Contributions to Efficient Nonstandard Time Integration with Application in Compressible Flows

Arjun Thenery Manikantan

UHASSELT
KNOWLEDGE IN ACTION

www.uhasselt.be
Hasselt University
Martelarenlaan 42 | BE-3500 Hasselt

UHASSELT
KNOWLEDGE IN ACTION

2025 | Faculty of Sciences

Doctoral dissertation submitted to obtain the degree of
Doctor of Sciences: Mathematics, to be defended by

Arjun Thenery Manikantan

DOCTORAL DISSERTATION

Contributions to Efficient Nonstandard Time Integration with Application in Compressible Flows

Promoter: Prof. Dr Jochen Schütz | UHasselt

UHASSELT

KNOWLEDGE IN ACTION

2025 | Faculty of Sciences

Doctoral dissertation submitted to obtain the degree of
Doctor of Sciences: Mathematics, to be defended by

Arjun Thenery Manikantan

DOCTORAL DISSERTATION

Contributions to Efficient Nonstandard Time Integration with Application in Compressible Flows

DSI DATA SCIENCE INSTITUTE

UHASSELT

D/2025/2451/121

Promoter: Prof. Dr Jochen Schütz| UHasselt

Contributions to Efficient Nonstandard Time Integration with Application in Compressible Flows

Doctoral dissertation submitted to obtain the degree of

Doctor of Sciences: Mathematics.

Arjun Thenery Manikantan

Promotor:	Prof. Dr. Jochen Schütz	| Hasselt University
Committee members:	Prof. Dr. Iuliu Sorin Pop	| Hasselt University
	Prof. Dr. Fred Vermolen	| Hasselt University
Jury members:	Prof. Dr.-Ing. Andrea Beck	| University of Stuttgart
	Dr. Robert Speck	| Jülich Supercomputing Centre
	Prof. Dr. Carina Bringedal	| Western Norway University
		| of Applied Sciences

In loving memory of Noolpuzhamma and Ammamma,

and dedicated to my Achan, Amma and Maalty

for their love and support.

Acknowledgments

We grow through the support of the remarkable people around us—those who guide us, help us learn from our mistakes, and inspire us to take bold steps in life. I have been fortunate to encounter many such individuals throughout my journey, each of whom has provided inspiration and support in one way or another. I would like to take this opportunity to express my heartfelt gratitude to all of them.

First and foremost, I would like to express my heartfelt gratitude to my promoter, Prof. Dr. Jochen Schütz, for his invaluable guidance, insightful feedback, and unwavering support throughout my research. I truly appreciate the time and attention he dedicated to my work. I am also thankful for his encouragement to engage in non-academic activities, which significantly broadened my professional experience. I have greatly enjoyed our discussions on research, music, and many other interesting topics. Thank you for providing an inspiring and supportive working environment over the past four years. I am also grateful to Dr. Jonas Zeifang for his valuable guidance and support during the early stages of my research, for helping me understand the codes, for introducing me to the VSC supercomputer, and for our fruitful collaborations.

I would like to express my sincere gratitude to Prof. Dr. Iuliu Sorin Pop and Prof. Dr. Fred Vermolen for being members of my doctoral committee. Thank you for always being available for discussions and for providing valuable guidance and suggestions throughout my PhD journey. Once again, thank you, Sorin, Fred and Jochen; your great sense of humor made our lunch and coffee breaks enjoyable and

memorable.

I would also like to thank Prof. Dr.-Ing. Andrea Beck, Dr. Robert Speck, and Prof. Dr. Carina Bringedal for kindly agreeing to be part of my jury. I am especially grateful to Andrea and the members of the Numerics Research Group at the Institute of Aerodynamics and Gas Dynamics for hosting me in Stuttgart, where I had the opportunity to learn more about FLEXI, a tool I used extensively in my research. I also thank Robert for the fruitful discussions during his visit to Hasselt and during the SDC days in Grenoble. I would also like to thank Dr. Afsaneh Moradi, Dr. Ange Pacifique Ishimwe, and Dr. Ruth Partzsch for their insightful discussions and valuable collaborations.

Furthermore, I would like to express my heartfelt gratitude to all my former and current colleagues in the Computational Mathematics Group and the Department of Mathematics for their invaluable support, assistance, and the many memorable moments we have shared throughout my journey. I would like to thank my first office mates, Dr. Stephan Lunowa and Dr. Sohely Sharmin, for the engaging discussions and enjoyable times together. I am particularly grateful to Sohely and her family—Mainul and their children, Mahdi and Aisha—for inviting me to celebrate special occasions with them and for all the delicious food. I would also like to thank Ayesha and Sabia (my Urdu ustanis!), and Dr. Koondanibha Mitra (Koondi), for the fun discussions and wonderful times we have shared, with a special thanks to Ayesha for her thoughtful gifts and for becoming my best friend. My appreciation extends to Jeremy and Wilbert for our great discussions, valuable suggestions, and for taking over the oral exams. I would also like to thank Vipul and Otavio for the engaging conversations, wonderful times, and enjoyable board game nights we shared. I would like to express my thanks to Prof. Dr. Inneke Van Nieuwenhuyse, Hoang-An, Eleni, Ansfried, Bram, Jesse, Sasan, Astrid, Filipe, George, and Yarne for the engaging discussions and pleasant breaks together.

I was fortunate to have a wonderful circle of friends in Hasselt who made my time there both enjoyable and memorable. I extend my sincere gratitude to Arish, Akshara, Shradha, and Velinda, as well as to Vishal, Meera, and little Jani, for their friendship, support and hospitality. I am also thankful to Manu, Leona, Sariga, Oviya,

and Keerthana for the enjoyable gatherings, and to Akshay, Yogesh, Umair, and other Xior friends for their cheerful company. My appreciation extends to my badminton friends — Giri, Sownder (my tournament partner), Partha, Vedant, Raju, Akshat, Yagnitha, Ganesh, and Mihir — and to the other members of the Herkse Badminton Club, especially Arno, Dirk, Rita, Marc, Kan, and Willy, for the engaging matches. I am equally grateful to Digvijay, Souvik, Lisa, Shameer, the PhD council members, and my volleyball teammates for their support and pleasant interactions. Finally, I wish to thank Suresh, Faran, Raza Ali, Ihtesham, Santhosh, Sujith, Priya, Harish, Jeffy, Abhijith, and Keerthi for their good company, and my piano teachers, Ellen and Liesbeth, whose guidance was both inspiring and restorative.

Having many close friends nearby greatly contributed to making my years in Europe memorable. I am grateful to Isha and Siddiq, and Saeed and Renna for the memorable moments we shared and the delightful meals. I also extend my appreciation to Harith, Ranjithettan, and Soniaji for the trips, the laughter, and your consistent support—especially Harith, for the wonderful days in Utrecht, jamming and cooking together. I sincerely cherish the time spent with Mitra during our weekends in LoLaN, filled with cooking, games and road trips. Thank you all for the unforgettable days and the dramatic card game climaxes! I am also grateful to Sruthi for her encouragement and all the wonderful moments. I would like to acknowledge Swetha, Lakshmi, Arya, Alwin, PS, Akhilesh, Ashwanth, Mirzana, Sandeepettan, Ashwin, and all my friends in the Netherlands for the good times we shared.

I would like to extend my sincere thanks to my IISER friends—particularly Rashad, Ajmal, Thampan, Varma, and Mymu—for their support and for making even challenging times in Thiruvananthapuram both enjoyable and memorable. I would also like to acknowledge Sarath, Kunji, Jyothish, KM, DV, Alex, Devu, Paru, Appu, Revathi, Joshyakka, and many others for their friendship and for being an unforgettable part of my journey. I wish to express my heartfelt gratitude to Dr. K. R. Arun, Prof. Rajeev N. Kini, Dr. Dharmatti Sheetal, and all other faculty at IISER for their guidance and support throughout my studies and beyond.

My time at Amrita University provided the opportunity to meet a remarkable group of people who have become close friends and an important part of my life. I

sincerely thank Kiran, Bintuettan, Ranjith, Jeevan, Sreedeep, Kripa, Dhaneesh bro, Prajeeshettan, Prajishechi, and Sandhya for their care and unwavering support. I would also like to express my gratitude to Pavithra, Anju, Aiswarya, Pramod, Sarika, Roshima, Jyothirmayi, Ashokan uncle, Bindu aunty, and all other colleagues and students in the Department of Mathematics for their encouragement and cooperation.

I would like to express my gratitude to my friends and teachers from school, who have played a significant role in helping me achieve important milestones in my life. I am especially grateful to my best friend, Noel, whose influence on me has been profound—no words can fully convey my appreciation—and to Ornella for her friendship and support. A special thanks to Paul Sir and Soly aunty, Noel's parents, and to my school principal, George Sir, for their encouragement, guidance, and kindness during pivotal moments in my life. I am particularly thankful to Sanoop, whose friendship of more than twenty-five years I deeply cherish, for his constant presence and the countless memories we have shared. I would also like to extend my heartfelt gratitude to Sanoop's father, the late Subhash maaman, for the invaluable support and care he provided. I would also like to mention my classmates—Sachin, Aswin, Rahul, Joel, Salman, Vipin, and others—for their friendship and the memorable moments we shared during school.

No words can fully capture the love, support, and inspiration my family has given me throughout my life. I would like to express my heartfelt gratitude to my father, Manikantan (Achan), my mother, Visalakshy (Amma), and my sister, Ajna (Maalty), for their unwavering love, encouragement, and support. I also wish to fondly remember my late grandmothers, Lakshmi (Noolpuzhamma) and Kalyani (Ammamma), whose love and care continue to inspire me, despite the irreplaceable void their absence has left in my life. I am sincerely grateful to my brother-in-law, Sumesh, for his constant support, and to my niece, Aditi, whose presence continues to bring immense joy and happiness to our family. Finally, I extend my heartfelt gratitude to my grandfathers, the late Appuchetty (Achachan) and Narayanan (Noolpuzhachan), my uncles and aunts—especially Surendran (Surumaaman) and Rajitha, Sajeevan (Kuttamaaman) and Pavithra—and my cousins for their love and constant support. I also fondly remember the late Veluthayyan, whose friendship, stories, and support to our

family will always be remembered with gratitude.

Finally, I would like to express my sincere gratitude to Hasselt University, the Special Research Fund (BOF) at Hasselt University (project no. BOF21KP12), the Data Science Institute (DSI), the Computational Mathematics group (CMAT) and the Doctoral School of Sciences & Technology for providing the necessary facilities, support, and resources throughout the past four years. I am especially thankful to the Doctoral Schools of Hasselt University for awarding me the mobility grant for my research stay at the University of Stuttgart. Additionally, I would like to acknowledge the Vlaams Supercomputer Centrum (VSC), funded by the Research Foundation - Flanders (FWO) and the Flemish Government, for providing the necessary computing resources.

During the preparation of this thesis, I used Grammarly solely to improve sentence structure and language clarity, and not for text generation. After using this tool, I thoroughly reviewed and edited the content myself, and I take full responsibility for the final version of this thesis.

(അർജുൻ തെനേരി മണികണ്ഠൻ)

Arjun Thenery Manikantan

Hasselt, December 2025

Contents

List of Publications

This thesis is based on the following peer-reviewed publications and preprints:

- Paper I : Zeifang, J., Thenery Manikantan, A., & Schütz, J. (2023). Time parallelism and Newton-adaptivity of the two-derivative deferred correction discontinuous Galerkin method. Applied Mathematics and Computation, 457, 128198. https://doi.org/10.1016/j.amc.2023.128198

- Paper II : Thenery Manikantan, A., & Schütz, J. (2024). Multi-step Hermite-Birkhoff predictor-corrector schemes. Applied Numerical Mathematics, 205, 281–295. https://doi.org/10.1016/j.apnum.2024.07.011

- Paper III: Thenery Manikantan, A., Zeifang, J., & Schütz, J.(2025) On the stability of two-derivative time discretizations. UHasselt Computational Mathematics Preprint, Hasselt University. https://www.uhasselt.be/media/4gbfm0hf/up2302.pdf

- Paper IV : Schütz, J., Ishimwe, A., Moradi, A., & Thenery Manikantan, A.(2024) A class of multirate multiderivative schemes. UHasselt Computational Mathematics Preprint, Hasselt University. (Major revision). https://www.uhasselt.be/media/5fopnoey/up2403.pdf

- Paper V : Thenery Manikantan, A., & Schütz, J.(2025) Two-derivative schemes for low-Mach flows. UHasselt Computational Mathematics Preprint, Hasselt University. (Accepted for publication in Communications in Computational Physics). https://www.uhasselt.be/media/hzonmb0o/paper.pdf.

Chapter 1

Introduction

The world has witnessed a tremendous improvement in science and technology over the past two centuries[1]. Ordinary differential equations (ODEs) and partial differential equations (PDEs) are among the most critical analytical tools facilitating this achievement. The history of ODEs and PDEs traces back to the late 1600s when eminent scientists like Newton and Leibniz laid the groundwork for the essential ingredients. Thereafter, it was a handy tool in providing mathematical frameworks to model and solve several complex physical phenomena.

At their core, differential equations are the mathematical language that describes change—capturing the rhythm of the universe itself, from the grand motion of celestial bodies to the delicate splitting of chromosomes within living cells. Thus, they provide a powerful perspective for exploring and understanding the universe.

In the present world, the applications of differential equations span a wide range of fields. In engineering, they are used, for example, to simulate airflow around aircraft wings [1, 2, 3], design vehicles with optimal fluid dynamic properties [4, 5, 6], and construct buildings and skyscrapers [7, 8] that can withstand various natural forces. In biology and medicine, differential equations help model blood circulation [9, 10, 11, 12], population dynamics [13, 14, 15], the spread of diseases [16, 17, 18], and the growth of tumors [19, 20, 21, 22, 23]. They are also used to develop tools for

[1]The Industrial Revolution began in the late 18th century

predicting the weather [24, 25, 26, 27, 28, 29] and assessing the intensity of natural disasters [30, 31, 32, 33]. Differential equations are used in economics and social sciences to model economic growth, inflation, market fluctuations, etc [34, 35, 36]. When it comes to energy exploration, they are inevitable in modeling fuel extraction methods [37, 38, 39], windmills [40], and hydroelectric power stations [41]. Overall, differential equations are essential for understanding and predicting changes in both natural [42, 43, 44, 45, 46] and human-made [47, 48, 49, 50, 51] systems. The references cited above represent only a small fraction of the extensive and diverse applications of differential equations across various fields.

Fig. 1.1 shows the image of the landslide that happened in the Wayanad district of Kerala, India, on July 30, 2024. The disaster has resulted in 254 fatalities, 397 injuries, and 118 people reported missing. Approximately 86,000 square meters of land had moved down the hill. More technical details about the Wayanad landslide can be found in [52]. Modeling such a phenomenon can result in an extensive system of PDEs with many physical constraints. For instance, the fully dynamical model equations for erosive landslides presented in [33] can be cast into the conservative form

$$\mathbf{w}_t + \nabla \cdot \mathbf{F}(\mathbf{w}) = \mathbf{S}(\mathbf{w}).$$

Solving the above equations numerically is highly computationally expensive. As it is a life-endangering event, solving and predicting the possible dangers must be done quickly. So, this highlights the requirement for numerical methods that can provide highly accurate solutions with minimal wall-clock time. The explicit formulation of the above equation is an extended version of the shallow water equations, as presented in [33, Eqs. 35–37]. The detailed equations are not presented here so as not to distract the reader from the main focus of this thesis. However, we present numerical approaches for solving isentropic equations (see Eq.(1.17)), which have a structure similar to the shallow water equations in the case $\mathbf{S}(\mathbf{w}) = 0$. More details are given in the upcoming sections.

In general, making a prediction or deriving information from such models requires solving the governing equations. In most cases, the resulting differential equations are too complex to yield a closed-form solution. Hence, a high precision numerical ap-

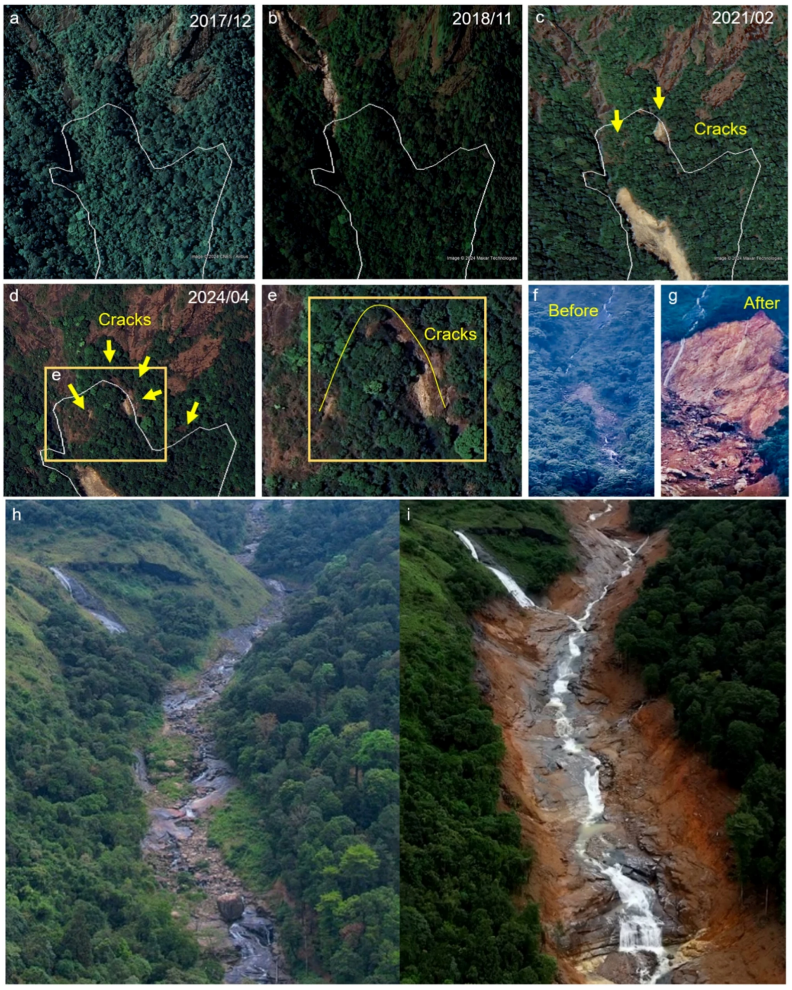


Figure 1.1: Satellite images and aerial views of the Chooralmala landslide source area in Wayanad district, Kerala, before and after the landslide on July 30, 2024. Reproduced with permission from Springer Nature. Yunus, A. P., Sajinkumar, K. S., Gopinath, G., *et al.* "Chronicle of destruction: the Wayanad landslide of July 30, 2024", *Landslides* (2025) **22**:1891–1908. © Springer Nature 2025.

proximation becomes essential. However, certain physical phenomena demand lower computational times to enable timely predictions of their outcomes. Examples include real-time simulations in climate modeling, to predict the spread of natural disasters, and more.

In this thesis, we study numerical methods for time-dependent equations. To provide an overview of existing time-stepping methods, we consider the following general form of an ODE for $t \in (0, T_{\text{end}}]$:

$$\frac{\mathrm{d}\mathbf{w}}{\mathrm{d}t} = f(\mathbf{w}), \tag{1.1}$$

where $\mathbf{w} \in \mathbb{R}^m$ is the unknown variable and $f : \mathbb{R}^m \to \mathbb{R}^m$ for $m \in \mathbb{N}$. The initial condition is given by

$$\mathbf{w}(0) = \mathbf{w}_0.$$

In the case of PDEs, spatial discretization is typically applied first, resulting in a large system of ODEs. This approach is known as the method of lines (MOL) [53]. In practice, the spatially discretized systems are obtained using various methods such as finite difference [54, 55, 56], finite volume [57, 58, 59, 60, 61], finite element [62, 63, 64], discontinuous Galerkin [65, 66, 67] methods and more. Therefore, the above formulation (1.1) serves as a general representation for both systems of ODEs and spatially discretized PDEs. The spatial discretization methods used in this thesis are described in Sec. 1.2.

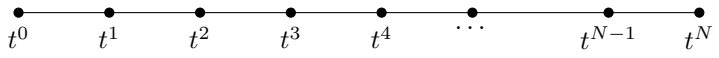
There are also space-time methods [68, 69, 70, 71, 72] which, unlike the MOL, solve the PDEs by discretizing both space and time simultaneously. These methods offer several advantages: they allow for straightforward parallelization [73] due to the presence of a single global system, enable simultaneous adaptivity [74, 73] in both space and time, simplify the treatment of problems involving moving geometries and more. However, space-time methods are more complex, computationally intensive and have higher memory demands due to their coupled space-time discretization approach. Therefore, these methods are used less often.

1.1 Background on standard timestepping methods

As mentioned in the general introduction, if the differential equation under consideration does not admit a closed-form solution, the only option is to find an approximate solution. Consider the ODE given in Eq. (1.1). The basic steps that any timestepping

method follows to approximate the solution at time $t = T_{\text{end}}$ are:

1. Choose discrete time nodes in $[0, T_{\text{end}}]$, with $t^0 = 0$ and $t^N = T_{\text{end}}$. The time



   nodes can be either uniformly or non-uniformly spaced. The time step size is defined as

$$\Delta t_n := t^n - t^{n-1}, \quad \text{for} \quad 1 \leq n \leq N.$$

   For simplicity of presentation, we assume that the nodes are uniformly spaced. In that case, the constant time step size is given by $\Delta t = \frac{T_{\text{end}}}{N}$. If variable time step sizes are used in any of the upcoming sections, it will be explicitly indicated.

2. Approximate the solution at each time instance t^n for $1 \leq n \leq N$ using the available information at previous time instances. Let **TimeStep** denotes the sequences of steps involved in approximating the solution at t^n. Then the procedure is given by

Input: Initialization $\mathbf{w}^0 = \mathbf{w}_0$
for $n \leftarrow 0$ **to** $(N-1)$ **do**
  $\mathbf{w}(t^{n+1}) \approx \mathbf{w}^{n+1} = \textbf{TimeStep}(\mathbf{w}^{n+1}, \mathbf{w}^n, \mathbf{w}^{n-1}, \dots)$
end
Output: Approximate solution $\mathbf{w}^N$

Several classifications of the timestepping methods are possible depending on the dependencies and structure of the **TimeStep** used. In terms of their dependency on the solution at different time levels, timestepping schemes are broadly categorized as explicit or implicit.

Before proceeding further, we introduce some frequently used definitions related to time-stepping schemes.

Definition 1 (Order of accuracy)**.** *Let the error between the approximated solution*

Explicit Methods	**Implicit Methods**
⇒ Use only information from previous time levels	⇒ Use information from previous and the current time levels.
⇒ $\mathbf{w}^{n+1} = \mathbf{TimeStep}(\mathbf{w}^n, \mathbf{w}^{n-1}, \dots)$	⇒ $\mathbf{w}^{n+1} = \mathbf{TimeStep}(\mathbf{w}^{n+1}, \mathbf{w}^n, \dots)$
⇒ E.g: Euler scheme [75] $\mathbf{w}^{n+1} = \mathbf{w}^n + \Delta t f(\mathbf{w}^n)$	⇒ E.g: Backward Euler scheme [75, 76] $\mathbf{w}^{n+1} = \mathbf{w}^n + \Delta t f(\mathbf{w}^{n+1})$
⇒ No algebraic equations to solve	⇒ Involves solving algebraic equations

Table 1.1: Comparison of explicit and implicit time-stepping methods.

$\mathbf{w}^N$ *and the exact solution* $\mathbf{w}(T_{end})$ *at time* $T_{end} = \Delta t N$ *be given by*

$$E(\Delta t) := \left\| \mathbf{w}(T_{end}) - \mathbf{w}^N \right\|,$$

for a given norm $\|\cdot\|$*. A time-stepping scheme is said to have order* p*, if*

$$E(\Delta t) = \mathcal{O}(\Delta t^p).$$

The order of accuracy of the schemes, Euler and the backward Euler, given in Tab. 1.1 is one. In the upcoming sections, we review several time-stepping methods, classified according to their structure, the information used from previous time levels and stages, the manner in which the function f and its derivatives are incorporated into the scheme, the ease of extending the method to higher orders of accuracy, and other relevant factors.

1.1.1 Multi-Derivative Runge–Kutta Methods

Runge–Kutta Methods

A large class of classical time-stepping schemes can be expressed as Runge-Kutta (RK) methods, each defined by a corresponding Butcher tableau. When comparing time-stepping schemes such as Euler's method, Runge–Kutta (RK) methods distinguish themselves by utilizing information from intermediate stages $\{t_1^n, t_2^n, t_3^n, \cdots, t_s^n\}$, which allows them to achieve higher-order accuracy. The stage values corresponding to the intermediate time levels $t_j^n = t^n + c_j \Delta t$, are evaluated by

$$\mathbf{w}^{(i)} = \mathbf{w}^n + \Delta t \sum_{j=1}^{s} a_{ij} f(\mathbf{w}^{(j)}), \quad \text{for } 1 \le i \le s,$$

and the solution is then updated using

$$\mathbf{w}^{n+1} = \mathbf{w}^n + \Delta t \sum_{i=1}^{s} b_i f(\mathbf{w}^{(i)}).$$

The associated Butcher tableau is represented as

$$\begin{array}{c|c} c & A \\ \hline & b^T \end{array} \equiv \begin{array}{c|ccc} c_1 & a_{11} & \cdots & a_{1s} \\ c_2 & a_{21} & \cdots & a_{2s} \\ \vdots & \vdots & \ddots & \vdots \\ c_s & a_{s1} & \cdots & a_{ss} \\ \hline & b_1 & \cdots & b_s \end{array},$$

in fact it defines the method.

Extension to Multi-Derivative Runge–Kutta Methods

Multi-Derivative Runge–Kutta (MDRK) methods generalize classical Runge–Kutta (RK) schemes by incorporating higher derivatives of the solution into the time-stepping process. This extension enables higher-order accuracy or enhanced stability without increasing the number of stages, at the cost of evaluating additional deriva-

tives. Classical RK methods are recovered as a special case when only the first derivative is used.

Since higher-order time derivatives are required for constructing MDRK schemes, they must be evaluated from the original differential equation (1.1). Starting from the initial equation,

$$\frac{\mathrm{d}\mathbf{w}}{\mathrm{d}t} = f(\mathbf{w}) =: f^{(0)}(\mathbf{w}),$$

the second time derivative is obtained by differentiating $f^{(0)}(\mathbf{w})$ with respect to time

$$\frac{\mathrm{d}^2\mathbf{w}}{\mathrm{d}t^2} = \frac{\mathrm{d}f^{(0)}(\mathbf{w})}{\mathrm{d}t} = f'(\mathbf{w})f(\mathbf{w}) =: f^{(1)}(\mathbf{w}),$$

where $f'(\mathbf{w})$ represents the Jacobian $\frac{\mathrm{d}f}{\mathrm{d}\mathbf{w}}$. Similarly, the third time derivative is obtained by differentiating $f^{(1)}(\mathbf{w})$

$$\frac{\mathrm{d}^3\mathbf{w}}{\mathrm{d}t^3} = \frac{\mathrm{d}f^{(1)}(\mathbf{w})}{\mathrm{d}t} = f''(\mathbf{w})(f(\mathbf{w}))^2 + (f'(\mathbf{w}))^2 f(\mathbf{w}) =: f^{(2)}(\mathbf{w}).$$

Recursively, the r-th time derivative ($r > 1$) can be expressed as

$$\frac{\mathrm{d}^r\mathbf{w}}{\mathrm{d}t^r} = \frac{\mathrm{d}}{\mathrm{d}t} f^{(r-2)}(\mathbf{w}) =: f^{(r-1)}(\mathbf{w}).$$

Using the derivatives defined above, an r-derivative, s-stage MDRK scheme updates the solution as

$$\mathbf{w}^{n+1} = \mathbf{w}^n + \sum_{d=1}^{r} \Delta t^d \sum_{i=1}^{s} b_i^{(d)} f^{(d-1)}(\mathbf{w}^{(i)}),$$

where each stage value $\mathbf{w}^{(i)}$ is computed via:

$$\mathbf{w}^{(i)} = \mathbf{w}^n + \sum_{d=1}^{r} \Delta t^d \sum_{j=1}^{s} a_{ij}^{(d)} f^{(d-1)}(\mathbf{w}^{(j)}), \quad \text{for } 1 \le i \le s.$$

This method is compactly represented by a generalized Butcher tableau:

$$\begin{array}{c|c|c|c|c} c & A^{(1)} & A^{(2)} & \cdots & A^{(r)} \\ \hline & b^{(1)T} & b^{(2)T} & \cdots & b^{(r)T} \end{array}$$

A few examples for the MDRK scheme are:

- Classical Runge–Kutta (RK) methods correspond to $r = 1$, using only the first derivative $f(u)$.
  - Explicit RK (ERK) if $a_{ij} = 0$ for all $j \geq i$ (no implicit dependencies).
  - Diagonally Implicit RK (DIRK) if $a_{ii} \neq 0$ and $a_{ij} = 0$ for $j > i$.
  - Fully Implicit RK (FIRK) if there exist $a_{ij} \neq 0$ for both $j < i$ and $j \geq i$.

  Refer to [75, 76, 77, 78, 79] for examples of RK schemes. Examples of DIRK and FIRK schemes are given in Tables 1.2 and 1.3, respectively.

$$\begin{array}{c|cc} \frac{1}{2}+\frac{1}{2\sqrt{3}} & \frac{1}{2}+\frac{1}{2\sqrt{3}} & 0 \\ \frac{1}{2}-\frac{1}{2\sqrt{3}} & -\frac{1}{\sqrt{3}} & \frac{1}{2}+\frac{1}{2\sqrt{3}} \\ \hline & \frac{1}{2} & \frac{1}{2} \end{array}$$

Table 1.2: Butcher tableau for a 2-stage, third-order DIRK method [80].

$$\begin{array}{c|cc} \frac{1}{3} & \frac{5}{12} & -\frac{1}{12} \\ 1 & \frac{3}{4} & \frac{1}{4} \\ \hline & \frac{3}{4} & \frac{1}{4} \end{array}$$

Table 1.3: Butcher tableau for the 2-stage, third-order Radau IIA method [76].

- The r-th order explicit Taylor Methods [75, 81] are given by

$$\mathbf{w}^{n+1} = \mathbf{w}^n + \sum_{k=1}^{r} \frac{\Delta t^k}{k!} f^{(k-1)}(\mathbf{w}^n).$$

- A class of special explicit two-derivative Runge-Kutta (TDRK) schemes is presented in [82]. These schemes were constructed to require only one evaluation of the function $f^{(0)}$ per step.

- The strong stability preserving (SSP) multi-derivative schemes. An example of a one-stage second-order implicit multiderivative SSP [83] scheme is

$$\mathbf{w}^{n+1} = \mathbf{w}^n + \Delta t\, f^{(0)}(\mathbf{w}^{n+1}) - \frac{\Delta t^2}{2} f^{(1)}(\mathbf{w}^{n+1}).$$

  Consult [84, 85, 86, 87, 88, 89, 90, 83] for more multi-derivative RK schemes.

- Two-derivative Hermite–Birkhoff collocation schemes (HBRK) [91, 92]. These schemes are stiffly accurate, satisfying

$$b^{(d)T} = A^{(d)}(s,:), \quad \text{for all } 1 \le d \le r,$$

  which implies $u_{n+1} = u_s$. These schemes are constructed by fitting a Hermite-Birkhoff polynomial through the time instances $\{c_1\Delta t, c_2\Delta t, \ldots, c_s\Delta t\}$ and then integrating the result from t^n to t^{n+1}. A sixth-order, three-stage HBRK scheme [92] is shown in Tab. 1.4.

0	0	0	0	0	0	0
$\frac{1}{2}$	$\frac{101}{480}$	$\frac{8}{30}$	$\frac{55}{2400}$	$\frac{65}{4800}$	$-\frac{25}{600}$	$-\frac{25}{8000}$
1	$\frac{7}{30}$	$\frac{16}{30}$	$\frac{7}{30}$	$\frac{5}{300}$	0	$-\frac{5}{300}$
	$\frac{7}{30}$	$\frac{16}{30}$	$\frac{7}{30}$	$\frac{5}{300}$	0	$-\frac{5}{300}$

Table 1.4: Sixth-order HBRK scheme with three stages [92].

The classical RK schemes are well-studied and widely used due to their simplicity, flexibility, and efficiency. However, higher-order RK schemes often require several stages, which can significantly affect performance, especially when the function f is expensive to evaluate. This is where MDRK schemes offer advantages, as they improve accuracy by incorporating higher-order derivatives instead of increasing the number of stages. For instance, [93] demonstrated the use of Taylor methods of order

20 in astrodynamics and celestial mechanics, highlighting their improved accuracy. Although the addition of higher derivatives provides flexibility in improving stability (both linear and nonlinear) and accuracy for MDRK schemes, implementing them for solving PDEs can be challenging.

We utilized multi-derivative collocation schemes [91, 92] to design higher-order predictor-corrector methods, as elaborated in [94, Paper I] and [95, Paper II]. Furthermore, we investigated the stability characteristics of SSP schemes [83] and applied these methods to solve the Navier-Stokes equations using a DGSEM spatial discretization [96, Paper III].

1.1.2 General Linear Methods

When it comes to improving the convergence order of timestepping schemes, it is possible in RK methods by adding more stages to the scheme. An alternative way is to consider multi-step timestepping schemes. If the time stepping considers contributions from more than one previous time instance

$$\mathbf{w}^{n+1} = \mathbf{TimeStep}(\mathbf{w}^{n+1}, \mathbf{w}^{n}, \mathbf{w}^{n-1}, \dots, \mathbf{w}^{n+1-k}),$$

then it is termed to be a multistep timestepping scheme. In particular, the above one is a k-step timestepping scheme as it uses information from previous k time instances. The linear multistep methods (LMMs) are those that take the form

$$\sum_{j=0}^{k} \alpha_j \mathbf{w}^{n+1-j} = \Delta t \sum_{j=0}^{k} \beta_j f(\mathbf{w}^{n+1-j}).$$

where the coefficients α_j and β_j determines the method. A three-step fourth-order Adams–Bashforth method [75]

$$\mathbf{w}^{n+1} - \mathbf{w}^{n} = \Delta t \left(\frac{23}{12} f(\mathbf{w}^{n}) - \frac{4}{3} f(\mathbf{w}^{n-1}) + \frac{5}{12} f(\mathbf{w}^{n-2}) \right)$$

is an example of an explicit LMM.

By combining the concept of intermediate stages from Runge-Kutta (RK) methods

with the multistep nature of linear multistep methods (LMMs), a broader class of methods can be constructed. These schemes are known as General Linear Methods (GLMs). The GLMs are represented using four matrices $A \in \mathbb{R}^{s\times s}$, $U \in \mathbb{R}^{s\times k}$, $B \in \mathbb{R}^{k\times s}$, and $V \in \mathbb{R}^{k\times k}$ as

$$\begin{bmatrix} \mathbf{W} \\ \mathbf{w}^{[n+1]} \end{bmatrix} = \begin{bmatrix} A & U \\ B & V \end{bmatrix} \begin{bmatrix} \Delta t\mathbf{F} \\ \mathbf{w}^{[n]} \end{bmatrix},$$

where $\mathbf{w}^{[n]}$ and $\mathbf{w}^{[n+1]}$ are the previous and current updated vectors, each containing k components, given by

$$\mathbf{w}^{[n]} = \begin{bmatrix} \mathbf{w}_1^{[n]} \\ \mathbf{w}_2^{[n]} \\ \vdots \\ \mathbf{w}_k^{[n]} \end{bmatrix}, \quad \mathbf{w}^{[n+1]} = \begin{bmatrix} \mathbf{w}_1^{[n+1]} \\ \mathbf{w}_2^{[n+1]} \\ \vdots \\ \mathbf{w}_k^{[n+1]} \end{bmatrix}.$$

For $n = 0$, the values of $\mathbf{w}^{[0]}$ are either provided or computed using an appropriate initialization procedure. The vectors $\mathbf{W}$ and $\mathbf{F}$ represent the s-stage values and their function evaluations, respectively, defined as

$$\mathbf{W} = \begin{bmatrix} \mathbf{W}_1 \\ \mathbf{W}_2 \\ \vdots \\ \mathbf{W}_s \end{bmatrix}, \quad \mathbf{F} = \begin{bmatrix} f(\mathbf{W}_1) \\ f(\mathbf{W}_2) \\ \vdots \\ f(\mathbf{W}_s) \end{bmatrix}.$$

An example of a second-order two-stage two-step GLM [97] is given in Tab.1.5. For more general linear methods (GLMs) and their applications to differential equations, see [98, 99, 100, 101, 102].

Compared to classical RK schemes, GLMs achieve higher orders of accuracy by leveraging information from previous time instances while requiring fewer function evaluations. This approach significantly reduces computational effort by decreasing

$$\left[\begin{array}{c|c} A & U \\ \hline B & V \end{array}\right] = \left[\begin{array}{cc|cc} 0 & 0 & 1 & 0 \\ 2 & 0 & 0 & 1 \\ \hline \frac{5}{4} & \frac{1}{4} & \frac{1}{2} & \frac{1}{2} \\ \frac{3}{4} & -\frac{1}{4} & \frac{1}{2} & \frac{1}{2} \end{array}\right]$$

Table 1.5: A second-order two-stage two-step GLM [97].

the number of intermediate stages. The combination of stages and steps can be tuned to obtain schemes with the desired stability properties. However, the requirement for appropriate starting procedures to initialize the method is a downside, especially for a l-step GLM when l is large. Additionally, unlike RK methods, GLMs must be checked for null-stability, since they rely on solutions from previous time steps.

1.1.3 Multi-Derivative General Linear Methods

When multiple derivatives are incorporated into general linear methods (GLMs), the resulting class is known as multi-derivative general linear methods (MDGLMs). This broader class encompasses all the previously discussed methods, including Runge–Kutta (RK) methods, linear multistep methods (LMMs), and multi-derivative Runge–Kutta (MDRK) methods.

The MDGLMs are formulated in a manner similar to standard GLMs, but with additional matrices to account for multiple derivatives. The matrices involved are: $\{A^{(d)} \in \mathbb{R}^{s\times s}, \quad 1 \le d \le r\}$, $U \in \mathbb{R}^{s\times k}$, $\{B^{(d)} \in \mathbb{R}^{k\times s}, \quad 1 \le d \le r\}$, and $V \in \mathbb{R}^{k\times k}$. The MDGLM scheme is expressed as

$$\begin{bmatrix} \mathbf{W} \\ \mathbf{w}^{[n+1]} \end{bmatrix} = \begin{bmatrix} A^{(1)} & A^{(2)} & \dots & A^{(r)} & U \\ B^{(1)} & B^{(2)} & \dots & B^{(r)} & V \end{bmatrix} \begin{bmatrix} \Delta t \mathbf{F}^{(0)} \\ \Delta t^2 \mathbf{F}^{(1)} \\ \vdots \\ \Delta t^r \mathbf{F}^{(r-1)} \\ \mathbf{w}^{[n]} \end{bmatrix},$$

where, $\mathbf{w}^{[n+1]}$, $\mathbf{w}^{[n]}$ and $\mathbf{W}$ have the same interpretations as in standard GLMs. The

vectors $\mathbf{F}^{(l)}$ represent stage evaluations of the l-th derivative

$$\mathbf{F}^{(l)} = \begin{bmatrix} f^{(l)}(\mathbf{W}_1) \\ f^{(l)}(\mathbf{W}_2) \\ \vdots \\ f^{(l)}(\mathbf{W}_s) \end{bmatrix}, \quad 0 \leq l \leq r-1.$$

Any scheme that belongs to the class of Runge–Kutta (RK) methods, linear multistep methods (LMMs), or multi-derivative Runge–Kutta (MDRK) methods is already an example of an MDGLM. Below are a few examples of MDGLM schemes that fall outside the above-mentioned classes:

- Brown's schemes are one-stage multi-step multi-derivative schemes [84, 103]. An l-step k-derivative Brown's scheme is denoted by Brown(k, l). It is given by

$$\sum_{j=0}^{k} \alpha_j \mathbf{w}^{n+1-j} = \sum_{d=1}^{l} \Delta t^d \beta_d f^{(d-1)}(\mathbf{w}^{n+1}),$$

  where the coefficients α_i and β_i are chosen to achieve the maximum possible order of accuracy, which is $k + l - 1$, by fixing $\alpha_0 = (-1)^k k^{-l}$.

- Second Derivative Backward Differentiation Formula (SDBDF) methods [76] are a special case of Brown's schemes where the number of derivatives is fixed at 2. Specifically, the k-step SDBDF scheme corresponds to the Brown$(k, 2)$ scheme. The order of accuracy of the k-step SDBDF scheme is $k + 1$.

- For further details on multiderivative general linear methods (MDGLMs), including recent developments and applications, consult [104, 105, 106, 107, 108, 109].

As the MDGLMs represent a broader class of GLMs through the incorporation of higher-order derivatives, they inherit similar advantages to those offered by GLMs and MDRK schemes over classical RK methods. In addition, the flexibility to choose the number of stages, steps, and derivatives provides greater freedom in designing

optimal schemes. In contrast to GLMs, a p-th order Taylor method that uses $p-1$ derivatives of f can be employed as a starting procedure for MDGLMs. However, this strategy is not feasible when solving PDEs.

1.2 Spatial discretization

In this thesis, we consider time-dependent partial differential equations (PDEs) of the form

$$\mathbf{w}_t + \nabla \cdot \mathbf{F}(\mathbf{w}) = 0, \tag{1.2}$$

defined on a spatial domain $\Omega \subset \mathbb{R}^n$, where $\mathbf{w}(x,t) : \Omega \times \mathbb{R}^{\geq 0} \to \mathbb{R}^m$ is the state vector, and $\mathbf{F}(\mathbf{w}) : \mathbb{R}^m \to \mathbb{R}^m$ denotes the flux function. In particular, we focus on (viscous) conservation laws, where the flux is given by

$$\mathbf{F}(\mathbf{w}) := \mathbf{F}^{\text{inv}}(\mathbf{w}) - \mathbf{F}^{\nu}(\mathbf{w}, \nabla\mathbf{w}), \tag{1.3}$$

with $\mathbf{F}^{\text{inv}}$ representing the inviscid flux, and $\mathbf{F}^{\nu}$ denoting the viscous flux, which depends on both the state $\mathbf{w}$ and its gradient $\nabla\mathbf{w}$. The following conservation equations are considered in this thesis:

- Viscous Burgers' Equation

  $$\mathbf{F}^{\text{inv}}(\mathbf{w}) = \frac{\mathbf{w}^2}{2}, \quad \mathbf{F}^{\nu}(\mathbf{w}, \nabla\mathbf{w}) = \nabla\mathbf{w},$$

  where $\mathbf{w}(x,t) : \Omega \times \mathbb{R}_{\geq 0} \to \mathbb{R}$ for $\Omega \subset \mathbb{R}$.

- Compressible Navier–Stokes Equations. For the state vector $\mathbf{w} = [\rho \ \rho\mathbf{v} \ E]^T$, the inviscid and viscous fluxes are given by:

  $$\mathbf{F}^{\text{inv}}(\mathbf{w}) = \begin{pmatrix} \rho\mathbf{v} \\ \rho\mathbf{v} \otimes \mathbf{v} + p\mathbf{Id} \\ \mathbf{v}(E+p) \end{pmatrix}, \quad \mathbf{F}^{\nu}(\mathbf{w}, \nabla\mathbf{w}) = \begin{pmatrix} 0 \\ \tau \\ \tau \cdot \mathbf{v} + \mathbf{q} \end{pmatrix},$$

  where $\rho \in \mathbb{R}$ is the density, $\mathbf{v} \in \mathbb{R}^2$ is the velocity, $E \in \mathbb{R}$ is the total energy, p

is the pressure, τ is the viscous stress tensor and $\mathbf{q} = \lambda_T \nabla T$ is the heat flux.

- Isentropic Euler Equations. For the state vector $\mathbf{w} = [\rho\ \rho\mathbf{v}]^T$, the inviscid flux is given by:

$$\mathbf{F}^{\text{inv}}(\mathbf{w}) = \begin{pmatrix} \rho\mathbf{v} \\ \rho\mathbf{v} \otimes \mathbf{v} + p\mathbf{Id} \end{pmatrix},$$

and $\mathbf{F}^{\nu}(\mathbf{w}, \nabla\mathbf{w}) = 0$. The variables are defined as in the compressible Navier–Stokes equations.

Further details on the above conservation equations are provided in the subsequent chapters, wherever they are applied. Formally, the PDE (1.2) can be cast into an ODE

$$\mathbf{w}_t = \mathbf{R}^{(1)}(\mathbf{w}) \tag{1.4}$$

in some infinite dimensional function space where the function $\mathbf{R}^{(1)}(\mathbf{w})$ is defined as

$$\mathbf{R}^{(1)}(\mathbf{w}) := -\nabla \cdot \mathbf{F}(\mathbf{w}). \tag{1.5}$$

There are several methods available in the literature for discretizing the spatial operator $\mathbf{R}^{(1)}(\mathbf{w})$ that arises in partial differential equations. Finite difference (FD) methods are well-suited for structured meshes but often struggle in the presence of discontinuities and complex geometries. Finite volume (FV) methods, on the other hand, are capable of handling discontinuities and are compatible with unstructured meshes, offering greater flexibility for complex solution structures. However, in FV schemes, increasing the accuracy typically requires refining the mesh or higher-order polynomial reconstruction, or the use of ENO-type schemes.

In order to effectively combine spatial discretization with higher-order time-stepping methods, it is essential to use a spatial scheme that also provides high-order accuracy. For this reason, we employ the Discontinuous Galerkin (DG) method, which combines the local conservation and geometric flexibility of FV methods with the high-order accuracy of finite element (FE) methods. The DG method is particularly well-suited for solving conservation laws involving complex geometries and discontinuous solutions. The Discontinuous Galerkin (DG) scheme was first introduced in

[65]. It was later significantly developed by Cockburn and Shu, who provided a comprehensive theoretical foundation for the method; see [66, 67, 110, 111, 112]. For a broader overview and various applications of DG methods, the reader is referred to [113, 114].

In particular, we consider the specific DG method introduced in [115], known as the Discontinuous Galerkin Spectral Element Method (DGSEM), due to its computational efficiency and the availability of DGSEM-based open-source software FLEXI[2]. The FLEXI software [116] was developed at the Institute of Aerodynamics and Gas Dynamics at the University of Stuttgart. The FLEXI framework has since been extended into various other software packages, including FLEXImultiphase, GALAEXI [117] (a GPGPU-enabled extension), RELEXI (a reinforcement learning framework), and ELEXI [118] (a high-order numerical Eulerian–Lagrangian extension).

1.2.1 Discontinuous Galerkin Spectral Element Method

Consider the equation (1.2)

$$\mathbf{w}_t + \nabla \cdot \mathbf{F}(\mathbf{w}) = 0,$$

defined on a spatial domain $\Omega \subset \mathbb{R}^2$. The domain Ω is partitioned into $\mathcal{N}_E$ non-overlapping mesh elements,

$$\Omega = \bigcup_{e=1}^{\mathcal{N}_E} \Omega_e.$$

Define the space of broken polynomial as follows:

$$V_{\mathcal{N}_p} := \left\{ v \in L^2(\Omega) \;\middle|\; v|_{\Omega_e} \in \Pi_{\mathcal{N}_p}(\Omega_e),\ 1 \le e \le \mathcal{N}_E \right\},$$

where the space $\Pi_{\mathcal{N}_p}(\Omega_e)$ denotes the tensor product of one-dimensional Lagrange basis function of degree at most $\mathcal{N}_p$. The weak formulation of equation (1.2) is given by

$$\sum_{e=1}^{\mathcal{N}_E} (\mathbf{w}_t, \phi)_{\Omega_e} - (\mathbf{F}(\mathbf{w}), \nabla\phi)_{\Omega_e} + \left\langle \mathbf{F}^*(\mathbf{w}^L, \mathbf{w}^R) \cdot \mathbf{n}, \phi \right\rangle_{\partial\Omega_e} = 0,\ \forall \phi \in \Pi_{\mathcal{N}_p}, \tag{1.6}$$

[2] http://www.flexi-project.org

where $(:,:)_{\Omega_e}$ denotes the volume integral over Ω_e and $\langle :,: \rangle_{\partial\Omega_e}$ denotes the surface intergral over the element boundary $\partial\Omega_e$. The term $\mathbf{F}^*(\mathbf{w}^L, \mathbf{w}^R)$ represents the numerical flux, computed using the values $\mathbf{w}^L$ and $\mathbf{w}^R$ from the adjacent elements, and $\mathbf{n}$ is the outward unit normal to the surface.

For the ease of computation, each element Ω_e is mapped onto the reference element $[-1,1]^2$, and the equations are transformed into the corresponding reference space $\boldsymbol{\xi} = (\xi_1, \xi_2)$; see [119] for the detailed formulation. On the reference element, the unknowns are approximated by polynomials constructed as the tensor product of one-dimensional Lagrange interpolation polynomials (ℓ) of degree $\mathcal{N}_p$, which form a key component of the DGSEM. The solution is then approximated by

$$\mathbf{w}(\boldsymbol{\xi}) \approx \mathbf{w}_h(\boldsymbol{\xi}) := \sum_{i,j=0}^{\mathcal{N}_p} \hat{\mathbf{w}}_{ij}\, \ell_i(\xi_1)\ell_j(\xi_2), \tag{1.7}$$

and the transformed flux functions are approximated as

$$\mathcal{F}^m(\boldsymbol{\xi}) \approx \mathcal{F}^m_h(\boldsymbol{\xi}) := \sum_{i,j=0}^{\mathcal{N}_p} \hat{\mathcal{F}}^m_{ij}\, \ell_i(\xi_1)\ell_j(\xi_2), \quad m = 1, 2, \tag{1.8}$$

where $(\hat{\bullet})_{ij}$ denotes the (i,j)-th coefficient in the polynomial representation of the corresponding quantity.

The Gauss-Legendre quadrature for approximating the integrals in Eq.(1.6) and the one-dimensional Lagrange interpolation polynomial uses the same $\mathcal{N}_p + 1$ Gauss-Legendre nodes. Substituting polynomial representations of $\mathbf{w}_h$ and $\mathcal{F}^m_h$ in Eq.(1.6), and applying the quadrature approximations in an element, and rearranging the terms, we get the spatial DGSEM operator $\mathbf{R}^{(1)}_h(\mathbf{w}_h)$, see [119],

$$\begin{aligned}\mathbf{R}^{(1)}_h(\mathbf{w}_h)_{ij} := -\frac{1}{\mathbf{J}_{geo_{ij}}}\Big[&\sum_{\lambda=0}^{\mathcal{N}_p} \hat{\mathcal{F}}^1_{\lambda j}\hat{D}_{\lambda j} + \Big([\mathbf{f}^*\hat{s}]^{+\xi_1}_j \hat{\ell}_i(1) + [\mathbf{f}^*\hat{s}]^{-\xi_1}_j \hat{\ell}_i(-1)\Big) \\ &+ \sum_{\mu=0}^{\mathcal{N}_p} \hat{\mathcal{F}}^2_{i\mu}\hat{D}_{i\mu} + \Big([\mathbf{f}^*\hat{s}]^{+\xi_2}_i \hat{\ell}_j(1) + [\mathbf{f}^*\hat{s}]^{-\xi_2}_i \hat{\ell}_j(-1)\Big)\Big], \ \forall i,j,\end{aligned} \tag{1.9}$$

where,

$$\hat{\ell}_i := \frac{\ell_i}{\omega_i} \text{ and } \hat{D}_{ij} := \frac{\omega_i}{\omega_j} \frac{\partial \ell_i(\xi)}{\partial \xi}\Big|_{\xi=\xi_j}, \tag{1.10}$$

with the Gauss-Legendre quadrature weight ω_i and $\mathbf{J}_{geo}$ is the Jacobian of the geometrical transformation. The four sides of the unit reference element are denoted by $-\xi_1$, $+\xi_1$, $-\xi_2$ and $+\xi_2$. The transformed flux $[\mathbf{f}^* \hat{s}]$ is evaluated at the four edges at each nodes in the Eq.(1.9). The states $\mathbf{w}^L$ and $\mathbf{w}^R$ are computed by evaluating the solution polynomials at the cell edges. See [119], for the definition of the surface element $\hat{s}$ and for the complete derivation [115, 119]. We use the global Lax-Friedrichs flux here,

$$\mathbf{f}^*(\mathbf{w}^L, \mathbf{w}^L, \mathbf{n}) = \frac{1}{2}\Big(\mathbf{F}(\mathbf{w}^L) + \mathbf{F}(\mathbf{w}^R)\Big) \cdot \mathbf{n} + \lambda\Big(\mathbf{w}^L - \mathbf{w}^R\Big) \tag{1.11}$$

for a globally constant value λ. This simple form of numerical flux is chosen to attain a straightforward expression for the second derivative term, as shown in Eq. (1.15). In contrast, other commonly used numerical fluxes may complicate the derivation of the second derivative form. Moreover, in [120, Paper V], the global Lax-Friedrichs flux with carefully chosen values of λ plays a key role in ensuring the asymptotic preservation property (refer to Definition 2) of the schemes considered for low-Mach flows.

Evaluation of the second time derivative

When it comes to multi-derivative time-stepping schemes, they require higher-order temporal derivatives of Equation (1.2) for the computation. Here, we restrict ourselves to second-derivative time-stepping schemes, which require only the first and second time derivatives of Equation (1.2). The spatial operator for the second derivative is evaluated by introducing the artificial quantity,

$$\sigma := \mathbf{R}^{(1)}(\mathbf{w}) \equiv \mathbf{w}_t \tag{1.12}$$

as done in [91]. Differentiating Eq.(1.2) with respect to time and applying σ, we get,

$$\mathbf{w}_{tt} = (\mathbf{R}^{(1)}(\mathbf{w}))_t = -\nabla \cdot \left(\frac{\partial \mathbf{F}}{\partial \mathbf{w}} \mathbf{R}^{(1)}(\mathbf{w}) \right) = -\nabla \cdot \left(\frac{\partial \mathbf{F}}{\partial \mathbf{w}} \sigma \right) := \mathbf{R}^{(2)}(\mathbf{w}, \sigma). \tag{1.13}$$

Similarly to the first temporal derivative, the spatial DGSEM operator for the second derivative is derived by introducing the new flux function $\frac{\partial \mathbf{F}}{\partial \mathbf{w}} \sigma$

$$\begin{aligned} \mathbf{R}_h^{(2)}(\mathbf{w}_h, \sigma_h)_{ij} := -\frac{1}{\mathbf{J}_{geo_{ij}}} \Big[\sum_{\lambda=0}^{\mathcal{N}_p} \frac{\partial \hat{\mathcal{F}}^1_{\lambda j}}{\partial \hat{\mathbf{w}}_{\lambda j}} \hat{\sigma}_{\lambda j} \hat{D}_{\lambda j} + \Big([\tilde{\mathbf{f}}^* \hat{s}]_j^{+\xi_1} \hat{\ell}_i(1) + [\tilde{\mathbf{f}}^* \hat{s}]_j^{-\xi_1} \hat{\ell}_i(-1) \Big) \\ + \sum_{\mu=0}^{\mathcal{N}_p} \frac{\partial \hat{\mathcal{F}}^2_{i\mu}}{\partial \mathbf{w}_{i\mu}} \hat{\sigma}_{i\mu} \hat{D}_{i\mu} + \Big([\tilde{\mathbf{f}}^* \hat{s}]_i^{+\xi_2} \hat{\ell}_j(1) + [\tilde{\mathbf{f}}^* \hat{s}]_i^{-\xi_2} \hat{\ell}_j(-1) \Big) \Big], \ \forall i, j, \end{aligned} \tag{1.14}$$

where $\sigma_h := \mathbf{R}^{(1)}(\mathbf{w}_h)$ and the numerical flux $\tilde{\mathbf{f}}^*(\mathbf{w}^L, \mathbf{w}^R, \sigma^L, \sigma^R, \mathbf{n})$. The states σ^L and σ^R can also be computed by evaluating the solution polynomials at the cell edges. Similar to the numerical flux in Eq.(1.9), we use the global Lax-Friedrichs flux for $\tilde{\mathbf{f}}^*$ as well,

$$\tilde{\mathbf{f}}^*(\mathbf{w}^L, \mathbf{w}^R, \sigma^L, \sigma^R, \mathbf{n}) = \frac{1}{2} \left(\frac{\partial \mathbf{F}(\mathbf{w}^L)}{\partial \mathbf{w}^L} \sigma^L + \frac{\partial \mathbf{F}(\mathbf{w}^R)}{\partial \mathbf{w}^R} \sigma^R \right) \cdot \mathbf{n} + \lambda \Big(\sigma^L - \sigma^R \Big). \tag{1.15}$$

1.3 Stiffness in differential equations

Stiffness in differential equations refers to the presence of components that evolve at significantly different rates. This phenomenon can arise in several ways, a few of them are listed below

- Consider the Van der Pol problem

$$\mathbf{w}_1'(t) = \mathbf{w}_2, \quad \mathbf{w}_2'(t) = \frac{(1 - \mathbf{w}_1^2)\mathbf{w}_2 - \mathbf{w}_1}{\varepsilon} \tag{1.16}$$

  This nonlinear oscillator is a classical model introduced in the 1920s by the Dutch electrical engineer Balthasar van der Pol [121] to describe the behavior of early vacuum tube circuits. As $\varepsilon \ll 1$, the solution components exhibit rapid changes over short time intervals, making the system stiff.

- When a PDE is spatially discretized, the resulting system of ODEs can exhibit eigenvalues with widely varying magnitudes, particularly when the mesh contains elements of significantly different sizes. This disparity often leads to a stiff system, making the application of numerical schemes more challenging.

- Consider the dimensionless isentropic Euler equations

$$\begin{pmatrix} \rho \\ \rho \mathbf{v} \end{pmatrix}_t = -\nabla \cdot \begin{pmatrix} \rho \mathbf{v} \\ \rho \mathbf{v} \otimes \mathbf{v} + \frac{1}{\varepsilon^2} p \mathbf{Id} \end{pmatrix}, \tag{1.17}$$

  where the variables are defined as in Sec. 1.2. The system is closed with the equation of state $p(\rho) := \kappa \rho^\gamma$, where $\kappa > 0$ and $\gamma > 1$. The stiffness of the system is governed by the reference Mach number parameter ε, which is the ratio of the fluid speed to the speed of sound. Depending on the value of the stiffness parameter, equations model different flow regimes

  - $\varepsilon < 1$: low Mach (nearly incompressible),
  - $\varepsilon = 1$: compressible flows with moderate speeds,
  - $\varepsilon > 1$: highly compressible or supersonic flows[3].

  A notable application where such variations in ε are relevant is the de Laval nozzle—a specially shaped nozzle used in jet engines and rockets to accelerate gases to supersonic speeds[4]. In contrast, examples of low Mach number ($\varepsilon \ll 1$) flows include water in pipes and air flow through ducts.

However, the notion of stiffness is not always straightforward. Several factors can introduce stiffness into a problem or its discretized equations. A comprehensive review of the historical development of the concept of stiffness in differential equations is presented in [122].

[3] For $\varepsilon > 1$, supersonic flows can create shock waves, leading to entropy changes that violate the assumption of isentropic flow. Consequently, such flows must be treated with caution. However, this regime is not addressed in the present thesis.

[4] https://www.grc.nasa.gov/www/k-12/airplane/isentrop.html

1.3.1 Necessity of implicitness

Starting from the Van der Pol problem (1.16), if we apply a time-stepping scheme to solve it numerically, we must ensure that the time step size Δt lies within the stability region of the chosen method. As explicit schemes usually come with a bounded stability region, the stiffness of the problem forces explicit methods to use extremely small time steps to maintain stability, which can make them inefficient or impractical for long-time simulations.

Consider the isentropic Euler equation (1.17). The eigenvalues of the Jacobian of the flux function in two dimensions are given by

$$\lambda_1 = \mathbf{v} \cdot \mathbf{n}, \quad \text{and} \quad \lambda_{2,3} = \mathbf{v} \cdot \mathbf{n} \pm \frac{1}{\varepsilon}\sqrt{\kappa\gamma\rho^{\gamma-1}}, \tag{1.18}$$

for some direction vector $\mathbf{n}$. For explicit schemes, the stability restriction imposed by the CFL condition [123] forces small time step sizes, with

$$\Delta t = \mathcal{O}(\varepsilon \Delta x)$$

where Δx is the spatial mesh size. Hence, as $\varepsilon \to 0$, the time step Δt must also tend to zero, making it infeasible to use.

In order to circumvent the above-mentioned issues, implicit schemes [124]–which possess unbounded stability regions–can be employed. As they impose no, or significantly less severe, time step restrictions, implicit methods are well suited for the numerical solution of stiff equations. However, implicit methods also bring computationally expensive tasks of solving nonlinear equations. The computational complexity increases even further when the fluxes are highly nonlinear and stiff. Another drawback of implicit methods is their tendency to introduce more numerical diffusion, which can overly smooth sharp gradients or essential features in the solution.

1.3.2 Implicit-Explicit schemes

As discussed above, implicit methods can provide numerical solutions to stiff equations without compromising on the timestep sizes. However, they require computationally

expensive tasks like solving nonlinear equations and have drawbacks such as unwanted numerical diffusion. One way to reduce the computational burden of inverting highly nonlinear equations is to split the terms into stiff and non-stiff parts. For example, in the case of ODEs, the equation (1.1) can be split as

$$\frac{du}{dt} = f(u) =: \underbrace{f_{\mathbf{I}}(u)}_{\text{stiff}} + \underbrace{f_{\mathbf{E}}(u)}_{\text{non-stiff}} . \tag{1.19}$$

Similarly, in the case of PDEs, the flux in Eq. (1.2) can be decomposed as

$$\mathbf{w}_t = -\nabla \cdot \mathbf{F}(\mathbf{w}) =: -\nabla \cdot \left(\underbrace{\mathbf{F}_{\mathrm{I}}(\mathbf{w})}_{\text{stiff}} + \underbrace{\mathbf{F}_{\mathrm{E}}(\mathbf{w})}_{\text{non-stiff}} \right).$$

The stiff part can be chosen to be nearly linear, allowing it to be treated implicitly, while the non-stiff part can be handled explicitly. Such timestepping schemes are referred to as implicit-explicit (IMEX) schemes. These class of methods take advantage of the speed of explicit schemes and the unconditional stability of implicit methods. The two-derivative IMEX-Taylor scheme [125]

$$\mathbf{w}^{n+1} = \mathbf{w}^n + \Delta t \left(f_{\mathbf{I}}^{(0)}(\mathbf{w}^{n+1}) + f_{\mathbf{E}}^{(0)}(\mathbf{w}^n) \right) - \frac{\Delta t^2}{2} \left(f_{\mathbf{I}}^{(1)}(\mathbf{w}^{n+1}) - f_{\mathbf{E}}^{(1)}(\mathbf{w}^n) \right) \tag{1.20}$$

is an example for second-order IMEX method.

Designing an effective splitting strategy for a given problem is an art and is far from trivial. However, developing such a splitting is beyond the scope of this thesis. Therefore, we adopt an established splitting method that is well-suited to the problem under consideration. For example, in the case of the dimensionless isentropic equations (1.17), we use the splitting proposed in [126], defined as

$$\mathbf{F}_{\mathrm{I}}(\mathbf{w}) = \begin{pmatrix} \rho \mathbf{v} \\ \frac{1-\varepsilon^2}{\varepsilon^2} p \mathbf{Id} \end{pmatrix}, \quad \text{and} \quad \mathbf{F}_{\mathrm{E}}(\mathbf{w}) = \begin{pmatrix} 0 \\ \rho \mathbf{v} \otimes \mathbf{v} + p \mathbf{Id} \end{pmatrix}. \tag{1.21}$$

The eigenvalues of the Jacobian of the explicit part are given by

$$\lambda_1 = 0, \quad \lambda_2 = \mathbf{v} \cdot \mathbf{n} \quad \text{and} \quad \lambda_3 = 2\mathbf{v} \cdot \mathbf{n},$$

and they do not depend on the stiffness parameter. Since the timestep restriction is influenced only by the non-stiff part, which is treated explicitly, we have

$$\Delta t = \mathcal{O}(\Delta x).$$

In [127], the authors presented a strategy to obtain suitable flux splittings through characteristic decomposition of the Jacobian. For further details on various IMEX methods, consult the following references: Runge–Kutta methods [128, 129, 130, 131, 132, 133]; linear multistep methods (LMMs) [134, 135]; multi-derivative methods [125, 92, 83, 136, 137, 138]; and second-derivative general linear methods (GLMs) [109]. For additional developments, see also [139, 140, 141, 142].

The equations (1.16) and (1.17) are examples of singularly perturbed problems, meaning their mathematical type changes as $\varepsilon \to 0$:

- Van der Pol equation(1.16): As $\varepsilon \to 0$, the system reduces to

$$\mathbf{w}_1'(t) = \mathbf{w}_2, \quad 0 = (1 - \mathbf{w}_1^2)\mathbf{w}_2 - \mathbf{w}_1, \tag{1.22}$$

  a differential-algebraic equation (DAE), involving both a differential and an algebraic constraint.

- Compressible isentropic Euler equations (1.17): In the limit $\varepsilon \to 0$, the equations reduce to the incompressible Euler equations

$$\begin{aligned} \rho_{(0)} \equiv \text{const} > 0, \quad \nabla \cdot \mathbf{v}_{(0)} = 0, \\ (\mathbf{v}_{(0)})_t + \nabla \cdot (\mathbf{v}_{(0)} \otimes \mathbf{v}_{(0)}) + \frac{\nabla p_{(2)}}{\rho_{(0)}} = 0. \end{aligned} \tag{1.23}$$

  This limit is obtained under the assumption that the solution $\mathbf{w}$ of (1.17) admits a Hilbert expansion [143] of the form

$$\mathbf{w} = \mathbf{w}_{(0)} + \varepsilon \mathbf{w}_{(1)} + \varepsilon^2 \mathbf{w}_{(2)} + \mathcal{O}(\varepsilon^3).$$

Therefore, solving such problems numerically introduces additional challenges. To

ensure both efficiency and stability, the temporal and spatial discretizations must account for the changing behavior of the equations as $\varepsilon \to 0$.

Definition 2. *A time stepping scheme is said to be asymptotic preserving (AP) if, in the formal limit as $\varepsilon \to 0$, the numerical solution of the scheme with well prepared initial data is a consistent discretization of the limiting equation. The initial data is said to be well-prepared if it possesses a well-defined Hilbert expansion.*

In the case of the Van der Pol equation (1.16) and the compressible isentropic Euler equations (1.17), an asymptotic preserving (AP) time stepping scheme ensures that, as $\varepsilon \to 0$, the numerical solution of the singularly perturbed problem converges to a consistent approximation of the solution to the corresponding limiting problem (Eq. (1.22) or (1.23)). For further insights on AP schemes, see [144, 145, 146, 147, 148, 149, 150, 151, 152, 125].

In [120, Paper V], we have shown that the two-derivative second-order IMEX Taylor scheme for the compressible isentropic Euler equations, using the splitting (1.21) given in [126], is asymptotic preserving under periodic boundary conditions.

1.3.3 Multirate schemes: Explicit approach

Unlike IMEX (Implicit-Explicit) schemes, which treat stiff terms implicitly, multirate methods apply a specialized explicit treatment. The main reason for the implicit treatment of stiff terms in IMEX schemes is to eliminate severe timestep restrictions imposed by stiffness. In contrast, multirate schemes treat both stiff and non-stiff components explicitly but with different timestep sizes.

In a typical multirate approach, the slow (non-stiff) part is integrated using a Runge-Kutta method with a timestep Δt, while the fast (stiff) part is computed using a different Runge-Kutta method with a smaller timestep $\Delta\tau < \Delta t$. The smaller timestep is usually chosen as

$$\Delta\tau = \frac{\Delta t}{M}$$

where $M \in \mathbb{N}$ is the number of fast substeps corresponding to the fast component within one slow timestep. The key idea is that the overall stability of the multirate scheme depends on the choice of M.

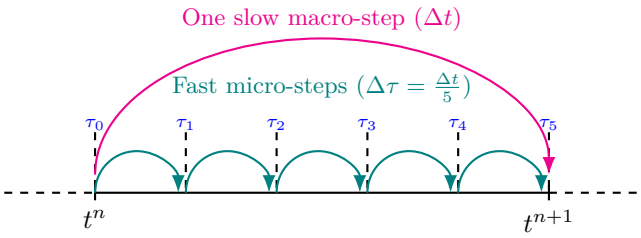


Figure 1.2: Schematic representation of a multirate scheme: one macro-step for the slow component and five micro-steps ($M = 5$) for the fast component.

For example consider the ODE

$$\frac{du}{dt} = f(u) = f_{\mathbf{I}}(u) + f_{\mathbf{E}}(u),$$

where $f_{\mathbf{E}}$ is the slow component and $f_{\mathbf{I}}$ is the fast component. When the Euler scheme is applied to the above ODE, the solution is given by

$$u^{n+1} = u^n + \Delta t\, f(u^n).$$

This can also be interpreted as the exact solution of an auxiliary ODE defined by

$$z(0) = u^n, \quad \frac{dz(\tau)}{d\tau} = f(u^n),$$

with respect to a new temporal variable τ. Thus, we have $u^{n+1} = z(\Delta t)$. The concept of multirate schemes is built on this idea. During one slow macro-timestep (Δt), the auxiliary ODE is reformulated as

$$z(0) = u^n, \quad \frac{dz(\tau)}{d\tau} = f_{\mathbf{E}}(u^n) + f_{\mathbf{I}}(z(\tau)).$$

It can be noted that if $f_{\mathbf{I}}$ is taken to be zero, then multirate scheme reduces to the original base scheme (e.g., Euler). Hence, by solving the above ODE exactly for $z(\tau)$, or by applying a Runge-Kutta scheme with finer timestep sizes $\Delta\tau$, the potential stability issues caused by the fast component $f_{\mathbf{I}}(u)$ can be mitigated. When the ODE is solved exactly, the resulting multirate scheme inherits the stability properties of

the base method used for the slow component. If a numerical scheme is used for solving the auxiliary ODE, the stability of the combined method can be controlled by adjusting M. As M increases, the stability region of the multirate method typically expands [153]. Therefore, as the stiffness of the problem increases, the parameter M can be increased to maintain numerical stability and achieve an accurate solution. A general approach for constructing one-derivative multirate schemes is presented in [154].

In [153], a multirate scheme is applied to ocean models, where fast gravity waves contribute to the stiff part, while the remaining dynamics contribute to the non-stiff part. For higher-order multirate schemes and their applications, consult [154, 155, 156, 157, 158, 159, 160, 161, 162, 163]. Multirate schemes are well suited for mildly stiff problems. However, they are difficult to use for highly stiff equations ($\varepsilon \ll 1$).

Existing multirate schemes are primarily based on one-derivative formulations. In [164, Paper IV], we extended the multirate approach to multi-derivative schemes. We mathematically derived the non-trivial order conditions and proposed novel schemes. The stability of this new class of multirate multi-derivative schemes was investigated, and their numerical properties were studied using a variety of ODE and PDE test problems.

1.4 Predictor-Corrector Schemes

As discussed in the previous sections, implicitness is essential for the development of asymptotic preserving (AP) schemes. For implicit schemes, it is highly desirable that they are A-stable, or at least $A(\alpha)$-stable with α close to 90°, see [76, 103, 165, 166, 167]. This guarantees stability even as the problem stiffness increases. Therefore, when designing a scheme that possesses properties such as being asymptotic preserving (AP), A-stabilty, L-stability, and suitable for highly stiff problems, the scheme must allow sufficient flexibility. This consideration naturally leads to the use of fully implicit Runge-Kutta (FIRK) type methods.

Collocation schemes are among the suitable choices of FIRK methods. See Tab. 1.3 for a one-derivative collocation scheme, and Tab. 1.4 for a two-derivative collocation scheme. However, due to their coupled stage equations, collocation schemes lead to

an increase in the problem size when solving PDEs. If the discretized PDE has N_{DOF} degrees of freedom, a collocation scheme with s implicit stages requires solving one large system of size $s \cdot N_{\text{DOF}}$ simultaneously, where the Newton method must invert a Jacobian matrix of size $(s \cdot N_{\text{DOF}}) \times (s \cdot N_{\text{DOF}})$. In contrast, for schemes with decoupled stages, such as DIRK schemes, it is only necessary to invert a Jacobian matrix of size $(N_{\text{DOF}} \times N_{\text{DOF}})$ for each of the s stages separately. Computationally, this results in a significant difference.

To mitigate the computational complexity associated with solving all coupled stages of collocation schemes simultaneously, an iterative solution strategy is often employed. One such approach is the Spectral Deferred Correction (SDC) method, introduced by Dutt, Greengard, and Rokhlin in [168]. The SDC method is part of a broader class of Deferred Correction (DC) schemes, which construct higher-order time integration methods from lower-order building blocks using iterative refinement.

In review by Ong and Spiteri [169], DC schemes are categorized into four main classes based on the formulation of their associated error equations. The Classical Deferred Correction (CDC) method [170] and the method proposed in [168] (often referred to as DGR) are based on solving initial value problems (IVPs) for the error function. On the other hand, the SDC method [171] and the Integral Deferred Correction (IDC) schemes [172, 173] utilize the integral form of the error equation. In principle, all of the aforementioned DC methods result in similar schemes.

Each correction step involves solving an error equation and updating the current approximation with an increased order of accuracy. This iterative process continues until the method reaches its designed order of consistency. Eventually the iterative scheme converges to the solution of the underlying collocation scheme. For further details and developments in DC methods, readers are referred to [174, 175, 176, 177, 178, 179, 180] and the references therein.

1.4.1 Hermite–Birkhoff Predictor–Corrector Schemes for Implicit and IMEX Integration

A two-derivative SDC-type asymptotic preserving IMEX scheme was developed in [125]. This work was later extended to higher-order schemes with parallel-in-time ca-

pabilities in [92]. The stability properties of these methods were subsequently studied and improved in [167]. The IMEX-HBPC scheme is presented below for the general splitting in Eq. (1.19). The scheme begins with a predicted solution computed using a two-derivative, second-order IMEX-Taylor method:

1. Predict. Solve the following expression for $\mathbf{w}^{n,[0],l}$ and $1 \le l \le s$:

$$\begin{aligned}\mathbf{w}^{n,[0],l} := \mathbf{w}^n &+ c_l \Delta t \left(f_{\mathbf{I}}^{(0)}(\mathbf{w}^{n,[0],l}) + f_{\mathbf{E}}^{(0)}(\mathbf{w}^n) \right) \\ &- \frac{(c_l \Delta t)^2}{2} \left(f_{\mathbf{I}}^{(1)}(\mathbf{w}^{n,[0],l}) - f_{\mathbf{E}}^{(1)}(\mathbf{w}^n) \right),\end{aligned} \tag{1.24}$$

2. Correct. Solve the following for $\mathbf{w}^{n,[k],l}$, for each $1 \le l \le s$ and each $1 \le k \le k_{\max}$:

$$\begin{aligned}\mathbf{w}^{n,[k],l} := \mathbf{w}^n &+ \theta_1 \Delta t \left(f_{\mathbf{I}}^{(0)}(\mathbf{w}^{n,[k],l}) - f_{\mathbf{I}}^{(0)}(\mathbf{w}^{n,[k-1],l}) \right) \\ &- \frac{\Delta t^2}{2} \theta_2 \left(f_{\mathbf{I}}^{(1)}(\mathbf{w}^{n,[k],l}) - f_{\mathbf{I}}^{(1)}(\mathbf{w}^{n,[k-1],l}) \right) \\ &+ \mathcal{I}_l \left(\mathbf{w}^{n,[k-1],1}, \ldots, \mathbf{w}^{n,[k-1],s} \right),\end{aligned} \tag{1.25}$$

   where (θ_1, θ_2) are stability-optimized coefficients [167], and $\mathcal{I}_l$ is the quadrature rule, see [92].

3. Update.

$$\mathbf{w}^{n+1} := \mathbf{w}^{n,[k_{\max}],s}$$

Remark 1. *The fully implicit HBPC scheme can be derived from the IMEX-HBPC algorithm by replacing $f_{\mathbf{I}}$ with f and setting $f_{\mathbf{E}} = 0$.*

In each correction step, the order of convergence increases by one until it reaches the order of the underlying quadrature rule (collocation scheme). Specifically, the convergence order after the k-th correction step is given by $\min(q, k+2)$, where q is the order of the quadrature rule. A proof of the convergence order is provided in [125].

In [181], implicit HBPC schemes were applied to the Navier–Stokes equations, and results up to eighth-order accuracy were presented. Since multiderivative schemes can mix stiff and non-stiff components during the evaluation of higher derivatives, an

explicitness-preserving IMEX-HBPC scheme was proposed in [137] to avoid unnecessary inversions of the non-stiff terms. In [120, Paper V], we have shown that the IMEX-HBPC (explicitness-preserving) scheme for the compressible isentropic Euler equations, using the splitting (1.21) given in [126], is asymptotic preserving (AP) under periodic boundary conditions. Numerical results are presented for schemes of order up to eight.

1.5 Parallel-in-time methods

Numerical schemes that provide highly accurate approximate solutions are in great demand. However, all the higher-order schemes discussed in the previous sections require additional stages, derivatives, steps, or other computational efforts, which can significantly increase the run time of these methods. Leveraging modern computer architectures with multiple CPUs or GPUs, parallelizing these schemes provides an effective way to reduce the increased run times.

One of the parallel-in-time (PinT) schemes is the parareal method. In a broad sense, the idea behind these schemes is to divide the time domain into N sub-intervals (or windows) and utilize N processors to solve N initial value problems in parallel. The resulting solution is then analyzed and iteratively refined to improve accuracy. For further insights and applications of the parareal method, consult the works in [182, 183, 184, 185]. In [186], a hybrid parareal algorithm is presented that incorporates SDC methods within the parareal framework. We have not employed any parareal methods in this thesis.

Deferred correction (DC) schemes lend themselves naturally to parallelization due to their iterative and stage-based structure. Several parallel DC methods exploiting pipeline parallelism have been proposed in the literature [187, 188, 189, 171]. These methods represent another class of PinT schemes, referred to as parallel across the method.

In [92], parallel-in-time HBPC methods were developed and implemented for ODEs. We have extended the PinT HBPC method for the Navier-Stokes equations in [94, Paper I], with further parallelization of stages in the prediction and first correction steps. The results were demonstrated on various 2D and 3D test cases.

1.6 Solving non-linear systems

1.6.1 Newton method

In the thesis, the results concerning the solution of ordinary differential equations (ODEs) using implicit time-stepping schemes are presented using MATLAB-based codes. The resulting nonlinear equations are solved using a damped Newton method. For systems of ODEs, the Jacobian matrices in the resulting linear solvers are inverted using MATLAB backslash operator.

When solving partial differential equations (PDEs) using the Discontinuous Galerkin Spectral Element Method (DGSEM), spatial discretization is coupled with Newton's method, while the resulting linear systems at each Newton iteration are solved using the GMRES algorithm [190, 191, 192]. Let $\mathbf{G}$ denote the resulting non-linear system:

$$\mathbf{G}(w) = 0,$$

where w is the unknown. The corresponding Jacobian $\mathcal{J}(w)$ of the nonlinear system is defined as

$$\mathcal{J}(w) := \frac{\partial \mathbf{G}(w)}{\partial w}.$$

Then, the Newton iterations are given by

$$\begin{aligned} \mathcal{J}^{(r)} \cdot \delta w^{(r)} &= -\mathbf{G}(w^{(r)}), \\ w^{(r+1)} &= w^{(r)} + \delta w^{(r)}. \end{aligned}$$

Choosing a stopping criterion for the Newton procedure is always a delicate task. Typically, the error tolerances are set very low to ensure that the nonlinear solver does not dominate the temporal or spatial discretization errors. However, this approach can lead to oversolving, resulting in unnecessary computational effort and increased run time.

An adaptive Newton strategy [193, 194] can help mitigate this issue by adjusting the solver tolerance according to the accuracy required at each time step. In [94,

Paper I], we developed such a strategy for HBPC schemes, using temporal error estimation based on a lower-order embedded scheme [193, 195, 196, 194]. Results show a considerable reduction in the number of nonlinear iterations when the adaptive Newton strategy is used compared to fixed tolerances.

1.6.2 GMRES method: Using a matrix-free approach

In each Newton iteration, the GMRES method is used to solve the linear system arising at each Newton step. The solution is approximated in the Krylov subspace

$$\{r_0, \mathcal{J}r_0, \mathcal{J}^2 r_0, \mathcal{J}^3 r_0, \ldots, \mathcal{J}^{K_{\text{dim}}} r_0\}$$

where r_0 is the residue and K_{dim} is the dimension of the Krylov subspace. It is important to note that GMRES requires only matrix-vector products to approximate the solution, making it particularly well-suited for matrix-free implementations. In such an approach, these matrix-vector products are approximated using finite differences with a small perturbation. For further details on matrix-free implementations and finite difference approximations, see [197, 198, 199].

1.6.3 Preconditioning of the extended system

To accelerate the convergence of GMRES, the use of an appropriate preconditioner is essential. In [181, Sec. 4], the authors evaluate various preconditioners based on their effects on both linear and nonlinear iteration performance. The study shows that problem-specific extended Block-Jacobi preconditioners significantly outperform standard approaches. A visual comparison of the preconditioner matrices is presented in [181, Fig. 1].

In this thesis, we adopt a similar matrix-free strategy with extended Block-Jacobi preconditioning, following the methodology outlined in [181, Sec. 4.2]. The structure of the extended nonlinear systems is presented in the respective sections where they are applied. See [94, Paper I], [96, Paper III] and [120, Paper V].

1.7 Outline of the thesis

In Chapter 2, the peer-reviewed publications and submitted papers are presented.

- As mentioned in the introduction, the results presented by the authors regarding the HBPC scheme, in terms of error reduction, computational efficiency, and optimized stability properties for ordinary differential equations (ODEs) [125, 92, 167], were promising. In [181], the authors combined the serial HBPC scheme with the DGSEM spatial discretization for the Navier–Stokes equations, achieving convergence rates of up to eight for various test cases. In order to prevent the Newton error from dominating the overall solution error, solving the nonlinear systems with higher accuracy may be necessary. However, this can result in oversolving and lead to increased computational time. Additionally, in long-run simulations, the serial nature of the method becomes a significant constraint on overall efficiency.

  In [94, Paper I], we have presented a parallel-in-time HBPC scheme for the Navier–Stokes equations using the DGSEM spatial discretization. The code was implemented in FLEXI [116], which supports spatial parallelization. Therefore, the schemes are parallel in time and space.

  Additionally, we developed an adaptive Newton strategy in [94, Paper I] to avoid oversolving the nonlinear equations and thereby improve computational efficiency. The PinT schemes significantly reduce computational time compared to serial schemes. Numerical results demonstrated a parallel efficiency of approximately 60% to 70%.

  My contributions: Methodology, Validation, Investigation, Writing – review & editing.

- In previous works on HBPC schemes [125, 92, 181], including our work on PinT HBPC schemes for DGSEM [94, Paper I], higher orders of accuracy were achieved by utilizing a higher-order quadrature rule devised using intermediate stage values. However, increasing the number of intermediate stage values requires solving more implicit stage equations, which in turn raises the computa-

tional effort needed. To address this, we focus on two-derivative schemes where incorporating a multistep nature can reduce the need for intermediate stages. As a preliminary step, we explore the feasibility of one-stage, two-derivative multi-step predictor-corrector schemes.

In [95, Paper II], we have developed the m-step HBPC (mS-HBPC) scheme using a multi-step quadrature rule. Since the initial stability regions were poor, we analyzed the schemes and optimized to attain A(α)-stability. The optimization parameters were determined for various stability angles. Numerical results were presented for several stiff ODEs and diffusion-dominated PDE. The sixth-order schemes exhibited the expected convergence for all stability angles, while a reduction in order was observed in the eighth-order schemes for certain stability angles.

My contributions: Conceptualization, Data curation, Formal analysis, Investigation, Methodology, Software, Validation, Visualization, Writing – original draft, Writing – review & editing.

- The influence of stability properties on achieving stable solutions for ODEs and PDEs under various stiff conditions for HBPC schemes can be found in [167], and for mS-HBPC in [95, Paper II]. It is observed that carefully adjusting the coefficients of Δt and Δt^2 in the implicit part significantly enhances the stability of the scheme. As a result, two-derivative schemes offer greater flexibility in positioning the stability regions of the timestepping schemes compared to one-derivative schemes. We utilize this capability to improve the linear stability properties of the strong stability-preserving (SSP) schemes presented in [83].

