On improving the efficiency of ADER methods

On improving the efficiency of ADER methods

Maria Han Veiga[a], Lorenzo Micalizzi[b], Davide Torlo[c,*]

[a]*Department of Mathematics, Ohio State University, 231 W 18th Ave, Columbus, OH 43210, United States of America*
[b]*Institute of Mathematics, University of Zurich, Winterthurerstrasse 190, Zurich, 8057, Switzerland*
[c]*SISSA Mathlab, Mathematics Area, SISSA, via Bonomea 265, 34136, Trieste, Italy*

Abstract

The (modern) arbitrary derivative (ADER) approach is a popular technique for the numerical solution of differential problems based on iteratively solving an implicit discretization of their weak formulation. In this work, focusing on an ODE context, we investigate several strategies to improve this approach. Our initial emphasis is on the order of accuracy of the method in connection with the polynomial discretization of the weak formulation. We demonstrate that precise choices lead to higher-order convergences in comparison to the existing literature. Then, we put ADER methods into a Deferred Correction (DeC) formalism. This allows to determine the optimal number of iterations, which is equal to the formal order of accuracy of the method, and to introduce efficient p-adaptive modifications. These are defined by matching the order of accuracy achieved and the degree of the polynomial reconstruction at each iteration. We provide analytical and numerical results, including the stability analysis of the new modified methods, the investigation of the computational efficiency, an application to adaptivity and an application to hyperbolic PDEs with a Spectral Difference (SD) space discretization.

Keywords: ADER, time-stepping, arbitrarily high order methods, adaptivity, efficient iterative method, deferred correction

1. Introduction

Differential problems play a crucial role in the world of modeling of natural and technological processes. In fact, countless systems in many applications are described through ODEs and PDEs. Due to the variety of existing phenomena to be modeled, no unified theory exists and very different system-dependent differential problems are reported in literature. Despite such heterogeneity, the vast majority of the mentioned mathematical models share a common feature: the impossibility to find analytical solutions, apart from very basic exceptions. At the moment, the only possible way to quantitatively cope with such models is to rely on numerical methods. Nevertheless, numerical approaches come with their own set of challenges. One of the biggest issues in such context is given by the computational cost associated to very accurate approximations of the sought analytical solutions, which require very refined discretizations, with consequent need for huge computational resources, in terms of computational time and memory consumption.

A classical expedient to reduce the computational cost, making hence the simulations more accessible, consists in the adoption of high order methods. In fact, such methods are well-known to require, at least on smooth problems, less computational resources for a fixed discretization error. This justifies the large interest, lately shown by the scientific community, in the construction of (arbitrary) high order frameworks for the numerical solution of differential problems, both for ODEs, with Runge-Kutta (RK) methods [14, 38, 66], predictor-corrector methods [35] and multistep methods [38], and PDEs, with finite difference methods [45, 36], finite volume methods [46, 28, 65, 36], finite element methods [27, 40, 17, 37], SD methods [43, 63, 64], Residual Distribution methods [3, 1, 2, 4] and ADER methods [6, 31, 9].

Compared to the other frameworks, ADER is relatively new. In fact, originally introduced by Titarev and Toro [61, 60] in 2001 for hyperbolic problems, as a time integrator for finite volume formulations, based on the Cauchy–Kovalevskaya theorem, it became popular in its modern formulation presented in 2008 [20]. The method consists in constructing nonlinear systems of algebraic equations, associated to high order discretizations of the weak

*Corresponding author
Email addresses: hanveiga.1@osu.edu (Maria Han Veiga), lorenzo.micalizzi@math.uzh.ch (Lorenzo Micalizzi), davide.torlo@sissa.it (Davide Torlo)

formulation of the analytical problem under investigation, which are solved through an iterative procedure. Despite its recent definition, the approach has been proven to be robust and reliable through numerous applications to many different fields such as ODEs contexts [39], Eulerian-Lagrangian formulations on unstructured moving meshes [8, 11, 10, 7, 32], structure-preserving [34, 48], magnetohydrodynamics [63, 6, 33, 25, 68], solid mechanics [12, 24], compressible fluid-dynamics [21, 9, 59, 64, 19, 58, 41], aeroacustics [23, 55, 56], adaptivity [48, 68], with parallel implementations used in the context of large scale simulations for real test cases [22, 54]. The mentioned literature does not pretend to be complete, however, it does indeed provide a clear proof of the level of maturity, of flexibility and of robustness of the ADER approach.

The goal of this paper is to introduce several strategies to drastically reduce the computational cost of ADER methods because, even if they have shown good performances in the context of large scale simulations, still they leave the door open for great improvements under many points of view, in particular at the level of the computational efficiency. More in detail, in an ODE context, we discuss the following main contributions:

- We start by investigating the role of the polynomial reconstruction of the numerical solution adopted in the context of the discretization of the weak formulation of the differential problem, leading to the definition of the ADER nonlinear system, later solved iteratively. In particular, we prove that careful choices on bases and quadrature points lead to schemes with order of accuracy higher than expected according to classical literature. This is done by reinterpreting the ADER nonlinear system as an implicit high order RK, which is analyzed in depth. The computational advantage achieved in this context is far from being negligible. In fact, usually in the context of ADER methods, order P is achieved by selecting polynomial discretizations of degree $P - 1$ in time [39, 63, 64, 67, 29, 53, 22]. We show that the same accuracy can be obtained with a polynomial degree which is approximately $\frac{P}{2}$, with related saving of computational resources both in terms of memory and time. A further result, in this context, concerns the link between ADER methods and Lobatto IIIC RK methods [66].

- We characterize the ADER methods obtained for general polynomial bases. As ADER methods do not require particular constraints with respect to the basis functions adopted for the discretization of the weak formulation of the differential problem, one could wonder whether the adoption of particular bases (e.g., modal bases) could lead to better schemes. We show that there is a strong link between ADER methods obtained with arbitrary bases and ADER methods with nodal bases, which are proven to be equivalent under the assumption that the adopted quadratures coincide.

- By using the fact that ADER methods can be put in a DeC framework [1, 47, 48], we exploit the DeC formulation of ADER methods, to construct efficient modifications of such schemes, following the idea introduced in [47] and generalized in [48]. The new modified schemes are based on increasing the degree of the polynomial reconstruction of the numerical solution along the iterative procedure in such a way that the achieved accuracy matches the discretization accuracy of the ADER iterations. Further, we show how to recast the new schemes as explicit RK methods, by defining the related Butcher tableaux, and we study their linear stability, proving that the efficient modifications do not affect stability.

- Lastly, we introduce a natural way to perform p-adaptivity. This aspect is particularly interesting, in fact, if on the one hand we have remarked how the adoption of high order methods results in a considerable computational advantage for a fixed discretization error, on the other hand one must notice that the error is not known a priori. In principle, users are not strictly interested in the order of the method adopted, but rather on the final error being smaller than a given tolerance. Therefore, in practical applications, high order methods should be used in combination with adaptive strategies able to estimate the error and to select the discretization and/or the order accordingly. Unfortunately, such adaptive strategies are in general not easy to design but the particular structure of the proposed modified ADER methods offers a natural way to do it.

A rich variety of numerical tests is provided to show the computational advantages of the proposed modifications, with large registered speed-ups.

The structure of this work is the following: we introduce the ADER methods for ODEs in Section 2. In Section 3, we study the accuracy of such methods with respect to the discretization of the weak formulation, showing that the adoption of Gauss–Lobatto (GLB) and Gauss–Legendre (GLG) bases leads to accuracy higher than expected in Section 3.1 and characterizing the ADER methods obtained with general polynomial bases in Section 3.2. In Section 4, we introduce the DeC framework and show how the ADER methods can be reinterpreted as DeC methods, hence fixing the optimal number of iterations. We present thus efficient modifications of ADER methods in Section 5 and their application to p-adaptivity. Then, in Section 6, we show how the modified methods

can be written in RK form, defining their Butcher tableaux, and we study their linear stability. In Section 7, we describe an application of the ADER methods to hyperbolic PDEs with an SD space discretization via the method of lines. The methods are validated in Section 8 and, finally, Section 9 is left for conclusions and further developments.

Moreover, Appendix A is dedicated to the proofs of several "minor" results presented along the paper. To maintain the flow of information smooth, we have placed these proofs in an appendix, so that readers can focus on the main concepts and conclusions, referring to the proofs at their convenience. In Appendix B, the proof of an equivalence theorem between particular ADER methods and Lobatto IIIC schemes is reported, while, an overview of all symbols used in the paper can be found in Appendix D.

2. ADER

In this section, we will present the original ADER method [20] in its simplified version for systems of ODEs, firstly described in [39]. Let us consider the following system

$$\begin{cases} \frac{d}{dt}\boldsymbol{u}(t) = \boldsymbol{G}(t, \boldsymbol{u}(t)), \quad t \in [0, T], \\ \boldsymbol{u}(0) = \boldsymbol{z}, \end{cases} \tag{1}$$

where $\boldsymbol{u} : \mathbb{R}_0^+ \to \mathbb{R}^Q$ is the sought solution, $\boldsymbol{z} \in \mathbb{R}^Q$ the initial condition and $\boldsymbol{G} : \mathbb{R}_0^+ \times \mathbb{R}^Q \to \mathbb{R}^Q$ a function that is continuous and Lipschitz-continuous with respect to $\boldsymbol{u}$ uniformly with respect to t with a Lipschitz constant C_{Lip}. As usual in the context of one-step methods, we focus on a generic time interval $[t_n, t_{n+1}]$, with $t_{n+1} = t_n + \Delta t$, and we look for a recipe for $\boldsymbol{u}_{n+1} \approx \boldsymbol{u}(t_{n+1})$ from $\boldsymbol{u}_n \approx \boldsymbol{u}(t_n)$. In particular, as commonly done in the context of consistency analyses, we assume $\boldsymbol{u}_n = \boldsymbol{u}(t_n)$.

2.1. Method

We consider the weak formulation of (1) over $[t_n, t_{n+1}]$, obtained by multiplying the equation by a smooth test function $\psi(t)$ and applying integration by parts on the time derivative term

$$\psi(t_{n+1})\boldsymbol{u}(t_{n+1}) - \psi(t_n)\boldsymbol{u}_n - \int_{t_n}^{t_{n+1}} \left(\frac{d}{dt}\psi(t)\right)\boldsymbol{u}(t)dt - \int_{t_n}^{t_{n+1}} \psi(t)\boldsymbol{G}(t, \boldsymbol{u}(t))dt = \boldsymbol{0}, \tag{2}$$

where we remark that $\boldsymbol{u}_n = \boldsymbol{u}(t_n)$ is known. We introduce now $M + 1$ subtimenodes t^m for $m = 0, 1, \ldots, M$ in the interval $[t_n, t_{n+1}]$ such that

$$t_n \le t^0 < t^1 < \cdots < t^M \le t_{n+1} \tag{3}$$

and we denote by $\boldsymbol{u}^m$ an approximation of the exact solution $\boldsymbol{u}(t)$ in t^m. We pass to a discrete setting by projecting (2) onto a finite dimensional functional space. In particular, we replace $\boldsymbol{u}(t)$ and $\boldsymbol{G}(t, \boldsymbol{u}(t))$ by their Lagrange interpolating polynomials of degree M associated to the $M + 1$ subtimenodes

$$\boldsymbol{u}_h(t) := \sum_{m=0}^{M} \boldsymbol{u}^m \psi^m(t), \qquad \boldsymbol{G}_h(t) := \sum_{m=0}^{M} \boldsymbol{G}(t^m, \boldsymbol{u}^m)\psi^m(t), \tag{4}$$

leading to the ADER implicit weak form (ADER-IWF)

$$\begin{aligned} \sum_{m=0}^{M} \left[\psi^\ell(t_{n+1})\psi^m(t_{n+1}) - \int_{t_n}^{t_{n+1}} \left(\frac{d}{dt}\psi^\ell(t)\right)\psi^m(t)dt\right]\boldsymbol{u}^m - \psi^\ell(t_n)\boldsymbol{u}_n \\ - \sum_{m=0}^{M} \left(\int_{t_n}^{t_{n+1}} \psi^\ell(t)\psi^m(t)dt\right)\boldsymbol{G}(t^m, \boldsymbol{u}^m) = \boldsymbol{0}, \quad \forall \ell = 0, \ldots, M, \end{aligned} \tag{5}$$

which is a nonlinear system in the unknowns $\boldsymbol{u}^m$.

As we are going to see in the next subsection, such system can be put in a compact matrix formulation and, under time step restrictions, it can be solved through an explicit iterative strategy. Its solution consists of the coefficients of the continuous representation (4) of the numerical solution $\boldsymbol{u}_h$, through which we get $\boldsymbol{u}_{n+1} := \boldsymbol{u}_h(t_{n+1})$. The order of accuracy of $\boldsymbol{u}_{n+1}$ with respect to the exact solution $\boldsymbol{u}(t_{n+1})$, denoted by N, depends on the choice of the subtimenodes and on the quadrature rule used for integrals. Summarizing, the ADER method consists in solving iteratively the ADER-IWF (5) with respect to the coefficients $\boldsymbol{u}^m$, which are later used to compute $\boldsymbol{u}_{n+1}$.

Remark 2.1 (On the order of accuracy)**.** *Several choices of subtimenodes and quadrature rules are possible. A generic distribution of $M+1$ subtimenodes t^m in the interval $[t_n, t_{n+1}]$, e.g., equispaced, guarantees, in general, an order of accuracy of the resulting ADER method equal to $N = M + 1$. However, particular choices yield higher accuracy, for example if we choose, both for the basis function definitions and for the quadrature rule, to use $M + 1$ GLB subtimenodes we obtain order of accuracy equal to $N = 2M$, while with $M + 1$ GLG nodes we get an accuracy of $N = 2M + 1$. The proofs of the accuracy are based on theoretical results in the context of the RK methods and are presented in Section 3, in Theorems 3.8 and 3.9.*

Remark 2.2 (On the computation of $\boldsymbol{u}_{n+1}$)**.** *The ADER-IWF* (5)*, together with the reconstruction* (4) *of $\boldsymbol{u}_h$ in t_{n+1}, could be interpreted as an implicit high order RK, referred as ADER-IWF-RK, as we are going to show in Theorem 3.1. In particular, in the context of the computation of $\boldsymbol{u}_{n+1}$ after the solution of the nonlinear system, the final interpolation could be equivalently replaced by the following integration*

$$\boldsymbol{u}_{n+1} = \boldsymbol{u}_n + \int_{t_n}^{t_{n+1}} \boldsymbol{G}_h(t)dt = \boldsymbol{u}_n + \sum_{m=0}^{M} \left(\int_{t_n}^{t_{n+1}} \psi^m(t)dt \right) \boldsymbol{G}(t^m, \boldsymbol{u}^m). \tag{6}$$

In the ADER formulation for hyperbolic PDEs introduced in [20], the final integration allows communications between neighboring cells after an iterative procedure used to solve local weak formulations in space-time control volumes.

Remark 2.3 (On the difference between ADER and ADER-IWF-RK and non-suitability for stiff problems)**.** *The ADER method is based on an explicit iterative procedure for the solution of the nonlinear system* (5) *and it has, hence, an explicit character, not suited for stiff problems. On the other hand, the ADER-IWF-RK is an implicit method. Therefore, an ADER method and its associated ADER-IWF-RK method do not share the same properties in terms of stability.*

Let us notice that implicit versions of ADER, based on an implicit treatment of $\boldsymbol{G}$ in the context of the iterative procedure and, hence, suitable for stiff problems, exist in literature [39] but they will not be investigated in this work. The application of the efficient modifications proposed in this work to implicit ADER methods is currently under investigation.

2.2. Matrix formulation and explicit iterative solution

The ADER-IWF (5) can be recast in the following matrix formulation

$$B\underline{\boldsymbol{u}} - \underline{\boldsymbol{r}} - \Delta t \Lambda \underline{\boldsymbol{G}}(\underline{\boldsymbol{u}}) = \boldsymbol{0}, \tag{7}$$

where B and Λ are matrices, while, $\underline{\boldsymbol{u}}$, $\underline{\boldsymbol{r}}$ and $\underline{\boldsymbol{G}}(\underline{\boldsymbol{u}})$ are vectors, defined by

$$\begin{aligned}
B_{\ell,m} &:= \psi^\ell(t_{n+1})\psi^m(t_{n+1}) - \int_{t_n}^{t_{n+1}} \left(\frac{d}{dt}\psi^\ell(t) \right) \psi^m(t)dt = \widehat{\psi}^\ell(1)\widehat{\psi}^m(1) - \int_0^1 \left(\frac{d}{d\xi}\widehat{\psi}^\ell(\xi) \right) \widehat{\psi}^m(\xi)d\xi, \\
\Lambda_{\ell,m} &:= \frac{1}{\Delta t} \int_{t_n}^{t_{n+1}} \psi^\ell(t)\psi^m(t)dt = \int_0^1 \widehat{\psi}^\ell(\xi)\widehat{\psi}^m(\xi)d\xi \\
\underline{\boldsymbol{u}} &:= \begin{pmatrix} \boldsymbol{u}^0 \\ \vdots \\ \boldsymbol{u}^M \end{pmatrix}, \quad \underline{\boldsymbol{r}} := \begin{pmatrix} \psi^0(t_n)\boldsymbol{u}_n \\ \vdots \\ \psi^M(t_n)\boldsymbol{u}_n \end{pmatrix}, \quad \underline{\boldsymbol{G}}(\underline{\boldsymbol{u}}) := \begin{pmatrix} \boldsymbol{G}(t^0, \boldsymbol{u}^0) \\ \vdots \\ \boldsymbol{G}(t^M, \boldsymbol{u}^M) \end{pmatrix}
\end{aligned} \tag{8}$$

with $\widehat{\psi}^m(\xi) := \psi^m(t_n + \Delta t\xi)$ being the Lagrange basis functions remapped onto the interval $[0, 1]$. For simplicity, the matrices B and Λ were defined for a scalar problem, they need to be block expanded for a vectorial problem. Furthermore, inverting the matrix B, from (7) we get

$$\underline{\boldsymbol{u}} - \underline{\boldsymbol{u}}_n - \Delta t B^{-1} \Lambda \underline{\boldsymbol{G}}(\underline{\boldsymbol{u}}) = \boldsymbol{0}, \quad \text{with} \quad \underline{\boldsymbol{u}}_n := \begin{pmatrix} \boldsymbol{u}_n \\ \vdots \\ \boldsymbol{u}_n \end{pmatrix}, \tag{9}$$

thanks to the following proposition, whose proof can be found in Appendix A.1.

Proposition 2.4. *If B and $\boldsymbol{r}$ are defined as in* (8) *and $\underline{\boldsymbol{u}}_n$ as in* (9)*, then $B^{-1}\boldsymbol{r} = \underline{\boldsymbol{u}}_n$.*

For Δt small enough, system (9) has a unique solution, which can be obtained as the limit of the following explicit iterative procedure

$$\underline{\boldsymbol{u}}^{(p)} := \underline{\boldsymbol{u}}_n + \Delta t B^{-1} \Lambda \underline{\boldsymbol{G}}(\underline{\boldsymbol{u}}^{(p-1)}), \quad p \geq 1, \tag{10}$$

which converges independently of the chosen initial guess $\underline{\boldsymbol{u}}^{(0)}$, due to the following proposition, whose proof can be found in Appendix A.2.

Proposition 2.5 (Convergence of the iterative procedure)**.** *For Δt small enough, a unique solution to* (9) *exists and it coincides with the limit of the iterative procedure* (10)*. In particular, the convergence is ensured for $\Delta t < \frac{1}{\|B^{-1}\Lambda\|_\infty C_{Lip}}$.*

The time step restriction resulting from Proposition 2.5 amounts to a classical time step restriction for explicit methods. In fact, the step size Δt is constrained by the inverse of the Lipschitz constant C_{Lip} of $\boldsymbol{G}$ up to a constant, $\left\|B^{-1}\Lambda\right\|_\infty$, independent of Δt. Let us notice that such estimate for the upper bound of Δt might not be optimal and could be improved, for example, by choosing other norms in the context of the proof.

Let us notice that, since the expected discretization accuracy of $\boldsymbol{u}_{n+1} := \boldsymbol{u}_h(t_{n+1})$ is N, there is no need to solve the nonlinear system (9) up to a tolerance that is stricter than the discretization error corresponding to such accuracy. In general, obtaining an N-th order accurate approximation of the solution of (9) is sufficient and this is possible by performing exactly N iterations of (10), as we are going to see in Section 4, by putting the ADER method in a DeC formalism and showing that each iteration increases the order of accuracy by one.

Before continuing, we state here a result concerning the invertibility of the matrix B.

Theorem 2.6 (Invertibility of B)**.** *The ADER matrix B, defined for any generic basis $\left\{\widehat{\phi}^m(\xi)\right\}_{m=0,\dots,M}$ of the space of the polynomials of degree M over $[0,1]$ as*

$$B_{\ell,m} := \widehat{\phi}^\ell(1)\widehat{\phi}^m(1) - \int_0^1 \left(\frac{d}{d\xi}\widehat{\phi}^\ell(\xi)\right)\widehat{\phi}^m(\xi)d\xi, \tag{11}$$

is invertible.

The construction of ADER methods is strongly based on the assumption that the matrix B is nonsingular, however, the problem of the existence of its inverse has been, up to authors' knowledge, never investigated in literature. In fact, the proof of Theorem 2.6 is less immediate than the proofs of the invertibility of classical mass and stiffness matrices of standard finite element formulations and can be found in Appendix A.3.

3. Accuracy of ADER-IWF

This section is divided into two subsections. In the first one, we will discuss the order of accuracy of ADER methods for equispaced, GLB and GLG subtimenodes. In the second one, we will present an original result concerning ADER methods with arbitrary bases.

Both subsections address important aspects in the context of the computational efficiency of ADER methods. In fact, in several works [39, 63, 64, 67, 29, 53, 22], $(M+1)$-th order of accuracy has been obtained via $M+1$ GLG subtimenodes leading to a polynomial reconstruction in time of degree M; however, as we are going to show, nearly half of the subtimenodes (with consequent shorter computational times and smaller memory consumption) can guarantee the same accuracy order. In particular, in Theorem 3.8 and Theorem 3.9, we show that $M+1$ subtimenodes correspond to an accuracy order equal to $2M$ and $2M+1$, respectively for GLB and GLG subtimenodes if the associated quadrature formulas are adopted to compute the integrals.

Moreover, we show in Theorem 3.10 that any general set of polynomial basis functions (not necessarily Lagrangian) lead to the same schemes defined above, under mild assumptions.

The quadrature formulas used to compute the integrals in the ADER terms will play a crucial role in this section. Whenever not specified, we assume an exact integration.

3.1. Accuracy of ADER with nodal bases

In this subsection, we will show that the order of accuracy of the ADER methods with $M+1$ subtimenodes is $N = M+1$ for equispaced subtimenodes with exact quadrature and, respectively, $N = 2M$ and $N = 2M+1$ for GLB and GLG subtimenodes, when adopting the associated quadrature formulas. We remark that the accuracy of an ADER method is referred to the approximation $\boldsymbol{u}_{n+1}$, obtained via the reconstruction (4), after the solution of the ADER-IWF (5). In the following, we will show how it is possible to associate an implicit high order RK scheme with $S = M+1$ stages to a general ADER method.

Let us recall the structure of a RK scheme [13] with S stages

$$\begin{cases} \boldsymbol{y}^s = \boldsymbol{u}_n + \Delta t \sum_{r=0}^{S-1} a_{s,r} \boldsymbol{G}(t_n + c_r \Delta t, \boldsymbol{y}^r), & \text{for } s = 0, \dots, S-1, \\ \boldsymbol{u}_{n+1} = \boldsymbol{u}_n + \Delta t \sum_{r=0}^{S-1} b_r \boldsymbol{G}(t_n + c_r \Delta t, \boldsymbol{y}^r), \end{cases} \tag{12}$$

where $a_{s,r}$, b_r and c_r for $s, r = 0, \dots, S-1$ are some coefficients that characterize the method, usually stored in matrix A and vectors $\boldsymbol{b}$ and $\boldsymbol{c}$ respectively. The associated Butcher tableaux defines the method as well

$$\begin{array}{c|c} \boldsymbol{c} & A \\ \hline & \boldsymbol{b} \end{array}.$$

In the following, we will denote by $\xi^m := \frac{t^m - t_n}{\Delta t}$ the ADER subtimenodes rescaled into $[0, 1]$, and by $w_m := \int_0^1 \widehat{\psi}^m d\xi$ the quadrature weights of the induced quadrature formula. Further, we collect the ξ^m values in the vector $\underline{\beta} := \left(\xi^0, \dots, \xi^M\right)^T$. We can now present the first result of this subsection concerning the link between ADER-$\overline{\text{I}}$WF and implicit RK methods.

Theorem 3.1 (The ADER-IWF (5) is an implicit RK)**.** *Solving the ADER-IWF* (5) *and then using reconstruction* (4) *to get* $\boldsymbol{u}_{n+1}$ *with* $M \geq 1$ *is equivalent to the implicit RK method characterized by* $A := B^{-1}\Lambda$, $\boldsymbol{c} := \underline{\beta}$ *and* $\boldsymbol{b}$ *defined by* $b_m := w_m$.

The proof can be found in Appendix A.4. Given an ADER method, we will refer to the associated RK method as ADER-IWF-RK. It is possible to show that such RK methods fulfill classical properties of high order RK schemes, for example the following proposition holds.

