2026 | Faculty of Sciences

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

Jeremy Chouchoulis

DOCTORAL DISSERTATION

Jacobian-free multiderivative timestepping methods

Promoter:	Prof. Dr Jochen Schütz | UHasselt
Co-promoter:	Prof. Dr Natalie Beenaerts | UHasselt

UHASSELT
KNOWLEDGE IN ACTION

2026 | Faculty of Sciences

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

Jeremy Chouchoulis

DOCTORAL DISSERTATION

Jacobian-free multiderivative timestepping methods

CMK
CENTRE FOR
ENVIRONMENTAL SCIENCES
UHASSELT

DSI
DATA SCIENCE INSTITUTE
UHASSELT

Promoter: Prof. Dr Jochen Schütz | UHasselt

Co-promoter: Prof. Dr Natalie Beenaerts | UHasselt

D/2026/2451/8

Jacobian-free multiderivative timestepping methods

Doctoral dissertation submitted to obtain the degree of Doctor of Sciences: Mathematics.

Jeremy Chouchoulis

Acknowledgments

My six-year doctoral journey at UHasselt was marked by an immense amount of support, and this thesis is a product of that collective encouragement. I am truly grateful for this opportunity to express my thanks to everyone who contributed.

First and foremost, I wish to express my deepest gratitude to my supervisor, Prof. dr. Jochen Schütz, for his unwavering support throughout my Ph.D. trajectory. I am profoundly thankful for your continuous availability, the low threshold for quick discussions, and your invaluable guidance in helping me maintain a clear overview of the project. It was a genuine pleasure to collaborate with you on both research and education. I have a deep appreciation for the hard work you dedicated to conceptualizing, prototyping, and implementing numerical algorithms and for your ability to make complex research understandable in your papers. Your approach to and philosophy on education were also a true source of insight.

A special thank you goes to Prof. dr. Natalie Beenaerts and Prof. dr. Francois Rineau, who allowed me to participate in the Ecotron project. This experience provided me with a highly interdisciplinary environment and broadened my perspective on research through access to the FRC facilities and the ongoing projects in Maasmechelen. Thank you for welcoming me into the team.

I also want to thank dr. Alexander Jaust for his assistance during the initial stages of my Ph.D. by taking the time to answer my coding questions and spending a week with me to help me better understand the hybridized discontinuous Galerkin software. Moreover, I am grateful to Prof. dr. Iuliu Sorin Pop for helping me balance my research and teaching responsibilities, for our teaching collaborations, and for being part of my committee.

Throughout my studies, I had the pleasure of working together with dr. Jonas Zeifang, who co-authored my first paper and helped me to develop an ODE solver in

Matlab, which was a crucial software tool for this thesis. I am also deeply thankful for the fruitful collaboration with dr. Afsaneh Moradi. After reading my first published paper, she contacted me to extend the approach, which ultimately led to a co-authored paper and a minisymposium in Lisbon.

I take this opportunity to thank the FWO (project K174622N) for the financial support that facilitated my Ph.D. project and my scientific visit to the 2022 SciCADE conference in Iceland. I would also like to acknowledge the assistance of the Doctoral School of Sciences & Technology staff members for their administrative support.

My learning environment was filled with many amazing individuals. I had the pleasure of being part of a great education team that was always willing to support one another. Thank you to dr. Maikel Bosschaert, dr. Ruben Henrard, dr. Jeroen Wynen, dr. Doryan Temmerman, Bram Lentjes, Ansfried Janssens, Wilbert den Hertog, Prof. dr. Fred Vermolen, and many others. Thank you to my warm office colleagues, Eleni Theodosiou, dr. Yiorgos Patsios, Hua Sun, Arjun Thenery Manikantan, and Hoang An Tran, for the numerous conversations and coffee breaks. Furthermore, I thank the following colleagues for the pleasant working atmosphere: Manuela, Sohely, Ayesha, Sabia, Roel, Karel, Koondi, Vipul, Peter, Renato, Adam-Christiaan, Lisa, Seppe, Shameer, Nasrin, Xander, Kristof, Hanne, and all the other wonderful people I met at the university.

Finally, I would like to thank my friends and family for their endless support and encouragement. The weekly five-hour commute between Belgium and the Netherlands was challenging, and I am incredibly grateful to my partner's family for always being ready to assist whenever and wherever needed. I am deeply thankful to my parents, who always treated me royally, providing a warm place to stay and spoiling me with delicious food. A special thank you to my sister for her mental support and great conversations and to my brother for our talks after I returned from cycling, which helped ease my mind. A special appreciation also goes to my office cats at home: Trigger, Alexi, and Spiro.

Lastly, I express my everlasting gratitude to my extraordinary partner, Anna. Without your endless patience, support, and countless times of cheering me on, this project would have ended long ago. Thank you for believing in me and keeping me motivated during all the challenging times.

Thank you, everyone, for making this journey possible.

Jeremy Chouchoulis

Contents

Chapter 1

Introduction

Climate change is a reality, and human activity plays a significant role in its progression. We are the primary factor for the rising levels of greenhouse gases and the resulting increase in global temperatures across the atmosphere, oceans, and land [22, SYR-A.1].

Understanding the science and addressing the issue effectively is crucial, though it is a complex challenge given its far-reaching impact on individuals, industries, and the natural environment alike. In this thesis, I aim to contribute from a computational and mathematical perspective, offering new insights and improvements to algorithms that solve equations describing fluid flow, which ultimately assist in the study of climate change. Concretely, my contributions are twofold:

- Making multiderivative multistage time-integrators more amenable as a means for solving conservation laws by taking a Jacobian-free approach;
- Bringing understanding to how stiff equations can negatively impact implicit multiderivative time-integrators through the conditioning of the linearized system, and what can be done to mitigate these adverse effects.

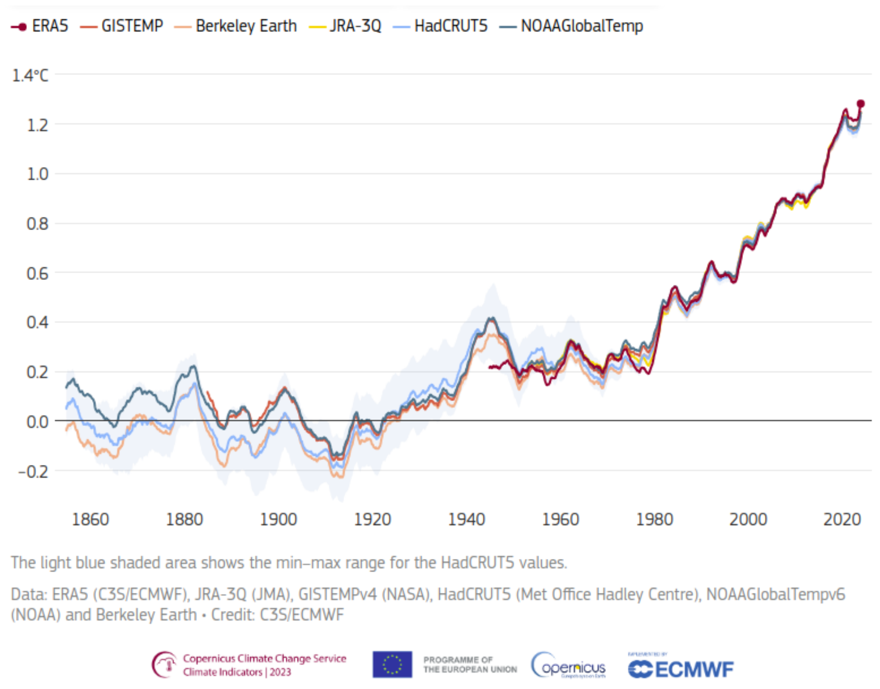
Further introductory details about my specific contributions are provided near the end of this chapter in the *"Jacobian-free"* paragraphs at pages 14 and 16.

Ecotrons Figure 1.1 displays a comparison of six widely-referenced datasets[1] that track the rise in five-year average global temperatures compared to the 1850-1900 pre-industrial average [11]. Global surface air temperatures are estimated to have risen by

[1] Simmons et al. ([45], 2021) discuss the six datasets in more detail, also take a look at the C3S temperature indicator website itself [11].

1.2–1.3°C since pre-industrial times. In Europe, the increase has been about 2.3°C, exceeding the global average by roughly one degree. The latest five-year averages are either the highest or nearly the highest on record, with 2024 the warmest year ever recorded, having an average global temperature of 1.6°C above the pre-industrial average [12].

Increase in global average temperature above 1850-1900 reference period



The light blue shaded area shows the min–max range for the HadCRUT5 values.

Data: ERA5 (C3S/ECMWF), JRA-3Q (JMA), GISTEMPv4 (NASA), HadCRUT5 (Met Office Hadley Centre), NOAAGlobalTempv6 (NOAA) and Berkeley Earth • Credit: C3S/ECMWF

Figure 1.1: Global average near-surface temperature for centered running 60-month periods as an increase above the 1850–1900 average, according to six datasets. Data sources: ERA5 (C3S/ECMWF), JRA-3Q (JMA), GISTEMPv4 (NASA), HadCRUT5 (Met Office Hadley Centre), NOAAGlobalTempv6 (NOAA) and Berkeley Earth. Copyright © 2024 European Centre for Medium-Range Weather Forecasts (ECMWF). This data is published under a Creative Commons Attribution 4.0 International License. `https://creativecommons.org/licenses/by/4.0/`. Disclaimer: ECMWF does not accept any liability whatsoever for any error or omission in the data, their availability, or for any loss or damage arising from their use. Credit: Copernicus Climate Change Service (C3S), ECMWF [11].

These rising temperatures are not merely statistical markers; they drive noticeable consequences for the planet's ecosystems. By altering seasonal water availability and disrupting the pace of crucial biogeochemical cycles for carbon, nitrogen, and water, this warming can shift the fundamental balance of an environment. Such disruptions have alarming implications, including the potential for forests to turn from vital

The UHasselt Ecotron facility

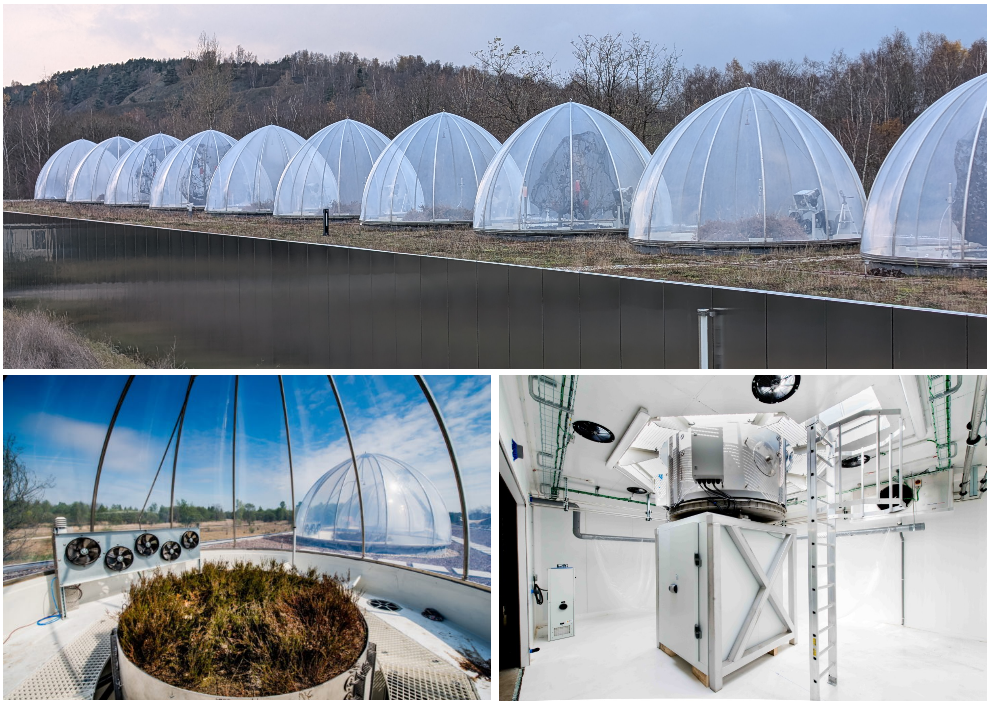


Figure 1.2: Upper picture: an outside view showing a subset of the Ecotron units with the National Park in the background, some units contain heathland, others contain pear trees. Bottom left picture: view inside the dome; on the center left one can see the black fans replicating wind flow on the basis of real-time measured data, a heathland extract of the National Park has been put in a lysimeter (the cylindrical iron tube). Bottom right picture: ground level view of an Ecotron unit; the lysimeter is shown in more detail including cabling for the measuring devices, the black fans in the ceiling control the temperature in the dome. Bottom pictures are courtesy of the UHasselt Field Research Center.

carbon sinks into carbon sources [9], thereby amplifying risks for both natural habitats and the human societies that depend on them.

Significant progress has been made in understanding how climate changes affect ecosystems, see, e.g., [17]. However, despite considerable progress, ecosystem response experiments studying climate change still face substantial hurdles, including the intricate interactions among environmental variables, limited climate treatment ranges and intensities, and a restricted focus on specific responses and interactions [39]. Past climate change experiments often reduced the potential interactions between factors due to technological constraints. Additionally, high equipment costs frequently restrict the number of experimental units, leading to simplified, two-level experiments comparing current conditions with future projections. This constraint is substantial, as the complex and nonlinear nature of ecosystem responses to various global change factors makes it difficult to create reliable models using data limited to two environmental levels. Therefore, experiments aimed at overcoming these obstacles are greatly needed. Due to the intricate processes and variety of organisms involved, a large, well-equipped, and carefully managed infrastructure is essential for studying how environmental changes impact ecosystem function. One proposed solution is through a multitude of *Ecotrons*, which are specialized facilities that replicate ecosystems off-site, allowing scientists to monitor and control environmental conditions with numerous sensors [40].

UHasselt itself, in 2016, has set up an Ecotron facility in the Hoge Kempen National Park [39]. The accommodation hosts twelve tightly controlled units, consisting of macrocosms (soil–canopy columns of 2m in diameter and 1.5m depth, see bottom pictures in Figure 1.2), some of which contain dry heathland extracted from the National park itself. Ranging from air temperature, soil water tension to greenhouse gas concentrations, about 184,000 data points are measured each day, allowing for the study of a large variety of ecosystem interactions. Within this infrastructure, the ecosystem is conceptualized as three primary, interacting compartments: the soil, the vegetation canopy, and the air. A key focus is understanding the exchange of mass and heat between these distinct layers.

Computational mathematics Ecotrons deliver precise information on ecosystem dynamics within a controlled setting. Despite overcoming many limitations typical of climate change studies, they remain bound by the constraints of physical experimentation. Adjusting and prototyping experiment parameters can be costly, time-intensive, and sometimes even unfeasible. Additionally, to manage both cost and practicality, careful prioritization is required, including decisions on sensor resolution and the specific parameters to be monitored. In this regard, supplementing Ecotron studies with digital experiments can be advantageous. The ultimate goal of this work is to deliver reliable and efficient mathematical algorithms that can in a subsequent step be used to, e.g., model evapotranspiration models in an Ecotron.

Achieving this requires modeling and mathematical representation of real natural phenomena. Ecosystems, in particular, are defined by biochemical cycles that drive flows within and between the soil and atmosphere. Predicting how ecosystems will respond to climate change, therefore, requires an in-depth understanding of the flow of water, moisture, and heat throughout the entire system. A significant challenge is that many current models do not sufficiently account for the thin vegetation layer separating the atmosphere and the soil. This canopy critically influences the interaction between these main compartments, as it is permeable to flow and acts as a source and sink for water and carbon. A robust model for the fluxes within the vegetation layer, and a computationally tractable method for coupling it to the soil and air flows, is still missing. The primary obstacle is the intricate geometry of the canopy; resolving the airflow around every individual plant is computationally infeasible. A guiding principle to overcome this is model reduction, where techniques like homogenization are used to derive an *upscaled* model that represents the averaged effects of the complex geometry without simulating it directly. This overarching scientific challenge served as the catalyst for the research presented in this thesis. A fundamental requirement for eventually modeling such a complex, coupled system is the development of high-fidelity simulation techniques for the airflow component, which governs much of the transport phenomena. To this end, the augmented[2] Euler and Navier–Stokes equations are often used to model airflow, which is particularly relevant in an Ecotron setting. In these controlled settings, evapotranspiration plays a significant role, emphasizing the need for, amongst others, accurate temperature modeling.

The mathematical models used to describe fluid flows are based on fundamental conservation principles, including the conservation of mass, momentum, and energy. These principles give rise to specific types of mathematical equations known as conservation laws, which, in their simplest one-dimensional form can be written as

$$\partial_t w + \partial_x f(w) = 0\,. \tag{1.1}$$

In here, w is unknown and depends on both space x and time t. The time derivative ∂_t indicates the evolution of w over time, while the spatial derivative ∂_x shows how the flux $f(w)$ changes across the domain. This combination leads to a system of nonlinear partial differential equations (PDEs), which is typically difficult, if not impossible, to solve analytically, as the complexity of the flux function $f(w)$ arises from the nonlinear interactions between mass, momentum, and energy. Nevertheless, solutions to eq. (1.1) provide valuable insights into technical and physical problems, making attempts to obtain approximate solutions still very much worthwhile. Yet, where and how does one typically begin the process of deriving a numerical approximation?

Before tackling that question, we want to emphasize that the PDEs considered in this thesis are deliberately simplified. We focus on well-established systems like the Euler equations, rather than a highly accurate Ecotron model. This strategic choice allows for a rigorous investigation into the primary goal of this work: the development of efficient numerical methods. Since these fundamental conservation laws share key challenges with the more complex Ecotron system, they serve as an ideal stepping stone for future applications on more advanced models.

[2]In this context *augmented* means extended by adding extra terms or equations to account for additional physical phenomena dictated by, for example moisture, chemical components or sometimes gravity.

Illustrative example of numerical timestepping Consider the two humans competing in a judo match at time n on the left of Figure 1.3. If one were asked to draw a follow-up sketch of these same people exactly one second later into the match at time $n + 1$, one would try to attain as much information as possible aiding in the prediction: Toward which direction are each of the athletes moving? How were both athletes positioned seconds before? Is it maybe easier to predict their positioning only a tenth of a second later? Can we use basic human anatomy to better understand the forces at play?

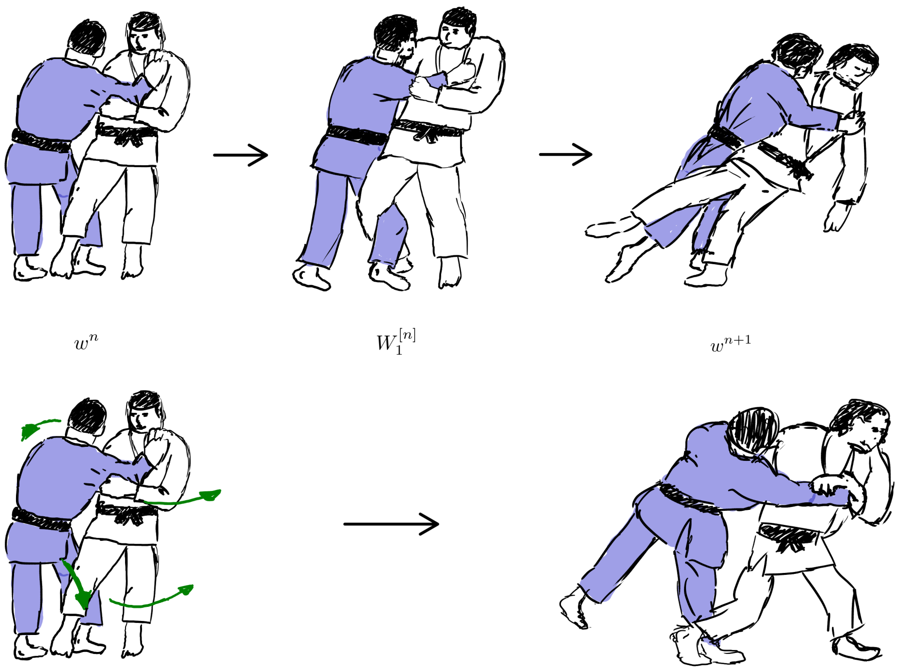


Figure 1.3: Two different scenarios in a judo bout. Both upper and lower paths start from the same situation at time n; the athletes nearly hooking each others right leg. In the upper path, a prediction for time $n+1$ is made purely on sight. An intermediate step is drawn to aid making a prediction. The athlete in the blue judogi seems to win the bout by sweeping the opponent's leg from the outside (this technique is called O-soto-gari). In the lower path, instead of having an intermediate step, forces are drawn onto the starting image. The arrows seem to indicate a counter-clockwise torsional movement. A possible prediction with this additional information might be that the athlete in the white judogi wants to block the opponent's leg by extending his right leg and using his body and arms to push him over that right leg (this technique is called Tai-otoshi).

In a first attempt, only the linear directions of movement are considered, based upon what is observed from the initial image. This yields the upper path in Figure

1.3. By means of an intermediate scenario, it seems that the athlete in the blue judogi (judo uniform) is winning by catching his rival off-guard with a leg sweep. The second attempt, instead of using an intermediate stage, puts all efforts in drawing forces applied by the athletes. With the addition of arrows in the lower path of Figure 1.3, it seems that there are counter-clockwise torsional forces involved. By blocking the opponent's leg with the right leg, and using his body and arms to throw him over that leg, the athlete in the white judogi seems to be in a winning position.

Without any extra knowledge, it will remain difficult to tell which path is more correct, or what the advantages and disadvantages of the different approaches are. Potentially, it might be better to balance efforts by gaining both information through intermediate steps and using force arrows. Therefore, establishing a means to compare accuracy is essential. In this particular example, the most straightforward approach would be to analyze a videotape of the bout. Such a videotape serves as the exact reality, also mathematically coined as *the exact solution*. Then, errors can be defined as

$$e^n = \|w^n - w(t^n)\|,$$

where $w(t^n)$ denotes the exact solution at time t^n, and w^n an approximation to it. Here, t^n is a discretization of the otherwise continuous time t. For simplicity, we rely on uniform discretizations, such that $t^n := n\Delta t$ for $n \in \mathbb{N}$. However, in many cases, such an exact solution does not exist. This raises two fundamental questions:

- Is it possible to quantify the prediction error without having an exact solution? And, if possible, can we bound this error?
- Can it be guaranteed that the path of prediction is the physically correct path?

Numerical mathematics concerns itself with developing techniques for which the answers to these questions are "yes". Over the years, many computational methods have been developed and are now standard practice for finding approximating solutions to conservation laws. This field of mathematical research, which focuses on numerical algorithms, has a rich history, with a primary emphasis on enhancing stability and efficiency — one of the main goals of this work.

Classical discretization approaches Below, a few conventional discretization approaches are provided, with the main focus put on temporal discretization. For a more comprehensive understanding, in particular in the context of high-order schemes, we refer to the books [28, 50, 51].

Method of Lines (MOL)

The most common way for approximating hyperbolic conservation laws is by means of the Method of Lines (MOL) procedure. In here, one separates the spatial discretization from the temporal discretization, and often, it is opted for discretizing the spatial domain first. For each spatial coordinate, this leads to a new set of ordinary differential equations (ODEs) which continuously depend on time t.

Typically, one gathers all the new unknowns into a vector function $y\colon \mathbb{R}^+ \to \mathbb{R}^M$, and writes the ODE system as

$$y'(t) = \Phi(y), \tag{1.2}$$

where $\Phi\colon \mathbb{R}^M \to \mathbb{R}^M$. In theory, one could consider solving the new ODE system analytically, but as to be expected, if the underlying PDE problem is complicated, one cannot reasonably expect the new ODE system to be any easier to deal with. This will only worsen with increasing spatial resolutions.

The benefit, instead, lies in the separate numerical treatment of the temporal component, given that any method of choice out of a vast catalog of ODE solvers can be applied. Especially for industrial applications, this modular approach is very much appreciated, allowing for the combination of state-of-the-art software packages for both space and time. Consider, for example, the classical Runge–Kutta fourth-order (RK4) scheme [29], one of the most well-known multistage methods. (For a more comprehensive history, see [20, §II.1].) In terms of the ODE system eq. (1.2), the computation of the stages is shown below. A color-coding scheme (outlined in the table that follows) has been applied to the indices, making it easier to visually track the role of each stage.

Color	Description
Magenta	*Inter-stage dependency*: a stage being used as input on the right-hand side of an equation to compute an intermediate stage.
Cyan	*Structural component*: a stage being defined (on the left-hand side), or a stage used as input for the final solution update.

The computation of the stages then proceeds as:

$$Y_1^{[n]} := y^n, \tag{1.3a}$$

$$Y_2^{[n]} := y^n + \frac{1}{2}\Delta t\Phi(Y_1^{[n]}), \tag{1.3b}$$

$$Y_3^{[n]} := y^n + \frac{1}{2}\Delta t\Phi(Y_2^{[n]}), \tag{1.3c}$$

$$Y_4^{[n]} := y^n + \Delta t\Phi(Y_3^{[n]}), \tag{1.3d}$$

which are used in turn to update the solution

$$y^{n+1} := y^n + \frac{\Delta t}{6}\Phi(Y_1^{[n]}) + \frac{\Delta t}{3}\Phi(Y_2^{[n]}) + \frac{\Delta t}{3}\Phi(Y_3^{[n]}) + \frac{\Delta t}{6}\Phi(Y_4^{[n]}). \tag{1.4}$$

Here, $y^n \approx y(t^n)$ and $y^{n+1} \approx y(t^{n+1})$ are approximations to $y(t)$ at times $t^n = n\Delta t$ and $t^{n+1} = (n+1)\Delta t$, respectively. Given that $y'(t) = \Phi(y)$, the sum in the update can be interpreted as a weighted average of the first-order derivatives. This representation offers a calculated measure of the rate of change of $y(t)$ with respect to time at the specific instance t^n.

The reasons for using multistage methods such as RK4 are straightforward: they are easy to implement, robust, and reasonably efficient, making them highly appealing for a wide range of applications. However, this apparent simplicity hides substantial performance costs which are not reflected in the amount of floating-point operations (i.e., specific types of computational tasks), but rather in data transfer requirements.

The memory footprint of Runge–Kutta methods comes from the intermediate stages $Y_1^{[n]}, Y_2^{[n]}, Y_3^{[n]}, Y_4^{[n]}$ that must be stored. For their computation, each stage requires reading at least one full vector and writing the result in another vector, which leads to substantial data transfer. On modern computer architectures, this can create a performance bottleneck, as processors are faster at performing calculations on data in their local cache memory than they are at retrieving it from main memory. This cost is particularly relevant in high-performance computing, where data movement between distributed processors introduces significant communication overhead; the RK4 method requires a communication round for each of its stages.

Instead, it may be advantageous to trade these data movements and communication costs for a higher amount of local computations, yielding an approach that favors large arithmetic intensity to attain high order accuracy. This can be achieved by means of a multiderivative approach, that makes use of higher-order derivatives of the solution $y(t)$, denoted by $y^{(k)}(t) =: \frac{\mathrm{d}^{k-1}}{\mathrm{d}t^{k-1}}\Phi(y(t))$ $(k > 1)$. Multiderivative methods offer more refined differential behavior of the ODE system, and, in turn, a better gauge for the sensitivity of y in relation to t. For example, the explicit fourth-order Taylor method,

$$y^{n+1} := y^n + \Delta t\Phi(y^n) + \frac{\Delta t^2}{2!}\frac{\mathrm{d}}{\mathrm{d}t}\Phi(y^n) + \frac{\Delta t^3}{3!}\frac{\mathrm{d}^2}{\mathrm{d}t^2}\Phi(y^n) + \frac{\Delta t^4}{4!}\frac{\mathrm{d}^3}{\mathrm{d}t^3}\Phi(y^n) \tag{1.5}$$

achieves the same order as the RK4 scheme, but without the use of stages[3]. The primary concern here is the difficulty of computing $\frac{\mathrm{d}^{k-1}}{\mathrm{d}t^{k-1}}\Phi(y^n)$ for $k = 1, 2, 3, 4$, which

[3]Please note that $\frac{\mathrm{d}^{k-1}}{\mathrm{d}t^{k-1}}\Phi(y)$ is a quantity derived from Φ, such that for the exact solution $y(t)$ to eq. (1.2), there holds $\frac{\mathrm{d}^k}{\mathrm{d}t^k}y(t) = \frac{\mathrm{d}^{k-1}}{\mathrm{d}t^{k-1}}\Phi(y(t))$. This leads to, e.g., $\frac{\mathrm{d}}{\mathrm{d}t}\Phi(y) = \Phi'(y)\Phi(y)$, where Φ' is the Jacobian of Φ with respect to y.

depends directly on the complexity of the function $\Phi(y)$ and requires it to be sufficiently smooth (C^3 for this fourth-order case). Although certain advanced discretizations (e.g., finite volume methods with limiters) can violate this requirement, the scope of this work is confined to problems where this assumption holds.

Even when this smoothness is guaranteed, the task of computing the derivatives remains highly complex. By repeatedly making use of the ODE system eq. (1.2), this leads to the tensor formulas

$$\frac{\mathrm{d}}{\mathrm{d}t}\Phi(y) = \Phi'(y)y^{(1)}\,, \tag{1.6a}$$

$$\frac{\mathrm{d}^2}{\mathrm{d}t^2}\Phi(y) = \Phi''(y) \bullet \left[y^{(1)}|y^{(1)}\right] + \Phi'(y)y^{(2)}\,, \tag{1.6b}$$

$$\frac{\mathrm{d}^3}{\mathrm{d}t^3}\Phi(y) = \Phi'''(y) \bullet \left[y^{(1)}|y^{(1)}|y^{(1)}\right] + 3\Phi''(y) \bullet \left[y^{(1)}|y^{(2)}\right] + \Phi'(y)y^{(3)}\,, \tag{1.6c}$$

where we ignore the $t-$dependency for the ease of presentation. The bullet operator is the tensor action, i.e.,

$$\Phi''' \bullet [u|v|w] := \sum_{j,k,l=1}^{M} \frac{\partial^3\Phi}{\partial y_j \partial y_k \partial y_l} u_j v_k w_l\,,$$

where $u, v, w \in \mathbb{R}^M$. Furthermore, not only is it a cumbersome procedure in general to explicitly embed all the terms used into eq. (1.6), the size M of the ODE solution also plays a crucial role. For practical case studies (e.g., think of a discretization of the equations describing evapotranspiration in a full Ecotron), it is not out of the ordinary that $M \gg 10^6$. And, therefore, because the MOL approach opts first for a spatial discretization, computing higher order derivatives $y^{(k)}$ is intrinsically interwoven with the selected spatial mesh, and thus M. This, in turn, is reflected in the tensors $\Phi'(y), \Phi''(y)$ and $\Phi'''(y)$ that grow massively in dimension; for instance, $\Phi^k(y)$ is an $M \times \cdots \times M$ (k-times) tensor.

To make the multiderivative approach viable, these tensors should never be explicitly formed or stored. Instead, Jacobian-free (or tensor-free) techniques, such as the recursive application of finite differences are preferred, yielding derivative terms $\frac{\mathrm{d}^{k-1}}{\mathrm{d}t^{k-1}}\Phi(y)$ as vectors of size M without the prohibitive memory cost of the tensors, thereby retaining the communication benefits of a single-stage method. This Jacobian-free philosophy is fundamental to the work presented in this thesis. For its specific application to ODE systems, a remedy based on suitable interpolation is analyzed in [Paper III].

Lax–Wendroff (LW) procedure

In light of the aforementioned information, it is easy to understand why a MOL approach in combination with a Runge–Kutta (RK) scheme is very attractive. Nonetheless, this modular approach is not devoid of limitations [37, 38, 43, 53]:

- **Loss of spatio-temporal interactions:** Fully separating the numerical discretizations can result in the loss of certain spatio-temporal interactions, making it more difficult to capture these details during the approximation process.
- **Stability constraints:** To ensure good stability properties, strong stability preserving (SSP) Runge–Kutta schemes (see, for example, [18]) are often preferred. These methods allow for relatively large CFL[4] condition numbers, enabling reasonably sized timesteps. However, they are subject to an order barrier: for orders higher than three, stability properties become less favorable. This either imposes a stricter CFL condition or necessitates additional stages in the algorithm, leading to efficiency losses in both cases.
- **Oscillatory behavior and limiting:** High-order linear methods are well known for exhibiting oscillatory behavior near discontinuities. To ensure physically meaningful results, a limiter is required that maintains high-order accuracy in smooth regions while reverting to first-order accuracy near shocks. In Runge–Kutta methods, this limiting process must be performed after each stage. Because this extra step adds computational cost, minimizing the number of stages can improve performance.

For these reasons, particularly in the last two decades, researchers have dedicated significant efforts to developing novel methods that overcome these limitations while maintaining practical applicability. Many of the most successful approaches build upon a 1960 paper [30] by Peter Lax and Burton Wendroff, who recognized the potential of Taylor methods for numerically solving partial differential equations. Conceptually, the Lax–Wendroff procedure reverses the MOL framework by first selecting the temporal scheme and then determining the appropriate spatial (and temporal) discretizations to ensure its effectiveness.

So, if, for example, we would like to maintain high efficiency with low communication costs, we can directly apply the explicit fourth-order Taylor method eq. (1.5) on the PDE system,

$$w(x, t^{n+1}) \approx w(x, t^n) + \sum_{k=1}^{4} \frac{\Delta t^k}{k!} \partial_t^k w(x, t^n),$$

[4]The Courant-Friedrichs-Lewy (CFL) number is a dimensionless quantity used in numerical simulations to describe the relationship between the speed of information, the timestep, and the size of the mesh (see, e.g., [31]).

which is approximately correct up to a term $\mathcal{O}(\Delta t^5)$ under the assumptions that w and $f(w)$ are smooth enough. However, now, to create a numerical scheme, something still has to be done about the continuous spatial variable x and the derivatives $\partial_t^k w(x, t^n)$. The latter is resolved in the Lax–Wendroff method by expressing temporal derivatives in terms of spatial derivatives of the fluxes through the Cauchy–Kowalevskaya procedure, which repeatedly makes use of eq. (1.1). For simplicity, we display the resulting derivatives for the one-dimensional case and given that f is scalar:

$$\begin{aligned}
\partial_t^2 w &= -\big(f'(w)\partial_t w\big)_x\,,\\
\partial_t^3 w &= -\big(f''(w)(\partial_t w)^2 + f'(w)\partial_t^2 w\big)_x\,,\\
\partial_t^4 w &= -\big(f'''(w)(\partial_t w)^3 + 3f''(w)\partial_t w \partial_t w)^2 + f'(w)\partial_t^3 w\big)_x\,.
\end{aligned}$$

In here, $f'(w) = \frac{\partial f(w)}{\partial w}$ denotes the Jacobian w.r.t. w, with higher derivatives following the same notation. If one then recursively substitutes the time derivatives into one another, only spatial derivatives ∂_x are left. Next, one could, for example, apply a discontinuous Galerkin discretization to obtain a complete numerical scheme. Alternatively, one could also decide to partially transition t-derivatives into x-derivatives with the intention of maintaining mixed derivatives and applying a finite difference approach.

Among the most acknowledged methods that use some variant of the Lax–Wendroff procedure are the ADER (Arbitrary order using DERivatives) methods; see, e.g., [14–16, 42, 47, 48] and the references therein. Also, higher-order extensions of the Lax–Wendroff procedure using WENO and discontinuous Galerkin (DG) reconstructions are very popular [19, 25, 32, 36–38].

Upon closer examination, the above formulas are very similar to the tensor formulas eqs. (1.6) for ODE systems. In fact, they can be viewed as an extension to hyperbolic PDEs. It is therefore not surprising that applying the Cauchy–Kowalevskaya procedure is at least as challenging to work with as in the ODE case, especially since the values $f'(w)$, $f''(w)$ and $f'''(w)$, which are generally tensors, are demanded in software implementations either exactly or through an approximation. Given the often intricate structure of the flux function $f(w)$, one resorts frequently to highly complex symbolic calculations.

Multistage multiderivative solvers When weighing the ups and downs of the classical Method of Lines and Lax–Wendroff procedures, it is natural to aim for a balanced method that extracts the best out of both. That is, preferably we keep the amount of stages low, and thus static data transfer at a lower amount, whilst retaining high convergence order through the use of additional temporal derivatives. Such methods exist, and they are aptly named multistage multiderivative solvers, or sometimes multistage Runge–Kutta (MDRK) methods. The general form of an m-derivative, s-stage MDRK method can, in ODE format, be written as

$$\begin{aligned} Y_l^{[n]} &:= y^{n-1} + \sum_{k=1}^{m} \Delta t^k \sum_{\nu=1}^{s} a_{l\nu}^{\{k\}} \frac{\mathrm{d}^{k-1}}{\mathrm{d}t^{k-1}} \Phi(Y_\nu^{[n]}), \quad l = 1, 2, \dots, s, \\ y^n &:= y^{n-1} + \sum_{k=1}^{m} \Delta t^k \sum_{l=1}^{s} b_l^{\{k\}} \frac{\mathrm{d}^{k-1}}{\mathrm{d}t^{k-1}} \Phi(Y_l^{[n]}), \end{aligned}$$

where the coefficients $a_{l\nu}^{\{k\}}$ and $b_l^{\{k\}}$ ($k = 1, \dots, m$ and $l, \nu = 1, \dots, s$) fully constitute the method. It is easy to see that the RK4 method (1.3)-(1.4) and the fourth-order Taylor method (1.5) are specific versions of an MDRK method. The first one is obtained by setting $m = 1$ and $s = 4$, whereas the latter can be retrieved by setting $m = 4$ and $s = 1$ (having only a single stage which is equal to the update).

In particular, keeping our focus on fourth order timestepping methods, a popular explicit fourth order 2DRK4 method [6] utilizes two derivatives to compute two stages

$$\begin{aligned} Y_1^{[n]} &:= y^n, \\ Y_2^{[n]} &:= y^n + \frac{\Delta t}{2} \Phi(Y_1^{[n]}) + \frac{\Delta t^2}{8} \frac{\mathrm{d}}{\mathrm{d}t} \Phi(Y_1^{[n]}), \end{aligned}$$

which are in turn used to compute a fourth-order update

$$y^{n+1} := y^n + \Delta t \Phi(Y_1^{[n]}) + \Delta t^2 \left(\frac{1}{6} \frac{\mathrm{d}}{\mathrm{d}t} \Phi(Y_1^{[n]}) + \frac{1}{3} \frac{\mathrm{d}}{\mathrm{d}t} \Phi(Y_2^{[n]}) \right).$$

Not only is the fourth order achieved, the linear stability properties are exactly the same as those of the RK4 method [43, §2.3.2].

This 2DRK method by itself illustrates the potential that MDRK methods possess and opens up a plethora of schemes that can be tweaked to favor data transfer and communication over floating-point operations, or the other way around, depending on the problem at hand. And, it does not stop at MDRK methods. General Linear Methods (GLMs), see for example the books of Butcher and Jackiewicz [4, 23], additionally add multiple steps (thus not only t^n and t^{n+1}, but more time instances) into the numerical scheme, which creates further room for designing schemes that balance between convergence orders, stability, memory, and efficiency. The only major difference between a GLM and an MDRK method is that, instead of directly using y^n and y^{n+1}, high-order approximations to some linear combination of the solution y and its derivatives are calculated. For further details, compare [Paper I] and [Paper II].

In the example above, only methods of order four were considered. However, there is no particular reason to restrict this choice. As expected, for even higher orders, additional stages and/or derivatives must be incorporated, bringing us back to the symbolic implementation of the Jacobians $f^{(k)}(w)$ for $k \geq 1$. One could argue that, although being very slow, obtaining symbolic formulas is a non-recurring task, as one can simply store the resulting outcome in code and use these for any future simulation. However, that is not completely true. The fact of the matter is that there is no single perfect flux function $f(w)$ that is utilized for all different flow settings. The flux function $f(w)$ depends on the equations at hand, and it differs depending on whether soil flow, low Mach air flow, or other fluid flows are considered. Not only this, also necessary equations of state used within f are not always explicitly given, e.g., for difficult media the equation of state relating pressure to density, energy, and velocity is given as a spline-reconstruction from data. Also, f changes if certain chemical elements are accounted for, or if alternative viscosity models are used. So there is a lot of variability in the flux function $f(w)$ in respect to flow dynamics.

Aside from the modelling aspect, numerically, there can also be various reasons as to why computing symbolic formulas possibly is not a single-time task. Similar as to how there is no single perfect flux function, there is also no absolute algorithm that reigns for all simulations. Preferably, the algorithm of choice cleverly manages to exploit the inherent dynamics of the problem to improve stability and efficiency of the computations. One such approach, for example, is by applying an Implicit-Explicit (IMEX) scheme that splits the flux function into separate parts, one of which is treated implicitly, and the other being treated explicitly [3]. In case of an additive splitting, this looks like $f(w) = f_I(w) + f_E(w)$. And, as expected, both $f_I(w)$ and $f_E(w)$ are handled separately in the symbolic calculations for the derivatives $\partial_t^k w$. Whether developing new splittings, or whether picking a particular splitting out of the inventory of IMEX methods, it is not preferable being delayed by the necessity of the symbolic arithmetic that is needed as a pre-processing step on top of the actual numerical simulation itself. Likely, it is because of these inconvenient calculations of flux derivatives that MD schemes have not been put much to practice, regardless of their nearly century old history – for a detailed overview, see [43].

Jacobian-free approach From a theoretical standpoint, an integrated mechanism capable of automatically approximating $\widetilde{w}^{(k)} \approx \partial_t^k w$ within the algorithm would be highly beneficial, allowing users to focus on the numerical scheme itself. Such a mechanism should be mathematically expressible, ensuring that control is maintained for analyzing convergence and evaluating the stability effects on the scheme. There is a potential downside to this approach however, that being the computational overhead introduced by the additional operations required for the approximation. Fortunately, advances in modern hardware capabilities mitigate these concerns, enabling techniques that were once deemed impractical. For instance, [53] proposed an explicit method based on recursive finite differences to bypass symbolic flux derivative computations. Finite differences are particularly well-suited for mathematical analysis, as demonstrated by Carrillo and Parés in [5], who in 2019 extended this method into the *compact approximate Taylor* (CAT) method, which improves upon the stability properties of the former.

The CAT method is a Jacobian-free approach, as it avoids the computation of the Jacobian $\frac{\partial f(w)}{\partial w}$ and higher order tensors that arise from the introduction of the derivatives $\partial_t^k w$. Their work is based on Taylor methods, and thus relies on obtaining high efficiency solely through the use of higher order derivatives. In contrast, traditional time-integrators do not use $\partial_t^2 w$ and higher order derivatives, yet obtain higher orders of convergence through the inclusion of either stages or previous steps. In [Paper I] and [Paper II] we try to strike a balance between these two. That is, by tinkering with the inclusion of multiple derivatives (MD), multiple stages and/or multiple steps, further flexibility is introduced, which enables the development of schemes that are both more stable and more efficient. Given that the primary aim of these studies is to assess practical applicability, only explicit schemes are considered.

In [Paper I], the CAT approach is extended by the inclusion of multiple stages. These methods, referred to as MDRKCAT methods, draw their name from the connection to traditional multistage integrators, commonly known as Runge–Kutta (RK) methods. A von Neumann stability analysis revealed that these methods have a high dependency both on the amount of stages and derivatives used, having the potential to either significantly improve or severely compromise their stability properties. Consequently, this finding underscores the importance of carefully selecting a well-tuned set of parameters to define the MDRK scheme.

The MDRKCAT methods are further expanded upon in [Paper II] that applies the same Jacobian-free strategy to multiderivative general linear methods (MDGLMs). As explained earlier, higher efficiency of integrators can as well be achieved by the inclusion of previous steps, as in a multistep method. To bridge the gap between multistage and multistep methods, General Linear Methods (GLMs) were introduced [23]. Unlike traditional methods that produce a single output, a GLM generates a collection of vectors, typically approximations of linear combinations of the time derivatives $\partial_t^k w$. As a result, MDGLMs show an added dependency on the time derivatives, arising both from their explicit use within the scheme and from their influence on the output vectors. This dependency makes Jacobian-free approaches particularly convenient and well-suited. Specifically in [Paper II], compact approximate MDGLMs that exhibit strong-stability preserving properties are constructed. These methods have two external stages, use up to four derivatives (thus $\partial_t^4 w$) and achieve convergence of up to order nine.

Jacobian-free approach for stiff equations The methods presented in [Paper I] and [Paper II], although being promising, might not be sufficient for the simulation of any fluid flow as they are *explicit*. Often, the dynamics of the flows are determined by factors that operate at different time scales. Think for example about a plant in the open; the blowing wind hitting the plant might have a much greater speed in comparison to the speed at which water droplets evaporate from that plant. However, both play an effective role in the flow dynamics of that ecosystem.

Such differences in time scales often lead to ordinary differential equations (ODEs) or PDEs that are described to be *stiff* [21, 46]. While there is no single definition of stiffness, it is often described heuristically as a property of equations that require excessively small timesteps for explicit time-integrators to maintain stability. When timesteps are too large, numerical solutions may exhibit strong oscillations. Conversely, sufficiently small timesteps ensure stability but make the method computationally slow. Several contributing factors might explain this behavior, for example in relation to fluid flow, this can occur when viscous effects are taken into consideration. Viscosity, being diffusive by nature, behaves fundamentally differently from advective flow, leading to stringent stability requirements that demand impractically small timesteps. This limitation makes explicit methods less appealing.

Implicit methods do not suffer that fate in relation to stiff problems, they maintain much better stability regardless of using larger timesteps. However, as the name suggests, the approximate solution is given implicitly in the numerical scheme. Therefore, in order to obtain an explicit solution $w(x, t)$, additional computational procedures are inevitable. An often employed tactic to extract w is by linearization of the implicit equations via Newton's method in combination with a solver for the resulting matrix system. When timesteps are very restrictive, the computational cost of these actions can be worthwhile.

As a next step in this thesis, it therefore makes sense to try devising and understanding implicit multiderivative methods, particularly in a stiff setting. Directing focus toward ODEs simplifies the analysis, as there are no spatial effects involved. However, it is not necessarily a huge restriction; many ODEs, having a large number of equations, arise from a spatial discretization of a PDE. For example, conservation laws with diffusive effects can be cast into ODEs of the form

$$y'(t) = \Phi(y) := F(y) + \frac{1}{\varepsilon}G(y),$$

with $\varepsilon \ll 1$ a small positive parameter that introduces the stiffness. In [Paper III] we study Jacobian-free implicit multiderivative Runge–Kutta (MDRK) methods for ODEs in the above form, based on the *approximate implicit Taylor* method by Baeza et al. [1].

The inclusion of higher-order derivatives in relation to eq. (1.3) leads to two undesirable outcomes: (1) each added k-th derivative introduces very small power terms ε^k, and (2) the complexity of the derivative formulas increases rapidly with each successive order (see eqs. (1.6)). These drawbacks become particularly evident when using Newton's method as a nonlinear solver: the condition number of the Newton matrix grows exponentially with each additional derivative, and the matrix must be derived from intricate formulas that require tensor calculations.

[Paper III] extensively investigates the origins of these issues and proposes a resolution. First, by introducing an additional equation to the ODE system for each derivative, the derivatives are incorporated as part of the unknown solution. This approach distributes the ε-dependencies among the newly added equations, thereby mitigating the exponential growth in the condition number. Second, by applying a Jacobian-free method, similar to those described in [Paper I] and [Paper II], the need for complicated formulas or tensor calculations is eliminated. Together, these strategies demonstrate notable advantages in numerical results compared to more conventional methods, especially in cases where higher-order derivatives dominate the system's behavior. However, this approach is not universally optimal, because adding new relations increases the overall system size, leading to greater computational costs and potentially diminishing returns for simpler problems. A takeaway from [Paper III] is the need to define a clear criterion for transitioning between the proposed resolution and conventional approaches to ensure optimal performance.

