow water equations
· Studied performance of the new numerical models compared to benchmarks

IFP Énergies Nouvelles July 2013 - February 2014
Intern *Solaize, France*

· Redesigned numerical model of the oil refinement process simulator to improve efficiency and modularity of the code
· Developed new kinetic models for oil refinement process
· Developed new models to accommodate new reaction catalysts
· Worked with clients to design new user interface for the simulator

PUBLICATIONS

· Gaudreault S., Charron M., **Dallerit V.**, Tokman M. High-order numerical solutions to the shallow-water equations on the rotated cubed-sphere grid, J. Comput. Phys. (2022), https://doi.org/10.1016/j.jcp.2021.110792
· **Dallerit V.**, Tokman M., Joseph I., Exponential integrators for non-linear diffusion, Technical report (2022), https://doi.org/10.2172/1860647
· **Dallerit V.**, Buvoli T., Tokman M., Gaudreault S. New Implicit-Exponential (IMEXP) Methods for Partitioned Nonlinear Differential Equation, *(to be submitted in July 2022)*
· **Dallerit V.**, Tokman M. A new approach to deriving stiffly accurate time integration schemes, *(to be submitted in September 2022)*

· Guédon J.P., Autrusseau F., Amouriq Y., Bléry P., Bouler J., Weiss P., Barbarin F., Dallet T., **Dallerit V.**. Exploring relationships between fractal dimension and trabecular bone characteristics, *Proc. SPIE 8317, Medical Imaging 2012: Biomedical Applications in Molecular, Structural, and Functional Imaging*

PRESENTATIONS

· **Dallerit V.** (2021) Introduction to exponential time integration and applications to nonlinear diffusion, *Lawrence Livermore National Laboratory, Fusion group meeting, Livermore*

· **Dallerit V.**, Gaudreault S., Charron M., Tokman M. (2021). New numerical methods for atmospheric models, *UC Merced SIAM Student Chapter SAMPLe Talk, Merced*

· **Dallerit V.**, Gaudreault S., Charron M., Tokman M. (2021). Numerical Solutions to the Shallow-Water Equations on the Cubed-Sphere Grid, *SIAM Conference on Computational Science and Engineering 2021, Online*

· **Dallerit V.**, Buvoli T., Loffeld J., Tokman M. (2019). New Multirate Implicit-Exponential (IMEXP) Methods for Partitioned Nonlinear Differential Equations, *Lawrence Livermore Summer Student Poster Symposium 2019, Livermore*

· **Dallerit V.**, Loffeld J. (2019). Towards the understanding of multirate schemes on Adaptive Mesh Refinement grids, *SIAM Conference on Computational Science and Engineering 2019, Spokane*

· **Dallerit V.**, Loffeld J. (2018). Towards the understanding of multirate schemes on Adaptive Mesh Refinement grids, *Lawrence Livermore Summer Student Poster Symposium 2018, Livermore*

· **Dallerit V.**, Tokman M. (2018). A General Time Integration Framework, *Integrating the Integrators for Nonlinear Evolution Equations: from Analysis to Numerical Methods, High-Performance-Computing and Applications, Banff*

AWARDS AND FELLOWSHIPS

Graduate Dean's Dissertation Fellowship 2022
University of California - Merced

Excellence in Research Award 2021
Applied Math Graduate Program, University of California - Merced

Excellence in Service Award 2021
Applied Math Graduate Program, University of California - Merced

School of Natural Science Dean Summer Research Fellowship 2021
University of California - Merced

Applied Mathematics Summer Research Fellowship 2020, 2021
University of California - Merced

TECHNICAL SKILLS

Programming Languages	C, C++, Python, Fortran, Matlab, R, Julia
High Performance Computing	MPI, OpenMP, GPGPU (openCL), PETSc
Systems and software	GNU/Linux, Bash script, LaTeX, Git/SVN

ABSTRACT OF THE DISSERTATION

Construction, analysis, and application of novel exponential time integrators for stiff problems

by Valentin Dallerit

Doctor of Philosophy in Applied Mathematics

University of California Merced, 2022

Committee Chair: Mayya Tokman

Stiff systems of ordinary differential equations (ODEs) play an essential role in the temporal integration of many scientific and engineering problems. With the increase in complexity and size of these problems, it is crucial to have efficient time integration methods. In this dissertation, we focus on the construction, analysis, and applications of exponential time integrators. We address the efficiency of these methods in three main directions. First, we construct new efficient exponential multi-step methods. These methods are carefully derived to combine the accuracy of high-order schemes with the efficiency of low-order methods. We validate and apply the new schemes to a simplified atmospheric model and show that they can capture more details in the solution with a limited computational cost. Secondly, we develop a new ansatz for deriving partitioned implicit-exponential integrators. These methods are applied to problems where the forcing term of the system is comprised of stiff additive terms. We propose a new way of analyzing the stability properties of partitioned methods. Our results show that the increased accuracy and stability of our schemes offer improved efficiency compared to state-of-the-art methods. Finally, we introduce a new theoretical framework for deriving stiffly resilient exponential methods. This framework addresses the complexity of deriving exponential schemes and enforces advantageous properties. We establish an analytical solution for the order conditions used to derive methods. We derive a new class of exponential integrators and new schemes with an order of convergence higher than currently available methods. We also evaluate the performance of exponential time integration for the reduced magnetohydrodynamics equations. Our initial results focus on a simplified model capturing the essence of the full equations. We show that exponential integrators are a valuable alternative to the schemes currently used for this problem.

Chapter 1

Introduction

A wide range of scientific and engineering applications requires computer simulation of complex time-dependent partial differential equations (PDEs). A common approach to approximate the solution of such equations is to use the method of lines (Schiesser, 1991). This method first spatially discretizes the problem leading to a large system of ordinary differential equations (ODEs) of the form:

$$\begin{cases} y'(t) = f(y(t)) \\ y(t_0) = y_0 \end{cases} \qquad \text{where: } y(t) \in \mathbb{R}^N \text{ and } t_0 \leq t \leq t_f \tag{1.1}$$

This system is then advanced in time from the initial solution y_0 at time t_0 to the final time using a time integration method. Many scientific problems tend to have a wide range of time scales in their forcing terms. For example, an atmospheric model needs to be able to represent a wide range of meteorological conditions. Another example can be found in a plasma physics model where the phenomenon developing close to the boundary can happen much faster than in the rest of the domain. Because of this range of time scales, the Jacobian of the right-hand side f of eq. (1.1) has both large and small eigenvalues. In other words the corresponding problem (1.1) is stiff. The stiffness in these problems can be caused by the entire right-hand side or, in some cases, by a specific forcing term. For instance, in an advection-diffusion PDE, the diffusion term might be dominant and cause the stiffness of the problem, while the advection term alone is not stiff. Furthermore, in recent years, the use of high-order spatial discretization methods coupled with high

resolution or locally refined grids have been common practice in many applications (Berger & Oliger, 1982; Godenschwager et al., 2013; Zhang et al., 2019; Holec et al., 2022). These highly accurate approximations of the spatial terms induce more stiffness and increase the dimension of the system of ODEs. This trend is likely to continue in the future with the emergence of exascale computing. Highly parallel architecture has been shown to be more efficient with high-order spatial methods (Sjogreen et al., 2009; Dongarra et al., 2014; Loffeld & Hittinger, 2019) and allows larger problems to be undertaken. Such large-scale problems will require accurate and efficient numerical methods to be solved.

Even though highly accurate numerical methods in space are common in applications, time integration techniques have not necessarily kept up and in many applications, researchers still use low order temporal integrators. The system of ODEs obtained from the spatial discretization is often approximated using either explicit or implicit time integration methods (J. C. Butcher, 2016; Wanner & Hairer, 1996). Due to their limited stability, explicit methods suffer from strict restrictions on the time step and therefore are not efficient when used with stiff problems. On the other hand, each time step of an implicit method requires solving large systems of nonlinear equations. Approximating such a solution is computationally expensive if no preconditioner is available for the problem. Preconditioners for complex multi-physics problems are challenging to derive and tend to scale poorly with large-scale systems. In recent years, the growing demand for more accurate and efficient time integration methods has led to the development of new methods such as exponential integrators.

In the 1960s, exponential time integration was introduced by Certaine (Certaine, 1960) as an alternative to explicit and implicit methods. These methods can solve linear problems exactly, have good stability properties, and can be computationally more efficient than implicit methods (Loffeld & Tokman, 2013). Some history and review of exponential time integration can be found in (Minchev & Wright, 2005; Hochbruck & Ostermann, 2010). When using an exponential integrator, the action of exponential-like φ_k matrix functions on vectors needs to be calculated to advance the solution from the current time to the next time step. For

example, one of the simplest exponential method is the 2^{nd} order exponential Euler method which uses the function $\varphi_1(A) = (e^A - I)A^{-1} = \sum_{i=0}^{\infty} \frac{A^i}{(i+1)!}$. Using this method, the solution of eq. (1.1) can be advanced in time from a known solution y_n using the update $y_{n+1} = y_n + \varphi_1(hJ)hf(y_n)$ where J is the Jacobian matrix of f at y_n. In general, the φ_k functions can be defined as the series $\varphi_k(A) = \sum_{i=0}^{\infty} \frac{A^i}{(k+i)!}$ similar to the exponential function. Higher-order φ_k functions are used with higher order schemes. The argument of these φ_k functions is either the Jacobian matrix of the full right-hand side f or a part of it. This matrix has dimension $N \times N$ where N is the size of the problem and, therefore, is large for the problems we are interested in. Without any efficient method to do this computation for large problems, exponential methods were only used for systems with a few equations or specific structures. In recent years, with the improvements in numerical linear algebra, exponential integrators are now becoming more attractive for stiff problems (van der Vorst, 1987; Saad, 1992; Higham, 2008). Since the introduction of exponential integrators, new classes of methods, improving the efficiency and accuracy, have been derived such as exponential Runge-Kutta (Hochbruck & Ostermann, 2005), exponential propagation iterative methods (Tokman, 2006, 2011), exponential Rosenbrock (Hochbruck, Ostermann, & Schweitzer, 2008; Luan & Ostermann, 2014) and exponential multi-step (Hochbruck & Ostermann, 2011) methods. These methods have been used in a wide range of applications including atmospheric models (Gaudreault & Pudykiewicz, 2016; Calandrini, Pieper, & Gunzburger, 2021), ocean models (Calandrini, Pieper, & Gunzburger, 2020; Pieper, Sockwell, & Gunzburger, 2019), nonlinear Schrodinger equations (de la Hoz & Vadillo, 2008; Hederi et al., 2016) and computer graphics (Michels, Luan, & Tokman, 2017; Y. J. Chen et al., 2020; Ascher et al., 2021). An introduction to the derivation of exponential methods and a description of a numerical method to compute the φ_k functions are available in Appendix A. Despite the advances in exponential time integration, further improvements in the efficiency of these techniques are possible. However, the construction of new methods is still a challenging task. Two main approaches are currently available to derive exponential schemes. The first approach is the classical order conditions commonly used to derive explicit and implicit methods.

These conditions are obtained by comparing the Taylor expansions of y_{n+1}, the numerical schemes, and $y(t_{n+1})$, the exact solution at the time $t_{n+1} = t_n + h$ where h is the time step. By matching these expansions up to the desired order of convergence p, the local error $e(h) = y(t_{n+1}) - y_{n+1}$ is bounded by $e(h) = O(h^{p+1})$. When computing the Taylor expansions, it is assumed that the solution of eq. (1.1) and its first p derivatives are smooth. However, when applied to stiff problems, the Jacobian matrix of f has large eigenvalues. In the limit where the stiffness goes to infinity, this can break the assumptions of the classical order conditions, and methods derived in this way can suffer from order reduction in our experiments. This behavior is presented in (Hochbruck & Ostermann, 2005) and the stiff order conditions are introduced to eliminate this problem. In this paper, the authors are deriving methods under the assumption that the Jacobian matrix can be unbounded, but the solution is still smooth. Such methods are called stiffly accurate, and several exponential methods have been derived using this framework (Hochbruck & Ostermann, 2011; Rainwater & Tokman, 2016; Luan, 2020). Both the classical and stiff order conditions yield a system of nonlinear equations that must be solved analytically. This system is hard to solve, and no generic way to find a solution has been presented before. Therefore each new method requires a significant amount of work and is derived on a case-by-case basis. As we will demonstrate in this thesis, it is possible to introduce an intermediate class of methods that are better suited for stiff problems compared to classically derived schemes while at the same time are easier to construct than fully stiffly accurate integrators.

As mentioned before, efficient time integration techniques are required to deal with modern stiff problems. One potential approach is to derive methods specifically for problems where the right-hand side f can be partitioned. The forcing terms of physical models can usually be partitioned as the sum of several forcing terms that correspond to different parts of the physics. In some cases, very efficient solvers have been developed for specific forcing terms. These solvers might, however, lose their efficiency when applied to the full problem. For this reason, we would like to take advantage of the system's structure in the construction of

numerical time integration methods. With such methods, we would like to treat the different components in different ways and use efficient solvers when available. A variety of techniques for partitioned problems have been explored such as splitting methods (MacNamara & Strang, 2016) and partitioned integrators (Belytschko, Yen, & Mullen, 1979; Rentrop, 1985; Ascher, Ruuth, & Wetton, 1995; Kennedy & Carpenter, 2003). Splitting methods have been extensively used in the literature to solve such problems (Gradinaru, 2007; Holden, Lubich, & Risebro, 2013), but their construction for orders larger than two tends to be challenging (Blanes & Casas, 2005). On the other hand, partitioned methods can offer a more straightforward extension to higher-order methods. Some of the best-known partitioned integrators are implicit-explicit (IMEX) methods (Ascher, Ruuth, & Spiteri, 1997; Higueras et al., 2014) which have been used in a wide range of applications (Pareschi & Russo, 2005; Keyes et al., 2013; Hundsdorfer, Verwer, & Hundsdorfer, 2003; Gardner et al., 2018). IMEX schemes treat one component of the forcing term implicitly and the other explicitly. This requires that only one of the forcing terms is responsible for stiffness in the system to avoid stability problems and therefore reduces the range of applications. To apply partitioned methods to problems where both components are stiff, implicit-implicit partitioned methods have been derived (Sandu & Günther, 2015). These methods, however, suffer from similar limitations as non-partitioned implicit methods and are usually computationally expensive. More recently, implicit-exponential (IMEXP) integration has been introduced (Luan, Tokman, & Rainwater, 2017; Ascher et al., 2021; Y. J. Chen et al., 2020) to combine the advantages of implicit and exponential methods. With such methods, it is possible to benefit from an efficient preconditioner available for the implicit part and use the efficiency and stability of exponential methods for the other part. However, the stability of these methods has not been studied, and they are currently limited to first and second order.

In this dissertation, we address the challenges of deriving and applying efficient exponential integrators in several directions. First, in Chapter 2, we construct new efficient exponential multi-step methods that combine the accuracy of high-order schemes with the efficiency of low-order methods. This is achieved by designing the

methods so that the computation of all the φ_k functions can be reduced to a single matrix-exponential-vector product in each step. This product is then approximated using the Krylov-based KIOPS (Krylov with incomplete orthogonalization procedure) algorithm presented in appendix A.2. As a result, the cost per iteration is similar to the cost of the exponential Euler method, independently of the order of accuracy. These schemes are based on the idea of multi-step methods and use the information available from the previous time steps to improve the accuracy. The new methods are applied to the shallow water equations on the sphere. These equations correspond to a simplified model of the atmosphere. A space-time tensor formalism is used to express the equations of motion covariantly and to describe the geometry of the rotated cubed-sphere grid. The spatial discretization is done using the direct flux reconstruction method, which is an alternative formulation of the discontinuous Galerkin approach. The proposed numerical algorithms produce realistic results and have the ability to perform accurate simulations with very large time steps. We show that the combination of high-order spatial and matching high-order temporal discretization leads to better accuracy and is capable of capturing more details in the solution. The convergence of the spatial and temporal patterns is well behaved as the order of accuracy is increased. Results are comparable to those obtained from more complicated methods. Experiments show that mass is conserved at machine precision. The performance of the new numerical methods is evaluated using a set of standard benchmark tests.

In Chapter 3, we introduce a new framework for deriving partitioned implicit-exponential integrators and construct several time integrators of this type. The new approach is suited for solving systems of ODEs where the forcing term is comprised of two additive nonlinear terms where both components can be stiff. We specifically derive new integrators that are A-stable, and have 2^{nd} order accuracy. Moreover, these methods are made efficient by only requiring one call to the linear system solver for the implicit part and one matrix-exponential-vector product for the exponential part at each time step. We also propose a novel approach to visualize the linear stability of partitioned schemes. This visualization allows a better understanding of the stability limitations of such methods. We analyze

the accuracy and stability of the new integrators and compare their performance to state of the art methods on a set of standard test problems. We show that our methods are more accurate and have better stability properties. The gained accuracy and stability of our methods provide an advantage in overall efficiency compared to existing methods.

In Chapter 4, we address the complexity of deriving stiffly accurate exponential methods that unlike classical techniques do not exhibit order reduction when applied to stiff problems. We introduce new order conditions and a new theoretical framework for the derivation of exponential integrators. We derive an analytical formula for the solution of the new order conditions making the derivation of exponential methods simple for arbitrary order. This framework is very general and can be used to derive exponential methods of many types. We present the derivation of exponential Runge-Kutta, exponential multi-step, and a new class of exponential multi-values methods. We demonstrate that the new order conditions can be interpreted as additionnal constraints on top of the classical order conditions. These extra constraints ensure that the dominant error terms of stiff problems are canceled. This has the effect of increasing the accuracy of the methods, and, similar to the stiff order condition, the methods derived in this framework are resilient to the stiffness of the problem. Such methods do not suffer from order reduction. We also show that methods derived in this framework minimize the number of matrix exponential computations. In addition, they have desirable properties such as continuous output and embedded lower order methods. All these reasons make the methods efficient and an interesting choice for applications.

Finally, in Chapter 5, we evaluate the performance of exponential time integrators for the solution of a nonlinear diffusion PDE. This equation is chosen as a simplified example of the system of reduced magnetohydrodynamics (RMHD). The RMHD system is a set of equations modeling an incompressible fluid plasma in a magnetic field. The solutions to the RMHD equations corresponding to the parameter regimes of interest often include boundary layers that are difficult to resolve numerically. The stiffness associated with such dynamics requires careful numerical treatment in the temporal integration. The system of RMHD equations

is usually approximated using low order implicit methods such as implicit Euler or implicit midpoint (Holec et al., 2022) as higher order implicit methods tend to be too computationally expensive. For this reason, there is a need for more accurate and efficient methods. In this project, we explore the use of exponential integrators as an alternative to the currently used implicit methods. Our experiments are done on simplified 1D and 2D nonlinear diffusion problems recreating the boundary layer present in the solution of the full equations. The spatial discretization is done with a finite element method using the *MFEM* library. We also added exponential integrators to this library as part of this project. Our results demonstrate that the use of exponential methods has the potential to improve the efficiency of the approximation of the RMHD equations compared to classical methods. Moreover, the accuracy of exponential methods can be improved without a significant impact on the performance.

Chapter 2

High-order numerical solutions to the shallow-water equations on the rotated cubed-sphere grid

The text of this chapter is a reprint of the material as it appears in Gaudreault, S., Charron, M., Dallerit, V., & Tokman, M. (2022). High-order numerical solutions to the shallow-water equations on the rotated cubed-sphere grid. *Journal of Computational Physics*, *449*, 110792

2.1 Introduction

A challenge of great interest in the numerical weather prediction community is to develop numerical methods for the governing equations that are conservative, highly accurate, geometrically flexible, computationally efficient, and simply formulated. Second-order numerical methods are often preferred in operational models due to their simplicity and robustness. High-order methods are sometimes perceived as less robust and complicated to implement. The aims of this paper are to present: 1) spatial numerical techniques that are almost as simple as finite differences; and 2) multistep exponential time integration methods with superior stability properties compared to explicit schemes.

For spatial discretization aspects, the direct flux reconstruction (DFR) scheme (J. Romero, Asthana, & Jameson, 2016) is applied to the cubed-sphere grid. The DFR method is a simplified formulation of a class of methods called flux reconstruction (FR) (Huynh, 2007). The DFR method is equivalent to the weak form of the nodal discontinuous Galerkin (NDG) scheme in one dimension and on multidimensional elements with tensor product bases (J. Romero, Asthana, & Jameson, 2016; Huynh, 2019). In this alternative formulation of the NDG method, the conservative equations are solved in differential form and the discretization is free from quadrature rules, resulting in a simple and computationally efficient algorithm. This method is also well suited to large geophysical fluid dynamics problems because it is local and has good conservation properties (Nair, Levy, & Lauritzen, 2011; Ullrich, 2014; Marras et al., 2016).

Time integration with high-order methods is a much more difficult problem. The implementation of high-order Eulerian techniques in operational models is made difficult by the Courant-Friedrichs-Lewy (CFL) condition that limits the time step in several explicit time integration schemes (Courant, Friedrichs, & Lewy, 1928). This motivated the recent investigation of exponential time integrators in geophysical applications (Clancy & Pudykiewicz, 2013; Gaudreault & Pudykiewicz, 2016; Gaudreault, Rainwater, & Tokman, 2018; Luan, Pudykiewicz, & Reynolds, 2019; Schreiber, Schaeffer, & Loft, 2019; Peixoto & Schreiber, 2019; Calandrini, Pieper, & Gunzburger, 2020; Shashkin & Goyman, 2020; Caliari et al., 2021; Calandrini, Pieper, & Gunzburger, 2021). These approaches allow for longer time steps and yield higher accuracy than traditional algorithms. In addition, these studies have shown that exponential integrators accurately calculate the entire spectrum of waves propagating in the atmosphere. Beside the obvious advantage of high accuracy, these methods also eliminate the need to divide the governing equations into linear and non-linear parts. This task is performed automatically by evaluating the action of the Jacobian operator. This consideration could greatly simplify the design of numerical models.

In this paper, new high-order exponential propagation iterative (EPI) methods are introduced and tested in combination with DFR for the shallow-water equa-

tions on the sphere. The paper is organized as follows. The governing equations in arbitrary coordinates are introduced in section 2.2. This sets the stage for section 2.3 and section 2.4 where the numerical schemes are described. Results from a series of numerical experiments are discussed in section 2.5. Conclusions and future works are presented in section 2.6.

2.2 General form of the equations of motion and the rotated cubed-sphere grid

Solving the shallow-water equations is often the first step towards the construction of a comprehensive numerical weather prediction model. Shallow-water dynamics represents a (2+1)-dimensional (i.e. two spatial and one temporal dimensions) nonlinear system of hyperbolic partial differential equations. They are obtained from the Euler equations by assuming that the depth of the fluid is small compared to the mean radius of the Earth, incompressibility, hydrostasy, weak stratification, and in the case of spherical geometry, that the spherical geopotential approximation holds.

Here, the shallow-water equations will be written in a space-time tensorial formalism. One advantage of using space-time tensor calculus and tensor-related objects is that the equations of motion may be expressed in arbitrary (possibly non-inertial and time-dependent) coordinates within a unified framework.

The synchronously covariant shallow-water equations for continuity and momentum in quasi-flux form are respectively

$$\frac{\partial}{\partial t}\left(\sqrt{g}H\right) + \frac{\partial}{\partial x^j}\left(\sqrt{g}Hu^j\right) = 0, \tag{2.1}$$

$$\begin{aligned}\frac{\partial}{\partial t}\left(\sqrt{g}Hu^i\right) + \frac{\partial}{\partial x^j}\left(\sqrt{g}\left[Hu^iu^j + \frac{1}{2}g_r h^{ij}H^2\right]\right) = \\ -2\sqrt{g}\,\Gamma^i_{j0}Hu^j - \sqrt{g}\,\Gamma^i_{jk}\left(Hu^ju^k + \frac{1}{2}g_r h^{jk}H^2\right) - \sqrt{g}Hg_r h^{ij}\frac{\partial h_B}{\partial x^j},\end{aligned} \tag{2.2}$$

where H is the fluid's thickness scalar field, h_B the height of the bottom orography, u^i $(i = 1, 2)$ the two components of the velocity field, g_r the constant effective gravitational acceleration, g the determinant of the covariant space-time metric

tensor, h^{ij} the space-only contravariant metric tensor, and Γ's the Christoffel symbols in space-time. Notice that the Coriolis effect is associated with the Christoffel symbols Γ^i_{j0}. The indices i, j take the values 1 and 2, and repeated indices are summed. The derivation of these equations is presented in Appendix B.1. The form of the governing equations provided by eqs. (2.1) and (2.2) will be discretized in the following sections.

The relevant space-time metric terms associated with the cubed-sphere coordinates on a rotating 2-sphere with radius a and constant angular velocity Ω are provided in Appendix B.2. It is assumed that the reader is familiar with the characteristics of the cubed-sphere grid, otherwise (Sadourny, 1972; Ronchi, Iacono, & Paolucci, 1996; Rančić, Purser, & Mesinger, 1996; Nair, Thomas, & Loft, 2005b, 2005a; Rossmanith, 2006; C. Chen & Xiao, 2008; Ullrich, Jablonowski, & Van Leer, 2010; Ullrich & Jablonowski, 2012; Bao, Nair, & Tufo, 2014; Nair, 2015) provide descriptions and details on its properties. In practical applications, it may be desirable to rotate a coordinate system to avoid the co-location of certain geographical points and grid peculiarities such as coordinate discontinuities. In Appendix B.2, an arbitrary rotation of the cubed-sphere coordinates is considered.

Coordinate lines at the interface of two panels may not be smooth as a result of the composite character of the global cubed-sphere coordinates. Although scalars have by definition the same value at a point on the interface of two panels with different coordinates, higher-rank tensor components are different if expressed with the coordinate basis of one or the other side of the interface. In Appendix B.3, consistency relations between all interfacing panels are provided for contravariant and covariant first-rank tensors.

2.3 Spatial discretization

The discontinuous Galerkin method is built on a weak integral formulation of the governing equations, where all unknown functions are approximated by high-order polynomials (see (Hesthaven & Warburton, 2007; Giraldo, 2020) for a review). This leads to the evaluation of several integrals using quadrature rules.

The recent direct flux reconstruction method (J. Romero, Asthana, & Jameson, 2016; J. D. Romero, 2017) is formulated using a differential form of the equations. Therefore, this method only requires Lagrange polynomials and avoids the need for quadrature rules, which simplifies its implementation. It has been shown that the DFR method results in a scheme equivalent to the weak form of the nodal discontinuous Galerkin method in one dimension and on multidimensional elements with tensor product bases (J. Romero, Asthana, & Jameson, 2016; Huynh, 2019). It should be noted that these ideas are not entirely new in geophysical applications since the 3rd order scheme presented in (C. Chen et al., 2015) may also be considered as a special case of the DFR method. In section 2.3.1, an overview of the basic properties of the method in one dimension is presented and its extension to two dimensions and curved geometry is discussed in section 2.3.2.

2.3.1 Direct flux reconstruction in one dimension

Consider the following conservation law:

$$\frac{\partial q}{\partial t} + \frac{\partial f(q)}{\partial x} = 0, \quad x \in \mathcal{D}. \tag{2.3}$$

The calculation domain $\mathcal{D}$ is divided into N_e non-overlapping elements $\mathcal{D}_j = [x_{j-\frac{1}{2}}, x_{j+\frac{1}{2}}]$, for $j = 1, ..., N_e$. The size of the element j is $\Delta_j = x_{j+\frac{1}{2}} - x_{j-\frac{1}{2}}$. In this subsection, all indices refer to positions in a one-dimensional grid and not to space-time indices as in section 2.2. Also, implicit summation over repeated indices is not assumed.

To conveniently treat a general non-uniform grid, a local coordinate variable ξ is defined by an affine transformation mapping each element $\mathcal{D}_j$ onto the so-called reference element $I_j = [-1, 1]$:

$$\xi(x) = \frac{2}{\Delta_j}(x - x_{j-\frac{1}{2}}) - 1 \tag{2.4}$$

with inverse

$$x(\xi) = \left(\frac{1-\xi}{2}\right) x_{j-\frac{1}{2}} + \left(\frac{1+\xi}{2}\right) x_{j+\frac{1}{2}}. \tag{2.5}$$

Derivatives in global and local coordinate systems are related by the chain rule:

$$\frac{\partial}{\partial x} = \frac{2}{\Delta_j}\frac{\partial}{\partial \xi}. \tag{2.6}$$

Similarly to several other high-order methods such as FR and NDG, a set of $N_s \geq 1$ solution points are defined within the reference element. These points are denoted ξ_k, for $k = 1, \ldots, N_s$. There are several ways to place these points, leading to schemes with different numerical properties (Jameson, Vincent, & Castonguay, 2012). In the present study, the Gauß-Legendre points have been chosen based on theoretical results which show the optimal nature of these solution points (Gassner & Kopriva, 2011; Witherden & Vincent, 2020). The placement of the Gauß-Legendre points corresponds to the roots of the Legendre polynomials. Although the solution and flux points are collocated, the method does not suffer from the so-called $2\Delta x$ numerical mode that typically affects the unstaggered finite difference schemes on uniform grids. In effect, the entries of the Gauß-Legendre differentiation matrices are all non-zero. This is in contrast with centered unstaggered finite differences on a uniform grid, where the coefficients vanish at the center of the stencil. The Runge phenomenon is also significantly mitigated by this choice of points.

The essence of the DFR method consists in approximating the exact solution q with an approximate solution $\hat{q}$. This approximate solution is obtained by taking the piecewise sum of N_e functions $\hat{q}_j$, each function being defined as a polynomial of degree $N_s - 1$ within the element $\mathcal{D}_j$ and exactly zero elsewhere. Thus, the approximate solution for a given element j can be represented on a reference element as

$$\hat{q}_j(\xi, t) = \sum_{k=1}^{N_s} \hat{q}_{j,k}(t)\, \ell_k(\xi), \tag{2.7}$$

where $\hat{q}_{j,k}(t)$ are coefficients defined at the solution points, and

$$\ell_k(\xi) = \prod_{\substack{l=1 \\ l \neq k}}^{N_s} \frac{\xi - \xi_l}{\xi_k - \xi_l} \tag{2.8}$$

are the Lagrange basis polynomials.

Similarly, the exact flux $f(q)$ in eq. (2.3) is approximated by the piecewise sum of polynomials $\hat{f}_j$ of degree $N_s - 1$ within each element and zero outside:

$$\hat{f}_j(\xi, t) = \sum_{k=1}^{N_s} \hat{f}_{j,k}(t)\, \ell_k(\xi), \tag{2.9}$$

where $\hat{f}_{j,k}(t) = f(\hat{q}_{j,k}(t))$. It is worth noting that even if $\hat{q}$ could be represented exactly with the basis functions, the term $\hat{f}$ is obtained from products of dynamical variables and the expansion (2.9) would not be exact in general. This could introduce aliasing errors which may require filtering.

At a given time t, the functions $\hat{q}$ and $\hat{f}$ are continuous within an element but are not uniquely defined at the element boundaries (i.e., at the points $\xi = \pm 1$). If for instance the flux at $\xi = -1$ on the element j is considered, its value may differ from the value at $\xi = 1$ on the element $j - 1$. These disparities constitute a Riemann problem. For smooth solutions, a simple upwinding might be an acceptable approximation for the common flux at the interface. If, however, the physical solution should include sharp gradients, the common flux is generally prescribed by an approximate Riemann solver borrowed from the finite volume methodology (see e.g. (LeVeque et al., 2002; Toro, 2013)). The study (Ullrich, Jablonowski, & Van Leer, 2010) adapted the Rusanov (Rusanov, 1962), Roe (Roe, 1981) and AUSM+-up (Liou, 2006) methods for the shallow-water equations. Their simulations showed that AUSM+-up provides the best overall accuracy when applied to various test cases, followed closely by the Roe solver. The Rusanov solver showed significantly worse performance in terms of accuracy and conservation error. However, the AUSM+-up method is much more computationally intensive than the Rusanov method, which explains why the latter is a more popular choice with discontinuous Galerkin methods. After exploring different possibilities, an adaptation of the basic AUSM solver (Liou & Steffen Jr, 1993) was found to be a good compromise between simplicity, accuracy and numerical efficiency. This Riemann solver is described in Appendix B.4.

To account for the interaction between adjacent elements, a continuous polynomial, denoted $F_j(\xi)$, is defined within each element. This continuous flux function must satisfy the following conditions:

1. $F_j(\xi)$ must take the value of the Riemann fluxes at both ends of the element $\mathcal{D}_j$:
$$F_j(-1) = \bar{f}\left(q^L_{j-\frac{1}{2}}, q^R_{j-\frac{1}{2}}\right), \quad F_j(1) = \bar{f}\left(q^L_{j+\frac{1}{2}}, q^R_{j+\frac{1}{2}}\right); \tag{2.10}$$
2. $F_j(\xi)$ must be a polynomial of degree $N_s + 1$;

3. $F_j(\xi)$ must approximate the discontinuous flux function $\hat{f}_j(\xi, t)$ at a given time t.

Condition 1 is necessary for the polynomial to be C^0. Condition 2 implies that the derivative of $F_j(\xi)$ is a polynomial of degree N_s and that the method has an accuracy of order N_s. To satisfy these two conditions, one defines the continuous polynomial over the extended set of solution points as the union of the set of interior solution points and the interface flux points, that is $\{\xi_0, \xi_1, \ldots, \xi_{N_s}, \xi_{N_s+1}\}$, with $\xi_0 = -1$, $\xi_{N_s+1} = 1$ and the usual Gauß-Legendre points for indices $1, \ldots, N_s$. Condition 3 may be satisfied by imposing that $F_j(\xi_k) = \hat{f}_{j,k}$ on the interior Gauß-Legendre points $k = 1, \ldots, N_s$. These three conditions are sufficient to define the following continuous flux polynomial:

$$F_j(\xi) = F_j(-1)\,\tilde{\ell}_0(\xi) + \left[\sum_{k=1}^{N_s} \hat{f}_{j,k}(t)\,\tilde{\ell}_k(\xi)\right] + F_j(1)\,\tilde{\ell}_{N_s+1}(\xi), \tag{2.11}$$

where $\tilde{\ell}_n$ are the Lagrange interpolation polynomial basis constructed from eq. (2.8) but now for the $N_s + 2$ interpolation points.

Finally, the derivative of the continuous flux function is given by

$$F_j'(\xi) = F_j(-1)\,\tilde{\ell}_0'(\xi) + \left[\sum_{k=1}^{N_s} \hat{f}_{j,k}(t)\,\tilde{\ell}_k'(\xi)\right] + F_j(1)\,\tilde{\ell}_{N_s+1}'(\xi), \tag{2.12}$$

with

$$\tilde{\ell}_n'(\xi) = \sum_{\substack{i=0 \\ i\neq n}}^{N_s+1} \left[\frac{1}{\xi_n - \xi_i} \prod_{\substack{m=0 \\ m\neq(i,n)}}^{N_s+1} \frac{\xi - \xi_m}{\xi_n - \xi_m}\right]. \tag{2.13}$$

After replacing variables and the spatial derivative in eq. (2.3) by their discrete counterparts, the following semi-discrete ordinary differential equation is obtained:

$$\frac{d}{dt}\hat{q}_{j,k} = -\frac{2}{\Delta_j} F_j'(\xi_k). \tag{2.14}$$

In practice, the derivative of the continuous flux function is calculated as a matrix-vector product using a polynomial derivative matrix (J. Romero et al., 2020). The equivalence of this scheme with the weak form of the NDG method may be verified easily by noting that: 1) $\tilde{\ell}_0'(\xi)$ and $\tilde{\ell}_k'(\xi)$ respectively take the same values as left and right Radau polynomials at the Gauß-Legendre points (Huynh,

2019), and 2) the part of the differentiation matrix applied to the solution points is identical to, say, Eq. (43) in (Kopriva & Gassner, 2010). Given this equivalence between DFR and NDG, one expects that the numerical properties of the latter, which are well studied in the literature (see e.g. (Hesthaven & Warburton, 2007; Giraldo, 2020)), also apply to the former.

2.3.2 Extension to two dimensions and curved geometry

The extension of the DFR method to quadrilateral and hexahedral elements, for which the basis polynomials are the tensor product of the one-dimensional basis functions, is straightforward. The basic idea is to first transform the governing equations from a physical element to the reference or standard element. Then, the one-dimensional DFR formulation is applied in each coordinate direction of the standard element without further complications. The discretization of the tensorial momentum equations is performed by treating each component separately.

Although the method is remarkably similar in flat and curved geometries, it is possible to simplify the formulation by avoiding the transformation of the derivatives given in eqs. (2.6) and (2.14). For instance, in eqs. (2.1) and (2.2) one may take advantage of the tensorial formulation by transforming the velocities u^i, metric h^{ij} and factor $\sqrt{g}$ as well as the components of the Christoffel symbols Γ^i_{0k} and Γ^i_{jk} from the cubed-sphere coordinates to the local coordinate system $[-1,1]\times[-1,1]$ on each element. The transformation rules are

$$\tilde{u}^j = \frac{2}{\Delta x^j} u^j, \tag{2.15}$$

$$\sqrt{\tilde{g}} = \frac{1}{4}\Delta x^1 \Delta x^2 \sqrt{g}, \tag{2.16}$$

$$\tilde{g}_{ij} = \frac{1}{4}\Delta x^i \Delta x^j g_{ij}, \tag{2.17}$$

$$\tilde{h}^{ij} = \frac{4}{\Delta x^i \Delta x^j} h^{ij}, \tag{2.18}$$

$$\tilde{\Gamma}^i_{jk} = \frac{1}{2}\Delta x^i \Gamma^i_{jk}, \tag{2.19}$$

$$\tilde{\Gamma}^i_{0k} = \Gamma^i_{0k}, \tag{2.20}$$

where Δx^1 and Δx^2 are the element dimensions in the cubed-sphere coordinate

system and the tilde symbol indicates the corresponding values in the coordinate system of the standard element. These transformations may be computed only once at the beginning of the simulation and then used consistently in all calculations. Note that these transformations do not affect the consistency relations at the interfaces of the cubed-sphere panels.

2.4 Time integration

The semi-discrete system obtained after applying the DFR to the shallow-water equations is an example of a general problem arising in the so-called method of lines, in which the space derivatives of a partial differential equation (PDE) are discretized first, leading to a large autonomous system of ordinary differential equations (ODEs) of the form

$$\frac{d}{dt}q(t) = \mathcal{F}(q(t)), \quad q(t_0) = q_0, \tag{2.21}$$

where $q(t)$ represents the unknown dynamical quantities and $\mathcal{F}$ is a function describing all forcing terms driving the system.

Numerical time integration of eq. (2.21) is a challenging task due to both high dimensionality of $\mathcal{F}(q)$ as well as the stiffness of its Jacobian $\partial\mathcal{F}/\partial q$. The latter property is due to the presence of a wide range of temporal frequencies characteristic of the evolution of this system. Traditionally, there have been two general approaches to temporal discretization of eq. (2.21): explicit and implicit methods (see e.g. (Mengaldo et al., 2019)). On the one hand, explicit methods are very efficient per time step since such algorithms require only evaluations of the right-hand side function $\mathcal{F}(q)$ using precomputed values of q and possible additions of such vectors. However, explicit integrators suffer from poor numerical stability properties, and for stiff systems the stability constraints impose a time step that is often too small for practical applications. Implicit methods, on the other hand, usually possess much better stability properties and allow time integration of the system using significantly larger time step sizes compared to explicit schemes. Such improvement though is accompanied by an increase in computational complexity per time step since implicit methods require solution of a large and stiff system of

nonlinear equations at each time step. Newton-Krylov algorithms (Knoll & Keyes, 2004) are the most commonly used in practice to solve such nonlinear systems with implicit time stepping. The nonlinearity and stiffness of many dynamical systems, such as the shallow-water equations, may result in slow convergence of both the Krylov method used to solve the underlying linear system within a Newton iteration and the Newton iteration itself. Preconditioning is used to alleviate this challenge, but developing an effective preconditioner for many problems can in itself be a difficult and time-consuming task. This is particularly true when higher-order spatial discretizations, such as the DFR method, are employed to approximate spatial operators. Given these considerations, there is a pressing need for new time integration techniques with better stability properties compared to explicit methods and improved computational complexity per time step compared to implicit schemes. In recent years, exponential integration emerged as a possible alternative to implicit methods in solving large stiff systems.

