Efficient Multiple Time-Stepping Algorithms of Higher Order

Abdullah Demirel[a], Jens Niegemann[b], Kurt Busch[c], Marlis Hochbruck[a]

[a] *Institut für Angewandte und Numerische Mathematik, Karlsruher Institut für Technologie, 76128 Karlsruhe, Germany*
[b] *ETH Zurich, Institute of Electromagnetic Fields (IEF), 8092 Zurich, Switzerland*
[c] *Humboldt-Universität zu Berlin, Institut für Physik, AG Theoretische Optik & Photonik, and Max-Born-Institut, 12489 Berlin, Germany*

Abstract

Multiple time-stepping (MTS) algorithms allow to efficiently integrate large systems of ordinary differential equations, where a few stiff terms restrict the timestep of an otherwise non-stiff system. In this work, we discuss a flexible class of MTS techniques, based on multistep methods. Our approach contains several popular methods as special cases and it allows for the easy construction of novel and efficient higher-order MTS schemes. In addition, we demonstrate how to adapt the stability contour of the non-stiff time-integration to the physical system at hand. This allows significantly larger timesteps when compared to previously known multistep MTS approaches. As an example, we derive novel predictor-corrector (PCMTS) schemes specifically optimized for the time-integration of damped wave equations on locally refined meshes. In a set of numerical experiments, we demonstrate the performance of our scheme on discontinuous Galerkin time-domain (DGTD) simulations of Maxwell's equations.

Keywords: multiple time-stepping (MTS), local time-stepping (LTS), multistep methods, grid-induced stiffness, exponential integrator, Discontinuous Galerkin time-domain (DGTD), Maxwell's equations

1. Introduction

When following a method-of-lines approach, time-dependent partial differential equations (PDEs) are typically reduced to large systems of coupled ordinary differential equations (ODEs). Often, such a system of ODEs is then integrated by an explicit ODE solver [1, 2] to obtain the time evolution of the unknowns. While explicit solvers are easily implemented, this approach has the well-known problem of conditional stability. In many cases, the system contains only a few terms which force the entire simulation to take small timesteps. In the following, we assume that we can split the system of ODEs into stiff terms (which mandate a small timestep) and non-stiff terms (which permit a larger timestep) as

$$\frac{\mathrm{d}}{\mathrm{d}t}\mathbf{u}(t) = \mathbf{f}(t, \mathbf{u}(t)) + \mathbf{g}(t, \mathbf{u}(t)), \tag{1}$$

where **f** contains the stiff part and **g** contains the non-stiff part. In particular we assume **g** to be Lipschitz continuous.

Such split systems are very common in scientific and technical applications, where the origin of the stiffness can be vastly different. Possibly the most common reasons are

- *Algebraic stiffness*, which directly stems from large differences in the timescales of the underlying equations. Some examples are discretized convection-diffusion-reaction equations [3], molecular dynamics [4] or electrical circuits [5].
- *Grid-induced stiffness*, which occurs in the spatial discretization of partial differential equations (PDEs) with non-uniform meshes. Caused by small geometrical features or due to local refinement, some of the elements may require much smaller timesteps than the rest of the mesh.

A first mitigation to this problem is to employ solvers specifically designed for stiff systems, e.g., implicit solvers [6]. Unfortunately, if the number of unknowns is very large, the solution of large (and possibly nonlinear) systems of equations often is prohibitively expensive. An alternative approach towards a more efficient procedure is to treat $\mathbf{f}$ and $\mathbf{g}$ with different timesteps, i.e., carrying out many small timesteps for $\mathbf{f}$ and a few large timesteps for $\mathbf{g}$. This approach is often called multiple time-stepping (MTS) or local time-stepping (LTS). Of course, one still has to take the coupling between $\mathbf{f}$ and $\mathbf{g}$ into account which makes the construction of such techniques quite involved. Probably one of the first methods in this direction was published by Rice in 1960 [7] and started the field of multirate Runge-Kutta schemes. Similar ideas were later also applied to multistep methods [8]. Another popular approach is to employ coupled implicit (for $\mathbf{f}$) and explicit (for $\mathbf{g}$) methods, resulting in implicit-explicit (IMEX) schemes. Such schemes have been constructed for both multistep (see [9] and references therein) and Runge-Kutta methods [10]. For a brief but very recent review on general LTS techniques, also see [11]. For recent applications of LTS and IMEX schemes in the context of advection dominated problems, see [12, 13, 14, 15, 16, 17, 18].

In this paper, we present an alternative and fairly general strategy of constructing MTS methods. In contrast to most of the commonly used techniques, our approach allows to employ different time-stepping strategies for the inner (stiff) and the outer (non-stiff) integration. As an illustration, we will combine a predictor-corrector scheme for the outer integration with a low-storage Runge-Kutta method for the inner integration. This feature increases the flexibility in the design of MTS methods and, in certain cases, leads to a significant performance enhancement. Furthermore, we will show how to tailor the stability domain of the outer multistep integrator to the underlying physical problem. This can result in a significant increase in the largest stable timestep and therefore further improves the performance of our method. For certain choices of the inner integrator, our approach reduces to previously known schemes.

2. Multiple time-stepping methods

In the following, we will assume that the evaluation of the stiff function $\mathbf{f}$ is computationally cheap in comparison to the non-stiff function $\mathbf{g}$. This is usually the case for grid-induced stiffness, where a few small elements limit the stability of an otherwise large mesh.

2.1. General explicit multiple time-stepping methods

For the construction of our methods we follow an idea presented recently in [19] for semilinear problems, where $\mathbf{f}(t, \mathbf{u}(t)) = -A\mathbf{u}(t)$ with a square matrix A of possibly large dimension, i.e.,

$$\frac{\mathrm{d}}{\mathrm{d}t}\mathbf{u}(t) = -A\mathbf{u}(t) + \mathbf{g}(t, \mathbf{u}(t)), \quad \mathbf{u}(t_0) = \mathbf{u}_0. \tag{2}$$

A representation of the exact solution of (2) can be obtained from the variation of constants formula

$$\mathbf{u}(t_{n+1}) = e^{-hA}\mathbf{u}(t_n) + h\int_0^1 e^{-(1-\theta)hA}\mathbf{g}(t_n+\theta h, \mathbf{u}(t_n+\theta h))\,\mathrm{d}\theta, \tag{3}$$

where $t_n := t_0 + nh$, $n \in \mathbb{N}_0$. Note that the matrix exponential e^{-tA} is uniformly bounded by one for all $t \geq 0$ if the field of values of A is contained in the right complex half plane, see, e.g., [6, Section IV.11]. An approximation to $\mathbf{u}(t_{n+1})$ can be obtained by replacing the non-stiff function $\mathbf{g}(t, \mathbf{u}(t))$ in (3) by a local interpolation polynomial as

$$\mathbf{u}(t_{n+1}) \approx \mathbf{u}_{n+1} = e^{-hA}\mathbf{u}_n + h\int_0^1 e^{-(1-\theta)hA}\mathbf{p}_n(t_n+\theta h)\,\mathrm{d}\theta \tag{4}$$

and to computing the integral in (4) exactly. The resulting methods are known as exponential multistep methods of Adams-type. They have been first discussed in [20] and in a more systematic way in [21]. A rigorous error analysis was given in [19]. As noted in [19], the approximation $\mathbf{u}_n \approx \mathbf{u}(t_n)$ of an exponential multistep method of Adams type coincides with the exact solution $\mathbf{v}_n(h)$ of the following initial value problem

$$\frac{\mathrm{d}}{\mathrm{d}\tau}\mathbf{v}_n(\tau) = -A\mathbf{v}_n(\tau) + \mathbf{p}_n(t_n+\tau), \quad \tau \in [0,h], \quad \mathbf{v}_n(0) = \mathbf{u}_n. \tag{5}$$

Here, $\mathbf{p}_n(t)$ is defined as the local interpolation polynomial which approximates $\mathbf{g}(t, \mathbf{u}(t))$ on the time interval $[t_n, t_{n+1}]$. The key idea of these methods is based on the local polynomial approximation of the non-stiff function $\mathbf{g}(t, \mathbf{u}(t))$ instead of using a local polynomial approximation of the complete right-hand side, as in the classical case.

Now, we generalize this approach to fully nonlinear systems of the type given in Eq. (1). Following the same idea, we replace the non-stiff function $\mathbf{g}(t, \mathbf{u}(t))$ by a polynomial approximation

$$\mathbf{p}_n(t_n+\theta h) = \sum_{i=0}^{k-1}\mathbf{g}_{n-k+1+i}\sum_{j=0}^{p-1} b_{i,j}\frac{\theta^j}{j!}, \qquad \mathbf{g}_j := \mathbf{g}(t_j, \mathbf{u}_j), \tag{6}$$

with as yet unknown coefficients $b_{i,j} \in \mathbb{R}$. The resulting initial value problem on each timestep is then given by

$$\frac{\mathrm{d}}{\mathrm{d}\tau}\mathbf{v}_n(\tau) = \mathbf{f}(\tau, \mathbf{v}_n(\tau)) + \mathbf{p}_n(t_n+\tau), \quad \tau \in [0,h], \quad \mathbf{v}_n(0) = \mathbf{u}_n. \tag{7}$$

In contrast to the semilinear case, we are no longer able to compute the exact solution of (7). Instead, we follow an alternative idea also presented in [19]. Namely, we are using multiple time-stepping techniques to numerically integrate (7) on each timestep to obtain an approximation of $\mathbf{u}_{n+1}$.