Proposition 3.2. *A general ADER-IWF-RK method with* $M \geq 1$ *satisfies condition*

$$\sum_{j=0}^{S-1} a_{i,j} = c_i, \quad i = 0, \dots, S-1. \tag{13}$$

The proof can be found in Appendix A.5. Further, thanks to basic interpolation properties, it is easy to show that an ADER-IWF-RK method with $M+1$ subtimenodes is at least of order $N = M+1$, as summarized in the next proposition.

Proposition 3.3. *ADER-IWF-RK with* $M+1$ *subtimenodes is at least of order* $M+1$.

The proof can be found in Appendix A.6. Therefore, a general distribution of subtimenodes, e.g., equispaced, yields accuracy $M+1$. Nonetheless, we will show that particular choices of subtimenodes lead to higher convergence rates. The order of accuracy of the ADER-IWF-RK schemes can be verified with the help of some conditions on the coefficients A, $\boldsymbol{b}$ and $\boldsymbol{c}$. Let us, hence, define, for $p, \eta, \zeta \in \mathbb{N}$, the following conditions [66]

$$\mathcal{B}(p): \quad \sum_{i=0}^{S-1} b_i c_i^{z-1} = \frac{1}{z}, \qquad z = 1, \dots, p; \tag{14a}$$

$$C(\eta): \quad \sum_{j=0}^{S-1} a_{i,j} c_j^{z-1} = \frac{c_i^z}{z}, \qquad i = 0, \dots, S-1,\ z = 1, \dots, \eta; \tag{14b}$$

$$\mathcal{D}(\zeta): \quad \sum_{i=0}^{S-1} b_i c_i^{z-1} a_{i,j} = \frac{b_j}{z}(1 - c_j^z), \qquad j = 0, \dots, S-1,\ z = 1, \dots, \zeta, \tag{14c}$$

which allow to easily verify the order of accuracy of implicit RK schemes through the following theorem.

Theorem 3.4 (Butcher 1964 [13])**.** *If the coefficients* $a_{i,j}, b_i, c_i$ *of a RK scheme satisfy* $\mathcal{B}(p)$, $C(\eta)$ *and* $\mathcal{D}(\zeta)$ *with* $p \leq \eta + \zeta + 1$ *and* $p \leq 2\eta + 2$, *then the method is of order* p.

We recall that the condition $\mathcal{B}(p)$ is necessary to reach order p, while $C(\eta)$ and $\mathcal{D}(\zeta)$ are only sufficient conditions [38]. Typically, explicit methods do not fulfill them.

We are now going to prove two preliminary results concerning the ADER-IWF-RK schemes, which will be later used to show their accuracy for GLB and GLG subtimenodes.

Lemma 3.5. *The ADER-IWF-RK methods, with integral terms evaluated through a quadrature formula $Q\{\cdot\}$ with degree of exactness at least $2S-3$, satisfy $C(S-1)$.*

Proof. Condition $C(S-1)$ can be verified by explicit computations in matricial form. Indeed, since $A = B^{-1}\Lambda$, then $C(S-1)$ can be rewritten as

$$A\,\underline{\boldsymbol{c}^{z-1}} = \frac{1}{z}\underline{\boldsymbol{c}^{z}} \Longleftrightarrow \Lambda\,\underline{\boldsymbol{c}^{z-1}} = \frac{1}{z}B\,\underline{\boldsymbol{c}^{z}}, \qquad z = 1,\dots,S-1, \tag{15a}$$

where $\underline{\boldsymbol{c}^{\alpha}}$ denotes the vector $\boldsymbol{c}$ with each entry to the power of α.

Let us compute the general ℓ-th component of both sides of the last equality. The right-hand side is

$$\begin{aligned}\frac{1}{z}\left(B\,\underline{\boldsymbol{c}^{z}}\right)_{\ell} &= \frac{1}{z}\sum_{m=0}^{M}\left[\widehat{\psi}^{\ell}(1)\widehat{\psi}^{m}(1) - \int_0^1\left(\frac{d}{d\xi}\widehat{\psi}^{\ell}(\xi)\right)\widehat{\psi}^{m}(\xi)d\xi\right](\xi^{m})^{z} \\ &= \frac{1}{z}\left\{\widehat{\psi}^{\ell}(1)\left[\sum_{m=0}^{M}\widehat{\psi}^{m}(1)\,(\xi^{m})^{z}\right] - \int_0^1\left(\frac{d}{d\xi}\widehat{\psi}^{\ell}(\xi)\right)\left[\sum_{m=0}^{M}\widehat{\psi}^{m}(\xi)\,(\xi^{m})^{z}\right]d\xi\right\}.\end{aligned} \tag{15b}$$

Since z is at most $S-1$ and the Lagrange basis functions $\widehat{\psi}^{m}$ associated to $M+1 = S$ points allow to exactly interpolate polynomials up to degree $M = S-1$, we have that the terms in square brackets are nothing but the exact interpolation of the function ξ^{z}. Therefore, due to the exactness of the quadrature formula for polynomials of degree $2S-3$, we can integrate by parts and obtain

$$\begin{aligned}\frac{1}{z}\left(B\,\underline{\boldsymbol{c}^{z}}\right)_{\ell} &= \frac{1}{z}\left\{\widehat{\psi}^{\ell}(1)\cdot 1^{z} - \int_0^1\left(\frac{d}{d\xi}\widehat{\psi}^{\ell}(\xi)\right)\xi^{z}d\xi\right\} \\ &= \frac{1}{z}\left\{\widehat{\psi}^{\ell}(1)\cdot 1^{z} - \widehat{\psi}^{\ell}(1)\cdot 1^{z} + \widehat{\psi}^{\ell}(0)\cdot 0^{z} + z\int_0^1\widehat{\psi}^{\ell}(\xi)\xi^{z-1}d\xi\right\} = \int_0^1\widehat{\psi}^{\ell}(\xi)\xi^{z-1}d\xi.\end{aligned} \tag{15c}$$

The left-hand side is slightly more delicate to handle, as the adopted quadrature formula does not allow to compute exactly the terms of the matrix Λ. We have

$$\left(\Lambda\,\underline{\boldsymbol{c}^{z-1}}\right)_{\ell} = \sum_{m=0}^{M} Q\left\{\widehat{\psi}^{\ell}(\xi)\widehat{\psi}^{m}(\xi)\right\}(\xi^{m})^{z-1} = Q\left\{\widehat{\psi}^{\ell}(\xi)\left[\sum_{m=0}^{M}\widehat{\psi}^{m}(\xi)\,(\xi^{m})^{z-1}\right]\right\} \tag{15d}$$

$$= Q\left\{\widehat{\psi}^{\ell}(\xi)\xi^{z-1}\right\} = \int_0^1\widehat{\psi}^{\ell}(\xi)\xi^{z-1}d\xi = \frac{1}{z}\left(B\,\underline{\boldsymbol{c}^{z}}\right)_{\ell}, \tag{15e}$$

where in (15d) we have used the linearity of the quadrature operator and in (15e) the fact that the subtimenodes exactly interpolate polynomials up to degree $M = S-1$ and (15c).

Note that the condition $C(S-1)$ is sharp. In fact, since the interpolation is not anymore exact for $z = S$, then $C(S)$ is not satisfied. □

The condition of the previous lemma is therefore satisfied by ADER-IWF-RK methods with GLB and GLG subtimenodes if the same subtimenodes are adopted as quadrature points, since they induce quadrature formulas respectively characterized by degree of exactness equal to $2S-3$ and $2S-1$. We refer to such schemes, characterized by having the subtimenodes as quadrature points, as ADER-IWF-RK-GLB and ADER-IWF-RK-GLG. They satisfy another important property stated in the next lemma.

Lemma 3.6. *The ADER-IWF-RK-GLB and ADER-IWF-RK-GLG methods satisfy $D(S-1)$.*

Proof. Let us observe that, for both ADER-IWF-RK-GLB and ADER-IWF-RK-GLG methods, the subtimenodes and the quadrature points coincide. Hence, in these particular cases, we have a diagonal matrix Λ. In fact, its general entry is

$$\Lambda_{\ell,m} := Q\left\{\widehat{\psi}^{\ell}(\xi)\widehat{\psi}^{m}(\xi)\right\} = w_{\ell}\delta_{\ell,m}, \tag{16a}$$

where $w_{\ell} = \int_0^1\widehat{\psi}^{\ell}d\xi = b_{\ell}$ is the quadrature weight associated to ξ^{ℓ}. This fact will be useful in the following.

We write explicitly condition $D(S-1)$ in matricial form, for all $z = 1,\dots,S-1$

$$A^{T}\underline{\boldsymbol{b}\boldsymbol{c}^{z-1}} = \frac{1}{z}\underline{\boldsymbol{b}(\mathbf{1}-\boldsymbol{c}^{z})} \Longleftrightarrow \left[\underline{\boldsymbol{b}\boldsymbol{c}^{z-1}}\right]^{T}A = \frac{1}{z}\left[\underline{\boldsymbol{b}(\mathbf{1}-\boldsymbol{c}^{z})}\right]^{T} \Longleftrightarrow \left[\underline{\boldsymbol{b}\boldsymbol{c}^{z-1}}\right]^{T} = \frac{1}{z}\left[\underline{\boldsymbol{b}(\mathbf{1}-\boldsymbol{c}^{z})}\right]^{T}\Lambda^{-1}B, \tag{16b}$$

where the general i-th entries of the vectors $\underline{\boldsymbol{b}\boldsymbol{c}^{z-1}}$ and $\underline{\boldsymbol{b}(\mathbf{1}-\boldsymbol{c}^z)}$ are respectively $b_i c_i^{z-1}$ and $b_i(1-c_i^z)$.

Recalling that $\Lambda_{i,j} = w_i\delta_{ij} = b_i\delta_{ij}$, the following expression holds $\left[\underline{\boldsymbol{b}(\mathbf{1}-\boldsymbol{c}^z)}\right]^T \Lambda^{-1} = \underline{(\mathbf{1}-\boldsymbol{c}^z)}^T$, with general i-th entry equal to $1-c_i^z$. Hence, it is left to prove that $\underline{\boldsymbol{b}\boldsymbol{c}^{z-1}} = \frac{1}{z}B^T\underline{\mathbf{1}-\boldsymbol{c}^z}$.

Expanding the ℓ-th component of the right-hand side, we get

$$\begin{aligned}\frac{1}{z}\left(B^T\underline{\mathbf{1}-\boldsymbol{c}^z}\right)_\ell &= \frac{1}{z}\sum_{m=0}^{M}\left[\widehat{\psi}^m(1)\widehat{\psi}^\ell(1) - \int_0^1\left(\frac{d}{d\xi}\widehat{\psi}^m(\xi)\right)\widehat{\psi}^\ell(\xi)d\xi\right]\left[1-(\xi^m)^z\right]\\ &= \frac{1}{z}\left\{\left(\sum_{m=0}^{M}\widehat{\psi}^m(1)\left[1-(\xi^m)^z\right]\right)\widehat{\psi}^\ell(1) - \int_0^1\left(\sum_{m=0}^{M}\frac{d}{d\xi}\widehat{\psi}^m(\xi)\left[1-(\xi^m)^z\right]\right)\widehat{\psi}^\ell(\xi)d\xi\right\}.\end{aligned} \tag{16c}$$

Since $(1-\xi^z)$ is a polynomial of degree at most $S-1$, its Lagrange interpolation is exact, hence

$$\begin{aligned}\frac{1}{z}\left(B^T\underline{\mathbf{1}-\boldsymbol{c}^z}\right)_\ell &= \frac{1}{z}\left\{(1-1^z)\cdot\widehat{\psi}^\ell(1) - \int_0^1\left[\frac{d}{d\xi}(1-\xi^z)\right]\widehat{\psi}^\ell(\xi)d\xi\right\}\\ &= \int_0^1\widehat{\psi}^\ell(\xi)\xi^{z-1}d\xi = w_\ell\cdot\left(\xi^\ell\right)^{z-1} = \left(\underline{\boldsymbol{b}\boldsymbol{c}^{z-1}}\right)_\ell.\end{aligned} \tag{16d}$$

In the last step, we have exploited the exactness of the quadrature rule in the GLB or GLG subtimenodes for the considered integral. Hence, condition $\mathcal{D}(S-1)$ holds. Note that also in this case the condition is sharp: the exact interpolation of polynomials of degree $S-1$, guaranteed for $z \le S-1$, was necessary and therefore $\mathcal{D}(S)$ is not satisfied. □

Remark 3.7 (ADER-IWF-RK-GLG is not a collocation method)**.** *From the proof of Lemma 3.5, we can observe that ADER-IWF-RK-GLG methods do not satisfy $C(S)$, despite having all the c_i coefficients distinct, hence, the methods are not collocation methods and they do not coincide with Gauss methods [38].*

To finally obtain the accuracy of ADER-IWF-RK-GLB and ADER-IWF-RK-GLG, we use Theorem 3.4.

Theorem 3.8. *ADER-IWF-RK-GLB is of order* $2S-2$.

Proof. The condition $\mathcal{B}(2S-2)$ holds, because the vectors $\boldsymbol{c}$ and $\boldsymbol{b}$ are the quadrature points and weights of the GLB quadrature formula characterized by degree of exactness $2S-3$. Lemmas 3.5 and 3.6 prove that ADER-IWF-RK-GLB satisfies $C(S-1)$ and $\mathcal{D}(S-1)$, so Theorem 3.4 is satisfied for order $p = 2S-2$ and $\eta = \zeta = S-1$. Moreover, since the quadrature is of order exactly $2S-2$, $\mathcal{B}(2S-1)$ does not hold and, hence, the method is not of order $2S-1$. This observation is based on the theory of order conditions of RK methods involving trees presented in [38]. If $\mathcal{B}(p)$ is not satisfied, not all the order conditions of [38, Theorem 2.13] hold for all trees of order $\le p$ and the method cannot be of order p. □

In particular, the ADER-IWF-RK-GLB methods coincide with Lobatto IIIC RK methods [66] and we prove it in Appendix B.

Theorem 3.9. *ADER-IWF-RK-GLG is of order* $2S-1$.

Proof. ADER-IWF-RK-GLG satisfies $\mathcal{B}(2S)$, because the vectors $\boldsymbol{c}$ and $\boldsymbol{b}$ are the quadrature points and weights of the GLG quadrature formula characterized by degree of exactness $2S-1$. Hence, also $\mathcal{B}(2S-1)$ holds. For Lemmas 3.5 and 3.6, it also satisfies $C(S-1)$ and $\mathcal{D}(S-1)$. Hence, Theorem 3.4 guarantees that the method is of order $2S-1$, since it is satisfied with $p = 2S-1$ and $\eta = \zeta = S-1$. Also here, it is possible to prove that ADER-IWF-RK-GLG is not of order $2S$, but sharply of order $2S-1$. Indeed, a method of order $2S$ must verify $C(S)$, see [14, Theorem 342C]. So, by contradiction, it must be that ADER-IWF-RK-GLG is not of order $2S$. □

3.2. Beyond (and within) nodal bases

In this subsection, we prove an interesting result concerning the equivalence between ADER schemes with arbitrary bases and ADER schemes with Lagrangian basis functions defined in the quadrature points of the former schemes. Let us start by introducing the ADER formulation for an arbitrary basis $\{\phi^m\}_{m=0,\dots,M}$ of the space of

polynomials with degree M over $[t_n, t_{n+1}]$. Again, moving from the weak formulation of the ODEs system (1) and projecting it onto a finite dimensional functional space, we get the nonlinear system

$$\begin{aligned}\sum_{m=0}^{M}\left[\phi^\ell(t_{n+1})\phi^m(t_{n+1}) - \int_{t_n}^{t_{n+1}}\left(\frac{d}{dt}\phi^\ell(t)\right)\phi^m(t)dt\right]\boldsymbol{u}^m - \phi^\ell(t_n)\boldsymbol{u}_n \\ - \int_{t_n}^{t_{n+1}}\phi^\ell(t)\boldsymbol{G}_h(t)dt = \boldsymbol{0}, \quad \ell = 0,\dots,M,\end{aligned} \tag{17}$$

where $\boldsymbol{G}_h(t) := \boldsymbol{G}(t, \boldsymbol{u}_h(t))$ with the reconstruction $\boldsymbol{u}_h(t) := \sum_{m=0}^{M}\boldsymbol{u}^m\phi^m(t)$. The unknown coefficients $\boldsymbol{u}^m$ are general representation coefficients of the numerical solution in the polynomial space spanned by the basis functions, not anymore nodal values. In literature, one can find different choices of basis functions alternative to nodal ones, for example modal Taylor basis functions [12, 7, 32, 48].

In order to get a fully discrete version, we need to specify the quadrature formula that we are going to use in the integral terms: $\{(\xi^q, w_q)\}_{q=0,\dots,M}$ with $\xi^q \in [0,1]$ nodes and $w_q = \int_0^1 \widehat{\psi}^q(\xi)d\xi$ weights of the quadrature, where $\widehat{\psi}^q$ are the Lagrangian basis functions associated to the quadrature points ξ^q. Then, (17) reads

$$\begin{aligned}\sum_{m=0}^{M}\left[\widehat{\phi}^\ell(1)\widehat{\phi}^m(1) - \sum_{q=0}^{M}\left(\frac{d}{d\xi}\widehat{\phi}^\ell(\xi^q)\right)\widehat{\phi}^m(\xi^q)w_q\right]\boldsymbol{u}^m - \widehat{\phi}^\ell(0)\boldsymbol{u}_n \\ - \Delta t\sum_{q=0}^{M}\widehat{\phi}^\ell(\xi^q)\boldsymbol{G}\left(\xi^q, \sum_{m=0}^{M}\boldsymbol{u}^m\widehat{\phi}^m(\xi^q)\right)w_q = \boldsymbol{0}, \quad \ell = 0,\dots,M,\end{aligned} \tag{18}$$

which is the ADER-IWF of the method defined by the basis $\{\phi^m\}_{m=0,\dots,M}$ and the quadrature $\{(\xi^q, w_q)\}_{q=0,\dots,M}$. The system can be solved iteratively as in (10) to obtain $\boldsymbol{u}_{n+1} := \boldsymbol{u}_h(t_{n+1})$. Again, we denote with $\widehat{(\cdot)}$ the quantities rescaled onto $[0,1]$.

Theorem 3.10 (Link between ADER schemes with arbitrary and nodal bases)**.** *Consider a basis $\{\phi^m\}_{m=0,\dots,M}$ of the space of the polynomials of degree M over $[t_n, t_{n+1}]$ and let us denote by $\{(\xi^q, w_q)\}_{q=0,\dots,M}$ the $(M+1)$-points GLB or GLG quadrature rule, which we will denote by GL*, and by $\{\psi^m\}_{m=0,\dots,M}$ the respective Lagrange polynomials. Then, the ADER-IWF* (18) *defined by $\{\phi^m\}_{m=0,\dots,M}$ and by the quadrature $\{(\xi^q, w_q)\}_{q=0,\dots,M}$ is equivalent to the ADER-IWF* (5) *defined by $\{\psi^m\}_{m=0,\dots,M}$ with integrals computed through the same quadrature.*

Proof. Let us observe that, being $\{\widehat{\phi}^m\}_{m=0,\dots,M}$ a basis of the space of the polynomials of degree M over $[0,1]$, there exists a unique vector of coefficients $\underline{\gamma} = (\gamma_0, \dots, \gamma_M)^T$ such that

$$\sum_{m=0}^{M}\gamma_m\widehat{\phi}^m \equiv 1. \tag{19a}$$

We aim at rewriting (18) into a matrix formulation, as in (9), and at comparing the resulting systems. We first introduce $\boldsymbol{v}^q = \boldsymbol{u}_h(t_n + \Delta t\xi^q) = \sum_{m=0}^{M}\boldsymbol{u}^m\widehat{\phi}^m(\xi^q)$, the reconstructed solution value in the GL* quadrature point ξ^q, so that $\boldsymbol{u}_h(t) = \sum_{m=0}^{M}\boldsymbol{u}^m\phi^m(t) = \sum_{q=0}^{M}\boldsymbol{v}^q\psi^q(t)$. We also define the change of basis matrix $\mathcal{H}$ such that $\underline{\boldsymbol{v}} = \mathcal{H}\underline{\boldsymbol{u}}$, where the general element of such matrix is defined as $\mathcal{H}_{\ell,m} = \widehat{\phi}^m(\xi^\ell)$.

Hence, system (18) can be equivalently recast as

$$B\underline{\boldsymbol{u}} - \underline{\boldsymbol{r}} - \Delta t\Lambda\underline{\boldsymbol{G}}(\underline{\boldsymbol{v}}) = \boldsymbol{0}, \tag{19b}$$

where

$$\begin{aligned}&B_{\ell,m} := \widehat{\phi}^\ell(1)\widehat{\phi}^m(1) - \sum_{q=0}^{M}\left(\frac{d}{d\xi}\widehat{\phi}^\ell(\xi^q)\right)\widehat{\phi}^m(\xi^q)w_q, \qquad \Lambda_{\ell,m} := \widehat{\phi}^\ell(\xi^m)w_m, \\ &\underline{\boldsymbol{u}} := \begin{pmatrix}\boldsymbol{u}^0\\ \vdots \\ \boldsymbol{u}^M\end{pmatrix}, \quad \underline{\boldsymbol{r}} := \begin{pmatrix}\phi^0(t_n)\boldsymbol{u}_n\\ \vdots \\ \phi^M(t_n)\boldsymbol{u}_n\end{pmatrix}, \quad \underline{\boldsymbol{G}}(\underline{\boldsymbol{v}}) := \begin{pmatrix}\boldsymbol{G}(t^0, \boldsymbol{v}^0)\\ \vdots \\ \boldsymbol{G}(t^M, \boldsymbol{v}^M)\end{pmatrix}, \quad \underline{\boldsymbol{v}} = \mathcal{H}\underline{\boldsymbol{u}}.\end{aligned} \tag{19c}$$

Again, for compactness, the structures are referred to a scalar problem and the matrices should be block expanded for a vectorial one. We can rewrite Λ as $\Lambda = \mathcal{H}^T W$ with W being a diagonal matrix having as entries the GL* quadrature weights w_m.

Then, inverting B and multiplying by $\mathcal{H}$, we have that system (19b) is equivalent to

$$\underline{\boldsymbol{v}} - \mathcal{H}B^{-1}\underline{\boldsymbol{r}} - \Delta t\mathcal{H}B^{-1}\mathcal{H}^T W\underline{\boldsymbol{G}}(\underline{\boldsymbol{v}}) = \boldsymbol{0}. \tag{19d}$$

Now, we can compare such system with the ADER system (9) obtained for GL* subtimenodes and quadrature. In particular, the two formulations coincide if and only if

$$\begin{cases} \mathcal{H}B^{-1}\underline{\boldsymbol{r}} = \underline{\boldsymbol{u}}_n \\ \mathcal{H}B^{-1}\mathcal{H}^T W = B_{GL*}^{-1}\Lambda_{GL*}, \end{cases} \tag{19e}$$

where B_{GL*} and Λ_{GL*} are the ADER structures given in (8) with GL* subtimenodes, also assumed as quadrature points for the computation of the integrals.

