# Two Families of W-Methods: Analysis and Application on Battery Models

Dajana Conte[1(✉)] and Giovanni Pagano[2]

[1] Department of Mathematics, University of Salerno, 84084 Fisciano, Italy
dajconte@unisa.it
[2] Department of Agricultural Sciences, University of Naples Federico II, 80055 Portici, Italy
giovanni.pagano5@unina.it

**Abstract.** This paper focuses on the efficient numerical solution of stiff initial value problems arising from the spatial discretization of Partial Differential Equations (PDEs). In particular, this work shows an efficient implementation of two families of linearly implicit numerical methods recently introduced in the scientific literature: the TASE-W methods and the singly TASE-Runge-Kutta methods. These methods are derived exploiting the so-called TASE (Time-Accurate and highly Stable Explicit) operators, and are particular cases of W-methods. We deeply analyze the properties of consistency, stability, and computational cost of TASE-W and singly TASE-Runge-Kutta methods, employing them for the solution of a system of two coupled PDEs for the description of the charge/discharge processes in electric batteries.

**Keywords:** W-methods · TASE-W methods · singly TASE-Runge-Kutta methods · stiff problems · PDEs · battery modeling

## 1 Introduction

This work focuses on the numerical solution of large stiff initial value problems (IVPs)

$$y'(t) = f(t, y), \quad y_0 = y(t_0), \qquad t \in [t_0, t_{\text{end}}], \quad y, f \in \mathbb{R}^m. \tag{1}$$

Stiff IVPs are characterized by the presence of rapidly changing components, and often require very small discretization step-sizes unless a stable numerical method is employed for their solution. Stiff systems often arise in fields like chemical kinetics or control theory, where some components evolve much faster than others. Stiff IVPs can also arise from the spatial semi-discretization of Partial Differential Equations (PDEs), that are widely used in engineering and physics to describe phenomena characterized by several independent variables, such as temperature distribution in a solid, evolution of different plant species in arid or semi-arid environments, corrosion of materials [30], charge/discharge processes in electric batteries involving several chemical substances [31]. Models of recent interest are also constituted by stochastic differential equations [16],

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2026
O. Gervasi et al. (Eds.): ICCSA 2025 Workshops, LNCS 15887, pp. 133–149, 2026.
https://doi.org/10.1007/978-3-031-97589-9_10

which introduce randomness into a system of ordinary differential equations, representing phenomena influenced by unpredictable noise, often used in finance, biology and physics to describe uncertain dynamics. Among the stiff problems of great interest in the deterministic and stochastic fields we mention, for example, Hamiltonian systems, for which it is known that the design of numerical methods capable of ensuring energy conservation on long time scales is very important [3,17,18,22]. Otherwise, small errors in numerical integration could accumulate, leading to unrealistic results. Currently, also stochastic partial differential equations, that combine the complexities of both randomness and spatial variability, are highly regarded in the field of numerical modeling [19]. For stiff deterministic problems several numerical methods have been proposed that attempt to combine low error coefficients and computational effort with good stability properties, in order to get an efficient solution [5,10,13,23,24,29]. Furthermore, an effort has been made in the construction of methods adapted to the problem, capable of unconditionally preserving the most important properties of the analytical solution in the discrete setting [11,25–28]. The extension of some of these methods to the stochastic case has also been considered [6,12,20,21].

Among the numerical methods for stiff IVPs recently introduced in the scientific literature, the so-called TASE-RK (Time-Accurate and highly Stable Explicit Runge-Kutta) schemes [4,9], which constitute a subclass of the linearly implicit W-methods [33], have garnered significant attention, see [1,2,7,8,14,15,32]. TASE-RK methods are indeed very simple to implement, as they require a slight modification of an explicit RK scheme through the so-called TASE operators. Also, TASE-RK methods have good stability properties and low computational cost. This makes them quite attractive for large stiff IVPs arising e.g. from the semi-discretization in space of PDEs. The main computational cost of the original $s$-stage TASE-RK methods [4,9] is given by the solution of $s^2$ linear systems at each time-step, to reach a given order of consistency $p = s = 2, 3, 4$. Such linear systems involve $s$ different coefficient matrices, which are of the type $(I - h\theta J)$, being: $I$ the identity of size $m$, $h$ the time step-size used by the method to provide the solution $y_n \approx y(t_n)$ at the points of the discrete grid $\{t_n = t_0 + nh,\ n = 0, \ldots, N,\ t_N = t_{\text{end}}\}$, $\theta$ a positive parameter, $J$ the Jacobian, i.e. $J = J(t, y) := \frac{\partial f}{\partial y}(t, y)$. To avoid trivial cases, we assume from now on that the Jacobian $J$ is not constant.

To improve the computational cost of the original TASE-RK methods [4,9], new operators and numerical schemes have been proposed that attempt to reduce the number of overall linear systems to be solved, and/or the number of coefficient matrices involved. In particular, by exploiting the connections between the TASE-RK methods and the W-methods, the family of TASE-W methods [32] has been proposed, which allows to reduce the overall number of linear systems to be solved to reach order $2, 3, 4$, involving only two coefficient matrices. A further improvement from this last point of view has been given by the so-called singly TASE operators [7], which used in the context of explicit RK methods allow to consider linear systems at any time-step where a single coefficient matrix is involved. In particular, the resulting singly TASE-RK methods [7] require the solution of $s^2$ linear systems at each step where the coefficient matrix is always

the same. Unlike the original TASE-RK methods [4,9], which have a relatively simple structure, the TASE-W [32] and singly TASE-RK methods [7] involve a rather intricate formulation. In this work, therefore, we show in detail the structure of the various stages of these methods, and highlight the steps required for an efficient computation of the time-marching solution.

The structure of this paper is as follows. In Sect. 2 we recall the family of TASE-W methods, propose a strategy for an efficient implementation, and analyze the properties of accuracy, stability, and computational cost of numerical schemes of order 2, 3, and 4. In Sect. 3 we do the same for the singly TASE-RK methods. In Sect. 4 we use the two considered families of methods for the solution of a PDEs model for the description of the electrodeposition process in batteries. Section 5 concludes the manuscript with comments on the results obtained and ideas for future research.