In other words, once we have the k previous function values $\mathbf{g}_{n-k+1}, \ldots, \mathbf{g}_n$ available, we can apply *any* ODE solver to integrate (7) and obtain an approximation of $\mathbf{u}(t_{n+1})$. The key point is, that on the right-hand side of (7), we only require the evaluation of the stiff function $\mathbf{f}(\tau, \mathbf{y}(\tau))$ and the polynomial $\mathbf{p}_n(t_n+\tau)$. Due to the stiffness of $\mathbf{f}$, we have to make small timesteps (if we employ an explicit solver), but the evaluation of $\mathbf{f}$ at each timestep is assumed still to be cheap

Algorithm 1 General explicit k-step multiple time-stepping (EMTS(k,p)) method

```
Initialise: $n = 0$,  $t = t_0$,  $\mathbf{u}_n = \mathbf{u}_0$
if $k \geq 2$ then
    Compute $k$ starting values $\mathbf{u}_0, \ldots, \mathbf{u}_{k-1}$ (cf. Ch. 5.2)
    Evaluate $\mathbf{g}_0 = \mathbf{g}(t_0, \mathbf{u}_0), \ldots, \mathbf{g}_{k-1} = \mathbf{g}(t_{k-1}, \mathbf{u}_{k-1})$
    Set $n = n + k - 1$,  $t = t + (k-1)h$,  $\mathbf{u}_n = \mathbf{u}_{k-1}$
end if
while ($t <$ endTime) do
    Employ ODE solver of choice to solve (7) approximately and compute $\mathbf{u}_{n+1} \approx \mathbf{v}(h)$
    Evaluate $\mathbf{g}_{n+1} = \mathbf{g}(t_{n+1}, \mathbf{u}_{n+1})$
    Set $n = n + 1$,  $t = t + h$
end while
Output: Approximations $\mathbf{u}_i \approx \mathbf{u}(t_i)$ with $t_i = t_0 + ih$ and $i = 1, \ldots, n$
```

in comparison to the evaluation of $\mathbf{g}(t, \mathbf{u})$. Differently put, we can employ any ODE solver to integrate our stiff components from time t_n to t_{n+1}. Then, once the inner integration is finished, we evaluate the expensive (but non-stiff) function $\mathbf{g}_{n+1}$ to proceed with the outer integration.

The size of the inner timestep will depend on the scheme used for the inner integration and there are two basic options: either solve the stiff equation with an explicit scheme using smaller step sizes (this might be suitable if the "$\mathbf{g}$-part" constitutes the bottleneck) or using an efficient implicit method for the stiff equation (this might be interesting, if the structure of the problems allows to solve the nonlinear equations efficiently). In this work we will mainly focus on the first option.

For a method which is of order p, we call this algorithm a k-step explicit multiple time-stepping method (EMTS(k,p)) and summarize the procedure in Alg. 1. For a practical implementation, clearly some parts are still missing. Most importantly, we have not yet discussed how to obtain the coefficients $b_{i,j}$ such that we obtain a certain order p. We will return to this point in Section 3.

2.2. *General predictor-corrector multiple time-stepping methods*

For $\mathbf{f} = 0$, the explicit multiple time-stepping method (7) reduces to the classical multistep method of Adams type. As a consequence, we have to expect that the timestep h of the macro-step will *at least* be limited by the same stability criterion as the classical multistep method of Adams type. Unfortunately, for increasing order p, those methods experience dramatic timestep restrictions [1, 2]. For classical multistep methods, it is therefore often more efficient to employ predictor-corrector schemes which allow significantly larger timesteps.

Now, we will show how to extend our explicit multiple time-stepping method to a predictor-corrector multiple time-stepping scheme (PCMTS) to profit from the larger stability contour for the macro-step. To keep the discussion brief, we focus on a particular kind of PCMTS schemes, namely a sequence of prediction, evaluation, correction and evaluation (PECE).

Prediction. Compute a polynomial $\mathbf{p}_n$ of the form (6). Solve the ODE (7) by a suitable ODE solver to obtain $\widehat{\mathbf{u}}_{n+1} \approx \mathbf{v}_n(h)$ as an initial approximation of $\mathbf{u}(t_{n+1})$.

Evaluation. Compute

$$\widehat{\mathbf{g}}_{n+1} \approx \mathbf{g}(t_{n+1}, \widehat{\mathbf{u}}_{n+1}). \tag{8}$$

Correction. Construct a polynomial

$$\widehat{\mathbf{p}}_n(t_n + \theta h) = \sum_{i=1}^{k} \widehat{\mathbf{g}}_{n-k+1+i} \sum_{j=0}^{p-1} \widehat{b}_{i,j} \frac{\theta^j}{j!}, \qquad \text{where} \qquad \widehat{\mathbf{g}}_j = \mathbf{g}_j, \quad j \le n, \tag{9}$$

as a new local polynomial approximation of $\mathbf{g}(t_n + \theta h, \mathbf{u}(t_n + \theta h))$. Solve the initial value problem

$$\frac{\mathrm{d}}{\mathrm{d}\tau}\widehat{\mathbf{v}}_n(\tau) = \mathbf{f}(\tau, \widehat{\mathbf{v}}_n(\tau)) + \widehat{\mathbf{p}}_n(t_n + \tau), \quad \tau \in [0, h], \quad \mathbf{v}_n(0) = \mathbf{u}_n. \tag{10}$$

by a suitable ODE solver to obtain the approximation $\mathbf{u}_{n+1} \approx \mathbf{v}(h) \approx \mathbf{u}(t_{n+1})$.

Evaluation. Finally, evaluate

$$\mathbf{g}_{n+1} = \mathbf{g}(t_{n+1}, \mathbf{u}_{n+1}). \tag{11}$$

The PCMTS algorithm is briefly summarized in Alg. 2.

Algorithm 2 General k-step predictor-corrector multiple time-stepping (PCMTS(k,p)) method

```
Initialise: $n = 0$,   $t = t_0$,   $\mathbf{u}_n = \mathbf{u}_0$

if $k \ge 2$ then
  Compute $k$ starting values $\mathbf{u}_0, \dots, \mathbf{u}_{k-1}$ (cf. Section 5.2)
  Evaluate $\mathbf{g}_0 = \mathbf{g}(t_0, \mathbf{u}_0), \dots, \mathbf{g}_{k-1} = \mathbf{g}(t_{k-1}, \mathbf{u}_{k-1})$
  Set $n = n + k - 1$,   $t = t + (k-1)h$,   $\mathbf{u}_n = \mathbf{u}_{k-1}$
end if
while ($t <$ endTime) do
  Solve the predictor equation (7) by a suitable ODE solver to compute $\widehat{\mathbf{u}}_{n+1} \approx \mathbf{v}_n(h)$
  Evaluate $\widehat{\mathbf{g}}_{n+1} = \mathbf{g}(t_{n+1}, \widehat{\mathbf{u}}_{n+1})$
  Solve the corrector equation (10) by a suitable ODE solver to compute $\mathbf{u}_{n+1} \approx \widehat{\mathbf{v}}_n(h)$
  Evaluate $\mathbf{g}_{n+1} = \mathbf{g}(t_{n+1}, \mathbf{u}_{n+1})$
  Set $n = n + 1$,   $t = t + h$
end while
Output: Approximations $\mathbf{u}_i \approx \mathbf{u}(t_i)$ with $t_i = t_0 + ih$ and $i = 1, \dots, n$
```

Remark 1. *For* $\mathbf{f} = 0$*, the EMTS(k,p) and the PCMTS(k,p) methods reduce to the classical explicit Adams methods and the classical predictor-corrector methods, if the polynomials* $\mathbf{p}_n$ *and* $\widehat{\mathbf{p}}_n$ *are chosen as the local interpolation polynomials. Then, the systems of ODEs* (7) *and* (10) *can be integrated exactly and yield the approximations*

$$\widehat{\mathbf{u}}_{n+1} = \mathbf{u}_n + h \sum_{i=0}^{k-1} \beta_i \, \mathbf{g}_{n-k+1+i}, \qquad \mathbf{u}_{n+1} = \mathbf{u}_n + h \sum_{i=1}^{k} \widehat{\beta}_i \, \widehat{\mathbf{g}}_{n-k+1+i} \tag{12}$$

with coefficients

$$\beta_i = \sum_{j=0}^{p-1} \frac{b_{i,j}}{(j+1)!} \qquad \text{and} \qquad \widehat{\beta}_i = \sum_{j=0}^{p-1} \frac{\widehat{b}_{i,j}}{(j+1)!}. \tag{13}$$

Remark 2. *Note that the choice of a PCMTS(k,p) scheme with an equal number of steps k for both the predictor and the corrector was made to keep the discussion compact. The above procedure can be easily generalized to any predictor-corrector method such as to P(EC)^m or P(EC)^{m}E modes, see [2].*

Remark 3. *For certain choices of the inner integrator, our schemes reduce to previously known methods. In particular, when combining EMTS(k,k) with an explicit Adams type integrator for the inner time-stepping, we recover the LTS-AB schemes discussed in [16]. For the choice of an implicit multistep method as an inner integrator, we obtain a number of well known multistep IMEX schemes. For example, using $p = k = 2$ and the Crank-Nicolson scheme as an inner integrator, we recover the "Crank-Nicolson Adams-Bashforth" (CNAB) method, see [9] and references therein.*

Remark 4. *Since the choice of the ODE solver used to solve* (7) *and* (10) *is arbitrary, one can employ our method recursively by using either an EMTS(k,p) or a PCMTS(k,p)-scheme for the inner integration.*

3. Order conditions

With both, the EMTS(k,p) and the PCMTS(k,p) methods at hand, we now discuss how to find suitable coefficients $b_{i,j}$ and $\widehat{b}_{i,j}$. As in the case of classical multistep methods, a certain order p will lead to a set of linear equations for the coefficients $b_{i,j}$ and $\widehat{b}_{i,j}$. To find those conditions, we return to the semilinear case analyzed in [19]. The complete error analysis for nonlinear problems of type (1) is a topic of future work.

Similarly to the order conditions of the exponential quadrature methods [22], it is possible to compute order conditions for general exponential multistep methods. We can conclude from the results in [22] and [23] that Theorem 5 holds for the order conditions of a general explicit multiple time-stepping method.

Theorem 5. *Let* $\mathbf{u}$ *be the solution of* (2) *and assume that the function* $\mathbf{z}(t) := \mathbf{g}(t, \mathbf{u}(t))$ *is sufficiently smooth. Then, a general explicit multiple time-stepping method is consistent of order p, if the order conditions*

$$\delta_{l,j} = \sum_{i=0}^{k-1} \frac{b_{i,j}(i+1-k)^l}{l!} \quad \text{with} \quad j = 0, \ldots, p-1 \quad \text{and} \quad \delta_{l,j} = \begin{cases} 1, & l = j \\ 0, & l \neq j \end{cases} \tag{14}$$

are fulfilled for $l = 0, \ldots, p-1$.