Let us start by the first equivalence of (19e). Let us simplify the notation of $\underline{\boldsymbol{r}}$ writing it as $\underline{\boldsymbol{r}} = \underline{\widehat{\phi}}(0)\boldsymbol{u}_n$, where $\boldsymbol{u}_n$ is meant to be multiplied to each component of $\underline{\widehat{\phi}}(0)$. With this, we just need to prove that

$$\mathcal{H}B^{-1}\underline{\widehat{\phi}}(0) = \mathbf{1}. \tag{19f}$$

Recalling that $\sum_{m=0}^{M}\gamma_m\widehat{\phi}^m(\xi) \equiv 1$, we can instead prove that

$$\underline{\widehat{\phi}}(0) = B\underline{\gamma}, \tag{19g}$$

as we would have $\left(\mathcal{H}B^{-1}\underline{\widehat{\phi}}(0)\right)_\ell = \left(\mathcal{H}\underline{\gamma}\right)_\ell = \sum_{m=0}^{M}\widehat{\phi}^m(\xi^\ell)\gamma_m = 1$. Equality (19g) can be proven expanding the right hand side:

$$\begin{aligned} \left(B\underline{\gamma}\right)_\ell &= \sum_{m=0}^{M}\left[\widehat{\phi}^\ell(1)\widehat{\phi}^m(1) - \int_0^1\left(\frac{d}{d\xi}\widehat{\phi}^\ell(\xi)\right)\widehat{\phi}^m(\xi)d\xi\right]\gamma_m \\ &= \widehat{\phi}^\ell(1)\left(\sum_{m=0}^{M}\widehat{\phi}^m(1)\gamma_m\right) - \int_0^1\left(\frac{d}{d\xi}\widehat{\phi}^\ell(\xi)\right)\left(\sum_{m=0}^{M}\widehat{\phi}^m(\xi)\gamma_m\right)d\xi \\ &= \widehat{\phi}^\ell(1)\cdot 1 - \int_0^1\left(\frac{d}{d\xi}\widehat{\phi}^\ell(\xi)\right)\cdot 1 d\xi = \widehat{\phi}^\ell(1) - \widehat{\phi}^\ell(1) + \widehat{\phi}^\ell(0) = \widehat{\phi}^\ell(0), \end{aligned} \tag{19h}$$

thanks again to (19a) and the exactness of the quadrature formula for polynomials of degree $M-1$.
For the second equality of (19e), let us first notice that, due to the assumption on the quadrature, we have $\Lambda_{GL*} = W$. Thus, we suffice to prove

$$\mathcal{H}B^{-1}\mathcal{H}^T = B_{GL*}^{-1}. \tag{19i}$$

Taking the inverse of the previous equality and inverting the $\mathcal{H}$ matrices, we get

$$\mathcal{H}B^{-1}\mathcal{H}^T = B_{GL*}^{-1} \iff (\mathcal{H}^T)^{-1}B(\mathcal{H})^{-1} = B_{GL*} \iff B = \mathcal{H}^T B_{GL*}\mathcal{H}.$$

Let us compute the general entry of the matrix at the right-hand side

$$\begin{aligned} \left(\mathcal{H}^T B_{GL*}\mathcal{H}\right)_{\ell,m} &= \sum_{i=0}^{M}\left(\mathcal{H}^T\right)_{\ell,i}\sum_{j=0}^{M}\left(B_{GL*}\right)_{i,j}\mathcal{H}_{j,m} \\ &= \sum_{i=0}^{M}\widehat{\phi}^\ell(\xi^i)\sum_{j=0}^{M}\left[\widehat{\psi}^i(1)\widehat{\psi}^j(1) - \int_0^1\left(\frac{d}{d\xi}\widehat{\psi}^i(\xi)\right)\widehat{\psi}^j(\xi)d\xi\right]\widehat{\phi}^m(\xi^j) \\ &= \sum_{i=0}^{M}\widehat{\phi}^\ell(\xi^i)\widehat{\psi}^i(1)\sum_{j=0}^{M}\widehat{\psi}^j(1)\widehat{\phi}^m(\xi^j) - \int_0^1\frac{d}{d\xi}\left(\sum_{i=0}^{M}\widehat{\phi}^\ell(\xi^i)\widehat{\psi}^i(\xi)\right)\sum_{j=0}^{M}\widehat{\psi}^j(\xi)\widehat{\phi}^m(\xi^j)d\xi. \end{aligned} \tag{19j}$$

Now, observing that $\sum_{i=0}^{M}\widehat{\phi}^\ell(\xi^i)\widehat{\psi}^i$ and $\sum_{j=0}^{M}\widehat{\phi}^m(\xi^j)\widehat{\psi}^j$ are the exact interpolations of $\widehat{\phi}^\ell$ and $\widehat{\phi}^m$ respectively, we finally have

$$\left(\mathcal{H}^T B_{GL*}\mathcal{H}\right)_{\ell,m} = \widehat{\phi}^\ell(1)\widehat{\phi}^m(1) - \int_0^1\left(\frac{d}{d\xi}\widehat{\phi}^\ell(\xi)\right)\widehat{\phi}^m(\xi)d\xi = B_{\ell,m}. \tag{19k}$$

Clearly, since the nodal values define a unique polynomial, also the final interpolation step will be the same for the two ADER methods involved in this theorem. □

The previous result is interesting under several points of view. Thanks to Theorem 3.10, we can give a precise characterization of the ADER methods for arbitrary bases. If, for some reason, a user wants to adopt another basis other than the GL* polynomials for the definition of the ADER method, for example a Taylor basis, then the properties of the resulting scheme do not change, provided that the function $\boldsymbol{G}$ is directly evaluated in the GL* quadrature points. Therefore, also the accuracy of the resulting scheme is known and it is characterized by the one of the ADER-IWF-RK-GL* methods in the previous section.

Notice that Theorem 3.10 applies also in the context of nodal bases: if one chooses $M+1$ equispaced subtimenodes for the definition of the ADER method but $\boldsymbol{G}$ is evaluated in the GL* quadrature points, instead of the basis nodal values, then the resulting accuracy will not be $M+1$, but rather $2M$ and $2M+1$ for GLB and GLG, respectively.

4. ADER as a Deferred Correction method

This section aims to determine the optimal number of iterations for an ADER method using the DeC formalism, an abstract framework used to approximate arbitrarily well the solution of implicit (nonlinear) discretizations of analytical problems, through an easy iterative procedure. Originally presented in 1949 [30], the DeC was applied in different flavors to ODEs [50, 44, 42, 49, 52, 47, 39] and PDEs [51, 57, 1, 16, 48, 4, 5, 2] contexts. Abgrall [1] proposed a formalization using two operators $\mathcal{L}^1_\Delta, \mathcal{L}^2_\Delta : X \to Y$, depending on a same parameter Δ, corresponding to two different discretizations of the same problem: $\mathcal{L}^2_\Delta$ is a *difficult-to-solve* high order nonlinear implicit discretization of the problem and $\mathcal{L}^1_\Delta$ is an *easy-to-solve* low order (explicit) one. Aiming at $\underline{\boldsymbol{u}}_\Delta \in X$, solution of $\mathcal{L}^2_\Delta(\underline{\boldsymbol{u}}_\Delta) = \boldsymbol{0}_Y$, we iteratively approximate it arbitrarily well through an *easy-to-solve* (explicit) iteration process as prescribed in the next theorem.

Theorem 4.1 (DeC; Abgrall [1])**.** *For a fixed $\underline{\boldsymbol{u}}^{(0)} \in X$, let us define the sequence of vectors $\underline{\boldsymbol{u}}^{(p)}$ as the solution of*

$$\mathcal{L}^1_\Delta(\underline{\boldsymbol{u}}^{(p)}) := \mathcal{L}^1_\Delta(\underline{\boldsymbol{u}}^{(p-1)}) - \mathcal{L}^2_\Delta(\underline{\boldsymbol{u}}^{(p-1)}), \quad p \geq 1. \tag{20}$$

If the following conditions on the operators $\mathcal{L}^1_\Delta$ and $\mathcal{L}^2_\Delta$ hold

1. $\exists!\, \underline{\boldsymbol{u}}_\Delta \in X$ *solution of* $\mathcal{L}^2_\Delta$ *such that* $\mathcal{L}^2_\Delta(\underline{\boldsymbol{u}}_\Delta) = \boldsymbol{0}_Y$ *(Existence of a unique solution to $\mathcal{L}^2_\Delta$);*
2. $\exists\, \alpha_1 \geq 0$ *independent of* Δ *such that*

$$\left\| \mathcal{L}^1_\Delta(\underline{\boldsymbol{v}}) - \mathcal{L}^1_\Delta(\underline{\boldsymbol{w}}) \right\|_Y \geq \alpha_1 \left\| \underline{\boldsymbol{v}} - \underline{\boldsymbol{w}} \right\|_X, \ \forall \underline{\boldsymbol{v}}, \underline{\boldsymbol{w}} \in X \text{(Coercivity-like property of } \mathcal{L}^1_\Delta); \tag{21}$$

3. $\exists\, \alpha_2 \geq 0$ *independent of* Δ *such that*

$$\left\| \left[\mathcal{L}^1_\Delta(\underline{\boldsymbol{v}}) - \mathcal{L}^2_\Delta(\underline{\boldsymbol{v}}) \right] - \left[\mathcal{L}^1_\Delta(\underline{\boldsymbol{w}}) - \mathcal{L}^2_\Delta(\underline{\boldsymbol{w}}) \right] \right\|_Y \leq \alpha_2 \Delta \left\| \underline{\boldsymbol{v}} - \underline{\boldsymbol{w}} \right\|_X, \ \forall \underline{\boldsymbol{v}}, \underline{\boldsymbol{w}} \in X; \tag{22}$$
(Lipschitz-continuity-like property of $\mathcal{L}^1_\Delta - \mathcal{L}^2_\Delta$)

then, we can prove the following error estimate

$$\left\| \underline{\boldsymbol{u}}^{(p)} - \underline{\boldsymbol{u}}_\Delta \right\|_X \leq \left(\Delta \frac{\alpha_2}{\alpha_1} \right)^p \left\| \underline{\boldsymbol{u}}^{(0)} - \underline{\boldsymbol{u}}_\Delta \right\|_X, \quad \forall p \in \mathbb{N}. \tag{23}$$

The proof can be found in [1, 47, 48] and it uses induction on the iterations. Estimate (23) tells that $\underline{\boldsymbol{u}}_\Delta$ can be approximated with arbitrarily high precision as $p \to +\infty$. However, $\underline{\boldsymbol{u}}_\Delta$ is itself an approximation of the exact solution $\underline{\boldsymbol{u}}_{ex}$ of the original analytical problem to which the operators are associated. If its order of accuracy is R, it suffices to find an R-th order accurate approximation of $\underline{\boldsymbol{u}}_\Delta$ to get the same formal order of accuracy. By triangular inequality, if $\underline{\boldsymbol{u}}^{(0)}$ is $O(\Delta)$-accurate, the order of accuracy of $\underline{\boldsymbol{u}}^{(p)}$ with respect to $\underline{\boldsymbol{u}}_{ex}$ is $\min(p, R)$, hence, the optimal choice for the final number of iterations P is $P = R$. Extra iterations may (slightly) improve the accuracy but they do not increase the order of accuracy and are essentially a waste of computational resources.

4.1. Link ADER-DeC

We will show here how the ADER method presented in Section 2 can be put in a DeC formalism with $\Delta = \Delta t$. We start by defining the N-th order accurate operator $\mathcal{L}^2_\Delta : \mathbb{R}^{(M+1)\times Q} \to \mathbb{R}^{(M+1)\times Q}$ as

$$\mathcal{L}^2_\Delta(\underline{\boldsymbol{u}}) := \underline{\boldsymbol{u}} - \underline{\boldsymbol{u}}_n - \Delta t B^{-1} \Lambda \underline{\boldsymbol{G}}(\underline{\boldsymbol{u}}). \tag{24}$$

Let us notice that the problem of finding $\underline{\boldsymbol{u}}_\Delta$ such that $\mathcal{L}^2_\Delta(\underline{\boldsymbol{u}}_\Delta) = \mathbf{0}$ is indeed equivalent to solving the nonlinear system (9). We introduce the low order operator $\mathcal{L}^1_\Delta : \mathbb{R}^{(M+1)\times Q} \to \mathbb{R}^{(M+1)\times Q}$ as a first order explicit approximation of (9)

$$\mathcal{L}^1_\Delta(\underline{\boldsymbol{u}}) := \underline{\boldsymbol{u}} - \underline{\boldsymbol{u}}_n - \Delta t B^{-1} \Lambda \underline{\boldsymbol{G}}(\underline{\boldsymbol{u}}_n). \tag{25}$$

Remark 4.2 (On the accuracy of the operators)**.** *We remark that the mentioned order of accuracy of the operators $\mathcal{L}^1_\Delta$ and $\mathcal{L}^2_\Delta$ is referred, in this context, to the accuracy of the approximation $\boldsymbol{u}_{n+1} = \boldsymbol{u}_h(t_{n+1})$ that the related solution coefficients $\underline{\boldsymbol{u}}$ induce via* (4)*.*

We can easily prove that the operators that we have defined satisfy the assumptions of Theorem 4.1.

Theorem 4.3 (Properties of $\mathcal{L}^1_\Delta$ and $\mathcal{L}^2_\Delta$; Han Veiga, Öffner, Torlo [39])**.** *The operators $\mathcal{L}^1_\Delta$ and $\mathcal{L}^2_\Delta$, given by* (25) *and* (24)*, satisfy the hypotheses of Theorem 4.1.*

The proof can be found in [39, Propositions 4.3 and 4.4]. Finally, the resulting DeC iteration (20), in this particular case, after an easy direct computation, reads

$$\underline{\boldsymbol{u}}^{(p)} = \underline{\boldsymbol{u}}_n + \Delta t B^{-1} \Lambda \underline{\boldsymbol{G}}(\underline{\boldsymbol{u}}^{(p-1)}), \tag{26}$$

which coincides with the iteration defined in (10). However, the DeC formalism allows us to select the optimal number of iterations, which is equal to N, i.e., the order of $\mathcal{L}^2_\Delta = 0$, provided that $\underline{\boldsymbol{u}}^{(0)}$ is $O(\Delta t)$-accurate. Therefore, we can set $\underline{\boldsymbol{u}}^{(0)} := \underline{\boldsymbol{u}}_n$ and $P = N$ to get the formal order of accuracy.

5. Novel modified ADER methods

In this section, we describe some efficient versions of ADER methods based on a modification of the original approach, following the strategy proposed in [47], in a DeC context for ODEs, and then generalized and applied to an ADER-DG framework in [48]. Actually, the basic idea was firstly introduced by Minion in [50] as ladder DeC methods for ODEs, counting several follow-ups [57**? ? ? ?**]. However, the approach was very specific for DeC time-integration methods for ODEs. The more general formulation presented in [48] allows for applications to many other contexts, e.g., to ADER schemes. In particular, the modification consists in redesigning the whole iterative process in such a way that the discretization accuracy increases accordingly to the order of accuracy of the numerical solution at each specific iteration. In practice, we will look for a solution in a different approximation space at each iteration p, i.e., the time reconstruction is such that $\boldsymbol{u}_h^{(p)}(t) \in \mathbb{P}_p$ up to a maximum reconstruction degree, where $\mathbb{P}_p$ is the space of polynomials of degree p over $[t_n, t_{n+1}]$. This is beneficial under many points of view, among which:

- we save computations in the early iterations, as we work with smaller vectors and matrices;
- p-adaptivity can be naturally embedded in the new methods, as there is no formal upper bound on the order of accuracy.

In order to change iteration structures along the iterative process, some embeddings between different spaces $\mathcal{E}^{(p-1)} : \mathbb{P}^{p-1} \hookrightarrow \mathbb{P}^p$, e.g., interpolations or L^2-projections, are needed to pass from $\underline{\boldsymbol{u}}^{(p-1)}$ to some $\underline{\boldsymbol{u}}^{*(p-1)}$, which is a suitable input for the p-th iteration to compute $\underline{\boldsymbol{u}}^{(p)}$. For the specific context of DeC methods, the following theorem holds.

Theorem 5.1 (Micalizzi, Torlo, Boscheri [48])**.** *Let us consider a problem with exact solution $\underline{\boldsymbol{u}}_{ex} \in Z$ and the normed vector spaces $(X^{(p)}, \|\cdot\|_{X^{(p)}})$ and $(Y^{(p)}, \|\cdot\|_{Y^{(p)}})$ for $p \in \mathbb{N}$, $p \geq 1$. Further, let us assume that some operators $\mathcal{L}^{1,(p)}_\Delta, \mathcal{L}^{2,(p)}_\Delta : X^{(p)} \to Y^{(p)}$ are defined for $p \geq 1$, dependent on the same parameter Δ and satisfying the assumptions in Theorem 4.1 for $\alpha_1^{(p)}, \alpha_2^{(p)} > 0$ and $\underline{\boldsymbol{u}}^{(p)}_\Delta \in X^{(p)}$. Let us also assume that $\forall p \in \mathbb{N}$ there exists an embedding operator $\mathcal{E}^{(p)} : X^{(p)} \to X^{(p+1)}$ and a projection $\Pi^{(p)} : Z \to X^{(p)}$. We define $\underline{\boldsymbol{u}}^{*(p)} := \mathcal{E}^{(p)}(\underline{\boldsymbol{u}}^{(p)}) \in X^{(p+1)}$ and $\underline{\boldsymbol{u}}^{(p)}_{ex} := \Pi^{(p)}(\underline{\boldsymbol{u}}_{ex}) \in X^{(p)}$. Let us consider the modified DeC method whose general p-th iteration is given by*

$$\begin{cases} \underline{\boldsymbol{u}}^{*(p-1)} := \mathcal{E}^{(p-1)}(\underline{\boldsymbol{u}}^{(p-1)}), \\ \mathcal{L}^{1,(p)}_\Delta(\underline{\boldsymbol{u}}^{(p)}) := \mathcal{L}^{1,(p)}_\Delta(\underline{\boldsymbol{u}}^{*(p-1)}) - \mathcal{L}^{2,(p)}_\Delta(\underline{\boldsymbol{u}}^{*(p-1)}), \end{cases} \tag{27}$$

for some $\underline{\boldsymbol{u}}^{(0)} \in X^{(0)}$. Moreover, assume that $\left\|\underline{\boldsymbol{u}}^{(p)}_\Delta - \underline{\boldsymbol{u}}^{(p)}_{ex}\right\|_{X^{(p)}} = O(\Delta^{p+1})$, for $p \geq 1$, that there exists $C > 0$ independent of Δ such that $\left\|\underline{\boldsymbol{u}}^{*(p)} - \underline{\boldsymbol{u}}^{(p+1)}_{ex}\right\|_{X^{(p+1)}} \leq C \left\|\underline{\boldsymbol{u}}^{(p)} - \underline{\boldsymbol{u}}^{(p)}_{ex}\right\|_{X^{(p)}}$, for $p \geq 0$ and that $\left\|\underline{\boldsymbol{u}}^{(0)} - \underline{\boldsymbol{u}}^{(0)}_{ex}\right\|_{X^{(0)}} = O(\Delta)$. Then, the following error estimate holds

$$\left\|\underline{\boldsymbol{u}}^{(p)} - \underline{\boldsymbol{u}}^{(p)}_{ex}\right\|_{X^{(p)}} = O(\Delta^{p+1}), \quad \forall p \in \mathbb{N}. \tag{28}$$

The proof can be found in [48]. A fundamental difference between the framework of Theorem 4.1 and the one of Theorem 5.1 is the fact that the former deals with converge towards the solution of a fixed operator $\mathcal{L}^2_\Delta$, see (23), while in the latter the error estimate (28) is referred, at each iteration, to a new and more accurate projection of the exact solution $\underline{\boldsymbol{u}}_{ex}$.

In the previous section, we have proved how the ADER methods can be put in a DeC framework. Now, we will see how the presented modification can be applied in this specific case. In particular, we propose three modifications, which differ in the way the embeddings between the different iterations are achieved, even though we will later prove that two of them are actually equivalent in the considered framework.

We assume equispaced subtimenodes at the beginning and we generalize for other types of subtimenodes later on.

We introduce here, for each p, the space $X^{(p)} := \mathbb{R}^{(M^{(p)}+1)\times Q}$ of the representation coefficients of vectorial polynomial functions in $(\mathbb{P}_{M^{(p)}})^Q$, where the used basis functions $\psi^{m,(p)}$ for $m = 0, \dots, M^{(p)}$ are the Lagrange polynomials of degree $M^{(p)}$ associated to $M^{(p)} + 1$ equispaced subtimenodes in the interval $[t_n, t_{n+1}]$, collected in the vector $\underline{t}^{(p)} := \left(t^{0,(p)}, \dots, t^{M^{(p)},(p)}\right)^T$, with $M^{(p)} = p$ for all $p \neq 0$ and $M^{(0)} = 1$. It is also useful to introduce here some structures associated to such basis functions and in particular the matrices $B^{(p)}$ and $\Lambda^{(p)}$, defined as in (8) but considering the functions $\left\{\psi^{m,(p)}\right\}_{m=0,\dots,M^{(p)}}$ in place of $\{\psi^m\}_{m=0,\dots,M}$.

In the following, we will explain in detail the procedure to pass from $\underline{\boldsymbol{u}}^{(p-1)} \in X^{(p-1)}$ to $\underline{\boldsymbol{u}}^{(p)} \in X^{(p)}$ with the three approaches.

5.1. ADERu

In this case, the embeddings consist in interpolations of the reconstructed numerical solution $\boldsymbol{u}(t)$ in $[t_n, t_{n+1}]$ between one iteration and the next one.

We start by $\underline{\boldsymbol{u}}^{(0)} := (\boldsymbol{u}_n, \boldsymbol{u}_n)^T \in X^{(0)} = \mathbb{R}^{2\times Q}$ associated to two subtimenodes, yielding $O(\Delta t)$-accuracy, and we perform the standard update (26) with structures associated to two subtimenodes

$$\underline{\boldsymbol{u}}^{(1)} = \underline{\boldsymbol{u}}^{(1)}_n + \Delta t \left(B^{(1)}\right)^{-1} \Lambda^{(1)} \underline{\boldsymbol{G}}(\underline{\boldsymbol{u}}^{(0)}) \quad \text{with} \quad \underline{\boldsymbol{u}}^{(1)}_n := \begin{pmatrix} \boldsymbol{u}_n \\ \boldsymbol{u}_n \end{pmatrix} \in \mathbb{R}^{2\times Q}. \tag{29}$$

We get $\underline{\boldsymbol{u}}^{(1)} \in X^{(1)} = \mathbb{R}^{2\times Q}$, corresponding to two subtimenodes and first order accurate, which allows to get an $O(\Delta t^2)$-accurate linear reconstruction of the numerical solution in $[t_n, t_{n+1}]$. Then, we perform the embedding and, by a simple interpolation of the linear reconstruction, via a suitable interpolation matrix $H^{(1)}$, we get

$$\underline{\boldsymbol{u}}^{*(1)} := H^{(1)} \underline{\boldsymbol{u}}^{(1)} \in X^{(2)} = \mathbb{R}^{3\times Q}, \tag{30}$$

corresponding to three equispaced subtimenodes, still $O(\Delta t^2)$-accurate. For the next iteration, $\underline{\boldsymbol{u}}^{*(1)}$ will be the starting point to compute $\underline{\boldsymbol{u}}^{(2)}$ with structures associated to three subtimenodes. Iteratively, at the generic iteration $p > 1$, we pass from $\underline{\boldsymbol{u}}^{(p-1)} \in X^{(p-1)} = \mathbb{R}^{p\times Q}$ to $\underline{\boldsymbol{u}}^{(p)} \in X^{(p)} = \mathbb{R}^{(p+1)\times Q}$ through an interpolation and a standard ADER iteration

$$\begin{cases} \underline{\boldsymbol{u}}^{*(p-1)} := H^{(p-1)} \underline{\boldsymbol{u}}^{(p-1)} \in X^{(p)} = \mathbb{R}^{(p+1)\times Q}, \\ \underline{\boldsymbol{u}}^{(p)} = \underline{\boldsymbol{u}}^{(p)}_n + \Delta t \left(B^{(p)}\right)^{-1} \Lambda^{(p)} \underline{\boldsymbol{G}}(\underline{\boldsymbol{u}}^{*(p-1)}) = \underline{\boldsymbol{u}}^{(p)}_n + \Delta t \left(B^{(p)}\right)^{-1} \Lambda^{(p)} \underline{\boldsymbol{G}}(H^{(p-1)} \underline{\boldsymbol{u}}^{(p-1)}), \end{cases} \tag{31}$$

with $\underline{\boldsymbol{u}}^{(p)} \in X^{(p)} = \mathbb{R}^{(p+1)\times Q}$ being p-th order accurate and associated to $p + 1$ equispaced subtimenodes. The interpolation matrices $H^{(p-1)}$ are defined by $H^{(p-1)}_{\ell,m} := \psi^{m,(p-1)}(t^{\ell,(p)})$ and the vector $\underline{\boldsymbol{u}}^{(p)}_n$ has $p + 1$ components equal to $\boldsymbol{u}_n$. Clearly, $H^{(p-1)}$ must be block-expanded in the context of a vectorial problem.

Remark 5.2 (On the optimal number of iterations)**.** *In such context, it is worth observing that $p + 1$ subtimenodes could guarantee $(p + 1)$-th order of accuracy. Therefore, if the final number of subtimenodes is fixed to be $M + 1$, we perform M iterations, getting $\underline{\boldsymbol{u}}^{(M)} \in X^{(M)} = \mathbb{R}^{(M+1)\times Q}$, plus one final iteration without interpolation, obtaining $\underline{\boldsymbol{u}}^{(M+1)} \in X^{(M+1)} = X^{(M)}$, to reach the maximal accuracy associated to such subtimenodes. This holds for equispaced subtimenodes, while in Section 5.4 we will generalize this idea to other subtimenodes types.*

5.2. ADERdu

Contrarily to the previous case, here the embedding is performed on the evolution operator $\boldsymbol{G}(t, \boldsymbol{u}(t))$, rather than on $\boldsymbol{u}(t)$. The name of the method is given to remark that the embedding is performed on the time derivative of the variable $\boldsymbol{u}$.

The iteration process is very similar to before and, starting again by two subtimenodes and a first iteration without interpolation, at the general iteration $p > 1$ we have

$$\begin{cases} \underline{\boldsymbol{G}}^{*(p-1)} := H^{(p-1)}\underline{\boldsymbol{G}}(\underline{\boldsymbol{u}}^{(p-1)}) \in \mathbb{R}^{(p+1)\times Q}, \\ \underline{\boldsymbol{u}}^{(p)} = \underline{\boldsymbol{u}}_n^{(p)} + \Delta t \left(B^{(p)}\right)^{-1} \Lambda^{(p)}\underline{\boldsymbol{G}}^{*(p-1)} = \underline{\boldsymbol{u}}_n^{(p)} + \Delta t \left(B^{(p)}\right)^{-1} \Lambda^{(p)} H^{(p-1)}\underline{\boldsymbol{G}}(\underline{\boldsymbol{u}}^{(p-1)}), \end{cases} \tag{32}$$

where $\underline{\boldsymbol{u}}^{(p)}$ is again p-th order accurate.

Also in this case, Remark 5.2 holds and, if the final number of subtimenodes is fixed to be $M + 1$, then $P = M + 1$ iterations are recommended without interpolation at the last iteration. In particular, the final iteration without interpolation saturates the $(M + 1)$-th order of accuracy associated to $M + 1$ equispaced subtimenodes.