Outline of the thesis The results of the doctoral research are presented as a cumulative thesis, meaning that the main findings are reported in one-to-one copies of the published papers. Chapter 2 compiles the published papers, organizing them in the order outlined in the introduction. The first two papers focus on the Jacobian-free approach, with the first applying it to explicit multiderivative Runge–Kutta methods and the second to explicit general linear methods. The third paper explores the application of implicit multiderivative methods, specifically addressing the conditioning of the linearized system arising from Newton's method when these techniques are applied to stiff problems. The thesis concludes in Chapter 3 with a summary of the findings and recommendations for future research.

Chapter 2

Published papers

This thesis is based on the following publications:

[Paper I]	J. Chouchoulis, J. Schütz, and J. Zeifang, *Jacobian-free explicit multiderivative Runge-Kutta methods for hyperbolic conservation laws*, Journal of Scientific Computing **90**.96 (2022), `https://link.springer.com/article/10.1007/s10915-021-01753-z`.
[Paper II]	A. Moradi, J. Chouchoulis, R. D'Ambrosio, and J. Schütz, *Jacobian-free explicit multiderivative general linear methods for hyperbolic conservation laws*, Numerical Algorithms (2024), `https://doi.org/10.1007/s11075-024-01771-6`.
[Paper III]	J. Chouchoulis and J. Schütz, *Jacobian-free implicit MDRK methods for stiff systems of ODEs*, Applied Numerical Mathematics **196** (2024), pp. 45–61, `https://www.sciencedirect.com/science/article/pii/S0168927423002672`.

On the following pages, the papers are presented in the same order as listed above. The styling and formatting of each paper 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. Due to the substantial amount of specialized syntax required to introduce the concepts, readers are advised to proceed carefully when reading and comparing the papers.

Paper I:
Jacobian-Free Explicit Multiderivative Runge–Kutta Methods for Hyperbolic Conservation Laws

Journal of Scientific Computing (2022) 90:96
https://doi.org/10.1007/s10915-021-01753-z

Jacobian-Free Explicit Multiderivative Runge–Kutta Methods for Hyperbolic Conservation Laws

Jeremy Chouchoulis[1] · Jochen Schütz[1] · Jonas Zeifang[1]

Received: 12 July 2021 / Accepted: 13 December 2021 / Published online: 11 February 2022

Abstract
Based on the recent development of Jacobian-free Lax–Wendroff (LW) approaches for solving hyperbolic conservation laws (Zorio et al. in J Sci Comput 71:246–273, 2017, Carrillo and Parés in J Sci Comput 80:1832–1866, 2019), a novel collection of explicit Jacobian-free multistage multiderivative solvers for hyperbolic conservation laws is presented in this work. In contrast to Taylor time-integration methods, multiderivative Runge–Kutta (MDRK) techniques achieve higher-order of consistency not only through the excessive addition of higher temporal derivatives, but also through the addition of Runge–Kutta-type stages. This adds more flexibility to the time integration in such a way that more stable and more efficient schemes could be identified. The novel method permits the practical application of MDRK schemes. In their original form, they are difficult to utilize as higher-order flux derivatives have to be computed analytically. Here we overcome this by adopting a Jacobian-free approximation of those derivatives. In this paper, we analyze the novel method with respect to order of consistency and stability. We show that the linear CFL number varies significantly with the number of derivatives used. Results are verified numerically on several representative testcases.

Keywords Hyperbolic conservation laws · Multiderivative Runge–Kutta · Lax–Wendroff · Finite differences

Mathematics Subject Classification 65M06 · 65M08 · 65M12 · 35L65

Jeremy Chouchoulis
jeremy.chouchoulis@uhasselt.be

Jochen Schütz
jochen.schuetz@uhasselt.be

Jonas Zeifang
jonas.zeifang@uhasselt.be

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

1 Introduction

In this work, we present a novel discretization method for the numerical approximation of one-dimensional hyperbolic conservation laws on domain $\Omega \subset \mathbb{R}$,

$$\begin{aligned} w_t + f(w)_x &= 0, \qquad \text{on} \quad (x,t) \in \Omega \times (0, T_{\text{end}}], \\ w(x,0) &= w_0(x). \end{aligned} \tag{1}$$

Our primary interest is on temporal integration. In recent years, there has been quite some progress on the further development of the *multiderivative* paradigm for temporal integration, see, e.g., [5,6,11,27,30] and the references therein. Assume that one is given a scalar ODE, e.g.,

$$y'(t) = \Phi(y) \tag{2}$$

for some flux function Φ. Multiderivative schemes make use of not only Φ, but also of the quantities $y''(t) \equiv \Phi'(y)\Phi(y)$, $y'''(t) \equiv \ldots$ and so on. Using this approach, one can derive stable, high-order and storage-efficient schemes very easily [28]. This can be extended to partial differential equations (PDEs) with a time-component, such as Eq. (1), depending on the method either directly through the method-of-lines-discretization [27] or through a Lax–Wendroff procedure, see, e.g., [3,4,16,18,22,36]. The Lax–Wendroff method expresses temporal derivatives of the unknown function w in terms of the fluxes through the Cauchy–Kowalevskaya procedure. As an example, we consider – for simplicity given that f is scalar – the second time-derivative of w. Due to Eq. (1), there holds

$$w_{tt} = -(f(w)_x)_t = -(f(w)_t)_x, \tag{3}$$

and

$$f(w)_t = f'(w)w_t = -f'(w)f(w)_x;$$

hence

$$w_{tt} = \left(f'(w)f(w)_x\right)_x = 2f'(w)f''(w)w_x^2 + f'(w)^2 w_{xx}. \tag{4}$$

Already at this stage, one can see that this approach is very tedious as it necessitates highly complex symbolic calculations.

Still, the potential LW-methods bear is very well recognized among researchers. Over the last two decades, plenty of authors have put effort into developing high-order variants of the LW-method for nonlinear systems. Particularly the ADER (Arbitrary order using DERivatives) methods, see, e.g., [8–10,29,33,34] and the references therein, gained a lot of interest. Also, higher-order extensions of the LW-method using WENO and discontinuous Galerkin (DG) reconstructions were investigated [13,15,19,22–24].

Our essential intent of this paper is to make explicit multistage multiderivative solvers more accessible as a means to solve PDEs. Although such solvers have been theoretically studied since the early 1940's (see [30] for an extensive review), the schemes have not been put much to practice, which is most likely due to the necessary cumbersome calculation of flux derivatives. In [3], Carrillo and Parés have, based on the earlier work [36], developed the *compact approximate Taylor* (CAT) method to circumvent having to symbolically compute flux derivatives. By means of an automatic procedure, the higher-order temporal derivatives of w, such as in Eq. (4), are approximated. Their work is based on Taylor methods, i.e., time integration is given by

$$w(x,t^{n+1}) = w(x,t^n) + \sum_{k=1}^{\mathtt{r}} \frac{\Delta t^k}{k!} \partial_t^k w(x,t^n) + \mathcal{O}(\Delta t^{\mathtt{r}+1}).$$

In this work, we extend their approach to more general multiderivative integration methods, more precisely, to multiderivative Runge–Kutta (MDRK) methods.

The paper is structured in the following manner: In Sect. 2 multiderivative Runge–Kutta (MDRK) time integrators for ODEs are introduced, given that they form the central mechanism of this work. Thereafter, in Sect. 3 we shortly revisit the Jacobian-free approach of the CAT method and introduce the explicit Jacobian-free MDRK solver for hyperbolic conservation laws, termed MDRKCAT. After describing the numerical scheme, in Sect. 4 we prove consistency, and in Sect. 5 analyze linear stability. Via several numerical cases we verify and expand on the theoretical results in Sect. 6. At last, we draw our conclusions and discuss future perspectives in Sect. 7.

2 Explicit Multiderivative Runge–Kutta Solvers

We start by considering the system of ODEs defined by Eq. (2) in which Φ is a function of the solution variable $y \in \mathbb{R}^{\mathrm{m}}$. In order to apply a time-marching scheme, we discretize the temporal domain with a fixed timestep Δt by iterating N steps such that $\Delta t = T_{\text{end}}/N$. Consequently, we define the time levels by

$$t^n := n\Delta t \qquad 0 \le n \le N.$$

Remark 1 Note that, although the fully space-time-discrete algorithm (Alg. 1) *seems* to have a multistep flavour, this is ultimately not the case. It is therefore of no necessity to consider a uniform timestep, which is also demonstrated numerically in Sect. 6.

The central class of time integrators in this work are *explicit* multiderivative Runge–Kutta (MDRK) methods. These form a natural generalization of classical explicit Runge–Kutta methods by adding extra temporal derivatives of $\Phi(w)$. The additional time derivatives can be recursively calculated via the chain rule, there holds

$$\frac{\mathrm{d}^k}{\mathrm{d}t^k}\Phi(y) = \frac{\mathrm{d}^{k-1}}{\mathrm{d}t^{k-1}}\left(\Phi'(y)\dot{y}\right) = \frac{\mathrm{d}^{k-1}}{\mathrm{d}t^{k-1}}\left(\Phi'(y)\Phi(y)\right).$$

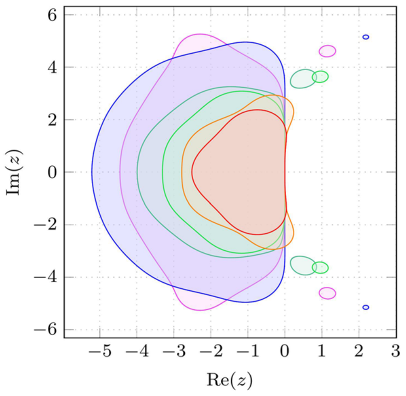
For a more detailed description, we refer to [30]. To present our ideas, let us formally define the MDRK scheme as follows:

Definition 1 [30, Definition 2] An explicit q-th order accurate $\mathtt{r}$-derivative Runge–Kutta scheme using $\mathtt{s}$ stages ($\mathtt{r}$DRKq-$\mathtt{s}$) is any method which can be formalized as

$$y^{n,l} := y^n + \sum_{k=1}^{\mathtt{r}} \Delta t^k \sum_{\nu=1}^{l-1} a_{l\nu}^{(k)} \frac{\mathrm{d}^{k-1}}{\mathrm{d}t^{k-1}}\Phi\left(y^{n,\nu}\right), \quad l = 1,\ldots,\mathtt{s},$$

where $y^{n,l}$ is a stage approximation at time $t^{n,l} := t^n + c_l\Delta t$. The update is given by

$$y^{n+1} := y^n + \sum_{k=1}^{\mathtt{r}} \Delta t^k \sum_{l=1}^{\mathtt{s}} b_l^{(k)} \frac{\mathrm{d}^{k-1}}{\mathrm{d}t^{k-1}}\Phi(y^{n,l}).$$



Scheme	$R(z)$
— 2DRK3-2	$\sum_{k=0}^{3} \frac{z^k}{k!}$
— 2DRK4-2	$\sum_{k=0}^{4} \frac{z^k}{k!}$
— 2DRK5-3	$\sum_{k=0}^{5} \frac{z^k}{k!} + \frac{z^6}{600}$
— 3DRK5-2	$\sum_{k=0}^{5} \frac{z^k}{k!} + \frac{z^6}{900}$
— 3DRK7-3	$\sum_{k=0}^{7} \frac{z^k}{k!} + c_8 z^8 + c_9 z^9$
— 4DRK6-2	$\sum_{k=0}^{6} \frac{z^k}{k!} + \frac{z^7}{6480} + \frac{z^8}{77760}$

Fig. 1 Regions of absolute stability $\mathcal{R} := \{z \in \mathbb{C} \mid |R(z)| \leq 1\}$ and stability functions $R(z)$ for the two-derivative schemes 2DRK3-2, 2DRK4-2, 2DRK5-3 [5], the three-derivative schemes 3DRK5-2 and 3DRK7-3 [21] and the four-derivative scheme 4DRK6-2 (see Appendix A). Except for 3DRK5-2 and 4DRK6-2, all schemes contain parts of the imaginary axis. Note that in the stability function of 3DRK7-3, we have defined $c_8 := \frac{1}{23520} - \frac{\sqrt{2}}{70560}$ and $c_9 := \frac{11}{1481760} - \frac{\sqrt{2}}{246960}$

The given coefficients $a_{l\nu}^{(k)}$ and $b_l^{(k)}$ determine the scheme; they are typically summarized in an extended Butcher tableau.

Remark 2 Note that standard Taylor methods can be cast in the framework of Definition 1 through setting $\mathtt{s} = 1, a_{11}^{(k)} = 0$ and $b_1^{(k)} = 1/k!$ $(k = 1, \ldots, \mathtt{r})$. The multiderivative Runge–Kutta schemes used in this work can be found through their extended Butcher tableaux in Appendix A.

Remark 3 The stability regions of the used Runge–Kutta methods are visualized in Fig. 1, see [5,14,21] for more details. Note that except for 3DRK5-2 and 4DRK6-2, all schemes contain parts of the imaginary axis.

3 Multiderivative Runge–Kutta Solvers for Hyperbolic Conservation Laws

Discretizing the spatial part of the hyperbolic conservation law (1) necessitates a discretization of the domain Ω. Hence, consider

$$\{x_1, \ldots, x_M\}$$

to be a uniform partition of Ω into M cells of size Δx. A natural extension of Definition 1 applied to Eq. (1) can then be expressed as

$$w_i^{n,l} := w_i^n - \sum_{k=1}^{\mathtt{r}} \Delta t^k \sum_{\nu=1}^{l-1} a_{l\nu}^{(k)} D_x D_t^{k-1} f(w_i^{n,\nu}), \tag{6a}$$

$$w_i^{n+1} := w_i^n - \sum_{k=1}^{\mathtt{r}} \Delta t^k \sum_{l=1}^{\mathtt{s}} b_l^{(k)} D_x D_t^{k-1} f(w_i^{n,l}), \tag{6b}$$

for $l = 1, \dots, \mathrm{s}$; with D_x and D_t being suitable approximations to ∂_x and ∂_t to be explained in the sequel. Contrary to the complete Cauchy-Kovalevskaya procedure as outlined for the second derivative in (4), only *one* time derivative of the solution is transformed into a spatial derivative (see (3)).

The core focus of this paper is to avoid the explicit use of Jacobians of the flux function f. Jacobians of f arise due to the usage of higher temporal derivatives, see, e.g., Eq. (4). In this, we follow the compact approximate Taylor (CAT) approach outlined in [3]. Since the CAT method heavily relies on discrete differentiation, first a small part is devoted to introducing the fundamental notation. Thereafter the method is described and applied to Eq. (6).

3.1 Discrete Differentiation

In this short section, we fix the notation on using finite differencing [1,25,26]. Considering central differences, the $(2p+1)$-point Lagrangian polynomials are given by

$$L_{p,j}(\omega) := \prod_{\substack{r=-p \\ r\neq j}}^{p} \frac{\omega - r}{j - r}, \qquad j = -p, \dots, p. \tag{7}$$

It is well-known that these polynomials can be used to interpolate $\varphi : \mathbb{R} \to \mathbb{R}$ in the points $x_{i-p}, \dots, x_{i+p}$ through

$$\mathcal{P}_i\varphi(x) := \sum_{j=-p}^{p} L_{p,j}\left(\frac{x - x_i}{\Delta x}\right)\varphi(x_{i+j}). \tag{8}$$

Similarly, from the $2p$-point Lagrangian polynomials

$$\ell_{p,j}(\omega) := \prod_{\substack{r=-p+1 \\ r\neq j}}^{p} \frac{\omega - r}{j - r}, \qquad j = -p+1, \dots, p, \tag{9}$$

(note that the index of the product begins at $r = -p + 1$) we obtain the unique polynomial of degree $2p - 1$ interpolating φ in the points $x_{i-p+1}, \dots, x_{i+p}$ through

$$\mathcal{Q}_i\varphi(x) := \sum_{j=-p+1}^{p} \ell_{p,j}\left(\frac{x - x_i}{\Delta x}\right)\varphi(x_{i+j}). \tag{10}$$

In the sequel, we use a similar notation as in [3]:

Definition 2 [3] For the k-th derivative $0 \le k \le 2p$, we define the following quantities:

$$\begin{aligned}
\delta^{k}_{p,j} &:= L^{(k)}_{p,j}(0), & j &= -p, \dots, p, \\
\gamma^{k,m}_{p,j} &:= \ell^{(k)}_{p,j}(m), & j &= -p+1, \dots, p, \quad m = -p+1, \dots, p.
\end{aligned}$$

Note that the k-th derivatives $L^{(k)}_{p,j}(0)$ and $\ell^{(k)}_{p,j}(m)$ are derived analytically from Eqs. (7) and (9), respectively.

Approximate derivatives can thus be derived from

$$(\mathcal{P}_i\varphi)^{(k)}(x_i) = \frac{1}{\Delta x^k}\sum_{j=-p}^{p} \delta^{k}_{p,j}\varphi(x_{i+j}), \tag{11a}$$

$$(\mathcal{Q}_i\varphi)^{(k)}(x_{i+m}) = \frac{1}{\Delta x^k}\sum_{j=-p+1}^{p} \gamma_{p,j}^{k,m}\varphi(x_{i+j}), \tag{11b}$$

with $m = -p+1, \ldots, p$. Since we are working in a discrete context, we define the linear operator counterparts of Eqs. (11a) and (11b) as

$$P^{(k)} : \mathbb{R}^{2p+1} \to \mathbb{R}, \qquad \mathbf{v} \mapsto \frac{1}{\Delta x^k}\sum_{j=-p}^{p} \delta_{p,j}^{k} v_j,$$

$$Q_m^{(k)} : \mathbb{R}^{2p} \to \mathbb{R}, \qquad \mathbf{w} \mapsto \frac{1}{\Delta x^k}\sum_{j=-p+1}^{p} \gamma_{p,j}^{k,m} w_j.$$

Remark 4 The spatial index i is neglected for the linear operators as its direct dependency on the node x_i is lost, cf. Eqs. (8) and (10). Notice also that $Q_m^{(k)}$ takes vectors in $\mathbb{R}^{2p}$ as input, whereas $P^{(k)}$ takes vectors in $\mathbb{R}^{2p+1}$ as input.

A non-centered $2p$-point finite difference method to approximate $\partial_t^k w(x_i, t^{n+m})$ for $m = -p+1, \ldots, p$ can therefore be written as

$$Q_m^{(k)}\mathbf{w}_i^{\langle n\rangle} = \frac{1}{\Delta t^k}\sum_{r=-p+1}^{p} \gamma_{p,j}^{k,m} w_i^{n+r},$$

with vector notation

$$\mathbf{w}_i^{\langle n\rangle} := \begin{pmatrix} w_i^{n-p+1} \\ \vdots \\ w_i^{n+p} \end{pmatrix}. \tag{12}$$

The angled brackets represent the local stencil function

$$\langle\cdot\rangle : \mathbb{Z} \to \mathbb{Z}^{2p} : n \mapsto \big(n-p+1, \ldots, n+p\big)^T \tag{13}$$

throughout this paper, and will be considered for both the spatial index i as the temporal index n. Note that the position of the angled bracket (top or bottom) determines whether derivation is w.r.t. time (top) or space (bottom).

To put the scheme into conservation form, in [36] auxiliary centered coefficients have been introduced. Here, the operators $P^{(k)}$ for $k \geq 1$ are written as differences of new 'half-way point' interpolation operators.

Definition 3 [36] Define $\lambda_{p,j}^{k-1}$ via the relations

$$\delta_{p,p}^{k} =: \lambda_{p,p}^{k-1}, \tag{14a}$$

$$\delta_{p,j}^{k} =: \lambda_{p,j}^{k-1} - \lambda_{p,j+1}^{k-1}, \quad j = -p+1, \ldots, p-1, \tag{14b}$$

$$\delta_{p,-p}^{k} =: -\lambda_{p,-p+1}^{k-1}. \tag{14c}$$

Remark 5 The relations given in Eq. (14) make up an overdetermined system, yet provide a unique solution $\lambda_{p,j}^{k-1}$ obtained from Eqs. (14a) and (14b), see [36, Theorem 2].

Notice the shift between k and $k-1$. This is justified because we enforce a first order derivative relation for the approximation $(\mathcal{P}_i\varphi)^{(k)}(x_i)$ by splitting the operator as

$$(\mathcal{P}_i\varphi)^{(k)}(x_i) = \frac{\left(\Lambda^{(k-1)}\varphi\right)(x_{i+1/2}) - \left(\Lambda^{(k-1)}\varphi\right)(x_{i-1/2})}{\Delta x}, \tag{15}$$

Table 1 A summary of the defined interpolation operators scaled and shifted to fit the uniform mesh locally at x_i

Functional	Linear	
$(\mathcal{P}_i\varphi)^{(k)}$	$P^{(k)}\mathbf{v}$	k-th derivative of the Lagrangian interpolation polynomial in the nodes $x_{i-p}, \dots, x_{i+p}$
$(\mathcal{Q}_i\varphi)^{(k)}$	$Q_m^{(k)}\mathbf{v}$	k-th derivative of the Lagrangian interpolation polynomial in the nodes $x_{i-p+1}, \dots, x_{i+p}$
$\left(\Lambda^{(k-1)}\varphi\right)(x_{i+1/2})$	$\varLambda^{(k-1)}\mathbf{v}$	$(k-1)$-th derivative of a half-way interpolation at $x_{i+1/2}$ using the nodes $x_{i-p+1}, \dots, x_{i+p}$ with the difference coefficients defined by Eq. (14)

in which $\Lambda^{(k-1)}$ is an operator mapping to $\mathbb{P}_{2p-1}$ so that

$$\left(\Lambda^{(k-1)}\varphi\right)(x_{i+1/2}) := \frac{1}{\Delta x^{k-1}} \sum_{j=-p+1}^{p} \lambda_{p,j}^{k-1}\varphi(x_{i+j}).$$

The linear operator alternative is defined by

$$\varLambda^{(k-1)}\colon \mathbb{R}^{2p} \longrightarrow \mathbb{R}, \qquad \mathbf{v} \mapsto \frac{1}{\Delta x^{k-1}} \sum_{j=-p+1}^{p} \lambda_{p,j}^{k-1} v_j. \tag{16}$$

An overview of all the defined interpolation operators is given in Table 1.

3.2 A Jacobian-Free MDRK Scheme

With all the building blocks at our disposal, we can now describe how the final class of methods, that we call MDRKCAT, is assembled. Starting from Eq. (6), we define the conservative updates of the solution via

$$w_i^{n,l} := w_i^n - \frac{\Delta t}{\Delta x}\left(F_{i+1/2}^{n,l} - F_{i-1/2}^{n,l}\right) \qquad l = 1, \dots, \mathtt{s}\,, \tag{17a}$$

$$w_i^{n+1} := w_i^n - \frac{\Delta t}{\Delta x}\left(F_{i+1/2}^{n} - F_{i-1/2}^{n}\right), \tag{17b}$$

in which the numerical fluxes are given by

$$F_{i+1/2}^{n,l} = \sum_{k=1}^{\mathtt{r}} \Delta t^{k-1} \sum_{\nu=1}^{l-1} a_{l\nu}^{(k)}\, \varLambda^{(0)}(\widetilde{\mathbf{f}}^{\nu})_{i,\langle 0\rangle}^{(k-1)}, \qquad l = 1, \dots, \mathtt{s}, \tag{18a}$$

$$F_{i+1/2}^{n} = \sum_{k=1}^{\mathtt{r}} \Delta t^{k-1} \sum_{l=1}^{\mathtt{s}} b_{l}^{(k)}\, \varLambda^{(0)}(\widetilde{\mathbf{f}}^{l})_{i,\langle 0\rangle}^{(k-1)}. \tag{18b}$$

For the calculation of $\widetilde{\mathbf{f}}_{i,\langle 0\rangle}^{(k-1)}$ the compact approximate Taylor (CAT) procedure [3] is used and the flux derivatives can be calculated according to Eq. (16) by

$$\varLambda^{(0)}\widetilde{\mathbf{f}}_{i,\langle 0\rangle}^{(k-1)} := \sum_{j=-p+1}^{p} \lambda_{p,j}^{0}\, \widetilde{f}_{i,j}^{(k-1)}.$$

$\widetilde{f}_{i,j}^{(k-1)} \approx \partial_t^{k-1} f(w)(x_{i+j}, t^n)$ indicates the local approximations for the time-derivatives of the flux and are given by

$$\widetilde{f}_{i,j}^{(k-1)} := Q_0^{(k-1)} (\mathfrak{f}_T)_{i,j}^{k-1,\langle n\rangle}, \qquad j = -p+1, \dots, p.$$

They rely on the approximate flux values $(\mathfrak{f}_T)_{i,j}^{k-1,n+r} \approx f(w(x_{i+j}, t^{n+r}))$. In other words, we take the $(k-1)$-st discrete temporal derivative in x_{i+j} using approximate fluxes

$$(\mathfrak{f}_T)_{i,j}^{k-1,n+r} := f\left(w_{i+j}^n + \sum_{m=1}^{k-1} \frac{(r\Delta t)^m}{m!} \widetilde{w}_{i,j}^{(m)} \right),$$

for $j, r = -p+1, \dots, p$. The only thing that is left to define are the quantities $\widetilde{w}_{i,j}^{(m)} \approx \frac{\partial^m}{\partial t^m} w(x_{i+j}, t^n)$. Their approximation makes heavy use of the Cauchy–Kovalevskaya identity $\partial_t^m w = -\partial_x \partial_t^{m-1} f(w)$, they are hence approximated by

$$\widetilde{w}_{i,j}^{(m)} := -Q_j^{(1)} \widetilde{\mathbf{f}}_{i,\langle 0\rangle}^{(m-1)}, \qquad j = -p+1, \dots, p.$$

Via the described steps, the vectors $\widetilde{\mathbf{f}}_{i,\langle 0\rangle}^{(k-1)}$ are recursively obtained, see also [3].

Definition 4 For a more precise terminology, we define the specific $\mathtt{r}$-derivative, q-th order, $\mathtt{s}$-stage MDRKCAT method as $\mathtt{r}$DRKCATq-$\mathtt{s}$.

A summary of the $\mathtt{r}$DRKCATq-$\mathtt{s}$ procedure to obtain the stage values is given in Alg. 1. Note that the flux at the left half-way point is obtained either from a shift of the index, i.e. $F_{i-1/2}^n = F_{i-1+1/2}^n$ or is given by the boundary condition.

Algorithm 1 Stages of $\mathtt{r}$DRKCATq-$\mathtt{s}$, an $\mathtt{r}$-derivative, q-th order, $\mathtt{s}$-stage MDRKCAT method

```
Stage solution ($l = 2, \dots, \mathtt{s}$):
for $j = -p+1$ to $p$ do
    $(\widetilde{f}^{l-1})_{i,j}^{(0)} = f(w_{i+j}^{n,l-1})$
end
$F_{i+1/2}^{n,l} = \sum_{\nu=1}^{l-1} a_{l\nu}^{(1)} \Lambda^{(0)} (\widetilde{\mathbf{f}}^{\nu})_{i,\langle 0\rangle}^{(0)}$
for $k = 2$ to $\mathtt{r}$ do
    Get $(\widetilde{f}^{l-1})_{i,j}^{(k-1)}$ via CAT procedure.
    $F_{i+1/2}^{n,l} \mathrel{+}= \Delta t^{k-1} \sum_{\nu=1}^{l-1} a_{l\nu}^{(k)} \Lambda^{(0)} (\widetilde{\mathbf{f}}^{\nu})_{i,\langle 0\rangle}^{(k-1)}$
end
$w_i^{n,l} = w_i^n - \frac{\Delta t}{\Delta x} (F_{i+1/2}^{n,l} - F_{i-1/2}^{n,l})$

CAT procedure [3] ($k = 2, \dots, \mathtt{r}$):
for $j = -p+1$ to $p$ do
    $\widetilde{w}_{i,j}^{(k-1)} = -Q_j^{(1)} \widetilde{\mathbf{f}}_{i,\langle 0\rangle}^{(k-2)}$
        $= -\frac{1}{\Delta x} \sum_{r=-p+1}^{p} \gamma_{p,r}^{1,j} \widetilde{f}_{i,r}^{(k-2)}$
    for $r = -p+1$ to $p$ do
        $(\mathfrak{f}_T)_{i,j}^{k-1,n+r} = f\left( w_{i+j}^n + \sum_{m=1}^{k-1} \frac{(r\Delta t)^m}{m!} \widetilde{w}_{i,j}^{(m)} \right)$
    end
    $\widetilde{f}_{i,j}^{(k-1)} = Q_0^{(k-1)} (\mathfrak{f}_T)_{i,j}^{k-1,\langle n\rangle}$
        $= \frac{1}{\Delta t^{k-1}} \sum_{r=-p+1}^{p} \gamma_{p,r}^{k-1,0} (\mathfrak{f}_T)_{i,j}^{k-1,n+r}$
end
```

4 Consistency Analysis

In this section, we show that the rDRKCATq-s methods are consistent. The order of consistency is, as to be expected, the minimum of the underlying Runge–Kutta order (q) and the order of the interpolation ($2p$). Let us make the following two important assumptions:

Assumption 1 We assume both f and w to be smooth functions in C^∞. Furthermore, we assume that Δt and Δx are asymptotically comparable in size, i.e.,

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

Throughout this section, we use the following notation to reduce the number of function arguments:

$$\partial_t^k f(w)_{i+j} \equiv \partial_t^k f(w(x_{i+j}, t^n)).$$

Whenever possible, similar notation is used for other functions. The time index n is only mentioned when necessary. We immediately state the main result and thereafter, in a successive form, deduce the necessary lemmas upon which its proof relies.

Theorem 1 *The consistency order of an explicit* rDRKCATq-s *method is given by* $\min(2p, q)$. *Here, q is the consistency order of the underlying MDRK method, while the stencil to update $w(x_i, t^n)$ is given by $\{i-p, i-p+1, \ldots, i+p\}$.*

Proof The proof relies on Lemmas that will be proven in the sequel. In Lemma 1, it is shown that the numerical flux difference gives the correct flux up to an order of $2p$. We can hence substitute the exact solution $w(x, t)$ into Eq. (17b), which immediately gives the requested result due to the fact that the Runge–Kutta update is an integration scheme of order $q + 1$:

$$\begin{aligned}
&w(x_i, t^{n+1}) - w(x_i, t^n) + \frac{\Delta t}{\Delta x}\left(F^n_{i+1/2} - F^n_{i-1/2}\right) \\
&\quad = w(x_i, t^{n+1}) - w(x_i, t^n) + \sum_{k=1}^{\mathrm{r}} \Delta t^k \sum_{l=1}^{\mathrm{s}} b_l^{(k)} \partial_x \partial_t^{k-1} f(w)_i^{n,l} + \mathcal{O}(\Delta x^{2p+1}) \\
&\quad = \mathcal{O}(\Delta t^{q+1}) + \mathcal{O}(\Delta x^{2p+1}).
\end{aligned}$$

□

Remark 6 Since the convergence order is $\min(2p, q)$, the optimal choice w.r.t. computational efficiency is to set $p = \lceil q/2 \rceil$. Hence, "rDRKCATq-s" does not contain the variable p.

Lemma 1 *The update numerical flux (18b) satisfies*

$$\frac{F^n_{i+1/2} - F^n_{i-1/2}}{\Delta x} = \sum_{k=1}^{\mathrm{r}} \Delta t^{k-1} \sum_{l=1}^{\mathrm{s}} b_l^{(k)} \partial_x \partial_t^{k-1} f(w)_i^{n,l} + \mathcal{O}(\Delta x^{2p}).$$

An analogous result holds for the stage flux (18a).

Proof From Lemmas 3 and 4, we obtain that for $k > 1$ and $l = 1, \ldots, \mathrm{s}$ there holds

$$\begin{aligned}
(\widetilde{f}^l)^{(k-1)}_{i,j} = \partial_t^{k-1} f(w)^{n,l}_{i+j} + \eta_p^{(k-1)} \left(R^{(k-1)}_{f,t}\right)^{n,l}_{i+j} \cdot \Delta t^{2p-k+1} \\
+ \xi^{(k-1)}_{p,j} \left(R^{(k-1)}_{f,x}\right)^{n,l}_{i} \cdot \Delta x^{2p-k+1} + \mathcal{O}(\Delta x^{2p-k+2}),
\end{aligned}$$

with $\eta_p^{(k-1)}$ and $\xi_{p,j}^{(k-1)}$ real-valued coefficients; and $R_{f,x}^{(k-1)}$, $R_{f,t}^{(k-1)}$ smooth functions of space and time. The above formula is put into use by substituting it into the numerical flux (18b):

$$\begin{aligned}
&\frac{\Lambda^{(0)}(\widetilde{\mathbf{f}}^{l})_{i,\langle 0\rangle}^{(k-1)} - \Lambda^{(0)}(\widetilde{\mathbf{f}}^{l})_{i-1,\langle 0\rangle}^{(k-1)}}{\Delta x} \\
&= \frac{\left(\Lambda^{(0)}\partial_t^{k-1} f(w)\right)_{i+1/2}^{n,l} - \left(\Lambda^{(0)}\partial_t^{k-1} f(w)\right)_{i-1/2}^{n,l}}{\Delta x} \\
&\quad + \Delta t^{2p-k+1} \cdot \eta_p^{(k-1)} \frac{\left(\Lambda^{(0)} R_{f,t}^{(k-1)}\right)_{i+1/2}^{n,l} - \left(\Lambda^{(0)} R_{f,t}^{(k-1)}\right)_{i-1/2}^{n,l}}{\Delta x} \\
&\quad + \Delta x^{2p-k+1} \cdot \frac{\left(R_{f,x}^{(k-1)}\right)_{i}^{n,l} - \left(R_{f,x}^{(k-1)}\right)_{i-1}^{n,l}}{\Delta x} \underbrace{\sum_{j=-p+1}^{p} \lambda_{p,j}^{0} \xi_{p,j}^{(k-1)}}_{\mathcal{O}(1)} \\
&\quad + \frac{1}{\Delta x}\mathcal{O}(\Delta x^{2p-k+2}) \\
&\overset{(15)}{=} \left(\mathcal{P}_i \partial_t^{k-1} f(w)\right)^{(1)} (x_i, t^{n,l}) \\
&\quad + \Delta t^{2p-k+1} \cdot \eta_p^{(k-1)} \left(\mathcal{P}_i R_{f,t}^{(k-1)}\right)^{(1)} (x_i, t^{n,l}) + \mathcal{O}(\Delta x^{2p-k+1}) \\
&= \partial_x \partial_t^{k-1} f(w)_i^{n,l} + \mathcal{O}(\Delta x^{2p-k+1}).
\end{aligned}$$

Please note that the term $\left(\left(R_{f,x}^{(k-1)}\right)_i^{n,l} - \left(R_{f,x}^{(k-1)}\right)_{i-1}^{n,l}\right)/\Delta x$ can be interpreted as a finite difference approximation of the derivative of the smooth function $R_{f,x}^{k-1}$ and therefore remains bounded, i.e., is $\mathcal{O}(1)$. □

Before we regard the consistency analysis of the CAT steps in Lemmas 3 and 4, the follwing identity on the difference coefficients is described.

Lemma 2 *Consider a local stencil index $j = -p+1, \dots, p$. Then there holds for $k = 1, \dots, 2p-1$:*

$$\sum_{r=-p+1}^{p} \gamma_{p,r}^{k,j} (r-j)^s = k!\delta_{s,k}, \quad s = 0, \dots, 2p-1.$$

The symbol $\delta_{s,k}$ here represents the Kronecker-delta function.

Proof We consider a mesh centered around $x_i = 0$ with spatial size $\Delta x = 1$. The operator $\mathcal{Q}_i$ exactly interpolates the polynomial function $\varphi(x) := (x-j)^s$ for $s = 0, \dots, 2p-1$ such that Eq. (10) becomes

$$(x-j)^s = \mathcal{Q}_i \varphi(x) = \sum_{r=-p+1}^{p} \ell_{p,r}(x)\varphi(r) = \sum_{r=-p+1}^{p} \ell_{p,r}(x)(r-j)^s.$$

Deriving the above relation k times in x and thereafter evaluating in $x = j$ gives the result. □

Now we can provide a consistency proof for the CAT procedure in Algorithm 1. To this purpose we establish the following notation for the exact Taylor approximation in time of order $k \in \mathbb{N}$,

$$T^k_{i,j}(\tau) := \sum_{m=0}^{k} \frac{\tau^m}{m!} \partial_t^m w(x_{i+j}, t^n). \tag{19}$$

It is the k-th order approximation of $w(x_{i+j}, t^n + \tau)$. The proof itself is built in a similar fashion as [3, Theorem 2] and [36, Proposition 1].

Lemma 3 *For $k = 1$ the steps of the CAT algorithm (Algorithm 1, right side) satisfy*

$$\begin{aligned}
\widetilde{w}^{(1)}_{i,j} &= \partial_t w_{i+j} + \xi^{(1)}_{p,j} \left(R^{(1)}_w\right)_i \cdot \Delta x^{2p-1} + \mathcal{O}(\Delta x^{2p}),\\
\widetilde{f}^{(1)}_{i,j} &= \partial_t f(w)_{i+j} + \eta^{(1)}_p \left(R^{(1)}_{f,t}\right)_{i+j} \cdot \Delta t^{2p-1} + \xi^{(1)}_{p,j} \left(R^{(1)}_{f,x}\right)_i \cdot \Delta x^{2p-1} + \mathcal{O}(\Delta x^{2p}),
\end{aligned}$$

with real-valued coefficients $\xi^{(1)}_{p,j}$, $\eta^{(1)}_p$; and smooth functions $\left(R^{(1)}_w\right)_i$, $\left(R^{(1)}_{f,t}\right)_{i+j}$, $\left(R^{(1)}_{f,x}\right)_i$. (Please note again that $(\cdot)_i$ stands for function-evaluation at $x = x_i$.)

Proof A straightforward computation on $\widetilde{w}^{(1)}_{i,j}$, using the Taylor expansion of $f(w)$, reveals that

$$\begin{aligned}
\widetilde{w}^{(1)}_{i,j} &= -Q^{(1)}_j \widetilde{\mathbf{f}}^{(0)}_{i,\langle 0\rangle} = -\frac{1}{\Delta x} \sum_{r=-p+1}^{p} \gamma^{1,j}_{p,r} \widetilde{f}^{(0)}_{i,r}\\
&= -\frac{1}{\Delta x} \sum_{r=-p+1}^{p} \gamma^{1,j}_{p,r} f(w(x_{i+r}, t^n))\\
&= -\frac{1}{\Delta x} \sum_{r=-p+1}^{p} \gamma^{1,j}_{p,r} \left(\sum_{s=0}^{2p} \frac{((r-j)\Delta x)^s}{s!} \partial_x^s f(w)_{i+j} + \mathcal{O}(\Delta x^{2p+1}) \right)\\
&= -\frac{1}{\Delta x} \sum_{s=0}^{2p-1} \frac{\Delta x^s}{s!} \partial_x^s f(w)_{i+j} \left(\sum_{r=-p+1}^{p} \gamma^{1,j}_{p,r} (r-j)^s \right)\\
&\quad - \left[\frac{1}{(2p)!} \left(\sum_{r=-p+1}^{p} \gamma^{1,j}_{p,r} (r-j)^{2p} \right) \partial_x^{2p} f(w)_i + \mathcal{O}(\Delta x) \right] \Delta x^{2p-1}\\
&\quad + \mathcal{O}(\Delta x^{2p})\\
&\overset{\text{Lemma 2}}{=} -\partial_x f(w)_{i+j} + \xi^{(1)}_{p,j} \left(R^{(1)}_w\right)_i \cdot \Delta x^{2p-1} + \mathcal{O}(\Delta x^{2p})\\
&= \partial_t w_{i+j} + \xi^{(1)}_{p,j} \left(R^{(1)}_w\right)_i \cdot \Delta x^{2p-1} + \mathcal{O}(\Delta x^{2p}),
\end{aligned}$$

in which we define $\xi^{(1)}_{p,j}$ and $\left(R^{(1)}_w\right)_i$ by

$$\xi^{(1)}_{p,j} := \frac{-1}{(2p)!} \sum_{r=-p+1}^{p} \gamma^{1,j}_{p,r} (r-j)^{2p}, \quad \left(R^{(1)}_w\right)_i := \partial_x^{2p} f(w)_i.$$

The succeeding step of the method evaluates the flux f in an approximate Taylor series. We find,

$$
\begin{aligned}
(\mathfrak{f}_T)_{i,j}^{1,n+r} &= f\left(w(x_{i+j}, t^n) + (r\Delta t)\widetilde{w}_{i,j}^{(1)}\right) \\
&\overset{(19)}{=} f\left(T_{i,j}^1(r\Delta t) + (r\Delta t)\xi_{p,j}^{(1)}\big(R_w^{(1)}\big)_i \cdot \Delta x^{2p-1} + \mathcal{O}(\Delta x^{2p+1})\right) \\
&= (f \circ T_{i,j}^1)(r\Delta t) \\
&\quad + (f' \circ T_{i,j}^1)(r\Delta t) \cdot \left[(r\Delta t)\xi_{p,j}^{(1)}\big(R_w^{(1)}\big)_i\right] \cdot \Delta x^{2p-1} + \mathcal{O}(\Delta x^{2p+1}) \\
&= (f \circ T_{i,j}^1)(r\Delta t) \\
&\quad + (f' \circ T_{i,j}^1)(0) \cdot \left[(r\Delta t)\xi_{p,j}^{(1)}\big(R_w^{(1)}\big)_i\right] \cdot \Delta x^{2p-1} + \mathcal{O}(\Delta x^{2p+1}) \\
&= (f \circ T_{i,j}^1)(r\Delta t) + f'(w)_i \left[(r\Delta t)\xi_{p,j}^{(1)}\big(R_w^{(1)}\big)_i\right] \cdot \Delta x^{2p-1} + \mathcal{O}(\Delta x^{2p+1}).
\end{aligned}
$$

Consequently, we can find for the temporal interpolation of the fluxes:

$$
\begin{aligned}
\widetilde{f}_{i,j}^{(1)} &= Q_0^{(1)} (\mathfrak{f}_T)_{i,j}^{1,\langle n\rangle} = \frac{1}{\Delta t} \sum_{r=-p+1}^{p} \gamma_{p,r}^{1,0} (\mathfrak{f}_T)_{i,j}^{1,n+r} \\
&= \frac{1}{\Delta t} \sum_{r=-p+1}^{p} \gamma_{p,r}^{1,0} \left[\sum_{s=0}^{2p} \left(\frac{(r\Delta t)^s}{s!} \frac{\mathrm{d}^s (f \circ T_{i,j}^1)}{\mathrm{d}\tau^s}(0)\right) + \mathcal{O}(\Delta x^{2p+1})\right] \\
&\quad + f'(w)_i \underbrace{\left(\sum_{r=-p+1}^{p} \gamma_{p,r}^{1,0} r\right)}_{=1,\text{ Lemma } 2} \xi_{p,j}^{(1)}\big(R_w^{(1)}\big)_i \cdot \Delta x^{2p-1} + \mathcal{O}(\Delta x^{2p}) \\
&= \frac{1}{\Delta t} \sum_{s=0}^{2p-1} \frac{\Delta t^s}{s!} \underbrace{\left(\sum_{r=-p+1}^{p} \gamma_{p,r}^{1,0} r^s\right)}_{=\delta_{1,s},\text{ Lemma } 2} \frac{\mathrm{d}^s (f \circ T_{i,j}^1)}{\mathrm{d}\tau^s}(0) \\
&\quad + \underbrace{\frac{1}{(2p)!} \left(\sum_{r=-p+1}^{p} \gamma_{p,r}^{1,0} r^{2p}\right)}_{=:\eta_p^{(1)}} \underbrace{\frac{\mathrm{d}^{2p} (f \circ T_{i,j}^1)}{\mathrm{d}\tau^{2p}}(0)}_{=:\left(R_{f,t}^{(1)}\right)_{i+j}} \cdot \Delta t^{2p-1} \\
&\quad + \xi_{p,j}^{(1)} \underbrace{f'(w)_i \big(R_w^{(1)}\big)_i}_{=:\left(R_{f,x}^{(1)}\right)_i} \cdot \Delta x^{2p-1} + \mathcal{O}(\Delta x^{2p}) \\
&= \frac{\mathrm{d}(f \circ T_{i,j}^1)}{\mathrm{d}\tau}(0) + \eta_p^{(1)}\big(R_{f,t}^{(1)}\big)_{i+j} \cdot \Delta t^{2p-1} + \xi_{p,j}^{(1)}\big(R_{f,x}^{(1)}\big)_i \cdot \Delta x^{2p-1} + \mathcal{O}(\Delta x^{2p}) \\
&= \partial_t f(w)_{i+j} + \eta_p^{(1)}\big(R_{f,t}^{(1)}\big)_{i+j} \cdot \Delta t^{2p-1} + \xi_{p,j}^{(1)}\big(R_{f,x}^{(1)}\big)_i \cdot \Delta x^{2p-1} + \mathcal{O}(\Delta x^{2p}).
\end{aligned}
$$

□

Via induction one can generalize this result.

Lemma 4 *For $k = 2, \ldots, 2p-1$ the steps of the CAT algorithm (Alg. 1, right side) satisfy*

$$
\widetilde{w}_{i,j}^{(k)} = \partial_t^k w_{i+j} + \xi_{p,j}^{(k)}\big(R_w^{(k)}\big)_i \cdot \Delta x^{2p-k} + \mathcal{O}(\Delta x^{2p-k+1}),
$$

$$\begin{aligned}\widetilde{f}_{i,j}^{(k)} &= \partial_t^k f(w)_{i+j} + \eta_p^{(k)} \big(R_{f,t}^{(k)}\big)_{i+j} \cdot \Delta t^{2p-k} \\ &\qquad + \xi_{p,j}^{(k)} \big(R_{f,x}^{(k)}\big)_i \cdot \Delta x^{2p-k} + \mathcal{O}(\Delta x^{2p-k+1}),\end{aligned}$$

with

$$\begin{aligned}&\eta_p^{(k)} := \frac{1}{(2p)!}\left(\sum_{r=-p+1}^{p} \gamma_{p,r}^{k,0} r^{2p}\right), \qquad \xi_{p,j}^{(k)} := -\sum_{r=-p+1}^{p} \gamma_{p,r}^{1,j} \xi_{p,r}^{(k-1)}, \\ &\big(R_w^{(k)}\big)_i := \big(f'(w)_i\big)^{k-1} \partial_x^{2p} f(w)_i, \quad \big(R_{f,t}^{(k)}\big)_{i+j} := \frac{\mathrm{d}^{2p}(f \circ T_{i,j}^k)}{\mathrm{d}\tau^{2p}}(0), \\ &\big(R_{f,x}^{(k)}\big)_i := f'(w)_i \big(R_w^{(k)}\big)_i.\end{aligned}$$

5 Von Neumann Stability of $\mathtt{r}$DRKCATq-$\mathtt{s}$ Methods

The original CAT procedure was developed with the intention to create a scheme which linearly reduces back to high-order Lax–Wendroff methods. As a consequence, the scheme is CFL-1 stable for linear equations [3, Theorem 1]. In this section, we discuss the stability properties of the $\mathtt{r}$DRKCATq-$\mathtt{s}$ scheme presented in this work.

Theorem 2 (MDRK-LW scheme) *Explicit $\mathtt{r}$DRKCATq-$\mathtt{s}$ methods for the linear advection flux $f(w) = \alpha w$ reduce to the numerical scheme*

$$w_j^{n,l} = w_j^n + \sum_{k=1}^{\mathtt{r}} (-1)^k \alpha^k \Delta t^k \left(\sum_{\nu=1}^{l-1} a_{l\nu}^{(k)} P^{(k)} \mathbf{w}_{\langle j\rangle}^{n,\nu}\right), \qquad l = 1, \ldots, \mathtt{s}, \tag{20a}$$

$$w_j^{n+1} = w_j^n + \sum_{k=1}^{\mathtt{r}} (-1)^k \alpha^k \Delta t^k \left(\sum_{l=1}^{\mathtt{s}} b_l^{(k)} P^{(k)} \mathbf{w}_{\langle j\rangle}^{n,l}\right), \tag{20b}$$

with $P^{(k)}\mathbf{w}_{\langle j\rangle}^{n,l}$ the centered difference approximation of the k-th spatial derivative.