Proof By definition of consistency order we have to show that $\mathbf{u}(t_k) - \mathbf{u}_k = O(h^{p+1})$ if the numerical approximation $\mathbf{u}_k$ is computed from exact starting values

$$\mathbf{u}_j = \mathbf{u}(t_j), \qquad j = 0, \ldots, k-1.$$

By using the definition of the φ-functions

$$\varphi_j(z) := \int_0^1 e^{(1-\theta)z} \frac{\theta^{j-1}}{(j-1)!} \, \mathrm{d}\theta \quad \text{with} \quad \varphi_j(0) = \frac{1}{j!} \quad \text{and} \quad j \in \mathbb{N},$$

we obtain the Taylor expansion of the exact solution (3) as

$$\begin{aligned} \mathbf{u}(t_k) &= e^{-hA}\mathbf{u}(t_{k-1}) + h\sum_{l=0}^{p-1}\int_0^1 e^{-(1-\theta)hA}\mathbf{z}^{(l)}(t_{k-1})\frac{(\theta h)^l}{l!}\,\mathrm{d}\theta + O(h^{p+1}) \\ &= e^{-hA}\mathbf{u}(t_{k-1}) + \sum_{l=0}^{p-1} h^{l+1}\varphi_{l+1}(-hA)\mathbf{z}^{(l)}(t_{k-1}) + O(h^{p+1}). \end{aligned} \tag{15}$$

For the numerical solution (4) we have

$$\mathbf{u}_k = e^{-hA}\mathbf{u}_{k-1} + h\sum_{i=0}^{k-1}\sum_{j=0}^{p-1} b_{i,j}\varphi_{j+1}(-hA)\mathbf{z}(t_i)\,\mathrm{d}\theta. \tag{16}$$

This yields

$$\mathbf{u}(t_k) - \mathbf{u}_k = \sum_{l=0}^{p-1} h^{l+1}\left(\varphi_{l+1}(-hA) - \sum_{i=0}^{k-1}\sum_{j=0}^{p-1}\frac{b_{i,j}(i+1-k)^l}{l!}\varphi_{j+1}(-hA)\right)\mathbf{z}^{(l)}(t_{k-1}) + O(h^{p+1}). \tag{17}$$

The order conditions now follow from the φ-functions being linearly independent.

Note that the constants in the O-terms above depend on bounds of the derivatives of $\mathbf{z}$ only but not on $\|A\|$.

If the assumptions of this theorem are satisfied, the method converges of order p. This can be shown exactly as in the proof of [19, Theorem 4.3] for exponential Adams methods.

□

Remark 6. *Note that for $A = 0$, exponential multistep methods reduce to classical multistep methods. Obviously, the order conditions then reduce to the classical order conditions so that both schemes are of the same classical (non-stiff) order.*

For explicit exponential multistep methods, the order conditions can be written in compact form as

$$\mathcal{V}\mathcal{B} = \mathcal{I}_p, \tag{18}$$

where $\mathcal{I}_p$ denotes the $p\times p$ identity matrix and

$$\mathcal{B} = \begin{pmatrix} b_{0,0} & \dots & b_{0,p-1} \\ \vdots & \vdots & \vdots \\ b_{k-1,0} & \dots & b_{k-1,p-1} \end{pmatrix} \quad \text{and} \quad \mathcal{V} = (\mathcal{V}_{l,i}), \quad \mathcal{V}_{l,i} = \frac{(i-k)^l}{(l-1)!}. \tag{19}$$

For $k = p$, the Vandermonde matrix $\mathcal{V}$ is invertible so that the coefficients of $\mathcal{B}$ are uniquely determined. In the case of $k > p$, we will show below how the $p(k-p)$ free parameters can be used to optimize the stability region of the method.

4. Tailoring the Stability Contours

As discussed above, in the absence of stiffness ($\mathbf{f} = 0$) our MTS methods reduce to the corresponding classical multistep methods. Thus, the timestep of our outer integration is at least limited by the same stability criteria as the classical methods. If we manage to increase the stability domain of the underlying multistep method, we can expect to also increase the stability of the resulting MTS scheme.

In [24], the authors increased the stability domain of a PE(CE)^m scheme by increasing the number of correction steps m. Here, we instead increase the number of steps k while maintaining a given order p. Following a similar optimization procedure as recently employed to low-storage Runge-Kutta methods [25] allows us to increase the stability domains of multistep methods significantly. In the classical case, we have $p - k$ free parameters which can be used to optimize the stability contour. It is important to note, that in our case an increase of k implies an increased memory consumption because all k previous steps need to be stored. Therefore, in the following we will limit the increase to $k \leq 10$.

4.1. Stability condition of classical multistep methods of Adams type

For a classical Adams type scheme

$$\mathbf{u}_{n+1} = \mathbf{u}_n + h \sum_{i=0}^{k-1} \beta_i \, \mathbf{g}_{n-k+1+i}, \tag{20}$$

a linear stability analysis [6, Section V.1] yields the difference equation

$$w^k - \sum_{i=0}^{k-1} \left(\delta_{i,k-1} + z\beta_i\right) w^i = 0. \tag{21}$$

Here, we have $z = h\lambda$ and $\delta_{i,j}$ denotes the usual Kronecker delta. Now, an Adams type scheme is linearly stable if and only if for each eigenvalue λ of the Jacobian $J = \partial \mathbf{g} / \partial \mathbf{u}$ the roots w_i of (21) fulfill the following condition:

$$\begin{aligned} |w_i| &\leq 1, \quad \text{if } w_i \text{ is a single root,} \\ |w_i| &< 1, \quad \text{if } w_i \text{ is a multiple root.} \end{aligned} \tag{22}$$

4.2. Stability condition of the classical PECE method

For the classical PECE method (12) with a k-step predictor and a k-step corrector a similar stability analysis [2] yields the difference equation

$$\left(w^k - w^{k-1}\right)\left(1 - z\widehat{\beta}_k\right) - z\left(\sum_{i=1}^{k} \widehat{\beta}_i w^i\right) - z^2\left(\widehat{\beta}_k \sum_{i=0}^{k-1} \beta_i w^i\right) = 0. \tag{23}$$

As for the classical Adams type schemes, a classical PECE scheme is linearly stable if and only if for each eigenvalue λ of the Jacobian $J = \partial \mathbf{g} / \partial \mathbf{u}$ the roots w_i of (23) fulfill the condition (22).

4.3. Optimization of the classical stability contours

Following the ideas of [25] we now aim to optimize the classical coefficients β_i (and in case of a PECE scheme also $\widehat{\beta_i}$) such that the stability domain is maximized. To keep the discussion brief, we only describe the procedure for the Adams type scheme, the optimization of a PECE method is performed analogously.

To start, we choose a target domain $\Lambda_{\text{target}}^{\text{opt}}$ which resembles the shape of the convex hull of the spectrum of A or of the field of values of A. We discretize the boundary of $\Lambda_{\text{target}}^{\text{opt}}$ with a finite number of points. Next, we fix the desired order p of the multistep method and pick a number of steps $k > p$. Our aim is to determine the coefficients $b_{i,j}$ such that the stability region of the corresponding multistep method contains the target domain.

This can be stated as the following optimization problem:

Problem 1. Given k, $p < k$, and $\Lambda_{\text{target}}^{\text{opt}} \subset \mathbb{C}$,

$$\begin{aligned} \underset{\beta_1 \ldots \beta_k}{\text{maximize}} \quad & h \\ \text{subject to} \quad & \beta_i \text{ fulfill the order conditions to order } p, \\ & \text{condition (22) holds for all } \lambda \in \Lambda_{\text{target}}^{\text{opt}} \end{aligned}$$

For the actual optimization, we use the Controlled Random Search with local mutation (CRS2) method [26] as implemented in the library `NLopt` [27]. This method is a derivative-free global optimization technique which was used for similar optimizations in [25]. Typically, for less than 10 free parameters, a good solution is found within minutes on a standard desktop computer.

4.4. An Example for Damped Wave Equations

Following the work in [25], as our target spectrum we pick a shifted circular shape given by

$$\Lambda_{\text{circle}} = \{\lambda \in \mathbb{C} \mid \operatorname{Re}[\lambda] \leq 0 \wedge |\lambda + \lambda_0| < 1\}. \tag{24}$$

with shift $\lambda_0 = \cos\left(\sin^{-1}\left(\frac{1}{2}\right)\right)$. This target region resembles the spectra of linear hyperbolic partial differential equations when discretized via a discontinuous Galerkin approach using an upwind flux [28].

For the optimization we select 50 equidistant points on the contour of Λ_{circle}. We then run a set of optimizations for different orders $p \in \{3, 4\}$, for various numbers of steps $k \in \{p + 1, \ldots, 10\}$ and for both, a classical Adams type and a PECE scheme. It should be noted that the PECE method requires *two* evaluations of the function $\mathbf{g}$ while the Adams type scheme only needs one. If we assume that those evaluations dominate the computational cost, it makes sense to introduce an effective timestep $h_{\text{eff}} = h/n_{\text{eval}}$ which is normalized by the number of function evaluations. In Figure 1, we show the largest stable effective timestep found for the classical Adams (dashed lines) and a PECE method (solid lines) as a function of the number of steps k. Here, the first point of each line where $k = p$ corresponds to the largest stable timestep of the classical scheme.

As expected, we observe that for the same number of steps k, the PECE schemes allow significantly larger effective timesteps. Therefore, in the following we will concentrate on predictor-corrector methods. As can be seen in Figure 1, by increasing k from $k = 3$ to $k = 6$, we obtain roughly a 40% increase in the largest stable h_{eff} for the third-order PECE method. Increasing

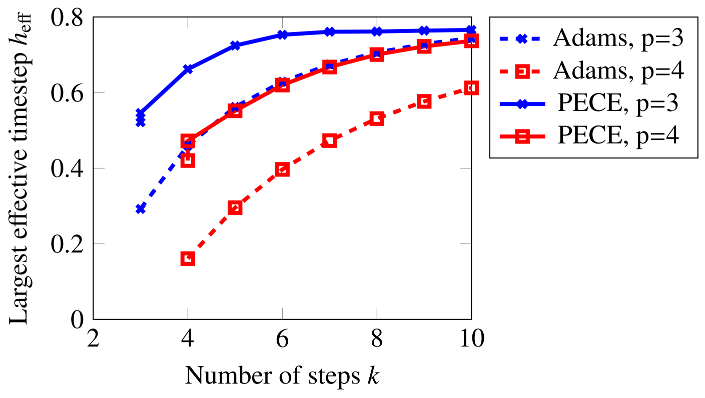


Figure 1: Largest stable effective timestep h_{eff} found for classical Adams and PECE methods of orders $p = 3, 4$.

k further does not lead to an additional increase of h_{eff}. In the case of $p = 4$, we find a 65% increase when using $k = 8$ instead of the classical method with $k = 4$. The relative stability contours of the optimized methods are shown in Figure 2(a) and Figure 2(b) for $p = 3$ and $p = 4$, respectively.