5.3. ADER-L^2

Also for this efficient ADER, the embedding is not performed on $\boldsymbol{u}(t)$, but on $\boldsymbol{G}(t, \boldsymbol{u}(t))$. The difference with ADERdu is that in ADER-L^2 the embedding consists in a Galerkin projection of the evolution operator from the polynomial space corresponding to $X^{(p-1)}$ to the polynomial space associated to $X^{(p)}$.

For the general iteration $p > 1$, we directly modify the discretization of the ADER-IWF (5) by looking for the new solution $\boldsymbol{u}_h^{(p)}(t)$ into the polynomial space of $X^{(p)}$, with test functions belonging as well to this space, while $\boldsymbol{G}(t, \boldsymbol{u}(t))$ is still represented in the space $X^{(p-1)}$, leading to

$$\begin{aligned} \sum_{m=0}^{M^{(p)}} \left[\psi^{\ell,(p)}(t_{n+1})\psi^{m,(p)}(t_{n+1}) - \int_{t_n}^{t_{n+1}} \left(\frac{d}{dt}\psi^{\ell,(p)}(t) \right) \psi^{m,(p)}(t) dt \right] \boldsymbol{u}^{m,(p)} - \psi^{\ell,(p)}(t_n)\boldsymbol{u}_n \\ - \sum_{m=0}^{M^{(p-1)}} \left(\int_{t_n}^{t_{n+1}} \psi^{\ell,(p)}(t)\psi^{m,(p-1)}(t) dt \right) \boldsymbol{G}(t^{m,(p-1)}, \boldsymbol{u}^{m,(p-1)}) = \boldsymbol{0}, \quad \ell = 0, \dots, M^{(p)}. \end{aligned} \tag{33}$$

System (33) can be written in matricial form as

$$B^{(p)}\underline{\boldsymbol{u}}^{(p)} - \underline{\boldsymbol{r}}^{(p)} - \Delta t \Lambda^{(p,p-1)}\underline{\boldsymbol{G}}(\underline{\boldsymbol{u}}^{(p-1)}) = \boldsymbol{0}, \tag{34}$$

where the matrix $\Lambda^{(p,p-1)}$ and the vector $\underline{\boldsymbol{r}}^{(p)}$ are defined by

$$\Lambda_{\ell,m}^{(p,p-1)} := \frac{1}{\Delta t} \int_{t_n}^{t_{n+1}} \psi^{\ell,(p)}(t)\psi^{m,(p-1)}(t) dt = \int_0^1 \widehat{\psi}^{\ell,(p)}(\xi)\widehat{\psi}^{m,(p-1)}(\xi) d\xi, \quad \underline{\boldsymbol{r}}_\ell^{(p)} := \psi^{\ell,(p)}(t_n)\boldsymbol{u}_n. \tag{35}$$

Inverting $B^{(p)}$ and making use of Proposition 2.4, we get

$$\underline{\boldsymbol{u}}^{(p)} := \underline{\boldsymbol{u}}_n^{(p)} + \Delta t \left(B^{(p)}\right)^{-1} \Lambda^{(p,p-1)}\underline{\boldsymbol{G}}(\underline{\boldsymbol{u}}^{(p-1)}). \tag{36}$$

In this formulation, the embedding is naturally realized by the mismatch between the spaces of $\boldsymbol{G}(t, \boldsymbol{u}(t))$ and of the test functions, without any need for additional structures.

Again, following Remark 5.2, we underline that a final extra iteration of the standard method without embedding is used for a fixed final number of subtimenodes to get the optimal accuracy.

We show the equivalence of ADERdu and ADER-L^2 in the next theorem. However, despite being equivalent in the context of ADER methods for ODEs, the two modifications have a deeply different nature and, hence, their generalization to the context of ADER methods for hyperbolic PDEs leads to different families of schemes. This is why both the approaches have been described.

Proposition 5.3 (Equivalence between ADERdu and ADER-L^2)**.** *The methods ADERdu* (32) *and ADER-L^2* (36) *are equivalent.*

Proof. In both cases the first iteration is the one of the standard method with two subtimenodes, therefore, let us focus on the generic iteration $p > 1$. The updates of the two methods, respectively given in (32) and (36), are clearly equivalent if $\Lambda^{(p)}H^{(p-1)} = \Lambda^{(p,p-1)}$. Exploiting the definition of $\Lambda^{(p)}H^{(p-1)}$, we get

$$\begin{aligned} \left(\Lambda^{(p)}H^{(p-1)}\right)_{\ell,m} &= \sum_{k=1}^{M^{(p)}} \int_0^1 \widehat{\psi}^{\ell,(p)}(\xi)\widehat{\psi}^{k,(p)}(\xi) d\xi\, \widehat{\psi}^{m,(p-1)}(\xi^{k,(p)}) \\ &= \int_0^1 \widehat{\psi}^{\ell,(p)}(\xi) \left(\sum_{k=1}^{M^{(p)}} \widehat{\psi}^{k,(p)}(\xi)\widehat{\psi}^{m,(p-1)}(\xi^{k,(p)}) \right) d\xi = \int_0^1 \widehat{\psi}^{\ell,(p)}(\xi)\widehat{\psi}^{m,(p-1)}(\xi) d\xi, \end{aligned} \tag{37}$$

which is the definition of $\Lambda_{\ell,m}^{(p,p-1)}$. In the previous computations, we have used the fact that the term in parenthesis is the exact interpolation of $\psi^{m,(p-1)}(t) \in \mathbb{P}_{p-1}$ into the polynomial space $\mathbb{P}_p$ defined by the Lagrange basis functions $\psi^{k,(p)}$ (modulo a remapping into $[0,1]$). Clearly, a final iteration of the standard method in the spirit of Remark 5.2, to achieve the optimal accuracy, does not spoil the equivalence. □

5.4. ADERu, ADERdu and ADER-L^2 for other choices of subtimenodes

As already said, the order N of a standard ADER method for ODEs depends on the distribution of the adopted subtimenodes. The modified methods for a general distribution, e.g., GLB or GLG, with a fixed number of subtimenodes equal to $M+1$, are hence constructed as follows. We start with two subtimenodes and we proceed like described in the equispaced case: we perform the first iteration of the standard method and we continue with iterations of the modified method ((31), (32) or (36)) increasing the number of subtimenodes until the iteration $p = M$ corresponding to $M+1$ subtimenodes and order M. At this point, we continue with $N - M$ iterations of the standard method to saturate the accuracy of the adopted distribution.

On the other hand, in order to reach a specific order P of accuracy in the most efficient way, we select M as the minimal integer such that $M+1$ subtimenodes guarantee the order of the ADER method to be $N \geq P$. This can be done for example for GLB nodes with $M = \left\lceil \frac{P}{2} \right\rceil$ and $M = \max\left(\left\lceil \frac{P-1}{2} \right\rceil, 1\right)$ for GLG. Then, we perform M iterations to reach $M+1$ subtimenodes and, further, we perform $P - M$ final iterations of the standard method to reach the desired accuracy.

We conclude this little section with some useful considerations. First, it is mandatory to start with at least two subtimenodes because a single node (in principle admissible for GLG) would not be enough to guarantee, after the first iteration, a first order reconstruction of the numerical solution, hence, spoiling the accuracy. Secondly, the embeddings must be performed starting already from the first iterations and it is not possible to postpone them after the saturation of the accuracy associated to an intermediate set of subtimenodes used along the iteration process. The reason is given by the fact that the p-th embedding is only p-th order accurate, i.e., the interpolation with $p+1$ subtimenodes is associated to a local truncation error $O(\Delta t^{p+1})$, and using it at later iterations would spoil the accuracy.

5.5. New adaptive ADER methods

The novel ADER methods can be easily further modified to design p-adaptive schemes.

From Theorem 5.1, we have that the iteration process yields a numerical solution $\underline{\boldsymbol{u}}^{(p)}$, which gains one order of accuracy at each iteration, with no saturation due to a fixed operator $\mathcal{L}_\Delta^2$. Under smoothness assumptions on the exact solution, we can therefore perform the iteration process, increasing the number of subtimenodes and hence the order of accuracy, until convergence up to a user defined tolerance ε. If we define the approximation of $\boldsymbol{u}_{n+1}$ obtained at the p-th iteration as $\boldsymbol{u}_{n+1}^{(p)} := \sum_{m=0}^{M^{(p)}} \boldsymbol{u}^{m,(p)} \psi^{m,(p)}(t_{n+1})$, we can set as stopping criterion

$$\frac{\left\| \boldsymbol{u}_{n+1}^{(p)} - \boldsymbol{u}_{n+1}^{(p-1)} \right\|}{\left\| \boldsymbol{u}_{n+1}^{(p)} \right\|} \leq \varepsilon. \tag{38}$$

A remarkable aspect is that p-adaptivity is naturally embedded in the described formulation and that no effort, other than a simple check at each iteration, is required to implement such feature. This is an interesting aspect of the novel methods and of the general approach introduced in [47, 48], which is particularly desirable in the context of real-world applications. Indeed, the final users want the error to be smaller than a predefined tolerance and to reduce the need for computational resources as much as possible. Per se, the adoption of high order methods does not guarantee the optimal balance. In fact, depending on the level of mesh refinement and on the prescribed tolerance, low order methods may actually be sufficient and cheaper than high order methods. The proposed approach is able to automatically detect the required order of accuracy. Furthermore, some of the authors are involved in other projects whose goal is to investigate such strategy in the context of non-smooth problems.

Let us finally remark that different criteria, other than convergence, can be chosen to halt the iterative process. For example, in [48], in a PDE context, the ADER iterations are stopped if the obtained numerical solution does not fulfill some physical constraints.

6. ADER as Runge–Kutta methods

In this section, we will show how the described ADER methods can be rewritten as explicit RK methods, i.e., in the form (12) with a strictly lower-triangular matrix A of the coefficients $a_{s,r}$. We will define their Butcher tableaux and we will study their linear stability.

We introduce here the iteration-dependent vectors of the subtimenodes in the reference inteval $[0, 1]$ as $\underline{\beta}^{(p)} := \left(\xi^{0,(p)}, \ldots, \xi^{M(p),(p)}\right)^T$ for the new modified methods, with $\xi^{m,(p)} := \frac{t^{m,(p)} - t_n}{\Delta t}$.

In all cases, for the sake of efficiency, the first iteration is replaced by a simple Euler step: this reduces the number of stages associated to the first iteration and does not spoil the accuracy. In fact, the first iteration is supposed to provide a first order accurate approximation.

We construct the tableaux for a general distribution of subtimenodes. For a fixed accuracy order P, we consider P iterations, assuming a final number of subtimenodes, which is chosen to be the minimal (and so the optimal) allowing to reach such accuracy, according to the theoretical analysis presented in Section 3, i.e., $M = P - 1$ for equispaced subtimenodes, $M = \left\lceil \frac{P}{2} \right\rceil$ for GLB subtimenodes and $M = \max\left(\left\lceil \frac{P-1}{2} \right\rceil, 1\right)$ for GLG subtimenodes.

The Butcher tableaux associated to the described ADER methods are reported in Tables 1, 5 and 6, while, the number of RK stages is reported in Tables 2, 3 and 4. The RK matrices A of the ADER methods under investigation have a block-diagonal structure, with each block being associated to an ADER iteration. Some of the computed rows of the RK A matrices have only zero elements, leading to "ghost" stages that do not contribute to the method. These rows have not been considered in the computation of the number of RK stages in Tables 2, 3 and 4.

In tables 2, 3 and 4, we report the *theoretical speed-ups* of ADERu and ADERdu (equivalent to ADER-L^2) with respect to the original ADER method without interpolations, simply denoted as "ADER". Further, for GLB and GLG subtimenodes, we also report the *theoretical speed-ups* of ADER, ADERu and ADERdu with respect to the non-optimal ADER method using a number of subtimenodes equal to the desired order of accuracy, as in [39, 63, 64, 67, 29, 53, 22], referring to such method as "classical ADER" (cADER). Such method makes use of an unnecessarily large number of subtimenodes at each iteration, indeed, as proven in Section 3, GLB and GLG subtimenodes can achieve the same order with nearly half of the subtimenodes.

The *theoretical speed-ups* are computed as the ratios between the number of RK stages of the reference methods over the ones of the novel methods. Therefore, a *theoretical speed-up* larger than one implies that the investigated scheme is, in theory, faster than the reference one. In Section 8, the *theoretical speed-ups* will be compared with the *numerical speed-ups*, defined as the ratios between the wall-clock computational times of the numerical simulations. Even if other factors might come into play, e.g., memory ordering or floating point operations of matrix multiplications between ADER structures, the two quantities should be highly correlated. This is also due to the fact that, for all the methods, all the matrices can be efficiently precomputed at the beginning of the simulation, as they do not depend on the specific time iteration.

6.1. ADER

We start by recalling the definition of the general ADER iteration (26)

$$\underline{\boldsymbol{u}}^{(p)} := \underline{\boldsymbol{u}}_n + \Delta t B^{-1} \Lambda \underline{\boldsymbol{G}}(\underline{\boldsymbol{u}}^{(p-1)}). \tag{39}$$

Collecting all the iterations and identifying each state $\underline{\boldsymbol{u}}^{m,(p)}$, associated to the subtimenode t^m, of $\underline{\boldsymbol{u}}^{(p)}$ as a RK stage $\boldsymbol{y}^s$, we get the RK formulation (12) characterized by the Butcher tableau reported in Table 1. The vector $\widetilde{\boldsymbol{b}}$ is defined as $\widetilde{\boldsymbol{b}}^T := \left(\underline{\widehat{\psi}}(1)\right)^T B^{-1}\Lambda$, with $\underline{\widehat{\psi}}(\xi) := (\widehat{\psi}^0(\xi), \ldots, \widehat{\psi}^M(\xi))^T$, and keeps into account the final ADER iteration and the interpolation. In fact, recalling that $\sum_{m=0}^{M} \widehat{\psi}^m \equiv 1$, we have

$$\boldsymbol{u}_{n+1} = \left(\underline{\widehat{\psi}}(1)\right)^T \underline{\boldsymbol{u}}^{(P)} = \left(\underline{\widehat{\psi}}(1)\right)^T \left[\underline{\boldsymbol{u}}_n + \Delta t B^{-1} \Lambda \underline{\boldsymbol{G}}(\underline{\boldsymbol{u}}^{(P-1)})\right] = \boldsymbol{u}_n + \Delta t \left(\underline{\widehat{\psi}}(1)\right)^T B^{-1} \Lambda \underline{\boldsymbol{G}}(\underline{\boldsymbol{u}}^{(P-1)}). \tag{40}$$

The number of stages is equal to $S = 1 + (P - 1)(M + 1)$ and is reported, for equispaced, GLB and GLG subtimenodes, in Tables 2, 3 and 4, neglecting the "ghost" stages. For cADER the same relation holds, but the number of subtimenodes is always $M + 1$ with $M = P - 1$ independently of the chosen subtimenodes.

We remark that, for equispaced subtimenodes, ADER and cADER coincide. Instead, for GLB and GLG, the adoption of the optimal number of subtimenodes for a given order, i.e., the choice of ADER over cADER, determines a substantial decrease in the number of stages, even without extra interpolation processes.

6.2. ADERu

Again, we recall the update at the generic iteration $p > 1$ characterized by interpolation of $\boldsymbol{u}(t)$

$$\begin{cases} \underline{\boldsymbol{u}}^{*(p-1)} := H^{(p-1)} \underline{\boldsymbol{u}}^{(p-1)}, \\ \underline{\boldsymbol{u}}^{(p)} = \underline{\boldsymbol{u}}_n^{(p)} + \Delta t \left(B^{(p)}\right)^{-1} \Lambda^{(p)} \underline{\boldsymbol{G}}(\underline{\boldsymbol{u}}^{*(p-1)}). \end{cases} \tag{41}$$

$\boldsymbol{c}$	$\boldsymbol{u}_n$	$\underline{\boldsymbol{u}}^{(1)}$	$\underline{\boldsymbol{u}}^{(2)}$	$\underline{\boldsymbol{u}}^{(3)}$	$\cdots$	$\underline{\boldsymbol{u}}^{(P-2)}$	$\underline{\boldsymbol{u}}^{(P-1)}$	A
0	0							$\boldsymbol{u}_n$
$\underline{\beta}$	$\underline{\beta}$	$\underline{\underline{0}}$						$\underline{\boldsymbol{u}}^{(1)}$
$\underline{\beta}$	$\underline{0}$	$B^{-1}\Lambda$	$\underline{\underline{0}}$					$\underline{\boldsymbol{u}}^{(2)}$
$\underline{\beta}$	$\underline{0}$	$\underline{0}$	$B^{-1}\Lambda$	$\underline{\underline{0}}$				$\underline{\boldsymbol{u}}^{(3)}$
	$\vdots$	$\vdots$		$\ddots$	$\ddots$			$\vdots$
	$\vdots$	$\vdots$			$\ddots$	$\ddots$		$\vdots$
$\underline{\beta}$	$\underline{0}$	$\underline{0}$	$\cdots$	$\cdots$	$\underline{\underline{0}}$	$B^{-1}\Lambda$	$\underline{\underline{0}}$	$\underline{\boldsymbol{u}}^{(P-1)}$
$\boldsymbol{b}$	0	$\underline{0}$	$\cdots$	$\cdots$	$\cdots$	$\underline{0}$	$\widetilde{\boldsymbol{b}}^T$	$\boldsymbol{u}_{n+1}$

Table 1: Butcher tableau of the original ADER method, $\boldsymbol{c}$ at the left, $\boldsymbol{b}$ at the bottom, A in the middle. References to the stages are reported on top and on the right side

Param		RK Stages			cADER/ADER-speed-up	
P	M	cADER/ADER	ADERu	ADERdu	ADERu	ADERdu
2	1	2	2	2	1.000	1.000
3	2	6	6	4	1.000	1.500
4	3	12	11	7	1.091	1.714
5	4	20	17	11	1.176	1.818
6	5	30	24	16	1.250	1.875
7	6	42	32	22	1.312	1.909
8	7	56	41	29	1.366	1.931
9	8	73	51	37	1.431	1.973
10	9	90	62	46	1.452	1.957
11	10	111	74	56	1.500	1.982
12	11	133	87	67	1.529	1.985
13	12	156	101	79	1.545	1.975
14	13	183	116	92	1.578	1.989

Table 2: Number of stages S for various ADER with equispaced subtimenodes and *theoretical speed-ups* with respect to ADER/cADER method computed as the ratio of the number of RK stages of ADER method over the number of RK stages of the method of interest

Param		RK Stages				cADER-speed-up			ADER-speed-up	
P	M	cADER	ADER	ADERu	ADERdu	ADER	ADERu	ADERdu	ADERu	ADERdu
2	1	2	2	2	2	1.000	1.000	1.000	1.000	1.000
3	2	6	6	6	4	1.000	1.000	1.500	1.000	1.500
4	2	12	9	9	7	1.333	1.333	1.714	1.000	1.286
5	3	20	16	15	11	1.250	1.333	1.818	1.067	1.455
6	3	30	20	19	15	1.500	1.579	2.000	1.053	1.333
7	4	43	30	27	21	1.433	1.593	2.048	1.111	1.429
8	4	56	35	32	26	1.600	1.750	2.154	1.094	1.346
9	5	73	48	42	34	1.521	1.738	2.147	1.143	1.412
10	5	91	54	48	40	1.685	1.896	2.275	1.125	1.350
11	6	110	71	60	50	1.549	1.833	2.200	1.183	1.420
12	6	133	78	67	57	1.705	1.985	2.333	1.164	1.368
13	7	156	96	81	69	1.625	1.926	2.261	1.185	1.391
14	7	183	104	89	77	1.760	2.056	2.377	1.169	1.351

Table 3: Number of stages S for various ADER with GLB subtimenodes and *theoretical speed-ups* with respect to cADER and to ADER computed as the ratio of the number of RK stages of the method of cADER or ADER, respectively, over the number of RK stages of the method of interest

Param		RK Stages				cADER-speed-up			ADER-speed-up	
P	M	cADER	ADER	ADERu	ADERdu	ADER	ADERu	ADERdu	ADERu	ADERdu
2	1	3	3	3	3	1.000	1.000	1.000	1.000	1.000
3	1	7	5	5	5	1.400	1.400	1.400	1.000	1.000
4	2	13	10	10	9	1.300	1.300	1.444	1.000	1.111
5	2	21	13	13	12	1.615	1.615	1.750	1.000	1.083
6	3	31	21	20	18	1.476	1.550	1.722	1.050	1.167
7	3	43	25	24	22	1.720	1.792	1.955	1.042	1.136
8	4	57	36	33	30	1.583	1.727	1.900	1.091	1.200
9	4	73	41	38	35	1.780	1.921	2.086	1.079	1.171
10	5	91	55	49	45	1.655	1.857	2.022	1.122	1.222
11	5	111	61	55	51	1.820	2.018	2.176	1.109	1.196
12	6	133	78	68	63	1.705	1.956	2.111	1.147	1.238
13	6	157	85	75	70	1.847	2.093	2.243	1.133	1.214
14	7	183	105	90	84	1.743	2.033	2.179	1.167	1.250

Table 4: Number of stages S for various ADER with GLG subtimenodes and *theoretical speed-ups* with respect to cADER and to ADER computed as the ratio of the number of RK stages of cADER or ADER, respectively, over the number of RK stages of the method of interest

It is possible to formulate the previous update in terms of the interpolated states by multiplying the second equation by $H^{(p)}$

$$\underline{\boldsymbol{u}}^{*(p)} = H^{(p)}\underline{\boldsymbol{u}}^{(p)} = \underline{\boldsymbol{u}}_n^{(p+1)} + \Delta t H^{(p)}\left(B^{(p)}\right)^{-1}\Lambda^{(p)}\underline{\boldsymbol{G}}(\underline{\boldsymbol{u}}^{*(p-1)}), \tag{42}$$

where we have exploited the fact that the sum of the elements in each row of the interpolation matrix $H^{(p)}$ is equal to 1. The Butcher tableau can be therefore constructed as in Table 5. The matrices $W^{(p)}$ are defined as

$$W^{(p)} := \begin{cases} H^{(p)}\left(B^{(p)}\right)^{-1}\Lambda^{(p)} \in \mathbb{R}^{(p+2)\times(p+1)}, & \text{if } p = 2,\dots,M-1, \\ \left(B^{(M)}\right)^{-1}\Lambda^{(M)} \in \mathbb{R}^{(M+1)\times(M+1)}, & \text{if } p = M. \end{cases} \tag{43}$$

Notice that we do not have interpolation in the final iterations. We perform the interpolations in the early iterations in order to reach the needed number of subtimenodes that allows to get the desired order of accuracy P, then, from $\boldsymbol{u}^{(M)}$ on, we keep iterating using the structures of the standard ADER method with $M+1$ subtimenodes. Vector $\widetilde{\boldsymbol{b}}^{(M)}$ is, in fact, defined as $\widetilde{\boldsymbol{b}}$ considering the final number of subtimenodes. The number of stages is equal to $S = 1 + (P-1)(M+1) - \frac{(M-1)(M-2)}{2}$, i.e., $\frac{(M-1)(M-2)}{2}$ less with respect to ADER, and is reported, for equispaced, GLB and GLG subtimenodes, in Tables 2, 3 and 4. We report also the *theoretical speed-up* factor with respect to ADER and cADER.

6.3. ADERdu and ADER-L^2

Since ADERdu and ADER-L^2 coincide in an ODE context, they are characterized by the same RK form. In order to obtain it and to construct the associated Butcher tableau, we recall the general update (32)

$$\underline{\boldsymbol{u}}^{(p)} = \underline{\boldsymbol{u}}_n^{(p)} + \Delta t\left(B^{(p)}\right)^{-1}\Lambda^{(p)}H^{(p-1)}\underline{\boldsymbol{G}}(\underline{\boldsymbol{u}}^{(p-1)}), \tag{44}$$

leading to the Butcher tableau reported in Table 6. The matrices $Z^{(p)}$ are defined as

$$Z^{(p)} := \begin{cases} \left(B^{(p)}\right)^{-1}\Lambda^{(p)}H^{(p-1)} \in \mathbb{R}^{(p+1)\times p}, & \text{if } p = 2,\dots,M, \\ \left(B^{(M)}\right)^{-1}\Lambda^{(M)} \in \mathbb{R}^{(M+1)\times(M+1)}, & \text{if } p = M+1. \end{cases} \tag{45}$$

Same considerations as the ones made in the context of the ADERu methods apply here: the final iterations are performed without interpolations to reach the desired accuracy. The number of stages is equal to $S = 1 + (P-1)(M+1) - \frac{M(M-1)}{2}$, i.e., $\frac{M(M-1)}{2}$ less with respect to ADER. Again, it can be found for equispaced, GLB and GLG subtimenodes, in Tables 2, 3 and 4, together with the *theoretical speed-up* with respect to ADER and cADER.