The proof is similar to [3, Theorem 1] and is hence left out. In this form it is possible to perform a von Neumann stability analysis (see for example [25,32]). That is, we fill in the Fourier mode $w_j^n = \mathcal{W}(t^n)e^{\mathrm{i}\kappa x_j}$ with wave number $\kappa \in \mathbb{Z}$ and search for the the amplification factors via the relations

$$\mathcal{W}(t^{n,l}) := g^{n,l}(\kappa)\mathcal{W}(t^n) \quad \text{and} \quad \mathcal{W}(t^{n+1}) := g(\kappa)\mathcal{W}(t^n).$$

Doing so gives an additional recurrence relation.

Proposition 1 *The amplification factors obtained from a von Neumann analysis on the MDRK-LW scheme Eq. (20) are defined by the recurrence relations*

$$g^{n,l}(\kappa) = 1 + \sum_{k=1}^{\mathtt{r}} (-1)^k \sigma^k\, \mathrm{P}^{(k)}(\kappa) \left(\sum_{\nu=1}^{l-1} a_{l\nu}^{(k)} g^{n,\nu}(\kappa)\right), \qquad l = 1, \ldots, \mathtt{s}, \tag{21a}$$

$$g(\kappa) = 1 + \sum_{k=1}^{\mathtt{r}} (-1)^k \sigma^k\, \mathrm{P}^{(k)}(\kappa) \left(\sum_{l=1}^{\mathtt{s}} b_l^{(k)} g^{n,l}(\kappa)\right), \tag{21b}$$

Table 2 Amplification factors $g(\kappa)$ of the considered MDRK schemes (see Appendix A) and the corresponding CFL values σ^* up to four decimals obtained from $|g(\kappa)| \le 1$

Scheme	$g(\kappa)$	CFL
2DRKCAT3-2	$1 - \sigma\, \mathrm{P}^{(1)} + \frac{1}{2}\sigma^2 \left(\frac{2}{3}\mathrm{P}^{(1)}\mathrm{P}^{(1)} + \frac{1}{3}\mathrm{P}^{(2)}\right) - \frac{1}{6}\sigma^3\, \mathrm{P}^{(1)}\mathrm{P}^{(2)}$	1.2954
2DRKCAT4-2	$1 - \sigma\, \mathrm{P}^{(1)} + \frac{1}{2}\sigma^2\, \mathrm{P}^{(2)} - \frac{1}{6}\sigma^3\, \mathrm{P}^{(1)}\mathrm{P}^{(2)} + \frac{1}{24}\sigma^4\, \mathrm{P}^{(2)}\mathrm{P}^{(2)}$	1.4718
2DRKCAT5-3	$1 - \sigma\, \mathrm{P}^{(1)} + \frac{1}{2}\sigma^2\, \mathrm{P}^{(2)} - \frac{1}{6}\sigma^3\, \mathrm{P}^{(1)}\mathrm{P}^{(2)} + \frac{1}{24}\sigma^4\, \mathrm{P}^{(2)}\mathrm{P}^{(2)}$ $-\frac{1}{120}\sigma^5\, \mathrm{P}^{(1)}\mathrm{P}^{(2)}\mathrm{P}^{(2)} + \frac{1}{600}\sigma^6\, \mathrm{P}^{(2)}\mathrm{P}^{(2)}\mathrm{P}^{(2)}$	1.0619
3DRKCAT5-2	$1 - \sigma\, \mathrm{P}^{(1)} + \frac{1}{2}\sigma^2\, \mathrm{P}^{(2)} - \frac{1}{6}\sigma^3\, \mathrm{P}^{(3)} + \frac{1}{24}\sigma^4\, \mathrm{P}^{(1)}\mathrm{P}^{(3)}$ $-\frac{1}{120}\sigma^5\, \mathrm{P}^{(2)}\mathrm{P}^{(3)} + \frac{1}{900}\sigma^6\, \mathrm{P}^{(3)}\mathrm{P}^{(3)}$	0.4275
3DRKCAT7-3	$1 - \sigma\, \mathrm{P}^{(1)} + \frac{1}{2}\sigma^2\, \mathrm{P}^{(2)} - \frac{1}{6}\sigma^3\, \mathrm{P}^{(3)} + \frac{1}{24}\sigma^4\, \mathrm{P}^{(1)}\mathrm{P}^{(3)}$ $-\frac{1}{120}\sigma^5\, \mathrm{P}^{(2)}\mathrm{P}^{(3)} + \frac{1}{720}\sigma^6\, \mathrm{P}^{(3)}\mathrm{P}^{(3)} - \frac{1}{5040}\sigma^7\, \mathrm{P}^{(1)}\mathrm{P}^{(2)}\mathrm{P}^{(3)}$ $+\left(\frac{1}{23520} - \frac{\sqrt{2}}{70560}\right)\sigma^8\, \mathrm{P}^{(2)}\mathrm{P}^{(3)}\mathrm{P}^{(3)} - \left(\frac{11}{1481760} - \frac{\sqrt{2}}{246960}\right)\sigma^9\, \mathrm{P}^{(3)}\mathrm{P}^{(3)}\mathrm{P}^{(3)}$	0.2300
4DRKCAT6-2	$1 - \sigma\, \mathrm{P}^{(1)} + \frac{1}{2}\sigma^2\, \mathrm{P}^{(2)} - \frac{1}{6}\sigma^3\, \mathrm{P}^{(3)} + \frac{1}{24}\sigma^4\, \mathrm{P}^{(4)}$ $-\frac{1}{120}\sigma^5\, \mathrm{P}^{(1)}\mathrm{P}^{(4)} + \frac{1}{720}\sigma^6\, \mathrm{P}^{(2)}\mathrm{P}^{(4)} - \frac{1}{6480}\sigma^7\, \mathrm{P}^{(3)}\mathrm{P}^{(4)} + \frac{1}{77760}\sigma^8\, \mathrm{P}^{(4)}\mathrm{P}^{(4)}$	0.8563

with wave number $\kappa \in \mathbb{Z}$, $\sigma := \frac{\alpha \Delta t}{\Delta x}$ the corresponding CFL number and

$$\mathrm{P}^{(k)}(\kappa) := \sum_{r=-p}^{p} \delta^k_{p,r} e^{\mathrm{i} r \kappa \Delta x}. \tag{22}$$

The term $\mathrm{P}^{(k)}(\kappa)$ can be interpreted as the k-th derivative of the centered $(2p+1)$-point Lagrangian interpolation Eq. (11a) using Fourier basis $\left\{e^{\mathrm{i}l\chi} \mid l \in \mathbb{Z}\right\}$ with the grid frequency $\chi = \kappa \Delta x$.

In Table 2 we display the amplification factors $g(\kappa)$ for the considered MDRK schemes summarized earlier in Fig. 1. Of main interest w.r.t. to these amplification factors is to obtain a critical CFL value $\sigma^* \in \mathbb{R}^+$ such that

$$|g(\kappa)| \le 1 \quad \Leftrightarrow \quad \frac{\alpha \Delta t}{\Delta x} \le \sigma^*.$$

We have used the Symbolic Math Toolbox in MATLAB [20] along with a bisection method on the CFL variable σ in Eq. (21b) to numerically obtain a σ^*:

- In our approach we take $\Delta x = 1$; only the frequency of $g(\kappa)$ is influenced by Δx, there is no change in absolute value. Assume we would compare mesh sizes Δx and $\Delta\tilde{x}$, then

$$\mathrm{P}^{(k)}(\kappa) = \sum_{r=-p}^{p} \delta^k_{p,r} e^{\mathrm{i} r \kappa \Delta x} = \sum_{r=-p}^{p} \delta^k_{p,r} e^{\mathrm{i} r \left(\kappa \frac{\Delta x}{\Delta \tilde{x}}\right) \Delta \tilde{x}} = \mathrm{P}^{(k)}(\widetilde{\kappa}),$$

  where $\widetilde{\kappa} := \kappa \frac{\Delta x}{\Delta \tilde{x}}$. Thus behavior of $g(\kappa)$ is the same up to a recalibration of the frequency space.
- Via the toolbox, the value $\max |g(\kappa)|$ is calculated on a uniform 1000-cell mesh of $\kappa \in [-\pi, \pi]$. This domain suffices for this purpose, since for $\Delta x = 1$ all terms $\mathrm{P}^{(k)}(\kappa)$ in Eq. (22) are 2π-periodic.

The critical CFL values σ^* up to four decimals are shown in Table 2. Notice that the two-derivative schemes improve the linear stability, whereas the other schemes reduce the stability compared to the original CAT method. To put this observation into perspective, let us point out that the CAT algorithm uses $2p$ derivatives, and thus is based on high-order Lax–Wendroff methods with $\mathtt{r}$ even. If, only for the sake of discussion, we take *uneven* $\mathtt{r}$, we get another picture. For $\mathtt{r} = 1$, we obtain the forward-time central-space scheme, which is infamous for being unconditionally unstable [17] and for odd $\mathtt{r} > 1$, we obtain CFL numbers smaller than one. Hence, we can make some important observations:

– Most importantly we can conclude that a method of lines (MOL) viewpoint is inadequate. Solely regarding the stability regions $\mathcal{R}$ in Fig. 1 would give the idea that the 3DRKCAT7-3 scheme provides the best stability. This is clearly not the case since the even-derivative schemes are shown to be better in terms of stability.
– Choosing even order derivatives gives better results than choosing an odd number of derivatives, i.e. the two- and four-derivative schemes show better stability than the three-derivative schemes. This is in very good agreement with the observations on the original CAT method.
– In contrast to Taylor-methods, where only the number of derivatives can be prescribed, MDRK methods have more free parameters, such as the number of derivatives *and* the number of stages. We observe that stages and derivatives highly influence the stability properties of the MDRKCAT method. This allows the identification of well-suited MDRK schemes and gives more flexibility compared to original Taylor-methods.

6 Numerical Results

In this section, we show numerical results validating our analytical findings. By means of several continuous test cases ranging from scalar PDEs to the system of Euler equations, we show that the expected orders of convergence are obtained. For brevity, we do not include flux limiting techniques to this work; hence, we avoid setups where shock formation occurs. Note that flux limiting can be incorporated in a straightforward way as in [3].

The measure for the accuracy in this section is the scaled l_1-error at time $t^N \equiv T_{\text{end}}$

$$\|\mathbf{w}(T_{\text{end}}) - \mathbf{w}^N\|_1 := \Delta x \sum_{i=1}^{M} |w(x_i, T_{\text{end}}) - w_i^N|,$$

$\mathbf{w} : t \mapsto \mathbf{w}(t)$ being a function of time returning a vector of exact solution values (or a reference solution) in the nodes $x_1, \ldots, x_M$, and $\mathbf{w}^N$ being the vector of approximations w_i^N at t^N. For systems, the sum of the l_1-errors corresponding to the separate solution variables is considered. All displayed convergence plots begin at $M = 8$ cells and double the amount of cells with each iteration. In order to enforce the adopted CFL value σ, the local eigenvalues $\lambda_{\text{eig},i}^n$ of the Jacobian w.r.t. w_i^n are computed. By means of the relation

$$\Delta t^n := \frac{\sigma \Delta x}{\max_i |\lambda_{\text{eig},i}^n|}$$

the timestep is then computed. As the maximum eigenvalue in the computational domain varies over time, a non-constant timestep is prescribed. This highlights the ability of the novel method to use varying timestep sizes, see Rem. 1.

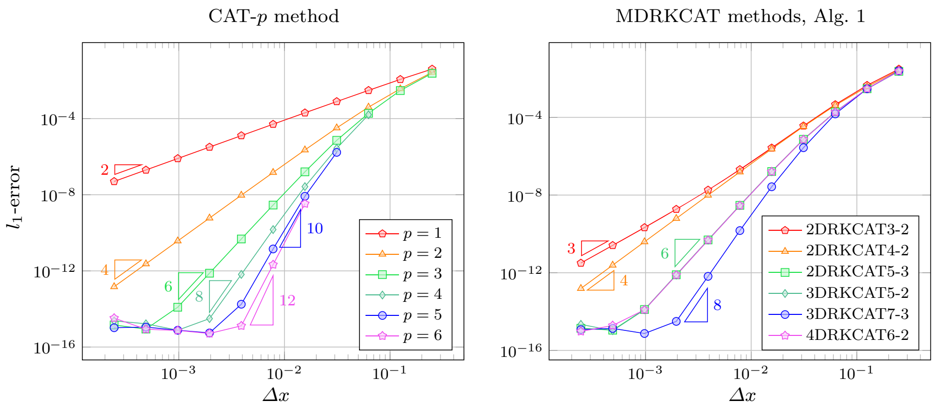


Fig. 2 Convergence order of the CAT and explicit MDRKCAT methods applied to Burgers equation on the cosine wave $w_0(x) = \frac{1}{4}\cos(\pi x)$ up to $T_{\text{end}} = 0.8$ with CFL $\sigma = 0.5$. The 2DRKCAT5-3, 3DRKCAT5-2 and 4DRKCAT6-2 schemes behave in a very alike manner. For large Δt and large p, we have observed stability problems for the CAT method. Those divergent results of the CAT method have been omitted in the convergence plot

6.1 Burgers Equation

First, we consider Burgers equation

$$\partial_t w + \partial_x \left(\frac{w^2}{2}\right) = 0,$$

with the cosine-wave initial condition with periodic boundary conditions

$$w(x,0) = \frac{1}{4}\cos(\pi x) \qquad \text{on} \quad x \in [0,2]\,, \tag{23}$$

to certify the accuracy $\min(2p, q)$ obtained in Thm. 1. Since the cosine-wave Eq. (23) has both positive and negative values, the characteristic lines must cross and a shock is formed at some point. The breaking time of the wave [17,35] is at $t^* = \frac{4}{\pi}$. Hence we set the final time to $T_{\text{end}} = 0.8$, well before shock formation.

Using the characteristic lines solution, l_1-errors have been calculated for the CAT-p methods ($p = 1, \ldots, 6$) and the MDRKCAT methods (Appendix A). The results with CFL $\sigma = 0.5$ are visualized in Fig. 2. All expected convergence orders are obtained; order $2p$ for the CAT-p methods and at least order q of the MDRK schemes. The schemes 2DRKCAT5-3, 3DRKCAT5-2 and 3DRKCAT7-3 behave better than expected. This is caused by the fact that the spatial order of accuracy is higher than the temporal one; and spatial errors dominate the overall behavior at least for 'large' Δt. We have observed similar behavior also for other schemes where $2p > q$. Given that CAT methods have been designed as a natural generalization of Lax–Wendroff methods with an even-order accuracy, odd-order MDRK schemes take advantage here. We expect this behavior to become more apparent when computing with finer machine precision. Convergence plots such as in Fig. 2 will then manifest as a stretched out S-curve.

Further, we notice that for higher values of p and larger values Δt, the CAT methods tend to be less stable. Unstable results have been left out in Fig. 2 for $p = 4$, 5 and 6. Even though CAT methods have been shown to be linearly stable under a CFL-1 condition [3], rapid divergence is observed well before shocks are formed for larger values of p at regions where

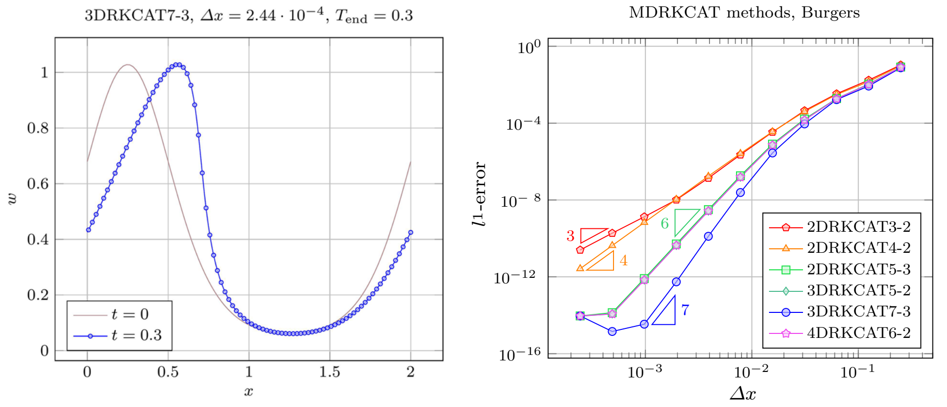


Fig. 3 Burgers equation applied to $w_0(x) = \frac{1}{4}e^{\cos(\pi x)+\sin(\pi x)}$ up to $T_{\text{end}} = 0.3$ with CFL $\sigma = 0.5$. Left: 3DRKCAT7-3 solution using $M = 8192$ cells, less nodal points are shown for better visual distinctness. Right: convergence order of the explicit MDRKCAT methods

the derivative of the solution $w'(x)$ is large in absolute value. The MDRKCAT methods used in this work do not suffer this fate.

Next, we perform a similar study having the initial solution,

$$w(x, 0) = \frac{1}{4}e^{\cos(\pi x)+\sin(\pi x)} \qquad \text{on} \quad x \in [0, 2]\,,$$

with periodic boundary conditions. Having a steeper peak than the cosine (23), the breaking time will be earlier. We find $t^* = \frac{4}{\pi e}$, and choose $T_{\text{end}} = 0.3$ accordingly.

In Fig. 3 the final solution at $T_{\text{end}} = 0.3$ is visualized and accuracy is studied in the convergence plots, solely focused on the MDRKCAT method. The behavior is very similar to the previous case, except that the 3DRKCAT7-3 scheme is driven back faster to order 7.

6.2 Buckley–Leverett Equation

Next, we consider the Buckley–Leverett flux [17],

$$f(w) = \frac{4w^2}{4w^2 + (1-w)^2}.$$

This flux is non-convex and introduces more nonlinearities compared to the Burgers flux. We consider the initial condition

$$w(x, 0) = 1 - \frac{3}{4}\cos^2\left(\frac{\pi}{2}x\right) \qquad \text{on} \quad x \in [-1, 1].$$

The typical Buckley–Leverett profile consists of a shock wave followed directly by a rarefaction wave. We set $T_{\text{end}} = 0.1$ to remain continuous and be able to calculate the exact solution via its characteristics. The solution and convergence plots are visualized in Fig. 4 with CFL $\sigma = 0.5$. We notice that the numerical solution tends toward the expected Buckley–Leverett profile. All schemes converge with the expected accuracy in a similar way as for Burgers equation.

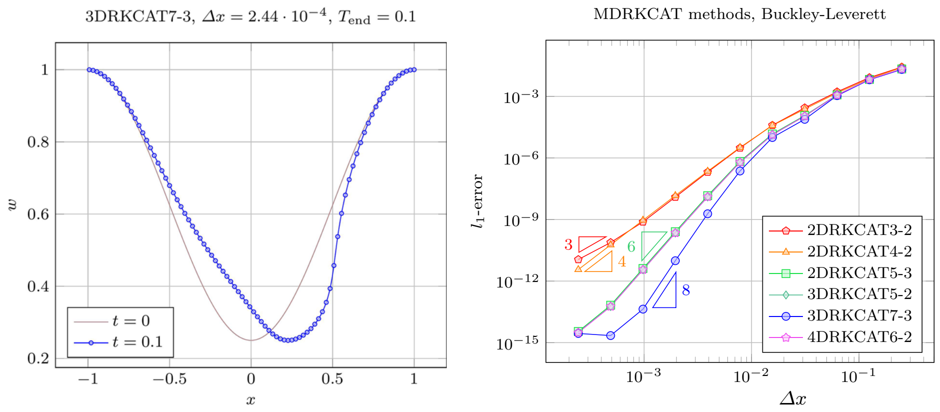


Fig. 4 Buckley–Leverett equation applied to $w_0(x) = 1 - \frac{3}{4}\cos^2\left(\frac{\pi}{2}x\right)$ up to $T_{\text{end}} = 0.1$ with CFL $\sigma = 0.5$. Left: 3DRKCAT7-3 solution using $M = 8192$ cells, less nodal points are shown for better visual distinctness. Right: convergence order of the explicit MDRKCAT methods

6.3 One-Dimensional Euler Equations

Finally, we consider the Euler equations of gas dynamics

$$\partial_t w + \partial_x f(w) = 0,$$

in which

$$w = \begin{pmatrix} \rho \\ \rho u \\ E \end{pmatrix}, \quad f(w) = \begin{pmatrix} \rho u \\ \rho u^2 + p \\ u(E + p) \end{pmatrix},$$

with ρ the density, u the velocity, E the energy and p the pressure. The system is closed via the equation of state for an ideal gas

$$p = (\gamma - 1)\left(E - \frac{1}{2}\rho u^2\right),$$

with γ being the ratio of specific heats, assumed to be 1.4 [17,35].

First we initialize the primitive variables (ρ, u, p) such that the Euler equations describe the linear advection of a density profile. To this end, we take

$$\rho(x, 0) = 1 + 0.3\sin(\pi x) \qquad \text{on} \quad x \in [0, 4]\,,$$

and set both $u(x, 0)$ and $p(x, 0)$ to be one. Periodic boundary conditions are used and T_{end} is set to 0.8. In Fig. 5 the corresponding convergence plots are displayed with CFL $\sigma = 0.5$. Immediately starting from the coarsest meshes the expected convergence orders are obtained.

Secondly, we consider the initial condition

$$w(x, 0) = \frac{1}{4}\begin{pmatrix} 3 \\ 1 \\ 3 \end{pmatrix} + \frac{\sin(\pi x)}{2}\begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} \qquad \text{on} \quad x \in [0, 2],$$

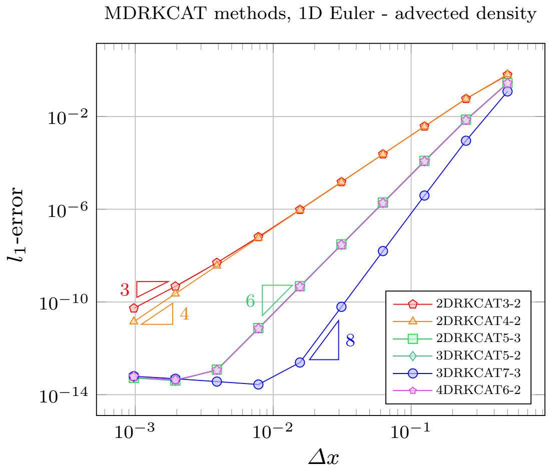


Fig. 5 Convergence order of the explicit MDRKCAT methods applied to $\rho_0(x) = 1 + 0.3\sin(\pi x)$, $u_0(x) = p_0(x) = 1$ on $x \in [0, 4]$ up to $T_{\text{end}} = 0.8$ with $\sigma = 0.5$

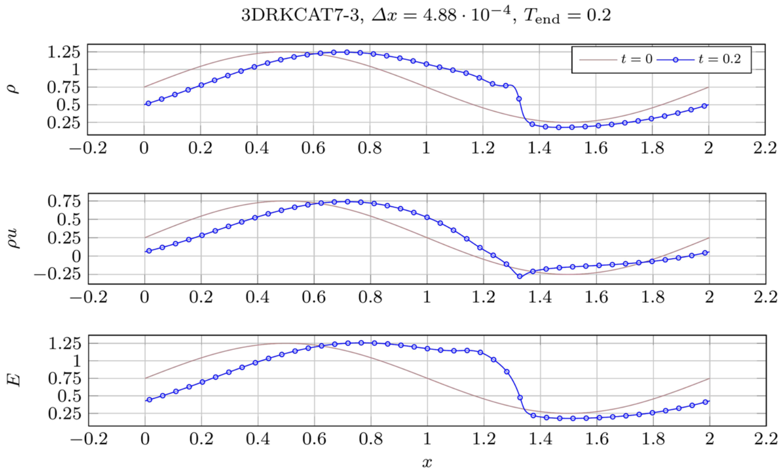


Fig. 6 MDRKCAT 1D Euler solution of $\rho_0(x) = 0.75 + 0.5\sin(\pi x)$, $(\rho u)_0(x) = 0.25 + 0.5\sin(\pi x)$ and $E_0(x) = 0.75 + 0.5\sin(\pi x)$ up to $T_{\text{end}} = 0.2$ with $\sigma = 0.5$. The 3DRKCAT7-3 scheme has been used on $M = 4096$ cells; less nodal points are shown for better visual distinctness

with periodic boundary conditions and $T_{\text{end}} = 0.2$ for w to remain continuous. In Fig. 6 the solution is visualized for the 3DRKCAT7-3 scheme with $p = 4$, CFL $\sigma = 0.5$ on $M = 4096$ cells.

In order to inspect accuracy, a reference solution has been computed via a discontinuous Galerkin (DG) method. We have used third-order polynomials in space, and a third-order strong-stability-preserving Runge–Kutta method in time [12]. The reference computation was executed on 10240 cells with a CFL number of $\sigma = 0.15$.

Convergence plots have been generated in Fig. 7 using CFL $\sigma = 0.15$. All expected orders were obtained. For smaller Δx, the l_1-error converges towards approximately $2 \cdot 10^{-11}$, which is the accuracy of the reference DG solution.

Akin to the earlier cases, a CFL value of $\sigma = 0.5$ was attempted for the construction of the convergence plots. However not all simulations were stable, more specifically the 3DRKCAT7-3 scheme using $p = 4$ diverged for $M = 8$, 16, 32 and 64 cells.

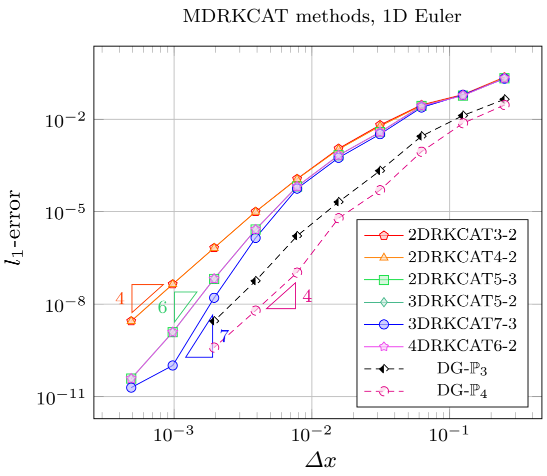


Fig. 7 Convergence order of the explicit MDRKCAT methods applied to $\rho_0(x) = 0.75 + 0.5\sin(\pi x)$, $(\rho u)_0(x) = 0.25 + 0.5\sin(\pi x)$ and $E_0(x) = 0.75 + 0.5\sin(\pi x)$ up to $T_{\text{end}} = 0.2$ with $\sigma = 0.15$. Very similar behavior can be seen between the 2DRKCAT3-2 and the 2DRKCAT4-2 schemes. The same can be said for the 5th order MDRKCAT schemes and 4DRKCAT6-2. For comparison, l_1-errors of a DG code with basis functions in $\mathbb{P}_3$ and $\mathbb{P}_4$ and CFL $\sigma = 0.1$ are visualized

In order to better grasp the efficiency of the MDRKCAT methods, in the same Fig. 7 convergence plots have been generated by means of a DG code that uses polynomial basis functions in $\mathbb{P}_3$ and $\mathbb{P}_4$ for each cell respectively. A fourth order SSP-RK scheme [31, order 4, p.21] has been used as time integrator. The same amount of cells $M = 8, \ldots, 1024$ has been used as for the MDRKCAT runs, the CFL $\sigma = 0.1$ with a maximum eigenvalue of 1.5 so that $\Delta t = \Delta x \frac{\sigma}{1.5}$. The l_1-errors, computed at cell-midpoints, have been generated relative to the earlier mentioned order three SSP-DG reference solution.

Overall, the MDRKCAT methods compare well with the DG solutions. A large discrepancy can be noticed in the manner at which the expected convergence order is achieved; the MDRKCAT methods gradually head toward order $\min(2p, q)$ with each refinement, whereas the DG solvers achieve convergence already going from 32 to 64 cells. This is to be expected: By definition the MDRKCAT methods only approximate the time derivatives of the flux $\partial_t^{k-1} f(w)$. Hence the achieved accuracy is intertwined with the mesh resolution of the problem at hand. For a lower amount of cells M the numerical flux $F^n_{i+1/2}$ at the faces can thus not be an accurate representation, whereas the DG solvers calculate the fluxes at the half-way points $i + 1/2$ on the basis of the exact flux $f(w)$. As soon as enough cells M are used to finely represent the initial data, full advantage can be taken of the CAT method.

Moreover, the difference between the methods should be brought into perspective by studying the amount of *effective* spatial degrees of freedom (DOF) and the *effective* spatial size that influences the order of accuracy. DG methods make use of numerical integration points on each cell for the integration of the solution variable multiplied with the chosen basis functions [7]. This illustrates why the DG solvers more quickly capture the expected convergence order and why a direct comparison of the DG schemes and the MDRKCAT schemes is difficult in Fig. 7. The actual amount of spatial DOF used by $\mathtt{r}$DRKCATq-$\mathtt{s}$ schemes is $(2p + 1)M$; each node uses its own local stencil in the calculations. However, as explained in [3], the local stencils are merely a manner to assure that the CAT methods linearly reduce back to Lax–Wendroff schemes. The same accuracy is achieved by the approximate Taylor methods in [36] of which the CAT procedure is established. Summing up, we can conclude that the novel $\mathtt{r}$DRKCATq-$\mathtt{s}$ schemes compare well with a state-of-the-art DG solver in terms of accuracy.

7 Conclusion and Outlook

In this paper we have formulated a family of Jacobian-free multistage multiderivative solvers for hyperbolic conservation laws, so-called MDRKCAT methods. Following the Compact Approximate Taylor (CAT) method in [3], instead of computing the exact flux derivative expressions, local approximations for the time derivatives of the fluxes are obtained recursively. There are many advantages by virtue of this procedure: no costly symbolic computations are needed; and we hope that many multiderivative Runge–Kutta (MDRK) schemes can now actually be of practical use.

Both theoretically and numerically it is proven that the desired convergence order $\min(2p, q)$ is achieved, $2p$ being the spatial order and q the temporal order. Universally among the different test cases the spatial accuracy is seen to be dominant. A comparison with SSP-DG methods for the Euler equations shows that MDRKCAT methods compare well with state-of-the-art schemes in terms of accuracy.

A von Neumann analysis revealed that the stability of the MDRKCAT methods depends heavily on the number of stages and the underlying high-order Lax–Wendroff method. The latter one solely utilizes centered differences for the spatial discretization. Consequently, odd-derivative Runge–Kutta schemes seem less adequate in conjuction with the CAT algorithm.

In the future, there are two main routes to follow: extend and apply the scheme to more challenging settings and to further examine the stability properties of the novel scheme. Concerning more challenging settings the investigation of multidimensional hyperbolic conservation laws with (possibly) unstructured meshes and parabolic PDEs with viscous effects are attractive. In order to accomplish such extensions it might be interesting to combine MDRKCAT methods with DG techniques [27]. Presumably, also implicit MDRK schemes need to be considered to take care of the diffusive effects. A possible starting point could be the implicit variant of the approximate Taylor methods, which have been recently developed for ODEs in [2]. Concerning the stability properties of the scheme one could think of exploring more types of MDRK schemes, possibly with SSP properties [6,11]. Moreover, at the same time, it will be possible to identify more efficient schemes.

Funding J. Zeifang was funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) through the Project No. 457811052. The HPC-resources and services used in this work were provided by the VSC (Flemish Supercomputer Center), funded by the Research Foundation-Flanders (FWO) and the Flemish Government.

Availability of Data and Materials The datasets generated and/or analyzed during the current study are available from the corresponding author on reasonable request mailto: jeremy.chouchoulis@uhasselt.be; jeremy.chouchoulis@uhasselt.be.

Declarations

Conflict 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.

Code availability The code can be downloaded from the personal webpage of Jochen Schütz at http://www.uhasselt.be/cmat or directly from http://www.uhasselt.be/Documents/CMAT/Code/MDRKCAT-CMAT.zip.

A Butcher Tableaux

In this section, we show the multiderivative Runge–Kutta methods used in this work through their Butcher tableaux. We use three two-derivative methods taken from [5], see Tables 3, 4 and 5; two three-derivative methods taken from [21], see Tables 6 and 7; and one four-derivative method, constructed for this paper, see Table 8. This last scheme has been derived from the idea that it should be of form

$$y^{n,l} = y^n + \sum_{k=1}^{\mathtt{r}-1} \frac{(c_l \Delta t)^k}{k!} \Phi^{(k-1)}\left(y^n\right) + \Delta t^{\mathtt{r}} \sum_{\nu=1}^{l-1} a_{l\nu}^{(\mathtt{r})} \Phi^{(\mathtt{r}-1)}\left(y^{n,\nu}\right),$$

for $l = 1, \ldots, \mathtt{s}$, with update

$$y^{n+1} = y^n + \sum_{k=1}^{\mathtt{r}-1} \frac{\Delta t^k}{k!} \Phi^{(k-1)}\left(y^n\right) + \Delta t^{\mathtt{r}} \sum_{l=1}^{\mathtt{s}} b_l^{(\mathtt{r})} \Phi^{(\mathtt{r}-1)}(y^{n,l}).$$

These forms have also been used in [5] and [21].

Table 3 2DRK3-2:Third order two-derivative Runge–Kutta scheme using two stages [5]

0	0	0	0	0
1	1	0	1/2	0
	2/3	1/3	1/6	0

Table 4 2DRK4-2:Fourth order two-derivative Runge–Kutta scheme using two stages [5]

0	0	0	0	0
1/2	1/2	0	1/8	0
	1	0	1/6	1/3

Table 5 2DRK5-3:Fifth order two-derivative Runge–Kutta scheme using three stages [5]

0	0	0	0	0	0	0
2/5	2/5	0	0	2/25	0	0
1	1	0	0	−1/4	3/4	0
	1	0	0	1/8	25/72	1/36

Table 6 3DRK5-2:Fifth order three-derivative Runge–Kutta scheme using two stages [21]

0	0	0	0	0	0	0
2/5	2/5	0	2/25	0	4/375	0
	1	0	1/2	0	1/16	5/48

Table 7 3DRK7-3:Seventh order three-derivative Runge–Kutta scheme using three stages [21]. The coefficients are given by $c_2 = \frac{3-\sqrt{2}}{7}$, $c_3 = \frac{3+\sqrt{2}}{7}$, $a_{32}^{(3)} = \frac{122+71\sqrt{2}}{7203}$, $b_1^{(3)} = \frac{1}{30}$, $b_2^{(3)} = \frac{1}{15} + \frac{13\sqrt{2}}{480}$, $b_3^{(3)} = \frac{1}{15} - \frac{13\sqrt{2}}{480}$

0	0	0	0	0	0	0	0	0	0
c_2	c_2	0	0	$c_2^2/2$	0	0	$c_2^3/6$	0	0
c_3	c_3	0	0	$c_3^2/2$	0	0	$c_2^3/6 - a_{32}^{(3)}$	$a_{32}^{(3)}$	0
	1	0	0	1/2	0	0	$b_1^{(3)}$	$b_2^{(3)}$	$b_3^{(3)}$

Table 8 4DRK6-2:Sixth order four-derivative Runge–Kutta scheme using two stages

0	0	0	0	0	0	0	0	0
1/3	1/3	0	1/18	0	1/162	0	1/1944	0
	1	0	1/2	0	1/6	0	1/60	1/40

References

1. Abramowitz, M., Stegun, I.A. (eds.): Handbook of Mathematical Functions with Formulas, Graphs and Mathematical Tables. Dover Publications Inc, New York (1964)
2. Baeza, A., Bürger, R., Martí, M.d.C., Mulet, P., Zorío, D.: On approximate implicit Taylor methods for ordinary differential equations. Comput. Appl. Math. **39**(4), 304 (2020)
3. Carrillo, H., Parés, C.: Compact approximate Taylor methods for systems of conservation laws. J. Sci. Comput. **80**(3), 1832–1866 (2019)
4. Carrillo, H., Parés, C., Zorío, D.: Lax–Wendroff approximate Taylor methods with fast and optimized weighted essentially non-oscillatory reconstructions. J. Sci. Comput. **86**, 15 (2021)
5. Chan, R., Tsai, A.: On explicit two-derivative Runge–Kutta methods. Numer. Algorithms **53**, 171–194 (2010)
6. Christlieb, A.J., Gottlieb, S., Grant, Z.J., Seal, D.C.: Explicit strong stability preserving multistage two-derivative time-stepping schemes. J. Sci. Comput. **68**, 914–942 (2016)
7. Cockburn, B., Karniadakis, G.E., Shu, C.-W.: The development of discontinuous Galerkin methods. In: Cockburn, B., Karniadakis, G.E., Shu, C.-W. (eds.) Discontinuous Galerkin Methods: Theory, Computation and Applications, volume 11 of Lecture Notes in Computational Science and Engineering, pp. 3–50. Springer, Berlin (2000)
8. Dumbser, M., Balsara, D.S., Toro, E.F., Munz, C.-D.: A unified framework for the construction of one-step finite volume and discontinuous Galerkin schemes on unstructured meshes. J. Comput. Phys. **227**(18), 8209–8253 (2008)
9. Dumbser, M., Enaux, C., Toro, E.F.: Finite volume schemes of very high order of accuracy for stiff hyperbolic balance laws. J. Comput. Phys. **227**(8), 3971–4001 (2008)
10. Dumbser, M., Fambri, F., Tavelli, M., Bader, M., Weinzierl, T.: Efficient implementation of ADER discontinuous Galerkin schemes for a scalable hyperbolic PDE engine. Axioms **7**(3), 63 (2018)
11. Gottlieb, S., Grant, Z.J., Hu, J., Shu, R.: High order unconditionally strong stability preserving multi-derivative implicit and IMEX Runge–Kutta methods with asymptotic preserving properties. arXiv preprint arXiv:2102.11939 (2021)
12. Gottlieb, S., Shu, C.-W., Tadmor, E.: Strong stability-preserving high-order time discretization methods. SIAM Rev. **43**(1), 89–112 (2001)
13. Guo, W., Qiu, J.-M., Qiu, J.: A new Lax–Wendroff discontinuous Galerkin method with superconvergence. J. Sci. Comput. **65**(1), 299–326 (2015)
14. Hairer, E., Wanner, G.: Solving ordinary differential equations II. Springer Series in Computational Mathematics (1991)
15. Jaust, A., Schütz, J., Seal, D.C.: Implicit multistage two-derivative discontinuous Galerkin schemes for viscous conservation laws. J. Sci. Comput. **69**, 866–891 (2016)
16. Lax, P., Wendroff, B.: Systems of conservation laws. Commun. Pure Appl. Math. **13**(2), 217–237 (1960)
17. LeVeque, R.J.: Numerical Methods for Conservation Laws. Birkhäuser, Basel (1990)
18. Li, J., Du, Z.: A two-stage fourth order time-accurate discretization for Lax–Wendroff type flow solvers I. Hyperbolic conservation laws. SIAM J. Sci. Comput. **38**(5), A3046–A3069 (2016)
19. Lu, C., Qiu, J.: Simulations of shallow water equations with finite difference Lax–Wendroff weighted essentially non-oscillatory schemes. J. Sci. Comput. **47**(3), 281–302 (2011)
20. MathWorks. Symbolic Math Toolbox. Natick, Massachusetts, United States (2020). https://www.mathworks.com/help/symbolic/
21. Ökten Turacı, M., Öziş, T.: Derivation of three-derivative Runge–Kutta methods. Numer. Algorithms **74**(1), 247–265 (2017)
22. Qiu, J.: Development and comparison of numerical fluxes for LWDG methods. Numer. Math. Theory Methods Appl. **1**(4), 435–459 (2008)
23. Qiu, J., Dumbser, M., Shu, C.-W.: The discontinuous Galerkin method with Lax–Wendroff type time discretizations. Comput. Methods Appl. Mech. Eng. **194**(42–44), 4528–4543 (2005)
24. Qiu, J., Shu, C.-W.: Finite difference WENO schemes with Lax–Wendroff-type time discretizations. SIAM J. Sci. Comput. **24**(6), 2185–2198 (2003)
25. Quarteroni, A., Sacco, R., Saleri, F.: Numerical Mathematics. Springer, Berlin (2007)

26. Ralston, A., Rabinowitz, P.: A First Course in Numerical Analysis. Dover Books on Mathematics. Dover Publications, Mineola (2001)
27. Schütz, J., Seal, D.C., Jaust, A.: Implicit multiderivative collocation solvers for linear partial differential equations with discontinuous Galerkin spatial discretizations. J. Sci. Comput. **73**, 1145–1163 (2017)
28. Schütz, J., Seal, D.C., Zeifang, J.: Parallel-in-time high-order multiderivative IMEX methods. J. Sci. Comput. **90**, 54 (2022)
29. Schwartzkopff, T., Dumbser, M., Munz, C.-D.: ADER: a high-order approach for linear hyperbolic systems in 2D. J. Sci. Comput. **17**, 231–240 (2002)
30. Seal, D.C., Güçlü, Y., Christlieb, A.: High-order multiderivative time integrators for hyperbolic conservation laws. J. Sci. Comput. **60**, 101–140 (2014)
31. Spiteri, R., Ruuth, S.: A new class of optimal high-order strong-stability-preserving time discretization methods. SIAM J. Numer. Anal. **40**(2), 469–491 (2002)
32. Strikwerda, J.C.: Finite Difference Schemes and Partial Differential Equations, 2nd edn. Society for Industrial and Applied Mathematics, Philadelphia (2004)
33. Titarev, V.A., Toro, E.F.: ADER: Arbitrary high order Godunov approach. J. Sci. Comput. **17**(1), 609–618 (2002)
34. Titarev, V.A., Toro, E.F.: ADER schemes for three-dimensional non-linear hyperbolic systems. J. Comput. Phys. **204**(2), 715–736 (2005)
35. Whitham, G.: Linear and Nonlinear Waves. Pure and Applied Mathematics: A Wiley Series of Texts, Monographs and Tracts. Wiley, New York (2011)
36. Zorío, D., Baeza, A., Mulet, P.: An approximate Lax–Wendroff-type procedure for high order accurate schemes for hyperbolic conservation laws. J. Sci. Comput. **71**, 246–273 (2017)

Paper II:
Jacobian-free Explicit Multiderivative General Linear Methods for Hyperbolic Conservation Laws

Numerical Algorithms
https://doi.org/10.1007/s11075-024-01771-6

ORIGINAL PAPER

Jacobian-free explicit multiderivative general linear methods for hyperbolic conservation laws

Afsaneh Moradi[1] · Jeremy Chouchoulis[2] · Raffaele D'Ambrosio[1] · Jochen Schütz[2]

Received: 22 November 2023 / Accepted: 23 January 2024

Abstract

We study explicit strong stability preserving (SSP) multiderivative general linear methods (MDGLMs) for the numerical solution of hyperbolic conservation laws. Sufficient conditions for MDGLMs up to four derivatives to be SSP are determined. In this work, we describe the construction of two external stage explicit SSP MDGLMs based on Taylor series conditions, and present examples of constructed methods up to order nine and three internal stages along with their SSP coefficients. It is difficult to apply these methods directly to the discretization of partial differential equations, as higher-order flux derivatives must be calculated analytically. We hence use a Jacobian-free approach based on the recent development of explicit Jacobian-free multistage multiderivative solvers (Chouchoulis et al. J. Sci. Comput. **90**, 96, 2022) that provides a practical application of MDGLMs. To show the capability of our novel methods in achieving the predicted order of convergence and preserving required stability properties, several numerical test cases for scalar and systems of equations are provided.

Keywords Hyperbolic conservation laws · Strong stability preserving · Multiderivative methods · General linear methods · Lax–Wendroff · Finite differences

Afsaneh Moradi
afsaneh.moradi@univaq.it

Jeremy Chouchoulis
jeremy.chouchoulis@uhasselt.be

Raffaele D'Ambrosio
raffaele.dambrosio@univaq.it

Jochen Schütz
jochen.schuetz@uhasselt.be

1 Department of Information Engineering and Computer Science and Mathematics, University of L'Aquila, L'Aquila, Italy

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

Mathematics Subject Classification (2010) 65M06 · 65M08 · 65M12 · 35L65

1 Introduction

In this work, we construct a novel class of high-order time integration schemes to solve one-dimensional hyperbolic conservation laws of form

$$Y_t + f(Y)_x = 0, \tag{1}$$

where Y is a conserved physical quantity and f is a flux function. The numerical approximation of such systems faces challenges when the exact solution becomes discontinuous, which can happen even if the initial profile is smooth. There exist a variety of successful approaches for developing high-resolution spatial discretizations capable of handling the presence of a discontinuity and shocks, see, e.g., [13, 44, 47, 49] and references therein for an overview. One of the most commonly used approaches in developing these schemes is the methods of lines (MOL) technique which decouples the spatial and temporal discretizations and reduces the partial differential equation (PDE) (1) to the semi-discrete form

$$y'(t) = \Phi(y). \tag{2}$$

The term $\Phi(y)$ is typically computed using a conservative spatial discretization D_x applied to the flux f,

$$\Phi(y) = D_x(-f(y)).$$

Commonly used spatial discretizations have special nonlinear stability properties (e.g., total variation stability or positivity preservation) if combined with the forward Euler method. Mathematically, this means that when the semi-discretized (2) is advanced using a first-order forward Euler method, the resulting numerical solution satisfies the following strong stability property if only the time-step Δt is sufficiently small:

$$\|y^n + \Delta t\Phi(y^n)\| \leq \|y^n\|, \quad 0 \leq \Delta t \leq \Delta t_{\mathrm{FE}}, \tag{3}$$

where $\|\cdot\|$ is a norm or semi-norm. Rather than using first-order time-stepping methods, we are interested in using higher-order time-stepping while still satisfying the strong stability property, i.e.,

$$\|y^{n+1}\| \leq \|y^n\|, \tag{4}$$

under a modified time-step limit, $\Delta t \leq \mathscr{C}\Delta t_{\mathrm{FE}}$. Methods that achieve (4) under the assumption that (3) holds, are called strong stability preserving (SSP) methods [43]; the coefficient $\mathscr{C}$ is referred to as the SSP coefficient [21]. Significant effort has been put into developing different classes of SSP methods with large enough SSP coefficients, such as SSP linear multistep methods (LMMs), Runge–Kutta (RK) methods, two-derivative RK methods, general linear methods (GLMs), and second-derivative GLMs to maximize the SSP coefficient, see, e.g., [8, 14, 19–21, 24–26, 28, 31]. In this paper, we aim to explore a novel class of SSP methods with higher derivatives: explicit multiderivative general linear methods (MDGLMs) up to four derivatives. Indeed, higher derivatives of the unknown solutions y are used in the method's formulation. While GLMs rely on only evaluating Φ, for multiderivative GLMs, we also make use of the quantities[1]

$$\dot{\Phi}(y) := y''(t) \equiv \Phi'(y)\Phi(y), \qquad \ddot{\Phi}(y) := y'''(t) \equiv \dots, \tag{5}$$

and so on. In the case of two-derivative methods, preserving just the forward Euler condition (3) is not enough, and another condition involving the second derivative is required. Considering a second-derivative condition in the form

$$\|y^n + \Delta t^2 \dot{\Phi}(y^n)\| \le \|y^n\|, \quad \Delta t \le \widehat{\alpha} \Delta t_{\text{FE}}, \tag{6}$$

Christlieb et al. [14] obtained SSP two-derivative RK methods up to order six preserving the strong stability properties of the forward Euler condition (3) along with the second-derivative condition (6). Here, the constant $\widehat{\alpha} > 0$ is associated with the stability condition of the second derivative and forward Euler terms. Building upon that work, Moradi et al. [31] developed the SSP approach to construct SSP second-derivative GLMs (SGLMs) as a class of multistep multistage second-derivative time-stepping methods, further studied in [32–37]. To accomplish the present work, we use, as before, the forward Euler and second-derivative conditions, but add to them Taylor series conditions for both third and fourth derivatives, and derive sufficient conditions for multiderivative GLMs to be SSP. By using multiple derivatives, the order of convergence can be increased without adding more stages. We focus on the construction of two and three-stage methods up to four derivatives.