  In [96, Paper III], we analyze the two-derivative strong stability preserving (SSP) schemes from [83] regarding their A(α)-stability properties. These schemes were then coupled with the DGSEM spatial discretization using the FLEXI code [116]. We proved that a third-order, two-stage diagonally implicit two-derivative scheme cannot be both SSP and A-stable. Numerical results are presented for both the Euler and Navier-Stokes equations. The second- and fourth-order SSP schemes exhibited the desired convergence; however, the third-order scheme

required finer timesteps to exhibit convergence. We further analyzed the third-order scheme by constructing a family of A(α)-stable third-order SSP schemes. Additionally, we developed a third-order adaptive SSP scheme that provides stable solutions for all time step sizes. We also utilized the tunability of stability regions in the two-derivative schemes to analyze the convergence anomalies exhibited by the fourth-order SSP scheme when applied to the Navier-Stokes equations.

My contributions: Conceptualization, Data curation, Formal analysis, Investigation, Methodology, Software, Validation, Visualization, Writing – original draft.

- Explicit schemes provide several advantages over implicit schemes, mainly because they do not require the solution of implicit systems. However, for stiff equations, implicit schemes become necessary due to the timestep limitations associated with explicit schemes. The use of implicit-explicit (IMEX) schemes can help reduce the complexity of the implicit part, making it easier to invert. Nonetheless, the challenge of inverting non-linear systems remains.

  In [164, Paper IV], we adopt an explicit approach to handle the rapidly changing stiff components, referred to as multirate schemes. This method allows us to compute the slow components using a Runge-Kutta scheme with a timestep Δt, while the fast components are computed using another Runge-Kutta scheme with a smaller timestep $\Delta\tau < \Delta t$.

  We extend the one-derivative multirate schemes in [154, 153] to a multi-derivative schemes. We derived the order conditions, and developed multi-derivative multirate schemes up to fourth order. We investigated the stability properties of the proposed schemes, and their performance was evaluated on a range of highly stiff ODEs and diffusion-dominated PDEs.

  My contributions: Conceptualization, Data curation, Formal analysis, Investigation, Methodology, Software, Validation, Visualization, Writing – original draft.

- Multirate schemes are well-suited for mildly stiff problems. However, in highly

stiff regimes, these schemes may require fast steps in the order of $\frac{1}{\varepsilon}$ to maintain stability, where ε is the stiffness parameter. IMEX schemes can be employed to overcome this limitation, as they alleviate the time step restrictions imposed by stiffness.

Nevertheless, constructing higher-order IMEX Runge-Kutta schemes is challenging due to the complexity of solving the associated order conditions. A key advantage of HBPC schemes is that they can be readily extended to IMEX schemes by replacing the predictor with an IMEX scheme. As a result, it is relatively straightforward to construct IMEX schemes of arbitrary order.

In [120, Paper V], we studied the second-order two-derivative IMEX Taylor method for the isentropic Euler equations using DeTa [126] flux splitting. We also investigated higher-order IMEX HBPC schemes for the same problem, using the IMEX Taylor method as the predictor. The spatial discretization was based on DGSEM. We proved that the overall schemes were asymptotic preserving (AP) under periodic boundary conditions and appropriate numerical fluxes. Numerical results demonstrated that the timestep sizes are independent of the stiffness parameter.

My contributions: Conceptualization, Data curation, Formal analysis, Investigation, Methodology, Software, Validation, Visualization, Writing – original draft.

Finally, in Chapter 3, the final conclusions of the thesis are presented, along with suggestions for future research directions.

Chapter 2

Publications

This thesis is based on the peer-reviewed publications and submitted papers listed below.

- [94, Paper I]: Zeifang, J., Thenery Manikantan, A., & Schütz, J. (2023). Time parallelism and Newton-adaptivity of the two-derivative deferred correction discontinuous Galerkin method. Applied Mathematics and Computation, 457, 128198. https://doi.org/10.1016/j.amc.2023.128198

- [95, Paper II]: Thenery Manikantan, A., & Schütz, J. (2024). Multi-step Hermite-Birkhoff predictor-corrector schemes. Applied Numerical Mathematics, 205, 281–295. https://doi.org/10.1016/j.apnum.2024.07.011

- [96, Paper III]: Thenery Manikantan, A., Zeifang, J., & Schütz, J.(2025) On the stability of two-derivative time discretizations. UHasselt Computational Mathematics Preprint, Hasselt University. https://www.uhasselt.be/media/4gbfm0hf/up2302.pdf

- [164, Paper IV]: Schütz, J., Ishimwe, A., Moradi, A., & Thenery Manikantan, A. (2024) A class of multirate multiderivative schemes. UHasselt Computational Mathematics Preprint, Hasselt University. (Major revision). https://www.uhasselt.be/media/5fopnoey/up2403.pdf

- [120, Paper V]: Thenery Manikantan, A., & Schütz, J.(2025) Two-derivative schemes for low-Mach flows. UHasselt Computational Mathematics Preprint, Hasselt University. (Accepted for publication in Communications in Computational Physics). https://www.uhasselt.be/media/hzonmb0o/paper.pdf

The articles [94, Paper I], [95, Paper II], and [120, Paper V] are accepted papers. The articles [96, Paper III] and [164, Paper IV] are submitted papers, with the latter currently under major revision.

The papers are presented in the same order as listed above. The styling and formatting of each article have been preserved exactly as they appeared in their original journals. As a result, many commonly used mathematical symbols may vary between papers and other chapters of this thesis. The data and code used in the publications are available upon request.

Paper I:

Time parallelism and Newton-adaptivity of the two-derivative deferred correction discontinuous Galerkin method

Applied Mathematics and Computation 457 (2023) 128198

Contents lists available at ScienceDirect

Applied Mathematics and Computation

journal homepage: www.elsevier.com/locate/amc

Time parallelism and Newton-adaptivity of the two-derivative deferred correction discontinuous Galerkin method

Jonas Zeifang, Arjun Thenery Manikantan, Jochen Schütz*

Faculty of Sciences & Data Science Institute, Hasselt University, Agoralaan Gebouw D, Diepenbeek BE-3590, Belgium

ARTICLE INFO

Article history:
Received 27 September 2022
Revised 17 May 2023
Accepted 18 June 2023

Keywords:
Implicit time stepping
Parallel-in-Time
Multiderivative schemes
Newton adaptivity

ABSTRACT

In this work, we consider a high-order discretization of compressible viscous flows allowing parallelization both in space and time.

The discontinuous Galerkin spectral element method, which is well-suited for massively parallel simulations, is used for spatial discretization. The main novelty in this work is the additional demonstration of time-parallel capabilities within an implicit two-derivative timestepping procedure to further increase the parallel speedup. Temporal parallelism is made possible by a predictor-corrector-type time discretization that allows to split the associated workload onto multiple processors.

We identify a homogeneous load balance with respect to the linear (GMRES) iterations on each processor as a key for parallel efficiency. To homogenize the load and to enable practical simulations, an adaptive strategy for Newton's method is introduced. It is shown that the time-parallel method provides a parallel efficiency of approx. $60-70\%$ on $4-7$ computational partitions. Moreover, the capabilities of the novel method for the simulation of large-scale problems are illustrated with a mixed temporal and spatial parallelization on more than 1000 processors.

1. Introduction

In this work, we are interested in solving the compressible Navier-Stokes equations, which can be cast into flux formulation

$$\mathbf{w}_t + \nabla_x \cdot \left(\mathbf{F}(\mathbf{w}) - \mathbf{F}^v(\mathbf{w}, \nabla_x \mathbf{w})\right) = 0, \quad \text{with} \quad \mathbf{w} = \begin{pmatrix} \rho \\ \rho \mathbf{v} \\ E \end{pmatrix}, \tag{1}$$

for the unknown quantities density ρ, velocity $\mathbf{v}$ and energy E. Note that we have closed the system by defining the pressure p via the ideal gas equation of state with the isentropic coefficient $\gamma = 1.4$ and reference Mach number ε. For a precise definition of the fluxes, consult Appendix A. All occuring quantities are non-dimensionalized.

In this work, we are interested in a parallel algorithm for the temporal discretization of Eq. (1). Upon defining

$$\mathbf{R}^{(1)}(\mathbf{w}) := -\nabla_x \cdot \left(\mathbf{F}(\mathbf{w}) - \mathbf{F}^v(\mathbf{w}, \nabla_x \mathbf{w})\right), \tag{2}$$

* Corresponding author.
E-mail addresses: jonas.zeifang@uhasselt.be (J. Zeifang), arjun.thenerymanikantan@uhasselt.be (A. Thenery Manikantan), jochen.schuetz@uhasselt.be (J. Schütz).

https://doi.org/10.1016/j.amc.2023.128198
0096-3003/

Eq. (1) can be cast as an ODE in some infinite-dimensional function space,

$$\mathbf{w}_t = \mathbf{R}^{(1)}(\mathbf{w}). \tag{3}$$

While classical timestepping methods only make use of the information of the first time derivative $\mathbf{w}_t$, the idea of two-derivative schemes is to additionally make use of the second temporal derivative. This adds an extra degree of freedom to the discretization and hence facilitates the development of storage- and runtime efficient high-order schemes. The second temporal derivative of $\mathbf{w}$ can be obtained by differentiating Eq. (1),

$$\mathbf{w}_{tt} = \mathbf{R}^{(2)}(\mathbf{w}, \mathbf{R}^{(1)}(\mathbf{w})), \tag{4}$$

where $\mathbf{R}^{(2)}$ for the Navier-Stokes equations is defined through

$$\mathbf{R}^{(2)}(\mathbf{w}, \mathbf{R}^{(1)}(\mathbf{w})) := \quad -\nabla_x \cdot \Big(\frac{\partial \mathbf{F}}{\partial \mathbf{w}} \mathbf{R}^{(1)}(\mathbf{w}) - \frac{\partial \mathbf{F}^v}{\partial \mathbf{w}} \mathbf{R}^{(1)}(\mathbf{w}) - \frac{\partial \mathbf{F}^v}{\partial \nabla_x \mathbf{w}} \nabla_x \mathbf{R}^{(1)}(\mathbf{w}) \Big). \tag{5}$$

For more details on the derivation of $\mathbf{w}_{tt}$, consult [1]. In [2], a novel class of implicit two-derivative deferred correction time discretization methods has been introduced. The concept is based on a predictor-corrector formulation and can - in principle - achieve arbitrary orders. After a predictor step based on the two-derivative Taylor method, successive correction steps improve the solution towards a background two-derivative Hermite-Birkhoff Runge-Kutta method, giving rise to the name Hermite-Birkhoff predictor-corrector methods (HBPC). In [3], HBPC schemes up to order 8 have been numerically investigated. The schemes are A(α)-stable with stability angles α close to 90°, see [4]. Recently, these schemes have been combined with a high order discontinuous Galerkin spectral element spatial discretization of the Euler and Navier-Stokes equations [1].

A common strategy to enable large-scale simulations of discretizations of Eq. (1) is the use of spatial parallelization. It typically comes with high parallel efficiencies. However, caused by an increase of the communication to computation ratio, the spatial parallelization tends to saturate as the assigned work per processor decreases. This has been observed by various authors, see e.g. [5–7]. One remedy is to additionally consider the parallelization of the temporal domain, which requires specifically designed strategies due to the causality principle. It has been shown that combining temporal and spatial parallelization can further reduce the required wallclocktimes, see e.g. [8–10]. An overview on parallel-in-time (PinT) algorithms can be found in the review articles [11] and [12]. Further literature and information can also be found on the PinT web page [13].

One particularly attractive property of the HBPC methods is that they offer a mild time parallelism. This class of time parallel methods is sometimes classified as "parallel-across-the-method" [14] or "direct time-parallel methods" [12]. This time-parallelism is based on the idea of distributing different correction steps to different processors, and has been introduced in [15], but has also been used for the RIDC (revisionist integral deferred correction) schemes in [16,17]. While being limited to a mild parallelization, i.e. using $\mathcal{O}(10)$ processors at maximum, this concept offers good parallel efficiencies [17]. Also for the HBPC schemes, a good speedup in computational time has been observed when solving ODEs, see [3]. One prerequisite for a good parallel speedup of this type of parallelization is equally expensive prediction/correction steps. However, already for the ODE examples investigated in [3], a large discrepancy of the computational work of the different prediction/correction steps has been observed. This is due to the different costs of the solution of the algebraic systems of equations in the prediction/corrections steps. This non-homogeneous work distribution also transfers to the Navier-Stokes equations discretized with the discontinuous Galerkin spectral element method. We illustrate this with an introductory example that describes an advection-diffusion process of a density sine-wave, see Eq. (25) for initial conditions. The same simulation setup as described in [1, Sec. 5.2.] is used. The required number of GMRES iterations per prediction/correction step is reported in Fig. 1.

One can see that especially the predictor (and, to a less extent, also the first correction step) requires significantly more computational work than the other correction steps. Therefore, in order to achieve a good parallel speedup when distributing different prediction/correction steps to different processors, one has two opportunities: try to harmonize the computational work and/or to develop a parallelization strategy that takes the different costs of the different iterates into account.

In this work, we harmonize the computational work per processor through a novel parallelization strategy where, in addition to the parallelization over the correction steps, there is also a parallelization over the stages of the predictor and the first corrector. Furthermore, to use computational resources as efficiently as possible, a novel strategy to adaptively determine the amount of Newton steps is developed. It is shown through several numerical testcases that this leads to a work distribution that is more homogeneous and hence more efficient than the straightforward application of the scheme in [1]. Time-parallel efficiencies of 60–70% are demonstrated. Also comparisons to established ESDIRK schemes are being made. As such, the main contributions of this work can be summarized as follows:

- An adaptive strategy for the Newton procedure, including a reliable error estimator, is developed in the context of the HBPC-DGSEM-methods. The strategy is numerically investigated.
- A parallelization strategy for the HBPC-DGSEM-methods that balances the loads over the different processors more evenly is developed.
- The actual parallel speedup is thoroughly investigated numerically.

The remainder of this paper is structured as follows: In Section 2 the implicit two-derivative predictor-corrector time discretization method and its temporal parallelization strategy are introduced. The fully discrete scheme is summarized in

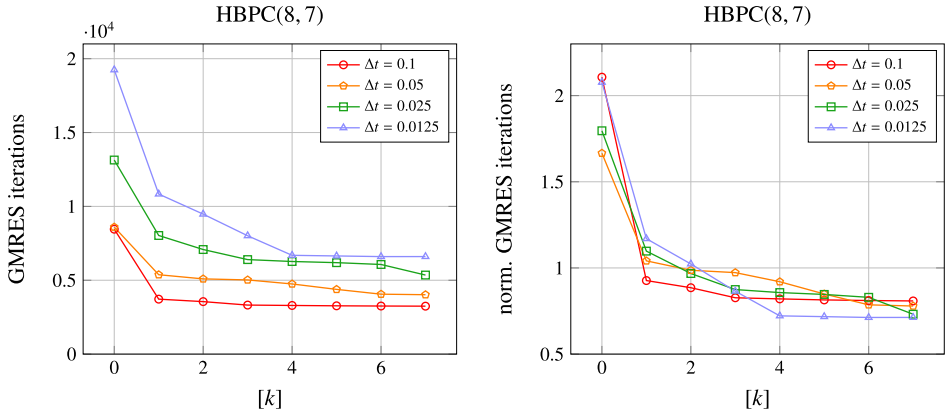


Fig. 1. Cumulated (left) and normalized (right) number of GMRES iterations per prediction/correction step for the Navier-Stokes example described in [1, Sec. 5.2.] using the serial HBPC(8, 7) scheme [3] with different timestep sizes for the temporal discretization. Normalization (right) has been done with the mean number of GMRES iterations per timestep. $[k]$ denotes the number of the correction step, [0] corresponding to the predictor.

Section 3. In order to homogenize the computational work and to enable efficient simulations, an adaptive strategy for the non-linear solver is introduced in Section 4. After having introduced all the ingredients of the novel method, its parallel performance and its efficiency compared to established serial methods is evaluated in Section 5. Finally, conclusion and outlook are given in Section 6.

2. Parallel-in-Time HBPC method

2.1. The Hermite-Birkhoff predictor-corrector method

The parallel-in-time algorithm described in this paper is based on the two-derivative deferred correction method introduced in [2] and [3], which relies on the approximate quantities

$$\mathbf{w}^{n,[k],l} \approx \mathbf{w}(t^n + c_l \Delta t), \quad 0 \le n \le \mathcal{N}_T, \quad 0 \le k \le k_{\max}, \quad 1 \le l \le s.$$

Here, $\mathcal{N}_T$ is the number of discrete time levels, c_l a Runge-Kutta-type relative timestep of an s-stage Runge-Kutta method and $k_{\max}$ denotes the number of correction steps of the underlying deferred correction procedure. The s-stage Runge-Kutta methods are given by their two-derivative Butcher tableaux consisting of typically dense matrices $B^{(1)}$, $B^{(2)} \in \mathbb{R}^{s\times s}$ and a vector $c \in \mathbb{R}^s$. They define the background Hermite-Birkhoff Runge-Kutta scheme and are given in the appendix, Eq. (B.1) and Eq. (B.3). More details can be found in [3]. The coefficients of the Butcher tableaux define a quadrature formula $\mathcal{I}_l$ of order q through

$$\mathcal{I}_l\left(\mathbf{w}^1, \dots, \mathbf{w}^s\right) := \Delta t \sum_{j=1}^{s} B_{lj}^{(1)} \mathbf{R}^{(1)}(\mathbf{w}^j) + \Delta t^2 \sum_{j=1}^{s} B_{lj}^{(2)} \mathbf{R}^{(2)}(\mathbf{w}^j) \tag{6}$$

for every stage $1 \le l \le s$. Note that we have omitted the additional dependencies of $\mathbf{R}^{(1)}$ and $\mathbf{R}^{(2)}$ given by Eq. (4) and Eq. (5) for the sake of brevity.

We use the parallel-in-time HBPC method according to [3, Alg. 2] and its improvement according to [4]. The predictor requires more computational load than higher correction steps, we have therefore modified the algorithm such that it allows for a parallelization of the stages for the predictor and the first correction step. The modifications in comparison to [4] have been marked in red color, they only apply to the definition of the quadrature rule. Note that while the original algorithm in [3] offers the possibility to use an IMEX splitting, here, only the implicit part is considered.

Algorithm 1. (HBPC$(q, k_{\max})$) To advance the solution to Eq. (3) in time, we compute values $\mathbf{w}^{n,[k],l}$. To account for the initial conditions $\mathbf{w}_0$, define

$$\mathbf{w}^{-1,[k],s} := \mathbf{w}_0.$$

First, the values $\mathbf{w}^{n,[0],l}$ are filled using a straightforward second-order implicit Taylor method departing from $\mathbf{w}^{n-1,[1],s}$.

1. **Predict.** Solve the following expression for $\mathbf{w}^{n,[0],l}$ and each $2 \le l \le s$:

$$\mathbf{w}^{n,[0],1} := \mathbf{w}^{n-1,[1],s},$$
$$\mathbf{w}^{n,[0],l} := \mathbf{w}^{n-1,[1],s} + c_l \Delta t \mathbf{R}^{(1)}(\mathbf{w}^{n,[0],l}) - \frac{(c_l \Delta t)^2}{2} \mathbf{R}^{(2)}(\mathbf{w}^{n,[0],l}). \tag{7}$$

2. **Correct.** Next, the corrected values $\mathbf{w}^{n,[k],l}$ for $1 \leq k \leq k_{\max}$ are computed through solving for each $2 \leq l \leq s$ and each $1 \leq k \leq k_{\max}$:

$$\mathbf{w}^{n,[k],1} := \mathbf{w}^{n-1,[k+1],s},$$
$$\mathbf{w}^{n,[k],l} := \mathbf{w}^{n-1,[k+1],s} + \theta_1 \Delta t \left(\mathbf{R}^{(1)}(\mathbf{w}^{n,[k],l}) - \mathbf{R}^{(1)}(\mathbf{w}^{n,[k-1],l})\right) - \theta_2 \frac{\Delta t^2}{2}\left(\mathbf{R}^{(2)}(\mathbf{w}^{n,[k],l}) - \mathbf{R}^{(2)}(\mathbf{w}^{n,[k-1],l})\right) + \mathcal{I}_l, \quad (8)$$

with

$$\mathcal{I}_l := \mathcal{I}_l\left(\mathbf{w}^{n,[0],1}, \dots, \mathbf{w}^{n,[0],s}\right), \qquad \text{for} \quad k = 1,$$
$$\mathcal{I}_l := \mathcal{I}_l\left(\mathbf{w}^{n,[k],1}, \dots, \mathbf{w}^{n,[k],l-1}, \mathbf{w}^{n,[k-1],l}, \dots, \mathbf{w}^{n,[k-1],s}\right), \quad \text{for} \quad k > 1. \quad (9)$$

$\mathcal{I}_l(\cdot)$ denotes the q-th order Hermite-Birkhoff quadrature rule given in Eq. (6). If $k = k_{\max}$, then the $k+1$ superscripts in Eq. (8) are replaced by $k_{\max}$ in order to close the recursion.

3. **Update.** In order to retain a first-same-as-last property, we update the solution with

$$\mathbf{w}^{n+1} := \mathbf{w}^{n,[k_{\max}],s}. \quad (10)$$

The coefficients $\theta = (\theta_1, \theta_2)$ are obtained by an optimization of the stability region, see [4]. For Alg. 1 with the Butcher tables given in Eq. (B.1) and Eq. (B.3) we find

$$\theta = (0.296, 0.0531) \quad \text{and} \quad \theta = (0.259, 0.0288) \quad (11)$$

for the sixth and the eighth order quadrature rules, respectively. The resulting methods are $A(\alpha)$-stable with the stability angles $\alpha > 89.81^\circ$ (HBPC$(6, k_{\max})$) and $\alpha > 88.66^\circ$ (HBPC$(8, k_{\max})$).

Remark 1. Please note that for efficiency considerations, we only treat background schemes with an explicit first stage. Furthermore, the last stage corresponds to collocation point $c_s = 1$. Strictly speaking, this is not necessary; the update step (10) has then to be modified accordingly.

Remark 2. For $k > 1$, we use a Gauß-Seidel type procedure in the quadrature formula (9). Obviously, this could be done for $k = 1$ as well. However, the way we have formulated it in Alg. 1 makes it possible to parallelize over the stages for $k = 0$ and $k = 1$. This concept was not present in the original work [3]. The parallelization concept will be described in the next section.

2.2. Parallelization of the HBPC method

The structure of Algorithm 1 allows to distribute the predictor and the correction steps on multiple processors, see [3]. The underlying basic idea of pipelining has been introduced in [15] and has also been used by the RIDC schemes [16–18].

Although the main ingredients of the parallelization of Algorithm 1 have been already introduced in [3], we summarize them here and describe the differences of the present algorithm. The keys to parallelize Algorithm 1 are:

- The stages of the prediction step at time instance n, i.e. $\mathbf{w}^{n,[0],l}$ only depend on the single value $\mathbf{w}^{n-1,[1],s}$ of the previous timestep. Hence, the different stages of the predictor can be calculated independently of each other.
- As the quadrature rule for the first correction step, i.e. $\mathbf{w}^{n,[1],l}$, only depends on values of the predictor, the different stages of the first correction step can also be calculated independently of each other.
- For $1 \leq k < k_{\max}$ the $[k]$-th correction step at time instance n, i.e. $\mathbf{w}^{n,[k],l}$, depends on the $[k-1]$-th iterate at the same time level n, as well as on the $[k+1]$-th correction step at the previous time step, $\mathbf{w}^{n-1,[k+1],s}$, see Eq. (8).
- The last correction step $[k_{\max}]$, i.e. $\mathbf{w}^{n,[k_{\max}],l}$ for $1 \leq l \leq s$, depends only on the $[k_{\max}-1]$-th iterate at the same time level n, as well as on the last correction iterate of the previous time level, $\mathbf{w}^{n-1,[k_{\max}],s}$.

The dependencies described above are visualized in Fig. 2 at the example of the sixth-order method. On the y-axis the different correction levels $0 \leq k \leq k_{\max}$ are illustrated, while on the x-axis, the different time levels $n, n+1, \dots$ are indicated. A full circle at position (n, k) corresponds to the computation of all stages of $\mathbf{w}^{n,[k],l}$, $l = 2, \dots, s$. (Calculation of the first stage $l = 1$ is trivial, see Eq. (8).) Splitted circles indicate computations of only one specific stage $l > 1$ of $\mathbf{w}^{n,[k],l}$. Note that the sixth order quadrature rule has two implicit stages; we hence split the circle in two semi-circles[1]. Numbers inside (semi-)circles indicate when the corresponding calculations can be performed: (semi-) circles with the same number can be computed at the same time in parallel, while those with a higher number have to wait for those with a lower number to finish.

[1] It should be three semi-circles for the eighth-order method.

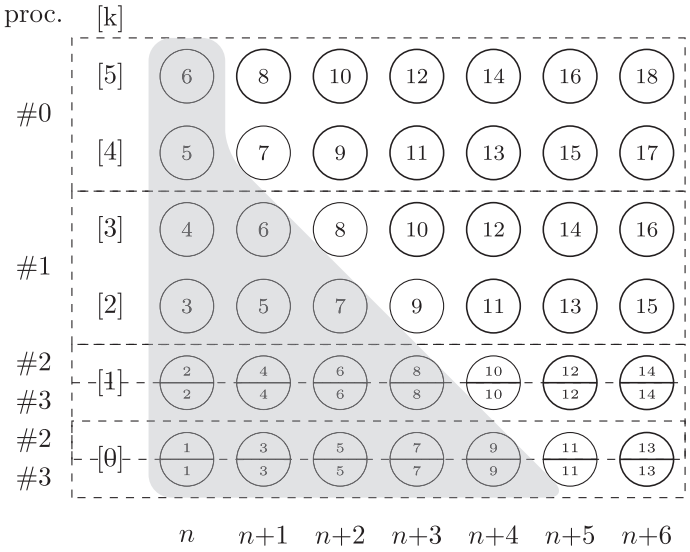


Fig. 2. Schematic overview on parallelization strategy of HBPC(6, 5) method. The parallelization is according to [3], with the only difference that for $k = 0$ and $k = 1$, parallelization is also done over the stages, which is indicated by semi-circles. Note that the sixth order quadrature rule has two implicit stages; we hence split the circle in two semi-circles. At the left, the processor index #i and the current iterate [k] are indicated. On the x-axis, time instances $n, n+1, \ldots$ are visualized. Numbers inside (semi-)circles indicate when the corresponding calculations can be performed. The gray-shaded area highlights solution steps where no adaptive Newton strategy can be performed, see Remark 11.

The main difference of the parallelization strategy performed here and the one described in [3, Alg. 2] is that we exploit the independence of different stages for the prediction and the first correction step. This is inspired by the observation that the calculation of the predictor and the first correction step is typically more expensive than the remaining correction steps, see [3]. This is also true for the PDE discretization considered in this work, see Fig. 1. The adaptive Newton strategy, which is described later in Section 4, will sharpen this observation, see Fig. 5. From Fig. 2 one can see that if one groups the correction iterates [k] and [$k+1$], one obtains consecutively numbered circles on all processors. For the predictor and the first correction step this is done in an analogous way, i.e. the predictor and corrector of one specific stage $l > 1$ are grouped together. The processor boundaries resulting from this grouping are visualized with dashed lines in Fig. 2.

Finally, one can see that each processor contains consecutively numbered circles, i.e. if communication is instantaneous and all calculations indicated with a (semi-)circle are equally expensive, there is no processor idle time.

Remark 3. The underlying assumption behind this is that *solving for one stage of the predictor or the first corrector has the same cost as solving for all stages of one of the following correction steps ($k > 1$).* While this is of course not true in a mathematically rigorous way, our numerical experience, see also Fig. 1, indicates that this assumption is reasonable.

Hence, the total amount of work packages per timestep is $2(s-1) + k_{\max} - 1$, where $2(s-1)$ work packages stem from the predictor and the first corrector, and $k_{\max} - 1$ work packages are due to the following correction steps. Note again that we have directly assumed that the calculation of the first stage is trivial. For the evaluation of the temporal parallelization's maximum achievable speedup, one additionally has to find the relation between the total amount of work packages and the work packages on a single processor where the initial startup phase is taken into account. While the amount of work packages on one single processor is $2\mathcal{N}_T$, the startup phase takes $k_{\max} - 1$ work packages until the processor with index #0 can start. Under those assumptions the maximum achievable speedup can be calculated by

$$\frac{\mathcal{N}_T(2(s-1) + k_{\max} - 1)}{2\mathcal{N}_T + k_{\max} - 1} \to \frac{k_{\max} + 1}{2} + s - 2, \quad \mathcal{N}_T \to \infty. \tag{12}$$

3. Fully discrete method

3.1. Two-derivative discontinuous Galerkin method

After having introduced the temporal discretization procedure, a spatial discretization of $\mathbf{R}^{(1)}$ and $\mathbf{R}^{(2)}$ is needed. In [19] it has been shown that a careful discretization of the second derivative operator $\mathbf{R}^{(2)}$ is required to retain the stability properties of the ODE integrator as it is desirable for a method-of-lines approach. This idea from [19] has been formulated for

a Discontinuous Galerkin Spectral Element Method (DGSEM [20]) discretization of nonlinear equations in [1]. Here, we will only very briefly recall this discretization for a purely hyperbolic PDE and refer the reader to [1] and the references therein for more details. The DGSEM is based on the weak formulation of Eq. (1),

$$\sum_{e=1}^{\mathcal{N}_E} (\mathbf{w}_t, \phi)_{\Omega_e} - (\mathbf{F}(\mathbf{w}), \nabla_x \phi)_{\Omega_e} + \left\langle \mathbf{F}^*(\mathbf{w}^L, \mathbf{w}^R) \cdot \mathbf{n}, \phi \right\rangle_{\partial\Omega_e} = 0, \qquad \forall \phi \in \Pi_{\mathcal{N}_p}, \tag{13}$$

where the function space $\Pi_{\mathcal{N}_p}$ of the test functions ϕ is the tensor-product of the one-dimensional Lagrange polynomials ℓ, each of degree $\mathcal{N}_p$. The domain Ω is split into $\mathcal{N}_E$ non-overlapping hexahedral (3d) or quadrangular (2d) elements. The integration over an element $\Omega_e \in \Omega$ is denoted by the scalar product $(\cdot,\cdot)$ and integration over the cell edges $\partial\Omega_e$ is denoted by $\langle\cdot,\cdot\rangle$. For the evaluation of the surface integral, the flux is substituted by a numerical flux $\mathbf{F}^*$, depending on the values of both adjacent elements of the edge ($\mathbf{w}^L$ and $\mathbf{w}^R$) and the outward pointing normal vector $\mathbf{n}$ of the current element. The numerical flux is chosen to be a global Lax-Friedrichs, see [1, Eq. (13)]. Using DGSEM techniques on (13), see [21], yields the discrete operator $\mathbf{R}_h^{(1)}(\mathbf{w}_h)$ as an approximation to $\mathbf{R}^{(1)}(\mathbf{w})$.

The second derivative operator $\mathbf{R}^{(2)}$ is defined through the artificial quantity

$$\boldsymbol{\sigma} := \mathbf{R}^{(1)}(\mathbf{w}) \equiv \mathbf{w}_t. \tag{14}$$

In [1] a DGSEM discretization of the second temporal derivative has been proposed via the weak formulation

$$\sum_{e=1}^{\mathcal{N}_E} (\mathbf{w}_{tt}, \phi)_{\Omega_e} - \left(\frac{\partial \mathbf{F}(\mathbf{w})}{\partial \mathbf{w}} \boldsymbol{\sigma}, \nabla_x \phi \right)_{\Omega_e} + \left\langle \frac{\partial \mathbf{F}^*(\mathbf{w}^L, \mathbf{w}^R)}{\partial \mathbf{w}^L} \boldsymbol{\sigma}^L \cdot \mathbf{n} + \frac{\partial \mathbf{F}^*(\mathbf{w}^L, \mathbf{w}^R)}{\partial \mathbf{w}^R} \boldsymbol{\sigma}^R \cdot \mathbf{n}, \phi \right\rangle_{\partial\Omega_e} = 0, \qquad \forall \phi \in \Pi_{\mathcal{N}_p}. \tag{15}$$

Note that the discretization of the second derivative operator is similar to the first derivative operator except for the flux which has to be substituted by $\partial \mathbf{F}(\mathbf{w})/\partial \mathbf{w} \cdot \boldsymbol{\sigma}$ (compare Eq. (13) and Eq. (15)). In analogy to the first derivative operator we obtain the discrete operator $\mathbf{R}_h^{(2)}(\mathbf{w}_h, \boldsymbol{\sigma}_h)$ for the second temporal derivative.

Considering the Navier-Stokes equations, see Eq. (1), second order spatial derivatives occur by the introduction of the viscous flux $\mathbf{F}^v(\mathbf{w}, \nabla_x \mathbf{w})$. They are discretized by following the BR2 lifting approach [22]. A detailed description of how the Navier-Stokes equations can be handled with the two-derivative DGSEM can be found in [1].

3.2. Solving for the stage values

In Section 2.1 and Section 3.1 we have introduced the temporal and the spatial discretization, respectively. Bringing both together, one has to solve for the stages $l > 1$ of the predictor and the correction steps in Eq. (7) and Eq. (8). The resulting non-linear system to be solved is very similar for the predictor and the corrector (see also [1, Sec. 3.2.1.]). Due to the non-linearity of the considered systems of equations, one has to use some non-linear solution procedure. As it is common for time-dependent PDE discretizations, we use Newton's method for that purpose.

We start by casting the predictor and corrector step (Eq. (7) and Eq. (8)) for the current timestep n, iterate k and stage l into a uniform formulation. Due to the introduction of the quantity $\boldsymbol{\sigma}_h^{n,[k],l} := \mathbf{R}_h^{(1)}(\mathbf{w}_h^{n,[k],l})$, we have to extend the state vector to consist of the discrete $\mathbf{w}_h^{n,[k],l}$ and $\boldsymbol{\sigma}_h^{n,[k],l}$, i.e. we introduce $\boldsymbol{X}^{[k]} := \left(\mathbf{X}_w^{[k]}, \mathbf{X}_\sigma^{[k]}\right)^T := \left(\mathbf{w}_h^{n,[k],l}, \boldsymbol{\sigma}_h^{n,[k],l}\right)^T$. The non-linear equation to be solved can then be written as

$$\boldsymbol{X}^{[k]} \stackrel{!}{=} \begin{pmatrix} \mathbf{w}_{\text{old}} \\ 0 \end{pmatrix} + \begin{pmatrix} \Phi\left(\boldsymbol{X}^{[k]}, \boldsymbol{X}^{[k-1],1:s}\right) \\ \mathbf{R}_h^{(1)}(\mathbf{w}_h^{n,[k],l}) \end{pmatrix} =: \boldsymbol{W}_{\text{old}} + \bar{\Phi}\left(\boldsymbol{X}^{[k]}, \boldsymbol{X}^{[k-1],1:s}\right), \tag{16}$$

with $\mathbf{w}_{\text{old}} := \mathbf{w}_h^{n-1,[k+1],s}$ for $k < k_{\max}$ and $\mathbf{w}_{\text{old}} := \mathbf{w}_h^{n-1,[k_{\max}],s}$ for $k = k_{\max}$. For the sake of notation, we use the abbreviation $\boldsymbol{X}^{[k-1],1:s} := \left(\boldsymbol{X}^{[k-1],1}, \boldsymbol{X}^{[k-1],2}, \dots, \boldsymbol{X}^{[k-1],s}\right)$. For the predictor, Φ is given by

$$\Phi\left(\boldsymbol{X}^{[k]}, \boldsymbol{X}^{[k-1],1:s}\right) := c_l \Delta t \mathbf{R}_h^{(1)}(\mathbf{w}_h^{n,[k],l}) - \frac{(c_l \Delta t)^2}{2} \mathbf{R}_h^{(2)}(\mathbf{w}_h^{n,[k],l}, \boldsymbol{\sigma}_h^{n,[k],l}), \tag{17}$$

and for the first corrector step by

$$\begin{aligned} \Phi\left(\boldsymbol{X}^{[k]}, \boldsymbol{X}^{[k-1],1:s}\right) := {} & \theta_1 \Delta t \mathbf{R}_h^{(1)}(\mathbf{w}_h^{n,[k],l}) - \frac{\theta_2 \Delta t^2}{2} \mathbf{R}_h^{(2)}(\mathbf{w}_h^{n,[k],l}, \boldsymbol{\sigma}_h^{n,[k],l}) \\ & - \theta_1 \Delta t \mathbf{R}_h^{(1)}(\mathbf{w}_h^{n,[k-1],l}) + \frac{\theta_2 \Delta t^2}{2} \mathbf{R}_h^{(2)}(\mathbf{w}_h^{n,[k-1],l}, \mathbf{R}_h^{(1)}(\mathbf{w}_h^{n,[k-1],l})) + \mathcal{I}_l\left(\mathbf{w}_h^{n,[k-1],1:s}\right). \end{aligned} \tag{18}$$

Remark 4. For the ease of presentation, we did not distinguish between the treatment of the quadrature rule $\mathcal{I}_l$ for $k = 1$ and $k > 1$, see (9). The treatment for $k > 1$ results in slightly different arguments of the quadrature formula and hence additional arguments in Φ. The modifications are straightforward and do not change the proposed arguments here, yet they make the notation more clumsy.

We use Newton's method to solve equations of type (16), in this particular case given by:

1. For r = 1, ... solve

$$\left(\mathrm{Id} - \frac{\partial\overline{\Phi}\left(\boldsymbol{X}_{r-1}^{[k]}, \boldsymbol{X}_{r}^{[k-1],1:s}\right)}{\partial \boldsymbol{X}^{[k]}}\right)\Delta\boldsymbol{X}_r = \boldsymbol{W}_{\text{old}} + \overline{\Phi}\left(\boldsymbol{X}_r^{[k]} - 1, \boldsymbol{X}_r^{[k-1],1:s}\right) - \boldsymbol{X}_{r-1}^{[k]}$$
$$\boldsymbol{X}_r^{[k]} = \boldsymbol{X}_{r-1}^{[k]} + \Delta\boldsymbol{X}_r. \tag{19}$$

2. If the convergence criterion is met, set

$$\mathbf{w}_h^{n,[k],l} := \boldsymbol{X}_{\mathrm{r,w}}^{[k]} \quad \text{and} \quad \boldsymbol{\sigma}_h^{n,[k],l} := \mathbf{R}_h^{(1)}(\mathbf{w}_h^{n,[k],l}). \tag{20}$$

Note that we have indicated that solutions from the previous correction step, i.e. $\boldsymbol{X}^{[k-1],1:s}$, are obtained via Newton's method terminated at some finite Newton iterate r′, which can be different for different stages and different k. To initialize the iterative procedure, some initial guess

$$\boldsymbol{X}_0^{[k]} \equiv \left(\mathbf{w}_{h,0}^{n,[k],l}, \mathbf{R}_h^{(1)}\left(\mathbf{w}_{h,0}^{n,[k],l}\right)\right)^T$$

has to be specified.

Remark 5. If not stated otherwise, we choose an explicit second order Taylor step to obtain the initial guess for the predictor. For the correction step $[k]$, the corresponding stage value of the previous iterate $[k-1]$ is used, i.e.

$$\begin{aligned}
\mathbf{w}_{h,0}^{n,[0],l} &= \mathbf{w}_{\text{old}} + c_l\Delta t\mathbf{R}_h^{(1)}(\mathbf{w}_{\text{old}}) + (c_l\Delta t)^2\mathbf{R}_h^{(2)}(\mathbf{w}_{\text{old}}, \mathbf{R}_h^{(1)}(\mathbf{w}_{\text{old}})) && \text{for } l = 2, \dots, s.\\
\mathbf{w}_{h,0}^{n,[k],l} &= \mathbf{w}_h^{n,[k-1],l} && \text{for } l = 2, \dots, s, \text{ and } k = 1, \dots, k_{\max}.
\end{aligned} \tag{21}$$

We have observed that using a second order explicit Taylor step to obtain an initial guess for the predictor is superior to performing an explicit first order step. However, using a third order Taylor step did not give noticeable advantages. Please note that the effectiveness of using the second order step remains problem- and timestep-dependent.

Remark 6. Please note that at the end of the Newton algorithm, we define $\boldsymbol{\sigma}_h^{n,[k],l} := \mathbf{R}_h^{(1)}(\mathbf{w}_h^{n,[k],l})$ in Eq. (20) rather than setting $\boldsymbol{\sigma}_h^{n,[k],l} = \boldsymbol{X}_{\mathrm{r},\sigma}^{[k]}$. If the Eq. (16) is solved exactly, $\boldsymbol{\sigma}_h^{n,[k],l}$ would be identical to $\mathbf{R}_h^{(1)}(\mathbf{w}_h^{n,[k],l})$. However, it is only solved to a certain accuracy, and hence, the identity does not necessarily hold. The definition in Eq. (20) avoids inconsistencies in $\mathbf{w}_h$, $(\mathbf{w}_h)_t$ and $(\mathbf{w}_h)_{tt}$ during the timestepping procedure and potential instabilities. Because of this, $\boldsymbol{\sigma}_h$ of a previous timestep, stage or correction iterate is no longer an independent variable and hence does not occur as an explicit argument in $\mathbf{R}_h^{(2)}(\mathbf{w}_h^{n,[k-1],l}, \mathbf{R}_h^{(1)}(\mathbf{w}_h^{n,[k-1],l}))$ and in $\mathcal{I}_l\left(\mathbf{w}_h^{n,[k-1],1:s}\right)$ in Eq. (18).

In order to solve the arising linear system in Eq. (19), we use the matrix-free GMRES approach with extended block-Jacobi preconditioning described in [1]. As initial condition for the GMRES method a zero vector is chosen. Choosing the negative right hand side times the timestep as initial guess as suggested in [23] can sometimes be advantageous. Similar as the authors in [23], we observed that this advantage is problem dependent and can in some cases have an unfavorable influence on the required iterations, which is especially the case for large timesteps. We therefore use the zero vector as initial conditions in all simulations performed in this work. Similar as it has been done in [1], we neglect the Hessian contribution in Eq. (19), when solving the linear system.

3.3. Error estimator of the HBPC method

Controlling the numerical error introduced through the integration scheme per timestep is obviously crucial for multiple purposes. When using classical implicit Runge-Kutta methods, an error estimate is typically obtained via an embedded quadrature rule, see e.g. [24,25]. This embedded quadrature rule uses the same nodes as the original scheme but utilizes different weights. This offers the opportunity to obtain either higher or lower order embedded schemes, see e.g. [25]. Inspired by these error estimates for Runge-Kutta methods, we define additional quadrature rules of order $\hat{q} = q - 1$ for the HBPC$(6, k_{\max})$ and the HBPC$(8, k_{\max})$ method,

$$\hat{\mathcal{I}}_l\left(\mathbf{w}^1, \dots, \mathbf{w}^s\right) := \Delta t\sum_{j=1}^{s}\hat{B}_{lj}^{(1)}\mathbf{R}^{(1)}(\mathbf{w}^j) + \Delta t^2\sum_{j=1}^{s}\hat{B}_{lj}^{(2)}\mathbf{R}^{(2)}(\mathbf{w}^j).$$

The coefficients of the tables $\hat{B}^{(1)}$ and $\hat{B}^{(2)}$ are obtained through collocation such that they utilize the same nodes, i.e. $c = \hat{c}$. In the collocation procedure, the (arbitrary) choice is made that $\mathbf{w}_{tt}$ at time instant $c_1 = 0$ is *not* taken into account. This leads to schemes that are one order lower than the original quadrature rules HBPC$(6, k_{\max})$ and HBPC$(8, k_{\max})$, respectively. The Butcher tableaux corresponding to these fifth and seventh order, respectively, methods are given in Eq. (B.2) and Eq. (B.4).

Error Estimate The error estimate $\|\mathcal{E}_t^{n,[k],l}\|_2$ is then obtained by

$$\|\mathcal{E}_t^{n,[k],l}\|_2 := \left|\left|\mathbf{w}_h^{n,[k],l} - \tilde{\mathbf{w}}_h^{n,[k],l}\right|\right|_2, \quad \text{with} \quad \tilde{\mathbf{w}}_h^{n,[k],l} := \mathbf{w}_h^{n-1,[k+1],s} + \hat{\mathcal{I}}_l\left(\mathbf{w}_h^{n,[k^*],1}, \dots, \mathbf{w}_h^{n,[k^*],s}\right), \tag{22}$$

where we have defined k^* as a function of k to be the closest odd integer that is larger or equal than k. Due to the pipelining strategy of Algorithm 1, this means that the k^*-th iterate is always the correction with the highest index available on one processor. Note that due to the construction of the parallelization strategy, see also Fig. 2, the processor(s) handling the predictor and the first corrector step for stages $\mathbf{w}^{n,[k],l}$ with $l \neq s$ (in the example in the figure, this would be proc. #3) are somewhat special, as they do not have acceess to the final stage $\mathbf{w}_h^{n,[k],s}$ of their corresponding $k \in \{0, 1\}$. Hence, the error estimates $\mathcal{E}_t^{n,[k],l}$ with $l \neq s$ are only needed for these processor(s). All other processors utilize $\mathcal{E}_t^{n,[k],s}$ for their error estimates.

Alternatively, instead of evaluating the quadrature rule directly, one can perform an additional correction step with $\hat{\mathcal{I}}_l$ to obtain an approximate quantity $\tilde{\tilde{\mathbf{w}}}^{n,[k],l}$. That means, solve the following for $\tilde{\tilde{\mathbf{w}}}^{n,[k],l}$:

$$\begin{aligned}\tilde{\tilde{\mathbf{w}}}_h^{n,[k],l} &= \mathbf{w}_h^{n-1,[k+1],s} + \theta_1 \Delta t\left(\mathbf{R}^{(1)}(\tilde{\tilde{\mathbf{w}}}_h^{n,[k],l}) - \mathbf{R}^{(1)}(\mathbf{w}_h^{n,[k^*],l})\right) \\ &\quad - \theta_2 \frac{\Delta t^2}{2}\left(\mathbf{R}^{(2)}(\tilde{\tilde{\mathbf{w}}}_h^{n,[k],l}, \tilde{\tilde{\boldsymbol{\sigma}}}_h^{n,[k],l}) - \mathbf{R}^{(2)}(\mathbf{w}_h^{n,[k^*],l}, \mathbf{R}^{(1)}(\mathbf{w}_h^{n,[k^*],l}))\right) + \hat{\mathcal{I}}_l\left(\mathbf{w}_h^{n,[k^*],1}, \dots, \mathbf{w}_h^{n,[k^*],s}\right),\end{aligned} \tag{23}$$

and following, calculate the error estimate via

$$\|\mathcal{E}_t^{n,[k],l}\|_2 := \left|\left|\mathbf{w}_h^{n,[k],l} - \tilde{\tilde{\mathbf{w}}}_h^{n,[k],l}\right|\right|_2. \tag{24}$$

Evaluation of Temporal Error Estimate The accuracy of the embedded error estimate is evaluated by considering the Navier-Stokes equations (Eq. (1)) with initial conditions

$$\rho(\mathbf{x}, t = 0) = 1 + 0.3 \sin(\pi(x_1 + x_2)), \quad \mathbf{v} = (0.3, 0.3)^T, \quad \text{and} \quad p = 1, \tag{25}$$

on the domain $\Omega = [-1, 1]^2$, equipped with periodic boundary conditions. Viscosity is chosen to be $\mu = 10^{-3}$ and the reference Mach number is $\varepsilon \in \{1, 10^{-1}\}$. The domain is discretized with $\mathcal{N}_E = 64^2$ elements with $\mathcal{N}_p = 7$. The 'exact' solution is obtained via an explicit simulation with a fourth order low-storage Runge-Kutta method [26] with very small timestep ($\Delta t \approx 2.9 \cdot 10^{-5}$ and $\Delta t \approx 7.53 \cdot 10^{-6}$ for $\varepsilon = 1$ and $\varepsilon = 10^{-1}$, respectively).

We now perform a single timestep with different sizes for $\varepsilon = 1$ and $\varepsilon = 10^{-1}$ with the HBPC(6, 9) and the HBPC(8, 9) and report the exact and the estimated errors after the predictor and each correction step in Fig. 3. One can observe a clear trend: the higher the stiffness of the problem, i.e. larger Δt and/or smaller ε, the worse do the error estimators approximate the true error. For larger stiffnesses, the procedure according to Eq. (23) is more accurate than evaluating the embedded formula directly. For lower stiffnesses, the error estimates coincide very well with the true error until some minimum error is reached for some $[k]$. This is due to the fact that the embedded quadrature formula is only of order $\hat{q} = q - 1$ and hence has a lower accuracy than the original quadrature rule. (Technically, the error of the *lower-order* method is approximated.) Similar results are obtained when the accuracy of the embedded error estimator is tested on different meshes (not shown here), which shows the robustness of the error estimator. Summing up, the error estimate in Eq. (23) requiring an additional solving step is slightly more accurate than directly evaluating the embedded quadrature rule; it is recommended for stiff problems.

4. Adaptive strategy for HBPC schemes

A key feature for an efficient implicit time discretization method is an adaptation strategy for the iterative solution procedure, see e.g. [5,27,28], as it is of utmost importance to keep Newton and GMRES iterations to an absolute minimum, while obviously guaranteeing a certain quality of the solution. This is very different to explicit schemes, where this part of the solution process simply does not exist. In this section, we are aiming for an adaptive Newton convergence criterion that preserves the accuracy of the time stepping method without 'oversolving' it, i.e., we envision that the error of the Newton procedure, defined by

$$\mathcal{E}_{\mathrm{r}}^{[k],l} := \boldsymbol{X}^{[k]} - \boldsymbol{X}_{\mathrm{r}}^{[k]}, \tag{26}$$

is of the same order as the time discretization error. Hence,

$$\|\mathcal{E}_t^{n,[k],l}\| \approx \|\mathcal{E}_{\mathrm{r}}^{[k],l}\|,$$

where we have omitted the superscript n for the error of Newton's procedure for the ease of presentation. In this way, one does not deteriorate the temporal accuracy, while at the same time one is not overdoing Newton iterations. A rough, but seemingly reliable estimate of Newton's error is devised through an analysis of the equations in Section 4.1.

Remark 7. While it is possible to only use one Newton step per prediction/correction, and take into account more correction steps as similarly done in [29], we have found that this approach does not really work well in our context. In particular the

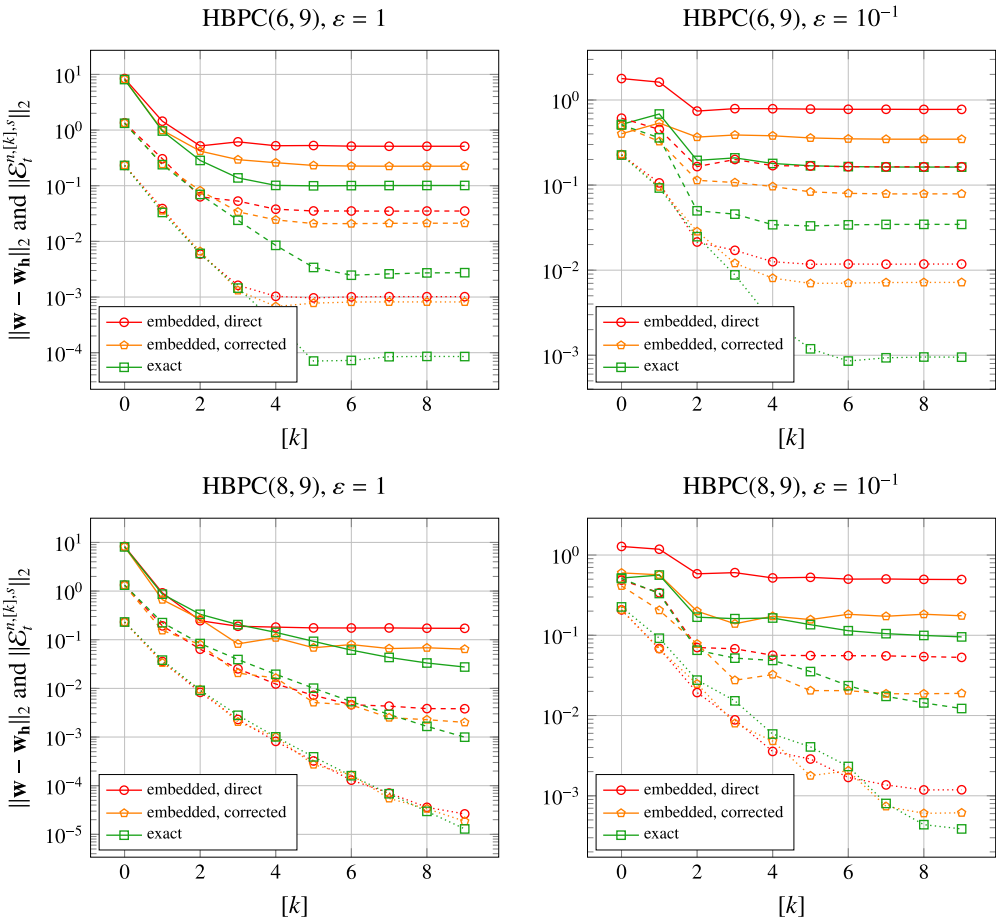


Fig. 3. Exact L_2-error and estimated errors via evaluating the embedded formula directly (Eq. (22)) and evaluating the embedded formula within one correction step (Eq. (23)) for HBPC(6, 9) (top) and HBPC(8, 9) scheme (bottom) after the first timestep. (Only one timestep each is performed, as the embedded formulae only measure local (in time) error contributions.) Left column shows results with $\varepsilon = 1$ and $\Delta t = 0.4$ (solid), $\Delta t = 0.2$ (dashed) and $\Delta t = 0.1$ (dotted). Right column shows results with $\varepsilon = 10^{-1}$ and $\Delta t = 0.1$ (solid), $\Delta t = 0.05$ (dashed) and $\Delta t = 0.025$ (dotted).

solution quality of the predictor and the corrector do have a significant influence on higher corrections. This can already be seen in the context of ODEs; and has in fact motivated the analysis to follow.

4.1. Adaptive Newton strategy

Convergence criteria for Newton's method have been addressed by several authors in the context of flow simulation with implicit timestepping methods relying on Newton-Krylov methods. Basically, two different approaches can be distinguished:

- An absolute tolerance for the Newton increment $\Delta \boldsymbol{X}_\mathrm{r}$ has been used in [30]. The inequality $\|\Delta \boldsymbol{X}_\mathrm{r}\|_2 \leq \mathrm{TOL}$, specified by a user-defined tolerance $\mathrm{TOL} \in \{10^{-5}, 10^{-7}\}$, is used as a criterion to terminate the Newton iterations.
- More used in practice seem to be convergence criteria based on the Newton residual N(X), which is the quantity to which the discrete solution fails to satisfy the equation. [31,32] and [33] start with a user defined accuracy TOL. An embedded Runge-Kutta method is then used to determine the corresponding timestep size and a modified tolerance TOL'. While [31] and [32] use $N(\boldsymbol{X}_\mathrm{r}) \leq N(\boldsymbol{X}_0) \cdot \mathrm{TOL}/5$ as convergence criterion ([32] also suggests the same treatment for the Newton increment), the authors in [33] use $N(\boldsymbol{X}_\mathrm{r}) \leq N(\boldsymbol{X}_0) \cdot \mathrm{TOL}'/10$. A slightly different approach is pursued in [5], where a fixed timestep is prescribed by the user and the convergence criterion $N(\boldsymbol{X}_\mathrm{r}) \leq N(\boldsymbol{X}_0) \cdot \min\left(10^{-3}, \|\mathcal{E}_t\|_2/3\right)$ is used, where $\|\mathcal{E}_t\|_2$ is computed through an embedded Runge-Kutta method. An *absolute tolerance for the Newton residual*

has been proposed in [28]. They use $N(\boldsymbol{X}_\mathrm{r}) \leq \|\mathcal{E}_t\|_2/10$, where the temporal error estimate is again based on an embedded Runge-Kutta method.

None of these approaches can directly be used for the HBPC schemes as different levels of accuracy for the different prediction/correction steps are not taken into account. Inspired by the approach outlined in [28], we derive a Newton convergence criterion that explicitly takes the different levels into account. We find that an absolute convergence criterion based on the Newton increment is a natural choice for this kind of methods.

4.1.1. Newton error estimate

In this section, we derive a heuristic that links the Newton error $\mathcal{E}_\mathrm{r}^{[k],l}$, see Eq. (26), to the Newton increment $\Delta\boldsymbol{X}_{\mathrm{r}+1}$ of the *following* Newton step and the Newton errors $\mathcal{E}_{\mathrm{r}'}^{[k-1],i}$ of *previous* correction steps. This allows, in a subsequent step, to derive a practically usable criterion on when to terminate Newton's algorithm. Terminating Newton's method (see Eq. (19) and Eq. (20)) at finite r, the introduced error $\mathcal{E}_\mathrm{r}^{[k],l}$ is given by

$$\mathcal{E}_\mathrm{r}^{[k],l} = \boldsymbol{X}^{[k]} - \boldsymbol{X}_\mathrm{r}^{[k]} = \boldsymbol{W}_\mathrm{old} + \bar{\Phi}\left(\boldsymbol{X}^{[k]}, \boldsymbol{X}^{[k-1],1:s}\right) - \boldsymbol{X}_\mathrm{r}^{[k]} + \bar{\Phi}\left(\boldsymbol{X}_r^{[k]}, \boldsymbol{X}_{\mathrm{r}'}^{[k-1],1:s}\right) - \bar{\Phi}\left(\boldsymbol{X}_r^{[k]}, \boldsymbol{X}_{\mathrm{r}'}^{[k-1],1:s}\right).$$

Performing a Taylor expansion one obtains

$$\begin{aligned}\boldsymbol{X}^{[k]} - \boldsymbol{X}_\mathrm{r}^{[k]} =& \frac{\partial\bar{\Phi}\left(\boldsymbol{X}_\mathrm{r}^{[k]}, \boldsymbol{X}_{\mathrm{r}'}^{[k-1],1:s}\right)}{\partial\boldsymbol{X}^{[k]}}\left(\boldsymbol{X}^{[k]} - \boldsymbol{X}_\mathrm{r}^{[k]}\right) + \sum_{i=1}^{s}\frac{\partial\bar{\Phi}\left(\boldsymbol{X}_\mathrm{r}^{[k]}, \boldsymbol{X}_{\mathrm{r}'}^{[k-1],1:s}\right)}{\partial\boldsymbol{X}^{[k-1],i}}\left(\boldsymbol{X}^{[k-1],i} - \boldsymbol{X}_{\mathrm{r}'}^{[k-1],i}\right)\\ &-\boldsymbol{X}_\mathrm{r}^{[k]} + \mathbf{W}_\mathrm{old} + \bar{\Phi}\left(\boldsymbol{X}_r^{[k]}, \boldsymbol{X}_{\mathrm{r}'}^{[k-1],1:s}\right) + \mathcal{O}\left(\left(\boldsymbol{X}^{[k]} - \boldsymbol{X}_\mathrm{r}^{[k]}\right)^2\right) + \mathcal{O}\left(\left(\boldsymbol{X}^{[k-1],1:s} - \boldsymbol{X}_{\mathrm{r}'}^{[k-1],1:s}\right)^2\right).\end{aligned}$$

Next, we truncate the higher order terms[2] and find

$$\mathcal{E}_\mathrm{r}^{[k],l} \doteq \left(\mathrm{Id} - \frac{\partial\bar{\Phi}\left(\boldsymbol{X}_\mathrm{r}^{[k]}, \boldsymbol{X}_{\mathrm{r}'}^{[k-1],1:s}\right)}{\partial\boldsymbol{X}^{[k]}}\right)^{-1}\cdot\left(\boldsymbol{W}_\mathrm{old} + \bar{\Phi}\left(\boldsymbol{X}_r^{[k]}, \boldsymbol{X}_{\mathrm{r}'}^{[k-1],1:s}\right) - \boldsymbol{X}_\mathrm{r}^{[k]} + \sum_{i=1}^{s}\frac{\partial\bar{\Phi}\left(\boldsymbol{X}_\mathrm{r}^{[k]}, \boldsymbol{X}_{\mathrm{r}'}^{[k-1],1:s}\right)}{\partial\boldsymbol{X}^{[k-1],i}}\left(\boldsymbol{X}^{[k-1],i} - \boldsymbol{X}_{\mathrm{r}'}^{[k-1],i}\right)\right).$$

We then can make use of the definition of Newton's method, see Eq. (19) to simplify the first part of the expression

$$\mathcal{E}_r^{[k],l} \doteq \underbrace{\Delta\boldsymbol{X}_{r+1}}_{\substack{\text{current Newton}\\ \text{procedure}}} + \underbrace{\left(\mathrm{Id} - \frac{\partial\overline{\Phi}\left(\boldsymbol{X}_r^{[k]}, \boldsymbol{X}_{r'}^{[k-1],1:s}\right)}{\partial\boldsymbol{X}^{[k]}}\right)^{-1}\cdot\left(\sum_{i=1}^{s}\frac{\partial\overline{\Phi}\left(\boldsymbol{X}_r^{[k]}, \boldsymbol{X}_{r'}^{[k-1],1:s}\right)}{\partial\boldsymbol{X}^{[k-1],i}}\mathcal{E}_{r'}^{[k-1],i}\right)}_{\text{accumulation of previous Newton errors}}.$$

The error hence consists of one part, where the Newton errors of previous prediction/correction steps are accumulated and another part influenced by the current Newton procedure, which equals the Newton increment of the next Newton iterate $\Delta\boldsymbol{X}_{\mathrm{r}+1}$. For the predictor, we then directly find

$$\mathcal{E}_\mathrm{r}^{[0],l} \doteq \Delta\boldsymbol{X}_{\mathrm{r}+1},$$

as there are no previous prediction/correction steps that can influence the error. For the corrector one finds

$$\begin{aligned}\mathcal{E}_\mathrm{r}^{[k],l} \doteq \Delta\boldsymbol{X}_{\mathrm{r}+1} - &\begin{pmatrix}\mathrm{Id} - \theta_1\Delta t\frac{\partial\mathbf{R}_h^{(1)}}{\partial\mathbf{w}_h^{n,[k],l}} + \frac{\theta_2\Delta t^2}{2}\frac{\partial\mathbf{R}_h^{(2)}}{\partial\mathbf{w}_h^{n,[k],l}} & \frac{\theta_2\Delta t^2}{2}\frac{\partial\mathbf{R}_h^{(2)}}{\partial\boldsymbol{\sigma}_h^{n,[k],l}}\\ -\frac{\partial\mathbf{R}_h^{(1)}}{\partial\mathbf{w}_h^{n,[k],l}} & \mathrm{Id}\end{pmatrix}^{-1}\\ &\cdot\left(\begin{pmatrix}\theta_1\Delta t\frac{\partial\mathbf{R}_h^{(1)}}{\partial\mathbf{w}_h^{n,[k-1],l}} - \frac{\theta_2\Delta t^2}{2}\left(\frac{\partial\mathbf{R}_h^{(2)}}{\partial\mathbf{w}_h^{n,[k-1],l}} + \frac{\partial\mathbf{R}_h^{(2)}}{\partial\boldsymbol{\sigma}_h^{n,[k-1],l}}\frac{\partial\mathbf{R}_h^{(1)}}{\partial\mathbf{w}_h^{n,[k-1],l}}\right) & 0\\ 0 & 0\end{pmatrix}\mathcal{E}_{\mathrm{r}'}^{[k-1],l}\right.\\ &\left. - \sum_{i=1}^{s}\begin{pmatrix}\Delta tB_{li}^{(1)}\frac{\partial\mathbf{R}_h^{(1)}}{\partial\mathbf{w}_h^{n,[k-1],i}} + \Delta t^2B_{li}^{(2)}\left(\frac{\partial\mathbf{R}_h^{(2)}}{\partial\mathbf{w}_h^{n,[k-1],i}} + \frac{\partial\mathbf{R}_h^{(2)}}{\partial\boldsymbol{\sigma}_h^{n,[k-1],i}}\frac{\partial\mathbf{R}_h^{(1)}}{\partial\mathbf{w}_h^{n,[k-1],i}}\right) & 0\\ 0 & 0\end{pmatrix}\mathcal{E}_{\mathrm{r}'}^{[k-1],i}\right),\end{aligned}$$

which can be simplified to (please note that due to construction, there holds $\frac{\partial\mathbf{R}_h^{(2)}}{\partial\boldsymbol{\sigma}_h} = \frac{\partial\mathbf{R}_h^{(1)}}{\partial\mathbf{w}_h}$[3])

[2] Please note that this is an assumption that we make. It is not clear – in particular for large Δt or stiff equations – that these terms are small. However, to obtain guidelines for the termination of Newton's algorithm, we will from now on neglect the higher order terms.

[3] That this is true can be seen the easiest from the continuous level: There holds $\mathbf{w}_t = \mathbf{R}^{(1)}(\mathbf{w})$ due to Eq. (3). Differentiating with respect to time yields $\mathbf{w}_{tt} = \frac{\partial\mathbf{R}^{(1)}(\mathbf{w})}{\partial\mathbf{w}}\mathbf{w}_t = \frac{\partial\mathbf{R}^{(1)}(\mathbf{w})}{\partial\mathbf{w}}\boldsymbol{\sigma} =: \mathbf{R}^{(2)}(\mathbf{w},\boldsymbol{\sigma})$. From this definition, the identity follows in a straightforward way. The same is true for the DG discretization, yet, it is more cumbersome (but not more difficult) to show this, departing from the weak formulations in (13) and (15).

$$\mathcal{E}_{\mathrm{r}}^{[k],l} \doteq \Delta \boldsymbol{X}_{\mathrm{r}+1} - \begin{pmatrix} S^{-1}\left(\theta_1 \Delta t \frac{\partial \mathbf{R}_h^{(1)}}{\partial \mathbf{w}_h^{n,[k-1],l}} - \frac{\theta_2 \Delta t^2}{2}\left(\frac{\partial \mathbf{R}_h^{(2)}}{\partial \mathbf{w}_h^{n,[k-1],l}} + \left(\frac{\partial \mathbf{R}_h^{(1)}}{\partial \mathbf{w}_h^{n,[k-1],l}}\right)^2\right)\right) & 0 \\ \frac{\partial \mathbf{R}_h^{(1)}}{\partial \mathbf{w}_h^{n,[k],l}} S^{-1}\left(\theta_1 \Delta t \frac{\partial \mathbf{R}_h^{(1)}}{\partial \mathbf{w}_h^{n,[k-1],l}} - \frac{\theta_2 \Delta t^2}{2}\left(\frac{\partial \mathbf{R}_h^{(2)}}{\partial \mathbf{w}_h^{n,[k-1],l}} + \left(\frac{\partial \mathbf{R}_h^{(1)}}{\partial \mathbf{w}_h^{n,[k-1],l}}\right)^2\right)\right) & 0 \end{pmatrix} \mathcal{E}_{\mathrm{r}'}^{[k-1],l}$$
$$+ \sum_{i=1}^{s} \begin{pmatrix} S^{-1}\left(\Delta t B_{li}^{(1)} \frac{\partial \mathbf{R}_h^{(1)}}{\partial \mathbf{w}_h^{n,[k-1],i}} + \Delta t^2 B_{li}^{(2)}\left(\frac{\partial \mathbf{R}_h^{(2)}}{\partial \mathbf{w}_h^{n,[k-1],i}} + \left(\frac{\partial \mathbf{R}_h^{(1)}}{\partial \mathbf{w}_h^{n,[k-1],i}}\right)^2\right)\right) & 0 \\ \frac{\partial \mathbf{R}_h^{(1)}}{\partial \mathbf{w}_h^{n,[k],l}} S^{-1}\left(\Delta t B_{li}^{(1)} \frac{\partial \mathbf{R}_h^{(1)}}{\partial \mathbf{w}_h^{n,[k-1],i}} + \Delta t^2 B_{li}^{(2)}\left(\frac{\partial \mathbf{R}_h^{(2)}}{\partial \mathbf{w}_h^{n,[k-1],i}} + \left(\frac{\partial \mathbf{R}_h^{(1)}}{\partial \mathbf{w}_h^{n,[k-1],i}}\right)^2\right)\right) & 0 \end{pmatrix} \mathcal{E}_{\mathrm{r}'}^{[k-1],i}, \tag{27}$$

with the Schur complement corresponding to the lower right block given by

$$S := \left(\mathrm{Id} - \theta_1 \Delta t \frac{\partial \mathbf{R}_h^{(1)}}{\partial \mathbf{w}_h^{n,[k],l}} + \frac{\theta_2 \Delta t^2}{2}\left(\frac{\partial \mathbf{R}_h^{(2)}}{\partial \mathbf{w}_h^{n,[k],l}} + \left(\frac{\partial \mathbf{R}_h^{(1)}}{\partial \mathbf{w}_h^{n,[k],l}}\right)^2\right)\right).$$

We now consider the limits of Eq. (27) and start with $\Delta t \to 0$, i.e.

$$\mathcal{E}_{\mathrm{r}}^{[k],l} \doteq \Delta \boldsymbol{X}_{\mathrm{r}+1} - \begin{pmatrix} \left(\mathrm{Id} + \mathcal{O}(\Delta t) + \mathcal{O}(\Delta t^2)\right)^{-1}\left(\mathcal{O}(\Delta t) + \mathcal{O}(\Delta t^2)\right) & 0 \\ \left(\mathrm{Id} + \mathcal{O}(\Delta t) + \mathcal{O}(\Delta t^2)\right)^{-1}\left(\mathcal{O}(\Delta t) + \mathcal{O}(\Delta t^2)\right) & 0 \end{pmatrix} \mathcal{E}_{\mathrm{r}'}^{[k-1],l}$$
$$+ \sum_{i=1}^{s} \begin{pmatrix} \left(\mathrm{Id} + \mathcal{O}(\Delta t) + \mathcal{O}(\Delta t^2)\right)^{-1}\left(\mathcal{O}(\Delta t) + \mathcal{O}(\Delta t^2)\right) & 0 \\ \left(\mathrm{Id} + \mathcal{O}(\Delta t) + \mathcal{O}(\Delta t^2)\right)^{-1}\left(\mathcal{O}(\Delta t) + \mathcal{O}(\Delta t^2)\right) & 0 \end{pmatrix} \mathcal{E}_{\mathrm{r}'}^{[k-1],i} \to \Delta \boldsymbol{X}_{\mathrm{r}+1}, \quad \Delta t \to 0.$$

We find that for vanishing Δt, the Newton errors introduced by previous stages and correction steps do not play a role and the error is directly given by the next Newton increment. The limit $\Delta t \to \infty$ is more difficult to obtain. We start by considering the $\mathcal{O}(\Delta t^2)$ terms

$$\lim_{\Delta t \to \infty} \mathcal{E}_{\mathrm{r}}^{[k],l} \doteq \Delta \boldsymbol{X}_{\mathrm{r}+1} - \begin{pmatrix} \left(\frac{\partial \mathbf{R}_h^{(2)}}{\partial \mathbf{w}_h^{n,[k],l}} + \left(\frac{\partial \mathbf{R}_h^{(1)}}{\partial \mathbf{w}_h^{n,[k],l}}\right)^2\right)^{-1}\left(\frac{\partial \mathbf{R}_h^{(2)}}{\partial \mathbf{w}_h^{n,[k-1],l}} + \left(\frac{\partial \mathbf{R}_h^{(1)}}{\partial \mathbf{w}_h^{n,[k-1],l}}\right)^2\right) & 0 \\ \frac{\partial \mathbf{R}_h^{(1)}}{\partial \mathbf{w}_h^{n,[k],l}}\left(\frac{\partial \mathbf{R}_h^{(2)}}{\partial \mathbf{w}_h^{n,[k],l}} + \left(\frac{\partial \mathbf{R}_h^{(1)}}{\partial \mathbf{w}_h^{n,[k],l}}\right)^2\right)^{-1}\left(\frac{\partial \mathbf{R}_h^{(2)}}{\partial \mathbf{w}_h^{n,[k-1],l}} + \left(\frac{\partial \mathbf{R}_h^{(1)}}{\partial \mathbf{w}_h^{n,[k-1],l}}\right)^2\right) & 0 \end{pmatrix} \mathcal{E}_{\mathrm{r}'}^{[k-1],l}$$
$$+ \sum_{i=1}^{s} \begin{pmatrix} \frac{2B_{li}^{(2)}}{\theta_2}\left(\frac{\partial \mathbf{R}_h^{(2)}}{\partial \mathbf{w}_h^{n,[k],l}} + \left(\frac{\partial \mathbf{R}_h^{(1)}}{\partial \mathbf{w}_h^{n,[k],l}}\right)^2\right)^{-1}\left(\frac{\partial \mathbf{R}_h^{(2)}}{\partial \mathbf{w}_h^{n,[k-1],l}} + \left(\frac{\partial \mathbf{R}_h^{(1)}}{\partial \mathbf{w}_h^{n,[k-1],i}}\right)^2\right) & 0 \\ \frac{2B_{li}^{(2)}}{\theta_2}\frac{\partial \mathbf{R}_h^{(1)}}{\partial \mathbf{w}_h^{n,[k],l}}\left(\frac{\partial \mathbf{R}_h^{(2)}}{\partial \mathbf{w}_h^{n,[k],l}} + \left(\frac{\partial \mathbf{R}_h^{(1)}}{\partial \mathbf{w}_h^{n,[k],l}}\right)^2\right)^{-1}\left(\frac{\partial \mathbf{R}_h^{(2)}}{\partial \mathbf{w}_h^{n,[k-1],l}} + \left(\frac{\partial \mathbf{R}_h^{(1)}}{\partial \mathbf{w}_h^{n,[k-1],i}}\right)^2\right) & 0 \end{pmatrix} \mathcal{E}_{\mathrm{r}'}^{[k-1],i}. \tag{28}$$

Remark 8. The above Eq. (28) is highly nonlinear, yet, it has an interesting structure. For *linear* equations, where the quantities $\frac{\partial \mathbf{R}_h^{(2)}}{\partial \mathbf{w}_h^{n,[k],l}}$ and $\frac{\partial \mathbf{R}_h^{(1)}}{\partial \mathbf{w}_h^{n,[k],l}}$ are constant, there holds $\left(\frac{\partial \mathbf{R}_h^{(2)}}{\partial \mathbf{w}_h^{n,[k],l}} + \left(\frac{\partial \mathbf{R}_h^{(1)}}{\partial \mathbf{w}_h^{n,[k],l}}\right)^2\right)^{-1}\left(\frac{\partial \mathbf{R}_h^{(2)}}{\partial \mathbf{w}_h^{n,[k-1],l}} + \left(\frac{\partial \mathbf{R}_h^{(1)}}{\partial \mathbf{w}_h^{n,[k-1],l}}\right)^2\right) = \mathrm{Id}$, and the equations reduce to

$$\lim_{\Delta t \to \infty} \mathcal{E}_{\mathrm{r}}^{[k],l} \doteq \Delta \boldsymbol{X}_{\mathrm{r}+1} - \begin{pmatrix} \mathrm{Id} & 0 \\ \frac{\partial \mathbf{R}_h^{(1)}}{\partial \mathbf{w}_h^{n,[k],l}} & 0 \end{pmatrix} \mathcal{E}_{\mathrm{r}'}^{[k-1],l} + \sum_{i=1}^{s} \begin{pmatrix} \frac{2B_{li}^{(2)}}{\theta_2}\mathrm{Id} & 0 \\ \frac{2B_{li}^{(2)}}{\theta_2}\frac{\partial \mathbf{R}_h^{(1)}}{\partial \mathbf{w}_h^{n,[k],i}} & 0 \end{pmatrix} \mathcal{E}_{\mathrm{r}'}^{[k-1],i}.$$

This automatically leads to the estimate

$$\|\mathcal{E}_{\mathrm{r,w}}^{[k],l}\|_2 \lesssim \|\Delta \boldsymbol{X}_{\mathrm{r}+1,\mathrm{w}}\|_2 + \|\mathcal{E}_{\mathrm{r}',\mathrm{w}}^{[k-1],l}\|_2 + \sum_{i=2}^{s} \frac{2|B_{li}^{(2)}|}{\theta_2} \|\mathcal{E}_{\mathrm{r}',\mathrm{w}}^{[k-1],i}\|_2, \quad \text{for } \Delta t \to \infty, \tag{29}$$

where by $\mathcal{E}_{\mathrm{r,w}}^{[k],l}$, we denote the component of $\mathcal{E}_{\mathrm{r}}^{[k],l}$ corresponding to the degrees of freedom of $\mathbf{w}_h^{n,[k],l}$. $\mathcal{E}_{\mathrm{r}'}^{[k-1],1} = 0$ due to the fact that the first stage is trivial to compute and does not need a Newton iteration. Please note that a similar computation is, to our knowledge, not possible for arbitrary nonlinear equations, in particular not for the compressible Navier-Stokes equations. We will, however, use (29) as a heuristic basis for our error estimation procedure.

Inspired by the behavior of the linear algorithm, we consider the following error estimate for small and large Δt:

$$\begin{aligned} &\|\mathcal{E}_{\mathrm{r,w}}^{[k],l}\|_2 \approx \|\Delta \boldsymbol{X}_{\mathrm{r}+1,\mathrm{w}}\|_2, && \text{for } \Delta t \to 0, \\ &\|\mathcal{E}_{\mathrm{r,w}}^{[k],l}\|_2 \approx \|\Delta \boldsymbol{X}_{\mathrm{r}+1,\mathrm{w}}\|_2 + C_l \|\mathcal{E}_{\mathrm{r}',\mathrm{w}}^{[k-1],l}\|_2 + \textstyle\sum_{i=2}^{s} C_i \frac{2|B_{li}^{(2)}|}{\theta_2} \|\mathcal{E}_{\mathrm{r}',\mathrm{w}}^{[k-1],i}\|_2, && \text{for } \Delta t \to \infty. \end{aligned} \tag{30}$$

Here, C_i are user-defined constants, which, later, will reduce into one global constant. Please note that we also use this form in case of the modified arguments for the correction steps $k > 1$ (see Eq. (9)).

Remark 9. Choosing the constants C_i basically means that we *assume* that terms of form

$$\left\| \left(\frac{\partial \mathbf{R}_h^{(2)}}{\partial \mathbf{w}_h^{n,[k],l}} + \left(\frac{\partial \mathbf{R}_h^{(1)}}{\partial \mathbf{w}_h^{n,[k],l}} \right)^2 \right)^{-1} \left(\frac{\partial \mathbf{R}_h^{(2)}}{\partial \mathbf{w}_h^{n,[k-1],l}} + \left(\frac{\partial \mathbf{R}_h^{(1)}}{\partial \mathbf{w}_h^{n,[k-1],l}} \right)^2 \right) \right\|$$

are bounded.

4.1.2. Newton convergence criterion

The findings in Eq. (30) still depend on the various stages, which make the error estimate even more tedious to evaluate than it already is. In the sequel, we do therefore assume that the stage-error is constant within a timestep, and define

$$\|\mathcal{E}_{\mathrm{r,w}}^{[k]}\|_2 := \|\mathcal{E}_{\mathrm{r,w}}^{[k],s}\|_2, \qquad \text{and} \qquad \|\mathcal{E}_t^{n,[k]}\|_2 := \|\mathcal{E}_t^{n,[k],s}\|_2, \tag{31}$$

so only the last stage is taken into account. The desired goal of the adaptive Newton strategy is that the errors introduced by not solving the non-linear Eq. (16) exactly are smaller than the errors introduced by the timestepping procedure itself. In formulae, this means

$$\|\mathcal{E}_{\mathrm{r,w}}^{[k]}\|_2 \leq \eta \|\mathcal{E}_t^{n,[k]}\|_2, \quad \text{for} \quad k = 0, \dots, k_{\max}, \tag{32}$$

where we have introduced some safety factor $0 < \eta < 1$. η is testcase-dependent, and will be explicitly stated for each numerical result.

Again, this is in good agreement with our numerical experience. Subsequently, based on the findings in Eq. (30) and definitions in Eq. (31), we assume that the Newton error $\|\mathcal{E}_{\mathrm{r,w}}^{[k]}\|_2$ can be written as

$$\|\mathcal{E}_{\mathrm{r,w}}^{[k]}\|_2 = \|\Delta \boldsymbol{X}_{\mathrm{r}+1,\mathrm{w}}^{[k]}\|_2 + C \cdot \|\mathcal{E}_{\mathrm{r}',\mathrm{w}}^{[k-1]}\|_2, \tag{33}$$

for a constant C. If we define $\|\mathcal{E}_{\mathrm{r}',\mathrm{w}}^{[-1]}\|_2 = 0$, this is valid for all $k \geq 0$. Please note that r' is a generic constant, as the amount of Newton steps in correction k can be different from the ones in correction $k-1$.

Remark 10. For $\Delta t \to 0$, there holds $C = 0$, see (30). For all the numerical experiments we made, C was always in the order of one, never exceeding five. In the algorithm itself, it is treated as a user-supplied constant.

We can recursively unfold formula (33) from $k = k_{\max}$ to obtain

$$\|\mathcal{E}_{\mathrm{r,w}}^{[k_{\max}]}\|_2 = \sum_{k=0}^{k_{\max}} C^{k_{\max}-k} \|\Delta \boldsymbol{X}_{\mathrm{r}'+1,\mathrm{w}}^{[k]}\|_2.$$

Please note again that r' is a generic amount of steps and can change from one correction to the other. Under these preliminaries, the inequality to be fulfilled for $k_{\max}$ is given by

$$\|\mathcal{E}_{\mathrm{r,w}}^{[k_{\max}]}\|_2 = \sum_{k=0}^{k_{\max}} C^{k_{\max}-k} \|\Delta \boldsymbol{X}_{\mathrm{r}'+1,\mathrm{w}}^{[k]}\|_2 \overset{!}{\leq} \eta \|\mathcal{E}_t^{n,[k_{\max}]}\|_2. \tag{34}$$

The scaling with $C^{k_{\max}-k}$ in (34) motivates the following heuristic choice for the Newton increment:

$$\begin{aligned} \|\Delta \boldsymbol{X}_{\mathrm{r}+1,\mathrm{w}}^{[0]}\|_2 &\leq \eta \min\left(C_1^{k_{\max}} \|\mathcal{E}_t^{n,[k_{\max}]}\|_2, \|\mathcal{E}_t^{n,[0]}\|_2 \right), \\ \|\Delta \boldsymbol{X}_{\mathrm{r}+1,\mathrm{w}}^{[k]}\|_2 &\leq \eta \min\left(C_2 C_1^{k_{\max}-k} \|\mathcal{E}_t^{n,[k_{\max}]}\|_2, \|\mathcal{E}_t^{n,[k]}\|_2 \right), \quad \text{for} \quad 1 \leq k \leq k_{\max}. \end{aligned} \tag{35}$$

Here, $C_1 \sim C^{-1}$ and C_2 are user-defined input parameters to the code. We have included C_2 into this heuristic as we have found that the quality of the predictor typically has a larger influence on the quality than the correction steps; typically, C_2 is taken to be smaller than one. We have found numerically that the $k_{\max}$-dependent choices $C_1 := \frac{k_{\max}-1}{C k_{\max}}$ and $C_2 := 1 - \frac{k_{\max}-1}{k_{\max}}$ seem to work very well; we will stick to this definition in the sequel.

Remark 11. Eq. (35) shows that for the evaluation of the convergence criterion of Newton's method for the current iterate $[k]$, one requires $\|\mathcal{E}_t^{n,[k]}\|_2$ and $\|\mathcal{E}_t^{n,[k_{\max}]}\|_2$. While the former can be evaluated independently on each processor, the latter requires some special treatment. The information about the solution $\mathbf{w}_h^{n,[k_{\max}],s}$ is only available on the processor with rank zero (#0), see Fig. 2. Hence, the temporal error estimate $\|\mathcal{E}_t^{n,[k_{\max}]}\|_2$ can only be evaluated on this processor. The error information is then communicated to the other processors along the standard information propagation path, i.e. along the diagonal. This leads to a delay of the adaptive procedure's start on the different processors, meaning that, instead of

$\|\mathcal{E}_t^{n,[k_{\max}]}\|_2$, the quantity $\|\mathcal{E}_t^{n^*,[k_{\max}]}\|_2$ with $n^* < n$ is evaluated. This delay is indicated with the gray shaded area in Fig. 2. As it is crucial to keep the parallel-in-time structure of the scheme, we need to hence make the important assumption that the errors $\|\mathcal{E}_t^{n,[k_{\max}]}\|_2$ are only changing mildly with n, so that $\|\mathcal{E}_t^{n^*,[k_{\max}]}\|_2$ is indeed a good approximation for $\|\mathcal{E}_t^{n,[k_{\max}]}\|_2$. Note that $n - n^*$ cannot exceed $k_{\max}$ due to construction.