Let $\{t_n\}_{n=0}^{M}$ be a set of nodes that represent some discretization of the integration interval $[t_0, t_{\text{end}}]$. The numerical solution of eq. (2.21) at time t_n is denoted as $q_n \approx q(t_n)$ and the next task is to develop a time integration method to compute the vectors q_n. The starting point for constructing an exponential integrator is often a Taylor expansion of $\mathcal{F}(q)$ around a known value of the solution q_n that allows writing the equation in the following form

$$\frac{dq}{dt} = \mathcal{F}(q) = \mathcal{F}(q_n) + \frac{\partial \mathcal{F}}{\partial q}(q_n)(q - q_n) + \mathcal{R}(q), \tag{2.22}$$

where $\mathcal{R}(q) = \mathcal{F}(q) - \mathcal{F}(q_n) - \frac{\partial \mathcal{F}}{\partial q}(q_n)(q - q_n)$ is the nonlinear remainder after the first two terms of the Taylor expansion. Denoting the Jacobian matrix $\mathcal{J}_n = \frac{\partial \mathcal{F}}{\partial q}(q_n)$ for brevity and using an integrating factor $e^{t\mathcal{J}_n}$, one may write the integral form of eq. (2.22), and the solution at $t_n + \Delta t$ is given by the variation of constants formula

$$q(t_n + \Delta t) = q_n + \mathcal{J}_n^{-1}(e^{\Delta t \mathcal{J}_n} - I)\mathcal{F}(q_n) + \int_{t_n}^{t_n + \Delta t} e^{(t_n + \Delta t - t)\mathcal{J}_n} \mathcal{R}(q(t))dt \tag{2.23}$$

or, if one sets $\varphi_1(z) = (e^z - 1)/z$, as

$$q(t_n + \Delta t) = q_n + \varphi_1(\Delta t \mathcal{J}_n)\Delta t \mathcal{F}(q_n) + \int_{t_n}^{t_n + \Delta t} e^{(t_n + \Delta t - t)\mathcal{J}_n} \mathcal{R}(q(t))dt. \tag{2.24}$$

An exponential integrator to estimate $q(t_n + \Delta t)$ is then constructed by approximating the nonlinear integral in eq. (2.24) to the desired order of accuracy. If the integral is neglected, one obtains a second-order method known as an Exponential Euler scheme (hereafter denoted EPI2 as in (Tokman, 2006)):

$$q(t_n + \Delta t) = q_n + \varphi_1(\Delta t \mathcal{J}_n)\Delta t \mathcal{F}(q_n). \tag{2.25}$$

To increase the order of accuracy, the nonlinear integral is usually approximated using Runge-Kutta or multistep-type approaches. Either of such schemes will result in using some form of a polynomial approximation to the function $\mathcal{R}(q(t))$ in variable t to estimate the nonlinear integral in eq. (2.24). As a result, the approximate solution $q_{n+1} \approx q(t_n + \Delta t)$ will be expressed in terms of products of matrix functions and vectors like $\varphi_k(\Delta t \mathcal{J}_n)\mathcal{R}_k$, where $\mathcal{R}_k$ is a vector obtained by computing the nonlinear function $\mathcal{R}(q_k)$ using a known value of q_k. The functions φ_k satisfy the relation

$$\varphi_0(z) = e^z, \quad \varphi_k(z) = \int_0^1 e^{(1-s)z} \frac{s^{k-1}}{(k-1)!} ds, \quad k \geq 1. \tag{2.26}$$

All $\varphi_k(z)$ are analytic functions defined on the complex plane. This definition can then be extended to matrices using any of the available definitions of matrix functions (see e.g. (Higham, 2008)). For instance, they could be defined using power series similar to the matrix exponential:

$$\varphi_k(A) = \sum_{n=0}^{\infty} \frac{A^n}{(n+k)!}.$$

Since $\mathcal{J}_n$ is generally a large stiff matrix, evaluation of the products $\varphi_k(\Delta t \mathcal{J}_n)\mathcal{R}_k$ is the most computationally intensive part of approximating q_{n+1} at every time step. Thus, choosing the right algorithm to approximate these products is key to implementing an efficient exponential integrator.

While there are a number of algorithms that have been proposed for estimating products of matrix functions and vectors, only a few of those are suitable for general large stiff problems, and even fewer are appropriate for problems where the argument matrix is not known or cannot be stored explicitly. Many of such algorithms are only computationally feasible for small matrices (Moler & Van Loan,

1978, 2003). Other methods can be used for large matrices but require some information about the norm or the spectrum of the matrices (Al-Mohy & Higham, 2011; Caliari et al., 2016; Caliari, Kandolf, & Zivcovich, 2018). For many systems including the one considered here, it is, however, common to have the Jacobian matrix only available in the so-called matrix-free form—i.e. only a function that multiplies the Jacobian matrix with a vector is available rather than having an explicitly stored Jacobian. When an approximation to the Jacobian-vector product is used, the resulting method is then called an inexact Krylov method. An analysis of inexact Krylov subspace methods for approximating the matrix exponential has been presented in (Dinh & Sidje, 2017).

A popular matrix-free method (see (Knoll & Keyes, 2004) and references therein) is the finite-difference approximation

$$\mathcal{J}_n v = \frac{\mathcal{F}(q_n + \epsilon \cdot v) - \mathcal{F}(q_n)}{\epsilon} + \mathcal{O}(\epsilon), \tag{2.27}$$

where $\epsilon \in \mathbb{R}$ is a small parameter and v is an arbitrary vector. It is well known that if ϵ is too small, then the rounding errors incurred in the evaluation of $\mathcal{F}$ begin to dominate. For this reason, a methodology based on the complex-step approximation is rather used (Squire & Trapp, 1998):

$$\mathcal{J}_n v = \Im\left[\frac{\mathcal{F}(q_n + \epsilon\ i \cdot v)}{\epsilon}\right] + \mathcal{O}(\epsilon^2), \tag{2.28}$$

where $i^2 = -1$ and $\Im$ denotes the imaginary part operator.

Hence, the result of the Jacobian-vector product is calculated using only one evaluation of the right-hand side function with complex arguments. The complex-step approximation has gained popularity in the field of machine learning in recent years (Goodfellow, Bengio, & Courville, 2016) but has seldom been used in the context of exponential integrators.

For large stiff matrix-free cases, Krylov-projection-type algorithms are the best approach to estimating products $\varphi_k(\Delta t \mathcal{J}_n)\mathcal{R}_k$. In particular, adaptive Krylov-based methods (Niesen & Wright, 2012), including the recently proposed KIOPS algorithm (Gaudreault, Rainwater, & Tokman, 2018), have been shown to deliver the most efficiency for such problems. Given a matrix A and a set of vectors b_0,

..., b_p these methods allow fast computation of the linear combinations of the type

$$\varphi_0(A)b_0 + \varphi_1(A)b_1 + \varphi_2(A)b_2 + ... + \varphi_p(A)b_p. \tag{2.29}$$

The adaptive Krylov-based methods compute the expression (2.29) using only one Krylov projection and an adaptive substepping mechanism which speeds up the computation by appropriately scaling the matrix A. The precise formulation of the KIOPS algorithm (its structure and implementation) is given in (Gaudreault, Rainwater, & Tokman, 2018).

As stated earlier, obtaining the matrix functions-vector products $\varphi_k(\Delta t\mathcal{J}_n)\mathcal{R}_k$ and their linear combinations represents the main computational load of an exponential integrator. One should therefore minimize the number of such operations and optimize their computation. Exponential Propagation Iterative (EPI) methods of Runge-Kutta (Tokman, 2011) and multistep (Tokman, 2006) types were designed to achieve this. Simulations with weather and climate models usually employ constant time steps because they are controlled not only by the resolved dynamics but also by constraints on sub-grid scale parameterizations. Therefore, only methods with constant time steps are considered here.

A third-order multistep-type EPI method is derived in (Tokman, 2006). The EPI3 algorithm to solve eq. (2.22) may be written as

$$q_{n+1} = q_n + \varphi_1(\Delta t\mathcal{J}_n)\Delta t\mathcal{F}_n + \tfrac{2}{3}\varphi_2(\Delta t\mathcal{J}_n)\Delta t\mathcal{R}_{n-1}, \tag{2.30}$$

where $\mathcal{F}_n = \mathcal{F}(q_n)$ and

$$\varphi_1(z) = \frac{e^z - 1}{z}, \tag{2.31}$$

$$\varphi_2(z) = \frac{e^z - 1 - z}{z^2}, \tag{2.32}$$

$$\mathcal{R}_{n-1} = \mathcal{F}(q_{n-1}) - \mathcal{F}_n - \mathcal{J}_n(q_{n-1} - q_n). \tag{2.33}$$

In the following, higher-order multistep-type EPI algorithms are presented. The following general ansatz will become useful:

$$q_{n+1} = q_n + \varphi_1(\Delta t\mathcal{J}_n)\Delta t\mathcal{F}(q_n) + \sum_{m=1}^{M} \varphi_m(\Delta t\mathcal{J}_n)v_m, \tag{2.34}$$

$$v_m = \sum_{i=1}^{P} \alpha_{m,i}\Delta t\mathcal{R}(q_{n-i}), \tag{2.35}$$

where P is the number of previous points used and M is the maximum order of the φ function. Notice that if methods constructed using this ansatz are combined with adaptive Krylov-type algorithms (such as KIOPS) for calculating the linear combinations of terms like $\varphi_m(\Delta t \mathcal{J}_n)v_m$, then each time step will require only one call to KIOPS (or to any other adaptive-Krylov method). Since approximating the linear combinations involving φ matrix functions-vector products represents the main computational cost of an EPI method, all such schemes will have a comparable cost. Therefore, significant increases in accuracy with only relatively small increases in computational cost are expected from such higher-order methods. This computational advantage of the newly introduced EPI schemes will hold for problems where the cost of the Krylov-projection evaluation of the exponential matrix functions and vectors dominates the computational cost per time step compared to the evaluation of the right-hand-side function or the actual product of the Jacobian and a vector. Should the latter computations become prohibitively expensive and dominant, one might still want to use a lower-order method.

To construct specific EPI schemes, the coefficients $\alpha_{m,i}$ in eq. (2.35) must be determined. This is done with the Butcher trees order condition theory, which greatly simplifies their derivation. Details on Butcher trees to construct exponential methods are found in (J. Butcher, 2010; Tokman, 2011). This machinery is used to derive systems of equations for the coefficients $\alpha_{m,i}$. In the case of multistep-type EPI methods, such systems are linear but typically under-determined. Therefore, as for standard Runge-Kutta and multistep methods, additional constraints may be added and the extended linear systems may be solved to obtain either a single method or a family of schemes. Such constraints may serve to introduce certain desired properties to the resulting scheme, for instance: 1) the largest possible number of zero coefficients, 2) coefficients with the smallest possible magnitude in the scheme itself or in the next order error term. The coefficients of a method may be represented as a matrix A, where $A_{i,j} = \alpha_{i,j}$. Therefore, a method using P previously computed nodes q_{n-i} $(i = 1, ..., P)$ and M functions φ_j $(j = 1, ..., M)$ will have P columns and M rows.

Here, it is chosen to supplement the order conditions by constraints that mini-

mize the magnitude of the coefficients in the first term of the error, In other words, if the method is of order p, then the coefficients of the error term of order $p+1$ will be minimized. Solving the order conditions with these constraints, methods of orders 4, 5 and 6 are obtained with the following coefficients:

$$A_4 = \begin{pmatrix} 0 & 0 \\ -\frac{3}{10} & \frac{3}{40} \\ \frac{32}{5} & -\frac{11}{10} \end{pmatrix}, \tag{2.36}$$

$$A_5 = \begin{pmatrix} 0 & 0 & 0 \\ -\frac{4}{5} & \frac{2}{5} & -\frac{4}{45} \\ 12 & -\frac{9}{2} & \frac{8}{9} \\ 3 & 0 & -\frac{1}{3} \end{pmatrix}, \tag{2.37}$$

$$A_6 = \begin{pmatrix} 0 & 0 & 0 & 0 \\ -\frac{49}{60} & \frac{351}{560} & -\frac{359}{1260} & \frac{367}{6720} \\ \frac{92}{7} & -\frac{99}{14} & \frac{176}{63} & -\frac{1}{2} \\ \frac{485}{21} & -\frac{151}{14} & \frac{23}{9} & -\frac{31}{168} \end{pmatrix}. \tag{2.38}$$

Note that the EPI multistep methods require computation of the starting nodes q_n to begin the time stepping process. These values may be calculated in many different ways. Here, the EPI2 method (see eq. (2.25)) with a smaller time step to compute the necessary starting values is used. Once the initial values are obtained and the remainder function $R(q_n)$ is evaluated and stored, the time stepping algorithm proceeds as follows. At each new time iteration, the solution at the next time step q_{n+1} is computed using eq. (2.34) as well as the following steps:

1. compute vectors $v_m = \sum_{i=1}^{P} \alpha_{m,i} \mathcal{R}(q_{n-i})$ using previously stored values q_{n-i} and $\mathcal{F}(q_{n-i})$. Note that while vectors $\mathcal{F}(q_{n-i})$ may be reused from previous iterations, vectors $\mathcal{R}(q_{n-i})$ cannot be reused because the Jacobian matrix

changes from one iteration to the next;

2. use the KIOPS algorithm to evaluate

$$w_n = \varphi_1(\Delta t \mathcal{J}_n)\mathcal{F}(q_n) + \sum_{m=1}^{M} \varphi_m(\Delta t \mathcal{J}_n) v_m;$$

3. advance the solution over the next time step $q_{n+1} = q_n + w_n \Delta t$.

Exponential integrators, much like implicit methods, possess very good stability properties since they represent an exact solution $q_{n+1} = e^{\Delta t \mathcal{J}} q_n$ to the linear problem $y' = \mathcal{J}y$ (i.e., $\mathcal{R}(q) = 0$ and $\mathcal{J}_n = \mathcal{J}$ in eq. (2.22)) with $\|e^{\Delta t \mathcal{J}_n}\| < 1$ if the real parts of the eigenvalues of $\mathcal{J}_n$ are all negative. In addition, exponential integration can lead to computational savings per time step compared to implicit schemes. This is the result of the Krylov methods performing more efficiently in evaluating exponential-like functions φ in expression (2.29) compared to computing the rational functions of a Jacobian, which have to be approximated within an implicit method. Exponential methods are also generally more accurate than explicit and implicit methods of the same order. In the context of geophysical fluid dynamics, this means that relevant wave dispersion relations are expected to be simulated more accurately.

Note that all exponential integrators are trivially A-stable since an exponential method solves the linear part of the system exactly ($q_n + \varphi_1(\Delta t \mathcal{J}_n)\Delta t \mathcal{F}(q_n)$) thus no unstable computational modes are generated. Surely when the exponential is approximated rather than computed exactly the method is not necessarily A-stable. However, when adaptive Krylov-based algorithm is used to compute the products of exponential matrix functions and vectors these computations can be performed in a way that does not pose challenges for stability. It has been shown in the literature (Tranquilli & Sandu, 2014a, 2014b) that combining, exponential Runge-Kutta integration with Krylov algorithm is similar to employing an explicit Runge-Kutta method with coefficients that depend on the eigenvalues (particularly those from the boundary of the spectral domain) of the Jacobian and therefore the region of stability of such method also depends on these eigenvalues. Obviously, such explicit Runge-Kutta method will be different for each time step, as the

Jacobian and the Krylov vectors are changing. Thus the stability region is not static and is accounting for the eigenvalues of the Jacobian. In addition, the adaptive Krylov allows to set tolerance on the approximation of the exponential matrix functions-vector products and it is easy to ensure that these computations are done with enough precision to not impact the overall time stepping process. It is also worth noting that a similar issue arises when implicit-Krylov methods are used and solving with an approximate Jacobian and within a certain tolerance is a common practice with implicit methods. The implications of this strategy in the context of Krylov subspace methods for approximating the matrix exponential have been studied recently in (Dinh & Sidje, 2017). Practice shows that employing exponential-Krylov time stepping approach is effective and allows integration of many practical problems without issues with stability (Loffeld & Tokman, 2013; Einkemmer, Tokman, & Loffeld, 2017).

In the next section, numerical examples will be used to demonstrate that the newly constructed EPI methods allow significant increases in accuracy of the time integration without incurring significant computational cost.

2.5 Numerical experiments

The numerical properties of the algorithms described in previous sections are studied using test cases found in the literature. In section 2.5.1, the accuracy and convergence of the EPI time integrators on three PDE problems are evaluated. In section 2.5.2, numerical properties of the spatial discretization are evaluated in isolation using benchmarks found in (Läuter, Handorf, & Dethloff, 2005). In section 2.5.3, the accuracy of the various time integrators and DFR accuracy orders is studied from simulating the barotropic test case found in (Galewsky, Scott, & Polvani, 2004). Finally, results from three test cases based on the benchmarks found in (Williamson et al., 1992) are presented. These experiments allow, among other things, to assess the conservation of various quantities.

Unless otherwise indicated, all tests with the shallow-water equations use a time step of 1 hour and the tolerance of the KIOPS solver is set to 10^{-10}. The ϵ

parameter of the complex-step approximation is set to $\approx 1.5 \times 10^{-8}$. The computational grid is made up of $6 \times N_e \times N_e$ elements. Each element consists of $N_s \times N_s$ solution points. The value of N_s and N_e are varied such that the total number of degrees of freedom is kept constant. For the considered shallow-water test cases, a global grid with 86400 degrees of freedom and a timestep of 1 hour correspond to Courant numbers ranging from 10 to 20. The grid rotation parameters are set to $\lambda_0 = 0$, $\phi_0 = \pi/4$ and $\alpha_0 = 0$ (see Appendix B.2) since this configuration poses a greater challenge for most test cases than, say, the unrotated configuration.

Beside the intrinsic numerical diffusion of the Riemann solver, no filter, artificial diffusion or limiter is used in this study. These aspects have been extensively studied in the literature (see e.g. (Jablonowski & Williamson, 2011; Marras et al., 2016)) and may be assessed in future works.

Errors associated with simulated cases for which known analytical solutions exist are evaluated from the height field using the following global norms:

$$L_1(t) = \frac{I[|H - H_T|\,; t]}{I[H_T; t]}, \tag{2.39}$$

$$L_2(t) = \sqrt{\frac{I[(H - H_T)^2; t]}{I[H_T^2; t]}}, \tag{2.40}$$

$$L_\infty(t) = \frac{\max|H(x^1, x^2, t) - H_T(x^1, x^2, t)|}{\max|H_T(x^1, x^2, t)|}, \tag{2.41}$$

where H_T is the height field of the analytical solution and I is an approximation to a global integral calculated with the Gauß-Legendre quadrature rules as follows:

$$I\left[\Psi; t\right] \approx \sum_{p=0}^{5} \sum_{k=1}^{N_e} \sum_{l=1}^{N_e} \sum_{m=1}^{N_s} \sum_{n=1}^{N_s} \Psi\left((x^1)_{m,k,l,p}, (x^2)_{n,k,l,p}, t\right) \times \sqrt{g}\left((x^1)_{m,k,l,p}, (x^2)_{n,k,l,p}, t\right) w_m w_n, \tag{2.42}$$

where p is the panel index, the indices k, l, m, n refer to the position of the points on the Gauß-Legendre grid and the w's are quadrature weights that lead to an exact integration of polynomials of degree $2N_s - 1$ or less.

2.5.1 Convergence of the time integrators

The performance of the new EPI methods introduced above is evaluated with a standard set of tests for large stiff systems of equations (Hochbruck & Ostermann, 2005). Their convergence is assessed from three common benchmarks:

- Advection-diffusion-reaction PDE:

$$\frac{\partial u}{\partial t} + \alpha \left(\frac{\partial u}{\partial x} + \frac{\partial u}{\partial y} \right) = \varepsilon \left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} \right) + R(u),$$

  where $R(u) = \gamma u(u - 1/2)(1 - u)$, $\varepsilon = 1/100, \alpha = -10, \gamma = 100$, on the domain $x, y \in [0, 1], t \in [0, 0.1]$ and with homogeneous Neumann boundary conditions. The initial condition is $u_0 = 256(xy(1 - x)(1 - y))^2 + 0.3$. The equation is discretized using a second-order finite difference method with $n = 1600$ grid points.

- Burger's equation:

$$\frac{\partial u}{\partial t} + \frac{1}{2} \frac{\partial u^2}{\partial x} = \varepsilon \frac{\partial^2 u}{\partial x^2},$$

  with $\varepsilon = 10^{-3}$. This equation is discretized using a second-order finite difference method with $n = 1024$ grid points on the domain $x \in [0, 1], t \in [0, 1]$. The initial condition $u_0 = \exp\left(-(x - \mu)^2/(2\sigma^2)\right)$ with $\mu = 0.3, \sigma = 0.05$ and homogeneous Dirichlet boundary conditions are used.

- Semilinear parabolic PDE (Hochbruck & Ostermann, 2005):

$$\frac{\partial u}{\partial t} - \frac{\partial^2 u}{\partial x^2} = \int_0^1 u(x, s) ds + \phi(x, t),$$

  where $\phi(x, t)$ is chosen such that the exact solution is $u(x, t) = x(1 - x)e^t$. This equation is discretized in space using a second-order finite difference method for the derivative and the trapezoidal rule for the integral with $n = 400$ grid points on the domain $x \in [0, 1], t \in [0, 1]$. The initial condition $u_0 = x(1 - x)$ and homogeneous Dirichlet boundary conditions are used.

The tolerance for KIOPS is set to 10^{-14} and the error is defined as the discrete 2-norm of the difference between the approximation to the solution and the reference solution. The exact Jacobian is used in all of the calculations. The reference

solutions for the first two cases are computed using MATLAB's `ode15s` integrator with absolute and relative tolerances set to 10^{-14}, whereas an exact solution exists for the third case.

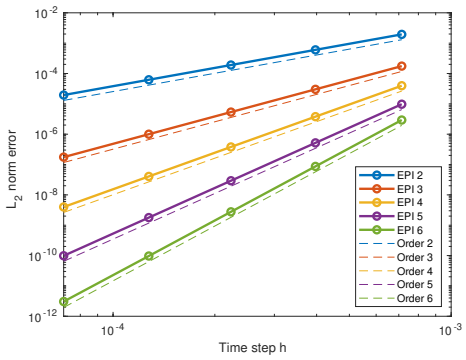
Figure 2.1 shows convergence plots obtained from the three test cases above. All methods converge to the reference solutions with the theoretically expected order of accuracy. Note that the semilinear parabolic PDE problem was constructed in (Hochbruck & Ostermann, 2005) to demonstrate that certain time integration methods can suffer from the order reduction if they were not derived using the stiff order conditions. It has not been proven, however, that the stiff order conditions in (Hochbruck & Ostermann, 2005, 2011) are necessary to avoid order reduction. While EPI schemes proposed here are not derived using these stiff order conditions from (Hochbruck & Ostermann, 2005, 2011), no order reduction is observed for these schemes in the test problems.

Figure 2.2 shows work-precision diagrams (CPU time vs. error) for all tested exponential scheme and benchmark cases. The points are in decreasing order of time step size, from left to right, with the following value of h: Advection-diffusion-reaction test case: $h = \{7.0 \times 10^{-4}, 2.2 \times 10^{-4}, 9.8 \times 10^{-5}\}$, Burger's equation test case: $h = \{3.1 \times 10^{-3}, 1.8 \times 10^{-3}, 1.0 \times 10^{-3}, 6.0 \times 10^{-4}, 3.0 \times 10^{-4}\}$ and semilinear parabolic test case: $h = \{1.1\times10^{-1}, 7.0\times10^{-2}, 5.0\times10^{-2}, 3.3\times10^{-2}, 2.1\times10^{-2}, 1.5\times 10^{-2}, 1.0 \times 10^{-3}\}$.

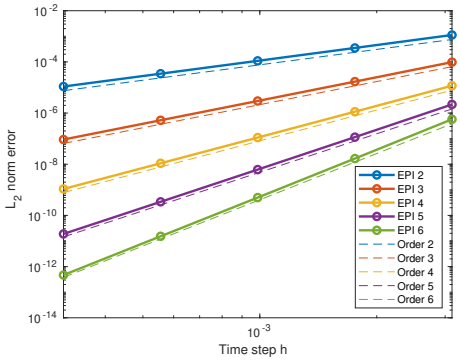
The work-precision diagrams clearly show that increasing the order of the EPI scheme does not lead to a significant increase in the computational time required to approximate the solutions with specified accuracy. This is because all EPI methods are implemented using a single computation of a linear combination of φ functions per time step and the computational cost of these linear combinations is comparable for all of the time integrators.

2.5.2 Convergence of the DFR method

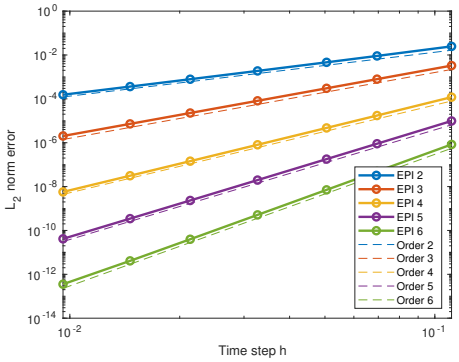
The diffusion-free, zonally balanced, time-dependent flow proposed in (Läuter, Handorf, & Dethloff, 2005) is a difficult test for the cubed-sphere when the grid is rotated at a 45° angle. The highest values of the height field cross eight panel edges,



(a) Advection-diffusion-reaction

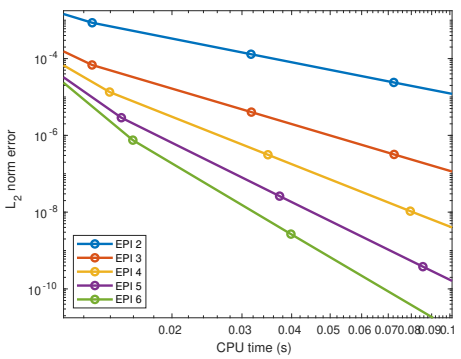


(b) Burger's equation

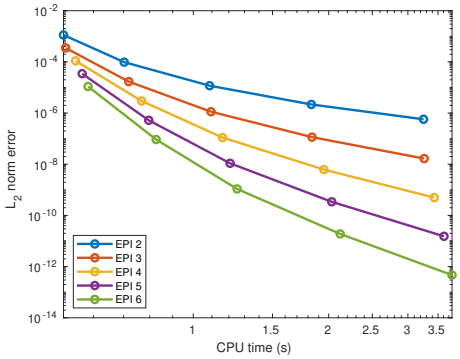


(c) Semilinear parabolic

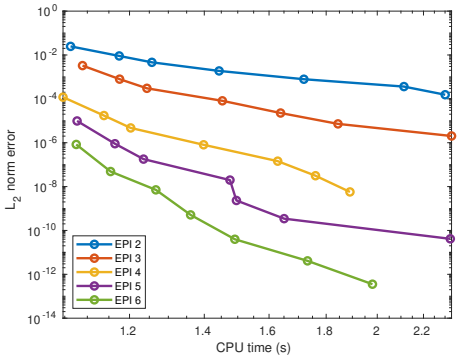
Figure 2.1: Convergence plots for multistep-type EPI methods of orders 2, 3, 4, 5 and 6.



(a) Advection-diffusion-reaction

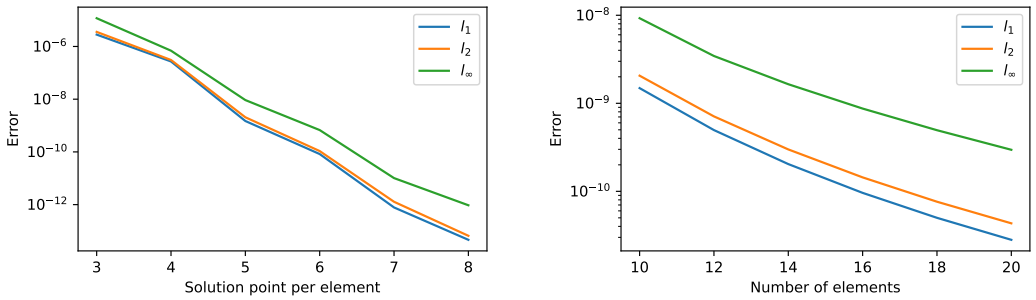


(b) Burger's equation



(c) Semilinear parabolic

Figure 2.2: Work-precision diagrams for multistep-type EPI methods of orders 2, 3, 4, 5 and 6.



(a) p-convergence ($N_e = 10$) (b) h-convergence ($N_s = 5$)

Figure 2.3: Error convergence at day 5 for the diffusion-free, zonally balanced, time-dependent flow.

four corners of the cube and follow two panel edges located along the equator. This test case has an analytical solution and is therefore frequently used to assess the convergence of numerical models. All simulations presented in this subsection are integrated in time with the EPI6 scheme. Error norms are computed at day 5.

To illustrate the numerical convergence of the DFR method applied to the shallow-water equations, the experiments are organized in two ways. First, a p-convergence test is performed. The results are shown in Figure 2.3a with $N_e = 10$ and the number of solution points is increased from $N_s = 3$ to $N_s = 8$. Then, an h-convergence test is conducted by varying N_e from 10 to 20, while keeping a fixed number of solution points $N_s = 5$. The results are shown in Figure 2.3b. Table 2.1 shows the order of accuracy calculated with respect to the errors with $N_e = 10$ and $N_e = 20$. It is observed that the DFR scheme reaches the expected formal order of accuracy, except at order 4 where there is a small order reduction.

2.5.3 Barotropic instability

A further description of convergence properties is obtained from a barotropic instability case using various numbers of elements and solution points. For this test proposed in (Galewsky, Scott, & Polvani, 2004), the initial condition is a zonal flow imitating a tropospheric jet at mid-latitudes in the northern hemisphere. A small perturbation of the height field is introduced to induce the development of

N_s	L_1	L_2	L_∞
3	3.501	3.504	3.007
4	3.724	3.705	3.668
5	5.724	5.571	4.964
6	6.030	6.061	6.222

Table 2.1: Computed order of accuracy for the diffusion-free, zonally balanced, time-dependent flow.

barotropic instability. This test case describes the evolution of a barotropic wave, where a continuous transfer of energy occurs at different spatial scales at mid-latitudes. This test is challenging for non-monotonic numerical schemes, such as those presented in this study.

The relative vorticity

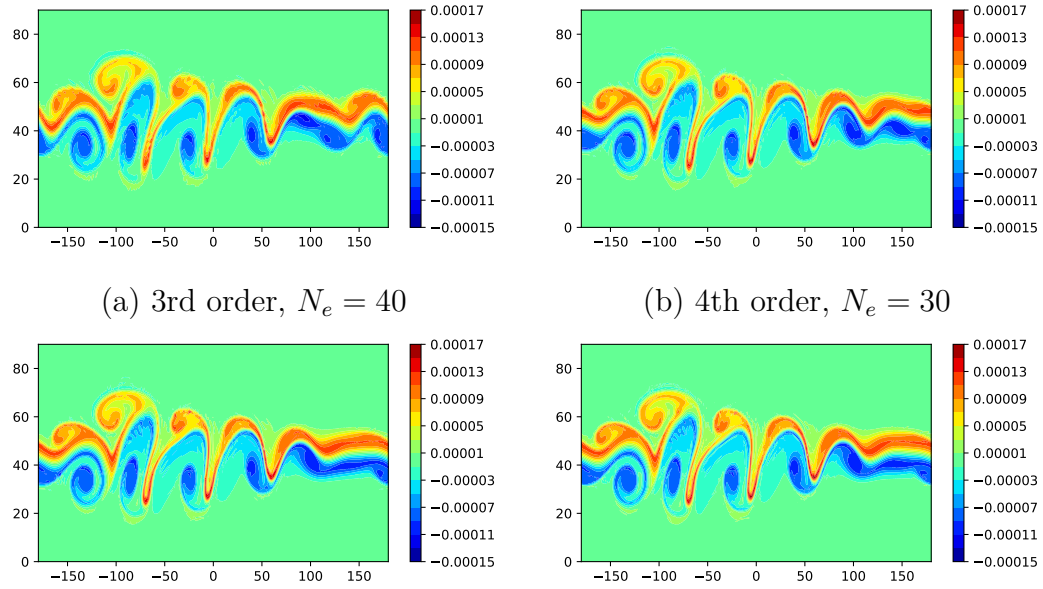
$$\zeta = \frac{\varepsilon^{0ij}}{\sqrt{g}} \frac{\partial}{\partial x^i} \left(g_{jk} u^k \right), \tag{2.43}$$

where $\varepsilon^{\alpha\mu\nu}$ is the Levi-Civita symbol, may be compared with the reference solution presented in (Galewsky, Scott, & Polvani, 2004). Figure 2.4 shows the relative vorticity for different values of N_s. The time integrator is chosen to match the order of convergence in space. The number of elements N_e is changed accordingly in order to keep a constant number of degrees of freedom. Although the grid spacing is rather coarse, a convergence to the reference solution is observed as the order increases. The shapes of the solutions at 5th and 6th orders are comparable to the reference solution.

2.5.4 Numerical conservation properties

Three standard tests suggested in (Williamson et al., 1992) are considered to assess the conservation properties of the numerical schemes presented in the previous sections. To monitor the residual temporal evolution of global invariants, one defines a normalized conservation error as follows:

$$\hat{\Psi}(t) = \frac{I\left[\Psi; t\right] - I\left[\Psi; 0\right]}{I\left[\Psi; 0\right]}. \tag{2.44}$$



(a) 3rd order, $N_e = 40$

(b) 4th order, $N_e = 30$

(c) 5th order, $N_e = 24$

(d) 6th order, $N_e = 20$

Figure 2.4: Relative vorticity field associated with the barotropic instability test at day 6 on a global grid with 86400 degrees of freedom. Only the Northern Hemisphere is shown.

The function Ψ is replaced by H for mass conservation, by

$$\frac{1}{2}\left[Hg_{ij}u^iu^j + g_r\left((H+h_B)^2 - h_B^2\right)\right]$$

for total energy conservation, and by

$$\frac{(\zeta+f)^2}{2H}$$

for potential enstrophy conservation, where

$$f = \frac{\varepsilon^{0ij}}{\sqrt{g}} g_{jk}\Gamma^k_{i0} \tag{2.45}$$

$$= \frac{2\Omega}{\delta}(\sin\phi_p - X\cos\phi_p\sin\alpha_p + Y\cos\phi_p\cos\alpha_p) \tag{2.46}$$

is the Coriolis parameter.

In the following numerical experiments, four different configurations are considered. The value of N_s varies from 3 to 6. The time integrator is chosen to match the order of convergence in space. The number of elements N_e is changed in order to keep a constant number of degrees of freedom. The parameter α introduced in (Williamson et al., 1992) is set to zero because the grid rotation mechanism in the present paper is embedded in the equations of motion.

The following approximation is proposed to compare the different mean resolutions of the cubed-sphere grids with Gauß-Legendre points to mean resolutions of other types of grids:

$$\Delta = \frac{90^\circ}{N_e N_s}. \tag{2.47}$$

For the configurations considered in this subsection, $\Delta \approx 1^\circ$. This resolution is comparable to the resolution of the models presented in (Qaddouri et al., 2012).

Steady-state geostrophically balanced flow

This test case is a steady-state solution to the shallow-water equations. The winds are a solid-body rotation, and the height is defined such that an exact geostrophic balance exists. The solution is expected to maintain this steady-state balance for at least 5 days.

Figure 2.5 depicts the normalized error of the height field for the steady-state geostrophically balanced flow after 5 days. As the order increases, the error rapidly

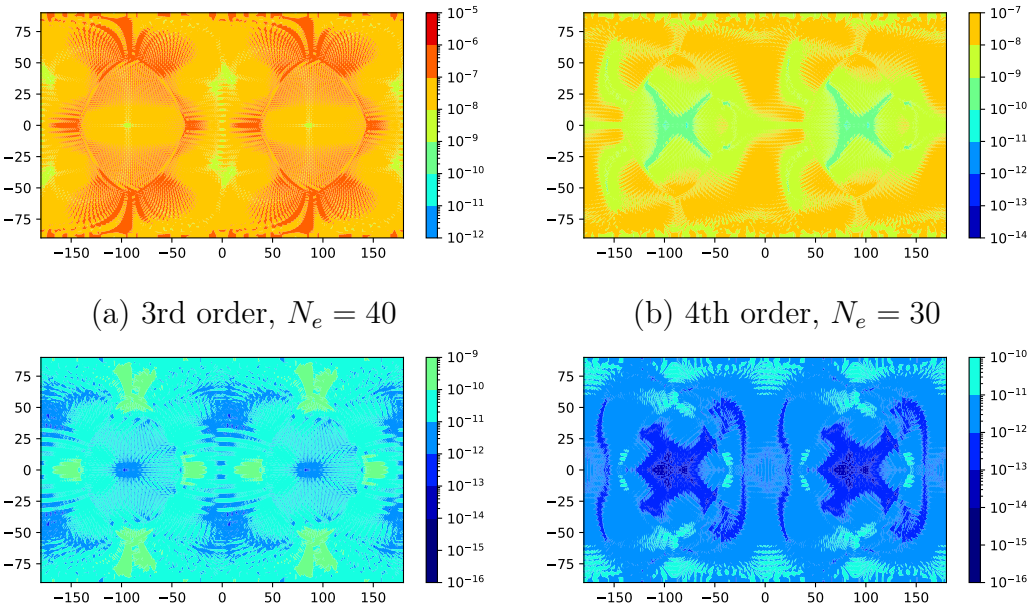
decreases. All configurations maintain a balance between advection, fluid height gradient, and Coriolis terms. Contrary to results found in other studies with the Yin-Yang overset grid (Qaddouri et al., 2012; Li et al., 2008) in which relatively large numerical errors occur where the zonal flow crosses the boundary between two panels, cubed-sphere panel boundaries and orientations do not impede convergence for higher-order configurations. The error associated with orders higher than 4 is smaller than the errors of the four models in (Qaddouri et al., 2012), including the reference spectral model.

Figure 2.6 shows the time trace of the normalized error for the height field over a period of 30 days. The evolution of the normalized conservation error of the mass, total energy and potential enstrophy is presented in Figure 2.7. Adjustments are observed during the first few time steps, likely due to imperfections in the initial conditions, but the conservation errors remain reasonably small for the rest of the simulations. Further experiments will be required to fully understand the convergence during the first few time steps. It is worth mentioning that nothing is done here to ensure that the discrete initial conditions are in geostrophic balance. An example of a procedure to numerically enforce the geostrophic equilibrium is proposed in (Weller, Thuburn, & Cotter, 2012). However, the numerical stability does not seem to be impacted by these initial imperfections since the error stabilizes quickly and remains low for the rest of the simulations.

Zonal flow over an isolated mountain

In this test case, a flow is perturbed by a topographically induced source term. The mountain shape is cone-like and not differentiable. In a model based on DFR, one must therefore approximate the mountain shape with piecewise polynomials, which may introduce spurious oscillations. In addition, the wind and height fields would be initially balanced only in the absence of a mountain. This results in a flow evolution that is difficult to predict numerically.