For such schemes, the calculation of temporal derivatives (5) directly from their definition tends to be computationally prohibitive. A commonly used approach to compute these time derivatives is a Lax–Wendroff (LW) type of approach [29], which expresses temporal derivatives of the unknown function y in terms of the fluxes through the Cauchy–Kowalevskaya procedure. The main drawback of this procedure comes from the fact that it results in highly complex symbolic calculations which increase computational costs and make a modular implementation more difficult. Nevertheless, LW methods have a great deal of potential that is well-known among researchers, and there have been considerable efforts on developing high-order variants of LW methods for nonlinear systems. For example the ADER (Arbitrary order using Derivatives) methods attracted a lot of attention, see, e.g., [16–18, 42, 45, 46] and references therein. To get rid of complex symbolic calculations of flux derivatives, Carrillo and Parés in [9] have developed the *compact approximate Taylor* (CAT) procedure based on Taylor series methods and Chouchoulis et al. in [11] extended this approach to the class of multiderivative Runge–Kutta (MDRK) methods and presented a novel collection of explicit Jacobian-free MDRK solvers for hyperbolic conservation laws. In this paper, after introducing SSP multiderivative GLMs, to avoid the necessary

[1] The dot $(\cdot)$ stands for the time derivative d/dt, whereas the prime $(')$ stands for the Jacobian of the vector-valued Φ w.r.t. y.

cumbersome calculation of flux derivatives, we extend the idea of the Jacobian-free technique to SSP multiderivative GLMs and present a new class of explicit Jacobian-free SSP multiderivative GLMs for hyperbolic conservation laws.

The structure of the paper is as follows. In Section 2, we introduce multiderivative GLMs for ODEs along with the necessary and sufficient conditions for such methods to be of order p and stage order q. Thereafter, in Section 3, sufficient conditions for multiderivative GLMs up to four derivatives to be SSP are derived, and examples of constructed SSP third-derivative GLMs up to order seven and SSP fourth-derivative GLMs up to order nine are given. After a short review of the Jacobian-free approach of the CAT method and MDRK solvers, we introduce the explicit Jacobian-free MDGLMs for hyperbolic conservation laws, referred to as CAMDGLM, in Section 4. To verify our theoretical results, we present several numerical cases in Section 5. Finally, we close this work with conclusions and a spotlight for future work in Section 6.

2 Explicit multiderivative general linear methods

The class of general linear methods (GLMs) was first introduced by Butcher [5] for the numerical solution of ODEs defined by (2) in which Φ is a function of the solution variable $y \in \mathbb{R}^d$. GLMs are a large family of schemes, containing traditional methods including RK methods and LMMs. The introduction of GLMs opened the possibility of developing new methods that were neither RK methods, nor LMMs, nor minor alterations of these methods, see, for instance, [6, 27].

GLMs are characterized by four integers (p, q, r, s) and four matrices indicated by $A^{\{1\}} \in \mathbb{R}^{s\times s}$, $U \in \mathbb{R}^{s\times r}$, $B^{\{1\}} \in \mathbb{R}^{r\times s}$, and $V \in \mathbb{R}^{r\times r}$. Here, p and q are order and stage order of the method, respectively, r is the number of input and output approximations, and s is the number of internal stages. Let $Y^{[n]}$ be an approximation of stage order q to the vector $y(t^{n-1} + ch)$, i.e.,

$$Y_l^{[n]} = y(t^{n-1} + c_l \Delta t) + O(\Delta t^{q+1}), \quad l = 1, 2, \ldots, s. \tag{7}$$

and the vector $\Phi(Y^{[n]})$ denotes the first derivative stages values, where $c = [c_1\ c_2\ \cdots\ c_s]^T$ is the abscissa vector. A GLM used for the numerical approximation of the solution (2), on the uniform grid $\{t^n\}_{n=0}^N$, $t^n = t^0 + nh$, $n = 0, 1, \cdots, N$, is defined by

$$Y_l^{[n]} = \sum_{\nu=1}^{r} u_{l\nu} y_\nu^{[n-1]} + \Delta t \sum_{\nu=1}^{l-1} a_{l\nu}^{\{1\}} \Phi(Y_\nu^{[n]}), \quad l = 1, 2, \ldots, s, \tag{8a}$$

$$y_l^{[n]} = \sum_{\nu=1}^{r} v_{l\nu} y_\nu^{[n-1]} + \Delta t \sum_{\nu=1}^{s} b_{l\nu}^{\{1\}} \Phi(Y_\nu^{[n]}), \quad l = 1, 2, \ldots, r. \tag{8b}$$

Here, $y_v^{[n-1]}$ is an approximation of order p to some linear combination of the solution y and its derivatives at time t^{n-1}, i.e.,

$$y_v^{[n-1]} = \sum_{\kappa=0}^{p} w_{v\kappa} y^{(\kappa)}(t^{n-1})\Delta t^{\kappa} + O(\Delta t^{p+1}), \quad v = 1, 2, \ldots, r \tag{9}$$

for real parameters $w_{v\kappa}$. As the method is of order p, this implies that the output values $y_v^{[n]}$ fulfill

$$y_v^{[n]} = \sum_{\kappa=0}^{p} w_{v\kappa} y^{(\kappa)}(t^{n})\Delta t^{\kappa} + O(\Delta t^{p+1}), \quad v = 1, 2, \ldots, r. \tag{10}$$

Algebraic analysis of order of GLMs (8) was developed by Butcher [6], see also [27]. Set

$$\omega_0 = e - U w_0, \quad \omega_\kappa = \frac{c^\kappa}{\kappa!} - \frac{A^{\{1\}} c^{\kappa-1}}{(\kappa-1)!} - U w_\kappa,$$
$$\widehat{\omega}_0 = w_0 - V w_0, \quad \widehat{\omega}_\kappa = \sum_{l=0}^{\kappa} \frac{w_l}{(\kappa-l)!} - \frac{B^{\{1\}} c^{\kappa-1}}{(\kappa-1)!} - V w_\kappa,$$

where $\kappa = 1, 2, \ldots, p$, $e = [1 \;\; 1 \;\; \ldots \;\; 1]^T \in \mathbb{R}^s$, and $c^i := [c_1^i \ldots c_s^i]^T$. It will be always assumed that $w_0 = e = [1 \; 1 \; \ldots \; 1]^T \in \mathbb{R}^r$, so that the stage preconsistency condition $\omega_0 = 0$, or $U w_0 = e$, and the preconsistency condition $\widehat{\omega}_0 = 0$, or $V w_0 = w_0$, are automatically satisfied. A GLM (8) has order p and stage order $q = p$ if and only if $\omega_\kappa = 0$, and $\widehat{\omega}_\kappa = 0$, $\kappa = 1, \ldots, p$.

There has been a great deal of research on numerical methods for solving systems of ODEs which use the second derivative of the solution, $y''(t) \equiv \Phi'(y)\Phi(y)$, as part of their integration formula, see, e.g., [10, 12, 23]. The family of second derivative general linear methods (SGLMs) was first introduced by Butcher and Hojjati in [7] and was later investigated by Abdi et al. in [1–4], just to name a few. In recent years, there has been some progress on developing a multiderivative RK framework for temporal integration, for instance see [11, 12, 14, 38, 41] and references therein. By following this direction, the main class of time-stepping methods in this work are explicit multiderivative general linear methods (MDGLMs) as a generalization of GLMs by adding extra-temporal derivatives of $\Phi(y)$ up to, at most, four derivatives. It is crucial to note that these temporal derivatives can be recursively calculated via the chain rule, so that

$$\frac{\mathrm{d}^k}{\mathrm{d}t^k}\Phi(y) = \frac{\mathrm{d}^{k-1}}{\mathrm{d}t^{k-1}} \left(\Phi'(y)\Phi(y)\right).$$

Consider $0 \leq t^0 < t^1 < \cdots < t^N = T_f$ to be a uniform partition of the temporal domain with a fixed timestep $\Delta t = \frac{T_f - t^0}{N}$. We define MDGLMs as follows:

Definition 1 Explicit m-derivative general linear methods of order p and stage order q are r-value and s-stage methods of the form

$$Y_l^{[n]} = \sum_{\nu=1}^{r} u_{l\nu} y_\nu^{[n-1]} + \sum_{k=1}^{m} \Delta t^k \sum_{\nu=1}^{l-1} a_{l\nu}^{\{k\}} \frac{\mathrm{d}^{k-1}}{\mathrm{d}t^{k-1}} \Phi(Y_\nu^{[n]}), \quad l = 1, 2, \ldots, s, \tag{12a}$$

$$y_l^{[n]} = \sum_{\nu=1}^{r} v_{l\nu} y_\nu^{[n-1]} + \sum_{k=1}^{m} \Delta t^k \sum_{\nu=1}^{s} b_{l\nu}^{\{k\}} \frac{\mathrm{d}^{k-1}}{\mathrm{d}t^{k-1}} \Phi(Y_\nu^{[n]}), \quad l = 1, 2, \ldots, r. \tag{12b}$$

Here, similar to the class of GLMs, $Y_l^{[n]}$ is an approximation of stage order q to the solution y of (2) at time $t^{n-1} + c_l \Delta t$, and $y_\nu^{[n-1]}$ is an approximation of order p to some linear combination of the solution y and its derivatives at time t^{n-1}, which are defined by (7) and (9), respectively. As the method is of order p, this implies that the output values $y_\nu^{[n]}$ fulfill (10). For the sake of convenience, we represent MDGLMs through a partitioned $(s + r) \times (ms + r)$ Butcher tableau:

$$\left[\begin{array}{c|c|c|c|c} A^{\{1\}} & A^{\{2\}} & \cdots & A^{\{m\}} & U \\ \hline B^{\{1\}} & B^{\{2\}} & \cdots & B^{\{m\}} & V \end{array}\right],$$

where $A^{\{k\}} := [a_{l\nu}^{\{k\}}]_{s\times s}$ and similarly for $B^{\{k\}}$, U, and V.

2.1 Order and stage order conditions

To derive order conditions for the stages and the output values using a straightforward application of Taylor's theorem, let us denote the vectors

$$Z := [1 \; z \; \cdots \; z^p]^T, \quad e^{cz} = [e^{c_1 z} \; e^{c_2 z} \; \cdots \; e^{c_s z}],$$

and introduce the matrix W as

$$W := [w_0 \; w_1 \; \ldots \; w_p],$$

with $w_\kappa := [w_{1\kappa} \; w_{2\kappa} \; \ldots \; w_{r\kappa}]^T$ for $\kappa = 0, 1, \ldots, p$. The necessary and sufficient conditions for MDGLMs to be of order p and stage order $q = p$ or $q = p - 1$, respectively, are given in the following theorem:

Theorem 1 *Suppose that $y_l^{[n-1]}$ satisfies (9). Then, the m-derivative GLM (12) of order p and stage order $q = p$ satisfies (7) and (10) iff*

$$\begin{aligned} e^{cz} &= \sum_{k=1}^{m} z^k A^{\{k\}} e^{cz} + UWZ + O(z^{p+1}), \\ e^z WZ &= \sum_{k=1}^{m} z^k B^{\{k\}} e^{cz} + VWZ + O(z^{p+1}). \end{aligned} \tag{13}$$

The m-derivative GLM (12) of order p and stage order $q = p - 1$ satisfies (7) and (10) iff

$$\begin{aligned} e^{cz} &= \sum_{k=1}^{m} z^k A^{\{k\}} e^{cz} + UWZ \\ &\quad + \left(\frac{c^p}{p!} - \sum_{k=1}^{m} A^{\{k\}} \frac{c^{p-k}}{(p-k)!} - Uw_p \right) z^p + O(z^{p+1}), \qquad (14) \\ e^z WZ &= \sum_{k=1}^{m} z^k B^{\{k\}} e^{cz} + VWZ + O(z^{p+1}). \end{aligned}$$

Here, the exponential is applied component-wise to a vector.

It should be noted that the proofs are simple generalizations of those given for SGLMs [2, 7], and are hence neglected here. In line with what has been done for GLMs and SGLMs, it can be clarified that (13) and (14) are equivalent to

$$\frac{c^\kappa}{\kappa!} - A^{\{1\}} \frac{c^{\kappa-1}}{(\kappa-1)!} - A^{\{2\}} \frac{c^{\kappa-2}}{(\kappa-2)!} - \cdots - A^{\{m\}} \frac{c^{\kappa-m}}{(\kappa-m)!} - Uw_\kappa = 0, \quad \kappa = 0, 1, \ldots, q,$$

and

$$\sum_{j=0}^{\kappa} \frac{w_{\kappa-j}}{j!} - B^{\{1\}} \frac{c^{\kappa-1}}{(\kappa-1)!} - B^{\{2\}} \frac{c^{\kappa-2}}{(\kappa-2)!} - \cdots - B^{\{m\}} \frac{c^{\kappa-m}}{(\kappa-m)!} - Vw_\kappa = 0, \quad \kappa = 0, 1, \ldots, p,$$

for $q = p$ or $q = p - 1$, respectively.

3 SSP conditions for multiderivative GLMs

3.1 Monotonicity theory for multiderivative GLMs

Following the formulation of SGLMs introduced in [31], to determine sufficient conditions for multiderivative GLMs (12) to be SSP, we reformulate (12) as

$$\begin{aligned} Y_l^{[n]} &= \sum_{\nu=1}^{r} s_{l\nu} y_\nu^{[n-1]} + \sum_{k=1}^{m} \Delta t^k \sum_{\nu=1}^{s+r} t_{l\nu}^{\{k\}} \frac{\mathrm{d}^{k-1}}{\mathrm{d}t^{k-1}} \Phi(Y_\nu^{[n]}), \quad l = 1, 2, \ldots, s+r, \\ y_l^{[n]} &= Y_{s+l}^{[n]}, \quad l = 1, 2, \ldots, r, \end{aligned} \qquad (15)$$

where $n = 1, 2, \ldots, N$. This method is characterized by the matrices $T^{\{k\}} = [t_{l\nu}^{\{k\}}] \in \mathbb{R}^{(s+r)\times(s+r)}$, $k = 1, 2, \ldots, m$, and $S = [s_{l\nu}] \in \mathbb{R}^{(s+r)\times r}$ defined by

$$T^{\{k\}} = \begin{pmatrix} A^{\{k\}} & \mathbf{0} \\ B^{\{k\}} & \mathbf{0} \end{pmatrix}, \quad \text{and} \quad S = \begin{pmatrix} U \\ V \end{pmatrix}.$$

In a similar manner to [26, 31], we assume that the components of the matrix S fulfill the condition

$$\sum_{\nu=1}^{r} s_{l\nu} = 1, \quad l = 1, 2, \ldots, s + r.$$

We will say that an explicit m-derivative GLM (15) is monotonic if

$$\left\| Y_l^{[n]} \right\| \leq \max_{1 \leq \nu \leq r} \left\| y_\nu^{[n-1]} \right\|, \quad \text{for} \quad l = 1, 2, \ldots, s + r. \tag{16}$$

To obtain the strong stability conditions for multiderivative GLMs up to four derivatives, let us to introduce the vector

$$\Phi(Y^{[n]}) := \left[\Phi(Y_1^{[n]})^T \quad \Phi(Y_2^{[n]})^T \quad \cdots \quad \Phi(Y_{s+r}^{[n]})^T \right]^T, \quad k = 1, 2, \ldots, m,$$

and similarly for $\frac{\mathrm{d}^{k-1}}{\mathrm{d}t^{k-1}} \Phi(Y^{[n]})$. To preserve (16), the forward Euler condition (3) and the second-derivative condition (6) are needed together with two further conditions that include third- and fourth-order temporal derivatives. These are given as Taylor series conditions, i.e.,

$$\left\| Y^{[n]} + \Delta t \Phi(Y^{[n]}) + \frac{\Delta t^2}{2} \dot{\Phi}(Y^{[n]}) + \frac{\Delta t^3}{6} \ddot{\Phi}(Y^{[n]}) \right\| \leq \left\| Y^{[n]} \right\|, \quad \forall \Delta t \leq \overline{\alpha} \Delta t_{\text{FE}} \tag{17}$$

and

$$\left\| Y^{[n]} + \Delta t \Phi(Y^{[n]}) + \frac{\Delta t^2}{2} \dot{\Phi}(Y^{[n]}) + \frac{\Delta t^3}{6} \ddot{\Phi}(Y^{[n]}) + \frac{\Delta t^4}{24} \dddot{\Phi}(Y^{[n]}) \right\| \leq \left\| Y^{[n]} \right\|, \quad \forall \Delta t \leq \widetilde{\alpha} \Delta t_{\text{FE}}. \tag{18}$$

Here, $\overline{\alpha}$, and $\widetilde{\alpha}$ are scaling factors that compare the stability condition of the m-derivative method, for $m = 3$ and $m = 4$ respectively, to that of the forward Euler method. Using these conditions, we are able to formulate sufficient conditions so that a fourth-derivative GLM satisfies the desired monotonicity condition under a given timestep.

Theorem 2 *Given spatial discretizations Φ, $\dot{\Phi}$, $\ddot{\Phi}$, and $\dddot{\Phi}$ that satisfy the forward Euler condition* (3)*, the second-derivative condition* (6) *and Taylor series conditions* (17) *and* (18)*, respectively, a multiderivative GLM of the form* (15) *with $m = 4$ preserves the strong stability property $\left\| Y^{[n+1]} \right\| \leq \left\| Y^{[n]} \right\|$ under the timestep restriction $\Delta t \leq$*

$\beta \Delta t_{\mathrm{FE}}$ *if it satisfies the conditions*

$$\begin{aligned}
&\mathbf{R}S \geq 0,\\
&\beta\mathbf{R}\left(T^{\{1\}} - 6\tfrac{\beta^2}{\overline{\alpha}^2}T^{\{3\}} + 24\tfrac{\beta}{\widetilde{\alpha}}(\tfrac{\beta^2}{\overline{\alpha}^2} - \tfrac{\beta^2}{\widetilde{\alpha}^2})T^{\{4\}}\right) \geq 0,\\
&\tfrac{\beta^2}{\widehat{\alpha}^2}\mathbf{R}\left(T^{\{2\}} - 3\tfrac{\beta}{\overline{\alpha}}T^{\{3\}} + 12\tfrac{\beta}{\widetilde{\alpha}}(\tfrac{\beta}{\overline{\alpha}} - \tfrac{\beta}{\widetilde{\alpha}})T^{\{4\}}\right) \geq 0,\\
&6\tfrac{\beta^3}{\overline{\alpha}^3}\mathbf{R}\left(T^{\{3\}} - 4\tfrac{\beta}{\widetilde{\alpha}}T^{\{4\}}\right) \geq 0,\\
&24\tfrac{\beta}{\widetilde{\alpha}^4}\mathbf{R}T^{\{4\}} \geq 0,
\end{aligned} \tag{19}$$

where

$$\begin{aligned}
\mathbf{R} = \Bigg(& I + \beta T^{\{1\}} + \frac{\beta^2}{\widehat{\alpha}}T^{\{2\}} + 3\frac{\beta^3}{\overline{\alpha}^3\widehat{\alpha}^2}\left(2\widehat{\alpha}^2 - 2\overline{\alpha}\widehat{\alpha}^2 - \overline{\alpha}^2\right)T^{\{3\}}\\
&+ \left(24\frac{\beta^4(1-\widetilde{\alpha})}{\widetilde{\alpha}^4} + 12\frac{\beta^4\,(\widetilde{\alpha}-\overline{\alpha})}{\overline{\alpha}\widehat{\alpha}^2\widetilde{\alpha}^2} + 24\frac{\beta^4\,(\overline{\alpha}-1)}{\widetilde{\alpha}\overline{\alpha}^3}\right)T^{\{4\}}\Bigg)^{-1},
\end{aligned}$$

for some $\beta > 0$. In the above conditions, the inequalities are to be understood component-wise.

Proof Considering the method (15) with $m = 4$ and adding $\beta T^{\{1\}}Y^{[n]}$, $\widehat{\beta}^2 T^{\{2\}}Y^{[n]}$, $\left(6\overline{\beta}^3 - 6\beta\overline{\beta}^2 - 3\widehat{\beta}^2\overline{\beta}\right)T^{\{3\}}Y^{[n]}$, and $\left(24\widetilde{\beta}^3(\widetilde{\beta}-\beta) + 12\widetilde{\beta}\widehat{\beta}^2(\overline{\beta}-\widetilde{\beta}) + 24\widetilde{\beta}\overline{\beta}^2(\beta-\overline{\beta})\right)$ $T^{\{4\}}Y^{[n]}$ to both sides, results in

$$\begin{aligned}
&\Big(I + \beta T^{\{1\}} + \widehat{\beta}^2 T^{\{2\}} + \left(6\overline{\beta}^3 - 6\beta\overline{\beta}^2 - 3\widehat{\beta}^2\overline{\beta}\right)T^{\{3\}}\\
&+ \left(24\widetilde{\beta}^3(\widetilde{\beta}-\beta) + 12\widetilde{\beta}\widehat{\beta}^2(\overline{\beta}-\widetilde{\beta}) + 24\widetilde{\beta}\overline{\beta}^2(\beta-\overline{\beta})\right)T^{\{4\}}\Big)Y^{[n]}\\
=&Sy^{[n-1]} + \beta\left(T^{\{1\}} - 6\overline{\beta}^2T^{\{3\}} + 24\widetilde{\beta}(\overline{\beta}^2 - \widetilde{\beta}^2)T^{\{4\}}\right)\left(Y^{[n]} + \frac{\Delta t}{\beta}\Phi(Y^{[n]})\right)\\
&+ \widehat{\beta}^2\left(T^{\{2\}} - 3\overline{\beta}T^{\{3\}} + 12\widetilde{\beta}(\overline{\beta}-\widetilde{\beta})T^{\{4\}}\right)\left(Y^{[n]} + \frac{\Delta t^2}{\widehat{\beta}^2}\dot{\Phi}(Y^{[n]})\right)\\
&+ 6\overline{\beta}^3\left(T^{\{3\}} - 4\widetilde{\beta}T^{\{4\}}\right)\left(Y^{[n]} + \frac{\Delta t}{\overline{\beta}}\Phi(Y^{[n]}) + \frac{\Delta t^2}{2\overline{\beta}^2}\dot{\Phi}(Y^{[n]}) + \frac{\Delta t^3}{6\overline{\beta}^3}\ddot{\Phi}(Y^{[n]})\right)\\
&+ 24\widetilde{\beta}^4T^{\{4\}}\left(Y^{[n]} + \frac{\Delta t}{\widetilde{\beta}}\Phi(Y^{[n]}) + \frac{\Delta t^2}{2\widetilde{\beta}^2}\dot{\Phi}(Y^{[n]}) + \frac{\Delta t^3}{6\widetilde{\beta}^3}\ddot{\Phi}(Y^{[n]}) + \frac{\Delta t^4}{24\widetilde{\beta}^4}\dddot{\Phi}(Y^{[n]})\right).
\end{aligned}$$

Supposing that the matrix in front of $Y^{[n]}$ on the left-hand side is invertible and setting

$$\begin{aligned}
\mathbf{R} =& \Big(I + \beta T^{\{1\}} + \widehat{\beta}^2 T^{\{2\}} + \Big(6\overline{\beta}^3 - 6\beta\overline{\beta}^2 - 3\widehat{\beta}^2\overline{\beta}\Big) T^{\{3\}} \\
&+ \Big(24\widetilde{\beta}^3(\widetilde{\beta} - \beta) + 12\widetilde{\beta}\widehat{\beta}^2(\overline{\beta} - \widetilde{\beta}) + 24\widetilde{\beta}\overline{\beta}^2(\beta - \overline{\beta})\Big) T^{\{4\}}\Big)^{-1}, \\
\mathbf{G} =& \beta\mathbf{R}\left(T^{\{1\}} - 6\overline{\beta}^2 T^{\{3\}} + 24\widetilde{\beta}(\overline{\beta}^2 - \widetilde{\beta}^2)T^{\{4\}}\right), \\
\mathbf{H} =& \widehat{\beta}^2\mathbf{R}\left(T^{\{2\}} - 3\overline{\beta}T^{\{3\}} + 12\widetilde{\beta}(\overline{\beta} - \widetilde{\beta})T^{\{4\}}\right), \\
\mathbf{J} =& 6\overline{\beta}^3\mathbf{R}\left(T^{\{3\}} - 4\widetilde{\beta}T^{\{4\}}\right), \quad \mathbf{L} = 24\widetilde{\beta}^4\mathbf{R}T^{\{4\}},
\end{aligned}$$

we obtain

$$\begin{aligned}
Y^{[n]} =& \mathbf{R}Sy^{[n-1]} + \mathbf{G}\left(Y^{[n]} + \frac{\Delta t}{\beta}\Phi(Y^{[n]})\right) + \mathbf{H}\left(Y^{[n]} + \frac{\Delta t^2}{\widehat{\beta}^2}\dot{\Phi}(Y^{[n]})\right) \\
&+ \mathbf{J}\left(Y^{[n]} + \frac{\Delta t}{\overline{\beta}}\Phi(Y^{[n]}) + \frac{\Delta t^2}{2\overline{\beta}^2}\dot{\Phi}(Y^{[n]}) + \frac{\Delta t^3}{6\overline{\beta}^3}\ddot{\Phi}(Y^{[n]})\right) \\
&+ \mathbf{L}\left(Y^{[n]} + \frac{\Delta t}{\widetilde{\beta}}\Phi(Y^{[n]}) + \frac{\Delta t^2}{2\widetilde{\beta}^2}\dot{\Phi}(Y^{[n]}) + \frac{\Delta t^3}{6\widetilde{\beta}^3}\ddot{\Phi}(Y^{[n]}) + \frac{\Delta t^4}{24\widetilde{\beta}^4}\dddot{\Phi}(Y^{[n]})\right).
\end{aligned}$$

An easy computation shows that $\mathbf{R} + \mathbf{G} + \mathbf{H} + \mathbf{J} + \mathbf{L} = I$. In combination with the fact that the elements of $\mathbf{G}$, $\mathbf{H}$, $\mathbf{J}$, $\mathbf{L}$, and $\mathbf{R}S$ are all non-negative, and using

$$\|Sy^{n-1}\| \leq \sum_{\nu=1}^{r} s_{l\nu} \max_j \|y_j^{[n-1]}\| \leq \max_j \|y_j^{[n-1]}\|, \quad l = 1, 2, \ldots, s + r,$$

these five terms describe a convex combination of terms which are SSP, and the resulting value is SSP as well:

$$\begin{aligned}
\left\|Y^{[n]}\right\| \leq& \mathbf{R}S\left\|y^{[n-1]}\right\| + \mathbf{G}\left\|Y^{[n]} + \frac{\Delta t}{\beta}\Phi(Y^{[n]})\right\| + \mathbf{H}\left\|Y^{[n]} + \frac{\Delta t^2}{\widehat{\beta}^2}\dot{\Phi}(Y^{[n]})\right\| \\
&+ \mathbf{J}\left\|Y^{[n]} + \frac{\Delta t}{\overline{\beta}}\Phi(Y^{[n]}) + \frac{\Delta t^2}{2\overline{\beta}^2}\dot{\Phi}(Y^{[n]}) + \frac{\Delta t^3}{6\overline{\beta}^3}\ddot{\Phi}(Y^{[n]})\right\| \\
&+ \mathbf{L}\left\|Y^{[n]} + \frac{\Delta t}{\widetilde{\beta}}\Phi(Y^{[n]}) + \frac{\Delta t^2}{2\widetilde{\beta}^2}\dot{\Phi}(Y^{[n]}) + \frac{\Delta t^3}{6\widetilde{\beta}^3}\ddot{\Phi}(Y^{[n]}) + \frac{\Delta t^4}{24\widetilde{\beta}^4}\dddot{\Phi}(Y^{[n]})\right\|,
\end{aligned}$$

under the time-step restrictions $\Delta t \leq \beta \Delta t_{\mathrm{FE}}$, $\Delta t \leq \widehat{\alpha}\widehat{\beta}\Delta t_{\mathrm{FE}}$, $\Delta t \leq \overline{\alpha}\overline{\beta}\Delta t_{\mathrm{FE}}$ and $\Delta t \leq \widetilde{\alpha}\widetilde{\beta}\Delta t_{\mathrm{FE}}$. Equating these four timestep restrictions leads to the optimal timestep, i.e., we require $\beta = \widehat{\alpha}\widehat{\beta} = \overline{\alpha}\overline{\beta} = \widetilde{\alpha}\widetilde{\beta}$. Therefore, for $\widehat{\beta} = \frac{\beta}{\widehat{\alpha}}$, $\overline{\beta} = \frac{\beta}{\overline{\alpha}}$, and $\widetilde{\beta} = \frac{\beta}{\widetilde{\alpha}}$, the SSP conditions (19) ensure that $\mathbf{G} \geq 0$, $\mathbf{H} \geq 0$, $\mathbf{J} \geq 0$, $\mathbf{L} \geq 0$, and $\mathbf{R}S \geq 0$ component-wise and the method (15) with $m = 4$ preserves the strong stability condition $\left\|Y^{[n+1]}\right\| \leq \left\|Y^{[n]}\right\|$ under the timestep restriction $\Delta t \leq \beta \Delta t_{\mathrm{FE}}$. $\square$

This theorem provides sufficient conditions for fourth-derivative GLMs to be SSP for any $\Delta t \leq \beta \Delta t_{\mathrm{FE}}$. It should be noted that SSP conditions for third-derivative GLMs can be derived by setting $T^{\{4\}} = 0$ in (19). Similar to [14, 31], the search for optimal SSP MDGLMs can be cast into an optimization problem with the aim of maximizing the SSP coefficient $\mathscr{C} = \max \beta$ subject to the SSP conditions (19) and order conditions (13) (for methods of order $p = q$) or (14) (for methods of order $p = q + 1$) as inequality and equality constraints, respectively.

3.2 Optimal SSP multiderivative GLMs

In this section, we develop optimal SSP third- and fourth-derivative GLMs and their corresponding SSP coefficients, and compare with SSP third-derivative RK methods developed in [39]. The value of the SSP coefficient $\mathscr{C}$ is a function of the parameters $\widehat{\alpha}$, $\overline{\alpha}$, and $\widetilde{\alpha}$, which depend on the spatial discretizations for the first, second, third, and fourth derivatives. Following the existing publications on SSP second-derivative methods, e.g., in [14], we consider the convection equation $Y_t = Y_x$. Φ is defined to be the original first-order upwind method

$$\Phi(y^n)_i := \frac{y^n_{i+1} - y^n_i}{\Delta x} \approx Y_x(x_i), \tag{20}$$

and $\dot{\Phi}$ is defined via the second-order centered discretization to Y_{xx} as

$$\dot{\Phi}(y^n)_i := \frac{y^n_{i+1} - 2y^n_i + y^n_{i-1}}{\Delta x^2} \approx Y_{xx}(x_i). \tag{21}$$

Both approaches were proved to be total variation diminishing (TVD) [14] in the following sense:

$$y^{n+1} = y^n + \Delta t \Phi(y^n), \qquad \text{is TVD for} \qquad \Delta t \leq \Delta x \tag{22}$$

$$y^{n+1} = y^n + \Delta t^2 \dot{\Phi}(y^n), \qquad \text{is TVD for} \qquad \Delta t \leq \frac{\sqrt{2}}{2}\Delta x. \tag{23}$$

To determine the values of $\overline{\alpha}$ and $\widetilde{\alpha}$ for the same problem, we consider the following discretization schemes to approximate third- and fourth-order temporal derivatives

$$\ddot{\Phi}(y^n)_i = \frac{y^n_{i+2} - 3y^n_{i+1} + 3y^n_i - y^n_{i-1}}{\Delta x^3} \approx Y_{xxx}(x_i), \tag{24}$$

$$\ddot{\Phi}(y^n)_i = \frac{y^n_{i+2} - 4y^n_{i+1} + 6y^n_i - 4y^n_{i-1} + y^n_{i-2}}{\Delta x^4} \approx Y_{xxxx}(x_i). \tag{25}$$

Using these discretizations, we can directly compute the values of $\overline{\alpha}$ and $\widetilde{\alpha}$ for which the Taylor series conditions (17) and (18) are TVD. That is, with $\widetilde{\alpha} := \frac{\Delta t}{\Delta x} \geq 0$, for the fourth-derivative Taylor series condition, we observe that

$$\begin{aligned}\|y^{n+1}\|_{TV} =& \Bigg\| \left(\frac{\widetilde{\alpha}^3}{6} + \frac{\widetilde{\alpha}^4}{24}\right) y^n_{i+2} + \left(\widetilde{\alpha} + \frac{\widetilde{\alpha}^2}{2} - \frac{\widetilde{\alpha}^3}{2} - \frac{\widetilde{\alpha}^4}{6}\right) y^n_{i+1} \\ &+ \left(1 - \widetilde{\alpha} - \widetilde{\alpha}^2 + \frac{\widetilde{\alpha}^3}{2} + \frac{\widetilde{\alpha}^4}{4}\right) y^n_i + \left(\frac{\widetilde{\alpha}^2}{2} - \frac{\widetilde{\alpha}^3}{6} - \frac{\widetilde{\alpha}^4}{6}\right) y^n_{i-1} + \frac{\widetilde{\alpha}^4}{24} y^n_{i-2} \Bigg\|_{TV} \\ \leq& \|y^n\|_{TV}\end{aligned}$$

provided that

$$\left.\begin{array}{l} \widetilde{\alpha} + \frac{\widetilde{\alpha}^2}{2} - \frac{\widetilde{\alpha}^3}{2} - \frac{\widetilde{\alpha}^4}{6} \geq 0, \\ 1 - \widetilde{\alpha} - \widetilde{\alpha}^2 + \frac{\widetilde{\alpha}^3}{2} + \frac{\widetilde{\alpha}^4}{4} \geq 0, \\ \frac{\widetilde{\alpha}^2}{2} - \frac{\widetilde{\alpha}^3}{6} - \frac{\widetilde{\alpha}^4}{6} \geq 0, \end{array}\right\} \Longleftrightarrow \widetilde{\alpha} \leq 0.73205\ldots.$$

In a similar way, setting $\overline{\alpha} := \frac{\Delta t}{\Delta x}$ and plugging the above definitions of Φ, $\dot{\Phi}$, and $\ddot{\Phi}$ into the third-order Taylor condition (17), we have

$$\left.\begin{array}{l} \overline{\alpha} + \frac{\overline{\alpha}^2}{2} - \frac{\overline{\alpha}^3}{2} \geq 0, \\ 1 - \overline{\alpha} - \overline{\alpha}^2 + \frac{\overline{\alpha}^3}{2} \geq 0, \\ \frac{\overline{\alpha}^2}{2} - \frac{\overline{\alpha}^3}{6} \geq 0, \end{array}\right\} \Longleftrightarrow \overline{\alpha} \leq 0.68889\ldots.$$

Considering the values of $\widehat{\alpha} = \frac{\sqrt{2}}{2}$, $\overline{\alpha} = 0.68889$ and $\widetilde{\alpha} = 0.73205$, we are able to derive SSP third-derivative GLMs (when $T^{\{4\}} = 0$) and fourth-derivative GLMs by solving an optimization problem with objective function of the form

$$\min \quad -\beta, \tag{26}$$

subject to inequality constraints corresponding to the SSP conditions (19), depending on the value of β and the coefficient matrices of the methods, and equality constraints corresponding to the order and stage order conditions (13) (for methods of order $p = q$) or (14) (for methods of order $p = q + 1$). In this work, we restrict our attention to MDGLMs (15) with $s = 2$ and $s = 3$ internal stages and $r = 2$ external stages of order p, with stage order $q = p$ and $q = p - 1$. We assume that the matrices

$A^{\{k\}} \in \mathbb{R}^{s\times s}$, $k = 1, 2, \dots, m$, are strictly lower triangular, i.e.,

$$A^{\{k\}} = \begin{bmatrix} 0 & & & & \\ a_{21}^{\{k\}} & 0 & & & \\ \vdots & \ddots & \ddots & & \\ a_{s-1,1}^{\{k\}} & \ddots & \ddots & 0 & \\ a_{s,1}^{\{k\}} & a_{s,2}^{\{k\}} & \cdots & a_{s,s-1}^{\{k\}} & 0 \end{bmatrix}.$$

To ensure zero-stability, we also assume that the matrix V is a rank one matrix of the form $V = ev^T$ with $e = [1\ 1]^T$, $v = [v_1\ v_2]^T$ and $v^T e = 1$.

Solving the optimization problem (26) using the MATLAB function `fmincon` choosing the sequential programming ("sqp") algorithm, we derive the following novel third- and fourth-derivative GLMs that are provably SSP in the sense of Theorem 2:

- The third-derivative methods SSP 3DGLMp-s, with $p = q = 4, 5$, and $s = 2$.
- The third-derivative methods SSP 3DGLMp-s, with $p = q+1 = 6, 7$, and $s = 3$.
- The fourth-derivative methods SSP 4DGLMp-s, with $p = q = 4, 5, 6$, and $s = 2$.
- The fourth-derivative method SSP 4DGLM7-s, with $p = q + 1 = 7$ and $s = 2$.
- The fourth-derivative methods SSP 4DGLMp-s, with $p = q+1 = 6, 7, 8, 9$, and $s = 3$.

The twelve developed methods are listed in the Appendix. The coefficients are rather cumbersome, as explained, they stem from a numerical optimization routine. The coefficients can therefore also be downloaded at https://www.uhasselt.be/nl/wie-is-wie/jochen-schuetz or obtained from the corresponding author upon reasonable request.

In Table 1, we report the obtained values of SSP coefficients β for the constructed schemes, as well as the SSP coefficient of the SSP 2DGLM methods developed in [31] and SSP 3DRK methods studied in [39], which reveals that multiderivative methods enable the attainment of higher-order accuracy with a minimal number of stages. In this table, dashes indicate that such methods cannot be constructed due to the lack of free parameters for solving both order and stage order conditions.

4 Multiderivative GLMs for hyperbolic conservation laws

On the uniform partition of the domain Ω with M cells of the size Δx, i.e., $\{x_1, \dots, x_M\}$, multiderivative GLMs (12) applied to (1) can be expressed as

$$Y_{i,l}^{[n]} = \sum_{\nu=1}^{r} u_{l\nu} y_{i,\nu}^{[n-1]} - \sum_{k=1}^{m} \Delta t^k \sum_{\nu=1}^{l-1} a_{l\nu}^{\{k\}} D_x D_t^{k-1} f(Y_{i,\nu}^{[n]}), \quad l = 1, 2, \dots, s, , \quad (27a)$$

$$y_{i,l}^{[n]} = \sum_{\nu=1}^{r} v_{l\nu} y_{i,\nu}^{[n-1]} - \sum_{k=1}^{m} \Delta t^k \sum_{\nu=1}^{s} b_{l\nu}^{\{k\}} D_x D_t^{k-1} f(Y_{i,\nu}^{[n]}), \quad l = 1, 2, \dots, r, \tag{27b}$$

where $n = 1, 2, \dots, N$, and $i = 1, 2, \dots, M$. Here, D_x and D_t stand for suitable approximations of ∂_x and ∂_t. This section aims to avoid using Jacobians of the flux function f that occur as a result of higher temporal derivatives used in the formulation of our methods. To do this, we follow the Jacobian-free technique outlined in [11]. This technique is based on the compact approximate Taylor (CAT) approach proposed in [9] as an extension to the work in [50]. These techniques heavily rely on discrete differentiation. In what follows, we introduce the required notations and describe the Jacobian-free technique when applied to MDGLMs (27) in the sequel.

4.1 Discrete differentiation

The aim of this part is to fix some notation on using finite differences similar as in [9, 11]. We will use two families of interpolatory formulas: the numerical approximations for the k-th derivative based on $(2\overline{p}+1)$-point stencils and $2\overline{p}$-point stencils.

Assuming that $\{x_i\}$ are the points of a uniform mesh of step Δx, the first family based on a $(2\overline{p}+1)$-point stencil is given by

$$(\mathscr{P}_i \varphi)^{(k)}(x_i) := \frac{1}{\Delta x^k} \sum_{j=-\overline{p}}^{\overline{p}} \delta_{\overline{p},j}^{k} \varphi(x_{i+j}), \tag{28}$$

where $\mathscr{P}_i\varphi$ are the Lagrangian interpolation polynomials of degree $2\overline{p}$ interpolating $\varphi : \mathbb{R} \to \mathbb{R}$ at the $2\overline{p}+1$ points $x_{i-\overline{p}}, \dots, x_{i+\overline{p}}$; and $\delta_{\overline{p},j}^{k}$ are related to the Lagrange

Table 1 SSP coefficients β of the two and three stage SSP second, third and fourth derivative GLMs of order p and stage order $q = p$ and $q = p - 1$

$p =$	$q =$	SSP 2DGLM	SSP 3DRK	SSP 3DGLM	SSP 4DGLM
$s = 2$					
4	4	0.738	1.120	1.119	1.436
5	5	–	0.679	0.755	1.223
6	6	–	–	–	1.013
7	6	–	–	–	0.775
$s = 3$					
6	5	–	0.677	0.589	1.819
7	6	–	–	0.282	1.344
8	7	–	–	–	0.826
9	8	–	–	–	0.328

SSP 2DGLM and SSP 3DRK were developed in [31] and [39], respectively,and are added for references polynomials

$$L_{\overline{p},j}(\omega) := \prod_{\substack{r=-\overline{p}\\ r\neq j}}^{\overline{p}} \frac{\omega - r}{j - r}, \quad j = -\overline{p}, \dots, \overline{p} \tag{29}$$

through

$$\delta^k_{\overline{p},j} := L^{(k)}_{\overline{p},j}(0), \quad j = -\overline{p}, \dots, \overline{p}.$$

We will also use the following numerical differentiation formulas based on a $2\overline{p}$-point stencil,

$$(\mathscr{Q}_i\varphi)^{(k)}(x_{i+\mathbf{m}}) := \frac{1}{\Delta x^k} \sum_{j=-\overline{p}+1}^{\overline{p}} \gamma^{k,\mathbf{m}}_{\overline{p},j}\varphi(x_{i+j}), \tag{30}$$

that approximate the k-th derivative at the points $x_i + \mathbf{m}\Delta x$, $\mathbf{m} = -\overline{p}+1, \dots, \overline{p}$. Here, $\mathscr{Q}_i\varphi$ are the Lagrangian interpolation polynomials of degree $2\overline{p}-1$ interpolating φ at the $2\overline{p}$ points $x_{i-\overline{p}+1}, \dots, x_{i+\overline{p}}$, and $\gamma^{k,\mathbf{m}}_{\overline{p},j}$ are related to the Lagrange polynomials

$$\ell_{\overline{p},j}(\omega) := \prod_{\substack{r=-\overline{p}+1\\ r\neq j}}^{\overline{p}} \frac{\omega - r}{j - r}, \quad j = -\overline{p}+1, \dots, \overline{p} \tag{31}$$

through

$$\gamma^{k,\mathbf{m}}_{\overline{p},j} := \ell^{(k)}_{\overline{p},j}(\mathbf{m}), \quad j, \mathbf{m} = -\overline{p}+1, \dots, \overline{p}.$$

The corresponding linear operators to (28) and (30) are defined by

$$P^{(k)} : \mathbb{R}^{2\overline{p}+1} \to \mathbb{R}, \quad \mathbf{v} \mapsto \frac{1}{\Delta x^k} \sum_{j=-\overline{p}}^{\overline{p}} \delta^k_{\overline{p},j} v_j,$$

$$Q^{(k)}_{\mathbf{m}} : \mathbb{R}^{2\overline{p}} \to \mathbb{R}, \quad \mathbf{w} \mapsto \frac{1}{\Delta x^k} \sum_{j=-\overline{p}+1}^{\overline{p}} \gamma^{k,\mathbf{m}}_{\overline{p},j} w_j.$$

In order to have the method in conservation form, auxiliary centered coefficients $\lambda^{k-1}_{\overline{p},j}$ have been introduced in [50] via the relations

$$\begin{aligned}
\delta^k_{\overline{p},\overline{p}} &=: \lambda^{k-1}_{\overline{p},\overline{p}},\\
\delta^k_{\overline{p},j} &=: \lambda^{k-1}_{\overline{p},j} - \lambda^{k-1}_{\overline{p},j+1}, \quad j = -\overline{p}+1, \dots, \overline{p}-1,\\
\delta^k_{\overline{p},-\overline{p}} &=: -\lambda^{k-1}_{\overline{p},-\overline{p}+1}.
\end{aligned}$$

Considering these auxiliary centered coefficients allows for an alternative form for (28) as differences of new ‘half-way point’ interpolation operators:

$$(\mathscr{P}_i\varphi)^{(k)}(x_i) = \frac{\left(\Lambda^{(k-1)}\varphi\right)(x_{i+1/2}) - \left(\Lambda^{(k-1)}\varphi\right)(x_{i-1/2})}{\Delta x}, \tag{32}$$

with $\Lambda^{(k-1)}$ as an operator mapping to $\mathbb{P}_{2\overline{p}-1}$ given by

$$\left(\Lambda^{(k-1)}\varphi\right)(x_{i+1/2}) := \frac{1}{\Delta x^{k-1}} \sum_{j=-\overline{p}+1}^{\overline{p}} \lambda_{\overline{p},j}^{k-1} \varphi(x_{i+j}). \tag{33}$$

For a detailed description of interpolation operators, we refer to [9, 11] and the references therein.

4.2 Jacobian-free MDGLMs

With the aim of assembling the class of compact approximate MDGLMs (CAMDGLMs), we define the conservative form of the solution via

$$Y_{i,l}^{[n]} := \sum_{\nu=1}^{r} u_{l\nu} y_{i,\nu}^{[n-1]} - \frac{\Delta t}{\Delta x}\left(\widetilde{F}_{i+1/2,l}^{[n]} - \widetilde{F}_{i-1/2,l}^{[n]}\right), \quad l = 1, 2, \ldots, s, \tag{34a}$$

$$y_{i,l}^{[n]} := \sum_{\nu=1}^{r} v_{l\nu} y_{i,\nu}^{[n-1]} - \frac{\Delta t}{\Delta x}\left(F_{i+1/2,l}^{[n]} - F_{i-1/2,l}^{[n]}\right), \quad l = 1, 2, \ldots, r, \tag{34b}$$

where the numerical fluxes are given by

$$\widetilde{F}_{i+1/2,l}^{[n]} = \sum_{k=1}^{m} \Delta t^{k-1} \sum_{\nu=1}^{l-1} a_{l\nu}^{\{k\}} \Lambda^{(0)} (\widetilde{\mathbf{f}}^{\nu})_{i,\langle 0\rangle}^{(k-1)}, \quad l = 1, 2, \ldots, s \tag{35a}$$

$$F_{i+1/2,l}^{[n]} = \sum_{k=1}^{m} \Delta t^{k-1} \sum_{\nu=1}^{s} b_{l\nu}^{\{k\}} \Lambda^{(0)} (\widetilde{\mathbf{f}}^{\nu})_{i,\langle 0\rangle}^{(k-1)}, \quad l = 1, 2, \ldots, r. \tag{35b}$$

Here, the angled brackets stand for the local stencil function

$$\langle\cdot\rangle : \mathbb{Z} \to \mathbb{Z}^{2\overline{p}} : w \mapsto (w - \overline{p} + 1, \ldots, w + \overline{p})^T,$$

and will be used for both the spatial index i and temporal index n. As in [11], to compute $\widetilde{\mathbf{f}}_{i,\langle 0\rangle}^{(k-1)}$, we use the compact approximate Taylor (CAT) procedure [9] and hence, the flux derivatives are computed by the linear operator alternative of (32) determined by

$$\Lambda^{(0)}\widetilde{\mathbf{f}}_{i,\langle 0\rangle}^{(k-1)} := \sum_{j=-\overline{p}+1}^{\overline{p}} \lambda_{\overline{p},j}^{0} \widetilde{\mathbf{f}}_{i,j}^{(k-1)},$$

in which

$$\widetilde{\mathbf{f}}_{i,j}^{(k-1)} := Q_0^{(k-1)}(\mathfrak{f}_T)_{i,j}^{k-1,\langle n\rangle}, \quad j = -\overline{p}+1, \ldots, \overline{p}$$

are local approximations for the temporal derivatives of the flux and depend on the approximate flux values $(\mathfrak{f}_T)_{i,j}^{k-1,n+r} \approx f(Y_{i+j}^{[n+r]})$. Indeed, the following approximate flux is taken to approximately evaluate the $(k-1)$-st discrete temporal derivative in x_{i+j}:

$$(\mathfrak{f}_T)_{i,j}^{k-1,n+r} := f\left(Y_{i+j}^{[n]} + \sum_{\ell=1}^{k-1} \frac{(r\Delta t)^\ell}{\ell!} \widetilde{Y}_{i,j}^{(\ell)}\right), \qquad j, r = -\overline{p}+1, \dots, \overline{p}.$$

All that remains to be defined are the quantities $\widetilde{Y}_{i,j}^{(\ell)} \approx \frac{\partial^\ell}{\partial t^\ell} Y_{i+j}^{[n]}$. To do this, we make use of the Cauchy–Kovalevskaya identity

$$\partial_t^\ell y = -\partial_x \partial_t^{\ell-1} f(y),$$

so, we have

$$\widetilde{Y}_{i,j}^{(\ell)} := -Q_j^{(1)} \widetilde{\mathbf{f}}_{i,\langle 0\rangle}^{(\ell-1)}, \quad j = -\overline{p}+1, \dots, \overline{p}.$$

A summary of the compact approximate mDGLMp-s procedure to obtain the stage values is provided in Algorithm 1. It should be noted that the flux at the left half-way point is determined by a shift of the index, i.e., $\widetilde{F}_{i-1/2,l}^{[n]} = \widetilde{F}_{i-1+1/2,l}^{[n]}$ or is given by the boundary condition.