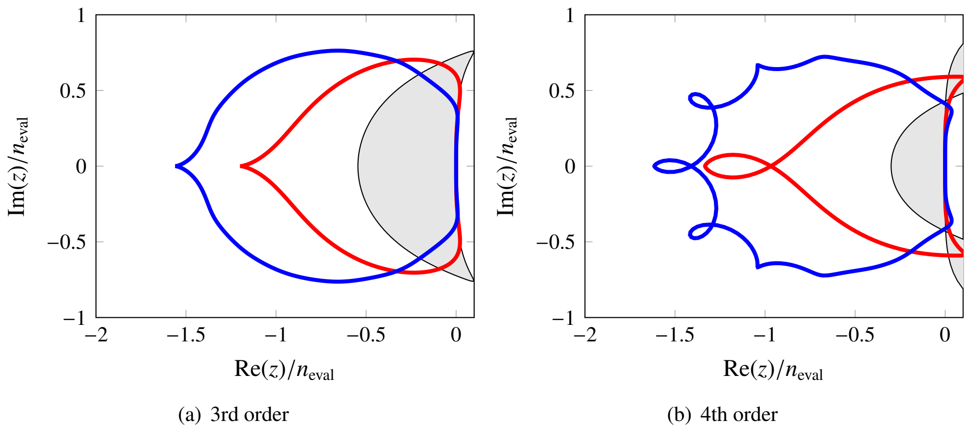


Figure 2: Optimized PECE stability contours for $k = 8$ (blue) in comparison with the standard ($k = p$) PECE methods (red). The stability region of the classical Adams method is sketched in gray.

Again, the contours in Figure 2(a) and Figure 2(b) were normalized by the number of function evaluations n_{eval} per step to allow a fair comparison between the Adams type and the predictor-corrector schemes.

5. Details of the Implementation

5.1. Generation of Optimized MTS Coefficients

In order to generate MTS schemes, which have optimized stability contours for the outer integration, we now combine the order conditions (18) with the optimized classical schemes discussed in Section 4. In particular, we use the optimized β-values from the classical Adams type schemes as an additional constraint to the free parameters in the matrix $\mathcal{B}$.

If $k > p$ we introduce $n = k - p$ and split the matrix equation (18) as

$$\left(\mathcal{V}_{\text{left}}^{p\times n} \;\middle|\; \mathcal{V}_{\text{right}}^{p\times p}\right)\left(\begin{array}{c}\mathcal{B}_{\text{top}}^{n\times p}\\ \hline \mathcal{B}_{\text{bottom}}^{p\times p}\end{array}\right) = \mathcal{I}_p \tag{25}$$

and solve for $\mathcal{B}_{\text{bottom}}$ to find

$$\mathcal{B}_{\text{bottom}} = \mathcal{V}_{\text{right}}^{-1}\left(\mathcal{I}_p - \mathcal{V}_{\text{left}}\mathcal{B}_{\text{top}}\right). \tag{26}$$

Thus, for a given $n \times p$-matrix $\mathcal{B}_{\text{top}}$, we can easily compute $\mathcal{B}_{\text{bottom}}$ and thus obtain the full matrix $\mathcal{B}$ of MTS coefficients. From comparison with the classical order conditions (13), we know that

$$\mathcal{B}\mathbf{v} = [\beta_0, \ldots, \beta_{p-1}]^T \quad \text{with} \quad v_j = \frac{1}{(j+1)!}, \quad j = 0, \ldots, p-1$$

describes the relation to the classical coefficient β_i. Thus, to incorporate an optimized set of optimized $\beta^{(\text{opt})}$-coefficients, we can simply use $\left(\beta_0^{(\text{opt})}, \ldots, \beta_{p-1}^{(\text{opt})}\right)^T$ as the first column of $\mathcal{B}_{\text{top}}$ and set the remaining entries to zero. Inserting this into (26) then yields $\mathcal{B}_{\text{bottom}}$ and the total matrix $\mathcal{B}$ takes the form

$$\mathcal{B} = \left(\begin{array}{cc} \begin{array}{c}\beta_0^{(\text{opt})}\\ \vdots\\ \beta_{p-1}^{(\text{opt})}\end{array} & 0 \\ \hline \mathcal{B}_{\text{bottom}} & \end{array}\right).$$

For a predictor-corrector scheme, the two matrices $\mathcal{B}$ and $\widehat{\mathcal{B}}$ are constructed independently from the corresponding coefficients $\beta^{(\text{opt})}$ and $\widehat{\beta}^{(\text{opt})}$. The coefficient matrices of the PCMTS(6,3) and PCMTS(8,4) schemes optimized for the damped wave equation are given in the appendix as (A.1) and (A.2), respectively.

5.2. *Computation of starting values*

Since our MTS schemes are based on multistep methods, we are facing the problem of initializing the vectors $\mathbf{g}_0, \ldots, \mathbf{g}_{k-1}$ for the first step. Possibly the simplest solution is to employ a self-starting method, e.g., a Runge-Kutta scheme, with a sufficiently small timestep to initialize the starting values. Indeed, this procedure is commonly used in practical implementations because of its simplicity. However, it is not trivial to properly select the timestep of the self-starting method to guarantee the required accuracy of the starting values. Therefore, we also present the alternative initialization from [22] which is based on a construction by Calvo and Palencia in [29].

For the semilinear system (2), we again start from the variation of constants formula

$$\mathbf{u}(t_m) = e^{-mhA}\mathbf{u}(t_0) + h\int_0^m e^{-(m-\theta)hA}\mathbf{g}(t_0+\theta h, \mathbf{u}(t_0+\theta h))\mathrm{d}\theta \quad \text{with} \quad m = 1,\dots,k-1,$$

and replace the nonlinear function $\mathbf{g}$ by a local interpolation polynomial $\mathbf{p}_0$ through the points $(t_0, g(t_0, \mathbf{u}_0)), \dots, (t_{k-1}, g(t_{k-1}, \mathbf{u}_{k-1}))$. This leads to a system of coupled equations

$$\mathbf{u}(t_m) = e^{-mhA}\mathbf{u}(t_0) + h\int_0^m e^{-(m-\theta)hA}\mathbf{p}_0(t_0+\theta h)\mathrm{d}\theta \quad \text{for} \quad m = 1,\dots,k-1. \tag{27}$$

for the unknown vectors $\mathbf{u}_m \approx \mathbf{u}(t_m)$ with $m = 1,\dots,k-1$. The Lipschitz condition on $\mathbf{g}$ and the boundedness of the matrix exponential implies that the right-hand side of (27) is a contraction for sufficiently small stepsizes h. Thus, one can use a simple fixpoint iteration to solve the system. As an initial guess, we use $\mathbf{p}_0 = \mathbf{u}_0$, which corresponds to the application of the exponential Euler method.

To generalize this procedure to nonlinear ODEs, we follow a similar approach as in Section 2. We consider a system of auxiliary ODEs

$$\frac{\mathrm{d}}{\mathrm{d}\tau}\mathbf{v}_0(\tau) = \mathbf{f}(\tau, \mathbf{v}_0(\tau)) + \mathbf{p}_0(t_0+\tau), \quad \tau \in [0, (k-1)h], \quad \mathbf{v}_0(0) = \mathbf{u}_0, \tag{28}$$

where we again replace the function $\mathbf{g}$ corresponding to a non-stiff ODE by the local interpolation polynomial $\mathbf{p}_0$ through the points $(t_0, \mathbf{g}_0), \dots, (t_{k-1}, \mathbf{g}_{k-1})$. The starting values $\mathbf{u}_m = \mathbf{u}(t_m)$, $m = 1,\dots,k-1$ are then found by repeatedly solving (28) with an arbitrary self-starting ODE solver. As in the semilinear case, we start with $\mathbf{p}_0 = \mathbf{u}_0$ and compute the approximate solutions $\mathbf{u}_m \approx \mathbf{v}_0(mh)$. From those values, we construct a new polynomial $\mathbf{p}_0$ through the points $(t_0, \mathbf{g}_0), \dots, (t_{k-1}, \mathbf{g}_{k-1})$ and again solve (28). This procedure is repeated until convergence. Our initialization routine is summarized in Algorithm 3.

Algorithm 3 Computation of the starting values via MTS

```
Input: $t_0$, $\mathbf{u}_0$, $h$, tol, $k$
Set $\mathbf{p}_0 = \mathbf{u}_0$
Employ self-starting ODE solver of choice to solve (28) and compute $\mathbf{u}_m \approx \mathbf{v}(mh)$ for $m = 1,\dots,k-1$
while error > tol do
    Compute local interpolation polynomial $\mathbf{p}_0$ through the points $(t_j, \mathbf{g}_j)$, $j = 0,\dots,k-1$
    Employ a suitable ODE solver to solve (28) and update $\mathbf{u}_m \approx \mathbf{v}(mh)$ for $m = 1,\dots,k-1$
end while
Output: Starting values $\mathbf{u}_1,\dots,\mathbf{u}_{k-1}$.
```

In contrast to the direct computation with a self-starting method, we can use the maximum of the relative change of the vectors $\mathbf{u}_m$, $m = 1,\dots,k-1$ as an error indicator. By repeating this procedure until the error is below a specified tolerance for each $\mathbf{u}_m$, we can ensure the accuracy of the starting values. In practice we used `tol` $= \max\{10^{-10}, 10h^p\}$ for the maximum of the relative change of $\mathbf{u}_1, \mathbf{u}_2, \dots, \mathbf{u}_{k-1}$ as a stopping criterion for a method or order p.