Orographic Rossby waves are induced from the beginning of the simulation. These waves then spread over almost the entire sphere, including the southern hemisphere. The height fields after 15 days are shown in Figure 2.8. These results



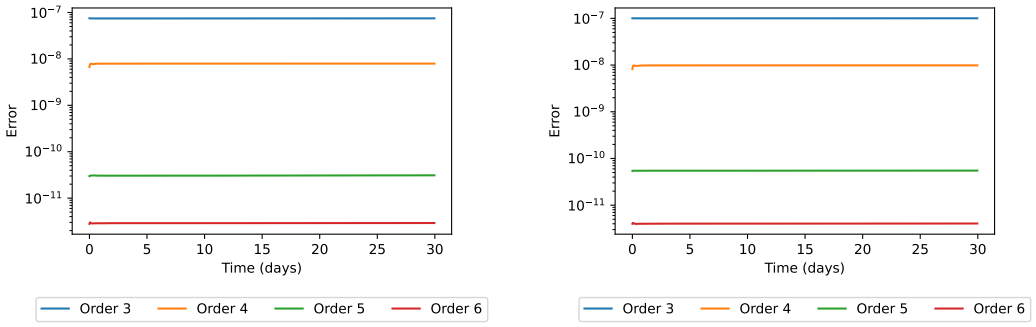
(a) 3rd order, $N_e = 40$

(b) 4th order, $N_e = 30$

(c) 5th order, $N_e = 24$

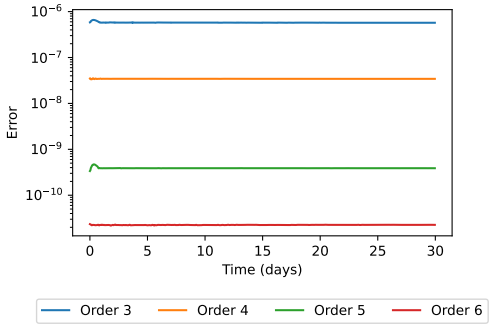
(d) 6th order, $N_e = 20$

Figure 2.5: Normalized error of the height field for the steady-state geostrophically balanced flow after 5 days using orders 3, 4, 5 and 6. A global grid with 86400 degrees of freedom is used.



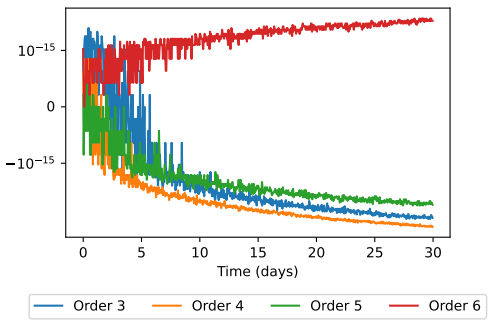
(a) L_1 error

(b) L_2 error

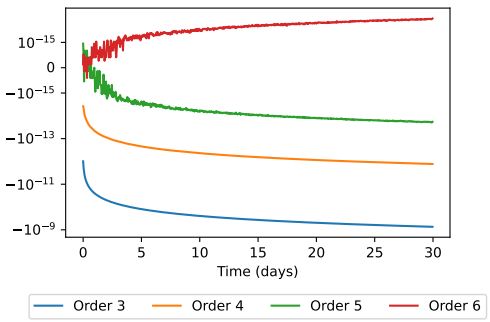


(c) L_∞ error

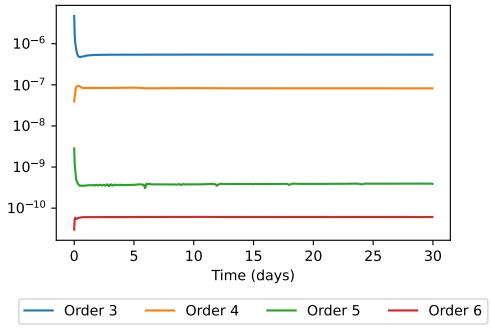
Figure 2.6: Normalized error of the height field as a function of time for the steady-state geostrophically balanced flow.



(a) Mass



(b) Energy



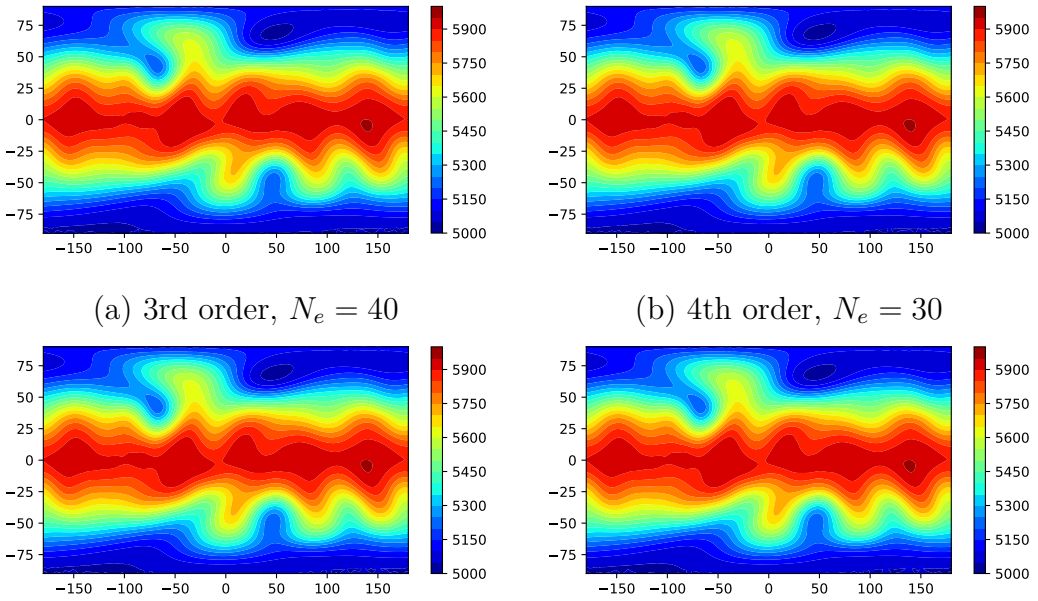
(c) Enstrophy

Figure 2.7: Time traces of the normalized errors of conserved quantities for the steady-state geostrophically balanced flow.

are similar to those reported by other authors (for instance (Ullrich, Jablonowski, & Van Leer, 2010; Qaddouri et al., 2012)). There are no apparent non-physical features such as oscillations around the mountain, although there are visible differences in the shape of the solution of the different configurations. In order to better distinguish the differences between the simulations, the relative vorticity ζ is shown in Figure 2.9. These results are comparable to those presented in (Bao, Nair, & Tufo, 2014). Small oscillations are visible in the relative vorticity field, especially around the mountain. This could be caused by the approximation of the mountain slope by piecewise polynomials or could be a sign that aliasing errors are present in higher-order simulations. The use of filters and other strategies against aliasing-driven errors will be studied in future works. Chapter 5 of (Hesthaven & Warburton, 2007) provides useful informations on this topic.

The normalized errors of mass, total energy and potential enstrophy conservation over 30 days are shown in Figure 2.10. The conservation of mass is accurate to machine precision and is comparable to results presented in (Nair, Thomas, & Loft, 2005a). The errors on the conservation of total energy and potential enstrophy are larger because the discretization of eqs. (2.1) and (2.2) does not explicitly enforce their conservation.

It should also be mentioned that some of the errors may come from the fact that this study focuses on dynamical aspects relevant to atmospheric models. For instance, the formulation and space-time discretization presented here do not maintain a perfect equilibrium for some special cases such as lakes at rest (see for example (Noelle et al., 2006)). Also, recall that nothing is done here to ensure that the discrete initial condition are in geostrophic balance. Thus, the restoring forces seek to compensate for these imbalances and other errors due to spatial discretization, leading to oscillations in the conservation error curves. This is particularly visible for energy and potential enstrophy, which are not conserved fields in the discrete system.



(a) 3rd order, $N_e = 40$

(b) 4th order, $N_e = 30$

(c) 5th order, $N_e = 24$

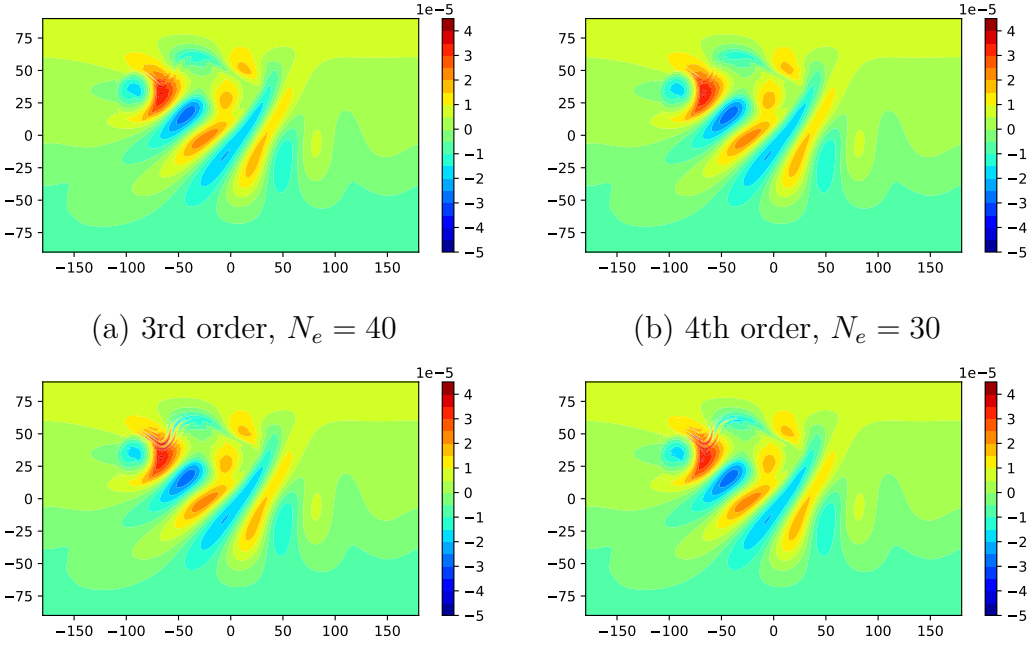
(d) 6th order, $N_e = 20$

Figure 2.8: Height field of the zonal flow over an isolated mountain at day 15 using orders 3, 4, 5 and 6. A global grid with 86400 degrees of freedom is used.

Rossby-Haurwitz wave

In this test case, the Rossby-Haurwitz wave number 4 is considered. This wave is an analytical solution to the non-linear barotropic vorticity equation. It is well known that this test case is susceptible to instabilities due to truncation in the initial conditions (Thuburn & Li, 2000) and that the numerical solution will eventually lose its structure. However, the solution is expected to remain stable over the 14 days required by (Williamson et al., 1992). The wave number 4 is expected to propagate steadily and retain its structure with only slight wavering in its shape.

The simulated height fields after 14 days are shown in Figure 2.11. The numerical solutions reproduce the reference solution reported in the literature (Qaddouri et al., 2012) with good accuracy. In particular, the Rossby-Haurwitz wave remains stable throughout the duration of the simulation. The history plots of the conservation errors for mass, total energy and potential enstrophy are shown in Figure 2.12. Mass is well conserved as in previous test cases. Conservation errors are



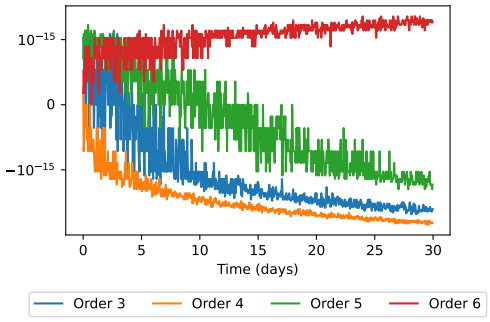
(a) 3rd order, $N_e = 40$

(b) 4th order, $N_e = 30$

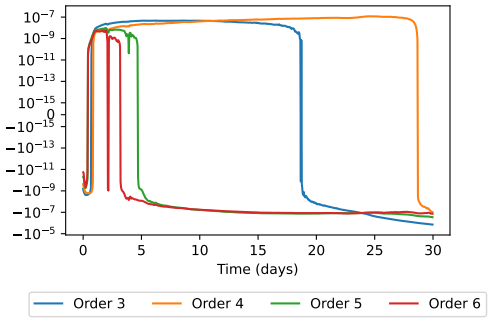
(c) 5th order, $N_e = 24$

(d) 6th order, $N_e = 20$

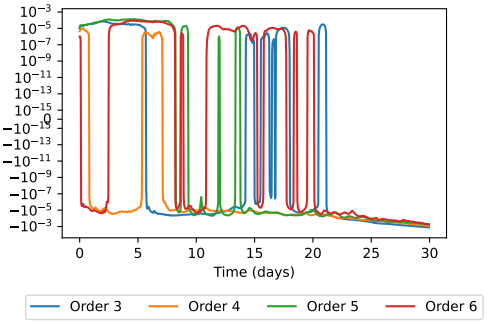
Figure 2.9: Relative vorticity field of the zonal flow over an isolated mountain at day 7 using orders 3, 4, 5 and 6. A global grid with 86400 degrees of freedom is used.



(a) Mass

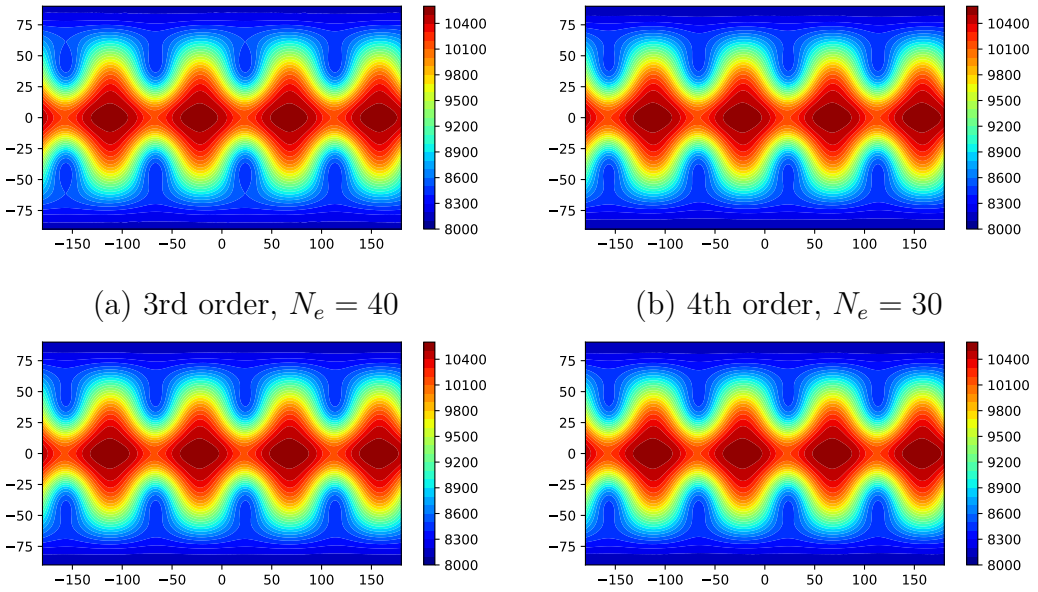


(b) Energy



(c) Potential enstrophy

Figure 2.10: Time traces of the normalized errors of conserved quantities for the zonal flow over an isolated mountain.



(a) 3rd order, $N_e = 40$

(b) 4th order, $N_e = 30$

(c) 5th order, $N_e = 24$

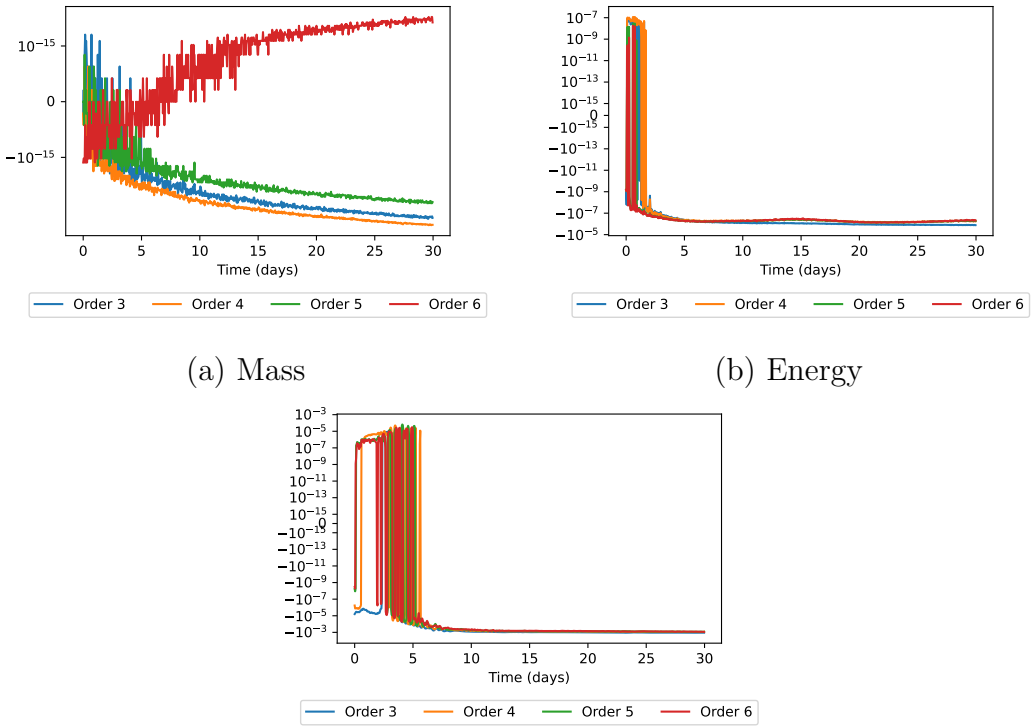
(d) 6th order, $N_e = 20$

Figure 2.11: Height field of Rossby-Haurwitz wave at day 14 using orders 3, 4, 5 and 6. A global grid with 86400 degrees of freedom is used.

much more important during the first 5 days of the simulation. Afterwards, these errors stabilize around a small value.

2.5.5 Large time step

As mentioned above, an important advantage of the exponential time integrators is that they allow large time steps regardless of the CFL condition. So far in this work, a time step of 1 hour has been used, which is larger than those typically allowed by an explicit Runge-Kutta method. Figures 2.13 and 2.15 show the results of the zonal flow over an isolated mountain and the Rossby-Haurwitz wave with a large time step of 4 hours. For these configurations, the Courant number varies from 40 to 80. These numerical solutions are similar to those of Figures 2.8 and 2.11. The differences between the results with a time step of 4 hours and 1 hour are shown in Figures 2.14 and 2.16. On the one hand, for the zonal flow over an isolated mountain, the solution with a time step of 4 hours seems to converge as the order is increased. On the other hand, in the case of the Rossby-Haurwitz

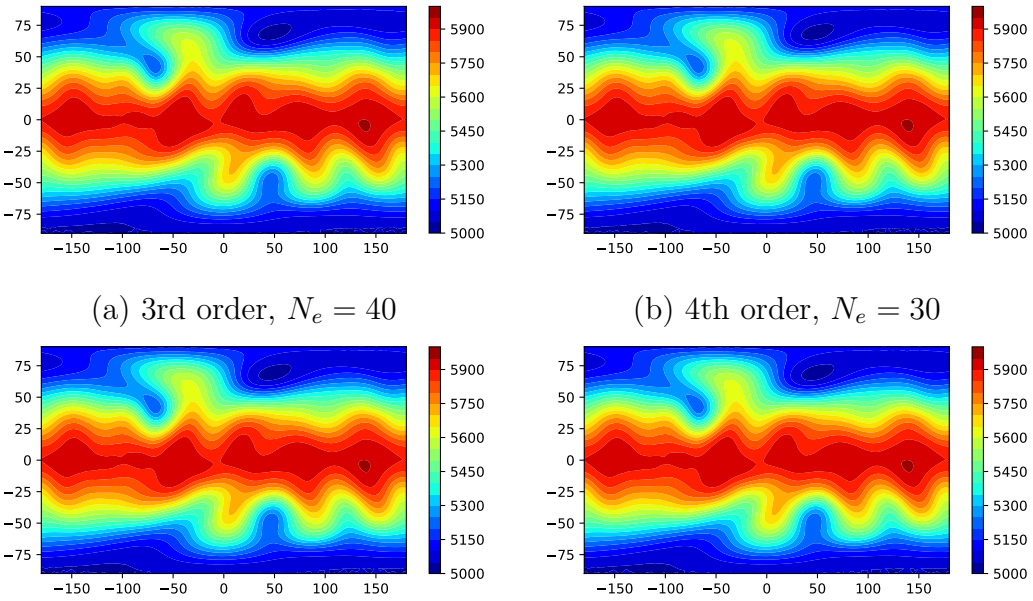


(a) Mass

(b) Energy

(c) Potential enstrophy

Figure 2.12: Time traces of the normalized errors of conserved quantities for the Rossby-Haurwitz wave.



(a) 3rd order, $N_e = 40$

(b) 4th order, $N_e = 30$

(c) 5th order, $N_e = 24$

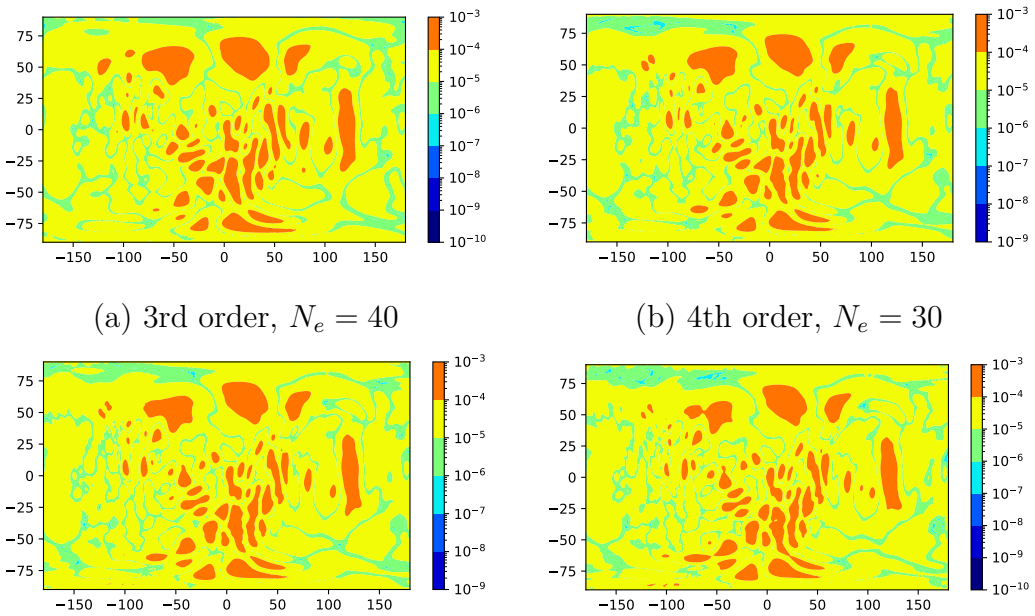
(d) 6th order, $N_e = 20$

Figure 2.13: Height field of the zonal flow over an isolated mountain at day 15 using orders 3, 4, 5 and 6 with a large timestep size of 4 hours. A global grid with 86400 degrees of freedom is used.

wave, significant differences are visible in the solutions of orders 5 and 6, while the solutions of orders 3 and 4 seem less sensitive to the increase in the time step size. This could be another indication that aliasing errors are present in high-order solutions. The success of these simulations can be largely attributed to the high accuracy with which the linear part of the equations is solved.

2.6 Summary and conclusions

The high-order methods discussed in this paper were applied to the shallow-water equations on the rotated cubed-sphere grid. Results from various test cases indicate that the proposed numerical algorithms produce realistic results. In particular, the convergence of the spatial and temporal patterns is well behaved as the order of accuracy is increased. Results are comparable to those obtained from more complicated methods based on the integral form of the equations of motion.



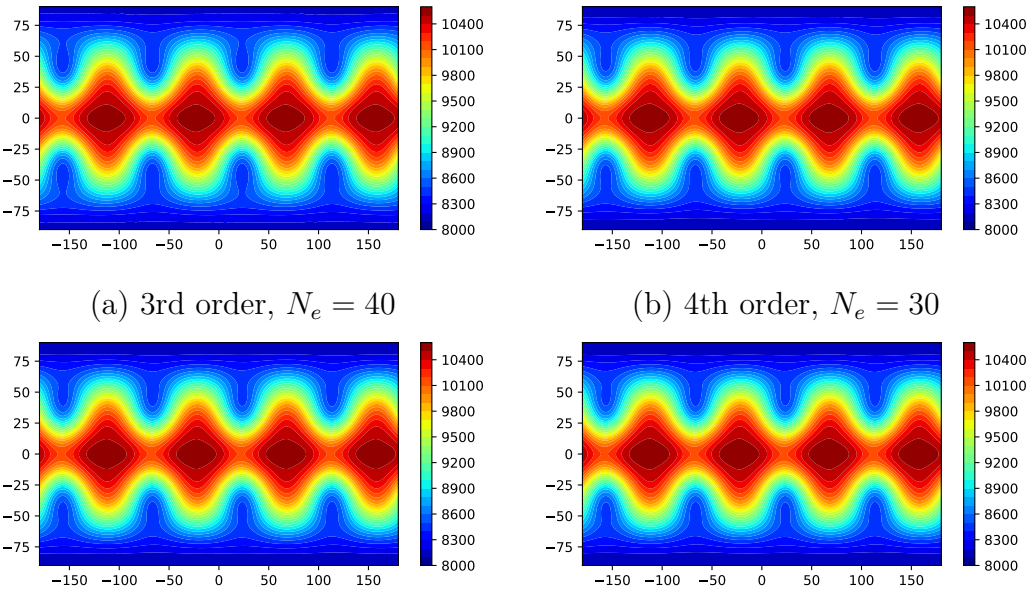
(a) 3rd order, $N_e = 40$

(b) 4th order, $N_e = 30$

(c) 5th order, $N_e = 24$

(d) 6th order, $N_e = 20$

Figure 2.14: Difference between timestep sizes of 4 hours and 1 hour for the zonal flow over an isolated mountain at day 15 using orders 3, 4, 5 and 6. A global grid with 86400 degrees of freedom is used.



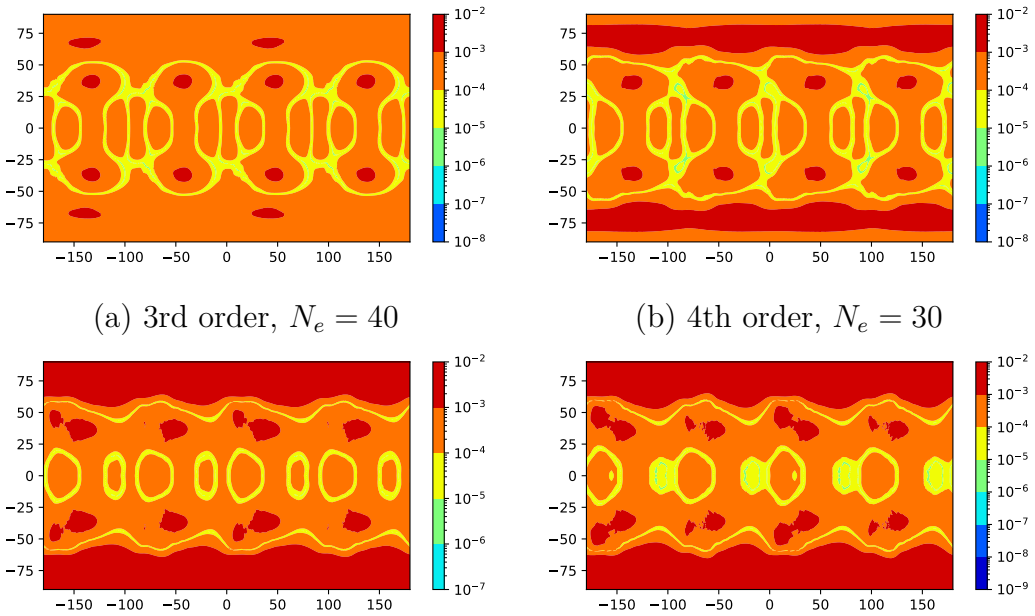
(a) 3rd order, $N_e = 40$

(b) 4th order, $N_e = 30$

(c) 5th order, $N_e = 24$

(d) 6th order, $N_e = 20$

Figure 2.15: Height field of the Rossby-Haurwitz wave at day 14 using orders 3, 4, 5 and 6 with a large timestep size of 4 hours. A global grid with 86400 degrees of freedom is used.



(a) 3rd order, $N_e = 40$

(b) 4th order, $N_e = 30$

(c) 5th order, $N_e = 24$

(d) 6th order, $N_e = 20$

Figure 2.16: Difference between timestep sizes of 4 hours and 1 hour for the Rossby-Haurwitz wave at day 14 using orders 3, 4, 5 and 6. A global grid with 86400 degrees of freedom is used.

Experiments show that mass is conserved at machine precision. A notable result is the ability of the schemes to perform accurate simulations with very large time steps for flows with complex structures.

The EPI integrators presented in this work offer not only high accuracy but new pathways to further improvements in making exponential integrators more computationally appealing. The authors plan to pursue this line of research in future works. An important task ahead is the extension of this work to a comprehensive three-dimensional atmospheric model with the space-time tensorial approach to describe the Euler equations. Improvements to the performance of the exponential solver is underway to bring the algorithm closer to the performance obtained with semi-implicit semi-Lagrangian methods (Robert, 1982).

2.7 Code availability

The code used to generate the results in this work is available under the GNU Lesser General Public License (LGPL) version 2.1 at
`https://doi.org/10.5281/zenodo.5014876` and
`https://gitlab.com/stephane.gaudreault/jcp2021_highorder_sw`.

The EPIC package implements the exponential integrators and test cases presented in section 2.5.1. The code and its license are available at
`http://faculty.ucmerced.edu/mtokman/#software`.

A MATLAB implementation of KIOPS is available under the GNU LGPL version 2.1 at `https://gitlab.com/stephane.gaudreault/kiops`.

Acknowledgements

The authors sincerely thank Christopher Subich, Ayrton Zadra and Janusz Pudykiewicz for providing valuable comments on an earlier draft of this manuscript. They also thank two anonymous reviewers and Pedro Peixoto for providing comments on this manuscript. This work was supported in part by a grant no. 1115978 from the National Science Foundation, Computational Mathematics Program.

Chapter 3

Second-order Rosenbrock-Exponential (ROSEXP) Methods for Partitioned Differential Equations

The text of this chapter is a reprint of an article that will be submitted soon Dallerit, V., Buvoli, T., Tokman, M., & Gaudreault, S. (2022). Second-order rosenbrock-exponential (ROSEXP) methods for partitioned differential equations. *to be submitted to SIAM Journal on Scientific Computing*

3.1 Introduction

Many scientific and engineering problems involve dynamics driven by several processes of different nature. Often such systems are modeled by differential evolution equations with a forcing term that is comprised of several additive components. These additive terms can represent the influence of each of the driving mechanisms. A well-known example of such system is an advection-diffusion equation where the evolution is governed by the advective and diffusive forces modeled by two additive terms with first-order and second-order derivatives respectively. In general, a

two-term forcing model can be written as an initial-value problem of the form

$$y' = f_1(y) + f_2(y) \tag{3.1a}$$

$$y(t_0) = y_0. \tag{3.1b}$$

Frequently, the forcing terms $f_i(y)$ represent processes occurring over a wide range of temporal scales. As a result the differential equations modeling such system are stiff with stiffness arising from either or both of the additive forcing terms. Such additive forcing structure can be exploited in construction of an efficient temporal numerical integrator to solve the model equations. This can be accomplished, for example, through the use of a splitting (MacNamara & Strang, 2016) or a partitioned approach (Ascher, Ruuth, & Wetton, 1995; Belytschko, Yen, & Mullen, 1979). Splitting methods have been extensively used in literature to solve such problems but construction of efficient splitting integrators of order larger than two tends to be challenging (Blanes & Casas, 2005). Partitioned methods, on the other hand, can offer easier extension to higher order methods. Some of the best-known partitioned integrators are implicit-explicit (IMEX) methods (Ascher, Ruuth, & Spiteri, 1997) which have been used for a wide range of applications (Kennedy & Carpenter, 2003; Pareschi & Russo, 2005; Keyes et al., 2013; Hundsdorfer, Verwer, & Hundsdorfer, 2003). IMEX techniques treat one component of the forcing term implicitly and the other explicitly. The idea of a partitioned method has also been extended to implicit-implicit methods (Sandu & Günther, 2015) and implicit-exponential (IMEXP) integration (Luan, Chinomona, & Reynolds, 2020; Ascher et al., 2021; Y. J. Chen et al., 2020). IMEX methods are appropriate on problems where one of the forcing terms is responsible for stiffness in the system. If the other term is stiff as well then implicit-implicit or implicit-exponential (IMEXP) approaches have been used. Fewer options have been introduced for problems with nonlinear-nonlinear additive forcing structure where both terms are stiff (Ascher et al., 2021; Y. J. Chen et al., 2020). Here we present a novel way to construct implicit-exponential-type methods for precisely such systems. In other words, we develop a new way to construct partitioned time integration schemes that treat f_1 implicitly and f_2 exponentially for problems where both of these functions are nonlinear and their Jacobians $J_1 = \frac{\partial f_1}{\partial y}$ and $J_2 = \frac{\partial f_2}{\partial y}$ are stiff. This approach

is particularly advantageous to use when one of the forcing terms can be treated implicitly in a very efficient way, e.g. when a fast preconditioner exists for this portion of the Jacobian. We extend the work in (Luan, Tokman, & Rainwater, 2017) to problems where both f_1 and f_2 are nonlinear and introduce a new anzatz for construction of such partitioned implicit-exponential integrators which can potentially be extended to higher order methods. We also describe a convenient way to visualize and to assess the stability of the methods and choose schemes with favorable stability properties. The efficiency and accuracy of the new techniques are demonstrated on a set of test problems in a numerical study which also includes a thorough comparison of the performance of the new methods with previously introduced partitioned schemes for such problems.

The article is organized as follows. The first section briefly reviews the exponential and Rosenbrock methods which serve as a building blocks of our new techniques. Section 3.3 introduces the novel ansatz for the partitioned implicit-exponential methods and presents construction of the new second-order schemes of this type. Linear stability analysis of the new methods is included in section 3.4 where we also show that some of our scheme are A-stable. Finally, in section 3.5 we validate and compare performance of our methods to other techniques using several numerical test problems.

3.2 Review of basic exponential and Rosenbrock methods

The new partitioned methods which will be introduced in section 3.3 use both exponential and Rosenbruck-type integration to advance eq. (3.1a) in time. Thus here we present a brief overview of these two approaches as the building blocks of our new schemes.

Consider the following (unpartitioned) system of ordinary differential equations (ODEs)

$$\frac{dy}{dt} = f(y), \qquad y(t_0) = y_0, \qquad y \in \mathbb{R}^N,\ f : \mathbb{R}^N \to \mathbb{R}^N \tag{3.2}$$

where y represents some unknown dynamically changing properties of the system

and f describes all forces driving the system. Suppose we are interested in computing the solution to this system over an interval $t \in [t_0, T]$. Letting h be the discretization step size and $y_n = y(t_n)$ denote the approximate solution at $t_n = hn$, one can expand eq. (3.2) in a Taylor series to obtain:

$$\frac{dy}{dt}(t) = f(y_n) + J \cdot (y(t) - y_n) + r(y(t)), \tag{3.3}$$

where $J = \dfrac{df}{dy}(y_n)$ and

$$r(z) = f(z) - f(y_n) - J \cdot (z - y_n). \tag{3.4}$$

Using the integrating factor $e^{-J\,t}$ on equation eq. (3.3) we can write it in the form

$$\frac{d}{dt}\left(e^{-J\,t}y(t)\right) = e^{-J\,t}(f(y_n) - Jy_n) + e^{-J\,t}r(y(t)). \tag{3.5}$$

Integrating over the time interval $[t_n, t_n + h]$ and multiplying by $e^{J(t_n+h)}$ leads to the integral form

$$y(t_n + h) = y_n + \varphi_1(hJ)f(y_n) + \int_{t_n}^{t_n+h} e^{J(t_n+h-t)}r(y(t))dt, \tag{3.6}$$

where the matrix function φ_1 is defined as $\varphi_1(A) = (e^A - I)A^{-1}$ and I is the identity matrix. This equation, often called the Voltera equation, is the starting point for construction of different integrators through introducing approximations to the terms of the right-hand-side to estimate $y_{n+1} \approx y(t_n + h)$.

For instance, a second-order exponentially fitted Euler method (EPI2) (Minchev & Wright, 2005) can be constructed by neglecting the nonlinear integral in eq. (3.6), e.g.:

$$y_{n+1} = y_n + h\varphi_1(hJ)f(y_n), \tag{3.7}$$

The action of the matrix function φ_1 on a vector can be either evaluated exactly or approximated depending on the properties of the matrix J. For example, this action can often be calculated exactly if N is small or if J is diagonal. When the Jacobian is large and sparse, a variety of approximation techniques such as Taylor expansions (Al-Mohy & Higham, 2011), Krylov-based algorithms (Gaudreault,

Rainwater, & Tokman, 2018) or Leja methods (Caliari, Vianello, & Bergamaschi, 2004) can be used.

A one-stage second-order Rosenbrock scheme (Rosenbrock, 1963), denoted here ROS2, could be derived by analogously neglecting the integral in eq. (3.6) but also replacing the φ_1 function by its Padé approximant of order [0/1] :

$$y_{n+1} = y_n + h\left(I - \frac{h}{2}J\right)^{-1} f(y_n) \tag{3.8}$$

It should be mentioned that the ROS2 scheme is very close in its formulation (differing only by a factor $\frac{1}{2}$ in front of the Jacobian) to the linearized Euler method

$$y_{n+1} = y_n + h\left(I - hJ\right)^{-1} f(y_n) \tag{3.9}$$

However, the linearized Euler method is only first-order.

Both EPI2 and ROS2 require evaluation of matrix functions of the full (unpartitioned) Jacobian J and share similar properties in terms of linear stability. The choice between the two therefore depends on the nature of the problem to be solved. For example, when an efficient solver for a linear system of equations is available (e.g. a direct solver or a preconditioned iterative method) then the ROS2 scheme may be a judicious choice. Otherwise, exponential approximation of $\varphi_1(hJ)f_n$ might be more efficient when used with a fast algorithm such as KIOPS method (Gaudreault, Rainwater, & Tokman, 2018). The case where an efficient linear solver is only available for a portion of the Jacobian will be discussed in the next section.

3.3 New nonlinear-nonlinear partitioned Rosenbrock-Exponential (ROSEXP) methods

3.3.1 General framework

Below we introduce a framework for developing efficient numerical schemes for solving nonlinear-nonlinear partitioned problems of the form:

$$y' = f(y) = f_1(y) + f_2(y), \qquad y(t_0) = y_0, \tag{3.10}$$

where f_1 and f_2 are both stiff. To develop such schemes we use the idea of a generalized EPI methods introduced in (Tokman, 2011). Specifically, in (Tokman, 2011) it was proposed to construct approximation to the solution of eq. (3.10) in the form

$$y_{n+1} = y_n + \sum_i \psi_i(hJ) f(z_i) \tag{3.11}$$

where $\psi_i(hJ)$ are some functions of a matrix J which in some way approximates the Jacobian or a portion of the Jacobian and z_i's are vectors approximating the solution on some nodes. The functions ψ_i are chosen to construct integrators of a particular type. For example, these functions can be exponential or rational depending on whether an exponential, an implicit or a hybrid method is being built.

We extend this idea to the case of a partitioned right-hand side and allow these functions to be a product of exponential or rational functions, each applied to either the Jacobian of f_1 or f_2. For low order methods, this idea can be expressed using the following ansatz:

$$y_{n+1} = y_n + Q_{1,1}(hJ_1)Q_{2,1}(hJ_2)hf_1(y_n) + Q_{1,2}(hJ_1)Q_{2,2}(hJ_2)hf_2(y_n) \tag{3.12}$$

where: $Q_{i,j}$ are analytic functions (rational or exponential-like functions), J_1 and J_2 are respectively the Jacobians of f_1 and f_2 evaluated at y_n. Note that the order of the functions $Q_{1,i}$ and $Q_{2,i}$ in the above ansatz can be changed to derive different schemes. Since matrices J_1 and J_2 do not necessarily commute we can also consider the following flipped ansatz that reverses the order of application of the functions:

$$y_{n+1} = y_n + Q_{2,1}(hJ_2)Q_{1,1}(hJ_1)hf_1(y_n) + Q_{2,2}(hJ_2)Q_{1,2}(hJ_1)hf_2(y_n) \tag{3.13}$$

Because the functions $Q_{i,j}$ are only applied to either J_1 or J_2, availability of efficient solvers that estimate $Q_{i,j}$'s applied to each of these matrices separately for some problems can result in significant computational savings compared to a method which involves only the full Jacobian $J = J_1 + J_2$. This ansatz is very general and allows the construction of many methods. In the following section, we present the derivation of several efficient second order schemes.