## 2 The Family of TASE-W Methods

TASE-W methods [32] were introduced taking inspiration from the connections between TASE-RK methods and W-methods, with the aim of reaching a given order $p$, up to $p = 4$, solving the smallest possible number of linear systems at each step, while still obtaining good stability properties. For reminders of stability definitions, see e.g. [32, p. 131]. The general formulation of TASE-W methods is as follows:

$$\begin{cases} (I - \theta_q h W)\, K_i^{[q]} = hf\left(t_n + c_i h, y_n + \displaystyle\sum_{j=1}^{i-1} a_{ij} K_j\right) + \displaystyle\sum_{j=1}^{i-1} \ell_{ij} K_j, \quad q = 1, \dots, r, \\ K_i = \displaystyle\sum_{q=1}^{r} \omega_q K_i^{[q]}, \quad i = 1, \dots, s, \\ y_{n+1} = y_n + \displaystyle\sum_{i=1}^{s} b_i K_i. \end{cases} \tag{2}$$

The matrix $W$ can be arbitrary (also constant throughout the numerical integration) without affecting the order of consistency of the methods. That is, the order conditions of these methods have been studied for arbitrary $W$. However, for stability reasons $W$ should correspond to the Jacobian of the problem at the current step, i.e. $W = J_n$ with $J_n = J(t_n, y_n)$, or at least to the Jacobian of the stiff part of the vector field. Indeed, the good stability properties of these methods have been proved under the assumption $W = J_n$.

*Remark 1.* Actually, solving non-autonomous problems (1) requires a slight modification of the formulation (2) of TASE-W methods, which consists in adding the term $\tilde{v} h^2 k_i^{[q]}$ in the right-hand side of the linear system that leads to determining $K_i^{[q]}$, being $k_i^{[q]}$ a given coefficient and $\tilde{v} \approx v(t_n, y_n)$, with $v(t, y) = \frac{\partial f}{\partial t}(t, y)$, see [32, Subsec. 3.2]. However, the order of consistency of the methods on non-autonomous problems is independent of the choice of $\tilde{v}$. Thus,

$\tilde{v}$ is typically considered equal to $\mathbf{0}$ [32, Subsec. 3.2], being $\mathbf{0}$ the vector of zeros of length $m$. In this way we recover formulation (2) also for non-autonomous problems.

The coefficients of the TASE-W methods (2) can be organized defining the following matrices and vectors:

$$\begin{aligned} &A = (a_{ij})_{i,j=1}^{s} \in \mathbb{R}^{s\times s} \text{ strictly lower triangular,} \\ &L = (\ell_{ij})_{i,j=1}^{s} \in \mathbb{R}^{s\times s} \text{ strictly lower triangular,} \\ &b = (b_i)_{i=1}^{s} \in \mathbb{R}^{s}, \quad c = (c_i)_{i=1}^{s} \in \mathbb{R}^{s}, \quad \theta = (\theta_i)_{i=1}^{r} \in \mathbb{R}^{r}, \quad \omega = (\omega_i)_{i=1}^{r} \in \mathbb{R}^{r}. \end{aligned} \tag{3}$$

From now on, we consider $c_1 = 0$. Also, we take $r = 2$ according to the following result, which can be directly deduced from [32, Th. 1 and Rem. 4].

**Proposition 1.** *TASE-W methods reach order of consistency $p = s = 2, 3, 4$ with $r = 2$. TASE-W methods cannot reach order $p = s = 3, 4$ with $r = 1$.*

As can be seen from (2), the more the value of $r$ increases, the more the number of linear systems to be solved increases. The choice $r = 2$ is therefore preferable, since it allows to work with small computational effort to reach order $p = 3, 4$, and also as we will show it allows to obtain L-stable methods.

For clarity and to highlight the steps required by the TASE-W methods (2), below we express them in full with $r = 2$.

$$\begin{aligned} &f_1 = f\left(t_n, y_n\right), \\ &\left(I - h\theta_1 W\right) K_1^{[1]} = h f_1, \quad \left(I - h\theta_2 W\right) K_1^{[2]} = h f_1, \\ &K_1 = \omega_1 K_1^{[1]} + \omega_2 K_1^{[2]}, \end{aligned}$$

$$\begin{aligned} &f_2 = f\left(t_n + c_2 h, y_n + a_{21} K_1\right), \\ &\left(I - h\theta_1 W\right) K_2^{[1]} = h f_2 + \ell_{21} K_1, \quad \left(I - h\theta_2 W\right) K_2^{[2]} = h f_2 + \ell_{21} K_1, \\ &K_2 = \omega_1 K_2^{[1]} + \omega_2 K_2^{[2]}, \end{aligned}$$

$$\vdots$$

$$\begin{aligned} &f_s = f\left(t_n + c_s h, y_n + \sum_{j=1}^{s-1} a_{sj} K_j\right), \\ &\left(I - h\theta_1 W\right) K_2^{[1]} = h f_s + \sum_{j=1}^{s-1} \ell_{sj} K_j, \quad \left(I - h\theta_2 W\right) K_2^{[2]} = h f_s + \sum_{j=1}^{s-1} \ell_{sj} K_j, \\ &K_s = \omega_1 K_s^{[1]} + \omega_2 K_s^{[2]}, \end{aligned}$$

$$y_{n+1} = y_n + \sum_{j=1}^{s} b_j K_j.$$

From here it is clear that the main computational effort of TASE-W methods is given by the solution of $2s$ linear systems at each step, involving 2 different coefficient matrices $(I - h\theta_1 W)$ and $(I - h\theta_2 W)$.

The next subsections discuss in detail an efficient implementation strategy of such methods, and show examples of L-stable TASE-W methods of order $p = 2, 3, 4$, with low error coefficients.

### 2.1 Implementation Steps

At a given grid point $t_n$, to compute the numerical solution at $t_{n+1}$, we perform the following steps.

1. Possible evaluation of the Jacobian of the problem at the current point, if the user chooses to select $W = J_n$.
2. Computation of the LU factorization of all the coefficient matrices involved in the linear systems to be solved: using the Matlab notation for employing the related `lu` built-in function, we thus write

$$[L_{\theta_1}, U_{\theta_1}] = \texttt{lu}\,(I - h\theta_1 W)\,, \quad [L_{\theta_2}, U_{\theta_2}] = \texttt{lu}\,(I - h\theta_2 W)\,.$$

   If the above coefficient matrices do not admit pure LU factorization, we compute a PLU one. As discussed below, this does not alter the implementation structure and main computational cost of the method.
3. Computation of the stages $K_i$, $i = 1, \ldots, s$: using the Matlab notation for employing the related backslash command, we derive $K_i$ as shown in the following Algorithm 1, solving 4 triangular linear systems.