Following [11], the expected order is the minimum of the underlying multiderivative GLMs and the order of the interpolation $(2\overline{p})$. Assuming that both f and y are smooth functions in C^∞, and $\mathcal{O}(\Delta t) = \mathcal{O}(\Delta x)$, we have the following theorem stating the order of Jacobian-free MDGLMs.

Theorem 3 *The order of an explicit compact approximate mDGLMp-s is given by* $\min(2\overline{p}, p)$ *where* p *is the order of the underlying MDGLM, while the stencil to update* $y(x_i, t^n)$ *in the CAT procedure is given by* $\{i-\overline{p}, i-\overline{p}+1, \dots, i+\overline{p}\}$.

Algorithm 1 Stages of compact approximate mDGLMp-s, an m-derivative, p-th order, s-stage CAMDGLM.

```
Stage solution ($l = 2, \dots, s$):
for $j = -\overline{p}+1$ to $\overline{p}$ do
    $(\widetilde{f}^{l-1})_{i,j}^{(0)} = f(Y_{i+j,l}^{[n]})$
end
$\widetilde{F}_{i+1/2,l}^{[n]} = \sum_{\nu=1}^{l-1} a_{l\nu}^{\{1\}} \Lambda^{(0)} (\widetilde{\mathbf{f}}^{\nu})_{i,\langle 0\rangle}^{(0)}$
for $k = 2$ to $\mathtt{r}$ do
    Get $(\widetilde{\mathbf{f}}^{l-1})_{i,j}^{(k-1)}$ via CAT procedure.
    $\widetilde{F}_{i+1/2,l}^{[n]} \mathrel{+}= \Delta t^{k-1} \sum_{\nu=1}^{l-1} a_{l\nu}^{\{k\}} \Lambda^{(0)} (\widetilde{\mathbf{f}}^{\nu})_{i,\langle 0\rangle}^{(k-1)}$
end
$Y_{i,l}^{[n]} = \sum_{\nu=1}^{2} u_{l\nu} y_{i,\nu}^{[n-1]} - \frac{\Delta t}{\Delta x}(\widetilde{F}_{i+1/2,l}^{[n]} - \widetilde{F}_{i-1/2,l}^{[n]})$
```

```
CAT procedure [9] ($k = 2, \dots, m$):
for $j = -\overline{p}+1$ to $\overline{p}$ do
    $\widetilde{Y}_{i,j}^{(k-1)} = -Q_j^{(1)} \widetilde{\mathbf{f}}_{i,\langle 0\rangle}^{(k-2)}$
        $= -\frac{1}{\Delta x} \sum_{r=-\overline{p}+1}^{\overline{p}} \gamma_{\overline{p},r}^{1,j} \widetilde{\mathbf{f}}_{i,r}^{(k-2)}$
    for $r = -\overline{p}+1$ to $\overline{p}$ do
        $(\mathfrak{f}_T)_{i,j}^{k-1,n+r} = f\left(Y_{i+j}^{[n]} + \sum_{\ell=1}^{k-1} \frac{(r\Delta t)^\ell}{\ell!} \widetilde{Y}_{i,j}^{(\ell)}\right)$
    end
    $\widetilde{\mathbf{f}}_{i,j}^{(k-1)} = Q_0^{(k-1)} (\mathfrak{f}_T)_{i,j}^{k-1,\langle n\rangle}$
        $= \frac{1}{\Delta t^{k-1}} \sum_{r=-\overline{p}+1}^{\overline{p}} \gamma_{\overline{p},r}^{k-1,0} (\mathfrak{f}_T)_{i,j}^{k-1,n+r}$
end
```

The proof is similar to [11, Theorem 1], and is hence left out.

4.3 A Jacobian-free starting procedure

Due to the structure of the input vector of a multiderivative GLM, a starting procedure is needed to approximate the initial vector $y^{[0]}$ using sufficient output information. As mentioned in Section 2, see (9), the components of the input vector $y^{[0]}_{i,\nu}$, $i = 1, 2, \dots, M$, are approximations of order p to the linear combinations of the solution y and its derivatives at the point (x_i, t^0), i.e.,

$$y^{[0]}_{i,\nu} = \sum_{\kappa=0}^{p} w_{\nu\kappa} \partial_t^{\kappa} y(x_i, t^0) + O(\Delta t^{p+1}). \tag{36}$$

To calculate higher derivatives $\partial_t^{\kappa} y(x_i, t^0)$ directly, we use the Cauchy-Kovalevskaya identity

$$\partial_t^{\kappa} y(x_i, t^0) = -\partial_x \partial_t^{\kappa-1} f(y(x_i, t^0)),$$

once again, and the approximate Taylor procedure [9] to derive the expression of the initial vector $y^{[0]}$. The flux derivatives can be computed by

$$\partial_t^{\kappa} y(x_i, t^0) \approx \widetilde{y}_i^{(\kappa)} := -P^{(1)} \widetilde{f}^{(\kappa-1)}_{\langle i \rangle},$$

in which the approximations $\widetilde{f}^{(\kappa-1)}_{i+j} \approx \partial_t^{\kappa-1} f(y(x_{i+j}, t^0))$ are given by

$$\widetilde{f}^{(\kappa-1)}_{i+j} := P^{(\kappa-1)} (\mathfrak{f}_T)^{\kappa-1,\langle 0 \rangle}_{i+j}, \quad j = -\overline{p}, \dots, \overline{p},$$

with

$$(\mathfrak{f}_T)^{\kappa-1,r}_{i+j} := f\left(y(x_{i+j}, t^0) + \sum_{\ell=1}^{\kappa-1} \frac{(r\Delta t)^{\ell}}{\ell!} \widetilde{y}^{(\ell)}_{i+j} \right),$$

for $r = -\overline{p}, \dots, \overline{p}$. Via the described steps, the values $\widetilde{f}^{(\kappa-1)}_{i+j}$ are recursively obtained.

A summary of the approximate Taylor (AT) procedure to obtain time-derivatives $\partial_t^{(\kappa)} y(x_i, t^0)$ is provided in Algorithm 2.

5 Numerical results

In this section, we show numerical experiments verifying the SSP theory, accuracy, and monotonicity properties of the constructed methods. To accomplish this, we consider several linear and nonlinear test cases with both smooth and discontinuous initial data. For accuracy tests, setups, where shock formation occurs, are avoided to obtain the expected order of convergence. Hence, we do not need to apply flux-limiting techniques in convergence studies. For monotonicity studies, on the other hand, it is vital to check the behavior of the numerical solutions close to a discontinuity or shock,

Algorithm 2 Values of initial vector $y^{[0]}$ obtained via an approximate Taylor procedure.

```
Time derivatives:
for $j = -\overline{p}$ to $\overline{p}$ do
    $\widetilde{f}^{(0)}_{i+j} = f(y(x_{i+j}, t^0))$
end
$\partial_t y(x_i, t^0) = -\frac{1}{\Delta x} \sum_{j=-\overline{p}}^{\overline{p}} \delta^1_{\overline{p},j} \widetilde{f}^{(0)}_{i+j}$
for $\kappa = 2$ to $p$ do
    Get $\widetilde{f}^{(\kappa-1)}_{i+j}$ via AT procedure.
    $\partial_t^\kappa y(x_i, t^0) =$
        $-\frac{1}{\Delta x} \sum_{j=-\overline{p}}^{\overline{p}} \delta^1_{\overline{p},j} \widetilde{f}^{(\kappa-1)}_{i+j}$
end
```

```
AT procedure [9] ($\kappa = 2, \dots, p$):
$\widetilde{y}^{(\kappa-1)}_{i+j} = -P^{(1)} \widetilde{\mathbf{f}}^{(\kappa-2)}_{\langle i \rangle}$
    $= -\frac{1}{\Delta x} \sum_{r=-\overline{p}}^{\overline{p}} \delta^1_{\overline{p},r} \widetilde{f}^{(\kappa-2)}_{i+r}$
for $r = -\overline{p}$ to $\overline{p}$ do
    $(\mathfrak{f}_T)^{\kappa-1,r}_{i+j} =$
        $f\left(y(x_{i+j}, t^0) + \sum_{\ell=1}^{\kappa-1} \frac{(r\Delta t)^\ell}{\ell!} \widetilde{y}^{(\ell)}_{i+j}\right)$
end
$\widetilde{f}^{(\kappa-1)}_{i+j} = P^{(\kappa-1)} (\mathfrak{f}_{\boldsymbol{T}})^{\kappa-1,\langle 0 \rangle}_{i+j}$
    $= \frac{1}{\Delta t^{\kappa-1}} \sum_{r=-\overline{p}}^{\overline{p}} \delta^{\kappa-1}_{\overline{p},r} (\mathfrak{f}_T)^{\kappa-1,r}_{i+j}$
```

which may produce strong non-physical numerical oscillations. To avoid the latter, we shall use flux limiters as in [9]. The van Albada flux limiter function is used here.

To measure accuracy, we use the scaled L_1-error at the final time $t^N \equiv T_f$ given by

$$\|\mathbf{y}(T_f) - \mathbf{y}^{[n]}\| := \Delta x \sum_{i=1}^{M} |y(x_i, T_f) - y_i^N|,$$

where $\mathbf{y}(T_f)$ represents a vector of reference solution values in the spatial grids $x_1, x_2, \dots, x_M$ at time T_f; and $\mathbf{y}^N$ stands for the vector of approximations y_i^N at time t^N.

In what follows, we perform several numerical experiments on a variety of 1d (scalar/systems of) conservation laws: Linear transport, Burgers, Buckley–Leverett, and Euler equations are used. In these test cases, both spatial and temporal grids are refined simultaneously by means of the relation

$$\Delta t := \frac{\sigma \Delta x}{\max_i |\lambda_{\mathrm{eig,i}}|},$$

where σ and $\lambda_{\mathrm{eig,i}}$ stand for a chosen CFL number, and the local eigenvalues of the Jacobian w.r.t initial value, respectively. For stable timestepping, we set $\sigma \leq \beta$, with β the obtained SSP coefficients of the new schemes. In all the test cases with smooth initial data, the expected order of convergence is preserved for the proposed Jacobian-free SSP multiderivative GLMs.

5.1 SSP property on the linear transport equation

To assess the TVD characteristics of the methods proposed in this study, we calculate the maximum observed rise in total variation during each computational step. This rise is defined through

$$\max_{1\leq n\leq N}\left(\|y^{n+1}\|_{TV}-\|y^{n}\|_{TV}\right). \tag{37}$$

Our primary objective is to identify the time step at which this maximum rise becomes significant, surpassing the threshold of 10^{-10}. We consider the linear advection problem:

$$\partial_t Y-\partial_x Y=0,$$

with domain $x\in[-2,2]$ and periodic boundary conditions. The initial condition is a step function defined as

$$Y_0(x)=\begin{cases}1 & \text{if } -\frac{1}{2}\leq x\leq\frac{1}{2},\\ 0 & \text{otherwise.}\end{cases}$$

We compute the spatial derivatives up to four using (20), (21), (24), and (25), respectively. As this testcase is meant to demonstrate the SSP abilities of our developed methods, we do not use both the CAT procedure and flux limiters in this subsection. Using a fixed grid size $\Delta x=\frac{1}{75}$ and a time-step $\Delta t=\text{CFL}\cdot\Delta x$, where the value of CFL ranges from 0.02 until reaching a point beyond which the TVD property is violated, we advance each method by $N=50$ time-steps. We then compare the performance of the developed time-stepping methods against non-SSP methods from the literature. In particular, we compare against the third-derivative method with $p=5$ and $s=2$ developed in [38], a third-derivative RK method with $p=7$ and $s=3$, and a fourth-derivative RK method with $p=6$ and $s=2$, the latter two derived in [11].

Figures 1 and 2 display the maximum per-step increase in total variation (37) for each CFL value $\frac{\Delta t}{\Delta x}$. First of all, it can be observed that the SSP schemes perform indeed as the name suggests, they preserve the total variation until a certain threshold CFL value, where a sudden 'shock' occurs in the total variation of the solution. It can be also seen from the picture that the β-value presented in Table 1 is typically way too pessimistic, the CFL threshold value is in all cases larger than the computed β, with the exception of 4DGLM6-3, where machine accuracy issues seem to play a role (note that the increase is smaller than 10^{-12} for this case for $\frac{\Delta t}{\Delta x}<1.819$).

Furthermore, our novel methods behave way better than the methods from literature which are not SSP, at least in the sense that the total variation is preserved.

5.2 Convergence and numerical stability studies

Problem 1 We consider again a linear advection problem $\partial_t Y+\partial_x Y=0$ with periodic boundary conditions and smooth sine-wave initial condition

$$Y(x,0)=\frac{1}{4}\sin(\pi x),\qquad x\in[0,2]. \tag{38}$$

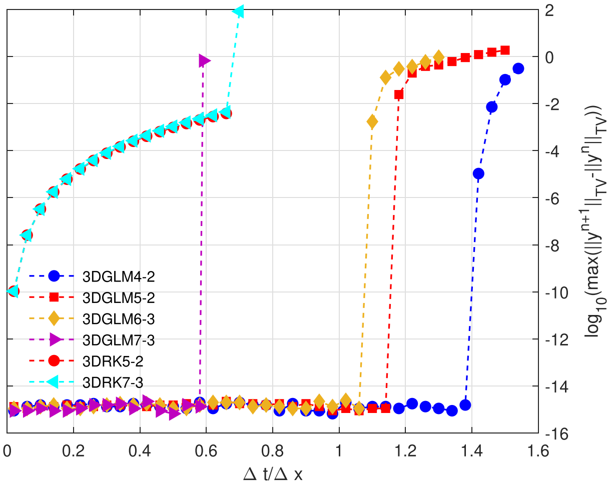


Fig. 1 Comparison of the rise in total variation as a function of the CFL number for third-derivative GLMs and RK methods

Adopting the Jacobian-free technique described in Algorithm 1, we apply the explicit SSP 3DGLMs and 4DGLMs constructed in this work together with the SSP 2DGLMs obtained in [31] to this problem. In order to calculate the L_1-error for the CAMDGLMs, we run the simulation up to $T_f = 0.8$ with co-refinement of the spatial and temporal grids. The results with different values of CFL corresponding to each method are visualized in Figs. 3 and 4, indicating that all expected convergence orders, $\min(2\overline{p}, p)$, are achieved. The compact approximate 3DGLM5-2 and 3DGLM6-3 behave in a very alike manner.

To show the capability of the constructed methods in preserving the stability properties near the appearance of shocks, we apply two stage compact approximate MDGLMs of order four with flux limiter technique: FL-2DGLM4-2, FL-3DGLM4-2 and FL-

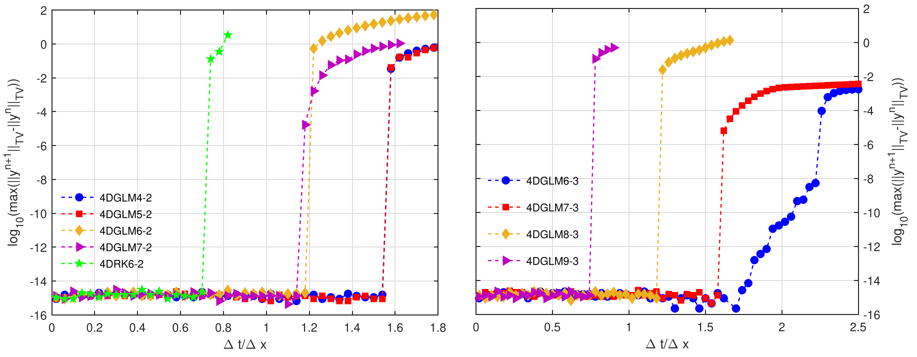


Fig. 2 Comparison of the rise in total variation as a function of the CFL number for fourth-derivative GLMs and RK methods

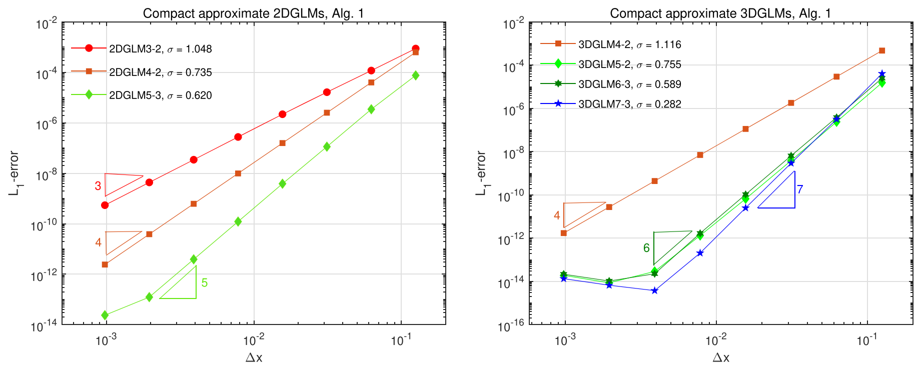


Fig. 3 Convergence order of explicit compact approximate SSP 2DGLMs (left) and 3DGLMs (right) applied to the linear transport equation on the sine wave $Y_0(x) = \frac{1}{4}\sin(\pi x)$ up to $T_f = 0.8$

4DGLM4-2, to the above-mentioned transport equation with initial condition

$$Y(x,0) = \begin{cases} 1, & 0 \le x < 0.2 \\ 2, & 0.2 \le x < 0.7 \\ 1, & 0.7 \le x \le 1 \end{cases} \tag{39}$$

and run the simulation up to $T_f = 1.0$ with $M = 100$ points and CFL values $\sigma = 0.8$ and $\sigma = 0.5$. The numerical simulations are shown in Fig. 5 where the van Albada flux limiter function is used. As it is clear, for $\sigma = 0.8$ the results given by FL-3DGLM4-2 and FL-4DGLM4-2 maintain stable near the discontinuities while FL-2DGLM4-2 with the same CFL value show strong oscillations. However, it can be seen from Fig. 5 (right) that for $\sigma = 0.5$, the FL-2DGLM4-2 behaves way better, yet, some slight oscillations still occur.

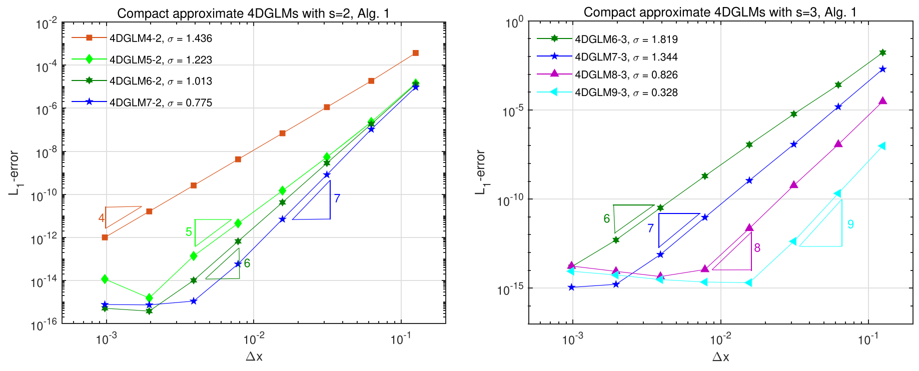


Fig. 4 Convergence order of explicit compact approximate SSP 4DGLMs applied to the linear transport equation on the sine wave $Y_0(x) = \frac{1}{4}\sin(\pi x)$ up to $T_f = 0.8$

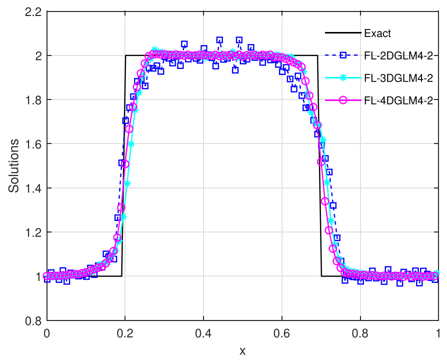


Fig. 5 Comparison of the performance of compact approximate MDGLMs of the same order for linear transport equation with initial condition (39), $\sigma = 0.8$ (left), $\sigma = 0.5$ (right) and $T_f = 1$

Problem 2 We consider Burger's equation

$$\partial_t Y + \partial_x \left(\frac{Y^2}{2} \right) = 0,$$

with the sine-wave initial condition (38) and periodic boundary conditions on the spatial domain $x \in [0, 2]$. As observed in [30, 48], a shock is formed at $t^* = \frac{4}{\pi} \approx 1.27$. To certify the theoretical accuracy of $\min(2\overline{p}, p)$, we set the final time to $T_f = 0.8$, before shock formation. As in the previous test case, we apply CAMDGLMs to this problem and run the simulations up to $T_f = 0.8$ to obtain L_1-errors when both the temporal and spatial grids are refined simultaneously. The obtained results are shown in Figs. 6 and 7. The orders coincide in all cases with the expected one, $\min(2\overline{p}, p)$. For the methods 3DGLM5-2 and 4DGLM5-2, the convergence order is slightly better than expected. Here the spatial order of accuracy is higher than the temporal one, and the spatial error dominates the overall behavior.

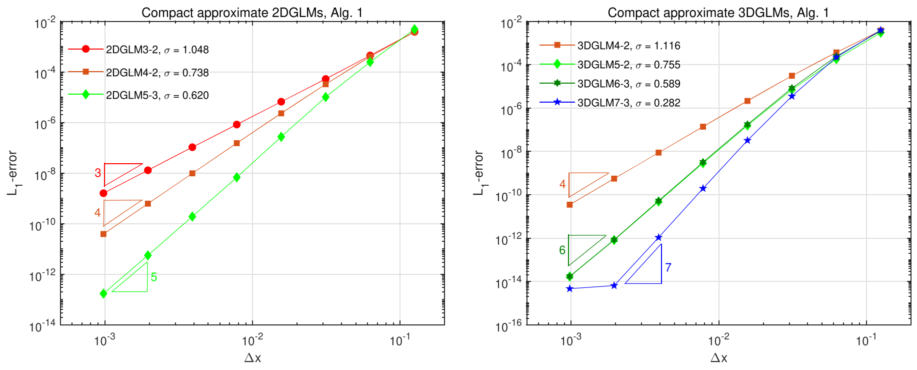


Fig. 6 Convergence order of explicit compact approximate SSP 2DGLMs and 3DGLMs applied to Burgers equation on the sine wave $Y_0(x) = \frac{1}{4}\sin(\pi x)$ up to $T_f = 0.8$

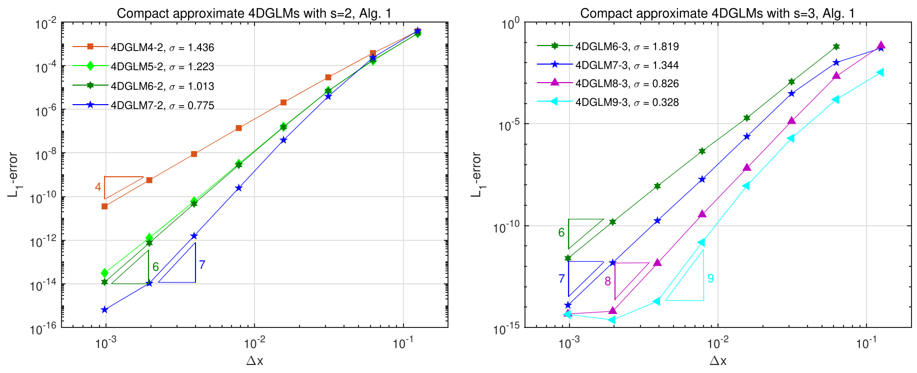


Fig. 7 Convergence order of explicit compact approximate SSP 4DGLMs applied to Burgers equation on the sine wave $Y_0(x) = \frac{1}{4}\sin(\pi x)$ up to $T_f = 0.8$

Next, the same problem with all the same parameters but different initial condition

$$Y(x,0) = \frac{1}{4}\exp(\cos(\pi x) + \sin(\pi x)),$$

is solved using FL-2DGLM4-2, FL-3DGLM4-2 and FL-4DGLM4-2. Using $\sigma = 0.8$ and $\sigma = 0.5$, a resolution of $M = 100$ points and $T_f = 1.0$, after shock formation –the breaking time is $t^* = \frac{4}{\pi e}$ [11]– the numerical results are depicted in Fig. 8 revealing that for $\sigma = 0.8$ (left side) the third-derivative and fourth-derivative GLMs with flux limiter function exhibit smooth behavior close to the shock while second-derivative GLM of the same order shows oscillations. It can also be seen from this figure (right side) that for $\sigma = 0.5$ all the methods behave correctly.

Problem 3 Next, we consider the Buckley–Leverett flux [30]

$$\partial_t Y + \partial_x \left(\frac{Y^2}{Y^2 + a(1-Y)^2} \right) = 0,$$

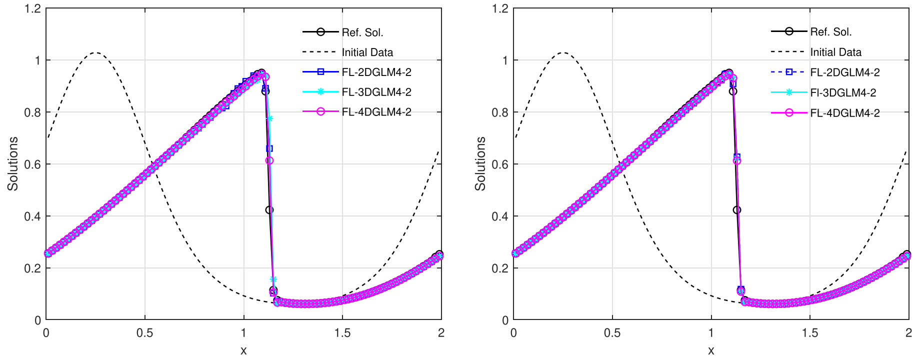


Fig. 8 Comparison of the performance of compact approximate MDGLMs of the same order for Burgers equation with initial condition $Y_0(x) = \frac{1}{4}\exp(\cos(\pi x) + \sin(\pi x))$, $\sigma = 0.8$ (left), $\sigma = 0.5$ (right) and $T_f = 1.0$

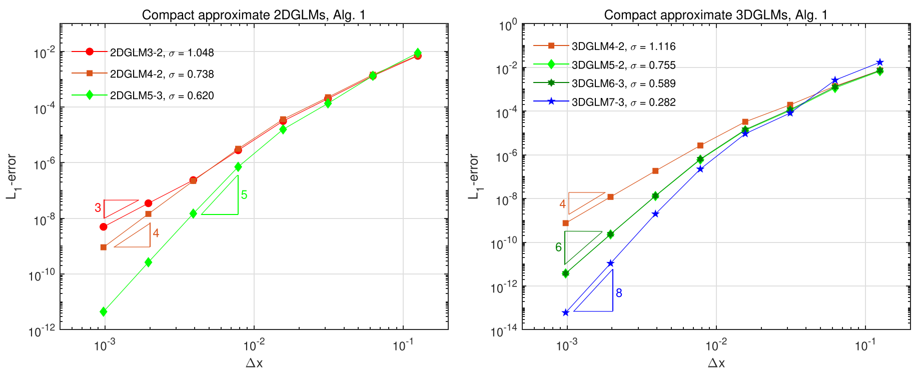


Fig. 9 Convergence order of explicit compact approximate SSP 2DGLMs and 3DGLMs applied to Buckley–Leverett equation with initial condition $Y_0(x) = 1 - \frac{3}{4}\cos^2(\frac{\pi}{2}x)$ up to $T_f = 0.1$

with $a = 1/4$. This equation models a two-phase flow through a porous medium. For this problem, we consider the initial condition

$$Y(x, 0) = 1 - \frac{3}{4}\cos^2\left(\frac{\pi}{2}x\right),$$

with periodic boundary conditions on domain $x \in [-1, 1]$. As a first test in this subsection, we set $T_f = 0.1$, so that we have a continuous solution which enables us to compute the exact solution via its characteristics. As in previous test cases, we refine both temporal and spatial grids concurrently and calculate the L_1-errors for CAMDGLMs. The convergence results are plotted in Figs. 9 and 10; the expected order of convergence are attained. Better performance than expected is observed for methods 2DGLM5-3, 3DGLM5-2, 4DGLM5-2, and 4DGLM7-3, as a result of the spatial order of accuracy being higher than the temporal one. Indeed, bearing in mind that the compact approximate approach has been investigated as a natural extension

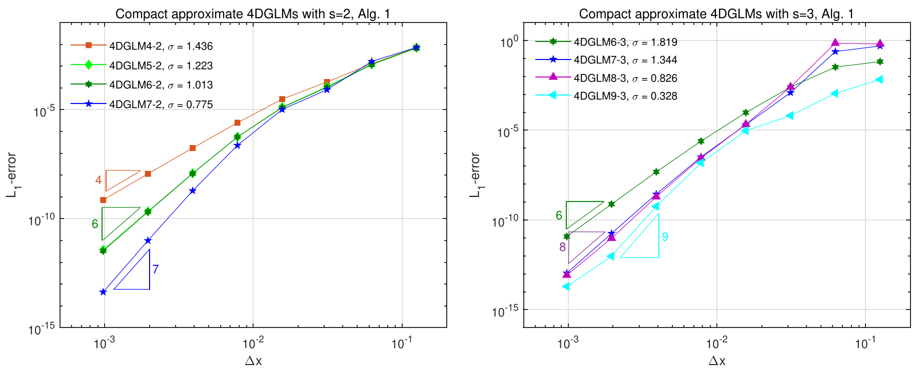


Fig. 10 Convergence order of explicit compact approximate SSP 4DGLMs applied to Buckley–Leverett equation with initial condition $Y_0(x) = 1 - \frac{3}{4}\cos^2(\frac{\pi}{2}x)$ up to $T_f = 0.1$

of Lax–Wendroff methods with an even-order accuracy, in most of the test cases when $2\overline{p} > p$, the odd-order MDGLMs take advantage and behave similar to the methods of order $2\overline{p}$.

In a similar way as for the previous examples, we conclude this test case by solving the same problem with all the same parameters except for the final time, which we set to $T_f = 0.4$, after shock appearance, and CFL numbers $\sigma = 0.8$ and $\sigma = 0.5$. Using two stage multiderivative GLMs of order four with the van Albada flux limiter function, we obtain numerical solutions with a quite similar behavior to those for the Burgers equation, as can be seen in the left side of Fig. 11: FL-3DGLM4-2 and FL-4DGLM4-2 indicate smooth behavior while 2DGLM4-2 shows numerical instability. The right side of this figure presents the behavior of the numerical solutions for $\sigma = 0.5$ indicating that all of the methods preserve the required stability property near shock.

Problem 4 Finally, we consider the nonlinear system of Euler equations defined by

$$\partial_t Y + \partial_x f(Y) = 0,$$

with

$$Y = \begin{pmatrix} \rho \\ \rho v \\ E \end{pmatrix}, \quad f(Y) = \begin{pmatrix} \rho v \\ \rho v^2 + P \\ v(E+P) \end{pmatrix},$$

where ρ stands for the density, E is the total energy, v is the velocity, and P is the pressure given by

$$P = (\gamma - 1)\left(E - \frac{1}{2}\rho v^2\right),$$

with $\gamma = 1.4$, for ideal gas [30, 48]. We consider the following initial condition

$$Y(x,0) = \frac{1}{4}\begin{pmatrix} 3 \\ 1 \\ 3 \end{pmatrix} + \frac{\sin(\pi x)}{2}\begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix},$$

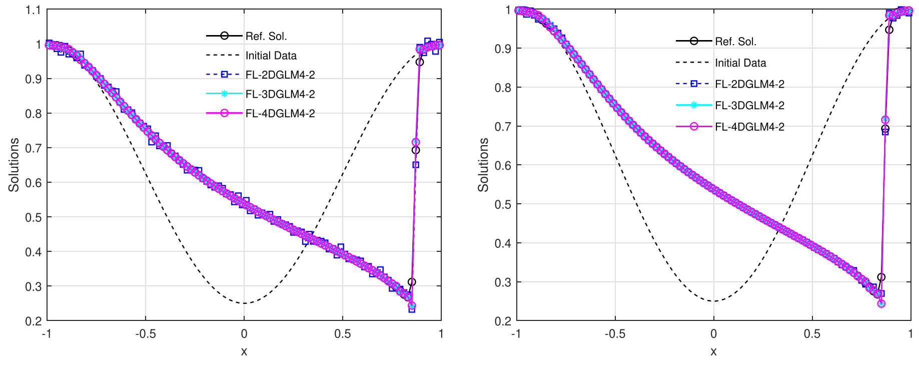


Fig. 11 Comparison of the performance of compact approximate MDGLMs of the same order for Buckley–Leverett equation with initial condition $Y_0(x) = 1 - \frac{3}{4}\cos^2(\frac{\pi}{2}x)$, $M = 100, \sigma = 0.8$ (left), $\sigma = 0.5$ (right) and $T_f = 0.4$

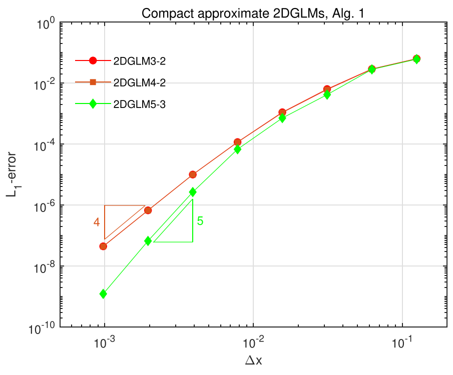


Fig. 12 Convergence order of explicit compact approximate SSP 2DGLMs and 3DGLMs applied to Euler equations with $\rho_0(x) = 0.75+0.5\sin(\pi x)$, $(\rho v)_0(x) = 0.25+0.5\sin(\pi x)$ and $E_0(x) = 0.75+0.5\sin(\pi x)$ on $x \in [0, 2]$ up to $T_f = 0.2$ with $\sigma = 0.15$

on $x \in [0, 2]$ with periodic boundary conditions and $T_f = 0.2$. To examine the accuracy of constructed methods for this test problem, a reference solution has been computed via a third-order discontinuous Galerkin (DG) method in space and a third-order SSP RK method in time [21] using $M = 10240$ cells and a CFL number of $\sigma = 0.15$. The L_1-errors for CAMDGLMs are calculated at final time $T_f = 0.2$. The convergence results are visualized in Figs. 12 and 13. It is clear from these figures that all expected orders were obtained. Very similar behavior can be seen between the methods that use the same $\overline{p}$.

As in the previous test cases, running the simulations for each methods with their corresponding CFL values, we observed that not all simulations were stable. Indeed, 3DGLMs and 4DGLMs with $s = 2$ diverged for $M = 16$, 32 and 64, and with $s = 3$ diverged for $M = 16$, 32, 64 and 128. In the case of 3DGLM7-3 and 4DGLM9-3, we

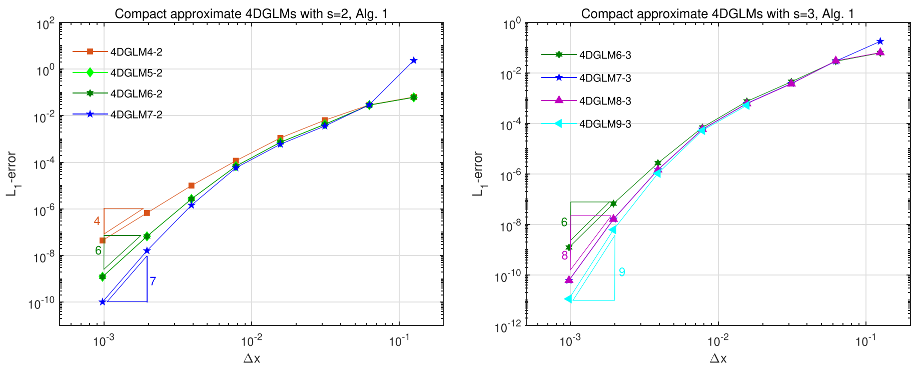


Fig. 13 Convergence order of explicit compact approximate SSP 4DGLMs applied to Euler equations with $\rho_0(x) = 0.75+0.5\sin(\pi x)$, $(\rho v)_0(x) = 0.25+0.5\sin(\pi x)$, and $E_0(x) = 0.75+0.5\sin(\pi x)$ on $x \in [0, 2]$ up to $T_f = 0.2$ with $\sigma = 0.15$

also observed divergence for $M = 16, 32$ and 64 (when the case is 4DGLM9-3) for CFL value $\sigma = 0.15$.

6 Conclusions

In this paper, following an introduction to multiderivative GLMs, we studied the monotonicity theory of multiderivative GLMs and derived sufficient conditions for multiderivative GLMs to be SSP. Solving an optimization problem with the aim of maximizing SSP coefficients and subject to the order and SSP conditions as equality and inequality constraints, respectively, we obtained high-order explicit SSP multiderivative GLMs with two external stages and $s = 2$ and $s = 3$ internal stages up to four derivatives and order nine for hyperbolic conservation laws. Such methods involve cumbersome calculation of the flux derivatives. To conquer this flaw, based on recent developments of explicit Jacobian-free multistage multiderivative solvers in [11], we adopted a Jacobian-free technique for multiderivative GLMs in which time derivatives of fluxes use local approximations based on discrete differentiations recursively. Aside from not requiring costly symbolic computations, by this Jacobian-free approach, higher-order multiderivative GLMs can be of more practical use in solving PDEs. Through a variety of numerical test cases, it is shown that the desired convergence order $\min(2\overline{p}, p)$ is attained, where $2\overline{p}$ stands for the spatial order and p for temporal order.

In future works, the novel scheme will be extended and applied to more challenging settings, for instance, a combination of MDGLMs and discontinuous Galerkin techniques [40] is of high interest. In the case of problems with diffusion, very small local mesh sizes are often needed and hence, acceptable time-steps become very small. In order to take care of these diffusive effects, or other "stiff" effects such as the singular perturbance that comes from low-Mach equations, the class of implicit and IMEX SSP MDGLMs with A- and L-stability properties will be considered.

Appendices

A Coefficients of third-derivative GLMs

Here, we give the coefficient matrices of the constructed third-derivative GLMs for $s = 2$ and $s = 3$. The coefficient matrices of two stage methods take the form

$$\left[\begin{array}{cc|cc|cc|cc}
0 & 0 & 0 & 0 & 0 & 0 & u_{11} & u_{12} \\
a_{21}^{\{1\}} & 0 & a_{21}^{\{2\}} & 0 & a_{21}^{\{3\}} & 0 & u_{21} & u_{22} \\
\hline
b_{11}^{\{1\}} & b_{12}^{\{1\}} & b_{11}^{\{2\}} & b_{12}^{\{2\}} & b_{11}^{\{3\}} & b_{12}^{\{3\}} & 1-v & v \\
b_{21}^{\{1\}} & b_{22}^{\{1\}} & b_{21}^{\{2\}} & b_{22}^{\{2\}} & b_{21}^{\{3\}} & b_{22}^{\{3\}} & 1-v & v
\end{array}\right],$$

and three stage methods take the following form

$$\left[\begin{array}{ccc|ccc|ccc|cc}
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & u_{11} & u_{12} \\
a_{21}^{\{1\}} & 0 & 0 & a_{21}^{\{2\}} & 0 & 0 & a_{21}^{\{3\}} & 0 & 0 & u_{21} & u_{22} \\
a_{31}^{\{1\}} & a_{32}^{\{1\}} & 0 & a_{31}^{\{2\}} & a_{32}^{\{2\}} & 0 & a_{31}^{\{3\}} & a_{32}^{\{3\}} & 0 & u_{31} & u_{32} \\
\hline
b_{11}^{\{1\}} & b_{12}^{\{1\}} & b_{13}^{\{1\}} & b_{11}^{\{2\}} & b_{12}^{\{2\}} & b_{13}^{\{2\}} & b_{11}^{\{3\}} & b_{12}^{\{3\}} & b_{13}^{\{3\}} & 1-v & v \\
b_{21}^{\{1\}} & b_{22}^{\{1\}} & b_{23}^{\{1\}} & b_{21}^{\{2\}} & b_{22}^{\{2\}} & b_{23}^{\{2\}} & b_{21}^{\{3\}} & b_{22}^{\{3\}} & b_{23}^{\{3\}} & 1-v & v
\end{array}\right].$$

1. SSP 3DGLM4-2

$a_{21}^{\{1\}} = 0.481524563110426$, $a_{21}^{\{2\}} = 0.107666492481270$, $a_{21}^{\{3\}} = 0.021992397120078$, $b_{11}^{\{1\}} = 0.498435904450369$,
$b_{12}^{\{1\}} = 0.522530753430964$, $b_{21}^{\{1\}} = 0.410124141823035$, $b_{22}^{\{1\}} = 0.463432449474462$, $b_{11}^{\{2\}} = 0.095786801825143$,
$b_{12}^{\{2\}} = 0.152075022730067$, $b_{21}^{\{2\}} = 0.137529315089621$, $b_{22}^{\{2\}} = 0.088062569968302$, $b_{11}^{\{3\}} = 0.018694337702556$,
$b_{12}^{\{3\}} = 0.027732589352203$, $b_{21}^{\{3\}} = 0.015749712492286$, $b_{22}^{\{3\}} = 0.005659319478742$, $u_{11} = 0.935650184213693$,
$u_{12} = 0.064349815786307$, $u_{21} = 0.886315321544333$, $u_{22} = 0.113684678455667$, $v = 0.142233555463516$,
$c_1 = 0.525747892280569$, $w_{10} = 1.0$, $w_{11} = 0.535233702910286$, $w_{12} = 0.136943388092356$,
$w_{13} = 0.027743278545692$, $w_{14} = 0.002690489400088$, $w_{20} = 1.0$, $w_{21} = 0.387823636326451$,
$w_{22} = 0.156555488157499$, $w_{23} = -0.027002414204259$, $w_{24} = 0.010351227670424$.

2. SSP 3DGLM5-2

$a_{21}^{\{1\}} = 0.444563089345968$, $a_{21}^{\{2\}} = 0.220433797971554$, $a_{21}^{\{3\}} = 0.047199086888770$, $b_{11}^{\{1\}} = 0.677030364676214$,
$b_{12}^{\{1\}} = 0.109344513124094$, $b_{21}^{\{1\}} = 0.600001026747633$, $b_{22}^{\{1\}} = 0.643878289010242$, $b_{11}^{\{2\}} = 0.187812461430404$,
$b_{12}^{\{2\}} = 0.167420842533560$, $b_{21}^{\{2\}} = 0.335797629018218$, $b_{22}^{\{2\}} = 0.027367894432842$, $b_{11}^{\{3\}} = 0.019973582828875$,
$b_{12}^{\{3\}} = 0.014734064127359$, $b_{21}^{\{3\}} = 0.076346090518484$, $b_{22}^{\{3\}} = 0.007807990410220$, $u_{11} = 0.691735860871995$,
$u_{12} = 0.308264139128005$, $u_{21} = 0.362379132344019$, $u_{22} = 0.637620867655981$, $v = 0.466935628326092$,
$c_1 = 0.404754745681296$, $w_{10} = 1.0$, $w_{11} = 0.263722533967065$, $w_{12} = 0.065333656992811$,
$w_{13} = 0.027169926386350$, $w_{14} = 0.002545864356240$, $w_{15} = -0.001260324593837$, $w_{20} = 1.0$,
$w_{21} = 0.721226971924633$, $w_{22} = 0.119117224325207$, $w_{23} = -0.025117506961899$, $w_{24} = -0.002085128124951$,
$w_{25} = 0.003121799371033$.

3. SSP 3DGLM6-3

$a_{21}^{\{1\}} = 0.315517321907129$, $a_{31}^{\{1\}} = 0.758864601103138$, $a_{32}^{\{1\}} = 0.067336415726727$, $a_{21}^{\{2\}} = 0.076494847917531$,
$a_{31}^{\{2\}} = 0.165996492414755$, $a_{32}^{\{2\}} = 0.131818841412823$, $a_{21}^{\{3\}} = 0.026098214000471$, $a_{31}^{\{3\}} = 0.010407236225120$,
$a_{32}^{\{3\}} = 0.011694043676070$, $b_{11}^{\{1\}} = 0.466407769409968$, $b_{12}^{\{1\}} = 0.588029782942878$, $b_{13}^{\{1\}} = 0.090753993408768$,
$b_{21}^{\{1\}} = 0.714352536776548$, $b_{22}^{\{1\}} = 0.004057520548612$, $b_{23}^{\{1\}} = 0.043314290009643$, $b_{11}^{\{2\}} = 0.101470387228004$,
$b_{12}^{\{2\}} = 0.309600095382871$, $b_{13}^{\{2\}} = 0.003137222126554$, $b_{21}^{\{2\}} = 0.208922227866803$, $b_{22}^{\{2\}} = 0.036586254679526$,
$b_{23}^{\{2\}} = 0.023218417766703$, $b_{11}^{\{3\}} = 0.016855407713879$, $b_{12}^{\{3\}} = 0.051517368037844$, $b_{13}^{\{3\}} = 0.001065531053630$,
$b_{21}^{\{3\}} = 0.035888517238621$, $b_{22}^{\{3\}} = 0.000525893188072$, $b_{23}^{\{3\}} = 0.007526287669863$, $u_{11} = 0.429393552342717$,
$u_{12} = 0.570606447657283$, $u_{21} = 0.714766096918914$, $u_{22} = 0.285233903081086$, $u_{31} = 0.373124486427176$,
$u_{32} = 0.626875513572824$, $v = 0.378628332116197$, $c_1 = 0.195376324234861$, $c_2 = 0.620324656318555$,
$w_{10} = 1.0$, $w_{11} = 0.414185180122275$, $w_{12} = 0.089420710803121$, $w_{13} = -0.015802200386162$,
$w_{14} = -0.001623142712040$, $w_{15} = 0.000303750873378$, $w_{16} = 0.218129334015903$, $w_{20} = 1.0$,
$w_{21} = 0.030717981695463$, $w_{22} = -0.033842454301531$, $w_{23} = 0.014069844764195$, $w_{24} = 0.001327849153771$,
$w_{25} = -0.000224421436006$, $w_{26} = 0.217923761039387$.