3.3.2 Construction of second-order schemes

In this section we derive the order conditions necessary for a scheme based on the ansatz eq. (3.12) to have second order of convergence. To do so, we assume that the numerical solution at time t_n is exact ($y(t_n) = y_n$) and match the numerical solution at the next time step y_{n+1} to $y(t_{n+1})$, the exact solution at time t_{n+1} up to second order. This will add some restrictions on the function $Q_{i,j}$ that will be used to derive second order schemes.

First, we assume that the functions $Q_{i,j}$ are analytic, so that we have the following Taylor series representation

$$Q_{i,j}(hJ) = \alpha_{i,j} + \beta_{i,j} hJ + O(h^2)$$

Without loss of generality, we can assume that $\alpha_{i,j} = 1$. If it is not the case the function can be rescaled. Moreover, the product of the scaling coefficient must be equal to 1 for consistency. We use these expansion of $Q_{i,j}$ to obtain the following form of the numerical solution

$$\begin{aligned}
y_{n+1} &= y_n + Q_{1,1}(hJ_1)\, Q_{2,1}(hJ_2)\, hf_1(y_n) + Q_{2,1}(hJ_1)\, Q_{2,2}(hJ_2)\, hf_2(y_n) \\
&= y_n + (1 + \beta_{1,1} hJ_1)(1 + \beta_{2,1} hJ_2)\, hf_1(y_n) \\
&\qquad + (1 + \beta_{1,2} hJ_1)(1 + \beta_{2,2} hJ_2)\, hf_2(y_n) + O\left(h^3\right) \\
&= y_n + h\left(f_1(y_n) + f_2(y_n)\right) \\
&\qquad + h^2\left[(\beta_{1,1} J_1 + \beta_{2,1} J_2)\, f_1(y_n) + (\beta_{1,2} J_1 + \beta_{2,2} J_2)\, f_2(y_n)\right] + O\left(h^3\right)
\end{aligned}$$

On the other side, the exact solution at time t_{n+1} can be expanded as follows,

$$y(t_{n+1}) = y(t_n) + h(f_1(y_n) + f_2(y_n)) + \frac{h^2}{2}(J_1 + J_2)(f_1(y_n) + f_2(y_n)) + O\left(h^3\right)$$

After matching the terms up two second order, we have the following conditions on the $Q_{i,j}$ functions:

$$\beta_{1,1} = \beta_{2,1} = \beta_{1,2} = \beta_{2,2} = 1/2$$

Table 3.1 presents several schemes that satisfy these conditions. These methods were obtained by choosing the $Q_{i,j}$ functions to be exponential or rational functions similar to those found in formulas for the EPI2 and ROS2 schemes. For this reason,

if we consider the extreme case partitioning $f_1 = 0, f_2 = f$ then all the schemes from table 3.1 reduce to the EPI2 method. Likewise, if $f_1 = f, f_2 = 0$ then all the schemes simplify to the ROS2 scheme.

Coefficients	Scheme
$Q_{1,1}(z) = Q_{1,2}(z) = \left(I - \frac{z}{2}\right)^{-1}$ $Q_{2,1}(z) = Q_{2,2}(z) = \varphi_1(z)$	**RosExp2** – ansatz eq. (3.12) $y_{n+1} = y_n + \left(I - \frac{h}{2}J_1\right)^{-1} \varphi_1(hJ_2)hf(y_n)$ **ExpRos2** – ansatz eq. (3.13) $y_{n+1} = y_n + \varphi_1(hJ_2)\left(I - \frac{h}{2}J_1\right)^{-1} hf(y_n)$
$Q_{1,1}(z) = Q_{1,2}(z) = \left(I - \frac{z}{2}\right)^{-1}$ $Q_{2,1}(z) = \frac{1}{2}\left(e^z + I\right)$ $Q_{2,2}(z) = \varphi_1(z)$	**PartRosExp2** – ansatz eq. (3.12) $y_{n+1} = y_n + \left(I - \frac{h}{2}J_1\right)^{-1} \left[\frac{1}{2}\left(e^{hJ_2} + I\right) hf_1(y_n) + \varphi_1(hJ_2)hf_2(y_n)\right]$ **PartExpRos2** – ansatz eq. (3.13) $y_{n+1} = y_n + \frac{1}{2}\left(e^{hJ_2} + I\right)\left(I - \frac{h}{2}J_1\right)^{-1} hf_1(y_n) + \varphi_1(hJ_2)\left(I - \frac{h}{2}J_1\right)^{-1} hf_2(y_n)$

Table 3.1: Second-order Rosenbrock-Exponential schemes

As mentioned above, implicit-exponential (IMEXP) schemes for linear-nonlinear partitioned problems were were introduced in (Luan, Tokman, & Rainwater, 2017). In particular, the scheme *HImExp2N* (equation (4.2) in (Luan, Tokman, & Rainwater, 2017)) is derived for problems of the type $y' = Ly + N(y)$ where L is a linear and N is the nonlinear operators. Interpreting this scheme in the context of our anzatz and the derived order conditions, we can easily see that the method *HImExp2N* also satisfies the order conditions (3.14a). Thus, *HImExp2N* can also be used for the nonlinear-nonlinear partitioned problems and is in fact, a second order scheme for problems of the form eq. (3.1a). Using the notations from this article, *HImExp2N* can be written as:

$$Y_1 = y_n + \frac{h}{2}\left(I - \frac{h}{2}J_1\right)^{-1} f(y_n) \tag{3.14a}$$

$$y_{n+1} = y_n + h\left(I - \frac{h}{2}J_1\right)^{-1} f(y_n) + 2h\varphi_2(hJ_2)(f_2(Y_1) - f_2(y_n)) \tag{3.14b}$$

Other ideas of partitioned nonlinear-nonlinear schemes were also explored in (Y. J. Chen et al., 2020; Ascher et al., 2021), but both of the methods derived in these publications are limited to first order of accuracy. We will include the SIERE and SBDF2ERE schemes derived in these papers in our comparisons:

- *SIERE* (Y. J. Chen et al., 2020):

$$y_{n+1} = y_n + h\left(I - hJ_1\right)^{-1}\left(f_1(y_n) + \varphi_1(hJ_2)f_2(y_n)\right)$$

- *SBDF2ERE* (Ascher et al., 2021):

$$y_{n+1} = y_n + \frac{1}{3}\left(I - \frac{2h}{3}J_1\right)^{-1}\left(y_n - y_{n-1} + 2hf_1(y_n) + 2h\varphi_1(hJ_2)f_2(y_n)\right)$$

Note that all the schemes are written so that the f_1 partition is treated using the rational function, while the f_2 partition is treated exponentially.

3.4 Linear stability

As mentioned in the previous section, our work focuses on nonlinear-nonlinear partitioning where both f_1 and f_2 are stiff. In this context it is important to have good stability properties. In order to study the linear stability of partitioned integrators, we consider the following problem:

$$y' = \lambda_1 y + \lambda_2 y \quad \text{where} \quad \lambda_1, \lambda_2 \in \mathbb{C}. \tag{3.15}$$

Any one-step method applied to eq. (3.15) reduces to the recurrence $y_{n+1} = R(z_1, z_2)y_n$ where $z_1 = h\lambda_1$, $z_2 = h\lambda_2$ and $R(z_1, z_2)$ is the stability function of the scheme.

	RosExp2, ExpRos2, HImExp2N	*PartRosExp2, PartExpRos2*	*SIERE*
$R(z_1, z_2)$	$1 + \frac{2\varphi_1(z_2)}{2-z_1}(z_1 + z_2)$	$\frac{2+z_1}{2-z_1}e^{z_2}$	$\frac{e^{z_2}}{1-z_1}$

Table 3.2: Stability functions for one-step methods.

The scheme is then stable if $|R(z_1, z_2)| \leq 1$. The stability functions for all one-step methods considered in this work are listed in table 3.2.

Because the *SBDF2ERE* scheme is a multi-step method, we determine stability differently. After applying the method to the linear problem eq. (3.15) we obtain the recurrence

$$y_{n+1} + R_1(z_1, z_2)y_n + R_0(z_1, z_2)y_{n-1} = 0$$

where $R_1(z_1, z_2) = -\frac{2}{3-2z_1}(1 + e^{z_2})$ and $R_0(z_1, z_2) = \frac{1}{3-2z_2}$. The method will be stable when w_1 and w_2, the roots of the polynomial $w^2 + R_1(z_1, z_2)w + R_0(z_1, z_2)$, satisfy $|w_i| \leq 1$.

Because both z_1 and z_2 are complex-valued, the stability regions for both one-step and multistep methods are challenging to visualize. To simplify our presentation of stability, we will use $A(\boldsymbol{\alpha})$-stability. For a stability function $R(z)$ of a single complex variable a method is said to be $A(\boldsymbol{\alpha})$-stable if it includes a sector of an angle $\boldsymbol{\alpha}$ in its stability region. $\boldsymbol{\alpha}$ is defined as:

$$\boldsymbol{\alpha} = \max\{\alpha : \forall z \quad (z \in \mathbb{C}^- \wedge |\arg(z) - \pi| \leq \alpha) \Rightarrow |R(z)| \leq 1\}. \tag{3.16}$$

For non-partitioned schemes $\boldsymbol{\alpha}$ is the maximum value of the angle such that the method is stable for all complex z values in the sector delimited by the lines with an angle $-\boldsymbol{\alpha}$ and $+\boldsymbol{\alpha}$ with respect to the negative real axis. This value ranges from 0° if the method is only stable on the negative real axis, to 90° for a method that is stable in the entire left half-plane. When the $\boldsymbol{\alpha}$-stability of a scheme is equal to 90°, we say that the method is A-stable.

By fixing either z_1 or z_2, we can reduce the stability function of a partitioned scheme to a function of a single complex variable. We can then compute the stability angle $\boldsymbol{\alpha}$ in the remaining free variable. This can be expressed mathematically

as

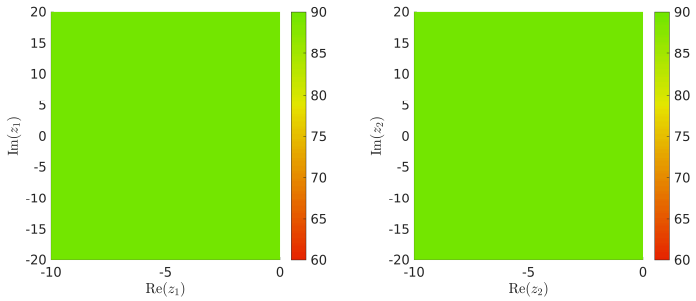
$$\text{fixing } z_1\text{:}\quad \boldsymbol{\alpha}(z_1) = \max\{\alpha : \forall z_2 \quad (z_2 \in \mathbb{C}^- \wedge |\arg(z_2) - \pi| \leq \alpha) \Rightarrow |R(z_1, z_2)| \leq 1\}, \tag{3.17a}$$

$$\text{fixing } z_2\text{:}\quad \boldsymbol{\alpha}(z_2) = \max\{\alpha : \forall z_1 \quad (z_1 \in \mathbb{C}^- \wedge |\arg(z_1) - \pi| \leq \alpha) \Rightarrow |R(z_1, z_2)| \leq 1\}, \tag{3.17b}$$

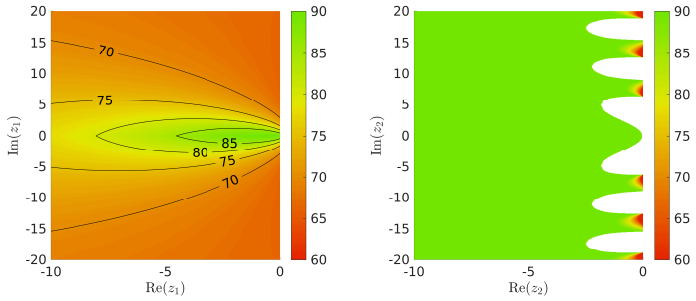
Fixing z_1 or z_2 over a grid of values and using color to represent the stability angle makes it possible to easily visualize the stability of each method. Figure 3.1 shows the $\boldsymbol{\alpha}$-stability for the schemes presented in the previous section. Note that ordinarily due to the high dimensionality of the stability function $R(z_1, z_2)$ it is difficult to assess the properties of the stability regions. However, using the approach described above visualization of the stability regions becomes intuitive and the geometry of the stability region can be easily discerned from the plots of type fig. 3.1. To our knowledge this approach to visualizing linear stability of a method has not been used before.

In fig. 3.1 (a), we see that $\boldsymbol{\alpha} = 90°$ for all values of z_1 and z_2. This implies that the schemes *PartRosExp2*, *PartExpRos2*, and *SIERE* are all A-stable. This can be formally proven by observing that the stability function for each of these methods is a product of two A-stable functions in z_1 and z_2, respectively (e.g. the stability function of *PartRosExp2* and *PartExpRos2* are the products of the functions $R_1(z) = e^z$ and $R_2(z) = \frac{2+z}{2-z}$). Since both of these functions are A-stable, the product must also be A-stable.

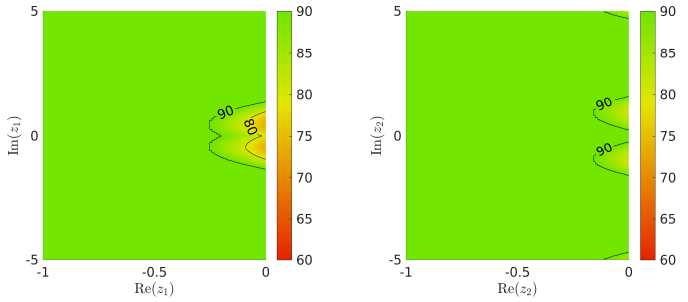
Figure 3.1 (b), shows that the stability of the schemes *RosExp2, ExpRos2* and *HImExp2N* is more restricted. Specifically, if z_2 is close to the imaginary axis, then the stability region for z_1 is either bounded or limited. Conversely, if the real part of z_2 is sufficiently negative, then the stability is retained. This indicates that these schemes should be used for problems where f_2 is sufficiently diffusive. Finally, the stability of the scheme *SBDF2ERE*, presented in fig. 3.1 (c), is good overall with some limitations close to the origin.



(a) Stability for the schemes *PartRosExp2, PartExpRos2*, and *SIERE*



(b) Stability for the schemes *RosExp2, ExpRos2* and *HImExp2N*



(c) Stability for the scheme *SBDF2ERE*

Figure 3.1: $\boldsymbol{\alpha}$-stability angles for the partitioned schemes when z_1 is fixed (left-column) or z_2 is fixed (right-column). The x and y axis of the plots in the left and right columns respectively correspond to the real and imaginary parts of z_1 and z_2. The color represents the stability angle $\boldsymbol{\alpha}$ defined in eqs. (3.17a) and (3.17b). The white regions correspond to parameter values where the stability region is bounded and therefore not $\boldsymbol{\alpha}$-stable even for $\boldsymbol{\alpha} = 0$.

3.5 Numerical experiments

In this section, we will verify the accuracy and test the performance of the newly derived schemes using several numerical examples.

The stability properties of the new schemes for linear equations with constant coefficients discussed in the previous section provide necessary, but not sufficient conditions for the stability of variable coefficients and nonlinear problems. In this section we summarize numerical experiments which confirm that the conclusions of our analysis for the simple linear problem can also be applied to more complicated problems. We choose three test problems commonly used to validate and compare the different schemes for stiff partitioned systems.

3.5.1 Advection-diffusion PDE (AdvDiff)

We consider the following advection-diffusion PDE:

$$\frac{\partial u}{\partial t} + \frac{\partial}{\partial x}\left(\alpha_0 u + \alpha_1 u^2\right) = \frac{\partial}{\partial x}\left[(\beta_0 + \beta_1 u)\frac{\partial u}{\partial x}\right], \quad x \in [0,1], \quad t \in [0, 0.1], \quad (3.18)$$

We use a Gaussian function as the initial condition $u(x,0) = e^{-5000(x-0.2)^2}$ and homogeneous Dirichlet boundary conditions $u(0,t) = u(1,t) = 0$. We also consider two sets of parameters: the first correspond to a linear problem with $\alpha_0 = 5$, $\alpha_1 = 0$, $\beta_0 = 10^{-2}$, and $\beta_1 = 0$, and the second representing a nonlinear problem with $\alpha_0 = 5$, $\alpha_1 = 5$, $\beta_0 = 5 \times 10^{-4}$, and $\beta_1 = 10^{-1}$. Figure 3.2 shows the solution u of this PDE at the initial and final time. Equation eq. (3.18) is discretized in space using standard second-order centered finite differences with 1000 grid points. This discretization leads to a system of N ordinary differential equations that can be written as:

$$u' = f_{\text{adv}}(u) + f_{\text{diff}}(u)$$

where $f_{\text{adv}}(u)$ correspond to the discretized advection term $\frac{\partial}{\partial x}\left(\alpha_0 u + \alpha_1 u^2\right)$ and $f_{\text{diff}}(u)$ correspond to the discretized diffusion term $\frac{\partial}{\partial x}\left[(\beta_0 + \beta_1 u)\frac{\partial u}{\partial x}\right]$. In the next section, we will explore the cases where $f_1 = f_{\text{adv}}, f_2 = f_{\text{diff}}$ (rational advection /

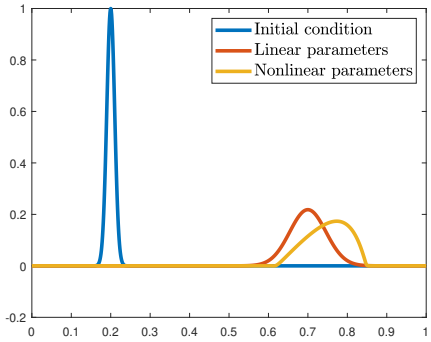


Figure 3.2: Solution at initial and final time for the PDE eq. (3.18) with both sets of parameters

exponential diffusion) as well as $f_1 = f_{\text{diff}}, f_2 = f_{\text{adv}}$ (rational diffusion / exponential advection).

3.5.2 Schnakenberg equation with non-linear diffusion (Schnakenberg_NL)

The following equations describe two reacting and diffusing chemical species (u and v):

$$\begin{aligned}\frac{\partial u}{\partial t} &= \gamma(a - u + u^2 v) + \nabla.(u^{\beta_1}\nabla u),\\ \frac{\partial v}{\partial t} &= \gamma(b - u^2 v) + d\ \nabla.(v^{\beta_2}\nabla v), \qquad (x, y) \in [0, 1]^2\end{aligned}$$

where $a = 0.1, b = 0.9, d = 10, t_{end} = 10^{-2}$ for $\gamma \in \{1000, 10000\}$, and $t_{end} = 10^{-3}$ for $\gamma = 10^5$. The parameter γ is used to control the stiffness of the reaction. The initial conditions were chosen to be a sum of cosine functions with different frequencies along both the x and y axes. The boundary conditions are periodic in both directions. The diffusion terms are discretized using the standard second-order finite differences on a uniform grid with $N_x = N_y = 128$. The reaction terms are treated exponentially while the diffusion terms are treated using the rational function.

3.5.3 1D Semilinear parabolic problem (Semilinear_para)

Finally, we use the following one-dimensional linear-nonlinear parabolic problem described in (Hochbruck & Ostermann, 2005):

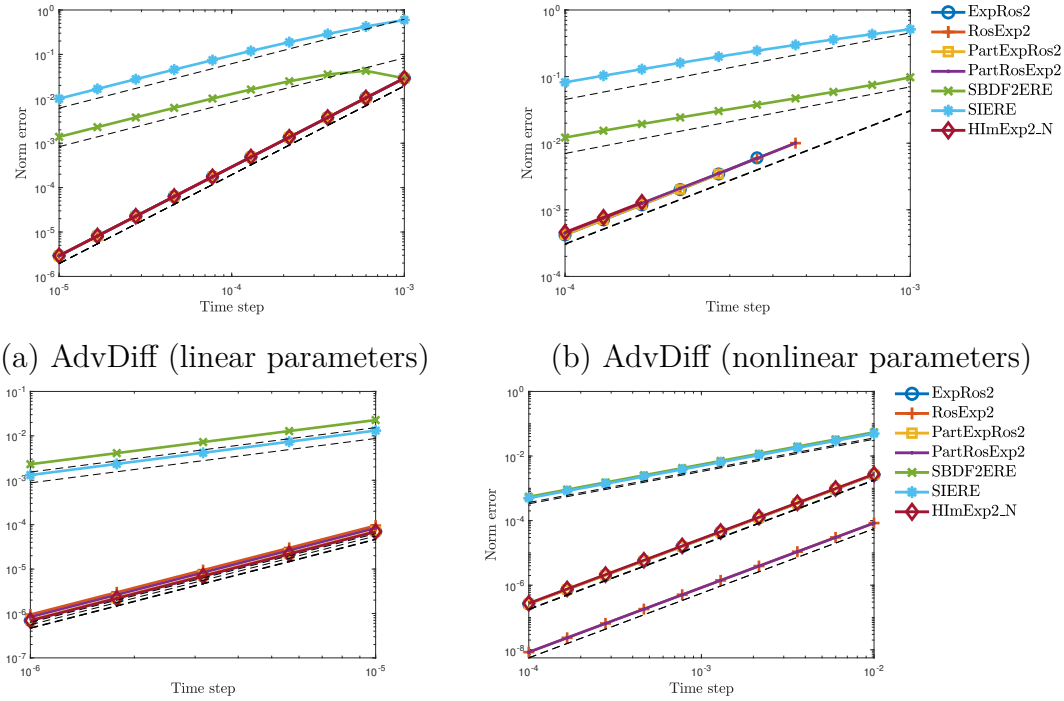
$$\frac{\partial u}{\partial t}(x,t) - \frac{\partial^2 u}{\partial x^2}(x,t) = \int_0^1 u(x,t)dx + \phi(x,t) \qquad x \in [0,1], t \in [0,1],$$

with the homogeneous Dirichlet boundary conditions. The source function ϕ is chosen so that $u(x,t) = x(1-x)e^t$ is the exact solution. This problem was originally designed to demonstrate the order reduction that some exponential integrators can suffer when applied to stiff problems. It is therefore used here to validate that no such order reduction is exhibited by our schemes. The diffusion term is discretized using the standard second-order finite differences on a uniform grid with $N_x = 400$. The nonlinear terms on the right-hand-side are treated exponentially while the diffusion term is treated using the rational function.

3.5.4 Numerical results

Numerical examples presented below verify the order of convergence of the newly derived methods and compare their performance with the existing methods described above. The implementation of the integrators was done in MATLAB 2020b. For all the schemes, we use the KIOPS method introduced in (Gaudreault, Rainwater, & Tokman, 2018) to approximate the products of exponential and $\varphi-$functions with vectors. This method allows us to approximate both exponential functions in the schemes *PartExpRos2* and *PartRosExp2* at once as a single computation. The rational functions are approximated using the GMRES method (Saad & Schultz, 1986) with an incomplete LU factorization preconditioner. Because the scheme *BDF2ERE* is a multi-step integrator where the solution at the current and previous time step must be known, the initial step must be treated differently. In this work, the initial time step is computed using the 2^{nd} order *EPI2* method. The error is defined as the discrete $2-$norm between the approximate solution and a reference solution computed using MATLAB's ode15s integrator with absolute and relative tolerances set to 10^{-14}.

In the first set of tests, we verify the order of convergence of all the methods on the problems presented above. Figure 3.3 shows the convergence plot (error vs time-step in log-log scale) on the linear and nonlinear advection diffusion PDE, Schnakenberg PDE and the semilinear parabolic problems. Note that for the advection diffusion PDE, we used $f_1 = f_{\text{adv}}$ and $f_2 = f_{\text{diff}}$. We can see that, as expected, the methods *SBDF2ERE* and *SIERE* both converge at first order while the methods introduced in table 3.1 and the *HImExp2N* scheme are converging at second order. We can also see that for the advection diffusion and the Schnakenberg PDE, the order of the $Q_{i,j}$ function does not influence the accuracy of the solution (ansatz eq. (3.12) vs eq. (3.13)). However, for the semilinear parabolic problem, the order does affect the accuracy. For this problem and this partitioning, applying the function of J_2 first leads to better accuracy. This case illustrates that the accuracy of the method does depend on the problem and the chosen partitioning.



(a) AdvDiff (linear parameters)

(b) AdvDiff (nonlinear parameters)

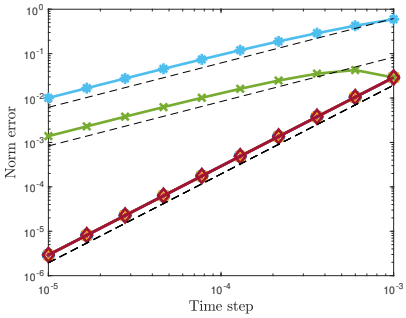
(c) Schnakenberg_NL ($\gamma = 10^3$)

(d) Semilinear_para

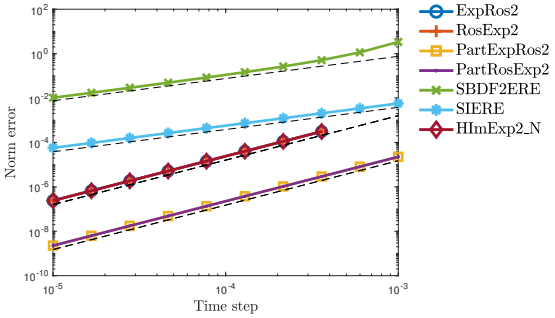
Figure 3.3: Convergence plots (error vs time step) for the linear and nonlinear AdvDiff, Schnakenberg_NL and Semilinear_para problems

Next, we want to validate the stability advantages of the new schemes that our analysis of section 3.4 predicted. We showed that for the schemes *ExpRos2*, *RosExp2* and *HImExp2N*, if z_2 is close to the imaginary axis, then stability for z_1 is either bounded or restricted. Figure 3.4 shows the convergence diagram for the advection-diffusion PDE problem. Figures 3.4a and 3.4b correspond to the problem with linear parameters while figs. 3.4c and 3.4d correspond to the nonlinear parameters. The plots on the left (figs. 3.4a and 3.4c) are obtained using the partitioning $f_1 = f_{\text{adv}}, f_2 = f_{\text{diff}}$ and the plots on the right (figs. 3.4b and 3.4d) are obtained using the partitioning $f_1 = f_{\text{diff}}, f_2 = f_{\text{adv}}$. The eigenvalues corresponding to the advection term f_{adv} are expected to be close to the imaginary axis while the eigenvalues of the diffusion term are expected to be along the negative real axis. Therefore, based on the stability analysis, we are expecting the schemes *ExpRos2*, *RosExp2* and *HImExp2N* to have worse stability for $f_2 = f_{\text{adv}}$ (right plots). For both the linear and nonlinear parameters, we see that this indeed the case and these methods are stable only for a more restrictive range of time step sizes.

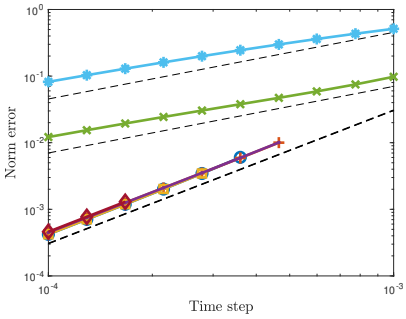
We now compare the performance of the methods on the different test problems. Figure 3.5 shows the precision diagrams (error vs CPU time) for the linear and nonlinear advection diffusion PDE, Schnakenberg PDE and the semilinear parabolic problems. We can see that the computation time for all the methods is similar. Only the *PartExpRos2* scheme is slightly more expensive because it requires two linear systems to be solved per iteration. All the other schemes require the solution of one system of linear equations and one evaluation of exponential functions. This is achieved through the use of the KIOPS method which allows evaluation of all the exponential terms at once as a single computation. Because of the increased order of convergence, the new methods are more efficient.



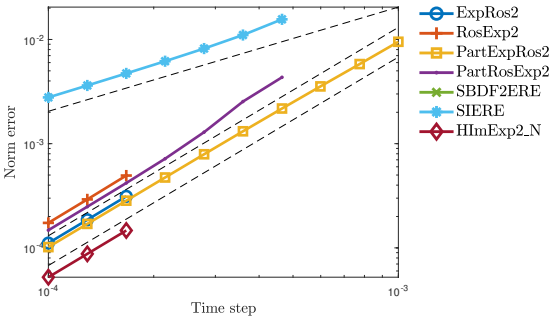
(a) AdvDiff (linear parameters) with $f_1 = f_{\text{adv}}, f_2 = f_{\text{diff}}$



(b) AdvDiff (linear parameters) with $f_1 = f_{\text{diff}}, f_2 = f_{\text{adv}}$

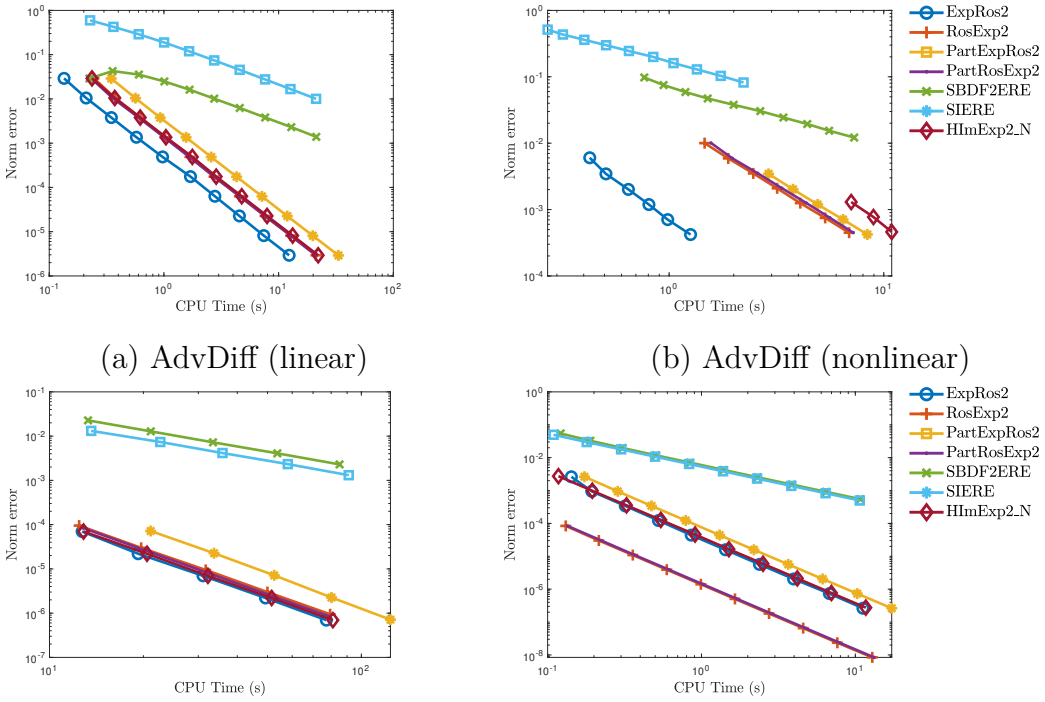


(c) AdvDiff (nonlinear parameters) with $f_1 = f_{\text{adv}}, f_2 = f_{\text{diff}}$



(d) AdvDiff (nonlinear parameters) with $f_1 = f_{\text{diff}}, f_2 = f_{\text{adv}}$

Figure 3.4: Stability comparison for the AdvDiff problem with different partitioning



(a) AdvDiff (linear)

(b) AdvDiff (nonlinear)

(c) Schnakenberg_NL ($\gamma = 10^3$)

(d) Semilinear_para ($\gamma = 10^3$)

Figure 3.5: Precision diagram (error vs CPU time) for the linear and nonlinear AdvDiff, Schnakenberg_NL and Semilinear_parabolic problems.

3.6 Conclusion

In this paper, we presented a new framework for deriving partitioned integrators for stiff systems of ODEs with nonlinear-nonlinear additive forcing terms. The new time integrators constructed using this framework are particularly efficient for problems where both nonlinear forcing terms are stiff but one of them can be solved efficiently using an implicit approach, and another can be integrated exponentially. The new ansatz that allowed us to derive specific second-order schemes can potentially be extended to construct higher-order methods which we plan to investigate in our future publications. We have used linear stability analysis and a novel way to visualize properties of a stability function to demonstrate that several of the new methods are A-stable and thus offer superior stability compared to existing schemes for similar problems. Convergence and efficient performance of the new methods have been demonstrated using several numerical examples. A thorough comparison of these schemes with integrators proposed for such problems in previous publications has been performed. We showed that the novel exponential-Rosenbrock-type methods are both more accurate and more stable than previously published methods and can be effectively used for a variety of applications.

Chapter 4

φ-order conditions and stiffness-resilient exponential time integrators

4.1 Introduction

In the previous chapters, we presented several exponential methods. As mentioned in chapter 1, two main approaches currently exist to derive such methods. The classical order conditions are obtained by comparing the Taylor expansions of the numerical scheme and exact solution. With these conditions, it is assumed that the solution and its derivatives are smooth. These assumptions can be invalidated when methods derived with classical order conditions are applied to stiff problems and can cause order reduction. To avoid order reduction in an exponential integrator applied to highly stiff problems, it is not enough to derive a scheme using classical order conditions. In (Hochbruck & Ostermann, 2005), it was shown that stiffly accurate methods have to be constructed for stiff problems. In this paper, the order reduction is avoided by deriving methods under the assumption that the Jacobian matrix can be unbounded, but the solution is still smooth. However, the solution of both the classical and stiff order conditions is highly complicated. In fact, there have been no exponential Runge-Kutta methods of order 6 and above constructed and the stiff order conditions are usually slightly weakened for order 4 and above. As a result, each new method requires a significant amount of work

and is derived on a case-by-case basis. In this chapter, we introduce the φ-order conditions based on a φ-series expansion of the exact solution. These conditions are more restrictive than the classical order conditions. The extra conditions imposed on the methods guarantee that the dominant error terms of stiff problems are canceled. For this reason, these methods are similar to the methods obtained with stiff order conditions and do not suffer from order reduction. However, we will show that the new conditions can be solved analytically making the derivation of new methods much easier. This chapter is organized as follows. First, in section 4.2, we give some background on classical order conditions and B-series theory. Then, in section 4.3, we introduce φ-order conditions and justify a general ansatz for deriving exponential schemes with useful properties. In section 4.4, we show that using this ansatz, the φ-order conditions can be solved analytically. In section 4.5, we show some of the properties that are obtained by deriving methods under this framework. Finally, section 4.6, we introduce new exponential methods using the φ-order conditions.

4.2 Classical order conditions and B-Series

This section introduces the classical order condition theory used to derive exponential schemes. This section only outlines the main results. For a more complete presentation of this theory, the reader can refer to (J. C. Butcher, 2021; J. Butcher, 2010; Tokman, 2011; Chartier, Hairer, & Vilmart, 2010).

4.2.1 Motivations

With classical order conditions, the order of convergence of a method is obtained as follows: The exact solution $y(t_n + h)$ of eq. (1.1) is comapred to the numerical scheme $y_{n+1} \approx y(t_n + h)$ around $h = 0$. A method is said to have classical order of convergence p if the local error $y(t_n + h) - y_{n+1}$ is such that $y(t_n + h) - y_{n+1} = O(h^{p+1})$.

To compute the local error, we first need to expand the exact solution $y(t_n + h)$

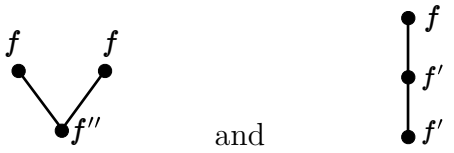
around $h = 0$.

$$y(t_n + h) = y(t_n) + h\frac{dy}{dt}(t_n) + \frac{h^2}{2}\frac{d^2y}{dt^2}(t_n) + \frac{h^3}{3!}\frac{d^3y}{dt^3}(t_n) + O(h^4)$$

Using the fact that y is the solution of the differential equation $\frac{dy}{dt} = f(y)$ and the chain rule, we get:

$$\begin{aligned} y(t_n + h) = y_n &+ hf(y_n) \\ &+ \frac{h^2}{2}\sum_{i=1}^{N}\frac{\partial f}{\partial y_i}(y_n)f_i(y_n) \\ &+ \frac{h^3}{3!}\sum_{i=1}^{N}\sum_{j=1}^{N}\left(\frac{\partial^2 f}{\partial y_i \partial y_j}f_i f_j + \frac{\partial f}{\partial y_i}\frac{\partial f_i}{\partial y_j}f_j\right)\Bigg|_{y=y_n} \\ &+ O(h^4) \end{aligned}$$

However, as the order increases, this expansion gets really complicated. The idea of Butcher's order condition theory is to associate each elementary differential to a rooted tree. This idea greatly simplifies the understanding and construction of higher order methods. For example, the elementary differentials of order 3, $\sum_{i,j}\frac{\partial^2 f}{\partial y_i \partial y_j}f_i f_j$ and $\sum_{i,j}\frac{\partial f}{\partial y_i}\frac{\partial f_i}{\partial y_f}f_j$ are respectively associated with the trees



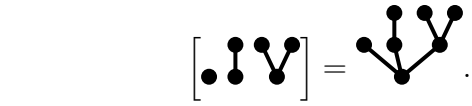
In general, the number of branches leaving from a node determines the order of the derivative and each branch is a vector that is applied to the derivative. The mapping between the rooted trees and the corresponding elementary differential is denoted: $F(\tau)$ for $\tau \in T$ where T is the set of rooted trees. Using this mapping, it is possible to rewrite the Taylor expansions of the exact solution $y(t_n + h)$ and the numerical solution y_{n+1} in terms of the rooted trees and their properties. In order to express the exact solution $y(t_n + h)$ and the numerical solution y_{n+1} in this new form, we first need to introduce some operations and functions on the rooted trees.

4.2.2 Operations and fonctions on rooted trees

First, we define the B^+ product on rooted trees.

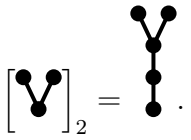
Definition 4.2.1 (B^+ product)**.** The tree $t = [t_1 t_2 ... t_n] = B^+(t_1, t_2, ..., t_n)$ is the tree obtained by connecting the root of each of the trees t_i to a new vertex. This vertex is the root of the new tree.

For example:



If the same tree is present several times, it is possible to use the exponential notation to indicate the number of repetitions. For example: $[\tau^3] = [\tau\ \tau\ \tau]$.

A special case of this product is the rooting of a tree. If the B^+ product is applied with a single tree, the operation corresponds to adding a new node as the root of the tree. When this operation is repeated k time we denote it by $t = [\tau]_k$. For example:



Based on this notation, we have the following theorem.

Theorem 4.2.1. *Let $J = f'(y_n)$ be the Jacobian matrix of the right hand side f of eq.* (1.1) *at y_n. We have:*

$$J^k F(\tau)(y_n) = F([\tau]_k)(y_n)$$

Proof. Direct consequence of the definition of the mapping $F(t)$. □

This theorem plays an important role in the construction of exponential methods as they use functions of the Jacobian matrix applied to vectors. After expanding these functions as a Taylor series, it allows the computation of the B-series of the numerical scheme.

Next, we present some functions on the rooted trees. These functions will be used later to compute the coefficients of some B-series.