Remark 12. A similar approach as the one outlined above in Section 4.1.1 and Section 4.1.2 can be done for standard diagonally implicit Runge Kutta methods. In Appendix C, we briefly introduce a similar adaptive Newton strategy for ESDIRK methods, which will then later be used for efficiency comparisons in Section 5.4.

4.1.3. Newton error extrapolation

Considering Eq. (34) and Eq. (35), one can see that we have found a condition for $\|\mathcal{E}_{\mathrm{r,w}}^{[k]}\|_2$ via the Newton increment $\|\Delta \boldsymbol{X}_{\mathrm{r+1,w}}^{[k]}\|_2$. That means that in order to obtain a condition for the error at iterate r one has to calculate or estimate the norm of the $(\mathrm{r}+1)$-th Newton increment. As the computation of $\mathcal{E}_{\mathrm{r,w}}^{[k]}$ is time-consuming, we propose to use an extrapolation procedure to obtain an estimate for $\|\Delta \boldsymbol{X}_{\mathrm{r+1}}\|_2$, which is then used within the estimate for $\|\mathcal{E}_{\mathrm{r,w}}\|_2$, see Eq. (34). We either use a linear extrapolation procedure,

$$\begin{aligned} \|\Delta \boldsymbol{X}_{3,w}^{[k]} \quad\|_2 &\approx \frac{\|\Delta \boldsymbol{X}_{2,w}^{[k]}\|_2^2}{\|\Delta \boldsymbol{X}_{1,w}^{[k]}\|_2}, \\ \|\Delta \boldsymbol{X}_{\mathrm{r+1,w}}^{[k]}\|_2 &\approx \frac{\|\Delta \boldsymbol{X}_{\mathrm{r-2,w}}^{[k]}\|_2 \|\Delta \boldsymbol{X}_{\mathrm{r-1,w}}^{[k]}\|_2 + \|\Delta \boldsymbol{X}_{\mathrm{r-1,w}}^{[k]}\|_2 \|\Delta \boldsymbol{X}_{\mathrm{r,w}}^{[k]}\|_2}{\|\Delta \boldsymbol{X}_{\mathrm{r-2,w}}^{[k]}\|_2^2 + \|\Delta \boldsymbol{X}_{\mathrm{r-1,w}}^{[k]}\|_2^2} \|\Delta \boldsymbol{X}_{\mathrm{r,w}}^{[k]}\|_2, \quad \text{for } \mathrm{r} \geq 3, \end{aligned} \tag{36}$$

or a quadratic extrapolation procedure

$$\begin{aligned} \|\Delta \boldsymbol{X}_{3,w}^{[k]} \quad\|_2 &\approx \frac{\|\Delta \boldsymbol{X}_{2,w}^{[k]}\|_2^3}{\|\Delta \boldsymbol{X}_{1,w}^{[k]}\|_2^2}, \\ \|\Delta \boldsymbol{X}_{\mathrm{r+1,w}}^{[k]}\|_2 &\approx \frac{\|\Delta \boldsymbol{X}_{\mathrm{r-2,w}}^{[k]}\|_2^2 \|\Delta \boldsymbol{X}_{\mathrm{r-1,w}}^{[k]}\|_2 + \|\Delta \boldsymbol{X}_{\mathrm{r-1,w}}^{[k]}\|_2^2 \|\Delta \boldsymbol{X}_{\mathrm{r,w}}^{[k]}\|_2}{\|\Delta \boldsymbol{X}_{\mathrm{r-2,w}}^{[k]}\|_2^4 + \|\Delta \boldsymbol{X}_{\mathrm{r-1,w}}^{[k]}\|_2^4} \|\Delta \boldsymbol{X}_{\mathrm{r,w}}^{[k]}\|_2^2, \quad \text{for } \mathrm{r} \geq 3, \end{aligned} \tag{37}$$

that considers at maximum the previous three calculated Newton increments. Note that for $\mathrm{r} \geq 3$, a least-squares approximation of the constants is used in both cases. The linear extrapolation procedure has to be applied if a fixed coarse relative tolerance for the GMRES solver is applied [32]. If the relative tolerances converge to zero fast enough, quadratic convergence of Newton's method can be expected. One opportunity to achieve this is to use the *Eisenstat-Walker* procedure introduced in [34].

4.1.4. Application of adaptive Newton procedure

In this section, all ingredients of the adaptive Newton strategy are combined and validated. The temporal error is estimated according to Eq. (22) and is evaluated only once after the first timestep. This error estimate is then used to determine Newton's convergence condition via Eq. (35), where the actual Newton increment is obtained via the extrapolation procedure, introduced in Section 4.1.3. The constants C and η are chosen to be 0.5 and 0.1, respectively. We either use a fixed relative GMRES tolerance of $\varepsilon_{\mathrm{GMRES}} = 5 \cdot 10^{-2}$, which corresponds to the same criterion as used in [28], with the linear extrapolation procedure of Newton's increment given in Eq. (36). Alternatively, we apply the adaptive Eisenstat-Walker procedure [34] to determine the relative GMRES tolerances and utilize the quadratic extrapolation of Newton's increment given by Eq. (37).

We choose the same setup used in Section 3.3 for the Navier-Stokes equations with the initial conditions given by Eq. (25) with $\varepsilon = 1$, $\mathcal{N}_p = 7$ and $\mathcal{N}_E = 32^2$. The final time is set to $T_{end} = 1.0$. In order to evaluate the adaptive Newton procedure, simulations with fixed relative tolerances are used. Additionally, an absolute convergence criterion

$$\|\Delta \boldsymbol{X}_{\mathrm{r+1,w}}^{[k]}\|_2 \leq 10^{-14} \frac{\sqrt{\mathrm{nDOF}}}{\min(1, \varepsilon)}, \tag{38}$$

is used for all cases including the simulations with adaptive Newton strategy, where nDOF describes the total number of spatial degrees of freedom and ε is the stiffness parameter of the considered physical equations[4]. As initial guess for Newton's method, i.e. $\boldsymbol{X}_0^{[k]}$, we choose the solution of the second order explicit Taylor step for the predictor and the corresponding stage value of the previous iterate $[k-1]$ for the correction steps. We start counting Newton's iterations after $k_{\max}$ timesteps (compare Fig. 2) to ensure that the full capabilities of the adaptive strategy are evaluated. Solution steps that cannot use the adaptive Newton strategy (gray shaded area in Fig. 2) utilize a relative convergence criterion of $\varepsilon_{\mathrm{Newton,rel}} = 10^{-10}$.

The resulting errors and average Newton iterations per stage of this series of simulations are visualized in Fig. 4. Missing points for the fixed tolerance $\varepsilon_{\mathrm{Newton}} = 10^{-2}$ indicate a diverging solution. Fig. 4, leftmost column, displays the numerical errors made for the adaptive choice of the Newton tolerance and several fixed Newton tolerances. It is only for HBPC(6, 5) with

[4] $\sqrt{\mathrm{nDOF}}$ is the Euclidean norm of the vector of size nDOF with each element being one. This quantity serves hence as a reference value – the larger nDOF, the lesser one can expect that fine target accuracies can be reached. The ε in the denominator is a safety factor to account for the stiffness of the problem.

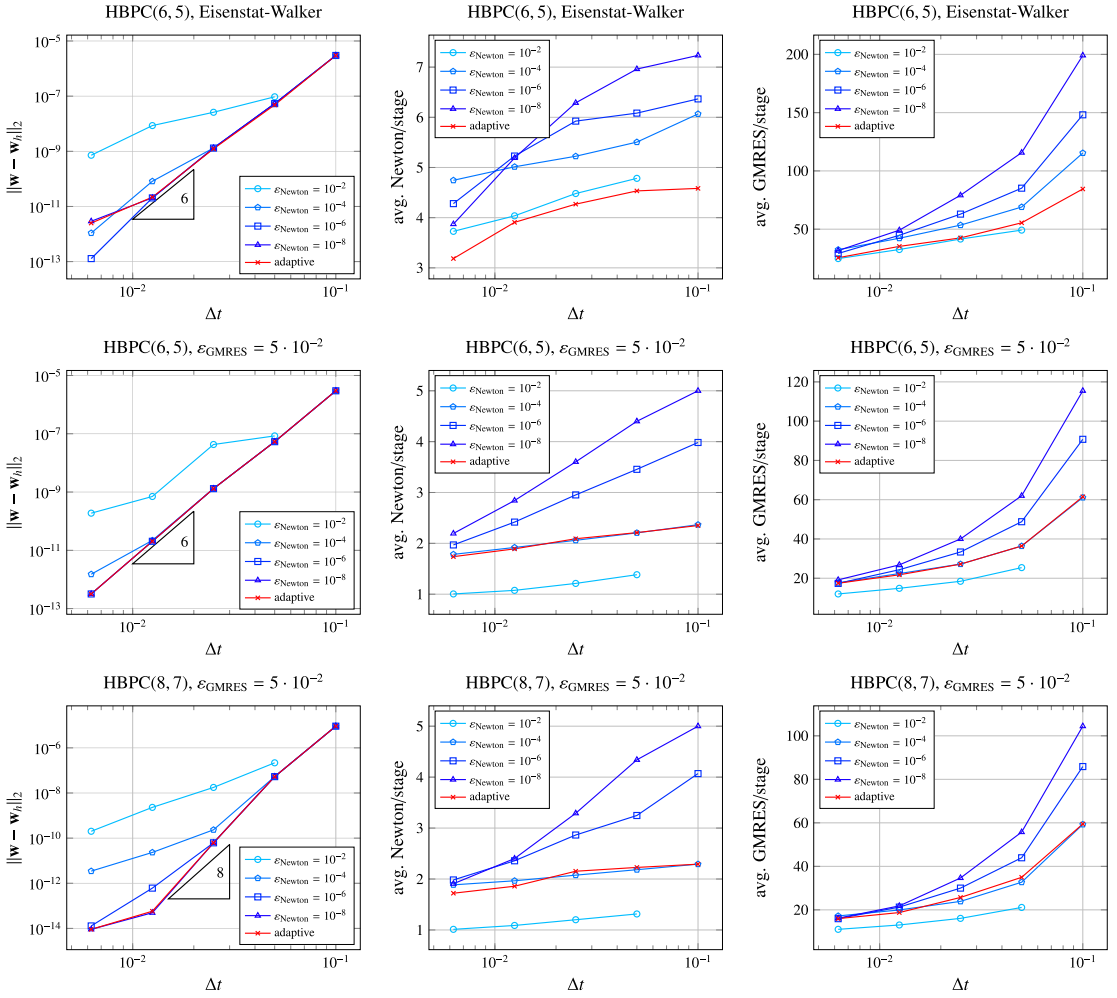


Fig. 4. Resulting L_2-error (left), average Newton iterations per stage (middle) and average GMRES iterations per stage (right) for HBPC(6, 5) with Eisenstat-Walker GMRES tolerance (top), HBPC(6, 5) with fixed GMRES tolerance (middle) and HBPC(8, 7) method with fixed GMRES tolerance (bottom) when choosing different convergence criteria for Newton's method. Adaptive Newton strategy is performed according to Eq. (35) with Newton increment extrapolation (Eq. (36) and Eq. (37)) and temporal error estimate according to Eq. (22).

the finest timestep size that the error curves associated to the adaptive strategy and the finest tolerance deviate slightly; in all the other cases, the obtained 'adaptive' error is equal to the one with the finest fixed Newton tolerance. This means that the adaptive strategy is succesful in the sense that it does not underresolve the algebraic systems of equations. The adaptive strategy is also successful w.r.t. the reduction of the required Newton and GMRES iterations, see Fig. 4 (middle, right). One can see that with the adaptive strategy, always at least two Newton iterations are performed. This is most likely caused by the fact that by choosing the Newton increment as convergence condition, we "lag" one Newton iteration, and the extrapolation procedure can only be applied after two Newton increments have been calculated.

Repetition of Introductory Example We now repeat the illustrative example shown in the introduction (see Fig. 1) with all the ingredients introduced in the previous sections. These are the parallelizable timestepping procedure described in Alg. 1, an improved initial Newton guess given by Eq. (21) and the adaptive Newton strategy with linear error extrapolation given by Eq. (35) and Eq. (36). Again, the constants C and η are chosen to be 0.5 and 0.1, respectively. Similar as for the introductory example, we use $\varepsilon_{\text{GMRES}} = 10^{-3}$ for the linear solver.

The total number of GMRES iterations and the normalized GMRES iterations are shown in Fig. 5. Compared to the simulation with the serial algorithm, using a fixed relative tolerance for the residual of Newton's method of $\varepsilon_{\text{Newton}} = 10^{-10}$, one can clearly see a much stronger dependency of the required absolute number of iterations on the chosen timestep size.

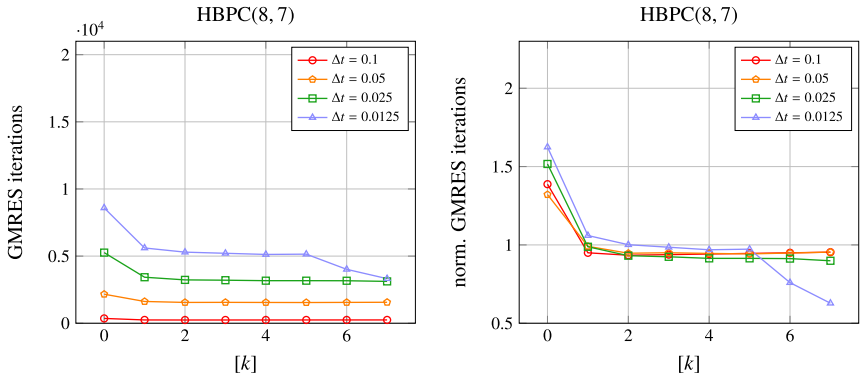


Fig. 5. Repetition of the numerical experiment from Section 1, Fig. 1, but with parallel-in-time algorithm (Alg. 1), adaptive Newton strategy (Eq. (35) and Eq. (36)) and improved initial Newton guess (Eq. (21)). Cumulated number (left) and normalized (right) GMRES iterations per prediction/correction step are shown. Normalization of the GMRES iterations per $[k]$ has been done with the mean GMRES iterations per timestep.

Moreover, the absolute values are much smaller than the ones reported in Fig. 1. Considering the normalized GMRES iterations, one can see that the relative cost of the predictor has been reduced and the costs of the correction steps have been homogenized. Moreover, the curves for the different timestep sizes coincide very well. The only exception from this behavior are the highest iterates, i.e. $k = 6, 7$, for the smallest timestep for which a reduced number of iterations is reported. This drop in iterations is most likely caused by hitting the absolute tolerance (see Eq. (38)).

5. Parallel performance

In this section, we investigate the parallel performance of the novel scheme. For that purpose, we first investigate the performance of the spatial parallelization. Next, we combine the spatial parallelization with the novel parallelization in time.

All simulations were performed on the *VSC Genius* cluster using up to 36 nodes. Each node has 192 GB RAM and consists of 18 cores, each equipped with 2 Xeon Gold 6240 CPUs@2.6 GHz (Cascadelake) processors. The connection between nodes is established with an Infiniband EDR network (25 GB/s bandwidth). The Fortran-written simulation code is compiled with the GCC compiler (6.4.0). It uses OpenMPI (v3.1.1) for the implementation of processor communication and OpenBLAS (v0.3.17) for the efficient implementation of the preconditioner's matrix inversion via LU-decomposition and matrix-matrix/matrix-vector multiplications. The single-derivative base-line code in which the novel method is implemented into is the open source code FLEXI[5], see also [35].

5.1. Parallel performance of spatial parallelization

The spatial parallelization is based on a domain decomposition via a space-filling curve. Each processor is responsible for its own set of elements and information between different processors is done via the surfaces of adjacent elements. A detailed overview on the parallelization strategy, including the communication pattern and an evaluation of the parallel efficiency with an explicit timestepping procedure can be found in [35].

We benchmark the spatial parallel performance of the implicit two-derivative method with two different settings: a two dimensional setup with $\mathcal{N}_p = 7$ and a three dimensional setup with $\mathcal{N}_p = 5$. For both cases we use Eq. (25) as initial condition, where $\mathbf{v} = (0.25, 0.25, 0.25)^T$ for the 3d and $\mathbf{v} = (0.3, 0.3)^T$ for the 2d-setup. The temporal discretization is done with the implicit two-derivative Taylor method and $\Delta t = 0.1$. We perform $\mathcal{N}_T = 10$ timesteps and use the relative tolerances $\varepsilon_{\text{Newton,rel}} = 10^{-3}$ and $\varepsilon_{\text{GMRES}} = 5 \cdot 10^{-2}$. The preconditioner is built once, and kept fixed for the whole simulation. Similar as it has been done in [1], we neglect the Hessian contribution when solving the linear system. A measure for the computational cost is the performance index (PID)

$$\text{PID} = \frac{\mathcal{T} \cdot \#\text{processors}}{\text{nDOF} \cdot \mathcal{N}_T \cdot (s-1)},$$

where $\mathcal{T}$ denotes the wallclocktime. PID measures the average time *per degree of freedom* that is necessary to perform one implicit stage of a single timestep. Note that, differently than for an explicit scheme, the absolute value of the PID does not transfer to different settings as it highly depends on the chosen test setup, i.e. on the implicit parameters and initial conditions[6] It can hence serve only as a relative measure. For the investigation of the parallel efficiency, a series of

[5] www.flexi-project.org, GNU GPL v3.0

[6] In particular, the conditioning of the nonlinear system of Eq. (16) plays an important role, which can be different for different test cases.

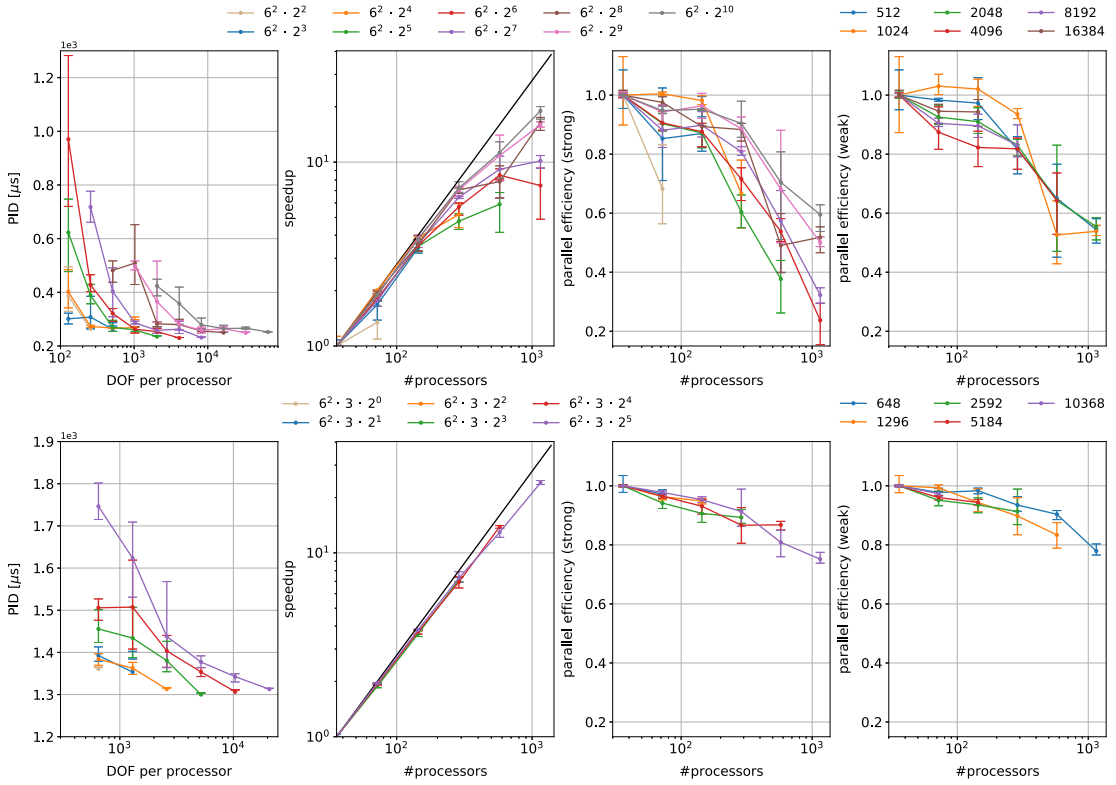


Fig. 6. Performance index (left), speedup (second to left), strong parallel efficiency (second to right) and weak parallel efficiency (right) for the 2d setup with $\mathcal{N}_p = 7$ (top) and the 3d setup with $\mathcal{N}_p = 5$ (bottom). For the PID, the strong scaling and the speedup, the legend indicates the different number of elements of different meshes. For the weak scaling, different lines correspond to different loads, i.e. different number of DOF per processor.

simulations on different meshes with varying number of processors is performed. For the 2d simulations, the smallest mesh has 12×12 elements to discretize the domain $\Omega = [-1, 1]^2$. Larger grids are obtained by doubling the number of elements and extending the domain Ω accordingly. The domain is extended to account for the fact that the CFL number should stay constant (note that $\Delta t = 0.1$ in this testcase), as otherwise, the behavior of the implicit algorithm changes drastically. The largest mesh has 192×192 elements for the domain $\Omega = [-128, 128]^2$. The meshes for the 3d simulations range from $6 \times 6 \times 3$ elements ($\Omega = [-2, 2] \times [-2, 2] \times [-1, 1]$) up to $24 \times 12 \times 12$ elements ($\Omega = [-8, 8] \times [-4, 4] \times [-4, 4]$). The lowest load, i.e. the lowest number of DOF per processor, is either obtained by using 1152 processors, or having 2 or 3 elements per processor for the 2d and 3d case, respectively. Each simulation is performed three times. The average, the minimum and the maximum PID of the simulations are reported in Fig. 6 (left). From those values, the weak and strong parallel efficiency as well as the speedup can be derived, see Fig. 6. Note that the minimum number of processors is 36, corresponding to one node.

One can see that the PID is a relatively constant quantity for small processor numbers. However, there is a strong increase if the load decreases below approximately 1000 DOF per processor and the number of processors increases. This increase additionally comes with an increasing variance of the measured PID. This is due to the increasing relative amount of communication and its high dependency on fluctuations of the machine's performance. Regarding the parallel efficiency, one can still observe a weak and strong efficiency of approx. 80% when using 1152 processors for the 3d simulations. For the 2d case, a significant dependency of the strong parallel efficiency on the amount of DOF can be observed. Depending on the number of DOF, it decreases until approx. 30% to 60% when using 1152 processors. Considering the weak parallel efficiency, one can observe a decrease towards approx. 55% when using 1152 processors. The main findings from this investigation are:

- Good weak scaling on up to more than 1000 processors indicates that the code is well-suited for large-scale applications.
- One can decrease the computational load per processor almost until the finest possible granularity (one element per processor) and still observe some speedup.

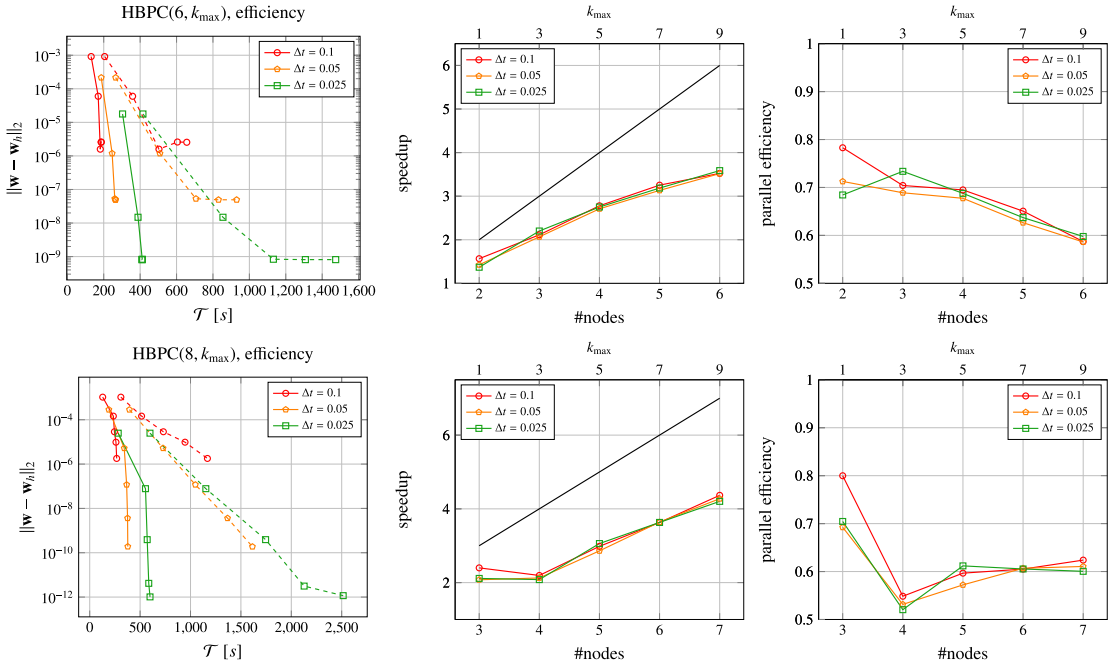


Fig. 7. Efficiency of temporal parallelization for HBPC(6, $k_{\max}$) (top) and HBPC(8, $k_{\max}$) method using different timestep sizes. The left plot shows required wallclocktime $\mathcal{T}$ vs. resulting error using different $k_{\max}$ for the parallel-in-time (solid) and serial method (dashed). The speedup (middle) gives the relation between serial and parallel wallclocktime. The black line indicates perfect speedup. The right plot displays the parallel efficiency, i.e. the relation between the actual speedup and the perfect speedup.

The significant differences between the 2d and the 3d simulations can be explained with the different ratios between internal work and communication, especially due to the increased work for matrix-matrix/matrix-vector multiplications.

5.2. Parallel performance of temporal parallelization

Next, we consider the parallel performance of the temporal parallelization. Setting up a fair evaluation problem for the parallel-in-time speedup is a non-trivial task [36]. One not only has to choose the problem, but also iterative procedures' parameters such that they are representative for the desired applications. We start with the same setup as used in the previous subsection with the initial conditions given by Eq. (25) with $\varepsilon = 1$, $\mathcal{N}_p = 7$ and use the adaptive Newton strategy introduced in Section 4. Again, the well-known values 0.5 and 0.1 are assigned to the constants C and η, respectively. Differently to the previous subsection, we choose $\mathcal{N}_E = 24^2$ for the domain $\Omega = [-1, 1]^2$ and the final time is set to $T_{end} = 10.0$. We then perform simulations with Alg. 1 running the HBPC(6, $k_{\max}$) and the HBPC(8, $k_{\max}$) serially and in parallel using $\Delta t = \{0.1, 0.05, 0.025\}$ which corresponds to performing $\mathcal{N}_T = \{100, 200, 400\}$ timesteps. The large number of timesteps ensures that the theoretical limit of the speedup for $\mathcal{N}_T \to \infty$ given by Eq. (12) is within reach. The preconditioner is rebuilt every 10^{th} timestep and the errors of the simulations are calculated by using the result of an explicit reference simulation with a fourth order low-storage Runge-Kutta method with a very small timestep ($\Delta t \approx 3.4 \cdot 10^{-4}$).

For the serial simulations we use one node with 36 processors corresponding to a load of 1024 DOF per processor for all $k_{\max} \in \{1, 3, 5, 7, 9\}$. For the parallel simulation we use multiple nodes, e.g. the parallel HBPC(8, 7) method uses 6 nodes with 36 processor each[7], resulting in 216 processors. Note that the load (i.e. DOF per processor) remains the same for all parallel-in-time and the serial simulations.

Parallel-in-Time Speedup In Fig. 7 we report the results of this series of simulations. On the left one can see the errors of the simulations w.r.t. the required wallclocktime $\mathcal{T}$. While different colors correspond to different timestep sizes, the solid and the dashed lines indicate the parallel and the serial simulations, respectively. Points being connected by a line indicate simulations with increasing $k_{\max} \in \{1, 3, 5, 7, 9\}$. One can see that the parallel method is more efficient (i.e. has a better

[7] HBPC(8, 7) is a method with four stages, the first stage being trivial. Hence, the splitted circles in Fig. 2 would be split into three rather than two. That means that three nodes are necessary for the handling of predictor and first corrector; and then one node each is necessary for $k = 2, 3$, $k = 4, 5$ and $k = 6, 7$. In total, this yields the six nodes used.

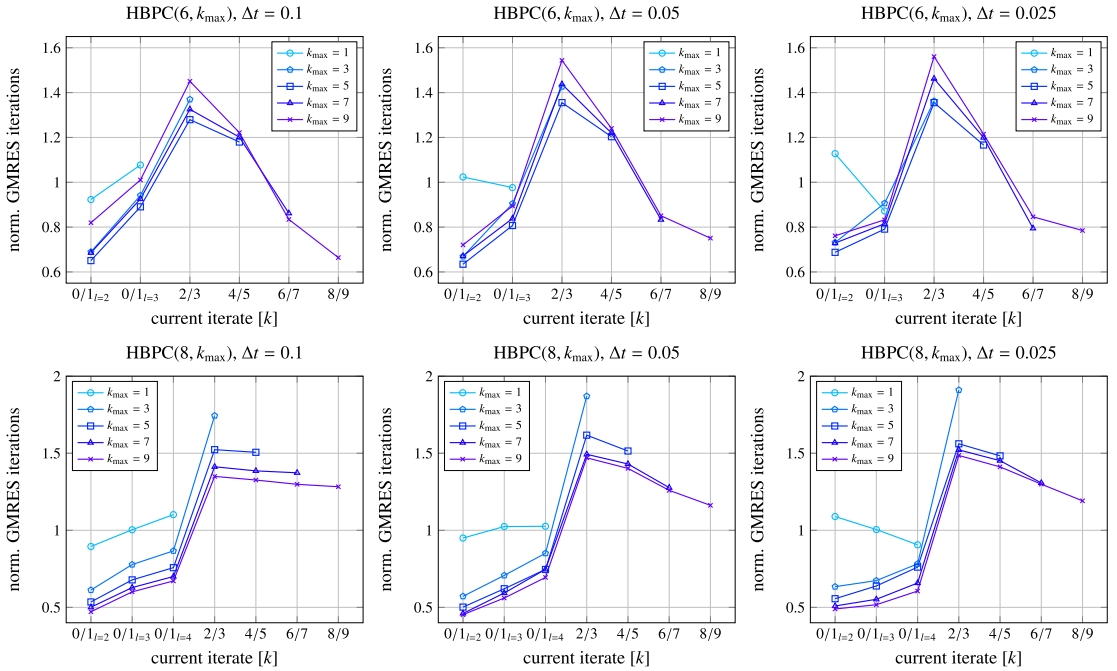


Fig. 8. Normalized GMRES iterations per iterate $[k]$ for the HBPC$(6, k_{\max})$ (top) and the HBPC$(8, k_{\max})$ (bottom) method using $\Delta t = 0.1$ (left), $\Delta t = 0.05$ (middle) and $\Delta t = 0.025$ (right). Note that the indexed iterates correspond to the different ranks of the temporal parallelization. Note that the iterates $k = 0/1$ are split up into the different implicit stages, see Fig. 2. For the normalization of the GMRES iterations, the mean GMRES iterations of all iterates k per simulation is used.

relation between accuracy and wallclocktime) than the serial method, which is indicated by a much steeper slope of the parallel methods. This behavior can be observed consistently for all considered timestep sizes and both, the HBPC$(6, k_{\max})$ and the HBPC$(8, k_{\max})$ method.

Calculating the ratio between the wallclocktimes of the serial and the parallel simulations, one obtains the speedup enabled by the temporal parallelization, visualized in Fig. 7 (middle). The black line indicates ideal speedup, i.e. when using 6 nodes, one expects a speedup of 6. Note that we have neglected the influence of a finite number of timesteps on the ideal speedup, see Eq. (12). Calculating the ratio between the ideal speedup and the actual achieved one, gives the parallel efficiency shown in Fig. 7 (right). Here, one can see that the parallel-in-time scheme achieves a parallel efficiency of approximately 60% when using $k_{\max} = 9$, corresponding to 6 and 7 nodes for the HBPC$(6, 9)$ and the HBPC$(8, 9)$ method, respectively.

A parallel efficiency of 60% on up to 7 partitions is in the same range as it has been reported for other PinT methods in literature: For solving ODEs with the implicit RIDC method, efficiencies of 90% and 69% were reported for 4 and 8 processors, respectively. For the HBPC scheme simulating ODEs, efficiencies up to 65% and 48% are measured for 4 and 18 processors. An inverted dual time stepping procedure is used in [37] and efficiencies of 95% and 45% are reported for 4 and 20 processors. Combined with additional spatial parallelization, an efficiency of 50% is obtained on 12 processors [9]. The reporting of parallel efficiencies of methods based on the parareal algorithm is difficult as the performance heavily depends on the problem to be solved [36]. Especially for hyperbolic dominated problems, the parareal algorithm can have relatively low efficiencies. In [38], fluid structure interaction problems with the incompressible Navier-Stokes equations are solved, parareal is used for the temporal parallelization and an efficiency up to 22% on 20 processors is reported. The compressible Navier-Stokes equations are solved in [8] and the authors show that, in combination with a spatial parallelization, the parareal algorithm shows efficiencies up to approx. 40% on 16 processors.

Work Distribution among the Parallel-in-Time Partitions To obtain some insight in the parallel efficiency, we consider the work distribution among the different parallel-in-time partitions in Fig. 8. We visualize the normalized GMRES iterations performed on each partition for all the parallel-in-time simulations. One important property of the novel method can be seen from the different graphs in Fig. 8: The curves for different $k_{\max}$ and different timestep sizes are very close to each other. This indicates that the adaptive strategy manages relatively well to balance the load over the processors.

Overall, the figure shows a qualitatively similar behavior for all simulations: While the partitions being responsible for one single stage of the predictor and the first corrector have the least load, the partition being responsible for $k = \{2, 3\}$ has

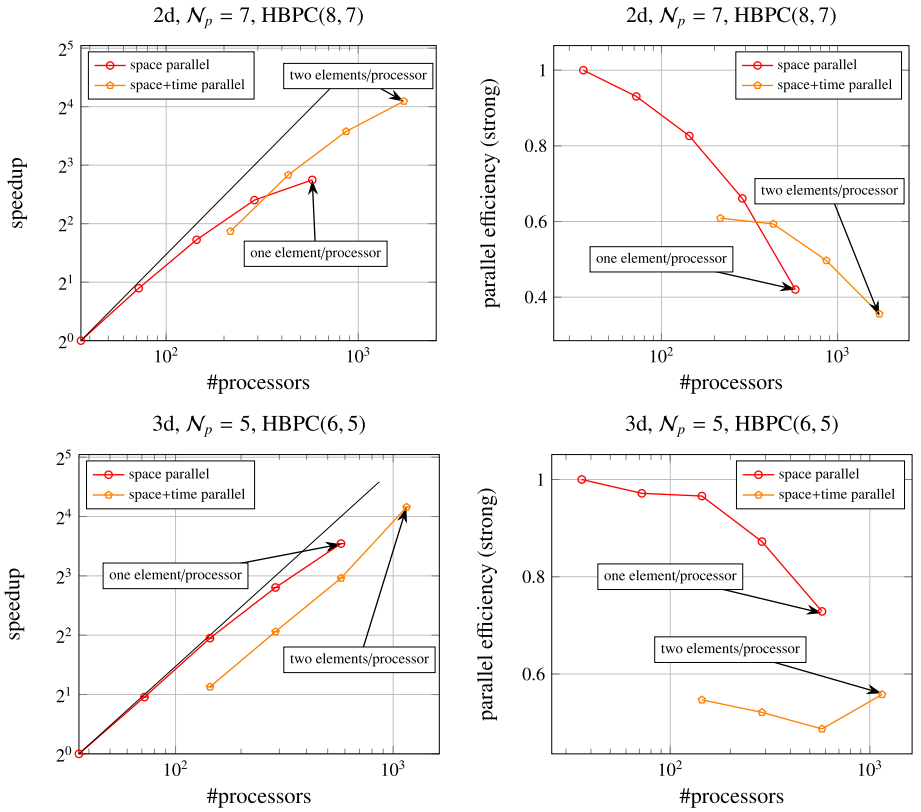


Fig. 9. Parallel speedup (left) and strong parallel efficiency (right) for a pure spatial and a mixed spatial/temporal parallelization for a 2d example using $\mathcal{N}_p = 7$ and HBPC(8, 7) (top) and a 3d example using $\mathcal{N}_p = 5$ and HBPC(6, 5) (bottom). Note that the very right point of the pure spatial parallelization (red, circles) corresponds to the finest possible granularity, i.e. one element per processor. The very right point of the mixed spatial/temporal parallelization corresponds to two elements per processor. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

the highest load. For higher iterates, the load decreases again. Comparing Fig. 8 and the parallel efficiency in Fig. 7, one can see that the value of the inverse of the maximum normalized GMRES iterations over all partitions translates almost directly to the achieved parallel efficiency. This highlights the importance of the homogenization of the work distribution among the different prediction/correction steps.

5.3. Space-time parallel performance

Next, we consider the parallel performance of the combined spatial and temporal parallelization. For that purpose, we again consider the initial conditions of the sine density wave given by Eq. (25). Two different representative configurations are investigated: A two dimensional problem on the domain $\Omega = [-2, 2]^2$ which is discretized with 24×24 elements using $\mathcal{N}_p = 7$, and a three dimensional problem on the domain $\Omega = [-2, 2]^3$ which is discretized with $8 \times 8 \times 9$ elements using $\mathcal{N}_p = 5$. For the 2d example the HBPC(8, 7) and for the 3d example the HBPC(6, 5) scheme is used leading to an 8^{th} order and a 6^{th} order scheme in space and time, respectively. The final time is set to $T_{end} = 5$ and the timestep is $\Delta t = 0.05$, leading to $\mathcal{N}_T = 100$ timesteps. The preconditioner is rebuilt every 10^{th} timestep and the linear solver tolerance is set to $\varepsilon_{\text{GMRES}} = 5 \cdot 10^{-2}$. C and η are set as before to 0.5 and 0.1, respectively.

In Fig. 9 the parallel speedups and the parallel efficiencies of a pure spatial and a mixed spatial/temporal parallelization are reported. Increasing the number of processors for the spatial discretization, the parallel efficiency decreases up to approx. 40% (2d) and 75% (3d). It is not possible to use more processors with the pure spatial parallelization for the considered test setups as the very right points of the red curves in Fig. 9 correspond to the finest possible granularity (i.e. one element per processor). If one wants to further reduce the required wallclocktime, spatial and temporal parallelization can be combined. One can clearly see that the temporal parallelization gives a further speedup. For the 2d simulation, one can see that the parallel efficiency can even be improved by combining spatial and temporal parallelization. Due to the very high parallel

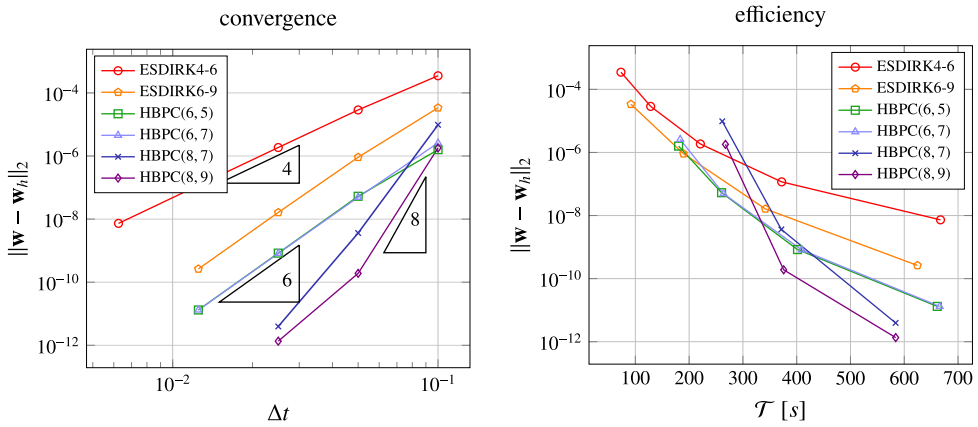


Fig. 10. Temporal convergence (left) and required wallclocktime (right) to achieve a certain error for different implicit schemes for the 2d sine wave problem with $\varepsilon = 1$ using different timestep sizes.

efficiency of the spatial parallelization for the 3d case, one cannot observe an increase in the parallel efficiency. However, due to the possibility to use more processors, a speedup can still be achieved.

Concluding, Fig. 9 shows that the temporal parallelization gives an efficiency gain if the pure spatial parallelization has an efficiency lower than the temporal one (i.e. lower than approx. 60%). I.e. the worse the parallel efficiency of the spatial operator, the more one can benefit from the temporal parallelization. Moreover, one can see that if the spatial parallelization reaches its limit, a further wallclocktime reduction can be achieved with the temporal parallelization.

5.4. Efficiency comparisons

In this final subsection, we compare the efficiency of the novel parallel-in-time two-derivative method with classical sequential-in-time single-derivative Runge-Kutta methods. We will use the 4^{th} order ARK4(3)6L[2]SA method [39, Appendix D] (abbreviated with ESDIRK4-6) and the 6^{th} order ESDIRK6(5)9L[2]SA method [40, Table 13] (abbreviated with ESDIRK6-9) as high order implicit ESDIRK methods[8]. For both, an adaptive Newton procedure that is based on the same principles as the one derived in Section 4 is used. Details on this can be found in Appendix C. We set $C = 0.5$ and $\eta = 0.1$.

5.4.1. Density sine wave

We start by considering the previously used example with the initial data given by Eq. (25) with $\varepsilon = 1$ and $\mathcal{N}_p = 7$ on the domain $\Omega = [-1, 1]^2$, discretized with $\mathcal{N}_E = 24 \times 24$ and with $T_{end} = 10$. We run the simulation by choosing different timesteps with the parallel-in-time HBPC(6, 5), HBPC(6, 7), HBPC(8, 7) and HBPC(8, 9) schemes. Additionally, the ESDIRK4-6 and ESDIRK6-9 are used as a reference. The simulations are performed on one node with 36 processors; for the parallel-in-time methods the respective multiples are used. The preconditioners are rebuilt every 10^{th} timestep and $\varepsilon_{\text{GMRES}} = 5 \cdot 10^{-2}$ is chosen for the linear solver. Initially, when no adaptive Newton criterion is available, we choose $\varepsilon_{\text{Newton,rel}} = 10^{-8}$. C and η are set to 0.5 and 0.1, respectively.

Fig. 10 shows the temporal convergence (left) and the efficiency (right) of the different methods. One can see that all methods show the desired order of convergence. The sixth order HBPC schemes show a smaller error than the sixth order ESDIRK method. Considering the efficiency, one can see that if small errors are desirable, the higher order methods pay off. Moreover, one can see that the sixth order HBPC method is superior to the ESDIRK6-9 scheme for smaller errors.

5.4.2. Cylinder flow

Next, we consider the two dimensional flow around a cylinder. Similar as it has been done by other authors, see e.g. [41,42], we consider the aerodynamic coefficients as a quality measure. The flow parameters for the cylinder flow with diameter $D = 1$ are given by the reference Mach number $\varepsilon = 0.1$ and the Reynolds number $\text{Re}_D = 200$. The initial conditions are given by the constant state

$$\rho_0 = 1, \quad \mathbf{v}_0 = (1, 0)^T, \quad p_0 = \frac{1}{\gamma},$$

[8] While HBPC(8, 7) and HBPC(8, 9) are schemes of order eight, we did, despite a thorough literature study, not find an eighth-order diagonally implicit Runge-Kutta scheme, see also [25]. Hence, only fourth- and sixth-order Runge-Kutta schemes are used for comparison.

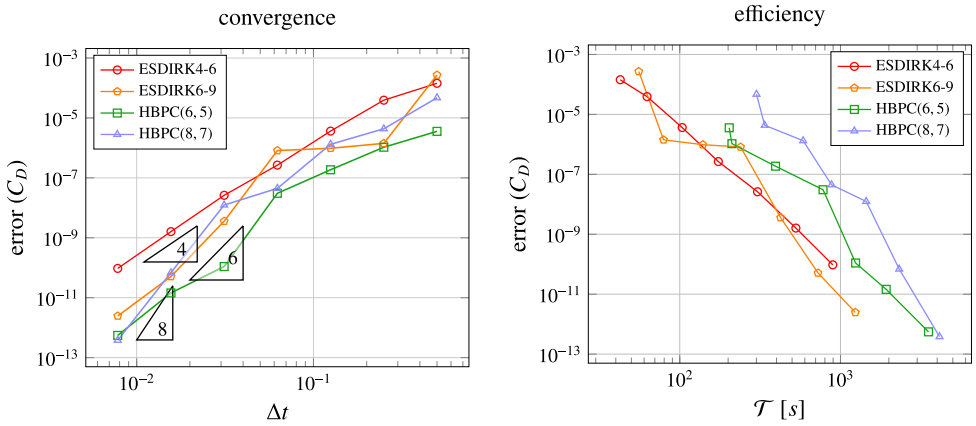


Fig. 11. Temporal convergence (left) of mean drag coefficient C_D and required wallclocktime (right) to achieve a certain error for different implicit schemes for the cylinder flow problem at $\mathrm{Re}_D = 200$ using different timestep sizes.

and the cylindrical domain with an outer diameter of $D_\infty = 200$ is discretized using $\mathcal{N}_E = 1200$ with $\mathcal{N}_p = 5$. On the cylinder surface, wall boundary conditions are prescribed. At the outer boundary, Dirichlet boundary conditions with the constant initial state are used. A more detailed description of the used mesh is available in [5, Sec. 5.1.1]. We run the simulation with a fourth order low-storage Runge-Kutta time discretization (ERK4) [26] up to $T = 400$. At this time, the flow is fully developed and a vortex shedding has been established. In the interval from $T = 400$ to $T = 500$, we obtain $C_D \approx 1.35$, a fluctuating lift coefficient of $C_{L'} \approx 0.501$ and a Strouhal number of $\mathrm{Sr} \approx 0.197$. Those aerodynamic measures agree well with data from literature, see e.g. [43–45]. Differences are most likely due to the relatively coarse spatial resolution.

To evaluate the performance of the novel scheme, we use the mean drag coefficient

$$C_D := \frac{2\bar{F}_x}{\rho_0 \|\mathbf{v}_0\|_2^2 D},$$

where $\bar{F}_x$ denotes the mean drag force at the cylinder surface as a measure for the accuracy. We restart the simulation at $T = 400$ and run it approximately for two vortex shedding periods ($T_{end} = 410$) using different timestepping methods and timestep sizes. As this is a quasi-steady flow, we estimate the temporal error only once during the adaptive Newton procedure. The aerodynamic forces are measured at the time intervals $\Delta t = 0.5$ and $\bar{F}_x$ is calculated via mean of these discrete values. A simulation with a very small explicit timestep for the ERK4 serves as a reference solution to calculate an error measure.

As we are using relatively large timesteps and $\varepsilon = 0.1$, we use Eq. (23) to estimate the temporal error and do not perform an explicit step for the initial Newton guess. Moreover, we set $C = 1$ in the adaptive Newton procedure, see Eq. (35). For the linear solver, the adaptive Eisenstat-Walker procedure is used. During the startup procedure of the adaptive Newton strategy, a relative tolerance of $\varepsilon_{\text{Newton,rel}} = 10^{-4}$ is used. The ESDIRK schemes initially use a Newton tolerance of $\varepsilon_{\text{Newton,rel}} = 10^{-6}$. We choose the safety factor $\eta = 0.1$ for the ESDIRK4-6 and the ESDIRK6-9 in Eq. (C.2).

In Fig. 11 the convergence and efficiency considering the drag coefficient are visualized on the left and right, respectively. One can see that the ESDIRK4-6 reaches the desired order of convergence. The sixth order schemes have difficulties to reach the desired order for large timesteps but reach the asymptotic regime for small timesteps. The eighth order HBPC(8, 7) scheme also achieves almost its desired order for the small timesteps. The figure shows that the HBPC(6, 5) scheme has smaller errors than the ESDIRK methods using the same timestep. However, the HBPC(8, 7) scheme could outperform the sixth order schemes in terms of error only for a few timesteps. Considering the efficiency, the ESDIRK4-6 seems to be superior to the other methods. The ESDIRK6-9 and the HBPC(6, 5) are within reach, but are only able to outperform the ESDIRK4-6 for very few settings. One reason for the good behavior of the forth order method could be its L-stability. As both, the ESDIRK6-9 and the HBPC(6, $k_{\max}$) are not L-stable, ESDIRK4-6 is naturally better suited for stiff problems. Note that the results of this investigation depend on the equipped error measure and physical setting. Using another error measure or using another mesh and/or Reynolds number could lead to slightly different results.

5.4.3. Taylor-Green-Vortex

Finally, we consider the three dimensional Taylor-Green-Vortex (TGV). It is a prototypical periodic test case to study the transition to turbulence and its decay. The initial data for the non-dimensional equations (see also [46]) are given by

$$\rho = 1, \quad \mathbf{v} = \begin{pmatrix} \cos(x)\cos(y)\cos(z) \\ -\cos(x)\sin(y)\cos(z) \\ 0 \end{pmatrix}, \quad \text{and} \quad p = \frac{\rho}{\gamma} + \frac{\rho\varepsilon^2}{16}(\cos(2x) + \cos(2y))(\cos(2z) + 2),$$

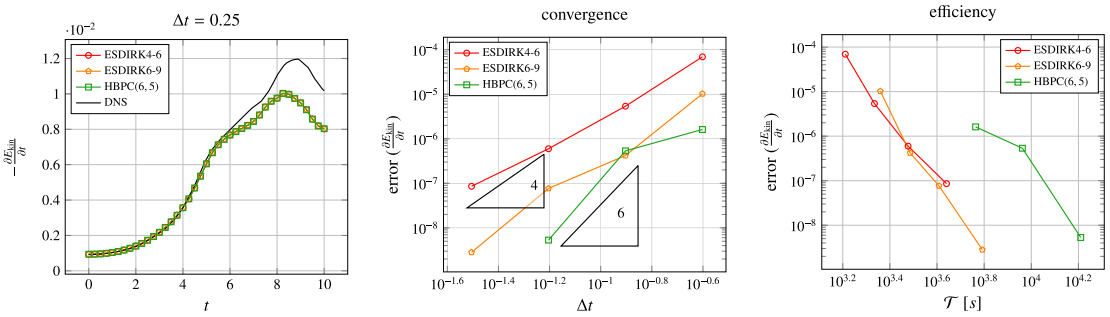


Fig. 12. Taylor-Green vortex at Re = 800 and $\varepsilon = 10^{-1}$: temporal evolution of kinetic energy dissipation rate (left) with $\Delta t = 0.25$, DNS data from [47]. Temporal convergence (middle) and required wallclocktime (right) to achieve a certain error for different implicit schemes using different timestep sizes.

on the periodic domain $\Omega = [0, 2\pi]^3$, which we discretize with $\mathcal{N}_E = 16 \times 16 \times 16$ and $\mathcal{N}_p = 3$. We use $\varepsilon = 10^{-1}$, a Reynolds number of Re = 800 and $T_{end} = 10$. A measure for the turbulent decay is the dissipation rate of the kinetic energy

$$\frac{\partial E_{\text{kin}}}{\partial t} = \frac{\mu}{\rho \|\Omega\|} \int_{\Omega} \nabla_x \mathbf{v} : \nabla_x \mathbf{v} d\mathbf{x}.$$

We use the dissipation rate as an error measure by calculating the L_1-norm of the measured dissipation rates in the time intervals $\Delta t = 0.25$. An explicit simulation with a very small timestep (ERK4, $\Delta t \approx 1.4 \cdot 10^{-3}$) serves as a reference. Similar as it has been done in Section 5.4.2, the ESDIRK4-6 and the ESDIRK6-9 use $\eta = 0.1$. For the HBPC(6, 5), the temporal error is estimated according to Eq. (23) and $C = 0.5$, $\eta = 0.1$. During the startup procedure of the adaptive Newton strategy, a relative tolerance of $\varepsilon_{\text{Newton,rel}} = 10^{-4}$ is used. The ESDIRK schemes initially use a Newton tolerance of $\varepsilon_{\text{Newton,rel}} = 10^{-6}$. At every 10-th timestep, the temporal error is estimated and the preconditioners are rebuilt. Simulations are performed on one node with 36 processors, or the respective multiples for the temporal parallelization. Please note that we do not report on HBPC(8, 7) results in this section. Due to limited computational resources, and the large amount of degrees of freedom present, we could not test the method sufficiently such that fair and reliable results can be guaranteed. Only very preliminary results are available that show that the method does not suffer from stability issues, and numerical errors are at least in the order of the other methods shown here.

The dissipation rate for the ESDIRK4-6, ESDIRK6-9 and the HBPC(6, 5), each with $\Delta t = 0.25$ are shown in Fig. 12 (left). Virtually, all schemes coincide; deviations from the DNS data [47] are due to the too coarse spatial resolution. In Fig. 12 (middle and right) the convergence and the efficiency w.r.t. the dissipation rate for the ESDIRK4-6, the ESDIRK6-9 and the HBPC(6, 5) are shown. One can see that the HBPC(6, 5) has smaller errors than the ESDIRK methods for the same chosen timestep sizes. However, this advantage does not directly transfer to higher efficiencies. This motivates further research on the development of the HBPC methods as outlined in the next section, possibly offering higher accuracies and efficiencies for stiff problems.

6. Conclusion and outlook

In this work, we have shown the application of a parallel-in-time implicit two-derivative discontinuous Galerkin method to the Navier-Stokes equations. As time discretization the HBPC scheme has been used. In previous works, this time discretization has been combined with the discontinous Galerkin method [1] to solve PDEs and the concept of time parallelism has been shown for ODEs [3]. The present work tackles practical aspects of combining a space-parallel discontinuous Galerkin PDE discretization with the time-parallel HBPC scheme.

A homogeneous distribution of linear iterations over the different processors has been identified as a key for parallel efficiency. Two main ingredients have been introduced for that purpose: an adaptive procedure for Newton's method, and an additional distribution of the predictor's and first corrector's stages to different processors. It has been shown that the temporal parallelization reaches a parallel efficiency of approx. 70 – 60% on 4 – 7 partitions. The pure spatial parallelization has been shown to be well suited for parallel computing as it provides 50 – 80% parallel efficiency for very fine granularities on more than 1000 processors. Combining spatial and temporal parallelization offers the possibility to obtain further speedup and in some cases also an improved efficiency over the pure spatial parallelization. This has also been demonstrated for settings with more than 1000 processors, highlighting the capability of the novel method to solve large-scale problems. Furthermore, the novel method has been compared with serial-in-time ESDIRK methods in terms of efficiency. We have shown that in some cases the novel method can outperform these schemes.

We consider the current paper as a milestone towards making two-derivative predictor corrector schemes a viable alternative to established schemes in applications from compressible flows. Obviously, the results in Section 5 show that there is still room for improvement. Active research lines include the identification of more suited background schemes in

Eq. (6) (other collocation points, extension to general linear methods); the use of IMEX schemes within this context, see [2,3,48] for first attempts in the context of ODEs; and Jacobian-free high-derivative schemes, where more than two temporal derivatives are added to the algorithm using a suitable finite difference approach, see [49] for first attempts. Further developments will consider full adaptivity in space and time, hence making use of spatial error estimators to determine both hp-adaptivity and non-constant timesteps.

Declaration of Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

Acknowledgments

J. Zeifang was funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) - project no. 457811052. A. Thenery Manikantan was funded by the "Bijzonder Onderzoeksfonds" (BOF) from UHasselt - project no. BOF21KP12. We acknowledge the VSC (Flemish Supercomputer Center) for providing computing resources. The VSC is funded by the Research Foundation - Flanders (FWO) and the Flemish Government.

Appendix A. Navier-Stokes fluxes

For the Navier-Stokes Eq. (1), inviscid and viscous fluxes $\mathbf{F}(\mathbf{w})$ and $\mathbf{F}^\nu(\mathbf{w}, \nabla_x \mathbf{w})$ are given by

$$\mathbf{F}(\mathbf{w}) = \begin{pmatrix} \rho \mathbf{v} \\ \rho \mathbf{v} \otimes \mathbf{v} + \frac{1}{\varepsilon^2} p \cdot \mathrm{Id} \\ \mathbf{v}(E+p) \end{pmatrix}, \quad \text{and} \quad \mathbf{F}^\nu(\mathbf{w}, \nabla_x \mathbf{w}) = \begin{pmatrix} 0 \\ \boldsymbol{\tau} \\ \boldsymbol{\tau} \cdot \mathbf{v} + \mathbf{q} \end{pmatrix}. \tag{A.1}$$

Pressure is coupled to density, momentum and energy via the ideal gas equation of state,

$$p = (\gamma - 1)\left(E - \frac{\varepsilon^2}{2} \rho \|\mathbf{v}\|_2^2\right).$$

The viscous stress tensor $\boldsymbol{\tau}$ and the heat flux $\mathbf{q}$ are defined as

$$\boldsymbol{\tau} := \mu\left(\nabla_x \mathbf{v} + (\nabla_x \mathbf{v})^T - \frac{2}{3}(\nabla_x \cdot \mathbf{v})\mathrm{Id}\right), \quad \text{and} \quad \mathbf{q} := \lambda_T \nabla_x T,$$

where T denotes temperature, μ dynamic viscosity, the thermal conductivity $\lambda_T = \frac{c_p \mu}{Pr}$, specific heat capacity $c_p = \frac{R\gamma}{\gamma - 1}$, specific gas constant $R = \frac{1}{\gamma \varepsilon^2}$, the ideal gas law $p = \rho R T$ and the fluid specific Prandtl number $Pr = 0.72$.

Appendix B. Butcher Tables of the background Hermite-Birkhoff Runge-Kutta Methods

We consider the following quadrature rules:

- A sixth-order method ($q = 6$) with three stages ($s = 3$, one being fully explicit), as also used in [3,19]

$$c = \begin{pmatrix} 0 \\ \frac{1}{2} \\ 1 \end{pmatrix}, \quad B^{(1)} = \begin{pmatrix} 0 & 0 & 0 \\ \frac{101}{480} & \frac{8}{30} & \frac{55}{2400} \\ \frac{7}{30} & \frac{16}{30} & \frac{7}{30} \end{pmatrix}, \quad B^{(2)} = \begin{pmatrix} 0 & 0 & 0 \\ \frac{65}{4800} & -\frac{25}{600} & -\frac{25}{8000} \\ \frac{1}{60} & 0 & -\frac{1}{60} \end{pmatrix}, \tag{B.1}$$

  with the fifth-order ($\hat{q} = 5$) embedded quadrature rule

$$\hat{c} = \begin{pmatrix} 0 \\ \frac{1}{2} \\ 1 \end{pmatrix}, \quad \hat{B}^{(1)} = \begin{pmatrix} 0 & 0 & 0 \\ \frac{31}{240} & \frac{4}{15} & \frac{5}{48} \\ \frac{2}{15} & \frac{8}{15} & \frac{1}{3} \end{pmatrix}, \quad \hat{B}^{(2)} = \begin{pmatrix} 0 & 0 & 0 \\ 0 & -\frac{23}{240} & -\frac{1}{60} \\ 0 & -\frac{1}{15} & -\frac{1}{30} \end{pmatrix}. \tag{B.2}$$

- An eighth-order method ($q = 8$) with four stages ($s = 4$, one being fully explicit), as also used in [3]

$$c = \begin{pmatrix} 0 \\ \frac{1}{3} \\ \frac{2}{3} \\ 1 \end{pmatrix}, \quad B^{(1)} = \begin{pmatrix} 0 & 0 & 0 & 0 \\ \frac{6893}{54432} & \frac{313}{2016} & \frac{89}{2016} & \frac{397}{54432} \\ \frac{223}{1701} & \frac{20}{63} & \frac{13}{63} & \frac{20}{1701} \\ \frac{31}{224} & \frac{81}{224} & \frac{81}{224} & \frac{31}{224} \end{pmatrix}, \quad B^{(2)} = \begin{pmatrix} 0 & 0 & 0 & 0 \\ \frac{1283}{272160} & -\frac{851}{30240} & -\frac{269}{30240} & -\frac{163}{272160} \\ \frac{43}{8505} & -\frac{16}{945} & -\frac{19}{945} & -\frac{8}{8505} \\ \frac{19}{3360} & -\frac{9}{1120} & \frac{9}{1120} & -\frac{19}{3360} \end{pmatrix}, \tag{B.3}$$

with the seventh-order ($\hat{q} = 7$) embedded quadrature rule

$$\hat{c} = \begin{pmatrix} 0 \\ \frac{1}{3} \\ \frac{2}{3} \\ 1 \end{pmatrix}, \quad \hat{B}^{(1)} = \begin{pmatrix} 0 & 0 & 0 & 0 \\ \frac{212}{2835} & \frac{47}{1680} & \frac{6}{35} & \frac{171}{2891} \\ \frac{214}{2835} & \frac{19}{105} & \frac{12}{35} & \frac{191}{2835} \\ \frac{8}{105} & \frac{117}{560} & \frac{18}{35} & \frac{337}{1680} \end{pmatrix}, \quad \hat{B}^{(2)} = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & -\frac{299}{4237} & -\frac{97}{1890} & -\frac{51}{9599} \\ 0 & -\frac{59}{945} & -\frac{62}{945} & -\frac{17}{2835} \\ 0 & -\frac{33}{560} & -\frac{3}{70} & -\frac{19}{1680} \end{pmatrix}, \tag{B.4}$$

Appendix C. Adaptive Criterion for ESDIRK Method

For the comparisons in Section 5.4, we use two different ESDIRK methods. The 4^{th} order ARK4(3)6L[2]SA method [39, Appendix D] (abbreviated with ESDIRK4-6) and the 6^{th} order ESDIRK6(5)9L[2]SA method [40, Table 13] (abbreviated with ESDIRK6-9)[9]. For those single-derivative implicit ESDIRK methods, the non-linear equation to be solved for each stage l is given by

$$\boldsymbol{X}^l = \mathbf{w}_{\text{old}} + \Phi\left(\boldsymbol{X}^l, \boldsymbol{X}^{1:l-1}\right),$$

with $\boldsymbol{X}^l := \mathbf{w}_h^{n,l}$, where $\mathbf{w}_h^{n,l}$ denotes the discrete $\mathbf{w}$ at time t^n and stage l. For the ease of notation, we use $\boldsymbol{X}^{1:l-1} := \left(\boldsymbol{X}^1, \boldsymbol{X}^2, \dots, \boldsymbol{X}^{l-1}\right)$. The function Φ is given by

$$\Phi\left(\boldsymbol{X}^l, \boldsymbol{X}^{1:l-1}\right) = \Delta t B_{ll}^{(1)} \mathbf{R}_h^{(1)}(\mathbf{w}_h^{n,l}) + \Delta t \sum_{i=1}^{l-1} B_{li}^{(1)} \mathbf{R}_h^{(1)}(\mathbf{w}_h^{n,i}).$$

Note that the introduction of $\boldsymbol{X}^l$ would not have been necessary for this method. Nevertheless, we introduce it here to highlight the similarities with the two-derivative method, see Eq. (16). As initial guess for Newton's method we use $\mathbf{w}_{h,0}^{n,l} = \mathbf{w}_h^{n,l-1}$. After defining the error introduced by Newton's method

$$\mathcal{E}_{\text{r}}^l := \boldsymbol{X}^l - \boldsymbol{X}_{\text{r}}^l,$$

where the subscript r denotes the solution of the r-th Newton step, we follow the steps outlined in Section 4.1.1 and find

$$\begin{aligned} \mathcal{E}_{\text{r}}^l \doteq & \Delta \boldsymbol{X}_{\text{r}+1} + \left(\text{Id} - \frac{\partial \Phi\left(\boldsymbol{X}^l, \boldsymbol{X}^{1:l-1}\right)}{\partial \boldsymbol{X}^l} \right)^{-1} \cdot \left(\sum_{i=1}^{l-1} \frac{\partial \Phi\left(\boldsymbol{X}^l, \boldsymbol{X}^{1:l-1}\right)}{\partial \boldsymbol{X}^i} \mathcal{E}_{\text{r}'}^i \right) \\ = & \Delta \boldsymbol{X}_{\text{r}+1} + \left(\text{Id} - \Delta t B_{ll}^{(1)} \frac{\partial \mathbf{R}_h^{(1)}}{\partial \mathbf{w}_h^{n,l}} \right)^{-1} \cdot \left(\Delta t \sum_{i=1}^{l-1} B_{li}^{(1)} \frac{\partial \mathbf{R}_h^{(1)}}{\partial \mathbf{w}_h^{n,i}} \mathcal{E}_{\text{r}'}^i \right). \end{aligned}$$

The limits of this equations can then be found as

$$\begin{aligned} \|\mathcal{E}_{\text{r}}^l\|_2 &\approx \|\Delta \boldsymbol{X}_{\text{r}+1}\|_2 && \text{for } \Delta t \to 0, \\ \|\mathcal{E}_{\text{r}}^l\|_2 &\approx \|\Delta \boldsymbol{X}_{\text{r}+1}\|_2 + \sum_{i=2}^{l-1} \tilde{C}_i \left| \frac{B_{li}^{(1)}}{B_{ll}^{(1)}} \right| \|\mathcal{E}_{\text{r}'}^i\|_2 && \text{for } \Delta t \to \infty, \end{aligned}$$

for some constants $\tilde{C}_i \approx \left\| \left(\frac{\partial \mathbf{R}_h^{(1)}}{\partial \mathbf{w}_h^{n,l}} \right)^{-1} \cdot \left(\frac{\partial \mathbf{R}_h^{(1)}}{\partial \mathbf{w}_h^{n,i}} \right) \right\|_2$, which equal one for a linear system. Note that due to the explicit evaluation of the first stage it holds $\mathcal{E}_{\text{r}}^1 = 0$. We can hence find a condition for the Newton increment of stage l via

$$\|\Delta \boldsymbol{X}_{\text{r}+1}^l\|_2 + C \sum_{i=2}^{l-1} \|\Delta \boldsymbol{X}_{\text{r}+1}^i\|_2 + \mathcal{O}(C^2) + \dots + \mathcal{O}(C^{l-2}) \leq \eta \|\mathcal{E}_t\|_2. \tag{C.1}$$

As for the used ESDIRK methods the last stage directly gives the solution at the new timestep, we have defined $\|\mathcal{E}_t\|_2 := \|\mathcal{E}_{\text{r}}^s\|_2$. Similar as for the HBPC methods, C is a function of the timestep, though, we choose $C = 0.5$ to be constant in this

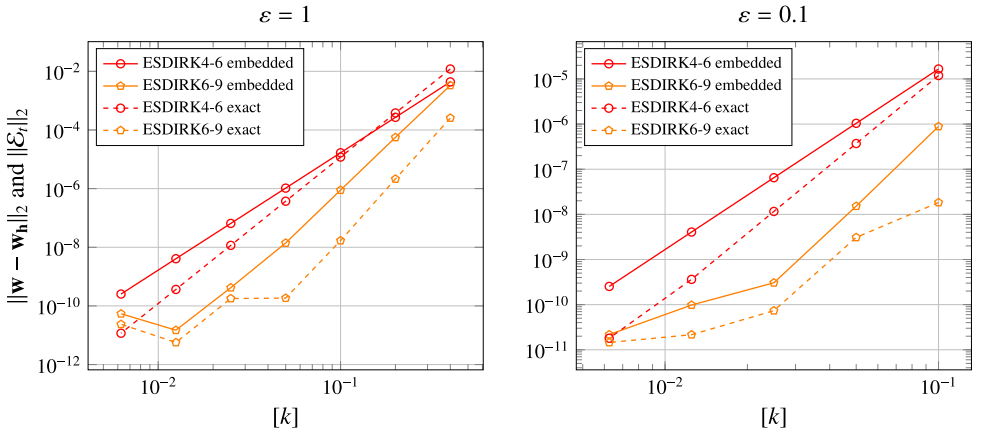


Fig. C.13. Accuracy of embedded error estimate for ESDIRK4-6 and ESDIRK6-9 using the initial conditions given by Eq. (25), $\mathcal{N}_p = 7$ and $\mathcal{N}_E = 32^2$. The reference Mach number is set to $\varepsilon = 1$ (left) and $\varepsilon = 10^{-1}$ (right).

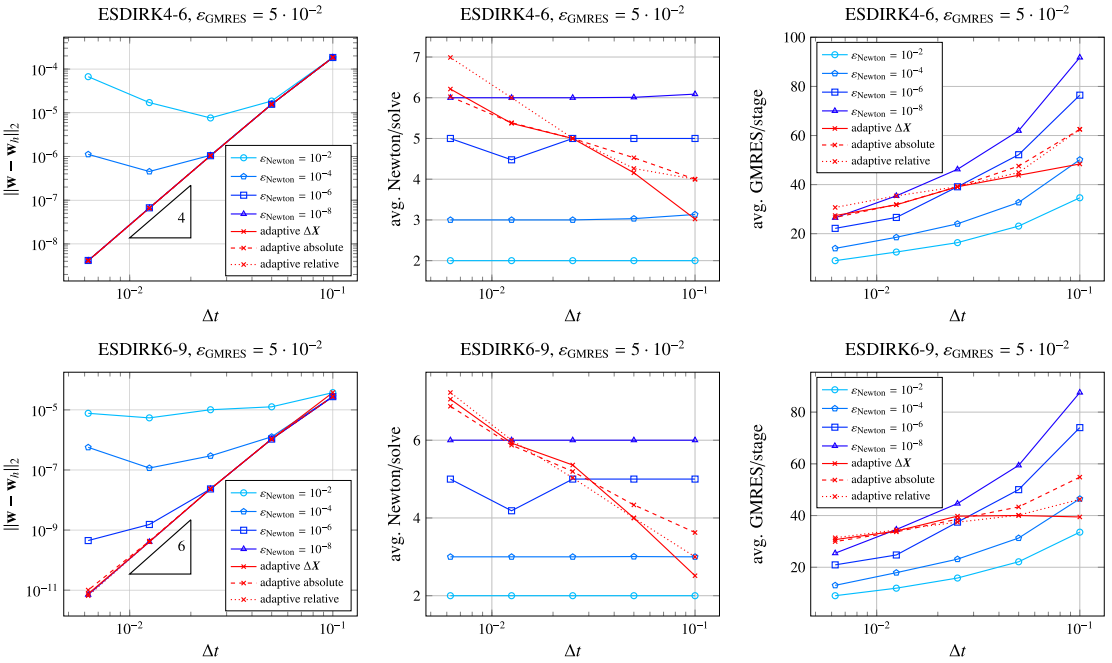


Fig. C.14. Resulting L_2-error (left), average Newton iterations per stage (middle) and average GMRES iterations per stage (right) for ESDIRK4-6 (top) and ESDIRK6-9 (bottom) with fixed GMRES tolerance when choosing different convergence criteria for Newton's method. Adaptive Newton strategy is performed according to Eq. (C.2) (adaptive $\Delta \boldsymbol{X}$), $N(\boldsymbol{X}_\mathrm{r}) \leq \eta \|\mathcal{E}_t\|_2$ (adaptive absolute) or $N(\boldsymbol{X}_\mathrm{r}) \leq \eta \|\mathcal{E}_t\|_2 N(\boldsymbol{X}_0)$ (adaptive relative). Note that the legend in the middle figures has been omitted for clarity but is the same as for the left and the right plot.

paper and truncate all higher order terms of C in Eq. (C.1). This corresponds to neglecting secondary effects of error propagation from previous stages. Demanding that the Newton increments of all stages are below a common threshold, we can then find

$$\|\Delta \boldsymbol{X}^l_{\mathrm{r}+1}\|_2 \leq \frac{1}{1 + C(s-2)} \eta \|\mathcal{E}_t\|, \qquad \text{for} \quad l = 2, \ldots, s, \tag{C.2}$$

[9] Note that in the original publication [40, Table 13] there is a typo for $\hat{b}_4$. The value should be $\hat{b}_4 = \frac{1376520686137389}{1064235527052079}$.

which can be combined with the Newton error extrapolation described in Eq. (36) and Eq. (37). Note that, similar as for the HBPC schemes, we have introduced a safety factor η that accounts for the non-exact error estimate via the embedded temporal error estimate. In Fig. C.13 the accuracy of the embedded error estimates are shown for the sine wave example and the same setup as used in Section 4.1.4.

The results suggest that choosing $\eta \leq 0.1$ is reasonable.

We now combine the embedded error estimate and the adaptive Newton criterion given by Eq. (C.2). The effectiveness of this adaptive strategy is highlighted in Fig. C.14. One can see that the novel adaptation strategy is at least as good as the standard strategies based on the Newton residual and outperforms them for large timesteps.

References

[1] J. Zeifang, J. Schütz, Implicit two-derivative deferred correction time discretization for the discontinuous Galerkin method, J Comput Phys 464 (2022) 111353.
[2] J. Schütz, D. Seal, An asymptotic preserving semi-implicit multiderivative solver, Appl. Numer. Math. 160 (2021) 84–101.
[3] J. Schütz, D.C. Seal, J. Zeifang, Parallel-in-time high-order multiderivative IMEX solvers, J Sci Comput 90 (54) (2022) 1–33.
[4] J. Zeifang, J. Schütz, D. Seal, Stability of implicit multiderivative deferred correction methods, BIT Numerical Mathematics (2022).
[5] S. Vangelatos, On the Efficiency of Implicit Discontinuous Galerkin Spectral Element Methods for the Unsteady Compressible Navier-Stokes Equations, University of Stuttgart, 2019 Ph.D. thesis.
[6] E. Ferrer, G. Rubio, G. Ntoukas, W. Laskowski, O. Mariño, S. Colombo, A. Mateo-Gabín, H. Marbona, F. Manrique de Lara, D. Huergo, J. Manzanero, A. Rueda-Ramírez, D. Kopriva, E. Valero, HORSES3D: a high-order discontinuous Galerkin solver for flow simulations and multi-physics applications, Comput Phys Commun 287 (2023) 108700.
[7] F.D. Witherden, A.M. Farrington, P.E. Vincent, PyFR: an open source framework for solving advection–diffusion type problems on streaming architectures using the flux reconstruction approach, Comput Phys Commun 185 (11) (2014) 3028–3040.
[8] T. Lunet, J. Bodart, S. Gratton, X. Vasseur, Time-parallel simulation of the decay of homogeneous turbulence using parareal with spatial coarsening, Comput Vis Sci 19 (1) (2018) 31–44.
[9] W. Chen, Y. Ju, C. Zhang, Parallel-in-time-space Chebyshev pseudospectral method for unsteady fluid flows (2021). https://www.researchgate.net/publication/350049339.
[10] R. Croce, D. Ruprecht, R. Krause, Parallel-in-space-and-time simulation of the three-dimensional, unsteady Navier–Stokes equations for incompressible flow, in: Modeling, Simulation and Optimization of Complex Processes-HPSC 2012, Springer, 2014, pp. 13–23.
[11] M.J. Gander, 50 years of time parallel time integration, in: Multiple Shooting and Time Domain Decomposition Methods, in: Contributions in Mathematical and Computational Sciences, volume 9, Springer, Cham, 2015, pp. 69–113.
[12] B.W. Ong, J.B. Schroder, Applications of time parallelization, Comput Vis Sci 23 (1–4) (2020) 11.
[13] Parallel-in-Time, https://parallel-in-time.org.
[14] C.W. Gear, Parallel methods for ordinary differential equations, Calcolo 25 (1) (1988) 1–20.
[15] W.L. Miranker, W. Liniger, Parallel methods for the numerical integration of ordinary differential equations, Math Comput 21 (1967) 303–320.
[16] A.J. Christlieb, C.B. Macdonald, B.W. Ong, Parallel high-order integrators, SIAM Journal on Scientific Computing 32 (2) (2010) 818–835.
[17] A. Christlieb, B. Ong, Implicit parallel time integrators, J Sci Comput 49 (2) (2011) 167–179.
[18] A.J. Christlieb, C.B. Macdonald, B.W. Ong, R.J. Spiteri, Revisionist integral deferred correction with adaptive step-size control, Comm App Math Comp Sci 10 (1) (2015) 1–25.
[19] J. Schütz, D.C. Seal, A. Jaust, Implicit multiderivative collocation solvers for linear partial differential equations with discontinuous Galerkin spatial discretizations, J Sci Comput 73 (2017) 1145–1163.
[20] D.A. Kopriva, Implementing Spectral Methods for Partial Differential Equations: Algorithms for Scientists and Engineers, Springer Science & Business Media, 2009.
[21] F. Hindenlang, G. Gassner, C. Altmann, A. Beck, M. Staudenmaier, C.-D. Munz, Explicit discontinuous Galerkin methods for unsteady problems, Computers & Fluids 61 (2012) 86–93.
[22] F. Bassi, S. Rebay, G. Mariotti, S. Pedinotti, M. Savini, A high-order accurate discontinuous finite element method for inviscid and viscous turbomachinery flows, Proceedings of 2nd European Conference on Turbomachinery, Fluid Dynamics and Thermodynamics (1997) 99–108.
[23] V. Linders, P. Birken, Locally conservative and flux consistent iterative methods (2022) arXiv preprint arXiv:2206.10943.
[24] A. Kværnø, Singly diagonally implicit Runge-Kutta methods with an explicit first stage, BIT Numerical Mathematics 44 (3) (2004) 489–502.
[25] C.A. Kennedy, M.H. Carpenter, Diagonally implicit Runge–Kutta methods for ordinary differential equations. A review, Technical Report, NASA Langley Research Center, 2016.
[26] M. Carpenter, C. Kennedy, Fourth-order 2*N*-storage Runge-Kutta schemes, Technical Report, NASA Langley Research Center, 1994.
[27] M. Franciolini, A. Crivellini, A. Nigro, On the efficiency of a matrix-free linearly implicit time integration strategy for high-order discontinuous galerkin solutions of incompressible turbulent flows, Computers & Fluids 159 (2017) 276–294.
[28] Y. Pan, Z.-G. Yan, J. Peiró, S. Sherwin, Development of a balanced adaptive time-stepping strategy based on an implicit JFNK-DG compressible flow solver, Communications on Applied Mathematics and Computation 4 (2022) 728–757.
[29] M. Han Veiga, P. Öffner, D. Torlo, Dec and ADER: similarities, differences and a unified framework, J Sci Comput 87 (1) (2021). Paper No. 2, 35
[30] V. Dolejší, M. Holík, J. Hozman, Efficient solution strategy for the semi-implicit discontinuous Galerkin discretization of the Navier-Stokes equations, J Comput Phys 230 (11) (2011) 4176–4200.
[31] D. Blom, P. Birken, H. Bijl, F. Kessels, A. Meister, A. van Zuilen, A comparison of rosenbrock and ESDIRK methods combined with iterative solvers for unsteady compressible flows, Adv Comput Math 42 (2016) 1401–1426.
[32] P. Birken, Numerical methods for unsteady compressible flow problems, Numerical Analysis and Scientific Computing, Chapman & Hall, 2021.
[33] G. Noventa, F. Massa, F. Bassi, A. Colombo, N. Franchina, A. Ghidoni, A high-order discontinuous Galerkin solver for unsteady incompressible turbulent flows, Computers & Fluids 139 (2016) 248–260.
[34] S.C. Eisenstat, H.F. Walker, Choosing the forcing terms in an inexact newton method, SIAM Journal on Scientific Computing 17 (1) (1996) 16–32.
[35] N. Krais, A. Beck, T. Bolemann, H. Frank, D. Flad, G. Gassner, F. Hindenlang, M. Hoffmann, T. Kuhn, M. Sonntag, et al., FLEXI: A high order discontinuous Galerkin framework for hyperbolic–parabolic conservation laws, Computers & Mathematics with Applications 81 (2021) 186–219.
[36] S. Götschel, M. Minion, D. Ruprecht, R. Speck, Twelve ways to fool the masses when giving parallel-in-time results, in: Workshops on Parallel-in-Time Integration, Springer, 2020, pp. 81–94.
[37] W. Chen, Y. Ju, C. Zhang, A parallel inverted dual time stepping method for unsteady incompressible fluid flow and heat transfer problems, Comput Phys Commun 260 (2021) 107325.
[38] N. Margenberg, T. Richter, Parallel time-stepping for fluid–structure interactions, Math Model Nat Phenom 16 (2021) 20.
[39] C.A. Kennedy, M.H. Carpenter, Additive Runge-Kutta schemes for convection-diffusion-reaction equations, Appl Numer Math 44 (2003) 139–181.
[40] C.A. Kennedy, M.H. Carpenter, Diagonally implicit Runge-Kutta methods for stiff ODEs, Appl. Numer. Math. 146 (2019) 221–244.
[41] H. Bijl, M.H. Carpenter, V.N. Vatsa, C.A. Kennedy, Implicit time integration schemes for the unsteady compressible Navier-Stokes equations: laminar flow, J Comput Phys 179 (1) (2002) 313–329.

[42] A. Nigro, C. De Bartolo, F. Bassi, A. Ghidoni, Up to sixth-order accurate a-stable implicit schemes applied to the discontinuous galerkin discretized Navier-Stokes equations, J Comput Phys 276 (2014) 136–162.
[43] L. Qu, C. Norberg, L. Davidson, S.-H. Peng, F. Wang, Quantitative numerical analysis of flow past a circular cylinder at reynolds number between 50 and 200, J Fluids Struct 39 (2013) 347–370.
[44] C. Liang, S. Premasuthan, A. Jameson, High-order accurate simulation of low-mach laminar flow past two side-by-side cylinders using spectral difference method, Computers & Structures 87 (11–12) (2009) 812–827.
[45] J. Meneghini, F. Saltara, C. Siqueira, J. Ferrari Jr, Numerical simulation of flow interference between two circular cylinders in tandem and side-by-side arrangements, J Fluids Struct 15 (2) (2001) 327–350.
[46] J. Zeifang, J. Schütz, K. Kaiser, A. Beck, M. Lukáčová-Medvid'ová, S. Noelle, A novel full-Eule low mach number IMEX splitting, Commun Comput Phys 27 (2020) 292–320.
[47] M.E. Brachet, D.I. Meiron, S.A. Orszag, B. Nickel, R.H. Morf, U. Frisch, Small-scale structure of the Taylor-Green vortex, J Fluid Mech 130 (1983) 411–452.
[48] E. Theodosiou, J. Schütz, D. Seal, An explicitness-preserving IMEX-split multiderivative method, UHasselt CMAT Preprint UP2301 (2023). https://www.uhasselt.be/media/zvvbnhh0/up2301.pdf.
[49] J. Chouchoulis, J. Schütz, J. Zeifang, Jacobian-free explicit multiderivative Runge-Kutta methods for hyperbolic conservation laws, J Sci Comput 90 (96) (2022).

Paper II:

Multi-step Hermite-Birkhoff predictor-corrector schemes

Appl. Numer. Math. 205 (2024) 281–295

Contents lists available at ScienceDirect

Applied Numerical Mathematics

journal homepage: www.elsevier.com/locate/apnum

Research Paper

Multi-step Hermite-Birkhoff predictor-corrector schemes

Arjun Thenery Manikantan *, Jochen Schütz

Faculty of Sciences & Data Science Institute, Hasselt University, Agoralaan Gebouw D, Diepenbeek, 3590, Belgium

ARTICLE INFO

MSC:
65L04
65L05
65L20
65M22

Keywords:
Multi-derivative
Multi-step
Time integration
Predictor-corrector
IVPs

ABSTRACT

In this study, we introduce a multi-step multi-derivative predictor-corrector time integration scheme analogous to the schemes in Schütz et al. (2022) [13], incorporating a multi-step quadrature rule. We conduct stability analysis up to order eight and optimize the schemes to achieve $A(\alpha)$-stability for large α. Numerical experiments are performed on ordinary differential equations exhibiting diverse stiffness conditions, as well as on partial differential equations showcasing non-linearity and higher-order terms. Results demonstrate the convergence and flexibility of the proposed schemes across diverse situations.

1. Introduction

In this work, we consider numerical solutions of initial value problems of form

$$y'(t) = \phi(y), \quad t \in (0, T_{end}),$$
$$y(0) = y_0. \tag{1}$$

There are plenty of schemes in literature to find the numerical solution of the above system (1), which can be broadly categorized into explicit and implicit schemes. Implicit schemes are chosen over explicit schemes for better stability and less severe timestep restriction properties, especially when it comes to solving stiff problems. Implicit schemes, while advantageous in certain aspects, come with drawbacks. These include the need to solve nonlinear equations, potentially leading to added errors in the solution. Moreover, implicit schemes often demand longer execution times and encounter additional bottlenecks when solving larger systems.