**Algorithm 1.** Computation of $K_i$ in TASE-W methods.

1: evaluate $f_i = f\left(t_n + c_i h, y_n + \sum_{j=1}^{i-1} a_{ij} K_j\right)$

2: compute

$$K_i = \omega_1 \cdot \left( U_{\theta_1} \backslash \left( L_{\theta_1} \backslash (h f_i + \sum_{j=1}^{i-1} \ell_{ij} K_j) \right) \right) + \omega_2 \cdot \left( U_{\theta_2} \backslash \left( L_{\theta_2} \backslash (h f_i + \sum_{j=1}^{i-1} \ell_{ij} K_j) \right) \right)$$

4. Computation of the time-marching solution:

$$y_{n+1} = y_n + b_1 K_1 + \ldots + b_s K_s.$$

This implementation involves at each $t_n$: 2 LU factorizations and $4s$ triangular linear systems, see the above steps 2 and 3. We underline that when $(I - h\theta_i W)$ does not admit pure LU factorization (for some $i$ or $W$), but only PLU, the above algorithm can still be used, provided that the coefficient vector

of the linear system to be solved is premultiplied by the permutation matrix. In this way, the computational cost of the proposed implementation is almost unchanged, since the linear systems to be solved remain of triangular form.

Finally, we observe that if the user chooses $W$ constant for the whole numerical integration, steps 1 and 2 need to be performed only once and not for each $n$: in this case the method would require 2 LU factorizations in total, and at each $t_n$ the solution of $4s$ triangular systems.

### 2.2 Examples of 2nd, 3rd and 4th Order TASE-W Methods

We now list efficient methods reporting the coefficients as in (3) of the TASE-W family of order 2, 3 and 4, from [32, p. 140]. For all the methods considered below, $\omega$ is as follows:

$$\omega = \left( -\frac{\theta_2}{\theta_1 - \theta_2}, \frac{\theta_1}{\theta_1 - \theta_2} \right)^T.$$

*2nd Order Method: TASEW2*
The coefficients of the method are as

$$b = (1,1)^T,\ c = \left(0, \frac{1}{2}\right)^T,\ \theta = \left(\frac{1}{2}, \frac{\sqrt{2}}{2}\right)^T,\ A = \begin{pmatrix} 0 & 0 \\ \frac{1}{2} & 0 \end{pmatrix},\ L = \begin{pmatrix} 0 & 0 \\ -1 & 0 \end{pmatrix}.$$

*3rd Order Method: TASEW3*
The coefficients of the method are as

$$b = \left(b_1, b_2, \frac{4}{9}\right)^T,\ c = \left(0, \frac{1}{2}, \frac{3}{4}\right)^T,\ \theta = \left(\frac{1}{2}, \frac{7}{10}\right)^T,$$
$$A = \begin{pmatrix} 0 & 0 & 0 \\ \frac{1}{2} & 0 & 0 \\ a_{31} & \frac{3}{4} & 0 \end{pmatrix},\ L = \begin{pmatrix} 0 & 0 & 0 \\ \ell_{21} & 0 & 0 \\ \ell_{31} & \ell_{32} & 0 \end{pmatrix},$$

with

$$b_1 = 1.6351578675654365735, \quad b_2 = 0.92024060176023148028,$$
$$a_{31} = 1.2776842708687388139, \quad \ell_{21} = -1.7035790278249849372,$$
$$\ell_{31} = -1.9014209311534939761, \quad \ell_{32} = -1.3205413539605210804.$$

*4th Order Method: TASEW4*
The coefficients of the method are as

$$b = \left(b_1, b_2, b_3, \frac{1}{6}\right)^T,\ c = \left(0, \frac{1}{2}, \frac{1}{2}, 1\right)^T,\ \theta = \left(\frac{1}{2}, \frac{7}{10}\right)^T,$$
$$A = \begin{pmatrix} 0 & 0 & 0 & 0 \\ \frac{1}{2} & 0 & 0 & 0 \\ a_{31} & \frac{1}{2} & 0 & 0 \\ a_{41} & a_{42} & 1 & 0 \end{pmatrix},\ L = \begin{pmatrix} 0 & 0 & 0 & 0 \\ \ell_{21} & 0 & 0 & 0 \\ \ell_{31} & \ell_{32} & 0 & 0 \\ \ell_{41} & \ell_{42} & \ell_{43} & 0 \end{pmatrix},$$

with

$$b_1 = 2.2125690141456160198, \quad b_2 = 1.0480568967728132179,$$
$$b_3 = 0.86264453466476542776, \quad a_{31} = 1.8930871174773045151,$$
$$a_{41} = 0.49836947454490920784, \quad a_{42} = 0.46104213203322702563,$$
$$\ell_{21} = -3.7861742349546090303, \quad \ell_{31} = -0.49836947454490920784,$$
$$\ell_{32} = -0.46104213203322702563, \quad \ell_{41} = -3.7063266658746600868,$$
$$\ell_{42} = -3.3662571165704262555, \quad \ell_{43} = -3.1758672079885927886.$$

*Properties of the Considered Methods*

Table 1 reports the properties of the above-listed TASE-W methods, in terms of stability, computational cost and error terms.

**Table 1.** Properties of the considered TASE-W methods: stability, number of triangular linear systems (N. tr. sys.) and LU factorizations (N. LU) required at each time-step, and maximum error coefficient (Err. coeff.) in absolute value. The symbol * refers to the choice $W = J_n$.

| Method | Stability* | N. tr. sys. | N. LU* | Err. coeff. |
|---|---|---|---|---|
| TASEW2 | L-stability | 8 | 2 | 0.1667 |
| TASEW3 | L-stability | 12 | 2 | 0.07229 |
| TASEW4 | L-stability | 16 | 2 | 0.05210 |

The properties of stability and the number of LU factorizations required at each time-step refer to the choice $W = J_n$. Indeed, as already highlighted, if $W$ is selected as a constant matrix different from $J_n$, the number of factorizations is reduced to 2 in total, and not at each time-step. However, the stability study of TASE-W methods has been carried out under the assumption $W = J_n$. If the reader is interested in techniques for the study of stability with inexact Jacobian, in the work [15] the authors analyzed it for a particular family of TASE-RK methods.