For a tree $t = [{t_1}^{m_1} {t_2}^{m_2} ... {t_k}^{m_k}]$, we can define the following functions:

- Order:

$$|t| = \begin{cases} 1 & \text{if } t = \bullet \\ 1 + \sum_i m_i |t_i| & \text{otherwise} \end{cases} \tag{4.1}$$

- Density:

$$\gamma(t) = \begin{cases} 1 & \text{if } t = \bullet \\ |t| \prod_{i=1}^{k} \gamma(t_i)^{m_i} & \text{otherwise} \end{cases} \tag{4.2}$$

- Symmetry:

$$\sigma(t) = \begin{cases} 1 & \text{if } t = \bullet \\ \prod_{i=1}^{k} m_i!\ \sigma(t_i)^{m_i} & \text{otherwise} \end{cases} \tag{4.3}$$

4.2.3 B-series

We can now rewrite Taylor expansions using the mappings from rooted trees to elementary differentials. For this, we first denote by T the set of rooted trees and $T^{\#}$ the set of rooted trees including the empty tree $\varnothing$ such that $T^{\#} = T \cup \{\varnothing\}$. We further define B^* as the set of all mappings from $T^{\#}$ to $\mathbb{R}$, B^0 the subset of B^* such that if $a \in \mathsf{B}^0$ then $a(\varnothing) = 0$ and B the subset of B^* such that if $a \in \mathsf{B}$ then $a(\varnothing) = 1$. The new expansion is called a B-series and is defined as follows:

Definition 4.2.2 (B-series)**.** The B-series $B(a, y_n)$ where $a \in \mathsf{B}^*$ and $y_n \in \mathbb{R}^n$ is defined as:

$$B(a, y_n) = a(\varnothing) y_n + \sum_{t \in T} a(t) \frac{h^{|t|}}{\sigma(t)} F(t)(y_n) \tag{4.4}$$

Many expressions can be expressed as a B-series. For example, the right hand side of eq. (1.1) at y_n can be expressed as: $hf(y_n) = B(d, y_n)$ where

$$d(t) = \begin{cases} 1 & \text{if } t = \bullet \\ 0 & \text{otherwise} \end{cases} \tag{4.5}$$

The following theorem presents another important example:

Theorem 4.2.2. *Let $y(t_n + ch) = B(a, y_n)$. Then,*

$$a(t) = \begin{cases} 1 & \text{if } t = \varnothing \\ \frac{c^{|t|}}{\gamma(t)} & \text{if } t \in T \end{cases}$$

Proof. replace $h = ch$ in exact solution. □

Theorem 4.2.3. *The exact solution of eq.* (1.1) *at* $t = t_n + \theta h$ *can be expressed as a B-series such that* $y(t_n + \theta h) = B(e_\theta, y_n)$ *where:*

$$e_\theta(\varnothing) = 1,$$
$$e_\theta(\tau) = \frac{\theta^{|\tau|}}{\gamma(\tau)}, \quad \tau \in T$$

Proof. See (J. C. Butcher, 2021, Section 3.4, p. 115) □

Several operations can also be applied to manipulate B-series. From the above definition, it is easy show that B-series are linear in the coefficients meaning that: $\alpha B(a, y_n) + \beta B(b, y_n) = B(c, y_n)$ with $c(t) = \alpha a(t) + \beta b(t)\ \forall t \in T^{\#}$. A less obvious property is the composition of B-series. If, in a B-series, y_n is replaced by another B-series the result is again a B-series: $B(b, B(a, y_n)) = B(a \circ b, y_n)$. The coefficients $a \circ b$ can be expressed in terms of the coefficients a and b as well as some properties of the trees. The general formula for these coefficients is complex and we refer the reader to (J. C. Butcher, 2021, Section 3.9) for more details. An interesting case of the composition rule is when $B(b, y_n) = hf(y_n)$.

Theorem 4.2.4. *Let* $Z = B(z, y_n)$ *such that* $z(\varnothing) = 1$. *Then* $hf(Z)$ *can be expressed as a B-series such that* $hf(Z) = B(a_f, y_n)$ *where*

$$a_f(t) = \begin{cases} 0 & \text{if } t = \varnothing \\ 1 & \text{if } t = \bullet \\ \prod_{i=1}^{m} z(t_i) & \text{if } t = [t_1\ t_2\ \ldots t_m] \end{cases}$$

Proof. See (J. C. Butcher, 2021, Theorem 3.4E) □

Using the notations introduced in this section, the classical order conditions can be expressed as follows:

Theorem 4.2.5. *Consider the numerical method expressed as a B-series* $y_{n+1} = B(\phi, y_n)$. *This method has classical order* p *if and only if*

$$\phi(t) = \frac{1}{\gamma(t)} \quad \text{for all trees such that } |t| \leq p$$

4.3 φ-order conditions

In this section, we introduce a new set of order conditions we call φ-order conditions. These order conditions will be motivated by a series expansion of the exact solution based on the φ_k functions.

First, we introduce the set T_D of all the trees with exactly one branch leaving from the root and $T_S = T \setminus T_D$ the set containing the tree with a single node ($\bullet$) and all trees with at least 2 branches leaving from the root. The set T can therefore be partitioned in 2 disjoint subsets: $T = T_D \cup T_S$.

We can note that any tree $t \in T_D$ as a unique representation as $t = [\tau]_k$ with $\tau \in T_S$. For example, the tree $t =$  $\in T_D$ can be expressed as $t =$ and  $\in T_S$.

Instead of expending the exact solution $y(t_n + h)$ using a Taylor series, we note that this expression can be written in terms of the φ functions.

Theorem 4.3.1. *Let y be the solution of the initial value problem eq.* (1.1) *with initial condition $y(t_n) = y_n$. The solution at time $t_n + h$ can be expressed as:*

$$y(t_n + h) = y_n + \sum_{k=1}^{\infty} \varphi_k(hJ)\Lambda_k \tag{4.6}$$

where $\Lambda_k = B(\lambda_k, y_n)$ and

$$\lambda_k(t) = \begin{cases} \frac{k!}{\gamma(t)} & \text{if } t \in T_S \text{ and } |t| = k \\ 0 & \text{otherwise} \end{cases} \tag{4.7}$$

Proof. Using the definitions of the B-series and λ_k, we have:

$$\begin{aligned} \varphi_k(hJ)\Lambda_k &= \varphi_k(hJ)\left(\lambda_k(\varnothing)y_n + \sum_{t\in T} \lambda_k(t)\frac{h^{|t|}}{\sigma(t)}F(t)(y_n)\right) \\ &= \varphi_k(hJ)\left(\sum_{\substack{t\in T_S \\ |t|=k}} \frac{k!}{\gamma(t)}\frac{h^k}{\sigma(t)}F(t)(y_n)\right) \end{aligned}$$

We can now apply the definition of the φ_k functions (eq. (A.4a)).

$$\varphi_k(hJ)\Lambda_k = \left(\sum_{i=0}^{\infty} \frac{(hJ)^i}{(k+i)!}\right)\left(\sum_{\substack{t\in T_S \\ |t|=k}} \frac{k!}{\gamma(t)}\frac{h^k}{\sigma(t)}F(t)(y_n)\right)$$
$$= \sum_{i=0}^{\infty}\sum_{\substack{t\in T_S \\ |t|=k}} \frac{k!}{(k+i)!}\frac{h^k}{\gamma(t)\sigma(t)}(hJ)^i F(t)(y_n)$$

We also know from theorem 4.2.1 that $J^i F(t)(y_n) = F([t]_i)(y_n)$. Therefore:

$$\varphi_k(hJ)\Lambda_k = \sum_{\substack{t\in T_S \\ |t|=k}} \left[\frac{h^k}{\gamma(t)\sigma(t)}F(t)(y_n) + \sum_{i=1}^{\infty}\frac{k!}{(k+i)!}\frac{h^{k+i}}{\gamma(t)\sigma(t)}F([t]_i)(y_n)\right]$$

Based on the properties of trees defined in section 4.2.2, we have that $|[t]_i| = |t| + i$, $\gamma([t]_i) = \frac{(|t|+i)!}{|t|!}\gamma(t)$ and $\sigma([t]_i) = \sigma(t)$. The expression simplifies to:

$$\varphi_k(hJ)\Lambda_k = \sum_{\substack{t\in T_S \\ |t|=k}} \left[\frac{h^{|t|}}{\gamma(t)\sigma(t)}F(t)(y_n) + \sum_{i=1}^{\infty}\frac{h^{|[t]_i|}}{\gamma([t]_i)\sigma([t]_i)}F([t]_i)(y_n)\right]$$

Because every tree $t \in T_D$ can be uniquely written as $t = [\tau]_i$ with $\tau \in T_S$, the set T_D can be defined as $T_D = \{\ [t]_k,\ t \in T_S \text{ and } k \geq 1\}$. Summing over all k, we get:

$$\sum_{k=1}^{\infty}\varphi_k(hJ)\Lambda_k = \sum_{t\in T_S}\frac{h^{|t|}}{\gamma(t)\sigma(t)}F(t)(y_n) + \sum_{t\in T_D}\frac{h^{|t|}}{\gamma(t)\sigma(t)}F(t)(y_n)$$
$$= \sum_{t\in T}\frac{h^{|t|}}{\gamma(t)\sigma(t)}F(t)(y_n)$$

Finally, we have:

$$y_n + \sum_{k=1}^{\infty}\varphi_k(hJ)\Lambda_k = y_n + \sum_{t\in T}\frac{h^{|t|}}{\gamma(t)\sigma(t)}F(t)(y_n)$$
$$= B(e, y_n)$$

where the B-series $B(e, y_n)$ is equal to the exact solution as defined in theorem 4.2.3 with $\theta = 1$. □

Note that in the expansion of the exact solution in theorem 4.3.1, the vectors Λ_1 and Λ_2 can be expressed simply. For $k = 1$, the only tree of order 1 in T_S is

the tree $t = \bullet$ and $F(\bullet)(y_n) = f(y_n)$ so $\Lambda_1 = hf(y_n)$. Moreover, for $k = 2$, there is a single tree of order 2 ($\left[\bullet\right]$) but this tree is T_D so $\Lambda_2 = 0$. The exact solution can then be expressed as

$$y(t_n + h) = y_n + \varphi_1(hJ)hf(y_n) + \sum_{k=3}^{\infty} \varphi_k(hJ)\Lambda_k \tag{4.8}$$

where Λ_k is defined as in theorem 4.3.1.

When using the classical order conditions, the exact solution is expanded as a single B-series and the structure in eq. (4.8) is lost. Here, we want to keep this structure and we will approximate each Λ_k B-series individually up to some order p. However, we can note that for $k > p, \Lambda_k = O(h^{p+1})$ and it can be approximated with the zero vector. Therefore, we consider the following truncated series when we seek a numerical method to approximate the exact solution:

$$y_{n+1} = y_n + \varphi_1(hJ)hf(y_n) + \sum_{k=3}^{p} \varphi_k(hJ)v_k \tag{4.9}$$

where each of the vectors v_k can be expressed as a B-series $v_k = B(\nu_k, y_n)$ and are approximation of the vector Λ_k. We now define the φ-order condition for methods following this ansatz.

Definition 4.3.1 (φ-order)**.** Consider a method of the form

$$y_{n+1} = y_n + \varphi_1(hJ)hf(y_n) + \sum_{k=3}^{q} \varphi_k(hJ)v_k$$

where $v_k = B(\nu_k, y_n)$ for $3 \leq k \leq q$ and $v_k = 0$ for $k > q$.

This method is said to have φ-order p if for all $k \geq 3$

$$\begin{cases} \nu_k(\varnothing) = \lambda_k(\varnothing) \\ \nu_k(t) = \lambda_k(t) \qquad \text{for all } t \in T \text{ such that } |t| \leq p \end{cases}$$

Note that a method cannot have φ-order p if $q < p$ because we would have $v_p = 0$ and some conditions could not be satisfied. In practice, we will usually consider the case $q = p$ as the vectors $v_k = 0$ for $k > p$ automatically meet the conditions.

Looking at the definition of λ_k from eq. (4.7), we notice that for all $k \geq 3$, $\lambda_k(\varnothing) = 0$, $\lambda_k(\bullet) = 0$ and $\lambda_k(t) = 0$ for all $t \in T_D$. Therefore, if we want our

method to have φ-order p, we need to make sure that at least $\nu_k(\varnothing) = \nu_k(\bullet) = \nu_k(t) = 0$ for all $t \in T_D$ and $3 \leq k \leq q$. The following theorem provides an easy way to enforce these constraints.

Theorem 4.3.2. *Let $r(z) = f(z) - f(y_n) - J(z - y_n)$ where f is the right hand side of eq. (1.1) and $J = f'(y_n)$ is the Jacobian matrix of f at y_n. For any $Z = B(z, y_n)$ such that $z(\varnothing) = 1$, $hr(Z)$ can be expressed as a B-series $B(a_r, y_n)$ with coefficients*

$$a_r(t) = \begin{cases} \prod_{i=1}^{m} a_z(t_i) & \text{if } t \in T_S \setminus \{\bullet\} \text{ and } t = [t_1, \ldots, t_m] \\ 0 & \text{otherwise} \end{cases} \tag{4.10}$$

Proof. Let $Z = B(z, y_n)$ such that $z(\varnothing) = 1$. From theorem 4.2.4, we know that $hf(z) = B(a_f, y_n)$ with:

$$a_f(t) = \begin{cases} 0 & \text{if } t = \varnothing \\ 1 & \text{if } t = \bullet \\ \prod_{i=1}^{m} z(t_i) & \text{if } t = [t_1, \ldots, t_m] \end{cases}$$

Thus $hf(z) - hf(y_n) = B(a_{df}, y_n)$ such that:

$$a_{df}(t) = \begin{cases} 0 & \text{if } t \in \{\varnothing, \bullet\} \\ \prod_{i=1}^{m} a_z(t_i) & \text{if } t = [t_1, \ldots, t_m] \end{cases}$$

Moreover, we have $w = z - y_n = B(a_w, y_n)$ such that $a_w(\varnothing) = 0$ and $a_w(t) = a_z(t)$ for all $t \in T$, then using theorem 4.2.1 we have $hJw = B(a_J, y_n)$ with:

$$a_J(t) = \begin{cases} a_z(\tau) & \text{if } t = [\tau] \in T_D \\ 0 & \text{otherwise} \end{cases}$$

We further note that, $a_{df}([\tau]) = a_z(\tau)$ for $t \in T$.

Finally, $hr(Z)$ can be expressed as the difference $hr(Z) = B(a_r, y_n) = B(a_{df}, y_n) - B(a_J, y_n)$, and we get:

$$a_r(t) = \begin{cases} 0 & \text{if } t \in \{\varnothing, \bullet\} \\ 0 & \text{if } t \in T_D \\ \prod_{i=1}^{m} a_z(t_i) & \text{otherwise with } t = [t_1, \ldots, t_m] \end{cases}$$

□

From this theorem, we can conclude that if v_k is computed as $v_k = hr(Z)$ with $Z = B(z, y_n), z \in \mathsf{B}$ or if v_k is a linear combination of such vectors then the conditions $\nu_k(\varnothing) = 0$, $\nu_k(\bullet) = 0$ and $\nu_k(t) = 0$ for all $t \in T_D$ and $3 \leq k \leq q$ are automatically satisfied. This motivates the choice of the vectors v_k in the ansatz eq. (4.9) to have the following form:

$$v_k = \sum_{i=1}^{m} \alpha_{k,i}\ hr(Z_i)$$

with $Z_i = B(z_i, y_n), z_i \in \mathsf{B}$. The φ-order conditions are then reduced to the following:

Theorem 4.3.3. *Consider a numerical method following the ansatz eq.* (4.9)*:*

$$y_{n+1} = y_n + \varphi_1(hJ)hf(y_n) + \sum_{k=3}^{q} \varphi_k(hJ)v_k$$

with $v_k = B(\nu_k, y_n) = \sum_{i=1}^{m} \alpha_{k,i}\ hr(Z_i)$ *and* $Z_i = B(z_i, y_n), z_i \in \mathsf{B}$*. This method satisfies the* φ*-order conditions of order* p *if and only if, for all* $3 \leq k \leq q$*:*

$$\nu_k(t) = \begin{cases} \frac{k!}{\gamma(t)} & \text{if } k = |t| \\ 0 & \text{if } k \neq |t| \end{cases} \quad \text{for all } t \in T_S \text{ such that } |t| \leq p \tag{4.11}$$

For example, for a method to have φ-order 4, we have the following six conditions:

$$\nu_3(\tau(\tau)) = 2 \qquad \nu_3(\tau(\tau\tau)) = 0 \qquad \nu_3(\tau^2(\tau)) = 0$$
$$\nu_4(\tau(\tau)) = 0 \qquad \nu_4(\tau(\tau\tau)) = 6 \qquad \nu_4(\tau^2(\tau)) = 3$$

As mentioned above, we want to compute the vectors v_k using linear combinations of the function r evaluated at different vectors Z_i. To minimize the computational cost of the final method, we want to minimize the number of such evaluations. However, because for each tree $t \in T_S$, $\nu_k(t) \neq 0$ only for $k = |t|$, the mappings ν_k must be linearly independent. Based on this conclusion, the corresponding vectors $v_k = B(\nu_k, y_n)$ must also be linearly independent. We will therefore compute the vectors v_k using $m = p-2$ evaluations of the function r. We denote these evaluations $W_i = hr(Z_i)$ and the corresponding B-series mappings w_i and z_i such that $W_i = B(w_i, y_n)$ and $Z_i = B(z_i, y_n)$.

Using these notations, we expressed the vectors v_k as follows:

$$v_k = \sum_{i=1}^{m} \alpha_{k,i} W_i \tag{4.12}$$

or if we want to express it in terms of the corresponding B-series mappings:

$$\nu_k(t) = \sum_{i=1}^{m} \alpha_{k,i} w_i(t) \tag{4.13}$$

The ansatz eq. (4.9) then becomes:

$$y_{n+1} = y_n + \varphi_1(hJ)hf(y_n) + \sum_{k=3}^{q} \varphi_k(hJ) \sum_{i=1}^{m} \alpha_{k,i}\ hr(Z_i) \tag{4.14}$$

4.4 Solution of φ-order conditions

In the previous section, we introduced the φ-order conditions and motivated the ansatz in eq. (4.14) for exponential methods. However, in order to derive a specific method, we need to solve the system of φ-order conditions. The unknowns of this system are the scalar $\alpha_{k,i}$ and the vectors Z_i. The goal of this section is to prove the following theorem:

Theorem 4.4.1. *Consider a method of the form*

$$y_{n+1} = y_n + \varphi_1(hJ)hf(y_n) + \sum_{k=3}^{q} \varphi_k(hJ) \left(\sum_{i=1}^{m} \alpha_{k,i} hr(Z_i) \right)$$

approximating the solution y of eq. (1.1) and let c_i for $1 \leq i \leq m$ be some scalars such that $c_i \neq 0$ and $c_i \neq c_j$ if $i \neq j$.

This method has φ-order p if and only if

$$\alpha_{k,i} = (-1)^{m-k}(k-1)!\ \frac{e_{p-k}(\{c_1, \ldots, c_m\} \setminus \{c_i\})}{{c_i}^2 \prod_{\substack{l=1 \\ l \neq i}}^{m} (c_i - c_l)} \tag{4.15}$$

and the vectors Z_i are approximations of order $p-2$ (in the classical sense) of $y(t_n + c_i h)$.

Proof. First, we note that eqs. (4.11) and (4.13) can be combined to form the system of φ-order conditions:

$$\sum_{i=1}^{m} \alpha_{k,i} w_i(t) = \begin{cases} \frac{k!}{\gamma(t)} & \text{if } |t| = k \\ 0 & \text{if } |t| \neq k \end{cases} \qquad \text{for all } t \in T_S \text{ such that } |t| \leq p \tag{4.16}$$

Because we want to apply the function r to the vectors Z_i, we need $z_i \in \mathsf{B}$ and therefore $z_i(\varnothing) = 1$. Then we define $c_i = z_i(\bullet)$ and the tree $\psi_j = [\tau^j]$ with $\tau = \bullet$. For example $\psi_3 =$  . Moreover, applying the definitions of section 4.2.2, we have $|\psi_j| = j + 1$ and $\gamma(\psi_j) = |\psi_j| = j + 1$. Finally, using theorem 4.2.4, we have $w_i(\psi_j) = z_i(\bullet)^j = {c_i}^j$.

Applying these expressions to our running example, we get the following conditions:

$$\begin{aligned} \alpha_{3,1}c_1^2 + \alpha_{3,2}c_2^2 &= 2 & \qquad \alpha_{4,1}c_1^2 + \alpha_{4,2}c_2^2 &= 0 \\ \alpha_{3,1}c_1^3 + \alpha_{3,2}c_2^3 &= 0 & \qquad \alpha_{4,1}c_1^3 + \alpha_{4,2}c_2^3 &= 6 \\ \alpha_{3,1}w_1(\text{  }) + \alpha_{3,2}w_2(\text{}) &= 0 & \qquad \alpha_{4,1}w_1(\text{}) + \alpha_{4,2}w_2(\text{}) &= 3 \end{aligned}$$

With this example, it is easy to show that this system of equations simplifies to:

$$\begin{cases} \alpha_{3,1} = \dfrac{2c_2}{c_1^2(c_2 - c_1)} \\ \alpha_{3,2} = \dfrac{2c_1}{c_2^2(c_1 - c_2)} \\ \alpha_{4,1} = -\dfrac{6}{c_1^2(c_2 - c_1)} \\ \alpha_{4,2} = -\dfrac{6}{c_2^2(c_1 - c_2)} \\ w_1(\text{}) = \dfrac{c_1}{2} \\ w_2(\text{}) = \dfrac{c_2}{2} \end{cases}$$

However, as we increase the order that we want to achieve, the number and complexity of these equations are increasing rapidly. We will now present a systematic way for solving these order conditions for an arbitrary order p.

First, we want to express the coefficient $\alpha_{k,i}$ in terms of c_i. For this, we consider the order conditions corresponding to the trees ψ_{j+1} introduced above for $j = \{1, \dots, m\}$ (corresponding to trees of order 3 to p).

Looking at the right-hand side of equation (4.16) with $t = \psi_{j+1}$, we have for $1 \leq i \leq m$ and $1 \leq j \leq m$:

$$\nu_{i+2}(\psi_{j+1}) = \begin{cases} \frac{(i+2)!}{\gamma(\psi_{j+1})} & \text{if } |\psi_{j+1}| = i+2 \\ 0 & \text{if } |\psi_{j+1}| \neq i+2 \end{cases}$$
$$= \begin{cases} (i+1)! & \text{if } i = j \\ 0 & \text{if } i \neq j \end{cases}$$

For all the values of i and j, we can collect the coefficient in a matrix $B \in \mathbb{R}^{m \times m}$ with components $B_{i,j} = \nu_{i+2}(\psi_{j+1}) = (i+1)!\ \delta_{i,j}$.

Now looking at the left-hand side of equation (4.16) with $t = \psi_j$, we have for $1 \leq i \leq m$ and $1 \leq j \leq m$:

$$\nu_{i+2}(\psi_{j+1}) = \sum_{n=1}^{m} \alpha_{i+2,n}\ w_n(\psi_{j+1})$$
$$= \sum_{n=1}^{m} \alpha_{i+2,n}\ {c_n}^{j+1} = \sum_{n=1}^{m} \alpha_{i+2,n}\ {c_n}^2\ {c_n}^{j-1}$$

For all the values of i and j, this can be express as the product of three matrices A, C^2 and V of size $m \times m$ with $A_{i,j} = \alpha_{i+2,j}$, $C_{i,j} = c_i\ \delta_{i,j}$ and $V_{i,j} = {c_i}^{j-1}$.

After combining both sides of the equation (4.16), we get the matrix equation $AC^2V = B$ or written more explicitly:

$$\begin{pmatrix} \alpha_{3,1} & \dots & \alpha_{3,m} \\ & \ddots & \\ \alpha_{p,1} & \dots & \alpha_{p,m} \end{pmatrix} \begin{pmatrix} {c_1}^2 & & \\ & \ddots & \\ & & c_m^2 \end{pmatrix} \begin{pmatrix} 1 & c_1 & \dots & {c_1}^{m-1} \\ 1 & c_2 & \dots & {c_2}^{m-1} \\ & & \vdots & \\ 1 & c_m & \dots & {c_m}^{m-1} \end{pmatrix} = \begin{pmatrix} 2! & & \\ & \ddots & \\ & & (m+1)! \end{pmatrix}$$

We can note that the matrix V is a Vandermonde matrix and therefore analytical formulas are available for its determinant and inverse. We want to solve the equation $AC^2V = B$ for the variable A. This equation will have a unique solution

if and only if $\det(C^2V) = \det(C)^2 \det(V) \neq 0$. We have:

$$\det(V) = \prod_{\substack{i,j=1 \\ i>j}}^{m} (c_i - c_j) \qquad\qquad \det(C) = \prod_{i=1}^{m} c_i$$

Therefore, we will have a unique solution if and only if all the c_i are distinct and non-zero. The solution will then be $A = BV^{-1}C^{-2}$ with:

$$(C^{-1})_{i,j} = \frac{1}{c_i}\ \delta_{i,j}$$

$$(V^{-1})_{i,j} = \frac{(-1)^{m-i}\ e_{m-i}(\{c_1, \ldots, c_m\} \setminus \{c_j\})}{\prod_{\substack{l=1 \\ l\neq j}}^{m}(c_j - c_l)}$$

where $e_k(S)$ is the k^{th} elementary symmetric function or the sum of all distinct products of k elements of the set S.

Finally, we have

$$\begin{aligned}
\alpha_{k,i} &= A_{k-2,i} \\
&= \sum_{j,l=1}^{m} B_{k-2,j}(V^{-1})_{j,l}(C^{-2})_{l,i} \\
&= \sum_{j,l=1}^{m} (k-1)!\ \delta_{k-2.j} \frac{(-1)^{m-j}\ e_{m-j}(\{c_1, \ldots, c_m\} \setminus \{c_l\})}{\prod_{\substack{n=1 \\ n\neq l}}^{m}(c_l - c_n)} \frac{1}{{c_l}^2}\ \delta_{l,i} \\
&= (-1)^{m-k}(k-1)!\ \frac{e_{p-k}(\{c_1, \ldots, c_m\} \setminus \{c_i\})}{{c_i}^2 \prod_{\substack{l=1 \\ l\neq i}}^{m}(c_i - c_l)}
\end{aligned}$$

We now want to find the conditions on the vectors Z_i. First, by looking at the equation (4.16), we see that the system will involve the unknown $w(t)$ for all $t \in T_S$ such that $|t| \leq p$. In turn, considering equation (4.10), we can conclude that the system will involve the unknown $z_i(\tau)$ for all $\tau \in T$ such that $|\tau| \leq p-2$. In fact, the coefficients $z_i(\tau)$ for trees of order higher than $p-2$ can only appear with coefficient $w(t)$ with $|t| \geq p$ because it would involve a tree of the form $t = [\tau\ t_2 \ldots t_n]$ with $\sum_{i=2}^{n} |t_i| \geq 1$ so $|t| \geq p$. We already know that $z_i(\varnothing) = 1$ and

$z_i(\bullet) = c_i$. We would like to be able to express the remaining unknown $z_i(t)$ for $t \in T$ such that $2 \leq |t| \leq p-2$ in terms of the values c_i using some of the order conditions. For this, we define the trees $\zeta(\tau) = [\tau\ \bullet]$. For these trees, we have the following properties: $|\zeta(\tau)| = |\tau| + 2$, and $\gamma(\zeta(\tau)) = (|\tau| + 2)\ \gamma(\tau)$

Looking at the right-hand side of equation (4.16) with $t = \zeta(\tau)$, we have for $1 \leq k \leq m$:

$$\sum_{n=1}^{m} \alpha_{i+2,n}\ w_n(\zeta(\tau)) = \begin{cases} \frac{(i+2)!}{(|\tau|+2)\ \gamma(\tau)} & \text{if } i+2 = |\tau|+2 \\ 0 & \text{if } i+2 \neq |\tau|+2 \end{cases}$$
$$= \begin{cases} \frac{(i+1)!}{\gamma(\tau)} & \text{if } i = |\tau| \\ 0 & \text{if } i \neq |\tau| \end{cases}$$

For all the values of i, we can collect the coefficient in a vector $b \in \mathbb{R}^m$ with components $b_i = \frac{(i+1)!}{\gamma(\tau)}\ \delta_{i,|\tau|}$.

Now looking at the left-hand side of equation (4.16) with $t = \zeta(\tau)$, we have for $1 \leq i \leq m$:

$$\sum_{n=1}^{m} \alpha_{i+2,n}\ w_n(\zeta(\tau)) = \sum_{n=1}^{m} \alpha_{i+2,n}\ c_n\ z_n(\tau)$$

This can be express as the matrix-product ACz with $A_{i,j} = \alpha_{i+2,j}$, $C_{i,j} = c_i\ \delta_{i,j}$ and $z_i = z_i(\tau)$.

After combining both sides of the equation (4.16), we get the following system of linear equations $ACz = b$. When solving for the coefficients $\alpha_{k,i}$, we already showed that $A = BV^{-1}C^{-2}$, $\det(V) \neq 0$ and $\det(C) \neq 0$. We also have $\det(B) = \prod_{i=1}^{m}(i+1)! \neq 0$. Therefore, $\det(AC) = \frac{\det(B)}{\det(V)\det(C)} \neq 0$ and the system $ACz = b$ has a unique solution. A simple way to solve this system is to compare the order conditions for the trees $t = \zeta(\tau)$ and $t = \psi_{|\tau|+1}$. For $t = \zeta(\tau)$, we have:

$$\sum_{n=1}^{m} \alpha_{i+2,n}\ c_n\ z_n(\tau)\gamma(\tau) = \begin{cases} (i+1)! & \text{if } i = |\tau| \\ 0 & \text{if } i \neq |\tau| \end{cases}$$

and for $t = \psi_{|\tau|+1}$, we have:

$$\sum_{n=1}^{m} \alpha_{i+2,n}\ {c_n}^{|\tau|+1} = \begin{cases} (i+1)! & \text{if } i = |\tau| \\ 0 & \text{if } i \neq |\tau| \end{cases}$$

After taking the difference of these two equations, we get:

$$\sum_{n=1}^{m} \alpha_{i+2,n} \ ({c_n}^{|\tau|+1} - c_n z_n(\tau)\gamma(\tau)) = 0$$

One possible solution for this equation is to take ${c_n}^{|\tau|+1} - c_n z_n(\tau)\gamma(\tau) = 0$ for all $n \in 1, \ldots, m$ which implies that $z_i(\tau) = \frac{{c_i}^{|\tau|}}{\gamma(\tau)}$. The solution is unique, therefore, this solution is the unique solution and we have the following constraints on the coefficients of z_i:

$$\begin{cases} z_i(\varnothing) = 1 \\ z_i(\bullet) = c_i \\ z_i(t) = \frac{{c_i}^{|t|}}{\gamma(t)} \quad \forall t \in T \text{ such that } |t| \leq p-2 \end{cases} \tag{4.17}$$

A consequence of theorem 4.2.3 is that the vectors $Z_i = B(z_i, y_n)$ are order $p-2$(classical) approximations of the vectors $y(t_n + c_i h)$.

We proved that considering the order conditions corresponding to the trees ψ_{j+1} for $1 \leq j \leq m$ we have a unique solution for the coefficients $\alpha_{k,i}$ such that:

$$\alpha_{k,i} = (-1)^{m-k}(k-1)! \ \frac{e_{p-k}(\{c_1, \ldots, c_m\} \setminus \{c_i\})}{{c_i}^2 \prod_{\substack{l=1 \\ l \neq i}}^{m} (c_i - c_l)}$$

Then, using the order conditions associated to the trees of the form $\zeta(t)$ for all tree t such that $|t| \leq p-2$, we get the following extra conditions on the coefficients $z_i(\tau)$:

$$\begin{cases} z_i(\varnothing) = 1 \\ z_i(t) = \frac{{c_i}^{|t|}}{\gamma(t)} \quad \forall t \in T \text{ such that } |t| \leq p-2 \end{cases}$$

These two sets of equations gave us conditions on all the unknowns of the system of order conditions. We are now verifying that the other equations that we haven't considered so far are automatically satisfied with the conditions presented above.

From equation (4.16), we have:

$$\sum_{i=1}^{m} \alpha_{k,i} w_i(t) = \begin{cases} \frac{k!}{\gamma(t)} & \text{if } |t| = k \\ 0 & \text{if } |t| \neq k \end{cases} \quad \text{for all } t \in T_S \text{ such that } |t| \leq p \tag{4.18}$$

Moreover, using the conditions on the $z_i(t)$ coefficients with equation (4.10), we get that for all trees $t = [t_1, \ldots, t_n]$ with $3 \leq |t| \leq p$:

$$\begin{aligned} w_i(t) &= \prod_{j=0}^{n} \frac{c_i^{|t_j|}}{\gamma(t_j)} \\ &= \frac{c_i^{|t|-1}|t|}{\gamma(t)} \end{aligned}$$

Therefore,

$$\sum_{i=1}^{m} \alpha_{k,i} c_i^{|t|-1} = \begin{cases} (k-1)! & \text{if } k = |t| \\ 0 & \text{if } k \neq |t| \end{cases} \quad \text{for all } t \in T_S \text{ such that } |t| \leq p$$

□

4.5 Properties of schemes

4.5.1 Comparison to classical order conditions and extra conditions

In addition to greatly reducing the number of order conditions, the choice we made to compute the vector v_k as linear combinations of evaluations of the function hr also enforces the following property.

Theorem 4.5.1. *Let $y_{n+1} = B(\phi, y_n)$ a numerical method of the form eq. (4.14).*

$$y_{n+1} = y_n + \varphi_1(hJ)hf(y_n) + \sum_{k=3}^{q} \varphi_k(hJ)v_k$$

where the vectors v_k are chosen such that $v_k = \sum_{i=1}^{m} \alpha_{k,i} hr(Z_i)$.

This method has φ-order p if and only if

$$\begin{cases} \phi(t) = e(t) \\ \phi([t]_k) = e([t]_k) \quad \forall\ k > 1 \end{cases} \quad \text{for all } t \in T \text{ such that } |t| \leq p$$

where the mapping e correspond to the coefficients of the B-series expansion of the exact solution as defined in theorem 4.2.3 with $\theta = 1$.

Proof. Let $v_1 = hf(y_n) = B(\nu_1, y_n)$ where the mapping $\nu_1 = d$ is defined in eq. (4.5) and $v_2 = 0 = B(\nu_2, y_n)$ where $\nu_2(t) = 0 \ \forall \ T \in T^{\#}$. We can rewrite the method as:

$$y_{n+1} = y_n + \sum_{k=1}^{q} \varphi_k(hJ)v_k$$

Moreover, for all $1 \leq k \leq q$, $\nu_k(t) = 0 \ \forall \ t \in T_D$ so we have:

$$v_k = \sum_{t \in T_S} \nu_k \frac{h^{|t|}}{\sigma(t)} F(t)(y_n)$$

Using eq. (A.4a), we get:

$$\begin{aligned}
\varphi_k(hJ)v_k &= \sum_{i=0}^{\infty} \frac{(hJ)^i}{(i+k)!} \sum_{t \in T_S} \nu_k \frac{h^{|t|}}{\sigma(t)} F(t)(y_n) \\
&= \sum_{t \in T_S} \frac{\nu_k}{k!} \frac{h^{|t|}}{\sigma(t)} F(t)(y_n) + \sum_{i=1}^{\infty} \sum_{t \in T_S} \frac{\nu_k}{(i+k)!} \frac{h^{|t|+i}}{\sigma(t)} F([t]_i)(y_n) \\
&= \sum_{t \in T_S} \frac{\nu_k}{k!} \frac{h^{|t|}}{\sigma(t)} F(t)(y_n) + \sum_{i=1}^{\infty} \sum_{t \in T_S} \frac{\nu_k}{(i+k)!} \frac{h^{|[t]_i|}}{\sigma([t]_i)} F([t]_i)(y_n)
\end{aligned}$$

We also know that $y_{n+1} = B(\phi, y_n)$ so using the previous equation, ϕ is defined as $\phi(\varnothing) = 1$ and

$$\phi(t) = \begin{cases} \sum_{k=1}^{q} \frac{\nu_k(t)}{k!} & \text{if } t \in T_S \\ \sum_{k=1}^{q} \frac{\nu_k(\tau)}{(i+k)!} & \text{if } t = [\tau]_i \in T_D \end{cases}$$

If we assume that y_{n+1} is a method with φ-order p then according to definition 4.3.1, $\nu_k(t) = \lambda_k(t)$ for all $t \in T$ such that $|t| \leq p$. Therefore if $t \in T_S$ such that $|t| \leq p$, $\phi(t) = \sum_{k=1}^{q} \frac{1}{\gamma(t)} \delta_{k,|t|} = \frac{1}{\gamma(t)}$ and if $t = [\tau]_i \in T_D$ such that $|\tau| \leq p$ and $i \geq 1$ then $\phi(t) = \sum_{k=1}^{q} \frac{k!}{(i+k)!\gamma(\tau)} \delta_{k,|\tau|} = \frac{|\tau|!}{(i+|\tau|)!\gamma(\tau)} = \frac{1}{\gamma(t)}$. Note that this is true because $q \geq p$ if the method has φ-order p as noted before. This proves that if the method has φ-order p then

$$\begin{cases} \phi(t) = e(t) \\ \phi([t]_k) = e([t]_k) \quad \forall \ k > 1 \end{cases} \qquad \text{for all } t \in T \text{ such that } |t| \leq p$$

To show the other direction, we assume that

$$\begin{cases} \phi(t) = e(t) \\ \phi([t]_k) = e([t]_k) \quad \forall \ k > 1 \end{cases} \qquad \text{for all } t \in T \text{ such that } |t| \leq p$$

Then, for $t \in T_S$ such that $|t| \leq p$, we have $\phi(t) = \sum_{k=1}^{q} \frac{\nu_k(t)}{k!} = \frac{1}{\gamma(t)}$ and $\phi([t]_i) = \sum_{k=1}^{q} \frac{\nu_k(t)}{(i+k)!} = \frac{1}{\gamma([t]_i)} = \frac{1}{\gamma(t)} \frac{|t|!}{(|t|+i)!}$. Thus,

$$\sum_{k=1}^{q} \nu_k \left(\frac{1}{k!} - \frac{(i+|t|)!}{|t|!(i+k)!} \right) = 0$$

The unique solution for this equation is $\nu_k(t) = 0 \ \forall \ k \neq |j|$. Therefore, $\phi(t) = \frac{\nu_{|t|}(t)}{|t|!} = \frac{1}{\gamma(t)}$ We also already know that $\nu_k(\varnothing) = 0$ and $\nu_k(t) = 0$ for $t \in T_D$ so,

$$\nu_k(t) = \begin{cases} \frac{k!}{\gamma(t)} & \text{if } k = |t| \\ 0 & \text{otherwise} \end{cases} \quad \text{for all } t \in T_S \text{ such that } |t| \leq p$$

This implies that the method has φ-order p. □

This theorem means that for a method with φ-order p and following the ansatz presented in eq. (4.14), not only the local error term for all the elementary differential terms up to order p is zero (classical order conditions) but it is also zero for all the elementary differential terms corresponding to trees of the form $[\tau]_k$ where $|\tau| \leq p$ and $k \geq 1$. For example, if such a method as φ-order 3, then it will have a zero local error term for the elementary differential term $F(\bullet\!\!\bullet)(y_n)$ but it will also have a zero local error term for $F(\bullet\!\!\bullet\!\!\bullet)(y_n)$, $F(\bullet\!\!\bullet\!\!\bullet\!\!\bullet)(y_n)$ and so on.

This property is of great interest for exponential methods as these methods are usually applied to stiff systems of differential equations using large time step. The stiffness of a systems of differential equations is usually characterized by large eigenvalues for the Jacobian matrix. In this context, terms of the form $J^k F(\tau)(y_n) = F([\tau]_k)(y_n)$ are usually large in magnitude as they can get amplified by the eigenvalues of the Jacobian matrix. In practice, this effect tends to reduce the accuracy of the method and limit the domain where the method is converging at the expected order. The methods matching the conditions of theorem 4.5.1 can circumvent this problem by eliminating all such terms from the error expansion and therefore help the accuracy of the method.

4.5.2 Embedded lower order methods

Theorem 4.5.2. *Consider a numerical method following the ansatz of eq.* (4.9) *with φ-order p:*

$$y_{n+1} = y_n + \varphi_1(hJ)hf(y_n) + \sum_{k=3}^{q} \varphi_k(hJ)v_k$$

Then the truncated method with $\tilde{q} < q$:

$$y_{n+1} = y_n + \varphi_1(hJ)hf(y_n) + \sum_{k=3}^{\tilde{q}} \varphi_k(hJ)v_k$$

has order $min(p, \tilde{q})$.