Remark 7. *In case of using a multistep procedure for the micro-step, we also require starting values for the inner integrator. However, these values can be readily computed by the ODE solver during the final iteration of the algorithm. We simply need to enforce the computation of the approximations at the required points.*

6. Numerical Verification

6.1. Verification of the Order

In a first experiment, we consider the nonlinear system of ODEs

$$\frac{\mathrm{d}}{\mathrm{d}t}u(t) = \frac{1}{u} - v\frac{\exp(t^2)}{t^2} - t, \tag{29a}$$

$$\frac{\mathrm{d}}{\mathrm{d}t}v(t) = \frac{1}{v} - \exp(t^2) - 2t\exp(-t^2). \tag{29b}$$

which was also used for a similar purpose in [25, 30]. The analytic solution of this system of equations for $t \geq 1$ and for initial conditions $u(1) = 1$, $v(1) = \exp(-1)$ is given by

$$u(t) = \frac{1}{t}, \qquad v(t) = \exp(-t^2). \tag{30}$$

In all the following calculations, we numerically integrate (29) from $t_0 = 1$ to $t_{\mathrm{end}} = 1.4$ for various step sizes h. To study the error of the time-integration, we define the total error after the simulation as

$$\mathcal{E} = |\tilde{u}(t_{\mathrm{end}}) - u(t_{\mathrm{end}})| + |\tilde{v}(t_{\mathrm{end}}) - v(t_{\mathrm{end}})|\,, \tag{31}$$

where $\tilde{u}$ and $\tilde{v}$ refer to the numerical results while u and v represent the analytic solutions (30).

To test our MTS schemes, we employ two different splittings of (29):

$$\mathbf{f}_1 = \left(\frac{1}{u}, \frac{1}{v}\right)^T, \qquad \mathbf{g}_1 = \left(-v\frac{\exp(t^2)}{t^2} - t, -\exp(t^2) - 2t\exp(-t^2)\right)^T, \tag{32a}$$

$$\mathbf{f}_2 = \left(\frac{1}{u} - v\frac{\exp(t^2)}{t^2} - t, 0\right)^T, \qquad \mathbf{g}_2 = \left(0, \frac{1}{v} - \exp(t^2) - 2t\exp(-t^2)\right)^T. \tag{32b}$$

It should be noted that for either splitting, $\mathbf{f}$ contains nonlinear terms. For the inner integrator we use an embedded Runge-Kutta(4,5) pair as implemented in the routine `ode45` of `MATLAB`.

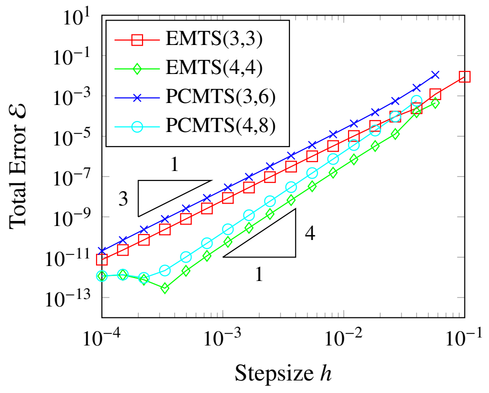
The error $\mathcal{E}$ as a function of the stepsize h is depicted in Figure 3(a) and Figure 3(b) for splitting (32a) and splitting (32b), respectively. In all cases, we observe the reduction of the error with the expected order. This indicates that our schemes maintain their order also for fully nonlinear problems.

6.2. Finite-Difference Discretization of the 1D Heat Equation

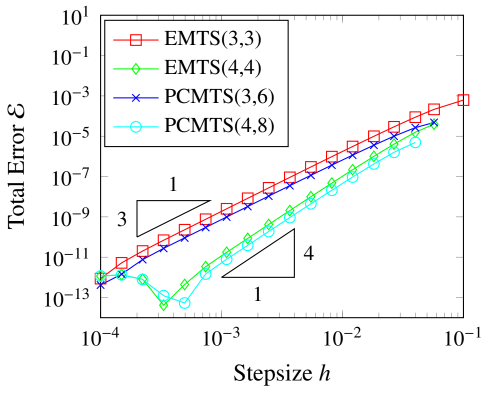
In this second example, we consider the one-dimensional heat equation

$$\frac{\partial}{\partial t}u(x,t) = \Delta u(x,t), \quad \mathbf{u}(x,t_0) = \mathbf{u}_0(x) = \sin\left(\frac{x\pi}{6}\right)$$

over the domain $\Omega = [0, 6]$ with homogeneous Dirichlet boundary conditions and the time interval $\mathcal{T} = [0, T]$ with $T = 10$. Furthermore, the domain Ω is locally refined as shown in Figure 4.



(a) Splitting (32a)



(b) Splitting (32b)

Figure 3: Total error $\mathcal{E}$ as a function of the outer timestep h for problem (30)

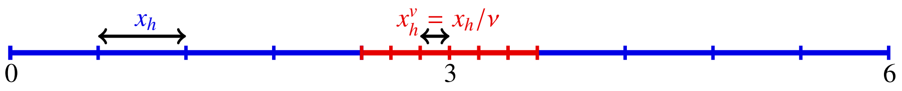


Figure 4: Locally refined domain $\Omega = [0, 6]$, where ν denotes the refinement factor of the element size.

The spatial discretization via finite differences leads to the following semi-discrete problem

$$\frac{\mathrm{d}}{\mathrm{d}t}\mathbf{u}(t) = A\mathbf{u}(t), \quad \mathbf{u}(t_0) = \mathbf{u}_0 = \sin\left(\frac{x\pi}{6}\right) \tag{33}$$

with a band-diagonal matrix A. The exact solution of (33) can be written as

$$\mathbf{u}(t) = e^{tA}\mathbf{u}_0. \tag{34}$$

The Laplace operator Δ and also its discretization A are symmetric negative semidefinite. Therefore, the spectra of the considered heat equation are real and contained in the interval $(-\infty, 0]$. Due to this fact, it is not ideal to use a method with a circular stability contour as discussed in Section 4.4. To demonstrate the flexibility of our approach, we instead construct methods with a rectangular target spectrum of $[-6, 0] \times [-0.05, 0.05]$. In particular, we generate optimized EMTS(8,4) and PCMTS(8,4) scheme with coefficients given in Appendix A by (A.3) and (A.4).

As a reference solution, we compute the matrix exponential (34) via diagonalization of the matrix and measure the error in the maximum norm, i.e., we consider the error of the time integration scheme only. In Fig. 5, we plot this error for EMTS(8,4) (blue) and for PCMTS(8,4) (red) as a function of the timestep h. From the slope of the error plot, we clearly observe that our methods are of order four. In both schemes we used the classical Runge-Kutta method of order four as the inner integrator with stepsize $\tau = h/(2\nu^2)$. For comparison, we also performed the classical Runge–Kutta method. Due to the CFL condition, it is stable only for the smallest four stepsizes used in Fig. 5 and then achieves an accuracy of about 10^{-14}. We omitted these results in the figure for the purpose of readability.

For the analysis of the stability properties, we computed the largest stable stepsize $h_{\max}$ of these methods via bisection for this particular problem. For comparison, we also computed $h_{\max}$

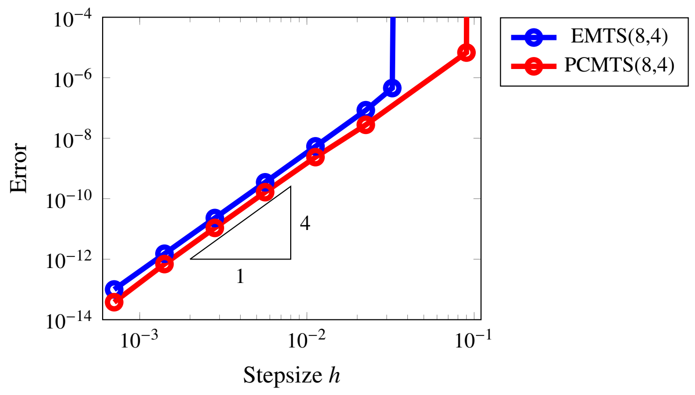


Figure 5: The logarithmic order plot shows that EMTS(8,4) and PCMTS(8,4) are of order 4. Here, the refinement factor was chosen as $\nu = 4$.

of the classical and exponential Adams methods of order 4 and the classical and exponential predictor-corrector methods of order 4 with 4 steps. The exponential versions of the corresponding classical methods can be constructed as described in [19]. Alternatively, the exponential schemes can also be considered as MTS schemes with an exact inner integrator. The results are shown in Figure 6.

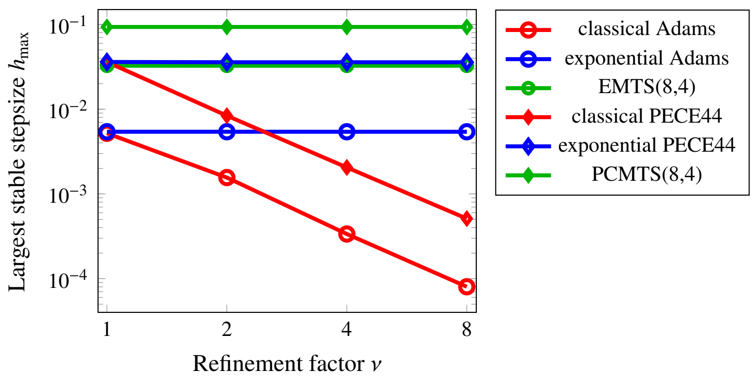


Figure 6: Largest stable stepsizes of the classical, exponential and the new MTS multistep methods as a function of the refinement factor $\nu = 1, 2, 4, 8$.

As expected, for the classical methods we observe a quadratic reduction of $h_{\max}$ with increasing refinement ν. In contrast, the exponential and the optimized MTS schemes maintain a fixed $h_{\max}$ independently of ν. However, due to the much larger stability contours, the EMTS(8,4) scheme is almost a factor 6 faster than the exponential Adams method of the same order. Furthermore, the PCMTS(8,4) scheme is approximately a factor 2.5 more efficient than its Adams-type counterparts. As a consequence, our PCMTS scheme clearly shows the best performance. For the heat equation with a moderate refinement of $\nu = 4$, our scheme allows timesteps which are about two orders of magnitude larger than those of classical schemes.