## 3 Runge-Kutta Methods with Singly TASE Operators

The original TASE-RK methods [4,9] require the solution of $s^2$ linear systems at each time-step to reach order $p = s = 2, 3, 4$, involving $s$ different coefficient matrices. In [7], the authors proposed the so-called singly TASE operators, which allow considering only one coefficient matrix at each time-step, thus also improving the TASE-W methods on this aspect. From [7, pp. 5–6], the resulting singly TASE-RK methods have the general formulation

$$\begin{cases} K_i = \left( \sum_{q=1}^{s} \omega_q \left( I - h\theta W \right)^{-q} \right) hf \left( t_n + c_i h, y_n + \sum_{j=1}^{i-1} a_{ij} K_j \right), \quad i = 1, \ldots, s, \\ y_{n+1} = y_n + \sum_{i=1}^{s} b_i K_i. \end{cases} \tag{4}$$

As for TASE-W methods, also for singly TASE-RK methods the matrix $W$ can be arbitrary without affecting the order of consistency, but for stability reasons it is advisable that it corresponds to $J_n$, or at least to the Jacobian of the stiff part of the vector field.

Similarly to TASE-W methods (2), we organize the coefficients of singly TASE-RK methods (4) defining the following matrices and vectors:

$$\begin{aligned} &A = (a_{ij})_{i,j=1}^{s} \in \mathbb{R}^{s\times s} \text{ strictly lower triangular,} \\ &b = (b_i)_{i=1}^{s} \in \mathbb{R}^{s}, \quad c = (c_i)_{i=1}^{s} \in \mathbb{R}^{s}, \quad \omega = (\omega_i)_{i=1}^{s} \in \mathbb{R}^{s}, \quad \theta \in \mathbb{R}. \end{aligned} \tag{5}$$

We rewrite the singly TASE-RK methods (4) in full below, with the purpose of highlighting: the steps needed to compute the solution, the connections with the TASE-W methods (2). From now on, we consider $c_1 = 0$.

$$\begin{aligned} &f_1 = f\left(t_n, y_n\right), \\ &\left(I - h\theta W\right)^q K_1^{[q]} = h f_1, \quad q = 1, \ldots, s, \\ &K_1 = \omega_1 K_1^{[1]} + \ldots + \omega_s K_1^{[s]}, \\ \\ &f_2 = f\left(t_n + c_2 h, y_n + a_{21} K_1\right), \\ &\left(I - h\theta W\right)^q K_2^{[q]} = h f_2, \quad q = 1, \ldots, s, \\ &K_2 = \omega_1 K_2^{[1]} + \ldots + \omega_s K_2^{[s]}, \\ \\ &\vdots \\ \\ &f_s = f\left(t_n + c_s h, y_n + \sum_{j=1}^{s-1} a_{sj} K_j\right), \\ &\left(I - h\theta W\right)^q K_s^{[q]} = h f_s, \quad q = 1, \ldots, s, \\ &K_s = \omega_1 K_s^{[1]} + \ldots + \omega_s K_s^{[s]}, \\ \\ &y_{n+1} = y_n + \sum_{j=1}^{s} b_j K_j. \end{aligned}$$

Note that, for each stage $K_i$, $i = 1, \ldots, s$, we need to compute $K_i^{[q]}$ from

$$\left(I - h\theta W\right)^q K_i^{[q]} = h f_i, \quad q = 1, \ldots, s.$$

This corresponds to perform the following Algorithm 2.

Therefore, the main computational effort of singly TASE-RK methods is given by the solution of $s^2$ linear systems at each step ($s$ linear systems for each stage) involving just 1 coefficient matrix $(I - h\theta W)$.

**Algorithm 2.** Computation of $K_i^{[q]}$, $q = 1, \ldots, s$, in singly TASE-RK methods.

1: compute $K_i^{[1]}$ by solving $(I - h\theta W)\, K_i^{[1]} = h f_i$
2: **for** $q = 2, \ldots, s$ **do**
3: solve $(I - h\theta W)\, K_i^{[q]} = K_i^{[q-1]}$
4: **end for**

*Remark 2.* Note that Algorithm 2 involves computing the $i$-th stage $K_i$ in (4) by sequential solution of the linear systems $(I - h\theta W)\, K_i^{[q]} = K_i^{[q-1]}$, $q = 2, \ldots, s$, starting from $K_i^{[1]}$ by $(I - h\theta W)\, K_i^{[1]} = h f_i$. Then, $K_i = \sum_{q=1}^{s} \omega_q K_i^{[q]}$. In [7, Subsec. 6], the authors propose to compute the $i$-th stage $K_i$ via a slightly different Horner-type algorithm, which can be well understood through an example. Let us fix e.g. $s = 4$, and consider the computation of the $i$-th stage $K_i$ in (4), which reads

$$K_i = \left( \sum_{q=1}^{4} \omega_q \left(I - h\theta W\right)^{-q} \right) h f_i, \quad f_i = \left( t_n + c_i h, y_n + \sum_{j=1}^{i-1} a_{ij} K_j \right).$$

This can be expressed as

$$K_i = \Bigg( \left(I - h\theta W\right)^{-1} \Big( \omega_1 I + \left(I - h\theta W\right)^{-1} \big( \omega_2 I + \left(I - h\theta W\right)^{-1} \left( \omega_3 I + \omega_4 \cdot \left(I - h\theta W\right)^{-1} \right) \big) \Big) \Bigg) h f_i.$$

Therefore, it is possible to determine the stage $K_i$ starting from the solution of the linear system

$$(I - h\theta W) \tilde{K}_i^{[4]} = \omega_4 h f_i.$$

Then, computing

$$\begin{aligned} (I - h\theta W) \tilde{K}_i^{[3]} &= \omega_3 h f_i + \tilde{K}_i^{[4]}, \\ (I - h\theta W) \tilde{K}_i^{[2]} &= \omega_2 h f_i + \tilde{K}_i^{[3]}, \\ (I - h\theta W) \tilde{K}_i^{[1]} &= \omega_1 h f_i + \tilde{K}_i^{[2]}, \end{aligned}$$

note that the stage $K_i$ corresponds to $\tilde{K}_i^{[1]}$. We emphasize that this algorithm does not imply changes in the number of linear systems to be solved at each time-step, but as mentioned in [7] it can bring advantages from a stability point of view.