Proof. Direct consequence of definition 4.3.1. □

This theorem can be useful for deriving methods with embedded error estimates. It is possible to compute the φ_k terms in two times and obtain one order p and one order $p-1$ approximation of the solution. These approximations can then be used to estimate the error and adapt the time-step of the method. Moreover, the formula to compute the coefficients $\alpha_{k,i}$ is very inexpensive and therefore, new coefficients can be computed at each step if necessary.

4.5.3 Single exponential projection

Using the results from appendix A.2, once the vectors v_k are available, it is possible to compute the linear combination $\varphi_1(hJ)hf(y_n) + \sum_{k=3}^{q} \varphi_k(hJ)v_k$ from eq. (4.9) using a single Krylov projection. This computation being the most expensive part of exponential schemes, it allows to minimize the computation time.

4.5.4 Dense output

The following theorem is an extension of theorem 4.3.1.

Theorem 4.5.3. *Let y be the solution of the initial value problem eq.* (1.1) *with initial condition $y(t_n) = y_n$. The solution at time $t_n + \theta h$ can be expressed as:*

$$y(t_n + \theta h) = y_n + \sum_{k=1}^{\infty} \varphi_k(\theta hJ)\theta^k \Lambda_k \tag{4.19}$$

where $\Lambda_k = B(\lambda_k, y_n)$ *and*

$$\lambda_k(t) = \begin{cases} \frac{k!}{\gamma(t)} & \text{if } t \in T_S \text{ and } |t| = k \\ 0 & \text{otherwise} \end{cases} \tag{4.20}$$

Proof. Using the same argument as in the proof of 4.3.1, we can get:

$$\begin{aligned} y_n + \sum_{k=1}^{\infty} \varphi_k(\theta h J)\theta^k \Lambda_k &= y_n + \sum_{t \in T} \frac{\theta^{|t|}}{\gamma(t)} \frac{h^{|t|}}{\sigma(t)} F(t)(y_n) \\ &= B(e_\theta, y_n) \end{aligned}$$

where the B-series $B(e_\theta, y_n)$ is equal to the exact solution as defined in theorem 4.2.3. □

Using this theorem and the definition of the φ-order, we can see that if a method using the ansatz of eq. (4.9):

$$y_{n+1} = y_n + \varphi_1(hJ)hf(y_n) + \sum_{k=3}^{q} \varphi_k(hJ)v_k,$$

is an approximation of the solution at time $t_n + h$ with φ-order p, then the methods

$$y_{n+1} = y_n + \varphi_1(\tau hJ)\tau hf(y_n) + \sum_{k=3}^{q} \varphi_k(\tau hJ)\tau^k v_k,$$

will be an approximation of the solution at time $t_n + \tau h$ with φ-order p. This expression therefore provides a way to get a full order dense output for any method following the ansatz of eq. (4.9). We can further notice that this expression coincides with the expression in theorem A.2.1. If the linear combination of φ functions is evaluated using the adaptive Krylov procedure, it is also possible to get the value for different values of τ along the way with no or little extra computation time. These values being full order approximations of the solution at the time $t_n + \tau h$, can be used for the vectors Z_i in the next time step as we will see in the next section.

4.6 Derivation of exponential schemes

In section 4.4, we showed that using the ansatz in eq. (4.14), the φ order conditions can be solved analytically. In theorem 4.4.1, the only constraint on

the vectors z_i is that they need to be an approximations of order $p-2$ of the exact solution at $t_n + c_i h$. By choosing the values of the nodes c_i, different classes of exponential integrators can be obtained. In this section, we will give a few examples of the methods that can be obtained by considering different values c_i.

4.6.1 Exponential Runge-Kutta

We first consider the class of exponential Runge-Kutta method. Methods of this type can be obtained by choosing the node values c_i such that $0 < c_i \leq 1$. The vectors z_i then correspond to the stages of the method at time $t_n + c_i h$.

We start with a method of order 3. Using the ansatz in eq. (4.14), the method will have the following form:

$$y_{n+1} = y_n + \varphi_1(hJ)hf(y_n) + \varphi_3(hJ)\ \alpha_{31}\ hr(z_1)$$

According to theorem 4.4.1, the values of α_{31} is given by $\alpha_{31} = \frac{2}{{c_1}^2}$ and z_1 is a first-order approximation of the exact solution. The value of z_1 can be obtained using the exponential Euler methods at time $t_n + c_1 h$. With these conditions, any methods with $0 < c \leq 1$ will have φ-order 3. However, by choosing $c = \frac{3}{4}$, the classical order of the method can be increased to 4. We then get the following method:

$$\begin{aligned} z_1 &= y_n + \varphi_1\left(\frac{3}{4}hJ\right)\frac{3}{4}hf(y_n) \\ y_{n+1} &= y_n + \varphi_1(hJ)hf(y_n) + \varphi_3(hJ)\frac{32}{9}hr(z_1) \end{aligned}$$

Similarly, if we want to derive a method with φ-order 4, we need the following conditions:

$$\begin{aligned} \alpha_{3,1} &= \frac{2c_2}{{c_1}^2(c_2 - c_1)} & \alpha_{3,2} &= \frac{2c_1}{{c_2}^2(c_1 - c_2)} \\ \alpha_{4,1} &= \frac{6}{{c_1}^2(c_1 - c_2)} & \alpha_{4,2} &= \frac{6}{{c_2}^2(c_2 - c_1)} \end{aligned}$$

By choosing $c_1 = \frac{1}{2}, c_2 = \frac{2}{3}$ and $c_1 = \frac{1}{8}, c_2 = \frac{1}{9}$ we rederive the methods from (Rainwater & Tokman, 2017). As for the third order method, the node values c_i can also be chosen carefully to minimize the error. For the order 4, this is done by

choosing c_i as the root of the polynomial $p(c) = 6 - 20x + 15x^2$ ($c_1 = \frac{10-\sqrt{10}}{15}$ and $c_2 = \frac{10+\sqrt{10}}{15}$). The method can then be expressed as:

$$\begin{aligned} z_1 &= y_n + \varphi_1\left(c_1 hJ\right) c_1 hf(y_n) \\ z_2 &= y_n + \varphi_1\left(c_2 hJ\right) c_2 hf(y_n) \\ y_{n+1} &= y_n + \varphi_1(hJ)hf(y_n) + \varphi_3(hJ)(\alpha_{3,1}hr(z_1) + \alpha_{3,2}hr(z_2)) \\ &\quad + \varphi_4(hJ)(\alpha_{4,1}hr(z_1) + \alpha_{4,2}hr(z_2)) \end{aligned}$$

where

$$\begin{pmatrix} \alpha_{3,1} & \alpha_{3,2} \\ \alpha_{4,1} & \alpha_{4,2} \end{pmatrix} = \begin{pmatrix} \frac{155+65\sqrt{10}}{18} & \frac{155-65\sqrt{10}}{18} \\ \frac{-100-55\sqrt{10}}{4} & \frac{-100+55\sqrt{10}}{4} \end{pmatrix} \tag{4.21}$$

If we want to derive a method with order higher than 4 then we cannot use the exponential Euler method anymore to compute the stage. However, to get a method of order p, it is possible to use a method of order $p-2$ to compute the stages. This can be done using the continuous output property of the methods presented in this chapter. For example, if we want to derivate a method of order 6, we can use the exponential Runge-Kutta presented above and evaluate it at the desired node. Below we give the coefficient α for a method of order 6 corresponding to the nodes $c_i = \frac{i}{4}$.

$$\alpha_{i+2,j} = \begin{pmatrix} 128 & -48 & \frac{128}{9} & -2 \\ -1664 & 912 & -\frac{896}{3} & 44 \\ 9216 & -6144 & \frac{7168}{3} & -384 \\ -20480 & 15360 & -\frac{20480}{3} & 1280 \end{pmatrix} \tag{4.22}$$

The full methods can be expressed as:

$$
\begin{aligned}
z_i &= y_n + \varphi_1(c_i hJ)\, c_i h f(y_n) && \text{for } i \in \{1,2\} \\
z_j &= y_n + \varphi_1(c_j hJ)\, c_j h f(y_n) + \varphi_3(c_j hJ)\, c_j^3 \sum_{k=1}^{2} \beta_{3,k} hr(z_k) \\
&\qquad + \varphi_4(c_j hJ)\, c_j^4 \sum_{k=1}^{2} \beta_{4,k} hr(w_k) && \text{for } j \in \{3,4,5,6\} \\
y_{n+1} = y_n + \varphi_1(hJ)\, hf(y_n) &\qquad + \varphi_3(hJ) \sum_{k=3}^{6} \alpha_{3,k} hr(z_k) + \varphi_4(hJ) \sum_{k=3}^{6} \alpha_{4,k} hr(z_k) \\
&\qquad + \varphi_5(hJ) \sum_{k=3}^{6} \alpha_{5,k} hr(z_k) + \varphi_6(hJ) \sum_{k=3}^{6} \alpha_{6,k} hr(z_k)
\end{aligned}
$$

with $c = [\frac{10-\sqrt{10}}{15}, \frac{10+\sqrt{10}}{15}, \frac{1}{4}, \frac{2}{4}, \frac{3}{4}, \frac{4}{4}]$, $\beta_{i,k}$ are the coefficient in eq. (4.21), and $\alpha_{i,k}$ are the coefficient in eq. (4.22).

4.6.2 Exponential multi-step

Exponential multi-step methods use the values of the previous iteration to approximate the value at the new step. In our framework, this corresponds to choosing the node c_i as negative itegers. For example, a method of order 3 using the value at the previous time step will give us: $c_1 = -1$ and

$$y_{n+1} = y_n + \varphi_1(hJ) hf(y_n) + \varphi_3(hJ) 2hr(y_{n-1}).$$

In general, for a method of order p, we choose $c_i = -i$ for $i \in \{1, \ldots, p-2\}$. Using the ansatz of eq. (4.14), the coefficients α for order 3 to 6 are given in table 4.1.

The methods obtained in this way are identical to the methods derived in (Hochbruck & Ostermann, 2011).

4.6.3 Exponential multi-value

The exponential multi-step methods presented above have two main drawbacks. First, in order to use such method with order p, we need $p-2$ previous iterates. This makes the methods hard to initialize in practice. Second, the accuracy is reduced as we reach further and further in the past because the approximations are more

Order	Coefficients α
3	$\begin{pmatrix} 2 \end{pmatrix}$
4	$\begin{pmatrix} 4 & -\frac{1}{2} \\ 6 & -\frac{3}{2} \end{pmatrix}$
5	$\begin{pmatrix} 6 & -\frac{3}{2} & \frac{2}{9} \\ 15 & -6 & 1 \\ 12 & -6 & \frac{4}{3} \end{pmatrix}$
6	$\begin{pmatrix} 8 & -3 & \frac{8}{9} & -\frac{1}{8} \\ 26 & -\frac{57}{4} & \frac{14}{3} & -\frac{11}{16} \\ 36 & -24 & \frac{28}{3} & -\frac{3}{2} \\ 20 & -15 & \frac{20}{3} & -\frac{5}{4} \end{pmatrix}$

Table 4.1: Coefficients for exponential multi-step methods

accurate when we use nodes closer to the current value. For this reason, we would like to develop methods where the nodes are closer to the current time. More specifically we will consider the values c_i corresponding to nodes occurring during the last step such that $-1 < c_i < 0$. The corresponding vector $z_i \approx y(t_n + c_i h)$ can be obtained for little to no extra cost if an adaptive pocedure such as the one presented in appendix A is used.

As an example, for a method of order 4, we can consider the nodes $c_1 = -\frac{1}{2}$ and $c_2 = -1$. The parameters α for these values are:

$$\alpha = \begin{pmatrix} 16 & -2 \\ 48 & -12 \end{pmatrix}$$

More generally, if we consider equidistant nodes from -1 to 0, we can obtain a method of order p with the node $c_i = -\frac{i}{p-2}$. Table 4.2 contains the coefficients α for these methods with order 3 to 6.

Order	Coefficients α
3	$\begin{pmatrix} 2 \end{pmatrix}$
4	$\begin{pmatrix} 16 & -2 \\ 48 & -12 \end{pmatrix}$
5	$\begin{pmatrix} 54 & -\frac{27}{2} & 2 \\ 405 & -162 & 27 \\ 972 & -486 & 108 \end{pmatrix}$
6	$\begin{pmatrix} 128 & -48 & \frac{128}{9} & -2 \\ 1664 & -912 & \frac{896}{3} & -44 \\ 9216 & -6144 & \frac{7168}{3} & -384 \\ 20480 & -15360 & \frac{20480}{3} & -1280 \end{pmatrix}$

Table 4.2: Coefficients for exponential multi-value methods

4.7 Conclusion

In this chapter, we introduced a new set of order conditions derived from a φ-series expansion of the exact solution. These order conditions naturally lead to a general ansatz for exponential methods. With this ansatz, we show that it is possible to solve these orders condition analytically. Moreover, we demonstrated that such methods have beneficial properties such as dense output and embedded lower order methods. Finally, we showed that this framework can be used to rederive known methods as well as new ones.

We are currently working on applying the schemes we introduced in this chapter to standard test problems to validate the order of convergence of the methods as well as their efficiency. Once this is done, we want to apply the methods to problems found in applications such as atmospheric models like the one we presented in chapter 2 and problems in plasma physics. We want to explore new classes of methods such as methods with complex node values similar to (Buvoli, 2021). In the future, we also want to explore the impact of the node values c on the precision and efficiency of the methods. The node values are currently degrees

of freedom that we can choose and we would like to be able to determine optimal values. We also want to study geometric properties of the methods derived in the new framework such as conservation of energy and simplecticity. These properties can be useful for some applications, and we would like to find out if it is possible to derive such methods as well.

Chapter 5

Application of exponential integrators for non-linear diffusion

The text of this chapter is a reprint of the material as it appears in Dallerit, V., Tokman, M., & Joseph, I. (2022). Exponential integrators for non-linear diffusion. https://doi.org/10.2172/1860647

5.1 Introduction

The goal of this project is to compare the performance of exponential time integrators with traditional methods such as diagonally implicit Runge-Kutta methods in the context of solving the system of reduced magnetohydrodynamics (RMHD). In this report, we present initial results of a proof of concept study that shows that exponential integrators can be an efficient alternative to traditional integration schemes.

For this work the spatial discretization is done using the finite element method and we utilize LLNL's MFEM software package for this purpose. Discretizing RMHD equations in space using finite elements leads to a large system of ordinary differential equations (ODEs) of the form:

$$\begin{cases} y'(t) = f(y(t)) \\ y(t_0) = y_n \end{cases} \tag{5.1}$$

where $y(t)$ is the vector of degrees of freedom of the system and $t_0 \leq t \leq t_f$. Due to the presence of a widely ranging timescales in the RMHD equations, system (5.1) is very stiff. Therefore, it is usually solved using an implicit time integrator, such as implicit Runge-Kutta or Backward Differentiation Formula (BDF) method. Each time step of an implicit method requires the solution of a large system of nonlinear equations. Approximating such a solution is computationally expensive, and a preconditioner is necessary. Often, preconditioners that yield sufficient efficiency are hard to construct. For these reasons, we want to explore the use of exponential integrators for this problem. These methods are attractive for large stiff problems since they have good stability properties, allow larger time step, and their cost per time step can be computationally cheaper than implicit methods.

In order to show the advantages of exponential time integrators, we formulate two nonlinear diffusion test problems corresponding to a simplified version of the RMHD model of interest. We then compare the efficiency of these exponential schemes to the traditional methods.

This report is organized as follows. First, we introduce the numerical exponential integrators used for this comparison. Then, we describe the numerical test problems and detail the implementation. Finally, we present the results of the comparison study and demonstrate that exponential time integrators can lead to a more accurate and more efficient solution compared to implicit schemes.

5.2 Time integration

5.2.1 Time stepping methods

In this section, we give a brief overview of exponential time integrators and present the schemes we will be using in the numerical experiments. We also provide details on the implementation and software we use.

Exponential integrators are usually derived in the following way. First, the right-hand side function f of (5.1) is expanded in a Taylor series around the solution

$y(t_n)$ at a given time t_n:

$$f(y) = f(y(t_n)) + J_n(y - y(t_n)) + R(y) \tag{5.2}$$

where J_n is the Jacobian matrix of f at time t_n and $R(y) = f(y) - f(y(t_n)) - J_n(y - y(t_n))$ is the nonlinear remainder of the Taylor expansion. Using the integrating factor $e^{-J_n t}$ and integrating the equation from t_n to t_{n+1}, we can get the solution at a future time $t_{n+1} = t_n + h$ as:

$$y(t_{n+1}) = y(t_n) + \varphi_1(hJ_n)hf(y(t_n)) + \int_0^1 e^{(1-\theta)hJ_n}\ hR(y(t_n + h\theta))\ d\theta \tag{5.3}$$

where the exponential-like functions φ_k are defined as:

$$\varphi_0(z) = e^z = \sum_{n=0}^{\infty} \frac{z^n}{n!}$$

$$\varphi_k(z) = \int_0^1 e^{(1-\theta)z} \frac{\theta^{k-1}}{(k-1)!} d\theta = \sum_{n=0}^{\infty} \frac{z^n}{(n+k)!} \quad \text{for} \quad k \geq 1$$

Equation (5.3) is the integral form of system (5.1) and a starting point for the derivation of most exponential integrators. An exponential integrator is derived by choosing a numerical approximation of the nonlinear integral in equation (5.3). For the study, we choose EPI type methods ((Tokman, 2006), (Tokman, 2011)) derived in (Rainwater & Tokman, 2016) as they have been shown to be efficient for this type of problems ((Tokman, Loffeld, & Tranquilli, 2012), (Einkemmer, Tokman, & Loffeld, 2017)). In particular, we will be testing the following two exponential schemes:

- EPI2 / Exponential Euler (2^{nd} order):

$$y_{n+1} = y_n + \varphi_1(hJ_n)hf_n \tag{5.4}$$

- EPIRK4 (4^{th} order) (Rainwater & Tokman, 2016) :

$$\begin{aligned} Y_1 &= y_n + \frac{1}{8}\varphi_1\left(\frac{1}{8}hJ_n\right)hf_n \\ Y_2 &= y_n + \frac{1}{9}\varphi_1\left(\frac{1}{9}hJ_n\right)hf_n \\ y_{n+1} &= y_n + \varphi_1\left(hJ_n\right)hf_n \\ &\quad + \varphi_3(hJ_n)(\alpha_{3,1}hR(Y_1) + \alpha_{3,2}hR(Y_2)) \\ &\quad + \varphi_4(hJ_n)(\alpha_{4,1}hR(Y_1) + \alpha_{4,2}hR(Y_2)) \end{aligned} \tag{5.5}$$

where $\alpha_{3,1} = -1024,\ \alpha_{3,2} = 1458,\ \alpha_{4,1} = 27648,\ \alpha_{4,2} = -34992$.

Note that the major computational cost of exponential integrator is the computation of the product between exponential functions of the jacobian matrix and vector. The EPI methods are specifically designed to reduce the number of such computations. For example, EPIRK4 requires only 2 such evaluations. As we explain below, these evaluations are usually done using Krylov projection type of technics so one would compare the number of such evaluations with the number of nonlinear iterations required at each time step of an implicit method.

For comparison we will use the following explicit and implicit methods implemented in MFEM:

- Explicit methods: explicit Euler (*ForwardEulerSolver*), Runge-Kutta of orders 2, 3 and 4 (*RK2Solver, RK3SSPSolver, RK4Solver*)
- Implicit methods: implicit Euler (*BackwardEulerSolver*), singly-diagonally implicit Runge-Kutta of orders 2 and 3 (*SDIRK23Solver, SDIRK23Solver*)

5.2.2 Exponential matrix function evaluation

The main computational challenge in the time stepping of exponential schemes is the evaluation of the matrix-vector products involving the φ_k functions. For the problems we are considering, the Jacobian matrix involved in these computations is large and stiff. Therefore, we need an efficient method adapted to this kind of problem. In this work, we are using the KIOPS algorithm (Gaudreault, Rainwater, & Tokman, 2018). It uses a Krylov subspace to project the large operator into a smaller space where the computation can be carried out more efficiently. This idea is similar to the one used in methods like GMRES or conjugate gradient. Moreover, some optimizations are used in this method to improve the efficiency. First, using a theorem from (Al-Mohy & Higham, 2011), it is possible to reduce the computation of a linear combination of φ functions of the form $\sum_{i=0}^{k} \varphi_i(hJ)v_i$ to the computation of a single matrix exponential times a vector. Then, in (Gaudreault, Rainwater, & Tokman, 2018), the authors show that during the Arnoldi process it is enough to orthogonalize the new vector with respect to the last 2 previous vec-

tors. This modification reduces the number of dot products from $O(m^2)$ to $O(m)$, where m is the size of the Krylov space, while keeping the procedure stable. This is especially important for parallel code, as dot products require a communication between the computation nodes. Unlike standard Arnoldi procedure and adaptive Krylov algorithm, KIOPS requires only 2 dot products per Krylov vector which significantly improves its parallel efficiency. The last optimization, originally presented in (Niesen & Wright, 2012), is to substep the computation of the matrix exponential by using the following property of the exponential function:

$$e^A v = e^{\tau_k A} ... e^{\tau_2 A} e^{\tau_1 A} v \quad \text{if} \quad \sum_{i=1}^{k} \tau_i = 1$$

Using this equation, it is possible to compute $e^A v$ by first computing $w_1 = e^{\tau_1 A} v$, then computing $w_2 = e^{\tau_2 A} w_1$, and so on until τ_k. By choosing $\tau_i < 1$, we have $||\tau_i A|| < ||A||$ and therefore the approximation requires a smaller Krylov subspace. This also allows the computation of $e^{\tau A}$ with several values of τ with a single Krylov projection. KIOPS uses an algorithm to compute the values of τ adaptively.

Using these optimizations, it is possible to advance the solution using the method EPIRK4 with only 2 Krylov projections instead of the 5 projections required with a naive implementation. To do so, we first compute the stages Y_1 and Y_2 in a single computation using the substepping. Then, we can compute the linear combination of φ functions with a second projection using the first optimization. For full details on EPI methods with KIOPS see (Gaudreault, Rainwater, & Tokman, 2018) and (Gaudreault et al., 2022).

5.3 Test problems

In order to compare the performance of the different time integration methods, we set up two test problems. These problems model nonlinear diffusion in a way that is similar to what is found in the RMHD problem of interest.

- 1D diffusion problem:

The first problem corresponds to a nonlinear diffusion PDE in 1 dimension.

$$\frac{\partial u}{\partial t} = \frac{\partial}{\partial x}\left(\mu(u)\frac{\partial u}{\partial x}\right) + s(x) \tag{5.6}$$

on the domain $x \in [0,1]$ with $\mu(u) = (\beta_1 + \beta_2 u^{5/2})$ and the source is defined by $s(x) = e^{-\frac{1}{2}((x-1/2)/\sigma)^2}$. The diffusion coefficients can be selected in order to adapt the stiffness and the amount of nonlinearity of the problem. With $\beta_2 = 0$, the problem is linear. In the other case, the nonlinearity is chosen to emulate the kind of diffusivity found in RMHD problems.

We found that in order to get the correct order of convergence, we need to rewrite the problem in the following form:

$$\frac{\partial u}{\partial t} = \frac{\partial^2}{\partial x^2}\left(g(u)\right) + s(x) \tag{5.7}$$

where $g(u) = \beta_1 u + \frac{2}{7}\beta_2 u^{7/2} = \int \mu(u) du$. Since mathematically equation (5.6) and (5.7) are equivalent, the discrepency is most likely due to a bug in the current version of MFEM.

- 2D anisotropic diffusion

  The second problem is an anisotropic diffusion PDE in 2 dimensions.

$$\frac{\partial u}{\partial t} = \nabla.\left(\mu(u)\nabla u\right) + s(x)$$

  on the domain $(x,y) \in [0,1]^2$ with $\mu(u) = \kappa\left[(\beta_1 + \beta_2 u^{5/2})(\hat{b}\otimes\hat{b}) + 10^{\alpha}(I - \hat{b}\otimes\hat{b})\right]$ and the source is defined by $s(x) = e^{-\frac{1}{2}((x-1/2)/\sigma)^2}$. The magnetic field b used in this experiment is the *2-wire* model and is represented in Figure 5.1 .

  The β parameters control the diffusion in the direction of the magnetic field. As in problem (5.6), it is possible to choose either a linear or nonlinear diffusion. The α coefficient controls the strength of the diffusion in the direction perpendicular to the magnetic field. As this parameter goes to 0, a boundary layer forms, making the problem stiffer.

  Similar to the 1D diffusion test, we need to rewrite the problem in the following form in order to get the correct order of convergence:

$$\frac{\partial u}{\partial t} = \nabla.\left(\left(\hat{b}\otimes\hat{b}\right)\nabla\left(\beta_1 u + \frac{2}{7}\beta_2 u^{7/2}\right)\right) + \nabla.\left((I - \hat{b}\otimes\hat{b})\nabla\left(10^{a}u\right)\right) + s(x)$$

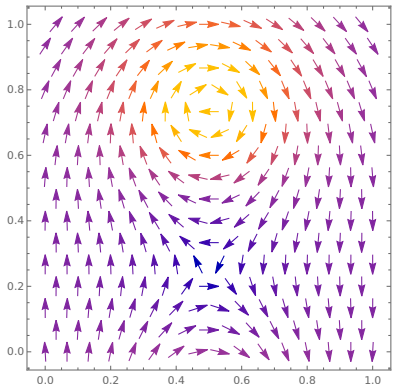


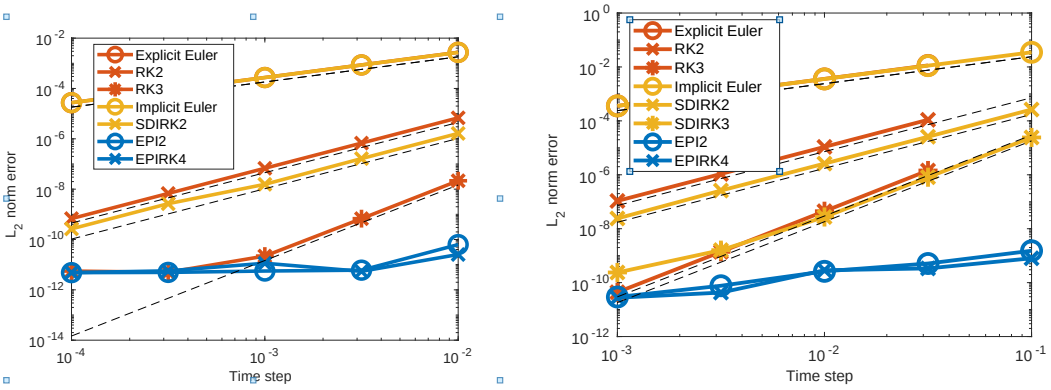
Figure 5.1: Magnetic field used for the anisotropic diffusion

5.4 Results

5.4.1 Validation of the order of convergence

Our first task is to validate the correct order of convergence of the implemented methods. Since for linear problems, the schemes (5.4) and (5.5) are exact (assuming that the φ_k functions are evaluated exactly), we want to verify that if we set the nonlinear diffusion coefficient to zero, we are getting the expected result. Figure 5.2 shows the norm of the error as a function of the timestep for the 1d test problem on the left and the 2d test problem on the right with linear diffusion. As expected, for the non-exponential methods (red and yellow curves), the error is growing with the time step. For the exponential schemes, for all values of the time step, the error stays very low. This error is nonzero because of the tolerance in the evaluation of the φ_k functions.

Figure 5.3 shows the convergence plot for the 1d test problem with nonlinear diffusion. The plots on the top correspond to a coarse grid making the problem non-stiff while the plots on the bottom are on a more refined grid making the problem stiffer. The left plots are comparing the explicit and exponential methods while the plots on the right are comparing the implicit and exponential methods. The dashed lines next to each curve correspond to the expected order of convergence. We can see that all the methods are converging at the expected rate. However, the explicit methods on the fine grid are only stable for the smallest time step while



(a) 1d test problem ($\beta_1 = 5 \times 10^{-3}, \beta_2 = 0$)

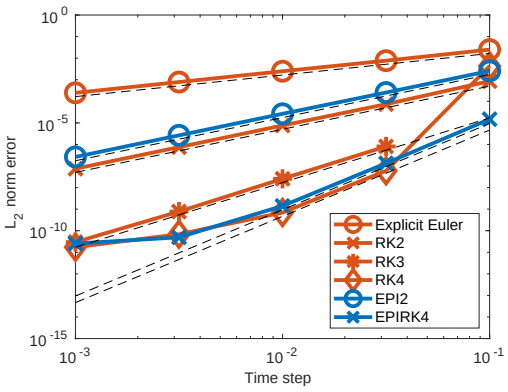
(b) 2d test problem ($\kappa = 10^{-2}, \alpha = 10^{-3}, \beta_1 = 1, \beta_2 = 0$)

Figure 5.2: Convergence diagrams for the 1d and 2d test problems with linear diffusion.

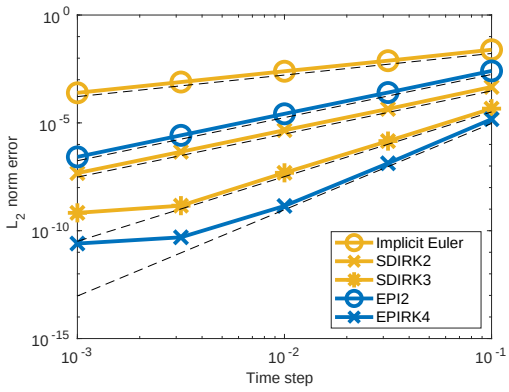
both the implicit and exponential methods do not have any stability issues. Figure 5.4 present the same results but for the 2d anisotropic diffusion test problem. With this problem, all methods still converge as expected and the explicit schemes still suffer from instability on the finer grid.

5.4.2 Performance comparison

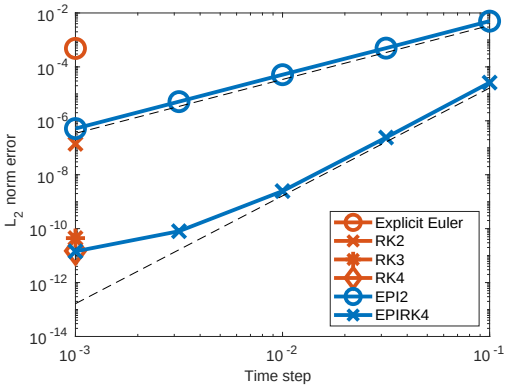
Figure 5.5 and 5.6 are showing the precision diagram (Error vs time to solution) for the 1d nonlinear diffusion and 2d anisotropic diffusion respectively. Each plot compares the performance of the exponential schemes to explicit (left) and implicit (right) methods on a coarse (top) and fine (bottom) grid. As expected, if we compare an explicit with an exponential scheme at the same order of accuracy, the time to solution for the explicit scheme will be lower as the cost per iteration is much cheaper with explicit scheme. However, as we can see from the previous section, explicit schemes quickly become unstable as the grid is refined and therefore can not be considered for stiff problems. On the other hand, we can see that the performance of exponential schemes is better than the implicit methods on both test problems. Moreover, the gap between the implicit and exponential methods is increasing going from the coarse to the finner grid. This indicates that exponential



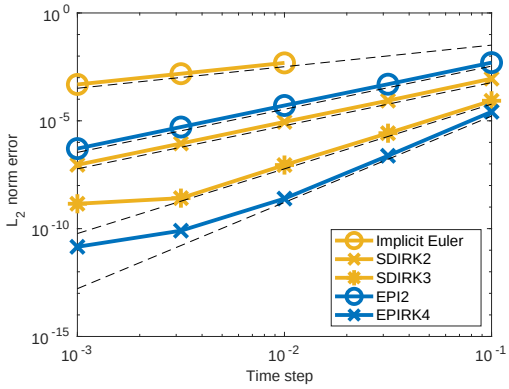
(a) Explicit methods - Coarse grid ($n_{elem} = 50$)



(b) Implicit methods - Coarse grid ($n_{elem} = 50$)



(c) Explicit methods - Fine grid ($n_{elem} = 200$)



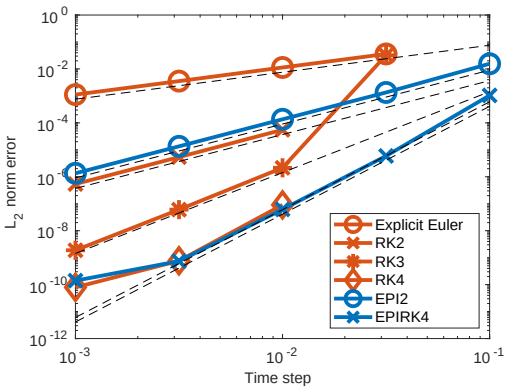
(d) Implicit methods - Fine grid ($n_{elem} = 200$)

Figure 5.3: Convergence diagrams for the 1d test problems with nonlinear diffusion ($\beta_1 = 5 \times 10^{-5}, \beta_2 = 5 \times 10^{-3}$).

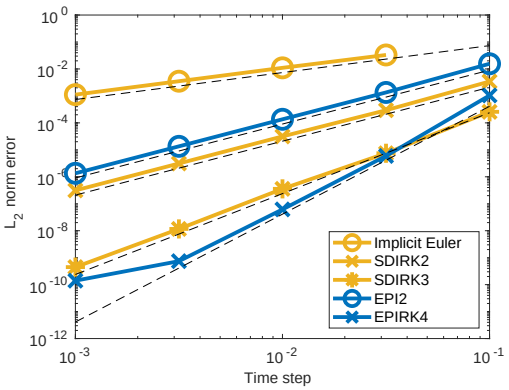
methods scale better as the problem gets stiffer.

5.5 Conclusion and future work

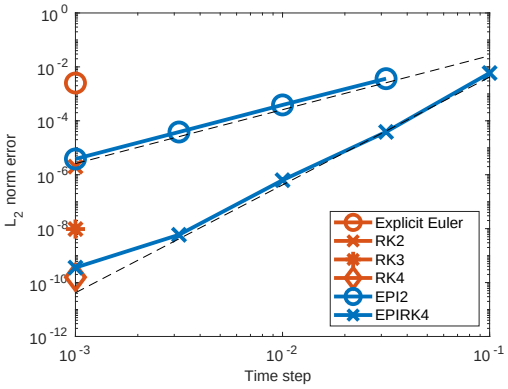
In this work, we introduced two nonlinear diffusion PDEs as test problems for the RMHD equations. We used these problems to compare the performance of exponential integrators with traditional methods. These integrators and test problems were validated by looking at the order of convergence. Moreover, the results demonstrate that exponential integrators have the potential to be an efficient



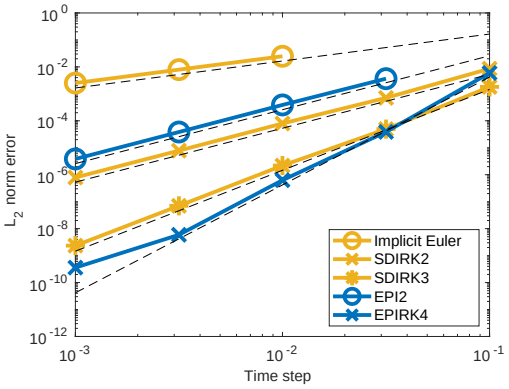
(a) Explicit methods - Coarse grid ($n_{elem} = 20^2$)



(b) Implicit methods - Coarse grid ($n_{elem} = 20^2$)



(c) Explicit methods - Fine grid ($n_{elem} = 50^2$)

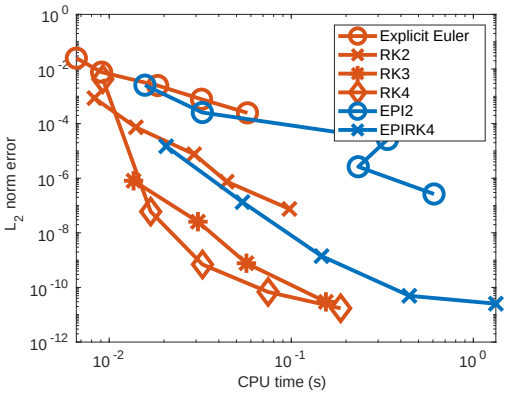


(d) Implicit methods - Fine grid ($n_{elem} = 50^2$)

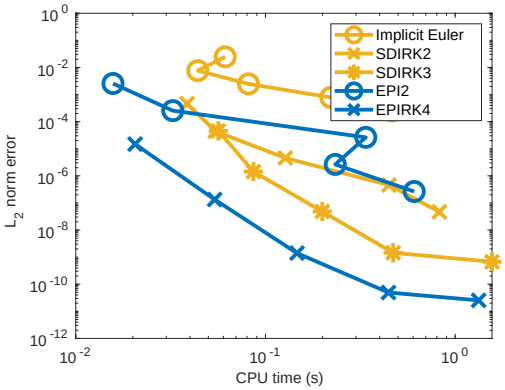
Figure 5.4: Convergence diagrams for the 2d test problems with anisotropic diffusion ($\kappa = 10^{-2}, \alpha = 10^{-3}, \beta_1 = 0, \beta_2 = 10$).

alternative to implicit methods for this kind of problem.

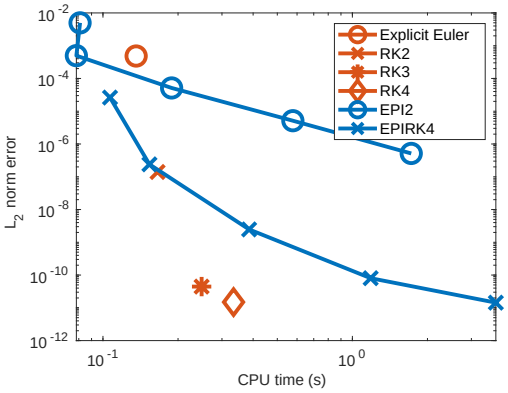
In the future, we are planning on extending this work to more realistic test problems. We also want to investigate the addition of algebraic constraints as part of the solve. These constraints are of great interest for RMHD problems and need to be treated correctly and efficiently.



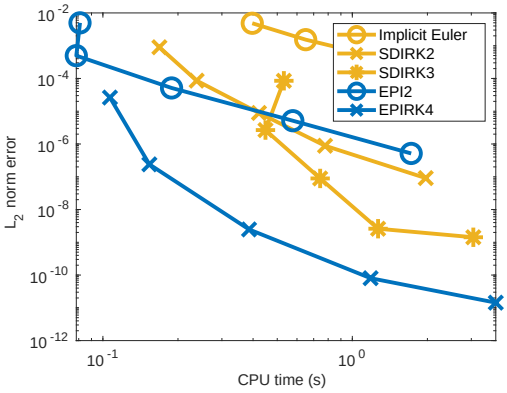
(a) Explicit methods - Coarse grid ($n_{elem} = 50$)



(b) Implicit methods - Coarse grid ($n_{elem} = 50$)

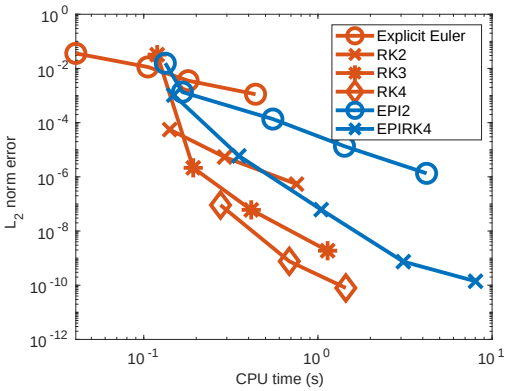


(c) Explicit methods - Fine grid ($n_{elem} = 200$)

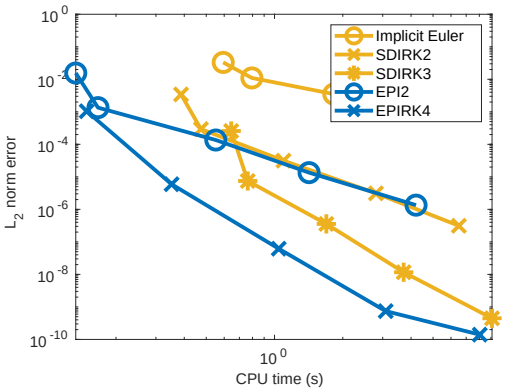


(d) Implicit methods - Fine grid ($n_{elem} = 200$)

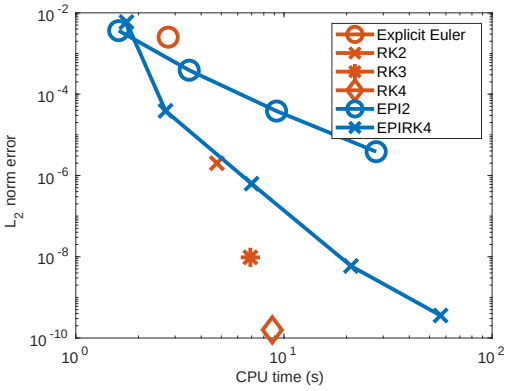
Figure 5.5: Precision diagrams for the 1d test problems with nonlinear diffusion ($\beta_1 = 5 \times 10^{-5}, \beta_2 = 5 \times 10^{-3}$).