In classical implicit time integration methods found in literature, the numerical scheme mostly relies on the first temporal derivative (y') of the problem. For one-derivative schemes, the orders of consistency can only be increased by incorporating additional stages or steps to the numerical scheme. At the same time, the addition of intermediate stages ends up with an arduous task of solving involved order conditions. In order to achieve higher-order schemes with fewer stages, one can include higher-order derivatives of the problem (1) to the scheme, which results in multi-derivative time stepping methods [1–11]. In this paper, we focus on two-derivative schemes, which necessitate the computation of the second-order derivative of the problem (1),

$$y''(t) = \phi'(y)y'(t) = \phi'(y)\phi(y) =: \dot{\phi}(y).$$

* Corresponding author.
E-mail addresses: arjun.thenerymanikantan@uhasselt.be (A. Thenery Manikantan), jochen.schuetz@uhasselt.be (J. Schütz).

https://doi.org/10.1016/j.apnum.2024.07.011
Received 8 March 2024; Received in revised form 5 July 2024; Accepted 23 July 2024

Available online 31 July 2024

In [12], the authors have introduced a fourth-order two-derivative asymptotic preserving IMEX time stepping scheme for solving stiff ODEs, which was structured in a predictor-corrector fashion. Later in [13], the schemes were extended to higher-orders with parallel-in-time property, termed as **H**ermite-**B**irkhoff **P**redictor-**C**orrector (HBPC(q, $k_{\max}$)) schemes, where q is the maximum achievable order and $k_{\max}$ is the number of correction steps used. The shemes in [13] were optimized in [14] to have $A(\alpha)$ stability with α close to 90^0. The optimized HBPC schemes were applied to the discontinuous Galerkin method in [15,16]. Explicitness-preserving IMEX-HBPC(q, $k_{\max}$) schemes were presented in [17] recently.

In order to achieve higher-orders, authors have used a higher-order quadrature rule in [13], which utilizes the intermediate stage values, see [13, Eq. (8)]. Generally, methods involving more intermediate implicit stages require longer execution time. Schemes with extended computational time pose a drawback when implemented for expansive systems, such as the Euler equations, Navier-Stokes equations, and more. One strategy for addressing this situation could involve investigating higher-order schemes that demand fewer intermediate implicit stages. Hence, an evident direction is to explore schemes involving multiple steps. Several multi-step multi-derivative schemes have been documented in literature, including Brown's schemes [18,19], second derivative BDF schemes [20], strong stability preserving schemes [21,11], schemes for chemical stiff equations [22], general linear methods [23,24] and more.

To integrate the concept of a multi-step method for HBPC(q, $k_{\max}$) schemes, we explore a higher-order quadrature rule that leverages the previously computed solutions rather than relying solely on intermediate stages. This results in ***m*-Step Hermite-Birkhoff Predictor-Corrector** schemes abbreviated as mS-HBPC(q, $k_{\max}$). The parameter m represents the number of steps used from the previous time instances, q is the maximum achievable order of convergence, and $k_{\max}$ is the maximum number of correction steps. The 1S-HBPC(4, $k_{\max}$) is none other than the serial HBPC(4, $k_{\max}$) scheme in [13]. In this paper we analyze the stability properties and convergence of the mS-HBPC(q, $k_{\max}$) schemes for orders up to eight (up to three-step schemes).

The paper is structured as follows: In Sec. 2, the algorithm for the mS-HBPC(q, $k_{\max}$) schemes is presented, followed by the study of their stability properties in Sec. 3. The optimization procedures for the mS-HBPC(q, $k_{\max}$) schemes for $A(\alpha)$-stability are discussed in further subsections of Sec. 3. Numerical results of the schemes on various test-cases that includes ordinary and partial differential equations, are shown in Sec. 4. Finally, the paper is concluded and an outlook is given in Sec. 5.

2. Numerical scheme

We consider a fixed timestep for the scheme throughout the paper. For a given number of total timesteps N, we have

$$\Delta t := \frac{T_{end}}{N}.$$

The approximate solution at time instance

$$t^n := n\Delta t, \quad 0 \le n \le N,$$

is denoted as $y^n \approx y(t^n)$. The mS-HBPC(q, $k_{\max}$) scheme and the HBPC(q, $k_{\max}$) schemes from [13] utilize an approach similar to the spectrally deferred correction (SDC) method [25–27]. These schemes are designed to initiate with a predicted solution, represented by $y^{[0],n}$ at t^n. Then the predicted solution is subsequently refined through a sequence of correction steps. The k^{th} corrected solution at t^n is denoted as $y^{[k],n}$. The notations ϕ^n, $\dot{\phi}^n$, $\phi^{[k],n}$ and $\dot{\phi}^{[k],n}$ correspond to the function evaluations $\phi(y^n)$, $\dot{\phi}(y^n)$, $\phi(y^{[k],n})$ and $\dot{\phi}(y^{[k],n})$. At time t^{n+1}, the solution is updated with the highest corrected solution $y^{[k_{\max}],n}$. The detailed algorithm is defined below:

Algorithm 1 *(mS-HBPC(q, $k_{\max}$)).* Given the solutions $y^n, y^{n-1}, y^{n-2}, \dots, y^{n+1-m}$, the updated solution at t^{n+1} for $n \ge m-1$, is computed using the following prediction and correction steps. The predicted solution is computed using an implicit second-order Taylor scheme.

1. **Predict.** Solve the following expression for $y^{[0],n}$:

$$y^{[0],n} := y^n + \Delta t \phi^{[0],n} - \frac{\Delta t^2}{2}\dot{\phi}^{[0],n} \tag{2}$$

2. **Correct.** Solve the following expression for $y^{[k+1],n}$, for each $0 \le k < k_{\max}$:

$$\begin{aligned} y^{[k+1],n} := y^n &+ \Delta t\theta_1\left(\phi^{[k+1],n} - \phi^{[k],n}\right) \\ &- \frac{\Delta t^2}{2}\theta_2\left(\dot{\phi}^{[k+1],n} - \dot{\phi}^{[k],n}\right) + \mathcal{I}\left(\phi^{n+1-m}, \dots, \phi^{n-1}, \phi^n, \phi^{[k],n}\right), \end{aligned} \tag{3}$$

   where the q^{th} order quadrature rule (see Table 1) is given by

$$\mathcal{I}\left(\psi_1, \psi_2 \dots, \psi_m, \psi_{m+1}\right) := \Delta t \sum_{i=1}^{m+1} b_i^{(1)}\psi_i + \Delta t^2 \sum_{i=1}^{m+1} b_i^{(2)}\dot{\psi}_i.$$

   Note that θ_1 and θ_2 are constants; more details are given in Remark 4.
3. **Update.** Setting the solution at the final corrected step as the updated solution:

$$y^{n+1} := y^{[k_{\max}],n}.$$

Table 1
The quadrature rules for the schemes up to order eight are given in the table. The methodology for constructing these quadrature rules is outlined in Remark 2. The $2(m+1)$-order quadrature rule for approximating $\int_{t^n}^{t^{n+1}} \psi(y(t))dt$ is given by $\mathcal{I}\left(\psi_{n+1-m},\dots,\psi_n,\psi_{n+1}\right) := \Delta t \sum_{i=1}^{m+1} b_i^{(1)} \psi_{n+i-m} + \Delta t^2 \sum_{i=1}^{m+1} b_i^{(2)} \dot{\psi}_{n+i-m}$, where Δt is the mesh size of a given set of equally spaced time nodes.

Scheme	Quadrature rule							
	$b^{(1)}$				$b^{(2)}$			
1S-HBPC(4, $k_{\max}$) [12]	$\frac{1}{2}$	$\frac{1}{2}$			$\frac{1}{12}$	$-\frac{1}{12}$		
2S-HBPC(6, $k_{\max}$)	$\frac{11}{240}$	$\frac{128}{240}$	$\frac{101}{240}$		$\frac{3}{240}$	$\frac{40}{240}$	$-\frac{13}{240}$	
3S-HBPC(8, $k_{\max}$)	$\frac{1985}{90720}$	$\frac{12015}{90720}$	$\frac{42255}{90720}$	$\frac{34465}{90720}$	$\frac{489}{90720}$	$\frac{7263}{90720}$	$\frac{22977}{90720}$	$-\frac{3849}{90720}$

Remark 1. In order to start up the procedure, the mS-HBPC(q, $k_{\max}$) scheme requires solutions at the first m time instances,

$$y^{\ell} := y(t^{\ell}), \quad 0 \le \ell \le m-1,$$

which are either given or found using an appropriate solver. For the numerical results in this paper, we use the explicit solutions of the test problems, if available, for the start-up procedure. In cases where explicit solutions are inaccessible, we employ the MATLAB solver `ode15s` [28] to compute a highly accurate solution.

Remark 2. The quadrature rule for the mS-HBPC(q, $k_{\max}$) scheme is derived by fitting a Hermite–Birkhoff polynomial interpolant through the uniformly spaced time instances $t^{n+1}, t^n, t^{n-1}, \dots, t^{n+1-m}$ and integrating the result from t^n to t^{n+1}. This construction results into a quadrature rule,

$$\int_{t^n}^{t^{n+1}} \psi(y(t))dt = \mathcal{I}\left(\psi(y(t^{n+1-m}),\dots,\psi(y(t^{n-1}),\psi(y(t^n),\psi(y(t^{n+1}))\right) + \mathcal{O}(\Delta t^{q+1}).$$

The $m+1$ points give the Hermite–Birkhoff polynomial of order $2m+1$, which gives an integral approximation of order $(2m+1)+2 =: q+1$. Hence, we have

$$q = 2(m+1).$$

The quadrature rules for the schemes up to order eight are given in Table 1. Studies on the stability and convergence of variable timestep size variable formula multi-step schemes are also available in the literature; see [29–31] and the references therein. The mS-HBPC(q, $k_{\max}$) schemes can also be extended to variable timestep size schemes. In such cases, the quadrature rule will have variable coefficients that depend on the previous timestep sizes. In this paper, we analyze only the convergence and stability of the mS-HBPC(q, $k_{\max}$) schemes with fixed timestep.

Remark 3. Similar to the results from [12,13], in every iteration, for each correction step $k \le k_{\max}$, the mS-HBPC(q, $k_{\max}$) provides an approximation to $y(t^{n+1})$ with an order of accuracy of $\min\{q, 2+k\}$, i.e.

$$y(t^{n+1}) = y^{[k],n} + \mathcal{O}(\Delta t^{\min\{q,2+k\}}).$$

Convergence results are shown numerically in Sec. 4.1. Moreover, the choice of a different predictor in Algorithm 1 can alter both its order and stability. Specifically, if the predictor is of order p, it results in a new scheme with an overall accuracy order of $\min\left\{q, p+k_{\max}\right\}$. In this paper, we stick to the implicit second-order Taylor predictor ($p=2$) due to its suitability for extending the scheme to an implicit-explicit version.

Remark 4. The parameters $(\theta_1,\theta_2) \in \mathbb{R}^2$ utilized in the correction step (3) are the tuning parameters aimed at optimizing the algorithm to ensure better stability properties. Sec. 3 of the paper delves into the analysis of stability and the optimization of parameters (θ_1,θ_2).

3. Stability analysis

In this section, the linear stability of the mS-HBPC(q, $k_{\max}$) method is analyzed using Dahlquist's equation $y' = \lambda y$, where $\lambda \in \mathbb{C}$. Define $z := \Delta t \lambda$ and the following functions:

$$\mathcal{R}_m^{[0]}(z) := \frac{2}{2-2z+z^2} \quad \text{and} \quad \mathcal{R}_i^{[0]}(z) := 0,\ \forall i < m,$$

$$\mathcal{P}_m(z) := 1 + b_m^{(1)} z + b_m^{(2)} z^2 \quad \text{and} \quad \mathcal{P}_i(z) := b_i^{(1)} z + b_i^{(2)} z^2,\ \forall i < m,$$

$$\mathcal{S}(z) := \{b_{m+1}^{(1)} - \theta_1\} z + \{b_{m+1}^{(2)} + \frac{\theta_2}{2}\} z^2 \quad \text{and} \quad \mathcal{T}(z) := 1 - \theta_1 z + \frac{\theta_2}{2} z^2.$$

For a k^{th} correction step, define the functions $\mathcal{R}_{\ell}^{[k]}$ for $1 \le \ell \le m$ such that

$$y^{[k]} =: \sum_{i=0}^{m-1} \mathcal{R}_{m-i}^{[k]}(z)\, y^{n-i}.$$

Apply the mS-HBPC(q, $k_{\max}$) scheme as defined in Algorithm 1 for the Dahlquist's equation $y' = \lambda y$. The predicted solution can be written as

$$y^{[0]} = \sum_{i=0}^{m-1} \mathcal{R}_{m-i}^{[0]}(z)\, y^{n-i} = \mathcal{R}_{m}^{[0]}(z) y^{n}.$$

Then for the k^{th} correction step, we have

$$\begin{aligned} y^{[k]} &= \frac{\mathcal{S}(z) y^{[k-1]} + \sum_{i=0}^{m-1} \mathcal{P}_{m-i}(z)\, y^{n-i}}{\mathcal{T}(z)} = \frac{\mathcal{S}(z) \sum_{i=0}^{m-1} \mathcal{R}_{m-i}^{[k-1]}(z)\, y^{n-i} + \sum_{i=0}^{m-1} \mathcal{P}_{m-i}(z)\, y^{n-i}}{\mathcal{T}(z)} \\ &= \sum_{i=0}^{m-1} \frac{\mathcal{S}(z) \mathcal{R}_{m-i}^{[k-1]}(z) + \mathcal{P}_{m-i}(z)}{\mathcal{T}(z)}\, y^{n-i} =: \sum_{i=0}^{m-1} \mathcal{R}_{m-i}^{[k]}(z)\, y^{n-i}. \end{aligned}$$

Hence the functions $\mathcal{R}_{\ell}^{[k]}$ are recursively given by

$$\mathcal{R}_{\ell}^{[k]}(z) = \frac{\mathcal{S}(z) \mathcal{R}_{\ell}^{[k-1]}(z) + \mathcal{P}_{\ell}(z)}{\mathcal{T}(z)} \quad \text{for } 1 \le \ell \le m.$$

Therefore the updated solution at t^{n+1} can be obtained as

$$\begin{aligned} y^{n+1} = y^{[k_{\max}]} &= \sum_{i=0}^{m-1} \mathcal{R}_{m-i}^{[k_{\max}]}(z)\, y^{n-i} \\ &=: \sum_{i=0}^{m-1} \mathcal{R}_{m-i}(z)\, y^{n-i}. \end{aligned} \tag{4}$$

Assume a solution of type $y^s := r^s$ to the difference equation Eq. (4) and substituting it in Eq. (4) leads to the following polynomial of degree m,

$$r^m - \mathcal{R}_m(z)\, r^{m-1} - \mathcal{R}_{m-1}(z)\, r^{m-2} \cdots - \mathcal{R}_2(z)\, r - \mathcal{R}_1(z) = 0. \tag{5}$$

If $r_i(z)$ are roots of the polynomial (5), then the stability of the scheme mandates each root $r_i(z)$ to have an absolute value less than or equal to one.

Theorem 1. *The multi-step HBPC schemes mS-HBPC(q, $k_{\max}$) are zero-stable for any values of (θ_1, θ_2).*

Proof. Substituting $\lambda = 0$ into the defined functions yields,

$$\begin{aligned} \mathcal{R}_m^{[0]}(z) = 1 \quad &\text{and} \quad \mathcal{R}_i^{[0]}(z) = 0,\ \forall i < m, \\ \mathcal{P}_m(z) = 1 \quad &\text{and} \quad \mathcal{P}_i(z) = 0,\ \forall i < m, \\ \mathcal{S}(z) = 0 \quad &\text{and} \quad \mathcal{T}(z) = 1, \end{aligned}$$

which are independent of (θ_1, θ_2) values. Therefore, we get

$$\mathcal{R}_{\ell}^{[k]}(z) = \begin{cases} 1, & \ell = m, \\ 0, & \ell \ne m, \end{cases}$$

for every k. Then the polynomial (5) is

$$r^m - r^{m-1} = 0,$$

with roots $r_i \in \{0, 1\}$, and hence their magnitude $|r_i| \le 1$. Being the multiplicity of the root $r_m = 1$ equal to one completes the proof. □

Define the function

$$\mathcal{R}_{max}(z) := \max_i \left\{ |r_i(z)| \right\}. \tag{6}$$

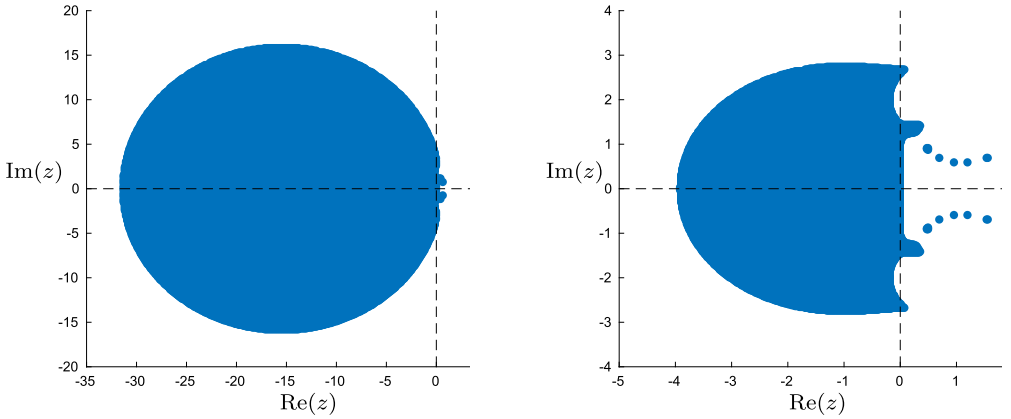


Fig. 1. Stability regions of 2S-HBPC(6,4) (left) and 3S-HBPC(8,6) (right) schemes with $(\theta_1, \theta_2) = (1, 1)$.

Then the stability region for the numerical scheme is given by

$$\text{Stability Region} = \left\{ z \in \mathbb{C} \,|\mathcal{R}_{max}(z) \leq 1 \right\}, \tag{7}$$

under the assumption that the polynomial (5) has only single roots. Since the scheme 1S-HBPC(4, $k_{\max}$) is the same as scheme HBPC(4, $k_{\max}$) in [12], it has been shown already in [14] that the scheme is $A(\alpha)$-stable, and A-stable with optimized coefficients $(\theta_1, \theta_2) = \left(\frac{1}{2}, \frac{1}{6}\right)$.

The plots in Fig. 1 show the stability regions of the 2S-HBPC(6,4) (left) and 3S-HBPC(8,6) (right) schemes with values θ_1 and θ_2 equal to one. The shaded regions in Fig. 1 show that the stability regions are bounded. To harness the advantage of an unconstrained timestep condition offered by an implicit scheme, it is crucial to possess a stability region that extends infinitely, contrasting with the finite boundaries illustrated in Fig. 1. Hence, the tuning parameters θ_1 and θ_2 are to be brought into service for expanding the stability region. In the following section, the tuning parameters are optimized to obtain an $A(\alpha)$-stable mS-HBPC(q, $k_{\max}$) scheme. In this paper, we specifically limit the optimization process to schemes 2S-HBPC(6,4) and 3S-HBPC(8,6).

3.1. Constraints on tuning parameters for $A(\alpha)$-stability

Lemmas 1 and 2 show the existence of positive values (θ_1, θ_2) ensuring that the multi-step HBPC schemes 2S-HBPC(6,4) and 3S-HBPC(8,6) are $A(\alpha)$-stable. The conditions on the tuning parameters are determined by examining the roots of the difference equation (4). The potential candidates are the tuning parameters (θ_1, θ_2), which ensure that the roots of the difference equation (4) have an absolute value less than one as z approaches $-\infty$.

Remark 5. We use the Symbolic Math Toolbox in MATLAB [32] for solving/simplifying/deriving certain equations/expressions in the paper due to their complexity. In particular,

- to find the roots, and their limits of the polynomials (8) and (9), in the proofs of Lemma 1 and Lemma 2, respectively;
- to find the error constant $C(\theta_1, \theta_2)$ defined in Sec. 3.2.1 and Sec. 3.2.2.

Remark 6 *(Methodology for evaluating stability angle).* We utilize the algorithm outlined in [14, Sec. 3] to compute the stability angles for A(α)-stable mS-HBPC(q, $k_{\max}$) schemes.

Lemma 1. *The 2S-HBPC(6,4) scheme is $A(\alpha)$-stable for any $\theta_1 > 0$ and $\theta_2 \geq 1.25868$. The minimum value of θ_2 provided is rounded to five decimal places.*

Proof. Begin with the polynomial equation in (5) for the 2S-HBPC(6,4) scheme

$$r^2 - \mathcal{R}_2(z)\, r - \mathcal{R}_1(z) = 0. \tag{8}$$

Its roots are given by

$$r_1(z) = \frac{\mathcal{R}_2(z)}{2} + \sqrt{\frac{\mathcal{R}_2^2(z)}{4} + \mathcal{R}_1(z)}, \quad \text{and} \quad r_2(z) = \frac{\mathcal{R}_2(z)}{2} - \sqrt{\frac{\mathcal{R}_2^2(z)}{4} + \mathcal{R}_1(z)}.$$

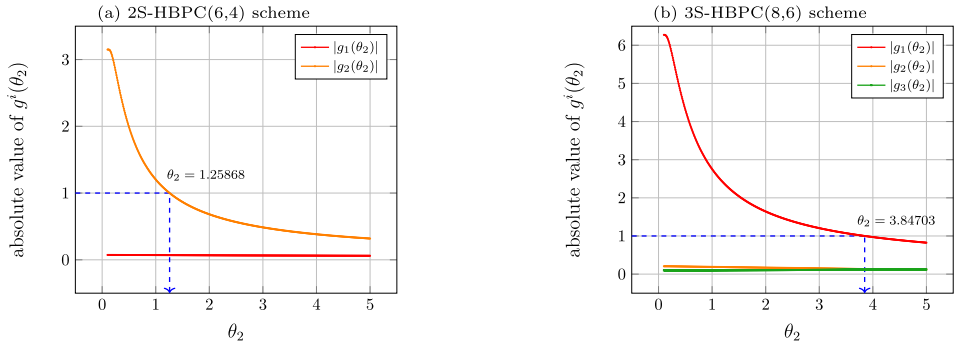


Fig. 2. Absolute values of the limits of the roots are plotted against different θ_2 values. The value of θ_2 for which $\max_i |g_i(\theta_2)| \leq 1$ is marked on the plot. This value of θ_2 ensures A(α)-stability for the scheme. (For interpretation of the colors in the figure(s), the reader is referred to the web version of this article.)

Table 2
A set of uniformly spaced values for the tuning parameters θ_1 and θ_2 has been chosen for computing the stability angles for the schemes 2S-HBPC(6,4) and 3S-HBPC(8,6), including the minimum required values for θ_2 found in Lemmas 1 and 2. The θ_1 and θ_2 values are represented in the leftmost column and topmost row of the respective tables. The stability angles (in degrees) for various (θ_1, θ_2) values are provided for $A(\alpha)$-stable 2S-HBPC(6,4) (left) and 3S-HBPC(8,6) (right) schemes in the left and right tables, respectively. These stability angles are computed using the algorithm described in Remark 6 and rounded to one decimal place. A trend of increase in stability angle when θ_2 is increased while keeping θ_1 fixed and conversely a decrease in stability angle when θ_1 is increased while keeping θ_2 fixed can be observed in the table above.

$\theta_1 \downarrow$ $\theta_2 \rightarrow$	1.25868	1.5	2	2.5	3	3.5	4	4.5	5
0.5	78.9	78.2	77.3	76.8	76.5	76.3	76.2	76.2	76.2
0.6	80.1	79.3	78.3	77.7	77.3	77.1	77.0	76.9	76.9
0.7	81.2	80.4	79.2	78.6	78.2	77.9	77.8	77.7	77.6
0.8	82.1	81.3	80.1	79.4	78.9	78.7	78.5	78.3	78.3
0.9	82.9	82.1	80.9	80.2	79.7	79.3	79.1	79.0	78.9
1.0	83.6	82.8	81.6	80.8	80.3	80.0	79.8	79.6	79.5
1.1	84.3	83.5	82.3	81.5	81.0	80.6	80.3	80.2	80.0
1.2	84.8	84.0	82.9	82.1	81.5	81.2	80.9	80.7	80.6
1.3	85.2	84.5	83.4	82.6	82.1	81.7	81.4	81.2	81.1
1.4	85.6	85.0	83.9	83.1	82.6	82.2	81.9	81.7	81.5
1.5	85.9	85.3	84.3	83.6	83.0	82.6	82.3	82.1	82.0
1.6	86.2	85.7	84.7	84.0	83.4	83.0	82.8	82.5	82.4
1.7	86.5	85.9	85.0	84.3	83.8	83.4	83.1	82.9	82.8
1.8	86.6	86.2	85.4	84.7	84.2	83.8	83.5	83.3	83.1
1.9	86.8	86.4	85.6	85.0	84.5	84.1	83.9	83.6	83.5
2.0	87.0	86.6	85.9	85.3	84.8	84.5	84.2	83.9	83.8
2.1	87.1	86.8	86.1	85.6	85.1	84.8	84.5	84.2	84.1
2.2	87.2	86.9	86.3	85.8	85.4	85.0	84.7	84.5	84.4
2.3	87.3	87.0	86.5	86.0	85.6	85.3	85.0	84.8	84.6
2.4	87.4	87.2	86.7	86.2	85.8	85.5	85.2	85.0	84.9
2.5	87.5	87.3	86.8	86.4	86.0	85.7	85.5	85.3	85.1
2.6	87.6	87.4	86.9	86.6	86.2	85.9	85.7	85.5	85.3
2.7	87.6	87.5	87.1	86.7	86.4	86.1	85.9	85.7	85.5
2.8	87.7	87.5	87.2	86.8	86.5	86.3	86.1	85.9	85.7
2.9	87.8	87.6	87.3	87.0	86.7	86.4	86.2	86.1	85.9
3.0	87.8	87.7	87.4	87.1	86.8	86.6	86.4	86.2	86.1

$\theta_1 \downarrow$ $\theta_2 \rightarrow$	3.84703	4.5	5	5.5	6	6.5	7	7.5	8
0.5	76.2	76.0	75.9	75.8	75.7	75.6	75.5	75.5	75.4
0.6	76.8	76.6	76.4	76.3	76.2	76.1	76.0	75.9	75.9
0.7	77.4	77.1	76.9	76.7	76.6	76.5	76.4	76.4	76.3
0.8	77.9	77.6	77.4	77.2	77.1	77.0	76.9	76.8	76.7
0.9	78.4	78.1	77.9	77.7	77.5	77.4	77.3	77.2	77.1
1.0	78.9	78.5	78.3	78.1	78.0	77.8	77.7	77.6	77.5
1.1	79.4	79.0	78.7	78.5	78.4	78.2	78.1	78.0	77.9
1.2	79.8	79.4	79.2	78.9	78.8	78.6	78.5	78.4	78.3
1.3	80.3	79.8	79.6	79.3	79.2	79.0	78.9	78.8	78.7
1.4	80.7	80.2	79.9	79.7	79.5	79.4	79.2	79.1	79.0
1.5	81.1	80.6	80.3	80.1	79.9	79.7	79.6	79.5	79.4
1.6	81.4	81.0	80.7	80.4	80.2	80.1	79.9	79.8	79.7
1.7	81.8	81.3	81.0	80.8	80.6	80.4	80.2	80.1	80.0
1.8	82.1	81.6	81.3	81.1	80.9	80.7	80.6	80.4	80.3
1.9	82.4	81.9	81.6	81.4	81.2	81.0	80.9	80.7	80.6
2.0	82.7	82.2	81.9	81.7	81.5	81.3	81.1	81.0	80.9
2.1	83.0	82.5	82.2	82.0	81.8	81.6	81.4	81.3	81.2
2.2	83.3	82.8	82.5	82.3	82.0	81.9	81.7	81.6	81.5
2.3	83.5	83.1	82.8	82.5	82.3	82.1	82.0	81.8	81.7
2.4	83.8	83.3	83.0	82.8	82.5	82.4	82.2	82.1	82.0
2.5	83.9	83.4	83.1	82.9	82.7	82.5	82.3	82.2	82.1
2.5	84.0	83.5	83.2	83.0	82.8	82.6	82.5	82.3	82.2
2.6	84.2	83.8	83.5	83.2	83.0	82.8	82.7	82.6	82.4
2.7	84.4	84.0	83.7	83.4	83.2	83.1	82.9	82.8	82.7
2.8	84.6	84.2	83.9	83.7	83.5	83.3	83.1	83.0	82.9
2.9	84.8	84.4	84.1	83.9	83.7	83.5	83.3	83.2	83.1
3.0	84.9	84.5	84.3	84.0	83.9	83.7	83.5	83.4	83.3

The limits of the roots $r_1(z)$ and $r_2(z)$ as z tends to $-\infty$ result into functions that depend on θ_2 only. It is because θ_2 occurs in the Algorithm 1 with terms involving Δt^2, whereas θ_1 pairs up with Δt terms. Therefore, we can write

$$\lim_{z\to-\infty} r_1(z) =: g_1(\theta_2),$$
$$\lim_{z\to-\infty} r_2(z) =: g_2(\theta_2).$$

As we need to expand the stability region, conditions on θ_2 can be found by analyzing the absolute values of $g_1(\theta_2)$ and $g_2(\theta_2)$.

In Fig. 2a it is shown that $|g_2(\theta_2)| < 1$ for all the given θ_2 values whereas $|g_1(\theta_2)| < 1$ only for values of $\theta_2 \gtrsim 1.25868$. Hence we have $\forall \theta_2 \geq 1.25868$

$$\lim_{z\to-\infty} \mathcal{R}_{max}(z) < 1.$$

The stability angles for various (θ_1, θ_2) values are given in Table 2 for A(α)-stable 2S-HBPC(6,4). As illustrated in Fig. 3 and Table 2, it is evident that the 2S-HBPC(6,4) scheme exhibits $A(\alpha)$-stability for θ_2 greater than or equal to 1.25868. The stability angle corresponding to the pair $(\theta_1, \theta_2) = (1, 1.25868)$ is 83.64°. □

Lemma 2. *The 3S-HBPC(8,6) scheme is $A(\alpha)$-stable for any $\theta_1 > 0$ and $\theta_2 \geq 3.84703$. The minimum value of θ_2 provided is rounded to five decimal places.*

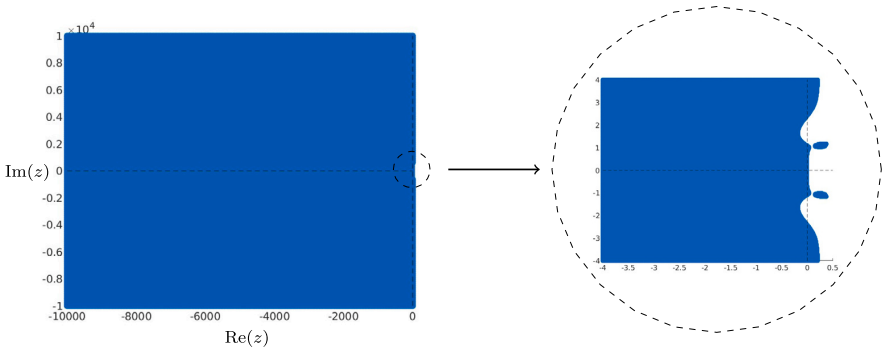


Fig. 3. Stability region of 2S-HBPC(6,4) scheme for $(\theta_1, \theta_2) = (1, 1.25868)$ with zoomed image on the right.

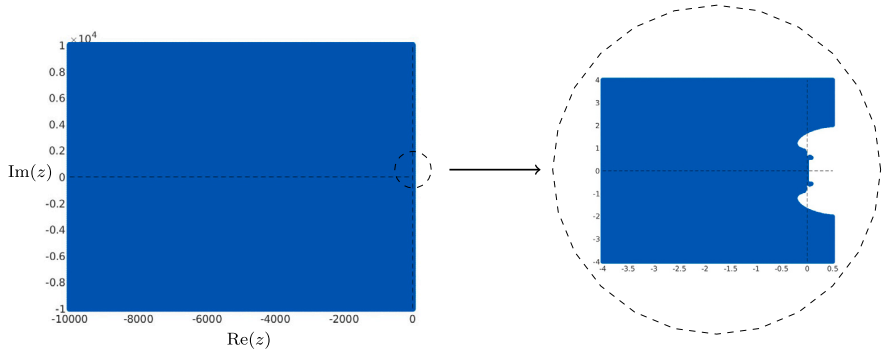


Fig. 4. Stability region of 3S-HBPC(8,6) scheme for $(\theta_1, \theta_2) = (1, 3.84703)$ with zoomed image on the right.

Proof. Begin with the polynomial equation in (5) for the 3S-HBPC(8,6) scheme

$$r^3 - \mathcal{R}_3(z)\, r^2 - \mathcal{R}_2(z)\, r - \mathcal{R}_1(z) = 0. \tag{9}$$

Here, we have a third degree polynomial and its roots can be found explicitly using Cardano's formula for cubic polynomials. The roots $r_1(z)$, $r_2(z)$ and $r_3(z)$ are not provided here due to their lengthy formulations. Similar to 2S-HBPC(6,4), the limits of the roots $r_1(z)$, $r_2(z)$ and $r_3(z)$ as z tends to $-\infty$ also result into functions that depend on θ_2 alone,

$$\lim_{z\to-\infty} r_k(z) =: g_k(\theta_2), \quad 1 \le k \le 3.$$

The explicit limits of the roots are omitted here due to the extended nature of the expressions.

In Fig. 2b, it is shown that $|g^2(\theta_2)| < 1$ and $|g^3(\theta_2)| < 1$ for all the given θ_2 values whereas $|g^1(\theta_2)| < 1$ only for values of $\theta_2 \gtrsim 3.84703$. Hence we have $\forall \theta_2 \ge 3.84703$

$$\lim_{z\to-\infty} \mathcal{R}_{max}(z) < 1.$$

The stability angles for various (θ_1, θ_2) values are given in Table 2 for A(α)-stable 3S-HBPC(8,6). From Fig. 4 and Table 2, it is evident that the 3S-HBPC(8,6) scheme exhibits $A(\alpha)$-stability for values of θ_2 greater than or equal to 3.84703. The stability angle corresponding to the pair $(\theta_1, \theta_2) = (1, 1.25868)$ is 78.93°. □

The detailed analysis on (θ_1, θ_2) values for 2S-HBPC(6,4) and 3S-HBPC(8,6) schemes are given in Sec. 3.2.

3.2. Optimization of the tuning parameters

When analyzing the stability angles in Table 2 for various (θ_1, θ_2) values with the minimum requirement on the θ_2 value for $A(\alpha)$-stability, it is observed that there is an increase in the stability angle as we increase θ_1 keeping θ_2 fixed, whereas a decrease in the stability angle as we increase θ_2 keeping θ_1 fixed. In this subsection, we seek optimized tuning parameters that can minimize the one-step error when using a linear test problem.

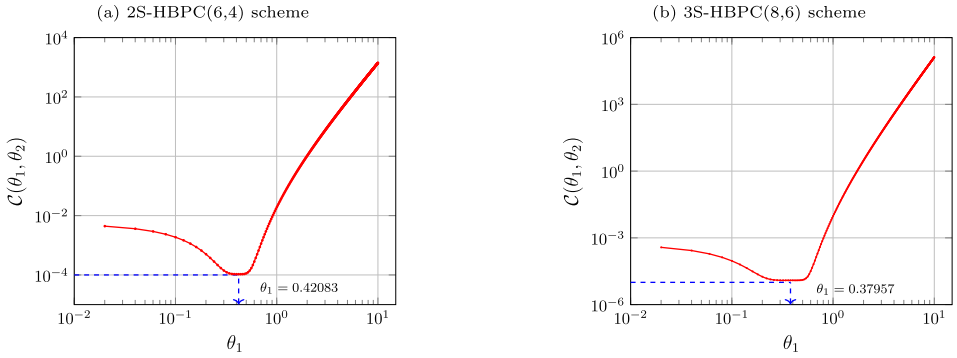


Fig. 5. Error constants for sixth and eighth-order schemes plotted for various θ_1 values. The value of θ_1, for which $C(\theta_1, \theta_2)$ is minimum, is marked on the plot.

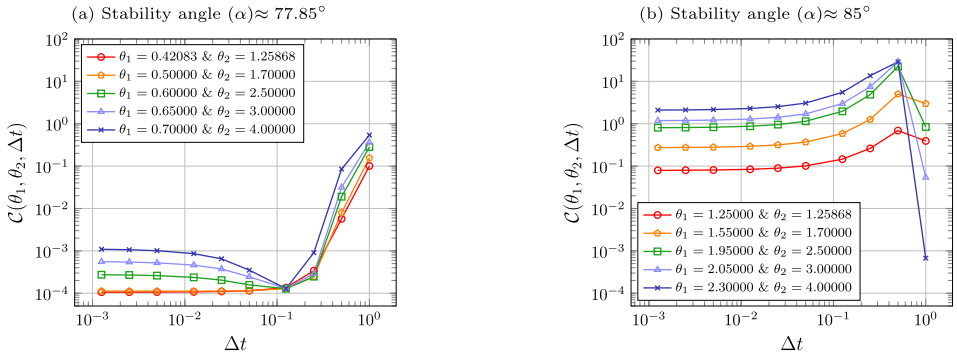


Fig. 6. The one-step error $C(\theta_1, \theta_2, \Delta t)$ for 2S-HBPC(6,4) scheme is plotted against different timesteps Δt.

Consider the test problem

$$y' = y, \quad y(0) = 1,$$

which admits an explicit solution $y(t) = e^t$. The one-step error $C(\theta_1, \theta_2, \Delta t)$ for an mS-HBPC(q, $k_{\max}$) scheme is defined as

$$C(\theta_1, \theta_2, \Delta t) := \left| \frac{e^{m\Delta t} - \left(\sum_{i=0}^{m-1} \mathcal{R}_{i+1}(\Delta t) e^{i\Delta t} \right)}{\Delta t^{q+1}} \right|. \tag{10}$$

3.2.1. 2S-HBPC(6,4) scheme

Begin with the error constant $C(\theta_1, \theta_2)$, which is obtained by considering $\lim_{\Delta t \to 0} C(\theta_1, \theta_2, \Delta t)$. The equation is given by

$$C(\theta_1, \theta_2) = \left| \frac{1}{6}\theta_1^4 - \frac{101}{360}\theta_1^3 + \frac{10201}{57600}\theta_1^2 - \frac{1030301}{20736000}\theta_1 + \frac{743168407}{139345920000} \right|. \tag{11}$$

It can be noted that the error constant is independent of θ_2. Therefore, it is plotted against various θ_1 values in Fig. 5a. It can be seen from Fig. 5a that the error constant reaches its minimum value at $\theta_1 \approx 0.42083$. Hence we consider only values of θ_1 starting from 0.42083 to obtain different stability angles.

Plots in Fig. 6 show the one-step error $C(\theta_1, \theta_2, \Delta t)$ versus timestep graph for various (θ_1, θ_2) pairs corresponding to stability angles approximately equal to 77.85° (left) and 85° (right) respectively. It can be observed from Fig. 6 that the one-step error $C(\theta_1, \theta_2, \Delta t)$ corresponding to a given stability angle gives a considerably minimal value for lowest (θ_1, θ_2) values. Hence, for an $A(\alpha)$-stable 2S-HBPC(6,4) scheme we fix θ_2 at the minimum value of 1.25868 (rounded to five decimal places). To enhance the optimization of the scheme, Fig. 8a illustrates the stability angles (α) across various θ_1 values. It can be seen from that Fig. 8a that the stability angle (α) increases with increased θ_1 values.

3.2.2. 3S-HBPC(8,6) scheme

Similar to the sixth-order scheme, the error constant $C(\theta_1, \theta_2)$ for 3S-HBPC(8,6) scheme is given by

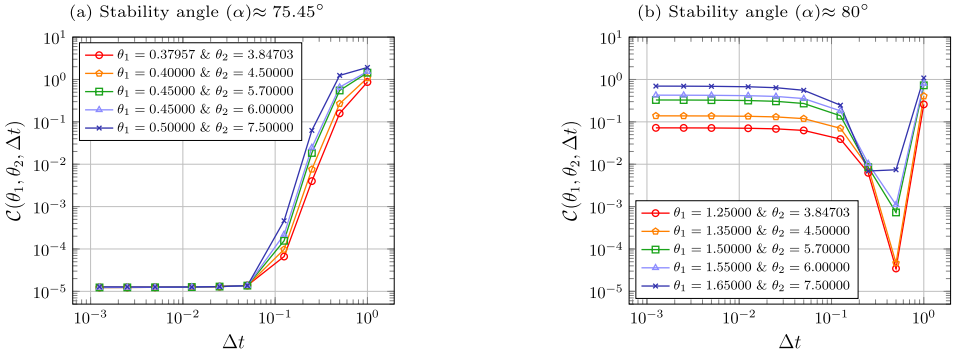


Fig. 7. The one-step error $C(\theta_1, \theta_2, \Delta t)$ for 3S-HBPC(8,6) scheme is plotted against different timesteps Δt.

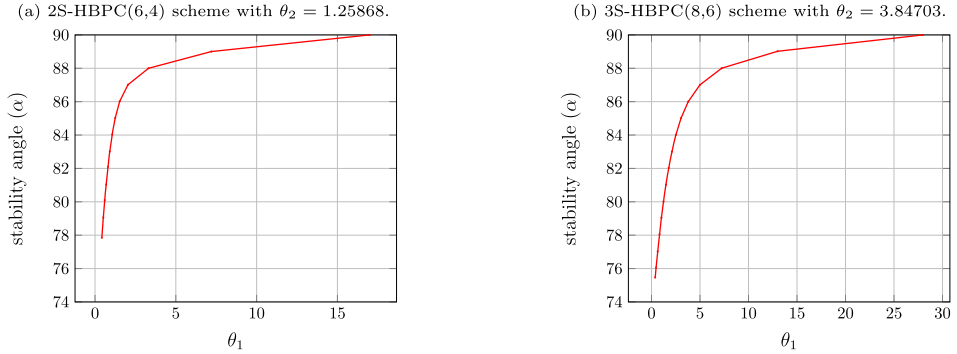


Fig. 8. Stability angles α for the $A(\alpha)$-stable sixth and eighth-order schemes plotted for varying θ_1 values.

$$C(\theta_1,\theta_2) = \left| \frac{1}{6}\theta_1^6 - \frac{6893}{18144}\theta_1^5 + \frac{237567245}{658409472}\theta_1^4 - \frac{1637551019785}{8959636094976}\theta_1^3 + \frac{11287639179378005}{216751516409659392}\theta_1^2 - \frac{155611393726905176 93}{196636975686843000 4224}\theta_1 + \frac{27475169567453025962 30737}{53516719302931190994960 38400} \right|. \quad (12)$$

It can be observed from Fig. 5b that the error constant reaches its minimum value at $\theta_1 \approx 0.37957$. Hence we consider only values of θ_1 starting from 0.37957 to obtain different stability angles.

Plots in Fig. 7 show the one-step error $C(\theta_1, \theta_2, \Delta t)$ versus timestep graph for various (θ_1, θ_2) pairs corresponds to stability angles approximately equal to 75.45° (left) and 80° (right) respectively. Likewise as in the sixth-order scheme, it can be noted from Fig. 7 that the one-step error $C(\theta_1, \theta_2, \Delta t)$ corresponding to a given stability angle gives a considerably minimal value for lowest (θ_1, θ_2) values. Therefore, to achieve an $A(\alpha)$-stable 2S-HBPC(6,4) scheme we fix θ_2 at the minimum value of 3.84703 (rounded to five decimal places). To enhance the optimization of the scheme, Fig. 8b illustrates the stability angles (α) across various θ_1 values. An increment in the stability angle (α) with increased θ_1 values can be observed from Fig. 8b.

3.3. Numerical stability results

We consider the Prothero–Robinson problem [33] to illustrate the effect of the improvement in the stability properties of the 2S-HBPC(6, 4) and 3S-HBPC(8, 6) schemes. It is given by

$$y'(t) = g'(t) + \lambda \{ y(t) - g(t) \}, \quad y(0) = y_0. \quad (13)$$

The exact solution for (13) is $y(t) = e^{\lambda t} \{ y_0 - g(t) \} + g(t)$. For the results, the function $g(t)$ is chosen to be $\cos(t)$, the initial solution is set to $y_0 = 0$, and $\lambda = -40$. The error is calculated at $T_{end} = 100$.

Definition 1. The **unoptimized** mS-HBPC(q, $k_{\max}$) schemes used in the subsequent sections refer to schemes with tuning parameters $(\theta_1, \theta_2) = (1, 1)$.

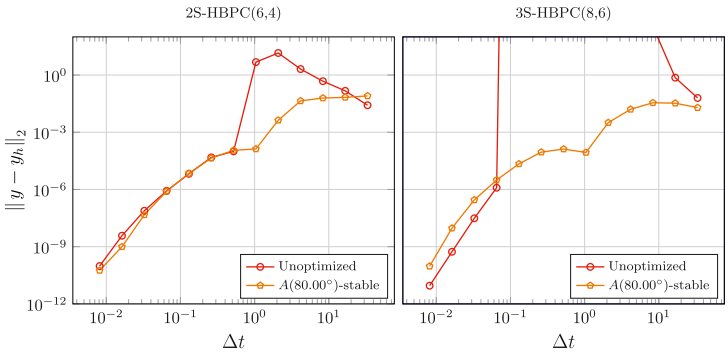


Fig. 9. Plots showcasing the effect of the improvement in the stability properties of the 2S-HBPC(6, 4) (left) and 3S-HBPC(8, 6) (right) schemes. The error is calculated at $T_{end} = 100$, and a large range of timesteps ($8 \times 10^{-3} < \Delta t < 34$) is chosen to illustrate the stability of both the unoptimized and optimized schemes for the Prothero–Robinson problem [33] given in (13). Consult Table 3 for (θ_1, θ_2) values corresponding to the given stability angles.

Table 3
Stability optimized tuning parameters for $A(\alpha)$-stable 2S-HBPC(6,4) and 3S-HBPC(8,6) schemes.

2S-HBPC(6,4) ($\theta_2 = 1.25868$)								
Approx. stability angle (α)	77.85°	80.00°	85.00°	86.00°	87.00°	88.00°	89.00°	89.99°
θ_1	0.42083	0.60000	1.25000	1.52500	2.03750	3.31250	7.2000	17.0000
Error constant	$1.05 \cdot 10^{-4}$	$2.77 \cdot 10^{-4}$	$7.88 \cdot 10^{-2}$	$2.47 \cdot 10^{-1}$	$1.13 \cdot 10^{0}$	$1.16 \cdot 10^{1}$	$3.52 \cdot 10^{2}$	$1.25 \cdot 10^{4}$
3S-HBPC(8,6) ($\theta_2 = 3.84703$)								
Approx. stability angle (α)	75.45°	80.00°	85.00°	86.00°	87.00°	88.00°	89.00°	89.99°
θ_1	0.37957	1.23750	3.05000	3.78750	4.98750	7.23750	13.0000	28.0000
Error constant	$1.23 \cdot 10^{-5}$	$6.63 \cdot 10^{-2}$	$6.03 \cdot 10^{1}$	$2.60 \cdot 10^{2}$	$1.59 \cdot 10^{3}$	$1.73 \cdot 10^{4}$	$6.73 \cdot 10^{5}$	$7.39 \cdot 10^{7}$

In Fig. 9, the Euclidean errors are plotted against a large range of timestep sizes ($8 \times 10^{-3} < \Delta t < 34$) for the schemes 2S-HBPC(6, 4) and 3S-HBPC(8, 6). It can be observed that the unoptimized schemes do not show convergence for relatively large timesteps, whereas the optimized schemes exhibit a reduction in error for the same chosen timesteps. Since the stability region for the unoptimized 3S-HBPC(8, 6) is smaller compared to the unoptimized 2S-HBPC(6, 4), the range of unstable timesteps is larger for the unoptimized 3S-HBPC(8, 6).

3.4. Optimized tuning parameters for numerical results

We found constraints on the tuning parameters in the preceding sections that minimize one-step errors (10) and the error constants (11) and (12) using a linear test problem. Consequently, the values obtained from these procedures are employed for the numerical results in the forthcoming section. Some of these values are presented in Table 3.

4. Numerical results

For the convergence analysis, we calculate the Euclidean error denoted as $\|y - y_h\|_2$, comparing the approximate solution y_h using the explicit solution $y(t)$ or a highly accurate solution found numerically. The error is calculated at a given final time T_{end}.

4.1. Convergence

We consider the non-stiff ODE

$$y'(t) = -y^{-\frac{5}{2}}, \quad y_0 = 1, \tag{14}$$

for showing the convergence of the mS-HBPC(q, $k_{\max}$) schemes. The explicit solution of the problem (14) is

$$y(t) = \left(y_0^{\frac{7}{2}} - \frac{7}{2} t \right)^{\frac{2}{7}}.$$

The error is calculated at $T_{end} = 0.25$.

In Fig. 10, the convergence results are shown for 2S-HBPC(6, 4) (top) and 3S-HBPC(8, 6) (bottom) for various correction steps k. The results are compared for unoptimized and $A(\alpha)$-stable mS-HBPC(q, $k_{\max}$) schemes. It can be seen from Fig. 10 that the unoptimized (first column) and $A(85^o)$-stable (third column), 2S-HBPC(6, 4) (top) and 3S-HBPC(8, 6) (bottom) schemes, clearly follows the trend of order increment by one after each correction step. This observation provides evidence in favor of the theoretical order of convergence

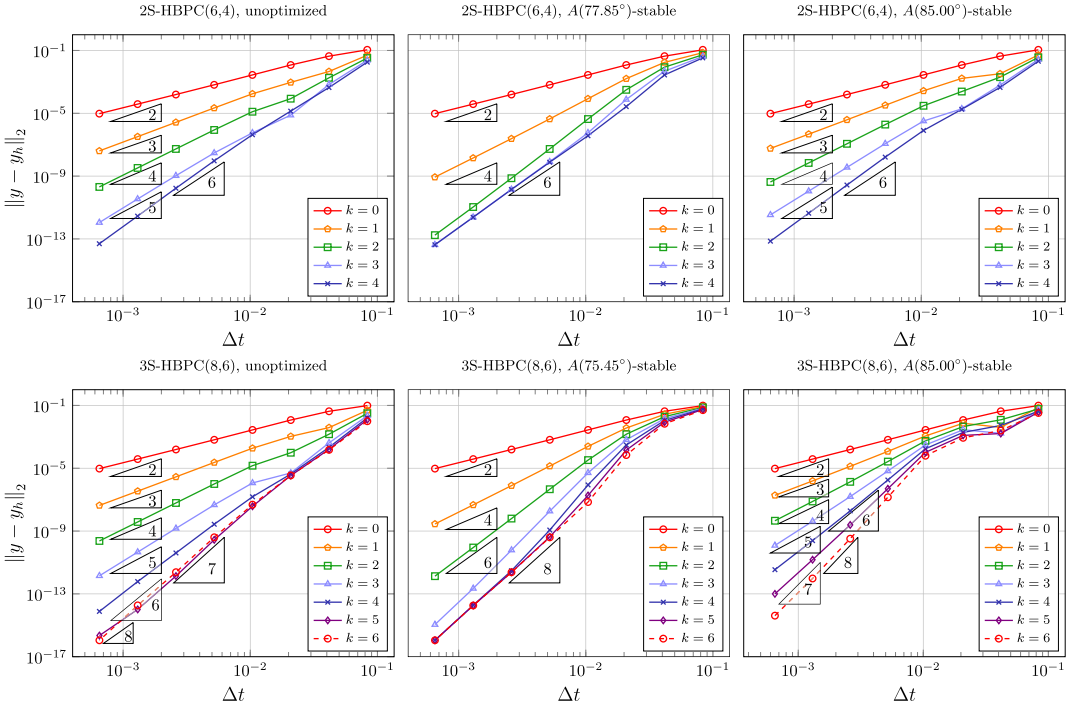


Fig. 10. Convergence plots of the 2S-HBPC(6,4) (top) and 3S-HBPC(8,6) (bottom) schemes for a simple ODE (14). The error is calculated at $T_{end} = 0.25$. Consult Table 3 for (θ_1, θ_2) values corresponding to the given stability angles.

of $\min\{q, 2+k\}$ for a k-th correction step. However, for $A(77.85°)$-stable 2S-HBPC(6, 4) (second column, top) and $A(75.45°)$-stable 3S-HBPC(8, 6) (second column, bottom) schemes, there is an order increment by two until the second and third correction steps, respectively. This exception arises from the dominance of $\mathcal{O}(\Delta t^2)$ terms over $\mathcal{O}(\Delta t)$ terms in the correction steps for parameters $\theta_1 \ll \theta_2$. Eventually, the schemes, 2S-HBPC(6, 4) and 3S-HBPC(8, 6) achieve their desired order of convergences six ($k_{\max} = 4$) and eight ($k_{\max} = 6$), respectively.

4.2. Pareschi–Russo problem

Here, we investigate the performance of mS-HBPC(q, $k_{\max}$) schemes on stiff ODEs. The model problem considered is a system of IVP given by

$$y_1'(t) = -y_2, \quad y_2'(t) = y_1 + \frac{\sin(y_1) - y_2}{\varepsilon}, \quad y_0 = \left(\frac{\pi}{2}, 1\right), \tag{15}$$

which was introduced in [34]. The error is calculated at $T_{end} = 5$, using a highly accurate solution found numerically. To investigate the performance of the scheme across non-stiff and stiff equations, we vary the stiffness parameter ε across different values, specifically $\varepsilon = 1$, $\varepsilon = 10^{-2}$, and $\varepsilon = 10^{-3}$.

In Fig. 11, the convergence plots for 2S-HBPC(6, 4) (top) and 3S-HBPC(8, 6) (bottom) with different stability angles are shown for various stiffness parameters. It can be seen that the unoptimized scheme becomes unstable as the problem becomes more stiff. For the non-stiff problem ($\varepsilon = 1$), all the schemes exhibit their expected convergence order except the schemes $A(77.85°)$-stable 2S-HBPC(6,4) and $A(75.45°)$-stable 3S-HBPC(8,6) that show some irregularities. This is a consequence of the additional decrease in their error constants, which are specifically connected to their (θ_1, θ_2) values. When stiffness parameter $\varepsilon = 10^{-2}$, the convergence of $A(\alpha)$-stable schemes with low stability angles becomes irregular when relatively large timesteps are employed to solve the problem. As the timesteps decrease, they eventually attain their desired convergence order. For extremely small stiffness parameters ($\varepsilon = 10^{-3}$), the inconsistencies in convergence order are noticeable over a wider range of time steps, particularly in schemes with lower stability angles. For all specified stiffness parameters, it is notable that the $A(\alpha)$-stable scheme exhibits smoother convergence and achieves the intended convergence order as the stability angles increase. However, this improvement comes at the cost of slight shifts in the error.

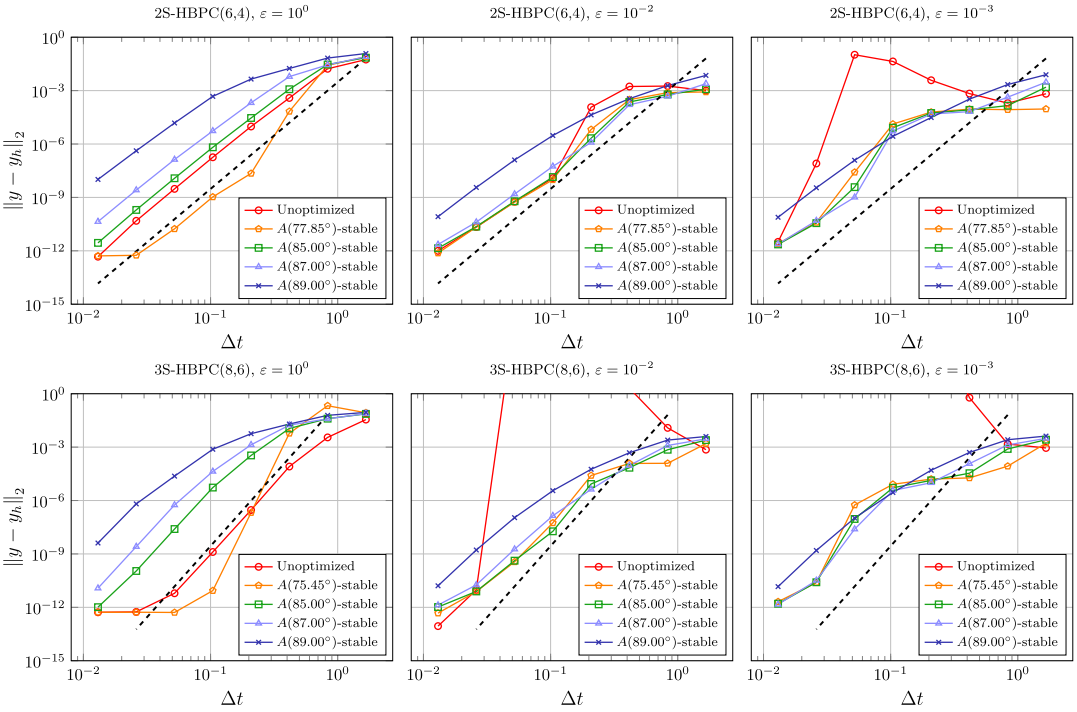


Fig. 11. Error of Pareschi–Russo problem (15) with 2S-HBPC(6,4) (top) and 3S-HBPC(8,6) (bottom) schemes at $T_{end} = 5$. The dashed black thick line represents convergence orders six (top) and eight (bottom), respectively. Consult Table 3 for (θ_1, θ_2) values corresponding to the given stability angles.

4.3. Van-der-Pol equation

To analyze the convergence behavior of mS-HBPC(q, $k_{\max}$) schemes on highly stiff problems, we consider Van-der-Pol problem. It is given by

$$y_1'(t) = y_2, \quad y_2'(t) = \frac{(1 - y_1^2)y_2 - y_1}{\varepsilon}, \quad y_0 = \left(2, -\frac{2}{3} + \frac{10}{81}\varepsilon\right). \tag{16}$$

The error is calculated at $T_{end} = 0.5$, using a refined solution found numerically. The stiffness parameters are varied across different values, from $\varepsilon = 10^{-1}$ to $\varepsilon = 10^{-5}$, for the numerical investigations.

In Fig. 12, the convergence plots for 2S-HBPC(6, 4) (top) and 3S-HBPC(8, 6) (bottom) with different stability angles are shown for various stiffness parameters. Similar to the Pareschi–Russo Problem, the unoptimized scheme becomes unstable as the problem becomes more stiff. For the equation (16) with a stiffness parameter $\varepsilon = 10^{-1}$, both sixth and eighth-order mS-HBPC(q, $k_{\max}$) schemes converge as expected, maintaining their designated order, although with some error shifts observed for $A(\alpha)$-stable schemes with larger stability angles. When $\varepsilon = 10^{-2}$, the sixth and eighth-order schemes demonstrate their expected convergence rates for lower stability angles. However, there is a slight reduction in order as the stability angle increases. In cases where the stiffness parameters are even smaller, such as $\varepsilon = 10^{-3}$, 10^{-4}, and 10^{-5}, the sixth and eighth-order $A(\alpha)$-stable schemes exhibit the desired convergence order for larger timesteps. However, as the timestep decreases, there's a noticeable reduction in order. Although the schemes with larger stability angles showcase significant error shifts in the plot for larger timesteps, the errors diminish in accordance with their respective convergence orders.

4.4. Viscous Burgers' equation

Here, we consider the viscous Burgers' equation to test the $A(\alpha)$-stable mS-HBPC(q, $k_{\max}$) schemes on partial differential equations. It is given by

$$\partial_t u + \partial_x \left(\frac{u^2}{2}\right) = \partial_{xx}^2 u, \quad u_0(x) = \frac{1 - \cos(2x)}{2}, \tag{17}$$

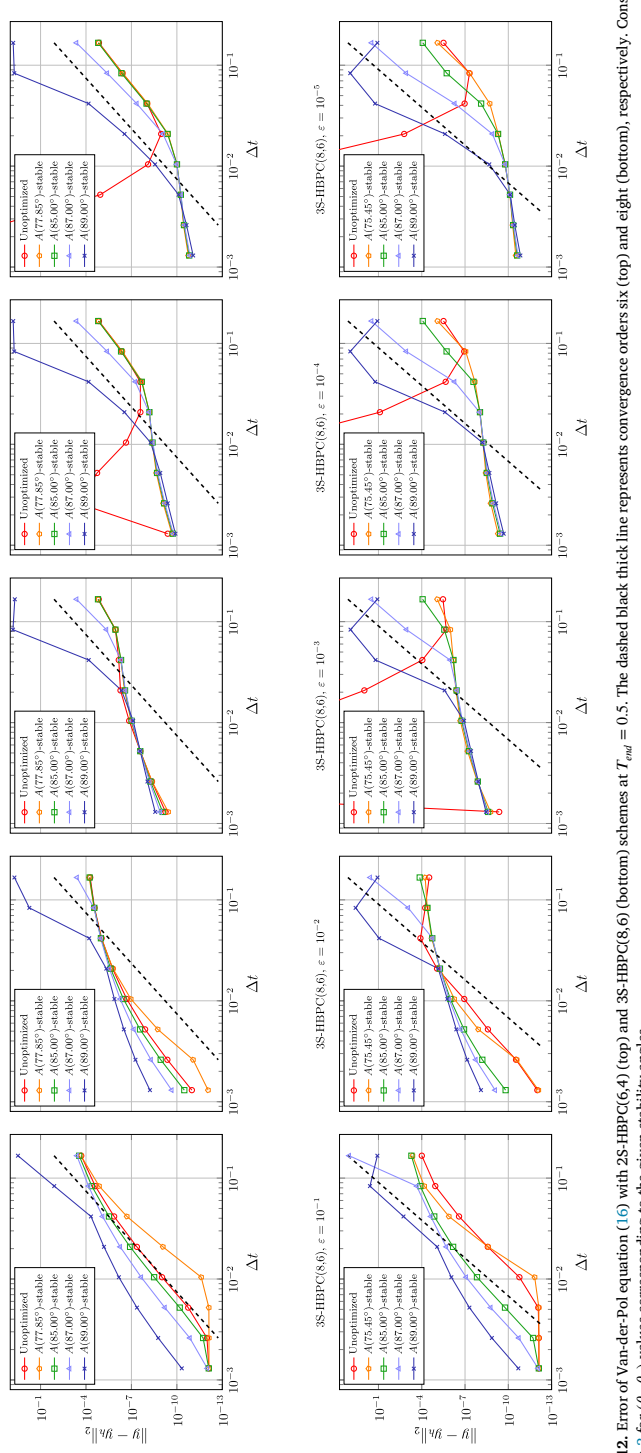


Fig. 12. Error of Van-der-Pol equation (16) with 2S-HBPC(6,4) (top) and 3S-HBPC(8,6) (bottom) schemes at $T_{end} = 0.5$. The dashed black thick line represents convergence orders six (top) and eight (bottom), respectively. Consult Table 3 for (θ_1, θ_2) values corresponding to the given stability angles.

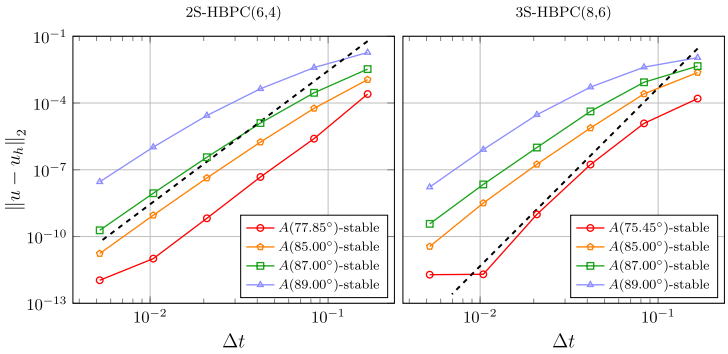


Fig. 13. Convergence results of viscous Burgers' equation (17) with 2S-HBPC(6,4) (left) and 3S-HBPC(8,6) (right) schemes at $T_{end} = 0.5$. The dashed black thick line represents convergence orders six (left) and eight (right), respectively. Consult Table 3 for (θ_1, θ_2) values corresponding to the given stability angles.

where $x \in [0, 2\pi]$ with periodic boundary conditions. The equation (17) has an explicit exact solution computed using Hopf-Cole transformation [35]. The spatial domain is discretized into a grid consisting of 200 elements with an eighth-order finite difference setting. The error is calculated at $T_{end} = 0.5$.

In Fig. 13, the convergence plots for $A(\alpha)$-stable 2S-HBPC(6, 4) (left) and 3S-HBPC(8, 6) (right) are shown for different stability angles. The sixth-order $A(\alpha)$-stable schemes exhibit desired order of convergence for all the given stability angles. The eight-order $A(\alpha)$-stable schemes achieve desired order of convergence for lower stability angles, whereas an order reduction is observed for larger stability angles. This phenomenon is more likely to result from larger values of θ_1 for 3S-HBPC(8, 6) compared to 2S-HBPC(6, 4), for the same stability angles.

5. Conclusion and outlook

In this work, we have presented a class of multi-derivative predictor-corrector timestepping schemes analogous to the schemes presented in [13], but with an underlying quadrature rule that is constructed using a multi-step fashion. However, the initial schemes are found not to be $A(\alpha)$-stable. Therefore, the schemes have been analyzed and optimized for better stability properties.

It has been found that the 2S-HBPC(6, 4) and 3S-HBPC(8, 6) schemes are $A(\alpha)$-stable for any values of θ_2 greater than or equal to 1.25868 and 3.84703, respectively. These threshold values have been fixed for θ_2 as they give the minimum one-step error for a fixed stability angle. Similarly, lower limit of $\theta_1 = 0.42083$ for the 2S-HBPC(6, 4) scheme and $\theta_2 = 0.37957$ for the 3S-HBPC(8, 6) scheme have been proposed by analyzing the error constant $C(\theta_1, \theta_2)$. Therefore, schemes can be designed to have higher stability angles for an increased value of θ_1. In Table 3, a selection of θ_1 values corresponding to specific stability angles (α) has been presented.

Convergence of the 2S-HBPC(6, 4) and 3S-HBPC(8, 6) schemes have been shown numerically. For non-stiff ordinary differential equations, the $A(\alpha)$-stable schemes converge with their expected rates, particularly demonstrating minimal error at lower stability angles. The $A(\alpha)$-stable schemes gradually converge towards their expected order of convergence in stiff problems. In contrast, a reduction in the order of convergence has been observed in very stiff problems. However, the unoptimized schemes were unstable for stiff problems. For higher-order non-linear partial differential equations, 2S-HBPC(6, 4) scheme has exhibited the desired convergence order for all the given stability angles, whereas 3S-HBPC(8, 6) scheme has exhibited desired order of convergence for lower stability angles.

As a future work, we are interested in combining the scheme with discontinuous Galerkin (DG) spatial discretization and studying the performance of the schemes over Navier-Stokes equations. An efficient scheme is paramount when the vast run time requirement for larger systems such as Navier-Stokes equations is concerned. Therefore, another possible direction for future investigation would be to utilize the parallelizability of the scheme across the correction steps and, hence, to implement a parallel-in-time mS-HBPC(q, $k_{\max}$) scheme.

CRediT authorship contribution statement

Arjun Thenery Manikantan: Conceptualization, Data curation, Formal analysis, Investigation, Methodology, Software, Validation, Visualization, Writing – original draft, Writing – review & editing. **Jochen Schütz:** Conceptualization, Formal analysis, Funding acquisition, Investigation, Methodology, Project administration, Resources, Software, Supervision, Validation, Visualization, Writing – original draft, Writing – review & editing.

Acknowledgements

Arjun Thenery Manikantan was funded by the "Bijzonder Onderzoeksfonds" (BOF) from UHasselt - project no. BOF21KP12. We acknowledge the VSC (Flemish Supercomputer Center) for providing computing resources. The VSC is funded by the Research Foundation - Flanders (FWO) and the Flemish Government.

Appendix A. Supplementary material

Supplementary material related to this article can be found online at https://doi.org/10.1016/j.apnum.2024.07.011.

References

[1] P. Turán, On the theory of the mechanical quadrature, Acta Sci. Math. 12 (1950) 30–37.
[2] A.H. Stroud, D.D. Stancu, Quadrature formulas with multiple Gaussian nodes, SIAM J. Numer. Anal. 2 (1965) 129–143.
[3] J.C. Butcher, G. Hojjati, Second derivative methods with RK stability, Numer. Algorithms 40 (2005) 415–429.
[4] R. Chan, A. Tsai, On explicit two-derivative Runge-Kutta methods, Numer. Algorithms 53 (2010) 171–194.
[5] E. Hairer, G. Wanner, Multistep-multistage-multiderivative methods for ordinary differential equations, Computing (Arch. Elektron. Rechnen) 11 (1973) 287–303.
[6] J. Chouchoulis, J. Schütz, J. Zeifang, Jacobian-free explicit multiderivative Runge–Kutta methods for hyperbolic conservation laws, J. Sci. Comput. 90 (2022) 96.
[7] S. Gottlieb, Z.J. Grant, J. Hu, R. Shu, High order strong stability preserving multiderivative implicit and IMEX Runge–Kutta methods with asymptotic preserving properties, SIAM J. Numer. Anal. 60 (2022) 423–449.
[8] S. Gottlieb, C.-W. Shu, E. Tadmor, Strong stability-preserving high-order time discretization methods, SIAM Rev. 43 (2001) 89–112.
[9] A.J. Christlieb, S. Gottlieb, Z.J. Grant, D.C. Seal, Explicit strong stability preserving multistage two-derivative time-stepping schemes, J. Sci. Comput. 68 (2016) 914–942.
[10] Z. Grant, S. Gottlieb, D.C. Seal, A strong stability preserving analysis for explicit multistage two-derivative time-stepping schemes based on Taylor series conditions, Commun. Appl. Math. Comput. 1 (2019) 21–59.
[11] A. Moradi, J. Farzi, A. Abdi, Strong stability preserving second derivative general linear methods, J. Sci. Comput. 81 (2019) 392–435.
[12] J. Schütz, D. Seal, An asymptotic preserving semi-implicit multiderivative solver, Appl. Numer. Math. 160 (2021) 84–101.
[13] J. Schütz, D.C. Seal, J. Zeifang, Parallel-in-time high-order multiderivative IMEX solvers, J. Sci. Comput. 90 (2022) 1–33.
[14] J. Zeifang, J. Schütz, D. Seal, Stability of implicit multiderivative deferred correction methods, BIT Numer. Math. (2022).
[15] J. Zeifang, J. Schütz, Implicit two-derivative deferred correction time discretization for the discontinuous Galerkin method, J. Comput. Phys. 464 (2022) 111353.
[16] J. Zeifang, A. Thenery Manikantan, J. Schütz, Time parallelism and Newton-adaptivity of the two-derivative deferred correction discontinuous Galerkin method, Appl. Math. Comput. 457 (2023) 128198.
[17] E. Theodosiou, J. Schütz, D. Seal, An explicitness-preserving imex-split multiderivative method, Comput. Math. Appl. 158 (2024) 139–149, https://doi.org/10.1016/j.camwa.2023.12.040.
[18] R.L. Brown, Multi-derivative numerical methods for the solution of stiff ordinary differential equations, Technical Report UIUCDCS-R-74-672, Department of Computer Science, University of Illinois, 1974.
[19] R. Jeltsch, A0-stability and stiff stability of Brown's multistep multiderivative methods, Numer. Math. 32 (1979) 167–181, https://doi.org/10.1007/BF01404873.
[20] E. Hairer, G. Wanner, Solving Ordinary Differential Equations II, Springer Series in Computational Mathematics, 1991.
[21] G. Califano, G. Izzo, Z. Jackiewicz, Strong stability preserving general linear methods with Runge–Kutta stability, J. Sci. Comput. 76 (2018) 943–968, https://doi.org/10.1007/s10915-018-0646-5.
[22] M.M. Khalsaraei, A. Shokri, M. Molayi, The new class of multistep multiderivative hybrid methods for the numerical solution of chemical stiff systems of first order IVPs, J. Math. Chem. 58 (2020) 1987–2012, https://doi.org/10.1007/s10910-020-01160-z.
[23] J.C. Butcher, General linear methods, Acta Numer. 15 (2006) 157–256.
[24] H. Zhang, A. Sandu, S. Blaise, Partitioned and implicit–explicit general linear methods for ordinary differential equations, J. Sci. Comput. 61 (2014) 119–144.
[25] S. Boscarino, J. Qiu, G. Russo, Implicit-explicit integral deferred correction methods for stiff problems, SIAM J. Sci. Comput. 40 (2018) A787–A816.
[26] A. Dutt, L. Greengard, V. Rokhlin, Spectral deferred correction methods for ordinary differential equations, BIT Numer. Math. 40 (2000) 241–266.
[27] M. Minion, Semi-implicit spectral deferred correction methods for ordinary differential equations, Commun. Math. Sci. 1 (2003) 471–500.
[28] MathWorks, ode15s documentation, Online Documentation, https://www.mathworks.com/help/matlab/ref/ode15s.html, 2022.
[29] Z. Zlatev, Stability properties of variable stepsize variable formula methods, Numer. Math. 31 (1978) 175–182, https://doi.org/10.1007/BF01397474.
[30] M. Crouzeix, F.J. Lisbona, The convergence of variable-stepsize, variable-formula, multistep methods, SIAM J. Numer. Anal. 21 (1984) 512–534, http://www.jstor.org/stable/2157066.
[31] D. Wang, S.J. Ruuth, Variable step-size implicit-explicit linear multistep methods for time-dependent partial differential equations, J. Comput. Math. 26 (2008) 838–855, http://www.jstor.org/stable/43693484, full publication date: November 2008.
[32] MathWorks, Symbolic Math Toolbox, Natick, Massachusetts, United States, 2020, https://www.mathworks.com/help/symbolic/.
[33] A. Prothero, A. Robinson, On the stability and accuracy of one-step methods for solving stiff systems of ordinary differential equations, Math. Comput. 28 (1974) 145–162, https://doi.org/10.1090/S0025-5718-1974-0331793-2.
[34] L. Pareschi, G. Russo, Implicit-explicit Runge-Kutta schemes for stiff systems of differential equations, Recent Trends Numer. Anal. 3 (2000) 269–289.
[35] E. Hopf, The partial differential equation $u_t + uu_x = \mu u_{xx}$, Commun. Pure Appl. Math. 3 (1950) 201–230.

Paper III:

On the stability of two-derivative time discretizations

On the stability of two-derivative time discretizations

Arjun Thenery Manikantan[*], Jonas Zeifang[†], Jochen Schütz[‡]

Abstract

In this paper, we analyze stability properties of the two-derivative strong stability preserving schemes presented in [Gottlieb et al., SIAM Journal on Numerical Analysis 60, 2022]. Stability analysis shows that the diagonally implicit two-derivative two-stage third-order strong stability preserving scheme can never be A-stable. We provide a detailed investigation of the third-order schemes and discuss stabilizing strategies. The stabilizing techniques are applicable to tune any general implicit two-derivative scheme. We implement the two-derivative strong stability preserving schemes for partial differential equations with a discontinuous Galerkin spectral element spatial discretization. We use Newton's method for non-linear stage equations and the generalized minimal residual method with a matrix-free approach for solving linear algebraic equations under suitable preconditioning. The method is applied for compressible Euler and Navier-Stokes equations with orders up to four. Numerical results show that the second and fourth-order strong stability preserving schemes attain their desired order of convergence for relatively large timesteps. In contrast, third-order schemes require smaller timesteps to exhibit convergence. Nevertheless, the improved adaptive third-order scheme yields stable solutions.

Keywords: Strong stability preserving; Implicit time stepping; Multiderivative schemes; Stability analysis; Discontinuous Galerkin spectral element method

AMS Subject Classification: 65L05; 65M20; 65M22; 65M60

1 Introduction

Consider the partial differential equations (PDE) defined on the spatial domain $\Omega \subset \mathbb{R}^n$,

$$\mathbf{w}_t + \nabla \cdot \mathbf{F}(\mathbf{w}) = 0, \tag{1}$$

where $\mathbf{w}(x,t) : \Omega \times \mathbb{R}_{\geq 0} \to \mathbb{R}^m$ is the state vector and $\mathbf{F}(\mathbf{w}) : \mathbb{R}^m \to \mathbb{R}^m$ is the flux. For the Navier-Stokes equation, the flux function is given by

$$\mathbf{F}(\mathbf{w}) := \mathbf{F}^{\text{inv}}(\mathbf{w}) - \mathbf{F}^{\nu}(\mathbf{w}, \nabla \mathbf{w}), \tag{2}$$

with the inviscid flux $\mathbf{F}^{\text{inv}}$ and the viscous flux $\mathbf{F}^{\nu}$, where fluxes are given in Eqs. (38) and (41) below. The Euler equations are obtained by setting $\mathbf{F}^{\nu} \equiv 0$ in Eq. (2). Formally, the PDE (1) can be cast into an ordinary differential equation (ODE)

$$\mathbf{w}_t = \mathbf{R}^{(1)}(\mathbf{w}) \tag{3}$$

in some infinite dimensional function space, where the function $\mathbf{R}^{(1)}(\mathbf{w})$ is defined as

$$\mathbf{R}^{(1)}(\mathbf{w}) := -\nabla \cdot \mathbf{F}(\mathbf{w}). \tag{4}$$

[*] arjun.thenerymanikantan@uhasselt.be, Faculty of Sciences & Data Science Institute, UHasselt, Belgium
[†] jonas.zeifang@uhasselt.be, Faculty of Sciences & Data Science Institute, UHasselt, Belgium
[‡] jochen.schuetz@uhasselt.be, Faculty of Sciences & Data Science Institute, UHasselt, Belgium

Widely used time stepping methods in the field of compressible computational fluid dynamics are of explicit nature. However, the conditional stability of the explicit integration schemes imposes timestep restrictions, which arise from the CFL condition. In order to overcome these timestep restrictions, we use *implicit time stepping schemes* which have less severe or no restrictions on the timestep. Multistep methods such as backward difference formulae (BDF) or multistage methods such as diagonally implicit Runge-Kutta (DIRK) methods are standard examples for implicit time stepping methods. In [12], some efficient implicit schemes based on Rosenbrock Runge-Kutta methods [2] and BDF methods can be found.

Classical implicit time stepping schemes in literature mostly use only the first-order derivative ($\mathbf{w}_t$) for temporal integration [12]. The consistency order of one-derivative Runge-Kutta methods can only be improved by increasing the number of stages; this requires additional memory and necessitates solving more involved order conditions. To overcome this situation, one can include higher-order derivatives in the time stepping procedure, resulting in *multiderivative methods*. The multistage multiderivative schemes were first considered half a century ago in [29, 28, 10]. In general, these methods belong to the larger class of multistep-multistage-multiderivative methods [10]. Including higher order derivatives in time stepping gives more flexibility to the method; hence it can achieve higher orders of consistency with the same number of stages as used in the one-derivative schemes [28, 4]. This current work is limited to only *two-derivative schemes*. Hence it is required to compute also the second-order derivative for the temporal integration,

$$\mathbf{w}_{tt} = (\mathbf{R}^{(1)}(\mathbf{w}))_t = -\nabla \cdot \left(\frac{\partial \mathbf{F}}{\partial \mathbf{w}}(\mathbf{w}) \mathbf{R}^{(1)}(\mathbf{w}) \right) := \mathbf{R}^{(2)}(\mathbf{w}, \mathbf{R}^{(1)}(\mathbf{w})). \tag{5}$$

An asymptotic preserving higher-order implicit two-derivative method for stiff ODEs was developed in a predictor-corrector fashion [25, 27], termed Hermite-Birkhoff Predictor-Corrector schemes (HBPC). In each correction step, the order of consistency is raised by one until some maximal order is reached; order reduction for low stiffness parameters was mitigated by adding more correction steps. The HBPC scheme was optimized in [30] to have $A(\alpha)$ stability almost up to $\alpha = 90^o$.

In [7], Gottlieb et al. developed a higher-order implicit two-derivative Runge-Kutta-type scheme that possesses the strong stability preserving (SSP) property. Refer to [8, 5, 9] for more two-derivative SSP schemes. Besides multistage methods, there are higher-order SSP two-derivative general linear methods in literature, see, e.g. [23]. In [5, 9], explicit two-derivative SSP schemes are combined with weighted essentially non-oscillatory methods to solve conservation laws. Applications of implicit multiderivative methods combined with discontinuous Galerkin spatial discretization for (non-)linear PDEs can be found in [26, 14, 24]. In [31], the HBPC schemes [25, 27] are combined with discontinuous Galerkin spectral element method (DGSEM) [17] and achieve order of accuracy up to eight. The scheme is parallelized in space and in [32] also in time.

In this work, we combine the two-derivative implicit SSP [7] time discretization approach with a spatial discretization of the DGSEM to solve Navier-Stokes equations. Linear stability properties of the SSP schemes [7] are investigated. A detailed stability analysis of the third-order SSP scheme is conducted, and several stabilizing strategies are discussed. We leverage the flexibility of two-derivative schemes, which allows for tuning the positions of the poles of the stability function, to enhance the stability of our scheme. As a result, we derive a family of A(α)-stable third-order SSP schemes and implement them in an adaptive manner to provide stable solutions. The implicit stages of the discretized system are solved using Newton's method and the generalized minimal residual method (GMRES) is used with appropriate preconditioning to solve the linear system arising in the Newton iterations. As the SSP schemes are structurally very similar to the stages of the HBPC schemes, we closely follow the preconditioning and the matrix-free approach implemented in [31].

The sections of this paper are structured as follows: In Sec. 2 the semi-descretization of the two-derivative SSP schemes are explained with Butcher form, followed by the investigation of the stability properties. The fully discrete formulation is briefed in Sec. 3 with the weak formulation of the PDEs. Effect of preconditioning and the

implementation of the preconditioner using a matrix-free approach is shortly recalled in Sec. 3.2. In Sec. 4.1.1, a detailed analysis of the third-order schemes is conducted, and adaptive schemes are derived to enhance stability properties. The validation of the SSP schemes on Euler and Navier-Stokes equations is presented in Secs. 4.2 and 4.3, including convergence results and statistics of the number of (non-)linear iterations. In the final section (Sec. 5), conclusions are made and an outlook is given.

2 Semi-discrete formulation

In this section, we discuss the semi-discrete formulation of the implicit two-derivative SSP Runge-Kutta method. We use the spatial operators $\mathbf{R}^{(1)}$ and $\mathbf{R}^{(2)}$, see Eq. (3) and Eq. (5), for the semi-discretization. The aforementioned spatial operators are discretized using DGSEM, which will be described in Sec. 3. Given the approximate solution $\mathbf{w}^n$ at time t^n, the s-stage diagonally implicit two-derivative SSP Runge-Kutta method can be written in Butcher form,

$$\mathbf{w}^{(i)} := \mathbf{w}^n + \Delta t \sum_{j=1}^{i} a_{ij}\mathbf{R}^{(1)}(\mathbf{w}^{(j)}) + \Delta t^2 \sum_{j=1}^{i} \dot{a}_{ij}\mathbf{R}^{(2)}(\mathbf{w}^{(j)}, \mathbf{R}^{(1)}(\mathbf{w}^{(j)})), \quad 1 \leq i \leq s. \tag{6}$$

The solution at time t^{n+1} is updated with

$$\mathbf{w}^{n+1} := \mathbf{w}^{(s)}. \tag{7}$$

The authors derived the diagonally implicit two-derivative SSP scheme in [7] for orders up to four; see Tab. 1. The Butcher coefficients $\mathbf{A} = \{a_{ij}\}$ and $\dot{\mathbf{A}} = \{\dot{a}_{ij}\}$ for the schemes can be found in [7, Sec. 2.3]. In Sec. 2.1, we analyze the linear stability of the implicit SSP schemes (Tab. 1).

Scheme	**Order**	**Stages**	A(α)-**Stability**
SSP-I2DRK21	2	1	A(90.00°)
SSP-I2DRK32	3	2	A(79.94°)
SSP-I2DRK45	4	5	A(84.52°)

Table 1: Strong stability preserving schemes from [7, Sec. 2.3]. The A(α) stability analysis can be found in Sec. 2.1 of the current paper.