### 3.1 Implementation Steps

According to [7, Sec. 2], to simplify some notation we make the change of variable $d = \theta^{-1}$. This leads to equivalently getting the stages $K_i$, $i = 1, \ldots, s$, of the

method (4) as

$$\begin{aligned} &(dI - hW)^q K_i^{[q]} = hf_i, \quad q = 1, \ldots, s, \\ &K_i = \beta_1 K_i^{[1]} + \ldots + \beta_s K_i^{[s]}. \end{aligned} \tag{6}$$

Here, the new $\beta_q$ coefficients and the $\omega_q$ coefficients in (4) are related as follows:

$$\beta_q = \omega_q \theta^{-q}, \quad q = 1, \ldots, s. \tag{7}$$

Using this, we report below the steps for an efficient implementation of the singly TASE-RK methods.

At a given grid point $t_n$, to compute the numerical solution at $t_{n+1}$, we perform the following steps.

1. Possible evaluation of the Jacobian of the problem at the current point, if the user chooses to select $W = J_n$.
2. Computation of the LU factorization of the only coefficient matrix involved in all the linear systems to be solved:

$$[L_d, U_d] = \mathtt{lu}\,(dI - hW)\,.$$

3. Computation of the stages $K_i$, $i = 1, \ldots, s$: according to Remark 2 and the change of variable $d = \theta^{-1}$ implying (6) and (7), we use the following Algorithm 3, leading to the solution of $2s$ triangular systems for each $K_i$.

**Algorithm 3.** Computation of $K_i$ in singly TASE-RK methods.

1: evaluate $f_i = f\left(t_n + c_i h, y_n + \sum_{j=1}^{i-1} a_{ij} K_j\right)$
2: initialize $\tilde{K}_i^{[s+1]} = \mathbf{0}$
3: **for** $q = s, \ldots, 1$ with step -1 **do**
4: compute $\tilde{K}_i^{[q]} = \left(U_d \backslash \left(L_d \backslash \left(\tilde{K}_i^{[q+1]} + \beta_q h f_i\right)\right)\right)$
5: **end for**
6: set $K_i = \tilde{K}_i^{[1]}$

4. Computation of the time-marching solution:

$$y_{n+1} = y_n + b_1 K_1 + \ldots + b_s K_s.$$

This implementation involves at each $t_n$: 1 LU factorization and $2s^2$ triangular linear systems, see the above steps 2 and 3. Similarly to the previous section, if $(dI - hW)$ does not admit pure LU factorization, we compute a PLU one and still use the above algorithm, premultiplying the coefficient vector of the linear system to be solved by the permutation matrix.

Finally, we observe that if the user chooses $W$ constant for the whole numerical integration, steps 1 and 2 need to be performed only once and not for each $n$: in this case the method would require 1 LU factorizations in total.

### 3.2 Examples of 2nd, 3rd and 4th Order Singly TASE-RK Methods

Below we report examples of efficient singly TASE-RK methods from [9, Subsecs. 3–5], by indicating the coefficients as in (5) where, instead of $\omega$ and $\theta$, we consider

$$\beta = (\beta_i)_{i=1}^{s} \in \mathbb{R}^s, \quad d = \theta^{-1} \in \mathbb{R},$$

according to the used change of variable and relations (7). For reasons of consistency, for all the methods considered below the $\beta_i$ coefficients are as follows [7, p. 6]:

$$\beta_i = \binom{s}{s-i} d^i (-1)^{-1+i}, \quad i = 1, \ldots, s.$$

*2nd Order Method: Singly-TRK2*
The coefficients of the method are as

$$d = \frac{1}{2},\ \beta = \left(1, -\frac{1}{4}\right)^T,\ b = (0,1)^T,\ c = \left(0, \frac{1}{2}\right)^T,\ A = \begin{pmatrix} 0 & 0 \\ \frac{1}{2} & 0 \end{pmatrix}.$$

*3rd Order Method: Singly-TRK3*
The coefficients of the method are as

$$b = \left(\frac{1}{6}, \frac{3}{10}, \frac{8}{15}\right)^T,\ c = \left(0, \frac{1}{3}, \frac{3}{4}\right)^T,\ A = \begin{pmatrix} 0 & 0 & 0 \\ \frac{1}{3} & 0 & 0 \\ -\frac{3}{16} & \frac{15}{16} & 0 \end{pmatrix},$$

and

$$\begin{aligned} &d = 0.532023879327774, \quad \beta_1 = 1.596071637983322, \\ &\beta_2 = -0.849148224524921, \quad \beta_3 = 0.150589044178680. \end{aligned}$$

*4th Order Method: Singly-TRK4*
The coefficients of the method are as

$$b = \left(\frac{1}{6}, \frac{1}{3}, \frac{1}{3}, \frac{1}{6}\right)^T,\ c = \left(0, \frac{1}{2}, \frac{1}{2}, 1\right)^T,\ A = \begin{pmatrix} 0 & 0 & 0 & 0 \\ \frac{1}{2} & 0 & 0 & 0 \\ 0 & \frac{1}{2} & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix},$$

and

$$\begin{aligned} &d = 0.399017909495830, \\ &\beta_1 = 1.596071637983322, \quad \beta_2 = -0.955291752590537, \\ &\beta_3 = 0.254119012051523, \quad \beta_4 = -0.025349509237986. \end{aligned}$$

**Table 2.** Properties of the considered singly TASE-RK methods: stability, number of triangular linear systems (N. tr. sys.) and LU factorizations (N. LU) required at each time-step, and maximum error coefficient (Err. coeff.) in absolute value. The symbol * refers to the choice $W = J_n$.

| Method | Stability* | N. tr. sys. | N. LU* | Err. coeff. |
|---|---|---|---|---|
| singly-TRK2 | strong A-stab. | 8 | 1 | 4 |
| singly-TRK3 | L($\alpha$)-stab., $\alpha = 88.99$ | 18 | 1 | 6.64 |
| singly-TRK4 | strong A($\alpha$)-stab., $\alpha = 87.17$ | 32 | 1 | 39.45 |

*Properties of the Considered Methods*

Table 2 reports the properties of the above-listed singly TASE-RK methods, in terms of stability, computational cost and error terms.

