An explicitness-preserving IMEX-split multiderivative method

E. Theodosiou, J. Schütz and D. C. Seal

UHasselt Computational Mathematics Preprint Nr. UP-23-01

January 25th, 2023

An explicitness-preserving IMEX-split multiderivative method

Eleni Theodosiou[a], Jochen Schütz[a], David Seal

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

Abstract

In the last decade, multi-derivative schemes for ordinary differential equations (ODE), i.e., schemes not only using the ODE's flux, but also derivatives thereof, have seen renewed interest in various applications. In [Schütz and Seal, *Applied Numerical Mathematics*, 160, 2021], the authors have introduced a two-derivative method for applications where a clear distinction can be made between stiff (to be treated implicitly) and non-stiff parts (to be treated explicitly); they hence obtained a two-derivative IMEX (implicit/explicit) scheme. In many applications, it is important that the non-stiff part is only treated explicitly. Think, e.g., of applications where the non-stiff part contains all the nonlinearities and should for efficiency reasons hence only be treated explicitly. However, this non-stiff part showed up in the second derivative of the stiff term and was hence treated implicitly as well. In this work, we further develop the algorithm from Schütz and Seal in such a way that the explicit part will be treated explicitly only. We define the algorithm, show that it preserves the asymptotic behavior of certain stiff equations, and investigate it numerically. We found that the modification we make has no negative effect on the convergence behavior of the scheme.

Keywords: IMEX, multiderivative method

1. Introduction

In this work, we consider multiderivative implicit/explicit (IMEX) schemes for the (typically singularly perturbed) ODE

$$w'(t) = \frac{1}{\varepsilon} Q(w(t); \varepsilon). \tag{1}$$

Our focus here is on the asymptotic case $\varepsilon \to 0$. $Q : \mathbb{R}^n \times \mathbb{R}^{>0} \to \mathbb{R}^n$ is a smooth operator that may depend on ε, yet only in a mild way, i.e., it is possible to expand $Q(w;\varepsilon)$ as

$$Q(w;\varepsilon) = Q_0(w) + \varepsilon Q_1(w) + \mathcal{O}(\varepsilon^2) \tag{2}$$

for smooth functions Q_0 and Q_1 that do not depend on ε. For brevity, in the remainder of this work, we leave out the explicit dependency of Q on ε and write $Q(w)$. As a prototypical example, van-der-Pol like equations [18]

$$y'(t) = z(t), \qquad z'(t) = \frac{g(y,z)}{\varepsilon} \tag{3}$$

are considered. Here, $g(\cdot,\cdot)$ is a sufficiently regular function that allows, together with suitable initial conditions $w(0)$, for a limit as $\varepsilon \to 0$. For more details on the existence of a limit, we refer the reader to standard textbooks, e.g., [20].

In this work, we further develop a method that has been proposed in [33], see also [13] and [34] for extensions, as an IMEX (implicit/explicit) method, i.e., a method where stiff parts of $\frac{1}{\varepsilon}Q(w)$ are treated implicitly, and nonstiff parts explicitly. To define the IMEX scheme, assume that the operator $Q(w)$ can be written as

$$Q(w) = Q_E(w) + Q_I(w). \tag{4}$$

Here, Q_E is the part that should be treated explicitly in the course of the timestepping procedure, while Q_I should be treated implicitly. In practice, Q_I covers 'stiff' parts of the equation, while Q_E covers those that are non-stiff and should, maybe for efficiency or accuracy reasons, be treated explicitly. Note that both Q_E and Q_I might mildly depend on ε as $Q(w)$. For some examples in the context of PDE discretizations, we refer to [10, 12, 17, 25, 29] and the references therein.

The method proposed in [33] is of the two-derivative type, see, e.g., [9, 19, 28, 34, 35, 15] and the references therein. More classical schemes only use the ODE's flux function $\frac{1}{\varepsilon}Q(w)$, which obviously equals $w'(t)$ for a solution to (1). Using (1), one can compute

$$w''(t) = \left(\frac{1}{\varepsilon}Q(w)\right)' = \frac{1}{\varepsilon}Q'(w)w' \overset{(1)}{=} \frac{1}{\varepsilon^2}Q'(w)Q(w) =: \frac{1}{\varepsilon^2}\dot{Q}(w). \tag{5}$$

Two-derivative integration schemes to (1) explicitly take $\frac{1}{\varepsilon^2}Q'(w)Q(w)$ into account. Through this procedure, higher-order schemes with fewer stages or steps can be devised.

As every IMEX scheme, the scheme proposed in [33] can also be used fully implicitly only, i.e., without an explicit contribution. Mathematically, this means that one defines $Q_E(w)$ in (4) as being uniformly zero, and hence, $Q_I(w) \equiv Q(w)$. This implicit part of the scheme has been extended in [37] and applied to the discontinuous Galerkin discretization of the compressible Navier-Stokes equation in [38, 36]. While we found that this method works very successfully on implicit schemes, it has a significant algorithmic drawback when applied to multiderivative IMEX schemes. Consider the process from (5) again, but this time with the IMEX splitting from (4):

$$\begin{aligned} w''(t) &= \frac{1}{\varepsilon}\left(Q(w)\right)' \overset{(4)}{=} \frac{1}{\varepsilon}\left(Q_E(w)+Q_I(w)\right)' \\ &\overset{(1),(4)}{=} \frac{1}{\varepsilon^2}Q_E'(w)\left(Q_E(w)+Q_I(w)\right) + \frac{1}{\varepsilon^2}Q_I'(w)\left(Q_E(w)+Q_I(w)\right) \end{aligned}$$

The quantities

$$\dot{Q}_E(w) := Q_E'(w)(Q_E(w)+Q_I(w)), \tag{6}$$

$$\dot{Q}_I(w) := Q_I'(w)(Q_E(w)+Q_I(w)) \tag{7}$$

are used in any two-derivative IMEX solver; $\dot{Q}_I(w)$ is treated implicitly. This, however, means that also $Q_E(w)$ is treated implicitly, as it is part of the definition of $\dot{Q}_I(w)$. This is not a desirable property. E.g., if the splitting is such that $Q_I(w)$ is linear, inverting $Q_E(w)$ would significantly deteriorate the overall performance as then, $\dot{Q}_I$ would not be linear any more, even though Q_I is. In this work, we propose a modification to the original algorithm in [33] such that this issue is cured. This is achieved by proposing a modification $\mathrm{d}Q_I$ to $\dot{Q}_I$, see (9) below.

This publication is structured as follows: In Sec. 2, the numerical algorithm is being described. The splitting of Q that this algorithm relies on is detailed in Sec. 3. Sec. 4 and Sec. 5 are on convergence and asymptotic properties of the algorithm, respectively. Numerical results for different test equations and different degrees of stiffness are shown and evaluated in Sec. 6 (ODEs) and Sec. 7 (prototypical PDEs). Finally, Sec. 8 offers conclusion and outlook.

2. Algorithm

In this section, we present the algorithm, following the notation introduced in [34]. The algorithm to be presented relies on a two-derivative Runge-Kutta method 'in the background'. This Runge-Kutta method is defined by two Butcher matrices $B^{(1)}$ and $B^{(2)}$ for the first derivative $Q(w)$ and the second derivative $\dot{Q}(w)$, see (5). For a definition of the stages and the update, see [35, Def. 1] and [34, Def. 1]. Please note that these stages/update are never explicitly computed in the algorithm to be presented here, which is why we do not show it. It is assumed that this Runge-Kutta method is globally stiffly accurate, i.e., the update is equal to the last stage. Butcher tableaux $B^{(1)}$ and $B^{(2)}$ that we use in this work are shown in [34, Example 1] and, for convenience, also in the Appendix of this work. The only way the Runge-Kutta method enters the algorithm is through the quadrature rule that it defines, given by [34, eq. (5)]:

$$\mathcal{I}_l(Q(w^1),\dots,Q(w^s)) := \frac{\Delta t}{\varepsilon}\sum_{j=1}^{s} B^{(1)}_{lj} Q(w^j) + \frac{\Delta t^2}{\varepsilon^2}\sum_{j=1}^{s} B^{(2)}_{lj}\dot{Q}(w^j). \tag{8}$$

All Runge-Kutta schemes are characterized by a quantity $q \in \mathbb{N}$ indicating the order of the scheme.