2.1 Linear stability of the implicit two-derivative SSP schemes

In this section, the linear stability of the implicit two-derivative SSP method has been analyzed using Dahlquist's equation $w' = \lambda w$, where $\lambda \in \mathbb{C}$. We first derive the stability function for an s-stage SSP scheme, and then the linear stability for each of the schemes (see Tab. 1) will be analyzed in the following subsections. Plugging in $\mathbf{R}^{(1)} = \lambda w$ and $\mathbf{R}^{(2)} = \lambda w' = \lambda^2 w$ in Eq. (6) we get

$$w^{(i)} = y^n + \sum_{j=1}^{i} \left(a_{ij}\Delta t\lambda + \dot{a}_{ij}\Delta t^2\lambda^2\right) w^{(j)} \tag{8}$$

for each of the stages up to s. Define $z := \Delta t\lambda$ and functions $\mathcal{S}_i(z)$, so that the stage values $w^{(i)}$ in Eq. (8) can be explicitly written as

$$w^{(i)} = \mathcal{S}_i(z) w^n.$$

For $i \geq 2$, these functions can be recursively written as

$$\mathcal{S}_i(z) = \frac{1 + \sum_{j=1}^{i-1} \left(a_{ij} z + \dot{a}_{ij} z^2\right) \mathcal{S}_j(z)}{\left(1 - a_{ii} z - \dot{a}_{ii} z^2\right)} \text{ with } \mathcal{S}_1(z) = (1 - a_{11} z - \dot{a}_{11} z^2)^{-1}.$$

Then the update w^{n+1} can be written as

$$w^{n+1} = \mathcal{S}_s(z) w^n = \left(\frac{1 + \sum_{j=1}^{s-1} \left(a_{sj} z + \dot{a}_{sj} z^2\right) \mathcal{S}_j(z)}{\left(1 - a_{ss} z - \dot{a}_{ss} z^2\right)} \right) y^n =: \mathcal{S}(z) w^n, \tag{9}$$

where $\mathcal{S}(z)$ is the stability function.

Second-order SSP scheme

The second order SSP scheme SSP-I2DRK21 is none other than second-order Taylor method. From Eq. (9), the stability function for SSP-I2DRK21 is given by

$$\mathcal{S}(z) = \mathcal{S}_1(z) = \frac{2}{2 - 2z + z^2} \text{ for } z \in \mathbb{C}. \tag{10}$$

Then for the modulus value $|\mathcal{S}(z)|$

$$\max_{z \in \mathbb{C}^-} |\mathcal{S}(z)| \leq \max_{y \in \mathbb{R}} |\mathcal{S}(iy)| = \max_{y \in \mathbb{R}} \frac{2}{|(2 - y^2) - 2yi|} = \max_{y \in \mathbb{R}} \frac{2}{\sqrt{y^4 + 4}} \leq 1,$$

which implies that the second-order SSP method is **A-stable**, see [11, Chap. 5]. The method is also **L-stable** because,

$$\lim_{z \to \infty} \mathcal{S}(z) = 0.$$

Third-order SSP scheme

The stability function for SSP-I2DRK32 is given by

$$\mathcal{S}(z) = \mathcal{S}_2(z) = \frac{18}{(6 + z^2)(3 - 3z + z^2)} \text{ for } z \in \mathbb{C}. \tag{11}$$

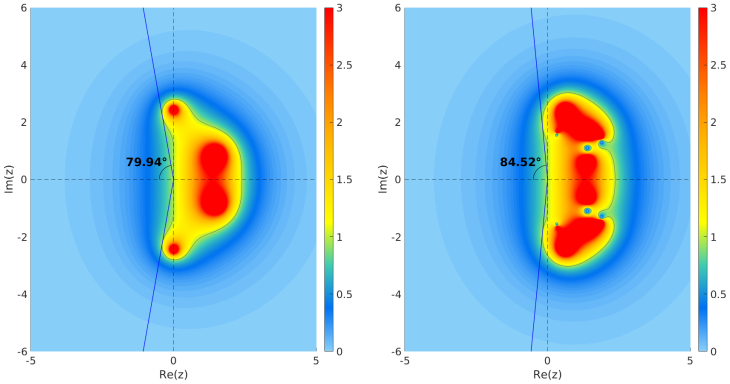
It can be found from Eq. (11) that $\mathcal{S}(z)$ has singularities on the boundary of the domain $\mathbb{C}^-$ at $z = \pm i\sqrt{6}$. In Fig. 1 (a), the stability region ($|\mathcal{S}(z)| < 1$) has been plotted for the values $-5 \leq \text{Re}(z) \leq 5$ and $-6 \leq \text{Im}(z) \leq 6$. The stability region is represented by the areas shaded in blue in Fig. 1 (a). It can be seen that the stability region does not cover the entire negative complex half-plane, which implies that the method is **not A-stable**. Using the algorithm from [30, Sec. 3], it can be found that the method is approximately A(79.94°) stable.

Fourth-order SSP scheme

The stability function for SSP-I2DRK45 is given by

$$\mathcal{S}(z) = \mathcal{S}_5(z) = \frac{1 + \sum_{j=1}^{4} \left(a_{5j} z + \dot{a}_{5j} z^2\right) \mathcal{S}_j(z)}{\left(1 - a_{55} z - \dot{a}_{55} z^2\right)} \text{ for } z \in \mathbb{C} \tag{12}$$

with the Butcher coefficients given in [7, Sec. 2.3]. Similarly to the third order SSP scheme, the stability region (blue shaded region in Fig. 1 (b)) of the fourth order scheme also does not cover the entire negative complex half-plane, which implies that the method is **not A-stable**; the method is approximately A(84.52°) stable.



(a) SSP-I2DRK32 (b) SSP-I2DRK45

Figure 1: Stability region ($|\mathcal{S}(z)| < 1$) for third order (a) and fourth order (b) SSP schemes. Stability angle is found using the algorithm from [30, Sec. 3] via the stability functions given in Eq. (11) and Eq. (12) respectively.

2.2 Non-existence of an A-stable implicit two-derivative two-stage third-order SSP scheme

As the SSP-I2DRK32 scheme exhibits a relatively low stability angle, we develop an improved scheme in this section.

Lemma 1. *There is no diagonally implicit two-derivative two-stage third-order scheme which is both* ***A-stable*** *and* ***SSP*** *according to the conditions in [7, Theorem 1].*

Proof. We utilize the order conditions for third-order accuracy from [7, Sec. 2.1] along with the SSP conditions from [7, Sec. 2.2] and [7, Theorem 1] to derive the Butcher coefficients

$$\mathbf{A} = \begin{bmatrix} \rho & 0 \\ \phi & \eta \end{bmatrix} \text{ and } \dot{\mathbf{A}} = \begin{bmatrix} \alpha & 0 \\ \gamma & \beta \end{bmatrix}. \tag{13}$$

So, for $0 < k \leq 1$ we have

$$\phi = k\rho \text{ and } \gamma = k\alpha \tag{14}$$

where $\rho \geq 0$, $\eta \geq 0$, $\alpha \leq 0$ and $\beta \leq 0$. Now, rewrite the Butcher coefficients (13) with the aforementioned SSP conditions and the first-order condition ($\phi + \eta = 1$) to obtain

$$\mathbf{A} = \begin{bmatrix} \rho & 0 \\ k\rho & 1 - k\rho \end{bmatrix} \text{ and } \dot{\mathbf{A}} = \begin{bmatrix} \alpha & 0 \\ k\alpha & \beta \end{bmatrix}. \tag{15}$$

Now, applying the second-order condition [7, Sec. 2.1, $p = 2$] we get

$$\beta = k(\rho - \rho^2 - \alpha) - \frac{1}{2}. \tag{16}$$

Consider Eq. (16) and apply the third-order conditions [7, Sec. 2.1, $p = 3$] to obtain

$$k(\rho^3 - 2\rho^2 + 2\rho\alpha + \rho - 2\alpha) = \frac{1}{3}, \tag{17}$$

$$k(2\rho^3 - 2\rho^2 + 4\rho\alpha + \rho - 2\alpha) = \frac{1}{3}. \tag{18}$$

We assume that $k \neq 0$ because the third order conditions (see [7, Sec. 2.1]) give rise to an inconsistent system of equations for $k = 0$ (see Eqs. (17) and (18)). Now, we can solve the equations (17) and (18) for ρ by eliminating k. Hence we have

$$\rho^3 + 2\alpha\rho = 0 \;\Rightarrow\; \rho = 0, \text{ or } \rho = \pm\sqrt{-2\alpha}.$$

We start with the non-zero solution. From the condition that $\alpha \leq 0$, take $\alpha = -2\mu^2$, for $\mu > 0$. Since the SSP property [7, Theorem 1] necessitates a non-negative ρ, the non-zero solution is

$$\rho = 2\mu, \;\; \mu > 0. \tag{19}$$

Substituting for α and ρ in (17) or (18) gives

$$k = \frac{1}{6(\mu - 2\mu^2)}, \;\; \mu > 0. \tag{20}$$

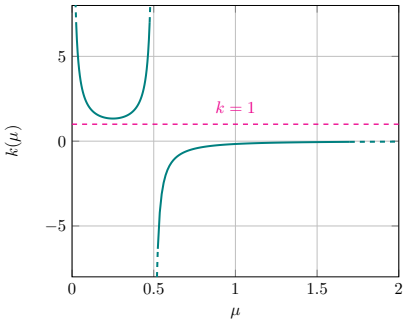


Figure 2: Graph of variable k (20) from the Butcher coefficients (15), obtained as a function of μ using the order equations (17) and (18). The function $k(\mu) = \frac{1}{6(\mu-2\mu^2)}$ never reaches values from (0,1] for any values of $\mu > 0$.

The values of k have been plotted in Fig. 2 for $\mu > 0$. It can been seen clearly from Fig. 2 that

$$k = \begin{cases} > 1, \; 0 < \mu < 0.5 \\ < 0, \; \mu > 0.5 \end{cases}. \tag{21}$$

So, there are no values for $\mu > 0$ such that $0 < k \leq 1$. Now consider the case when $\rho = 0$. Then from Eq. (17) and Eq. (16) we get

$$k\alpha = -\frac{1}{6}, \text{ and } \beta = -\frac{1}{3}. \tag{22}$$

Hence there are infinitely many possibilities for third-order SSP schemes when $\rho = 0$ and it is given by

$$\mathbf{A} = \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix} \text{ and } \dot{\mathbf{A}} = \begin{bmatrix} -\frac{1}{6k} & 0 \\ -\frac{1}{6} & -\frac{1}{3} \end{bmatrix}, \ 0 < k \leq 1. \tag{23}$$

The SSP scheme (23) is the same third-order SSP scheme referred to in [7] for $k = 1$. The stability function for the scheme (23) is

$$\mathcal{S}(z) = \frac{18k + 3(1-k)z^2}{(6k + z^2)(3 - 3z + z^2)}, \tag{24}$$

and the scheme (23) can **never be A-stable** as $\mathcal{S}(z)$ has singularities at $\pm i\sqrt{6k}$. □

2.3 A family of A(α)-stable implicit two-derivative two-stage third-order SSP scheme

Even though the third-order SSP scheme discussed above cannot be A-stable, the analysis in the previous section has led to a family of A(α)-stable implicit two-derivative, two-stage, third-order SSP schemes, as given in (23), parameterized by a free variable k. The α angles are plotted against different k values in Fig. 3. It can be noted from Fig. 3 that the schemes (23) are tending towards $A(90^o)$ as $k \to 0$. However, the Butcher coefficient $\dot{a}_{11} = -\frac{1}{6k} \to -\infty$, which can significantly affect the error constant of the schemes.

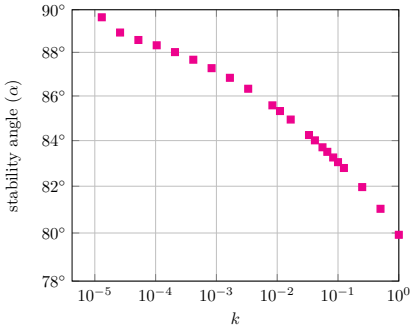


Figure 3: Stability angles α for the $A(\alpha)$-stable two-derivative two-stage third-order scheme (23) are plotted for different values of the free variable $0 < k \leq 1$.

Consider the stability regions plotted in Fig. 4 to analyze the effect of the free parameter k on the stability angle. It can be seen in Fig. 4 that, as $k \to 0$, the poles on the imaginary axis ($\pm i\sqrt{6k}$) move toward the origin. Meanwhile, the unstable region in the left half of the complex plane becomes smaller, resulting in an increase in the stability angle. This poses a significant issue because, as Δt approaches 0, the term $z = \Delta t\, \lambda$ tends to concentrate around the origin, where λ represents the eigenvalues of the discretized ODE system.

In the upcoming sections, the family of $A(\alpha)$ stable third-order SSP schemes (23) with $0 < k < 1$, will be denoted as SSP-I2DRK32(k). When $k = 1$, SSP-I2DRK32(1) refers to the original scheme SSP-I2DRK32. For values of $k > 1$, the schemes are no longer SSP, and we will denote them as NSSP-I2DRK32(k).

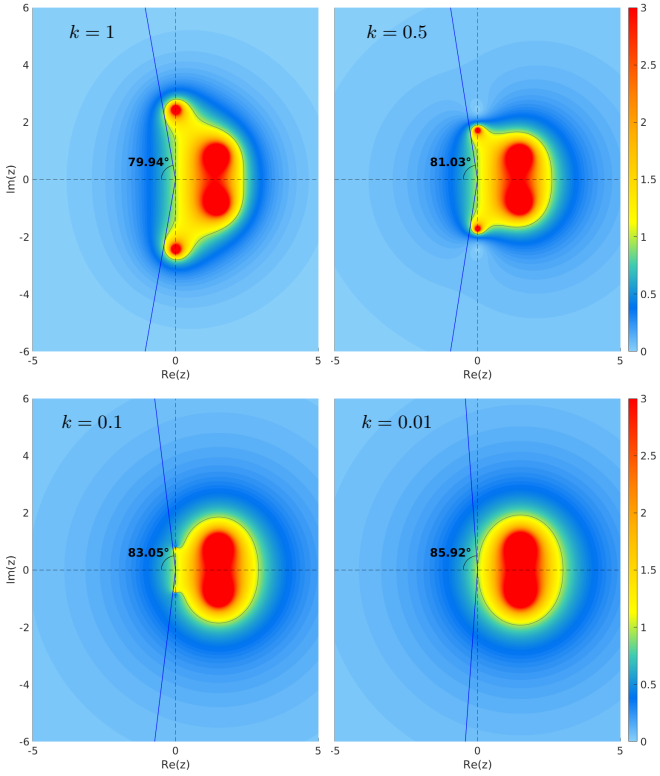


Figure 4: Stability region ($|\mathcal{S}(z)| < 1$) for the third-order SSP scheme SSP-I2DRK32(k), plotted using the algorithm from [30, Sec. 3] via the stability functions in Eq. (24) for different values of the free parameter k.

3 Fully discrete formulation

We use the discontinuous Galerkin spectral element method which has been introduced in [17] for the spatial discretization. The spatial domain Ω under consideration is subdivided into N_E quadrangular (2D) or hexahedral (3D) elements Ω_e. The discrete formulation of the DGSEM is recalled briefly in the following section by closely following the papers [31, 13]. The weak formulations for the first and second derivatives are provided only for the Euler equations in the following subsection; detailed formulations for the Navier–Stokes equations can be found in [31, Sec. 5].

3.1 Evaluation of the temporal derivatives with the DGSEM

The DGSEM utilizes the weak formulation of Eq. (1),

$$\sum_{e=1}^{\mathrm{N_E}}(\mathbf{w}_t,\phi)_{\Omega_e}-(\mathbf{F}(\mathbf{w}),\nabla\phi)_{\Omega_e}+\left\langle \mathbf{F}^*(\mathbf{w}^L,\mathbf{w}^R)\cdot\mathbf{n},\phi\right\rangle_{\partial\Omega_e}=0,\ \forall\phi\in\Pi_{\mathrm{N_p}}, \tag{25}$$

where $\Pi_{\mathrm{N_p}}$ is the space of test functions constructed by the tensor product of the one-dimensional Lagrange interpolation polynomials of degree $\mathrm{N_p}$. The scalar product $(\cdot,\cdot)_{\Omega_e}$ denotes the element-wise integration over the elements Ω_e and $\langle\cdot,\cdot\rangle_{\partial\Omega_e}$ is the integration along the cell-edges $\partial\Omega_e$. The flux function is replaced by a numerical flux function $F^*(\mathbf{w}^L,\mathbf{w}^R)$ on the cell-edges, which depends on the left and right states with respect to the cell-edge, and $\mathbf{n}$ is the outward pointing normal to the cell-edge. The global Lax-Friedrichs flux is used as the numerical flux (see [31, Eq. (13) and Eq. (17)]). The spatial DGSEM operator for the first time derivative $\mathbf{R}_h^{(1)}(\mathbf{w}_h)$ is given in [31, Eq. (12)].

As we use two-derivative time stepping schemes (6), it is required to compute the spatial DGSEM operator for the second derivative. The spatial operator for the second derivative is evaluated by introducing the artificial quantity

$$\boldsymbol{\sigma}:=\mathbf{R}^{(1)}(\mathbf{w})\equiv\mathbf{w}_t \tag{26}$$

as done in [26]. Differentiating Eq. (25) with respect to time and applying $\boldsymbol{\sigma}$, we get,

$$\sum_{e=1}^{\mathrm{N_E}}(\mathbf{w}_{tt},\phi)_{\Omega_e}-\left(\frac{\partial\mathbf{F}(\mathbf{w})}{\partial\mathbf{w}}\boldsymbol{\sigma},\nabla\phi\right)_{\Omega_e}+\left\langle\frac{\partial\mathbf{F}^*(\mathbf{w}^L,\mathbf{w}^R)}{\partial\mathbf{w}^L}\boldsymbol{\sigma}^L\cdot\mathbf{n}+\frac{\partial\mathbf{F}^*(\mathbf{w}^L,\mathbf{w}^R)}{\partial\mathbf{w}^R}\boldsymbol{\sigma}^R\cdot\mathbf{n},\phi\right\rangle_{\partial\Omega_e}=0,\forall\phi\in\Pi_{\mathrm{N_p}}. \tag{27}$$

The spatial DGSEM operator for the second time derivative $\mathbf{R}_h^{(2)}(\mathbf{w}_h,\mathbf{R}^{(1)})$ is also evaluated similarly to that of the first time derivative and it is given in [31, Eq. (16)].

The SSP schemes [7] for the conservation laws (1) are implemented in the open source code FLEXI[1], which was developed for solving hyperbolic-parabolic conservation equations in a discontinuous Galerkin setting [18].

3.2 Preconditioning of the extended linear system

The stage values in Eq. (6) can be solved using Newton's method. For the i^{th} stage, the equation can be cast into the non-linear form

$$\mathbf{G}(\mathbf{w}^{(i)}):=\mathbf{g}(\mathbf{w}^{(i)})-\mathbf{b}=0, \tag{28}$$

where the function $\mathbf{g}$ and vector $\mathbf{b}$ are given by

$$\begin{aligned}\mathbf{g}(\mathbf{w}^{(i)})&=\mathbf{w}^{(i)}-\Delta t a_{ii}\mathbf{R}^{(1)}(\mathbf{w}^{(i)})-\Delta t^2\dot{a}_{ii}\mathbf{R}^{(2)}(\mathbf{w}^{(i)},\mathbf{R}^{(1)}(\mathbf{w}^{(i)})),\\ \mathbf{b}&=\mathbf{w}^n+\Delta t\sum_{j=1}^{i-1}a_{ij}\mathbf{R}^{(1)}(\mathbf{w}^{(j)})+\Delta t^2\sum_{j=1}^{i-1}\dot{a}_{ij}\mathbf{R}^{(2)}(\mathbf{w}^{(j)},\mathbf{R}^{(1)}(\mathbf{w}^{(j)})).\end{aligned}$$

More details over the non-linear formulation (28) can be found in [31, Sec. 3.2.1]. The linear system which arises in every Newton step is solved using the GMRES method. For faster convergence of the GMRES, application of a preconditioner is necessary.

[1] http://www.flexi-project.org

In [31, Sec. 4], the authors have compared the effect of different preconditioners on linear and non-linear iterations (see [31, Fig. 2 and Fig. 4]) and found that problem-tailored extended Block-Jacobi preconditioners $\mathbf{BJ^H_{ext}}$ and $\mathbf{BJ_{ext}}$ perform better than standard Block-Jacobi and ILU(0) preconditioners, in terms of the GMRES iterations per timestep and the wall-clock time. Consult [31, Fig. 1] for the pictorial representation of the mentioned precondition matrices. The authors employed a matrix-free approach with the $\mathbf{BJ_{ext}}$ preconditioner for their numerical investigations in [31], as it was efficient and effective. See the papers [6] and [16] for the matrix-free implementation. In this paper, we also use the matrix free approach with $\mathbf{BJ_{ext}}$ preconditioning as implemented in [31, Sec. 4.2].

4 Numerical investigations

As discussed in the previous section, we use Newton's method for solving the implicit equations and employ GMRES to solve the linear equations that arise in each Newton step. For a given Newton tolerance $\varepsilon_{\text{Newton}}$, the stopping criterion for the k^{th} Newton iteration is given by

$$\left\|\mathbf{G}(\mathbf{X}^k)\right\| < \varepsilon_{\text{Newton}} \cdot \left\|\mathbf{G}(\mathbf{X}^0)\right\|,$$

where $\left\|\mathbf{G}(\mathbf{X}^k)\right\|$ and $\left\|\mathbf{G}(\mathbf{X}^0)\right\|$ are the Euclidean norms of the k^{th} and initial residuals, respectively. For a specified tolerance for the linear solver, $\varepsilon_{\text{GMRES}}$, the stopping criterion for the GMRES iterations for the k^{th} Newton increment is

$$\left\|\mathbf{r}^k\right\| < \varepsilon_{\text{GMRES}} \cdot \left\|\mathbf{G}(\mathbf{X}^{k-1})\right\|,$$

where $\mathbf{r}^k$ is the residual of the linear equation, as indicated in [31, Eq. 18]. Test cases are described in the following sections.

4.1 Linear advection equation

Consider the two-dimensional linear advection equation

$$\mathbf{w}_t + \nabla \cdot (\mathbf{a}\mathbf{w}) = 0, \tag{29}$$

where $\mathbf{a} \in \mathbb{R}^2$ is a constant vector. We consider an exact solution of the equation (29) given by

$$\mathbf{w}(x,t) = \sin\left(\pi \sum_{j=1}^{2} (x_j - a_j t)\right), \tag{30}$$

with $\mathbf{a} = (0.3, 0.3)$ and $x \in \Omega = [-1,1]^2$ with periodic boundary conditions to analyze the SSP schemes. The classic upwind flux is chosen as the numerical flux for the DGSEM spatial discretization.

The L_2-error, average Newton iterations per timestep per stage and average GMRES iterations per Newton iteration for linear advection of a sine wave with $T_{\text{end}} = 0.8$ and $T_{\text{end}} = 1.6$ are presented in Fig. 5. The plots in Fig. 5 show that both the second-order and fourth-order SSP schemes exhibit the expected convergence for all the timestep sizes considered for $T_{\text{end}} = 0.8$ and $T_{\text{end}} = 1.6$. However, the third-order scheme encounters convergence issues for certain timesteps and requires greater number of Newton and GMRES iterations compared to the second and fourth-order schemes. Additionally, the severity of divergence in the third-order SSP scheme increases for the larger final time of $T_{\text{end}} = 1.6$.

A comprehensive study on the third-order SSP scheme is presented in Sec. 4.1.1 using the linear advection equation discretized using DGSEM. Additionally, a few strategies are discussed in Sec. 4.1.2 to control the divergence of the algorithm.

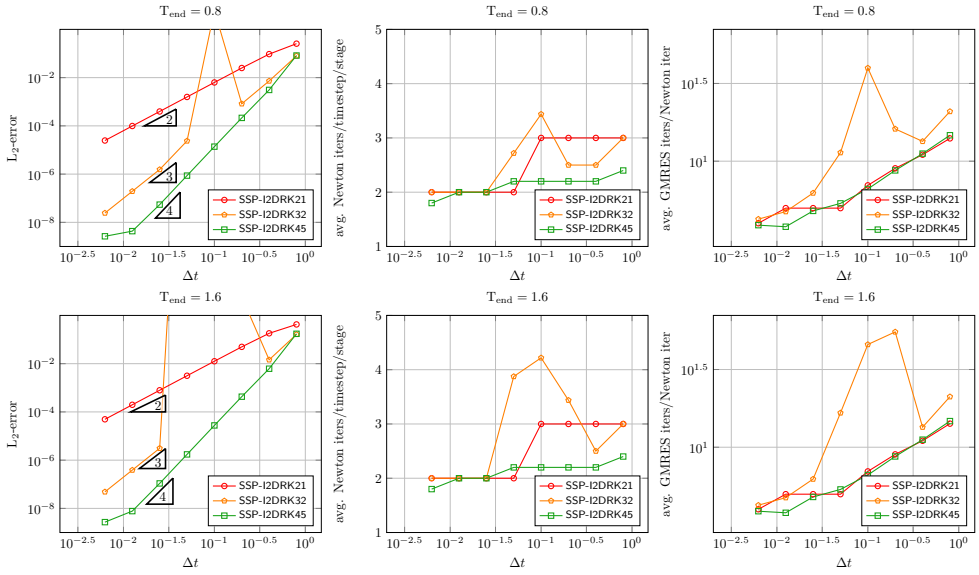


Figure 5: L_2-error (left), average Newton iterations per timestep per stage (middle) and average GMRES iterations per Newton iteration (right) for linear advection of a sine wave with $T_{end} = 0.8$ (top) and $T_{end} = 1.6$ (bottom). The initial condition is evaluated from (30) and simulated using the SSP schemes for different timesteps Δt. The spatial discretization is set to $N_E = 16 \times 16$ and $N_p = 5$. A tolerance of $\varepsilon_{Newton} = 10^{-6}$ and $\varepsilon_{GMRES} = 10^{-3}$ is used, with a limit of 10 Newton iterations per implicit solve and 2000 GMRES iterations per Newton iteration.

4.1.1 Analysis of the constraints in the third-order SSP scheme

As discussed in the previous section, the third-order SSP scheme exhibits convergence issues over a specific range of timestep sizes when applied to the linear advection of a sine wave. The initial condition was evaluated using equation (30), and the DGSEM spatial discretization was configured with $N_E = 16 \times 16$ elements and polynomial degree $N_p = 5$. To investigate the influence of the stability angle on the under-performance of the SSP-I2DRK32 scheme, we first consider a two-derivative, diagonally implicit, third-order Runge-Kutta scheme (I2DRK32(79.94°)), which possesses the same stability angle (79.94°) as the SSP-I2DRK32 scheme. The construction of the I2DRK32(79.94°) (44) is given in A.1.

In Fig. 6, we compare the convergence, average Newton and GMRES iterations required for implicit solves using the SSP-I2DRK32 scheme and the I2DRK32(79.94°) scheme. The results indicate that although the I2DRK32(79.94°) scheme has the same stability angle as the SSP-I2DRK32, it achieves the desired error reduction with a similar Newton and GMRES consumption as the second and fourth-order SSP schemes (see Fig. 5). Therefore, as a preliminary conclusion from Fig. 6, it is not the stability angle that affects the convergence of the SSP-I2DRK32 scheme. Hence, a detailed investigation on the stability regions is necessary.

The stability regions for the SSP-I2DRK32 and I2DRK32(79.94°) schemes are illustrated in Fig. 7. Although

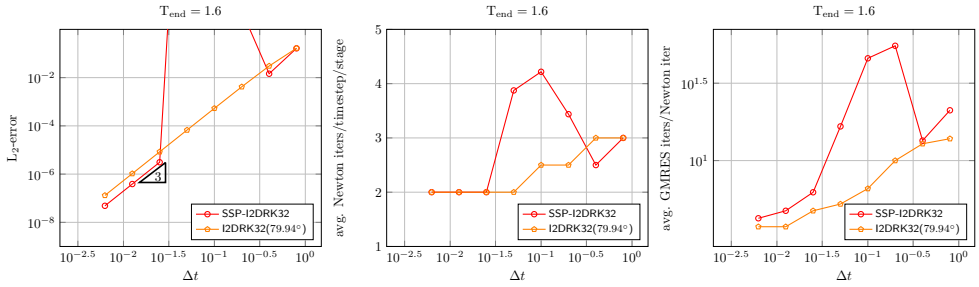


Figure 6: L_2-error (left), average Newton iterations per timestep per stage (middle) and average GMRES iterations per Newton iteration (right) for linear advection of a sine wave with $T_{end} = 1.6$. The initial condition is evaluated from (30). The spatial discretization is set to $N_E = 16 \times 16$ and $N_p = 5$. A tolerance of $\varepsilon_{Newton} = 10^{-6}$ and $\varepsilon_{GMRES} = 10^{-3}$ is used, with a limit of 10 Newton iterations per implicit solve and 2000 GMRES iterations per Newton iteration.

both schemes have the same stability angle of 79.94°, the locations of their unstable regions near the imaginary axis, which determine the stability angle, differ. To understand the influence of differences in unstable regions, we need

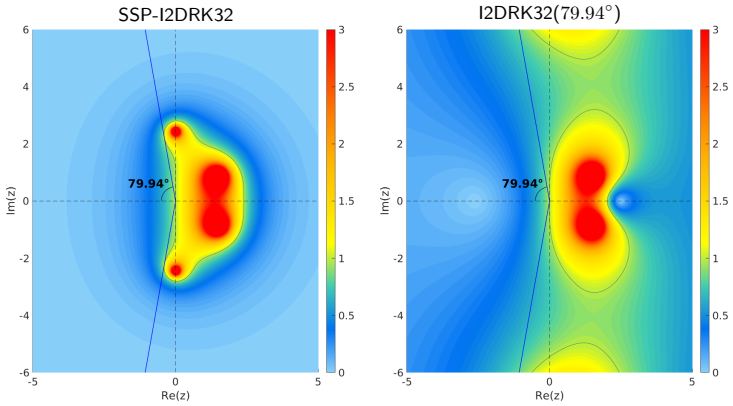


Figure 7: Stability region ($|\mathcal{S}(z)| < 1$) for SSP-I2DRK32 scheme (left) and I2DRK32(79.94°) scheme (right).

to examine the eigenvalues of the Jacobian of the spatially discretized part, $-\nabla \cdot \mathbf{F}$, from Eq. (1). Let Λ be the set of eigenvalues corresponding to the linear advection equation in Sec. 4.1.1 with discretization set to $N_E = 16 \times 16$ and $N_p = 5$. Note that for linear advection equations, the Jacobian corresponding to the spatially discretized ODE system is a constant matrix.

From Fig. 6, it can be observed that the range of timestep sizes that experience convergence issues is $\Delta t =$

$\{0.2, 0.1, 0.05\}$. Therefore, in Fig. 8 we plot the points

$$\mathbf{Z} = \Delta t \Lambda,$$

on the stability regions for both the SSP-I2DRK32 and I2DRK32(79.94°) schemes. It is clear that the values of $\mathbf{Z}$ fall into the unstable regions for all timestep sizes $\Delta t = \{0.2, 0.1, 0.05\}$ for the SSP-I2DRK32 scheme. Thus, the stability function (Eq. 11) generates values greater than one for the components associated with eigenvalues that are in the unstable regions. As the number of timesteps varies between 8 and 32, the error accumulates over these timesteps, leading to diverging solutions. The increase in the number of Newton and GMRES iterations observed in Figs. 5 and 7 might have resulted from several factors: error accumulation due to stability, a poor initial guess for the Newton method—since we used previous stage solutions as the initial guess—and higher residual norms, see Eq. (28).

In the case of the I2DRK32(79.94°) scheme, almost all values of $\mathbf{Z}$ fall into stable regimes for timestep sizes $\Delta t = \{0.1, 0.05\}$, and therefore, the desired convergence is achieved. Although a few eigenvalues fall into the unstable region for $\Delta t = 0.2$, the lower severity of the stability function ($|\mathcal{S}(z)| \lesssim 4$) and the reduced number of time steps help minimize error accumulation. In comparison, the absolute stability function ($|\mathcal{S}(z)|$) reaches values up to 40 in the unstable regimes for the SSP-I2DRK32 scheme.

Therefore, from the Figs. 7 and 8, it can be concluded that it is not the stability angle, but *the position of the unstable region near the imaginary axis* that contributes to the convergence issues for the SSP-I2DRK32 scheme.

4.1.2 A k-adaptive SSP scheme

As discussed in the previous section, the occurrence of values $\Delta t \Lambda$ in the unstable region leads to error accumulation, resulting in a solution divergence when using the SSP-I2DRK32 scheme. Fig. 9 shows that a convergent solution for a specified timestep Δt can be achieved by selecting an appropriate SSP-I2DRK32(k) or NSSP-I2DRK32(k) scheme. However, not all values of k yield convergent solutions. Therefore, we aim to explore the relationship between Δt and the free parameter k to develop a k-adaptive scheme, referred to as AD-SSP-I2DRK32 for the adaptive SSP scheme and AD-NSSP-I2DRK32 for the adaptive non-SSP scheme.

In [19], the authors analyzed the one-dimensional linear advection equation using a discontinuous Galerkin spatial discretization with upwinding and periodic boundary conditions. They found that the eigenvalues Λ of the discretization matrix can be bounded by a value that depends on the wave speed $a \in \mathbb{R}$, the spatial mesh size Δx, and the polynomial degree $\mathrm{N_p}$. They proved that the growth rate of the largest eigenvalues is less than $(\mathrm{N_p}+1)^2$, and more precisely, they conjectured that it is proportional to $(\mathrm{N_p}+1)^{1.75}$.

Taking into account the eigenvalue bounds described in [19], we modifiy it slightly for the two-dimensional linear advection equation (29), with wave velocity $\mathbf{a} \in \mathbb{R}^2$. Hence, we consider a bound on maximum absolute value of the eigenvalue given by

$$\Lambda_{\max} := \frac{\sum |\mathbf{a}_i|}{\Delta x} (\mathrm{N_p}+1)^{1.75}. \tag{31}$$

As seen in Sec. 2.2, the stability of the family of third-order schemes (24) has poles at $\pm\sqrt{6k}i$. Therefore by analyzing the distribution eigenvalues in Fig. 8, we consider the following assumption

$$\sqrt{6k} \geq \Delta t \Lambda_{\max} \tag{32}$$

to make sure that no values $\Delta t \Lambda$ is falling on the unstable region of the schemes (23). From Eq.(32), we derive

$$k_{\min} := \frac{(\Delta t \Lambda_{\max})^2}{6} \tag{33}$$

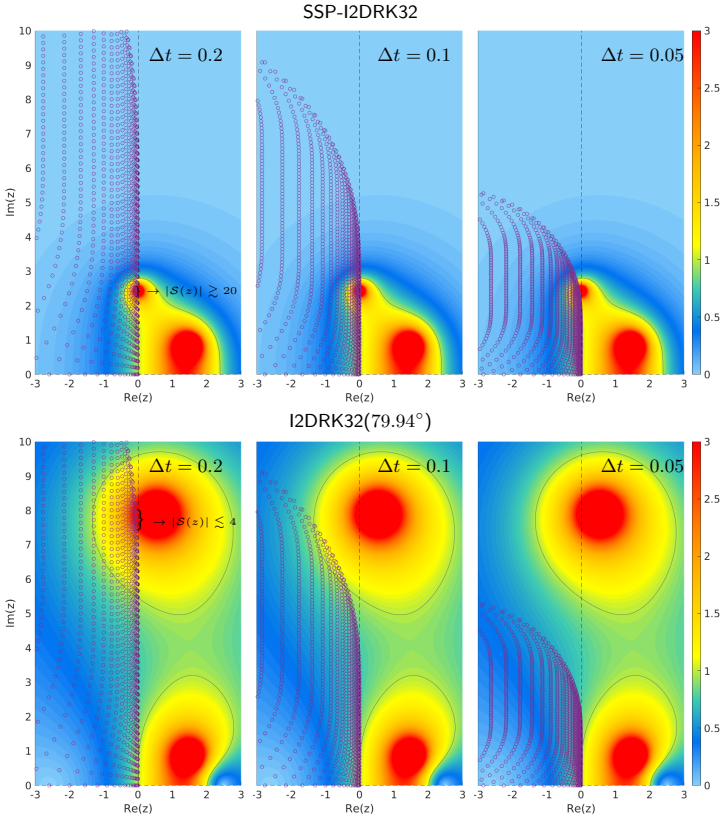


Figure 8: The stability regions for the SSP-I2DRK32 scheme and the I2DRK32(79.94°) scheme are given in the top and bottom rows, respectively. The values $\mathbf{Z} = \Delta t \Lambda$ are plotted (purple circles) for the eigenvalues Λ corresponding to the linear advection equation discussed in Sec. 4.1.1, with the discretization set to $\mathrm{N_E} = 16 \times 16$ and $\mathrm{N_p} = 5$. The points $\mathbf{Z} = \Delta t \Lambda$ are plotted for timestep sizes $\Delta t = \{0.2, 0.1, 0.05\}$. The real and imaginary axes are restricted to the ranges $-3 \leq \mathrm{Re}(z) \leq 3$ and $0 \leq \mathrm{Im}(z) \leq 10$.

which is the minimum k value needed for the scheme (23) so as to assure a converging solution for a given Δt. Hence the non-SSP adaptive scheme is given by

$$\text{AD-NSSP-I2DRK32} := \begin{cases} \text{SSP-I2DRK32}, & \Delta t \Lambda_{\max} \leq \sqrt{6} \\ \text{NSSP-I2DRK32}(k_{\min}), & \text{otherwise} \end{cases} . \tag{34}$$

Since the non-SSP does not guarantee $k_{\min} \leq 1$ for every Δt, we need a slightly different approach to develop an

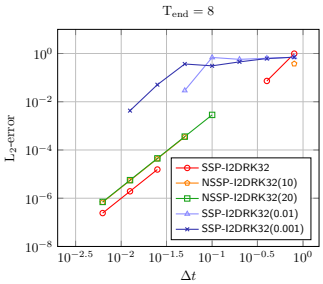


Figure 9: L_2-error for linear advection of a sine wave with $T_{end} = 8$ using various third-order schemes. The initial condition is evaluated from (30). The spatial discretization is set to $N_E = 16 \times 16$ and $N_p = 5$. A tolerance of $\varepsilon_{\text{Newton}} = 10^{-6}$ and $\varepsilon_{\text{GMRES}} = 10^{-3}$ is used, with a limit of 20 Newton iterations per implicit solve and 2000 GMRES iterations per Newton iteration. Note that the missing points on the plots indicate diverged solutions.

adaptive SSP scheme. Consider the absolute value of stability function for SSP-I2DRK32(k) along the imaginary axis

$$\begin{aligned} |\mathcal{S}(iy)| &= \left| \frac{18k + 3(1-k)(iy)^2}{(6k + (iy)^2)(3 - 3(iy) + (iy)^2)} \right|, \\ &= \left| \frac{18k + 3(k-1)y^2}{(6k - y^2)} \right| \frac{1}{\sqrt{y^4 + 3y^2 + 9}}. \end{aligned}$$

To find the points where the absolute value equals one on the imaginary axis, we set $|\mathcal{S}(iy)|^2$ to one, which yields

$$y^4 \left(y^4 + (3 - 12k)\, y^2 + \left(27k^2 - 18k\right) \right) = 0.$$

Analyzing the roots of this equation, we find that the real roots lie in the interval

$$\mathbf{I} := \left[-\sqrt{6k + \frac{3}{2}\left(\sqrt{4k^2+1} - 1\right)}, \sqrt{6k + \frac{3}{2}\left(\sqrt{4k^2+1} - 1\right)} \right].$$

For visual clarity, refer to the stability regions shown in Fig. 4. Let $l_\Lambda > 0$ be a value slightly less than the minimum modulus of eigenvalues in Λ that have a non-zero imaginary part and a real part very close to zero. For a clearer visual understanding of l_Λ, refer to Fig. 10. Therefore, to achieve a stable SSP solution for a given time step Δt, we require

$$\sqrt{6k + \frac{3}{2}\left(\sqrt{4k^2+1} - 1\right)} = \Delta t l_\Lambda. \tag{35}$$

The condition (35) ensures that the set $\mathbf{I}$ is contained within the interval $[-\Delta t l_\Lambda, \Delta t l_\Lambda]$. Consequently, this implies that there are no eigenvalues present in the unstable region. Hence, the stable SSP adaptive scheme is given by

$$\text{AD-SSP-I2DRK32} := \begin{cases} \text{SSP-I2DRK32}, & \Delta t \Lambda_{\max} \leq \sqrt{6} \\ \text{SSP-I2DRK32}(k_{\text{small}}), & \text{otherwise} \end{cases} \tag{36}$$

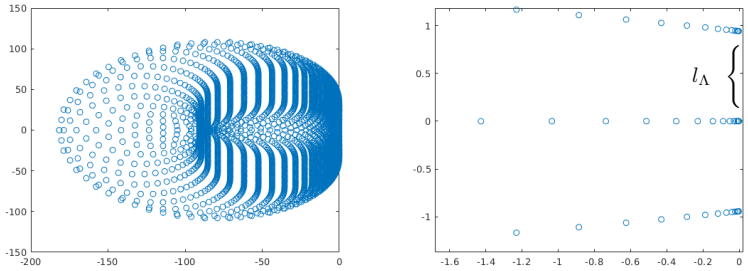


Figure 10: The eigenvalues Λ (shown on the left) corresponding to the linear advection equation discussed in Sec. 4.1.1, with a discretization set to $\mathrm{N_E} = 16 \times 16$ and $\mathrm{N_p} = 5$. On the right, a zoomed-in plot of the eigenvalues Λ is provided, with l_Λ marked on it.

with $k_{\text{small}} := \min(k_{\Delta t}, 1)$, where $k_{\Delta t}$ is the root of the equation defined above. Note that the value l_Λ must be known a priori. For the linear advection equation (29) with $\mathbf{a} = (0.3, 0.3)$, discretized using $\mathrm{N_E} = 16 \times 16$ and $\mathrm{N_p} = 5$, the value l_Λ is found to be less than 0.9, See Fig. 10. In a different spatial discretization setting, l_Λ can only be determined by analyzing the distribution of the eigenvalues, which is a current limitation. Further study and analysis are needed to find an empirical formulation for l_Λ, which is beyond the scope of the current paper.

In Fig. 11, the convergence plots for the linear advection equation are presented for the SSP-I2DRK32, AD-NSSP-I2DRK32 and AD-SSP-I2DRK32 schemes, using various final times $\mathrm{T_{end}}$. It is evident that the AD-NSSP-I2DRK32 and AD-SSP-I2DRK32 schemes provide stable solutions for all given timestep sizes Δt, while the SSP-I2DRK32 scheme diverges for the majority of timestep sizes. The adaptive schemes maintain stability across all final times, $\mathrm{T_{end}} = 8, 16, 32$. The AD-NSSP-I2DRK32 scheme exhibits the desired convergence order for all timestep sizes. Although the AD-SSP-I2DRK32 scheme yields stable solutions, there is a noticeable degradation in order for several timestep sizes. The error constant for the SSP-I2DRK32(k) scheme is given by

$$\left| \frac{1}{36k} - \frac{1}{24} \right|.$$

Thus, smaller values of k result in larger error constants, which lead to order reductions.

4.2 Euler equations

We consider the two dimensional Euler equations of gas dynamics

$$\mathbf{w}_t + \nabla \cdot \mathbf{F}^{\text{inv}}(\mathbf{w}) = 0, \tag{37}$$

with the state vector $\mathbf{w} = \begin{pmatrix} \rho \\ \rho \mathbf{v} \\ E \end{pmatrix}$. The flux function $\mathbf{F}^{\text{inv}}$ is given by

$$\mathbf{F}^{\text{inv}}(\mathbf{w}) = \begin{pmatrix} \rho \mathbf{v} \\ \rho \mathbf{v} \otimes \mathbf{v} + p \cdot \mathbf{Id} \\ \mathbf{v}(E + p) \end{pmatrix}, \tag{38}$$

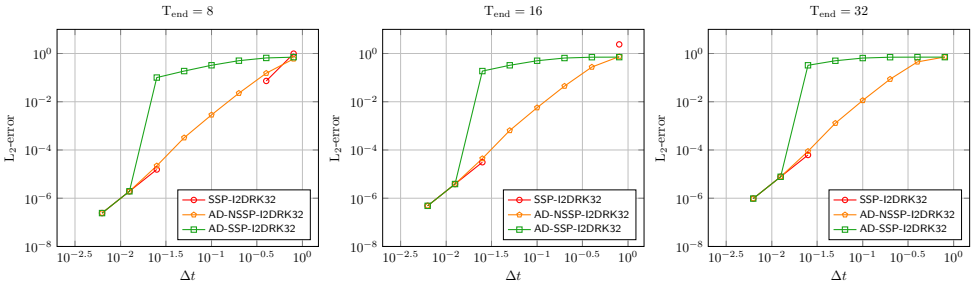


Figure 11: L_2-error for linear advection of a sine wave with $T_{end} = 8$ (left), $T_{end} = 16$ (middle) and $T_{end} = 8$ (right) using SSP-I2DRK32, AD-NSSP-I2DRK32 and AD-SSP-I2DRK32 schemes. The initial condition is evaluated from (30). The spatial discretization is set to $N_E = 16 \times 16$ and $N_p = 5$. A tolerance of $\varepsilon_{Newton} = 10^{-6}$ and $\varepsilon_{GMRES} = 10^{-3}$ is used, with a limit of 20 Newton iterations per implicit solve and 2000 GMRES iterations per Newton iteration. Note that the missing points on the plots indicate diverged solutions.

where ρ is density, $\mathbf{v} = (v_1, v_2)$ is velocity, E is total energy and p is pressure. Pressure is evaluated using the equation of state of a perfect gas,

$$p = (\gamma - 1)\Big(E - \frac{\rho}{2}\|\mathbf{v}\|_2^2\Big)$$

with the isentropic coefficient $\gamma = 1.4$. For the numerical validation of the Euler equations (37), we consider an extension of the linear advection equation (29) with a constant velocity $\mathbf{v} \equiv \mathbf{a} = (0.3, 0.3)^T$ and pressure $p = 1$. The initial conditions are therefore evaluated from the exact solution

$$\rho(x,t) = 1 + 0.3\sin\Big(\pi\sum_{j=1}^{2}(x_j - a_j t)\Big),\ \ \mathbf{v} = \mathbf{a},\ \ p = 1,\ \ x \in \Omega = [-1,1]^2, \tag{39}$$

and boundary conditions are taken to be periodic. The λ value for the global Lax-Friedrichs numerical flux (see [31, Eqs. (13) and (17)]) is chosen to be $\lambda = (1,1,1,1)$, as per the values given in [15].

The exact solution given in equation (39) serves as the reference solution for analyzing the L_2-error. The convergence results for the SSP-I2DRK21, SSP-I2DRK32, and SSP-I2DRK45 schemes are visualized in Fig. 12. For the SSP-I2DRK21 and SSP-I2DRK45 schemes, the errors decrease as the time step decreases, exhibiting the expected order of convergence. However, the SSP-I2DRK32 scheme diverges for $\Delta t \leq 0.2$. For the converged solutions, the number of Newton and GMRES iterations per implicit solve remains more or less the same.

In Fig. 13, we have presented the convergence plots, as well as the average Newton and GMRES iterations required for implicit solves, for the SSP-I2DRK32, AD-NSSP-I2DRK32 , and AD-SSP-I2DRK32 schemes. In contrast to the linear advection equations, the Euler equations have variable wave speeds given by $\{\mathbf{u}\cdot\mathbf{n}, \mathbf{u}\cdot\mathbf{n} \pm c\}$, where c is the speed of sound, defined as $c = \sqrt{\frac{\gamma p}{\rho}}$. As we consider Euler equations corresponding to the explicit solution (39), we can obtain an upper bound on the wave speeds in any direction $\mathbf{n}$. The upper bound is given by

$$\max\{\mathbf{u}\cdot\mathbf{n}, \mathbf{u}\cdot\mathbf{n} \pm c\} \leq |\mathbf{u}\cdot\mathbf{n}| + \sqrt{\frac{\gamma p}{\rho}} \leq 0.3 + \sqrt{\frac{1.4}{\min \rho}} = 0.3 + \sqrt{2}.$$

Consequently, we can evaluate $k_{\min}$ using the relation (31), replacing $\sum |\mathbf{a}_i| = 2(0.3 + \sqrt{2})$. The AD-NSSP-I2DRK32 shows desired convergence order for all the timestep sizes, with relatively few GMRES iterations compared to

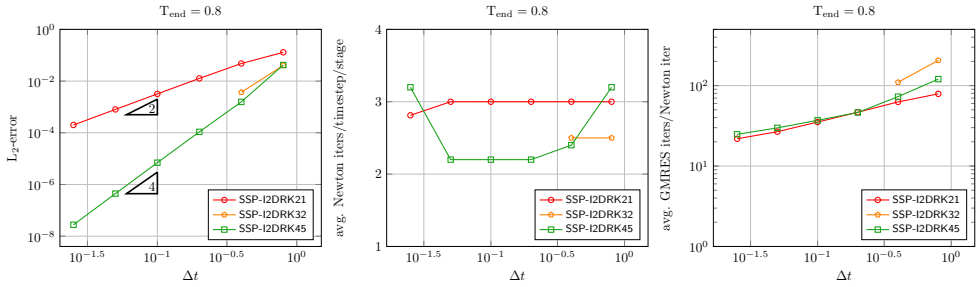


Figure 12: L_2-error (left), average Newton iterations per timestep per stage (middle) and average GMRES iterations per Newton iteration (right) for Euler equations with advection of density sine wave with $T_{end} = 0.8$ for the initial condition evaluated from the exact solution (39) using the SSP timestepping schemes. The plots are given for different timesteps Δt. The spatial discretization is set to $N_E = 16 \times 16$ and $N_p = 5$. A tolerance of $\varepsilon_{Newton} = 10^{-6}$ and $\varepsilon_{GMRES} = 10^{-3}$ is used, with a limit of 20 Newton iterations per implicit solve and 2000 GMRES iterations per Newton iteration. For the SSP-I2DRK45 scheme, the value of ε_{Newton} is set to 10^{-8} to avoid dominance of Newton error (only for $\Delta t = 0.025$). Note that the missing points on the plots indicate diverged solutions.

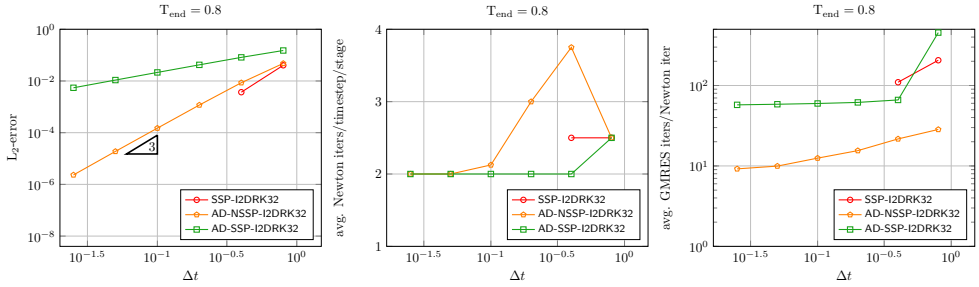


Figure 13: L_2-error (left), average Newton iterations per timestep per stage (middle) and average GMRES iterations per Newton iteration (right) for Euler equations with advection of density sine wave with $T_{end} = 0.8$ for the initial condition evaluated from the exact solution (39) using SSP-I2DRK32, AD-NSSP-I2DRK32 and AD-SSP-I2DRK32 schemes. The plots are given for different timesteps Δt. The spatial discretization is set to $N_E = 16 \times 16$ and $N_p = 5$. A tolerance of $\varepsilon_{Newton} = 10^{-4}$ and $\varepsilon_{GMRES} = 10^{-2}$ is used, with a limit of 20 Newton iterations per implicit solve and 2000 GMRES iterations per Newton iteration. Note that the missing points on the plots indicate diverged solutions.

SSP-I2DRK32 AD-SSP-I2DRK32 schemes. We consider the minimum of the wave speeds to compute l_Λ, which is necessary for the AD-SSP-I2DRK32 scheme. Using the explicit solution (39), we find that the minimum bound on the wave speeds

$$\min\{\mathbf{u} \cdot \mathbf{n}, \mathbf{u} \cdot \mathbf{n} \pm c\} = 0.3.$$

Since the minimum value of the wave speed is 0.3, we select the same value of $l_\Lambda = 0.9$ used for linear advection

equations. Similar to the findings in linear advection equations, the AD-SSP-I2DRK32 scheme provides stable solutions for the Euler equations with a reduction in error. However, the convergence order has decreased to almost one, likely due to the larger error constants.

4.3 Navier-Stokes equations

Next, we consider the two dimensional Navier-Stokes equations,

$$\mathbf{w}_t + \nabla \cdot \Big(\mathbf{F}^{\text{inv}}(\mathbf{w}) - \mathbf{F}^{\nu}(\mathbf{w}, \nabla \mathbf{w}) \Big) = 0, \tag{40}$$

with the state variables $\mathbf{w}$, the inviscid Euler flux $\mathbf{F}^{\text{inv}}$; and the viscous flux $\mathbf{F}^{\nu}$ given by

$$\mathbf{F}^{\nu}(\mathbf{w}, \nabla \mathbf{w}) = \begin{pmatrix} 0 \\ \tau \\ \tau.\mathbf{v} + \mathbf{q} \end{pmatrix} \tag{41}$$

where τ is the viscous tensor and $\mathbf{q}$ is the heat flux, given by

$$\tau := \mu\big(\nabla \mathbf{v} + (\nabla \mathbf{v})^T - \frac{2}{3}(\nabla \cdot \mathbf{v})\mathbf{Id}\big), \text{ and } \mathbf{q} := \lambda_T \nabla T, \tag{42}$$

respectively. The corresponding other parameters and constants used in the above equations are, dynamic viscosity μ, temperature T given by the ideal gas equation, thermal conductivity $\lambda_T = \frac{c_p \mu}{Pr}$, $Pr = 0.72$ is the fluid specific Prandtl number, specific heat capacity $c_p = \frac{R\gamma}{\gamma - 1}$ and the specific gas constant $R = \frac{1}{\gamma}$.

As the viscous flux $\mathbf{F}^{\nu}$ in the Navier-Stokes equations (40) depends on the state vector $\mathbf{w}$ as well as its gradient $\nabla \mathbf{w}$, it results into a second order PDE system. It is required to use an extended first order form for the equation (40) so as to utilize the fully discrete forms mentioned in the previous sections. Here, we use the BR2 lifting operator (see [1]) for the discretization of the second order equations. See [31, Sec. 5.1.1 - Sec. 5.1.3] for a detailed derivation.

For the numerical validation of the Navier-Stokes equations, we use the same set up as for the Euler equations. The viscosity is chosen to be $\mu = 10^{-3}$. In order to compute the L_2-error, a reference solution is computed via a fourth order explicit scheme [3] with a very small timestep $\Delta t = 10^{-6}$. The convergence results for the schemes are plotted in Fig. 14. The second order SSP scheme (SSP-I2DRK21) exhibit their desired order of convergence almost for every timesteps. However, the fourth order SSP scheme (SSP-I2DRK45) attains its actual order of convergence only for timesteps $\Delta t \leq 0.0125$. Even the second order schemes performs slightly better than the fourth order scheme for a couple of timesteps. The third order scheme (SSP-I2DRK32) diverges for timesteps $0.0125 \leq \Delta t \leq 0.1$ and provides stable solutions to other timesteps. Comparison of the linear and non-linear iterations for the Navier-Stokes equations for the three schemes in Fig. 14 shows a similar behavioral pattern as that of the Euler equations.

In Fig. 15, we present the convergence plots along with the average Newton and GMRES iterations required for implicit solves using the SSP-I2DRK32, AD-NSSP-I2DRK32 and AD-SSP-I2DRK32 schemes for the Navier-Stokes equations. Due to a lack of information regarding the distribution of eigenvalues for the spatially discretized advection-diffusion equation, we have chosen to use the minimum wave speed of $0.3 + \sqrt{2}$ from the hyperbolic part of the Navier-Stokes equation (40). Consequently, the value of $k_{\min}$ required for the AD-NSSP-I2DRK32 scheme is determined based on this wave speed bound of $0.3 + \sqrt{2}$, similar to the calculation used for the Euler equations.

The AD-NSSP-I2DRK32 scheme exhibits the desired convergence, requiring relatively few GMRES iterations compared to the SSP-I2DRK32 and AD-SSP-I2DRK32 schemes. For the AD-SSP-I2DRK32 scheme, we choose $l_\Lambda = 0.45$ heuristically. This scheme provides stable solutions for the Navier-Stokes equations with a reduction in error. The trend of order reduction observed in the Euler and linear advection equations is also evident here. However,

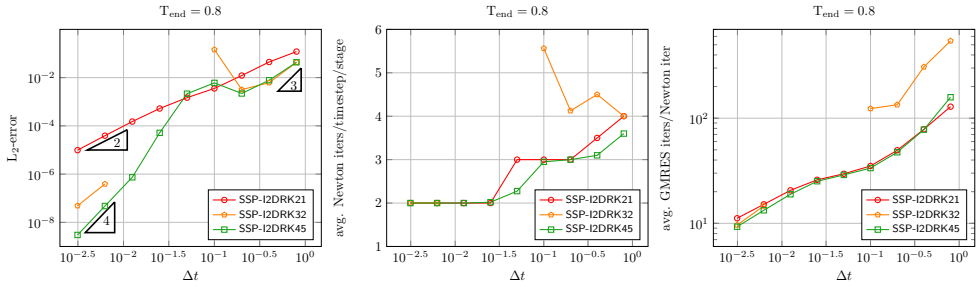


Figure 14: L_2-error (left), average Newton iterations per timestep per stage (middle) and average GMRES iterations per Newton iteration (right) for Navier-Stokes equations with advection and diffusion of density sine wave with $T_{end} = 0.8$ for the initial condition evaluated from the exact solution (39) using the SSP timestepping schemes. The plots are given for different timesteps Δt. The spatial discretization is set to $N_E = 16 \times 16$ and $N_p = 5$. A tolerance of $\varepsilon_{Newton} = 10^{-6}$ and $\varepsilon_{GMRES} = 10^{-3}$ is used, with a limit of 20 Newton iterations per implicit solve and 2000 GMRES iterations per Newton iteration. Note that the missing points on the plots indicate diverged solutions.

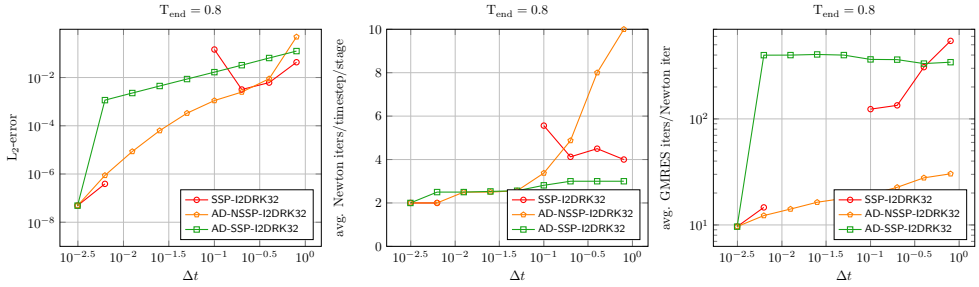


Figure 15: L_2-error (left), average Newton iterations per timestep per stage (middle) and average GMRES iterations per Newton iteration (right) for Navier-Stokes equations with advection and diffusion of density sine wave with $T_{end} = 0.8$ for the initial condition evaluated from the exact solution (39) using SSP-I2DRK32, AD-NSSP-I2DRK32 and AD-SSP-I2DRK32 schemes. The plots are given for different timesteps Δt. The spatial discretization is set to $N_E = 16 \times 16$ and $N_p = 5$. A tolerance of $\varepsilon_{Newton} = 10^{-6}$ and $\varepsilon_{GMRES} = 10^{-3}$ is used, with a limit of 20 Newton iterations per implicit solve and 2000 GMRES iterations per Newton iteration. Note that the missing points on the plots indicate diverged solutions.

the AD-SSP-I2DRK32 scheme requires more GMRES iterations per Newton iteration compared to the AD-NSSP-I2DRK32 scheme, which may be a consequence of smaller k values leading to a non-linear system.

For the fourth-order SSP scheme, we observed a significant deterioration in error reduction for timestep sizes ranging from 0.025 to 0.1. Since the SSP-I2DRK45 scheme is not A-stable, we further investigated stability in this context. We constructed a series of I2DRK45(α) schemes with varying stability angles α, maintaining the imaginary

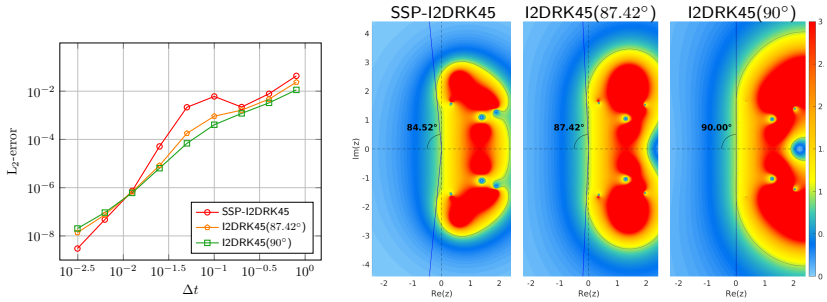


Figure 16: L_2-error (left) for Navier-Stokes equations with advection and diffusion of density sine wave with $T_{end} = 0.8$ for the initial condition evaluated from the exact solution (39) using SSP-I2DRK45, I2DRK45(87.42°) and I2DRK45(90°) schemes. The spatial discretization is set to $N_E = 16 \times 16$ and $N_p = 5$. A tolerance of $\varepsilon_{Newton} = 10^{-6}$ and $\varepsilon_{GMRES} = 10^{-3}$ is used, with a limit of 20 Newton iterations per implicit solve and 2000 GMRES iterations per Newton iteration. On the right, the stability angle and the stability region of the schemes SSP-I2DRK45, I2DRK45(87.42°) and I2DRK45(90°) are shown.

parts of the poles that cause instability at nearly the same positions. The detailed construction of the I2DRK45(α) schemes is provided in A.2.

In Fig. 16, we show the convergence plots for the SSP-I2DRK45, I2DRK45(87.42°), and I2DRK45(90°) schemes. It is clear that the I2DRK45(87.42°) and I2DRK45(90°) schemes show significant improvement compared to SSP-I2DRK45, consistently reducing errors for timestep sizes of $0.025 \leq \Delta t \leq 0.1$. Therefore, the results in Fig. 16 implies that the anomalies observed in the SSP-I2DRK45 scheme for the Navier-Stokes equations most likely stemmed from its instability.

5 Conclusion and outlook

In this work, we have analyzed the stability properties of the two-derivative SSP schemes [7], and have shown that the diagonally implicit two-derivative two-stage third order SSP can never be A-stable. SSP timestepping schemes [7] were implemented for Navier-Stokes equations in a DGSEM spatial framework. The complexity of the implementation of two-derivative Runge-Kutta scheme was outstripped by the introduction of the additional variable $\boldsymbol{\sigma}$ for the first order derivative as done in [26]. The stage values of non-linear equations were solved using Newton's method by employing GMRES method with a matrix-free approach on underlying preconditioned linear systems.

The second and fourth-order SSP schemes gave good convergence results on Euler and Navier-Stokes equations within a considerable number of Newton and GMRES iterations. The domination of temporal error over the spatial error for the chosen simulation setups was seen from the convergence plots. However, third-order SSP scheme encountered stability issues across a wide range of time step sizes due to the presence of poles of the stability function on the imaginary axis.

We have constructed and analyzed a family of A(α)-stable third-order schemes. The analysis reveals that the location of the unstable region in an A(α)-stable time-stepping scheme is more crucial than the stability angle. Based on the findings from the stability analysis, we have developed a stability adaptive third order SSP and non-SSP

schemes. The adaptive non-SSP scheme converged with the desired order for all test cases. Although the adaptive SSP scheme experienced an order reduction, it provided a stable solution in contrast to the original third-order SSP scheme. As we have devised a strategy to identify stable timestep sizes for A(α) schemes using eigenvalue bounds given in [19], it is possible to develop a hybrid SSP method that selectively incorporates higher-order SSP schemes to ensure both stability and overall accuracy.

While our analysis concentrates on a specific third-order method, the stabilizing techniques can be applied to a general class of two-derivative schemes. We also utilized these techniques to construct a series of A(α)-stable fourth-order schemes to investigate the anomalies observed in the fourth-order SSP scheme when applied to the Navier-Stokes equations. Numerical results showcased the importance of requiring A-stable time-stepping schemes when simulations are conducted over large final times, particularly for stiff systems.

There are four possible directions for future investigations. First, we are interested in implementing the asymptotic preserving IMEX timestepping schemes for low-Mach problems combined with DGSEM spatial discretization. Second, incorporation of the higher derivatives (order greater than two) using the Jacobian-free methods [4] and hence more flexibility can be achieved over the coefficients. Third, since there are higher-order strong stability preserving GLMs in literature [23, 21, 22], the implementation of these schemes into the DGSEM framework is subject to numerical investigations. Lastly, as there are only limited studies on strong stability preserving deferred correction schemes [20], investigating SSP-HBPC schemes is another potential direction for future research.

Acknowledgments

Arjun Thenery Manikantan was funded by the "Bijzonder Onderzoeksfonds" (BOF) from UHasselt - project no. BOF21KP12. Jonas Zeifang was funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) - project no. 457811052. We acknowledge the VSC (Flemish Supercomputer Center) for providing computing resources. The VSC is funded by the Research Foundation - Flanders (FWO) and the Flemish Government.

A Construction of reference schemes

A.1 The third-order I2DRK32(79.94°) scheme

The ξ-I2DRK32 schemes are constructed by keeping the implicit term of the first derivative (a_{11}) in the first stage as a free variable ξ. Using the order conditions [7, Sec. 2.1, p=1, 2, 3], the remaining coefficients were written as a function of ξ. Hence we have the Butcher coefficients for the two-derivative two-stage third order Runge-Kutta scheme ξ-I2DRK32

$$\mathbf{A} = \begin{bmatrix} \xi & 0 \\ 0 & 1 \end{bmatrix} \text{ and } \dot{\mathbf{A}} = \begin{bmatrix} -\frac{1}{6} & 0 \\ -\frac{1}{6(1-\xi)} & -\frac{1}{2} + \frac{1}{6(1-\xi)} \end{bmatrix}, \tag{43}$$

with $\xi \neq 1$. The ξ-I2DRK32 are A(α)-stable schemes.

The I2DRK32(79.94°) scheme is constructed to have the same stability angle as that of an SSP-I2DRK32 scheme. The scheme is obtained via slightly tuning the $\frac{1}{60}$-I2DRK32 scheme with a different coefficient $\dot{a}_{11} = -\frac{100}{6307}$. Hence the Butcher coefficients for two-derivative two-stage third order I2DRK32(79.94°) scheme is given by

$$\mathbf{A} = \begin{bmatrix} \frac{1}{60} & 0 \\ 0 & 1 \end{bmatrix} \text{ and } \dot{\mathbf{A}} = \begin{bmatrix} -\frac{100}{6307} & 0 \\ -\frac{10}{59} & -\frac{39}{118} \end{bmatrix}. \tag{44}$$

A.2 The fourth-order I2DRK45(α) schemes

For the SSP-I2DRK45 scheme, it was observed that the pole of the stability function

$$\mathcal{S}(z) = \frac{1 + \sum_{j=1}^{4} \left(a_{5j} z + \dot{a}_{5j} z^2\right) \mathcal{S}_j(z)}{\left(1 - a_{55} z - \dot{a}_{55} z^2\right)}$$

that is responsible for instability arise from the fourth stage

$$\frac{1}{1 - a_{44} z - \dot{a}_{44} z^2}. \tag{45}$$

To have a pole at the point $z = q \pm mi$ for Eq. (45), it is sufficient to choose the following coefficients

$$\begin{aligned} a_{44} &= \frac{2q}{q^2 + m^2}, \\ \dot{a}_{44} &= \frac{-1}{q^2 + m^2}. \end{aligned} \tag{46}$$

The poles of the Eq. (45) for the SSP-I2DRK45 scheme with coefficients $a_{44} = 0.191388711018110$ and $\dot{a}_{44} = -0.161628266349058$ is

$$q \pm mi \approx 0.592064480246234 \pm 2.4158841558212118i.$$

We construct I2DRK45(α) schemes by fixing $m \approx 2.4158841558212118$ and varying the real part q using the relationship given in Eq. (46). The coefficients a_{ii} and $\dot{a}_{ii}$ for $i \neq 4$ are taken directly from the Butcher tableau of the SSP-I2DRK45 scheme. The remaining lower diagonal coefficients a_{ij} and $\dot{a}_{ij}$ for $i < j$ are determined by solving order equations to achieve fourth-order accuracy.

The values of q corresponding to the schemes I2DRK45(87.42°) and I2DRK45(90°) are 1.4 and 2.5, respectively. The I2DRK45(α) scheme shows a trend of increasing stability angle α as q is increased. See, Fig. 16 for the stability regions.

Declarations

Conflict of interest The authors declare no competing interests

References

[1] F. Bassi, S. Rebay, G. Mariotti, S. Pedinotti, and M. Savini. A high-order accurate discontinuous Finite Element method for inviscid and viscous turbomachinery flows. *Proceedings of 2nd European Conference on Turbomachinery, Fluid Dynamics and Thermodynamics*, pages 99–108, 1997.

[2] Francesco Bassi, L. Botti, A. Colombo, Antonio Ghidoni, and F. Massa. Linearly implicit Rosenbrock-type Runge-Kutta schemes applied to the discontinuous Galerkin solution of compressible and incompressible unsteady flows. *Computers and Fluids*, 118:305–320, 09 2015.

[3] M.H. Carpenter and C.A. Kennedy. Fourth-order 2N-storage Runge-Kutta schemes. Technical report, NASA Langley Research Center, 1994.

[4] Jeremy Chouchoulis, Jochen Schütz, and Jonas Zeifang. Jacobian-free explicit multiderivative Runge–Kutta methods for hyperbolic conservation laws. *Journal of Scientific Computing*, 90(3):96, Feb 2022.

[5] Andrew J. Christlieb, Sigal Gottlieb, Zachary J. Grant, and David C Seal. Explicit strong stability preserving multistage two-derivative time-stepping schemes. *Journal of Scientific Computing*, 68:914–942, 2016.

[6] M. Franciolini, A. Crivellini, and A. Nigro. On the efficiency of a matrix-free linearly implicit time integration strategy for high-order discontinuous Galerkin solutions of incompressible turbulent flows. *Computers & Fluids*, 159:276–294, 2017.

[7] Sigal Gottlieb, Zachary J. Grant, Jingwei Hu, and Ruiwen Shu. High order strong stability preserving multiderivative implicit and IMEX Runge–Kutta methods with asymptotic preserving properties. *SIAM Journal on Numerical Analysis*, 60(1):423–449, 2022.

[8] Sigal Gottlieb, Chi-Wang Shu, and Eitan Tadmor. Strong stability-preserving high-order time discretization methods. *SIAM Review*, 43(1):89–112, 2001.

[9] Zachary Grant, Sigal Gottlieb, and David C. Seal. A strong stability preserving analysis for explicit multistage two-derivative time-stepping schemes based on Taylor series conditions. *Communications on Applied Mathematics and Computation*, 1(1):21–59, 2019.

[10] E. Hairer and G. Wanner. Multistep-multistage-multiderivative methods for ordinary differential equations. *Computing (Arch. Elektron. Rechnen)*, 11(3):287–303, 1973.

[11] E. Hairer and G. Wanner. *Solving ordinary differential equations II.* Springer Series in Computational Mathematics, 1991.

[12] Ralf Hartmann, Francesco Bassi, Igor Bosnyakov, Lorenzo Botti, Alessandro Colombo, Andrea Crivellini, Matteo Franciolini, Tobias Leicht, Emeric Martin, Francescocarlo Massa, et al. Implicit methods. In *TILDA: Towards Industrial LES/DNS in Aeronautics*, pages 11–59. Springer, 2021.

[13] F. Hindenlang, G. Gassner, C. Altmann, A. Beck, M. Staudenmaier, and C.-D. Munz. Explicit discontinuous Galerkin methods for unsteady problems. *Computers & Fluids*, 61:86–93, 2012.

[14] Alexander Jaust, Jochen Schütz, and David C. Seal. Implicit multistage two-derivative discontinuous Galerkin schemes for viscous conservation laws. *Journal of Scientific Computing*, 69:866–891, 2016.

[15] K. Kaiser and J. Schütz. A high-order method for weakly compressible flows. *Communications in Computational Physics*, 22(4):1150–1174, 2017.

[16] D. A. Knoll and D. E. Keyes. Jacobian-free Newton–Krylov methods: a survey of approaches and applications. *Journal of Computational Physics*, 193:357–397, 2004.

[17] D. A. Kopriva. *Implementing spectral methods for partial differential equations: Algorithms for scientists and engineers.* Springer Science & Business Media, 2009.

[18] Nico Krais, Andrea Beck, Thomas Bolemann, Hannes Frank, David Flad, Gregor Gassner, Florian Hindenlang, Malte Hoffmann, Thomas Kuhn, Matthias Sonntag, et al. FLEXI: A high order discontinuous Galerkin framework for hyperbolic–parabolic conservation laws. *Computers & Mathematics with Applications*, 81:186–219, 2021.

[19] Lilia Krivodonova and Ruibin Qin. An analysis of the spectrum of the discontinuous galerkin method. *Applied Numerical Mathematics*, 64:1–18, 2013.

[20] Yuan Liu, Chi-Wang Shu, and Mengping Zhang. Strong stability preserving property of the deferred correction time discretization. *Journal of Computational Mathematics*, 26(5):633–656, 2025/05/23/ 2008. Full publication date: September 2008.

[21] Afsaneh Moradi, Ali Abdi, and Javad Farzi. Strong stability preserving second derivative general linear methods with Runge–Kutta stability. *Journal of Scientific Computing*, 85(1):1, Sep 2020.

[22] Afsaneh Moradi, Ali Abdi, and Gholamreza Hojjati. Strong stability preserving implicit and implicit–explicit second derivative general linear methods with RK stability. *Computational and Applied Mathematics*, 41(4):135, Apr 2022.

[23] Afsaneh Moradi, Javad Farzi, and Ali Abdi. Strong stability preserving second derivative general linear methods. *Journal of Scientific Computing*, 81(1):392–435, Oct 2019.

[24] N. C. Nguyen, J. Peraire, and B. Cockburn. High-order implicit hybridizable discontinuous Galerkin methods for acoustics and elastodynamics. *Journal of Computational Physics*, 230:3695–3718, 2011.

[25] J. Schütz and D. Seal. An asymptotic preserving semi-implicit multiderivative solver. *Applied Numerical Mathematics*, 160:84–101, 2021.

[26] J. Schütz, D.C. Seal, and A. Jaust. Implicit multiderivative collocation solvers for linear partial differential equations with discontinuous Galerkin spatial discretizations. *Journal of Scientific Computing*, 73:1145–1163, 2017.

[27] Jochen Schütz, David C Seal, and Jonas Zeifang. Parallel-in-time high-order multiderivative IMEX solvers. *Journal of Scientific Computing*, 90(54):1–33, 2022.

[28] A. H. Stroud and D. D. Stancu. Quadrature formulas with multiple Gaussian nodes. *SIAM Journal on Numerical Analysis*, 2:129–143, 1965.

[29] P Turán. On the theory of the mechanical quadrature. *Acta Universitatis Szegediensis Acta Scientiarum Mathematicarum*, 12:30–37, 1950.

[30] J. Zeifang, J. Schütz, and D. Seal. Stability of implicit multiderivative deferred correction methods. *BIT Numerical Mathematics*, 2022.

[31] Jonas Zeifang and Jochen Schütz. Implicit two-derivative deferred correction time discretization for the discontinuous Galerkin method. *Journal of Computational Physics*, 464:111353, 2022.

[32] Jonas Zeifang, Arjun Thenery Manikantan, and Jochen Schütz. Time parallelism and Newton-adaptivity of the two-derivative deferred correction discontinuous Galerkin method. *Applied Mathematics and Computation*, 457:128198, 2023.

Paper IV:

A class of multirate multiderivative schemes

A class of multirate multiderivative schemes

Jochen Schütz[*], Ange Ishimwe[†⋆], Afsaneh Moradi[‡⋆], Arjun Thenery Manikantan[§⋆]

[⋆]These authors contributed equally to this work.

Abstract

In this work, we treat the numerical resolution of ordinary differential equations (ODEs) that contain both stiff and non-stiff terms, where these terms can be identified and separated, and where the stiff terms are 'easier' to evaluate than the non-stiff terms. In [Wensch, Knoth, Galant, BIT Numer Math 49 (2009), pp. 449–473], a class of so-called multirate schemes has been proposed to efficiently resolve said ODEs. Here, we extend this class of schemes by adding multiple temporal derivatives of the non-stiff part to the formulation. Order conditions and simplified order conditions of up to order four are derived in this work. Through this modification to the original algorithm, we can devise schemes with lesser stages at the same order. In particular, we devise a four-stage fourth-order scheme. The efficacy of the proposed methods is demonstrated through numerical experiments.

Keywords: Multirate; multiderivative; singularly perturbed ODE

MSC Classification: 65L05

1 Introduction

Ordinary differential equations serve as a fundamental tool for comprehending and exploring complex dynamics. Within these dynamics, phenomena can manifest across a spectrum of temporal scales. This implies that the characteristics of the phenomena may vary widely in terms of their time scales, with some exhibiting significantly smaller characteristic times compared to others. Such terms, known as stiff terms, pose unique challenges in computational analysis. To fix the ideas, let us consider the ODE

$$y'(t) = F(y) \equiv f(y) + g(y), \tag{1.1}$$

where g denotes the 'stiff' or fast part and f the 'non-stiff' or slow part. Obviously, the definition of stiffness is non-trivial [31]. For this work, we interpret it in the way that f should be treated with a 'large' timestep Δt, while g should be treated with a 'small' timestep $\Delta\tau < \Delta t$. This type of equations frequently arises in singularly perturbed equations such as, for example, the low-Mach Navier-Stokes equations [22] or relaxation problems [20]. Normally, because of the stiff term, a small time step has to be used for explicit schemes, leading to long simulations. However, in some cases, this stiff term g can be evaluated more easily than the non-stiff terms. For instance, this is the case for physical problems where one spatial dimension is significantly less important than the others. Atmospheric or hydrodynamic models are examples of this. In these models, the horizontal lengths of the domains are much larger than the vertical one. In this configuration, fast phenomena with small characteristics are computed using two

[*]jochen.schuetz@uhasselt.be (corresponding author), Faculty of Sciences & Data Science Institute, UHasselt, Belgium

[†]ange.ishimwe@uclouvain.be, Institute of Mechanics, Materials and Civil Engineering, Université catholique de Louvain, Belgium

[‡]afsaneh.moradi@ovgu.de, Institute of Analysis and Numerics, Otto von Guericke University Magdeburg, Germany

[§]arjun.thenerymanikantan@uhasselt.be, Faculty of Sciences & Data Science Institute, UHasselt, Belgium

dimensional terms, while the slow ones are represented by three dimensional ones. For atmospheric models [12, 32], waves with a characteristic speed typically of 300 m/s are much faster than velocity of the air. For ocean models [19], external gravity waves are the fast processes with a characteristic velocity that can be 100 times the one of others phenomena.

To address these potential challenges, high-order multirate schemes have been introduced in the seminal work [33]. They have been quite heavily extended, for a highly incomplete list of references, see [15, 24, 25, 28, 10, 1, 13, 14, 21] and the references therein. The underlying idea of these schemes is to compute the parts corresponding to f – the non-stiff part – with an explicit Runge-Kutta scheme of timestep Δt, and the parts corresponding to g using another explicit scheme of timestep $\Delta\tau < \Delta t$. Obviously, as for any coupled scheme, not any two schemes can be combined, but they need to fulfill certain coupling conditions to preserve the order of accuracy. Typically, these come in the form of algebraic requirements on the schemes' coefficients.

In this work, we extend the work of [33] to deal with multiderivative Runge-Kutta schemes for the slow part. Multiderivative schemes are schemes that do not only incorporate y', which is $f(y) + g(y)$ as visible from (1.1), but also (parts of) y'' and higher derivatives, see, e.g., [16, 7, 9, 30] and the references therein. In this work, we use those parts of the higher derivatives of y that are associated to f, i.e., we use for example the term $f(y)^{\{1\}} \equiv f'(y)F(y)$, see Rem. 1 for more information. This way, one arrives at higher order schemes with fewer stages than without this addition. In particular, we show a fourth-order multirate scheme with only four stages.

This article is structured as follows. It begins by introducing the multiderivative split-explicit time integrator method in Section 2, along with some illustrative and newly developed examples. The subsequent section presents the order conditions. Sections 4 and 5 explore the properties and numerical results of the presented schemes on various ODE and PDE. Finally, a conclusion is provided at the end of this article.

2 Split-explicit time integration methods

In this section, we give the general formulation of a multiderivative-multirate scheme; and then, in anticipation of the order conditions in Sec. 3, already show some novel schemes particularly developed for this work.

2.1 Formulation

Following [33], we define a multirate scheme in a semi-discrete way. While f is already discretized through an explicit multiderivative Runge-Kutta method, the equation associated to g is left continuous in time. In this work, we use multiderivative Runge-Kutta methods with up to four temporal derivatives in total, see also Rem. 1. In all what follows, $\Delta t > 0$ is a given time-step size that can be either constant throughout the algoritm or adaptive.

Definition 1. *(Multiderivative multirate scheme) For a given number of stages $s \in \mathbb{N}$ and a given number of derivatives $1 \le m \le 4$, we assume that the matrices α, γ, $\beta^{\{k\}} \in \mathbb{R}^{(s+1)\times(s+1)}$, $0 \le k \le m-1$, are given and are all strictly lower-triangular to obtain an explicit scheme. As in [33, eq. (2.3)], we assume the balancing condition*

$$d_i = \sum_{j=1}^{s+1} \beta_{ij}, \qquad 1 \le i \le s+1. \tag{2.1}$$

Then, the semi-discrete multiderivative multirate scheme is given by ($1 \le i \le s+1$)

$$\begin{aligned} Z_{ni}(0;\Delta t) &= y_n + \sum_{j=1}^{i-1} \alpha_{ij} \left(Y_{nj}(\Delta t) - y_n \right), \\ \partial_\tau Z_{ni}(\tau;\Delta t) &= \frac{1}{\Delta t} \sum_{j=1}^{i-1} \gamma_{ij} \left(Y_{nj}(\Delta t) - y_n \right) + d_i g(Z_{ni}(\tau;\Delta t)) \\ &\quad + \sum_{k=0}^{m-1} \sum_{j=1}^{i-1} \Delta t^k \beta_{ij}^{\{k\}} f^{\{k\}}(Y_{nj}(\Delta t)), \\ Y_{ni}(\Delta t) &= Z_{ni}(\Delta t;\Delta t); \end{aligned} \tag{2.2}$$

the update is then given by

$$y_{n+1} = Y_{n,(s+1)}(\Delta t).$$

For this definition, we have used

$$\begin{aligned} f^{\{0\}}(y) &:= f(y), \\ f^{\{1\}}(y) &:= f_y F, \\ f^{\{2\}}(y) &:= f_{yy}(F,F) + f_y(f_y + g_y)F, \\ f^{\{3\}}(y) &:= f_{yyy}(F,F,F) + 3f_{yy}((f_y + g_y)F, F) \\ &\quad + f_y \left(f_{yy}(F,F) + g_{yy}(F,F) \right) + f_y(f_y + g_y)(f_y + g_y)F. \end{aligned} \tag{2.3}$$

$f_y(y)$ denotes the Jacobian of f w.r.t. to y. $f_{yy}(\cdot,\cdot)$ and $f_{yyy}(\cdot,\cdot,\cdot)$ denote the second and third order derivatives of f w.r.t. to y, respectively. For an easier presentation, we have omitted the argument (y). The colors introduced here will also be used in the presentation of the order conditions.