Before studying the linear stability of the methods, as an example and for the sake of completeness, we report, in Figure 1, the sparsity pattern of the A matrices of the cADER, ADER, ADERu and ADERdu (equivalent to

$\boldsymbol{c}$	$\boldsymbol{u}_n$	$\underline{\boldsymbol{u}}^{*(1)}$	$\underline{\boldsymbol{u}}^{*(2)}$	$\underline{\boldsymbol{u}}^{*(3)}$	$\cdots$	$\underline{\boldsymbol{u}}^{*(M-2)}$	$\underline{\boldsymbol{u}}^{*(M-1)}$	$\underline{\boldsymbol{u}}^{(M)}$	$\underline{\boldsymbol{u}}^{(M+1)}$	$\cdots$	$\underline{\boldsymbol{u}}^{(P-2)}$	$\underline{\boldsymbol{u}}^{(P-1)}$	A
0	0												$\boldsymbol{u}_n$
$\underline{\beta}^{(2)}$	$\underline{\beta}^{(2)}$	$\underline{\underline{0}}$											$\underline{\boldsymbol{u}}^{*(1)}$
$\underline{\beta}^{(3)}$	$\underline{0}$	$\underline{\underline{W}}^{(2)}$	$\underline{\underline{0}}$										$\underline{\boldsymbol{u}}^{*(2)}$
$\underline{\beta}^{(4)}$	$\underline{0}$	$\underline{\underline{0}}$	$\underline{\underline{W}}^{(3)}$	$\underline{\underline{0}}$									$\underline{\boldsymbol{u}}^{*(3)}$
	$\vdots$	$\vdots$	$\ddots$	$\ddots$	$\ddots$								$\vdots$
	$\vdots$	$\vdots$		$\ddots$	$\ddots$	$\ddots$							$\vdots$
$\underline{\beta}^{(M)}$	$\underline{0}$	$\underline{\underline{0}}$	$\cdots$	$\cdots$	$\underline{\underline{0}}$	$\underline{\underline{W}}^{(M-1)}$	$\underline{\underline{0}}$						$\underline{\boldsymbol{u}}^{*(M-1)}$
$\underline{\beta}^{(M)}$	$\underline{0}$	$\underline{\underline{0}}$	$\cdots$	$\cdots$	$\cdots$	$\underline{\underline{0}}$	$\underline{\underline{W}}^{(M)}$	$\underline{\underline{0}}$					$\underline{\boldsymbol{u}}^{(M)}$
$\underline{\beta}^{(M)}$	$\underline{0}$	$\underline{\underline{0}}$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\underline{\underline{0}}$	$\underline{\underline{W}}^{(M)}$	$\underline{\underline{0}}$				$\underline{\boldsymbol{u}}^{(M+1)}$
$\vdots$	$\vdots$	$\vdots$						$\ddots$	$\ddots$	$\ddots$			$\vdots$
$\vdots$	$\vdots$	$\vdots$							$\ddots$	$\ddots$	$\ddots$		$\vdots$
$\underline{\beta}^{(M)}$	$\underline{0}$	$\underline{\underline{0}}$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$		$\underline{\underline{0}}$	$\underline{\underline{W}}^{(M)}$	$\underline{\underline{0}}$	$\underline{\boldsymbol{u}}^{(P-1)}$
$\boldsymbol{b}$	0	$\underline{0}$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\underline{0}$	$(\widetilde{\boldsymbol{b}}^{(M)})^T$	$\boldsymbol{u}_{n+1}$

Table 5: Butcher tableau of the ADERu method, $\boldsymbol{c}$ at the left, $\boldsymbol{b}$ at the bottom, A in the middle. References to the stages are reported on top and on the right side

$\boldsymbol{c}$	$\boldsymbol{u}_n$	$\underline{\boldsymbol{u}}^{(1)}$	$\underline{\boldsymbol{u}}^{(2)}$	$\underline{\boldsymbol{u}}^{(3)}$	$\cdots$	$\underline{\boldsymbol{u}}^{(M-2)}$	$\underline{\boldsymbol{u}}^{(M-1)}$	$\underline{\boldsymbol{u}}^{(M)}$	$\underline{\boldsymbol{u}}^{(M+1)}$	$\cdots$	$\underline{\boldsymbol{u}}^{(P-2)}$	$\underline{\boldsymbol{u}}^{(P-1)}$	A
0	0												$\boldsymbol{u}_n$
$\underline{\beta}^{(1)}$	$\underline{\beta}^{(1)}$	$\underline{\underline{0}}$											$\underline{\boldsymbol{u}}^{(1)}$
$\underline{\beta}^{(2)}$	$\underline{0}$	$\underline{\underline{Z}}^{(2)}$	$\underline{\underline{0}}$										$\underline{\boldsymbol{u}}^{(2)}$
$\underline{\beta}^{(3)}$	$\underline{0}$	$\underline{\underline{0}}$	$\underline{\underline{Z}}^{(3)}$	$\underline{\underline{0}}$									$\underline{\boldsymbol{u}}^{(3)}$
	$\vdots$	$\vdots$	$\ddots$	$\ddots$	$\ddots$								$\vdots$
	$\vdots$	$\vdots$		$\ddots$	$\ddots$	$\ddots$							$\vdots$
$\underline{\beta}^{(M-1)}$	$\underline{0}$	$\underline{\underline{0}}$	$\cdots$	$\cdots$	$\underline{\underline{0}}$	$\underline{\underline{Z}}^{(M-1)}$	$\underline{\underline{0}}$						$\underline{\boldsymbol{u}}^{(M-1)}$
$\underline{\beta}^{(M)}$	$\underline{0}$	$\underline{\underline{0}}$	$\cdots$	$\cdots$	$\cdots$	$\underline{\underline{0}}$	$\underline{\underline{Z}}^{(M)}$	$\underline{\underline{0}}$					$\underline{\boldsymbol{u}}^{(M)}$
$\underline{\beta}^{(M)}$	$\underline{0}$	$\underline{\underline{0}}$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\underline{\underline{0}}$	$\underline{\underline{Z}}^{(M+1)}$	$\underline{\underline{0}}$				$\underline{\boldsymbol{u}}^{(M+1)}$
$\vdots$	$\vdots$	$\vdots$						$\ddots$	$\ddots$	$\ddots$			$\vdots$
$\vdots$	$\vdots$	$\vdots$							$\ddots$	$\ddots$	$\ddots$		$\vdots$
$\underline{\beta}^{(M)}$	$\underline{0}$	$\underline{\underline{0}}$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$		$\underline{\underline{0}}$	$\underline{\underline{Z}}^{(M+1)}$	$\underline{\underline{0}}$	$\underline{\boldsymbol{u}}^{(P-1)}$
$\boldsymbol{b}$	0	$\underline{0}$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\underline{0}$	$(\widetilde{\boldsymbol{b}}^{(M)})^T$	$\boldsymbol{u}_{n+1}$

Table 6: Butcher tableau of the ADERdu and ADER-L^2 method, $\boldsymbol{c}$ at the left, $\boldsymbol{b}$ at the bottom, A in the middle. References to the stages are reported on top and on the right side

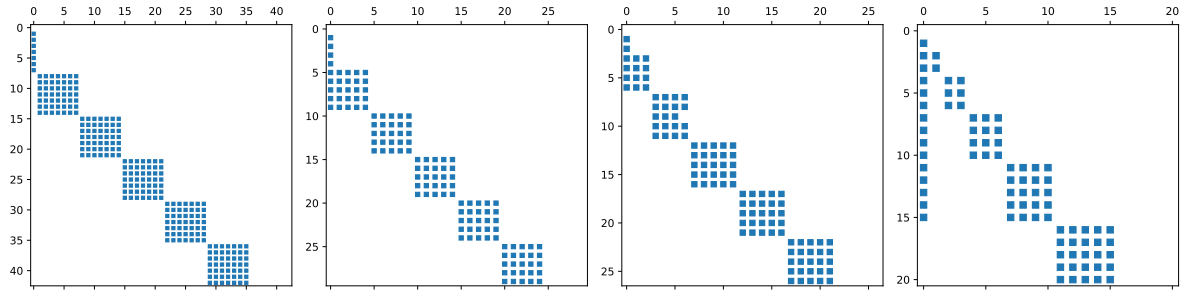

Figure 1: Sparsity pattern of the A matrix with GLB subtimenodes and order 7. From left to right: cADER, ADER, ADERu and ADERdu (equivalent to ADER-L^2). The references to the stages indices are reported on the left and on top. In cADER the blocks have size 7, in ADER they have size 5, in ADERu and ADERdu they have increasing sizes from 2 to 5

ADER-L^2) methods for order 7 and GLB subtimenodes. The block-structure of all matrices is essentially identical, with each block corresponding to one iteration, however, the sizes of the blocks differ a lot. All the blocks in the cADER matrix are larger than the ones in the ADER matrix, as the latter method makes use of the optimal number of subtimenodes. Moreover, for both cADER and ADER, the size of the blocks is always the same, instead, for ADERu and ADERdu, the size of the blocks increases along the iterations, as a result of the interpolations. We remark that the non-zero entries in the first column of the ADERdu matrix are determined by the ghost stages, corresponding, in practice, to the first stage $\boldsymbol{u}_n$.

6.4. Linear stability

The linear stability of a RK method is studied by analyzing the asymptotic behavior of the numerical solution of the method applied to Dahlquist's equation $\frac{d}{dt}u(t) = \lambda u(t)$, where $\lambda \in \mathbb{C}$ with $Re(\lambda) < 0$. The linearity of the problem and of the method makes possible to express u_{n+1} as $u_{n+1} = R(\lambda\Delta t)u_n$, where $R(\cdot)$ is the so-called stability function of the method. In the context of the novel methods, the following result holds.

Theorem 6.1. *The stability function of any ADER, ADERu and ADERdu (and so ADER-L^2) method of order P is*

$$R(z) = \sum_{r=0}^{P} \frac{z^r}{r!}, \tag{46}$$

independently of the distribution of the subtimenodes.

The proof of the previous result is identical to the proof of Theorem 6.2 presented in [47] in the context of the study of the linear stability of the bDeC, bDeCu and bDeCdu methods and it is based on the particular block-structure of the RK matrix A of the investigated methods. The reader is referred to [47] for further details. An interesting consequence of Theorem 6.1 is given by the following corollary.

Corollary 6.2. *The bDeC, bDeCu, bDeCdu, ADER, ADERu and ADERdu (and so ADER-L^2) methods of order P share the same stability function, independently of the distribution of the subtimenodes.*

The methods mentioned in the previous result are therefore equivalent on linear problems and characterized by the same stability region.

The explicit characterization of the stability function can be used to find the largest Δt for which the methods are stable. From classical RK analysis [38], one should determine the stability region from the stability function and choose Δt in such a way that all $\Delta t\lambda_i(\boldsymbol{u})$ lie inside the stability region, with $\lambda_i(\boldsymbol{u})$ being the generic eigenvalue of the Jacobian matrix $\frac{\partial \boldsymbol{G}}{\partial \boldsymbol{u}}(\boldsymbol{u})$. We recall that the set of complex numbers $\mathcal{S} := \{z \in \mathbb{C} : |R(z)| < 1\}$ constitutes the stability region of the scheme. For the simple scalar problem $\frac{d}{dt}u = -u$, the bounds for Δt, reported in Table 7, guarantee the stability of the methods. Indeed, for systems with real eigenvalues, the same bounds rescaled by a factor $\frac{1}{\rho(\boldsymbol{u})}$, with $\rho(\boldsymbol{u})$ being the spectral radius of $\frac{\partial \boldsymbol{G}}{\partial \boldsymbol{u}}(\boldsymbol{u})$, guarantee (linear) stability. For more general systems with complex eigenvalues, the choice of Δt is more difficult and one has to take into account the whole stability region and the distribution of the eigenvalues in the complex plane. We remark that, since the methods are equivalent on linear problems, they are subjected to the same linear stability bounds.

Order	Δt	Order	Δt	Order	Δt	Order	Δt
1	2	4	2.79	7	3.95	10	5.07
2	2	5	3.22	8	4.31	11	5.45
3	2.51	6	3.55	9	4.70	12	5.85

Table 7: Stability bounds on Δt for the scalar problem $\frac{d}{dt}u = -u$ for different orders for bDeC, bDeCu and bDeCdu. Up to a simple rescaling by $\frac{1}{\rho(\boldsymbol{u})}$, with $\rho(\boldsymbol{u})$ being the spectral radius of the Jacobian matrix $\frac{\partial \boldsymbol{G}}{\partial \boldsymbol{u}}(\boldsymbol{u})$, the bounds can be assumed for well–posed ODEs with real eigenvalues

7. Application to PDEs with Spectral Difference schemes

In this section, we show how to apply ADER methods in a PDE context with the SD space discretization. Let us consider the monodimensional hyperbolic PDE

$$\frac{\partial}{\partial t}\boldsymbol{u}(x,t) + \frac{\partial}{\partial x}\boldsymbol{F}(\boldsymbol{u}(x,t)) = \boldsymbol{0}, \quad (x,t) \in [x_L, x_R] \times \mathbb{R}_0^+, \tag{47}$$

where $\boldsymbol{u} : [x_L, x_R] \times \mathbb{R}_0^+ \to \mathbb{R}^Q$ is the unknown solution and $\boldsymbol{F} : \mathbb{R}^Q \to \mathbb{R}^Q$ is the flux. We introduce a tessellation $\mathcal{T}_h$ of $[x_L, x_R]$ with non-overlapping segments K and we adopt a classical nodal DG discretization of the numerical solution. Thus, globally the solution is approximated as a discontinuous piecewise polynomial function in $(V_M)^Q$ where $V_M := \left\{ g \in L^2(\Omega) \; s.t. \; g|_K \in \mathbb{P}_M(K) \right\}$, leading to accuracy $M+1$ for sufficiently smooth solutions. Locally, in each element K, we represent the solution by interpolating it in $M+1$ solution points x_i^s

$$\boldsymbol{u}_h(x,t) := \sum_{i=1}^{M+1} \boldsymbol{u}_i^s(t)\varphi_i^s(x), \quad \forall x \in K, \tag{48}$$

where the functions φ_i^s are the Lagrange polynomials of degree M associated to the solution points x_i^s and $\boldsymbol{u}_i^s$ the time-dependent values in the same points. For the sake of compactness, the label K on the local coefficients and basis functions is omitted. In each element, out of the approximation (48), we can reconstruct the flux by interpolating it in $M+2$ flux points, x_i^f. In particular, we set $\boldsymbol{F}_i^f(t) := \boldsymbol{F}(\boldsymbol{u}_h(x_i^f, t))$ and we define

$$\boldsymbol{F}_h(x,t) := \sum_{i=1}^{M+2} \boldsymbol{F}_i^f(t)\varphi_i^f(x), \quad \forall x \in K. \tag{49}$$

In order to guarantee a coupling between the elements K and to avoid the instabilities of central schemes, we include the extrema of each segment K among the flux points x_i^f and we use a numerical flux $\boldsymbol{F}^{num}$, rather than a direct evaluation, to define the flux value $\boldsymbol{F}_i^f(t)$ in such extrema, keeping into account the trace of $\boldsymbol{u}_h$ from the neighboring segments. Hence, the approximated flux is given by

$$\boldsymbol{F}_h(x,t) := \boldsymbol{F}^{num}(x_1^f, t)\varphi_1^f(x) + \sum_{i=2}^{M+1} \boldsymbol{F}_i^f(t)\varphi_i^f(x) + \boldsymbol{F}^{num}(x_{M+2}^f, t)\varphi_{M+2}^f(x), \quad \forall x \in K, \tag{50}$$

where x_1^f and x_{M+2}^f are the extrema of the cell K.

The semidiscretization of the SD method is finally obtained by imposing that the discretizations of the solution and of the flux satisfy the PDE (47) in each solution point

$$\frac{\partial}{\partial t}\boldsymbol{u}_i^s(t) + \frac{\partial}{\partial x}\boldsymbol{F}_h(x_i^s, t) = \boldsymbol{0}, \quad \forall x_i^s \in K, \quad \forall K \in \mathcal{T}_h, \tag{51}$$

which is a system of ODEs, in the unknowns $\boldsymbol{u}_i^s$, that must be solved in time. System (51) is in the form (1) and can be therefore solved with the described explicit ADER methods. We use a Courant–Friedrichs–Lewy condition adapted to the SD scheme, taken from the linear stability analysis presented in [62], and compute the time step as

$$\Delta t = \frac{C}{M+1}\frac{\Delta x}{v_{max}}, \tag{52}$$

for some $C \in (0, 1]$, with v_{max} being the maximum wave speed of the system in absolute value, i.e., the spectral radius of the Jacobian of $\boldsymbol{F}$.

Figure 2: SD element for second and third order in 1-dimension

Remark 7.1 (On the spatial coupling in SD methods)**.** *If we identify the solution points as degrees of freedom of classical finite element formulations, we have that the spatial coupling, in the context of SD methods, does not differ much from the one of a standard DG formulation, with the exception that no mass matrix is present in SD methods. Indeed, the local interpolation of the flux determines the coupling of the degrees of freedom within each cell, while, the coupling between neighboring cells is guaranteed, as already remarked, by the inclusion of the extrema of each segment among the flux points and the adoption of a numerical flux to define the flux values there.*

We presented the SD method in a 1-dimensional setting but the extension to the multidimensional case is straightforward on Cartesian grids, applying the same arguments dimension by dimension, as proposed, *inter alia*, in [63, 64].

The stability of SD methods has been shown to be independent of the choice of the solution and flux points under mild assumptions (i.e., solution points are located between flux points) in [18]. Further, it has been proven that there exist flux point placements for which the method is stable, both for 1-dimensional intervals and Cartesian meshes. In [43], the stability of SD schemes is established when the interior flux collocation points are the zeros of the Legendre polynomials. For further information on the SD-ADER scheme, the reader is referred to [63, 64].

Remark 7.2 (CFL condition and convergence of the ADER iterative procedure)**.** *It is possible to notice a relation between the CFL condition* (52) *and the ADER convergence condition from Proposition 2.5. Indeed, after linearization and after applying the Fourier transform, as done in [62], we observe that the PDE can be rewritten as a system of ODEs like in* (1)*. In particular, the Lipschitz-continuity constant of the resulting* $\boldsymbol{G}$ *is proportional to* $\frac{v_{max}}{\Delta x}$*, but it also depends on another coefficient, i.e.,* $M+1$ *[62]. This is actually less restrictive with respect to classical finite element schemes, where the constant is proportional to* $2M+1$*. For practical purposes, we include also a safety factor* $C \in \mathbb{R}^+$*, keeping the* Δt *a little lower than the theoretical bound, as* v_{max} *is known only at time* t_n *but not on the whole interval* $[t_n, t_{n+1}]$*.*

The treatment of shocks and oscillations is done through an *a-posteriori* limiter described in Appendix C.

8. Numerical results

In this section, we will numerically investigate the properties of the proposed improvements of ADER methods on several benchmarks both for ODEs and PDEs. Since the interpretation of ADER as DeC is not new [39], we do not investigate the *numerical speed-up* coming from the adoption of the optimal number of iterations rather than solving iteratively the ADER-IWF up to machine precision. Instead, we always assume a number of iterations equal to the desired order of accuracy and we focus on the impact of the modifications proposed in Sections 3 and 5, which are the main novelties of this work. In particular, we study the *numerical speed-up* of the methods with reduced number of GLB and GLG subtimenodes for a fixed order, according to the results presented in Section 3, simply indicated as ADER, with respect to cADER, characterized by a number of subtimenodes always equal to the desired order. Moreover, we investigate the *numerical speed-ups* of the novel ADERu and ADERdu methods, and their adaptive versions, with respect to both ADER and cADER. We recall that cADER methods make use of a number of subtimenodes equal to desired order of accuracy. As proven in Section 3, such number is non-optimal for GLB and GLG subtimenodes.

The integral terms of the ADER structures are computed exactly for equispaced and GLG subtimenodes using the GLG quadrature formula, while for GLB subtimenodes we adopt the associated quadrature leading to an underintegrated diagonal matrix Λ. This choices, for GLB and GLG subtimenodes, determine the high order implicit RK methods associated to the ADER methods to be respectively the ADER-IWF-RK-GLB and ADER-IWF-RK-GLG methods presented in Section 3. All the ADER matrices defined in the previous sections are precomputed at the beginning of the simulation, as they remain identical in every time step. This also holds for ADERu and ADERdu, for which the structures change along the iterative procedure but do not depend on the specific time step.

Finally, since ADER, ADERu and ADERdu are equivalent for order 2 and since the focus of this work is on (arbitrary) high order, we will investigate the methods from order 3 on.

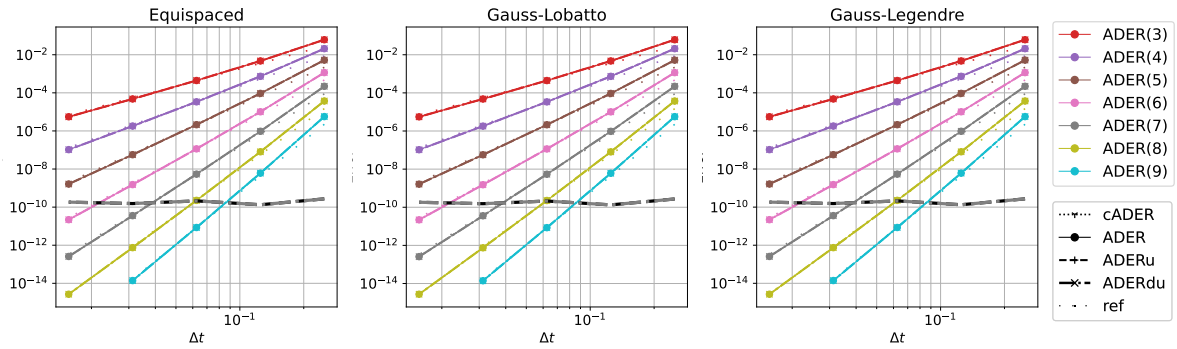

Figure 3: Linear system: Error decay for various methods and orders. The "ref" line is the reference for every order of accuracy

8.1. ODE tests

In this section, we focus on two ODE benchmarks: a simple linear system, which allows to verify the equivalence result presented in Theorem 6.1, and a more involved problem, the C5 test presented in [26], which allows to assess the performance of the methods in the context of real applications.

8.1.1. Linear system

The linear system under investigation reads

$$\begin{cases} u' = -5u + v \\ v' = 5u - v \end{cases}, \qquad \begin{pmatrix} u_0 \\ v_0 \end{pmatrix} = \begin{pmatrix} 0.9 \\ 0.1 \end{pmatrix}, \tag{53}$$

and the exact solution is given by $u(t) = u_0 + (1 - e^{-6t})(-5u_0 + v_0)$ and $v(t) = 1 - u(t)$. The problem is indeed very simple and, in fact, it has been chosen for the main purpose of verifying the analytical result summarized in Theorem 6.1, i.e., the equivalence on linear systems of ADER, ADERu and ADERdu of a given order independently of the adopted subtimenodes. In particular, due to the independence of the choice of subtimenodes, the same result applies to the cADER, from which we expect the same behavior as for the other schemes for any order. The final time is set to be $T = 1$.

The results of the convergence analysis are displayed in Figure 3. The expected order of accuracy is obtained for all the methods and also the expected equivalence on linear problems is numerically confirmed. The error against the computational time is reported in Figure 4. The adoption of the optimal number of subtimenodes is associated to a clear computational advantage, with the errors of ADER being always much smaller than cADER ones for a given computational time. Moreover, the novel ADERu and ADERdu methods are definitely faster than ADER in the context of equispaced subtimenodes. The computational performance of ADERu is slightly worse than ADER for GLB and GLG subtimenodes, while, ADERdu guarantees a computational advantage with respect to ADER for GLB subtimenodes and has the same computational performance as ADER for GLB subtimenodes.

The adaptive versions of ADERu and ADERdu, respectively in black and gray, have been tested with the convergence criterion (38) and a tolerance $\varepsilon = 10^{-8}$. The methods are able to automatically choose the order of accuracy to fulfill the desired error condition, as it can be observed by the error lines that lie uniformly below the tolerance level in Figure 3. For the prescribed tolerance, their performances are similar to the ones of very high order schemes, as can be seen in Figure 4, with error lines approaching the Pareto front for big time steps. In Figure 5, we observe how the average number of iterations required to achieve the expected accuracy changes with respect to Δt. As expected, larger time steps are associated to a higher number of iterations to reach the convergence tolerance. The small standard deviation in the number of iterations means that, on average, fixed tolerance ε and time step Δt correspond to a fixed order.

8.1.2. C5 problem

The next test we study is the C5 nonstiff problem proposed in [26]. It consists of a five body problem in three dimensions and it represents the description of the five outer planets (including Pluto) around the solar system. Each body has three coordinates y_{1j}, y_{2j}, y_{3j} and each coordinate satisfies

$$y''_{ij} = k_2 \left(\frac{-(m_0 + m_j) y_{ij}}{r_j^3} + \sum_{k \neq j} m_k \left[\frac{y_{ik} - y_{ij}}{d_{ik}^3} - \frac{y_{ik}}{r_k^3} \right] \right), \quad r_j^2 = \sum_{i=1}^{3} y_{ij}^2, \quad d_{kj} = \sum_{i=1}^{3} (y_{ik} - y_{ij})^2. \tag{54}$$

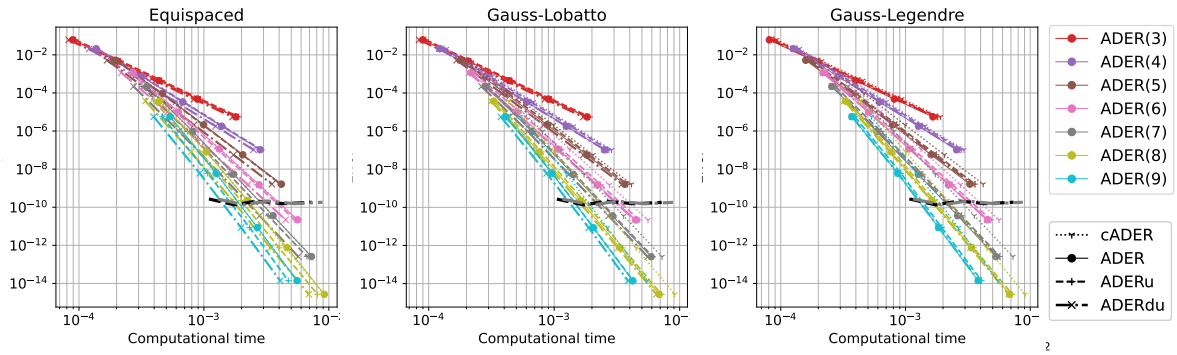

Figure 4: Linear system: Error with respect to computational time.