In Table 2, the properties of stability and the number of LU factorizations required at each time-step refer to the choice $W = J_n$. Indeed, as already highlighted, if $W$ is chosen as a constant matrix different from $J_n$, the number of factorizations is reduced to 1 in total, and not at each time-step. However, as for the TASE-W methods, the stability study of singly TASE-RK methods has been carried out under the assumption $W = J_n$. Finally, we recall that strong A-stability or strong A($\alpha$)-stability means that the value assumed by the stability function of the methods at infinity is strictly less than 1. In particular, for the singly-TRK2 and singly-TRK4 this value is 0.5 and 0.27, respectively.

## 4 Experiments on the DIB System for Battery Modeling

In this section, we apply the methods illustrated in this paper to solve the 2D DIB model, which was introduced to characterize the electrodeposition process in batteries [31]. It is given by the following system of coupled PDEs:

$$\begin{cases} \dfrac{\partial \eta_1}{\partial t} = \Delta \eta_1 + \rho \left( A_1 (1 - \eta_2) \eta_1 - A_2 \eta_1^3 - B(\eta_2 - a) \right), \\ \dfrac{\partial \eta_2}{\partial t} = d \Delta \theta + \rho \big( C \left(1 + k_2 \eta_1\right) \left(1 - \eta_2\right) \left(1 - \gamma \left(1 - \eta_2\right)\right) - \\ \qquad D \eta_2 \left(1 + \gamma \eta_1\right) \left(1 + k_3 \eta_1\right) \big). \end{cases}$$

Here, $\eta_1 : \Omega \times [0, t_{\text{end}}] \to \mathbb{R}$ and $\eta_2 : \Omega \times [0, t_{\text{end}}] \to [0, 1]$. We consider $\Omega = [0, 5]^2$, zero Neumann boundary conditions, and as initial condition

$$\eta_1(x_1, x_2, t_0) = 10^{-5}\, \texttt{rand}(x_1, x_2), \quad \eta_2(x_1, x_2, t_0) = a + 10^{-5}\, \texttt{rand}(x_1, x_2).$$

Above, `rand` refers to the Matlab function that generates uniformly distributed random numbers. We consider the following values for the parameters: $a = 0.5$, $\gamma = 0.2$, $\rho = 1$, $A_1 = 10$, $A_2 = 30$, $B = 66$, $C = 3$, $d = 20$, $D = 2.4545$, $k_2 = 2.5$, $k_3 = 1.5$. We carry out the space discretization with order-two finite differences, using the same number of points in both directions $x_1$ and $x_2$. The Neumann conditions are imposed introducing ghost points, see [34]. For the space discretization, we consider step-size $\Delta x = 0.5$, thus obtaining

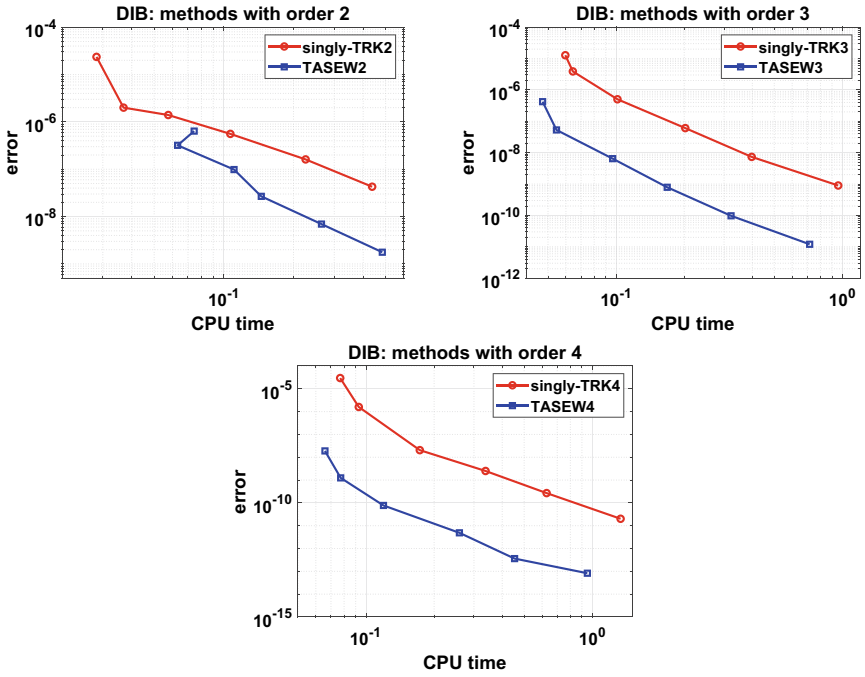


**Fig. 1.** Performance of the TASE-W and singly TASE-RK methods of order $p = 2, 3, 4$ on the DIB model for $h = (t_{\text{end}} - t_0)/N$, $N = 2^5, \ldots, 2^{10}$, with $W = J_0$.

**Table 3.** CPU times in seconds used by methods for different values of $N$ for $W = J_0$.

| Method | $N = 2^5$ | $N = 2^6$ | $N = 2^7$ | $N = 2^8$ | $N = 2^9$ | $N = 2^{10}$ |
|---|---|---|---|---|---|---|
| singly-TRK2 | 0.03 | 0.04 | 0.06 | 0.11 | 0.23 | 0.44 |
| TASEW2 | 0.07 | 0.06 | 0.11 | 0.15 | 0.26 | 0.48 |
| singly-TRK3 | 0.06 | 0.06 | 0.10 | 0.20 | 0.40 | 0.96 |
| TASEW3 | 0.05 | 0.05 | 0.10 | 0.17 | 0.32 | 0.72 |
| singly-TRK4 | 0.08 | 0.09 | 0.17 | 0.34 | 0.63 | 1.33 |
| TASEW4 | 0.07 | 0.08 | 0.12 | 0.26 | 0.45 | 0.95 |

11 grid points in each direction. Therefore, the resulting system of ODEs has size $m = 2 \cdot 11^2 = 242$. We apply the TASE-W and singly TASE-RK methods for several values of $h = (t_{\text{end}} - t_0)/N$, $N = 2^5, \ldots, 2^{10}$, selecting both constant $W$ corresponding to the Jacobian only at the initial point, namely $W = J_0$, and $W$ corresponding exact Jacobian at each time-step, namely $W = J_n$.