Remark 1. *Let us make some comments here:*

- *Please note that for the exact solution y to (1.1) and $k \in \mathbb{N}^{\ge 0}$, there holds*

$$\frac{\mathrm{d}^k}{\mathrm{d}t^k} f(y(t)) = f^{\{k\}}(y(t)).$$

  If g was identically zero, this would mean $y''(t) = f^{\{1\}}(y(t))$, $y'''(t) = f^{\{2\}}(y(t))$ and so on, hence the term multiderivative. While a two-derivative scheme (i.e., using $f^{\{1\}}(y)$) has been proven to be of relevance in practice, see, e.g, [34], the higher-derivative schemes are included for scientific curiosity. If they are to be used, the higher derivatives should typically be approximated by some sort of finite difference, see [3, 8, 35, 11]

- *The scheme from Def. 1 is still continuous in the pseudo-time τ. In the numerical experiments, this will be discretized using an explicit* one-derivative *Runge-Kutta method with a finer timestep-size $\Delta\tau \le \Delta t$. The reason that we do not use a multi-derivative Runge-Kutta method lies in the fact that for $g^{\{k\}}$, $k \ge 1$, one would need evaluations of f and its derivatives again. This would be unattractive given that we consider f to be more difficult to evaluate than g.*

- *The magenta terms in Def. 1 distinguish the algorithm that we propose here from the one in [33]. These extra terms will be the reason that we can construct schemes with less stages while keeping the order of convergence.*

As in [33, Thm. 2.1], Def. 1 yields a multiderivative Runge-Kutta method in the case that $g \equiv 0$, with Butcher tableaux

$$A^{\{k\}} := R\beta^{\{k\}}, \qquad 0 \le k \le m-1, \tag{2.4}$$

where

$$R := (\mathrm{Id} - \alpha - \gamma)^{-1}. \tag{2.5}$$

Typically, we will indicate schemes by giving A, α and γ rather than the β.

2.2 Schemes

Based on the order conditions to be shown in Sec. 3, we have developed a couple of novel schemes. In particular, here, we present

- a class of two-stage, third-order two-derivative schemes depending on a parameter ξ, called Mul3s2m2(ξ)
- a four-stage, fourth-order two-derivative scheme, called Mul4s4m2,
- a three-stage, fourth-order three-derivative scheme, called Mul4s3m3.

Third order two-stage two-derivative scheme Mul3s2m2(ξ) Let $\xi \in \mathbb{R}^{\neq \frac{-1}{6}}$ be a free parameter and define

$$c_1 := 2\xi + \frac{1}{3}, \qquad \dot{b}_1 = \frac{3\xi}{6\xi+1}, \qquad \dot{b}_2 = \frac{1/2}{6\xi+1}.$$

Then, the Butcher-tableaux of the schemes are defined by

$$A^{\{0\}} = \begin{pmatrix} 0 & & \\ c_1 & 0 & \\ 1 & 0 & 0 \end{pmatrix}, \qquad A^{\{1\}} = \begin{pmatrix} 0 & & \\ \xi & 0 & \\ \dot{b}_1 & \dot{b}_2 & 0 \end{pmatrix}. \tag{2.6}$$

If we set $\alpha = \begin{pmatrix} 0 & & \\ 0 & 0 & \\ 0 & 1 & 0 \end{pmatrix}$, and $\gamma = 0$, then it can be deduced from the order conditions to be presented in Sec. 3 that the multirate scheme with $\beta = R^{-1}A$ and $\dot{\beta} = R^{-1}\dot{A}$ is third-order convergent.

Fourth order four-stage two-derivative scheme Mul4s4m2 This scheme, and also the next scheme, have been developed by solving the simplified order conditions, see Thm. 2, numerically in Matlab with the help of fsolve.

$$A^{\{0\}} = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 \\ 0.644528962237943 & 0 & 0 & 0 & 0 \\ 0 & 0.793930203564751 & 0 & 0 & 0 \\ 0 & 0.651368938661906 & 0.234630026296709 & 0 & 0 \\ 0.368783295148086 & 0.361990106948867 & 0.147750352586748 & 0.121476245316299 & 0 \end{pmatrix}, \tag{2.7}$$

$$A^{\{1\}} = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 \\ 0.019204137009700 & 0 & 0 & 0 & 0 \\ 0 & 1.074197913721907 & 0 & 0 & 0 \\ 0 & -0.328894199359934 & -0.868581157332243 & 0 & 0 \\ 0.046047593117438 & -0.004291996212853 & 0 & 0 & 0 \end{pmatrix} \tag{2.8}$$

α is set to $\alpha = \begin{pmatrix} 0 & & & \\ 0 & 0 & & \\ 0 & 1 & 0 & \\ 0 & 0 & 1 & 0 \end{pmatrix}$, and $\gamma = 0$.

Fourth-order three-stage three-derivative scheme Mul4s3m3

$$A^{\{0\}} = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 1.009283680769299 & 0 & 0 & 0 \\ 3.720878355840538 & -2.718495837492225 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{pmatrix}$$

$$A^{\{1\}} = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0.253296309203584 & 0 & 0 & 0 \\ -3.356309948891324 & -2.584529228478059 & 0 & 0 \\ -0.331202647364177 & 0.855031437707487 & -0.023828790343315 & 0 \end{pmatrix} \tag{2.9}$$

$$A^{\{2\}} = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0.075395834891222 & 0 & 0 & 0 \\ 0.989887257282753 & 1.428802815206199 & 0 & 0 \\ -0.297547643762234 & -0.882455016628254 & 0.507585613307806 & 0 \end{pmatrix}$$

α and γ are set in exactly the same way as for the scheme before. For comparison, please note that the fourth-order one-derivative scheme presented in [5] has five stages and uses a γ that is different from zero.

3 Order conditions

In this section, we investigate the order of consistency of the scheme presented in Def. 1. Although other approaches are possible as well, e.g., via generalized additive Runge-Kutta methods [15], we rely very heavily on the approach presented in [33] and later extended in [5]. As only the magenta terms in Def. 1 change in comparison to [33, 5], it is clear that the order conditions also look rather similar, with the addition of the multiderivative order conditions. For comparison, order conditions of up to order four for explicit two derivative schemes are, e.g., given in [9], for three derivative schemes in [26]; and for one-derivative multirate schemes in [5].

Theorem 1 (Order conditions for scheme in Def. 1)**.** *Define D as a diagonal matrix with $D_{ii} = d_i$; and $b^{\{k\}} \in \mathbb{R}^{1\times(s+1)}$ as the last row of $A^{\{k\}}$. $\mathbb{1} \in \mathbb{R}^{(s+1)\times 1}$ denotes a columvector filled with ones. As usual for Runge-Kutta schemes, define*

$$c_i^{\{k\}} = \sum_j A_{ij}^{\{k\}}.$$

For notational simplicity, we will set $b \equiv b^{\{0\}}$ and $c \equiv c^{\{0\}}$. Further, we define

$$\tilde{c} = \alpha c, \qquad \tilde{b} = (RD)_{(s+1),\text{-}} \in \mathbb{R}^{1\times(s+1)},$$

i.e., $\tilde{b}$ denotes the last row of the matrix RD.
Then, the scheme in Def. 1 is of order $1 \leq q \leq 4$, if it fulfills the following order conditions up to order q:

Order	*Algebraic condition*			*Elem. diff.*[1]
1	$b\mathbb{1}$	$=$	1	F
2	$bc + b^{\{1\}}\mathbb{1}$	$=$	$\frac{1}{2}$	$f_y F$
	$\tilde{b}(c+\tilde{c})$	$=$	1	$g_y F$
3	$bc^2 + 2b^{\{1\}}c + 2b^{\{2\}}\mathbb{1}$	$=$	$\frac{1}{3}$	$f_{yy}(F,F)$
	$bAc + b^{\{1\}}c + bc^{\{1\}} + b^{\{2\}}\mathbb{1}$	$=$	$\frac{1}{6}$	$f_y f_y F$
	$\tilde{b}(Id+\alpha)(Ac + c^{\{1\}})$	$=$	$\frac{1}{3}$	$g_y f_y F$
	$3\tilde{b}(\alpha+\gamma/2)RD(c+\tilde{c}) + \tilde{b}D(c+2\tilde{c})$	$=$	1	$g_y g_y F$
	$bRD(c+\tilde{c}) + 2b^{\{1\}}c + 2b^{\{2\}}\mathbb{1}$	$=$	$\frac{1}{3}$	$f_y g_y F$
	$\tilde{b}(c^2 + \tilde{c}^2 + c\cdot\tilde{c})$	$=$	1	$g_{yy}(F,F)$
4	$4bc^3 + 12b^{\{1\}}c^2 + 24b^{\{2\}}c + 24b^{\{3\}}\mathbb{1}$	$=$	*1*	$f_{yyy}(F,F,F)$
	$12bRD(c+\tilde{c})c + 12b^{\{1\}}(2c^2 + RD(c+\tilde{c}))$			
	$+\ 72b^{\{2\}}c + 72b^{\{3\}}\mathbb{1}$	$=$	*3*	$f_{yy}(g_y F, F)$
	$24bcAc + 24bcc^{\{1\}} + 24b^{\{1\}}Ac$			
	$+24b^{\{1\}}c^{\{1\}} + 24b^{\{1\}}c^2 + 72b^{\{2\}}c + 72b^{\{3\}}\mathbb{1}$	$=$	*3*	$f_{yy}(F, f_y F)$
	$12bAc^2 + 24bA^{\{1\}}c + 12b^{\{1\}}c^2 + 24b^{\{2\}}c + 24bc^{\{2\}} + 24b^{\{3\}}\mathbb{1}$	$=$	*1*	$f_y f_{yy}(F,F)$
	$24bA^2c + 24bAc^{\{1\}} + 24bA^{\{1\}}c + 24b^{\{1\}}Ac + 24b^{\{1\}}c^{\{1\}}$			
	$+\ 24bc^{\{2\}} + 24b^{\{2\}}c + 24b^{\{3\}}\mathbb{1}$	$=$	*1*	$f_y f_y f_y F$
	$12bARD(c+\tilde{c}) + 24bA^{\{1\}}c + 12b^{\{1\}}RD(c+\tilde{c})$			
	$+24bc^{\{2\}} + 24b^{\{2\}}c + 24b^{\{3\}}\mathbb{1}$	$=$	*1*	$f_y f_y (g_y) F$
	$4bRD(c^2 + \tilde{c}^2 + c\tilde{c}) + 12b^{\{1\}}c^2 + 24b^{\{2\}}c + 24b^{\{3\}}\mathbb{1}$	$=$	*1*	$f_y g_{yy}(F,F)$
	$12bRD(Id+\alpha)(Ac + c^{\{1\}})$			
	$+24b^{\{1\}}(Ac + c^{\{1\}}) + 24b^{\{2\}}c + 24b^{\{3\}}\mathbb{1}$	$=$	*1*	$f_y g_y f_y F$
	$4bRD\left(3(\alpha+\frac{\gamma}{2})RD(c+\tilde{c}) + D(c+2\tilde{c})\right)$			
	$+12b^{\{1\}}RD(c+\tilde{c}) + 24b^{\{2\}}c + 24b^{\{3\}}\mathbb{1}$	$=$	*1*	$f_y g_y g_y F$
	$\tilde{b}(c^3 + c^2\tilde{c} + c\tilde{c}^2 + \tilde{c}^3)$	$=$	*1*	$g_{yyy}(F,F,F)$
	$\tilde{b}\left((4+8\alpha)\tilde{c} + (8+4\alpha)c\right) \odot (Ac + c^{\{1\}})$	$=$	*3*	$g_{yy}(F, f_y F)$
	$6\tilde{b}(Id+\alpha)(Ac^2 + 2A^{\{1\}}c + 2c^{\{2\}})$	$=$	*1*	$g_y f_{yy}(F,F)$
	$6\tilde{b}(Id+\alpha)(2AAc + 2Ac^{\{1\}} + 2A^{\{1\}}c + 2c^{\{2\}})$	$=$	*1*	$g_y f_y f_y F$
	$\tilde{b}\left(6(Id+\alpha)(ARD(c+\tilde{c}) + 2A^{\{1\}}c + 2c^{\{2\}})\right)$	$=$	*1*	$g_y f_y g_y F$
	$\tilde{b}\left(6\alpha RD(c+\tilde{c})^2 + \gamma RD(c+\tilde{c})(4c+2\tilde{c}) + 6D\tilde{c}c + 3Dc^2 + 3D\tilde{c}^2\right)$	$=$	*3*	$g_{yy}(F, g_y F)$
	$\tilde{b}\left(4(\alpha+\frac{\gamma}{2})RD(c^2 + \tilde{c}^2 + c\tilde{c}) + 3D\tilde{c}^2 + 2D\tilde{c}c + Dc^2\right)$	$=$	*1*	$g_y g_{yy}(F,F)$
	$\tilde{b}\left(12(\alpha+\frac{\gamma}{2})RD(Id+\alpha)(Ac + c^{\{1\}}) + D(4Id+8\alpha)(Ac + c^{\{1\}})\right)$	$=$	*1*	$g_y g_y f_y F$
	$\tilde{b}\left(4(\alpha+\frac{\gamma}{2})RD(3(\alpha+\frac{\gamma}{2})RD(c+\tilde{c}) + D(c+2\tilde{c}))\right.$			
	$\left.+D(6\alpha+2\gamma)RD(c+\tilde{c}) + D^2(c+3\tilde{c})\right)$	$=$	*1*	$g_y g_y g_y F$

Proof. The proof of these order conditions is in big parts very similar to the proof in [33]; it can be found in the appendix. □

[1] Elementary differential

The order conditions presented here are rather lengthy; it is 27 conditions for order four. It has been realized in [33] that some older methods can be cast into the framework of Def. 1 with a particular definition of α and γ, that is called 'Property A' in [33] and (slightly specialized) 'MIS-KW' in [5]. This property A is also very helpful in our setting:

Theorem 2. *For meaning of the variables, consult Thm. 1. Let $\gamma = 0$, and let α be such that $\alpha_{ij} \in \{0, 1\}$, and $\sum_{j=1}^{s+1} \alpha_{ij} \in \{0, 1\}$, see [33, Def. 5.1]. Then, the scheme in Def. 1 is of order $1 \le q \le 4$, if it fulfills the following order conditions up to order q:*

Order	*Algebraic condition*			*Elem. diff.*
1	$b\mathbb{1}$	$=$	1	F
2	$bc + b^{\{1\}}\mathbb{1}$	$=$	$\frac{1}{2}$	$f_y F$
3	$bc^2 + 2b^{\{1\}}c + 2b^{\{2\}}\mathbb{1}$	$=$	$\frac{1}{3}$	$f_{yy}(F, F)$
	$bAc + b^{\{1\}}c + bc^{\{1\}} + b^{\{2\}}\mathbb{1}$	$=$	$\frac{1}{6}$	$f_y f_y F$
	$\tilde{b}(Id + \alpha)(Ac + c^{\{1\}})$	$=$	$\frac{1}{3}$	$g_y f_y F$
4	$4bc^3 + 12b^{\{1\}}c^2 + 24b^{\{2\}}c + 24b^{\{3\}}\mathbb{1}$	$=$	1	$f_{yyy}(F, F, F)$
	$24bcAc + 24bcc^{\{1\}} + 24b^{\{1\}}Ac$			
	$+24b^{\{1\}}c^{\{1\}} + 24b^{\{1\}}c^2 + 72b^{\{2\}}c + 72b^{\{3\}}\mathbb{1}$	$=$	3	$f_{yy}(F, f_y F)$
	$12bAc^2 + 24bA^{\{1\}}c + 12b^{\{1\}}c^2$			
	$+24b^{\{2\}}c + 24bc^{\{2\}} + 24b^{\{3\}}\mathbb{1}$	$=$	1	$f_y f_{yy}(F, F)$
	$24bA^2c + 24bAc^{\{1\}} + 24bA^{\{1\}}c + 24b^{\{1\}}Ac + 24b^{\{1\}}c^{\{1\}}$			
	$+24b^{\{2\}}c + 24bc^{\{2\}} + 24b^{\{3\}}\mathbb{1}$	$=$	1	$f_y f_y f_y F$
	$12bRD(Id + \alpha)(Ac + c^{\{1\}}) + 24b^{\{1\}}(Ac + c^{\{1\}})$			
	$+24b^{\{2\}}c + 24b^{\{3\}}\mathbb{1}$	$=$	1	$f_y g_y f_y F$
	$\tilde{b}\left((4Id + 8\alpha)\tilde{c} + (8Id + 4\alpha)c\right) \odot (Ac + c^{\{1\}})$	$=$	3	$g_{yy}(F, f_y F)$
	$6\tilde{b}(Id + \alpha)(Ac^2 + 2A^{\{1\}}c + 2c^{\{2\}})$	$=$	1	$g_y f_{yy}(F, F)$
	$6\tilde{b}\left((Id + \alpha)(2AAc + 2Ac^{\{1\}} + 2A^{\{1\}}c + 2c^{\{2\}}\right)$	$=$	1	$g_y f_y f_y F$
	$\tilde{b}\left(12\alpha RD(I + \alpha)(Ac + c^{\{1\}}) + D(4 + 8\alpha)(Ac + c^{\{1\}})\right)$	$=$	1	$g_y g_y f_y F$

Proof. The proof is very similar to the one of [33, Thm. 5.1]. In particular, one can use the same identities

$$\tilde{c}^k = \alpha c^k, \quad k \ge 1, \qquad D = C - \tilde{C},$$

where C and $\tilde{C}$ are diagonal matrices with c and $\tilde{c}$, respectively, on the diagonals. For the conditions up to order three, the procedure is the same as in [33], with the obvious incorporation of the multiderivative parts. For order four then, it is straightforward to see that the conditions corresponding to the elementary differentials $g_{yyy}(F, F, F)$, $g_y g_{yy}(F, F)$, $g_{yy}(g_y F, F)$ and $g_y g_y g_y F$ are equivalent to $c_{s+1} = 1$ which, then again, is equivalent to $b\mathbb{1} = 1$. The conditions corresponding to the elementary differentials $f_{yy}g_y F$, $f_y g_{yy}(F, F)$ and $f_y g_y g_y F$ are equivalent to the one corresponding to $f_{yyy}(F, F, F)$. The ones corresponding to $f_y f_y g_y F$ are equivalent to the ones of $f_y f_{yy}$; and the ones of $g_y f_y g_y F$ are equivalent to the ones of $g_y f_{yy}(F, F)$. □

Lemma 1. *Under the assumptions of Thm. 2, there is no two stage scheme ($s = 2$) of order four, even not with higher derivatives.*

Proof. The conditions corresponding to $g_{yy}(f_y F, F)$ and $g_y g_y f_y F$ cannot be fulfilled simultaneously under these

assumptions, as they will lead to

$$A_{31}^{\{1\}} + A_{32}^{\{1\}} - A_{21}^{\{0\}}(A_{31}^{\{0\}} - 1) = \frac{3}{8},$$
$$A_{31}^{\{1\}} + A_{32}^{\{1\}} - A_{21}^{\{0\}}(A_{31}^{\{0\}} - 1) = \frac{1}{4}$$

which obviously cannot be fulfilled simultaneously. □

4 Stability regions

To analyze the stability of the methods, we apply, as is customary for IMEX schemes [2], our developed schemes to the model equation

$$y'(t) = i\mu y(t) + \lambda y(t),$$

where i denotes the imaginary unit and $\mu \in \mathbb{R}$, $\lambda \in \mathbb{R}$ are assumed to be constants. This equation is a prototype of a convection-diffusion equation, as the eigenvalues of a pure convection operator lie on the imaginary axes, and those of a pure diffusion operator on the (negative) real axis. In [33, Sec. 5.1], a similar problem has been considered.

The term $f(y) := i\mu y$ is assumed to be the slow part; the second term $g(y) := \lambda y$ the fast part. We follow the steps of [33] and observe that for some given constant c, the solution to $y' = c + \lambda y$ is given by

$$y(t) = e^{\lambda t} y_0 + t\phi(\lambda t)c,$$

with $\phi(z) := \frac{e^z - 1}{z}$. Applying this to (2.2), one finds that

$$\begin{aligned} Y_{ni}(\Delta t) =& e^{d_i \lambda \Delta t} \left(y_n + \sum_{j=1}^{i-1} \alpha_{ij}(Y_{nj}(\Delta t) - y_n) \right) \\ &+ \phi(d_i \lambda \Delta t) \left(\sum_{j=1}^{i-1} \gamma_{ij} \left(Y_{nj}(\Delta t) - y_n \right) + \sum_{k=0}^{m-1} \sum_{j=1}^{i-1} \Delta t^{k+1} \beta_{ij}^{\{k\}} (i\mu)^{k+1} Y_{nj}(\Delta t) \right). \end{aligned}$$

Recursively unfolding this to $Y_{n,s+1}$ yields the stability function $R(\tilde{\lambda}, \tilde{\mu})$ as a function of $\tilde{\mu} := i\Delta t\mu$ and $\tilde{\lambda} := \Delta t\lambda$.

For the schemes presented in this work, we have plotted the stability regions in Fig. 1. The first to notice is that the stability regions incorporate the whole negative real axis. This means that the splitting method is consistent so that if one solved exactly the fast part, it does not give rise to any stability problems. The second observation is that the stability region is smaller for higher-order methods. Lastly, for the same order of accuracy, the stability region is larger for schemes with more stages. This is because having more stages provides more degrees of freedom to select the coefficients, potentially leading to a larger stability region.

Let us note that all the developments in this section are based on the assumption that the fast part is solved exactly or with high order temporal schemes combined with a really small time step. In practice, this is not always possible. In this case, the stability regions will also depend on the method used to solved the stiff terms.

5 Numerical results

Please note that in principle, multirate schemes are agnostic to the fine solver. This is also the case for this work here. For the numerical results, the fine solver is either a standard RK4 with timestepsize $\Delta\tau = \frac{\Delta t}{M}$ for some given M; or it is exactly integrated through a highly resolved numerical computation.

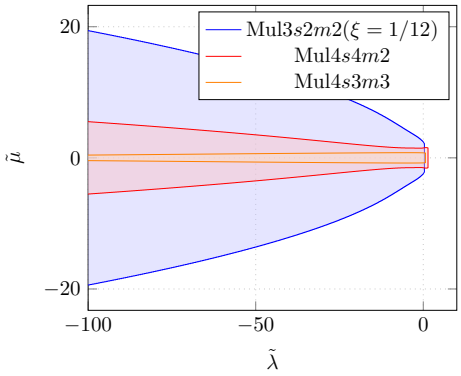


Figure 1: Stability domains for the different methods presented in this work, i.e., parts of the domain where there holds $|R(\tilde{\lambda}, \tilde{\mu})| \leq 1$. For this picture, we treat the part λy as the 'fast' (i.e., stiff) part. It can be seen very well that all the schemes encompass the negative real axis, which is natural, as the λy part is treated exactly.

5.1 A singularly perturbed ODE: van der Pol equation

In this section, we discuss numerical results for the van der Pol equation

$$y_1'(t) = y_2(t), \quad y_2'(t) = \frac{1}{\varepsilon}\left((1 - y_1(t)^2)y_2(t) - y_1(t)\right), \tag{5.1}$$

$$y_1(0) = 2, \qquad y_2(0) = -\frac{2}{3} + \frac{10}{81}\varepsilon - \frac{292}{2187}\varepsilon^2. \tag{5.2}$$

Van der Pol's equation constitutes a singularly perturbed problem as $\varepsilon \to 0$, making it particularly challenging for a numerical solver to accurately resolve the solution. For $\varepsilon \to 0$, stiffness increases. The initial conditions are chosen in such a way that the solution is asymptotically smooth as $\varepsilon \to 0$ and final time T_{end} small enough, so we do not have to deal with any sharp gradients that spoil the numerical solution [17]. We integrate until time $T_{\text{end}} = 0.5$, which is small enough for having solutions without sharp gradients. The numerical error is then defined as the Eulerian norm of the difference of exact and numerical solution at time T_{end}. As is frequently done in the context of IMEX schemes, see, e.g., [6], we set non-stiff and stiff parts as

$$f(y) = \begin{pmatrix} y_2(t) \\ 0 \end{pmatrix}, \quad g(y) = \frac{1}{\varepsilon}\begin{pmatrix} 0 \\ (1 - y_1(t)^2)y_2(t) - y_1(t) \end{pmatrix}.$$

In Fig. 2, we report on numerical results using the third-order scheme (2.6) with $\xi = \frac{1}{12}$ (Mul3s2m2(1/12)) for various ε. First (top picture left), we assess the quality of the outer iteration, and we set the resolution of the pseudo-time τ in (2.2) to be an exact solution (as there is no exact analytical solution to the van der Pol equation, a very highly resolved numerical computation is used). For the other three pictures, we use the classical Runge-Kutta 4 (RK4) scheme for the inner iteration using a $\Delta\tau = \frac{\Delta t}{M}$ with $M = 4$ (top right), $M = 10$ (bottom left) and $M = \frac{1}{\varepsilon}$ (bottom right). Values not plotted are NaN, hence, the method was not stable in this case. For the case with an exact fast solver, it is clearly visible that the method converges with order three for larger values of ε ('non-stiff

case'), which is to be expected based on the analysis from Sec. 3. For the smaller ε, order reduction appears due to the stiff nature of the problem. Furthermore, for increasing stiffness, more iterations on the fast solver need to be done for the method to be stable. Obviously, this is not unexpected for a fully explicit scheme. In particular, only for $\Delta\tau = \varepsilon\Delta t$, so a highly-resolved fast solver, there is no convergence issue at all. Also $\Delta\tau = \sqrt{\varepsilon}\Delta t$ has been checked, this did not lead to a uniformely stable scheme throughout the ε-values considered. The message at this point is pretty clear, it is that the multirate scheme, in combination with a fully explicit interior scheme, is only useful for moderately stiff problems, or in combination with an *implicit* fast solution process.

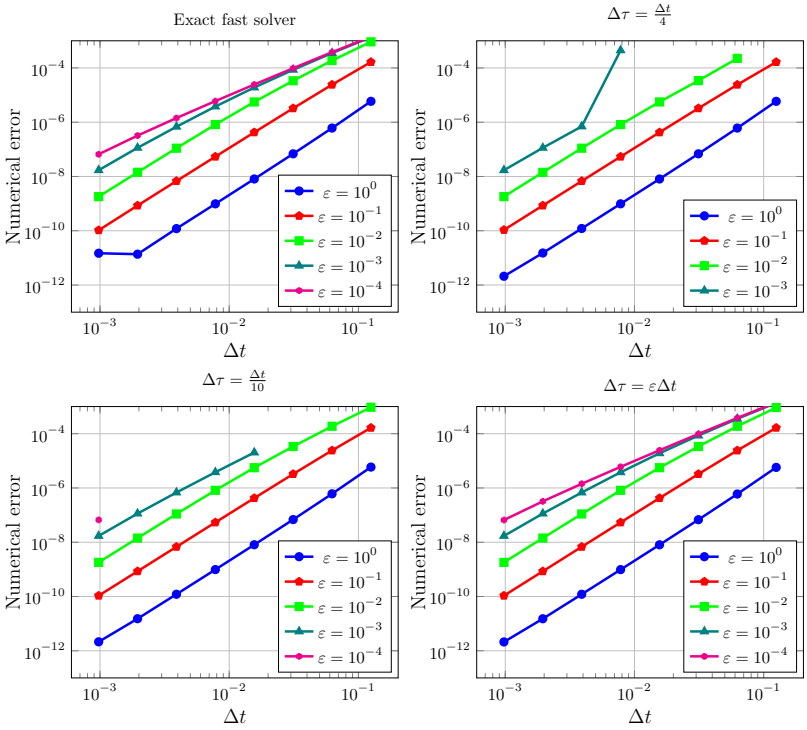


Figure 2: Numerical results for the van der Pol equation (5.1), generated using the third-order two-derivative multirate scheme (2.8) with a value of $\xi = \frac{1}{12}$. Integration of the pseudo-time τ is done using an RK4 scheme with $\Delta\tau = \frac{\Delta t}{M}$ with $M = 4$ (top right), $M = 10$ (bottom left) and $M = \frac{1}{\varepsilon}$ (bottom right). Please note that values not plotted were NaN, i.e., the overall method was not stable there.

From now on, we will hence only consider moderately stiff problems, i.e., we make the somewhat arbitrary choice of $10^{-2} \leq \varepsilon \leq 1$. In Fig. 3 on the left, we compute solutions to van der Pol equation (5.1) using the fourth-order two-derivative scheme Mul4s4m2 given in (2.8). The pseudo-time τ is integrated using an RK4-scheme with

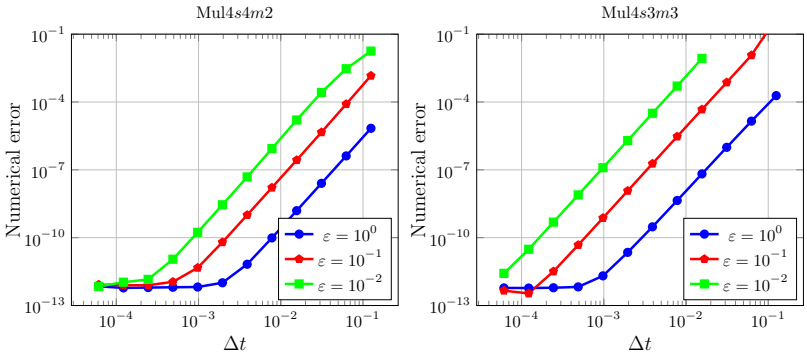


Figure 3: Numerical results for the van der Pol equation (5.1), generated using the fourth-order two-derivative multirate scheme (2.8) (left) and the fourth-order three derivative multirate scheme (2.9) (right). Integration of the pseudo-time τ is done using an RK4 scheme with $\Delta\tau = \frac{\Delta t}{10}$. Please note that values not plotted were NaN, i.e., the method was not stable there.

$\Delta\tau = \frac{\Delta t}{10}$. For ε considered here, this is enough to guarantee stability for all values of Δt considered. It is visible from the numerical results that the error converges with the design order of the scheme, which is four in this case. Fig. 3 on the right shows numerical results for the fourth-order three-derivative scheme Mul4s3m3, see (2.9). For large Δt and $\varepsilon = 10^{-2}$, the scheme has stability issues. This is not surprising given the size of the stability region shown in Sec. 4.

5.2 Stiff hyperbolic equation

To examine the properties of the method more closely, we consider in this section the following prototypical stiff hyperbolic equation, taken from [29]:

$$w_t + Aw_x = 0, \qquad (x,t) \in [0, 2\pi] \times [0, T_{end}] \tag{5.3}$$

with

$$A = \begin{pmatrix} a & 1 & 0 \\ \frac{1}{\varepsilon^2} & a & \frac{1}{\varepsilon^2} \\ 0 & 1 & a \end{pmatrix}$$

for some given parameter $a > 0$ (that we choose $a = 1$ in the numerical experiments). T_{end} is set to 0.5. Boundary conditions are assumed to be periodic; as initial conditions, we choose

$$w(x,0) = \begin{pmatrix} e^{-\sin(x)} \\ e^{-\sin(x)^2} \\ \cos(x) \end{pmatrix}.$$

This hyperbolic system has the three wave speeds a and $a \pm \frac{\sqrt{2}}{\varepsilon}$. For $\varepsilon \ll 1$, there is a slow wave speed (mimicking the 'convective' wave speed of the Euler equations) and two fast wave speeds (the equivalent for Euler equations would be the 'acoustic' wave speeds). Using an explicit time integration scheme on this equation, such as, e.g., explicit Euler, would result in the timestep restriction

$$\Delta t \lesssim \frac{\Delta x}{a + \frac{\sqrt{2}}{\varepsilon}} = \mathcal{O}\left(\varepsilon \Delta x\right).$$

As in [29, Sec. 4], we split the matrix A into $A = \widehat{A} + \widetilde{A}$, with

$$\widehat{A} = \begin{pmatrix} a & \varepsilon & 0 \\ \frac{1}{\varepsilon} & a & \frac{1}{\varepsilon} \\ 0 & \varepsilon & a \end{pmatrix}, \qquad \widetilde{A} = \begin{pmatrix} 0 & 1-\varepsilon & 0 \\ \frac{1-\varepsilon}{\varepsilon^2} & 0 & \frac{1-\varepsilon}{\varepsilon^2} \\ 0 & 1-\varepsilon & 0 \end{pmatrix}.$$

The wavespeeds of $\widehat{A}$ are a and $a \pm \sqrt{2}$; the wavespeeds of $\widetilde{A}$ are 0 and $\pm \frac{\sqrt{2}(1-\varepsilon)}{\varepsilon}$. In the following, the contribution $\widehat{A} w_x$ is treated as the non-stiff part, and $\widetilde{A} w_x$ as the stiff part. We discretize the equation in space through a standard first-order Finite Volume method with a local Lax-Friedrichs / Rusanov numerical flux.

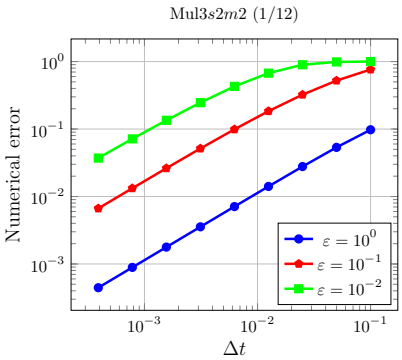


Figure 4: Numerical results to the hyperbolic problem (5.3) using a first-order Finite Volume scheme with a local Lax-Friedrichs/Rusanov flux, hence the first-order convergence. As time integration, Mul3$s2m2$ is chosen. We use $M = 1$, $M = 5$ and $M = 56$ for $\varepsilon = 1$, $\varepsilon = 0.1$ and $\varepsilon = 0.01$, respectively. The errors are scaled by the norm of the solution, which is $\varepsilon-$dependent.

In Fig. 4, we plot numerical results for the problem integrated with the third-order Mul3$s2m2(1/12)$ scheme. As basis for our investigations, we use the *non-stiff* CFL condition

$$\Delta t \lesssim \frac{\Delta x}{a + \sqrt{2}}.$$

This leads to an initial setup of 20 spatial cells and 5 time steps; which is then multiplied by two in each subsequent iteration. To choose $\Delta\tau$, we need the *stiff* CFL condition

$$\Delta\tau \lesssim \Delta x \frac{\varepsilon}{\sqrt{2}(1-\varepsilon)}.$$

This leads to $M = 1$, $M = 5$ and $M = 56$ for $\varepsilon = 1$, $\varepsilon = 0.1$ and $\varepsilon = 0.01$, respectively. $\Delta\tau$ is then defined as $\Delta\tau = \frac{\Delta t}{M}$. It can be clearly seen from the numerical results that the resulting algorithm is stable.

5.3 Diffusion-dominated equation

We consider the viscous Burgers' equation to study the effect of multirate schemes on diffusion-dominated equations. The equation is given by:

$$u_t = -\partial_x f(u) + \nu \partial_x g(u_x), \tag{5.4}$$

where $x \in [0, 2\pi]$ and ν is the diffusion coefficient. The functions f and g are defined by $f(u) = \frac{u^2}{2}$ and $g(q) = q_x$, respectively. Let the initial condition be

$$u(x, 0) = u_0(x), \tag{5.5}$$

with periodic boundary conditions.

For $\nu > 0$, an exact solution to Eqs. (5.4) and (5.5) exists, and it is obtained using the Cole-Hopf transformation [18]. Classical explicit time-stepping schemes require a timestep

$$\Delta t = \mathcal{O}\left(\frac{\Delta x^2}{\nu}\right), \tag{5.6}$$

for highly diffusive problems in order to maintain stability. Upon treating the nonlinear convection (f) as the non-stiff part and the diffusion (g) as the stiff part, the timestep restriction can be brought down to

$$\Delta t = \mathcal{O}\left(\Delta x\right). \tag{5.7}$$

As diffusivity increases, the number of fast steps (M) can be adjusted accordingly to ensure stability while maintaining the new timestep condition (5.7).

The spatial part is discretized using the discontinuous Galerkin spectral element method (DGSEM) [23] with BR1 lifting procedure [4, 27] for the diffusion terms and global Lax-Friedrichs numerical flux for the convective part. Therefore, the second-order equation (5.4) is reduced to a first-order equation using an auxiliary variable q. The equations are given by

$$q = \partial_x u, \qquad u_t = -\partial_x f(u) + \partial_x g(q). \tag{5.8}$$

Divide the spatial domain $\Omega = [0, 2\pi]$ into N_e non-overlapping mesh elements as follows

$$\Omega = \bigcup_{e=1}^{N_e} \Omega_e.$$

Let $\mathcal{T}$ be the set of all mesh elements Ω_e, $1 \le e \le N_e$, and $\partial\mathcal{T}$ be the set of all mesh element boundaries $\partial\Omega_e$. Define the space of broken polynomial as follows:

$$V_{N_p} := \left\{ v \in L^2(\Omega) \;\middle|\; v|_{\Omega_e} \in \Pi_{N_p}(\Omega_e),\ 1 \le e \le N_e \right\},$$

where the space $\Pi_{N_p}(\Omega_e)$ denotes polynomials of degree at most N_p. As basis functions, DGSEM uses the Lagrangian basis functions on Gaussian nodes. We define the following bilinear operators

$$(\mathbf{a}, \phi)_{\mathcal{T}} := \sum_{e=1}^{N_e} \int_{\Omega_e} \mathbf{a}\phi \, \mathrm{d}x, \quad \langle \mathbf{a}, \phi \rangle_{\partial\mathcal{T}} := \sum_{e=1}^{N_e} \left(\mathbf{a}\phi \Big|_{\partial\Omega_e} \right),$$

for functions $\mathbf{a}, \phi \in V_{N_p}$. Let the average value and the jump, respectively, of a function at a boundary be represented by

$$\{h(u)\} := \frac{h(u^L) + h(u^R)}{2}, \qquad [h(u)] := h(u^L) - h(u^R),$$

where values u^L and u^R are the left and right values derived from the neighboring elements of a boundary $\partial\Omega_e$, respectively. Then, the discontinuous Galerkin variational form of equation (5.8) is given by

$$\begin{aligned}(q, \psi)_{\mathcal{T}} &= -(u, \psi_x)_{\mathcal{T}} + \langle \{u\}, \psi \rangle_{\partial\mathcal{T}}, \\ (u_t, \phi)_{\mathcal{T}} &= \underbrace{\frac{1}{2}\left(u^2, \phi_x\right)_{\mathcal{T}} - \frac{1}{2}\left\langle \{u^2\} + [u], \phi \right\rangle_{\partial\mathcal{T}}}_{\text{non-stiff}} \underbrace{- (\nu q, \phi_x)_{\mathcal{T}} + \langle \{\nu q\}, \phi \rangle_{\partial\mathcal{T}}}_{\text{stiff}},\end{aligned}$$

for all $\psi, \phi \in V_{N_p}$. The non-stiff part is hence given by

$$\left(u_t^{NS}, \phi\right)_{\mathcal{T}} = \frac{1}{2}\left(u^2, \phi_x\right)_{\mathcal{T}} - \frac{1}{2}\left\langle \{u^2\} + [u], \phi \right\rangle_{\partial\mathcal{T}}, \tag{5.9}$$

and the stiff part is given by

$$\left(u_t^{S}, \phi\right)_{\mathcal{T}} = -(\nu q, \phi_x)_{\mathcal{T}} + \langle \{\nu q\}, \phi \rangle_{\partial\mathcal{T}}. \tag{5.10}$$

In this test case, we restrict our results to the two-derivative multirate schemes. The second derivative of the non-stiff term is evaluated using the Lax-Wendroff procedure through

$$u_{tt}^{NS} = -\partial_t\left(\partial_x f(u)\right) = -\partial_x\left(\partial_t f(u)\right) = -\partial_x\left(f'(u)u_t\right),$$

and the variational form is given by

$$\left(u_{tt}^{NS}, \phi\right)_{\mathcal{T}} = (uu_t, \phi_x)_{\mathcal{T}} + \left\langle \{-uu_t\} - \frac{1}{2}[u_t], \phi \right\rangle_{\partial\mathcal{T}}. \tag{5.11}$$

The two-derivative multirate method uses the discretized fluxes from equations (5.9), (5.10), and (5.11) to perform time integration. The DGSEM approximates the exact integrations in equations (5.9), (5.10), and (5.11), using the same quadrature nodes employed in constructing the one-dimensional Lagrange interpolation polynomials. Refer to [34] for the detailed formulation of the DGSEM method.

In Fig. 5, the results for the two-derivative multirate schemes are shown for the viscous Burgers equation (5.4) with the initial condition

$$u_0(x) = \sin^2(x).$$

The third-order (Mul3s2m2 (1/12)) multirate scheme is combined with DGSEM with $N_p = 2$, and the fourth-order (Mul4s4m2) multirate scheme is combined with DGSEM with $N_p = 3$ to achieve overall accuracies of three and four, respectively. For a given set of spatial discretization parameters, the timestep for the multirate scheme is calculated using equation

$$\Delta t = CFL_{scale} \cdot \frac{\Delta x}{(2N_p + 1)\min(u)},$$

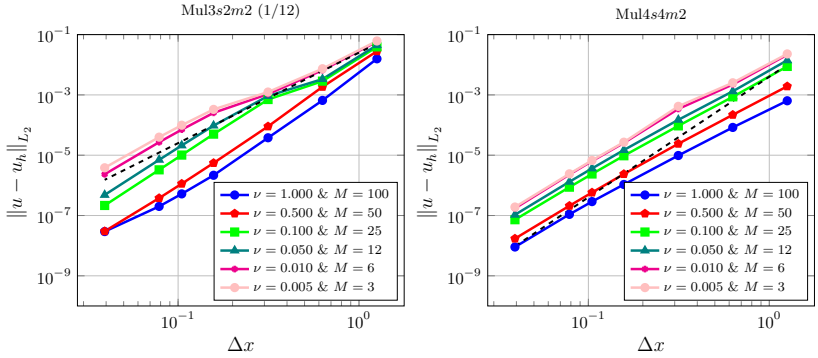


Figure 5: Numerical results for the viscous Burgers' equation (5.4) with $u_0(x) = \sin^2(x)$ are shown for a two-derivative third-order (Mul3$s2m2$ (1/12)) multirate scheme (left) and a two-derivative fourth-order (Mul4$s4m2$) multirate scheme (right). The L_2-error is plotted against the mesh size Δx, for a fixed $CFL_{scale} = 0.9$ for the third-order scheme and $CFL_{scale} = 0.25$ for the fourth-order scheme. The error is evaluated at $T_{\text{end}} = 0.5$ in comparison to the exact solution [18]. The DGSEM spatial discretization method is used, with $N_p = 2$ for the third-order scheme and $N_p = 3$ for the fourth-order scheme. The dashed black thick line represents convergence orders of three (left) and four (right). Results are shown for various diffusion coefficients ranging from $\nu = 0.005$ to $\nu = 1$, with their required fast steps (M).

where CFL_{scale} is a value less than one, Δx is the mesh size, and $\min(u)$ is the minimum eigenvalue of the convective flux. The CFL_{scale} for the Mul3$s2m2$ (1/12) scheme is set to 0.9. However, CFL_{scale} for the Mul4$s4m2$ scheme is set to 0.25 due to its narrow stability region (see Fig. 1). The L_2-error is plotted against the mesh size Δx in Fig. 5. The error is evaluated at $T_{\text{end}} = 0.5$ in comparison to the exact solution [18]. As diffusivity increases, the schemes require more fast steps (M). Although the required number of steps M depends on spatial discretization, we have chosen a value of M large enough to work for all chosen mesh sizes. The convergence plots are shown for $N_e = \{5, 10, 20, 40, 60, 80, 160\}$. It can be seen in Fig. 5 that the multirate-DGSEM exhibits the desired convergence order for almost all the diffusion coefficients considered.

6 Conclusion and outlook

In this paper, we focused on differential equations containing both stiff and non-stiff terms. These equations are challenging because they typically require a very small time step for the entire equation, even though only some terms necessitate it. To address this issue, we presented a multiderivative scheme that incorporates multiple derivatives of the stiff part. The idea behind this scheme is to solve the non-stiff terms with a large time step while solving the stiff part exactly or with a very small time step. This approach is particularly relevant when the term with a small time characteristic is less computationally intensive than the other terms.

This class of temporal schemes has been previously studied, but only for single derivatives. We proposed schemes that include up to three derivatives of the non-stiff term and derived the order conditions to develop temporal methods up to order four. We presented three schemes that vary in the number of stages and derivatives. The stability regions were demonstrated using a classical convection-diffusion problem, where the eigenvalues of the

stiff terms are purely real, while others are imaginary. Additionally, these schemes were tested on two benchmarks, confirming their order and precision.

Future work is on extending the framework to more realistic settings, e.g., for the compressible Navier-Stokes equations at low Mach numbers, or the shallow water equations at low Froude numbers. In particular, it seems challenging to choose fast-scale solvers that are efficient and keep the stability region as close to the 'exact' stability region as possible.

Acknowledgements

Arjun Thenery Manikantan was funded by the "Bijzonder Onderzoeksfonds" (BOF) from UHasselt - project no. BOF21KP12. During the preparation of this work, A. Ishimwe used Deepl in order to improve some few sentences w.r.t. language. After using this tool, the authors reviewed and edited the content as needed and take full responsibility for the content of the publication.

Conflict of interests

The authors have no competing interests to declare that are relevant to the content of this article.

References

[1] Assyr Abdulle, Marcus J. Grote, and Giacomo Rosilho de Souza. Explicit stabilized multirate method for stiff differential equations. *Mathematics of Computation*, 91(338):2681–2714, 2022.

[2] U. M. Ascher, S. Ruuth, and R. Spiteri. Implicit-explicit Runge-Kutta methods for time-dependent partial differential equations. *Applied Numerical Mathematics*, 25:151–167, 1997.

[3] Antonio Baeza, Sebastiano Boscarino, Pep Mulet, Giovanni Russo, and David Zorío. On the stability of Approximate Taylor methods for ODE and their relationship with Runge-Kutta schemes. *arXiv preprint arXiv:1804.03627*, 2018.

[4] F. Bassi and S. Rebay. A high-order accurate discontinuous finite-element method for the numerical solution of the compressible Navier-Stokes equations. *Journal of Computational Physics*, 131:267–279, 1997.

[5] Tobias Peter Bauer and Oswald Knoth. Extended multirate infinitesimal step methods: Derivation of order conditions. *Journal of Computational and Applied Mathematics*, 387:112541, 2021.

[6] S. Boscarino. On an accurate third order implicit-explicit Runge-Kutta method for stiff problems. *Applied Numerical Mathematics*, 59:1515–1528, 2009.

[7] John C. Butcher and Gholamreza Hojjati. Second derivative methods with RK stability. *Numerical Algorithms*, 40(4):415–429, 2005.

[8] H. Carrillo and C. Parés. Compact approximate Taylor methods for systems of conservation laws. *Journal of Scientific Computing*, 80(3):1832–1866, 2019.

[9] R. Chan and A. Tsai. On explicit two-derivative Runge-Kutta methods. *Numerical Algorithms*, 53:171–194, 2010.

[10] Rujeko Chinomona and Daniel R. Reynolds. Implicit-explicit multirate infinitesimal GARK methods. *SIAM Journal on Scientific Computing*, 43(5):A3082–A3113, 2021.

[11] Jeremy Chouchoulis, Jochen Schütz, and Jonas Zeifang. Jacobian-free explicit multiderivative Runge-Kutta methods for hyperbolic conservation laws. *Journal of Scientific Computing*, 90(96), 2022.

[12] Dale R. Durran and Peter N. Blossey. Implicit–explicit multistep methods for fast-wave–slow-wave problems. *Monthly Weather Review*, 140(4):1307 – 1325, 2012.

[13] Alex C. Fish and Daniel R. Reynolds. Adaptive time step control for multirate infinitesimal methods. *SIAM Journal on Scientific Computing*, 45(2):A958–A984, 2023.

[14] Alex C. Fish, Daniel R. Reynolds, and Steven B. Roberts. Implicit-explicit multirate infinitesimal stage-restart methods. *Journal of Computational and Applied Mathematics*, 438:Paper No. 115534, 23, 2024.

[15] M. Günther and A. Sandu. Multirate generalized additive Runge Kutta methods. *Numerische Mathematik*, 133:497–524, 2016.

[16] E. Hairer and G. Wanner. Multistep-multistage-multiderivative methods for ordinary differential equations. *Computing (Arch. Elektron. Rechnen)*, 11(3):287–303, 1973.

[17] E. Hairer and G. Wanner. *Solving ordinary differential equations II.* Springer Series in Computational Mathematics, Berlin, Heidelberg, 1991.

[18] E. Hopf. The partial differential equation $u_t + uu_x = \mu u_{xx}$. *Communications on Pure and Applied Mathematics*, 3:201–230, 1950.

[19] A. Ishimwe, E. Deleersnijder, V. Legat, and J. Lambrechts. A split-explicit second order Runge–Kutta method for solving 3D hydrodynamic equations. *Ocean Modeling*, (186):102273, 2023.

[20] S. Jin. Asymptotic preserving (AP) schemes for multiscale kinetic and hyperbolic equations: A review. *Rivista di Matematica della Universita Parma*, 3:177–216, 2012.

[21] Shinhoo Kang, Alp Dener, Aidan Hamilton, Hong Zhang, Emil M. Constantinescu, and Robert L. Jacob. Multirate partitioned Runge-Kutta methods for coupled Navier-Stokes equations. *Computers & Fluids. An International Journal*, 264:Paper No. 105964, 15, 2023.

[22] S. Klainerman and A. Majda. Singular limits of quasilinear hyperbolic systems with large parameters and the incompressible limit of compressible fluids. *Communications on Pure and Applied Mathematics*, 34:481–524, 1981.

[23] D. A. Kopriva. *Implementing spectral methods for partial differential equations: Algorithms for scientists and engineers.* Springer Science & Business Media, Dordrecht, 2009.

[24] Vu Thai Luan, Rujeko Chinomona, and Daniel R. Reynolds. A new class of high-order methods for multirate differential equations. *SIAM Journal on Scientific Computing*, 42(2):A1245–A1268, 2020.

[25] Andreas Naumann and Jörg Wensch. Multirate finite step methods. *Numerical Algorithms*, 81(4):1547–1571, 2019.

[26] Mukaddes Ökten Turaci and Turgut Öziş. Derivation of three-derivative Runge-Kutta methods. *Numerical Algorithms*, 74(1):247–265, 2017.

[27] Sigrun Ortleb. A comparative Fourier analysis of discontinuous Galerkin schemes for advection–diffusion with respect to BR1, BR2, and local discontinuous Galerkin diffusion discretization. *Mathematical Methods in the Applied Sciences*, 43(13):7841–7863, 2020.

[28] Adrian Sandu. A class of multirate infinitesimal GARK methods. *SIAM Journal on Numerical Analysis*, 57(5):2300–2327, 2019.

[29] J. Schütz and S. Noelle. Flux splitting for stiff equations: A notion on stability. *Journal of Scientific Computing*, 64(2):522–540, 2015.

[30] D. C. Seal, Y. Güçlü, and A. Christlieb. High-order multiderivative time integrators for hyperbolic conservation laws. *Journal of Scientific Computing*, 60:101–140, 2014.

[31] Gustaf Söderlind, Laurent Jay, and Manuel Calvo. Stiffness 1952–2012: sixty years in search of a definition. *BIT. Numerical Mathematics*, 55(2):531–558, 2015.

[32] J. M. Straka, Robert B. Wilhelmson, Louis J. Wicker, John R. Anderson, and Kelvin K. Droegemeier. Numerical solutions of a non-linear density current: A benchmark solution and comparisons. *International Journal for Numerical Methods in Fluids*, 17(1):1–22, 1993.

[33] J. Wensch, O. Knoth, and A. Galant. Multirate infinitesimal step methods for atmospheric flow simulation. *BIT Numerical Mathematics*, 49:449–473, 2009.

[34] J. Zeifang and J. Schütz. Implicit two-derivative deferred correction time discretization for the discontinuous Galerkin method. *Journal of Computational Physics*, 464:111353, 2022.

[35] D. Zorío, A. Baeza, and P. Mulet. An approximate Lax–Wendroff-type procedure for high order accurate schemes for hyperbolic conservation laws. *Journal of Scientific Computing*, 71:246–273, 2017.

7 Appendix: Proof of the order conditions

In this appendix section, we prove Thm. 1. The proof goes very much along the lines of [33, Sec. 3], most of equations are very similar. However, some of the details vary, which is why we provide it here. The notation used is the notation from [33]. We restrict ourselves to four derivatives at most, i.e., $m \leq 4$. The function Z_{ni} from (2.2) is a function of τ and Δt, obviously. As in [33], we define $Z^{(k,l)}$ to be the $k-$th derivative of Z in direction τ, and the $l-$th derivative of Z in direction of Δt. $Y^{(l)}$ denotes the $l-$th derivative of Y in direction Δt. Evaluation is in both cases at $\tau = \Delta t = 0$. By $(f^{\{k\}})^l$, we denote the $l-$th derivative of $f^{\{k\}}(Y)$ in direction Δt. The understanding is that $(f^{\{k\}})^l$ is zero if $l < 0$. If $l = 0$, we assume $(f^{\{k\}})^0 = f^{\{k\}}$; consult (2.3) for the definition.

With these defintions, there holds

$$
\begin{aligned}
Z_{ni}^{(0,l)} &= \sum_j \alpha_{ij} Y_{nj}^{(l)}, \qquad l \geq 1 \\
Z_{ni}^{(1,l)} &= \frac{1}{l+1} \sum_j \gamma_{ij} Y_{nj}^{(l+1)} + d_i g(Z_{ni})^{(0,l)} + \sum_{k=0}^{m-1} \frac{l!}{(l-k)!} \sum_{j=1}^{i-1} \beta_{ij}^{\{k\}} f^{\{k\}}(Y_{nj})^{(l-k)} \\
Z_{ni}^{(k,l)} &= d_i g(Z_{ni})^{(k-1,l)}, \qquad k \geq 2
\end{aligned}
$$

In the matrix notation of [33], this amounts to

$$\begin{aligned}
Z^{(0,l)} =& \alpha Y^{(l)}, \qquad l \geq 1 \\
Z^{(1,l)} =& \frac{1}{l+1}(\gamma \otimes I)Y^{(l+1)} + (D \otimes I)g(Z)^{(0,l)} + \sum_{k=0}^{m-1} \frac{l!}{(l-k)!}(\beta^{\{k\}} \otimes I)f^{\{k\}}(Y)^{(l-k)} \\
Z^{(k,l)} =& (D \otimes I)g(Z)^{(k-1,l)}, \qquad k \geq 2
\end{aligned}$$

Here, Y denotes the upright vector $Y = (Y_{n1}, \dots, Y_{n,s+1})^T$, and similarly for Z. We understand $f^{\{k\}}(Y)$ and similar expressions as the vector $(f^{\{k\}}(Y_{n1}), \dots, f^{\{k\}}(Y_{n,s+1}))^T$. Using the chain rule, we obtain that for $Y^{(\kappa)}$, there holds

$$\begin{aligned}
Y^{(\kappa)} =& \sum_{l=0}^{\kappa} \binom{\kappa}{l} Z^{(l,\kappa-l)} \\
=& \sum_{l=1}^{\kappa} \binom{\kappa}{l} (D \otimes I)g(Z)^{(l-1,\kappa-l)} + ((\gamma+\alpha) \otimes I)Y^{(\kappa)} \\
&+ \sum_{k=0}^{m-1} \frac{\kappa!}{(\kappa-k-1)!}(\beta^{\{k\}} \otimes I)f^{\{k\}}(Y)^{\kappa-k-1}.
\end{aligned}$$

Upon using the definitions of R and $A^{\{k\}}$ as in (2.4) and (2.5), we obtain

$$Y^{(\kappa)} = \sum_{l=1}^{\kappa} \binom{\kappa}{l} (RD \otimes I)g(Z)^{(l-1,\kappa-l)} + \sum_{k=0}^{m-1} \frac{\kappa!}{(\kappa-k-1)!}(A^{\{k\}} \otimes I)f^{\{k\}}(Y)^{\kappa-k-1}.$$

In the following, we define

$$c^{\{k\}} := A^{\{k\}}\mathbb{1}.$$

For $k = 0$, this is the usual definition of the time instances of the stages for Runge-Kutta schemes. For easier notation, we write $c \equiv c^{\{0\}}$. We now go through the terms recursively as in [33].

Zeroth and first order To zeroth and first order, there holds

$$\begin{aligned}
Y^{(0)} =& Z^{(0,0)} = \mathbb{1} \otimes y_n, \\
Y^{(1)} =& (RD \otimes I)g(Z)^{(0,0)} + (A^{\{0\}} \otimes I)f^{\{0\}}(Y)^{(0)} = (RD\mathbb{1}) \otimes g + (A^{\{0\}}\mathbb{1}) \otimes f^{\{0\}} \\
=& c \otimes F.
\end{aligned}$$

The last step is true because $D\mathbb{1} = \beta^{\{0\}}\mathbb{1}$ due to the balancing condition.

Second order Further, there holds

$$\begin{aligned}
Z^{(0,1)} =& \alpha Y^{(1)} = (\alpha c \otimes I)F =: \tilde{c} \otimes F, \\
Z^{(1,0)} =& (\gamma \otimes I)Y^{(1)} + (\beta^{\{0\}} \otimes I)f^{\{0\}}(Y)^{(0)} + (D \otimes I)g(Z)^{(0,0)}
\end{aligned}$$

$$
\begin{aligned}
&=(\gamma c \otimes I)F + (\beta^{\{0\}}\mathbb{1} \otimes I)f^{\{0\}} + (D\mathbb{1} \otimes I)g \\
&=((\gamma c + \beta^{\{0\}}\mathbb{1}) \otimes I)F.
\end{aligned}
$$

Now we have

$$
\beta\mathbb{1} = R^{-1}A^{\{0\}}\mathbb{1} = R^{-1}c,
$$

and hence

$$
Z^{(1,0)} = ((\gamma + R^{-1})c \otimes I)F = ((I - \alpha)c) \otimes I)F = (c - \tilde{c}) \otimes F.
$$

Now, to second order:

$$
\begin{aligned}
Y^{(2)} =& \sum_{l=1}^{2} \binom{2}{l} (RD \otimes I)g(Z)^{(l-1,2-l)} + 2(A^{\{0\}} \otimes I)f^{\{0\}}(Y)^{(1)} + 2(A^{\{1\}} \otimes I)f^{\{1\}}(Y)^{(0)} \\
=& 2(RD \otimes I)g(Z)^{(0,1)} + (RD \otimes I)g(Z)^{(1,0)} + 2(A^{\{0\}} \otimes I)f_y Y^{(1)} + 2(A^{\{1\}} \otimes I)f^{\{1\}}(Y)^{(0)} \\
=& 2(RD \otimes I)g_y Z^{(0,1)} + (RD \otimes I)g_y Z^{(1,0)} + 2A^{\{0\}}c \otimes f_y F + 2A^{\{1\}}\mathbb{1} \otimes f_y F \\
=& 2RD\tilde{c} \otimes g_y F + RD(c - \tilde{c}) \otimes g_y F + 2A^{\{0\}}c \otimes f_y F + 2c^{\{1\}} \otimes f_y F \\
=& RD(c + \tilde{c}) \otimes g_y F + (2A^{\{0\}}c + 2c^{\{1\}}) \otimes f_y F.
\end{aligned}
$$

Third order Continuing with higher-order terms, we obtain

$$
\begin{aligned}
Z^{(0,2)} =& \alpha Y^{(2)} = \alpha RD(c + \tilde{c}) \otimes g_y F + (2\alpha A^{\{0\}}c + 2\alpha c^{\{1\}}) \otimes f_y F \\
Z^{(1,1)} =& \frac{1}{2}(\gamma \otimes I)Y^{(2)} + (\beta^{\{0\}} \otimes I)f^{\{0\}}(Y)^{(1)} + (\beta^{\{1\}} \otimes I)f^{\{1\}}(Y)^{(0)} + (D \otimes I)g(Z)^{(0,1)} \\
=& \frac{1}{2}(\gamma \otimes I)Y^{(2)} + (\beta^{\{0\}} \otimes I)f_y Y^{(1)} + (\beta^{\{1\}}\mathbb{1}) \otimes f_y F + (D \otimes I)g_y Z^{(0,1)} \\
=& \frac{1}{2}(\gamma \otimes I)Y^{(2)} + \beta c \otimes f_y F + (R^{-1}c^{\{1\}}) \otimes f_y F + (D\tilde{c}) \otimes g_y F. \\
Z^{(2,0)} =& (D \otimes I)g(Z)^{(1,0)} = (D \otimes I)g_y Z^{(1,0)} = D(c - \tilde{c}) \otimes g_y F.
\end{aligned}
$$

Now finally, we end up with the third derivative of Y:

$$
\begin{aligned}
Y^{(3)} =& \sum_{l=1}^{3} \binom{3}{l} (RD \otimes I)g(Z)^{(l-1,3-l)} + 3(A^{\{0\}} \otimes I)f^{\{0\}}(Y)^{(2)} \\
&+ 6(A^{\{1\}} \otimes I)f^{\{1\}}(Y)^{(1)} + 6(A^{\{2\}} \otimes I)f^{\{2\}}(Y)^{(0)} \\
=& (RD \otimes I)\sum_{l=1}^{3} \binom{3}{l} g(Z)^{(l-1,3-l)} + 3(A^{\{0\}} \otimes I)f(Y)^{(2)} \\
&+ 6(A^{\{1\}} \otimes I)f^{\{1\}}(Y)^{(1)} + 6(A^{\{2\}} \otimes I)f^{\{2\}}(Y)^{(0)}.
\end{aligned} \tag{7.1}
$$

First, the sum is treated. If one realizes that $\beta = R^{-1}A$ and $R^{-1} = I - \alpha - \gamma$, then in a straightforward way, one obtains

$$
\sum_{l=1}^{3} \binom{3}{l} g(Z)^{(l-1,3-l)} = \left(c^2 + \tilde{c}^2 + c \odot \tilde{c}\right) \otimes g_{yy}(F, F)
$$

$$
\begin{aligned}
&+\left(3(I+\alpha)(A^{\{0\}}c+c^{\{1\}})\right)g_y f_y F\\
&+\left(3(\alpha+\frac{\gamma}{2})RD(c+\tilde{c})+D(c+2\tilde{c})\right)g_y g_y F.
\end{aligned}
$$

Now we treat the remainder of (7.1). Before doing this, we make the following computation about the Jacobian of the first total derivative:

$$
f_y^{\{1\}} = (f_y F)_y = (f_y(f+g))_y = f_y f_y + f_y g_y + f_{yy}F,
$$

Furthermore, there holds

$$
f^{\{2\}} = f_{yy}(F,F) + f_y F_y F.
$$

Hence,

$$
\begin{aligned}
&3(A^{\{0\}}\otimes I)f^{\{0\}}(Y)^{(2)} + 6(A^{\{1\}}\otimes I)f^{\{1\}}(Y)^{(1)} + 6(A^{\{2\}}\otimes I)f^{\{2\}}(Y)^{(0)}\\
=&3(A^{\{0\}}\otimes I)\left(f_{yy}(Y^{(1)},Y^{(1)}) + f_y Y^{(2)}\right) + 6(A^{\{1\}}\otimes I)f_y^{\{1\}}Y^{(1)} + 6(A^{\{2\}}\otimes I)f^{\{2\}}(Y)^{(0)}\\
=&3(A^{\{0\}}c^2)\otimes f_{yy}(F,F) + 3(A^{\{0\}}\otimes I)f_y Y^{(2)} + 6(A^{\{1\}}c)\otimes(f_y f_y F + f_y g_y F + f_{yy}(F,F))\\
&+6A^{\{2\}}\mathbb{1}\otimes\left(f_{yy}(F,F)+f_y F_y F\right).
\end{aligned}
$$

Now, we can collect all terms in $Y^{(3)}$ to conclude:

$$
\begin{aligned}
Y^{(3)} =& RD\left(c^2+\tilde{c}^2+c\odot\tilde{c}\right)\otimes g_{yy}(F,F)\\
&+RD\left(3(I+\alpha)(A^{\{0\}}c+c^{\{1\}})\right)\otimes g_y f_y F\\
&+RD\left(3(\alpha+\frac{\gamma}{2})RD(c+\tilde{c})+D(c+2\tilde{c})\right)\otimes g_y g_y F\\
&+3(A^{\{0\}}c^2+2A^{\{1\}}c+2A^{\{2\}}\mathbb{1})\otimes f_{yy}(F,F)\\
&+3\left(A^{\{0\}}RD(c+\tilde{c})+2A^{\{1\}}c+2A^{\{2\}}\mathbb{1}\right)\otimes f_y g_y F\\
&+3\left(2A^{\{0\}}A^{\{0\}}c+2A^{\{0\}}c^{\{1\}}+2A^{\{1\}}c+2A^{\{2\}}\mathbb{1}\right)\otimes f_y f_y F.
\end{aligned}
$$

Fourth order We conclude this investigation with the conditions for fourth order. The computations are tedious but straightforward. As we are not aiming for fifth order, we do not go into all the details here.

$$
\begin{aligned}
Z^{(0,3)} =& \alpha Y^{(3)}\\
Z^{(1,2)} =& \frac{1}{3}(\gamma\otimes I)Y^{(3)} + (D\otimes I)g(Z)^{(0,2)} + (\beta^{\{0\}}\otimes I)f^{\{0\}}(Y)^{(2)}\\
&+2(\beta^{\{1\}}\otimes I)f^{\{1\}}(Y)^{(1)} + 2(\beta^{\{2\}}\otimes I)f^{\{2\}}(Y)^{(0)}\\
=&\frac{1}{3}(\gamma\otimes I)Y^{(3)} + (D\tilde{c}^2\otimes I)g_{yy}(F,F) + (D\otimes I)g_y Z^{(0,2)}\\
&+(\beta^{\{0\}}c^2\otimes I)f_{yy}(F,F) + (\beta^{\{0\}}\otimes I)f_y Y^{(2)}
\end{aligned}
$$

$$
\begin{aligned}
&+2(\beta^{\{1\}}\otimes I)f_y^{\{1\}}Y^{(1)}+2(\beta^{\{2\}}\otimes I)f^{\{2\}}(Y)\\
Z^{(2,1)}=&(D\otimes I)g(Z)^{(1,1)}=(D\otimes I)\left(g_{yy}(Z^{(0,1)},Z^{(1,0)})+g_yZ^{(1,1)}\right)\\
=&(D\tilde{c}\odot(c-\tilde{c})\otimes I)g_{yy}(F,F)+(D\otimes I)g_yZ^{(1,1)}\\
Z^{(3,0)}=&(D\otimes I)g(Z)^{(2,0)}=(D\otimes I)\left(g_{yy}(Z^{(1,0)},Z^{(1,0)})+g_yZ^{(2,0)}\right)\\
=&(D(c-\tilde{c})^2\otimes g_{yy}(F,F)+DD(c-\tilde{c})\otimes g_yg_yF
\end{aligned}
$$

Finally, with this, one can derive the fourth-order conditions to be

$$
\begin{aligned}
Y^{(4)}=&\sum_{l=1}^{4}\binom{4}{l}(RD\otimes I)g(Z)^{(l-1,4-l)}\\
&+4(A^{\{0\}}\otimes I)f^{\{0\}}(Y)^{(3)}+12(A^{\{1\}}\otimes I)f^{\{1\}}(Y)^{(2)}\\
&+24(A^{\{2\}}\otimes I)f^{\{2\}}(Y)^{(1)}+24(A^{\{3\}}\otimes I)f^{\{3\}}(Y)^{(0)}.
\end{aligned}
$$

Please note that there holds

$$
\begin{aligned}
f^{\{0\}}(Y)^{(3)}=&f_{yyy}(Y^{(1)},Y^{(1)},Y^{(1)})+3f_{yy}(Y^{(2)},Y^{(1)})+f_yY^{(3)},\\
f^{\{1\}}(Y)^{(2)}=&f_{yyy}(Y^{(1)},Y^{(1)},F)+f_{yy}(Y^{(2)},F)+2f_{yy}(Y^{(1)},F_yY^{(1)})\\
&+f_yF_{yy}(Y^{(1)},Y^{(1)})+f_yF_yY^{(2)}\\
f^{\{2\}}(Y)^{(1)}=&f_{yyy}(Y^{(1)},F,F)+2f_{yy}(F_yY^{(1)},F)+f_{yy}(Y^{(1)},f_yF)\\
&+f_yf_{yy}(Y^{(1)},F)+f_yf_yF_yY^{(1)}+f_{yy}(Y^{(1)},g_yF)\\
&+f_yg_{yy}(Y^{(1)},F)+f_yg_yF_yY^{(1)}.
\end{aligned}
$$

Now, upon observing that

$$
\begin{aligned}
&\sum_{l=1}^{4}\binom{4}{l}(RD\otimes I)g(Z)^{(l-1,4-l)}\\
=&4(RD\otimes I)g(Z)^{(0,3)}+6(RD\otimes I)g(Z)^{(1,2)}+4(RD\otimes I)g(Z)^{(2,1)}+(RD\otimes I)g(Z)^{(3,0)}
\end{aligned}
$$

and

$$
\begin{aligned}
g(Z)^{(0,3)}=&g_{yyy}(Z^{(0,1)},Z^{(0,1)},Z^{(0,1)})+3g_{yy}(Z^{(0,2)},Z^{(0,1)})+g_yZ^{(0,3)}\\
g(Z)^{(1,2)}=&g_{yyy}(Z^{(1,0)},Z^{(0,1)},Z^{(0,1)})+2g_{yy}(Z^{(0,1)},Z^{(1,1)})+g_{yy}(Z^{(1,0)},Z^{(0,2)})+g_yZ^{(1,2)}\\
g(Z)^{(2,1)}=&g_{yyy}(Z^{(1,0)},Z^{(1,0)},Z^{(0,1)})+2g_{yy}(Z^{(1,1)},Z^{(1,0)})+g_{yy}(Z^{(2,0)},Z^{(0,1)})+g_yZ^{(2,1)}\\
g(Z)^{(3,0)}=&g_{yyy}(Z^{(1,0)},Z^{(1,0)},Z^{(1,0)})+3g_{yy}(Z^{(2,0)},Z^{(1,0)})+g_yZ^{(3,0)},
\end{aligned}
$$

one can in a straightforward way obtain the order conditions as in Thm. 1.

Paper V:

Two-derivative schemes for low-Mach flows

Preprint (2025) 1–23

Two-derivative schemes for low-Mach flows

Arjun Thenery Manikantan[a], Jochen Schütz[a]

[a] *Faculty of Sciences & Data Science Institute, Hasselt University, Agoralaan Gebouw D, BE-3590 Diepenbeek, Belgium*

Abstract

In this paper, we analyze two-derivative IMEX (implicit/explicit) schemes applied to the isentropic Euler equations. In particular, we investigate a broad range of stiffness, varying from a weakly compressible regime ($\varepsilon \ll 1$) to a fully compressible regime ($\varepsilon \approx 1$). The high-order discontinuous Galerkin spectral element method is used as spatial discretization. We prove that the considered two-derivative IMEX schemes for the splitting described in [Degond, Tang, 2011] are asymptotically preserving. We show experimentally that the schemes are stable and convergent under a convective CFL condition independent of the stiffness parameter ε.

Keywords: Asymptotic preserving, isentropic Euler, IMEX schemes, discontinuous Galerkin, predictor-corrector schemes

1. Introduction

Solving partial differential equations (PDE) is an ubiquitous task in computational modelling of all sorts of physical phenomena. In particular, it is of greatest importance in fluid dynamics. Approximating PDEs is a widely studied field, yet there are still open issues and challenges. Solving time-dependent conservation equations in fluid dynamics is highly demanding and, at the same time, computationally expensive. Therefore, new strategies that utilize the most modern computational architecture are constantly being developed. Generally, the procedure of finding a numerical solution to these equations starts with discretizing the spatial derivatives in the PDE, resulting in a time-dependent system of ordinary differential equations (ODE). Then, well-suited timestepping procedures are employed to solve this system of ODEs numerically.

In many cases, this ODE is stiff, with stiffness arising from, e.g., the equations themselves or the spatial discretization. Typically, stiffness manifests itself through vast differences in orders of magnitude of the eigenvalues of the underlying system. In this work, we consider a model for compressible flow of form

$$\mathbf{w}_t + \nabla \cdot \mathbf{F}(\mathbf{w}) = 0, \tag{1}$$

with a flux term $\mathbf{F}(\mathbf{w})$ that depends on the scaled reference Mach number ε that is supposed to be small, see [1, Sec. 2]. In this prototypical equation, the stiffness is hence characterized by ε. In particular, for $\varepsilon \to 0$, the eigenvalues are of vastly different size, see Eq. (4).

Widely used timestepping schemes for compressible fluid equations are explicit schemes due to their easy implementation and their high efficiency for non-stiff equations. However, the stability restriction that arises from the CFL

Email addresses: arjun.thenerymanikantan@uhasselt.be (Arjun Thenery Manikantan), jochen.schuetz@uhasselt.be (Jochen Schütz)

conditions [2] can often force unfeasible bounds on the timestep size. In the case of equation (3), it mandates choosing the timestep size Δt to be $O(\varepsilon\Delta x)$, where Δx is the spatial mesh size. Hence, it is impractical to use a timestep

$$\Delta t = O(\varepsilon\Delta x), \quad \text{as} \quad \varepsilon \to 0.$$

In this situation, using implicit timestepping schemes [3] helps to overcome timestep restrictions. However, it brings on the cumbersome task of solving large non-linear algebraic systems of equations. Additionally, if the fluxes are highly non-linear and/or very stiff, the implicit solver may require many iterations to converge (or not converge at all), resulting in a longer execution time for the timestepping scheme.

Considering the challenges of timestepping schemes mentioned above, methods that simultaneously leverage the computational efficiency of an explicit scheme and the unconstrained timestep size requirement from implicit schemes, are highly demanding. The aforementioned methods that use a split flux evaluation are the implicit-explicit (IMEX) time-stepping schemes, see, e.g., [4, 5, 6, 7, 8, 9]. The flux function is generally split into stiff and non-stiff parts, i.e.,

$$\mathbf{w}_t + \nabla \cdot \Big(\underbrace{\mathbf{F}_{\mathrm{I}}(\mathbf{w})}_{\text{stiff}} + \underbrace{\mathbf{F}_{\mathrm{E}}(\mathbf{w})}_{\text{non-stiff}} \Big) = 0. \tag{2}$$

Then, the stiff and non-stiff parts are evaluated implicitly and explicitly, respectively. The effective way would be to choose a (nearly) linear stiff part, making the implicit solver work faster, and a non-stiff part, which can give a more reasonable timestep restriction that is independent of ε, i.e.,

$$\Delta t = O(\Delta x).$$

Classical time integration schemes are of the one-derivative type, i.e., they rely on the first temporal derivative ($\mathbf{w}_t$) of the problem only. Adding intermediate stages or utilizing solutions from the previous timesteps is the only way to increase their order of convergence. Devising higher-order schemes using these often comes with the strenuous chore of solving many order conditions. Also, utilizing solutions from previous timesteps beyond a threshold can severely affect stability properties. Considering multi-derivative schemes results in simpler order conditions and more flexibility over coefficients that are to be found. This class of schemes utilizes higher-order derivatives of the problem ($\mathbf{w}_{tt}$ and more), see Eq. (9). For some multiderivative schemes, we refer to [10, 11, 12, 13, 14, 15] and [16, 17, 18] for some IMEX multi-derivative schemes.

In [16], a two-derivative asymptotic preserving (AP) IMEX scheme for ordinary differential equations, that works in a predictor-corrector fashion, has been developed. The scheme has been extended later, see [19, 20, 21], and termed Hermite-Birkhoff Predictor-Corrector (HBPC) scheme. When solving PDEs, time stepping schemes are to be combined with higher-order spatial discretizations for highly accurate numerical solutions. In [19, 22], the authors have combined the HBPC scheme with the higher-order Discontinuous Galerkin Spectral Element Methods (DGSEM) [23] for viscous compressible flow equations.

When numerically solving the singularly perturbed problem (3) for $\varepsilon \ll 1$, there is the additional difficulty that for $\varepsilon \to 0$, the equations change type (hence singularly perturbed). To be efficient and stable, both temporal and spatial discretization must take this behavior into account, a property that is called 'asymptotic preservation' (AP) [24, 25, 26, 27, 16, 28]. An asymptotically preserving scheme means that the $\varepsilon \to 0$ limit of the numerical solution is consistent with the solution in the singular limit of the problem (6) for finite values of Δt and Δx.

In this paper, we analyze and showcase the effect of IMEX-HBPC schemes [16, 19, 29] combined with DGSEM [23] spatial discretization on the low-Mach isentropic Euler equations. Considering the reference Mach number ε, the latter are given by

$$\mathbf{w}_t + \nabla \cdot \mathbf{F}(\mathbf{w}) = 0, \quad \text{where} \quad \mathbf{w} = \begin{pmatrix} \rho \\ \rho\mathbf{v} \end{pmatrix} \quad \text{and} \quad \mathbf{F}(\mathbf{w}) = \begin{pmatrix} \rho\mathbf{v} \\ \rho\mathbf{v} \otimes \mathbf{v} + \frac{p}{\varepsilon^2}\mathbf{Id} \end{pmatrix}. \tag{3}$$

The system is closed with the equation of state $p(\rho) := \kappa\rho^\gamma$, where $\kappa > 0$ and $\gamma > 1$.[1] The eigenvalues of the Jacobian in the direction of some vector $\mathbf{n}$, i.e., $\left(\frac{\partial F}{\partial \mathbf{w}} \cdot \mathbf{n}\right)$ are in two dimensions given by

$$\lambda_1 = \mathbf{v} \cdot \mathbf{n} \quad \text{and} \quad \lambda_{2,3} = \mathbf{v} \cdot \mathbf{n} \pm \frac{\sqrt{p'(\rho)}}{\varepsilon}. \tag{4}$$

[1] Please note that more general equations of state are possible, as long as $p'(\rho) > 0$ is guaranteed. The modifications to the analysis are straightforward.

To achieve a well-behaved limit of Eq. (3) as $\varepsilon \to 0$ [30], we consider well-prepared initial data

$$\rho \big|_{t=0} = \text{const} + O(\varepsilon^2) \quad \text{and} \quad \nabla \cdot (\rho \mathbf{v}) \big|_{t=0} = O(\varepsilon), \tag{5}$$

and we assume that the solution $\mathbf{w}$ of (3) has a Hilbert expansion, given by

$$\mathbf{w} = \mathbf{w}_{(0)} + \varepsilon \mathbf{w}_{(1)} + \varepsilon^2 \mathbf{w}_{(2)} + O(\varepsilon^3).$$

Formally, for $\varepsilon \to 0$, (3) yields the incompressible Euler equations given by

$$\begin{aligned} \rho_{(0)} \equiv \text{const} > 0, \quad \nabla \cdot \mathbf{v}_{(0)} = 0, \\ (\mathbf{v}_{(0)})_t + \nabla \cdot (\mathbf{v}_{(0)} \otimes \mathbf{v}_{(0)}) + \frac{\nabla p_{(2)}}{\rho_{(0)}} = 0. \end{aligned} \tag{6}$$

There exist several well-studied splittings $\mathbf{F}_\mathrm{I}$ and $\mathbf{F}_\mathrm{E}$ in literature, see, e.g., [1, 31, 32] and the references therein. As developing or comparing splittings is not at the core of this work, we use the splitting of [31], given by

$$\mathbf{F}_\mathrm{I}(\mathbf{w}) = \begin{pmatrix} \rho \mathbf{v} \\ \frac{1-\varepsilon^2}{\varepsilon^2} p \mathbf{Id} \end{pmatrix} \quad \text{and} \quad \mathbf{F}_\mathrm{E}(\mathbf{w}) = \begin{pmatrix} 0 \\ \rho \mathbf{v} \otimes \mathbf{v} + p \mathbf{Id} \end{pmatrix}. \tag{7}$$

Please note that other splitting choices are possible, and will not alter the algorithm significantly. We show that the combination of HBPC with DGSEM, applied to this splitting, results in an asymptotically preserving algorithm. To our knowledge, this is the first result in this direction for a two-derivative scheme. The result can be extended to more elaborate equations representing compressible flow.

The sections of this paper are structured as follows: In Sec. 2, we introduce the two-derivative IMEX-Taylor method followed by the semi-discrete and fully-discrete analysis of the method in Sec. 2.1 and Sec. 2.2 to show the asymptotic preservation. Then, the AP analysis of the IMEX-HBPC schemes is given in Sec. 3. In Sec. 4, we focus on the solution of the algebraic system of equations. Subsequently, numerical results are shown in Sec. 5. The paper concludes in Sec. 6 with an outlook.

2. Two-derivative IMEX-Taylor method

The PDE (1) can be formally cast into an ODE of form

$$\mathbf{w}_t = -\nabla \cdot \mathbf{F}(\mathbf{w}) =: \mathbf{R}^{(1)}(\mathbf{w}) \tag{8}$$

in some infinite dimensional space. The timestepping schemes considered in this work require the second time derivative of the equation. Hence, we have

$$\mathbf{w}_{tt} = \Big(-\nabla \cdot \mathbf{F}(\mathbf{w}) \Big)_t = -\nabla \cdot \left(\frac{\partial \mathbf{F}(\mathbf{w})}{\partial \mathbf{w}} \mathbf{w}_t \right) = -\nabla \cdot \left(\frac{\partial \mathbf{F}(\mathbf{w})}{\partial \mathbf{w}} \mathbf{R}^{(1)}(\mathbf{w}) \right) =: \mathbf{R}^{(2)}(\mathbf{w}). \tag{9}$$

Consider a general splitting of the flux term in (3) into implicit (**I**) and explicit (**E**) parts, given by

$$\mathbf{F}(\mathbf{w}) = \mathbf{F}_\mathrm{I}(\mathbf{w}) + \mathbf{F}_\mathrm{E}(\mathbf{w}), \tag{10}$$

with $\mathbf{F}_\mathrm{I}(\mathbf{w})$ and $\mathbf{F}_\mathrm{E}(\mathbf{w})$ as defined in (7). Then, in Tab. 1, we define similar notations as in Eqs. (8) and (9) for the split flux (10). Using the definitions from Tab. 1, we have the split flux PDE cast into the ODE

$$\mathbf{w}_t = \mathbf{R}_\mathrm{I}^{(1)}(\mathbf{w}) + \mathbf{R}_\mathrm{E}^{(1)}(\mathbf{w}) \tag{11}$$

for the first derivative and

$$\mathbf{w}_{tt} = \mathbf{R}_\mathrm{I}^{(2)}(\mathbf{w}, \mathbf{w}) + \mathbf{R}_\mathrm{E}^{(2)}(\mathbf{w}) \tag{12}$$

for the second derivative.

Notation	Definition w.r.t a general splitting
$\mathbf{R}^{(1)}(\mathbf{w})$	$-\nabla \cdot \mathbf{F}(\mathbf{w})$
$\mathbf{R}_{\mathrm{E}}^{(1)}(\mathbf{w})$	$-\nabla \cdot \mathbf{F}_{\mathrm{E}}(\mathbf{w})$
$\mathbf{R}_{\mathrm{I}}^{(1)}(\mathbf{w})$	$-\nabla \cdot \mathbf{F}_{\mathrm{I}}(\mathbf{w})$
$\mathbf{R}^{(2)}(\mathbf{w})$	$-\nabla \cdot \left(\frac{\partial \mathbf{F}(\mathbf{w})}{\partial \mathbf{w}} \mathbf{R}^{(1)}(\mathbf{w})\right)$
$\mathbf{R}_{\mathrm{E}}^{(2)}(\mathbf{w})$	$-\nabla \cdot \left(\frac{\partial \mathbf{F}_{\mathrm{E}}(\mathbf{w})}{\partial \mathbf{w}} \mathbf{R}^{(1)}(\mathbf{w})\right) = -\nabla \cdot \left(\frac{\partial \mathbf{F}_{\mathrm{E}}(\mathbf{w})}{\partial \mathbf{w}} \left(\mathbf{R}_{\mathrm{I}}^{(1)}(\mathbf{w}) + \mathbf{R}_{\mathrm{E}}^{(1)}(\mathbf{w})\right)\right)$
$\mathbf{R}_{\mathrm{I}}^{(2)}(\bar{\mathbf{w}}, \mathbf{w})$	$-\nabla \cdot \left(\frac{\partial \mathbf{F}_{\mathrm{I}}(\mathbf{w})}{\partial \mathbf{w}} \left(\mathbf{R}_{\mathrm{I}}^{(1)}(\mathbf{w}) + \mathbf{R}_{\mathrm{E}}^{(1)}(\bar{\mathbf{w}})\right)\right)$, (Refer to Rem. 1)

Table 1. Definition of notations used in paper.