Remark 1. *Originally in [33], the two-derivative trapezoidal rule (fourth-order consistency) has been used, which corresponds to a two-stage scheme with trivial zero quadrature rule for the first stage, and for the second stage a trapezoidal rule*

$$\mathcal{I}_2(Q(w^1),Q(w^2)) := \frac{\Delta t}{2\varepsilon}\left(Q(w^1)+Q(w^2)\right) + \frac{\Delta t^2}{12\varepsilon^2}\left(\dot{Q}(w^1)-\dot{Q}(w^2)\right).$$

The algorithm to be presented depends on this quadrature rule $\mathcal{I}_l$. The key change in comparison to the works in [33] is the change in the quantity $\dot{Q}_I(w)$. Instead of the quantity $\dot{Q}_I(w)$ defined in (6), we use the function $\mathrm{d}Q_I$, defined as

$$\mathrm{d}Q_I(u,v) := Q_I'(v)(Q_E(u)+Q_I(v)). \tag{9}$$

Note that $\dot{Q}_I(w) \equiv \mathrm{d}Q_I(w,w)$. The idea is to put the implicitness on v rather than on u. In this way, the scheme can for example make use of the fact that in many cases, $Q_I(w)$ is 'easier' to invert than $Q_E(w)$.

With these preliminaries, we can now formulate the numerical algorithm which is given in predictor-corrector format. For the ease of presentation, we rely on constant timesteps Δt which, however, is not a necessity. As usual, we define $t^n := n\Delta t$; the same notation as in [34] is used. In particular, we define $Q_I^{n,[k],l} := Q_I(w^{n,[k],l})$ (and similar for other functions) in the following. The meaning of the indices is clarified in Rem. 2 below.

Algorithm 1 (HBPC$(q, k_{\max})$ [34])**.** *The algorithm introduced in [34] is given in predictor-corrector form. In a first step, stage-values are predicted using a second-order IMEX-Taylor scheme; subsequently, these values are corrected towards the background Runge-Kutta scheme.*

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

$$\begin{aligned} w^{n,[0],l} := w^n &+ \frac{c_l \Delta t}{\varepsilon} \left(Q_I^{n,[0],l} + Q_E(w^n) \right) \\ &+ \frac{(c_l \Delta t)^2}{2\varepsilon^2} \left(\dot{Q}_E(w^n) - \mathrm{d}Q_I(w^n, w^{n,[0],l}) \right). \end{aligned} \tag{10}$$

2. ***Correct.*** *The following expression has to be solved for $w^{n,[k+1],l}$ for each $1 \le l \le s$ and $0 \le k \le k_{\max} - 1$:*

$$\begin{aligned} w^{n,[k+1],l} :=& w^n + \theta_1 \frac{\Delta t}{\varepsilon} \left(Q_I^{n,[k+1],l} - Q_I^{n,[k],l} \right) \\ &- \theta_2 \frac{\Delta t^2}{2\varepsilon^2} \left(\mathrm{d}Q_I(w^{n,[k],l}, w^{n,[k+1],l}) - \dot{Q}_I^{n,[k],l} \right) \\ &+ \mathcal{I}_l(Q^{n,[k],0}, \dots, Q^{n,[k],s}). \end{aligned} \tag{11}$$

3. ***Update.***

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

Remark 2. *The method's formulation is rather complex. Let us therefore clarify the following points:*

- *The quantity $w^{n,[k],l}$ is an approximation to $w(t^n + c_l \Delta t)$. The c_l, $1 \le l \le s$, are given by the underlying two-derivative Runge-Kutta method that defines the quadrature rule (8). The integer k, $0 \le k \le k_{\max}$, where $k_{\max}$ is a user-defined parameter, denotes the correction level of the method. Roughly speaking, this corresponds to an accuracy level of order $\Delta t^{\min\{q,k+2\}}$, see Thm. 4.1 below.*

- *The constant scalar values θ_1 and θ_2 are used to improve upon the stabilization of the method, see [37]. Based on the analysis in [37], they are chosen to be $(\theta_1, \theta_2) = (\frac{1}{2}, \frac{1}{6})$ for the fourth-order scheme, $(\theta_1, \theta_2) = (0.283, 0.0528)$ for the sixth-order scheme, and $(\theta_1, \theta_2) = (0.395, 0.0375)$ for the eighth-order scheme, respectively.*

- *The method can, for a given and fixed $k_{\max}$, be written as a two-derivative Runge-Kutta method.*

- *Alg. 1 has been formulated in a time-parallel version [34], which comes down to a slight modification of the terms involving w^n. Then, it can not be written as a two-derivative Runge-Kutta method any more, but the method is of second-derivative GLM-type (general linear method [1, 8, 27]). The time-parallel variant has been explored broadly in [34, 38], it is completely independent of the IMEX modification done here. Therefore, for the sake of an easier presentation, we do not use this modification in this publication.*

- *Please note that $\mathcal{I}_l$ is always evaluated using Q rather than Q_I or Q_E. This is possible as the evaluation of $\mathcal{I}_l$ is always* explicit*, as it is evaluated on lower-level corrections.*

3. Splitting properties

Obviously, not all splittings (4) are reasonable. In particular when approximating PDEs, this can become quite cumbersome and influence the stability of an overall scheme [32]. In this subsection, we define important basic properties on Q_I and Q_E that are needed in the sequel:

Assumption 3.1. *All occuring functions Q, Q_I, Q_E as well their temporal derivatives $\dot{Q}$, $\dot{Q}_I$, $\dot{Q}_E$ are assumed to be smooth and Lipschitz continuous. We assume there exist appropriate constants $L_Q, L_{\dot{Q}} > 0$ with*

$$\|Q(w_1) - Q(w_2)\| \leq L_Q \|w_1 - w_2\|, \quad \|\dot{Q}(w_1) - \dot{Q}(w_2)\| \leq L_{\dot{Q}} \|w_1 - w_2\|,$$

similarly for $\dot{Q}_I$ and $\dot{Q}_E$.

Definition 1 (Admissible splitting)**.** *A splitting of $Q(w)$ into $Q_I(w)$ and $Q_E(w)$ is* admissible *if there holds:*

- *(Consistency)* $Q(w) = Q_I(w) + Q_E(w)$.
- *(ε-smoothness) Both Q_I and Q_E can be written in terms of a Hilbert expansion as in (2), i.e.,*

$$Q_I(w) = Q_{I,0}(w) + \varepsilon Q_{I,1}(w) + \mathcal{O}(\varepsilon^2), \tag{12}$$
$$Q_E(w) = Q_{E,0}(w) + \varepsilon Q_{E,1}(w) + \mathcal{O}(\varepsilon^2). \tag{13}$$

- *(Non-stiffness of Q_E) It is assumed that $Q_{E,0} \equiv 0$.*
- *(Asymptotic preserving) The equality*

$$Q'_{I,0}(w)Q_{I,0}(w) = 0 \tag{14}$$

implies $Q_0(w) = 0$.

Remark 3. *Please note the following easy consequences of Def. 1:*

- *From the ε-smoothness, there follows that also $\mathrm{d}Q_I$ has a Hilbert expansion, i.e.,*

$$\mathrm{d}Q_I(w) = \mathrm{d}Q_{I,0}(w) + \varepsilon \mathrm{d}Q_{I,1}(w) + \mathcal{O}(\varepsilon^2).$$

- *From the fact that $Q_{E,0} \equiv 0$, there follows that $\dot{Q}_E(w) = \mathcal{O}(\varepsilon^2)$ if only $Q(w) = \mathcal{O}(\varepsilon)$. This is a straightforward consequence of the definition $\dot{Q}_E := Q'_E(w)Q(w)$.*

Remark 4. *While the first three points of Def. 1 are relatively obvious, the fourth one deserves some explanation. Assume that the simplest IMEX scheme, IMEX-Euler, is being considered, i.e.,*

$$\frac{w^{n+1} - w^n}{\Delta t} = \frac{1}{\varepsilon}\left(Q_E(w^n) + Q_I(w^{n+1})\right). \tag{15}$$

The ε-expansion to lowest order of (1) is given by

$$Q_0(w_0) = 0,$$

where a Hilbert expansion of w, i.e., $w = w_0 + \varepsilon w_1 + \mathcal{O}(\varepsilon^2)$, is assumed. It is thus reasonable to use initial conditions w^0 such that there holds $Q_0(w_0^0) = 0$, where w_0^0 is the lowest-order expansion of w^0. Assuming that w^1 has a Hilbert

expansion as well, which can be proved in some cases [5, 25], reveals that to lowest order in ε, (15) *yields*

$$0 = Q_{E,0}(w_0^0) + Q_{I,0}(w_0^1).$$

Due to the asymptotic preserving property (just multiply by $Q'_{I,0}(w)$), this implies that $Q_0(w_0^1)$ is zero, which hence means that the numerical methods respects the continuous asymptotics.