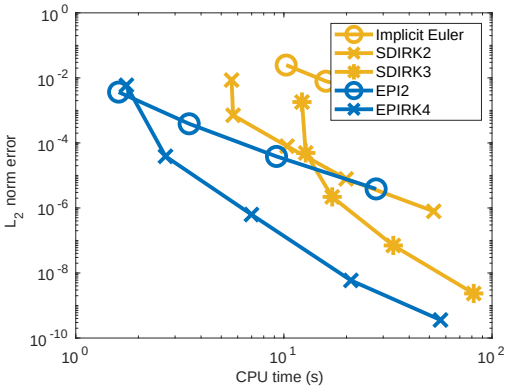
(a) Explicit methods - Coarse grid ($n_{elem} = 20^2$)



(b) Implicit methods - Coarse grid ($n_{elem} = 20^2$)



(c) Explicit methods - Fine grid ($n_{elem} = 50^2$)



(d) Implicit methods - Fine grid ($n_{elem} = 50^2$)

Figure 5.6: Precision diagrams for the 2d test problems with anisotropic diffusion ($\kappa = 10^{-2}, \alpha = 10^{-3}, \beta_1 = 0, \beta_2 = 10$).

Chapter 6

Conclusion and future work

In this dissertation, we advanced the field of exponential integration in several directions. We first introduced new high-order exponential multi-step methods that require the computation of a single matrix-exponential-vector product to be computed at each step. These methods were validated on a simplified atmospheric model. We showed that when combined with high-order spatial discretization, these methods can perform accurate simulations with very large time steps, even in the case of complex flows. We believe that further gain in efficiency could be achieved by using the multi-values methods introduced in Chapter 4. We intend to explore the use of exponential integrators for more complex atmospheric models.

We also introduced a new framework for deriving implicit-exponential partitioned integrators for stiff systems of ODEs with nonlinear-nonlinear additive forcing terms. The new time integrators are made efficient by only requiring one call to the linear solver and one call to the matrix-exponential solver. We demonstrated that these methods offer superior stability and accuracy compared to existing schemes and can be effectively used for various applications. We plan to extend the ansatz introduced in this work to allow the construction of higher-order methods. This can be done by computing intermediate stages to derive Runge-Kutta type methods or using the values from previous iterations to derive multi-step type methods.

Stiffness resilient methods were also introduced. These methods are similar to the stiffly accurate methods and do not suffer from order reduction. However,

they can be easily derived, and some of their properties were presented. We are currently working on applying the schemes we introduced in this work to standard test problems to validate the order of convergence of the methods as well as their efficiency. We believe that all the methods present in this work are also stiffly accurate. However, this still needs to be proven, and the exact conditions to have stiff accuracy in this framework are not fully understood yet. Moreover, many aspects of these methods still need to be explored. For example, we want to study the impact of the node values c on the methods' accuracy, stability, and efficiency. The node values are currently degrees of freedom that we can choose, and we would like to be able to determine optimal values. We also want to study the geometric properties of the methods derived in the new framework, such as conservation of energy and simplecticity. These properties can be important for some applications, and we would like to determine if it is possible to derive methods with such guarantees in this framework. We plan on exploring new classes of methods and extending the framework to new types of schemes. We would like to consider methods with complex node values similar to the work presented in (Buvoli, 2021) and methods where the stages are computed using non-exponential methods. We will further consider the extension of the framework to allow schemes where the matrix in the φ_k functions does not need to be the Jacobian of the full right-hand side but can be any linear operator. Some initial work has been done in this direction and indicates that similar results can be obtained in this case. In parallel, we also want to apply the stiffness resilient methods to the applications presented in this dissertation. With the methods we presented in this dissertation, much attention was paid minimizing the number of matrix exponential and φ_k evaluations that must be computed at each step. To further improve the efficiency of exponential methods, we believe that further progress needs to be achieved in the computation of this term. Further advances in the adaptive procedure of the KIOPS method presented in Appendix A are possible. Moreover, for each substep, the Krylov subspace is currently discarded. Reusing the basis or a part of it could help reduce the number of right-hand side evaluations and communications. We also want to explore other techniques to compute the matrix

exponential and associated φ_k functions, such as the Leja point approximation, Taylor expansion method, and contour integral approximation. We believe that depending on the application, one or a combination of these techniques can provide faster computations.

Finally, we demonstrated that exponential methods are a good alternative to the implicit schemes often used to simulate the reduced magnetohydrodynamics (RMHD) equations. In the future, we plan to extend this analysis to a broader selection of exponential methods and make a similar comparison on the Kelvin Helmholtz instability problems. This problem is closer to the RMHD equation and is a step toward a more realistic problem. However, on top of the system of ordinary differential equations, this system contains an algebraic constraint. Computing this constraint for each evaluation of the right-hand side might be too expensive. For this reason, we will explore more efficient techniques to enforce this constraint. One possibility is to use exponential methods that allow the matrix in the φ_k function to be only a part of the Jacobian of the right-hand side. This would allow us to remove the constraint from the Jacobian and compute the constraint only once per time step. In summary, in this thesis, we constructed new time integration methods, developed a novel framework for constructing novel classes of time integrators, and explored the performance of the new schemes and methodologies on test problems and important applications in weather and climate modeling and plasma physics. This work generated both new ideas as well as opened new research directions we plan to pursue in the future.

Bibliography

Abdulle, A. (2002). Fourth order chebyshev methods with recurrence relation. *SIAM Journal on Scientific Computing*, *23*(6), 2041–2054.

Al-Mohy, A. H., & Higham, N. J. (2011). Computing the action of the matrix exponential, with an application to exponential integrators. *SIAM journal on scientific computing*, *33*(2), 488–511.

Ascher, U. M., Larionov, E., Sheen, S. H., & Pai, D. K. (2021). Simulating deformable objects for computer animation: A numerical perspective. *arXiv preprint arXiv:2103.01891*.

Ascher, U. M., Ruuth, S. J., & Spiteri, R. J. (1997). Implicit-explicit runge-kutta methods for time-dependent partial differential equations. *Applied Numerical Mathematics*, *25*(2-3), 151–167.

Ascher, U. M., Ruuth, S. J., & Wetton, B. T. (1995). Implicit-explicit methods for time-dependent partial differential equations. *SIAM Journal on Numerical Analysis*, *32*(3), 797–823.

Bader, P., Blanes, S., & Casas, F. (2019). Computing the matrix exponential with an optimized taylor polynomial approximation. *Mathematics*, *7*(12), 1174.

Bao, L., Nair, R. D., & Tufo, H. M. (2014). A mass and momentum flux-form high-order discontinuous Galerkin shallow water model on the cubed-sphere. *Journal of Computational Physics*, *271*, 224–243.

Belytschko, T., Yen, H.-J., & Mullen, R. (1979). Mixed methods for time integration. *Computer Methods in Applied Mechanics and Engineering*, *17*, 259–275.

Bergamaschi, L., Caliari, M., & Vianello, M. (2003). Efficient approximation of the exponential operator for discrete 2d advection–diffusion problems. *Numerical linear algebra with applications*, *10*(3), 271–289.

Berger, M. J., & Oliger, J. E. (1982). Adaptive mesh refinement for hyperbolic partial differential equations.

Blanes, S., & Casas, F. (2005). On the necessity of negative coefficients for operator splitting schemes of order higher than two. *Applied Numerical Mathematics*, *54*(1), 23–37.

Butcher, J. (2010). Trees, B-series and exponential integrators. *IMA journal of numerical analysis*, *30*(1), 131–140.

Butcher, J. C. (2016). *Numerical methods for ordinary differential equations*. John Wiley & Sons.

Butcher, J. C. (2021). *B-series: Algebraic analysis of numerical methods* (Vol. 55). Springer Nature.

Buvoli, T. (2021). Exponential polynomial block methods. *SIAM J. Sci. Comput.*, *43*, A1692–A1722.

Calandrini, S., Pieper, K., & Gunzburger, M. (2021). Numerical analyses of exponential time-differencing schemes for the solution of atmospheric models. *Quarterly Journal of the Royal Meteorological Society*, *147*(736), 1477–1496.

Calandrini, S., Pieper, K., & Gunzburger, M. D. (2020). Exponential time differencing for the tracer equations appearing in primitive equation ocean models. *Computer Methods in Applied Mechanics and Engineering*, *365*, 113002.

Caliari, M., Einkemmer, L., Moriggl, A., & Ostermann, A. (2021). An accurate and time-parallel rational exponential integrator for hyperbolic and oscillatory PDEs. *Journal of Computational Physics*, *437*, 110289.

Caliari, M., Kandolf, P., Ostermann, A., & Rainer, S. (2016). The Leja method revisited: Backward error analysis for the matrix exponential. *SIAM Journal on Scientific Computing*, *38*(3), A1639–A1661.

Caliari, M., Kandolf, P., & Zivcovich, F. (2018). Backward error analysis of polynomial approximations for computing the action of the matrix exponential. *BIT Numerical Mathematics*, *58*(4), 907–935.

Caliari, M., Vianello, M., & Bergamaschi, L. (2004). Interpolating discrete advection–diffusion propagators at leja sequences. *Journal of Computational and Applied Mathematics*, *172*(1), 79–99.

Certaine, J. (1960). The solution of ordinary differential equations with large time constants. *Mathematical methods for digital computers*, *1*, 128–132.

Charron, M., & Zadra, A. (2014). On the dynamical consistency of geometrically approximated equations for geophysical fluids in arbitrary coordinates. *Q. J. R. Meteorol. Soc.*, *140*, 2078–2083.

Charron, M., Zadra, A., & Girard, C. (2014). Four-dimensional tensor equations for a classical fluid in an external gravitational field. *Q. J. R. Meteorol. Soc.*, *140*, 908–916.

Chartier, P., Hairer, E., & Vilmart, G. (2010). Algebraic structures of b-series. *Foundations of Computational Mathematics*, *10*(4), 407–427.

Chen, C., Li, X., Shen, X., & Xiao, F. (2015). A high-order conservative collocation scheme and its application to global shallow-water equations. *Geoscientific Model Development*, *8*(2), 221–233.

Chen, C., & Xiao, F. (2008). Shallow water model on cubed-sphere by multi-moment finite volume method. *Journal of Computational Physics*, *227*(10), 5019–5044.

Chen, Y. J., Sheen, S. H., Ascher, U. M., & Pai, D. K. (2020). Siere: A hybrid semi-implicit exponential integrator for efficiently simulating stiff deformable objects. *ACM Transactions on Graphics (TOG)*, *40*(1), 1–12.

Clancy, C., & Pudykiewicz, J. A. (2013). On the use of exponential time integration methods in atmospheric models. *Tellus A: Dynamic Meteorology and Oceanography*, *65*(1), 20898.

Courant, R., Friedrichs, K., & Lewy, H. (1928). Über die partiellen Differenzengleichungen der mathematischen Physik. *Mathematische Annalen*, *100*(1), 32–74.

Dallerit, V., Buvoli, T., Tokman, M., & Gaudreault, S. (2022). Second-order rosenbrock-exponential (ROSEXP) methods for partitioned differential equations. *to be submitted to SIAM Journal on Scientific Computing*.

Dallerit, V., Tokman, M., & Joseph, I. (2022). Exponential integrators for nonlinear diffusion. https://doi.org/10.2172/1860647

de la Hoz, F., & Vadillo, F. (2008). An exponential time differencing method for the nonlinear schrö dinger equation. *Comput. Phys. Commun.*, *179*, 449–456.

Dinh, K. N., & Sidje, R. B. (2017). Analysis of inexact Krylov subspace methods for approximating the matrix exponential. *Mathematics and Computers in Simulation*, *138*, 1–13.

Dongarra, J., Hittinger, J., Bell, J., Chacon, L., Falgout, R., Heroux, M., Hovland, P., Ng, E., Webster, C., & Wild, S. (2014). *Applied mathematics research for exascale computing* (tech. rep.). Lawrence Livermore National Lab.(LLNL), Livermore, CA (United States).

Einkemmer, L., Tokman, M., & Loffeld, J. (2017). On the performance of exponential integrators for problems in magnetohydrodynamics. *Journal of Computational Physics*, *330*, 550–565. https://doi.org/https://doi.org/10.1016/j.jcp.2016.11.027

Galewsky, J., Scott, R. K., & Polvani, L. M. (2004). An initial-value problem for testing numerical models of the global shallow-water equations. *Tellus A: Dynamic Meteorology and Oceanography*, *56*(5), 429–440.

Gardner, D. J., Guerra, J. E., Hamon, F. P., Reynolds, D. R., Ullrich, P. A., & Woodward, C. S. (2018). Implicit–explicit (imex) runge–kutta methods for non-hydrostatic atmospheric models. *Geoscientific Model Development*, *11*(4), 1497–1515. https://doi.org/10.5194/gmd-11-1497-2018

Gassner, G., & Kopriva, D. A. (2011). A comparison of the dispersion and dissipation errors of Gauss and Gauss–Lobatto discontinuous Galerkin spectral element methods. *SIAM Journal on Scientific Computing*, *33*(5), 2560–2579.

Gaudreault, S., Charron, M., Dallerit, V., & Tokman, M. (2022). High-order numerical solutions to the shallow-water equations on the rotated cubed-sphere grid. *Journal of Computational Physics*, *449*, 110792.

Gaudreault, S., & Pudykiewicz, J. A. (2016). An efficient exponential time integration method for the numerical solution of the shallow water equations on the sphere. *Journal of Computational Physics*, *322*, 827–848.

Gaudreault, S., Rainwater, G., & Tokman, M. (2018). Kiops: A fast adaptive krylov subspace solver for exponential integrators. *Journal of Computational Physics*, *372*, 236–255.

Giraldo, F. X. (2020). *An introduction to element-based Galerkin methods on tensor-product bases: Analysis, algorithms, and applications*. Springer Nature.

Godenschwager, C., Schornbaum, F., Bauer, M., Köstler, H., & Rüde, U. (2013). A framework for hybrid parallel flow simulations with a trillion cells in complex geometries. *Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis*, 1–12.

Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep learning*. MIT press.

Gradinaru, V. (2007). Strang splitting for the time-dependent schrödinger equation on sparse grids. *SIAM J. Numer. Anal.*, *46*, 103–123.

Hederi, M., Islas, A. L., Reger, K., & Schober, C. M. (2016). Efficiency of exponential time differencing schemes for nonlinear schrödinger equations. *Math. Comput. Simul.*, *127*, 101–113.

Hesthaven, J. S., & Warburton, T. (2007). *Nodal discontinuous Galerkin methods: Algorithms, analysis, and applications*. Springer Science & Business Media.

Higham, N. J. (2005). The scaling and squaring method for the matrix exponential revisited. *SIAM Journal on Matrix Analysis and Applications*, *26*(4), 1179–1193.

Higham, N. J. (2008). *Functions of matrices: Theory and computation*. Society for Industrial; Applied Mathematics.

Higueras, I., Happenhofer, N., Koch, O., & Kupka, F. (2014). Optimized strong stability preserving imex runge-kutta methods. *J. Comput. Appl. Math.*, *272*, 116–140.

Hochbruck, M., & Ostermann, A. (2005). Explicit exponential Runge–Kutta methods for semilinear parabolic problems. *SIAM Journal on Numerical Analysis*, *43*(3), 1069–1090.

Hochbruck, M., & Ostermann, A. (2010). Exponential integrators. *Acta Numerica*, *19*, 209–286.

Hochbruck, M., & Ostermann, A. (2011). Exponential multistep methods of Adams-type. *BIT*, *51*(4), 889–908. https://doi.org/10.1007/s10543-011-0332-6

Hochbruck, M., Ostermann, A., & Schweitzer, J. (2008). Exponential rosenbrock-type methods. *SIAM J. Numer. Anal.*, *47*, 786–803.

Holden, H., Lubich, C., & Risebro, N. H. (2013). Operator splitting for partial differential equations with burgers nonlinearity. *Math. Comput.*, *82*, 173–185.

Holec, M., Zhu, B., Joseph, I., Vogl, C. J., Southworth, B. S., Campos, A., Dimits, A. M., & Pazner, W. (2022). Arbitrary order energy and enstrophy conserving finite element methods for 2d incompressible fluid dynamics and drift-reduced magnetohydrodynamics. *ArXiv*, *abs/2202.13022*.

Hundsdorfer, W. H., Verwer, J. G., & Hundsdorfer, W. (2003). *Numerical solution of time-dependent advection-diffusion-reaction equations* (Vol. 33). Springer.

Huynh, H. T. (2007). A flux reconstruction approach to high-order schemes including discontinuous Galerkin methods. *18th AIAA Computational Fluid Dynamics Conference*, 4079.

Huynh, H. T. (2019). Discontinuous Galerkin via interpolation: The direct flux reconstruction method. *AIAA Aviation 2019 Forum*, 3064.

Jablonowski, C., & Williamson, D. L. (2011). The pros and cons of diffusion, filters and fixers in atmospheric general circulation models. In *Numerical techniques for global atmospheric models* (pp. 381–493). Springer.

Jameson, A., Vincent, P. E., & Castonguay, P. (2012). On the non-linear stability of flux reconstruction schemes. *Journal of Scientific Computing*, *50*(2), 434–445.

Kennedy, C. A., & Carpenter, M. H. (2003). Additive runge–kutta schemes for convection–diffusion–reaction equations. *Applied numerical mathematics*, *44*(1-2), 139–181.

Keyes, D. E., McInnes, L. C., Woodward, C., Gropp, W., Myra, E., Pernice, M., Bell, J., Brown, J., Clo, A., Connors, J., et al. (2013). Multiphysics simulations: Challenges and opportunities. *The International Journal of High Performance Computing Applications*, *27*(1), 4–83.

Knoll, D. A., & Keyes, D. E. (2004). Jacobian-free Newton–Krylov methods: A survey of approaches and applications. *Journal of Computational Physics*, *193*(2), 357–397.

Kopriva, D. A., & Gassner, G. (2010). On the quadrature and weak form choices in collocation type discontinuous Galerkin spectral element methods. *Journal of Scientific Computing*, *44*(2), 136–155.

Läuter, M., Handorf, D., & Dethloff, K. (2005). Unsteady analytical solutions of the spherical shallow water equations. *Journal of Computational Physics*, *210*(2), 535–553.

LeVeque, R. J., et al. (2002). *Finite volume methods for hyperbolic problems* (Vol. 31). Cambridge university press.

Li, X., Chen, D., Peng, X., Takahashi, K., & Xiao, F. (2008). A multimoment finite-volume shallow-water model on the yin–yang overset spherical grid. *Monthly weather review*, *136*(8), 3066–3086.

Liou, M.-S. (2006). A sequel to AUSM, Part II: AUSM+-up for all speeds. *Journal of computational physics*, *214*(1), 137–170.

Liou, M.-S., & Steffen Jr, C. J. (1993). A new flux splitting scheme. *Journal of Computational physics*, *107*(1), 23–39.

Loffeld, J., & Tokman, M. (2013). Comparative performance of exponential, implicit, and explicit integrators for stiff systems of odes. *Journal of Computational and Applied Mathematics*, *241*, 45–67. https://doi.org/https://doi.org/10.1016/j.cam.2012.09.038

Loffeld, J., & Hittinger, J. A. F. (2019). On the arithmetic intensity of high-order finite-volume discretizations for hyperbolic systems of conservation laws. *The International Journal of High Performance Computing Applications*, *33*, 25–52.

Luan, V. T. (2020). Efficient exponential runge–kutta methods of high order: Construction and implementation. *BIT Numerical Mathematics*, 1–26.

Luan, V. T., Chinomona, R., & Reynolds, D. R. (2020). A new class of high-order methods for multirate differential equations. *SIAM Journal on Scientific Computing*, *42*(2), A1245–A1268.

Luan, V. T., & Ostermann, A. (2014). Exponential rosenbrock methods of order five - construction, analysis and numerical comparisons. *J. Comput. Appl. Math.*, *255*, 417–431.

Luan, V. T., Pudykiewicz, J. A., & Reynolds, D. R. (2019). Further development of efficient and accurate time integration schemes for meteorological models. *Journal of Computational Physics*, *376*, 817–837.

Luan, V. T., Tokman, M., & Rainwater, G. (2017). Preconditioned implicit-exponential integrators (imexp) for stiff pdes. *Journal of Computational Physics*, *335*, 846–864.

MacNamara, S., & Strang, G. (2016). Operator splitting. In *Splitting methods in communication, imaging, science, and engineering* (pp. 95–114). Springer.

Marras, S., Kelly, J. F., Moragues, M., Mü ller, A., Kopera, M. A., Vázquez, M., Giraldo, F. X., Houzeaux, G., & Jorba, O. (2016). A review of element-based Galerkin methods for numerical weather prediction: Finite elements, spectral elements, and discontinuous Galerkin. *Archives of Computational Methods in Engineering*, *23*(4), 673–722.

Mengaldo, G., Wyszogrodzki, A., Diamantakis, M., Lock, S.-J., Giraldo, F. X., & Wedi, N. P. (2019). Current and emerging time-integration strategies in global numerical weather and climate prediction. *Archives of Computational Methods in Engineering*, *26*(3), 663–684.

Michels, D. L., Luan, V. T., & Tokman, M. (2017). A stiffly accurate integrator for elastodynamic problems. *ACM Transactions on Graphics (TOG)*, *36*, 1–14.

Minchev, B., & Wright, W. (2005). *A review of exponential integrators for first order semi-linear problems. technical report 2.* (tech. rep.). Norwegian University of Science and Technology.

Moler, C., & Van Loan, C. (1978). Nineteen dubious ways to compute the exponential of a matrix. *SIAM Review*, *20*(4), 801–836.

Moler, C., & Van Loan, C. (2003). Nineteen dubious ways to compute the exponential of a matrix, twenty-five years later. *SIAM Review*, *45*(1), 3–49.

Nair, R. D. (2015). Quadrature-free implementation of a discontinuous Galerkin global shallow-water model via flux correction procedure. *Monthly Weather Review*, *143*(4), 1335–1346.

Nair, R. D., Levy, M. N., & Lauritzen, P. H. (2011). Emerging numerical methods for atmospheric modeling. In *Numerical techniques for global atmospheric models* (pp. 251–311). Springer.

Nair, R. D., Thomas, S. J., & Loft, R. D. (2005a). A discontinuous Galerkin global shallow water model. *Monthly weather review*, *133*(4), 876–888.

Nair, R. D., Thomas, S. J., & Loft, R. D. (2005b). A discontinuous Galerkin transport scheme on the cubed sphere. *Monthly Weather Review*, *133*(4), 814–828.

Niesen, J., & Wright, W. M. (2012). Algorithm 919: A Krylov subspace algorithm for evaluating the φ-functions appearing in exponential integrators. *ACM Transactions on Mathematical Software (TOMS)*, *38*(3), 22.

Noelle, S., Pankratz, N., Puppo, G., & Natvig, J. R. (2006). Well-balanced finite volume schemes of arbitrary order of accuracy for shallow water flows. *Journal of Computational Physics*, *213*(2), 474–499.

Pareschi, L., & Russo, G. (2005). Implicit–explicit runge–kutta schemes and applications to hyperbolic systems with relaxation. *Journal of Scientific computing*, *25*(1), 129–155.

Peixoto, P. S., & Schreiber, M. (2019). Semi-Lagrangian exponential integration with application to the rotating shallow water equations. *SIAM Journal on Scientific Computing*, *41*(5), B903–B928.

Pieper, K., Sockwell, K. C., & Gunzburger, M. D. (2019). Exponential time differencing for mimetic multilayer ocean models. *J. Comput. Phys.*, *398*.

Qaddouri, A., Pudykiewicz, J., Tanguay, M., Girard, C., & Côté, J. (2012). Experiments with different discretizations for the shallow-water equations on

a sphere. *Quarterly Journal of the Royal Meteorological Society*, *138*(665), 989–1003.

Rainwater, G., & Tokman, M. (2016). A new approach to constructing efficient stiffly accurate epirk methods. *Journal of Computational Physics*, *323*, 283–309.

Rainwater, G., & Tokman, M. (2017). Designing efficient exponential integrators with epirk framework.

Rančić, M., Purser, R., & Mesinger, F. (1996). A global shallow-water model using an expanded spherical cube: Gnomonic versus conformal coordinates. *Quarterly Journal of the Royal Meteorological Society*, *122*(532), 959–982.

Rentrop, P. (1985). Partitioned runge-kutta methods with stiffness detection and stepsize control. *Numerische Mathematik*, *47*, 545–564.

Robert, A. (1982). A semi-Lagrangian and semi-implicit numerical integration scheme for the primitive meteorological equations. *Journal of the Meteorological Society of Japan. Ser. II*, *60*(1), 319–325.

Roe, P. L. (1981). Approximate riemann solvers, parameter vectors, and difference schemes. *Journal of computational physics*, *43*(2), 357–372.

Romero, J., Asthana, K., & Jameson, A. (2016). A simplified formulation of the flux reconstruction method. *Journal of Scientific Computing*, *67*(1), 351–374.

Romero, J., Crabill, J., Watkins, J., Witherden, F., & Jameson, A. (2020). ZEFR: A GPU-accelerated high-order solver for compressible viscous flows using the flux reconstruction method. *Computer Physics Communications*, 107169.

Romero, J. D. (2017). *On the development of the direct flux reconstruction scheme for high-order fluid flow simulations* (Doctoral dissertation). Stanford University.

Ronchi, C., Iacono, R., & Paolucci, P. S. (1996). The "cubed sphere": A new method for the solution of partial differential equations in spherical geometry. *Journal of computational physics*, *124*(1), 93–114.

Rosenbrock, H. H. (1963). Some general implicit processes for the numerical solution of differential equations. *The Computer Journal*, *5*(4), 329–330. https://doi.org/10.1093/comjnl/5.4.329

Rossmanith, J. A. (2006). A wave propagation method for hyperbolic systems on the sphere. *Journal of Computational Physics*, (2), 629–658.

Rusanov, V. V. (1962). The calculation of the interaction of non-stationary shock waves and obstacles. *USSR Computational Mathematics and Mathematical Physics*, *1*(2), 304–320.

Saad, Y., & Schultz, M. H. (1986). Gmres: A generalized minimal residual algorithm for solving nonsymmetric linear systems. *SIAM Journal on Scientific and Statistical Computing*, *7*(3), 856–869. https://doi.org/10.1137/0907058

Saad, Y. (1992). Analysis of some krylov subspace approximations to the matrix exponential operator. *SIAM Journal on Numerical Analysis*, *29*(1), 209–228.

Sadourny, R. (1972). Conservative finite-difference approximations of the primitive equations on quasi-uniform spherical grids. *Monthly Weather Review*, *100*(2), 136–144.

Sandu, A., & Günther, M. (2015). A generalized-structure approach to additive Runge–Kutta methods. *SIAM Journal on Numerical Analysis*, *53*(1), 17–42.

Schiesser, W. E. (1991). The numerical method of lines: Integration of partial differential equations.

Schreiber, M., Schaeffer, N., & Loft, R. (2019). Exponential integrators with parallel-in-time rational approximations for the shallow-water equations on the rotating sphere. *Parallel Computing*, *85*, 56–65.

Shashkin, V. V., & Goyman, G. S. (2020). Semi-Lagrangian exponential time-integration method for the shallow water equations on the cubed sphere grid. *Russian Journal of Numerical Analysis and Mathematical Modelling*, *35*(6), 355–366.

Sidje, R. B. (1998). Expokit: A software package for computing matrix exponentials. *ACM Transactions on Mathematical Software (TOMS)*, *24*(1), 130–156.

Sjogreen, B., Yee, H. C., Djomehri, J., Lazanoff, A., & Henshaw, W. D. (2009). Parallel performance of adpdis3d - a high order multiblock overlapping grid solver for hypersonic turbulence.

Squire, W., & Trapp, G. (1998). Using complex variables to estimate derivatives of real functions. *SIAM review*, *40*(1), 110–112.

Thuburn, J., & Li, Y. (2000). Numerical simulations of Rossby–Haurwitz waves. *Tellus A*, *52*(2), 181–189.

Tokman, M., Loffeld, J., & Tranquilli, P. (2012). New Adaptive Exponential Propagation Iterative Methods of Runge–Kutta Type. *SIAM Journal on Scientific Computing*, *34*(5), A2650–A2669. https://doi.org/10.1137/110849961

Tokman, M. (2006). Efficient integration of large stiff systems of odes with exponential propagation iterative (EPI) methods. *Journal of Computational Physics*, *213*(2), 748–776.

Tokman, M. (2011). A new class of exponential propagation iterative methods of Runge– Kutta type (EPIRK). *Journal of Computational Physics*, *230*(24), 8762–8778.

Toro, E. F. (2013). *Riemann solvers and numerical methods for fluid dynamics: A practical introduction*. Springer Science & Business Media.

Tranquilli, P., & Sandu, A. (2014a). Exponential-Krylov methods for ordinary differential equations. *Journal of Computational Physics*, *278*, 31–46. https://doi.org/https://doi.org/10.1016/j.jcp.2014.08.013

Tranquilli, P., & Sandu, A. (2014b). Rosenbrock–Krylov methods for large systems of differential equations. *SIAM Journal on Scientific Computing*, *36*(3), A1313–A1338. https://doi.org/10.1137/130923336

Ullrich, P. A. (2014). A global finite-element shallow-water model supporting continuous and discontinuous elements. *Geoscientific Model Development*, *7*(6), 3017–3035.

Ullrich, P. A., & Jablonowski, C. (2012). Mcore: A non-hydrostatic atmospheric dynamical core utilizing high-order finite-volume methods. *Journal of Computational Physics*, *231*(15), 5078–5108.

Ullrich, P. A., Jablonowski, C., & Van Leer, B. (2010). High-order finite-volume methods for the shallow-water equations on the sphere. *Journal of Computational Physics*, *229*(17), 6104–6134.

van der Vorst, H. A. (1987). An iterative solution method for solving f (a) x = b , using krylov subspace information obtained for the symmetric positive definite matrix a. *Journal of Computational and Applied Mathematics*, *18*, 249–263.

Wanner, G., & Hairer, E. (1996). *Solving ordinary differential equations ii* (Vol. 375). Springer Berlin Heidelberg.

Weller, H., Thuburn, J., & Cotter, C. J. (2012). Computational modes and grid imprinting on five quasi-uniform spherical C grids. *Monthly weather review*, *140*(8), 2734–2755.

Williamson, D. L., Drake, J. B., Hack, J. J., Jakob, R. ü., & Swarztrauber, P. N. (1992). A standard test set for numerical approximations to the shallow water equations in spherical geometry. *Journal of Computational Physics*, *102*(1), 211–224.

Witherden, F., & Vincent, P. (2020). On nodal point sets for flux reconstruction. *Journal of Computational and Applied Mathematics*, 113014.

Zhang, W., Almgren, A., Beckner, V., Bell, J., Blaschke, J., Chan, C., Day, M., Friesen, B., Gott, K., Graves, D., et al. (2019). Amrex: A framework for block-structured adaptive mesh refinement. *Journal of Open Source Software*, *4*(37), 1370–1370.

Appendix A

Introduction to the derivation and computation of exponential time integrators

This appendix contains a short introduction to the derivation of exponential time integration methods and presents some of the common exponential schemes used in the literature. It also presents a numerical method to compute the φ_k function used in exponential schemes.

A.1 Derivation of exponential integrators

We consider the system (1.1) and its solution at time t, $y(t)$. We want to derive a scheme that can advance the problem from a known solution at time t_n, $y(t_n)$ to the solution $y(t_{n+1})$ at a future time $t_{n+1} = t_n + h$. Exponential integrators can be derived using the following reformulation. First, the function f is expanded in a Taylor series around $y(t_n)$:

$$f(z) = f(y(t_n)) + J_n(z - y(t_n)) + R(z) \tag{A.1}$$

where J_n is the Jacobian matrix of f at $y(t_n)$ and $R(z) = f(z) - f(y(t_n)) - J_n(z - y(t_n))$ is the nonlinear remainder of the Taylor expansion. Using the integrating factor $e^{-J_n t}$ and integrating the equation from t_n to t_{n+1}, we can write the solution

at time t_{n+1} as:

$$y(t_{n+1}) = e^{hJ_n} y(t_n) + \int_{t_n}^{t_{n+1}} e^{J_n(t_{n+1}-t)} \left(f(y(t_n)) - J_n y(t_n) + R(y(t)) \right) \, dt \quad \text{(A.2)}$$

Simplifying this expression and using the change of variable $t = t_n + \theta h$, we get:

$$y(t_{n+1}) = y(t_n) + \varphi_1(hJ_n) h f(y(t_n)) + \int_0^1 e^{(1-\theta)hJ_n} \, hR(y(t_n + h\theta)) \, d\theta \quad \text{(A.3)}$$

where $\varphi_1(z) = (e^z - 1) \, z^{-1}$.

This equation is the starting point for the derivation of an exponential integrator and no approximation has been used yet. From here, it is necessary to approximate the nonlinear reminder R to derive a numerical scheme. We define exponential-like functions φ_k as:

$$\varphi_0(z) = e^z = \sum_{n=0}^{\infty} \frac{z^n}{n!} \quad \text{(A.4a)}$$

$$\varphi_k(z) = \int_0^1 e^{(1-\theta)z} \frac{\theta^{k-1}}{(k-1)!} d\theta = \sum_{n=0}^{\infty} \frac{z^n}{(n+k)!} \quad \text{for} \quad k \geq 1 \quad \text{(A.4b)}$$

If the nonlinear reminder is approximated using a polynomial in θ: $R(y(t_n + h\theta)) \approx R_0 + \theta R_1 + ... + \frac{\theta^p}{p!} R_p$ where $R_i \in \mathbb{R}^N$ then eq. (A.3) becomes:

$$y(t_{n+1}) \approx y(t_n) + \varphi_1(hJ_n) h (f(y(t_n)) + R_0) + \varphi_2(hJ_n) h R_1 + ... + \varphi_{p+1}(hJ_n) h R_p$$

This shows that the φ_k functions are a natural basis to derive exponential integrators. In practice, exponential schemes are usually not derived directly using a polynomial to approximate the function R but using the order condition theory presented.

We are now presenting some exponential methods that can be derived from eq. (A.3). We assume that we have an approximation $y_n \approx y(t_n)$ of the solution at time t_n and compute the approximation $y_{n+1} \approx y_(t_{n+1})$ of the solution at time t_{n+1}. The simplest and one of the earliest exponential method is the exponential Euler method which can be obtained by approximating the nonlinear residual by 0. For a historical review of exponential integrators, we invite the reader to refer to (Minchev & Wright, 2005).

- Exponential Euler (2^{nd} order):

$$y_{n+1} = y_n + \varphi_1(hJ_n) h f(y_n)$$

If we wish to get higher order methods, just like with classical methods, two techniques are usually used. First, one can use the values from the previous iterates and obtain linear multistep methods e.g.

- 3^{rd} order exponential multistep method (Tokman, 2006):

$$y_{n+1} = y_n + \varphi_1(hJ_n)hf(y_n) + \frac{2}{3}\varphi_2(hJ_n)hR(y_{n-1})$$

Another strategy is to build intermediate stage values as in Runge-Kutta methods e.g.

- 4^{th} order exponential Runge-Kutta method (Rainwater & Tokman, 2016):

$$Y_1 = y_n + \frac{1}{9}\varphi_1\left(\frac{1}{9}hJ_n\right)hf(y_n)$$
$$Y_2 = y_n + \frac{1}{8}\varphi_1\left(\frac{1}{8}hJ_n\right)hf(y_n)$$
$$y_{n+1} = y_n + \varphi_1\left(hJ_n\right)hf(y_n) + \varphi_3(hJ_n)(1458\ hR(Y_1) - 1024\ hR(Y_2)) + \varphi_4(hJ_n)(-34992\ hR(Y_1) + 27648\ hR(Y_2))$$

One common feature of all these methods is the expressions involving products of φ functions with vectors. The major computational expense of an exponential integrator comes from the evaluation of these terms. Therefore, the fewer evaluations a method requires, the more efficient it will be. In the next section, we will show that a linear combination of the form

$$\sum_{i=1}^{p} \tau^i \varphi_i(\tau hJ_n)v_i \tag{A.5}$$

where τ is a scalar and v_i are vectors, can be computed for the cost of a single evaluation. Moreover, we will see that it is also possible to get this result for different values of τ at once.

A.2 Computing matrix exponential and related matrix functions

All of the exponential schemes require the computation of so-called φ functions of matrix arguments and their products with vectors. As mentioned before,

this is the most computationally expensive part of the time stepping algorithm. Many numerical methods are available to compute the exponential of a matrix (Moler & Van Loan, 1978). However, most of these methods are not adapted to large stiff matrices. For large matrices, numerical methods such as Taylor series methods (Bader, Blanes, & Casas, 2019; Al-Mohy & Higham, 2011), Chebychev methods (Abdulle, 2002; Bergamaschi, Caliari, & Vianello, 2003), Leja approximation (Caliari, Vianello, & Bergamaschi, 2004; Caliari et al., 2016) or Krylov subspace methods (Saad, 1992; Sidje, 1998; Gaudreault, Rainwater, & Tokman, 2018) can be used. In this section, we are going to present the main results of the *KIOPS* algorithm introduced in (Gaudreault, Rainwater, & Tokman, 2018) as it can be used to compute linear combination of these terms efficiently using an approximation in a Krylov subspace.

A.2.1 From φ functions to matrix exponential

The following theorem allows us to reduce the computation of a linear combinations of φ functions to the computation of a single matrix exponential time a vector.

Theorem A.2.1. *Let* $A \in \mathbb{R}^{N\times N}$, $V = [v_p, \dots, v_2, v_1] \in \mathbb{R}^{N\times p}$, $K = \begin{bmatrix} 0 & I_{p-1} \\ 0 & 0 \end{bmatrix} \in \mathbb{R}^{p\times p}$, $v = \begin{bmatrix} v_0, e_p \end{bmatrix}^\intercal \in \mathbb{R}^{N+p}$ *and* τ. *We define the augmented matrix*

$$\tilde{A} = \begin{bmatrix} A & B \\ 0 & K \end{bmatrix} \in \mathbb{R}^{(N+p)\times(N+p)} \tag{A.6}$$

and let $w = e^{\tau\tilde{A}}v$. *Then, the first* N *elements of the vector* w *are given by*

$$w(1:N) = \sum_{j=0}^{p} \tau^j \varphi_j(\tau A) v_j \tag{A.7}$$

The proof of this theorem can be found in (Al-Mohy & Higham, 2011). Using this result, it is possible to reduce the computation of any linear combination of the form eq. (A.5) to a single matrix exponential-vector product. Note that the matrix used in the matrix exponential is slightly bigger than the original. This extension has zero to little impact on the computation time for large-scale applications.