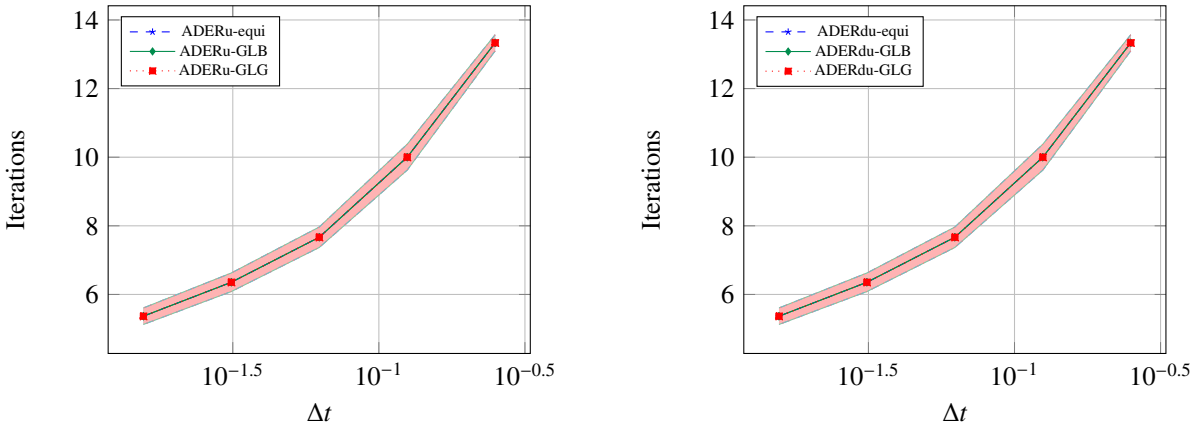

Figure 5: Linear system: Average number of iterations (± half standard deviation) of adaptive ADERu (left) and ADERdu (right) for different time steps.

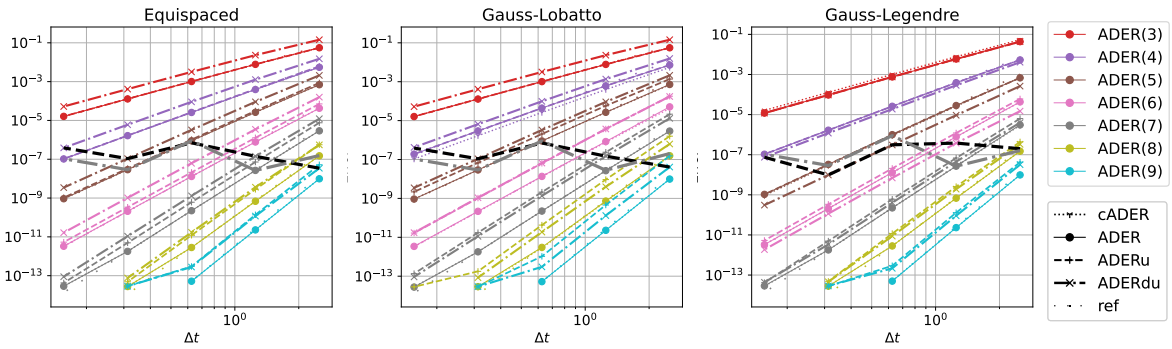

Figure 6: C5: Error decay for various methods and orders. The "ref" line is the reference for every order of accuracy

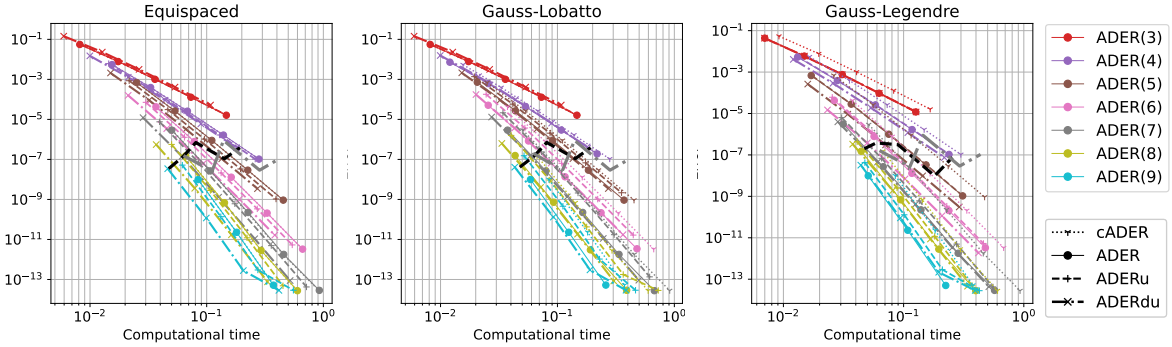

Figure 7: C5: Error with respect to computational time.

All mass coefficients, gravitational constant, final time and initial conditions can be found in [26]. This second order systems can be rewritten in terms of coordinates and velocities into a 30 equations system.

We are interested in this problem as the number of equations guarantees that the leading computational cost of the methods is directly proportional to the number of right-hand side evaluations. We use, as a reference, the solution obtained with ADER GLG of order 9 with 256 timesteps. All the errors are computed with respect to this reference solution.

In Figure 6, we observe that all schemes converge with the expected order of accuracy, even if, differently from the previous linear test, they do not have the same errors. For equispaced and GLB subtimenodes, we have that the smaller errors are almost always achieved by ADER and cADER methods, while for GLG nodes different orders have different behaviors. For example, for order 5 and 6 ADERdu is the method with lowest errors. As for the previous test, the adaptive methods with a tolerance of 10^{-8} lead, more or less, to errors of the magnitude of the tolerance, independently of the mesh discretization for all subtimenodes.

In terms of computational time, see Figure 7, we can observe that, most of the time, the ADERdu is the fastest scheme for comparable errors. For adaptive schemes, we observe again that the errors obtained with a fixed tolerance does not vary changing the time discretization, but the best computational times are obtained with coarse meshes. In Figure 8, we plot the *numerical speed-up* of ADER, ADERu and ADERdu methods against cADER. We recall that the *numerical speed-up* is defined as the computational time of the reference method, in this case of cADER, divided by the computational time of the method of interest, in our case ADER, ADERu and ADERdu. In this test problem, the simulation cost is mainly dictated by the number of right hand side evaluations, and this can be seen in Figure 8 as the *numerical speed-up* values do not vary much for different Δt. The *theoretical speed-up* values can be found in Tables 2, 3 and 4 under the cADER-speed-up column. Comparing those values, we notice that the *numerical speed-ups* are very close to the theoretical ones within a 10% error maximum.

Finally, for the adaptive methods we observe in Figure 9 that the chosen order for a given simulation is quite stable along the simulations, as we see almost no variance in all plots, and it scales very well changing the time discretization scale.

In Tables 8, 9 and 10, we can observe the *numerical speed-ups* compared with the theoretical ones for equispaced, GLB and GLG subtimenodes respectively. We recall that the *theoretical speed-up* is the ratio between the

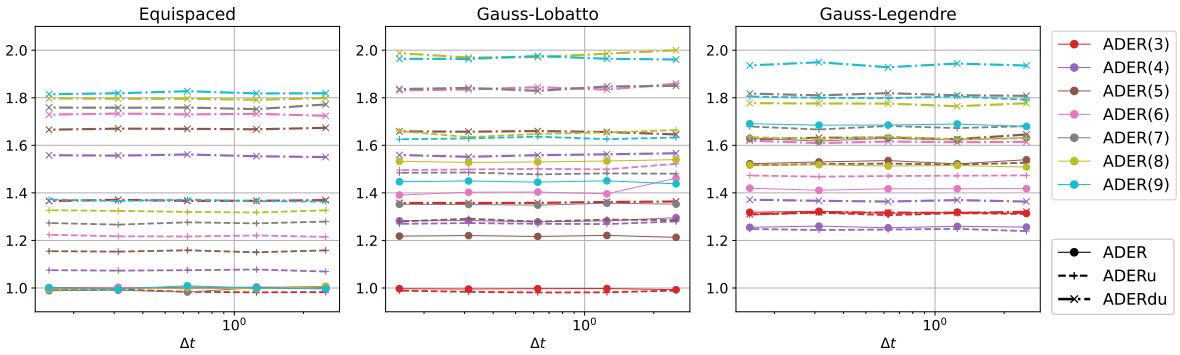

Figure 8: C5 test: *Numerical speed-up* with respect to cADER method computed as the computational time of the cADER method over the computational time of method in consideration.

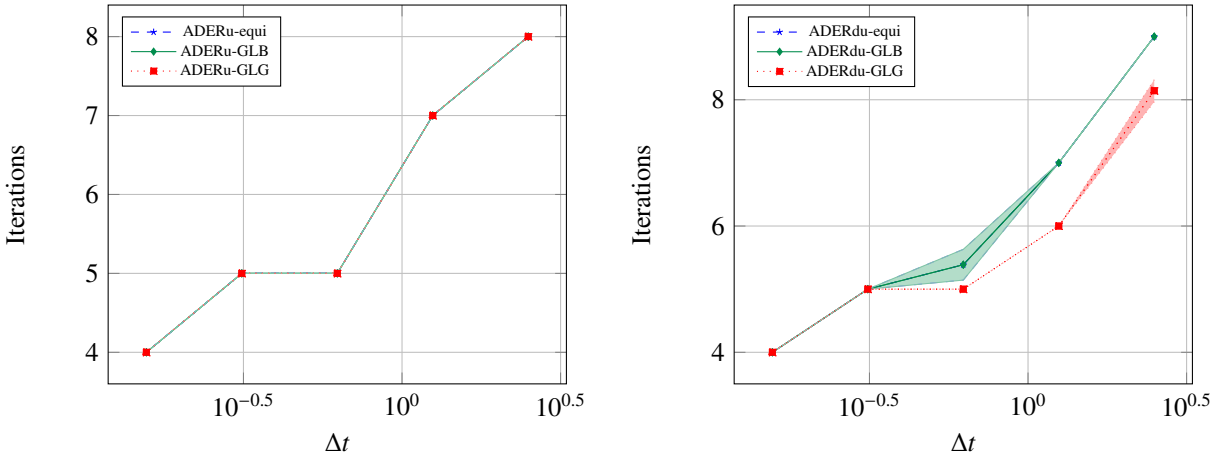

Figure 9: C5 test: Average number of iterations (± half standard deviation) of adaptive ADERu (left) and ADERdu (right) for different time steps.

Speed-up		ADERu vs ADER			ADERdu vs ADER		
P	*M*	Th.	LS	C5	Th.	LS	C5
3	2	1	0.956	0.993	1.5	1.069	1.366
4	3	1.091	0.975	1.076	1.714	1.132	1.561
5	4	1.176	1.013	1.154	1.818	1.185	1.664
6	5	1.25	1.049	1.225	1.875	1.247	1.731
7	6	1.312	1.088	1.288	1.909	1.279	1.779
8	7	1.366	1.124	1.333	1.931	1.321	1.806
9	8	1.431	1.163	1.371	1.973	1.359	1.813

Table 8: Comparison between the *theoretical speed-ups* and the numerical ones obtained for linear system (LS) test and the C5 problem. Equispaced subtimenodes

Speed-up		ADER vs cADER			ADERu vs cADER			ADERdu vs cADER			ADERu vs ADER			ADERdu vs ADER		
P	*M*	Th.	LS	C5	Th.	LS	C5	Th.	LS	C5	Th.	LS	C5	Th.	LS	C5
3	2	1	0.997	0.997	1	0.951	0.989	1.5	1.066	1.357	1	0.954	0.992	1.5	1.070	1.361
4	2	1.333	1.141	1.283	1.333	1.052	1.270	1.714	1.133	1.559	1	0.922	0.990	1.286	0.994	1.216
5	3	1.25	1.130	1.218	1.333	1.076	1.280	1.818	1.192	1.659	1.067	0.952	1.050	1.455	1.055	1.362
6	3	1.5	1.250	1.391	1.579	1.177	1.496	2	1.281	1.831	1.053	0.942	1.075	1.333	1.024	1.317
7	4	1.433	1.220	1.352	1.593	1.187	1.484	2.048	1.300	1.837	1.111	0.973	1.098	1.429	1.066	1.359
8	4	1.6	1.328	1.533	1.75	1.277	1.660	2.154	1.398	1.988	1.094	0.962	1.083	1.346	1.053	1.297
9	5	1.521	1.294	1.447	1.738	1.290	1.625	2.147	1.426	1.964	1.143	0.998	1.123	1.412	1.102	1.357

Table 9: Comparison between the *theoretical speed-ups* and the numerical ones obtained for linear system (LS) test and the C5 problem. GLB subtimenodes

Speed-up		ADER vs cADER			ADERu vs cADER			ADERdu vs cADER			ADERu vs ADER			ADERdu vs ADER		
P	M	Th.	LS	C5	Th.	LS	C5	Th.	LS	C5	Th.	LS	C5	Th.	LS	C5
3	1	1.4	1.146	1.319	1.4	1.056	1.309	1.4	1.104	1.311	1	0.922	0.992	1	0.963	0.994
4	2	1.3	1.146	1.255	1.3	1.053	1.248	1.444	1.131	1.371	1	0.919	0.995	1.111	0.987	1.092
5	2	1.615	1.280	1.523	1.615	1.164	1.516	1.75	1.222	1.625	1	0.910	0.996	1.083	0.955	1.067
6	3	1.476	1.238	1.420	1.55	1.158	1.473	1.722	1.240	1.620	1.05	0.936	1.038	1.167	1.002	1.141
7	3	1.72	1.372	1.629	1.792	1.275	1.679	1.955	1.333	1.818	1.042	0.929	1.031	1.136	0.971	1.116
8	4	1.583	1.330	1.517	1.727	1.266	1.635	1.9	1.350	1.777	1.091	0.952	1.078	1.2	1.016	1.172
9	4	1.78	1.432	1.691	1.921	1.360	1.805	2.086	1.433	1.936	1.079	0.949	1.068	1.171	1.000	1.145

Table 10: Comparison between the *theoretical speed-ups* and the numerical ones obtained for linear system (LS) test and the C5 problem. GLG subtimenodes

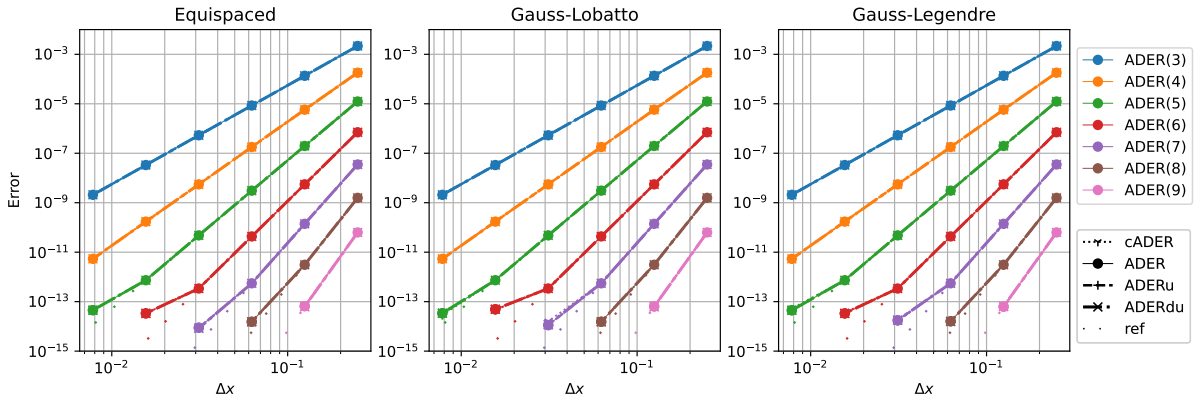

Figure 10: Linear advection: Error decay for various methods and orders. The "ref" line is the reference for every order of accuracy

number of RK stages of the reference method over the number of RK stages of the method of interest. On the other hand, the *numerical speed-up* is computed as the ratio between the computational times of such methods. The two measures are quite comparable as, in particular for complicated problems, the cost of the evaluation of $\boldsymbol{G}$ is the most expensive operation of the method, modulo parallelization and vectorization. For the ODE tests, our Python implementation, based on `numpy`, performs very similarly to the expected theoretical predictions. Looking at the tables, we observe that in the context of problem C5, for which the complexity of the function $\boldsymbol{G}$ is higher than for the linear system, the *numerical speed-ups* coincide with the theoretical ones within an error which is at most 10%. On the other hand, for the linear system, lower *numerical speed-ups* are obtained, as the ODE is so simple that the cost of basic operations is not negligible with respect to the cost of the computation of $\boldsymbol{G}$. As already remarked, the linear system is not a suitable problem for verifying the computational advantages of the new methods and its only purpose is verifying some analytical results, yet computational advantages are registered also in this case.

8.2. PDE tests

In this section, we will apply the novel ADER methods to hyperbolic PDEs through SD spatial semidiscretization as described in Section 7. In particular, the accuracy of the ADER methods will always be chosen equal to the spatial accuracy. The timings reported in this section are CPU process times.

8.2.1. Linear advection

We consider the linear advection equation

$$\partial_t u + \partial_x(au) = 0, \tag{55}$$

where $u(x,t)$ is a scalar wave, advected by a nonzero constant $a \in \mathbb{R}$. We consider the spatial domain $x \in [0,1]$, $a = 1$, initial condition $u_0(x) = \sin(2\pi x)$ with periodic boundary conditions. The analytical solution is given by $u(x,t) = u_0(x - at)$. The CFL is chosen according to (52), with $C = 0.8$, and the final time is set to be $T = 1$.

In Figure 10, we show the convergence plots and observe that the theoretical convergence rate is attained. In Figure 11, the error as a function of the computational time of the different ADER methods is shown. We can observe that the proposed modifications succeed in reducing the computational costs as Δx decreases, being

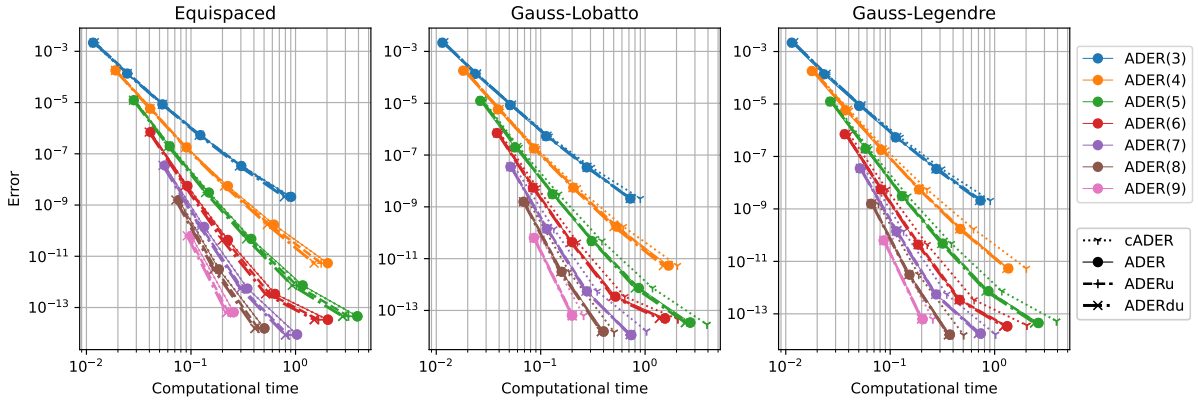

Figure 11: Linear advection: Error with respect to computational time

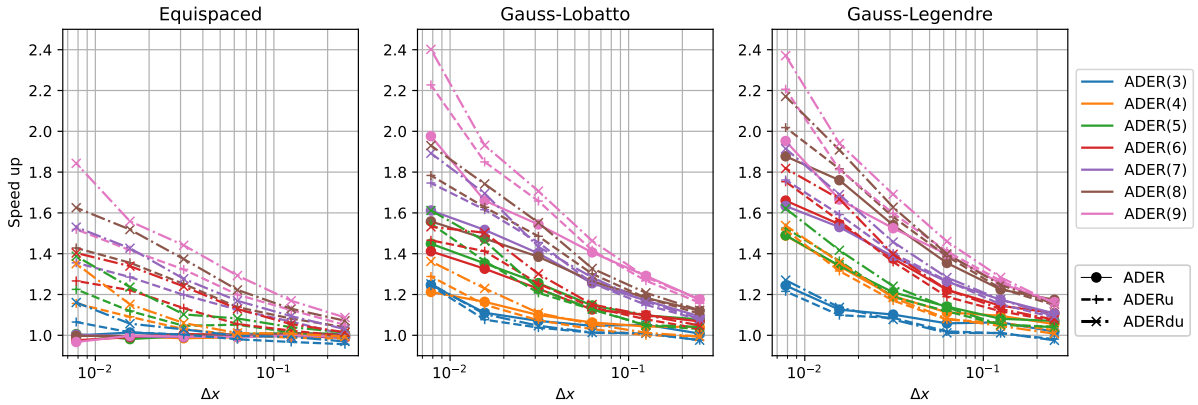

Figure 12: Linear advection: *Numerical speed-up* factor with respect to cADER method computed as the computational time of the cADER method over the computational time of method in consideration

ADERdu and ADERu the best performing schemes, followed by ADER with optimal number of subtimenodes and, finally, by cADER. Lastly, in Figure 12, we show the *numerical speed-ups* attained when using the reformulated ADER, ADERu and ADERdu with respect to cADER. In particular, we empirically observe *numerical speed-ups* that vary a lot depending on the spatial discretization and that they increase as the spatial grid is refined and the polynomial order increases. The authors are not completely sure about the reason of this behavior, but they suspect that the implementation through Python packages and functions as `einsum` of `numpy` could be the source of this outcome. The phenomenon is currently under investigation.

However, despite the less homogeneous behavior of the *numerical speed-ups* with respect to the ODE tests, the results obtained for small values of Δx (in principle more reliable) are in good agreement with the theoretical predictions. In general, we remark again that the ADERdu is the fastest method for not too coarse meshes.

8.2.2. Euler equations

We now consider the 1D Euler equations

$$\partial_t \begin{pmatrix} \rho \\ \rho v \\ E \end{pmatrix} + \partial_x \begin{pmatrix} \rho v \\ \rho v^2 + p \\ (E + p)v \end{pmatrix} = 0, \tag{56}$$

where ρ is the mass density, v the velocity, $E = e + \frac{1}{2}\rho v^2$ the total energy, equal to the sum of the internal energy density e and the kinetic energy density. The system is closed with the equation of state of an ideal gas $p = (\gamma - 1)e$, with γ the constant adiabatic index, set at $\gamma = 1.4$.

We consider first a nonlinear sound wave test case, as in [64], that consists of a nonlinear acoustic perturbation

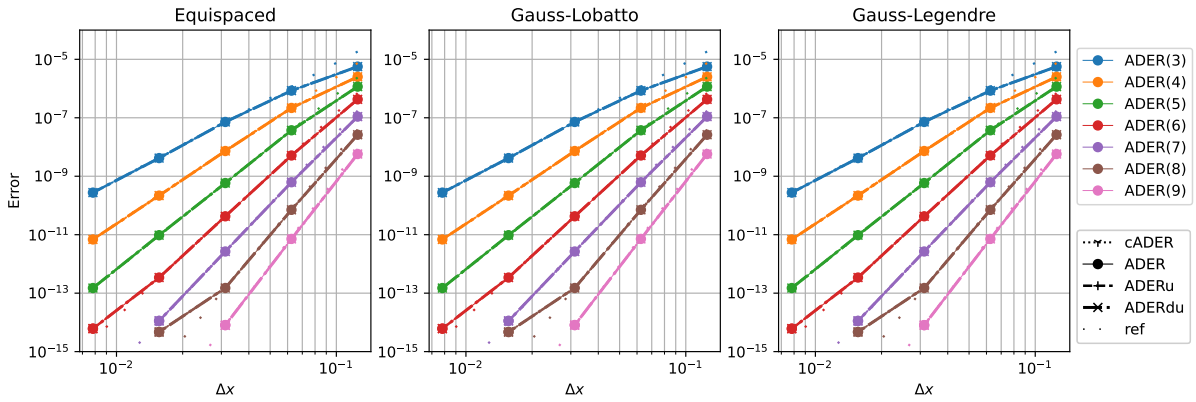

Figure 13: 1D Euler equations with sound wave initial condition (57): Error decay for various methods and orders. The "ref" line is the reference for every order of accuracy

over a uniform equilibrium state

$$\begin{aligned} \rho_0(x) &= 1 + A\sin(kx)/c_{s,0}, \\ v_0(x) &= A\sin(kx), \\ p_0(x) &= 1 + \gamma p_0 A\sin(kx)/c_{s,0}, \end{aligned} \tag{57}$$

in the spatial computational domain $[0, 1]$, with $c_{s,0} = \sqrt{\gamma}$, $k = 20\pi$ and $A = 10^{-6}$.