The results are reported in Figs. 1 and 2, where we compare the CPU time used by the methods in seconds, and the absolute error in infinity norm computed at the last point of the time grid. The reference solution has been computed with the MATLAB function `ode15s`, setting `AbsTol`$= 10^{-14}$ `RelTol`$= 10^{-14}$. In particular, Fig. 1 shows the case $W = J_0$, while Fig. 2 shows the case $W = J_n$. Also,

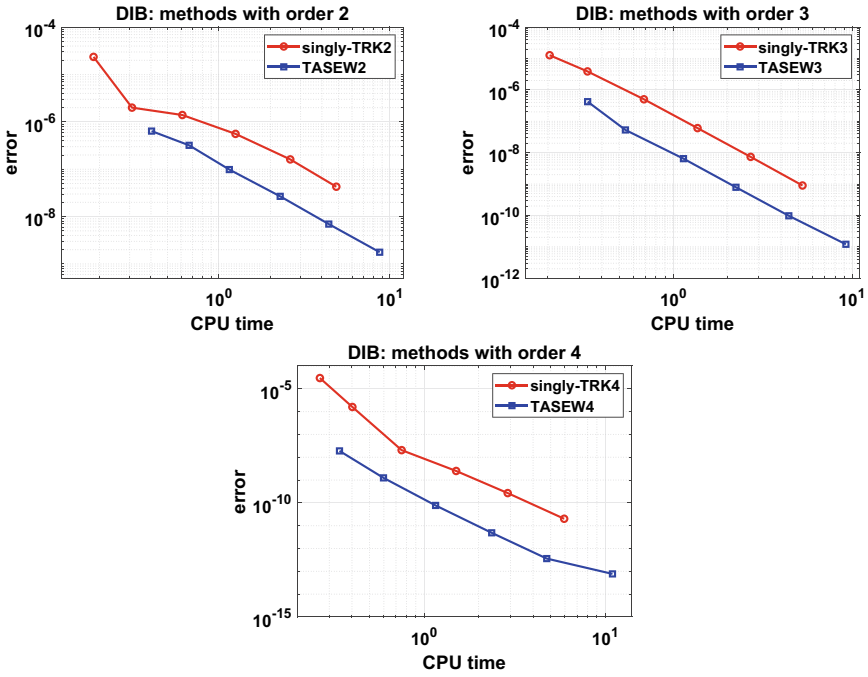


**Fig. 2.** Performance of the TASE-W and singly TASE-RK methods of order $p = 2, 3, 4$ on the DIB model for $h = (t_{\text{end}} - t_0)/N$, $N = 2^5, \ldots, 2^{10}$, with $W = J_n$.

Tables 3 and 4 report the values of the computing times used by the methods for $W = J_0$ and $W = J_n$, respectively. Note that, for $W = J_0$, the TASE-W and singly TASE-RK methods exhibit comparable computing times, but the former provide the solution with the lowest errors. This is consistent with the maximum error coefficients of the methods given in Tables 1 and 2. Note also that, although we are not using exact Jacobian in this case, all the methods are stable for any considered value of the step-size $h$. For $W = J_n$, clearly the computing times of the methods increase, but the singly TASE-RK schemes become faster than the TASE-W schemes, for the same step-size $h$. This is consistent with the fact that, if $W = J_n$, we need to compute at each step a factorization of the coefficient matrices of the linear systems to be solved. In this case, it is more convenient to work with the smallest possible number of coefficient matrices. From this point of view, as explained in the paper, singly TASE-RK methods are advantageous over TASE-W methods.

All the tests have been done using the version R2023b of MATLAB and a laptop with processor Intel(R) Xeon(R) CPU E3-1245 v5 3.50 GHz, RAM of 16 Gb, on Windows 10 Enterprise 64–bit 22H2.

**Table 4.** CPU times in seconds used by methods for different values of $N$ for $W = J_n$.

| Method | $N = 2^5$ | $N = 2^6$ | $N = 2^7$ | $N = 2^8$ | $N = 2^9$ | $N = 2^{10}$ |
|---|---|---|---|---|---|---|
| singly-TRK2 | 0.18 | 0.31 | 0.61 | 1.25 | 2.61 | 4.86 |
| TASEW2 | 0.40 | 0.67 | 1.15 | 2.28 | 4.39 | 8.71 |
| singly-TRK3 | 0.20 | 0.33 | 0.69 | 1.37 | 2.71 | 5.27 |
| TASEW3 | 0.33 | 0.54 | 1.14 | 2.24 | 4.42 | 9.20 |
| singly-TRK4 | 0.27 | 0.40 | 0.75 | 1.50 | 2.90 | 5.94 |
| TASEW4 | 0.34 | 0.60 | 1.16 | 2.36 | 4.76 | 10.98 |

## 5 Conclusions

In this work, we have described in detail the TASE-W and singly TASE-RK methods, recently introduced in the scientific literature for the efficient solution of large stiff IVPs. In particular, we have analyzed their properties of consistency, stability and computational cost, proposing an efficient implementation strategy. Numerical tests on a system of PDEs modeling charge/discharge processes in electric batteries have shown the good performance of both families of methods. In particular, they have shown that the TASE-W methods are advantageous for obtaining a solution with high accuracy, even with large step-size $h$. Instead, to obtain a stable solution by sacrificing great accuracy, in order to use low computing times, for the same step-size the singly TASE-RK methods are advantageous compared to the TASE-W methods.

**Acknowledgments.** The authors are members of the research group GNCS-INdAM. This work has been supported by GNCS-INdAM projects and by the Italian Ministry of University and Research (MUR), through the PRIN PNRR 2022 projects BATMEN (P20228C2PP, CUP: F53D23010020001) and MatForPat (P2022WC2ZZ, CUP: E53D23018030001).

**Disclosure of Interests.** The authors have no competing interests to declare that are relevant to the content of this article.