A.2.2 Krylov approximation of the matrix exponential

In this section, we will describe the most common methods for approximating the matrix exponential in a Krylov subspace. First, a subspace is built on which the matrix is projected. Then, the action of the matrix exponential on this subspace is computed. Finally, the solution is projected back onto the original space to get the final approximation. The subspace used for this work will be a Krylov subspace which is defined as:

$$K_m(A, v) = \text{span}\left\{v, Av, ..., A^{m-1}v\right\},$$

Building Krylov space: Arnoldi iteration

The first step of the approximation is to build a basis for the Krylov space. The vectors $A^k v$ become increasingly aligned as k increases, so the naive basis is not stable for numerical applications. For this reason, the Arnoldi iteration is used (see Algorithm 1).

Algorithm 1 Arnoldi process

Input: $A \in \mathbb{R}^{N\times N}$, $B \in \mathbb{R}^{N\times p}$, $V \in \mathbb{R}^{N+p\times m_{\max}+1}$, $m \in \mathbb{N}^+$
for $j = 1$ **to** m **do**
  $v_{j+1} = Av_j$
  for $i = 1$ **to** j **do**
    $h_{i,j} = v_i^\intercal \cdot v_{j+1}$
    $v_{j+1} = v_{j+1} - h_{i,j}\ v_i$
  end for
  $h_{i+1,j} = \|v_{j+1}\|$
  if $h_{i+1,j} \approx 0$ **then**
    break
  end if
  $v_{j+1} = \frac{1}{h_{i+1,j}}\ v_{j+1}$
end for
return $V_m = [v_1, ..., v_j]$, $H_m = h_{i,j}$

This procedure produces the matrix V_m corresponding to an orthonormal basis of K_m and the matrix H_m where m is the size of the Krylov space. It can be noted that lines 4 to 7 in Algorithm 1 correspond to the modified Gram–Schmidt algorithm. Therefore, we have $h_{i,j} = {v_i}^T A v_j$ or equivalently $H_m = {V_m}^T A V_m$. Then, H_m is the projection of the matrix A onto the Krylov subspace K_m.

Approximation in Krylov space

The second step is to approximate the action of the matrix exponential using the subspace created before. In (Saad, 1992), the author proves that for any polynomial function p_{m-1} of degree $\leq m-1$, we have:

$$p_{m-1}(\tau A)v = V_m p_{m-1}(\tau H_m){V_m}^T v = \beta V_m p_{m-1}(\tau H_m)e_1$$

where $\beta = ||v||$ and $e_1 = (1, 0, ..., 0)^T \in \mathbb{R}^m$

Moreover, the exponential can be expressed as: $e^x = \sum_{n=0}^{\infty} \frac{x^n}{n!}$. The terms of this series converging quickly to zero motivate the following approximation:

$$e^{\tau A}v \approx V_m \; e^{\tau H_m} \; {V_m}^T v = \beta \; V_m \; e^{\tau H_m} \; e_1$$

Following (Saad, 1992), the error made by this approximation can be expressed as:

$$e^{\tau A}v - \beta \; V_m \; e^{\tau H_m} \; e_1 = \tau\beta \; h_{m+1,m} \sum_{j=1}^{\infty} {e_m}^T \varphi_j(\tau H_m) \; e_1 (\tau A)^{j-1} v_{m+1}$$

For practical application, keeping only the first term of the series provides a good approximation of the error:

$$\left|\left| e^{\tau A}v - \beta \; V_m \; e^{\tau H_m} \; e_1 \right|\right| \approx \tau\beta \; h_{m+1,m} \; |{e_m}^T \varphi_1(\tau H_m) \; e_1|$$

The term $\varphi_1(\tau H_m) \; e_1$ can be computed by applying Theorem 1 again. The other terms are directly available. This error estimate can be used to select the size of the Krylov space to get an approximation within some tolerance.

A.2.3 Matrix exponential of a small matrix

The previous section presented the approximation in the Krylov space. However, it is still necessary to compute the matrix exponential of the smaller matrix H_m. Since $m << N$, $e^{\tau H_m}$ can be computed using any standard technique. (see (Higham, 2008) for a review). Here, we present a diagonal Padé approximation combined with the scaling and squaring method (Higham, 2005). A $[k/m]$ Padé approximation of a function f is a rational function r_{km} such that

$$r_{km}(x) = \frac{p_k(x)}{q_m(x)}$$

where p_k and $q_m$are polynomials of degree at most k and m respectively, $q_m(0) = 1$ and $f(x) - r_{km}(x) = O(x^{k+m+1})$. In the case $k = m$, the approximant is called a diagonal Padé approximation.

The matrix exponential e^A is well approximated with a diagonal Padé function when the norm of A is small. For this reason, the Padé approximation is combined with the scaling and squaring method.

The scaling and squaring method uses the property of the exponential: $e^A = \left(e^{A/\sigma}\right)^{\sigma}$. By taking σ to be a power of 2: $\sigma = 2^s, s \in \mathbb{N}$, the result can be computed efficiently by repeatedly squaring the matrix.

In summary, the algorithm to compute the exponential of the small matrix H_m is as follow:

1. $\sigma = 2^s$ is selected such that $||H_m||$ is sufficiently small.
2. $E = e^{H_m/\sigma}$ is computed using a diagonal Padé approximation.
3. The matrix E is squared s times to get the final result.

A.2.4 KIOPS optimizations

This section describes new improvements to the Krylov approximation introduced by the KIOPS algorithm (Gaudreault, Rainwater, & Tokman, 2018).

Incomplete orthogonalization

To build a Krylov basis of size m using the Arnoldi process presented above, it is necessary to evaluate m matrix-vector products and $\frac{m(m+1)}{2}$ dot products. Therefore, as m increases, the cost of the orthogonalization is increasing relative to the cost of the matrix product and can become dominant. This is even more true in the case of parallel code. The action of the matrix on a vector usually scales nicely with the number of threads but the dot product always requires some communication.

In (Gaudreault, Rainwater, & Tokman, 2018), the authors are showing that during the Arnoldi process it is enough to orthogonalize the new vector with respect to the last 2 previous vectors. This simple modification reduces the number of dot products from $\frac{m(m+1)}{2}$ to $2m$ while keeping the procedure stable.

Adaptive Krylov

Using incomplete orthogonalization allows reducing the computation cost of the Arnoldi process from $O(m^2)$ to $O(m)$. However, the computation of the matrix exponential in the Krylov space: $e^{\tau H_m}$ still requires $O(m^3)$ operations if the diagonal Padé approximation is used (due to the matrix inversion).

The idea of the adaptive Krylov method, originally presented in (Niesen & Wright, 2012), is to substep the computation of the matrix exponential by using the following property of the exponential:

$$e^A v = e^{\tau_k A} ... e^{\tau_2 A} e^{\tau_1 A} v \quad \text{if } \sum_i \tau_i = 1$$

Using this equation, it is possible to compute e^A by first computing $w_1 = e^{\tau_1 A} v$, then computing $w_2 = e^{\tau_2} A w_1$, and so on until τ_k. By choosing $\tau_i < 1$, we have $||\tau_i A|| < ||A||$ and therefore the approximation requires a smaller Krylov subspace. The goal is to choose the value τ_i such that it minimizes the total CPU time. In (Gaudreault, Rainwater, & Tokman, 2018), a strategy for choosing the values of τ_i is presented.

Appendix B

High-order numerical solutions to the shallow-water equations on the rotated cubed-sphere grid

B.1 The covariant shallow-water equations in 2+1 dimensions

Here, the (2+1)-dimensional shallow-water equations will be derived from the (3+1)-dimensional covariant form of the Euler equations when the fluid is externally forced by a gravitational potential Φ. Following (Charron, Zadra, & Girard, 2014),

$$\frac{\partial}{\partial t}\left(\sqrt{g}\rho\right) + \frac{\partial}{\partial x^j}\left(\sqrt{g}\rho u^j\right) = 0, \tag{B.1}$$

$$\frac{\partial u^i}{\partial t} + u^j\frac{\partial u^i}{\partial x^j} + \Gamma^i_{00} + 2\Gamma^i_{j0}u^j + \Gamma^i_{jk}u^j u^k = -h^{ij}\left(\frac{1}{\rho}\frac{\partial p}{\partial x^j} + \frac{\partial \Phi}{\partial x^j}\right) \tag{B.2}$$

describe the continuity and inviscid momentum equations ($i = 1, 2, 3$), respectively. Repeated Latin (spatial) indices are summed from 1 to 3, unless otherwise indicated. These equations hold under the classical assumptions of Newtonian space-time: time intervals are absolute, and space is also absolute. The symbol ρ represents the fluid's density field, p the pressure field, and $u^i \equiv dx^i/dt$ the velocity field components.

One may define an infinitesimal space interval dl in an inertial frame as $dl^2 = h_{\mu\nu}dx^\mu dx^\nu$, where repeated Greek indices are summed from 0 to 3 (0 being a time index), and an infinitesimal time interval $dx^0 = u^0 dt$ as $(dx^0)^2 = \delta^0_\mu \delta^0_\nu dx^\mu dx^\nu$, where δ^ν_μ is the Kronecker tensor and u^0 an arbitrary non-zero constant with units of velocity. The time unit may be chosen such that $u^0 = 1$ without loss of generality and without impacting the form of the equations. Therefore, $u^0 = 1$ is assumed from now on. A (3+1)-dimensional distance ds is written $ds^2 = dl^2 + (dx^0)^2 = g_{\mu\nu}dx^\mu dx^\nu = (h_{\mu\nu} + \delta^0_\mu \delta^0_\nu)dx^\mu dx^\nu$. Therefore, the contravariant tensor $h^{\mu\nu}$ is obtained from the (3+1)-dimensional contravariant metric tensor $g^{\mu\nu}$ (where $g^{\mu\nu}g_{\nu\alpha} = \delta^\mu_\alpha$):

$$h^{\mu\nu} = g^{\mu\alpha}g^{\beta\nu}h_{\alpha\beta} = g^{\mu\alpha}g^{\beta\nu}(g_{\alpha\beta} - \delta^0_\alpha \delta^0_\beta) = g^{\mu\nu} - g^{0\mu}g^{0\nu},$$

with $g^{00} = 1$ in Newtonian mechanics. This definition implies that $h^{\mu 0} = h^{0\mu}$ vanishes. Notice that in Newtonian space-time, there are in fact two tensors describing the geometry (for instance, $g_{\mu\nu}$ for space-time and $h^{\mu\nu}$ for space-only) because both spatial distances *dl and* temporal intervals dx^0 are absolute and invariant.

The Γ's in Eq. (B.2) are Christoffel symbols of the second kind: Γ^i_{00} corresponds to the centripetal acceleration, Γ^i_{j0} is associated with the local Coriolis acceleration, and Γ^i_{jk} with the nonlinear metric terms. They are obtained from the usual definition in a metric space-time:

$$\Gamma^\mu_{\alpha\beta} = \frac{1}{2}g^{\mu\nu}\left(\frac{\partial g_{\nu\alpha}}{\partial x^\beta} + \frac{\partial g_{\nu\beta}}{\partial x^\alpha} - \frac{\partial g_{\alpha\beta}}{\partial x^\nu}\right) = \frac{1}{2}h^{\mu\nu}\left(\frac{\partial g_{\nu\alpha}}{\partial x^\beta} + \frac{\partial g_{\nu\beta}}{\partial x^\alpha} - \frac{\partial g_{\alpha\beta}}{\partial x^\nu}\right). \quad \text{(B.3)}$$

The last equality in Eq. (B.3) stems from the fact that $\Gamma^0_{\alpha\beta}$ vanishes in Newtonian mechanics. The term $\sqrt{g}$ is the square root of the determinant of the covariant metric tensor $g_{\mu\nu}$. In Newtonian mechanics, it also corresponds to the inverse of the square root of the 3-dimensional determinant of h^{ij}. Details on this (3+1)-dimensional formalism are provided in (Charron, Zadra, & Girard, 2014).

Consider spherical geometry under the thin-shell (or shallow-atmosphere) approximation. The x^3 axis is taken to represent the radial direction from the origin of the coordinate system and is chosen to indicate a geometric distance from the center of the Earth. The contravariant metric tensor components $g^{\mu\nu}$—and therefore $g_{\mu\nu}$ and h^{ij}—become independent of the coordinate x^3 because $x^3 = a$, where

a is the mean radius of the Earth in spherical geometry, as explained in (Charron, Zadra, & Girard, 2014; Charron & Zadra, 2014). The Christoffel symbols must not be directly approximated using $x^3 = a$ but rather recalculated from Eq. (B.3) with the approximated metric tensors (this preserves conservation laws). It turns out that this choice of coordinates together with the thin-shell approximation implies that the components $\Gamma^i_{\mu 3}$ of the Christoffel symbols vanish, as shown in (Charron, Zadra, & Girard, 2014) (their Eqs. (117)–(118)), and that $h^{13} = h^{23} = 0$ and $h^{33} = 1$ (their Eq. (60)).

Applying the spherical geopotential approximation in a geopotential coordinate, it may be shown that

$$h^{ij}\frac{\partial \Phi}{\partial x^j} + \Gamma^i_{00} = \begin{cases} 0 & \text{for } i = 1, 2; \\ g_r & \text{for } i = 3, \end{cases} \tag{B.4}$$

where $g_r \approx 9.80616 \text{ m s}^{-2}$ is the constant effective gravitational acceleration at the surface of the Earth.

The "horizontal" (i.e. any spatial direction perpendicular to x^3) momentum equations become

$$\frac{\partial u^i}{\partial t} + u^j\frac{\partial u^i}{\partial x^j} + 2\Gamma^i_{j0}u^j + \Gamma^i_{jk}u^j u^k = -\frac{1}{\rho}h^{ij}\frac{\partial p}{\partial x^j} \tag{B.5}$$

for $i = 1, 2$, while the radial momentum equation leads to the hydrostatic balance:

$$\frac{\partial p}{\partial x^3} = -\rho g_r. \tag{B.6}$$

Recall that in Eqs. (B.1) and (B.5), all the metric terms (the Γ's, h^{ij} and $\sqrt{g}$) are independent of x^3 because the thin-shell approximation is being used.

To obtain the set of shallow-water equations, one makes the additional assumption that the fluid's density ρ is uniform and constant. Integrating Eq. (B.1) over x^3 from $x^3 = h_B(t, x^1, x^2)$, where h_B is prescribed, to $x^3 = h(t, x^1, x^2)$ while assuming that "horizontal" motion does not vary significantly over the depth $h - h_B$, one obtains

$$(h - h_B)\frac{\partial(\sqrt{g})}{\partial t} + \sqrt{g}\left(u^3(h) - u^3(h_B)\right) + (h - h_B)\frac{\partial}{\partial x^j}\left(\sqrt{g}u^j\right) = 0,$$

where in this case j is summed from 1 to 2. Writing $u^3(h) = dh/dt$ and $u^3(h_B) = dh_B/dt$, and expanding the operator $d/dt = \partial/\partial t + u^j \partial/\partial x^j$ (where j is summed from 1 to 2 because h is independent of x^3), one gets the shallow-water continuity equation in arbitrary "horizontal" coordinates:

$$\frac{\partial}{\partial t}\left(\sqrt{g}H\right) + \frac{\partial}{\partial x^j}\left(\sqrt{g}Hu^j\right) = 0, \tag{B.7}$$

where $H \equiv h - h_B$ is the thickness of the incompressible fluid and j is summed from 1 to 2.

The 2-dimensional shallow-water momentum equations follow from assuming that the pressure at $x^3 = h(t, x^1, x^2)$ is a constant p_0. Therefore from Eq. (B.6), $p = \rho g_r(h - x^3) + p_0$, and the "horizontal" pressure gradient in Eq. (B.5) becomes $\partial p/\partial x^j = \rho g_r \partial h/\partial x^j$ $(j = 1, 2)$.

A quasi-flux form may be obtained by adding Eq. (B.5) multiplied by $\sqrt{g}H$ and Eq. (B.7) multiplied by u^i:

$$\begin{aligned}\frac{\partial}{\partial t}\left(\sqrt{g}Hu^i\right) + \frac{\partial}{\partial x^j}\left(\sqrt{g}Hu^iu^j\right) = -2\sqrt{g}\,\Gamma^i_{j0}Hu^j - \sqrt{g}\,\Gamma^i_{jk}Hu^ju^k \\ -\frac{1}{2}\sqrt{g}\,g_r h^{ij}\frac{\partial H^2}{\partial x^j} - \sqrt{g}Hg_r h^{ij}\frac{\partial h_B}{\partial x^j},\end{aligned} \tag{B.8}$$

where $i = 1, 2$ and j, k are summed from 1 to 2. From the identity $\partial(\sqrt{g}h^{ij})/\partial x^j = -\sqrt{g}h^{jk}\Gamma^i_{jk}$, Eq. (B.8) may be rewritten as

$$\begin{aligned}\frac{\partial}{\partial t}\left(\sqrt{g}Hu^i\right) + \frac{\partial}{\partial x^j}\left(\sqrt{g}\left[Hu^iu^j + \frac{1}{2}g_r h^{ij}H^2\right]\right) = -2\sqrt{g}\,\Gamma^i_{j0}Hu^j \\ -\sqrt{g}\,\Gamma^i_{jk}\left(Hu^ju^k + \frac{1}{2}g_r h^{jk}H^2\right) - \sqrt{g}Hg_r h^{ij}\frac{\partial h_B}{\partial x^j}.\end{aligned} \tag{B.9}$$

Note that Eqs. (B.7) and (B.9) may be rewritten as a single expression in a (2+1)-dimensional formalism:

$$\frac{\partial}{\partial x^\nu}(\sqrt{g}\,T^{\mu\nu}) = -2\sqrt{g}\,\Gamma^\mu_{j0}T^{j0} - \sqrt{g}\,\Gamma^\mu_{jk}T^{jk} - \sqrt{g}Hg_r h^{\mu j}\frac{\partial h_B}{\partial x^j}, \tag{B.10}$$

where

$$T^{\mu\nu} = Hu^\mu u^\nu + \frac{1}{2}g_r h^{\mu\nu}H^2 \tag{B.11}$$

is the mass-momentum tensor for the shallow-water equations. Repeated Greek indices are summed from 0 to 2.

B.2 Metric terms associated with the rotated cubed-sphere grid

Consider a given panel p (out of 6 identical ones). Its coordinates x^1 and x^2 are great circles intersecting at right angle at the panel's center, which is assumed to be located at geographical longitude λ_p and geographical latitude ϕ_p. The coordinate line $x^1 = 0$ is assumed to be rotated clockwise by an angle α_p with respect to a geographical meridian (oriented northward) when looking at the panel's center directly from above. Define the quantities $X = \tan x^1$, $Y = \tan x^2$, and $\delta^2 = 1 + X^2 + Y^2$. The transformation from geographical longitude and latitude (λ, ϕ) to cubed-sphere coordinates (x^1, x^2) is performed via the relations

$$X = \frac{\sin(\lambda - \lambda_p)\cos\alpha_p + \sin\phi_p \cos(\lambda - \lambda_p)\sin\alpha_p - \tan\phi\cos\phi_p \sin\alpha_p}{\cos\phi_p\cos(\lambda - \lambda_p) + \tan\phi\sin\phi_p}, \quad \text{(B.12)}$$

$$Y = \frac{\sin(\lambda - \lambda_p)\sin\alpha_p - \sin\phi_p \cos(\lambda - \lambda_p)\cos\alpha_p + \tan\phi\cos\phi_p \cos\alpha_p}{\cos\phi_p\cos(\lambda - \lambda_p) + \tan\phi\sin\phi_p}. \quad \text{(B.13)}$$

Each panel of the cubed-sphere grid is a gnomonic projection from the 2-sphere to the plane tangent to the panel's center. For instance, a panel with $\lambda_p = 0$ and $\alpha_p = 0$ centered at the north pole ($\phi_p = \pi/2$) has $\tan x^1 = y/z$ and $\tan x^2 = -x/z$, where (x, y, z) are 3-dimensional Cartesian coordinates of the Euclidean space in which the 2-sphere is embedded.

Tensors and Christoffel symbols are transformed following the usual laws (see for instance Eqs. (6), (7) and (17) in (Charron, Zadra, & Girard, 2014)). It may be shown that the covariant metric tensor in 2+1 dimensions is

$$g_{00} = 1 + \frac{a^2}{\delta^2}\Omega^2\left(\delta^2 - [\sin\phi_p - X\cos\phi_p\sin\alpha_p + Y\cos\phi_p\cos\alpha_p]^2\right), \quad \text{(B.14)}$$

$$g_{01} = \frac{a^2}{\delta^2}\Omega(1 + X^2)\left(\cos\phi_p\cos\alpha_p - Y\sin\phi_p\right) = g_{10}, \quad \text{(B.15)}$$

$$g_{02} = \frac{a^2}{\delta^2}\Omega(1 + Y^2)\left(\cos\phi_p\sin\alpha_p + X\sin\phi_p\right) = g_{20}, \quad \text{(B.16)}$$

$$g_{11} = \frac{a^2}{\delta^4}(1 + X^2)^2(1 + Y^2), \quad \text{(B.17)}$$

$$g_{12} = -\frac{a^2}{\delta^4}XY(1 + X^2)(1 + Y^2) = g_{21}, \quad \text{(B.18)}$$

$$g_{22} = \frac{a^2}{\delta^4}(1 + X^2)(1 + Y^2)^2. \quad \text{(B.19)}$$

The spatial metric tensor h^{ij}, where $h^{ij}g_{jk} = \delta^i_k$, takes the form

$$h^{11} = \frac{\delta^2}{a^2(1+X^2)}, \tag{B.20}$$

$$h^{12} = \frac{XY\delta^2}{a^2(1+X^2)(1+Y^2)} = h^{21}, \tag{B.21}$$

$$h^{22} = \frac{\delta^2}{a^2(1+Y^2)}. \tag{B.22}$$

The term $\sqrt{g}$ is most simply calculated as the inverse of the square root of the determinant of h^{ij}:

$$\sqrt{g} = \frac{a^2(1+X^2)(1+Y^2)}{\delta^3}. \tag{B.23}$$

The Christoffel symbols of the second kind with mixed space-time components depend on the position of the panel because the rotation at constant angular velocity Ω around a given axis breaks the (space-time) symmetry. These Christoffel symbols associated with the Coriolis acceleration may be calculated based on a space-time tensor formalism, as in (Charron, Zadra, & Girard, 2014). They are

$$\Gamma^1_{01} = \frac{\Omega XY}{\delta^2}\left(\sin\phi_p - X\cos\phi_p\sin\alpha_p + Y\cos\phi_p\cos\alpha_p\right) = \Gamma^1_{10}, \tag{B.24}$$

$$\Gamma^1_{02} = -\frac{\Omega(1+Y^2)}{\delta^2}(\sin\phi_p - X\cos\phi_p\sin\alpha_p + Y\cos\phi_p\cos\alpha_p) = \Gamma^1_{20}, \tag{B.25}$$

$$\Gamma^2_{01} = \frac{\Omega(1+X^2)}{\delta^2}(\sin\phi_p - X\cos\phi_p\sin\alpha_p + Y\cos\phi_p\cos\alpha_p) = \Gamma^2_{10}, \tag{B.26}$$

$$\Gamma^2_{02} = -\frac{\Omega XY}{\delta^2}\left(\sin\phi_p - X\cos\phi_p\sin\alpha_p + Y\cos\phi_p\cos\alpha_p\right) = \Gamma^2_{20}. \tag{B.27}$$

The spatial components of the Christoffel symbols of the second kind are

$$\Gamma^1_{11} = \frac{2XY^2}{\delta^2}, \tag{B.28}$$

$$\Gamma^1_{12} = -\frac{Y(1+Y^2)}{\delta^2} = \Gamma^1_{21}, \tag{B.29}$$

$$\Gamma^1_{22} = 0, \tag{B.30}$$

$$\Gamma^2_{11} = 0, \tag{B.31}$$

$$\Gamma^2_{12} = -\frac{X(1+X^2)}{\delta^2} = \Gamma^2_{21}, \tag{B.32}$$

$$\Gamma^2_{22} = \frac{2X^2Y}{\delta^2}. \tag{B.33}$$

Because of spherical symmetry, the tensor $h^{\mu\nu}$, the quantity $\sqrt{g}$ and the spatial components of the Christoffel symbols of the second kind have the same form on all six panels.

The 2-dimensional cubed-sphere coordinates have an interesting property: it may be shown that the contraction $h^{jk}\Gamma^i_{jk}$ ($i = 1, 2$ and j, k are summed from 1 to 2) vanishes (see also (Ullrich, Jablonowski, & Van Leer, 2010)), thus allowing a simplification of Eq. (2.2). Note however that $h^{jk}\Gamma^i_{jk}$ does not vanish in general and that this term cannot be discarded from the equations of motion when other coordinates are employed.

A general rotation of the global cubed-sphere grid may be characterized by three angles which are here assumed to be $(\lambda_0, \phi_0, \alpha_0)$ associated with panel 0. The other angles $(\lambda_p, \phi_p, \alpha_p)$ associated with panels 1 to 5 are obtained as functions of these $(\lambda_0, \phi_0, \alpha_0)$. They are found to be

$$\lambda_1 = \tan^{-1}\left(\frac{\cos\lambda_0\cos\alpha_0 + \sin\lambda_0\sin\phi_0\sin\alpha_0}{\cos\lambda_0\sin\phi_0\sin\alpha_0 - \sin\lambda_0\cos\alpha_0}\right), \tag{B.34}$$

$$\phi_1 = -\sin^{-1}\left(\cos\phi_0\sin\alpha_0\right), \tag{B.35}$$

$$\alpha_1 = \tan^{-1}\left(\frac{\sin\phi_0}{\cos\phi_0\cos\alpha_0}\right), \tag{B.36}$$

$$\lambda_2 = \lambda_0 + \pi, \tag{B.37}$$

$$\phi_2 = -\phi_0, \tag{B.38}$$

$$\alpha_2 = -\alpha_0, \tag{B.39}$$

$$\lambda_3 = -\tan^{-1}\left(\frac{\cos\lambda_0\cos\alpha_0 + \sin\lambda_0\sin\phi_0\sin\alpha_0}{\sin\lambda_0\cos\alpha_0 - \cos\lambda_0\sin\phi_0\sin\alpha_0}\right), \tag{B.40}$$

$$\phi_3 = \sin^{-1}\left(\cos\phi_0\sin\alpha_0\right), \tag{B.41}$$

$$\alpha_3 = -\tan^{-1}\left(\frac{\sin\phi_0}{\cos\phi_0\cos\alpha_0}\right), \tag{B.42}$$

$$\lambda_4 = \tan^{-1}\left(\frac{\cos\lambda_0\sin\alpha_0 - \sin\lambda_0\sin\phi_0\cos\alpha_0}{-\cos\lambda_0\sin\phi_0\cos\alpha_0 - \sin\lambda_0\sin\alpha_0}\right), \tag{B.43}$$

$$\phi_4 = \sin^{-1}\left(\cos\phi_0\cos\alpha_0\right), \tag{B.44}$$

$$\alpha_4 = \tan^{-1}\left(\frac{\cos\phi_0\sin\alpha_0}{-\sin\phi_0}\right), \tag{B.45}$$

$$\lambda_5 = \tan^{-1}\left(\frac{\sin\lambda_0 \sin\phi_0 \cos\alpha_0 - \cos\lambda_0 \sin\alpha_0}{\cos\lambda_0 \sin\phi_0 \cos\alpha_0 + \sin\lambda_0 \sin\alpha_0}\right), \tag{B.46}$$

$$\phi_5 = -\sin^{-1}\left(\cos\phi_0 \cos\alpha_0\right), \tag{B.47}$$

$$\alpha_5 = \tan^{-1}\left(\frac{\cos\phi_0 \sin\alpha_0}{\sin\phi_0}\right). \tag{B.48}$$

Care must be taken when choosing the correct branch of the arctangent operators for various panels. In the special case where $\phi_0 = \alpha_0 = 0$, different formulas for λ_4, α_4, λ_5 and α_5 must be used due to the singularity at the poles. In this case, $\lambda_4 = 0 = \lambda_5$ and $\alpha_4 = -\lambda_0 = -\alpha_5$. Examples are provided in the accompanying implementation code (see § 2.7).

B.3 Consistency relations at the interfaces of panels

At the interface of two panels, consistency relations on tensor components may be established. Consider for instance the contravariant vector components $A^1_{(0)}$, $A^2_{(0)}$ on panel 0 and their relations to the contravariant vector components $A^1_{(1)}$, $A^2_{(1)}$ on panel 1. Note that due to spherical symmetry, the form of these relations is not altered by rotating the grid. One may then consider the case $\lambda_0 = \phi_0 = \alpha_0 = \phi_1 = \alpha_1 = 0$ and $\lambda_1 = \pi/2$. From $X_{(0)} = \tan\lambda$, $Y_{(0)} = \tan\phi/\cos\lambda$, $X_{(1)} = -1/\tan\lambda$ and $Y_{(1)} = \tan\phi/\sin\lambda$, the transformation laws from coordinates on panel 0 to coordinates on panel 1 lead to

$$\frac{\partial x^1_{(0)}}{\partial x^1_{(1)}} = 1, \tag{B.49}$$

$$\frac{\partial x^1_{(0)}}{\partial x^2_{(1)}} = 0, \tag{B.50}$$

$$\frac{\partial x^2_{(0)}}{\partial x^1_{(1)}} = \frac{Y_{(1)}(1+X^2_{(1)})}{X^2_{(1)}(1+Y^2_{(1)})} \quad \Rightarrow \quad \frac{\partial x^2_{(0)}}{\partial x^1_{(1)}} = \frac{2Y_{(1)}}{1+Y^2_{(1)}} \text{ at } X_{(1)} = -1, \tag{B.51}$$

$$\frac{\partial x^2_{(0)}}{\partial x^2_{(1)}} = -\frac{X_{(1)}(1+Y^2_{(1)})}{X^2_{(1)}+Y^2_{(1)}} \quad \Rightarrow \quad \frac{\partial x^2_{(0)}}{\partial x^2_{(1)}} = 1 \text{ at } X_{(1)} = -1. \tag{B.52}$$

These relations are used at the interface $X_{(1)} = -1$ to convert contravariant vector components from panel 1 to panel 0. A similar approach is used to convert covari-

ant vector components. This procedure may be performed at the 12 interfaces of the cubed-sphere grid.

The consistency relations for contravariant vector components $A^i_{(p)}$ and covariant vector components $A_{(p)i}$ on panel p at the 12 interfaces are provided below. At the interface of panels $(p,q) = (0,1)$, $(1,2)$, $(2,3)$, $(3,0)$, one obtains

$$A^1_{(p)} = A^1_{(q)}, \tag{B.53}$$

$$A^2_{(p)} = \frac{2Y}{1+Y^2} A^1_{(q)} + A^2_{(q)}, \tag{B.54}$$

$$A_{(p)1} = A_{(q)1} - \frac{2Y}{1+Y^2} A_{(q)2}, \tag{B.55}$$

$$A_{(p)2} = A_{(q)2}. \tag{B.56}$$

At the interface of panels $(4,0)$, one obtains

$$A^1_{(4)} = A^1_{(0)} - \frac{2X}{1+X^2} A^2_{(0)}, \tag{B.57}$$

$$A^2_{(4)} = A^2_{(0)}, \tag{B.58}$$

$$A_{(4)1} = A_{(0)1}, \tag{B.59}$$

$$A_{(4)2} = \frac{2X}{1+X^2} A_{(0)1} + A_{(0)2}, \tag{B.60}$$

with X defined on panel 0. At the interface of panels $(4,1)$, one obtains

$$A^1_{(4)} = -A^2_{(1)}, \tag{B.61}$$

$$A^2_{(4)} = A^1_{(1)} - \frac{2X}{1+X^2} A^2_{(1)}, \tag{B.62}$$

$$A_{(4)1} = -\frac{2X}{1+X^2} A_{(1)1} - A_{(1)2}, \tag{B.63}$$

$$A_{(4)2} = A_{(1)1}, \tag{B.64}$$

with X defined on panel 1. At the interface of panels $(4,2)$, one obtains

$$A^1_{(4)} = -A^1_{(2)} + \frac{2X}{1+X^2} A^2_{(2)}, \tag{B.65}$$

$$A^2_{(4)} = -A^2_{(2)}, \tag{B.66}$$

$$A_{(4)1} = -A_{(2)1}, \tag{B.67}$$

$$A_{(4)2} = -\frac{2X}{1+X^2} A_{(2)1} - A_{(2)2}, \tag{B.68}$$

with X defined on panel 2. At the interface of panels $(4,3)$, one obtains

$$A^1_{(4)} = A^2_{(3)}, \tag{B.69}$$

$$A^2_{(4)} = -A^1_{(3)} + \frac{2X}{1+X^2}A^2_{(3)}, \tag{B.70}$$

$$A_{(4)1} = \frac{2X}{1+X^2}A_{(3)1} + A_{(3)2}, \tag{B.71}$$

$$A_{(4)2} = -A_{(3)1}, \tag{B.72}$$

with X defined on panel 3. At the interface of panels $(5,0)$, one obtains

$$A^1_{(5)} = A^1_{(0)} + \frac{2X}{1+X^2}A^2_{(0)}, \tag{B.73}$$

$$A^2_{(5)} = A^2_{(0)}, \tag{B.74}$$

$$A_{(5)1} = A_{(0)1}, \tag{B.75}$$

$$A_{(5)2} = -\frac{2X}{1+X^2}A_{(0)1} + A_{(0)2}, \tag{B.76}$$

with X defined on panel 0. At the interface of panels $(5,1)$, one obtains

$$A^1_{(5)} = A^2_{(1)}, \tag{B.77}$$

$$A^2_{(5)} = -A^1_{(1)} - \frac{2X}{1+X^2}A^2_{(1)}, \tag{B.78}$$

$$A_{(5)1} = -\frac{2X}{1+X^2}A_{(1)1} + A_{(1)2}, \tag{B.79}$$

$$A_{(5)2} = -A_{(1)1}, \tag{B.80}$$

with X defined on panel 1. At the interface of panels $(5,2)$, one obtains

$$A^1_{(5)} = -A^1_{(2)} - \frac{2X}{1+X^2}A^2_{(2)}, \tag{B.81}$$

$$A^2_{(5)} = -A^2_{(2)}, \tag{B.82}$$

$$A_{(5)1} = -A_{(2)1}, \tag{B.83}$$

$$A_{(5)2} = \frac{2X}{1+X^2}A_{(2)1} - A_{(2)2}, \tag{B.84}$$

with X defined on panel 2. At the interface of panels $(5, 3)$, one obtains

$$A^1_{(5)} = -A^2_{(3)}, \tag{B.85}$$

$$A^2_{(5)} = A^1_{(3)} + \frac{2X}{1+X^2} A^2_{(3)}, \tag{B.86}$$

$$A_{(5)1} = \frac{2X}{1+X^2} A_{(3)1} - A_{(3)2}, \tag{B.87}$$

$$A_{(5)2} = A_{(3)1}, \tag{B.88}$$

with X defined on panel 3.

Consistency relations for higher-rank tensors may be obtained from these rules. For instance, if a symmetric contravariant second-rank tensor is considered at a point on an interface, then T^{00} remains identical on both panels; T^{0i} and T^{i0} transform as A^i; and T^{ij} as the product $A^i A^j$.

B.4 AUSM Riemann solver for the shallow-water equations

The Advection Upstream Splitting Method (AUSM) is a simple flux splitting method. It works by splitting the advective component of the flux from the pressure component. This appendix presents a brief overview of the elements that are relevant for its implementation in the context of the shallow-water equations. The reader is referred to (Liou & Steffen Jr, 1993) for more details.

From Eq. (B.11), define a tensor density $q^\mu \equiv \sqrt{g}\, T^{\mu 0} = \sqrt{g} H u^\mu$. One may express $\sqrt{g}\, T^{\mu\nu}$ as a function of q^μ:

$$\sqrt{g}\, T^{\mu\nu} = \frac{q^\mu q^\nu}{q^0} + \frac{1}{2\sqrt{g}}\, g_r h^{\mu\nu} \left(q^0\right)^2. \tag{B.89}$$

In AUSM, the components of the tensor densities appearing in the spatial derivative of Eq. (B.10) are splitted into

$$\sqrt{g}\, T^{\mu i} = M^i A^{\mu i} + P^{\mu i}, \tag{B.90}$$

(no sum over i) where $A^{\mu i} = q^\mu \sqrt{g} H a^i / q^0$ is the advective component, $M^i = u^i / a^i$ is the Froude number, a^i is the intrinsic gravity wave velocity (derived in Appendix B.5) and $P^{\mu i} = g_r h^{\mu i} \left(q^0\right)^2 / (2\sqrt{g})$ is the so-called pressure component.

The Froude number and pressure component are decomposed into a part evaluated on one side of the interface (denoted by a "+" superscript) and another part evaluated on the other side (denoted by a "−" superscript). Specifically,

$$M^i = (M^i)^+ + (M^i)^- \tag{B.91}$$

and

$$P^{\mu i} = (P^{\mu i})^+ + (P^{\mu i})^-, \tag{B.92}$$

where $(M^i)^\pm$ and $(P^{\mu i})^\pm$ are respectively defined by Eq. 6 and Eq. 8 of (Liou & Steffen Jr, 1993). Analogously to the original AUSM formulation, the Froude number is splitted a second time as follows

$$M^i = \max[0, (M^i)^+ + (M^i)^-] + \min[0, (M^i)^+ + (M^i)^-]. \tag{B.93}$$

Then the AUSM method with "double Froude number splitting" is

$$\sqrt{g}\,T^{\mu i} = \left[(\sqrt{g}\,T^{\mu i})^+ + (\sqrt{g}\,T^{\mu i})^-\right] \tag{B.94}$$

where

$$(\sqrt{g}\,T^{\mu i})^+ = \max[0, (M^i)^+ + (M^i)^-]\,(A^{\mu i})^+ + (P^{\mu i})^+, \tag{B.95}$$

$$(\sqrt{g}\,T^{\mu i})^- = \min[0, (M^i)^+ + (M^i)^-]\,(A^{\mu i})^- + (P^{\mu i})^-. \tag{B.96}$$

B.5 Gravity wave velocities

The components (ν, α) of the three flux Jacobian matrices $\mathcal{J}^\mu$ are defined as

$$(\mathcal{J}^\mu)^\nu{}_\alpha \equiv \frac{\partial(\sqrt{g}\,T^{\mu\nu})}{\partial q^\alpha}. \tag{B.97}$$

Note that the left-hand side of Eq. (B.10) may be rewritten with the flux Jacobian matrices as

$$\frac{\partial}{\partial x^\nu}(\sqrt{g}\,T^{\mu\nu}) = (\mathcal{J}^\mu)^\nu{}_\alpha \frac{\partial q^\alpha}{\partial x^\nu}. \tag{B.98}$$

They are written explicitly as $\mathcal{J}^0 = I_{3\times 3}$ (the identity matrix) and

$$\mathcal{J}^1 = \begin{pmatrix} 0 & 1 & 0 \\ g_r h^{11} H - u^1 u^1 & 2u^1 & 0 \\ g_r h^{12} H - u^1 u^2 & u^2 & u^1 \end{pmatrix}; \quad \mathcal{J}^2 = \begin{pmatrix} 0 & 0 & 1 \\ g_r h^{12} H - u^1 u^2 & u^2 & u^1 \\ g_r h^{22} H - u^2 u^2 & 0 & 2u^2 \end{pmatrix}. \quad \text{(B.99)}$$

The three eigenvalues of $\mathcal{J}^0$ correspond to $u^0 = 1$. The three eigenvalues of $\mathcal{J}^1$ are

$$u^1,\ u^1 \pm \sqrt{h^{11} g_r H}, \quad \text{(B.100)}$$

and those of $\mathcal{J}^2$ are

$$u^2,\ u^2 \pm \sqrt{h^{22} g_r H}. \quad \text{(B.101)}$$

Since the shallow-water equations are hyperbolic, the eigenvalues (B.100) and (B.101) are real and distinct. The intrinsic gravity wave velocities are deduced from these eigenvalues as $a^1 \equiv \sqrt{h^{11} g_r H}$ and $a^2 \equiv \sqrt{h^{22} g_r H}$.