Remark 1. *The idea of IMEX time-stepping schemes is that the explicit flux terms are never inverted. However, when it comes to multi-derivative schemes, the derivatives of the implicit flux term are forced to carry contributions from the explicit flux due to their definition, as in the implicit part of* $\mathbf{w}_{tt}$ *given by*

$$-\nabla \cdot \left(\frac{\partial \mathbf{F}_I(\mathbf{w})}{\partial \mathbf{w}} \left(\mathbf{R}_I^{(1)}(\mathbf{w}) + \mathbf{R}_E^{(1)}(\mathbf{w})\right)\right). \tag{13}$$

Hence, the term (13) *indirectly inverts the explicit flux contributions, adversely affecting the schemes. To tackle this condition of the explicit flux contribution on the higher derivatives of the implicit flux terms, the authors in [29] have introduced a slightly modified definition of the implicit part of* $\mathbf{w}_{tt}$ *for the use in IMEX schemes. It is given by*

$$-\nabla \cdot \left(\frac{\partial \mathbf{F}_I(\mathbf{w})}{\partial \mathbf{w}} \left(\mathbf{R}_I^{(1)}(\mathbf{w}) + \mathbf{R}_E^{(1)}(\bar{\mathbf{w}})\right)\right) =: \mathbf{R}_I^{(2)}(\bar{\mathbf{w}}, \mathbf{w}), \tag{14}$$

where $\bar{\mathbf{w}}$ *will be an* ***explicitly known*** *value in the algorithm, hence preventing explicit contributions from getting inverted, see, e.g., Eq.* (15) *below this remark. We utilize the latter definition* (14) *for the implicit part of* $\mathbf{w}_{tt}$ *in all the schemes and its analysis provided in the paper.*

Algorithm 1 (Two-derivative IMEX-Taylor)**.** *Given the solution* $\mathbf{w}^n$ *at time instance* t^n*, the solution update at* t^{n+1} *by the two-derivative IMEX-Taylor method is given by*

$$\mathbf{w}^{n+1} := \mathbf{w}^n + \Delta t \left[\mathbf{R}_E^{(1)}(\mathbf{w}^n) + \mathbf{R}_I^{(1)}(\mathbf{w}^{n+1})\right] + \frac{\Delta t^2}{2} \left[\mathbf{R}_E^{(2)}(\mathbf{w}^n) - \mathbf{R}_I^{(2)}(\mathbf{w}^n, \mathbf{w}^{n+1})\right]. \tag{15}$$

2.1. Semi-discrete asymptotic analysis

We assume that the differential equations are defined on the domain $\Omega \times [0, T_{end}]$ with $\Omega \subset \mathbb{R}^2$ and periodic boundary conditions. As stated in (7), we consider the splitting from [31] (called DeTa in the sequel),

$$\underbrace{\begin{pmatrix} \rho \\ \rho \mathbf{v} \end{pmatrix}_t}_{\mathbf{w}_t} + \nabla \cdot \underbrace{\begin{pmatrix} \rho \mathbf{v} \\ \frac{1-\varepsilon^2}{\varepsilon^2} p \mathbf{Id} \end{pmatrix}}_{\mathbf{F}_{\mathrm{I}}(\mathbf{w})} + \nabla \cdot \underbrace{\begin{pmatrix} 0 \\ \rho \mathbf{v} \otimes \mathbf{v} + p \mathbf{Id} \end{pmatrix}}_{\mathbf{F}_{\mathrm{E}}(\mathbf{w})} = 0. \tag{16}$$

Take $\mathbf{w} = (\rho, u, v)^T$ and $\bar{\mathbf{w}} = (\bar{\rho}, \bar{u}, \bar{v})^T$ with the pressures p and $\bar{p}$ respectively. The derivative of the pressure term is denoted as $p' = \kappa \gamma \rho^{\gamma - 1}$. Expanded definitions of the divergence terms mentioned in Tab. 1 for the DeTa splitting (16) are given in Tab. 2.

Notation	Definition w.r.t DeTa splitting (16)
$\mathbf{R}^{(1)}(\mathbf{w})$	$-\begin{pmatrix} \rho u \\ \rho u^2 + \frac{1}{\varepsilon^2} p \\ \rho uv \end{pmatrix}_x - \begin{pmatrix} \rho v \\ \rho uv \\ \rho v^2 + \frac{1}{\varepsilon^2} p \end{pmatrix}_y$
$\mathbf{R}^{(1)}_{\mathrm{E}}(\mathbf{w})$	$-\begin{pmatrix} 0 \\ \rho u^2 + p \\ \rho uv \end{pmatrix}_x - \begin{pmatrix} 0 \\ \rho uv \\ \rho v^2 + p \end{pmatrix}_y$
$\mathbf{R}^{(1)}_{\mathrm{I}}(\mathbf{w})$	$-\begin{pmatrix} \rho u \\ \frac{1-\varepsilon^2}{\varepsilon^2} p \\ 0 \end{pmatrix}_x - \begin{pmatrix} \rho v \\ 0 \\ \frac{1-\varepsilon^2}{\varepsilon^2} p \end{pmatrix}_y$
$\mathbf{R}^{(2)}(\mathbf{w})$	$\begin{pmatrix} \left(\rho u^2 + \frac{p}{\varepsilon^2}\right)_x + (\rho uv)_y \\ \left(-u^2 + \frac{p'}{\varepsilon^2}\right)\left((\rho u)_x + (\rho v)_y\right) + 2u\left(\left(\rho u^2 + \frac{p}{\varepsilon^2}\right)_x + (\rho uv)_y\right) \\ -uv\left((\rho u)_x + (\rho v)_y\right) + v\left(\left(\rho u^2 + \frac{p}{\varepsilon^2}\right)_x + (\rho uv)_y\right) + u\left((\rho uv)_x + \left(\rho v^2 + \frac{p}{\varepsilon^2}\right)_y\right) \end{pmatrix}_x$ + $\begin{pmatrix} (\rho uv)_x + \left(\rho v^2 + \frac{p}{\varepsilon^2}\right)_y \\ -uv\left((\rho u)_x + (\rho v)_y\right) + v\left(\left(\rho u^2 + \frac{p}{\varepsilon^2}\right)_x + (\rho uv)_y\right) + u\left((\rho uv)_x + \left(\rho v^2 + \frac{p}{\varepsilon^2}\right)_y\right) \\ \left(-v^2 + \frac{p'}{\varepsilon^2}\right)\left((\rho u)_x + (\rho v)_y\right) + 2v\left((\rho uv)_x + \left(\rho v^2 + \frac{p}{\varepsilon^2}\right)_y\right) \end{pmatrix}_y$
$\mathbf{R}^{(2)}_{\mathrm{E}}(\mathbf{w})$	$\begin{pmatrix} 0 \\ \left(-u^2 + p'\right)\left((\rho u)_x + (\rho v)_y\right) + 2u\left(\left(\rho u^2 + \frac{p}{\varepsilon^2}\right)_x + (\rho uv)_y\right) \\ -uv\left((\rho u)_x + (\rho v)_y\right) + v\left(\left(\rho u^2 + \frac{p}{\varepsilon^2}\right)_x + (\rho uv)_y\right) + u\left((\rho uv)_x + \left(\rho v^2 + \frac{p}{\varepsilon^2}\right)_y\right) \end{pmatrix}_x$ + $\begin{pmatrix} 0 \\ -uv\left((\rho u)_x + (\rho v)_y\right) + v\left(\left(\rho u^2 + \frac{p}{\varepsilon^2}\right)_x + (\rho uv)_y\right) + u\left((\rho uv)_x + \left(\rho v^2 + \frac{p}{\varepsilon^2}\right)_y\right) \\ \left(-v^2 + p'\right)\left((\rho u)_x + (\rho v)_y\right) + 2v\left((\rho uv)_x + \left(\rho v^2 + \frac{p}{\varepsilon^2}\right)_y\right) \end{pmatrix}_y$
$\mathbf{R}^{(2)}_{\mathrm{I}}(\bar{\mathbf{w}}, \mathbf{w})$	$\begin{pmatrix} \frac{1-\varepsilon^2}{\varepsilon^2} p_x + \left(\left(\bar{\rho}\bar{u}^2 + \bar{p}\right)_x + (\bar{\rho}\bar{u}\bar{v})_y\right) \\ \frac{1-\varepsilon^2}{\varepsilon^2} p'\left((\rho u)_x + (\rho v)_y\right) \\ 0 \end{pmatrix}_x$ + $\begin{pmatrix} \frac{1-\varepsilon^2}{\varepsilon^2} p_y + \left((\bar{\rho}\bar{u}\bar{v})_x + \left(\bar{\rho}\bar{v}^2 + \bar{p}\right)_y\right) \\ 0 \\ \frac{1-\varepsilon^2}{\varepsilon^2} p'\left((\rho u)_x + (\rho v)_y\right) \end{pmatrix}_y$

Table 2. Expansion of the divergence terms mentioned in Tab. 1 for the DeTa splitting (16). Let $\mathbf{w} = (\rho, u, v)^T$ and $\bar{\mathbf{w}} = (\bar{\rho}, \bar{u}, \bar{v})^T$ with the pressures p and $\bar{p}$ respectively. The derivative of the pressure is denoted as p', where $p' = \kappa\gamma\rho^{\gamma-1}$.

Definition 1 (Well-prepared numerical solution)**.** *Let* $\mathbf{w}^n$ *be the numerical solution of Eq.*(3) *depending on a small*

parameter $\varepsilon < 1$. We say that $\mathbf{w}^n$ admits an asymptotic expansion in ε at time t^n if

$$\mathbf{w}^n = \mathbf{w}^n_{(0)} + \varepsilon\, \mathbf{w}^n_{(1)} + \varepsilon^2\, \mathbf{w}^n_{(2)} + O(\varepsilon^3),$$

where $\mathbf{w}^n_{(k)}$, $k = 0, 1, 2, \ldots$, are the asymptotic coefficients. Then, the solution at t^n is said to be well-prepared if

$$\rho^n = const + O(\varepsilon^2), \quad \text{and} \quad \nabla \cdot \mathbf{v}^n = O(\varepsilon).$$

Lemma 1. *The quantities $\rho^{n+1}_{(0)}$ and $\rho^{n+1}_{(1)}$ in the asymptotic expansion of ρ^{n+1} in Eq. (17) are constant in space, with an assumption of well-prepared solution at t^n and periodic boundary conditions.*

Proof. Substituting the corresponding flux terms from Tab. 2 in the time-stepping scheme (15), we get

$$\begin{aligned}
\begin{pmatrix}\rho\\ \rho u\\ \rho v\end{pmatrix}^{n+1} &= \begin{pmatrix}\rho\\ \rho u\\ \rho v\end{pmatrix}^{n} - \Delta t \Bigg(\underbrace{\begin{pmatrix}(\rho u)_x + (\rho v)_y\\ \frac{1-\varepsilon^2}{\varepsilon^2} p_x\\ \frac{1-\varepsilon^2}{\varepsilon^2} p_y\end{pmatrix}^{n+1}}_{-\mathbf{R}^{(1)}_{\mathrm{I}}(\mathbf{w}^{n+1})} + \underbrace{\begin{pmatrix}0\\ \left(\rho u^2 + p\right)_x + (\rho u v)_y\\ (\rho u v)_x + \left(\rho v^2 + p\right)_y\end{pmatrix}^{n}}_{-\mathbf{R}^{(1)}_{\mathrm{E}}(\mathbf{w}^{n})} \Bigg) \\
&\quad - \frac{\Delta t^2}{2} \Bigg(\underbrace{\begin{pmatrix} \left\{\frac{1-\varepsilon^2}{\varepsilon^2} p_x\right\}^{n+1} + \left\{\left(\rho u^2 + p\right)_x + (\rho u v)_y\right\}^{n} \\ \left\{\frac{1-\varepsilon^2}{\varepsilon^2} p' \left((\rho u)_x + (\rho v)_y\right)\right\}^{n+1} \\ 0 \end{pmatrix}^{n}_{x} + \begin{pmatrix} \left\{\frac{1-\varepsilon^2}{\varepsilon^2} p_y\right\}^{n+1} + \left\{(\rho u v)_x + \left(\rho v^2 + p\right)_y\right\}^{n} \\ 0 \\ \left\{\frac{1-\varepsilon^2}{\varepsilon^2} p' \left((\rho u)_x + (\rho v)_y\right)\right\}^{n+1} \end{pmatrix}^{n}_{y}}_{\mathbf{R}^{(2)}_{\mathrm{I}}(\mathbf{w}^{n}, \mathbf{w}^{n+1})} \Bigg) \\
&\quad + \frac{\Delta t^2}{2} \Bigg(\underbrace{\begin{pmatrix} 0 \\ \left(-u^2 + p'\right)\left((\rho u)_x + (\rho v)_y\right) + 2u\left(\left(\rho u^2 + \frac{p}{\varepsilon^2}\right)_x + (\rho u v)_y\right) \\ -uv\left((\rho u)_x + (\rho v)_y\right) + v\left(\left(\rho u^2 + \frac{p}{\varepsilon^2}\right)_x + (\rho u v)_y\right) + u\left((\rho u v)_x + \left(\rho v^2 + \frac{p}{\varepsilon^2}\right)_y\right) \end{pmatrix}^{n}_{x} \\
&\qquad + \begin{pmatrix} 0 \\ -uv\left((\rho u)_x + (\rho v)_y\right) + v\left(\left(\rho u^2 + \frac{p}{\varepsilon^2}\right)_x + (\rho u v)_y\right) + u\left((\rho u v)_x + \left(\rho v^2 + \frac{p}{\varepsilon^2}\right)_y\right) \\ \left(-v^2 + p'\right)\left((\rho u)_x + (\rho v)_y\right) + 2v\left((\rho u v)_x + \left(\rho v^2 + \frac{p}{\varepsilon^2}\right)_y\right) \end{pmatrix}^{n}_{y}}_{\mathbf{R}^{(2)}_{\mathrm{E}}(\mathbf{w}^{n})} \Bigg) .
\end{aligned} \tag{17}$$

Consider the $O(\varepsilon^{-2})$ terms from (17) with well-preparedness assumption on $\mathbf{w}^n$. From the density equation, we have

$$\partial_{xx} p^{n+1}_{(0)} + \partial_{yy} p^{n+1}_{(0)} = 0, \tag{18}$$

and from the momentum equations, we have

$$\begin{aligned}
-\Delta t \partial_x p^{n+1}_{(0)} - \frac{\Delta t^2}{2} \partial_x \left(p'_{(0)} \left((\rho_{(0)} u_{(0)})_x + (\rho_{(0)} v_{(0)})_y \right) \right)^{n+1} = 0, \\
-\Delta t \partial_y p^{n+1}_{(0)} - \frac{\Delta t^2}{2} \partial_y \left(p'_{(0)} \left((\rho_{(0)} u_{(0)})_x + (\rho_{(0)} v_{(0)})_y \right) \right)^{n+1} = 0.
\end{aligned} \tag{19}$$

Similarly considering the $O(\varepsilon^{-1})$ terms, we get

$$\partial_{xx} p^{n+1}_{(1)} + \partial_{yy} p^{n+1}_{(1)} = 0 \tag{20}$$

from the density equation and

$$\begin{aligned}
-\Delta t \partial_x p^{n+1}_{(1)} - \frac{\Delta t^2}{2} \partial_x \left(p'_{(1)} \left((\rho_{(0)} u_{(0)})_x + (\rho_{(0)} v_{(0)})_y \right) + p'_{(0)} \left((\rho_{(0)} u_{(1)} + \rho_{(1)} u_{(0)})_x + (\rho_{(0)} v_{(1)} + \rho_{(1)} v_{(0)})_y \right) \right)^{n+1} = 0, \\
-\Delta t \partial_y p^{n+1}_{(1)} - \frac{\Delta t^2}{2} \partial_y \left(p'_{(1)} \left((\rho_{(0)} u_{(0)})_x + (\rho_{(0)} v_{(0)})_y \right) + p'_{(0)} \left((\rho_{(0)} u_{(1)} + \rho_{(1)} u_{(0)})_x + (\rho_{(0)} v_{(1)} + \rho_{(1)} v_{(0)})_y \right) \right)^{n+1} = 0
\end{aligned} \tag{21}$$

from the momentum equations. The $O(1)$ term from the density equation gives

$$\rho_{(0)}^{n+1} = \rho_{(0)}^{n} - \Delta t \left((\rho_{(0)}u_{(0)})_x + (\rho_{(0)}v_{(0)})_y\right)^{n+1} - \frac{\Delta t^2}{2}\left(\partial_{xx}p_{(2)}^{n+1} + \partial_x\left(\left(\rho_{(0)}u_{(0)}^2\right)_x + (\rho_{(0)}u_{(0)}v_{(0)})_y\right)^n\right) - \frac{\Delta t^2}{2}\left(\partial_{yy}p_{(2)}^{n+1} + \partial_y\left((\rho_{(0)}u_{(0)}v_{(0)})_x + \left(\rho_{(0)}v_{(0)}^2\right)_y\right)^n\right), \tag{22}$$

considering well-prepared assumptions at t^n. Since the boundary conditions are set to be periodic, it implies the existence of Fourier series expansion for $p_{(0)}^{n+1}$ and $p_{(1)}^{n+1}$. Substituting the Fourier series solution in (18) and (20) forces zero values to the Fourier coefficients. Hence it can be proved that the solutions to (18) and (20) are independent of space variables. Therefore the equation of state $p(\rho) := \kappa\rho^\gamma$ implies that density terms $\rho_{(0)}^{n+1}$ and $\rho_{(1)}^{n+1}$ are independent of space variables.

Now integrating Eq. (22) over the spatial domain, we obtain

$$\int_\Omega \frac{\rho_{(0)}^{n+1} - \rho_{(0)}^{n}}{\Delta t} = -\int_\Omega \rho_{(0)}^{n+1}\,\nabla\cdot(\mathbf{v}_{(0)})^{n+1} - \frac{\Delta t}{2}\int_\Omega \nabla\cdot(\nabla p_{(2)})^{n+1} - \frac{\Delta t}{2}\int_\Omega \rho_{(0)}^{n}\,\nabla\cdot\left(\left(u_{(0)}^2\right)_x + (u_{(0)}v_{(0)})_y, (u_{(0)}v_{(0)})_x + \left(v_{(0)}^2\right)_y\right)^n$$

$$\left(\rho_{(0)}^{n+1} - \rho_{(0)}^{n}\right)\frac{|\Omega|}{\Delta t} = -\rho_{(0)}^{n+1}\int_{\partial\Omega}\mathbf{n}\cdot\mathbf{v}_{(0)}^{n+1} - \frac{\Delta t}{2}\int_{\partial\Omega}\mathbf{n}\cdot(\nabla p_{(2)})^{n+1} - \frac{\Delta t}{2}\rho_{(0)}^{n}\int_{\partial\Omega}\mathbf{n}\cdot\left(\left(u_{(0)}^2\right)_x + (u_{(0)}v_{(0)})_y, (u_{(0)}v_{(0)})_x + \left(v_{(0)}^2\right)_y\right)^n$$

$$\Rightarrow \quad \rho_{(0)}^{n+1} - \rho_{(0)}^{n} = 0 \quad (\textit{due to periodic boundary conditions}). \tag{23}$$

Similarly using the $O(\varepsilon)$ terms from the density equation we can obtain

$$\rho_{(1)}^{n+1} - \rho_{(1)}^{n} = 0. \tag{24}$$

Hence from Eq. (23) and Eq. (24), we get

$$\rho^{n+1} = \text{const} + O(\varepsilon^2). \tag{25}$$

□

Lemma 2. *The quantity $\mathbf{v}_{(0)}^{n+1}$ is divergence free under the assumptions of Lemma 1.*

Proof. From Eq. (19), we get

$$\left.\begin{aligned}\partial_x\left((u_{(0)})_x + (v_{(0)})_y\right)^{n+1} &= 0\\ \partial_y\left((u_{(0)})_x + (v_{(0)})_y\right)^{n+1} &= 0\end{aligned}\right\} \Rightarrow \nabla\cdot\mathbf{v}_{(0)}^{n+1} = C,$$

where $C \in \mathbb{R}$ is a constant. Now substituting $\nabla\cdot\mathbf{v}_{(0)}^{n+1} = C$ in (22) and integrating over the spatial domain gives

$$0 = -\int_\Omega \rho_{(0)}^{n+1}\,C - \frac{\Delta t}{2}\int_\Omega \nabla\cdot(\nabla p_{(2)})^{n+1} - \frac{\Delta t}{2}\int_\Omega \rho_{(0)}^{n}\,\nabla\cdot\left(\left(u_{(0)}^2\right)_x + (u_{(0)}v_{(0)})_y, (u_{(0)}v_{(0)})_x + \left(v_{(0)}^2\right)_y\right)^n$$

$$0 = -\rho_{(0)}^{n+1}C\,|\Omega| - \frac{\Delta t}{2}\int_{\partial\Omega}\mathbf{n}\cdot(\nabla p_{(2)})^{n+1} - \frac{\Delta t}{2}\rho_{(0)}^{n}\int_{\partial\Omega}\mathbf{n}\cdot\left(\left(u_{(0)}^2\right)_x + (u_{(0)}v_{(0)})_y, (u_{(0)}v_{(0)})_x + \left(v_{(0)}^2\right)_y\right)^n$$

$$\Rightarrow \quad 0 = C \quad (\textit{due to periodic boundary conditions}).$$

Hence we get

$$\nabla\cdot\mathbf{v}^{n+1} = O(\varepsilon). \tag{26}$$

□

Lemma 3. *The $O(1)$ terms from the momentum equations in Eq. (17) are the semi-discretization of the limiting equation*

$$(\mathbf{v}_{(0)})_t + \frac{\nabla p_{(2)}}{\rho_{(0)}} + \nabla \cdot (\mathbf{v}_{(0)} \otimes \mathbf{v}_{(0)}) = 0, \tag{27}$$

under the assumptions of Lemma 1, with an implicit treatment of pressure term $p_{(2)}$ and an explicit treatment of the tensor product $(\mathbf{v}_{(0)} \otimes \mathbf{v}_{(0)})$.

Proof. Consider the $O(1)$ terms from the momentum equations along with the results from Eq. (25) and Eq. (26) with the stiffness parameter $\varepsilon \to 0$,

$$\begin{aligned}
\rho_{(0)}^{n+1} u_{(0)}^{n+1} &= \rho_{(0)}^{n} u_{(0)}^{n} - \Delta t \partial_x p_{(2)}^{n+1} - \Delta t \rho_{(0)}^{n} \left(\left(u_{(0)}^2 \right)_x + \left(u_{(0)} v_{(0)} \right)_y \right)^n - \frac{\Delta t^2}{2} \partial_x \left(\left(p' \left((\rho u)_x + (\rho v)_y \right) \right)_{(2)} \right)^{n+1} \\
&\quad + \frac{\Delta t^2}{2} \rho_{(0)}^{n} \partial_x \left(2 u_{(0)} \left(\left(u_{(0)}^2 + \frac{p_{(2)}}{\rho_{(0)}} \right)_x + \left(u_{(0)} v_{(0)} \right)_y \right) \right)^n \\
&\quad + \frac{\Delta t^2}{2} \rho_{(0)}^{n} \partial_y \left(v_{(0)} \left(\left(u_{(0)}^2 + \frac{p_{(2)}}{\rho_{(0)}} \right)_x + \left(u_{(0)} v_{(0)} \right)_y \right) + u_{(0)} \left(\left(u_{(0)} v_{(0)} \right)_x + \left(v_{(0)}^2 + \frac{p_{(2)}}{\rho_{(0)}} \right)_y \right) \right)^n \\
\rho_{(0)}^{n+1} v_{(0)}^{n+1} &= \rho_{(0)}^{n} v_{(0)}^{n} - \Delta t \partial_y p_{(2)}^{n+1} - \Delta t \rho_{(0)}^{n} \left(\left(u_{(0)} v_{(0)} \right)_x + \left(v_{(0)}^2 \right)_y \right)^n - \frac{\Delta t^2}{2} \partial_y \left(\left(p' \left((\rho u)_x + (\rho v)_y \right) \right)_{(2)} \right)^{n+1} \\
&\quad + \frac{\Delta t^2}{2} \rho_{(0)}^{n} \partial_x \left(v_{(0)} \left(\left(u_{(0)}^2 + \frac{p_{(2)}}{\rho_{(0)}} \right)_x + \left(u_{(0)} v_{(0)} \right)_y \right) + u_{(0)} \left(\left(u_{(0)} v_{(0)} \right)_x + \left(v_{(0)}^2 + \frac{p_{(2)}}{\rho_{(0)}} \right)_y \right) \right)^n \\
&\quad + \frac{\Delta t^2}{2} \rho_{(0)}^{n} \partial_y \left(2 v_{(0)} \left(\left(u_{(0)} v_{(0)} \right)_x + \left(v_{(0)}^2 + \frac{p_{(2)}}{\rho_{(0)}} \right)_y \right) \right)^n,
\end{aligned}$$

where

$$\left(p' \left((\rho u)_x + (\rho v)_y \right) \right)_{(2)} = p'_{(0)} \left((\rho_{(0)} u_{(2)} + \rho_{(2)} u_{(0)})_x + (\rho_{(0)} v_{(2)} + \rho_{(2)} v_{(0)})_y \right).$$

Upon dividing the above equations by the term $\rho_{(0)}$, we get

$$\begin{aligned}
\mathbf{v}_{(0)}^{n+1} = \; & \mathbf{v}_{(0)}^{n} + \Delta t \left\{ \nabla \cdot \left(-\frac{p_{(2)}}{\rho_{(0)}} \mathbf{Id} \right)^{n+1} + \nabla \cdot \left(-\mathbf{v}_{(0)} \otimes \mathbf{v}_{(0)} \right)^n \right\} \\
& + \frac{\Delta t^2}{2} \left\{ \partial_x \left(\mathcal{J}_x^{(0)} \nabla \cdot \left(\mathbf{v}_{(0)} \otimes \mathbf{v}_{(0)} + \frac{p_{(2)}}{\rho_{(0)}} \mathbf{Id} \right) \right)^n + \partial_y \left(\mathcal{J}_y^{(0)} \nabla \cdot \left(\mathbf{v}_{(0)} \otimes \mathbf{v}_{(0)} + \frac{p_{(2)}}{\rho_{(0)}} \mathbf{Id} \right) \right)^n \right. \\
& \left. - \nabla \cdot \left(\frac{p_{(2)}}{\rho_{(0)}} \left(\rho_{(0)} \nabla \cdot \mathbf{v}_{(2)} + \mathbf{v}_{(0)} \cdot \nabla \rho_{(2)} \right) \mathbf{Id} \right)^{n+1} \right\}
\end{aligned} \tag{28}$$

where $\mathcal{J}_x^{(0)}$ and $\mathcal{J}_y^{(0)}$ are the Jacobian matrices given by

$$\mathcal{J}_x^{(0)} = \begin{pmatrix} 2u_{(0)} & 0 \\ v_{(0)} & u_{(0)} \end{pmatrix} \quad \text{and} \quad \mathcal{J}_y^{(0)} = \begin{pmatrix} v_{(0)} & u_{(0)} \\ 0 & 2v_{(0)} \end{pmatrix}. \tag{29}$$

Eq. (28) is nothing but the semi-discretization of the limiting equation (6) using the two-derivative IMEX-Taylor rule (15) when applied to the splitting

$$\begin{pmatrix} u_{(0)} \\ v_{(0)} \end{pmatrix}_t + \nabla \cdot \underbrace{\begin{pmatrix} \frac{p_{(2)}}{\rho_{(0)}} & 0 \\ 0 & \frac{p_{(2)}}{\rho_{(0)}} \end{pmatrix}}_{\mathbf{F}_{\mathrm{I}}(\mathbf{w}_{(0)})} + \nabla \cdot \underbrace{\begin{pmatrix} u_{(0)}^2 & u_{(0)} v_{(0)} \\ u_{(0)} v_{(0)} & v_{(0)}^2 \end{pmatrix}}_{\mathbf{F}_{\mathrm{E}}(\mathbf{w}_{(0)})} = 0,$$

which can be simplified to

$$(\mathbf{v}_{(0)})_t + \frac{\nabla p_{(2)}}{\rho_{(0)}} + \nabla \cdot \left(\mathbf{v}_{(0)} \otimes \mathbf{v}_{(0)} \right) = 0.$$

□

We are now ready to prove the following theorem:

Theorem 1. *The two-derivative IMEX-Taylor method combined with the DeTa splitting (16) is AP in the sense that the formal limit $\varepsilon \to 0$ of the numerical solution using (15), with well-prepared data and periodic boundary conditions, is a consistent discretization of the incompressible Euler equations (6).*

Proof. The proof of the theorem is a direct consequence of Lemmas 1, 2 and 3. Lemma 1 and Lemma 2 show the well-prepared property of the updated solution at t^{n+1}. Lemma 3 shows that the limiting method is the consistent semi-discretization of the limiting equation. □

2.2. Fully-discrete asymptotic analysis

Let the spatial domain $\Omega \subset \mathbb{R}^2$ be divided into $\mathcal{N}_E$ non-overlapping mesh elements

$$\Omega = \bigcup_{e=1}^{\mathcal{N}_E} \Omega_e.$$

Let $\mathcal{T}$ denotes the set of all mesh elements Ω_e and $\partial\mathcal{T}$ denotes the set of all element boundaries. In each Ω_e, we define a space of polynomials of degree $\mathcal{N}_p$, denoted by $\Pi_{\mathcal{N}_p}(\Omega_e)$. The space $\Pi_{\mathcal{N}_p}(\Omega_e)$ is constructed by the tensor product of the one-dimensional Lagrange interpolation polynomials of degree $\mathcal{N}_p$. Now, we define the space of broken polynomials $V_{\mathcal{N}_p}$ on the basis of the above spatial discretization by setting

$$V_{\mathcal{N}_p} := \left\{ \nu \in L^2(\Omega) \;\middle|\; \nu|_{\Omega_e} \in \Pi_{\mathcal{N}_p}(\Omega_e) \;\forall e \le \mathcal{N}_E \right\},$$

and let $V^k_{\mathcal{N}_p}$ denote the product space

$$V^k_{\mathcal{N}_p} := \underbrace{V_{\mathcal{N}_p} \times V_{\mathcal{N}_p} \times \cdots \times V_{\mathcal{N}_p}}_{k \text{ times}}.$$

As per definition, any function in $V^k_{\mathcal{N}_p}$ is allowed to have discontinuities over a cell boundary. Therefore, for any $\mathbf{x} \in \partial\Omega_e$ and the outward pointing normal vector n_e to Ω_e, we define

$$\xi^{\mathrm{R}} := \lim_{\delta \to 0} \xi(\mathbf{x} + |\delta| n_e),$$

as the exterior and

$$\xi^{\mathrm{L}} := \lim_{\delta \to 0} \xi(\mathbf{x} - |\delta| n_e),$$

as the interior values of $\xi(x)$ at $\mathbf{x}$. Finally, we define the following bilinear operators

$$(\mathbf{a}, \mathbf{b})_{\mathcal{T}} : (\mathbf{a}, \mathbf{b}) \in V^k_{\mathcal{N}_p} \times V^k_{\mathcal{N}_p} \to \sum_{e=1}^{\mathcal{N}_E} \int_{\Omega_e} \mathbf{a} \cdot \mathbf{b} \, dx$$

for the integral over whole domain Ω, and

$$\{\mathbf{a}, \mathbf{b}\}_{\partial\mathcal{T}} : (\mathbf{a}, \mathbf{b}) \in V^k_{\mathcal{N}_p} \times V^k_{\mathcal{N}_p} \to \sum_{e=1}^{\mathcal{N}_E} \int_{\partial\Omega_e} \left(\left(\mathbf{a}^{\mathrm{R}} + \mathbf{a}^{\mathrm{L}} \right) \cdot \mathbf{n}_e \right) \cdot \mathbf{b} \, ds,$$

$$[\![\mathbf{a}, \mathbf{b}]\!]_{\partial\mathcal{T}} : (\mathbf{a}, \mathbf{b}) \in V^k_{\mathcal{N}_p} \times V^k_{\mathcal{N}_p} \to \sum_{e=1}^{\mathcal{N}_E} \int_{\partial\Omega_e} \left(\mathbf{a}^{\mathrm{L}} - \mathbf{a}^{\mathrm{R}} \right) \cdot \mathbf{b} \, ds,$$

for the integral over domain boundaries $\partial\Omega$.

The DGSEM utilizes the weak formulation of Eq. (3). Therefore, at each time step t^n, we search for an approximate solution $\mathbf{w} = (\rho^n, \rho\mathbf{v}^n)$ to the two-dimensional isentropic Euler equations (3) in $V^3_{N_p}$. The approximation to the first derivative $\mathbf{w}_t$ is given by

$$(\mathbf{w}_t, \phi)_{\mathcal{T}} - (\mathbf{F}(\mathbf{w}), \nabla\phi)_{\mathcal{T}} + \frac{1}{2}\{\mathbf{F}^*(\mathbf{w}), \phi\}_{\partial\mathcal{T}} + \boldsymbol{\lambda}[\![\mathbf{w}, \phi]\!]_{\partial\mathcal{T}} = 0, \ \forall\phi \in V^3_{N_p}; \tag{30}$$

the second derivative $\mathbf{w}_{tt}$ is given by

$$(\mathbf{w}_{tt}, \phi)_{\mathcal{T}} - \left(\frac{\partial\mathbf{F}(\mathbf{w})}{\partial\mathbf{w}}\boldsymbol{\sigma}, \nabla\phi\right)_{\mathcal{T}} + \frac{1}{2}\left\{\frac{\partial\mathbf{F}^*(\mathbf{w})}{\partial\mathbf{w}}\boldsymbol{\sigma}, \phi\right\}_{\partial\mathcal{T}} + \boldsymbol{\lambda}[\![\boldsymbol{\sigma}, \phi]\!]_{\partial\mathcal{T}} = 0, \ \forall\phi \in V^3_{N_p}, \tag{31}$$

with the artificial quantity

$$\boldsymbol{\sigma} := \mathbf{R}^{(1)}(\mathbf{w}) \equiv \mathbf{w}_t \tag{32}$$

as introduced in [33]. $\mathbf{F}^*(\mathbf{w})$ is the global Lax-Friedrichs numerical flux that depends on the left ($\mathbf{w}^{\mathrm{L}}$) and right ($\mathbf{w}^{\mathrm{R}}$) states with respect to the cell-edge, and $\boldsymbol{\lambda}$ is a globally constant vector. For isentropic Euler equations, the $\boldsymbol{\lambda}$ values for the explicit and implicit numerical fluxes (corresponding to $\mathbf{F}_{\mathrm{E}}$ and $\mathbf{F}_{\mathrm{I}}$ from Eq. (16)) are chosen to be

$$\boldsymbol{\lambda}_{\mathrm{E}} = (\varepsilon, \varepsilon, \varepsilon), \quad \boldsymbol{\lambda}_{\mathrm{I}} = \left(\frac{1}{\varepsilon^2}, 1, 1\right),$$

respectively. If the full flux $\mathbf{F}$ is used for reference, we choose

$$\boldsymbol{\lambda} = \left(\frac{1}{\varepsilon^2}, 1, 1\right).$$

The particular $\boldsymbol{\lambda}$ values given above are chosen to ensure the asymptotic preserving property of the schemes [34].

For the two-dimensional isentropic Euler equations (16), the fully discrete two-derivative IMEX-Taylor method (Alg. 1) is given by

$$\begin{aligned}
0 = &\left(\mathbf{w}^{n+1} - \mathbf{w}^n, \phi\right)_{\mathcal{T}} + \Delta t \left\{-\left(\mathbf{F}_{\mathrm{I}}(\mathbf{w}^{n+1}), \nabla\phi\right)_{\mathcal{T}} + \frac{1}{2}\left\{\mathbf{F}_{\mathrm{I}}(\mathbf{w}^{n+1}), \phi\right\}_{\partial\mathcal{T}} + \frac{1}{2}\,\mathrm{Diag}\left[\frac{1}{\varepsilon^2}, 1, 1\right][\![\mathbf{w}^{n+1}, \phi]\!]_{\partial\mathcal{T}}\right\} \\
&+\Delta t \left\{-(\mathbf{F}_{\mathrm{E}}(\mathbf{w}^n), \nabla\phi)_{\mathcal{T}} + \frac{1}{2}\{\mathbf{F}_{\mathrm{E}}(\mathbf{w}^n), \phi\}_{\partial\mathcal{T}} + \varepsilon[\![\mathbf{w}^n, \phi]\!]_{\partial\mathcal{T}}\right\} \\
&-\frac{\Delta t^2}{2}\left\{-\left(\frac{\partial\mathbf{F}_{\mathrm{I}}(\mathbf{w}^{n+1})}{\partial\mathbf{w}}\boldsymbol{\sigma}^{n+1}, \nabla\phi\right)_{\mathcal{T}} + \frac{1}{2}\left\{\frac{\partial\mathbf{F}_{\mathrm{I}}(\mathbf{w}^{n+1})}{\partial\mathbf{w}}\boldsymbol{\sigma}^{n+1}, \phi\right\}_{\partial\mathcal{T}} + \frac{1}{2}\,\mathrm{Diag}\left[\frac{1}{\varepsilon^2}, 1, 1\right][\![\boldsymbol{\sigma}^{n+1}, \phi]\!]_{\partial\mathcal{T}}\right\} \\
&+\frac{\Delta t^2}{2}\left\{-\left(\frac{\partial\mathbf{F}_{\mathrm{E}}(\mathbf{w}^n)}{\partial\mathbf{w}}\mathbf{R}^{(1)}(\mathbf{w}^n), \nabla\phi\right)_{\mathcal{T}} + \frac{1}{2}\left\{\frac{\partial\mathbf{F}_{\mathrm{E}}(\mathbf{w}^n)}{\partial\mathbf{w}}\mathbf{R}^{(1)}(\mathbf{w}^n), \phi\right\}_{\partial\mathcal{T}} + \varepsilon[\![\mathbf{R}^{(1)}(\mathbf{w}^n), \phi]\!]_{\partial\mathcal{T}}\right\}.
\end{aligned} \tag{33}$$

We solve the above expression (33) for $\mathbf{w}^{n+1}$ along with a discretization of (32), given by

$$\begin{aligned}
0 = &\left(\boldsymbol{\sigma}^{n+1}, \phi\right)_{\mathcal{T}} - \left(\mathbf{F}_{\mathrm{I}}(\mathbf{w}^{n+1}), \nabla\phi\right)_{\mathcal{T}} + \frac{1}{2}\left\{\mathbf{F}_{\mathrm{I}}(\mathbf{w}^{n+1}), \phi\right\}_{\partial\mathcal{T}} + \frac{1}{2}\,\mathrm{Diag}\left[\frac{1}{\varepsilon^2}, 1, 1\right][\![\mathbf{w}^{n+1}, \phi]\!]_{\partial\mathcal{T}} \\
&- (\mathbf{F}_{\mathrm{E}}(\mathbf{w}^n), \nabla\phi)_{\mathcal{T}} + \frac{1}{2}\{\mathbf{F}_{\mathrm{E}}(\mathbf{w}^n), \phi\}_{\partial\mathcal{T}} + \varepsilon[\![\mathbf{w}^n, \phi]\!]_{\partial\mathcal{T}}.
\end{aligned} \tag{34}$$

Consider the asymptotic expansion of the numerical solution

$$\mathbf{w}^{n+1} = \mathbf{w}^{n+1}_{(0)} + \varepsilon\mathbf{w}^{n+1}_{(1)} + \varepsilon^2\mathbf{w}^{n+1}_{(2)} + O(\varepsilon^3) \quad \text{and} \quad \boldsymbol{\sigma}^{n+1} = \boldsymbol{\sigma}^{n+1}_{(0)} + \varepsilon\boldsymbol{\sigma}^{n+1}_{(1)} + \varepsilon^2\boldsymbol{\sigma}^{n+1}_{(2)} + O(\varepsilon^3),$$

with $\boldsymbol{\sigma} = \left(\boldsymbol{\sigma}_\rho, \boldsymbol{\sigma}_{\rho u}, \boldsymbol{\sigma}_{\rho v}\right)^T =: \left(\boldsymbol{\sigma}_\rho, \boldsymbol{\sigma}_{\rho \mathbf{v}}\right)^T$. Expanding the components of the $\boldsymbol{\sigma}$ equations, we get

$$0 = \left(\boldsymbol{\sigma}_\rho^{n+1}, \phi\right)_{\mathcal{T}} - \left(\rho^{n+1}\mathbf{v}^{n+1}, \nabla\phi\right)_{\mathcal{T}} + \frac{1}{2}\left\{\rho^{n+1}\mathbf{v}^{n+1}, \phi\right\}_{\partial\mathcal{T}} + \frac{1}{2\varepsilon^2}\left[\!\left[\rho^{n+1}, \phi\right]\!\right]_{\partial\mathcal{T}} + \varepsilon\left[\!\left[\rho^n, \phi\right]\!\right]_{\partial\mathcal{T}}, \tag{35}$$

$$\begin{aligned} 0 = \left(\boldsymbol{\sigma}_{\rho\mathbf{v}}^{n+1}, \phi\right)_{\mathcal{T}} &- \left(\frac{1-\varepsilon^2}{\varepsilon^2}p^{n+1}\mathbf{Id}, \nabla\phi\right)_{\mathcal{T}} + \frac{1}{2}\left\{\frac{1-\varepsilon^2}{\varepsilon^2}p^{n+1}\mathbf{Id}, \phi\right\}_{\partial\mathcal{T}} + \left[\!\left[\rho^{n+1}\mathbf{v}^{n+1}, \phi\right]\!\right]_{\partial\mathcal{T}} \\ &- \left((\rho\mathbf{v}\otimes\mathbf{v} + p\mathbf{Id})^n, \nabla\phi\right)_{\mathcal{T}} + \frac{1}{2}\left\{(\rho\mathbf{v}\otimes\mathbf{v} + p\mathbf{Id})^n, \phi\right\}_{\partial\mathcal{T}} + \varepsilon\left[\!\left[\rho^n\mathbf{v}^n, \phi\right]\!\right]_{\partial\mathcal{T}}. \end{aligned} \tag{36}$$

Lemma 4. *The quantities $\rho_{(0)}^{n+1}$ and $\rho_{(1)}^{n+1}$ in the asymptotic expansion of ρ^{n+1} are continuous across the entire spatial domain.*

Proof. The $O(\varepsilon^{-2})$ and $O(\varepsilon^{-1})$ terms from Eq. (35) give

$$\left[\!\left[\rho_{(0)}^{n+1}, \phi\right]\!\right]_{\partial\mathcal{T}} = 0, \quad \text{and} \quad \left[\!\left[\rho_{(1)}^{n+1}, \phi\right]\!\right]_{\partial\mathcal{T}} = 0. \tag{37}$$

[35, Lemma 2] applied to Eq. (37) implies the continuity of $\rho_{(0)}^{n+1}$ and $\rho_{(1)}^{n+1}$ over the entire spatial domain. □

Lemma 5. *The quantities $\rho_{(0)}^{n+1}$ and $\rho_{(1)}^{n+1}$ in the asymptotic expansion of ρ^{n+1} are constant in space.*

Proof. The $O(\varepsilon^{-2})$ and $O(\varepsilon^{-1})$ terms from Eq. (36) give

$$\begin{aligned} -\left(p_{(0)}^{n+1}\mathbf{Id}, \nabla\phi\right)_{\mathcal{T}} + \frac{1}{2}\left\{p_{(0)}^{n+1}\mathbf{Id}, \phi\right\}_{\partial\mathcal{T}} = 0, \\ -\left(p_{(1)}^{n+1}\mathbf{Id}, \nabla\phi\right)_{\mathcal{T}} + \frac{1}{2}\left\{p_{(1)}^{n+1}\mathbf{Id}, \phi\right\}_{\partial\mathcal{T}} = 0. \end{aligned}$$

Applying integration by parts to the above equations with the continuity results from Lemma 4 gives

$$\begin{aligned} \left(\nabla\cdot p_{(0)}^{n+1}\mathbf{Id}, \phi\right)_{\mathcal{T}} + \frac{1}{2}\left[\!\left[p_{(0)}^{n+1}\mathbf{Id}\cdot n_e, \phi\right]\!\right]_{\partial\mathcal{T}} = 0 \quad &\Rightarrow \left(\nabla p_{(0)}^{n+1}, \phi\right)_{\mathcal{T}} = 0, \\ \left(\nabla\cdot p_{(1)}^{n+1}\mathbf{Id}, \phi\right)_{\mathcal{T}} + \frac{1}{2}\left[\!\left[p_{(1)}^{n+1}\mathbf{Id}\cdot n_e, \phi\right]\!\right]_{\partial\mathcal{T}} = 0 \quad &\Rightarrow \left(\nabla p_{(1)}^{n+1}, \phi\right)_{\mathcal{T}} = 0. \end{aligned}$$

By choosing $\phi = \nabla p_{(m)}^{n+1}$ and substituting it into the equations above, we obtain

$$\left(\nabla p_{(m)}^{n+1}, \nabla p_{(m)}^{n+1}\right)_{\mathcal{T}} = 0 \quad \Rightarrow \nabla p_{(m)}^{n+1} = 0, \ \forall m \in \{0, 1\}.$$

Hence, the quantities $\rho_{(0)}^{n+1}$ and $\rho_{(1)}^{n+1}$ are constant in space. □

Lemma 6. *The quantities $\rho_{(0)}^{n+1}$ and $\rho_{(1)}^{n+1}$ in the asymptotic expansion of ρ^{n+1} are constant in time, given periodic boundary conditions are considered.*

Proof. Consider the density equation from Eq. (33)

$$\begin{aligned} 0 = \left(\rho^{n+1} - \rho^n, \phi\right)_{\mathcal{T}} + \Delta t\left\{-\left(\rho^{n+1}\mathbf{v}^{n+1}, \nabla\phi\right)_{\mathcal{T}} + \frac{1}{2}\left\{\rho^{n+1}\mathbf{v}^{n+1}, \phi\right\}_{\partial\mathcal{T}} + \frac{1}{2\varepsilon^2}\left[\!\left[\rho^{n+1}, \phi\right]\!\right]_{\partial\mathcal{T}}\right\} \\ -\frac{\Delta t^2}{2}\left\{-\left(\boldsymbol{\sigma}_{\rho\mathbf{v}}^{n+1}, \nabla\phi\right)_{\mathcal{T}} + \frac{1}{2}\left\{\boldsymbol{\sigma}_{\rho\mathbf{v}}^{n+1}, \phi\right\}_{\partial\mathcal{T}} + \frac{1}{2\varepsilon^2}\left[\!\left[\boldsymbol{\sigma}_\rho^{n+1}, \phi\right]\!\right]_{\partial\mathcal{T}}\right\}. \end{aligned} \tag{38}$$

The $O(1)$ terms from density equation (38) are

$$\begin{aligned} 0 = \left(\rho_{(0)}^{n+1} - \rho_{(0)}^n, \phi\right)_{\mathcal{T}} + \Delta t\left\{-\left(\rho_{(0)}^{n+1}\mathbf{v}_{(0)}^{n+1}, \nabla\phi\right)_{\mathcal{T}} + \frac{1}{2}\left\{\rho_{(0)}^{n+1}\mathbf{v}_{(0)}^{n+1}, \phi\right\}_{\partial\mathcal{T}} + \frac{1}{2}\left[\!\left[\rho_{(2)}^{n+1}, \phi\right]\!\right]_{\partial\mathcal{T}}\right\} \\ -\frac{\Delta t^2}{2}\left\{-\left(\boldsymbol{\sigma}_{\rho\mathbf{v}(0)}^{n+1}, \nabla\phi\right)_{\mathcal{T}} + \frac{1}{2}\left\{\boldsymbol{\sigma}_{\rho\mathbf{v}(0)}^{n+1}, \phi\right\}_{\partial\mathcal{T}} + \frac{1}{2}\left[\!\left[\boldsymbol{\sigma}_{\rho(2)}^{n+1}, \phi\right]\!\right]_{\partial\mathcal{T}}\right\}. \end{aligned}$$

Substituting $\phi \equiv 1$ in the above equation, we get

$$0 = \left(\rho_{(0)}^{n+1} - \rho_{(0)}^{n}, 1\right)_{\mathcal{T}} + \Delta t \left\{ \frac{1}{2} \left\{ \rho_{(0)}^{n+1} \mathbf{v}_{(0)}^{n+1}, 1 \right\}_{\partial\mathcal{T}} + \frac{1}{2} \left[\!\left[\rho_{(2)}^{n+1}, 1 \right]\!\right]_{\partial\mathcal{T}} \right\} - \frac{\Delta t^2}{2} \left\{ \frac{1}{2} \left\{ \boldsymbol{\sigma}_{\rho\mathbf{v}(0)}^{n+1}, 1 \right\}_{\partial\mathcal{T}} + \frac{1}{2} \left[\!\left[\boldsymbol{\sigma}_{\rho(2)}^{n+1}, 1 \right]\!\right]_{\partial\mathcal{T}} \right\}.$$

As we assume periodic boundary conditions, the boundary integrals will add up to zero. Combining the result from Lemma 5, we obtain

$$\left(\rho_{(0)}^{n+1} - \rho_{(0)}^{n}\right)(1,1)_{\mathcal{T}} = 0 \quad \Rightarrow \rho_{(0)}^{n+1} - \rho_{(0)}^{n} = 0.$$

Similarly, it can be shown that $\rho_{(1)}^{n+1} - \rho_{(1)}^{n} = 0$, using $O(\varepsilon)$ terms from density equation (38). Therefore, the quantities $\rho_{(0)}^{n+1}$ and $\rho_{(1)}^{n+1}$ are constant in space and time. □

Lemma 7. *The $O(1)$ terms from the density equation (38) form a consistent discretization of $\nabla \cdot \mathbf{v}_{(0)} = 0$, assuming well-prepared solution at t^n and periodic boundary conditions.*

Proof. Consider the $O(1)$ terms from the density equation (38) with the results from Lemmas 5 and 6

$$\begin{aligned} 0 = \Delta t \Bigg\{ &- \left(\mathbf{v}_{(0)}^{n+1}, \nabla\phi\right)_{\mathcal{T}} + \frac{1}{2} \left\{ \mathbf{v}_{(0)}^{n+1}, \phi \right\}_{\partial\mathcal{T}} + \frac{1}{2} \left[\!\left[\frac{\rho_{(2)}^{n+1}}{\rho_{(0)}}, \phi \right]\!\right]_{\partial\mathcal{T}} \Bigg\} \\ &- \frac{\Delta t^2}{2} \left\{ - \left(\frac{\boldsymbol{\sigma}_{\rho\mathbf{v}(0)}^{n+1}}{\rho_{(0)}}, \nabla\phi \right)_{\mathcal{T}} + \frac{1}{2} \left\{ \frac{\boldsymbol{\sigma}_{\rho\mathbf{v}(0)}^{n+1}}{\rho_{(0)}}, \phi \right\}_{\partial\mathcal{T}} + \frac{1}{2} \left[\!\left[\frac{\boldsymbol{\sigma}_{\rho(2)}^{n+1}}{\rho_{(0)}}, \phi \right]\!\right]_{\partial\mathcal{T}} \right\}. \end{aligned}$$

The above equation is nothing but the implicit two-derivative DGSEM discretization of $\nabla \cdot \mathbf{v}_{(0)}$ along with the $O(1)$ terms from Eqs. (35) and (36). □

Reconsider the momentum equations from Eq. (33),

$$\begin{aligned} 0 = &\left(\rho^{n+1}\mathbf{v}^{n+1} - \rho^{n}\mathbf{v}^{n}, \phi\right)_{\mathcal{T}} + \Delta t \left\{ - \left(\frac{1-\varepsilon^2}{\varepsilon^2} p^{n+1} \mathbf{Id}, \nabla\phi \right)_{\mathcal{T}} + \frac{1}{2} \left\{ \frac{1-\varepsilon^2}{\varepsilon^2} p^{n+1} \mathbf{Id}, \phi \right\}_{\partial\mathcal{T}} + \frac{1}{2} \left[\!\left[\rho^{n+1}\mathbf{v}^{n+1}, \phi \right]\!\right]_{\partial\mathcal{T}} \right\} \\ &+ \Delta t \left\{ - \left((\rho\mathbf{v}\otimes\mathbf{v} + p\mathbf{Id})^{n}, \nabla\phi \right)_{\mathcal{T}} + \frac{1}{2} \left\{ (\rho\mathbf{v}\otimes\mathbf{v} + p\mathbf{Id})^{n}, \phi \right\}_{\partial\mathcal{T}} + \varepsilon \left[\!\left[\rho^{n}\mathbf{v}^{n}, \phi \right]\!\right]_{\partial\mathcal{T}} \right\} \\ &- \frac{\Delta t^2}{2} \left\{ - \left(\frac{1-\varepsilon^2}{\varepsilon^2} p'^{n+1} \boldsymbol{\sigma}_{\rho}^{n+1} \mathbf{Id}, \nabla\phi \right)_{\mathcal{T}} + \frac{1}{2} \left\{ \frac{1-\varepsilon^2}{\varepsilon^2} p'^{n+1} \boldsymbol{\sigma}_{\rho}^{n+1} \mathbf{Id}, \phi \right\}_{\partial\mathcal{T}} + \frac{1}{2} \left[\!\left[\boldsymbol{\sigma}_{\rho\mathbf{v}}^{n+1}, \phi \right]\!\right]_{\partial\mathcal{T}} \right\} \\ &+ \frac{\Delta t^2}{2} \left\{ - \left(\left(p'\mathbf{Id} - \mathbf{v}\otimes\mathbf{v}\right)^{n} \mathbf{R}_{\rho}^{(1)}(\mathbf{w}^{n}) + \mathcal{J}^{n}\, \mathbf{R}_{\rho\mathbf{v}}^{(1)}(\mathbf{w}^{n}), \nabla\phi \right)_{\mathcal{T}} \right. \\ &\left. + \frac{1}{2} \left\{ \left(p'\mathbf{Id} - \mathbf{v}\otimes\mathbf{v}\right)^{n} \mathbf{R}_{\rho}^{(1)}(\mathbf{w}^{n}) + \mathcal{J}^{n}\, \mathbf{R}_{\rho\mathbf{v}}^{(1)}(\mathbf{w}^{n}), \nabla\phi \right\}_{\partial\mathcal{T}} + \varepsilon \left[\!\left[\boldsymbol{\sigma}_{\rho\mathbf{v}}^{n}, \phi \right]\!\right]_{\partial\mathcal{T}} \right\}, \end{aligned} \tag{39}$$

where $\mathbf{R}^{(1)} = \left(\mathbf{R}_{\rho}^{(1)}, \mathbf{R}_{\rho u}^{(1)}, \mathbf{R}_{\rho v}^{(1)}\right)^{T} =: \left(\mathbf{R}_{\rho}^{(1)}, \mathbf{R}_{\rho\mathbf{v}}^{(1)}\right)^{T}$, and $\mathcal{J} = \left[\mathcal{J}_x \,\middle|\, \mathcal{J}_y\right]$ is given by (compare with (29))

$$\mathcal{J}_x = \begin{pmatrix} 2u & 0 \\ v & u \end{pmatrix} \quad \text{and} \quad \mathcal{J}_y = \begin{pmatrix} v & u \\ 0 & 2v \end{pmatrix}.$$

Lemma 8. *The $O(1)$ terms from the momentum equations in Eq. (39) are the two-derivative IMEX-Taylor DGSEM discretization of the limiting equation*

$$\left(\mathbf{v}_{(0)}\right)_t + \frac{\nabla p_{(2)}}{\rho_{(0)}} + \nabla \cdot \left(\mathbf{v}_{(0)} \otimes \mathbf{v}_{(0)}\right) = 0, \tag{40}$$

under the assumptions of well-prepared solution at t^n and Lemmas 5 and 6, with an implicit treatment of pressure term $p_{(2)}$ and an explicit treatment of the tensor product $(\mathbf{v}_{(0)} \otimes \mathbf{v}_{(0)})$.

Proof. Consider the $O(1)$ terms from the momentum equations in Eq. (39) along with results from Lemmas 5 and 6, and applying integration by parts to the explicit pressure term gives

$$\begin{aligned}
0 = \left(\mathbf{v}_{(0)}^{n+1} - \mathbf{v}_{(0)}^{n}, \phi\right)_{\mathcal{T}} &+ \Delta t \left\{ -\left(\frac{p_{(2)}^{n+1}}{\rho_{(0)}}\mathbf{Id}, \nabla\phi\right)_{\mathcal{T}} + \frac{1}{2}\left\{\frac{p_{(2)}^{n+1}}{\rho_{(0)}}\mathbf{Id}, \phi\right\}_{\partial\mathcal{T}} + \frac{1}{2}\left[\!\left[\mathbf{v}_{(0)}^{n+1}, \phi\right]\!\right]_{\partial\mathcal{T}} \right\} \\
&+ \Delta t \left\{ -\left(\mathbf{v}_{(0)}^{n} \otimes \mathbf{v}_{(0)}^{n}, \nabla\phi\right)_{\mathcal{T}} + \frac{1}{2}\left\{\mathbf{v}_{(0)}^{n} \otimes \mathbf{v}_{(0)}^{n}, \phi\right\}_{\partial\mathcal{T}} \right\} \\
&- \frac{\Delta t^2}{2} \left\{ -\left(\frac{{p'_{(2)}}^{n+1}}{\rho_{(0)}}\boldsymbol{\sigma}_{\rho(0)}^{n+1}\mathbf{Id}, \nabla\phi\right)_{\mathcal{T}} + \frac{1}{2}\left\{\frac{{p'_{(2)}}^{n+1}}{\rho_{(0)}}\boldsymbol{\sigma}_{\rho(0)}^{n+1}\mathbf{Id}, \phi\right\}_{\partial\mathcal{T}} + \frac{1}{2}\left[\!\left[\frac{\boldsymbol{\sigma}_{\rho\mathbf{v}(0)}^{n+1}}{\rho_{(0)}}, \phi\right]\!\right]_{\partial\mathcal{T}} \right\} \\
&+ \frac{\Delta t^2}{2} \left\{ -\left(\frac{\mathcal{J}_{(0)}^{n}}{\rho_{(0)}}\mathbf{R}_{\rho\mathbf{v}(0)}^{(1)}(\mathbf{w}^n), \nabla\phi\right)_{\mathcal{T}} + \frac{1}{2}\left\{\frac{\mathcal{J}_{(0)}^{n}}{\rho_{(0)}}\mathbf{R}_{\rho\mathbf{v}(0)}^{(1)}(\mathbf{w}^n), \nabla\phi\right\}_{\partial\mathcal{T}} \right\},
\end{aligned}$$

where $\boldsymbol{\sigma}_{\rho(0)}$ and $\boldsymbol{\sigma}_{\rho\mathbf{v}(0)}$ are the respective $O(1)$ terms from Eqs. (35) and (36). The above equation is nothing but the two-derivative IMEX-Taylor DGSEM discretization of the limiting equation (6) when applied to the following splitting

$$\begin{pmatrix} u_{(0)} \\ v_{(0)} \end{pmatrix}_t + \nabla \cdot \underbrace{\begin{pmatrix} \frac{p_{(2)}}{\rho_{(0)}} & 0 \\ 0 & \frac{p_{(2)}}{\rho_{(0)}} \end{pmatrix}}_{\mathbf{F}_{\mathrm{I}}(\mathbf{w}_{(0)})} + \nabla \cdot \underbrace{\begin{pmatrix} u_{(0)}^2 & u_{(0)}v_{(0)} \\ u_{(0)}v_{(0)} & v_{(0)}^2 \end{pmatrix}}_{\mathbf{F}_{\mathrm{E}}(\mathbf{w}_{(0)})} = 0.$$

□

Theorem 2. *The two-derivative IMEX-Taylor method combined with the DeTa splitting (16) and spatially discretized with DGSEM is asymptotically consistent, in the sense that the formal limit $\varepsilon \to 0$ of the numerical solution with well-prepared data and periodic boundary conditions, is a consistent discretization of the incompressible Euler equations (6).*

Proof. The proof of the theorem follows from Lemmas 4, 5 and 6, which show the well preparedness of the updated solution. Lemmas 7 and 8 shows that the limiting method is a consistent discretization of the limiting equations. □

3. Extension to higher order methods

The two-derivative IMEX-Taylor method (15) discussed previously gives a second-order accurate solution. When higher-order implicit or explicit Runge-Kutta methods are concerned, devising them involves a cumbersome process of solving many equations. The number of unknowns and equations exponentially increases as we search for Runge-Kutta schemes with higher orders of accuracy. Due to additional conditions, the scenario worsens when probing for higher-order IMEX schemes. The use of predictor-corrector schemes [16, 19, 20, 29] discussed in the introduction reduces the algebraic burden of devising high order schemes. The schemes are constructed in such a way that the order is increased in each correction step until it hits its maximum achievable order.

Extending implicit predictor-corrector schemes to IMEX schemes is trouble-free. An IMEX predictor step instead of an implicit predictor can change the overall scheme to an IMEX method with slight modifications on the correction steps. Hence, we have the two-derivative IMEX-HBPC scheme given in Alg. 2, which uses the two-derivative IMEX-Taylor scheme as the prediction step. We consider the serial IMEX version of the HBPC algorithm described in [16, 19] incorporating the stability parameters (θ_1, θ_2) introduced in [20].

Algorithm 2 (IMEX-HBPC($q, k_{\max}$))**.** *To advance the solution in time, we compute values $\mathbf{w}^{n,[k],l}$. To account for the initial conditions $\mathbf{w}_0 \equiv \mathbf{w}(t=0)$, define*

$$\mathbf{w}^{-1,[k],s} := \mathbf{w}_0.$$

1. ***Predict.*** *Solve the following expression for $\mathbf{w}^{n,[0],l}$ and each $1 \le l \le s$:*

$$\mathbf{w}^{n,[0],l} := \mathbf{w}^{n} + c_l \Delta t \left(\mathbf{R}_I^{(1)}(\mathbf{w}^{n,[0],l}) + \mathbf{R}_E^{(1)}(\mathbf{w}^{n})\right) + \frac{(c_l \Delta t)^2}{2}\left(\mathbf{R}_E^{(2)}(\mathbf{w}^{n}) - \mathbf{R}_I^{(2)}(\mathbf{w}^{n}, \mathbf{w}^{n,[0],l})\right). \tag{41}$$

2. ***Correct.*** *Next, the corrected values* $\mathbf{w}^{n,[k],l}$ *for* $1 \le k \le k_{\max}$ *are computed through solving for each* $1 \le l \le s$ *and each* $1 \le k \le k_{\max}$*:*

$$\mathbf{w}^{n,[k],l} := \mathbf{w}^n + \Delta t \theta_1 \left(\mathbf{R}_I^{(1)}(\mathbf{w}^{n,[k],l}) - \mathbf{R}_I^{(1)}(\mathbf{w}^{n,[k-1],l}) \right) - \frac{\Delta t^2}{2} \theta_2 \left(\mathbf{R}_I^{(2)}(\mathbf{w}^{n,[k-1],l}, \mathbf{w}^{n,[k],l}) - \mathbf{R}_I^{(2)}(\mathbf{w}^{n,[k-1],l}) \right) + \mathcal{I}_l, \tag{42}$$

with

$$\mathcal{I}_l := \mathcal{I}_l \left(\mathbf{w}^{n,[k-1],1}, \dots, \mathbf{w}^{n,[k-1],s} \right). \tag{43}$$

$\mathcal{I}_l(\cdot)$ *denotes the q-th order Hermite-Birkhoff quadrature rule.*

3. ***Update.*** *In order to retain a first-same-as-last property, we update the solution with*

$$\mathbf{w}^{n+1} := \mathbf{w}^{n,[k_{\max}],s}. \tag{44}$$

Remark 2. *The order of accuracy of the IMEX-HBPC*$(q, k_{\max})$ *scheme outlined in Alg. 2 is* $\min(2 + k_{\max}, q)$*. The scheme begins with a second-order predicted solution, and the order of accuracy is improved by one in each correction step until it reaches the maximum accuracy order q determined by the underlying quadrature rule* (43)*. For a proof, refer to [16, 19, 29].*

Theorem 3. *The two-derivative IMEX-HBPC scheme combined with the DeTa splitting* (16) *and spatially discretized with DGSEM is asymptotically consistent in the sense that the formal limit* $\varepsilon \to 0$ *of the numerical solution using Alg.* (2) *with well-prepared data and periodic boundary conditions, is a consistent discretization of the incompressible Euler equations* (6)*.*

Proof. The proof of the asymptotic consistency for the predictor step of the IMEX-HBPC scheme follows the same steps as of Theorem 2. Similar arguments can be extended to the corrector steps to complete the proof. □

4. Solving the (non-)linear system of equations

We use Newton's method to solve the non-linear equations arising in the implicit formulations of the IMEX-HBPC scheme. For a given system of non-linear equations $\mathcal{G}(X) := 0$, Newton's method starts with an initial guess X^0. Then, the following linear equation is solved for the Newton increment ΔX:

$$\frac{\partial \mathcal{G}(X^r)}{\partial X} \cdot \Delta X = -\mathcal{G}(X^r),$$

ΔX is hence used to update the new iterate through

$$X^{r+1} = X^r + \Delta X.$$

The iterative procedure is continued until a desired solution is found that satisfies the imposed stopping criterion. The formulations of the non-linear equations associated with each implicit step of Alg. 1 and Alg. 2 are given by

$$\mathcal{G}(X) \equiv \begin{pmatrix} \mathcal{G}^w(X) \\ \mathcal{G}^\sigma(X) \end{pmatrix} := \begin{pmatrix} \mathbf{w} - \alpha_1 \Delta t \mathbf{R}_{\mathrm{I}}^{(1)}(\mathbf{w}) + \frac{\alpha_2 \Delta t^2}{2} \mathbf{R}_{\mathrm{I}}^{(2)}(\mathbf{w}, \boldsymbol{\sigma}) \\ \boldsymbol{\sigma} - \mathbf{R}_{\mathrm{I}}^{(1)}(\mathbf{w}) \end{pmatrix} - \begin{pmatrix} b(\bar{\mathbf{w}}) \\ \mathbf{R}_{\mathrm{E}}^{(1)}(\bar{\mathbf{w}}) \end{pmatrix} = 0,$$

where $b(\bar{\mathbf{w}})$ and the parameters α_1 and α_2 depend on the respective schemes. The unknown value is denoted as $\mathbf{w}$, whereas $\bar{\mathbf{w}}$ is the known solution from a previous step. The associated Jacobian for the above non-linear equation system is

$$\mathcal{J}(X) := \begin{pmatrix} \mathbf{Id} - \alpha_1 \Delta t \frac{\partial \mathbf{R}_{\mathrm{I}}^{(1)}(\mathbf{w})}{\partial \mathbf{w}} + \frac{\alpha_2 \Delta t^2}{2} \frac{\partial \mathbf{R}_{\mathrm{I}}^{(2)}(\mathbf{w}, \boldsymbol{\sigma})}{\partial \mathbf{w}} & \frac{\alpha_2 \Delta t^2}{2} \frac{\partial \mathbf{R}_{\mathrm{I}}^{(1)}(\mathbf{w})}{\partial \mathbf{w}} \\ -\frac{\partial \mathbf{R}_{\mathrm{I}}^{(1)}(\mathbf{w})}{\partial \mathbf{w}} & \mathbf{Id} \end{pmatrix}. \tag{45}$$

There holds $\frac{\partial \mathbf{R}_{\mathrm{I}}^{(2)}(\mathbf{w}, \boldsymbol{\sigma})}{\partial \boldsymbol{\sigma}} = \frac{\partial \mathbf{R}_{\mathrm{I}}^{(1)}(\mathbf{w})}{\partial \mathbf{w}}$ due to the structure of the spatial discretization. Hence, the terms $\frac{\partial \mathbf{R}_{\mathrm{I}}^{(2)}(\mathbf{w}, \boldsymbol{\sigma})}{\partial \boldsymbol{\sigma}}$ are replaced by $\frac{\partial \mathbf{R}_{\mathrm{I}}^{(1)}(\mathbf{w})}{\partial \mathbf{w}}$ in the Jacobian defined above.

The linear equations arising in the Newton iterations are solved using the GMRES method. As the GMRES method utilizes the matrix-vector products $\mathcal{J}\Delta X$ for the solution search, we use a matrix-free approach, as done in [36, Sec. 4]. Consult the papers [37] and [38] for more details on the matrix-free implementation. Applying the matrix-free approach, then the Jacobian-vector product is

$$\mathcal{J}\Delta X = \begin{pmatrix} \Delta\mathbf{w} - \alpha_1\Delta t \frac{\mathbf{R}_1^{(1)}(\mathbf{w}+\varepsilon_{\mathrm{FD}}^{w}\Delta\mathbf{w})-\mathbf{R}_1^{(1)}(\mathbf{w})}{\varepsilon_{\mathrm{FD}}^{w}} + \frac{\alpha_2\Delta t^2}{2}\frac{\mathbf{R}_1^{(2)}(\mathbf{w}+\varepsilon_{\mathrm{FD}}^{w}\Delta\mathbf{w},\boldsymbol{\sigma})-\mathbf{R}_1^{(2)}(\mathbf{w},\boldsymbol{\sigma})}{\varepsilon_{\mathrm{FD}}^{w}} + \frac{\alpha_2\Delta t^2}{2}\frac{\mathbf{R}_1^{(1)}(\mathbf{w}+\varepsilon_{\mathrm{FD}}^{\sigma}\Delta\boldsymbol{\sigma})-\mathbf{R}_1^{(1)}(\mathbf{w})}{\varepsilon_{\mathrm{FD}}^{\sigma}} \\ -\frac{\mathbf{R}_1^{(1)}(\mathbf{w}+\varepsilon_{\mathrm{FD}}^{w}\Delta\mathbf{w})-\mathbf{R}_1^{(1)}(\mathbf{w})}{\varepsilon_{\mathrm{FD}}^{w}} + \Delta\boldsymbol{\sigma} \end{pmatrix},$$

replacing the derivatives with finite difference approximations. The small parameter $\varepsilon_{\mathrm{FD}}^{(\bullet)}$ is defined by

$$\varepsilon_{\mathrm{FD}}^{(\bullet)} := \frac{\sqrt{\varepsilon_{\mathrm{machine}}}}{\varepsilon\,\|\Delta(\bullet)\|_2},$$

where ε is the reference Mach number, and $\varepsilon_{\mathrm{machine}}$ denotes the approximate machine accuracy, see [36]. In fortran, there exists an intrinsic *epsilon* function, which provides an approximate value of $\varepsilon_{\mathrm{machine}} \approx 2\cdot 10^{-16}$.

An appropriate and efficient preconditioner is necessary for the faster convergence of the GMRES method. We use a problem-tailored extended Block-Jacobi preconditioner as used in [36]. Refer to the figures [36, Fig. 2 and Fig. 4] and the section [36, Sec. 3.2.3] for more details on the construction of the extended Block-Jacobi preconditioner. As shown in [36], neglecting the Hessian contribution ($\frac{\partial\mathbf{R}^{(2)}(\mathbf{w})}{\partial\mathbf{w}}$) from the linear equation and preconditioner exerts only a negligible influence on Newton's iterations; we also adopted it in this paper.

4.1. Newton stopping criteria

For a given $\varepsilon_{\mathrm{Newton}}$, the stopping criterion for the k^{th} Newton iteration is given by

$$\left\|\mathcal{G}^{w}(X^{k})\right\| < \varepsilon_{\mathrm{Newton}}\cdot\left\|\mathcal{G}^{w}(X^{0})\right\|,$$

where $\|\cdot\|$ is the Euclidean norm.

The residual norm stagnates in the Newton procedure as the stiffness of the problem increases ($\varepsilon \le 10^{-2}$). The implicit solver cannot reduce the residual norm beyond a specific value for low values of $\varepsilon_{\mathrm{Newton}}$. In order to address this condition and modify the Newton stopping criteria, we use a stagnated residual norm stopping criteria. Consider the sequence of residual norms $\left\{\left\|\mathcal{G}^{w}(X^{0})\right\|, \left\|\mathcal{G}^{w}(X^{1})\right\|, \left\|\mathcal{G}^{w}(X^{2})\right\|, \ldots, \left\|\mathcal{G}^{w}(X^{k})\right\|\right\}$ in the Newton iteration. For each Newton iteration k, we define a measure of stagnation (**Ms**) using

$$\mathbf{Ms}_k := \frac{\left|\left\|\mathcal{G}^{w}(X^{k})\right\| - \left\|\mathcal{G}^{w}(X^{k-1})\right\|\right|}{\left\|\mathcal{G}^{w}(X^{k-1})\right\|}.$$

In each Newton iteration, $\mathbf{Ms}_k$ will be computed. If $\mathbf{Ms}_k \le \mathcal{M}_{ms}$ and $\left\|\mathcal{G}^{w}(X^{k})\right\| \le \mathcal{M}_{\eta}$ holds for predefined values $\mathcal{M}_{ms}$ and $\mathcal{M}_{\eta}$, the stagnation counter increases by one. When the stagnation counter hits a predefined value $\mathcal{M}_C$, the Newton algorithm stops, indicating that the residual was at a stagnated value for at least $\mathcal{M}_C$ iterations. If the residual norm oscillates along with the stagnation condition, the algorithm chooses the Newton solution with minimal residual norm. The values $\mathcal{M}_{ms}$, $\mathcal{M}_{\eta}$ and $\mathcal{M}_C$ are user-defined; hence, they will be mentioned for the obtained numerical results in the upcoming section.

5. Numerical Results

The aforementioned time-stepping schemes are implemented in the open-source code FLEXI[2], which is developed to solve hyperbolic-parabolic conservation equations in a discontinuous Galerkin setting [39].

[2] www.flexi-project.org, GNU GPL v3.0

5.1. Higher-order traveling vortex (HOTV)

In this section, we cite an explicit solution to the isentropic Euler equations (3) with parameters $\kappa = 0.5$ and $\gamma = 2$ from [40, 34, 41]. The spatial domain under consideration is $\Omega = [0, 1] \times [0, 1]$, with periodic boundary conditions. The initial conditions are

$$\rho(x, 0) = 2 + (500\varepsilon)^2 \cdot \begin{cases} 0.5e^{\frac{2}{\Delta r}}\Delta r - \mathrm{Ei}(\frac{2}{\Delta r}), & \text{for } r < 0.5, \\ 0, & \text{otherwise} \end{cases},$$

$$\mathbf{v}(x, 0) = \begin{pmatrix} 0.5 \\ 0 \end{pmatrix} + 500 \begin{pmatrix} -x_2 + 0.5 \\ x_1 - 0.5 \end{pmatrix} \cdot \begin{cases} e^{\frac{1}{\Delta r}}, & \text{for } r < 0.5, \\ 0, & \text{otherwise} \end{cases},$$

where $r := \sqrt{(x_1 - 0.5)^2 + (x_2 - 0.5)^2}$ and $\Delta r := r^2 - 0.25$. The exponential integral function Ei(x) is given by

$$\mathrm{Ei}(x) := \int_{-\infty}^{x} \frac{e^t}{t} dt,$$

and it is computed using the algorithm given in [42, Sec. 6.3]. The exact solution corresponding to the above given initial conditions is

$$\rho(x, t) = \rho\left(\begin{pmatrix} x_1 - 0.5t \\ x_2 \end{pmatrix}, 0\right), \quad \mathbf{v}(x, t) = \mathbf{v}\left(\begin{pmatrix} x_1 - 0.5t \\ x_2 \end{pmatrix}, 0\right). \tag{46}$$

5.2. Timestep selection for the IMEX schemes with DGSEM spatial discretization

We use the formulation of the allowable timestep given in [43] for an explicit scheme under the constraints of CFL conditions for discontinuous Galerkin schemes. For the IMEX scheme, it is given by

$$\Delta t = cfl_{\mathrm{conv}} \cdot \frac{\Delta l}{(2\mathcal{N}_p + 1) \cdot |\lambda_{\mathrm{E}}|_{\max}}, \tag{47}$$

where cfl_{conv} is the convective CFL value, $|\lambda_{\mathrm{E}}|_{\max}$ is the maximum of the absolute values of the eigenvalues associated with the explicit flux given by

$$\lambda_1 = 0, \quad \lambda_2 = \mathbf{v} \cdot \mathbf{n} \quad \text{and} \quad \lambda_3 = 2\mathbf{v} \cdot \mathbf{n}, \tag{48}$$

and Δl is the characteristic length of the considered mesh element. The convective CFL value is chosen to be $cfl_{\mathrm{conv}} \leq 1$ for stability.

5.3. Comparison of explicitness-preserving and non-explicitness-preserving IMEX schemes

There are two ways to consider the implicit part of the second derivative, as mentioned in Rem. 1. The schemes that use the formulation in Eq. (14) are called *explicitness-preserving*, and those that use Eq. (13) can be termed *non-explicitness-preserving* IMEX schemes. Hence, any two-derivative IMEX schemes can have these two variants. We have only shown the analysis for explicitness-preserving IMEX-Taylor and IMEX-HBPC schemes in the paper. However, we have verified that Theorems 1, 2 and 3 also hold for their non-explicitness-preserving variants.

Explicitness-preserving schemes are more natural as they give more justice to the concept of an IMEX splitting. However, we compare the explicitness and non-explicitness-preserving schemes using IMEX-HBPC(4, 3) to understand their computational disparities. Regarding the implicit solver, there is a slight difference on the third block of the Jacobian (45), where it is $-\frac{\partial \mathbf{R}^{(1)}(\mathbf{w})}{\partial \mathbf{w}}$ for the non-explicitness-preserving and $-\frac{\partial \mathbf{R}_1^{(1)}(\mathbf{w})}{\partial \mathbf{w}}$ for the explicitness-preserving scheme. The difference is also adapted in the respective preconditioner.

For the comparison test, the spatial discretization is set to $\mathcal{N}_E \in \{16 \times 16, 24 \times 24, 32 \times 32\}$ with $\mathcal{N}_p = 4$. The cfl_{conv} value is fixed at 0.9. The relative tolerances for the Newton and GMRES iterations are chosen to be 10^{-6} and 10^{-2} respectively with values $\mathcal{M}_{ms} = 0.1$, $\mathcal{M}_\eta = 10^{-5}$ and $\mathcal{M}_C = 3$ for the residual norm stagnation criteria. Then the L_2-errors are computed at final time $T_{\mathrm{end}} = 0.1$, compared to the explicit solution (46).

In the first column of Fig. 1, the L_2-error is plotted for various values of stiffness parameters ε ranging from 10^{-3} to 10^{0}. The scheme converges for explicitness-preserving and non-explicitness-preserving schemes without fail for all the values of $\varepsilon \geq 10^{-3}$, and convergence is nearly impossible to distinguish.

In the third column of Fig. 1, the average number of GMRES iterations per timestep is plotted for various values of stiffness parameters. There is a notable difference in the linear iterations between explicitness-preserving and non-explicitness-preserving schemes utilized per timestep. The non-explicitness-preserving strategy consumes extra GMRES iterations to achieve the same error reduction as the explicitness-preserving strategy. The difference increases as the stiffness of the system increases. The GMRES iteration consumption is directly reflected in the computational time required for the schemes to complete the simulation. Therefore, in the second column of Fig. 1, it can be seen that the non-explicitness-preserving strategy consumed enormous extra time compared to the explicitness-preserving strategy. These dissimilarities seen above are mainly due to the additional non-linearity and unnecessary function inversions that the non-explicitness-preserving brings into the scheme.

The disparities discussed above and the results shown in Fig. 1 show the importance of choosing the explicitness-preserving strategy for a multi-derivative IMEX scheme. Therefore, we use the explicitness-preserving scheme for the higher-order extensions in the coming sections to obtain the numerical results.

5.4. Numerical results of IMEX-HBPC schemes on HOTV

Fig. 2 shows the L_2-errors of the IMEX-HBPC schemes up to order eight for various stiffness values. The spatial discretization is set to $\mathcal{N}_E = \{16 \times 16\}$ and two sets of polynomial degrees $\mathcal{N}_p$. The first set includes polynomial degrees 3, 5, and 7 for IMEX-HBPC(4, $k_{\max}$), IMEX-HBPC(6, $k_{\max}$) and IMEX-HBPC(8, $k_{\max}$), respectively. The second set takes polynomial degrees 4, 6, and 8. The errors are calculated compared to the explicit solution (46) at the final time $T_{\text{end}} = 2$. The relative tolerances for the Newton and GMRES iterations are chosen to be 10^{-6} and 10^{-2}, respectively, with values $\mathcal{M}_{ms} = 0.3$, $\mathcal{M}_{\eta} = 10^{-5}$ and $\mathcal{M}_C = 1$ for the residual norm stagnation criteria.

The errors are computed for IMEX-HBPC(4, $k_{\max}$) and IMEX-HBPC(6, $k_{\max}$) schemes at a fixed $cfl_{\text{conv}} = 0.9$. The fourth-order scheme showcases convergence for all $\varepsilon \geq 10^{-3}$ for correction steps $k_{\max} \leq 3$. However, the sixth-order scheme mandates more correction steps as the stiffness increases to achieve the desired convergence. The correction steps ranged from $5 \leq k_{\max} \leq 12$ for $10^{-3} \leq \varepsilon \leq 1$. The fourth-order scheme is A-stable for all values of $k_{\max}$, and the sixth order scheme is at least $A(89.7^{\circ})$-stable for all possible values of $k_{\max}$, see the stability plots in [20, Fig. 3]. The demand for additional correction steps in IMEX-HBPC(6, $k_{\max}$) likely arises from two main factors: the need for enhanced stability and the necessity to mitigate the decrease in convergence order, as outlined in [16, Fig. 2]. Since the stability angle approaches values close to 90° when $k_{\max} > 5$, the latter factor may have contributed more significantly to the requirement for additional correction steps as stiffness increases.

Compared to the IMEX-HBPC(4, $k_{\max}$) and IMEX-HBPC(6, $k_{\max}$) schemes, the eight-order scheme IMEX-HBPC $(8, k_{\max})$ has relatively low stability angle for lower values of $k_{\max}$, see [20, Fig. 3]. The scheme is approximately $A(89.4^{\circ})$-stable for $k_{\max} = 7$, and the stability angle drops monotonically to approximately 88.65° for $k_{\max} = 15$. Then, the stability angle gradually increases (with oscillations) to 89.4° for $k_{\max} = 50$. The severity of the oscillations decreases as $k_{\max}$ is increased. The above-mentioned oscillating stability properties of the IMEX-HBPC(8, $k_{\max}$) scheme have caused difficulty choosing (cfl_{conv} , $k_{\max}$) combinations for stiffer problems. In Fig. 2, the errors were computed for $\varepsilon \geq 10^{-2}$, and schemes exhibited convergence for the given (cfl_{conv} , $k_{\max}$) combinations. In contrast to the IMEX-HBPC(6, $k_{\max}$) scheme, the need for stability improvement may have contributed primarily to the requirement for additional correction steps. The result for $\varepsilon = 10^{-3}$ has been skipped due to the high computational costs required.

5.5. Isentropic Navier-Stokes equations

In this section, we investigate the applicability of the IMEX-HBPC schemes to the Navier-Stokes equations, specifically focusing on the isentropic Navier-Stokes equations presented in [1]. The equations are given by

$$\begin{pmatrix} \rho \\ \rho \mathbf{v} \end{pmatrix}_t + \nabla \cdot \begin{pmatrix} \rho \mathbf{v} \\ \rho \mathbf{v} \otimes \mathbf{v} + \frac{p}{\varepsilon^2} \mathbf{Id} \end{pmatrix} = \nabla \cdot \begin{pmatrix} 0 \\ \mu \nabla \mathbf{v} \end{pmatrix}, \tag{49}$$

where μ denotes the dynamic viscosity of the fluid. The above system (49) is closed with the equation of state $p(\rho) := \kappa \rho^{\gamma}$.