4. SSP 3DGLM7-3

$a_{21}^{\{1\}} = 0.379665884421877,$ $a_{31}^{\{1\}} = 0.515488221507126,$ $a_{32}^{\{1\}} = 0.008239732993902,$ $a_{21}^{\{2\}} = 0.060210073470162,$

$a_{31}^{\{2\}} = 0.114667967793747,$ $a_{32}^{\{2\}} = 0.010050929935034,$ $a_{21}^{\{3\}} = 0.027175182796611,$ $a_{31}^{\{3\}} = 0.093175561916561,$

$a_{32}^{\{3\}} = 0.008173511911974,$ $b_{11}^{\{1\}} = 1.107792953001359,$ $b_{12}^{\{1\}} = 0.090330329859403,$ $b_{13}^{\{1\}} = 0.000000066635348,$

$b_{21}^{\{1\}} = 0.664653168823024,$ $b_{22}^{\{1\}} = 0.309711371128129,$ $b_{23}^{\{1\}} = 0.015546721568938,$ $b_{11}^{\{2\}} = 0.189649927487829,$

$b_{12}^{\{2\}} = 0.600387177342980,$ $b_{13}^{\{2\}} = 0.000000000094460,$ $b_{21}^{\{2\}} = 0.107129004361048,$ $b_{22}^{\{2\}} = 0.214103657229089,$

$b_{23}^{\{2\}} = 0.018964113794848,$ $b_{11}^{\{3\}} = 0.003171262661914,$ $b_{12}^{\{3\}} = 0.067436297519191,$ $b_{13}^{\{3\}} = 0.000000000076816,$

$b_{21}^{\{3\}} = 0.006076703572795,$ $b_{22}^{\{3\}} = 0.000052086287613,$ $b_{23}^{\{3\}} = 0.015421809057899,$ $u_{11} = 0.050799054224709,$

$u_{12} = 0.949200945775291,$ $u_{21} = 0.271811785253415,$ $u_{22} = 0.728188214746585,$ $u_{31} = 0.848358688706512,$

$u_{32} = 0.151641311293488,$ $v = 0.951545856064458,$ $c_1 = 0.310210488718125,$ $c_2 = 0.735893895336771,$

$w_{10} = 1.0,$ $w_{11} = 0.507845599546811,$ $w_{12} = 0.239955454479258,$ $w_{13} = -0.006520488569236,$

$w_{14} = -0.005551486351131,$ $w_{15} = 0.000113698479916,$ $w_{16} = 0.000223765806822,$ $w_{17} = 1.999940921849224,$

$w_{20} = 1.0,$ $w_{21} = 0.299633511570792,$ $w_{22} = 0.037848427850458,$ $w_{23} = 0.005590515047944,$

$w_{24} = 0.000703599013788,$ $w_{25} = 0.000019134996526,$ $w_{26} = -0.000010671520193,$ $w_{27} = 2.0.$

B Coefficients of Fourth-derivative GLMs

Here, we give the coefficients matrices of the constructed fourth-derivative GLMs for $s = 2$ and $s = 3$. The coefficient matrices of two stage methods take the form

$$\left[\begin{array}{cc|cc|cc|cc|cc}
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & u_{11} & u_{12} \\
a_{21}^{\{1\}} & 0 & a_{21}^{\{2\}} & 0 & a_{21}^{\{3\}} & 0 & a_{21}^{\{4\}} & 0 & u_{21} & u_{22} \\
\hline
b_{11}^{\{1\}} & b_{12}^{\{1\}} & b_{11}^{\{2\}} & b_{12}^{\{2\}} & b_{11}^{\{3\}} & b_{12}^{\{3\}} & b_{11}^{\{4\}} & b_{12}^{\{4\}} & 1-v & v \\
b_{21}^{\{1\}} & b_{22}^{\{1\}} & b_{21}^{\{2\}} & b_{22}^{\{2\}} & b_{21}^{\{3\}} & b_{22}^{\{3\}} & b_{21}^{\{4\}} & b_{22}^{\{4\}} & 1-v & v
\end{array}\right],$$

and three stage methods take the following form

$$\left[\begin{array}{ccc|ccc|ccc|ccc|cc}
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & u_{11} & u_{12} \\
a_{21}^{\{1\}} & 0 & 0 & a_{21}^{\{2\}} & 0 & 0 & a_{21}^{\{3\}} & 0 & 0 & a_{21}^{\{4\}} & 0 & 0 & u_{21} & u_{22} \\
a_{31}^{\{1\}} & a_{32}^{\{1\}} & 0 & a_{31}^{\{2\}} & a_{32}^{\{2\}} & 0 & a_{31}^{\{3\}} & a_{32}^{\{3\}} & 0 & a_{31}^{\{4\}} & a_{32}^{\{3\}} & 0 & u_{31} & u_{32} \\
\hline
b_{11}^{\{1\}} & b_{12}^{\{1\}} & b_{13}^{\{1\}} & b_{11}^{\{2\}} & b_{12}^{\{2\}} & b_{13}^{\{2\}} & b_{11}^{\{3\}} & b_{12}^{\{3\}} & b_{13}^{\{3\}} & b_{11}^{\{4\}} & b_{12}^{\{4\}} & b_{13}^{\{4\}} & 1-v & v \\
b_{21}^{\{1\}} & b_{22}^{\{1\}} & b_{23}^{\{1\}} & b_{21}^{\{2\}} & b_{22}^{\{2\}} & b_{23}^{\{2\}} & b_{21}^{\{3\}} & b_{22}^{\{3\}} & b_{23}^{\{3\}} & b_{21}^{\{4\}} & b_{22}^{\{4\}} & b_{23}^{\{4\}} & 1-v & v
\end{array}\right].$$

1. SSP 4DGLM4-2

$a_{21}^{\{1\}} = 0.492502401255857,$ $a_{21}^{\{2\}} = 0.121549058540707,$ $a_{21}^{\{3\}} = 0.019860332137553,$ $a_{21}^{\{4\}} = 0.002469334621926,$

$b_{11}^{\{1\}} = 0.493035280152582,$ $b_{12}^{\{1\}} = 0.519129890196680,$ $b_{21}^{\{1\}} = 0.492159016664885,$ $b_{22}^{\{1\}} = 0.507134722289626,$

$b_{11}^{\{2\}} = 0.120822362679484,$ $b_{12}^{\{2\}} = 0.118796068553098,$ $b_{21}^{\{2\}} = 0.121514758761671,$ $b_{22}^{\{2\}} = 0.129069521144118,$

$b_{11}^{\{3\}} = 0.019733454185988,$ $b_{12}^{\{3\}} = 0.020130886150193,$ $b_{21}^{\{3\}} = 0.019811609177391,$ $b_{22}^{\{3\}} = 0.021899025203918,$

$$
\begin{aligned}
b_{11}^{\{4\}} &= 0.002452844681870, & b_{12}^{\{4\}} &= 0.002512408254589, & b_{21}^{\{4\}} &= 0.002463116135828, & b_{22}^{\{4\}} &= 0.002754839201374,\\
u_{11} &= 0.043647614240132, & u_{12} &= 0.956352385759868, & u_{21} &= 0.054763091495361, & u_{22} &= 0.945236908504639,\\
v &= 0.945129564550485, & c_1 &= 0.507354526641232, & w_{10} &= 1.0, & w_{11} &= 0.519664150763746,\\
w_{12} &= 0.117804247188642, & w_{13} &= 0.020428440495935, & w_{14} &= 0.002458895114308, & w_{20} &= 1.0,\\
w_{21} &= 0.506792719368996, & w_{22} &= 0.129201783102532, & w_{23} &= 0.021827294362888, & w_{24} &= 0.002774578642803.
\end{aligned}
$$

2. SSP 4DGLM5-2

$$
\begin{aligned}
a_{21}^{\{1\}} &= 0.500492666826293, & a_{21}^{\{2\}} &= 0.114756600078513, & a_{21}^{\{3\}} &= 0.018148178751572, & a_{21}^{\{4\}} &= 0.002714589878061,\\
b_{11}^{\{1\}} &= 0.431186537508394, & b_{12}^{\{1\}} &= 0.404171392100681, & b_{21}^{\{1\}} &= 0.483662234321112, & b_{22}^{\{1\}} &= 0.560179604710392,\\
b_{11}^{\{2\}} &= 0.081227126476428, & b_{12}^{\{2\}} &= 0.047086826677170, & b_{21}^{\{2\}} &= 0.109480320931676, & b_{22}^{\{2\}} &= 0.135274055565348,\\
b_{11}^{\{3\}} &= 0.013178225742208, & b_{12}^{\{3\}} &= 0.008882729874764, & b_{21}^{\{3\}} &= 0.017492912408121, & b_{22}^{\{3\}} &= 0.025272005673387,\\
b_{11}^{\{4\}} &= 0.001589026148517, & b_{12}^{\{4\}} &= 0.000129548838296, & b_{21}^{\{4\}} &= 0.002446465182670, & b_{22}^{\{4\}} &= 0.003740837321860,\\
u_{11} &= 0.088185135295424, & u_{12} &= 0.911814864704576, & u_{21} &= 0.176145583277641, & u_{22} &= 0.823854416722359,\\
v &= 0.789711162108572, & c_1 &= 0.517845671243588, & w_{10} &= 1.0, & w_{11} &= 0.327746943580494,\\
w_{12} &= 0.050980139124519, & w_{13} &= 0.011125839350588, & w_{14} &= -0.001282009647015, & w_{15} &= 0.000603507965945,\\
w_{20} &= 1.0, & w_{21} &= 0.536230853002924, & w_{22} &= 0.142119178095184, & w_{23} &= 0.024306987806757,\\
w_{24} &= 0.003410108652328, & w_{25} &= 0.000281973075456.
\end{aligned}
$$

3. SSP 4DGLM6-2

$$
\begin{aligned}
a_{21}^{\{1\}} &= 0.517293370447585, & a_{21}^{\{2\}} &= 0.133925352925481, & a_{21}^{\{3\}} &= 0.027371115505543, & a_{21}^{\{4\}} &= 0.004940436426576,\\
b_{11}^{\{1\}} &= 0.562577084182751, & b_{12}^{\{1\}} &= 0.421763934840916, & b_{21}^{\{1\}} &= 0.572954574408108, & b_{22}^{\{1\}} &= 0.478728558134980,\\
b_{11}^{\{2\}} &= 0.163935914431203, & b_{12}^{\{2\}} &= 0.100241069539339, & b_{21}^{\{2\}} &= 0.170412942747712, & b_{22}^{\{2\}} &= 0.165570428503403,\\
b_{11}^{\{3\}} &= 0.035658092350845, & b_{12}^{\{3\}} &= 0.016327465256022, & b_{21}^{\{3\}} &= 0.037215832658113, & b_{22}^{\{3\}} &= 0.035849937970705,\\
b_{11}^{\{4\}} &= 0.005287177412273, & b_{12}^{\{4\}} &= 0.002944889939538, & b_{21}^{\{4\}} &= 0.006694878863016, & b_{22}^{\{4\}} &= 0.000000025053928,\\
u_{11} &= 0.991894960507822, & u_{12} &= 0.008105039492178, & u_{21} &= 0.677452760618707, & u_{22} &= 0.322547239381293,\\
v &= 0.232528801933385, & c_1 &= 0.461531427232185, & w_{10} &= 1.0, & w_{11} &= 0.460985616742624,\\
w_{12} &= 0.105968926131271, & w_{13} &= 0.016230449697124, & w_{14} &= 0.001904410077557, & w_{15} &= 0.000179896607685,\\
w_{16} &= 0.000011946083550, & w_{20} &= 1.0, & w_{21} &= 0.528327730262044, & w_{22} &= 0.172187361061285,\\
w_{23} &= 0.035327454889158, & w_{24} &= 0.000197418142938, & w_{25} &= 0.000484463733041, & w_{26} &= 0.000194265652679.
\end{aligned}
$$

4. SSP 4DGLM7-2

$$
\begin{aligned}
a_{21}^{\{1\}} &= 0.499507160885063, & a_{21}^{\{2\}} &= 0.119968933288827, & a_{21}^{\{3\}} &= 0.017081896566529, & a_{21}^{\{4\}} &= 0.004031320198658,\\
b_{11}^{\{1\}} &= 0.666614451332563, & b_{12}^{\{1\}} &= 0.303306256194171, & b_{21}^{\{1\}} &= 0.576409644843040, & b_{22}^{\{1\}} &= 0.462190555937428,\\
b_{11}^{\{2\}} &= 0.183560532895898, & b_{12}^{\{2\}} &= 0.155207228786631, & b_{21}^{\{2\}} &= 0.259780581201447, & b_{22}^{\{2\}} &= 0.034723177133322,\\
b_{11}^{\{3\}} &= 0.031183451078632, & b_{12}^{\{3\}} &= 0.041000165610189, & b_{21}^{\{3\}} &= 0.057101861689900, & b_{22}^{\{3\}} &= 0.010915027907463,\\
b_{11}^{\{4\}} &= 0.001433739889497, & b_{12}^{\{4\}} &= 0.004322242467946, & b_{21}^{\{4\}} &= 0.001529153305801, & b_{22}^{\{4\}} &= 0.002575963845641,\\
u_{11} &= 0.618158814106136, & u_{12} &= 0.381841185893864, & u_{21} &= 0.688026611571673, & u_{22} &= 0.311973388428327,
\end{aligned}
$$

$v = 0.437966138773607$, $c_1 = 0.505291324039624$, $w_{10} = 1.0$, $w_{11} = 0.479066664889029$,

$w_{12} = 0.127521779354434$, $w_{13} = 0.041705623460671$, $w_{14} = 0.003986082383866$, $w_{15} = -0.000245284111217$,

$w_{16} = 0.000137223973872$, $w_{17} = 0.210228581679632$, $w_{20} = 1.0$, $w_{21} = 0.547746158142762$,

$w_{22} = 0.127882876390368$, $w_{23} = -0.011206034972369$, $w_{24} = 0.000660308249256$, $w_{25} = 0.001115949397675$

$w_{26} = -0.000161611421203$, $w_{27} = 0.210270682891451$.

5. SSP 4DGLM6-3

$a_{21}^{\{1\}} = 0.374778451641205$, $a_{21}^{\{2\}} = 0.063406760818824$, $a_{21}^{\{3\}} = 0.008463987299019$, $a_{21}^{\{4\}} = 0.000850022276759$,
$a_{31}^{\{1\}} = 0.357597546030056$, $a_{31}^{\{2\}} = 0.062908428776938$, $a_{31}^{\{3\}} = 0.008164853981020$, $a_{31}^{\{4\}} = 0.000653850758305$,
$a_{32}^{\{1\}} = 0.325477000217070$, $a_{32}^{\{2\}} = 0.046441087898065$, $a_{32}^{\{3\}} = 0.005297221144835$, $a_{32}^{\{4\}} = 0.000532701706906$,
$b_{11}^{\{1\}} = 0.334959394295523$, $b_{12}^{\{1\}} = 0.315171444182360$, $b_{13}^{\{1\}} = 0.410190435750393$, $b_{21}^{\{1\}} = 0.349958069221630$,
$b_{22}^{\{1\}} = 0.292876373365019$, $b_{23}^{\{1\}} = 0.258165929344463$, $b_{11}^{\{2\}} = 0.060633404000675$, $b_{12}^{\{2\}} = 0.041104145868688$,
$b_{13}^{\{2\}} = 0.041008760892016$, $b_{21}^{\{2\}} = 0.055477164332920$, $b_{22}^{\{2\}} = 0.047166235937992$, $b_{23}^{\{2\}} = 0.051817450556552$,
$b_{11}^{\{3\}} = 0.007643277336581$, $b_{12}^{\{3\}} = 0.004674306560848$, $b_{13}^{\{3\}} = 0.005497486923106$, $b_{21}^{\{3\}} = 0.007256291024332$,
$b_{22}^{\{3\}} = 0.005093765887795$, $b_{23}^{\{3\}} = 0.006947776401714$, $b_{11}^{\{4\}} = 0.000621015210638$, $b_{12}^{\{4\}} = 0.000470045478575$,
$b_{13}^{\{4\}} = 0.000552841163009$, $b_{21}^{\{4\}} = 0.000622129834196$, $b_{22}^{\{4\}} = 0.000341706070841$, $b_{23}^{\{4\}} = 0.000698685934132$,
$u_{11} = 0.630474771711679$, $u_{12} = 0.369525228288321$, $u_{21} = 0.573012085130541$, $u_{22} = 0.426987914869459$,
$u_{31} = 0.601083078559332$, $u_{32} = 0.398916921440668$, $v = 0.378614942286516$, $c_1 = 0.321608164825949$,
$c_2 = 0.687231609392625$, $w_{10} = 1.0$, $w_{11} = 0.380481257618408$, $w_{12} = 0.048570228152514$,

$w_{13} = 0.003531837263071$, $w_{14} = 0.000965597584732$ $w_{15} = -0.000093904905574$, $w_{16} = 0.837871652069336$,
$w_{20} = 1.0$, $w_{21} = 0.221160355321244$, $w_{22} = 0.057082983028313$, $w_{23} = 0.008977334456590$,
$w_{24} = -0.000441185296213$, $w_{25} = 0.000237808975927$, $w_{26} = 0.837823139921717$.

6. SSP 4DGLM7-3

$a_{21}^{\{1\}} = 0.350457763059077$, $a_{21}^{\{2\}} = 0.059030590724736$, $a_{21}^{\{3\}} = 0.006167354392834$, $a_{21}^{\{4\}} = 0.000820001027662$,
$a_{31}^{\{1\}} = 0.387500300301159$, $a_{31}^{\{2\}} = 0.084838889640679$, $a_{31}^{\{3\}} = 0.012031303235926$, $a_{31}^{\{4\}} = 0.000492349845276$,
$a_{32}^{\{1\}} = 0.312881241973284$, $a_{32}^{\{2\}} = 0.056243073304936$, $a_{32}^{\{3\}} = 0.009017317429851$, $a_{32}^{\{4\}} = 0.001125124849635$,
$b_{11}^{\{1\}} = 0.369607055094061$, $b_{12}^{\{1\}} = 0.403154224734759$, $b_{13}^{\{1\}} = 0.127967119709400$, $b_{21}^{\{1\}} = 0.390879999306208$,
$b_{22}^{\{1\}} = 0.239443987884715$, $b_{23}^{\{1\}} = 0.494290470310205$, $b_{11}^{\{2\}} = 0.058565784856034$, $b_{12}^{\{2\}} = 0.104280749110758$,
$b_{13}^{\{2\}} = 0.008303419949253$, $b_{21}^{\{2\}} = 0.087915116062722$, $b_{22}^{\{2\}} = 0.058148914188203$, $b_{23}^{\{2\}} = 0.049797213094552$,
$b_{11}^{\{3\}} = 0.006572057700620$, $b_{12}^{\{3\}} = 0.018705647077826$, $b_{13}^{\{3\}} = 0.001466993217758$, $b_{21}^{\{3\}} = 0.012839897311860$,
$b_{22}^{\{3\}} = 0.006947653665000$, $b_{23}^{\{3\}} = 0.008829144278492$, $b_{11}^{\{4\}} = 0.000610929864500$, $b_{12}^{\{4\}} = 0.002527896988956$,
$b_{13}^{\{4\}} = 0.000199724984841$, $b_{21}^{\{4\}} = 0.000420835098031$, $b_{22}^{\{4\}} = 0.000858837048376$, $b_{23}^{\{4\}} = 0.001202326356901$,
$u_{11} = 0.558164113618959$, $u_{12} = 0.441835886381041$, $u_{21} = 0.589995146907839$, $u_{22} = 0.410004853092161$,
$u_{31} = 0.537204663265761$, $u_{32} = 0.462795336734239$, $v = 0.443402333155673$, $c_1 = 0.294925929008912$,
$c_2 = 0.638257167504055$, $w_{10} = 1.0$, $w_{11} = 0.196005034140514$, $w_{12} = 0.013033449258411$,
$w_{13} = 0.007614583616032$, $w_{14} = -0.001144243875588$, $w_{15} = -0.000104282181364$, $w_{16} = 0.000081762590702$,
$w_{17} = 0.749648748323158$, $w_{20} = 1.0$, $w_{21} = 0.419891092103421$, $w_{22} = 0.081966742096937$,
$w_{23} = 0.000057305547194$, $w_{24} = 0.002158981114169$, $w_{25} = 0.000173822525851$, $w_{26} = -0.000101220718724$,
$w_{27} = 0.749692058749461$.

7. SSP 4DGLM8-3

$a_{21}^{\{1\}} = 0.175790178224457,\quad a_{21}^{\{2\}} = 0.053976619001249,\quad a_{21}^{\{3\}} = 0.014560267078991,\quad a_{21}^{\{4\}} = 0.002914857159230,$

$a_{31}^{\{1\}} = 0.193854979888018,\quad a_{31}^{\{2\}} = 0.027727867380224,\quad a_{31}^{\{3\}} = 0.005382677196853,\quad a_{31}^{\{4\}} = 0.001097886442691,$

$a_{32}^{\{1\}} = 0.344954266207200,\quad a_{32}^{\{2\}} = 0.013247065452726,\quad a_{32}^{\{3\}} = 0.001182532004156,\quad a_{32}^{\{4\}} = 0.000163946507126,$

$b_{11}^{\{1\}} = 0.350470082796973,\quad b_{12}^{\{1\}} = 0.207739507894165,\quad b_{13}^{\{1\}} = 0.323978091749971,\quad b_{21}^{\{1\}} = 0.336851998589390,$

$b_{22}^{\{1\}} = 0.379664713420180,\quad b_{23}^{\{1\}} = 0.475765720663504,\quad b_{11}^{\{2\}} = 0.055651506187291,\quad b_{12}^{\{2\}} = 0.102852985251848,$

$b_{13}^{\{2\}} = 0.027042199636409,\quad b_{21}^{\{2\}} = 0.114564248168959,\quad b_{22}^{\{2\}} = 0.121907339183384,\quad b_{23}^{\{2\}} = 0.134966509007059,$

$b_{11}^{\{3\}} = 0.005747532077953,\quad b_{12}^{\{3\}} = 0.007546452267841,\quad b_{13}^{\{3\}} = 0.007094916636282,\quad b_{21}^{\{3\}} = 0.030041698640272,$

$b_{22}^{\{3\}} = 0.027573371342666,\quad b_{23}^{\{3\}} = 0.038269689121708,\quad b_{11}^{\{4\}} = 0.001039724633659,\quad b_{12}^{\{4\}} = 0.000079054390091,$

$b_{13}^{\{4\}} = 0.001572203392119,\quad b_{21}^{\{4\}} = 0.003491131979020,\quad b_{22}^{\{4\}} = 0.006063710347766,\quad b_{23}^{\{4\}} = 0.002985797295388,$

$u_{11} = 0.991567328795643,\quad u_{12} = 0.008432671204357,\quad u_{21} = 0.184632179631794,\quad u_{22} = 0.815367820368206,$

$u_{31} = 0.678379155693658,\quad u_{32} = 0.321620844306342,\quad v = 0.379923611962994,\quad c_1 = 0.364072745591117,$

$c_2 = 0.790089277348930,\quad w_{10} = 1.0,\quad w_{11} = 0.361457818520213,\quad w_{12} = 0.064938219272829,$

$w_{13} = 0.007747210371254,\quad w_{14} = 0.000708041762913,\quad w_{15} = 0.000054105005786,\quad w_{16} = 0.000003491031849,$

$w_{17} = 0.000000163872239,\quad w_{18} = 0.982635662271492,\quad w_{20} = 1.0,\quad w_{21} = 0.671552568752178,$

$w_{22} = 0.223400791323222,\quad w_{23} = 0.042813264929839,\quad w_{24} = 0.003555229580558,\quad w_{25} = -0.000040886308458,$

$w_{26} = -0.000026939548375,\quad w_{27} = 0.000000679871749,\quad w_{28} = 0.982637254619139.$

8. SSP 4DGLM9-3

$a_{21}^{\{1\}} = 0.568808081207977,\quad a_{21}^{\{2\}} = 0.048114396995161,\quad a_{21}^{\{3\}} = 0.010788380615035,\quad a_{21}^{\{4\}} = 0.006008124562918,$

$a_{31}^{\{1\}} = 0.344187432821678,\quad a_{31}^{\{2\}} = 0.042382496857406,\quad a_{31}^{\{3\}} = 0.003582825049991,\quad a_{31}^{\{4\}} = 0.000689205385252,$

$a_{32}^{\{1\}} = 0.340398899955966,\quad a_{32}^{\{2\}} = 0.014335262109803,\quad a_{32}^{\{3\}} = 0.000313906335632,\quad a_{32}^{\{4\}} = 0.000145276246639,$

$b_{11}^{\{1\}} = 0.982954717825614,\quad b_{12}^{\{1\}} = 0.014490770169346,\quad b_{13}^{\{1\}} = 0.013584892713300,\quad b_{21}^{\{1\}} = 0.443809258398672,$

$b_{22}^{\{1\}} = 0.135975551664887,\quad b_{23}^{\{1\}} = 0.118857631207925,\quad b_{11}^{\{2\}} = 0.212259343381131,\quad b_{12}^{\{2\}} = 0.273158542191652,$

$b_{13}^{\{2\}} = 0.002916848832074,\quad b_{21}^{\{2\}} = 0.142520928035879,\quad b_{22}^{\{2\}} = 0.136853198913849,\quad b_{23}^{\{2\}} = 0.018625818520990,$

$b_{11}^{\{3\}} = 0.018327446754578,\quad b_{12}^{\{3\}} = 0.002170604033030,\quad b_{13}^{\{3\}} = 0.002168545523328,\quad b_{21}^{\{3\}} = 0.056096865472845,$

$b_{22}^{\{3\}} = 0.096389327446308,\quad b_{23}^{\{3\}} = 0.012468767290534,\quad b_{11}^{\{4\}} = 0.000413806824034,\quad b_{12}^{\{4\}} = 0.000000761021627,$

$b_{13}^{\{4\}} = 0.001208578960785,\quad b_{21}^{\{4\}} = 0.006136911418370,\quad b_{22}^{\{4\}} = 0.033053608345553,\quad b_{23}^{\{4\}} = 0.004291265251185,$

$u_{11} = 0.957045804986185,\quad u_{12} = 0.042954195013815,\quad u_{21} = 0.684034016720173,\quad u_{22} = 0.315965983279827,$

$u_{31} = 0.929009329287935,\quad u_{32} = 0.070990670712065,\quad v = 0.035309880170621,\quad c_1 = 0.324171924094799,$

$c_2 = 0.807694415324405,\quad w_{10} = 1.0,\quad w_{11} = 0.337590296565330,\quad w_{12} = 0.046071695897029,$

$w_{13} = 0.001607713367915,\quad w_{14} = 0.000143963069728,\quad w_{15} = 0.000063618469522,\quad w_{16} = 0.000006654666131,$

$w_{17} = -0.000000395816761,\quad w_{18} = -0.000000198789037,\quad w_{19} = 0.986560378780447,\quad w_{20} = 1.0,$

$w_{21} = 0.025202357128554,\quad w_{22} = 0.196744343505741,\quad w_{23} = 0.096360260165478,\quad w_{24} = 0.007504764895165,$

$w_{25} = -0.000722929866516,\quad w_{26} = -0.000110745594193,\quad w_{27} = 0.000010556807649,\quad w_{28} = 0.000004499558562,$

$w_{29} = 0.986558896598726.$

Author contribution Afsaneh Moradi: conceptualization, formal analysis, investigation, methodology, software, validation, visualization, writing—orginal draft, writing—review and editing. Jeremy Choi-choulis: conceptualization, formal analysis, investigation, methodology, software, validation, visualization, writing—review and editing. Jochen Schutz: conceptualization, formal analysis, investigation, methodology, software, supervision, writing—review and editing. Rafeale D'Ambrosio: conceptualization, formal analysis, investigation, methodology, software, supervision, writing—review and editing.

Funding R. D'Ambrosio is supported by GNCS-INDAM project and PRIN2017-MIUR project 2017JYCLSF "Structure preserving approximation of evolutionary problems."

Data Availability No datasets were generated or analyzed during the current study.

Declarations

Conflict of interest The authors declare no competing interests.

References

1. Abdi, A.: Construction of high-order quadratically stable second-derivative general linear methods for the numerical integration of stiff ODEs. J. Comput. Appl. Math. **303**, 218–228 (2016)
2. Abdi, A., Behzad, B.: Efficient Nordsieck second derivative general linear methods: construction and implementation. Calcolo **55**(28), 1–16 (2018)
3. Abdi, A., Braś, M., Hojjati, G.: On the construction of second derivative diagonally implicit multistage integration methods. Appl. Numer. Math. **76**, 1–18 (2014)
4. Abdi, A., Conte, D.: Implementation of second derivative general linear methods. Calcolo **57**, 20 (2020)
5. Butcher, J.C.: On the convergence of numerical solutions to ordinary differential equations. Math. Comput. **20**, 1–10 (1966)
6. Butcher, J.C.: Numerical methods for ordinary differential equations. Wiley, New York (2016)
7. Butcher, J.C., Hojjati, G.: Second derivative methods with RK stability. Numer. Algorithms **40**, 415–429 (2005)
8. Califano, G., Izzo, G., Jackiewicz, Z.: Strong stability preserving general linear methods with Runge-Kutta stability. J. Sci. Comput. **76**, 943–968 (2018)
9. Carrillo, H., Parés, C.: Compact approximate Taylor methods for systems of conservation laws. J. Sci. Comput. **80**(3), 1832–1866 (2019)
10. Cash, J.R.: Second derivative extended backward differentiation formulas for the numerical integration of stiff systems. SIAM J. Numer. Anal. **18**, 21–36 (1981)
11. Chouchoulis, J., Schütz, J., Zeifang, J.: Jacobian-free explicit multiderivative Runge-Kutta methods for hyperbolic conservation laws. J. Sci. Comput. **90**, 96 (2022)
12. Chan, R.P.K., Tsai, A.Y.J.: On explicit two-derivative Runge-Kutta methods. Numer. Algorithms **53**, 171–194 (2010)
13. Cheng, J.B., Toro, E.F., Jiang, S., Tang, W.: A sub-cell WENO reconstruction method for spatial derivatives in the ADER scheme. J. Comput. Phys. **251**, 53–80 (2013)
14. Christlieb, A.J., Gottlieb, S., Grant, Z., Seal, D.C.: Explicit strong stability preserving multistage two-derivative time-stepping schemes. J. Sci. Comput. **68**, 914–942 (2016)
15. Dahlquist, G.: A special stability problem for linear multistep methods. BIT **3**, 27–43 (1963)
16. Dumbser, M., Balsara, D.S., Toro, E.F., Munz, C.-D.: A unified framework for the construction of one-step finite volume and discontinuous Galerkin schemes on unstructured meshes. J. Comput. Phys. **227**(18), 8209–8253 (2008)
17. Dumbser, M., Enaux, C., Toro, E.F.: Finite volume schemes of very high order of accuracy for stiff hyperbolic balance laws. J. Comput. Phys. **227**(8), 3971–4001 (2008)
18. Dumbser, M., Fambri, F., Tavelli, M., Bader, M., Weinzierl, T.: Efficient implementation of ADER discontinuous Galerkin schemes for a scalable hyperbolic PDE engine. Axioms **7**(3), 63 (2018)
19. Gottlieb, S.: On high order strong stability preserving Runge-Kutta and multi step time discretizations. J. Sci. Comput. **25**, 105–128 (2005)
20. Gottlieb, S., Ketcheson, D.I., Shu, C.-W.: Strong stability preserving Runge-Kutta and multistep time discretizations. World Scientific, Hackensack (2011)
21. Gottlieb, S., Shu, C.-W., Tadmor, E.: Strong stability-preserving high-order time discretization methods. SIAM Rev. **43**, 89–112 (2001)
22. Grant, Z., Gottlieb, S., Seal, D.C.: A strong stability preserving analysis for explicit multistage two-derivative time-stepping schemes based on Taylor series conditions. Commun. Appl. Math. Comput. **1**, 21–59 (2019)

23. Hairer, E., Wanner, G.: Solving ordinary differential equations II: stiff and differential-algebraic problems. Springer, Berlin (2010)
24. Higueras, I.: Representations of Runge-Kutta methods and strong stability preserving methods. SIAM J. Numer. Anal. **43**, 924–948 (2005)
25. Hundsdorfer, W., Ruuth, S.J.: On monotonicity and boundedness properties of linear multistep methods. Math. Comput. **75**, 655–672 (2005)
26. Izzo, G., Jackiewicz, Z.: Strong stability preserving general linear methods. J. Sci. Comput. **65**, 271–298 (2015)
27. Jackiewicz, Z.: General linear methods for ordinary differential equations. Wiley, Hoboken (2009)
28. Ketcheson, D.I., Gottlieb, S., Macdonald, C.B.: Strong stability preserving two-step Runge-Kutta methods. SIAM J. Numer. Anal. **49**, 2618–2639 (2011)
29. Lax, P., Wendroff, B.: Systems of conservation laws. Commun. Pure Appl. Math. **13**(2), 217–237 (1960)
30. LeVeque, R.J.: Numerical methods for conservation laws. Birkhäuser, Basel (1990)
31. Moradi, A., Farzi, J., Abdi, A.: Strong stability preserving second derivative general linear methods. J. Sci. Comput. **81**, 392–435 (2019)
32. Moradi, A., Abdi, A., Farzi, J.: Strong stability preserving second derivative general linear methods with Runge-Kutta stability. J. Sci. Comput. **85**(1), 1–39 (2020)
33. Moradi, A., Abdi, A., Farzi, J.: Strong stability preserving second derivative diagonally implicit multistage integration methods. Appl. Numer. Math. **150**, 536–558 (2020)
34. Moradi, A., Sharifi, M., Abdi, A.: Transformed implicit-explicit second derivative diagonally implicit multistage integration methods with strong stability preserving explicit part. Appl. Numer. Math. **156**, 14–31 (2020)
35. Moradi, A., Abdi, A., Farzi, J.: Strong stability preserving diagonally implicit multistage integration methods. Appl. Numer. Math. **150**, 536–558 (2020)
36. Moradi, A., Abdi, A., Hojjati, G.: High order explicit second derivative methods with strong stability properties based on Taylor series conditions. ANZIAM J., 1–28 (2022)
37. Moradi, A., Abdi, A., Hojjati, G.: Strong stability preserving second derivative general linear methods based on Taylor series conditions for discontinuous Galerkin discretizations. J. Sci. Comput. **98**(20), 1–21 (2024)
38. Ökten Turacı, M., Öziş, T.: Derivation of three-derivative Runge-Kutta methods. Numer. Algorithms **74**(1), 247–265 (2017)
39. Qin, X., Jiang, Z., Yu, J., Huang, L., Yan, C.: Strong stability-preserving three-derivative Runge-Kutta methods. Comput. Appl. Math. **42**(171), 1–24 (2023)
40. Schütz, J., Seal, D.C., Jaust, A.: Implicit multiderivative collocation solvers for linear partial differential equations with discontinuous Galerkin spatial discretizations. J. Sci. Comput. **73**, 1145–1163 (2017)
41. Seal, D.C., Gülü, Y., Christlieb, A.: High-order multiderivative time integrators for hyperbolic conservation laws. J. Sci. Comput. **60**, 101–140 (2014)
42. Schwartzkopff, T., Dumbser, M., Munz, C.-D.: ADER: a high-order approach for linear hyperbolic systems in 2D. J. Sci. Comput. **17**, 231–240 (2002)
43. Shu, C.-W.: Total-variation diminishing time discretizations. J. Sci. Comput. **9**, 1073–1084 (1988)
44. Shu, C.-W.: Essentially non-oscillatory and weighted essentially non-oscillatory schemes for hyperbolic conservation laws. In: Quarteroni, A. (ed.) Advanced numerical approximation of nonlinear hyperbolic equations, pp. 325–432. Springer, Berlin (1998)
45. Titarev, V.A., Toro, E.F.: ADER: arbitrary high order Godunov approach. J. Sci. Comput. **17**(1), 609–618 (2002)
46. Titarev, V.A., Toro, E.F.: ADER schemes for three-dimensional non-linear hyperbolic systems. J. Comput. Phys. **204**(2), 715–736 (2005)
47. Toro, E.F., Titarev, V.A.: Derivative Riemann solvers for systems of conservation laws and ADER methods. J. Comput. Phys. **212**, 150–165 (2006)
48. Whitham, G.: Linear and nonlinear waves. A Wiley Series of Texts, Monographs and Tracts. Wiley, New York, Pure and Applied Mathematics (2011)
49. Zhang, X., Shu, C.-W.: On maximum-principle-satisfying high order schemes for scalar conservation laws. J. Comput. Phys. **229**, 3091–3120 (2010)
50. Zorìo, D., Baeza, A., Mulet, P.: An approximate Lax-Wendroff-type procedure for high order accurate schemes for hyperbolic conservation laws. J. Sci. Comput. **71**, 246–273 (2017)

Paper III: Jacobian-free Implicit MDRK Methods for stiff systems of ODEs

Contents lists available at ScienceDirect

Applied Numerical Mathematics

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

Jacobian-free implicit MDRK methods for stiff systems of ODEs

Jeremy Chouchoulis [*], Jochen Schütz

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

A R T I C L E I N F O

Keywords:
Multiderivative Runge-Kutta
Jacobian-free
ODE integrator

A B S T R A C T

In this work, an approximate family of implicit multiderivative Runge-Kutta (MDRK) time integrators for stiff initial value problems is presented. The approximation procedure is based on the recent Approximate Implicit Taylor method (Baeza et al., 2020 [7]). As a Taylor method can be written in MDRK format, the novel family constitutes a multistage generalization. Two different alternatives are investigated for the computation of the higher order derivatives: either directly as part of the stage equation, or either as a separate formula for each derivative added on top of the stage equation itself. From linearizing through Newton's method, it turns out that the conditioning of the Newton matrix behaves significantly different for both cases. We show that direct computation results in a matrix with a conditioning that is highly dependent on the stiffness, increasing exponentially in the stiffness parameter with the amount of derivatives. Adding separate formulas has a more favorable behavior, the matrix conditioning being linearly dependent on the stiffness, regardless of the amount of derivatives. Despite increasing the Newton system significantly in size, through several numerical results it is demonstrated that doing so can be considerably beneficial.

1. Introduction

We are interested in developing stable and efficient *implicit* multiderivative time integrators, see, e.g., [1–9] and the references therein, for stiff ordinary differential equations (ODEs)

$$y'(t) = \Phi(y), \tag{1}$$

where $\Phi : \mathbb{R}^M \to \mathbb{R}^M$ is the flux and $y : \mathbb{R}^+ \to \mathbb{R}^M$ the unknown solution variable. In our case, stiffness is introduced through a variable $\varepsilon \ll 1$ into the flux, which is given by

$$\Phi_i(y) = f_i(y_1, \dots, y_M) + \frac{g_i(y_1, \dots, y_M)}{\varepsilon}, \quad 1 \le i \le M, \tag{2}$$

for smooth functions f_i and g_i that do not explicitly depend on ε. Multiderivative methods not only take into account the first derivative $y'(t)$, but as well higher order time derivatives

$$y^{(k)}(t) := \frac{\mathrm{d}^k}{\mathrm{d}t^k} y(t).$$

Repeatedly making use of the ODE system (1) leads to the formulas

* Corresponding author.
E-mail addresses: jeremy.chouchoulis@uhasselt.be (J. Chouchoulis), jochen.schuetz@uhasselt.be (J. Schütz).

https://doi.org/10.1016/j.apnum.2023.10.007
Received 6 February 2023; Received in revised form 18 October 2023; Accepted 23 October 2023

Available online 26 October 2023

$$y^{(2)} = \Phi'(y)y^{(1)}, \tag{3a}$$

$$y^{(3)} = \Phi''(y) \bullet \left[y^{(1)}|y^{(1)}\right] + \Phi'(y)y^{(2)}, \tag{3b}$$

$$y^{(4)} = \Phi'''(y) \bullet \left[y^{(1)}|y^{(1)}|y^{(1)}\right] + 3\Phi''(y) \bullet \left[y^{(1)}|y^{(2)}\right] + \Phi'(y)y^{(3)}, \tag{3c}$$

and so forth. (Please note that, for the ease of presentation, we have not explicitly stated the t-dependency of $y^{(k)}$.) The bullet operator is the tensor action, i.e.,

$$\Phi''' \bullet [u|v|w] := \sum_{j,k,l=1}^{M} \frac{\partial^3 \Phi}{\partial y_j \partial y_k \partial y_l} u_j v_k w_l, \tag{4}$$

where $u, v, w \in \mathbb{R}^M$. Already at this introductory level, it can be seen that it is quite cumbersome to explicitly put all the terms used in (3) into an algorithm. Furthermore, plugging $\Phi_i(y)$ into (3) reveals that $y^{(k)} = \mathcal{O}(\varepsilon^{-k})$. As a result, the derivatives $y^{(k)}$ tend to become extremely large with each added order of the derivative, potentially leading to a huge disparity in values handled in a multiderivative solver. Therefore, one can expect the typical limitations associated to floating-point arithmetic. In particular, the algebraic system of equations that results from the nonlinear timescheme is strongly influenced. It is shown numerically in this work that the conditioning of the linearized equation system behaves as $\mathcal{O}(\varepsilon^{-k})$ if no algorithmic measures are taken.

In 2018, Baeza et al. [6] have constructed a recursive algorithm on the basis of centered finite differences that approximates the derivatives $y^{(k)}$, accordingly named the Approximate Taylor (AT) method. This approach directly stems from a recursive finite difference scheme that was designed for the circumvention of the Cauchy-Kovalevskaya procedure in the context of hyperbolic conservation laws [10]. In order to deal with stiffness and strict timestepping restrictions, more recently Baeza et al. [7] extended the AT method with an implicit variant, named the Approximate Implicit Taylor method. To simplify the computation of the Newton-Jacobian, [7] suggests including additional equations into the ODE system for the calculation of the derivatives $y^{(k)}$. We show that this as well can improve the conditioning of the Jacobian compared to the $\mathcal{O}(\varepsilon^{-k})$ behavior that is achieved by directly incorporating (3).

In this work, we generalize the approximate implicit Taylor method to more general multiderivative Runge-Kutta (MDRK) schemes. This improves the solution quality significantly. While a Taylor method has order of convergence $\mathcal{O}(\Delta t^k)$, with k denoting the maximally used derivative, MDRK schemes can achieve the same order through less derivatives by incorporating more stages. Furthermore, we thoroughly investigate multiple ways of solving the resulting algebraic system of equations. Although all methods are equivalent with *infinite* machine precision, we observe that numerically, the methods differ quite significantly.

First, in Sect. 2 traditional MDRK time integrators for ODEs are introduced, highlighting the variety of ways to compute the time derivatives $y^{(k)}$, among which a review of the AT procedure is given. Next, Sect. 3 is devoted to understanding the stability of the linear system obtained from applying Newton's method. In settings with timesteps large compared to the stiffness parameter ε, we show that the Newton-Jacobian has a condition number that grows exponentially with the amount of derivatives. As an alternative for the traditional MDRK approach, in Sect. 4, along the lines of the approximate implicit Taylor method, we introduce the MDRK-DerSol approach, where the derivatives are computed as solution variables via new relations in a larger ODE system. We verify numerically that the Newton-Jacobian of this bigger system has a more favorable conditioning asymptotically for ε going to 0. Subsequently, in Sect. 5, we compare the developed method to more classical schemes for stiff ODEs. Finally, our conclusions are summarized and future endeavors are explored in Sect. 6.

2. Implicit multiderivative Runge-Kutta solvers

In order to apply a time-marching scheme to Eq. (1), we discretize the temporal domain with a constant[1] timestep Δt and iterate N steps such that $\Delta t = T_{\text{end}}/N$. Consequently, we define the time levels by

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

The central class of time integrators in this work are *implicit* MDRK methods. By adding extra temporal derivatives of $\Phi(y)$, these form a natural generalization of classical implicit Runge-Kutta methods. To present our ideas, let us formally define the MDRK scheme as follows:

Definition 1 *(Kastlunger, Wanner [1, Section 1]).* A q-th order implicit $\mathtt{r}$-derivative Runge-Kutta scheme using $\mathtt{s}$ stages ($\mathtt{r}$DRKq-$\mathtt{s}$) is any method which can, for given coefficients $a_{l\nu}^{(k)}$, $b_l^{(k)}$ and c_l, be formalized as

$$y^{n,l} := y^n + \sum_{k=1}^{\mathtt{r}} \Delta t^k \sum_{\nu=1}^{\mathtt{s}} a_{l\nu}^{(k)} \frac{\mathrm{d}^{k-1}}{\mathrm{d}t^{k-1}} \Phi\left(y^{n,\nu}\right), \quad l = 1, \ldots, \mathtt{s}, \tag{5a}$$

where $y^{n,l}$ is a stage approximation of y at time $t^{n,l} := t^n + c_l \Delta t$. The update is given by

[1] A fixed Δt is used for solving any ODE system described within this work. Nevertheless, all presented methods can readily be applied with a variable timestep Δt^n if needed.

$$y^{n+1} := y^n + \sum_{k=1}^{\mathtt{r}} \Delta t^k \sum_{l=1}^{\mathtt{s}} b_l^{(k)} \frac{\mathrm{d}^{k-1}}{\mathrm{d}t^{k-1}} \Phi(y^{n,l}) . \tag{5b}$$

Typically, the values of $a_{l\nu}^{(k)}$, $b_l^{(k)}$ and c_l are summarized in an extended Butcher tableau, see Appendix A for some examples.

As can be seen from Eq. (3), at least the k-th order Jacobian tensor

$$\Phi^k(y) = \frac{\partial^k \Phi}{\partial y^k}(y) \tag{6}$$

is needed for the derivation of $\frac{\mathrm{d}^k}{\mathrm{d}t^k}\Phi$. For systems of ODEs, Φ^k is an $M \times \ldots \times M$ (k-times) tensor. The generalization of (3), named Faá Di Bruno's formula (see [7, Prop. 1]), can therefore be very expensive. A more sensible way to obtain the time derivatives of Φ is from the recursive relation

$$\frac{\mathrm{d}^k}{\mathrm{d}t^k}\Phi(y) = \left[\frac{\mathrm{d}^{k-1}\Phi(y)}{\mathrm{d}t^{k-1}}\right]' y^{(1)} . \tag{7}$$

The prime symbol denotes the Jacobian derivative with respect to y. Here, the quantities $\left[\frac{\mathrm{d}^{k-1}\Phi}{\mathrm{d}t^{k-1}}\right]'$ are $M \times M$ matrices, regardless of k.

Remark 1. Although Faá Di Bruno's formula is mathematically equivalent to the recursive relation (7), numerical results do in actuality differ. Due to the many tensor actions with Φ^k in Faá Di Bruno's formula, numerical computations are much more prone to round-off errors. To illustrate this, we will apply both Faá Di Bruno's formula, as in (3), and the recursive relation (7). We refer to the tensors Φ^k with the wording "Exact Jacobians" (EJ) from hereon.

2.1. *Approximating the time derivatives*

Despite the availability of the recursive relation (7), the Jacobian derivatives w.r.t. y within this relation can nevertheless be quite intricate to deal with. And as such, avoiding Jacobian derivation by hand often leads to the use of symbolic computing software to allow the user to focus directly on the numerical procedure. There are two major downsides here, first, it being that symbolic software is computationally expensive, and secondly, not always feasible to apply. For large numerical packages for example, generally it is not desirable to significantly alter vital portions of code. To overcome symbolic procedures completely, a high-order centered differences approximation strategy has recently been developed by Baeza et al. [6,7] to obtain values

$$\widetilde{y}^{(k)} = y^{(k)} + \mathcal{O}(\Delta t^{\mathtt{r}-k+1}) = \frac{\mathrm{d}^{k-1}}{\mathrm{d}t^{k-1}}\Phi(y) + \mathcal{O}(\Delta t^{\mathtt{r}-k+1}), \tag{8}$$

for $k = 2, \ldots, \mathtt{r}$. An overview of the method is given here; first, necessary notation is introduced.