Remark 5. *The standard – and most straightforward – splitting for* (3) *as, e.g., used in [4], yields*

$$Q_I(w) = \begin{pmatrix} 0 \\ g(y,z) \end{pmatrix}, \qquad Q_E(w) = \begin{pmatrix} \varepsilon z \\ 0 \end{pmatrix}. \tag{16}$$

Most splitting requirements are easy to show. The asymptotic preserving (AP) property necessitates that $\partial_z g(y,z) \neq 0$, which is a natural condition to require from a singular perturbation kind of view, see [20].

4. Order of convergence

In [33, 34], a thorough consistency analysis has been done. It turns out that the order of the method is the order of the predictor – in this special case here two – increased by one for each additional correction step, until the maximum order of the underlying Runge-Kutta scheme is obtained. This is formalized in the following theorem:

Theorem 4.1. *Let the solution to* (1) *be sufficiently smooth. Furthermore, let $T_{end} > 0$ be fixed and consider Δt such that it is possible to find $N \in \mathbb{N}$ with $N \cdot \Delta t = T_{end}$. Then, for the approximations generated using Alg. 1, there holds that*

$$\|w^N - w(T_{end})\|_\infty = \mathcal{O}(\Delta t^{\min\{q, 2+k_{\max}\}}),$$

where q is the order of the underlying Runge-Kutta method (8)*.*

The proof is a straightforward extension of the proof done in [33], it is hence omitted here. Please note that, in contrast to the setting in [34], we do not treat the time-parallel case, hence, the method can be formulated as a one-step method. Convergence follows thus trivially from consistency.

5. Asymptotic properties of the algorithm

With these preliminaries in place, we can now show that Alg. 1 is asymptotic preserving, meaning that $\frac{Q(w)}{\varepsilon}$ is bounded (for $\varepsilon \to 0$) also for the discrete solution. We start by showing this property for the predictor, which is the most difficult part; the corrector then follows easily.

Theorem 5.1. *Let $\overline{w}$ be defined as*

$$\overline{w} = w^n + \frac{\Delta t}{\varepsilon}\left(Q_I(\overline{w}) + Q_E(w^n)\right) + \frac{\Delta t^2}{2\varepsilon^2}\left(\dot{Q}_E(w^n) - \mathrm{d}Q_I(w^n, \overline{w})\right). \quad (17)$$

It is assumed that $\overline{w}$ has a Hilbert expansion. Then, there holds

$$Q_0(\overline{w}_0) = 0.$$

Proof. There holds

$$\begin{aligned} Q_I(\overline{w}) &= Q_{I,0}(\overline{w}) + \varepsilon Q_{I,1}(\overline{w}) + \mathcal{O}(\varepsilon^2) \\ &= Q_{I,0}(\overline{w}_0) + \varepsilon\left(Q'_{I,0}(\overline{w}_0)\overline{w}_1 + Q_{I,1}(\overline{w}_0)\right) + \mathcal{O}(\varepsilon^2). \end{aligned}$$

Similarly, $\mathrm{d}Q_I$ is expanded as

$$\mathrm{d}Q_I(w^n, \overline{w}) = \mathrm{d}Q_{I,0}(w^n, \overline{w}_0) + \varepsilon\left(\mathrm{d}Q'_{I,0}(w^n, \overline{w}_0)\overline{w}_1 + \mathrm{d}Q_{I,1}(w^n, \overline{w}_0)\right) + \mathcal{O}(\varepsilon^2).$$

Plugging this into (17) and separating scales yields that to highest order $\mathcal{O}(\varepsilon^{-2})$, there holds

$$\dot{Q}_{E,0}(w_0^n) - \mathrm{d}Q_{I,0}(\overline{w}_0) = 0.$$

It is easy to see that

$$\mathrm{d}Q_{I,0}(\overline{w}_0) = Q'_{I,0}(\overline{w}_0)(Q_{E,0}(w_0^n) + Q_{I,0}(\overline{w}_0)).$$

From the non-stiffness of the explicit part, it is known that $Q_{E,0} = 0$. Then, this and the property (14) implies that $Q_0(\overline{w}_0) = 0$ and also $Q_{I,0}(\overline{w}_0) = 0$. (This last fact is true because Q_0 and $Q_{I,0}$ must coincide due to $Q_{E,0} \equiv 0$.) □

Remark 6. *Obviously, the same statement is also true for the predictor of Alg. 1, as the c_l do not depend on ε.*

Theorem 5.2. *Let w^{n+1} be the solution from Alg. 1. Furthermore, assume that both w^n and w^{n+1} possess Hilbert expansions. Then, there holds $Q_0(w_0^{n+1}) = 0$.*

Proof. The proof works inductively on k, and is very much alike to the one of Thm. 5.1, which is why we omit it here. □

Remark 7. *Proving that a Hilbert expansion exists has been done for a related algorithm in [33] using the theorem of Newton-Kantorovich.*

6. Numerical results: ODEs

In this section, we show numerical results, demonstrating the theoretical findings and the capabilities of the presented algorithm. As background schemes, we choose the fourth, sixth, and eighth order scheme, respectively, from [34, Example 1]; leading to the HBPC(4, $k_{\max}$), HBPC(6, $k_{\max}$) and HBPC(8, $k_{\max}$), respectively. The underlying schemes from [34] are classical Hermite-Birkhoff collocation schemes, set up through an interpolation with equally spaced nodes, and two derivatives per node. Van der Pol's equation is used in the sequel. This is eq. (3) with function g defined by

$$g(y, z) = (1 - y^2)z - y.$$

Initial conditions are given by [3]

$$\begin{pmatrix} y(0) \\ z(0) \end{pmatrix} = \begin{pmatrix} 2 \\ -\frac{2}{3} + \frac{10}{81}\varepsilon - \frac{292}{2187}\varepsilon^2 \end{pmatrix}, \tag{18}$$

this ensures that the solution remains smooth, independently of ε, at least for some (finite) time t. To keep away from the critical time where the solution becomes non-smooth in the transition $\varepsilon \to 0$, we use $T_{\text{end}} = 0.5$. We use the convention that $w = (y, z)$.

The splitting that we use is the straightforward one from eq. (16), where all terms in Q depending on ε will be put to the explicit part; the others will be put to the implicit part. The following three different algorithms are considered here:

- (Novel IMEX) The IMEX scheme developed in this work, see Alg. 1.
- (Classical IMEX) The IMEX scheme developed in [33, 34]. Based on a splitting $Q(w) = Q_I(w) + Q_E(w)$, the only difference to this work here is the replacement of $\mathrm{d}Q_I$ in Alg. 1 by the actual quantity $\dot{Q}_I(w) := Q_I'(w)Q(w)$.
- (Fully implicit) The fully implicit one. This corresponds to formally setting $Q_I(w) \equiv Q(w)$, and hence $Q_E(w) \equiv 0$, in Alg. 1.