6.3. Discontinuous Galerkin Discretization of Maxwell's Equations

For a more realistic application, we next consider the numerical solution of the three-dimensional Maxwell equations via a nodal discontinuous Galerkin time-domain (DGTD) method [31, 32, 33]. In this method, Maxwell's equations are first discretized in space to obtain a large, semi-discrete system of ODEs

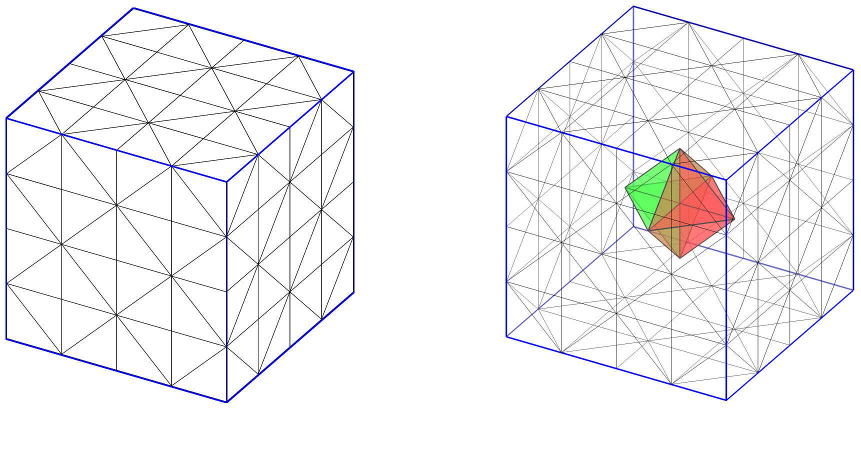
$$\frac{\mathrm{d}\mathbf{E}^k}{\mathrm{d}t} = \frac{1}{\epsilon^k}\left[\mathbf{D}^k \times \mathbf{H}^k + (\mathcal{M}^k)^{-1}\mathcal{F}^k\left[(\Delta\mathbf{E} - \hat{n}(\hat{n}\cdot\Delta\mathbf{E})) + Z^+\hat{n}\times\Delta\mathbf{H}\right]/\overline{Z}\right], \tag{35a}$$

$$\frac{\mathrm{d}\mathbf{H}^k}{\mathrm{d}t} = \frac{1}{\mu^k}\left[-\mathbf{D}^k \times \mathbf{E}^k + (\mathcal{M}^k)^{-1}\mathcal{F}^k\left[(\Delta\mathbf{H} - \hat{n}(\hat{n}\cdot\Delta\mathbf{H})) - Y^+\hat{n}\times\Delta\mathbf{E}\right]/\overline{Y}\right]. \tag{35b}$$

The discontinuous Galerkin approach allows a discretization of arbitrary spatial order p_{DG}. In the nodal formulation on a tetrahedral grid, the vectors $\mathbf{E}^k$ and $\mathbf{H}^k$ together contain a total of $N_k = (p_{\mathrm{DG}}+1)(p_{\mathrm{DG}}+2)(p_{\mathrm{DG}}+3)$ unknown electric and magnetic field values that are associated with the tetrahedral element k. Furthermore, $\Delta\mathbf{E}$ and $\Delta\mathbf{H}$ signify the field discontinuities, i.e., the jumps across each element's interface, and $\hat{n}$ denotes the normal vector of those interfaces. The definition of the matrices $\mathbf{D}^k$, $\mathcal{M}^k$ and $\mathcal{F}^k$, as well as details on the material parameters Y and Z can be found in [34].

To test the performance of our PCMTS algorithm, we consider the time-evolution of an eigenmode in an empty cubic cavity (see [35, Chapter 8.3]) of edge length $a = b = c = 2$ with perfect electric conductor (PEC) boundary conditions. The fields are initialized by the second eigenmode ($n = 2$) and each simulation runs for 25 oscillation periods ($T_{\mathrm{end}} = 25\frac{2}{\sqrt{3}} \approx 28.87$).

The mesh consists of 320 tetrahedra and is depicted in Figure 7(a). Then, in order to introduce grid-induced stiffness, we shift the center point of the mesh along the x-axis without changing the connectivity of the mesh. As a consequence, we compress four tetrahedra. Following [32, 33], we



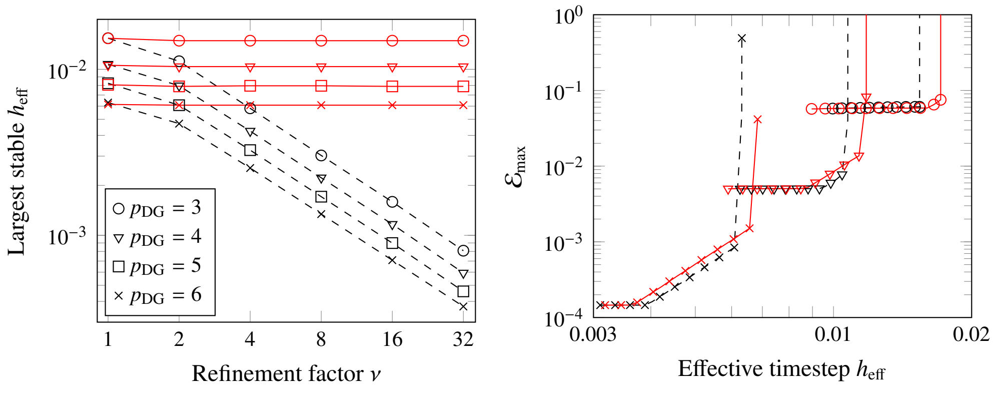
(a) The mesh (b) The 8 inner tetrahedra

Figure 7: The basic mesh used for testing different time-integration methods. To change the stiffness of the system, the center node of the mesh is shifted along the x-axis, resulting in four compressed tetrahedra.

use the radius of the smallest insphere as a geometric measure for the mesh-dependent part of the

Courant-Friedrichs-Lewy (CFL) criterion. Therefore, we expect that the largest stable timestep $h_{\max}$ scales linearly with the radius of the smallest insphere in our mesh. For the following numerical experiments, we use six different meshes, where the inradius of the four compressed tetrahedra is successively reduced by the refinement factor $\nu = 2^n$ for $n = 0 \dots 5$.

For comparison, we first run the simulations with the optimized 4th-order 14-stage low-storage Runge-Kutta integrator LSRK(14,4) proposed in [25]. The maximum stable effective step sizes h_{eff} are plotted over ν for different degrees p_{DG} as dashed lines in Figure 8(a). Indeed it shows the expected linear behavior for $\nu > 1$.



(a) Largest stable timestep

(b) Convergence

Figure 8: A comparison of the LSRK(14,4)-scheme (black, dashed lines) and the optimized PCMTS(8,4) method (red, solid lines). In (a) we depict the largest stable timestep for various spatial discretization orders as a function of the element size ratio. In (b) we plot the error $\mathcal{E}_{\max}$ as a function of the effective timestep for $\nu = 1$. The legend is the same as in (a) and the spatial order $p_{\text{DG}} = 5$ was omitted to improve readability.

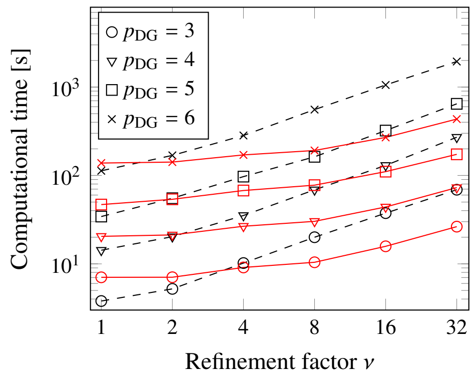
For the PCMTS method, we split our total system operator on the element level. This means, for the inner (stiff) operation **f**, we employ the right-hand side of (35) for the four small elements only. The function **g** simply corresponds to the right-hand side of (35) for the remaining 316 tetrahedra. This splitting was chosen for simplicity, since the action of the right-hand side of (35) is implemented element-wise in our code (as probably in most production codes). Thus, we can leave our implementation unchanged and simply call the existing routines with the list of elements to evaluate. For the actual time-integration we employ the PCMTS(8,4) method with coefficients (A.2), where the inner integration is performed by the LSRK(14,4) scheme.

As shown in Figure 8(a), without stiffness ($\nu = 1$) we observe almost the same largest stable h_{eff} for LSRK(14,4) and PCMTS(8,4). This shows that our multistep method has roughly the same effective stability contour as the LSRK(14,4) scheme. However, for increasing stiffness, the outer stepsize of the PCMTS(8,4) scheme remains constant.

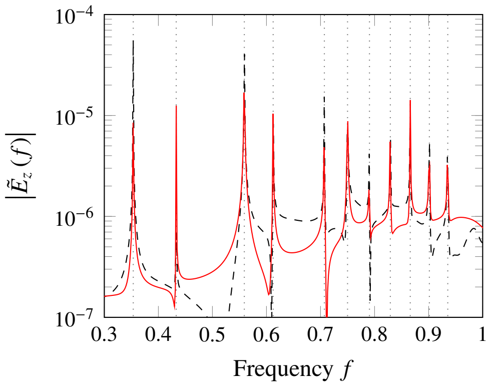
To facilitate a fair comparison, it is important to also investigate the accuracy of the LSRK(14,4) and the PCMTS(8,4) schemes. To do so, we define $\mathcal{E}_{\max}$ as the maximum deviation of the electrical field components from the analytical solution over all points and all timesteps. In Fig. 8(b), we plot $\mathcal{E}_{\max}$ as a function of the effective timestep h_{eff} for $\nu = 1$. We find that for order $p_{\text{DG}} = 3$ the error is almost entirely dominated by the spatial discretization. For $p_{\text{DG}} = 4$, we start to find a small region where we can observe the expected fourth-order convergence of both time integration schemes until the error is again dominated by the spatial discretization. Only for higher spatial orders, we observe significant error contributions from the time integration. While the error is slightly larger for the PCMTS(8,4) scheme in comparison the the LSRK(14,4), the difference can be easily compensated by a very small reduction in h.

For practical applications, it is also interesting to consider the actual CPU time required for a computation. It is important to note that such numbers are strongly dependent on the actual implementation and on the computer the code was executed on. In our case, we put significant effort into the optimization of both, the classical and the multistep formulation. As shown in Figure 9(a), the difference in computational time depends on both, the spatial order p_{DG} and the refinement factor ν. We find that for higher spatial discretization orders p_{DG}, the advantage of our MTS approach becomes more apparent. This is to be expected since the computational effort of the calculations per element grows as N_k^2, while the total time spent on the interpolation (evaluation of the polynomial) only grows linearly with N_k. Thus, the overhead of the interpolation becomes less important for higher degrees p_{DG}. However, even for lower spatial order such as $p_{\mathrm{DG}} = 3$, the PCMTS algorithm outperforms an optimized low-storage Runge-Kutta scheme once the refinement factor is above $\nu = 4$.