Definition 2. For any number $p \in \mathbb{N}$, define the locally centered stencil function having $2p+1$ nodes by means of angled brackets

$$\langle \cdot \rangle : \mathbb{Z} \to \mathbb{Z}^{2p+1} : z \mapsto \begin{pmatrix} z-p, & \ldots, & z+p \end{pmatrix}^T . \tag{9}$$

In this manner it is possible to write the vectors

$$\mathbf{y}^{\langle n \rangle} := \begin{pmatrix} y^{n-p} \\ \vdots \\ y^{n+p} \end{pmatrix} \quad \text{and} \quad y(\mathbf{t}^{\langle n \rangle}) := \begin{pmatrix} y(t^{n-p}) \\ \vdots \\ y(t^{n+p}) \end{pmatrix} . \tag{10}$$

Such representation allows us to concisely write down approximations $\widetilde{y}^{(k)}$ to $y^{(k)}$. Let $k = 1, \ldots, \mathtt{r}$ be the derivative order of interest which we would like to approximate.

Lemma 1 *(Carrillo, Parés [11, Proposition 4], Zorío et al. [10, Proposition 2]). For $k \geq 1$ and $p \geq \lfloor \frac{k+1}{2} \rfloor$ ($p \in \mathbb{N}$), there exist $2p+1$ quantities $\delta_{p,j}^k \in \mathbb{R}$ for $j = -p, \ldots, p$, such that the linear operator*

$$P^{(k)} : \mathbb{R}^{2p+1} \to \mathbb{R}, \qquad \mathbf{v} \mapsto \frac{1}{\Delta t^k} \sum_{j=-p}^{p} \delta_{p,j}^k v_j \tag{11}$$

approximates the k-th derivative up to order $\omega := 2p - 2\lfloor \frac{k-1}{2} \rfloor$, i.e.

$$P^{(k)} y(\mathbf{t}^{\langle n \rangle}) = y^{(k)}(t^n) + \mathcal{O}(\Delta t^\omega) . \tag{12}$$

The linear operator $P^{(k)}$, however, is difficult to apply in practice, since it introduces additional unknown values $y(t^{n+1}), \ldots, y(t^{n+p})$ into the stencil. In order to bypass the issue of creating more unknowns, in [6,7,10] a recursive strategy that includes Taylor approximations into the centered difference operator is incorporated. This gives the following computations of the values $\widetilde{y}^{(k)}$:

MDRK	Approximate	Exact Jacobians	Recursive
Direct	A-Direct DIMDRK: M^2 FSMDRK: $(\mathtt{s}M)^2$ Eqs. (13)-(14)	EJ-Direct DIMDRK: M^2 FSMDRK: $(\mathtt{s}M)^2$ Eqs. (3)	rec-Direct DIMDRK: M^2 FSMDRK: $(\mathtt{s}M)^2$ Eq. (7)
Derivatives as Solutions	A-DerSol DIMDRK: $((\mathtt{r}+1)M)^2$ FSMDRK:$((\mathtt{r}+1)\mathtt{s}M)^2$ Eqs. (41)-(43)	EJ-DerSol DIMDRK: $((\mathtt{r}+1)M)^2$ FSMDRK:$((\mathtt{r}+1)\mathtt{s}M)^2$ Eqs. (28) and (36)	rec-DerSol DIMDRK: $((\mathtt{r}+1)M)^2$ FSMDRK:$((\mathtt{r}+1)\mathtt{s}M)^2$ Eqs. (29) and (36)

Fig. 1. An overview of the different MDRK approaches applied in this work. In here M represents the size of the ODE system being solved, $\mathtt{r}$ and $\mathtt{s}$ represent the amount of derivatives and the amount of stages, respectively, of the MDRK scheme (Definition 1). For both DIMDRK and FSMDRK schemes (Subsect. 2.2) the size of the linear system resulting from applying Newton's method is displayed. In comparison to the other Direct approaches, EJ-Direct necessitates the most computations through tensor calculations, making it less fit for efficient time integration. Moreover, we found that using rec-Dersol leads to algebraic systems with extremely large condition numbers in relation to the other DerSol procedures; rec-Dersol hence turned out to be less suited for stiff equations than the others.

$$\begin{aligned} \widetilde{y}^{(1)} &:= \Phi(y^n), \\ \widetilde{y}^{(k)} &:= P^{(k-1)}\boldsymbol{\Phi}_T^{k-1,\langle n\rangle}, \quad 2 \le k \le \mathtt{r}, \end{aligned} \tag{13}$$

in which

$$\Phi_T^{k-1,n+j} := \Phi\left(y^n + \sum_{m=1}^{k-1} \frac{(j\Delta t)^m}{m!}\widetilde{y}^{(m)} \right) \tag{14}$$

is an approximation to $\Phi\big(y(t^{n+j})\big)$. By adopting the recursive Taylor approach (13)-(14) into Definition 1, we acquire a novel family of time-marching schemes:

Definition 3 *(AMDRK method).* The $\mathtt{r}$DRKq-$\mathtt{s}$ scheme (Definition 1) in which the time derivatives $\frac{\mathrm{d}^{k-1}}{\mathrm{d}t^{k-1}}\Phi(y)$ are approximated by using the formulas (13)-(14) is called the *Approximate* MDRK method, denoted by the short-hand notation $A\mathtt{r}$DRKq-$\mathtt{s}$.

Without proof – it is very similar to related cases, see for example [6] in the context of explicit Taylor schemes for ODEs and [12] for explicit MDRK schemes applied to hyperbolic PDEs – we state the order of convergence:

Theorem 2. *The consistency order of an $A\mathtt{r}DRKq$-$\mathtt{s}$ method is $\min(2p+1, q)$, the variable q being the consistency order of the underlying MDRK method, and p denoting the use of the $2p+1$ points $\{t^{n-p}, \dots, t^{n+p}\}$ in Eq. (13).*

Remark 2. Note that the variable p is not defined in the terminology "$A\mathtt{r}$DRKq-$\mathtt{s}$". Since the consistency order is $\min(2p+1, q)$, the optimal choice w.r.t. computational efficiency is to set $p = \lfloor q/2 \rfloor$. Throughout this paper p is chosen along this line of reasoning for all numerical results.

Remark 3. If Φ is a linear function in y, then the procedure in (13)-(14) is exact, see [13, Proposition 1]. This also means that (linear) stability properties such as $A-$ and L-stability are not influenced by the approximate Taylor approach.

2.2. *Specific MDRK schemes*

As is the case for standard Runge-Kutta methods, the (A)$\mathtt{r}$DRKq-$\mathtt{s}$ method has a lot of flexibility in choosing the coefficients $a_{l\nu}^{(k)}$ and $b_l^{(k)}$. We put spotlight on two varying implementations of the (A)$\mathtt{r}$DRKq-$\mathtt{s}$ method:

- *Full Storage MDRK (FSMDRK)*:
  Under the assumption that the Butcher tableau consists of dense matrices, all $\mathtt{s}$ stages are coupled and must be solved simultaneously. This approach can be applied for any existing MDRK scheme, but might not be efficient as it often leads to large systems of equations.

- *Diagonally Implicit MDRK (DIMDRK)*:
  If each stage l only depends on previous stages $\nu = 1, \ldots, l-1$, and is only implicit in itself, it can be more efficient to solve for each stage one at a time.

In Appendix A the extended Butcher tableaux used in this paper are displayed, three families are considered:

- *Taylor schemes (Tables A.3-A.4)*:
  The explicit and implicit Taylor method can be reformulated as a single-stage MDRK scheme. This respectively leads to the Approximate Explicit Taylor methods in [6] and the Approximate Implicit Taylor methods in [7]. Having only one stage, the FSMDRK and DIMDRK approaches are equivalent.
- *Hermite-Birkhoff (HB) schemes (Tables A.5-A.10)*:
  The coefficients are obtained from the Hermite-Birkhoff quadrature rule [8,14] which integrates a Hermite polynomial that also takes derivative data into account, with possibly a varying amount of derivative data per point. By taking equispaced abscissa c_l, the resulting tableau is fully implicit whilst having a fully explicit first stage. Hence, for $\mathtt{s} > 2$, stages 2 to $\mathtt{s}$ should be solved with an FSMDRK method.
- *Strong-Stability Preserving (SSP) schemes (Tables A.11-A.12)*:
  In [9], Gottlieb et al. have constructed implicit multiderivative SSP schemes. The tableaux are diagonally implicit, and therefore each stage can be solved for one after another. Both the DIMDRK and FSMDRK approach are thus valid, with the DIMDRK approach likely being more efficient.

In the first row of Fig. 1 an overview of the thus far presented MDRK methods is given, with differences focused around the computations of the derivatives $y^{(k)}$. As all of these MDRK methods exclusively solve the Eqs. (5a)-(5b), we refer to them as "Direct" in this work.

2.3. Nonlinear solver

The implicit (A)MDRK scheme (Definitions 1 and 3) requests a nonlinear solver, irrespective of whether the derivatives are either calculated exactly through (3), recursively obtained with (7) or approximated by means of (13)-(14). In case that a single stage $Y = y^{n,l}$ (with $l = 1, \ldots, \mathtt{s}$) is considered, as for DIMDRK schemes, or all the unknown stages are combined into a single vector $Y = \left(y^{n,1}, \ldots, y^{n,\mathtt{s}}\right)$ as in the FSMDRK approach, it is possible to write the stage equation(s) (5a) as

$$F(Y) = 0, \tag{15}$$

and then choose any nonlinear solver of preference. Computationally, solving Eq. (15) is the most expensive portion of the numerical method. Hence, it is vital for the efficiency of the overall method to well understand the behavior of the selected solver. In this paper we use Newton's method, and thus require the Jacobian matrix $F'(Y)$. Given an initial value $Y^{[0]}$, the linearized system

$$F'(Y^{[i]})\Delta Y^{[i]} = -F(Y^{[i]}), \quad Y^{[i+1]} = Y^{[i]} + \Delta Y^{[i]} \tag{16}$$

is solved for $i = 0, \ldots, N_{\text{iter}} - 1$ or until some convergence criteria are satisfied. In this work, criteria are invoked on the residuals,

$$\|F(Y^{[i]})\|_2 < 10^{-n_{\text{tol}}} \quad \text{or} \quad \frac{\|F(Y^{[i]})\|_2}{\|F(Y^{[0]})\|_2} < 10^{-n_{\text{tol0}}}, \tag{17}$$

where $n_{\text{tol}}, n_{\text{tol0}} \in \mathbb{N}$. Under the assumptions that $Y^{[0]}$ is in a neighborhood close enough to the exact solution Y, and the Jacobian matrix is nonsingular, Newton's method converges quadratically [14, Theorem 7.1].

Remark 4. In what follows, we avoid the superscript index i whenever possible, and instead write $F'(Y)$ or even F'.

Remark 5. The term 'Jacobian-free' in the title of this work does not refer to Newton's method, so $F'(Y^{[i]})$ is computed exactly, it rather refers to the approximation of $y^{(k)}$ in (3) by $\tilde{y}^{(k)}$ in (13).

3. Newton stability of direct (A)MDRK methods

In order to investigate the conditioning of the Newton-Jacobian $F'(Y)$ in the linearized Newton system (16), we consider the Pareschi-Russo (PR) problem [15], given by

$$y_1'(t) = -y_2, \qquad y_2'(t) = y_1 + \frac{\sin(y_1) - y_2}{\varepsilon}, \qquad y(0) = \left(\frac{\pi}{2}, 1\right). \tag{18}$$

Let us first verify that the consistency order given in Theorem 2 is achieved and compare it with the exact MDRK method as in Definition 1 (using relation (7)). In Fig. 2, convergence plots are shown for five different MDRK schemes (three Hermite-Birkhoff and two SSP, see Appendix A). Final time is set to $T_{\text{end}} = 5$; the coarsest computation uses $N = 4$ timesteps. To separate convergence order from stiffness, ε is set to 1. We can clearly see that the AMDRK method achieves the appropriate convergence orders for all the

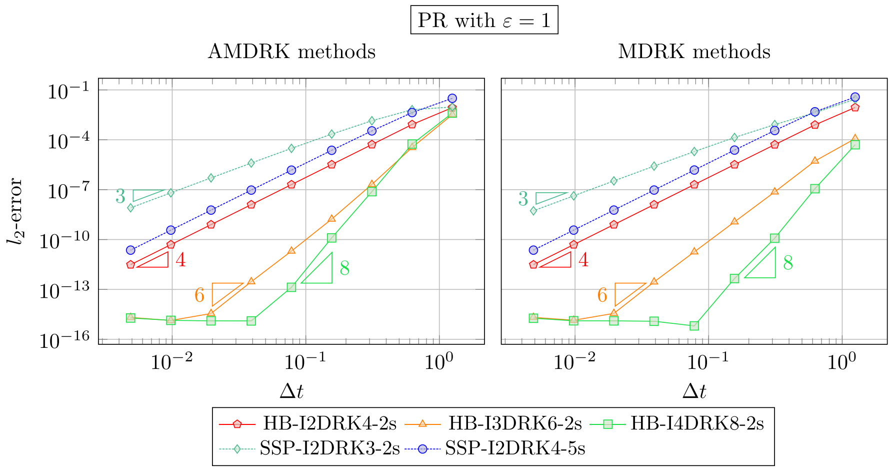


Fig. 2. Pareschi-Russo problem, $\varepsilon = 1$: Convergence of the AMDRK scheme (Definition 3) and the MDRK scheme (Definition 1). The final time is set to $T_{\text{end}} = 5$, timesteps start from $N = 4$. Three Hermite-Birkhoff schemes and two SSP schemes are considered, see Appendix A.

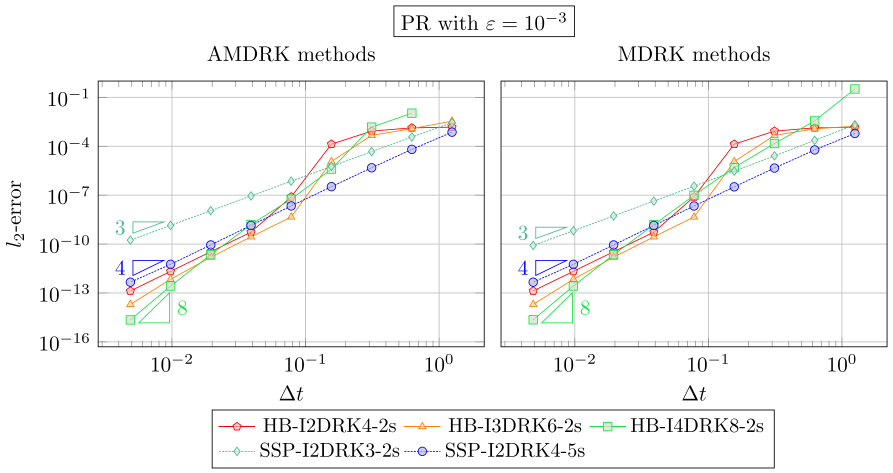


Fig. 3. Pareschi-Russo problem, $\varepsilon = 10^{-3}$: Convergence of the AMDRK scheme (Definition 3) and the MDRK scheme (Definition 1). The final time is set to $T_{\text{end}} = 5$, timesteps start from $N = 4$. Three Hermite-Birkhoff schemes and two SSP schemes are considered, see Appendix A. For $N = 4$ the HB-I4DRK8-2s scheme diverges, hence no node is shown.

considered schemes. Also, when compared to their exact MDRK counterpart, differences are barely visible. Only for high values of Δt and large orders of consistency, differences are visible.

When stiffness is increased by decreasing ε to $\varepsilon = 10^{-3}$, see Fig. 3, the same methods as well show convergence for $\Delta t \to 0$. However, due to order reduction phenomena, it is more difficult to observe the appropriate order here. Above all, for values $\varepsilon \ll \Delta t$, the scheme HB-I4DRK8-2s (Table A.10) has not properly converged in the Newton iterations; for $N = 4$ the AMDRK method diverges immediately, hence there being no node in the left plot of Fig. 3, whereas the exact MDRK method shows a large error.

3.1. Numerical observations of the Newton conditioning

So, even though all three approaches (3), (7) and (13)-(14) for calculating the derivatives $y^{(k)}$ yield valid high-order algorithms, numerically we observe stability issues for stiff problems $\varepsilon \ll \Delta t$. More specifically, when $\varepsilon \ll \Delta t$, the Newton-Jacobian $F'(Y)$ is badly conditioned. In Table 1 we display the arithmetic mean of the condition numbers in the 1-norm w.r.t. the Newton iterations,

$$\mu\big(\text{cond}(F')\big) := \frac{\sum_{i=1}^{N_{\text{iter}}} \text{cond}(F'(Y^{[i]}))}{N_{\text{iter}}},$$

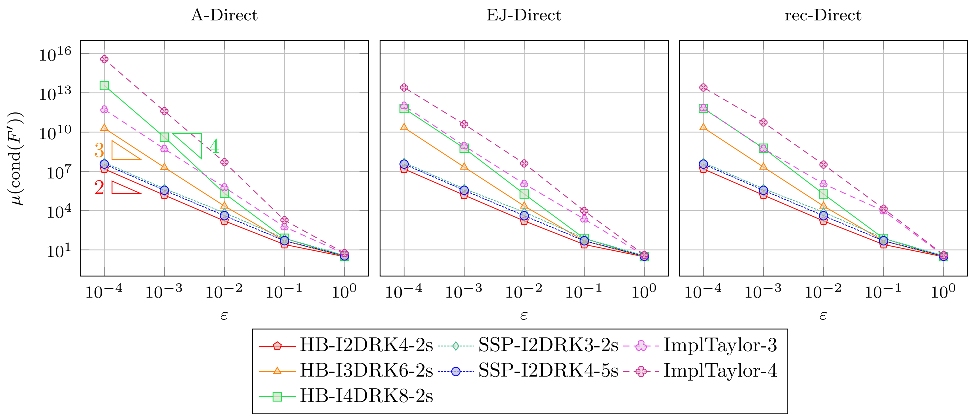


Fig. 4. DIMDRK schemes applied as a Direct method (Fig. 1) to the PR problem (18). The average condition number in the 1-norm of the Newton Jacobian obtained from the last RK-stage is shown for different values of ε. The behavior $\mathrm{cond}(F') = \mathcal{O}(\varepsilon^{-\mathtt{r}})$ is observed, $\mathtt{r}$ being the amount of derivatives. A single timestep ($N = 1$) of size $T_{\mathrm{end}} = 1.25$ ($\Delta t = 1.25$) has been considered with tolerances, Eq. (17), set to 10^{-12} under a maximum of 1000 iterations.

which we have obtained from solving Eq. (18) with the approximate implicit Taylor method of order $\mathtt{r} = 3$ for different values of ε. To account for large timesteps, only a single step $N = 1$ of size $\Delta t = 1$ was applied. Newton tolerances, Eq. (17), were set to 10^{-12} under a maximum of 10000 iterations.

In order to put the obtained condition numbers into perspective, the empirical orders w.r.t. ε

$$\mathrm{EO}_\varepsilon := \frac{\log\left(\frac{\mu(\mathrm{cond}(F'_\varepsilon))}{\mu\left(\mathrm{cond}(F'_{10\varepsilon})\right)}\right)}{\log(10)} \tag{19}$$

are additionally computed (where F'_ε denotes F' for a particular value of ε). In this case, the experimental order seems to equal the order ($\mathtt{r} = 3$ here) of the implicit Taylor method. And in fact, the same behavior is observed for any amount of derivatives $\mathtt{r}$ used. That means, we numerically observe the asymptotic behavior

$$\mathrm{cond}(F') = \mathcal{O}(\varepsilon^{-\mathtt{r}}) \tag{20}$$

to hold true for any order of the implicit Taylor scheme. As a result of bad conditioning, in Table 1 we can therefore observe that the (A)MDRK methods do not converge for Pareschi-Russo's equation (18) when $\varepsilon = 10^{-5}$. In general, when considering any DIMDRK scheme, the same derivative-dependent behavior holds true for any implicit stage. In Fig. 4 we plot the condition number for the final stage of different DIMDRK schemes.

The FSMDRK implementation shows a behavior similar to the DIMDRK implementation, see Fig. 5. An intuitive reasoning can be found in the block-matrix structure of the Newton-Jacobian. Due to the stages all being solved for at once, the Jacobian reads as

$$F'(Y) = \frac{\partial}{\partial Y}\begin{bmatrix} F_1 \\ \vdots \\ F_{\mathtt{s}} \end{bmatrix} = \left[\begin{array}{c|c|c} \partial_{Y_1} F_1 & \dots & \partial_{Y_{\mathtt{s}}} F_1 \\ \hline \vdots & \ddots & \vdots \\ \hline \partial_{Y_1} F_{\mathtt{s}} & \dots & \partial_{Y_{\mathtt{s}}} F_{\mathtt{s}} \end{array}\right], \tag{21}$$

with $\partial_{Y_\nu} F_l$ the partial derivative of the l-th stage equation w.r.t. the ν-th stage variable Y_ν. This block-structure assures a dependency of $\mathrm{cond}(F')$ on the conditioning $\mathrm{cond}(\partial_{Y_\nu} F_l)$ of the separate blocks. Hence, if $\mathrm{cond}(\partial_{Y_\nu} F_l) = \mathcal{O}(\varepsilon^{-\mathtt{r}})$, as is often the case from what is observed in the DIMDRK implementation for an $\mathtt{r}$-derivative scheme, the complete Jacobian likely also behaves at least as $\mathcal{O}(\varepsilon^{-\mathtt{r}})$.

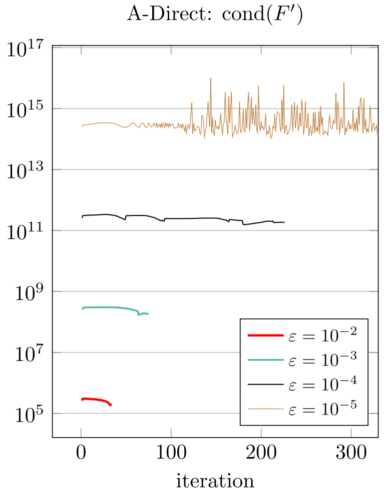
Remark 6. If the first stage is explicit, s.t. $y^{n,1} = y^n$, we make the assumption that the FSMDRK approach instead solves for $Y = \left(y^{n,2}, \dots, y^{n,\mathtt{s}}\right)$. The Hermite-Birkhoff schemes (Tables A.5-A.10) are good examples of RK-schemes with an explicit first stage.

Moreover, when we consider other problems with a similar dependency on a small non-dimensional value ε as for the PR problem (18), then as well $\mathcal{O}(\varepsilon^{-\mathtt{r}})$ behavior is observed. Similar condition number plots alike the ones in Figs. 4 and 5 have been obtained for van der Pol and Kaps problems described in [16].

Remark 7. Albeit mathematically equivalent, in Table 1 we can numerically see different results between using exact Jacobians (in the sense that we apply Faá die Bruno's formula) and using recursive formulas for calculating the derivatives $y^{(k)}$.

Table 1
Newton statistics of the implicit Taylor method of order 3 applied for a single timestep ($N = 1$) of size $T_{\text{end}} = 1$ ($\Delta t = 1$) to the PR problem (18). Tolerances, Eq. (17), were set to 10^{-12} under a maximum of 10000 iterations. Left: The amount of iterations N_{iter} and the average condition number in the 1-norm of the Newton-Jacobian $\mu(\text{cond}(F'))$. EO$_\varepsilon$ is the experimental order of the average w.r.t. ε according to Eq. (19). For $\varepsilon = 10^{-5}$, none of the methods converged, with A-Direct diverging at 702 iterations. Right: The first 330 iterations of A-Direct. We can observe that for $\varepsilon = 10^{-5}$ the scheme becomes unstable and diverges eventually at iteration 702.

Method	ε	N_{iter}	$\mu(\text{cond}(F'))$	EO$_\varepsilon$
A-Direct	1	5	$4.45 \cdot 10^{0}$	
	10^{-1}	6	$2.89 \cdot 10^{2}$	1.81
	10^{-2}	34	$2.69 \cdot 10^{5}$	2.97
	10^{-3}	75	$2.71 \cdot 10^{8}$	3.00
	10^{-4}	226	$2.51 \cdot 10^{11}$	2.97
	10^{-5}	702	$8.20 \cdot 10^{14}$	3.51
EJ-Direct	1	5	$3.44 \cdot 10^{0}$	
	10^{-1}	7	$2.78 \cdot 10^{2}$	1.91
	10^{-2}	124	$3.00 \cdot 10^{5}$	3.03
	10^{-3}	45	$5.57 \cdot 10^{8}$	3.27
	10^{-4}	10000	$6.58 \cdot 10^{10}$	2.07
	10^{-5}	10000	$2.70 \cdot 10^{13}$	2.61
rec-Direct	1	5	$3.44 \cdot 10^{0}$	
	10^{-1}	7	$2.78 \cdot 10^{2}$	1.91
	10^{-2}	124	$3.00 \cdot 10^{5}$	3.03
	10^{-3}	45	$5.57 \cdot 10^{8}$	3.27
	10^{-4}	10000	$6.32 \cdot 10^{10}$	2.05
	10^{-5}	10000	$3.31 \cdot 10^{13}$	2.72



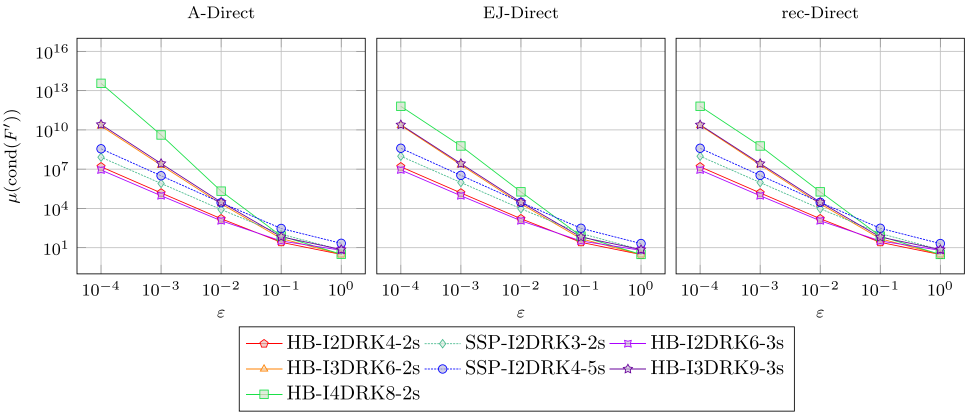


Fig. 5. FSMDRK schemes applied as a Direct method (Fig. 1) to the PR problem (18). The average condition number in the 1-norm of the Newton Jacobian is shown for different values of ε. The behavior $\text{cond}(F') = \mathcal{O}(\varepsilon^{-\mathfrak{r}})$ is observed, $\mathfrak{r}$ being the amount of derivatives. A single timestep ($N = 1$) of size $T_{\text{end}} = 1.25$ ($\Delta t = 1.25$) has been considered with tolerances, Eq. (17), set to 10^{-12} under a maximum of 1000 iterations.

3.2. Conditioning of a two-variable ODE system

Eq. (18), but also van der Pol and Kaps equation, can be put into the form

$$y_1'(t) = f_1(y_1, y_2) \tag{22}$$

$$y_2'(t) = f_2(y_1, y_2) + \frac{g(y_1, y_2)}{\varepsilon}, \tag{23}$$

in which f_1, f_2 and g are smooth functions. In order to get a basic understanding of how the condition number of the Newton-Jacobian behaves in terms of ε, we consider the simplified system

$$y_1'(t) = y_2, \quad y_2'(t) = \alpha y_1 + \frac{g(y_1, y_2)}{\varepsilon}, \quad 0 \le t \le T, \tag{24}$$

where $\alpha \in \mathbb{R}$ and $g : \mathbb{R}^2 \to \mathbb{R}$ is smooth. We are interested in the analytical form of the Newton-Jacobian obtained from the (A)MDRK method in the case that $\varepsilon \ll \Delta t$.

Example 1. Applying implicit Taylor order $\mathtt{r} = 2$ to the system of ODEs (24) yields a system $F = (y_1^n, y_2^n)^T$ with

$$F = \begin{bmatrix} y_1^{n+1} - \Delta t y_2^{n+1} + \frac{\Delta t^2}{2}\left(\alpha y_1^{n+1} + \frac{g^{n+1}}{\varepsilon}\right) \\ y_2^{n+1} - \Delta t\left(\alpha y_1^{n+1} + \frac{g^{n+1}}{\varepsilon}\right) + \frac{\Delta t^2}{2}\left(\alpha y_2^{n+1} + \frac{\partial_{y_1} g^{n+1}}{\varepsilon} y_2^{n+1} + \frac{\partial_{y_2} g^{n+1}}{\varepsilon}(\alpha y_1^{n+1} + \frac{g^{n+1}}{\varepsilon})\right) \end{bmatrix}. \tag{25}$$

Note that g^{n+1} has been defined as $g(y_1^{n+1}, y_2^{n+1})$.

Proposition 3. *Assume that g and all its partial derivatives are $\mathcal{O}(1)$, and assume that $\varepsilon \ll \Delta t$. Then, the Newton-Jacobian F' obtained from solving the system of ODEs (24) with the implicit Taylor method of order $\mathtt{r} = 2$ behaves in the 1-norm as*

$$\|F'\|_1 = \mathcal{O}\left(\frac{\Delta t^2}{\varepsilon^2}\right), \quad \textit{and} \quad \operatorname{cond}(F') = \mathcal{O}\left(\varepsilon^{-1}\right).$$

Remark 8 (part 1). The behavior shown in Proposition 3 is not what we observe from the numerical experiments in Figs. 4 and 5, where we obtained $\operatorname{cond}(F') = \mathcal{O}(\varepsilon^{-2})$ for two-derivative schemes. There is no contradiction here though. We reason in part 2 of this remark that, often, an order of ε is gained through the determinant $\det(F')$.

Proof of Proposition 3. For simplicity, the notation $(u, v) = (y_1, y_2)$ will be used in what follows. From the construction of F as given in Eq. (25), it is apparent that the Newton-Jacobian satisfies

$$\|F'\|_1 = \mathcal{O}\left(\frac{\Delta t^2}{\varepsilon^2}\right),$$

under the assumption that $\varepsilon \ll \Delta t$. For the behavior of the inverse matrix F'^{-1} we make use of the identity $A^{-1} = \frac{1}{\det(A)} \operatorname{adj}(A)$. As F' is a 2×2 matrix, its adjugate is obtained from simply shuffling terms and possibly adding a minus sign. Consequently, the behavior of its norm remains unaffected w.r.t. ε and Δt. The determinant can be explicitly computed as

$$\det(F') = 1 + \frac{1}{4}\frac{\Delta t^4}{\varepsilon^3} \underbrace{\left(\partial_u g \partial_{vv} g - \partial_v g \partial_{uv} g\right)}_{Dg} g + \mathcal{O}(\varepsilon^{-2}). \tag{26}$$

So in total:

$$\operatorname{cond}(F') = \|F'\|_1 \cdot \|F'^{-1}\|_1 = \frac{\|F'\|_1 \|\operatorname{adj}(F')\|_1}{|\det(F')|} = \mathcal{O}\left(\frac{\Delta t^2}{\varepsilon^2}\right)\mathcal{O}\left(\frac{\varepsilon^3}{\Delta t^4}\right)\mathcal{O}\left(\frac{\Delta t^2}{\varepsilon^2}\right) = \mathcal{O}(\varepsilon^{-1}),$$

under the assumption that $\varepsilon \ll \Delta t$. $\square$

Remark 8 (part 2). In equation (26) we observe that $\det(F') = \mathcal{O}\left(\frac{\Delta t^4}{\varepsilon^3}\right)$ under the assumption that $\varepsilon \ll \Delta t$. In many cases we nonetheless observe $\det(F') = \mathcal{O}(\varepsilon^{-2})$:

1. The values are mainly decided by $g(y_1, y_2)$ and a function of partial derivatives which we have denoted $Dg(y_1, y_2)$. In case of the PR-problem (18), $\alpha = 1$ and $g(y_1, y_2) = \sin(y_1) - y_2$. Therefore, any mixed partial derivatives of g, or second partial derivative of g w.r.t y_2 equals 0. So for the PR-problem $Dg = 0$.
2. In general, it does not need to hold true that $Dg = 0$. The van der Pol problem (as in [16]) for instance has $g(y_1, y_2) = (1 - y_1^2)y_2 - y_1$, and therefore yields $Dg = 2y_1(1 - y_1^2)$. Here, a clarification can be given by the (very) harsh restriction set in Proposition 3 that g and all its partial derivatives are $\mathcal{O}(1)$, which typically is not true. For well-prepared initial conditions and an asymptotically consistent algorithm, $g = \mathcal{O}(\varepsilon)$ [17].

A similar type of effect takes place for a higher amount of derivatives $\mathtt{r}$; the resulting conditioning is $\mathcal{O}(\varepsilon^{-\mathtt{r}})$.

4. Derivatives as members of the solution

One of the main issues for the $\mathcal{O}(\varepsilon^{-\mathtt{r}})$ conditioning of the direct (A)MDRK method is the fact that with each higher derivative $y^{(k)}$, the order of ε increases simultaneously. Such behavior is to be expected due to a built-in dependency on the lower order derivatives, i.e.

$$\begin{aligned} y^{(1)} &= \Phi(y), \\ y^{(k)} &= \Psi_k(y, y^{(1)}, \dots, y^{(k-1)}), \quad 2 \le k \le \mathtt{r}. \end{aligned} \tag{27}$$

The operator Ψ_k is then either the relation that uses the Exact Jacobians (EJ) as in (3), so that

$$\Psi_2 = \Phi'(y) y^{(1)}, \tag{28a}$$

$$\Psi_3 = \Phi''(y) \bullet \left[y^{(1)} | y^{(1)}\right] + \Phi'(y) y^{(2)}, \tag{28b}$$

$$\Psi_4 = \Phi'''(y) \bullet \left[y^{(1)} | y^{(1)} | y^{(1)}\right] + 3\Phi''(y) \bullet \left[y^{(1)} | y^{(2)}\right] + \Phi'(y) y^{(3)}, \tag{28c}$$

and so forth, or either is given recursively from (7), so that

$$\Psi_{k+1} = \left[\frac{\mathrm{d}^{k-1}\Phi(y)}{\mathrm{d}t^{k-1}}\right]' y^{(1)}, \tag{29}$$

for $k = 1, \ldots, \mathtt{r} - 1$.

Example 2 (part 1). Consider the implicit Taylor scheme of order $\mathtt{r} = 3$, then there is only a single stage $Y = y$ to solve for. In terms of the relations (27), the Newton system $F(y) = 0$ simply writes as

$$y - \Delta t \Phi(y) + \frac{\Delta t^2}{2} \Psi_2 - \frac{\Delta t^3}{6} \Psi_3 - y^n = 0. \tag{30}$$

From the above example it is clear that computing $F'(Y)$ necessitates deriving the formulas Ψ_k with respect to y,

$$\frac{\partial y^{(k)}}{\partial y} = \partial_y \Psi_k + \sum_{m=1}^{k-1} \partial_{y^{(m)}} \Psi_k \cdot \frac{\partial y^{(m)}}{\partial y}. \tag{31}$$

It is exactly because of this recursive dependency on lower order derivatives that the order of ε increases in $\mathrm{cond}(F')$. A similar recursion holds true when calculating the approximate values $\widetilde{y}^{(k)}$ with the recursive formulas (13)-(14).

In order to better understand the ε-behavior, we investigate a linear problem in the sequel. To reduce the complexity of involved formulas, we only consider scalar problems ($m = 1$) in this section.

4.1. ε-scaled Dahlquist test equation

We consider an ε-scaled Dahlquist test problem

$$y' = \frac{\lambda}{\varepsilon} y, \qquad y(0) = 1, \tag{32}$$

with the exact solution $y(t) = \mathrm{e}^{(\lambda/\varepsilon)t}$. As the equation is linear, the AMDRK method (A-Direct) coincides with the MDRK method that uses EJ (EJ-Direct), see for example [13, Proposition 1].[2] The rationale behind the observed behavior follows immediately from the next lemma.

Lemma 4. *The derivatives $y^{(k)}$ and their Jacobians $\partial_y y^{(k)}$ of the ε-scaled Dahlquist test are $\mathcal{O}(\varepsilon^{-k})$, i.e.*

$$y^{(k)} = \frac{\mathrm{d}^{k-1}}{\mathrm{d}t^{k-1}} \Phi = \left(\frac{\lambda}{\varepsilon}\right)^k y = \mathcal{O}(\varepsilon^{-k}), \qquad \frac{\partial y^{(k)}}{\partial y} = \left[\frac{\mathrm{d}^{k-1}\Phi}{\mathrm{d}t^{k-1}}\right]' = \left(\frac{\lambda}{\varepsilon}\right)^k = \mathcal{O}(\varepsilon^{-k}).$$

It would be better for the conditioning of the Jacobian to unfold the ε-dependency through its recursion given by Ψ_k in Eqs. (27). When applying EJ (and thus also for AMDRK) there holds,

$$\Psi_k = \frac{\lambda}{\varepsilon} y^{(k-1)}, \tag{33}$$

whereas recursion (rec-Direct) gives the relation

$$\Psi_k = \left(\frac{\lambda}{\varepsilon}\right)^{k-1} y^{(1)}, \tag{34}$$

for $k = 1, \ldots, \mathtt{r}$. Already here we can notice that the first out of these two is more favorable, as it unfolds the ε-dependency more thoroughly.

4.2. Recursive dependencies as additional system equations

In order to achieve such an unfolding of the ε-dependency, Baeza et al. [7] suggest to take the derivatives as members of the solution. Instead of directly solving for Y, additionally, the independent unknowns

$$z_k \approx y^{(k)}, \qquad 1 \le k \le \mathtt{r}, \tag{35}$$

are sought for using the same recursive dependencies

[2] The AMDRK method approximates the derivatives $y^{(k)}$ on the basis of finite differences. For linear problems finite differences are exact.

$$\begin{aligned} z_1 &= \Phi(z_0), \\ z_k &= \Psi_k(z_0, z_1, \dots, z_{k-1}), \quad 2 \le k \le \mathtt{r}, \end{aligned} \tag{36}$$

where we have defined $z_0 := Y$. In contrast to the single relation $F(Y) = 0$, we now solve the $\mathtt{r}+1$ relations as a bigger system $\mathcal{F}(z) = 0$, with $z := (z_0, z_1, \dots, z_{\mathtt{r}})$. In summary, the recursive dependency in one single formula is traded off for a larger system containing the $\mathtt{r}$ additional relations given by (36).

Example 2 (part 2). For the third order Taylor scheme (30),

$$z_0 - \Delta t z_1 + \frac{\Delta t^2}{2} z_2 - \frac{\Delta t^3}{6} z_3 - y^n = 0, \tag{37}$$

and

$$\mathcal{F}(z) = \begin{bmatrix} z_0 - \Delta t z_1 + \frac{\Delta t^2}{2} z_2 - \frac{\Delta t^3}{6} z_3 - y^n \\ \Phi(z_0) - z_1 \\ \Psi_2(z_0, z_1) - z_2 \\ \Psi_3(z_0, z_1, z_2) - z_3 \end{bmatrix}. \tag{38}$$

The Jacobian is now less clustered, in our example

$$\mathcal{F}'(z) = \begin{bmatrix} 1 & -\Delta t & \frac{\Delta t^2}{2} & -\frac{\Delta t^3}{6} \\ \Phi'(z_0) & -1 & 0 & 0 \\ \partial_{z_0}\Psi_2 & \partial_{z_1}\Psi_2 & -1 & 0 \\ \partial_{z_0}\Psi_3 & \partial_{z_1}\Psi_3 & \partial_{z_2}\Psi_3 & -1 \end{bmatrix}. \tag{39}$$

In the case of the ε-scaled Dahlquist test (32), the relations (33) and (34) respectively yield

$$\mathcal{F}'_{\mathrm{EJ}}(z) = \begin{bmatrix} 1 & -\Delta t & \frac{\Delta t^2}{2} & -\frac{\Delta t^3}{6} \\ -\frac{1}{\varepsilon} & -1 & 0 & 0 \\ 0 & -\frac{1}{\varepsilon} & -1 & 0 \\ 0 & 0 & -\frac{1}{\varepsilon} & -1 \end{bmatrix} \quad \text{and} \quad \mathcal{F}'_{\mathrm{rec}}(z) = \begin{bmatrix} 1 & -\Delta t & \frac{\Delta t^2}{2} & -\frac{\Delta t^3}{6} \\ -\frac{1}{\varepsilon} & -1 & 0 & 0 \\ 0 & -\frac{1}{\varepsilon} & -1 & 0 \\ 0 & \frac{1}{\varepsilon^2} & 0 & -1 \end{bmatrix}. \tag{40}$$

Regarding AMDRK schemes, Baeza et al. [7] introduce the scaled unknowns $z_k \approx \Delta t^{k-1} \widetilde{y}^{(k)}$, $1 \le k \le \mathtt{r}$. With this choice, analogous relations

$$\begin{aligned} z_1 &= \Phi(z_0), \\ z_k &= \widetilde{\Psi}_k(z_0, z_1, \dots, z_{k-1}), \quad 2 \le k \le \mathtt{r}, \end{aligned} \tag{41}$$

are found on the basis of the formulas (13)-(14), namely

$$\widetilde{\Psi}_k := \Delta t^{k-1} P^{(k-1)} \mathbf{\Phi}_T^{k-1,\langle n \rangle}. \tag{42}$$

In here,

$$\Phi_T^{k-1,n+j} := \Phi\left(z_0 + \Delta t \sum_{m=1}^{k-1} \frac{j^m}{m!} z_m \right), \tag{43}$$

for $j = -p, \dots, p$. Note the slight redefinition of $\Phi_T^{k-1,n+j}$ in contrast to Eq. (14) to account for the Δt dependency of the $\widetilde{\Psi}_k$. For a specific example of the AMDRK method, and its Jacobian $\widetilde{\mathcal{F}}'(z)$, we refer the reader to [7, Subsection 4.2].

As a counterpart to the "Direct" MDRK methods in Section 2, we denote the MDRK approach in which the derivatives are taken as members of the solution by "DerSol". A summary of the six different MDRK approaches is presented in Fig. 1. From the specific Taylor example that we have investigated in this section, there are two important observations to be made:

- Most importantly, compared to $\mathcal{F}'_{\mathrm{EJ}}$ and $\mathcal{F}'_{\mathrm{rec}}$, no second order Jacobian Φ'' occurs for the approximate procedure. From (42)-(43) it can be observed that the AMDRK method solely relies on finite difference computations of Φ. Hence, Φ' is sufficient for retrieving partial derivatives of $\widetilde{\Psi}_k$. If the problem is not scalar anymore ($m > 1$) no tensor calculations are needed, whereas such calculations cannot be avoided for an exact MDRK scheme.
- Starting from three derivatives, the matrices $\mathcal{F}'_{\mathrm{EJ}}$ and $\mathcal{F}'_{\mathrm{rec}}$ are not the same anymore, i.e. $\mathcal{F}'_{\mathrm{rec}}$ will only fill up the first two columns (and the diagonal), whereas $\mathcal{F}'_{\mathrm{EJ}}$ has a full lower-triangular submatrix. In the numerical results below it will be demonstrated that there is significantly different behavior in the conditioning of these Jacobians.

When effectively applying the DerSol approach to several DIMDRK schemes, different orders of ε can be observed in the condition numbers, see Fig. 6. In comparison to the Direct MDRK approach (see Fig. 4), many schemes behave as

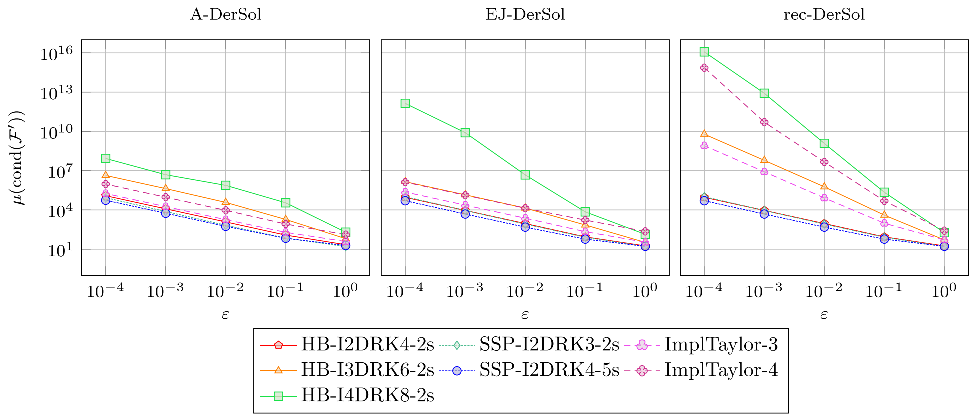


Fig. 6. DIMDRK schemes applied as a DerSol method (Fig. 1) to the PR problem (18). The average condition number in the 1-norm of the Newton Jacobian obtained from the last RK-stage is shown for different values of ε. The behavior $\mathrm{cond}(\mathcal{F}') = \mathcal{O}(\varepsilon^{-1})$ is observed for the A-DerSol and EJ-DerSol methods, the rec-DerSol methods seem to behave as $\mathcal{O}(\varepsilon^{-\mathtt{r}+1})$, $\mathtt{r}$ being the amount of derivatives. A single timestep ($N = 1$) of size $T_{\mathrm{end}} = 1.25$ ($\Delta t = 1.25$) has been considered with tolerances, Eq. (17), set to 10^{-12} under a maximum of 1000 iterations.

$$\mathrm{cond}(\mathcal{F}') = \mathcal{O}(\varepsilon^{-1}), \tag{44}$$

confirming the successful unfolding of the ε-dependency through the $\mathtt{r}$ additional equations in the A-DerSol and (partially) in the EJ-DerSol approach. The same can not be said for the rec-DerSol approach, where the order seems to behave as $\mathcal{O}(\varepsilon^{-\mathtt{r}+1})$. This behavior was foreshadowed in the relation (34); exactly one recursion order is resolved, therefore as well unfolding exactly one order in the ε-dependency. For that reason, it is highly disadvised to apply the rec-DerSol approach for practical purposes.

The EJ-DerSol approach as well does not seem to be flawless when we consider the scheme HB-I4DRK8-2s (Table A.10). Instead, the order $\mathcal{O}(\varepsilon^{-3})$ seems to be achieved. In fact, numerically we observe that the scheme tends toward $\mathcal{O}(\varepsilon^{-2})$ up to $\varepsilon = 10^{-8}$. From running all schemes in Fig. 6 up to $\varepsilon = 10^{-8}$, this behavior appears to be unique among the applied DIMDRK schemes. Even more so, when considering different problems (van der Pol and Kaps, see [16]), all the same schemes show $\mathcal{O}(\varepsilon^{-1})$ up to $\varepsilon = 10^{-8}$, except for HB-I4DRK8-2s applied to van der Pol. For both the A-DerSol and the EJ-DerSol approach, around $\varepsilon \approx 10^{-6}$ there is a sudden change from $\mathcal{O}(\varepsilon^{-1})$ to $\mathcal{O}(\varepsilon^{-4})$ and worse.

This leads us to believe that the observed phenomena of the HB-I4DRK8-2s scheme come as a result of floating-point arithmetic. The double-precision format in MATLAB has a machine precision of $2^{-52} \approx 2.22 \cdot 10^{-16}$. Given a value $\varepsilon = 10^{-4}$, a four-derivative Runge-Kutta method yields values $\varepsilon^4 = 10^{-16}$ in the denominator of $z_4 = \Psi_3(z_0, z_1, z_2, z_3)$. Albeit the implicit Taylor method of order 4 giving the requested behavior for the condition number, the Butcher coefficients are larger compared with those of the HB-I4DRK8-2s scheme (see Tables A.4 and A.10). The application of many-derivative schemes to stiff problems having very small values ε should therefore be regarded with sufficient awareness of the machine accuracy being used.

4.3. *The (A)MDRK scheme for a general amount of stages*

In the most general case, it is not possible to solve for each stage one at a time, an FSMDRK approach is therefore a necessity. Thus, there is a need to solve for $Y = \left(y^{n,1}, \ldots, y^{n,\mathtt{s}}\right)$ at once. This entails that for each stage $\mathtt{r}+1$ separate equations have to be solved, leading to a Jacobian matrix $\mathcal{F}(z)$ of size $((\mathtt{r}+1)\mathtt{s}M)^2$.