In all our computations, we use Newton's method as a method to solve the arising nonlinear systems of equations. Newton's solver's absolute and relative tolerance are set to 10^{-12}, with a maximum of 2000 iterations. For

all convergence plots, the coarsest computation uses only four sub intervals, the finest one 2^9 subintervals. This comes hence down to time steps ranging from $\Delta t = \frac{T_{\text{end}}}{2^2}$ to $\Delta t = \frac{T_{\text{end}}}{2^9}$. As error measure, we use the quantity

$$e_{\Delta t} := \|w^N - w(T_{\text{end}})\|_2,$$

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

Remark 8. *If Q_I is a linear function in w, then the equations (10) and (11) are linear, so for the novel IMEX method, Newton is not needed (or, equivalently, only one Newton step is needed). This is the case for the viscous Burgers equation in Sec. 7.1. For the ODE used here, this is not the case, so a nonlinear root-finding algorithm (Newton) is still necessary. We have chosen van der Pol in order to show that Alg. 1 behaves qualitatively and quantitatively similar to the original algorithm, which is of course important for a later extension.*

6.1. Asymptotic behavior and comparison: Fourth-order scheme

Of utmost importance to a succesful IMEX method is obviously its behavior for stiff equations. It has already been shown in Thm. 5.2 that the method preserves the asymptotics of the equation. In this subsection, numerical results for different ε are shown, along with a comparison of the splitting developed in [33, 34]. Fig. 1 shows numerical computations of the HBPC(4, $k_{\max}$)-scheme for the van der Pol equation for various ε, $k_{\max}$ and the three different splitting types considered here (the newly developed one, the one from [33], and the fully implicit trivial splitting).

Several remarks need to be made: First of all, for 'moderately small' $\varepsilon > 10^{-4}$, fourth order is achieved for all three splittings throughout all Δt considered. For smaller ε, it needs more $\Delta t-$refinement to reach the asymptotic zone. Second, it is clearly visible that, as $k_{\max}$ increases, the errors become substantially smaller for the splitted schemes – in particular for the very small $\varepsilon = 10^{-6}$ that ultimately also shows fourth order convergence. The fully implicit scheme behaves very well in the first place for all ε. Also, it is visible that all the schemes seem to converge to a common scheme for larger $k_{\max}$, which is not a surprise, as in this case, all schemes converge against the background scheme. Furthermore, for $k_{\max} = 20$, the numerical error is hardly dependent on ε, a similar trend can already be seen for lower $k_{\max}$.

All these observations are very much in line with the observations made in [33, 34]. Concerning the asymptotic results, the ones shown here also compare very favorably with similar results (on another equation) to those made in [15]. The most important conclusion of this section is that the novel splitting, even though it contains more 'explicit' terms than the one from [33], does not interfere with either numerical or asymptotic accuracy of the method.

Similar investigations for Pareschi-Russo and Kaps equation ([30] and [22], both not shown here for brevity) have been done. They show similar behavior, although for Pareschi-Russo equation, the background scheme does not perform too well, quite some order reduction is visible for small ε.

6.2. Comparison to IMEX Runge-Kutta schemes

Obviously, IMEX Runge-Kutta schemes have a long tradition in numerical analysis, see, e.g., [2, 6, 7, 31, 26] (which, obviously, is a highly incomplete list!) and the references therein. We have chosen to compare our scheme to a couple of IMEX Runge-Kutta schemes from literature, see Fig. 2. Testcase is van der Pol equation with $\varepsilon = 0.1$ and $\varepsilon = 10^{-5}$, respectively; and we compare to the fourth-order scheme from Alg. 1 for various values of $k_{\max}$. For the non-stiff case of $\varepsilon = 0.1$, it can be seen that the HBPC scheme outperforms the schemes that were chosen for comparison in terms of Δt versus size of the error. In particular, as one would expect, for this case there is hardly any difference between choosing $k_{\max} = 2$, 5 or 20, respectively. For the stiff case, results are in some way similar in the sense that for $k_{\max} > 2$, the HBPC scheme outperforms all schemes, with the exception of the ARK4SA scheme from [26] for large Δt. As a conclusion, it can be seen from these plots that the scheme presented in Alg. 1 performs very well also in comparison to more established schemes from literature.

6.3. Asymptotic behavior and comparison: Higher-order schemes

While in the last subsections, we have been discussing fourth-order schemes, here, we consider sixth- and eighth-order schemes. The results are pretty much in line with the results in the last section. In Fig. 3, numerical results are shown for the eighth-order scheme HBPC(8, $k_{\max}$) and different values of ε and $k_{\max}$. We have inclued higher values of $k_{\max}$ as before, as it is clearly visible from the figures that the method based on the novel splitting takes significantly more steps to converge towards the background scheme. While classical splitting and fully implicit scheme are already converged at

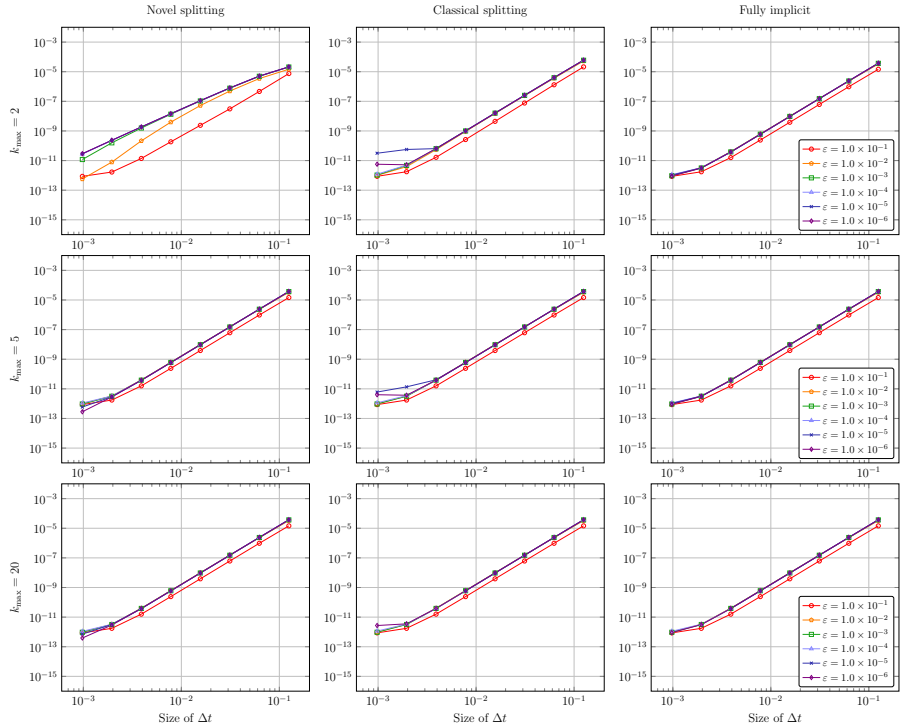

Figure 1: Numerical results for van der Pol equation (3) and (18) with varying ε. Plotted is the error $e_{\Delta t}$ versus Δt. The first row shows numerical results for $k_{\max} = 2$, the second for $k_{\max} = 5$, and the third for $k_{\max} = 20$. The used scheme is the HBPC(4, $k_{\max}$) scheme. In the first column, we present numerical results for the novel IMEX splitting, in the second for the classical splitting from [33], and the third column is fully implicit, without a splitting. The legend in the right plot is valid for all plots.

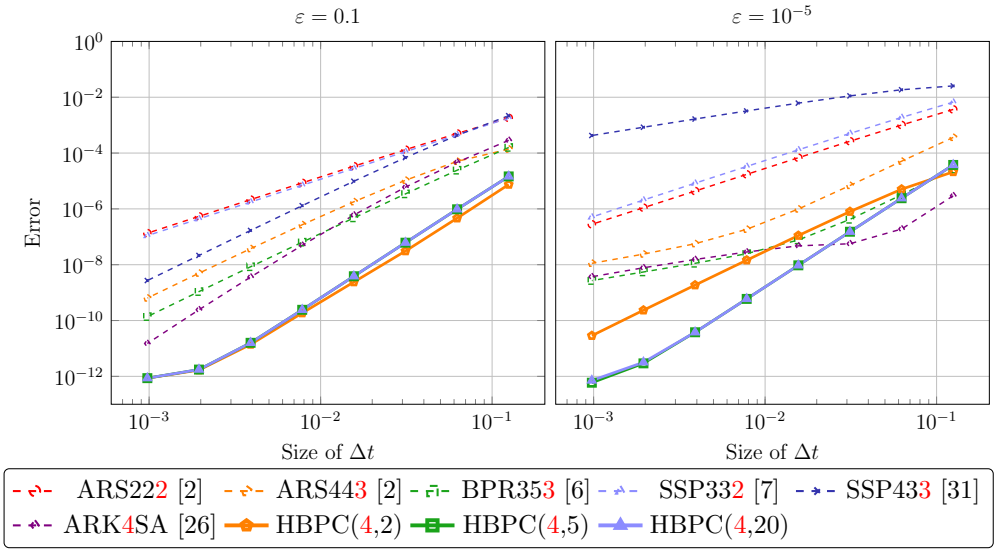

Figure 2: Comparison among different IMEX one-derivative Runge-Kutta schemes and the scheme HBPC(4, $k_{\max}$) with $k_{\max} = 2$, $k_{\max} = 5$, and $k_{\max} = 20$. Numerical results for van der Pol equation (3) and (18) with $\varepsilon = 0.1$ (left) and $\varepsilon = 10^{-5}$ (right). Plotted is the error $e_{\Delta t}$ versus Δt. The order of the IMEX Runge-Kutta schemes can be seen from the red digit in the legend.