The corresponding time-dependent solution for the velocity field has the following analytical solution, valid up to second-order (in a perturbative sense),

$$v(x,t) = A\sin(kx - \omega t) + A^2\frac{\gamma+1}{4}\frac{\omega t}{c_{s,0}}\cos\left[2(kx-\omega t)\right],$$

for $\omega = k\gamma$.

We compute the error of the numerical solution with respect to this analytical solution at the final time $T = 1/(c_{s,0}k)$, corresponding to the time needed by the sound wave to perform one complete orbit over the periodic domain. The CFL is chosen according to (52), with $C = 0.4$.

First, we verify in Figure 13 the convergence rates for increasing degree of the polynomial basis. We observe essentially no difference between the different versions of ADER. As the errors are identical for all types of schemes, we report directly the *numerical speed-up* as a measure of the computational advantage. In Figure 15, ADERdu outperforms all other methods reaching *numerical speed-up* factors of the order of the theoretical ones in Section 6. Also in this case, there is more variance in these results with respect to the ODEs ones and the authors believe this may be due to the implementation of the ADER-SD method using Python packages and functions as `einsum` of `numpy`, where, for instance, the role of the memory layout plays a big role in the computational costs. Nevertheless, just like in the previous test, for small values of Δx the *numerical speed-ups* get closer to the analytical ones.

The next test is the well known Sod shock tube problem [**?**], characterized by the following initial condition

$$\rho_0(x) = \begin{cases} 1, & x < 0.5, \\ 0.1, & x \geq 0.5, \end{cases} \qquad v_0(x) = 1, \qquad p_0(x) = \begin{cases} 1, & x < 0.5, \\ 0.125, & x \geq 0.5, \end{cases} \tag{58}$$

in the spatial computational domain $[0, 1]$ and zero gradient boundary conditions. We let the numerical solution evolve until the final time $T = 0.2$. In order to tackle the discontinuities occurring in the solution, we adopt an *a posteriori* limiting strategy similar to the one presented in [64] and described in detail in Appendix C.

In this case, we do not perform convergence analysis as the convergence order would be at most 1, due to the mentioned discontinuities. However, we can notice, from Figure 16, that the quality of the solution improves as the (formal) order of the discretization increases and that there is no noticeable difference between the quality of the solution for the different versions of the ADER scheme. In Figure 17, we report the numerical speed-ups obtained for this particular test. As expected, we observe that ADERu, ADERdu and ADER outperform cADER. Further, ADERu and ADERdu outperform ADER. Again, as in the previous PDE tests, we observe a dependency on the grid size Δx, with larger gains when the mesh is more refined. Let us notice that the computational complexity

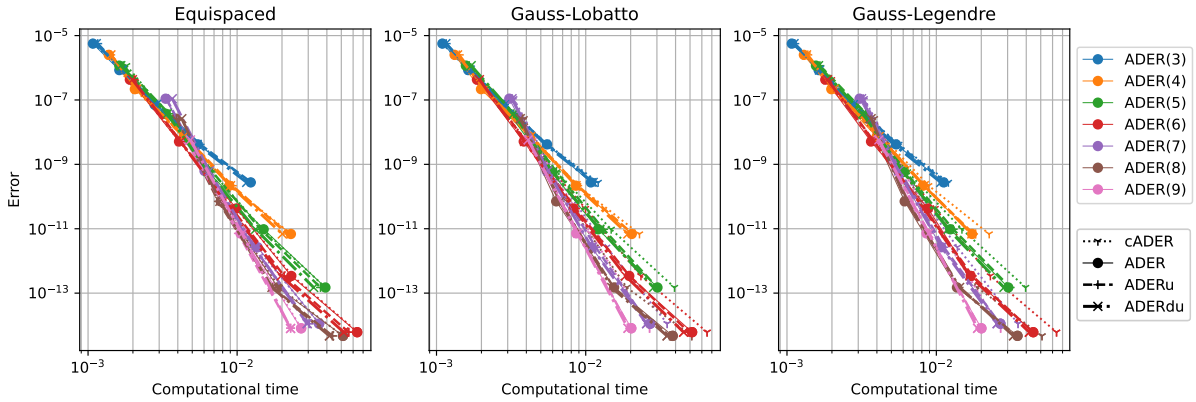

Figure 14: 1D Euler equations with sound wave initial condition (57): Error with respect to computational time

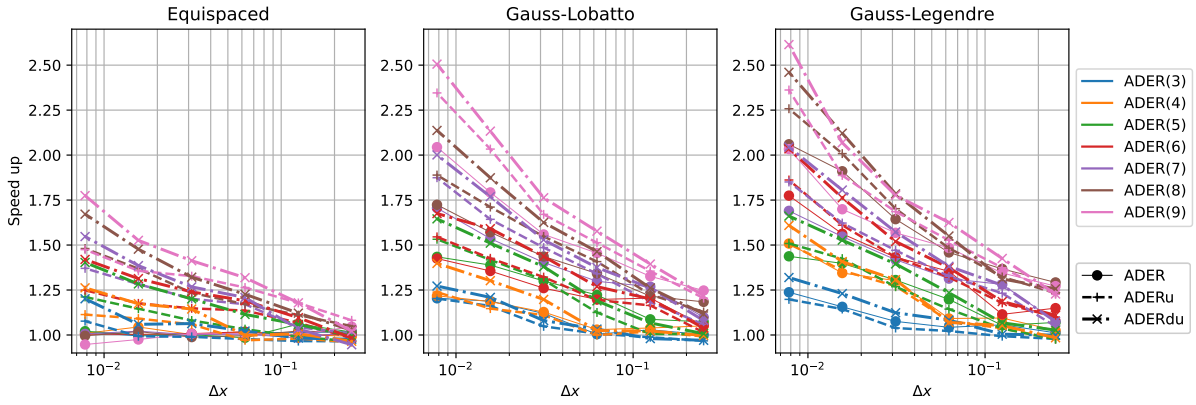

Figure 15: 1D Euler equations with sound wave initial condition (57): *Numerical speed-up* factor with respect to cADER method computed as the computational time of the cADER over the computational time of the method in consideration

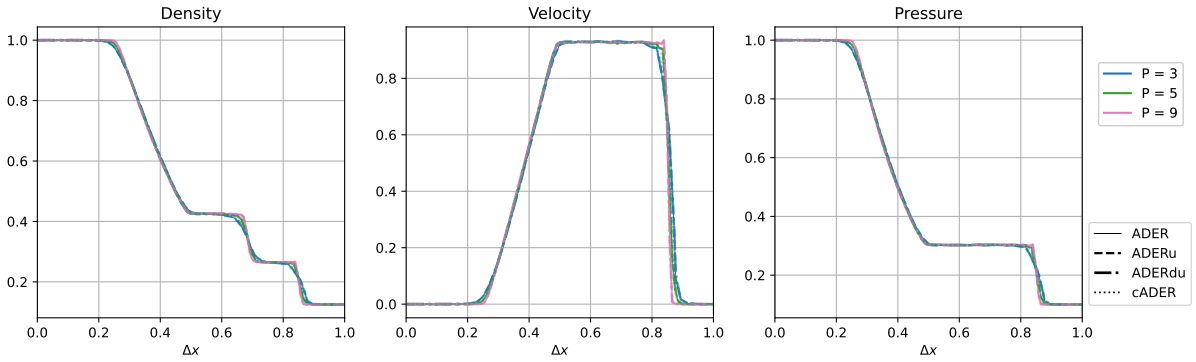

Figure 16: 1D Euler equations with Sod shock tube initial condition (58): Numerical solutions computed for different orders P for a fixed resolution characterized by 16 elements

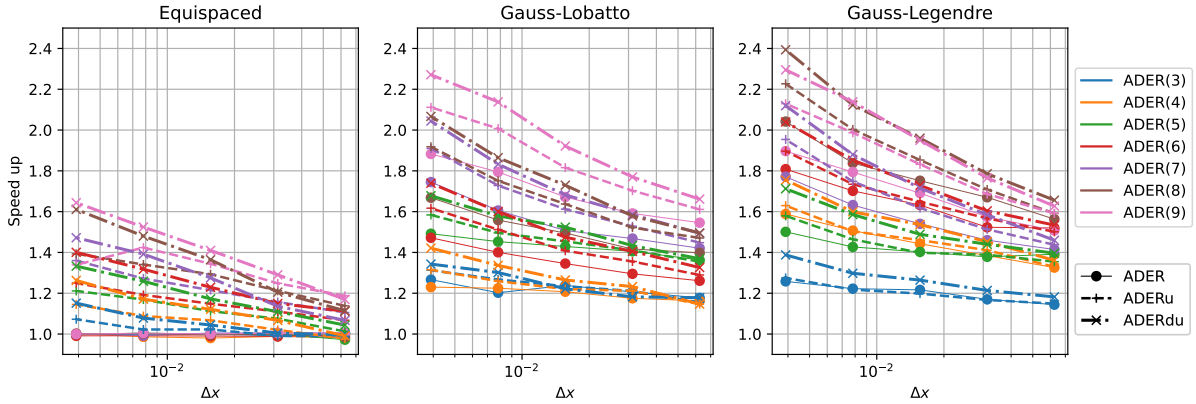

Figure 17: 1D Euler equations with Sod shock tube initial condition (58): *Numerical speed-up* factor with respect to cADER method computed as the computational time of the cADER method over the computational time of the method in consideration

of the limiter is directly related to the number of subtimenodes and this implies further advantages in ADER, ADERu and ADERdu with respect to cADER.

9. Conclusions and further developments

Summarizing, in this work we have showed different techniques to save computational times when adopting ADER methods. In the first part of the paper, we have shown how the weak formulations can be optimally discretized using a minimal reconstruction degree to maximize the order of accuracy. Then, we have designed efficient methods, based on increasing the polynomial degree of the reconstructed numerical solution at each iteration, saving computational time especially in early stages. This allowed us to easily set up p-adaptive versions of the new schemes, where the iterations can be stopped on-the-fly when certain criteria are met, e.g., the matching of an error tolerance. The whole presentation is accompanied by theoretical and numerical analysis that validate the proposed methods, showing strong improvements in the computational times without degradation of the accuracy nor stability of the methods.

We believe these strategies can have a strong impact in the community that uses these algorithms, as they allow to speed-up the simulations, save computational resources and they give new hints for adaptive methods. From this work, new research directions arise, both in the ODE and hyperbolic PDE framework, namely, for hp-adaptive methods, implicit and structure preserving schemes.

Acknowledgments

L. Micalizzi has been funded by the SNF grant 200020_204917 “Structure preserving and fast methods for hyperbolic systems of conservation laws” and by the Forschungskredit grant FK-21-098. D. Torlo has been funded

by a SISSA Mathematical Fellowship. M. Han Veiga has been funded by the Michigan Institute for Data Science (MIDAS) and the Van Loo Postdoctoral fellowship at the University of Michigan.

Declarations of interest

None.

Appendix A. Proofs

In this section, we report the proofs of several "minor" propositions and theorems that have been presented throughout the paper.

Appendix A.1. Proof of Proposition 2.4

Proof. We can equivalently prove that

$$\boldsymbol{r} = B\underline{\boldsymbol{u}}_n. \tag{A.1a}$$

By a direct computation of the ℓ-th component of both sides of the previous equation, thanks to the fact that the Lagrange functions are such that $\sum_{m=0}^{M} \psi^m \equiv 1$, we have

$$\begin{aligned}\sum_{m=0}^{M} B_{\ell,m}\boldsymbol{u}_n &= \sum_{m=0}^{M}\left[\psi^\ell(t_{n+1})\psi^m(t_{n+1}) - \int_{t_n}^{t_{n+1}}\left(\frac{d}{dt}\psi^\ell(t)\right)\psi^m(t)dt\right]\boldsymbol{u}_n \\ &= \psi^\ell(t_{n+1})\boldsymbol{u}_n - \int_{t_n}^{t_{n+1}}\left(\frac{d}{dt}\psi^\ell(t)\right)dt\boldsymbol{u}_n \\ &= [\psi^\ell(t_{n+1}) - \psi^\ell(t_{n+1}) + \psi^\ell(t_n)]\boldsymbol{u}_n = \psi^\ell(t_n)\boldsymbol{u}_n = \boldsymbol{r}_\ell,\end{aligned} \tag{A.1b}$$

which implies (A.1a). □

Appendix A.2. Proof of Proposition 2.5: Convergence of the iterative procedure

Proof. The proof relies on the definition of the map $\mathcal{J} : \mathbb{R}^{(M+1)\times Q} \to \mathbb{R}^{(M+1)\times Q}$, defined by

$$\mathcal{J}(\underline{\boldsymbol{u}}) := \underline{\boldsymbol{u}}_n + \Delta t B^{-1}\Lambda\underline{\boldsymbol{G}}(\underline{\boldsymbol{u}}), \quad \forall \underline{\boldsymbol{u}} \in \mathbb{R}^{(M+1)\times Q}. \tag{A.2a}$$

We will now show that it is a contraction. We take two general vectors $\underline{\boldsymbol{v}}, \underline{\boldsymbol{w}} \in \mathbb{R}^{(M+1)\times Q}$ and, from the definition of $\underline{\boldsymbol{G}}$ and the Lipschitz-continuity of $\boldsymbol{G}$ with respect to $\boldsymbol{u}$ uniformly with respect to t, basic computations give

$$\left\|\mathcal{J}(\underline{\boldsymbol{v}}) - \mathcal{J}(\underline{\boldsymbol{w}})\right\|_\infty = \left\|\Delta t B^{-1}\Lambda\left[\underline{\boldsymbol{G}}(\underline{\boldsymbol{v}}) - \underline{\boldsymbol{G}}(\underline{\boldsymbol{w}})\right]\right\|_\infty \leq \Delta t \left\|B^{-1}\Lambda\right\|_\infty C_{Lip}\left\|\underline{\boldsymbol{v}} - \underline{\boldsymbol{w}}\right\|_\infty. \tag{A.2b}$$

The entries of B and Λ, as well as C_{Lip}, are constants independent of Δt, therefore, for $\Delta t < \frac{1}{\widetilde{C}_{Lip}}$ with $\widetilde{C}_{Lip} := \left\|B^{-1}\Lambda\right\|_\infty C_{Lip}$ we have that $\mathcal{J}$ is a contraction over $\mathbb{R}^{(M+1)\times Q}$ with respect to the infinity norm. Thus, thanks to the Banach fixed-point theorem, we have that for $\Delta t < \frac{1}{\widetilde{C}_{Lip}}$ the map $\mathcal{J}$ has a unique fixed point, which can be obtained as the limit of the iteration $\underline{\boldsymbol{u}}^{(p)} := \mathcal{J}(\underline{\boldsymbol{u}}^{(p-1)})$, independently of the choice of $\underline{\boldsymbol{u}}^{(0)}$. This fixed-point iteration is equivalent to the iterative procedure (10). Observing that a fixed point of $\mathcal{J}$ is also a solution of the nonlinear system (9) and vice versa, we get the desired result. □

Appendix A.3. Proof of Theorem 2.6: Invertibility of B

Proof. As shown in Theorem 3.10 and more precisely in Equation (19k), the matrix B exactly integrated, for any generic basis $\left\{\widehat{\phi}(\xi)\right\}_{m=0,\dots,M}$, is equivalent to the matrix B_{GL*} of the ADER method with $GL*$ subtimenodes up to the multiplication by the change of basis matrix $\mathcal{H}$ and its transpose, i.e., $B = \mathcal{H}^T B_{GL*}\mathcal{H}$.

Since the matrix $\mathcal{H}$ is invertible, we can prove the invertibility of B for a particular basis and obtain the desired result for all the other bases. Let us consider the modal basis functions $\widehat{\phi}^m(\xi) := \xi^m$ for $m = 0, \dots, M$. A direct computation shows that, in such a case, the matrix B is defined as

$$B_{\ell,m} = \widehat{\phi}^\ell(1)\widehat{\phi}^m(1) - \int_0^1\left(\frac{d}{d\xi}\widehat{\phi}^\ell(\xi)\right)\widehat{\phi}^m(\xi)\,d\xi = \begin{cases}1, & \text{if } \ell = 0,\\ \frac{m}{\ell+m}, & \text{if } \ell = 1,\dots,M.\end{cases} \tag{A.3}$$

The first row of the matrix B is given by $B_{0,m} = 1$ for all $m = 0, \dots, M$, while, the first column by $B_{\ell,0} = 0$ for $\ell = 1, \dots, M$. Hence, the determinant of the matrix is equal to the determinant of the submatrix $B_{1:,1:}$ and we can focus on it. In particular, we will show that it is nonsingular and, therefore, that its determinant is nonzero.

From basic linear algebra, a matrix is nonsingular if and only if its columns are linearly independent. Furthermore, the columns of a matrix are linearly independent if and only if multiplied by nonzero factors they are still linearly independent. Hence, let us consider the matrix $\widetilde{B}$ with generic entry given by $\widetilde{B}_{\ell,m} := \frac{1}{m} B_{\ell,m} = \frac{1}{\ell+m}$ for $\ell, m = 1, \dots, M$. Such matrix is the so-called "Hilbert" matrix, which is invertible [**?**], and so also $B_{1:,1:}$ is invertible. Hence, B has nonzero determinant and is invertible. □

Appendix A.4. Proof of Theorem 3.1: The ADER-IWF is an implicit RK

Proof. We have already observed that the nonlinear systems (9) and (5) are equivalent and, identifying $\boldsymbol{u}^m$ with RK stage values $\boldsymbol{y}^s$, it is clear that (9) is the nonlinear system of a RK method, expressed by the first equation in (12), characterized by $A = B^{-1}\Lambda$ and $\boldsymbol{c} = \underline{\beta}$. We are left to check that the reconstruction formula (4) can be written as the RK final update, expressed by the second equation in (12), with coefficients $b_m = w_m$. Manipulating the nonlinear system (5) we get

$$\begin{aligned}\sum_{m=0}^{M}\left[\psi^\ell(t_{n+1})\psi^m(t_{n+1}) - \int_{t_n}^{t_{n+1}}\left(\frac{d}{dt}\psi^\ell(t)\right)\psi^m(t)dt\right]\boldsymbol{u}^m = \psi^\ell(t_n)\boldsymbol{u}_n \\ + \sum_{m=0}^{M}\left(\int_{t_n}^{t_{n+1}}\psi^\ell(t)\psi^m(t)dt\right)\boldsymbol{G}(t^m,\boldsymbol{u}^m), \quad \ell = 0,\dots,M,\end{aligned} \tag{A.4a}$$

and, summing over ℓ and recalling the fact that $\sum_{\ell=0}^{M}\psi^\ell \equiv 1$, we obtain

$$\boldsymbol{u}_h(t_{n+1}) = \sum_{m=0}^{M}\psi^m(t_{n+1})\boldsymbol{u}^m = \boldsymbol{u}_n + \sum_{m=0}^{M}\left(\int_{t_n}^{t_{n+1}}\psi^m(t)dt\right)\boldsymbol{G}(t^m,\boldsymbol{u}^m) = \boldsymbol{u}_n + \Delta t\sum_{m=0}^{M} w_m\boldsymbol{G}(t^m,\boldsymbol{u}^m). \tag{A.4b}$$

□

Appendix A.5. Proof of Proposition 3.2

Proof. Proving (13) is equivalent to prove that $A\mathbf{1} = \boldsymbol{c}$, where $\mathbf{1}$ is a vector with all entries equal to 1. Knowing that $A = B^{-1}\Lambda$ and $\boldsymbol{c} = \underline{\beta}$, we can prove (13) by showing that $\Lambda\mathbf{1} = B\underline{\beta}$.

By a direct computation of the ℓ-th component of both the sides, since $\sum_{m=0}^{M}\widehat{\psi}^m \equiv 1$, we have

$$(\Lambda\mathbf{1})_\ell = \sum_{m=0}^{M}\int_0^1 \widehat{\psi}^\ell(\xi)\widehat{\psi}^m(\xi)d\xi = \int_0^1 \widehat{\psi}^\ell(\xi)d\xi, \tag{A.5a}$$

$$\begin{aligned}\left(B\underline{\beta}\right)_\ell &= \sum_{m=0}^{M}\left[\widehat{\psi}^\ell(1)\widehat{\psi}^m(1) - \int_0^1\left(\frac{d}{d\xi}\widehat{\psi}^\ell(\xi)\right)\widehat{\psi}^m(\xi)d\xi\right]\xi^m \\ &= \widehat{\psi}^\ell(1)\left(\sum_{m=0}^{M}\xi^m\widehat{\psi}^m(1)\right) - \int_0^1\left(\frac{d}{d\xi}\widehat{\psi}^\ell(\xi)\right)\left(\sum_{m=0}^{M}\xi^m\widehat{\psi}^m(\xi)\right)d\xi.\end{aligned} \tag{A.5b}$$

Let us focus on $\left(B\underline{\beta}\right)_\ell$. Since $\sum_{m=0}^{M}\xi^m\widehat{\psi}^m$ is nothing but the interpolation of the linear function ξ, which is exact when interpolated in at least $2 \le M+1$ nodes, we can write

$$\left(B\underline{\beta}\right)_\ell = \widehat{\psi}^\ell(1)\cdot 1 - \int_0^1\left(\frac{d}{d\xi}\widehat{\psi}^\ell(\xi)\right)\xi d\xi, \tag{A.5c}$$

and integrating by parts we obtain

$$\left(B\underline{\beta}\right)_\ell = \widehat{\psi}^\ell(1)\cdot 1 - \left[\widehat{\psi}^\ell(1)\cdot 1 - \widehat{\psi}^\ell(0)\cdot 0\right] + \int_0^1\widehat{\psi}^\ell(\xi)d\xi = \int_0^1\widehat{\psi}^\ell(\xi)d\xi, \tag{A.5d}$$

and, thus, the desired result. □

Appendix A.6. Proof of Proposition 3.3

Proof. By direct computation, we will estimate the order of magnitude of the local truncation error. The properties of interpolation with $M+1$ subtimenodes guarantee that

$$\boldsymbol{u}(t) = \sum_{m=0}^{M} \boldsymbol{u}(t^m)\psi^m(t) + O(\Delta t^{M+1}), \qquad \boldsymbol{G}(t,\boldsymbol{u}(t)) = \sum_{m=0}^{M} \boldsymbol{G}(t^m,\boldsymbol{u}(t^m))\psi^m(t) + O(\Delta t^{M+1}). \tag{A.6a}$$

Thus, if we insert the exact solution of the ODEs system (1) in the ADER-IWF (5), we get

$$\begin{aligned}
&\sum_{m=0}^{M}\left[\psi^\ell(t_{n+1})\psi^m(t_{n+1}) - \int_{t_n}^{t_{n+1}}\left(\frac{d}{dt}\psi^\ell(t)\right)\psi^m(t)dt\right]\boldsymbol{u}(t^m) - \psi^\ell(t_n)\boldsymbol{u}_n \\
&\qquad - \sum_{m=0}^{M}\left(\int_{t_n}^{t_{n+1}}\psi^\ell(t)\psi^m(t)dt\right)\boldsymbol{G}(t^m,\boldsymbol{u}(t^m)) \\
&= \psi^\ell(t_{n+1})\boldsymbol{u}(t_{n+1}) - \int_{t_n}^{t_{n+1}}\left(\frac{d}{dt}\psi^\ell(t)\right)\boldsymbol{u}(t)dt - \psi^\ell(t_n)\boldsymbol{u}_n \\
&\qquad - \int_{t_n}^{t_{n+1}}\psi^\ell(t)\boldsymbol{G}(t,\boldsymbol{u}(t))dt + O(\Delta t^{M+1}) \\
&= \int_{t_n}^{t_{n+1}}\psi^\ell(t)\left[\frac{d}{dt}\boldsymbol{u}(t) - \boldsymbol{G}(t,\boldsymbol{u}(t))\right]dt + O(\Delta t^{M+1}), \quad \ell = 0,\dots,M.
\end{aligned} \tag{A.6b}$$

Hence, the coefficients $\boldsymbol{u}^m$ are $O(\Delta t^{M+1})$ accurate with respect to the exact values $\boldsymbol{u}(t^m)$. In the proof of Theorem 3.1, we showed that the final interpolation step to get $\boldsymbol{u}_{n+1}$ is equivalent to the integration

$$\boldsymbol{u}_{n+1} = \boldsymbol{u}_n + \sum_{m=0}^{M}\left(\int_{t_n}^{t_{n+1}}\psi^m(t)dt\right)\boldsymbol{G}(t^m,\boldsymbol{u}^m) = \boldsymbol{u}_n + \int_{t_n}^{t_{n+1}}\left(\sum_{m=0}^{M}\boldsymbol{G}(t^m,\boldsymbol{u}^m)\psi^m(t)\right)dt. \tag{A.6c}$$

So, recalling that we are assuming $\boldsymbol{u}_n = \boldsymbol{u}(t_n)$, we have

$$\begin{aligned}
\boldsymbol{u}_{n+1} &= \boldsymbol{u}_n + \int_{t_n}^{t_{n+1}}\left[\boldsymbol{G}(t,\boldsymbol{u}(t)) + O(\Delta t^{M+1})\right]dt = \boldsymbol{u}(t_n) + \int_{t_n}^{t_{n+1}}\boldsymbol{G}(t,\boldsymbol{u}(t))dt + O(\Delta t^{M+2}) \\
&= \boldsymbol{u}(t_{n+1}) + O(\Delta t^{M+2}),
\end{aligned} \tag{A.6d}$$

which concludes the proof. □

Appendix B. Equivalence of ADER-IWF-RK-GLB and Lobatto IIIC

In this section, we show the equivalence between ADER-IWF-RK-GLB and Lobatto IIIC schemes. First of all, let us recall how Lobatto IIIC are defined.