When using the DerSol approach, one has two options in which one can order all the unknown variables. Either all the variables of the same stage are grouped together, or either the variables are collected by degree of the derivatives. In this work we have chosen to do the ordering in a *stage-based* manner

$$z = \left(z^{n,1}, \ldots, z^{n,\mathtt{s}}\right), \tag{45}$$

with for each stage $z^{n,l} := (z_0^{n,l}, z_1^{n,l}, \ldots, z_{\mathtt{r}}^{n,l})$. This allows us to obtain an anologous block-structure (21) as in the Direct implementation:

$$\mathcal{F}'(z) = \left[\begin{array}{c|c|c} \partial_{z^{n,1}}\mathcal{F}_1 & \ldots & \partial_{z^{n,\mathtt{s}}}\mathcal{F}_1 \\ \hline \vdots & \ddots & \vdots \\ \hline \partial_{z^{n,1}}\mathcal{F}_{\mathtt{s}} & \ldots & \partial_{z^{n,\mathtt{s}}}\mathcal{F}_{\mathtt{s}} \end{array}\right], \tag{46}$$

where each block-matrix $\partial_{z^{n,\nu}}\mathcal{F}_l$ inside is of size $((\mathtt{r}+1)M)^2$ with a similar construction as the matrix (39) in Example 2 (part 2).

Fig. 7 displays the average condition numbers $\mu\big(\mathrm{cond}(\mathcal{F}')\big)$ for different MDRK schemes. The results are very similar to the ones of the DIMDRK implementation in Fig. 6, thus the previous remarks remaining valid pertaining to the FSMDRK implementation. It is clear that $\mathcal{F}'(z)$ will quickly grow large for an increasing amount of derivatives $\mathtt{r}$ and stages $\mathtt{s}$, and that this consequently has an

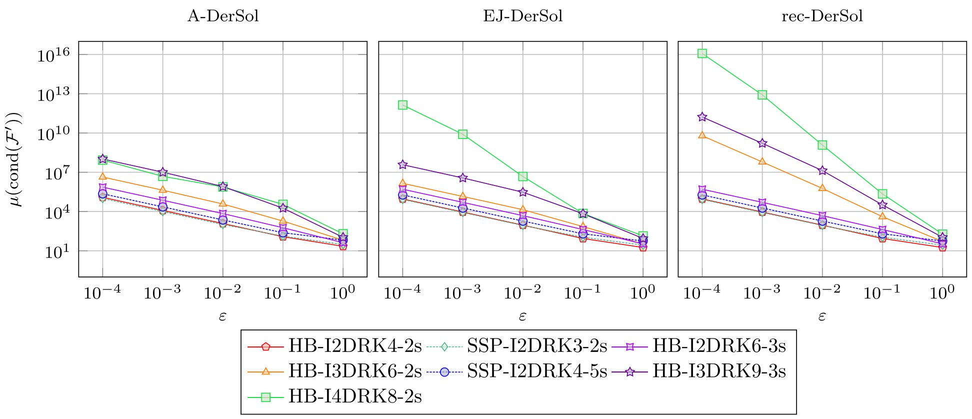


Fig. 7. FSMDRK schemes applied as a DerSol method (Fig. 1) to the PR problem (18). The average condition number in the 1-norm of the Newton Jacobian is shown for different values of ε. The behavior $\mathrm{cond}(\mathcal{F}') = \mathcal{O}(\varepsilon^{-1})$ is observed for the A-DerSol and EJ-DerSol methods, the rec-DerSol methods seem to behave as $\mathcal{O}(\varepsilon^{-\mathfrak{r}+1})$, $\mathfrak{r}$ being the amount of derivatives. A single timestep ($N = 1$) of size $T_{\text{end}} = 1.25$ ($\Delta t = 1.25$) has been considered with tolerances, Eq. (17), set to 10^{-12} under a maximum of 1000 iterations.

impact on the performance of the MDRK method. Still, it might be beneficial to introduce the additional derivative relations for the overall efficiency of the method. As highlighted before w.r.t. the condition of the block-Jacobian (21), here as well $\mathrm{cond}(\mathcal{F}'(z))$ is strongly dependent on the condition of the separate blocks. If $\mathrm{cond}(\partial_{z^{n,\nu}} \mathcal{F}_l) = \mathcal{O}(\varepsilon^{-1})$ can be guaranteed, there might be a significant difference in the total used amount of Newton iterations compared to the Direct counterpart. Furthermore, there is more certainty that the method itself will converge after all, which, for example, is not always the case for A-Direct methods (see Table 1).

5. Numerical results and comparison

In this last section, we aim to show some representative numerical computations, with a special focus on the comparison with Radau schemes, which are classical schemes for stiff equations. In here, we use the two- and three-stage schemes RadauIIA-3-2s and RadauIIA-5-3s of order three and five, respectively, see [18].

As two representative test cases, we use the Pareschi-Russo equation (18) and van der Pol's equation

$$y_1'(t) = y_2, \qquad y_2'(t) = \frac{(1-y_1)^2 y_2 - y_1}{\varepsilon}, \qquad y(0) = \left(2, -\frac{2}{3} + \frac{10}{81}\varepsilon - \frac{292}{2187}\varepsilon^2\right), \tag{47}$$

see [19], with $\varepsilon = 1$ and $\varepsilon = 10^{-4}$. Results for various integration schemes are plotted in Figs. 8,9. The Newton tolerances were chosen dependently on both ε and the testcase, they have been chosen in a manner that all the schemes converge within at most 1000 Newton steps. The tolerances have been chosen uniformly over the methods. It can be seen from the numerical results that with respect to error versus mesh size, the methods presented in this work behave very well in comparison to the Radau schemes for large $\varepsilon = 1$, they behave a bit worse, but still good, for the small $\varepsilon = 10^{-4}$. Runtime efficiency shows a different story. For large values of Δt, and relatively large error levels, the multiderivative schemes tend to be better for Pareschi-Russo, but not for van der Pol. Asymptotically, for $\Delta t \to 0$, the Radau schemes always outperform the multiderivative schemes. For van der Pol and very small ε, this is quite significant, while for Pareschi-Russo, the schemes lie at least in the same bulk part.

The results show that the methods can in many cases compete with more classical and established schemes, in particular with respect to mesh resolution versus error. However, they can not outperform them yet in terms of runtime. Similar results, in the context of a comparison 'Jacobian-free' against 'Jacobian-based', have been made in [6,7]. Still, the difference is not many orders of magnitude, but typically an order-one factor, which is why we have the hope that significant improvements can be made, in particular in the context of the predictor-corrector type methods, as already shown in [20,8]. This framework offers a built-in parallel-in-time capacity, an advantage that Radau and other schemes do not necessarily have. The more efficient realization of the Jacobian-free approach, an adaptive Newton-framework, and many more improvements, are left for future work.

6. Conclusion and outlook

We have developed a family of implicit Jacobian-free multiderivative Runge-Kutta (MDRK) solvers for stiff systems of ODEs. These so-called AMDRK methods have been tailored to deal with the unwanted outcomes that come from the inclusion of a higher amount of derivatives: (1) each added k-th derivative yields a power term ε^k in the denominator, and (2) the complexity of the formulas for the derivatives increases rapidly with each derivative order.

When adopting Newton's method as a nonlinear solver, these two negatives become noticeable in the Jacobian: the condition number of the Jacobian grows exponentially with each added derivative, as well as the Jacobian having to be obtained from intricate

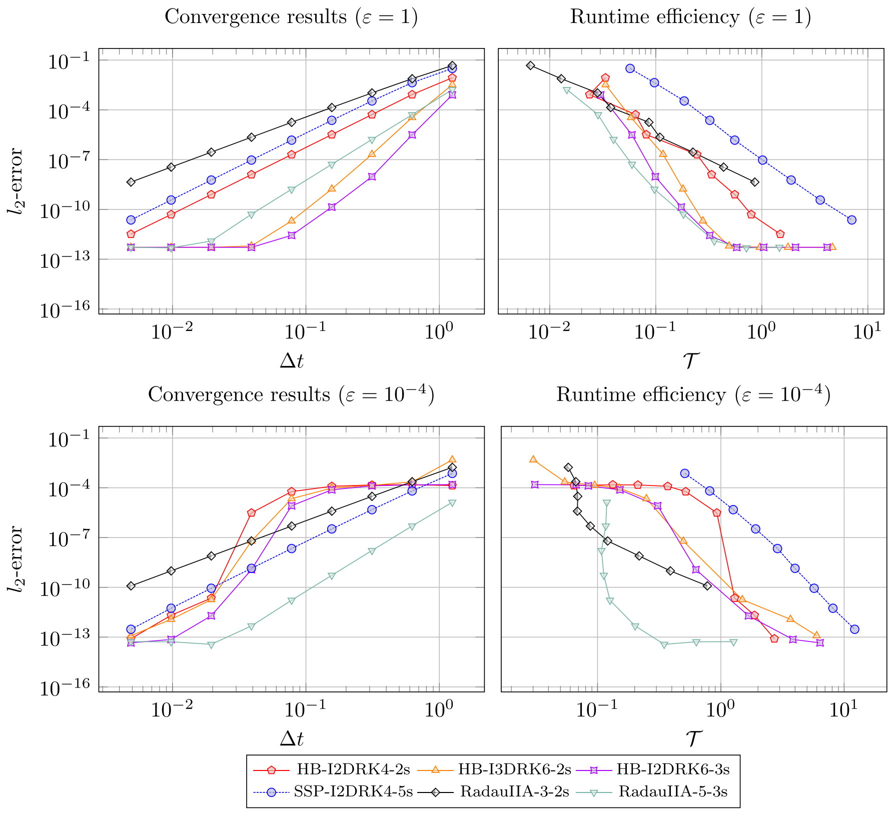


Fig. 8. Convergence results for the Pareschi-Russo equation (18). Top row: $\varepsilon = 1$, bottom row: $\varepsilon = 10^{-4}$. Left are classical convergence results as error over Δt, while on the right, one sees the time-to-solution over Δt. Final time is set to $T_{\text{end}} = 5$; timesteps start from $N = 4$. $\mathcal{T}$ is in seconds.

formulas that request tensor calculations. In order to manage these negatives, the AMDRK methods have been established along the lines of the Approximate Implicit Taylor method in [7].

First, by adding an additional equation to the ODE system for each derivative, the derivatives become a part of the unknown solution, which we named MDRK-DerSol. In this manner, the ε-dependencies are distributed among the newly added relations. Numerically we have shown that this procedure alleviates the exponential growth in the condition number that is typical for direct MDRK methods (correspondingly named MDRK-Direct), for some cases resulting in much less Newton iterations per timestep. Second, by recursively approximating the derivatives using centered differences, no complicated formulas or tensor calculation are needed. The desired convergence order $\min(2p+1, q)$ is achieved, $2p+1$ denoting the amount of stencil points used for the centered differences and q being the order of the MDRK scheme.

Despite the (A)MDRK-DerSol methods for $\varepsilon \to 0$ having a more favorable behavior in the condition number in comparison to (A)MDRK-Direct methods, the total system grows in size, and therefore might be less efficient. In order to balance on the one hand the amount of Newton iterations per timestep, and on the other hand the computing time that is needed for solving the linear system, it might be beneficial in the future to establish a threshold value that switches between (A)MDRK-DerSol and (A)MDRK-Direct methods. Such threshold can play a significant role when transitioning to parabolic PDEs with viscous effects, where the size of linear systems depends on the spatial resolution. A careful consideration w.r.t. efficiency will be needed in the development of MDRK-DerSol approaches for PDEs with viscous effects.

The multiderivative approach puts a high demand on the regularity of the right-hand side Φ. While this makes sense from a conceptual point of view – multiderivative methods are devised to obtain high orders of accuracy, which necessitates smooth solutions y and hence smooth $\Phi(y)$ – it can be a severe drawback in some applications where Φ is weakly non-smooth; and the Jacobians of Φ cannot be computed. One can, e.g., think of the spatial discretization of convective PDEs using some sort of limiting, which is typically non-differentiable. The approximate multiderivative approach weakens this issue, as no Jacobians have to be computed explicitly. A more thorough analysis of the AMDRK method for weakly non-smooth problems is now ongoing.

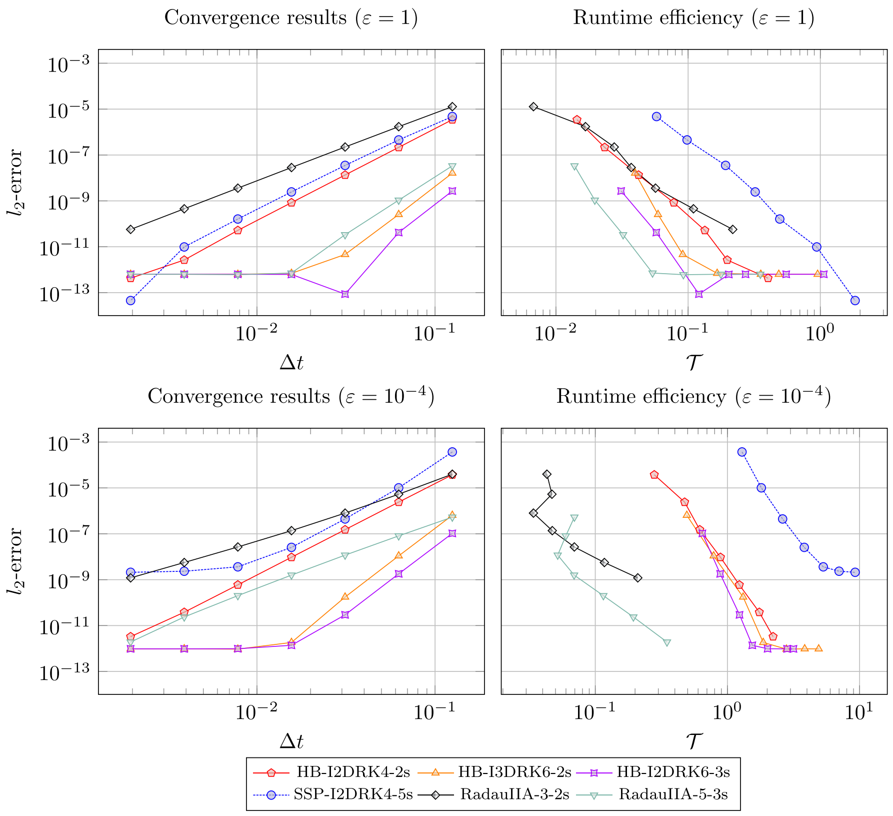


Fig. 9. Convergence results for the van der Pol equation (47). Top row: $\varepsilon = 1$, bottom row: $\varepsilon = 10^{-4}$. Left are classical convergence results as error over Δt, while on the right, one sees the time-to-solution over Δt. Final time is set to $T_{\text{end}} = 0.5$; timesteps start from $N = 4$. $\mathcal{T}$ is in seconds.

Table A.2
A general extended Butcher tableau for a multiderivative Runge-Kutta scheme having $\mathtt{r}$ derivatives and $\mathtt{s}$ stages. The associated matrices and vectors are of size $A^{(k)} \in \mathbb{R}^{\mathtt{s}\times\mathtt{s}}$, $b^{(k)} \in \mathbb{R}^{1\times\mathtt{s}}$ and $c \in \mathbb{R}^{\mathtt{s}\times 1}$ for $k = 1, \dots, \mathtt{r}$.

c	$A^{(1)}$	$\dots$	$A^{(\mathtt{r})}$
	$b^{(1)}$	$\dots$	$b^{(\mathtt{r})}$

Availability of data and material

The datasets generated and/or analyzed during the current study are available from the corresponding author on reasonable request via jeremy.chouchoulis@uhasselt.be.

Declaration of competing 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.

Appendix A. Butcher tableaux

All the used multiderivative Runge-Kutta methods in this paper are displayed in this section. A typical multiderivative Runge-Kutta method can be summarized in an extended Butcher tableau of the form as in Table A.2.

We use the explicit and implicit Taylor method reformulated as RK scheme, two-derivative Hermite-Birkhoff (HB) schemes taken from [8] together with new higher-derivative HB-schemes designed along the same line of reasoning, and Strong-Stability Preserving schemes taken from [9]. The corresponding Butcher tableaux of the HB schemes have been generated using a short MATLAB code which can be downloaded from the personal webpage of Jochen Schütz at www.uhasselt.be/cmat or directly from http://www.uhasselt.be/Documents/CMAT/Code/generate_HBRK_tables.zip.

Table A.3
r-th order explicit Taylor.

0	0	0	…	0
	1	1/2	…	1/r!

Table A.4
r-th order implicit Taylor.

1	1	−1/2	…	$(-1)^{r+1}/r!$
	1	−1/2	…	$(-1)^{r+1}/r!$

Table A.5
HB-I2DRK4-2s: Fourth order implicit two-derivative Hermite-Birkhoff scheme using two stages [8].

0	0	0	0	0
1	1/2	1/2	1/12	−1/12
	1/2	1/2	1/12	−1/12

Table A.6
HB-I2DRK6-3s: Sixth order implicit two-derivative Hermite-Birkhoff scheme using three stages [8].

0	0	0	0	0	0	0
1/2	101/480	8/30	55/2400	65/4800	−25/600	−25/8000
1	7/30	16/30	7/30	5/300	0	−5/300
	7/30	16/30	7/30	5/300	0	−5/300

Table A.7
HB-I2DRK8-4s: Eighth order implicit two-derivative Hermite-Birkhoff scheme using four stages [8].

0	0	0	0	0	0	0	0	0
1/3	6893/54432	313/2016	89/2016	397/54432	1283/272160	−851/30240	−269/30240	−163/272160
2/3	223/1701	20/63	13/63	20/1701	43/8505	−16/945	−19/945	−8/8505
1	31/224	81/224	81/224	31/224	19/3360	−9/1120	9/1120	−19/3360
	31/224	81/224	81/224	31/224	19/3360	−9/1120	9/1120	−19/3360

Table A.8
HB-I3DRK6-2s: Sixth order implicit three-derivative Hermite-Birkhoff scheme using two stages.

0	0	0	0	0	0	0
1	1/2	1/2	1/10	−1/10	1/120	1/120
	1/2	1/2	1/10	−1/10	1/120	1/120

Table A.9
HB-I3DRK9-3s: Ninth order implicit three-derivative Hermite-Birkhoff scheme using three stages.

0	0	0	0	0	0	0	0	0	0
1/2	5669/26880	32/105	−421/26880	303/17920	−1/32	47/17920	169/322560	1/315	−41/322560
1	41/210	64/105	41/210	1/70	0	−1/70	1/2520	2/315	1/2520
	41/210	64/105	41/210	1/70	0	−1/70	1/2520	2/315	1/2520

Table A.10
HB-I4DRK8-2s: Eighth order implicit four-derivative Hermite-Birkhoff scheme using two stages.

0	0	0	0	0	0	0	0	0
1	1/2	1/2	3/28	−3/28	1/84	1/84	1/1680	−1/1680
	1/2	1/2	3/28	−3/28	1/84	1/84	1/1680	−1/1680

Table A.11
SSP-I2DRK3-2s: Third order implicit two-derivative Strong-Stability Preserving scheme using two stages [9].

0	0	0	−1/6	0
1	0	1	−1/6	−1/3
	0	1	−1/6	−1/3

Table A.12
SSP-I2DRK4-5s: Fourth order implicit two-derivative Strong-Stability Preserving scheme using five stages [9].

0.660949255604937	0.660949255604937	0	0	0	0
0.903150646005785	0.660949255604937	0.242201390400848	0	0	0
2.020339810245656	0.660949255604937	0.221847558352979	1.137542996287740	0	0
0.374733308278053	0.060653001401867	0.020022818960029	0.102668776898047	0.191388711018110	0
1.000000000000000	0.060653001401867	0.020022818960029	0.102668776898047	0.191388711018110	0.625266691721946
	0.060653001401867	0.020022818960029	0.102668776898047	0.191388711018110	0.625266691721946
	−0.177750705279127	0	0	0	0
	−0.177750705279127	−0.354733903778084	0	0	0
	−0.177750705279127	−0.324923198367868	−0.403963513682271	0	0
	−0.016311560509453	−0.029325895786881	−0.036459667895230	−0.161628266349058	0
	−0.016311560509453	−0.029325895786881	−0.036459667895230	−0.161628266349058	−0.218859021269943
	−0.016311560509453	−0.029325895786881	−0.036459667895230	−0.161628266349058	−0.218859021269943

References

[1] K. Kastlunger, G. Wanner, On Turan type implicit Runge-Kutta methods, Computing 9 (1972) 317–325.
[2] E. Hairer, G. Wanner, Multistep-multistage-multiderivative methods for ordinary differential equations, Comput. (Arch. Elektron. Rechnen) 11 (1973) 287–303.
[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] D.C. Seal, Y. Güçlü, A. Christlieb, High-order multiderivative time integrators for hyperbolic conservation laws, J. Sci. Comput. 60 (2014) 101–140.
[6] A. Baeza, S. Boscarino, P. Mulet, G. Russo, D. Zorío, Reprint of: approximate Taylor methods for ODEs, Comput. Fluids 169 (2018) 87–97, Recent progress in nonlinear numerical methods for time-dependent flow & transport problems.
[7] A. Baeza, R. Bürger, M. d, C. Martí, P. Mulet, D. Zorío, On approximate implicit Taylor methods for ordinary differential equations, Comput. Appl. Math. 39 (2020) 304.
[8] J. Schütz, D.C. Seal, J. Zeifang, Parallel-in-time high-order multiderivative IMEX solvers, J. Sci. Comput. 90 (2022) 1–33.
[9] 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.
[10] D. Zorío, A. Baeza, P. Mulet, An approximate Lax–Wendroff-type procedure for high order accurate schemes for hyperbolic conservation laws, J. Sci. Comput. 71 (2017) 246–273.
[11] H. Carrillo, C. Parés, Compact approximate Taylor methods for systems of conservation laws, J. Sci. Comput. 80 (2019) 1832–1866.
[12] J. Chouchoulis, J. Schütz, J. Zeifang, Jacobian-free explicit multiderivative Runge-Kutta methods for hyperbolic conservation laws, J. Sci. Comput. 90 (2022).
[13] A. Baeza, S. Boscarino, P. Mulet, G. Russo, D. Zorío, On the stability of approximate Taylor methods for ODE and their relationship with Runge-Kutta schemes, arXiv preprint, arXiv:1804.03627, 2018, arXiv:1804.03627.
[14] A. Quarteroni, R. Sacco, F. Saleri, Numerical Mathematics, Springer, 2007.
[15] L. Pareschi, G. Russo, Implicit-explicit Runge-Kutta schemes for stiff systems of differential equations, Recent Trends Numer. Anal. 3 (2000) 269–289.
[16] J. Schütz, D. Seal, An asymptotic preserving semi-implicit multiderivative solver, Appl. Numer. Math. 160 (2021) 84–101.
[17] S. Boscarino, G. Russo, On a class of uniformly accurate IMEX Runge-Kutta schemes and applications to hyperbolic systems with relaxation, SIAM J. Sci. Comput. 31 (2009) 1926–1945.
[18] E. Hairer, S.P. Nørsett, G. Wanner, Solving Ordinary Differential Equations I, Springer Series in Computational Mathematics, 1987.
[19] S. Boscarino, Error analysis of IMEX Runge-Kutta methods derived from differential-algebraic systems, SIAM J. Numer. Anal. 45 (2007) 1600–1621.
[20] 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.

Chapter 3

Conclusions and future work

User-friendly multiderivative temporal schemes We have extended the CAT method [5] for hyperbolic partial differential equations (PDEs) to more general temporal integration schemes. Specifically, we applied it to Multiderivative Runge–Kutta (MDRK) methods, leading to the development of MDRKCAT[1] methods introduced in [Paper I], and to Multiderivative General Linear Methods (MDGLMs), resulting in the CAMDGLMs presented in [Paper II]. The CAT method is intrinsically based on Taylor schemes, requiring p temporal derivatives to achieve convergence order p. In contrast, MDRK schemes and MDGLMs can attain the same order with fewer temporal derivatives by employing additional stages or previous steps.

By virtue of the Jacobian-free procedure, which approximates the high-order derivatives in a multiderivative scheme, numerical timestepping using any combination of stages, derivatives, and steps becomes practically feasible. In the past, such methods were often avoided due to the high computational cost of symbolic differentiation.

However, it is important to clarify that MDRKCAT methods and CAMDGLMs are not inherently superior to more traditional timestepping schemes. The underlying quadrature rules still rely on several stages, derivatives and steps, often resulting in highly involved expressions. The main benefit of Jacobian-free methods lies in the flexibility they provide for identifying new schemes with the potential to outperform existing ones. Previously, many of these underlying schemes were rarely considered due to their computational demands.

In [Paper II], we capitalized on this advantage to develop novel SSP MDGLMs that incorporate up to four derivatives and achieve a convergence order of nine.

[1]Perhaps a better naming would have been CAMDRK methods, as the letters "AT" refer to an Approximate Taylor method.

Implicit Jacobian-free MDRK methods for stiff ODEs We have done a first analysis on implicit Jacobian-free MDRK solvers for *stiff* systems of ordinary differential equations (ODEs) in [Paper III]. The stiffness in these equations is determined by a variable $\varepsilon \ll 1$ into the flux,

$$\Phi(y) := F(y) + \frac{1}{\varepsilon} G(y),$$

where F and G do not explicitly depend on ε. The motivation for this analysis directly stems from the pursuit of applying Jacobian-free MDRK methods to parabolic PDEs with viscosity effects, such as the Navier-Stokes equations. In such cases, stiffness emerges due to distinct dynamics operating at different time scales. As a result, implicit time integration is necessary, since explicit schemes would require prohibitively small timesteps to maintain stability due to imposed CFL restrictions (see, for example, [24]).

[Paper III] examines the challenges posed by the variable ε in the inclusion of higher-order derivatives, characterizes their impact on the conditioning of the linearized matrix system, and proposes a resolution in the form of a novel family of Jacobian-free MDRK methods. Using Newton's method as an example for linearization, the study demonstrates that incorporating an additional equation for each higher-order derivative spreads out the ε-dependencies. This, in turn, numerically alleviates exponential growth in the condition number, lowering the amount of Newton iterations per timestep, and, under suitable conditions, leads to a more efficient solver.

Improving CFL restrictions In [Paper I], a Von Neumann analysis showed that there is a strong correlation between the CFL number and the amount of derivatives that the MDRKCAT method uses, resulting in better conditions for even-derivative schemes. We believe this stems from the centered-difference approach that is adopted for the approximation of the higher-order derivatives. Likely, other interpolation routines (maybe based on popular ENO/WENO schemes [44]) will yield better Von Neumann stability for explicit odd-derivative schemes.

Yet, for parabolic conservations laws such as the Navier-Stokes equations, CFL conditions imposed on explicit schemes remain too restrictive. Building on the work in [Paper III], we are also interested in developing implicit CAT-like methods. However, an immediate generalization of the family of methods in [Paper III] likely will be inefficient, as the size of resulting linear systems scales with spatial resolution. Hence, trading off ε-dependency for additional equations will have to be carefully evaluated. Introducing a threshold that balances the number of Newton iterations per timestep with the computational cost of solving the linear system could help optimize this trade-off. Another direction worthwhile exploring might be hyperbolization of the parabolic PDE [26].

Moreover, based on the mathematical form in which the stiffness parameter ε appears, it can be reasonable to use an IMEX approach. IMEX methods split the flux in different components, use explicit timestepping for those components that are independent of ε, and use a tailored (semi-)implicit approach for the stiff components. In parts, this approach is pursued in the very novel semi-implicit CAT2 scheme developed by Macca and Boscarino [33].

Realistic frameworks The overarching goal remains to apply efficient multiderivative methods to accurately simulate real-world scenarios, such as Ecotron experiments. In pursuit of that goal, many numerical and programming challenges have to be overcome. For instance, an Ecotron's non-convex geometry necessitates non-Cartesian meshes, contrasting with the Cartesian mesh dependence of CAT-like finite difference approaches. Consequently, key research questions arise emerge:

- Can Jacobian-free multiderivative methods be developed for irregular meshes (e.g., triangular)?
- And if so, can these methods be integrated into element-based solvers like the Discontinuous Galerkin method?

Beyond the challenges of geometric discretization, real-world experiments like those in an Ecotron present fundamental multi-physics problems. The system comprises three distinct layers: soil, canopy, and air. Each of these layers is governed by different physical laws. A complete model would therefore require coupling several models, for instance, by linking the Navier-Stokes equations for free airflow with the Richards equation for porous media flow in the soil. Furthermore, because resolving the intricate canopy geometry directly is computationally infeasible, an effective, upscaled model must be derived via homogenization. The interfaces between these distinct physical domains then demand specialized numerical treatment to capture relevant dynamics.

Additionally, the scale of such simulations imposes substantial computational demands, exceeding the capabilities of general-purpose laptops. Hence, if the intentions are to perform simulations on high-performance computing (HPC) systems, it might be beneficial to fully leverage their potential: algorithmically, by employing advanced parallel-in-time timestep methods [41, 52]; and architecture-wise, through GPU parallelization for computational acceleration (see, for instance, [27]).

Humidity modelling For Ecotrons it is very important to model humidity, as the plant's behavior in the experiment is strongly influenced by the water cycle. With the intent of studying and quantifying the emission, absorption and retention rates of greenhouse gases, preferably, evapotranspiration is taken into account. In the following, we present a model for evapotranspiration based on the assumption that air is a binary mixture of water vapor and dry air [2]. Typically, this is done by defining the *specific humidity* $y_{\text{vap}} := m_{\text{vap}}/m_{\text{totAir}}$ as the mass fraction of vapor in the complete air, such that

$$y_{\text{vap}} + y_{\text{dryAir}} = 1.$$

Fick's law tells us how the water vapor and dry air diffuse into one another, which, in turn, allows us to constitute a conservation law for vapor mass and adjust the heat flux to account for both components (vapor and dry air). In summary, the "humid Navier-Stokes" equations can be formulated as

$$\begin{aligned}
\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \mathbf{u}) &= 0 \\
\frac{\partial (\rho y_{\text{vap}})}{\partial t} + \nabla \cdot (\rho y_{\text{vap}} \mathbf{u}) &= \nabla \cdot (\rho D_{\text{vap}} \nabla y_{\text{vap}}) \\
\frac{\partial (\rho \mathbf{u})}{\partial t} + \nabla \cdot (\rho \mathbf{u} \otimes \mathbf{u}) &= \nabla \cdot (-p\mathrm{I} + \boldsymbol{\tau}) \\
\frac{\partial E}{\partial t} + \nabla \cdot (E\mathbf{u}) &= \nabla \cdot \Big((-p\mathrm{I} + \boldsymbol{\tau})\, \mathbf{u} + \kappa \nabla T + (c_{p_{\text{vap}}} - c_{p_{\text{dryAir}}}) T \rho D_{\text{vap}} \nabla y_{\text{vap}} \Big),
\end{aligned}$$

where D_{vap} represents the diffusion coefficient of water vapor in the air, $c_{p_{\text{vap}}}$ and $c_{p_{\text{dryAir}}}$ signifies the specific heat capacities at constant pressure and κ the thermal conductivity. Future work should focus on the efficient solution of the humid Navier-Stokes equations through CAT on unstructured meshes.

Artificial intelligence and machine learning Lastly, from a more general research perspective, recent mathematical and technological advancements in artificial intelligence — leading to famous large language models such as ChatGPT and Google Gemini — highlight the value of staying informed about developments in the field of machine learning. Machine learning is widely recognized for its capability to analyze observed data and reveal intricate patterns and evolving dynamics.

In Jacobian-free methods, many aspects of the solver involve pattern recognition. Similarly as to how mesh refinement is used to maintain high numerical accuracy in regions with rapidly changing dynamics, a machine learning model could potentially be trained to identify elements requiring higher-order derivatives. This approach could pave the way for a "variable-derivative" time-integrator, dynamically adjusting numerical treatment based on local solution behavior. It is highly important to mention that development of such a machine learning model does not have to come fully out of the blue. Established numerical techniques already exist to address related challenges, such as the erroneous oscillations that can arise from high-order schemes in the presence of sharp gradients. For instance, the a posteriori Multi-dimensional Optimal Order Detection (MOOD) limiting approach [10, 13, 34] provides a relevant framework. This method locally inspects each cell against a set of admissibility criteria after a computational step. If the criteria are not met, the solution in that cell is rejected and recomputed with a progressively lower-order scheme until an acceptable result is achieved, ultimately defaulting to a robust first-order method if necessary. It is therefore plausible that pattern recognition techniques, inspired by existing smoothness indicators and admissibility detectors from methods like MOOD, could form the basis for a powerful and dynamic multi-derivative integration procedure.

Additionally, machine learning may offer a more tractable way to handle stiffness. Viscous effects, for instance, do not always manifest uniformly across the physical domain. This is evident from the variety of existing wall models. Machine learning could help detect stiffness either directly from solution data or through patterns in local matrix structures, improving solver efficiency and adaptability.

Nonetheless, advancements in the application of machine learning techniques to computational fluid dynamics (CFD) are progressing rapidly (see for example [49] for a comprehensive 2024 review). As a first step, we suggest focusing on *machine-learning-assisted numerical methods*, which closely resemble traditional CFD approaches but selectively replace specific solver components to enhance efficiency or accuracy.

Bibliography

[1] A. Baeza, R. Bürger, M. d. C. Martí, P. Mulet, and D. Zorío, *On approximate implicit Taylor methods for ordinary differential equations*, Computational and Applied Mathematics **39**.4 (2020), p. 304, `https://doi.org/10.1007/s40314-020-01356-8`.

[2] R. Bird, E. Lightfoot, and W. Stewart, *Transport Phenomena*, J. Wiley, 2002.

[3] S. Boscarino and 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), pp. 3–49, `https://doi.org/10.1007/s40324-024-00351-x`.

[4] J. C. Butcher, *Numerical Methods for Ordinary Differential Equations*, John Wiley & Sons, Ltd, 2016, `https://onlinelibrary.wiley.com/doi/abs/10.1002/9781119121534`.

[5] H. Carrillo and C. Parés, *Compact approximate Taylor methods for systems of conservation laws*, Journal of Scientific Computing **80** (2019), pp. 1832–1866, `https://doi.org/10.1007/s10915-019-01005-1`.

[6] R. Chan and A. Tsai, *On explicit two-derivative Runge-Kutta methods*, Numerical Algorithms **53** (2010), pp. 171–194.

[Paper III] J. Chouchoulis and J. Schütz, *Jacobian-free implicit MDRK methods for stiff systems of ODEs*, Applied Numerical Mathematics **196** (2024), pp. 45–61, `https://www.sciencedirect.com/science/article/pii/S0168927423002672`.

[Paper I] J. Chouchoulis, J. Schütz, and J. Zeifang, *Jacobian-free explicit multi-derivative Runge-Kutta methods for hyperbolic conservation laws*, Journal of Scientific Computing **90**.96 (2022), `https://link.springer.com/article/10.1007/s10915-021-01753-z`.

[9] P. Ciais, M. Reichstein, N. Viovy, A. Granier, J. Ogée, V. Allard, M. Aubinet, N. Buchmann, C. Bernhofer, A. Carrara, F. Chevallier, N. De Noblet, A. D. Friend, P. Friedlingstein, T. Grünwald, B. Heinesch, P. Keronen, A. Knohl, G. Krinner, D. Loustau, G. Manca, G. Matteucci, F. Miglietta, J. M. Ourcival, D. Papale, K. Pilegaard, S. Rambal, G. Seufert, J. F. Soussana, M. J. Sanz, E. D. Schulze, T. Vesala, and R. Valentini, *Europe-wide reduction in primary productivity caused by the heat and drought in 2003*, Nature **437**.7058 (2005), pp. 529–533, `https://doi.org/10.1038/nature03972`.

[10] S. Clain, S. Diot, and R. Loubère, *A high-order finite volume method for systems of conservation laws—Multi-dimensional Optimal Order Detection (MOOD)*, Journal of Computational Physics **230**.10 (2011), pp. 4028–4050, `https://www.sciencedirect.com/science/article/pii/S002199911100115X`.

[11] Copernicus Climate Change Service (C3S), *Climate Indicators, Temperature indicator* (2024), retrieved October 9th, 2024, `https://climate.copernicus.eu/climate-indicators/temperature`.

[12] Copernicus Climate Change Service (C3S), *Global Climate Highlight 2024* (2025), retrieved March 2nd, 2025, `https://climate.copernicus.eu/global-climate-highlights-2024`.

[13] S. Diot, S. Clain, and R. Loubère, *Improved detection criteria for the Multi-dimensional Optimal Order Detection (MOOD) on unstructured meshes with very high-order polynomials*, Computers & Fluids **64** (2012), pp. 43–63, `https://www.sciencedirect.com/science/article/pii/S0045793012001909`.

[14] M. Dumbser, D. S. Balsara, E. F. Toro, and C.-D. Munz, *A unified framework for the construction of one-step finite volume and discontinuous Galerkin schemes on unstructured meshes*, Journal of Computational Physics **227**.18 (2008), pp. 8209–8253, `https://doi.org/10.1016/j.jcp.2008.05.025`.

[15] M. Dumbser, C. Enaux, and E. F. Toro, *Finite volume schemes of very high order of accuracy for stiff hyperbolic balance laws*, Journal of

Computational Physics **227**.8 (2008), pp. 3971–4001, https://www.sciencedirect.com/science/article/pii/S0021999107005578.

[16] M. Dumbser, F. Fambri, M. Tavelli, M. Bader, and T. Weinzierl, *Efficient Implementation of ADER Discontinuous Galerkin Schemes for a Scalable Hyperbolic PDE Engine*, Axioms **7**.3 (2018), https://www.mdpi.com/2075-1680/7/3/63.

[17] H. Goosse, *Climate System Dynamics and Modelling*, Cambridge University Press, 2015.

[18] S. Gottlieb, C.-W. Shu, and E. Tadmor, *Strong Stability-Preserving High-Order Time Discretization Methods*, SIAM Review **43**.1 (2001), pp. 89–112.

[19] W. Guo, J.-M. Qiu, and J. Qiu, *A new Lax–Wendroff discontinuous Galerkin method with superconvergence*, Journal of Scientific Computing **65**.1 (2015), pp. 299–326.

[20] E. Hairer, S. P. Nørsett, and G. Wanner, *Solving ordinary differential equations I*, Springer Series in Computational Mathematics, 1987, https://doi.org/10.1007/978-3-540-78862-1.

[21] E. Hairer and G. Wanner, *Solving ordinary differential equations II*, Springer Series in Computational Mathematics, 1996, https://doi.org/10.1007/978-3-642-05221-7.

[22] IPCC, *Summary for Policymakers*, in: *Climate Change 2023: Synthesis Report. Contribution of Working Groups I, II and III to the Sixth Assessment Report of the Intergovernmental Panel on Climate Change*, ed. by Core Writing Team, H. Lee, and J. Romero, IPCC, Geneva, Switzerland, 2023, pp. 1–34.

[23] Z. Jackiewicz, *General Linear Methods for Ordinary Differential Equations*, John Wiley & Sons, Ltd, 2009, https://doi.org/10.1002/9780470522165.

[24] A. Jaust, *Novel implicit unconditionally stable time-stepping for DG-type methods and related topics*, PhD thesis, Hasselt University, https://documentserver.uhasselt.be/handle/1942/27133, 2018.

[25] A. Jaust, J. Schütz, and D. C. Seal, *Implicit multistage two-derivative discontinuous Galerkin schemes for viscous conservation laws*, Journal of Scientific Computing **69** (2016), pp. 866–891.

[26] D. I. Ketcheson and A. Biswas, *Approximation of arbitrarily high-order PDEs by first-order hyperbolic relaxation*, arXiv preprint arXiv:2405.16841 (2024).

[27] D. B. Kirk and W. H. Wen-Mei, *Programming massively parallel processors: a hands-on approach*, Morgan kaufmann, 2016.

[28] D. A. Kopriva, *Implementing spectral methods for partial differential equations: Algorithms for scientists and engineers*, Springer Science & Business Media, 2009.

[29] W. Kutta, *Beitrag zur näherungsweisen Integration totaler Differentialgleichungen*, Zeitschrift für Mathematik und Physik **46** (1901), pp. 435–53.

[30] P. Lax and B. Wendroff, *Systems of conservation laws*, Communications on Pure and Applied Mathematics **13**.2 (1960), pp. 217–237, `https://onlinelibrary.wiley.com/doi/abs/10.1002/cpa.3160130205`.

[31] R. J. LeVeque, *Numerical methods for conservation laws*, vol. 214, Springer, 1992.

[32] C. Lu and J. Qiu, *Simulations of Shallow Water Equations with Finite Difference Lax-Wendroff Weighted Essentially Non-oscillatory Schemes*, Journal of Scientific Computing **47**.3 (June 2011), pp. 281–302, `https://doi.org/10.1007/s10915-010-9437-3`.

[33] E. Macca and S. Boscarino, *Semi-implicit-Type Order-Adaptive CAT2 Schemes for Systems of Balance Laws with Relaxed Source Term*, Communications on Applied Mathematics and Computation **7**.1 (2025), pp. 151–178, `https://doi.org/10.1007/s42967-024-00414-w`.

[34] E. Macca, R. Loubère, C. Parés, and G. Russo, *An almost fail-safe a-posteriori limited high-order CAT scheme*, Journal of Computational Physics **498** (2024), p. 112650, `https://www.sciencedirect.com/science/article/pii/S0021999123007453`.

[Paper II] A. Moradi, J. Chouchoulis, R. D'Ambrosio, and J. Schütz, *Jacobian-free explicit multiderivative general linear methods for hyperbolic conservation laws*, Numerical Algorithms (2024), `https://doi.org/10.1007/s11075-024-01771-6`.

[36] J. Qiu, *Development and Comparison of Numerical Fluxes for LWDG Methods*, Numerical Mathematics: Theory, Methods and Applications **1**.4 (2008), pp. 435–459.

[37] J. Qiu, M. Dumbser, and C.-W. Shu, *The discontinuous Galerkin method with Lax–Wendroff type time discretizations*, Computer Methods in Applied Mechanics and Engineering **194**.42-44 (2005), pp. 4528–4543.

[38] J. Qiu and C.-W. Shu, *Finite difference WENO schemes with Lax-Wendroff-type time discretizations*, SIAM Journal of Scientific Computing **24**.6 (2003), pp. 2185–2198.

[39] F. Rineau, R. Malina, N. Beenaerts, N. Arnauts, R. D. Bardgett, M. P. Berg, A. Boerema, L. Bruckers, J. Clerinx, E. L. Davin, H. J. D. Boeck, T. D. Dobbelaer, M. Dondini, F. D. Laender, J. Ellers, O. Franken, L. Gilbert, L. Gudmundsson, I. A. Janssens, D. Johnson, S. Lizin, B. Longdoz, P. Meire, D. Meremans, A. Milbau, M. Moretti, I. Nijs, A. Nobel, I. S. Pop, T. Puetz, W. Reyns, J. Roy, J. Schütz, S. I. Seneviratne, P. Smith, F. Solmi, J. Staes, W. Thiery, S. Thijs, I. Vanderkelen, W. V. Landuyt, E. Verbruggen, N. Witters, J. Zscheischler, and J. Vangronsveld, *Towards more predictive and interdisciplinary climate change ecosystem experiments*, en, Nature Climate Change **9** (2019), pp. 809–816, `https://doi.org/10.1038/s41558-019-0609-3`.

[40] J. Roy, F. Rineau, H. J. De Boeck, I. Nijs, T. Pütz, S. Abiven, J. A. Arnone III, C. V. M. Barton, N. Beenaerts, N. Brüggemann, M. Dainese, T. Domisch, N. Eisenhauer, S. Garré, A. Gebler, A. Ghirardo, R. L. Jasoni, G. Kowalchuk, D. Landais, S. H. Larsen, V. Leemans, J.-F. Le Galliard, B. Longdoz, F. Massol, T. N. Mikkelsen, G. Niedrist, C. Piel, O. Ravel, J. Sauze, A. Schmidt, J.-P. Schnitzler, L. H. Teixeira, M. G. Tjoelker, W. W. Weisser, B. Winkler, and A. Milcu, *Ecotrons: Powerful and versatile ecosystem analysers for ecology, agronomy and environmental science*, Global Change Biology **27**.7 (2021), pp. 1387–1407, `https://onlinelibrary.wiley.com/doi/abs/10.1111/gcb.15471`.

[41] J. Schütz, D. C. Seal, and J. Zeifang, *Parallel-in-Time High-Order Multiderivative IMEX Solvers*, Journal of Scientific Computing **90**.54 (2022), pp. 1–33, `https://doi.org/10.1007/s10915-021-01733-3`.

[42] T. Schwartzkopff, M. Dumbser, and C.-D. Munz, *ADER: A High-Order Approach for Linear Hyperbolic Systems in 2D*, Journal of Scientific Computing **17**.1-4 (2002), pp. 231–240.

[43] D. C. Seal, Y. Güçlü, and A. Christlieb, *High-Order Multiderivative Time Integrators for Hyperbolic Conservation Laws*, Journal of Scientific Computing **60**.1 (2014), pp. 101–140, `https://doi.org/10.1007/s10915-013-9787-8`.

[44] C.-W. Shu, *Essentially non-oscillatory and weighted essentially non-oscillatory schemes*, Acta Numerica **29** (2020), pp. 701–762.

[45] A. Simmons, H. Hersbach, J. Munoz-Sabater, J. Nicolas, F. Vamborg, P. Berrisford, P. de Rosnay, K. Willett, and J. Woollen, *Low frequency variability and trends in surface air temperature and humidity from ERA5 and other datasets*, eng, ECMWF Technical Memoranda 881 (Feb. 2021), `https://www.ecmwf.int/node/19911`.

[46] G. Söderlind, L. Jay, and M. Calvo, *Stiffness 1952–2012: sixty years in search of a definition*, BIT. Numerical Mathematics **55**.2 (2015), pp. 531–558, `https://doi.org/10.1007/s10543-014-0503-3`.

[47] V. A. Titarev and E. F. Toro, *ADER schemes for three-dimensional non-linear hyperbolic systems*, Journal of Computational Physics **204**.2 (2005), pp. 715–736, `https://www.sciencedirect.com/science/article/pii/S0021999104004358`.

[48] V. A. Titarev and E. F. Toro, *ADER: Arbitrary High Order Godunov Approach*, Journal of Scientific Computing **17**.1 (2002), pp. 609–618.

[49] H. Wang, Y. Cao, Z. Huang, Y. Liu, P. Hu, X. Luo, Z. Song, W. Zhao, J. Liu, J. Sun, S. Zhang, L. Wei, Y. Wang, T. Wu, Z.-M. Ma, and Y. Sun, *Recent Advances on Machine Learning for Computational Fluid Dynamics: A Survey*, CoRR **abs/2408.12171** (2024), `https://doi.org/10.48550/arXiv.2408.12171`.

[50] Z. Wang, K. Fidkowski, R. Abgrall, F. Bassi, D. Caraeni, A. Cary, H. Deconinck, R. Hartmann, K. Hillewaert, H. Huynh, N. Kroll, G. May, P.-O. Persson, B. van Leer, and M. Visbal, *High-Order CFD Methods: Current Status and Perspective*, International Journal for Numerical Methods in Fluids **72** (Jan. 2013), pp. 811–845.

[51] Z. Wang, *High-order methods for the Euler and Navier–Stokes equations on unstructured grids*, Progress in Aerospace Sciences **43**.1 (2007), pp. 1–41, `https://www.sciencedirect.com/science/article/pii/S0376042107000450`.

[52] J. Zeifang, A. Thenery Manikantan, and J. Schütz, *Time parallelism and Newton-adaptivity of the two-derivative deferred correction discontinuous Galerkin method*, Applied Mathematics and Computation **457** (2023), pp. 128–198, `https://www.sciencedirect.com/science/article/pii/S0096300323003673`.

[53] 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**.1 (2017), pp. 246–273, `https://doi.org/10.1007/s10915-016-0298-2`.