$k_{\max} = 20$, it takes (for very low values of ε) more than 100 correction steps for the novel splitting. This can be explained through the fact that the novel splitting comes with less implicitness, making a convergence in k more difficult. However, we see that even for low values of $k_{\max}$, there is no stability issue and, at least for moderate values of ε, errors are in the regime of the background scheme. Please do also note that convergence in k is not a necessity for the algorithm to work; and that for practical problems, iterations of the novel scheme will be significantly cheaper than those of the others. The fact that it takes longer to converge to the background scheme is something that manifests itself also for other problems, e.g., for Pareschi-Russo equation (not shown here).

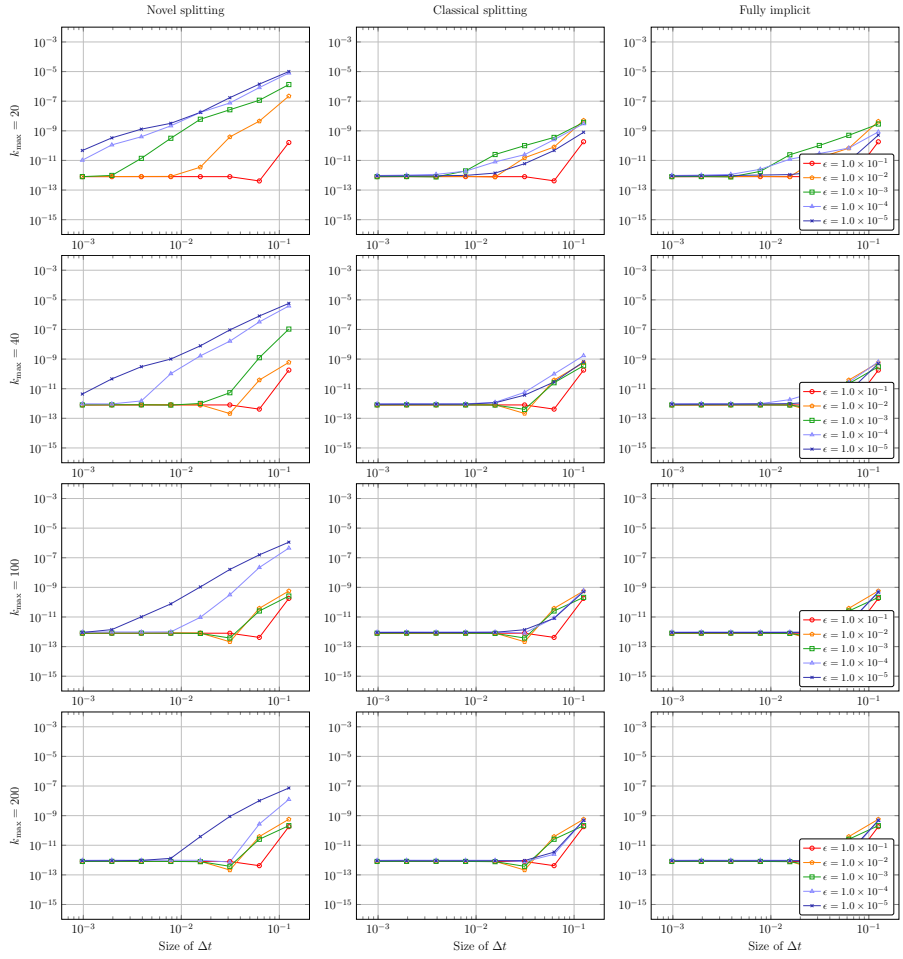

Figure 3: Numerical results for van der Pol equation (3) and (18) with varying ε. Plotted is the error $e_{\Delta t}$ versus Δt. The first row shows numerical results for $k_{\max} = 20$, the second for $k_{\max} = 40$, third for $k_{\max} = 100$ and the last for $k_{\max} = 200$. The used scheme is the HBPC(8, $k_{\max}$) scheme. In the first column, we present numerical results for the novel IMEX splitting, in the second for the classical splitting from [33], and the third column is fully implicit, without a splitting. The legend in the right plot is valid for all plots.

7. Numerical results: PDEs

In this section, we consider the application of Alg. 1 to prototypical partial differential equations. This serves as a proof-of-concept and hence, we will use rather straightforward spatial discretization procedures in the following, namely Lagrange-type finite differences. The algorithm will be tested on Burgers equation and Cahn-Hilliard equation.

7.1. Viscous Burgers equation

As a testcase where obviously two very different operators (nonlinear convection and linear diffusion) are involved, we consider the viscous Burgers equation

$$w_t + \left(\frac{w^2}{2}\right)_x = w_{xx}, \qquad (x,t) \in (0, 2\pi) \times (0, T_{\text{end}} = 0.5), \tag{19}$$

$$w(x,0) = \sin(x)^2, \qquad x \in (0, 2\pi), \tag{20}$$

with periodic boundary conditions in space. An exact solution can be obtained through the Cole-Hopf transformation [21]. As underlying spatial discretization, we choose finite differences based on the eighth-order Lagrange polynomial, yielding

$$\Delta x \left(\frac{w^2}{2}\right)_x \approx \sum_{i=-4}^{4} \alpha_i \left(\frac{w_i^2}{2}\right), \quad \Delta x^2 w_{xx} \approx \sum_{i=-4}^{4} \beta_i w_i.$$

w_i denotes an approximation to w at point x_i (for the respective time level, obviously); and the coefficients α and β are given by

$$\alpha = \frac{1}{5040}\left(18, -192, 1008, -4032, 0, 4032, -1008, 192, -18\right),$$
$$\beta = \frac{1}{5040}\left(-9, 128, -1008, 8064, -14350, 8064, -1008, 128, -9\right).$$

As we are in the diffusion-dominated regime in this work, we do not consider any upwinding here. This, however, could be easily incorporated through a suited WENO scheme or the like.

An obvious splitting here is to treat the diffusion w_{xx} implicitly, while treating the convection term $\left(\frac{w^2}{2}\right)_x$ explicitly. The explicit part is then nonlinear, while the implicit part is linear. This is exactly the case that we

had in mind when developing the current algorithm. The quantity $\mathrm{d}Q_I(u, v)$ will remain linear in v, hence, there is no nonlinearity that needs to be solved in the IMEX procedure of Alg. 1.

In this work, we are interested in temporal convergence only. Hence, we fix the spatial resolution to consist of $N_x = 140$ points; mesh spacing is then given by $\Delta x = \frac{2\pi}{140}$. The coarsest computation uses four temporal subintervals, i.e., $\Delta t = \frac{1}{8}$. Please note that on the coarsest level, this corresponds to a CFL number at time $t = 0$ ($\max_x |\sin(x)^2| = 1$) of $\frac{\Delta t}{\Delta x} = \frac{140}{16\pi} \approx 2.8$ for this high-order scheme. This is only possible because the viscosity is treated implicitly. The finest computation uses 2^9 spatial subintervals. As an error measure, we use the quantity

$$e_{\Delta t} := \|w^N - w(T_{\text{end}})\|_2,$$

where we have defined as before the quantity $N := \frac{T_{\text{end}}}{\Delta t}$. The vector w^N is the collection of the spatial values of the function at time T_{end}, i.e.,

$$w^N = (w_1^N, \ldots, w_{N_x}^N)^T,$$

where $w_i^N \approx w(x_i, T_{\text{end}})$. The $i-$th component of $w(T_{\text{end}})$ is given by the exact solution at point (x_i, T_{end}), with $x_i = i\Delta x$. Fig. 4 shows convergence results for the fourth-, the sixth- and the eighth-order scheme for several values of $k_{\max}$ and the three splittings discussed in this work. It can be clearly seen that convergence orders are met until the spatial accuracy for this given mesh resolution at an error level of about 10^{-11} is hit. No stability issue occurs.