Finally, to demonstrate long-time stability, we ran additional calculations where we initialize all field components of the cavity with uniformly distributed random numbers from the interval $[-1, 1]$. For all combinations of ν, k and p_{DG} used above, we then perform the simulations with the previously determined largest stable timestep until $T_{\mathrm{end}} = 10^6$ in our dimensionless units. This corresponds to between 5×10^6 and 10^8 timesteps. As expected, all the simulations remain stable. In Fig. 9(b) we show the spectra from calculations with LSRK(14,4) and with PCMTS(8,4) for $p_{\mathrm{DG}} = 4$ and $\nu = 4$. As can be seen, all resonances (indicated by the dotted vertical lines) are accurately reproduced and no spurious resonances can be observed.



(a) Computational time



(b) Spectra of randomly initialized cavity

Figure 9: A comparison of the LSRK(14,4)-scheme (black, dashed lines) and the optimized PCMTS(8,4) method (red, solid lines). In (a) we plot the actual computational time for a corresponding computation. All computations were performed on a single core of an Intel Xeon(R) W3680 processor. In (b) we plot the Fourier transform of the z-component of the electrical field recorded at position $(0.123, 0.213, 0.321)$ for a cavity initialized with random values. Both calculations were performed for $p_{\mathrm{DG}} = 4$ and $\nu = 4$. The dotted vertical lines indicate the analytical resonance frequencies of the cavity.

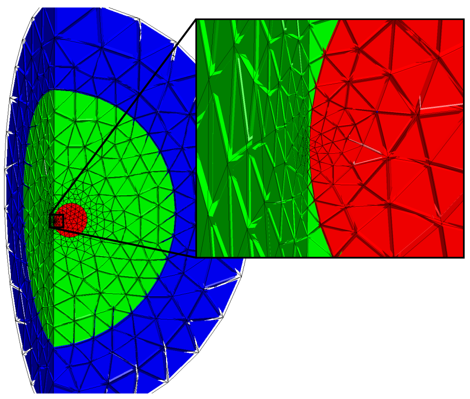
6.4. A realistic, inhomogeneous problem

Finally, we consider a realistic calculation from the field of plasmonics. We again solve the linear Maxwell equations, discretized by a nodal discontinuous Galerkin approach based on an upwind flux as in the previous section. However, in this calculation we study a dimer of two gold nano-spheres of radius $r = 80\text{nm}$ with a very narrow gap of $g = 1\text{nm}$ between them. The dispersion of gold is included in the simulation via a Drude-Lorentz model

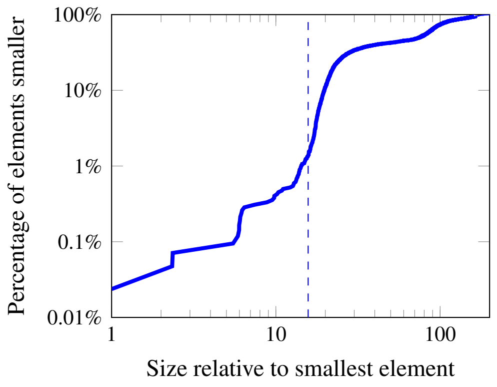
$$\epsilon(\omega) = \epsilon_\infty - \frac{\omega_D^2}{\omega(\omega - i\gamma_D)} + \frac{\Delta\epsilon\omega_L^2}{\omega^2 - \omega_L^2 + i\omega\gamma_L}$$

with parameters $\epsilon_\infty = 6.21$, $\omega_D = 8.79\text{eV}$, $\gamma_D = 0.069\text{eV}$, $\Delta\epsilon = 1.0$, $\omega_L = 2.65\text{eV}$ and $\gamma_L = 0.38\text{eV}$. Our system is terminated by basic first-order absorbing boundary conditions as described in [33]. The dimer is excited by a plane wave with the electric field polarized along the dimer axis. The wave is injected via a total-field/scattered-field (TF/SF) contour and its temporal profile is given by a broad-band Gaussian pulse with a wavelength spectrum ranging from 400nm to 1200nm.

The intrinsic symmetries of this system allow to introduce two mirror planes to reduce the size of the computational domain by a factor four [36]. The actual mesh used for the computations is shown in Figure 10(a). The same structure was used in [36] for the comparison of several frequency-domain Maxwell solvers and can be considered as a challenging but typical plasmonic system.



(a)



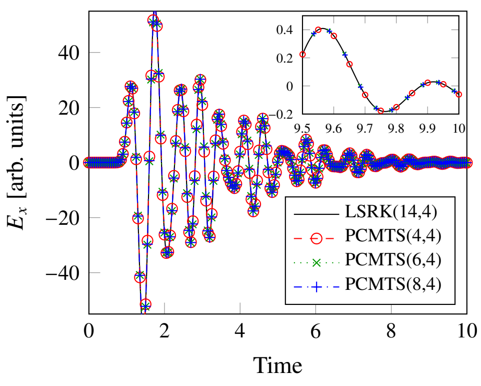
(b)

Figure 10: (a) The mesh used for the plasmonic dimer. Red elements indicate the metallic particle, green elements belong to the total-field region and blue elements indicate the scattered-field region. (b) Cumulative distribution of the element size relative to the smallest element. The y-axis shows the number of elements in percent of the total number. The dashed vertical line indicates the splitting between the small and the large elements.

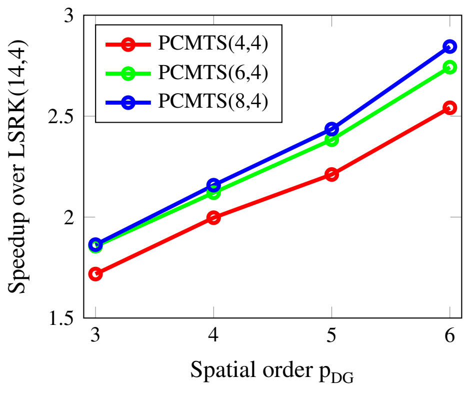
The critical issue with this structure is the very narrow gap between the sphere and the mirror wall. Here, the mesh generator has to introduce a few, very small elements. This leads to strong grid-induced stiffness which leads to very small timesteps for explicit time-domain solvers. In Figure 10(b) we plot the cumulative size distribution of the insphere radii of all tetrahedra in our mesh. In this double-logarithmic plot we observe a steep increase of the element count around a factor of 15. Thus, we decide to split the system at this point. To be precise, we consider the smallest 58 elements as stiff and treat the remaining 4164 tetrahedra as non-stiff. Our splitting is indicated by the dashed vertical line in Figure 10(b). It should be noted that the exact splitting is somewhat arbitrary and it is certainly possible to improve the computational time by using more advanced methods to find the optimal splitting point.

As in the previous section, we perform computations for different spatial orders p_{DG} and with different time-stepping schemes. Specifically, we use PCMTS(k,4) schemes with $k \in \{4, 6, 8\}$ and compare them to the LSRK(14,4) method. We again employ bisection to find the largest stable effective timestep for each scheme.

To assess the accuracy of the PCMTS schemes we compare the temporal evolution of the x-component of the electric field at the center of the dimer. The results for $p_{\mathrm{DG}} = 6$ are depicted in Fig. 11(a). We find that the maximum absolute deviation of all PCMTS(k,4) simulations to the LSRK(14,4) [25] data is below 10^{-6} and we therefore consider the error introduced by the time integration to be negligible. This is not surprising since the timestep for the outer integration is still small compared to the fastest oscillation period (shortest wavelength) excited in our simulation. In Figure 11(b), we then depict the actual speedup of our PCMTS(k,4) schemes with $k \in \{4, 6, 8\}$ over the same calculation with the explicit LSRK(14,4) scheme.



(a) Time evolution



(b) Computational speedup

Figure 11: (a) The evolution of the x-component of the electric field in the center between two metallic spheres. Spatial order was $p_{\mathrm{DG}} = 6$ and the inset shows a zoom for late times (b) The speedup in computational time for the simulation of a plasmonic dimer with a narrow gap.

In total, we find a speedup factor between two and three over the LSRK(14,4) integrator. As in the previous section, the gain of the MTS methods increases with the order of the spatial discretization p_{DG}. Unfortunately, we do not observe the full speedup of up to 15 in this example because of the computational time required for the evaluation of the polynomials (6) and (9). A profiling of our code reveals that between 70% and 90% of the computational time in the inner integrator are spent in the matrix-vector products used to evaluate the polynomials. Since this operation is strongly memory bound, it is not negligible in comparison to the DG kernel used to evaluate $\mathbf{g}$. As discussed in the previous section, this effect is reduced when using a more complex $\mathbf{g}$. Still, a speedup of a factor two to three over a highly optimized simulator and for a realistic simulation represents a significant gain.

7. Summary and Outlook

In summary, we have presented a flexible approach to construct efficient multiple time-stepping algorithms. More specifically, we have demonstrated how a multistep MTS scheme can be constructed from two ingredients: 1) Any choice of explicit Adams type or predictor-corrector scheme for the outer (non-stiff) integration and 2) any choice of ODE solver for the inner (stiff) integration. Exploiting this decoupling allowed us to construct MTS schemes with optimized stability contours for both the inner and outer integration. In contrast to many other higher-order LTS schemes, our methods are easily implemented and do not require any modification of existing codes besides the splitting into a stiff and a non-stiff function. As an example, we constructed a number of EMTS and PCMTS schemes which allow significantly larger macro-timesteps when compared to previously known explicit multistep MTS schemes.

In a series of numerical experiments, we demonstrated that our methods are stable and maintain their order for inhomogeneous and nonlinear problems. Furthermore, we showed significant reductions in computational time for realistic numerical simulations.

As an outlook, we believe that our approach can be further generalized by using more general extrapolation schemes. In addition, our methods can also be extended to include an error estimator which would allow for adaptive timestepping. Furthermore, when considering wave propagation problems with grid-induced stiffness, it might be sufficient to interpolate only a subset of the unknowns around the small elements. This would reduce the memory requirements and further increase the performance for those kinds of simulations. However, one then has to take care that the coupling between both parts is stable (which is trivial for the methods considered in this paper). Finally, from a theoretical point of view, a rigorous stability analysis of the fully nonlinear system is certainly a worthwhile topic for future research.