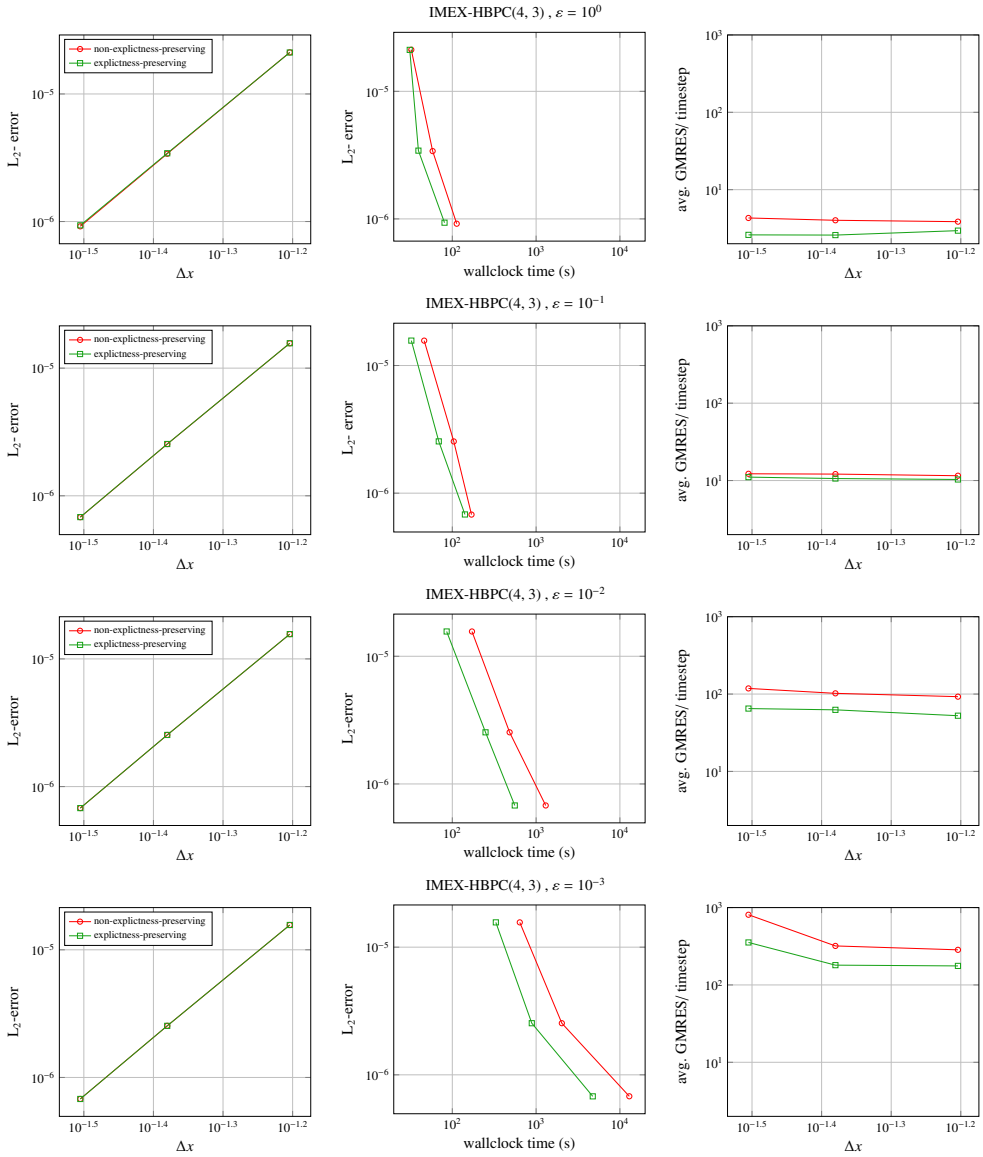


Figure 1. L_2-error vs meshsize (left), L_2-error vs wallclock time (middle), and average number of GMRES iterations per timestep vs meshsize (right) for HOTV (Sec. 5.1) with $T_{\text{end}} = 0.1$ using IMEX-HBPC(4, 3) timestepping scheme plotted for a fixed $cfl_{\text{conv}} = 0.9$. The comparison is made for the strategies *non-explicitness-preserving* and *explicitness-preserving* for different values of the stiffness parameter ε. The spatial discretization is set to $\mathcal{N}_E \in \{16 \times 16, 24 \times 24, 32 \times 32\}$ with $\mathcal{N}_p = 4$. The Newton and GMRES tolerances are taken to be 10^{-6} and 10^{-2}, respectively, within a limit of 50 Newton iterations per implicit solve and 700 GMRES iterations with 50 restarts per Newton iteration. For the residual norm stagnation criteria, the values are chosen to be $\mathcal{M}_{ms} = 0.1$, $\mathcal{M}_\eta = 10^{-5}$ and $\mathcal{M}_C = 3$. The simulations use spatial parallelization on 36 processors.

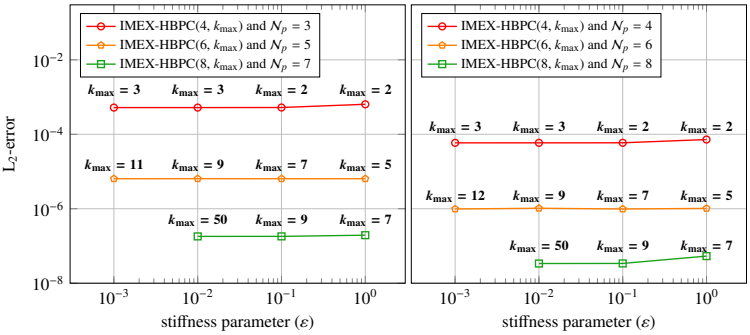


Figure 2. Total L_2-error for HOTV (Sec.5.1) at $T_{end} = 2$ for the IMEX-HBPC schemes with a constant $cfl_{conv} = 0.9$ for various stiffness parameters. The IMEX-HBPC$(8, k_{max})$ scheme has used a different $cfl_{conv} = 0.45$ for $\varepsilon = 10^{-1}$. The spatial discretization is set to $\mathcal{N}_E = \{16 \times 16\}$ with two sets of polynomial degrees $\mathcal{N}_p$. On the right, $\mathcal{N}_p$ equals 3, 5, and 7 for the schemes IMEX-HBPC$(4, k_{max})$, IMEX-HBPC$(6, k_{max})$, and IMEX-HBPC$(8, k_{max})$, respectively. On the left, $\mathcal{N}_p$ equals 4, 6, and 8 for the schemes IMEX-HBPC$(4, k_{max})$, IMEX-HBPC$(6, k_{max})$, and IMEX-HBPC$(8, k_{max})$, respectively. The Newton and GMRES tolerances are taken to be 10^{-6} and 10^{-2}, respectively, with a limit of 50 Newton iterations per implicit solve and 700 GMRES iterations with 50 restarts per Newton iteration. For the residual norm stagnation criteria, the values are chosen to be $\mathcal{M}_{ms} = 0.3$, $\mathcal{M}_\eta = 10^{-5}$, and $\mathcal{M}_C = 1$. The simulations on the left use spatial parallelization on 36 processors, and simulations on the right use spatial parallelization on 72 processors. IMEX-HBPC$(8, 50)$ with $\mathcal{N}_p = 7$ is also run on 72 processors. The result for $\varepsilon = 10^{-3}$ for IMEX-HBPC$(8, k_{max})$ has been skipped due to the high computational costs required. The correction steps k_{max} for the IMEX-HBPC schemes are chosen for different values of $\varepsilon < 1$ such that they yield stable solutions with maximum error reduction (as observed for $\varepsilon = 1$) for the given $cfl_{conv} = 0.9$. Moreover, the required k_{max} values decrease as the cfl_{conv} value is reduced.

Considering well-prepared initial data as in (5) with periodic boundary conditions, we obtain in the limit $\varepsilon \to 0$ that Eq. (49) reduces to the incompressible isentropic Navier–Stokes equations [1]. These limiting equations take the form

$$\begin{aligned} \rho_{(0)} \equiv \text{const} > 0, \quad \nabla \cdot \mathbf{v}_{(0)} = 0, \\ (\mathbf{v}_{(0)})_t + \nabla \cdot (\mathbf{v}_{(0)} \otimes \mathbf{v}_{(0)}) + \frac{\nabla p_{(2)}}{\rho_{(0)}} = \frac{\mu}{\rho_{(0)}} \nabla \cdot \nabla \mathbf{v}_{(0)} \,. \end{aligned} \tag{50}$$

Following the implementation described in [1], we explicitly account for the parabolic component in the flux splitting. Therefore, we define the implicit and explicit flux components for the isentropic Navier-Stokes equations (49) as

$$\mathbf{F}_{\mathrm{I}}(\mathbf{w}) = \begin{pmatrix} \rho \mathbf{v} \\ \frac{1-\varepsilon^2}{\varepsilon^2} p \mathbf{Id} \end{pmatrix} \quad \text{and} \quad \mathbf{F}_{\mathrm{E}}(\mathbf{w}) = \begin{pmatrix} 0 \\ \rho \mathbf{v} \otimes \mathbf{v} + p \mathbf{Id} \end{pmatrix} - \begin{pmatrix} 0 \\ \mu \nabla \mathbf{v} \end{pmatrix}. \tag{51}$$

To discretize the second-order equations, we employ the BR1 lifting approach [44, 45]. The lifting procedure extends the second-order equations into an equivalent system of first-order PDEs [46, 19]. We consider the following two test cases for the numerical investigations:

(a) The equation (49) with parameters $\kappa = 0.5$, $\gamma = 2$, and $\mu = 0.01$, initialized with a vortex as described in Sec. 5.1. The spatial domain is $\Omega = [0, 1] \times [0, 1]$ equipped with periodic boundary conditions.

(b) The equation (49) with parameters $\kappa = 1$, $\gamma = 2$, and $\mu = 0.01$, initialized with

$$\begin{aligned} \rho(x, 0) &= 1 + \varepsilon^2 \sin^2 (2\pi(x_1 + x_2)) \,, \\ u(x, 0) &= \sin (2\pi(x_1 - x_2)) \,, \\ v(x, 0) &= \sin (2\pi(x_1 - x_2)) \,. \end{aligned} \tag{52}$$

The spatial domain is again $\Omega = [0, 1] \times [0, 1]$ with periodic boundary conditions.

Since no exact solutions are available for these two cases, we utilize a fine explicit reference solution corresponding to each value of ε to evaluate the error. As the parabolic term is treated explicitly, the allowable timestep for the IMEX schemes are determined using the criteria

$$\Delta t = \min\left\{ cfl_{\text{conv}} \cdot \frac{\Delta l}{(2\mathcal{N}_p + 1) \cdot |\lambda_{\text{E}}|_{\max}} \,,\; cfl_{\text{diff}} \cdot \frac{\Delta l^2}{(2\mathcal{N}_p + 1)^2 \cdot \nu} \right\}, \tag{53}$$

where cfl_{diff} is a scheme-dependent safety factor for the diffusive stability restriction [39], and $\nu = \frac{\mu}{\rho}$ represents the kinematic viscosity.

In Fig. 3, the L_2-error of the IMEX-HBPC schemes at $T_{\text{end}} = 0.5$ for the isentropic Navier-Stokes equations (49) is presented for two test cases with various stiffness parameters. The spatial discretization consists of $\mathcal{N}_E = 16 \times 16$ elements, utilizing polynomial degrees $\mathcal{N}_p = 3$ for IMEX-HBPC(4, $k_{\max}$) and $\mathcal{N}_p = 5$ for IMEX-HBPC(6, $k_{\max}$). The parameters cfl_{conv} and cfl_{diff} generally depend on the selected polynomial degree and the time-stepping scheme. However, these values are yet to be optimized for the IMEX-HBPC schemes. Here, we set cfl_{conv} to 0.9 for all schemes, while cfl_{diff} is determined to be 0.495 for IMEX-HBPC(4, $k_{\max}$) and 0.288 for IMEX-HBPC(6, $k_{\max}$). For the Newton and GMRES methods, the tolerances are set to 10^{-6} and 10^{-2}. The criteria for the residual norm stagnation are established with values of $\mathcal{M}_{ms} = 0.3$, $\mathcal{M}_\eta = 10^{-5}$, and $\mathcal{M}_C = 1$.

It can be observed in Fig. 3 that both the fourth-order and sixth-order IMEX-HBPC schemes are stable and are converging towards their respective explicit reference solutions across all tested stiffness parameters. The correction steps were determined based on trends observed in the numerical results for the purely hyperbolic problem discussed in Sec. 5.1. While the eighth-order scheme is also expected to exhibit similar error reduction as mentioned in Sec. 5.1, its results have been omitted due to the increased computational costs required for the scheme and obtaining reference solutions for error comparison. Furthermore, while the asymptotic analysis of the isentropic Navier-Stokes equations has not been conducted in this paper, the numerical results indicate that the IMEX-HPBC scheme effectively preserves the asymptotic behavior of the parabolic equations (49) for the splitting given in Eq. (51).

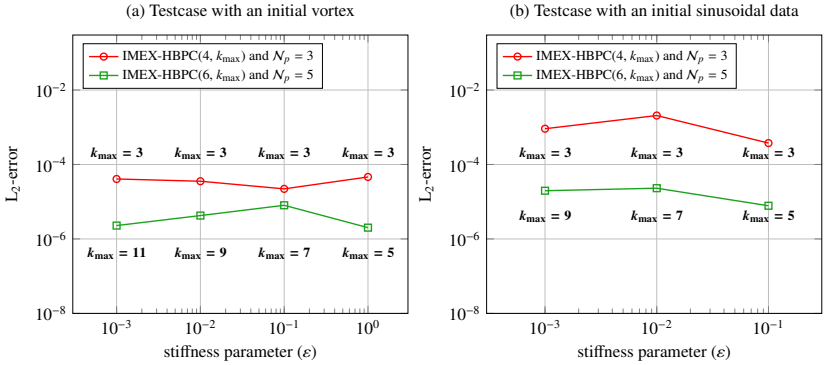


Figure 3. Total L_2-error at $T_{\text{end}} = 0.5$ for the isentropic Navier-Stokes equations (49) with vortex (left) and sinusoidal (right) initial data, computed using IMEX-HBPC schemes for various stiffness parameters. The boundary conditions are periodic. The spatial discretization consists of $\mathcal{N}_E = 16 \times 16$ elements with polynomial degrees $\mathcal{N}_p = 3$ for IMEX-HBPC(4, $k_{\max}$) and $\mathcal{N}_p = 5$ for IMEX-HBPC(6, $k_{\max}$). The value cfl_{conv} is set to 0.9 for all schemes, while cfl_{diff} is set to 0.495 and 0.288 for IMEX-HBPC(4, $k_{\max}$) and IMEX-HBPC(6, $k_{\max}$), respectively. The correction steps $k_{\max}$ were determined based on trends observed in the numerical results for the purely hyperbolic problem discussed in Sec. 5.1. The Newton and GMRES tolerances are taken to be 10^{-6} and 10^{-2}, respectively, with a limit of 50 Newton iterations per implicit solve and 700 GMRES iterations with 50 restarts per Newton iteration. For the residual norm stagnation criteria, the values are chosen to be $\mathcal{M}_{ms} = 0.3$, $\mathcal{M}_\eta = 10^{-5}$, and $\mathcal{M}_C = 1$. The simulations use spatial parallelization on 36 processors. The result for IMEX-HBPC(8, $k_{\max}$) has been skipped due to the high computational costs required.

5.6. *Efficiency comparisons with existing IMEX schemes*

To evaluate the efficiency of IMEX-HBPC schemes in terms of wall-clock time required for error reduction, we compare them with several existing higher-order IMEX schemes. (Please note that we are not aware of IMEX Runge-

Kutta schemes with order greater than five.) For the comparison, we consider the one-derivative third-order IMEX scheme IMEX-ARS443, as outlined in [5], and the fourth-order scheme IMEX-ARK4A2 from [47].

The L_2 error and efficiency plots for the methods described above are presented in Fig. 4 for the HOTV-vortex problem defined in equation (46) at $T_{end} = 0.5$ with a fixed $cfl_{conv} = 0.5$. We selected IMEX-HBPC(4, 1) as a candidate for a third-order two-derivative IMEX scheme since it achieves third-order accuracy when we limit the number of correction steps to one. The IMEX-HBPC(4, k_{max}) requires at least $k_{max} = 2$ to theoretically achieve fourth-order accuracy. The spatial discretization is set to $N_E = 16 \times 16$ elements, consistent with previous numerical results. The polynomial degrees N_p are chosen to be 2 and 3, corresponding to third and fourth-order schemes, respectively. For the implicit solvers, the Newton and GMRES methods, the tolerances are set to 10^{-6} and 10^{-2}. Additionally, the preconditioner is rebuilt after every 10 timesteps.

In Fig. 4, it can be observed that the computational time for all IMEX schemes increases with stiffness. For stiffness parameters $\varepsilon \geq 10^{-1}$, the run time is nearly the same across all schemes. However, for $\varepsilon \leq 10^{-2}$, one-derivative schemes achieve the same level of error with lower computational time compared to two-derivative schemes, and this difference becomes more pronounced as stiffness increases. This distinction can be attributed to two main factors: first, two-derivative methods require an additional evaluation of the second derivative; second, the incorporation of an additional variable σ to reduce the stencil and enhance stability has led to larger systems of nonlinear equations. Consequently, the computational cost for building the preconditioner and solving the nonlinear system can be higher.

On the other hand, IMEX-HBPC schemes can be easily extended to higher orders beyond five, which is a notable advantage over IMEX Runge-Kutta schemes, in particular for applications that desire very tight numerical tolerances. Furthermore, parallel-in-time IMEX-HBPC schemes [19] could be a promising approach for minimizing computational time; however, this aspect is not studied in the current paper.

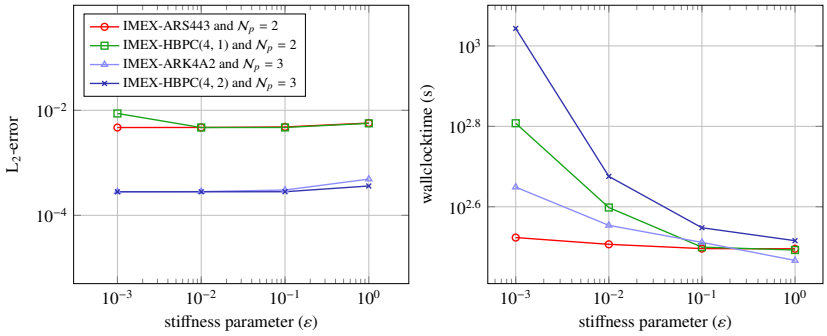


Figure 4. Total L_2-error (left) and the wall-clock time (right) for HOT-vortex at $T_{end} = 0.5$ shown for third (IMEX-ARS443,IMEX-HBPC(4, 1)) and fourth-order (IMEX-ARK4A2, IMEX-HBPC(4, 2)) IMEX methods with a constant $cfl_{conv} = 0.5$ for various stiffness parameters. The spatial discretization is set to $N_E = \{16 \times 16\}$ with polynomial degrees N_p equals 2 and 3 for the third and fourth-order methods, respectively. The Newton and GMRES tolerances are taken to be 10^{-6} and 10^{-2}, respectively, with a limit of 50 Newton iterations per implicit solve and 700 GMRES iterations with 50 restarts per Newton iteration. For the residual norm stagnation criteria, the values are chosen to be $M_{ms} = 0.3$, $M_{\eta} = 10^{-5}$, and $M_C = 1$. The simulations use spatial parallelization on 36 processors.

6. Conclusions and Outlook

In this paper, we have combined the higher-order two-derivative IMEX-HBPC scheme with higher-order discontinuous Galerkin spatial discretization for isentropic Euler equations. The IMEX-HBPC scheme utilized the DeTa flux splitting [31]. The asymptotic analysis has been done for the semi-discrete and fully-discrete formulations of the isentropic Euler equations. The overall scheme is proven to be asymptotic preserving. Numerical results showed that the IMEX schemes are stable under convective CFL conditions independent of ε.

The implicit part of the second derivative flux has been evaluated in two ways: explicitness-preserving (14) and non-explicitness-preserving (13). It has been shown in the comparison test that the explicitness-preserving scheme is

more efficient than the non-explicitness-preserving scheme in terms of the consumed linear iterations. The schemes were theoretically proved asymptotically consistent with the appropriate choice of numerical fluxes. Numerical experiments were performed on the high-order traveling vortex (HOTV) (46) with periodic boundary conditions. Results have been shown for IMEX-HBPC schemes up to order eight, see Fig. 2. All the schemes have exhibited a reduction in the error by their order of accuracy for fixed convective CFL conditions, uniformly for all the ε values.

By adapting the strategy of explicitly treating diffusion terms, numerical results were also obtained for isentropic Navier-Stokes equations. Results for fourth and sixth-order IMEX-HBPC schemes were shown, demonstrating convergence across all stiffness ranges under the advection-diffusion CFL restrictions (53). In terms of efficiency in error reduction with respect to computational time, the one-derivative schemes showed better performance at lower ε values.

One of the future research directions is to investigate other possible flux splitting options for the isentropic Euler equations. Also, we are interested in studying the IMEX-HBPC schemes for the full Navier-Stokes equations. However, as the PDE system becomes more complex, selecting appropriate flux splitting and numerical fluxes for the full Navier-Stokes equations will be cumbersome. Furthermore, for the parallel-in-time HBPC schemes [19], it is interesting to study and conduct asymptotic analysis on their IMEX versions.

Acknowledgments

Arjun Thenery Manikantan was funded by the “Bijzonder Onderzoeksfonds” (BOF) from UHasselt - project no. BOF21KP12. We acknowledge the VSC (Flemish Supercomputer Center) for providing computing resources. The VSC is funded by the Research Foundation - Flanders (FWO) and the Flemish Government.

References

[1] J. Haack, S. Jin, J.-G. Liu, An all-speed asymptotic-preserving method for the isentropic Euler and Navier-Stokes equations, Communications in Computational Physics 12 (2012) 955–980.
[2] R. Courant, K. Friedrichs, H. Lewy, Über die partiellen Differenzengleichungen der mathematischen Physik, Mathematische Annalen 100 (1) (1928) 32–74.
[3] R. Hartmann, F. Bassi, I. Bosnyakov, L. Botti, A. Colombo, A. Crivellini, M. Franciolini, T. Leicht, E. Martin, F. Massa, et al., Implicit methods, in: TILDA: Towards Industrial LES/DNS in Aeronautics, Springer, 2021, pp. 11–59.
[4] U. M. Ascher, S. Ruuth, B. Wetton, Implicit-Explicit methods for time-dependent partial differential equations, SIAM Journal on Numerical Analysis 32 (1995) 797–823.
[5] U. M. Ascher, S. Ruuth, R. Spiteri, Implicit-explicit Runge-Kutta methods for time-dependent partial differential equations, Applied Numerical Mathematics 25 (1997) 151–167.
[6] S. Boscarino, Error analysis of IMEX Runge-Kutta methods derived from differential-algebraic systems, SIAM Journal on Numerical Analysis 45 (2007) 1600–1621.
[7] S. Boscarino, G. Russo, On a class of uniformly accurate IMEX Runge-Kutta schemes and applications to hyperbolic systems with relaxation, SIAM Journal on Scientific Computing 31 (3) (2009) 1926–1945.
[8] L. Pareschi, G. Russo, Implicit-explicit Runge-Kutta schemes and applications to hyperbolic systems with relaxation, Journal of Scientific Computing 25 (1-2) (2005) 129–155.
[9] S. Boscarino, G. Russo, Asymptotic preserving methods for quasilinear hyperbolic systems with stiff relaxation: a review, SeMA Journal. Boletin de la Sociedad Española de Matemática Aplicada 81 (1) (2024) 3–49. doi:10.1007/s40324-024-00351-x.
URL https://doi.org/10.1007/s40324-024-00351-x
[10] S. Gottlieb, C.-W. Shu, E. Tadmor, Strong stability-preserving high-order time discretization methods, SIAM Review 43 (1) (2001) 89–112.
[11] R. Chan, A. Tsai, On explicit two-derivative Runge-Kutta methods, Numerical Algorithms 53 (2010) 171–194.
[12] E. Hairer, G. Wanner, Multistep-multistage-multiderivative methods for ordinary differential equations, Computing (Arch. Elektron. Rechnen) 11 (3) (1973) 287–303.
[13] D. Seal, Y. Güçlü, A. Christlieb, High-order multiderivative time integrators for hyperbolic conservation laws, Journal of Scientific Computing 60 (2014) 101–140.
[14] A. Moradi, A. Abdi, J. Farzi, Strong stability preserving second derivative general linear methods with Runge–Kutta stability, Journal of Scientific Computing 85 (1) (2020) 1.
[15] J. Chouchoulis, J. Schütz, J. Zeifang, Jacobian-free explicit multiderivative Runge–Kutta methods for hyperbolic conservation laws, Journal of Scientific Computing 90 (3) (2022) 96.
[16] J. Schütz, D. Seal, An asymptotic preserving semi-implicit multiderivative solver, Applied Numerical Mathematics 160 (2021) 84–101.
[17] S. Gottlieb, Z. J. Grant, J. Hu, R. Shu, High order strong stability preserving multiderivative implicit and IMEX Runge–Kutta methods with asymptotic preserving properties, SIAM Journal on Numerical Analysis 60 (1) (2022) 423–449.
[18] A. Moradi, A. Abdi, G. Hojjati, Strong stability preserving implicit and implicit–explicit second derivative general linear methods with RK stability, Computational and Applied Mathematics 41 (4) (2022) 135.

[19] J. Schütz, D. C. Seal, J. Zeifang, Parallel-in-time high-order multiderivative IMEX solvers, Journal of Scientific Computing 90 (54) (2022) 1–33.
[20] J. Zeifang, J. Schütz, D. Seal, Stability of implicit multiderivative deferred correction methods, BIT Numerical Mathematics (2022).
[21] A. Thenery Manikantan, J. Schütz, Multi-step Hermite-Birkhoff predictor-corrector schemes, Applied Numerical Mathematics 205 (2024) 281–295. doi:https://doi.org/10.1016/j.apnum.2024.07.011.
URL https://www.sciencedirect.com/science/article/pii/S0168927424001910
[22] J. Zeifang, A. Thenery Manikantan, J. Schütz, Time parallelism and Newton-adaptivity of the two-derivative deferred correction discontinuous Galerkin method, Applied Mathematics and Computation 457 (2023) 128198.
[23] D. A. Kopriva, Implementing spectral methods for partial differential equations: Algorithms for scientists and engineers, Springer Science & Business Media, 2009.
[24] S. Jin, Asymptotic preserving (AP) schemes for multiscale kinetic and hyperbolic equations: A review, Rivista di Matematica della Universita Parma 3 (2012) 177–216.
[25] F. Cordier, P. Degond, A. Kumbaro, An asymptotic-preserving all-speed scheme for the Euler and Navier-Stokes equations, Journal of Computational Physics 231 (2012) 5685–5704.
[26] S. Noelle, G. Bispen, K. Arun, M. Lukáčová-Medvid'ová, C.-D. Munz, A weakly asymptotic preserving low Mach number scheme for the Euler equations of gas dynamics, SIAM Journal on Scientific Computing 36 (2014) B989–B1024.
[27] G. Bispen, M. Lukáčová-Medvid'ová, L. Yelash, Asymptotic preserving IMEX finite volume schemes for low Mach number Euler equations with gravitation, Journal of Computational Physics 335 (2017) 222–248.
[28] X. Xie, H. Dong, M. Li, Numerical simulations of the shallow water equations with coriolis forces in full froude number by an asymptotic preserving dg scheme, Journal of Applied Mathematics and Computing (2025). doi:10.1007/s12190-025-02501-4.
URL https://doi.org/10.1007/s12190-025-02501-4
[29] E. Theodosiou, J. Schütz, D. Seal, An explicitness-preserving imex-split multiderivative method, Computers & Mathematics with Applications 158 (2024) 139–149. doi:https://doi.org/10.1016/j.camwa.2023.12.040.
URL https://www.sciencedirect.com/science/article/pii/S089812212400021X
[30] S. Klainerman, A. Majda, Singular limits of quasilinear hyperbolic systems with large parameters and the incompressible limit of compressible fluids, Communications on Pure and Applied Mathematics 34 (1981) 481–524.
[31] P. Degond, M. Tang, All speed scheme for the low Mach number limit of the isentropic Euler equation, Communications in Computational Physics 10 (2011) 1–31.
[32] K. Kaiser, J. Schütz, R. Schöbel, S. Noelle, A new stable splitting for the isentropic Euler equations, Journal of Scientific Computing 70 (2017) 1390–1407.
[33] J. Schütz, D. Seal, A. Jaust, Implicit multiderivative collocation solvers for linear partial differential equations with discontinuous Galerkin spatial discretizations, Journal of Scientific Computing 73 (2017) 1145–1163.
[34] K. Kaiser, J. Schütz, A high-order method for weakly compressible flows, Communications in Computational Physics 22 (4) (2017) 1150–1174.
[35] K. Kaiser, J. Schütz, Asymptotic error analysis of an IMEX Runge–Kutta method, Journal of Computational and Applied Mathematics 343 (2018) 139–154.
[36] J. Zeifang, J. Schütz, Implicit two-derivative deferred correction time discretization for the discontinuous Galerkin method, Journal of Computational Physics 464 (2022) 111353.
[37] M. Franciolini, A. Crivellini, A. Nigro, On the efficiency of a matrix-free linearly implicit time integration strategy for high-order discontinuous Galerkin solutions of incompressible turbulent flows, Computers & Fluids 159 (2017) 276–294.
[38] D. A. Knoll, D. E. Keyes, Jacobian-free Newton–Krylov methods: a survey of approaches and applications, Journal of Computational Physics 193 (2004) 357–397.
[39] N. Krais, A. Beck, T. Bolemann, H. Frank, D. Flad, G. Gassner, F. Hindenlang, M. Hoffmann, T. Kuhn, M. Sonntag, et al., FLEXI: A high order discontinuous Galerkin framework for hyperbolic–parabolic conservation laws, Computers & Mathematics with Applications 81 (2021) 186–219.
[40] G. Bispen, K. R. Arun, M. Lukáčová-Medvid'ová, S. Noelle, IMEX large time step finite volume methods for low Froude number shallow water flows, Communications in Computational Physics 16 (2014) 307–347.
[41] J. Zeifang, K. Kaiser, A. Beck, J. Schütz, C.-D. Munz, Efficient high-order discontinuous Galerkin computations of low Mach number flows, Communications in Applied Mathematics and Computational Science 13 (2018) 243–270.
[42] William H. Press, S. A. Teukolsky, W. T. Vetterling, B. P. Flannery, Numerical recipes 3rd edition, 3rd Edition, Cambridge University Press, Cambridge, England, 2007.
[43] B. Cockburn, C. W. Shu, Runge-Kutta discontinuous Galerkin methods for convection-dominated problems, Journal of Scientific Computing 16 (2001) 173–261.
[44] F. Bassi, S. Rebay, A high-order accurate discontinuous finite-element method for the numerical solution of the compressible Navier-Stokes equations, Journal of Computational Physics 131 (1997) 267–279.
[45] S. Ortleb, A comparative Fourier analysis of discontinuous Galerkin schemes for advection–diffusion with respect to BR1, BR2, and local discontinuous Galerkin diffusion discretization, Mathematical Methods in the Applied Sciences 43 (13) (2020) 7841–7863.
[46] D. N. Arnold, F. Brezzi, B. Cockburn, L. D. Marini, Unified analysis of discontinuous Galerkin methods for elliptic problems, SIAM Journal on Numerical Analysis 39 (2002) 1749–1779.
[47] H. Liu, J. Zou, Some new additive Runge–Kutta methods and their applications, Journal of Computational and Applied Mathematics 190 (1-2) (2006) 74–98.

Chapter 3

Conclusions and future work

The singularly perturbed conservation equations pose significant challenges for explicit time integration schemes, particularly when the stiffness parameter $\varepsilon \ll 1$. The primary difficulty stems from the severe timestep restriction imposed by the CFL condition, which depends on ε. For example, in the case of the low-Mach compressible isentropic Euler equations, this leads to a timestep size of

$$\Delta t = \mathcal{O}(\varepsilon \Delta x),$$

which makes explicit schemes unsuitable for such stiff problems as $\varepsilon \to 0$. Therefore, in this thesis, we have explored various time stepping methods that incorporate elements of implicitness, either through algebraic implicit formulations or through repeated explicit sub-steps. As a result, implicitness facilitates the alleviation of severe timestep restrictions. Additionally, most of these methods have a multi-derivative nature, enabling the achievement of higher orders of accuracy. Furthermore, we have presented a new class of multirate multi-derivative explicit schemes capable of handling moderately stiff problems.

In [94, Paper I], we have presented a parallel-in-time (PinT) implit two-derivative discontinous Galerking spectral element method (DGSEM) for Navier-Stokes equations. These schemes are an extension of the HBPC-DGSEM method [181], incor-

porating parallel-in-time strategies developed for ODEs in [92]. As the DGSEM is implemented using the open-source code FLEXI [116], which supports spatial parallelization, this resulted in a scheme that is parallel in both time and space. In comparison to the previous version presented in [92], we further improved the current scheme by parallelizing the stages of the predictor and first corrector steps. In addition, we adopted an adaptive Newton procedure. These developments have enhanced the efficiency of the scheme by reducing computation time, benefiting both from improved parallelization and a reduction in (non-)linear iterations. The temporal parallelization demonstrated a parallel efficiency of approximately 60% to 70%. The combination of parallel-in-time and parallel-in-space methods showed improved performance, particularly in regimes where the efficiency of pure spatial parallelization declined. While PinT schemes have demonstrated superior performance over ESDIRK schemes [193] of the same order on simpler problems, achieving similar benefits for more complex flows—such as the two-dimensional cylinder flow or the three-dimensional Taylor–Green vortex—remains challenging.

There is room for further development, particularly in exploring background quadrature rules within the framework of General Linear Methods (GLMs), as well as extensions to higher-order derivatives using Jacobian-free approaches. Inspired by the preconditioned spectral deferred correction (SDC) schemes in [171, 200], a promising future direction lies in the design of a new class of multi-derivative SDC (MD-SDC) methods. These methods have the potential to incorporate strategies that enable faster convergence to the underlying background schemes. In particular, the HBPC method can be viewed as a special case of the MD-SDC approach. It employs a quadrature rule with uniformly spaced nodes and a diagonal preconditioner.

In [95, Paper II], we developed an m-step HBPC (mS-HBPC) scheme, where the multistep character is embedded in the construction of the underlying background scheme. The sixth- and eighth-order schemes were optimized to achieve A(α)-stability, with the number of correction steps fixed at four and six, respectively. The stability parameters (θ_1, θ_2) were given for various stability angles. It was observed that increasing the stability angle led to an increase in the error constants for both schemes. Numerical results were presented for various non-stiff and stiff ODEs, as well as for

nonlinear PDEs. While the sixth-order schemes consistently achieved the expected convergence across all stability angles, the eighth-order schemes experienced a reduction in convergence order at certain stability angles.

As previously mentioned, the mS-HBPC schemes can serve as a foundation for future work on developing HBPC methods with General Linear Method (GLM)-type background schemes. A thorough mathematical analysis, particularly on order reduction and related aspects, is necessary. Additionally, extending the mS-HBPC schemes to PinT frameworks and applying them to the Navier–Stokes equations are further promising directions for improvement.

In [96, Paper III], we have combined the two-derivative strong-stability-preserving (SSP) schemes from [83] with DGSEM using the open-source code FLEXI [116]. We analyzed the schemes for their A(α)-stability properties. The second-order scheme was found to be A-stable, while the third- and fourth-order schemes were A(α)-stable with stability angles of approximately 79.94° and 84.51°, respectively.

For the third-order, two-stage diagonally implicit two-derivative scheme, we proved that it cannot be both A-stable and SSP, based on the conditions outlined in [83, Theorem 1]. We presented numerical results for both the Euler and Navier-Stokes equations. The second and fourth-order SSP schemes demonstrated the desired convergence; however, the third-order scheme encountered convergence issues due to the presence of poles of the stability function along the imaginary axis. By constructing a family of A(α)-stable third-order SSP schemes, we showed that the location of the instability is more critical than the stability angle itself.

We devised an adaptive coefficient third-order SSP and non-SSP scheme based on the findings from the stability analysis and the distribution of the eigenvalues of the spatially discretized linear advection equations. The adaptive schemes were then applied to the Euler and Navier-Stokes equations under certain assumptions. The non-SSP scheme converged with the desired order for all timestep sizes. The adaptive SSP scheme produced stable solutions for all the given timestep sizes; however, a reduction in order was observed, likely due to large error constants. We leveraged the flexibility in positioning the stability region for the two-derivative schemes to investigate the convergence issues encountered by the fourth-order SSP schemes when

applied to the Navier-Stokes equations. The overall results indicate that A-stability is crucial for time-stepping schemes to reduce error accumulation during simulations that run for extended final times, without compromising the timestep sizes.
Analyzing and implementing the IMEX SSP schemes from [83] for low-Mach number problems using DGSEM is one of the possible future directions. A natural next step would be to study the Jacobian-free multi-derivative SSP schemes and the two-derivative SSP general linear methods (GLMs) [109] within the framework of DGSEM spatial discretization. Since there are only a few studies on deferred correction schemes with the SSP property [201], extending HBPC schemes to possess SSP properties is another possible direction for future research.

As mentioned at the beginning of the conclusion section, we have developed a class of multi-derivative multirate schemes in [164, Paper IV], extending the single-derivative multirate schemes presented in [154, 153]. Unlike IMEX schemes, multirate schemes utilize the efficiency of explicit treatment for both the stiff and non-stiff parts of the ODE or PDE under consideration. The non-stiff part is computed with large timesteps, whereas the stiff part is computed using smaller timesteps or, if possible, solved exactly. We have derived the order conditions up to order four. The multi-derivative treatment was performed up to three derivatives for the non-stiff part. We have investigated the stability properties of the proposed schemes, and their performance was evaluated on a range of highly stiff ODEs and diffusion-dominated PDEs. As the stiffness parameter (ε) of the problem increased, the number of fast steps increased to the order of $\frac{1}{\varepsilon}$ to maintain stable solutions. Therefore, multirate schemes are well-suited for mildly stiff problems.
As future work, the schemes can be extended to the compressible Navier–Stokes equations for low-Mach number flows or to other conservation equations. Additionally, investigating the parallelizability of the fast steps could further improve efficiency on highly stiff problems. Developing order trees for multi-derivative multirate schemes to simplify their presentation is a worthwhile future work.

Finally, in [120, Paper V], we have analyzed the two-derivative schemes for low-Mach flows. We studied the two-derivative IMEX schemes on the isentropic Euler equations using a flux splitting approach from [126], combined with DGSEM spa-

tial discretization. As a first step, we proved the two-derivative second-order Taylor method is asymptotic-preserving (AP) in both the semi-discrete and fully discrete settings, under periodic boundary conditions. The numerical fluxes are carefully chosen for both the stiff and non-stiff parts to achieve the asymptotic-preserving (AP) property. Unlike in the single-derivative IMEX schemes, mixing of stiff and non-stiff components can occur in the higher derivatives of the fluxes. Therefore, the implicit part of the second derivative is treated using the strategy presented in [137] to avoid dependence on the non-stiff terms during function inversions. Consequently, the entire analysis presented in this paper applies to explicitness-preserving schemes as described in [137]. The non-explicitness-preserving schemes can also be shown to be asymptotic-preserving by adopting a similar analysis as presented in the paper. The numerical results comparing explicitness-preserving and non-explicitness-preserving schemes show that the latter requires significantly more computational time, primarily due to the additional non-linearity and unnecessary function inversions introduced by the non-explicitness-preserving approach. The schemes were extended to higher orders, up to eight, using the IMEX-HBPC schemes. We also proved that the IMEX-HBPC schemes are asymptotic-preserving (AP). Numerical results show that all the schemes exhibited a reduction in error consistent with their order of accuracy for fixed convective CFL conditions, uniformly across all values of ε.

As future work, the IMEX-HBPC schemes should be studied on the full Navier–Stokes equations with appropriate splitting strategies. Furthermore, conducting asymptotic-preserving (AP) analysis for parallel-in-time IMEX versions of the schemes is a promising direction. As mentioned previously, a detailed study of multi-derivative spectral deferred correction (MD-SDC) schemes and their IMEX variants opens windows for future research. Additionally, incorporating the compact approximate Taylor (CAT) [202, 203, 204] methods to include higher derivatives using Jacobian-free techniques is another potential extension.

In this thesis, we have presented various nonstandard time integration schemes to alleviate the numerical difficulties caused by stiffness in ordinary and partial differential equations, particularly in flow equations. The contributions made in this work incorporate equation splitting techniques, parallel computing architectures, and

improved nonlinear solvers to enhance the overall efficiency of the proposed schemes.

The proposed future directions aim to further advance these methods to address more complex challenges. From the outcomes discussed in this thesis, parallel-in-time (PinT) schemes have shown promising results in reducing computational time. However, these schemes need further improvement to handle more complex problems effectively. Since the PinT HBPC scheme already possesses a multistep nature, extending the concept of multistep HBPC schemes to multi-derivative GLMs and developing corresponding PinT schemes could offer additional advantages. Incorporating information from previous steps can help reduce the number of implicit stages, thereby increasing computational efficiency. The inclusion of GLMs can also provide additional flexibility to improve both linear stability properties (such as A-stability) and nonlinear stability properties (such as SSP). Therefore, a detailed study and analysis aimed at developing PinT multistep multi-derivative inexact asymptotic-preserving solvers for complex problems could serve as a promising overall future direction of this thesis.

Bibliography

[1] A. Jameson, Time dependent calculations using multigrid, with applications to unsteady flows past airfoils and wings, AIAA Paper 91-1596 (1991).

[2] P. Sváček, M. Feistauer, J. Horáček, Numerical simulation of flow induced airfoil vibrations with large amplitudes, Journal of Fluids and Structures 23 (3) (2007) 391–411. doi:https://doi.org/10.1016/j.jfluidstructs.2006.10.005.

[3] Y. Cao, J. Huang, J. Yin, Numerical simulation of three-dimensional ice accretion on an aircraft wing, International Journal of Heat and Mass Transfer 92 (2016) 34–54. doi:https://doi.org/10.1016/j.ijheatmasstransfer.2015.08.027.

[4] C. Othmer, Adjoint methods for car aerodynamics, Journal of Mathematics in Industry 4 (1) (2014) 6. doi:10.1186/2190-5983-4-6.

[5] F. Mariani, C. Poggiani, F. Risi, L. Scappaticci, Formula-sae racing car: Experimental and numerical analysis of the external aerodynamics, Energy Procedia 81 (2015) 1013–1029, 69th Conference of the Italian Thermal Engineering Association, ATI 2014. doi:https://doi.org/10.1016/j.egypro.2015.12.111.

[6] A. Guerrero, R. Castilla, Aerodynamic study of the wake effects on a formula 1 car, Energies 13 (19) (2020). doi:10.3390/en13195183.

[7] R. Rahgozar, A. R. Ahmadi, Y. Sharifi, A simple mathematical model for approximate analysis of tall buildings, Applied Mathematical Modelling 34 (9) (2010) 2437–2451. doi:https://doi.org/10.1016/j.apm.2009.11.009.

[8] H.-T. Thai, B. Uy, M. Khan, Z. Tao, F. Mashiri, Numerical modelling of concrete-filled steel box columns incorporating high strength materials, Journal of Constructional Steel Research 102 (2014) 256–265. doi:https://doi.org/10.1016/j.jcsr.2014.07.014.

[9] N. P. Smith, D. P. Nickerson, E. J. Crampin, P. J. Hunter, Multiscale computational modelling of the heart, Acta Numerica 13 (2004) 371–431. doi:10.1017/S0962492904000200.

[10] A. Quarteroni, A. Veneziani, Analysis of a geometrical multiscale model based on the coupling of ode and pde for blood flow simulations, Multiscale Modeling & Simulation 1 (2) (2003) 173–195. doi:10.1137/S1540345902408482.

[11] A. Quarteroni, A. Manzoni, C. Vergara, The cardiovascular system: Mathematical modelling, numerical algorithms and clinical applications, Acta Numerica 26 (2017) 365–590. doi:10.1017/S0962492917000046.

[12] E. Vidotto, T. Koch, T. Köppl, R. Helmig, B. Wohlmuth, Hybrid models for simulating blood flow in microvascular networks, Multiscale Modeling & Simulation 17 (3) (2019) 1076–1102. doi:10.1137/18M1228712.

[13] O. Angulo, J. López-Marcos, Numerical schemes for size-structured population equations, Mathematical Biosciences 157 (1) (1999) 169–188. doi:https://doi.org/10.1016/S0025-5564(98)10081-0.

[14] F. Brauer, C. Castillo-Chavez, C. Castillo-Chavez, Mathematical models in population biology and epidemiology, Vol. 2, Springer, 2012.

[15] M. Kot, Elements of Mathematical Ecology, Cambridge University Press, 2001.

[16] J. Medlock, M. Kot, Spreading disease: integro-differential equations old and new, Mathematical Biosciences 184 (2) (2003) 201–222. doi:https://doi.org/10.1016/S0025-5564(03)00041-5.

[17] M. Medrek, Z. Pastuszak, Numerical simulation of the novel coronavirus spreading, Expert Systems with Applications 166 (2021) 114109. doi:https://doi.org/10.1016/j.eswa.2020.114109.

[18] M. Corti, F. Bonizzoni, P. F. Antonietti, Structure preserving polytopal discontinuous galerkin methods for the numerical modeling of neurodegenerative diseases, Journal of Scientific Computing 100 (2) (2024) 39. doi:10.1007/s10915-024-02581-7.

[19] M. Orme, M. Chaplain, A mathematical model of vascular tumour growth and invasion, Mathematical and Computer Modelling 23 (10) (1996) 43–60. doi:https://doi.org/10.1016/0895-7177(96)00053-2.

[20] S. Wise, J. Lowengrub, H. Frieboes, V. Cristini, Three-dimensional multispecies nonlinear tumor growth—i: Model and numerical method, Journal of Theoretical Biology 253 (3) (2008) 524–543. doi:https://doi.org/10.1016/j.jtbi.2008.03.027.

[21] Mathematical model and its fast numerical method for the tumor growth.

[22] M. Fritz, P. K. Jha, T. Köppl, J. T. Oden, A. Wagner, B. Wohlmuth, Modeling and simulation of vascular tumors embedded in evolving capillary networks, Computer Methods in Applied Mechanics and Engineering 384 (2021) 113975. doi:https://doi.org/10.1016/j.cma.2021.113975.

[23] Q. Peng, F. J. Vermolen, D. Weihs, Physical confinement and cell proximity increase cell migration rates and invasiveness: A mathematical model of cancer cell invasion through flexible channels, Journal of the Mechanical Behavior of Biomedical Materials 142 (2023) 105843. doi:https://doi.org/10.1016/j.jmbbm.2023.105843.

[24] A. J. Gadd, A split explicit integration scheme for numerical weather prediction, Quarterly Journal of the Royal Meteorological Society 104 (441) (1978) 569–582.

[25] S. Reich, Linearly implicit time stepping methods for numerical weather pre-

diction, BIT Numerical Mathematics 46 (3) (2006) 607–616. doi:10.1007/s10543-006-0065-0.

[26] J. Coiffier, Fundamentals of Numerical Weather Prediction, Cambridge University Press, 2011.

[27] E. H. Müller, R. Scheichl, Massively parallel solvers for elliptic partial differential equations in numerical weather and climate prediction, Quarterly Journal of the Royal Meteorological Society 140 (685) (2014) 2608–2624. doi:10.1002/qj.2327.

[28] C. Cotter, J. Shipton, Mixed finite elements for numerical weather prediction, Journal of Computational Physics 231 (21) (2012) 7076–7091. doi:https://doi.org/10.1016/j.jcp.2012.05.020.

[29] G. Mengaldo, A. Wyszogrodzki, M. Diamantakis, S.-J. Lock, F. X. Giraldo, N. P. Wedi, Current and emerging time-integration strategies in global numerical weather and climate prediction, Archives of Computational Methods in Engineering 26 (3) (2019) 663–684. doi:https://doi.org/10.1007/s11831-018-9261-8.

[30] P. Moczo, J. Kristek, P. Pazak, M. Balazovjech, M. Galis, The finite-difference and finite-element modeling of seismic wave propagation and earthquake motion, Acta Physica Slovaca 57 (2) (2007) 177–406.

[31] E. T. Flouri, N. Kalligeris, G. Alexandrakis, N. A. Kampanis, C. E. Synolakis, Application of a finite difference computational model to the simulation of earthquake generated tsunamis, Applied Numerical Mathematics 67 (2013) 111–125, nUMAN 2010. doi:https://doi.org/10.1016/j.apnum.2011.06.003.

[32] J. Behrens, F. Dias, New computational methods in tsunami science, Philosophical Transactions of the Royal Society A 373 (2015) 20140382. doi:10.1098/rsta.2014.0382.

[33] S. P. Pudasaini, M. Krautblatter, The mechanics of landslide mobility

with erosion, Nature Communications 12 (1) (2021) 6793. doi:10.1038/s41467-021-26959-5.

[34] R. Boucekkine, O. Licandro, C. Paul, Differential-difference equations in economics: On the numerical solution of vintage capital growth models, Journal of Economic Dynamics and Control 21 (2) (1997) 347–362. doi:https://doi.org/10.1016/S0165-1889(96)00935-9.

[35] S. M. Nuugulu, F. Gideon, K. C. Patidar, A robust numerical scheme for a time-fractional black-scholes partial differential equation describing stock exchange dynamics, Chaos, Solitons & Fractals 145 (2021) 110753. doi:https://doi.org/10.1016/j.chaos.2021.110753.

[36] S. D. Odintsov, V. K. Oikonomou, I. Giannakoudi, F. P. Fronimos, E. C. Lymperiadou, Recent advances in inflation, Symmetry 15 (2023) 1701. doi:10.3390/sym15091701.

[37] C. J. van Duijn, L. A. Peletier, I. S. Pop, A new class of entropy solutions of the buckley–leverett equation, SIAM Journal on Mathematical Analysis 39 (2) (2007) 507–536. doi:10.1137/05064518X.

[38] D. Ingham, A. Bejan, E. Mamut, I. Pop, Emerging Technologies and Techniques in Porous Media, NATO Science Series II: Mathematics, Physics and Chemistry, Springer Netherlands, 2012.

[39] B. Jha, R. Juanes, Coupled multiphase flow and poromechanics: A computational model of pore pressure effects on fault slip and earthquake triggering, Water Resources Research 50 (5) (2014) 3776–3808. doi:10.1002/2013WR015175.

[40] R. Zárate-Miñano, M. Anghel, F. Milano, Continuous wind speed models based on stochastic differential equations, Applied Energy 104 (2013) 42–49. doi:https://doi.org/10.1016/j.apenergy.2012.10.064.

[41] Y. Zeng, L. Zhang, T. Xu, H. Dong, Building and analysis of hydro turbine dynamic model with elastic water column, in: 2010 Asia-Pacific Power and

Energy Engineering Conference, 2010, pp. 1–5. doi:10.1109/APPEEC.2010.5449286.

[42] K. Wada, C. A. Norman, Numerical models of the multiphase interstellar matter with stellar energy feedback on a galactic scale, The Astrophysical Journal 547 (1) (2001) 172. doi:10.1086/318344.

[43] C. Bellos, V. Hrissanthou, Numerical simulation of morphological changes in rivers and reservoirs, Computers & Mathematics with Applications 45 (1) (2003) 453–467. doi:https://doi.org/10.1016/S0898-1221(03)80030-5.

[44] E. Nikinmaa, R. Sievänen, T. Hölttä, Dynamics of leaf gas exchange, xylem and phloem transport, water potential and carbohydrate concentration in a realistic 3-d model tree crown, Annals of Botany 114 (4) (2014) 653–666. arXiv:https://academic.oup.com/aob/article-pdf/114/4/653/16992611/mcu068.pdf, doi:10.1093/aob/mcu068.

[45] V. Agranat, V. Perminov, Mathematical modeling of wildland fire initiation and spread, Environmental Modelling & Software 125 (2020) 104640. doi:https://doi.org/10.1016/j.envsoft.2020.104640.

[46] M. Sanz-Ramos, E. Bladé, P. Oller, G. Furdada, Numerical modelling of dense snow avalanches with a well-balanced scheme based on the 2d shallow water equations, Journal of Glaciology 69 (278) (2023) 1646–1662. doi:10.1017/jog.2023.48.

[47] T. Wey, C. Li, Numerical simulation of shuttle ascent transonic flow using an unstructured-grid approach, Computers & Structures 39 (1) (1991) 207–218. doi:https://doi.org/10.1016/0045-7949(91)90088-4.

[48] M. Sefidgar, M. Soltani, K. Raahemifar, M. Sadeghi, H. Bazmara, M. Bazargan, M. Mousavi Naeenian, Numerical modeling of drug delivery in a dynamic solid tumor microvasculature, Microvascular Research 99 (2015) 43–56. doi:https://doi.org/10.1016/j.mvr.2015.02.007.

[49] V. Taralova, O. Iliev, Y. Efendiev, Derivation and numerical validation of a homogenized isothermal li-ion battery model, Journal of Engineering Mathematics 101 (1) (2016) 1–27. doi:10.1007/s10665-015-9842-6.

[50] R. Alchikh, S. Khuri, Numerical solution of a fractional differential equation arising in optics, Optik 208 (2020) 163911. doi:https://doi.org/10.1016/j.ijleo.2019.163911.

[51] J. A. Rojas-Quintero, J. Villalobos-Chin, V. Santibanez, Optimal control of robotic systems using finite elements for time integration of covariant control equations, IEEE Access 9 (2021) 104980–105001. doi:10.1109/ACCESS.2021.3099131.

[52] A. P. Yunus, K. S. Sajinkumar, G. Gopinath, S. S. Subramanian, S. Kaushal, J. Thanveer, A. L. Achu, S. M. U. Islam, A. Ishan, V. K. Krishnapriya, A. Rajaneesh, M. Dewrari, S. Dixit, S. Singh, P. Srivastava, T. Oommen, N. Nedumpallile-Vasu, S. Sen, A. C. Narayana, V. Ambili, G. S. Pradeep, S. L. Kuriakose, Chronicle of destruction: the wayanad landslide of july 30, 2024, Landslides 22 (6) (2025) 1891–1908. doi:10.1007/s10346-025-02494-y.

[53] W. Schiesser, The Numerical Method of Lines: Integration of Partial Differential Equations, Academic Press, 2012.

[54] G. A. Sod, A survey of several finite difference methods for systems of nonlinear hyperbolic conservation laws, Journal of Computational Physics 27 (1) (1978) 1 – 31. doi:http://dx.doi.org/10.1016/0021-9991(78)90023-2.

[55] J. Nordström, M. H. Carpenter, High-order finite difference methods, multidimensional linear problems, and curvilinear coordinates, Journal of Computational Physics 173 (2001) 149–174.

[56] J. C. Strikwerda, Finite Difference Schemes and Partial Differential Equations, Second Edition, Society for Industrial and Applied Mathematics, 2004. doi:10.1137/1.9780898717938.

[57] A. Jameson, W. Schmidt, E. Turkel, Numerical solutions of the Euler equations by finite volume methods using Runge-Kutta time-stepping schemes, AIAA Paper 81-1259 (1981).

[58] R. J. LeVeque, Numerical Methods for Conservation Laws, Birkhäuser Basel, 1990.

[59] F. Bramkamp, P. Lamby, S. Müller, An adaptive multiscale finite volume solver for unsteady and steady state flow computations, Journal of Computational Physics 197 (2004) 460–490.

[60] T. Barth, M. Ohlberger, Finite volume methods: foundation and analysis, Encyclopedia of Computational Mechanics (2004).

[61] M. Dumbser, C. Enaux, E. F. Toro, Finite volume schemes of very high order of accuracy for stiff hyperbolic balance laws, Journal of Computational Physics 227 (8) (2008) 3971–4001. doi:https://doi.org/10.1016/j.jcp.2007.12.005.

[62] S. C. Brenner, L. R. Scott, The Mathematical Theory of Finite Element Methods, Vol. 15 of Texts in Applied Mathematics, Springer-Verlag, New York, 1994.

[63] M. Ainsworth, J. T. Oden, A posteriori error estimation in Finite Element analysis, Computer Methods in Applied Mechanics and Engineering 142 (1997) 1–88.

[64] E. Süli, P. Houston, Adaptive Finite Element approximations of hyperbolic problems, in: T. J. Barth, H. Deconinck (Eds.), Error Estimation and Adaptive Discretization Methods in Computational Fluid Dynamics, Vol. 25 of Lecture Notes in Computational Science and Engineering, Springer Verlag, 2002, pp. 269–344.

[65] W. Reed, T. Hill, Triangular mesh methods for the neutron transport equation, Tech. rep., Los Alamos Scientific Laboratory (1973).

[66] B. Cockburn, C.-W. Shu, The Runge-Kutta local projection p^1-discontinuous Galerkin finite element method for scalar conservation laws, RAIRO Mathematical modelling and numerical analysis 25 (1991) 337–361.

[67] B. Cockburn, C.-W. Shu, TVB Runge-Kutta local projection discontinuous Galerkin finite element method for conservation laws II: General framework, Mathematics of Computation 52 (1989) 411–435.

[68] C. M. Klaij, J. J. W. van der Vegt, H. van der Ven, Space-time discontinuous Galerkin method for the compressible Navier-Stokes equations, Journal of Computational Physics 217 (2) (2006) 589–611. doi:10.1016/j.jcp.2006.01.018.

[69] L. Pesch, J. J. W. van der Vegt, A space-time discontinuous Galerkin Finite-Element discretization of the Euler equations using entropy variables, in: P. Wesseling, E. Oñate, J. Périaux (Eds.), ECCOMAS CFD 2006, European Conference on Computational Fluid Dynamics, TU Delft, The Netherlands, 2006.

[70] C. Klaij, M. van Raalte, H. van der Ven, J. J. W. van der Vegt, h-multigrid for space-time discontinuous Galerkin discretizations of the compressible Navier–Stokes equations, Journal of Computational Physics 227 (2007) 1024–1045.

[71] S. Rhebergen, B. Cockburn, J. van der Vegt, A space-time discontinuous Galerkin method for the incompressible Navier-Stokes equations, Journal of Computational Physics 233 (2013) 339–358.

[72] A. Hiltebrand, S. Mishra, Efficient preconditioners for a shock capturing space-time discontinuous Galerkin method for systems of conservation laws, Tech. Rep. 2014-04, Seminar for Applied Mathematics, ETH Zürich (2014).

[73] S. M. Findeisen, A parallel and adaptive space-time method for Maxwell's equations, Ph.D. thesis, Karlsruhe Institute of Technology (KIT) (jul 2016).

[74] M. Neumüller, O. Steinbach, Refinement of flexible space–time finite element meshes and discontinuous galerkin methods, Computing and Visualization in Science 14 (5) (2011) 189–205. doi:10.1007/s00791-012-0174-z.

[75] E. Hairer, S. P. Norsett, G. Wanner, Solving ordinary differential equations I, Springer Series in Computational Mathematics, 1987.

[76] E. Hairer, G. Wanner, Solving ordinary differential equations II, Springer Series in Computational Mathematics, 1991.

[77] M. Carpenter, C. Kennedy, Fourth-order $2N$-storage Runge-Kutta schemes, Tech. rep., NASA Langley Research Center (1994).

[78] C. A. Kennedy, M. H. Carpenter, Additive Runge-Kutta schemes for convection-diffusion-reaction equations, Applied Numerical Mathematics 44 (2003) 139–181.

[79] C. A. Kennedy, M. H. Carpenter, Diagonally implicit Runge–Kutta methods for stiff ODEs, Applied Numerical Mathematics 146 (2019) 221–244.

[80] R. Alexander, Diagonally implicit Runge-Kutta methods for stiff O.D.E.'s., SIAM Journal of Numerical Analysis 14 (1977) 1006–1021.

[81] R. Burden, J. Faires, Numerical Analysis, Cengage Learning, 2010.

[82] R. Chan, A. Tsai, On explicit two-derivative Runge-Kutta methods, Numerical Algorithms 53 (2010) 171–194.

[83] S. Gottlieb, Z. J. Grant, J. Hu, R. Shu, High order strong stability preserving multiderivative implicit and IMEX Runge–Kutta methods with asymptotic preserving properties, SIAM Journal on Numerical Analysis 60 (1) (2022) 423–449.

[84] R. L. Brown, Multi-derivative numerical methods for the solution of stiff ordinary differential equations, Technical Report UIUCDCS-R-74-672, Department of Computer Science, University of Illinois (1974).

[85] D. Seal, Y. Güçlü, A. Christlieb, High-order multiderivative time integrators for hyperbolic conservation laws, Journal of Scientific Computing 60 (2014) 101–140.

[86] A. J. Christlieb, S. Gottlieb, Z. J. Grant, D. C. Seal, Explicit strong stability preserving multistage two-derivative time-stepping schemes, Journal of Scientific Computing 68 (2016) 914–942.

[87] D. Yakubu, A. Kwami, Implicit two-derivative runge–kutta collocation methods for systems of initial value problems, Journal of the Nigerian Mathematical Society 34 (2015) 128–142.

[88] A. Jaust, J. Schütz, D. C. Seal, Implicit multistage two-derivative discontinuous Galerkin schemes for viscous conservation laws, Journal of Scientific Computing 69 (2016) 866–891.

[89] Z. Grant, S. Gottlieb, D. C. Seal, A strong stability preserving analysis for explicit multistage two-derivative time-stepping schemes based on Taylor series conditions, Communications on Applied Mathematics and Computation 1 (1) (2019) 21–59.

[90] I. B. Aiguobasimwin, R. I. Okuonghae, A class of two-derivative two-step Runge-Kutta methods for non-stiff ODEs, Journal of Applied Mathematics (2019) Art. ID 2459809, 9.

[91] J. Schütz, D. Seal, A. Jaust, Implicit multiderivative collocation solvers for linear partial differential equations with discontinuous Galerkin spatial discretizations, Journal of Scientific Computing 73 (2017) 1145–1163.

[92] J. Schütz, D. C. Seal, J. Zeifang, Parallel-in-time high-order multiderivative IMEX solvers, Journal of Scientific Computing 90 (54) (2022) 1–33.

[93] F. Biscani, D. Izzo, Revisiting high-order taylor methods for astrodynamics and celestial mechanics, Monthly Notices of the Royal Astronomical Society 504 (2) (2021) 2614–2628. doi:10.1093/mnras/stab1032.

[94] J. Zeifang, A. Thenery Manikantan, J. Schütz, Time parallelism and Newton-adaptivity of the two-derivative deferred correction discontinuous Galerkin method, Applied Mathematics and Computation 457 (2023) 128198.

[95] A. Thenery Manikantan, J. Schütz, Multi-step Hermite-Birkhoff predictor-corrector schemes, Applied Numerical Mathematics 205 (2024) 281–295. doi:https://doi.org/10.1016/j.apnum.2024.07.011.

[96] A. Thenery Manikantan, J. Zeifang, J. Schütz, On the stability of two-derivative time discretizations, UHasselt Computational Mathematics Preprint, https://www.uhasselt.be/media/4gbfm0hf/up2302.pdf (2025).

[97] J. C. Butcher, General linear methods, Acta Numerica 15 (2006) 157–256.

[98] Z. Jackiewicz, General linear methods for ordinary differential equations, John Wiley & Sons, 2009.

[99] A. Jaust, J. Schütz, General linear methods for time-dependent PDEs, in: Submitted to Proceedings of HYP2016.

[100] G. Califano, G. Izzo, Z. Jackiewicz, Strong stability preserving general linear methods with Runge–Kutta stability, Journal of Scientific Computing 76 (2) (2018) 943–968. doi:10.1007/s10915-018-0646-5.

[101] P. Vos, C. Eskilsson, A. Bolis, S. Chun, R. M. Kirby, S. J. Sherwin, A generic framework for time-stepping partial differential equations (PDEs): general linear methods, object-oriented implementation and application to fluid problems, International Journal of Computational Fluid Dynamics 25 (3) (2011) 107–125.

[102] H. Zhang, A. Sandu, S. Blaise, Partitioned and implicit–explicit general linear methods for ordinary differential equations, Journal of Scientific Computing 61 (1) (2014) 119–144.

[103] R. Jeltsch, A0-stability and stiff stability of Brown's multistep multiderivative methods, Numerische Mathematik 32 (2) (1979) 167–181. doi:10.1007/BF01404873.

[104] A. Moradi, J. Farzi, A. Abdi, Strong stability preserving second derivative general linear methods, Journal of Scientific Computing 81 (1) (2019) 392–435.

[105] R. I. Okuonghae, M. N. O. Ikhile, L(α)-stable multi-derivative GLM, Journal of Algorithms & Computational Technology 9 (4) (2015) 339–376. doi:10.1260/1748-3018.9.4.339.

[106] A. Abdi, D. Conte, Implementation of second derivative general linear methods, Calcolo 57 (3) (2020) Paper No. 20, 29.

[107] M. M. Khalsaraei, A. Shokri, M. Molayi, The new class of multistep multi-derivative hybrid methods for the numerical solution of chemical stiff systems of first order IVPs, Journal of Mathematical Chemistry 58 (9) (2020) 1987–2012. doi:10.1007/s10910-020-01160-z.

[108] A. Moradi, A. Abdi, J. Farzi, Strong stability preserving second derivative general linear methods with Runge–Kutta stability, Journal of Scientific Computing 85 (1) (2020) 1.

[109] A. Moradi, A. Abdi, G. Hojjati, Strong stability preserving implicit and implicit–explicit second derivative general linear methods with RK stability, Computational and Applied Mathematics 41 (4) (2022) 135.

[110] B. Cockburn, S. Y. Lin, C.-W. Shu, TVB Runge-Kutta local projection discontinuous Galerkin finite element method for conservation laws III: One dimensional systems, Journal of Computational Physics 84 (1989) 90–113.

[111] B. Cockburn, S. Hou, C.-W. Shu, The Runge–Kutta local projection discontinuous Galerkin finite element method for conservation laws IV: The multidimensional case, Mathematics of Computation 54 (1990) 545–581.

[112] B. Cockburn, C.-W. Shu, The Runge-Kutta discontinuous Galerkin Method for conservation laws V: Multidimensional Systems, Mathematics of Computation 141 (1998) 199–224.

[113] B. Cockburn, G. E. Karniadakis, C.-W. Shu, discontinuous Galerkin Methods, Springer, 2000.

[114] D. Di Pietro, A. Ern, Mathematical aspects of discontinuous Galerkin Methods, Vol. 69, Springer Science & Business Media, 2011.

[115] D. A. Kopriva, Implementing spectral methods for partial differential equations: Algorithms for scientists and engineers, Springer Science & Business Media, 2009.

[116] N. Krais, A. Beck, T. Bolemann, H. Frank, D. Flad, G. Gassner, F. Hindenlang, M. Hoffmann, T. Kuhn, M. Sonntag, et al., FLEXI: A high order discontinuous Galerkin framework for hyperbolic–parabolic conservation laws, Computers & Mathematics with Applications 81 (2021) 186–219.

[117] M. Kurz, D. Kempf, M. P. Blind, P. Kopper, P. Offenhäuser, A. Schwarz, S. Starr, J. Keim, A. Beck, Galæxi: Solving complex compressible flows with high-order discontinuous galerkin methods on accelerator-based systems, Computer Physics Communications 306 (2025) 109388, code available at https://github.com/flexi-framework/galaexi. doi:https://doi.org/10.1016/j.cpc.2024.109388.

[118] P. Kopper, A. Schwarz, S. M. Copplestone, P. Ortwein, S. Staudacher, A. Beck, A framework for high-fidelity particle tracking on massively parallel systems, Computer Physics Communications (2023) 108762doi:10.1016/j.cpc.2023.108762.

[119] F. Hindenlang, G. Gassner, C. Altmann, A. Beck, M. Staudenmaier, C.-D. Munz, Explicit discontinuous Galerkin methods for unsteady problems, Computers & Fluids 61 (2012) 86–93.

[120] A. Thenery Manikantan, J. Schütz, Two-derivative schemes for low-mach flows, UHasselt Computational Mathematics Preprint, https://www.uhasselt.be/media/hzonmb0o/paper.pdf (2025).

[121] B. van der Pol Jun. and, Lxxxviii. on "relaxation-oscillations", The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science 2 (11) (1926) 978–992. doi:10.1080/14786442608564127.

[122] G. Söderlind, L. Jay, M. Calvo, Stiffness 1952–2012: Sixty years in search of a definition, BIT Numerical Mathematics 55 (2) (2015) 531–558. doi:10.1007/s10543-014-0503-3.

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

[124] R. Hartmann, F. Bassi, I. Bosnyakov, L. Botti, A. Colombo, A. Crivellini, M. Franciolini, T. Leicht, E. Martin, F. Massa, et al., Implicit methods, in: TILDA: Towards Industrial LES/DNS in Aeronautics, Springer, 2021, pp. 11–59.

[125] J. Schütz, D. Seal, An asymptotic preserving semi-implicit multiderivative solver, Applied Numerical Mathematics 160 (2021) 84–101.

[126] P. Degond, M. Tang, All speed scheme for the low Mach number limit of the isentropic Euler equation, Communications in Computational Physics 10 (2011) 1–31.

[127] J. Schütz, S. Noelle, Flux splitting for stiff equations: A notion on stability, Journal of Scientific Computing 64 (2) (2015) 522–540.

[128] U. M. Ascher, S. Ruuth, B. Wetton, Implicit-Explicit methods for time-dependent partial differential equations, SIAM Journal on Numerical Analysis 32 (1995) 797–823.

[129] U. M. Ascher, S. Ruuth, R. Spiteri, Implicit-explicit Runge-Kutta methods for time-dependent partial differential equations, Applied Numerical Mathematics 25 (1997) 151–167.

[130] S. Boscarino, Error analysis of IMEX Runge-Kutta methods derived from differential-algebraic systems, SIAM Journal on Numerical Analysis 45 (2007) 1600–1621.

[131] W. Hundsdorfer, S.-J. Ruuth, IMEX extensions of linear multistep methods with general monotonicity and boundedness properties, Journal of Computational Physics 225 (2) (2007) 2016–2042.

[132] L. Pareschi, G. Russo, Implicit-explicit Runge-Kutta schemes and applications to hyperbolic systems with relaxation, Journal of Scientific Computing 25 (1-2) (2005) 129–155.

[133] K. Kaiser, J. Schütz, Asymptotic error analysis of an IMEX Runge–Kutta method, Journal of Computational and Applied Mathematics 343 (2018) 139–154.

[134] D. Wang, S. J. Ruuth, Variable step-size implicit-explicit linear multistep methods for time-dependent partial differential equations, Journal of Computational Mathematics 26 (6) (2008) 838–855, full publication date: November 2008.

[135] S. Boscarino, G. Russo, On a class of uniformly accurate IMEX Runge-Kutta schemes and applications to hyperbolic systems with relaxation, SIAM Journal on Scientific Computing 31 (3) (2009) 1926–1945.

[136] A. J. Dittmann, High-order multiderivative IMEX schemes, Applied Numerical Mathematics 160 (2021) 205 – 216.

[137] E. Theodosiou, J. Schütz, D. Seal, An explicitness-preserving imex-split multiderivative method, Computers & Mathematics with Applications 158 (2024) 139–149. doi:https://doi.org/10.1016/j.camwa.2023.12.040.

[138] J. Stiller, Stable semi-implicit sdc methods for conservation laws, Journal of Scientific Computing 103 (2) (2025) 37. doi:10.1007/s10915-025-02857-6.

[139] S. Noelle, R. Klein, J. Schütz, H. Zakerzadeh, RS-IMEX schemes: derivation and asymptotic stability(in preparation) (2015).

[140] J. Zeifang, J. Schütz, K. Kaiser, A. Beck, M. Lukáčová-Medvid'ová, S. Noelle, A novel full-Euler low Mach number IMEX splitting, Communications in Computational Physics 27 (2020) 292–320.

[141] J. Zeifang, A. Beck, A low Mach number IMEX flux splitting for the level set ghost fluid method, Communications on Applied Mathematics and Computation (2021) 1–29.

[142] S. Boscarino, G. Russo, Asymptotic preserving methods for quasilinear hyperbolic systems with stiff relaxation: a review, SeMA Journal. Boletin de la Sociedad Española de Matemática Aplicada 81 (1) (2024) 3–49. doi:10.1007/s40324-024-00351-x.

[143] S. Klainerman, A. Majda, Singular limits of quasilinear hyperbolic systems with large parameters and the incompressible limit of compressible fluids, Communications on Pure and Applied Mathematics 34 (1981) 481–524.

[144] S. Jin, Efficient asymptotic-preserving (AP) schemes for some multiscale kinetic equations, SIAM Journal on Scientific Computing 21 (1999) 441–454.

[145] P. Degond, S. Jin, J.-G. Liu, Mach-number uniform asymptotic-preserving gauge schemes for compressible flows, Bulletin-Institute of Mathematics Academia Sinica (New Series) 2 (4) (2007) 851–892.

[146] S. Jin, Asymptotic preserving (AP) schemes for multiscale kinetic and hyperbolic equations: A review, Rivista di Matematica della Universita Parma 3 (2012) 177–216.

[147] F. Cordier, P. Degond, A. Kumbaro, An asymptotic-preserving all-speed scheme for the Euler and Navier-Stokes equations, Journal of Computational Physics 231 (2012) 5685–5704.

[148] S. Noelle, G. Bispen, K. Arun, M. Lukáčová-Medvid'ová, C.-D. Munz, A weakly asymptotic preserving low Mach number scheme for the Euler equations of gas dynamics, SIAM Journal on Scientific Computing 36 (2014) B989–B1024.

[149] P. Degond, A. Lozinski, J. Narski, C. Negulescu, An asymptotic-preserving method for highly anisotropic elliptic equations based on a micro-macro decomposition, Journal of Computational Physics 231 (2012) 2724–2740.

[150] J. Haack, S. Jin, J.-G. Liu, An all-speed asymptotic-preserving method for the isentropic Euler and Navier-Stokes equations, Communications in Computational Physics 12 (2012) 955–980.

[151] G. Bispen, M. Lukáčová-Medvid'ová, L. Yelash, Asymptotic preserving IMEX finite volume schemes for low Mach number Euler equations with gravitation, Journal of Computational Physics 335 (2017) 222–248.

[152] J. Hu, S. Jin, Q. Li, Asymptotic-preserving schemes for multiscale hyperbolic and kinetic equations, Handbook of Numerical Analysis 18 (2017) 103–129.

[153] A. Ishimwe, E. Deleersnijder, V. Legat, J. Lambrechts, A split-explicit second order Runge–Kutta method for solving 3D hydrodynamic equations, Ocean Modeling (186) (2023) 102273.

[154] J. Wensch, O. Knoth, A. Galant, Multirate infinitesimal step methods for atmospheric flow simulation, BIT Numerical Mathematics 49 (2009) 449–473.

[155] M. Günther, A. Sandu, Multirate generalized additive Runge Kutta methods, Numerische Mathematik 133 (2016) 497–524.

[156] V. T. Luan, R. Chinomona, D. R. Reynolds, A new class of high-order methods for multirate differential equations, SIAM Journal on Scientific Computing 42 (2) (2020) A1245–A1268. doi:10.1137/19M125621X.

[157] A. Naumann, J. Wensch, Multirate finite step methods, Numerical Algorithms 81 (4) (2019) 1547–1571. doi:10.1007/s11075-019-00763-1.

[158] A. Sandu, A class of multirate infinitesimal GARK methods, SIAM Journal on Numerical Analysis 57 (5) (2019) 2300–2327. doi:10.1137/18M1205492.

[159] R. Chinomona, D. R. Reynolds, Implicit-explicit multirate infinitesimal GARK methods, SIAM Journal on Scientific Computing 43 (5) (2021) A3082–A3113. doi:10.1137/20M1354349.

[160] A. Abdulle, M. J. Grote, G. Rosilho de Souza, Explicit stabilized multirate method for stiff differential equations, Mathematics of Computation 91 (338) (2022) 2681–2714. doi:10.1090/mcom/3753.

[161] A. C. Fish, D. R. Reynolds, Adaptive time step control for multirate infinitesimal methods, SIAM Journal on Scientific Computing 45 (2) (2023) A958–A984. doi:10.1137/22M1479798.

[162] A. C. Fish, D. R. Reynolds, S. B. Roberts, Implicit-explicit multirate infinitesimal stage-restart methods, Journal of Computational and Applied Mathematics 438 (2024) Paper No. 115534, 23. doi:10.1016/j.cam.2023.115534.

[163] S. Kang, A. Dener, A. Hamilton, H. Zhang, E. M. Constantinescu, R. L. Jacob, Multirate partitioned Runge-Kutta methods for coupled Navier-Stokes equations, Computers & Fluids. An International Journal 264 (2023) Paper No. 105964, 15. doi:10.1016/j.compfluid.2023.105964.

[164] J. Schütz, A. Ishimwe, A. Moradi, A. Thenery Manikantan, A Class of Multirate Multiderivative Schemes, UHasselt Computational Mathematics Preprint, UP-24-03, https://www.uhasselt.be/media/5fopnoey/up2403.pdf (2024).

[165] A. M. Stuart, A. R. Humphries, Model problems in numerical stability theory for initial value problems, SIAM Review 36 (2) (1994) 226–257.

[166] J. C. Butcher, G. Hojjati, Second derivative methods with RK stability, Numerical Algorithms 40 (4) (2005) 415–429.

[167] J. Zeifang, J. Schütz, D. Seal, Stability of implicit multiderivative deferred correction methods, BIT Numerical Mathematics (2022).

[168] A. Dutt, L. Greengard, V. Rokhlin, Spectral deferred correction methods for ordinary differential equations, BIT. Numerical Mathematics 40 (2) (2000) 241–266.

[169] B. W. Ong, R. J. Spiteri, Deferred correction methods for ordinary differential equations, Journal of Scientific Computing 83 (3) (2020) Paper No. 60, 29.

[170] P. Zadunaisky, On the estimation of errors propagated in the numerical integration of ordinary differential equations., Numerische Mathematik 27 (1976/1977) 21–40.

[171] R. Speck, Parallelizing spectral deferred corrections across the method, Computing and Visualization in Science 19 (3) (2018) 75–83. doi:10.1007/s00791-018-0298-x.

[172] A. Christlieb, B. Ong, J.-M. Qiu, Comments on high-order integrators embedded within integral deferred correction methods, Communications in Applied Mathematics and Computational Science 4 (1) (2009) 27 – 56. doi:10.2140/camcos.2009.4.27.

[173] A. Christlieb, B. Ong, J.-M. Qiu, Integral deferred correction methods constructed with high order Runge-Kutta integrators, Mathematics of Computation 79 (270) (2010) 761–783.

[174] K. Böhmer, P. W. Hemker, H. J. Stetter, The Defect Correction Approach, Springer Vienna, Vienna, 1984, pp. 1–32. doi:10.1007/978-3-7091-7023-6_1.

[175] M. Minion, Semi-implicit spectral deferred correction methods for ordinary differential equations, Communications in Mathematical Sciences 1 (3) (2003) 471–500.

[176] J. Huang, J. Jia, M. Minion, Accelerating the convergence of spectral deferred correction methods, Journal of Computational Physics 214 (2) (2006) 633–656. doi:10.1016/j.jcp.2005.10.004.

[177] Y. Xia, Y. Xu, C.-W. Shu, Efficient time discretization for local discontinuous galerkin methods (2007). doi:10.3934/dcdsb.2007.8.677.

[178] M. Weiser, Faster sdc convergence on non-equidistant grids by dirk sweeps, BIT Numerical Mathematics 55 (4) (2015) 1219–1241. doi:10.1007/s10543-014-0540-y.

[179] R. Speck, D. Ruprecht, M. Emmett, M. Minion, M. Bolten, R. Krause, A multi-level spectral deferred correction method, BIT Numerical Mathematics 55 (3) (2015) 843–867. doi:10.1007/s10543-014-0517-x.

[180] M. F. Causley, D. C. Seal, On the convergence of spectral deferred correction methods, Communications in Applied Mathematics and Computational Science 14 (1) (2019) 33–64.

[181] J. Zeifang, J. Schütz, Implicit two-derivative deferred correction time discretization for the discontinuous Galerkin method, Journal of Computational Physics 464 (2022) 111353.

[182] M. J. Gander, S. Vandewalle, Analysis of the parareal time-parallel time-integration method, SIAM Journal on Scientific Computing 29 (2) (2007) 556–578. doi:10.1137/05064607X.

[183] X. Dai, Y. Maday, Stable parareal in time method for first- and second-order hyperbolic systems, SIAM Journal on Scientific Computing 35 (1) (2013) A52–A78. doi:10.1137/110861002.

[184] G. Gurrala, A. Dimitrovski, S. Pannala, S. Simunovic, M. Starke, Parareal in time for fast power system dynamic simulations, IEEE Transactions on Power Systems 31 (3) (2016) 1820–1830. doi:10.1109/TPWRS.2015.2434833.

[185] T. Lunet, J. Bodart, S. Gratton, X. Vasseur, Time-parallel simulation of the decay of homogeneous turbulence using parareal with spatial coarsening, Computing and Visualization in Science 19 (1) (2018) 31–44.

[186] M. L. Minion, A hybrid parareal spectral deferred corrections method, Communications in Applied Mathematics and Computational Science 5 (2) (2010) 265–301. doi:10.2140/camcos.2010.5.265.

[187] A. J. Christlieb, C. B. Macdonald, B. W. Ong, Parallel high-order integrators, SIAM Journal on Scientific Computing 32 (2) (2010) 818–835.

[188] A. Christlieb, B. Ong, Implicit parallel time integrators, Journal of Scientific Computing 49 (2) (2011) 167–179.

[189] A. J. Christlieb, C. B. Macdonald, B. W. Ong, R. J. Spiteri, Revisionist integral deferred correction with adaptive step-size control, Communications in Applied Mathematics and Computational Science 10 (1) (2015) 1–25.

[190] Y. Saad, M. H. Schultz, GMRES: A generalized minimal residual algorithm for solving non-symmetric linear systems, SIAM Journal on Scientific and Statistical Computing 7 (1986) 856–869.

[191] A. Soulaimani, N. B. Salah, Y. Saad, Enhanced GMRES acceleration techniques for some CFD problems, International Journal of Computational Fluid Dynamics 16 (1) (2002) 1–20.

[192] P.-O. Persson, J. Peraire, Newton-gmres preconditioning for discontinuous Galerkin discretizations of the navier-stokes equations, SIAM Journal on Scientific Computing 30 (6) (2008) 2709–2733.

[193] D. Blom, P. Birken, H. Bijl, F. Kessels, A. Meister, A. van Zuilen, A comparison of Rosenbrock and ESDIRK methods combined with iterative solvers for unsteady compressible flows, Advances in Computational Mathematics 42 (2016) 1401–1426.

[194] P. Birken, Numerical Methods for Unsteady Compressible Flow Problems, Numerical Analysis and Scientific Computing, Chapman & Hall, 2021.

[195] S. Vangelatos, On the efficiency of implicit discontinuous Galerkin spectral element methods for the unsteady compressible Navier-Stokes equations, Ph.D. thesis, University of Stuttgart (2019).

[196] Y. Pan, Z.-G. Yan, J. Peiró, S. Sherwin, Development of a balanced adaptive time-stepping strategy based on an implicit JFNK-DG compressible flow solver, Communications on Applied Mathematics and Computation (2021).

[197] N. Qin, D. K. Ludlow, S. T. Shaw, A matrix-free preconditioned Newton/GMRES method for unsteady Navier-Stokes solutions, International Journal for Numerical Methods in Fluids 33 (2000) 223–248.

[198] M. Franciolini, A. Crivellini, A. Nigro, On the efficiency of a matrix-free linearly implicit time integration strategy for high-order discontinuous Galerkin solutions of incompressible turbulent flows, Computers & Fluids 159 (2017) 276–294.

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

[200] G. Čaklović, T. Lunet, S. Götschel, D. Ruprecht, Improving efficiency of parallel across the method spectral deferred corrections, SIAM Journal on Scientific Computing 47 (1) (2025) A430–A453. doi:10.1137/24M1649800.

[201] Y. Liu, C.-W. Shu, M. Zhang, Strong stability preserving property of the deferred correction time discretization, Journal of Computational Mathematics 26 (5) (2008) 633–656, full publication date: September 2008.

[202] H. Carrillo, C. Parés, Compact approximate Taylor methods for systems of conservation laws, Journal of Scientific Computing 80 (3) (2019) 1832–1866.

[203] J. Chouchoulis, J. Schütz, J. Zeifang, Jacobian-free explicit multiderivative Runge–Kutta methods for hyperbolic conservation laws, Journal of Scientific Computing 90 (3) (2022) 96.

[204] J. Chouchoulis, J. Schütz, Jacobian-free implicit mdrk methods for stiff systems of odes, Applied Numerical Mathematics 196 (2024) 45–61. doi:https://doi.org/10.1016/j.apnum.2023.10.007.

Curriculum vitae

Arjun Thenery Manikantan was born on January 31, 1993, in Wayanad, Kerala, India. He completed his schooling at St. Peter's & St. Paul's English School, Meenangadi, and The GreenHills Public School, Sulthan Bathery, graduating in 2012. He received the INSPIRE Fellowship (2012–2017), a prestigious national scholarship supporting undergraduate and postgraduate studies in the Natural and Basic Sciences.

In 2017, he obtained a dual BS-MS degree in Mathematical Sciences, with a minor in Physical Sciences, from the Indian Institute of Science Education and Research (IISER), Thiruvananthapuram, India. After qualifying the JRF(NET)-CSIR, he joined Amrita Vishwa Vidyapeetham, Kollam, as an Assistant Professor in the Department of Mathematics in 2018.

In February 2022, Arjun began his doctoral program in Mathematics at Hasselt University, Belgium, under the supervision of Prof. Dr. Jochen Schütz. During his PhD, he participated in several summer schools, workshops, and international conferences. The main results of his research are presented in this dissertation. His PhD position was funded by the "Bijzonder Onderzoeksfonds" (BOF) at Hasselt University (project no. BOF21KP12).