This testcase is prototypical for the applications we have in mind, which are Euler equations at low Mach numbers, see also Sec. 8: The implicit part is a linear function in the unknown, while the explicit part is a highly nonlinear function. It is for exactly this kind of testcases that the novel method should have an advantage over the existing ones as, due to the linearity of the problem, one only needs one Newton step per solve rather than possibly multiple. In Fig. 5, the amount of Newton steps per algebraic solve is plotted for the HBPC(8, $k_{\max}$) scheme for various $k_{\max}$. Newton's tolerances, both relative and absolute, have been set to 10^{-10} with a maximum of 10 iterations. The following observations can be made: For 'stiffer' problems – in this case this means that Δt is large – the difference between the fully implicit/classical scheme and the novel one from Alg. 1 is quite huge, up to a factor of three.

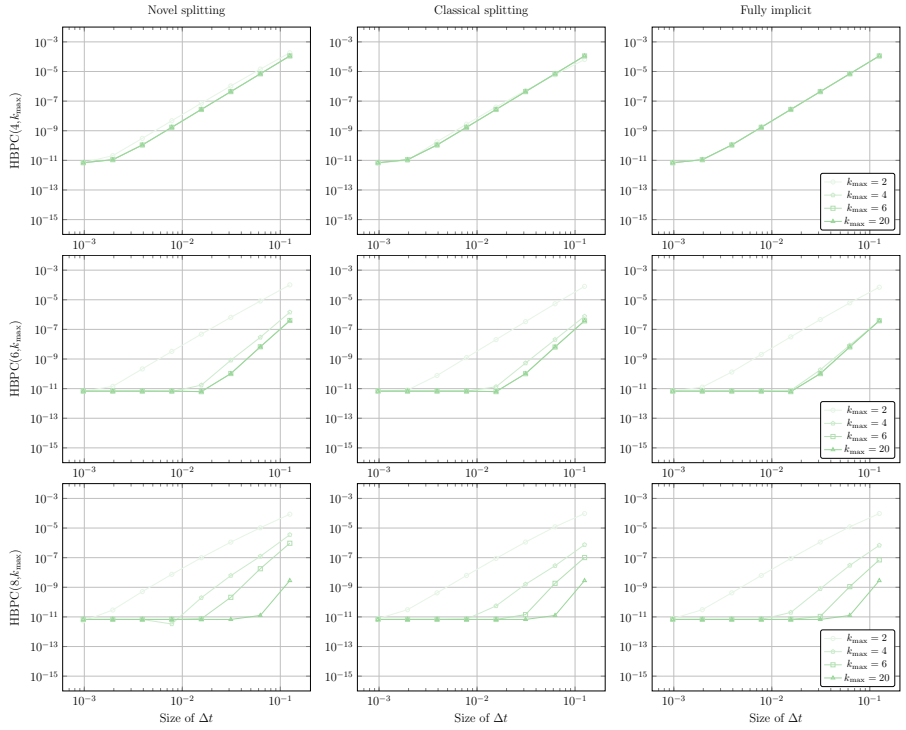

Figure 4: Numerical results for Burgers equation (19). Plotted is the error $e_{\Delta t}$ versus Δt for different values for the $k_{\max}$. The first row shows numerical results for the method HBPC(4, $k_{\max}$), the second for the method HBPC(6, $k_{\max}$) and the third for the method HBPC(8, $k_{\max}$). In the first column, we present numerical results for the fully implicit scheme, in the second for the splitting from [33], and the third column is the novel splitting. The legend in the right plot is valid for all plots.

If the problem becomes less stiff, i.e., if Δt becomes smaller, then this difference also becomes smaller. Furthermore, with growing $k_{\max}$, the difference becomes smaller. Also this is to be expected, because the higher corrections will not add much to the solution in this case, and hence, the algebraic solves can be done with one Newton step also for the fully implicit and the classical scheme.

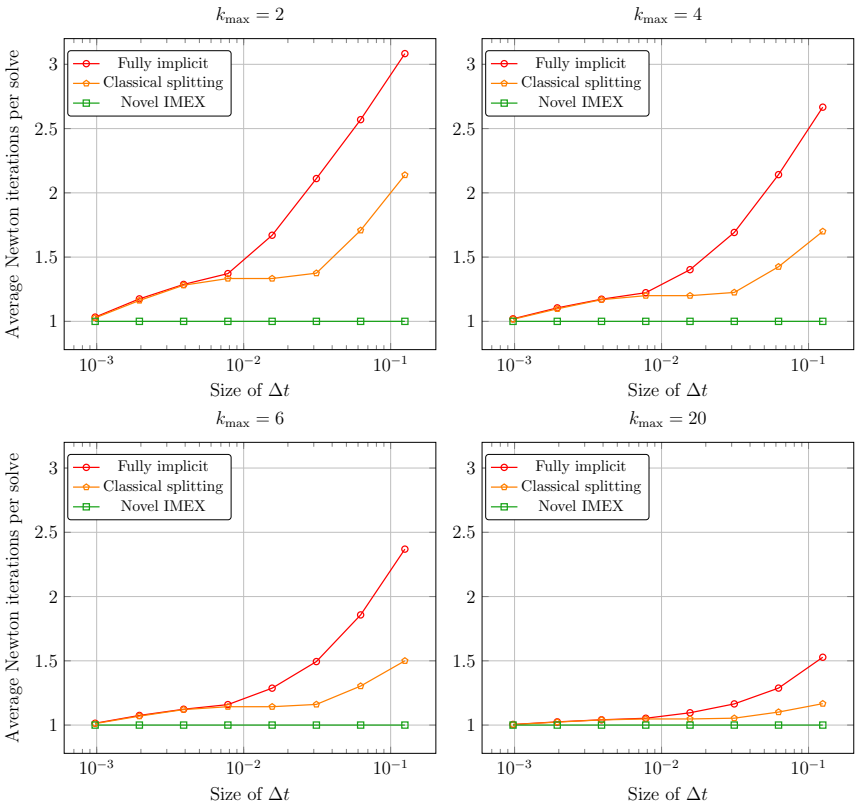

Figure 5: Numerical results for Burgers equation (19). Plotted is the relative number of Newton iterations versus Δt for all the splittings for the method HBPC(8, $k_{\max}$) and different number of correction steps $k_{\max} = 2, 4, 6, 20$.

7.2. Cahn-Hilliard equation

As a final test, we consider the Cahn-Hilliard equation

$$w_t = (\underbrace{\Phi'(w) - \varepsilon w_{xx}}_{:=\mu})_{xx}, \quad (x,t) \in (0,1) \times (0, T_{\text{end}} = 0.5), \tag{21}$$

where the chemical potential $\Phi(w)$ is defined as $\Phi(w) = \frac{1}{4}(w^2-1)^2$. ε is a positive constant describing the thickness of the interface of the phase field; it is set to $\varepsilon = 0.001$ in the sequel. For more information, consult [23]. Boundary and initial conditions are set as in [11],

$$\begin{aligned} 100w(x,0) &= 10\sin(2\pi x) + \cos(4\pi x) + 6\sin(4\pi x) + 2\cos(10\pi x), \\ w_x(x,t) &= \mu_x(x,t) = 0, \quad (x,t) \in \{0,1\} \times (0, T_{\text{end}}). \end{aligned}$$

From a numerical point of view, this equation is quite challenging due to the occurence of a fourth-order derivative in combination with the nonlinearity of the chemical potential. It is worth noting that explicit schemes need a restriction of the timestep, $\Delta t = \mathcal{O}(\Delta x^4)$ [16]. To obtain a suitable splitting, we use the convex-concave splitting introduced by [14], i.e., $\Phi'(w)$ is split into

$$\Phi'(w) = w^3 - w,$$

where w^3 is treated implicitly, and w is treated explicitly. The double-diffusion part εw_{xxxx} is also treated implicitly. Spatial discretization is through a straightforward second-order finite difference approach.