Appendix A. Coefficients for the Optimized Schemes

PCMTS(6,3) with circular stability region:

$$\mathcal{B} = \begin{bmatrix} -0.027438448850 & 0 \\ 0.004205433197 & 0 \\ -0.005757000197 & 0 \\ -0.074759827110 & 0 \\ 0.287161166273 & -1 \\ 0.816588676687 & 1 \end{bmatrix}, \quad \tilde{\mathcal{B}} = \begin{bmatrix} 0.0246201522600 & 0 & 0 \\ -0.0005352246566 & 0 & 0 \\ -0.0546888084000 & 0 & 0 \\ -0.0789237494604 & -1/2 & 1 \\ 1.2009540614472 & 0 & -2 \\ -0.0914264311902 & 1/2 & 1 \end{bmatrix}. \tag{A.1}$$

PCMTS(8,4) with circular stability region:

$$\mathcal{B} = \begin{bmatrix} 0.048992366370 & 0 & 0 \\ -0.011407158170 & 0 & 0 \\ -0.027817310550 & 0 & 0 \\ 0.006136109166 & 0 & 0 \\ -0.023738957620 & 0 & 0 \\ -0.545158997856 & 1/2 & 1 \\ 1.001573369088 & -2 & -2 \\ 0.551420579572 & 3/2 & 1 \end{bmatrix}, \quad \tilde{\mathcal{B}} = \begin{bmatrix} -0.02689484047 & 0 & 0 & 0 \\ 0.02714562621 & 0 & 0 & 0 \\ 0.04728737387 & 0 & 0 & 0 \\ 0.01190410100 & 0 & 0 & 0 \\ -0.12208325045 & 1/6 & 0 & -1 \\ -0.02044133663 & -1 & 1 & 3 \\ 1.14846927729 & 1/2 & -2 & -3 \\ -0.06538695082 & 1/3 & 1 & 1 \end{bmatrix}. \tag{A.2}$$

EMTS(8,4) with rectangular stability region:

$$\mathcal{B} = \begin{bmatrix} -0.092436748185 & 0 & 0 & 0 \\ -0.034882222033 & 0 & 0 & 0 \\ 0.271029601208 & 0 & 0 & 0 \\ 0.284302074046 & 0 & 0 & 0 \\ 0.085426318875 & -1/3 & -1 & -1 \\ -2.207982370599 & 3/2 & 4 & 3 \\ 2.523680051842 & -3 & -5 & -3 \\ 0.170863294846 & 11/6 & 2 & 1 \end{bmatrix}. \tag{A.3}$$

PCMTS(8,4) with rectangular stability region:

$$\mathcal{B} = \begin{bmatrix} 0.119290989092 & 0 & 0 \\ -0.070763889414 & 0 & 0 \\ 0.000508218466 & 0 & 0 \\ -0.082227604557 & 0 & 0 \\ -0.164764495336 & 0 & 0 \\ -0.461075501035 & 1/2 & 1 \\ 1.332360226815 & -2 & -2 \\ 0.326672055969 & 3/2 & 1 \end{bmatrix}, \quad \tilde{\mathcal{B}} = \begin{bmatrix} -0.12885251374 & 0 & 0 & 0 \\ 0.15957818116 & 0 & 0 & 0 \\ 0.22581846012 & 0 & 0 & 0 \\ -0.13209979425 & 0 & 0 & 0 \\ -0.41151106644 & 1/6 & 0 & -1 \\ 0.08117743480 & -1 & 1 & 3 \\ 1.41598371535 & 1/2 & -2 & -3 \\ -0.21009441700 & 1/3 & 1 & 1 \end{bmatrix}. \tag{A.4}$$

References

[1] E. Hairer, S. P. Nørsett, G. Wanner, Solving Ordinary Differential Equations I. Nonstiff Problems, 2nd Edition, Vol. 8 of Springer Series in Computational Mathematics, Springer, Berlin, Heidelberg, 1993.

[2] J. C. Butcher, Numerical methods for ordinary differential equations, John Wiley & Sons, New York, 2003.

[3] W. Hundsdorfer, J. Verwer, Numerical Solution of Time-Dependent Advection-Diffusion-Reaction Equations, Springer Series in Computational Mathematics, Springer, 2007.

[4] A. R. Leach, Molecular Modelling: Principles and Applications, Pearson Education, Prentice Hall, 2001.

[5] M. Günther, U. Feldmann, J. ter Maten, Modelling and discretization of circuit problems, in: W. Schilders, E. ter Maten (Eds.), Numerical Methods in Electromagnetics, Vol. 13 of Handbook of Numerical Analysis, Elsevier, 2005, pp. 523 – 659.

[6] E. Hairer, G. Wanner, Solving Ordinary Differential Equations II: Stiff and Differential-Algebraic Problems, Springer Series in Computational Mathematics, Springer, 2004.

[7] J. R. Rice, Split Runge-Kutta method for simultaneous equations, J. Res. Natl. Bur. Standards 64B (3) (1960) 151–170.

[8] C. W. Gear, Multirate methods for ordinary differential equations, Tech. Rep. UIUCDCS-F-74-880, Department of Computer Sciences, Universityof Illinois (1974). doi:10.2172/4254117.

[9] U. M. Ascher, S. J. Ruuth, B. T. R. Wetton, Implicit-explicit methods for time-dependent partial differential equations, SIAM J. Numer. Anal. 32 (3) (1995) 797–823.

[10] U. M. Ascher, S. J. Ruuth, R. J. Spiteri, Implicit-explicit Runge-Kutta methods for time-dependent partial differential equations, Appl. Numer. Math 25 (1997) 151–167.

[11] M. J. Gander, L. Halpern, Techniques for locally adaptive timestepping developed over the last two decades, in: Domain decomposition methods in science and engineering XX, Vol. 91 of Lect. Notes Comput. Sci. Eng., Springer, Berlin, 2013, pp. 377–385.

[12] S. Piperno, Symplectic local time-stepping in non-dissipative DGTD methods applied to wave propagation problems, Research Report RR-5643, INRIA (2005).

[13] M. A. Botchev, J. G. Verwer, Numerical integration of damped Maxwell equations, SIAM J. Sci. Comput. 31 (1) (2009) 1322–1346.

[14] V. Dolean, H. Fahs, L. Fezoui, S. Lanteri, Locally implicit discontinuous Galerkin method for time domain electromagnetics, J. Comput. Phys. 229 (2) (2010) 512–526.

[15] S. Descombes, S. Lanteri, L. Moya, Locally implicit time integration strategies in a discontinuous Galerkin method for maxwells equations, J. Sci. Comput. 56 (1) (2013) 190–218.

[16] M. J. Grote, T. Mitkova, High-order explicit local time-stepping methods for damped wave equations, J. Comput. Appl. Math. 239 (1) (2013) 270 – 289.

[17] L. Angulo, J. Alvarez, F. Teixeira, M. Pantoja, S. Garcia, Causal-path local time-stepping in the discontinuous galerkin method for maxwells equations, J. Comput. Phys. 256 (2014) 678 – 695.

[18] M. J. Grote, M. Mehlin, T. Mitkova, Runge-Kutta based explicit local time-stepping methods for wave propagation, Preprint, University of Basel (2014) 28 pages.

[19] M. Hochbruck, A. Ostermann, Exponential multistep methods of Adams-type, BIT 51 (2011) 889–908.

[20] J. Certaine, The solution of ordinary differential equations with large time constants, in: Mathematical Methods for Digital Computers, Wiley, New York, 1960, pp. 128–132.

[21] S. P. Nørsett, An *A*-stable modification of the Adams-Bashforth methods, in: Conf. on Numerical Solution of Differential Equations (Dundee, 1969), Springer, Berlin, 1969, pp. 214–219.

[22] M. Hochbruck, A. Ostermann, Exponential integrators, Acta Numerica 19 (2010) 209–286.

[23] A. Ostermann, M. Thalhammer, W. M. Wright, A class of explicit exponential general linear methods, BIT 46 (2) (2006) 409–431.

[24] M. Liu, K. Sirenko, H. Bagci, An efficient discontinuous galerkin finite element method for highly accurate solution of maxwell equations, Antennas and Propagation, IEEE Transactions on 60 (8) (2012) 3992–3998.

[25] J. Niegemann, R. Diehl, K. Busch, Efficient low-storage Runge-Kutta schemes with optimized stability regions, J. Comput. Phys. 231 (2012) 364–372.

[26] P. Kaelo, M. M. Ali, Some variants of the controlled random search algorithm for global optimization, J. Optim. Theory Appl. 130 (2) (2006) 253–264.

[27] S. G. Johnson, The NLopt nonlinear-optimization package.
URL `http://ab-initio.mit.edu/nlopt`

[28] R. Diehl, K. Busch, J. Niegemann, Comparison of low-storage Runge-Kutta schemes for discontinuous Galerkin time-domain simulations of Maxwell's equations, J. Comput. Theor. Nanosci. 7 (2010) 1572–1580.

[29] M. P. Calvo, C. Palencia, A class of explicit multistep exponential integrators for semilinear problems, Numer. Math. 102 (3) (2006) 367–381.

[30] D. Stanescu, W. G. Habashi, 2N-storage low dissipation and dispersion Runge-Kutta schemes for computational acoustics, J. Comput. Phys. 143 (2) (1998) 674–681.

[31] J. Hesthaven, T. Warburton, Nodal high-order methods on unstructured grids - I. Time-domain solution of Maxwell's equations, J. Comput. Phys. 181 (1) (2002) 186–221.

[32] J. Hesthaven, T. Warburton, Nodal Discontinuous Galerkin Methods, Springer, 2007.

[33] K. Busch, M. König, J. Niegemann, Discontinuous Galerkin methods in nanophotonics, Laser Photon. Rev. 5 (6) (2011) 773–809.

[34] J. Niegemann, M. König, K. Stannigel, K. Busch, Higher-order time-domain methods for the analysis of nano-photonic systems, Photon. Nanostruct. Fundam. Appl. 7 (1) (2009) 2.

[35] C. A. Balanis, Advanced Engineering Electromagnetics, John Wiley & Sons, 1989.

[36] J. Hoffmann, C. Hafner, P. Leidenberger, J. Hesselbarth, S. Burger, Comparison of electromagnetic field solvers for the 3d analysis of plasmonic nanoantennas, in: Proc. of SPIE, Vol. 7390, 2009, pp. 73900J–1.