Theorem Appendix B.1 (Chipman 1971 [15])**.** *The Lobatto IIIC scheme with S stages is uniquely defined by S GLB quadrature points and associated weights, respectively as RK coefficients $\boldsymbol{c}$ and $\boldsymbol{b}$, and coefficients $a_{i,j}$ determined imposing the conditions*

$$a_{i,0} = b_0, \qquad \text{for } i = 0,\dots,S-1 \tag{B.1}$$

and $C(S-1)$, with definition given in (14b)*.*

Then, we can show that also ADER-IWF-RK-GLB satisfies the same conditions.

Lemma Appendix B.2. *ADER-IWF-RK-GLB satisfies condition* (B.1)*.*

Proof. Let us recall the RK structures of the ADER-IWF-RK methods, given in Theorem 3.1

$$A := B^{-1}\Lambda, \quad \boldsymbol{c} := \underline{\beta}, \quad b_m := \frac{1}{\Delta t}\int_{t_n}^{t_{n+1}}\psi^m(t)dt = \int_0^1 \widehat{\psi}^m(\xi)d\xi = w_m. \tag{B.2a}$$

Assuming GLB subtimenodes and quadrature points (exact for polynomials of degree $2M-1$), we have that the matrix B is computed exactly, while Λ is under-integrated (as its terms involve integrals of polynomials of degree $2M$), leading to a diagonal Λ

$$\begin{cases} B_{\ell,m} := \widehat{\psi}^{\ell}(1)\widehat{\psi}^{m}(1) - \int_0^1 \left(\frac{d}{d\xi}\widehat{\psi}^{\ell}(\xi)\right)\widehat{\psi}^{m}(\xi)d\xi, \\ \Lambda_{\ell,m} := w_\ell \delta_{\ell,m} = \sum_{k=0}^{M} w_k \widehat{\psi}^{\ell}(\xi_k)\widehat{\psi}^{m}(\xi_k) \approx \int_0^1 \widehat{\psi}^{\ell}(\xi)\widehat{\psi}^{m}(\xi)d\xi. \end{cases} \tag{B.2b}$$

To prove that $a_{i,0} = b_0$ for all $i = 0, \dots, S-1$, we have to prove that

$$(B^{-1}\Lambda)_{i,0} \stackrel{!}{=} w_0 \quad \Longleftrightarrow \quad (B^{-1}\Lambda)_{:,0} \stackrel{!}{=} w_0 \underline{1} \quad \Longleftrightarrow \quad B^{-1}\Lambda_{:,0} \stackrel{!}{=} w_0 \underline{1} \quad \Longleftrightarrow \quad \Lambda_{:,0} \stackrel{!}{=} w_0 B \underline{1},$$

where the symbol colon ":" is used to select all the elements of a given row or column.

This amounts to show that $\sum_{m=0}^{M} B_{\ell,m} w_0 = \delta_{\ell,0} w_0$, which can be easily proven exploiting the definition of B and the fact that $\sum_{m=0}^{M} \widehat{\psi}^{m} \equiv 1$:

$$\sum_{m=0}^{M} B_{\ell,m} w_0 = \widehat{\psi}^{\ell}(1) \cdot w_0 - \int_0^1 \left(\frac{d}{d\xi}\widehat{\psi}^{\ell}(\xi)\right) d\xi w_0 = [\widehat{\psi}^{\ell}(1) - \widehat{\psi}^{\ell}(1) + \widehat{\psi}^{\ell}(0)] w_0 = \widehat{\psi}^{\ell}(0) w_0. \tag{B.2c}$$

Since only the first GLB basis function, for $\ell = 0$, is 1 in $\xi = 0$, while the other ones have value 0 there, (B.1) holds. □

Theorem Appendix B.3. *ADER-IWF-RK-GLB methods are Lobatto IIIC methods.*

Proof. ADER-IWF-RK-GLB methods are such that: $\boldsymbol{c}$ and $\boldsymbol{b}$ coincide with the GLB quadrature points and weights, $C(S-1)$ holds for for Lemma 3.5, while Lemma Appendix B.2 guarantees condition (B.1). Hence, ADER-IWF-RK-GLB methods satisfy the properties of Theorem Appendix B.1, which uniquely characterize the Lobatto IIIC methods. Thus, the two methods coincide. □

Appendix C. *A posteriori* limiting strategy

In this appendix, we describe the shock capturing strategy adopted to perform the last numerical test for PDEs characterized by shocks. The high-order space discretization is achieved via the SD scheme, presented in Section 7. We adopt an *a posteriori* limiting strategy similar to the one described in [64] and references therein.

The general methodology of the *a posteriori* limiting consists in correcting the high-order fluxes, that are responsible for oscillatory behavior, after the computations of one time step have been completed. This is in contrast with *a priori* limiting, which modifies the high-order flux during the time step computation. In order to identify which fluxes must be corrected, we consider two physical criteria

1. positiveness of density and pressure;
2. discrete maximum principle for the density.

As shown in [64], the SD method is equivalent to a Finite Volume explicit Euler method using high-order fluxes $\widehat{\boldsymbol{f}}_i^{HO,m}$ and a nonuniform (space-time) subcell grid, such that the update (51) for the final iteration after ADER discretization can be equivalently written as

$$\bar{\boldsymbol{u}}_i^{-1} = \boldsymbol{u}_{n,i}, \tag{C.1}$$

$$\tilde{\boldsymbol{u}}_i^{m} = \bar{\boldsymbol{u}}_i^{m-1} - (t^m - t^{m-1}) \frac{\widehat{\boldsymbol{f}}_{i+1/2}^{HO,m-1} - \widehat{\boldsymbol{f}}_{i-1/2}^{HO,m-1}}{\Delta x_i}, \qquad \text{for } m = 0, \dots, M, \tag{C.2}$$

$$\boldsymbol{u}_{n+1,i} = \boldsymbol{u}_{n,i} - \Delta t \sum_{m=0}^{M} w_m \partial_x \boldsymbol{F}_h^{(P)}(x_i^s, t^m) = \tag{C.3}$$

$$= \bar{\boldsymbol{u}}_i^{M} - (t_{n+1} - t^M) \frac{\widehat{\boldsymbol{f}}_{i+1/2}^{HO,M} - \widehat{\boldsymbol{f}}_{i-1/2}^{HO,M}}{\Delta x_i} \tag{C.4}$$

$$= \boldsymbol{u}_{n,i} - \sum_{m=0}^{M} (t^m - t^{m-1}) \frac{\widehat{\boldsymbol{f}}_{i+1/2}^{HO,m-1} - \widehat{\boldsymbol{f}}_{i-1/2}^{HO,m-1}}{\Delta x_i}, \tag{C.5}$$

with the convention of $t^{-1} = t_n$.

Hence, the limiting strategy is equivalent to replacing the high-order fluxes $\widetilde{\boldsymbol{f}}^{m}_{i+1/2}$ in the subcells that trigger the aforementioned criteria with a low-order flux $\boldsymbol{f}^{LO,m}_{i+1/2}$. In this work, we use a simple first order Finite Volume scheme on the (space-time) subcell grid, referred as *parachute scheme*.

The space-time update of the SD-ADER method with limiting is shown in Algorithm 1. For further details, the interested reader is referred to [64]. Finally, we point out that the cost of the *a posteriori* limiter is proportional to the number of subtimenodes.

Input: Numerical solution $\boldsymbol{u}_n$ at t_n, Δt
Output: Numerical solution $\boldsymbol{u}_{n+1}$ at t_{n+1}
Initialize solution in the subtimenodes $\boldsymbol{u}^{m,(0)} = \boldsymbol{u}_n$ for $m = 0, ..., M$
Compute the high order update of the ADER-SD scheme up to the final iteration $\boldsymbol{u}^{m,(P)}$ for all $m = 0, \dots, M$
Convert the high-order fluxes into the subcell version $\widetilde{\boldsymbol{f}}^{HO,m}_{i+1/2} = \widetilde{\boldsymbol{f}}^{HO,m}_{i+1/2}(\boldsymbol{u}^{m,(P)})$ for $m = 0, ..., M$, $\forall i$,
for $m = 0, ..., M$ **do**
  Compute low-order fluxes using solution $\bar{\boldsymbol{u}}^m$ with parachute scheme
  Compute high-order candidate solution (shown i-th subcell) as in (C.2)
  Perform physical criteria check
  if $\tilde{\boldsymbol{u}}^{m+1}_i$ is *troubled* **then**
    Replace $\widetilde{\boldsymbol{f}}^{HO,m}_{i\pm1/2}$ with fluxes of parachute scheme fluxes $\widetilde{\boldsymbol{f}}^{LO,m}_{i\pm1/2}$ in (C.2)
    Replace surrounding fluxes $\widetilde{\boldsymbol{f}}^{HO,m}_{i\pm3/2}$ using parachute scheme fluxes $\widetilde{\boldsymbol{f}}^{LO,m}_{i\pm3/2}$ in (C.2)
  end
end
Compute $\boldsymbol{u}_{n+1,i}$ using (C.4) with HO or LO fluxes according to the previous criteria.

Algorithm 1: Timestep evolution of SD-ADER with the *a posteriori* limiting

Appendix D. Table of notation adopted in the paper

In Table D.11, we summarize the notation adopted for the main structures in the context of the ADER methods throughout the whole paper.

Bibliography

[1] R. Abgrall. High order schemes for hyperbolic problems using globally continuous approximation and avoiding mass matrices. *Journal of Scientific Computing*, 73(2-3):461–494, 2017.

[2] R. Abgrall, P. Bacigaluppi, and S. Tokareva. High-order residual distribution scheme for the time-dependent Euler equations of fluid dynamics. *Computers & Mathematics with Applications*, 78(2):274–297, 2019.

[3] R. Abgrall and M. Ricchiuto. *High-Order Methods for CFD*, pages 1–54. John Wiley & Sons, Ltd, 2017.

[4] R. Abgrall and D. Torlo. High order asymptotic preserving deferred correction implicit-explicit schemes for kinetic models. *SIAM Journal on Scientific Computing*, 42(3):B816–B845, 2020.

[5] P. Bacigaluppi, R. Abgrall, and S. Tokareva. a posteriori limited high order and robust schemes for transient simulations of fluid flows in gas dynamics. *Journal of Computational Physics*, 476:111898, 2023.

[6] D. S. Balsara, T. Rumpf, M. Dumbser, and C.-D. Munz. Efficient, high accuracy ADER-WENO schemes for hydrodynamics and divergence-free magnetohydrodynamics. *Journal of Computational Physics*, 228(7):2480–2516, 2009.

[7] W. Boscheri and D. S. Balsara. High order direct Arbitrary-Lagrangian-Eulerian (ALE) PNPM schemes with WENO Adaptive-Order reconstruction on unstructured meshes. *Journal of Computational Physics*, 398:108899, 2019.

[8] W. Boscheri and M. Dumbser. Arbitrary–Lagrangian–Eulerian One–Step WENO Finite Volume Schemes on Unstructured Triangular Meshes. *Communications in Computational Physics*, 14:1174–1206, 2013.

[9] W. Boscheri, M. Dumbser, and E. Gaburro. Continuous finite element subgrid basis functions for discontinuous Galerkin schemes on unstructured polygonal Voronoi meshes. *Communications in Computational Physics*, 32(1):259–298, 2022.

[10] W. Boscheri and R. Loubère. High order accurate direct Arbitrary-Lagrangian-Eulerian ADER-MOOD finite volume schemes for non-conservative hyperbolic systems with stiff source terms. *Communications in Computational Physics*, 21:271–312, 2017.

[11] W. Boscheri, R. Loubère, and M. Dumbser. Direct Arbitrary-Lagrangian-Eulerian ADER-MOOD finite volume schemes for multidimensional hyperbolic conservation laws. *Journal of Computational Physics*, 292:56–87, 2015.

[12] S. Busto, S. Chiocchetti, M. Dumbser, E. Gaburro, and I. Peshkov. High order ADER schemes for continuum mechanics. *Frontiers in Physics*, 8:32, 2020.

[13] J. C. Butcher. Implicit Runge-Kutta processes. *Mathematics of computation*, 18(85):50–64, 1964.

[14] J. C. Butcher. *Numerical methods for ordinary differential equations*. John Wiley & Sons, Auckland, 2016.

[15] F. Chipman. A-stable Runge-Kutta processes. *BIT Numerical Mathematics*, 11(4):384–388, 1971.

t_n	timenode
$\boldsymbol{u}_n$	approximation of the solution of (1) in t_n
Δt	timestep
N	accuracy of the ADER-IWF (5)
P	number of ADER iterations
$M+1$	number of subtimenodes
$t^m,\ m=0,\dots,M$	subtimenode in the interval $[t_n, t_{n+1}]$
$\boldsymbol{u}^m$	reconstruction coefficients of solution of (1)
ψ^m	Lagrange basis function associated to t^m
$\widehat{\psi}^m$	Lagrange basis function associated to ξ^m onto $[0,1]$
ξ^m	subtimenode t^m remapped into the reference interval $[0,1]$
ω_m	quadrature weight associated to $\widehat{\psi}^m$
ϕ^m	generic basis function in the interval $[t_n, t_{n+1}]$
$\widehat{\phi}^m$	generic basis function remapped into the reference interval $[0,1]$
$\underline{\beta}$	vector of the subtimenodes ξ^m
$\boldsymbol{u}_h$	interpolation of the numerical solution in the interval $[t_n, t_{n+1}]$
B, Λ	ADER matrices
$\underline{\boldsymbol{u}}, \underline{\boldsymbol{r}}, \underline{\boldsymbol{G}}(\underline{\boldsymbol{u}}), \underline{\boldsymbol{u}}_n, \underline{\boldsymbol{v}},$	ADER vectors
$\underline{\boldsymbol{u}}^{(p)},\ p=0,\dots,P$	vector of the approximations coefficients at the iteration p
$\mathcal{H}$	change of basis matrix
S	number of RK stages
$\boldsymbol{y}^s,\ s=0,\dots,S-1$	RK stage values
$a_{s,r}, b_r, c_r$	RK coefficients
$A, \boldsymbol{b}, \boldsymbol{c}$	RK structures
$\mathcal{L}^1_\Delta, \mathcal{L}^2_\Delta$	DeC operators
$\underline{\boldsymbol{u}}_\Delta$	solution of the $\mathcal{L}^2_\Delta$ operator
$\mathcal{L}^{1,(p)}_\Delta, \mathcal{L}^{2,(p)}_\Delta, \mathcal{E}^{(p)}, \Pi^{(p)}$	structures of efficient DeC methods
$\psi^{m,(p)}\quad 0,\dots,M^{(p)}$	basis functions of ADERu, ADERdu, ADER-L^2
$B^{(p)}, \Lambda^{(p)}, H^{(p)}, \underline{\boldsymbol{u}}_n^{(p)}, \underline{\boldsymbol{u}}^{*(p)}, \underline{\boldsymbol{G}}^{*(p)}, \Lambda^{(p,p-1)}, \underline{\boldsymbol{r}}^{(p)}, \underline{\beta}^{(p)}$	structures of ADERu, ADERdu, ADER-L^2

Table D.11: Table of symbols

[16] M. Ciallella, L. Micalizzi, P. Öffner, and D. Torlo. An arbitrary high order and positivity preserving method for the shallow water equations. *Computers & Fluids*, 247:105630, 2022.
[17] B. Cockburn, G. E. Karniadakis, and C.-W. Shu. The development of discontinuous Galerkin methods. In *Discontinuous Galerkin Methods*, pages 3–50. Springer, Berlin, 2000.
[18] K. V. den Abeele, C. Lacor, and Z. J. Wang. On the stability and accuracy of the spectral difference method. *Journal of Scientific Computing*, 37:162–188, 2008.
[19] M. Dumbser. Arbitrary high order PNPM schemes on unstructured meshes for the compressible Navier–Stokes equations. *Computers & Fluids*, 39(1):60–76, 2010.
[20] 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):8209–8253, 2008.
[21] 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):39714001, 2008.
[22] 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):63, 2018.
[23] M. Dumbser and C.-D. Munz. ADER discontinuous Galerkin schemes for aeroacoustics. *Comptes Rendus Mécanique*, 333(9):683–687, 2005.
[24] M. Dumbser, I. Peshkov, E. Romenski, and O. Zanotti. High order ader schemes for a unified first order hyperbolic formulation of newtonian continuum mechanics coupled with electro-dynamics. *Journal of Computational Physics*, 348:298–342, 2017.
[25] M. Dumbser and O. Zanotti. Very high order PNPM schemes on unstructured meshes for the resistive relativistic MHD equations. *Journal of Computational Physics*, 228(18):6991–7006, 2009.
[26] W. H. Enright and J. D. Pryce. Two FORTRAN packages for assessing initial value methods. *ACM Transactions on Mathematical Software (TOMS)*, 13(1):1–27, 1987.
[27] A. Ern and J.-L. Guermond. *Theory and practice of finite elements*, volume 159. Springer, New York, 2004.
[28] R. Eymard, T. Gallouët, and R. Herbin. Finite volume methods. *Handbook of numerical analysis*, 7:713–1018, 2000.
[29] E. G. Fernández, M. C. Díaz, M. Dumbser, and T. M. de Luna. An arbitrary high order well-balanced ADER-DG numerical scheme for the multilayer shallow-water model with variable density. *Journal of Scientific Computing*, 90(1):52, 2022.
[30] L. Fox and E. Goodwin. Some new methods for the numerical integration of ordinary differential equations. In *Mathematical Proceedings of the Cambridge Philosophical Society*, volume 45, pages 373–388. Cambridge University Press, 1949.
[31] E. Gaburro. A unified framework for the solution of hyperbolic PDE systems using high order direct Arbitrary-Lagrangian–Eulerian schemes on moving unstructured meshes with topology change. *Archives of Computational Methods in Engineering*, 28(3):1249–1321, 2021.
[32] E. Gaburro, W. Boscheri, S. Chiocchetti, C. Klingenberg, V. Springel, and M. Dumbser. High order direct Arbitrary-Lagrangian-Eulerian schemes on moving Voronoi meshes with topology changes. *Journal of Computational Physics*, 407:109167, 2020.
[33] E. Gaburro and M. Dumbser. A Posteriori Subcell Finite Volume Limiter for General PNPM Schemes: Applications from Gasdynamics to Relativistic Magnetohydrodynamics. *Journal of Scientific Computing*, 86(3):1–41, 2021.
[34] E. Gaburro, P. Öffner, M. Ricchiuto, and D. Torlo. High order entropy preserving ADER-DG scheme. *Applied Mathematics and Computation*, 440:127644, 2023.
[35] W. Gautschi. *Numerical analysis*. Springer Science & Business Media, Indiana, 2011.
[36] E. Godlewski and P.-A. Raviart. *Numerical approximation of hyperbolic systems of conservation laws*, volume 118. Springer Science & Business Media, New York, 2013.
[37] D. Gottlieb and J. S. Hesthaven. Spectral methods for hyperbolic problems. *Journal of Computational and Applied Mathematics*, 128(1-2):83–131, 2001.
[38] E. Hairer, S. Nørsett, and G. Wanner. *Solving Ordinary Differential Equations I: Nonstiff Problems*. Springer-Verlag, Berlin, 1987.
[39] M. Han Veiga, P. Öffner, and D. Torlo. Dec and Ader: similarities, differences and a unified framework. *Journal of Scientific Computing*, 87(1):1–35, 2021.
[40] J. S. Hesthaven and T. Warburton. *Nodal discontinuous Galerkin methods: algorithms, analysis, and applications*. Springer Science & Business Media, New York, 2007.
[41] A. Hidalgo and M. Dumbser. ADER schemes for nonlinear systems of stiff advection–diffusion–reaction equations. *Journal of Scientific Computing*, 48(1-3):173–189, 2011.
[42] J. Huang, J. Jia, and M. Minion. Accelerating the convergence of spectral deferred correction methods. *Journal of Computational Physics*, 214(2):633–656, 2006.
[43] A. Jameson. A proof of the stability of the spectral difference method for all orders of accuracy. *Journal of Scientific Computing*, 45:348–358, 2010.
[44] A. T. Layton and M. L. Minion. Implications of the choice of quadrature nodes for Picard integral deferred corrections methods for ordinary differential equations. *BIT Numerical Mathematics*, 45(2):341–373, 2005.
[45] R. J. LeVeque. *Finite difference methods for ordinary and partial differential equations: steady-state and time-dependent problems*. SIAM, Philadelphia, 2007.
[46] R. J. LeVeque et al. *Finite volume methods for hyperbolic problems*, volume 31. Cambridge university press, 2002.
[47] L. Micalizzi and D. Torlo. A new efficient explicit Deferred Correction framework: analysis and applications to hyperbolic PDEs and adaptivity. *Communications on Applied Mathematics and Computation*, 2023.
[48] L. Micalizzi, D. Torlo, and W. Boscheri. Efficient iterative arbitrary high order methods: an adaptive bridge between low and high order. *Communications on Applied Mathematics and Computation*, 2023.
[49] M. Minion. A hybrid parareal spectral deferred corrections method. *Communications in Applied Mathematics and Computational Science*, 5(2):265–301, 2011.
[50] M. L. Minion. Semi-implicit spectral deferred correction methods for ordinary differential equations. *Communications in Mathematical Sciences*, 1(3):471–500, 2003.
[51] M. L. Minion. Semi-implicit projection methods for incompressible flow based on spectral deferred corrections. *Applied numerical mathematics*, 48(3-4):369–387, 2004.
[52] P. Öffner and D. Torlo. Arbitrary high-order, conservative and positivity preserving Patankar-type deferred correction schemes. *Appl. Numer. Math.*, 153:15–34, 2020.
[53] L. Rannabauer, M. Dumbser, and M. Bader. ADER-DG with a-posteriori finite-volume limiting to simulate tsunamis in a parallel

adaptive mesh refinement framework. *Computers & Fluids*, 173:299–306, 2018.
[54] L. Río-Martín, S. Busto, and M. Dumbser. A massively parallel hybrid finite volume/finite element scheme for computational fluid dynamics. *Mathematics*, 9(18):2316, 2021.
[55] T. Schwartzkopff, M. Dumbser, and C.-D. Munz. Fast high order ADER schemes for linear hyperbolic equations. *Journal of Computational Physics*, 197(2):532–539, 2004.
[56] T. Schwartzkopff, C.-D. Munz, and E. F. Toro. ADER: A high-order approach for linear hyperbolic systems in 2d. *Journal of Scientific Computing*, 17(1-4):231–240, 2002.
[57] R. Speck, D. Ruprecht, M. Emmett, M. Minion, M. Bolten, and R. Krause. A multi-level spectral deferred correction method. *BIT Numerical Mathematics*, 55(3):843–867, 2015.
[58] V. Titarev and E. Toro. ADER: Arbitrary high order Godunov approach. *Journal of Scientific Computing*, 17(1-4):609–618, 2002.
[59] V. Titarev and E. Toro. ADER schemes for three-dimensional nonlinear hyperbolic systems. *Journal of Computational Physics*, 204:715–736, 2005.
[60] V. A. Titarev and E. F. Toro. ADER: Arbitrary high order Godunov approach. *Journal of Scientific Computing*, 17(1-4):609–618, 2002.
[61] E. Toro, R. Millington, and L. Nejad. Towards very high order Godunov schemes. In *Godunov methods*, pages 907–940. Springer, New York, 2001.
[62] J. Vanharen, G. Puigt, X. Vasseur, J.-F. Boussuge, and P. Sagaut. Revisiting the spectral analysis for high-order spectral discontinuous methods. *Journal of Computational Physics*, 337:379–402, 2017.
[63] M. H. Veiga, D. A. Velasco-Romero, Q. Wenger, and R. Teyssier. An arbitrary high-order spectral difference method for the induction equation. *Journal of Computational Physics*, 438:110327, 2021.
[64] D. A. Velasco Romero, M. Han-Veiga, and R. Teyssier. Spectral difference method with a posteriori limiting: application to the Euler equations in one and two space dimensions. *Monthly Notices of the Royal Astronomical Society*, 520(3):3591–3608, 2023.
[65] H. K. Versteeg and W. Malalasekera. *An introduction to computational fluid dynamics: the finite volume method*. Pearson education, Harlow, 2007.
[66] G. Wanner and E. Hairer. *Solving ordinary differential equations II: Stiff and Differential-Algebraic Problems*, volume 375. Springer Berlin Heidelberg, Berlin, 1996.
[67] O. Zanotti, F. Fambri, and M. Dumbser. Solving the relativistic magnetohydrodynamics equations with ADER discontinuous Galerkin methods, a posteriori subcell limiting and adaptive mesh refinement. *Monthly Notices of the Royal Astronomical Society*, 452(3):3010–3029, 2015.
[68] O. Zanotti, F. Fambri, M. Dumbser, and A. Hidalgo. Space–time adaptive ADER discontinuous Galerkin finite element schemes with a posteriori sub-cell finite volume limiting. *Computers & Fluids*, 118:204–224, 2015.