Still, our interest is on temporal convergence, which is why we fix the spatial resolution. We report on the results for $N_x = 50$ with associated mesh spacing $\Delta x = \frac{1}{50}$. The coarsest computation uses four subintervals, i.e., $\Delta t = \frac{1}{8}$. The finest computation uses 2^9 spatial subintervals. Error is computed as in Sec. 7.1. Newton tolerance is set to 10^{-7} with a maximum of 1000 iterations; the exact solution is computed through Matlab's ode15s routine with very fine precision.

Fig. 6 shows the corresponding convergence plots. This testcase is way more challenging than the one before, which is also expressed in the convergence results. It is way more difficult to obtain the appropriate order; only at rather fine Δt, the actual order can be observed in most of the combinations. In any cases, no stability issues were observed.

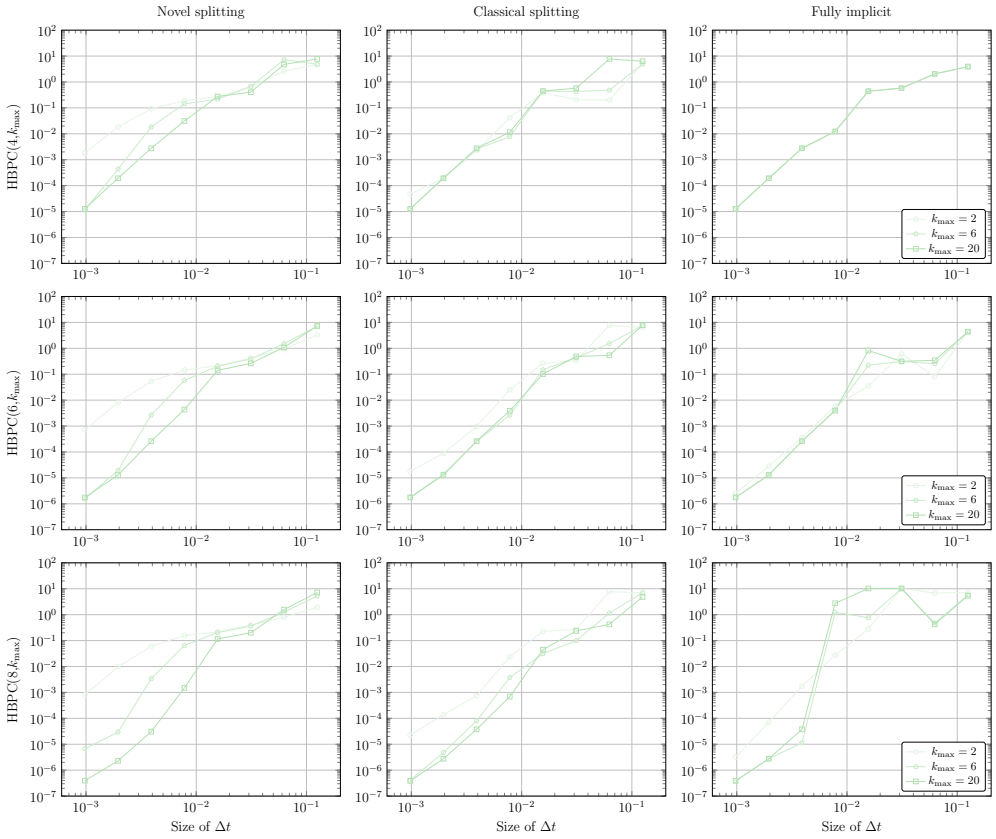

Figure 6: Numerical results for Cahn-Hilliard equation (21). Plotted is the error $e_{\Delta t}$ versus Δt for different values for the $k_{\max}$. ε is set to 10^{-3}; the amount of space nodes is $N_x = 50$. The first row shows numerical results for the method HBPC(4, $k_{\max}$), the second for the method HBPC(6, $k_{\max}$) and the third for the method HBPC(8, $k_{\max}$). In the first column, we present numerical results for the fully implicit scheme, in the second for the splitting from [33], and the third column is the novel splitting. The legend in the right plot is valid for all plots.

8. Conclusion and Outlook

In this work, we have presented a novel way of defining the implicit second temporal derivative of a flux-split IMEX scheme. This has been done in such a way that the non-stiff (explicit!) contribution never has to be inverted and remains explicit throughout the whole procedure. This has not been done before; it will now open many possibilities for future work. Current research focuses on the extension of this to the low-Mach Navier-Stokes equations that themselves form a singularly perturbed problem. There have been many splitting attempts in the past where the stiff splitting is rather easy, typically linear, it is now feasible to also use them in the context of a second derivative scheme. As an example, consider the isentropic Euler equations at low Mach number [24],

$$\begin{aligned} \rho_t + \nabla \cdot (\rho \mathbf{u}) &= 0, \\ (\rho \mathbf{u})_t + \nabla \cdot \left(\rho \mathbf{u} \otimes \mathbf{u} + \frac{1}{\varepsilon^2} p(\rho) \, \mathrm{Id} \right) &= 0. \end{aligned}$$

Here, ρ, $\mathbf{u}$ and p denote density, velocity and pressure of an isentropic fluid, where pressure is a function of density. In [17], a splitting of this equation into implicit ('stiff') and explicit ('non-stiff') parts has been proposed as

$$\begin{aligned} \rho_t + \nabla \cdot (\alpha \rho \mathbf{u}) &+ \nabla \cdot ((1-\alpha)\rho \mathbf{u}) = 0, \\ (\rho \mathbf{u})_t + \nabla \cdot \left(\rho \mathbf{u}^2 + \frac{p(\rho) - a(t)\rho}{\varepsilon^2} \, \mathrm{Id} \right) &+ \nabla \cdot \left(\frac{a(t)\rho}{\varepsilon^2} \, \mathrm{Id} \right) = 0, \end{aligned}$$

where the second divergence terms of each equation denote the implicit part. It is straightforward to see that these implicit terms are linear in the unknowns ρ and $\rho\mathbf{u}$. For a concise definition of α and $a(t)$, we refer to [17]. Currently, we analyze the algorithm presented in this work together with a high-order spatial discretization to solve splitted low-Mach Euler equations. First results show that also here, the method is asymptotically preserving. Furthermore, the fact that Alg. 1 preserves the linearity of the implicit splitting will result in a huge computational saving.

An important ingredient to the algorithm is the background scheme. We have seen that for higher orders of consistency, the HBPC scheme with uniform collocation points as used in this work does not excel at high degrees of stiffness. Currently, we are experimenting with Gaussian-type collocation points to improve upon this.

Some other open questions are the extension to higher temporal derivatives and to other second- or multi-derivative IMEX schemes. We envision that the approach pursued here can be extended by the means of a Taylor expansion in the explicit part, the behavior and stability have to be investigated both numerically and analytically.

Acknowledgments

E. Theodosiou was funded by the Fonds voor Wetenschappelijk Onderzoek (FWO, Belgium) - project no. G052419N

9. Appendix: Butcher tableaux

For convenience, we list the Butcher tableaux that we use in this work here, see also [34, Example 1] and the references therein.

- $q = 4$:

$$B^{(1)} = \begin{pmatrix} 0 & 0 \\ \frac{1}{2} & \frac{1}{2} \end{pmatrix}, \quad B^{(2)} = \begin{pmatrix} 0 & 0 \\ \frac{1}{12} & \frac{-1}{12} \end{pmatrix}. \tag{22}$$

- $q = 6$:

$$B^{(1)} = \begin{pmatrix} 0 & 0 & 0 \\ \frac{101}{480} & \frac{8}{30} & \frac{55}{2400} \\ \frac{7}{30} & \frac{16}{30} & \frac{7}{30} \end{pmatrix}, \quad B^{(2)} = \begin{pmatrix} 0 & 0 & 0 \\ \frac{65}{4800} & -\frac{25}{600} & -\frac{25}{8000} \\ \frac{5}{300} & 0 & -\frac{5}{300} \end{pmatrix}. \tag{23}$$

- $q = 8$:

$$\begin{aligned} B^{(1)} &= \begin{pmatrix} 0 & 0 & 0 & 0 \\ \frac{6893}{54432} & \frac{313}{2016} & \frac{89}{2016} & \frac{397}{54432} \\ \frac{223}{1701} & \frac{20}{63} & \frac{13}{63} & \frac{20}{1701} \\ \frac{31}{224} & \frac{81}{224} & \frac{81}{224} & \frac{31}{224} \end{pmatrix}, \\ B^{(2)} &= \begin{pmatrix} 0 & 0 & 0 & 0 \\ \frac{1283}{272160} & -\frac{851}{30240} & -\frac{269}{30240} & -\frac{163}{272160} \\ \frac{43}{8505} & -\frac{16}{945} & -\frac{19}{945} & -\frac{8}{8505} \\ \frac{19}{3360} & -\frac{9}{1120} & \frac{9}{1120} & -\frac{19}{3360} \end{pmatrix}. \end{aligned} \tag{24}$$

References

[1] A. Abdi and G. Hojjati. An extension of general linear methods. *Numerical Algorithms*, 57(2):149–167, 2011.

[2] U. M. Ascher, S. Ruuth, and R. Spiteri. Implicit-explicit Runge-Kutta methods for time-dependent partial differential equations. *Applied Numerical Mathematics*, 25:151–167, 1997.

[3] S. Boscarino. Error analysis of IMEX Runge-Kutta methods derived from differential-algebraic systems. *SIAM Journal on Numerical Analysis*, 45:1600–1621, 2007.

[4] S. Boscarino. On an accurate third order implicit-explicit Runge-Kutta method for stiff problems. *Applied Numerical Mathematics*, 59:1515–1528, 2009.

[5] S. Boscarino and L. Pareschi. On the asymptotic properties of IMEX Runge–Kutta schemes for hyperbolic balance laws. *Journal of Computational and Applied Mathematics*, 316:60 – 73, 2017.

[6] S. Boscarino, L. Pareschi, and G. Russo. Implicit-explicit Runge–Kutta schemes for hyperbolic systems and kinetic equations in the diffusion limit. *SIAM Journal on Scientific Computing*, 35(1):A22–A51, 2013.

[7] S. Boscarino, J.-M. Qiu, G. Russo, and T. Xiong. A high order semi-implicit IMEX WENO scheme for the all-Mach isentropic Euler system. *Journal of Computational Physics*, 392:594–618, 2019.

[8] J. C. Butcher. General linear methods. *Acta Numerica*, 15:157–256, 2006.

[9] R. Chan and A. Tsai. On explicit two-derivative Runge-Kutta methods. *Numerical Algorithms*, 53:171–194, 2010.

[10] F. Cordier, P. Degond, and A. Kumbaro. An asymptotic-preserving all-speed scheme for the Euler and Navier-Stokes equations. *Journal of Computational Physics*, 231:5685–5704, 2012.

[11] E. V. L. de Mello and O. T. da Silveira Filho. Numerical study of the Cahn-Hilliard equation in one, two and three dimensions. *Physica A. Statistical Mechanics and its Applications*, 347(1-4):429–443, 2005.

[12] P. Degond and M. Tang. All speed scheme for the low Mach number limit of the isentropic Euler equation. *Communications in Computational Physics*, 10:1–31, 2011.

[13] A. Dittmann. High-order multiderivative IMEX schemes. *Applied Numerical Mathematics*, 160:205 – 216, 2021.

[14] D. Eyre. Unconditionally gradient stable time marching the Cahn-Hilliard equation. *MRS Proceedings*, 529:39, 1998.

[15] S. Gottlieb, Z. Grant, J. Hu, and R. Shu. High order strong stability preserving multiderivative implicit and IMEX Runge–Kutta methods with asymptotic preserving properties. *SIAM Journal on Numerical Analysis*, 60(1):423–449, 2022.

[16] R. Guo and Y. Xu. Efficient solvers of discontinuous Galerkin discretization for the Cahn–Hilliard equations. *Journal of Scientific Computing*, 58(2):380–408, jun 2013.

[17] J. Haack, S. Jin, and J.-G. Liu. An all-speed asymptotic-preserving method for the isentropic Euler and Navier-Stokes equations. *Communications in Computational Physics*, 12:955–980, 2012.

[18] E. Hairer, S. P. Nørsett, and G. Wanner. *Solving ordinary differential equations I.* Springer Series in Computational Mathematics, 1987.

[19] E. Hairer and G. Wanner. Multistep-multistage-multiderivative methods for ordinary differential equations. *Computing (Arch. Elektron. Rechnen)*, 11(3):287–303, 1973.

[20] E. Hairer and G. Wanner. *Solving ordinary differential equations II.* Springer Series in Computational Mathematics, 1991.

[21] E. Hopf. The partial differential equation $u_t + uu_x = \mu u_{xx}$. *Communications on Pure and Applied Mathematics*, 3:201–230, 1950.

[22] P. Kaps. Rosenbrock-type methods. In G. Dahlquist and R. Jeltsch, editors, *Oberwolfach 28.6.–4.7.1981*, Bericht Nr. 9. Institut für Geometrie und Praktische Mathematik, RWTH Aachen, 1981.

[23] J. Kim. Phase-field models for multi-component fluid flows. *Communications in Computational Physics*, 12(3):613–661, 2012.

[24] S. Klainerman and A. Majda. Singular limits of quasilinear hyperbolic systems with large parameters and the incompressible limit of compressible fluids. *Communications on Pure and Applied Mathematics*, 34:481–524, 1981.

[25] V. Kučera, M. Lukáčová-Medvid'ová, S. Noelle, and J. Schütz. Asymptotic properties of a class of linearly implicit schemes for weakly compressible Euler equations. *Numerische Mathematik*, 150:79–103, 2021.

[26] H. Liu and J. Zou. Some new additive Runge–Kutta methods and their applications. *Journal of Computational and Applied Mathematics*, 190(1-2):74–98, 2006.

[27] A. Moradi, A. Abdi, and J. Farzi. Strong stability preserving second derivative general linear methods with Runge-Kutta stability. *Journal of Scientific Computing*, 85(1):Paper No. 1, 39, 2020.

[28] A. Moradi, A. Abdi, and G. Hojjati. Implicit-explicit second derivative general linear methods with strong stability preserving explicit part. *Applied Numerical Mathematics*, 181:23–45, 2022.

[29] S. Noelle, G. Bispen, K.R. Arun, M. Lukáčová-Medvid'ová, and C.-D. Munz. A weakly asymptotic preserving low Mach number scheme for the Euler equations of gas dynamics. *SIAM Journal on Scientific Computing*, 36:B989–B1024, 2014.

[30] L. Pareschi and G. Russo. Implicit-explicit Runge-Kutta schemes for stiff systems of differential equations. *Recent Trends in Numerical Analysis*, 3:269–289, 2000.

[31] L. Pareschi and G. Russo. Implicit–explicit Runge–Kutta schemes and applications to hyperbolic systems with relaxation. *Journal of Scientific computing*, 25:129–155, 2005.

[32] J. Schütz and S. Noelle. Flux splitting for stiff equations: A notion on stability. *Journal of Scientific Computing*, 64(2):522–540, 2015.

[33] J. Schütz and D. Seal. An asymptotic preserving semi-implicit multiderivative solver. *Applied Numerical Mathematics*, 160:84–101, 2021.

[34] J. Schütz, D. C. Seal, and J. Zeifang. Parallel-in-time high-order multiderivative IMEX solvers. *Journal of Scientific Computing*, 90(54):1–33, 2022.

[35] D. C. Seal, Y. Güçlü, and A. Christlieb. High-order multiderivative time integrators for hyperbolic conservation laws. *Journal of Scientific Computing*, 60:101–140, 2014.

[36] J. Zeifang and J. Schütz. Implicit two-derivative deferred correction time discretization for the discontinuous Galerkin method. *Journal of Computational Physics*, 464:111353, 2022.

[37] J. Zeifang, J. Schütz, and D. Seal. Stability of implicit multiderivative deferred correction methods. *BIT Numerical Mathematics*, 2022.

[38] 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*, 457:128198, 2023.

UHasselt Computational Mathematics Preprint Series

www.uhasselt.be/cmat