## References

1. Aceto, L., Conte, D., Pagano, G.: On a generalization of time-accurate and highly-stable explicit operators for stiff problems. Appl. Numer. Math. **200**, 2–17 (2024). https://doi.org/10.1016/j.apnum.2023.04.001
2. Aceto, L., Conte, D., Pagano, G.: Modified TASE Runge-Kutta methods for integrating stiff differential equations. SIAM J. Sci. Comput. **47**(3), A1652–A1680 (2025). https://doi.org/10.1137/24M1667336
3. Amodio, P., Brugnano, L., Frasca-Caccia, G., Iavernaro, F.: Arbitrarily high-order energy-conserving methods for Hamiltonian problems with quadratic holonomic constraints. J. Comput. Math. **42**(4), 1145–1171 (2024). https://doi.org/10.1016/j.cam.2024.115826

4. Bassenne, M., Fu, L., Mani, A.: Time-accurate and highly-stable explicit operators for stiff differential equations. J. Comput. Phys. **424**, 109847 (2021). https://doi.org/10.1016/j.jcp.2020.109847
5. Brugnano, L., Frasca-Caccia, G., Iavernaro, F., Vespri, V.: A new framework for polynomial approximation to differential equations. Adv. Comput. Math. **48**(6), 76 (2022). https://doi.org/10.1007/s10444-022-09992-w
6. Buckwar, E., D'Ambrosio, R.: Exponential mean-square stability properties of stochastic linear multistep methods. Adv. Comput. Math. **47**(4), 1–14 (2021). https://doi.org/10.1007/s10444-021-09879-2
7. Calvo, M., Fu, L., Montijano, J.I., Rández, L.: Singly TASE operators for the numerical solution of stiff differential equations by explicit Runge-Kutta schemes. J. Sci. Comput. 96(1) (2023). https://doi.org/10.1007/s10915-023-02232-3
8. Calvo, M., Montijano, J.I., Rández, L.: Modified Singly-Runge–Kutta-TASE methods for the numerical solution of stiff differential equations. J. Sci. Comput. 103(1) (2025). https://doi.org/10.1007/s10915-025-02813-4
9. Calvo, M., Montijano, J.I., Rández, L.: A note on the stability of time-accurate and highly-stable explicit operators for stiff differential equations. J. Comput. Phys. **436**, 110316 (2021). https://doi.org/10.1016/j.jcp.2021.110316
10. Cardone, A., Conte, D., Paternoster, B.: Stability of two-step spline collocation methods for initial value problems for fractional differential equations. Commun. Nonlinear Sci. Numer. Simul. **115**, 106726 (2022). https://doi.org/10.1016/j.cnsns.2022.106726
11. Cardone, A., Frasca-Caccia, G.: Numerical conservation laws of time fractional diffusion PDEs. Fract. Calc. Appl. Anal. **25**(4), 1459–1483 (2022). https://doi.org/10.1007/s13540-022-00059-7
12. Citro, V., D'Ambrosio, R.: Long-term analysis of stochastic theta-methods for damped stochastic oscillators. Appl. Numer. Math. **150**, 18–26 (2020). https://doi.org/10.1016/j.apnum.2019.08.011
13. Conte, D., D'Ambrosio, R., Jackiewicz, Z.: Two-step Runge-Kutta methods with quadratic stability functions. J. Sci. Comput. **2**, 191–218 (2010). https://doi.org/10.1007/s10915-010-9378-x
14. Conte, D., González-Pinto, S., Hernández-Abreu, D., Pagano, G.: On approximate matrix factorization and TASE W-methods for the time integration of parabolic partial differential equations. J. Sci. Comput. **100**(2), 34 (2024). https://doi.org/10.1007/s10915-024-02579-1
15. Conte, D., Martin-Vaquero, J., Pagano, G., Paternoster, B.: Stability theory of TASE–Runge–Kutta methods with inexact Jacobian. SIAM J. Sci. Comput. **46**(6), A3628–A3657 (2024). https://doi.org/10.1137/24M1631869
16. R. D'Ambrosio: Numerical Approximation of Ordinary Differential Problems - From Deterministic to Stochastic Numerical Methods. Springer (2023). https://doi.org/10.1007/978-3-031-31343-1
17. D'Ambrosio, R., Di Giovacchino, S.: Strong backward error analysis of symplectic integrators for stochastic Hamiltonian systems. Appl. Math. Comput. **467**, 128488 (2024). https://doi.org/10.1016/j.amc.2023.128488
18. D'Ambrosio, R., Di Giovacchino, S.: Long-term analysis of stochastic Hamiltonian systems under time discretizations. SIAM J. Sci. Comput. **45**(2), A257–A288 (2023). https://doi.org/10.1137/21M1458612
19. D'Ambrosio, R., Di Giovacchino, S.: Numerical conservation issues for the stochastic Korteweg-de Vries equation. J. Comput. Appl. Math. **424**, 114967 (2023). https://doi.org/10.1016/j.cam.2022.114967

20. D'Ambrosio, R., Di Giovacchino, S.: Mean-square contractivity of stochastic theta-methods. Commun. Nonlinear Sci. Numer. Simul. **96**, 105671 (2021). https://doi.org/10.1016/j.cnsns.2020.105671
21. D'Ambrosio, R., Di Giovacchino, S.: Nonlinear stability issues for stochastic Runge-Kutta methods. Commun. Nonlinear Sci. Numer. Simul. **94**, 105549 (2021). https://doi.org/10.1016/j.cnsns.2020.105549
22. D'Ambrosio, R., Giordano, G., Paternoster, B., Ventola, A.: Perturbative analysis of stochastic Hamiltonian problems under time discretizations. Appl. Math. Lett. **120**, 107223 (2021). https://doi.org/10.1016/j.aml.2021.107223
23. D'Ambrosio, R., Izzo, G., Jackiewicz, Z.: Search for highly stable two-step Runge-Kutta methods for ODEs. Appl. Numer. Math. **62**(10), 1361–1379 (2012). https://doi.org/10.1016/j.apnum.2012.06.012
24. D'Ambrosio, R., Paternoster, B.: Multivalue collocation methods free from order reduction. J. Comput. Appl. Math. **387**, 112515 (2021). https://doi.org/10.1016/j.cam.2019.112515
25. Frasca-Caccia, G.: Finite difference schemes with non polynomial local conservation laws. J. Comput. Appl. Math. **458**, 116330 (2025). https://doi.org/10.1016/j.cam.2024.116330
26. Frasca-Caccia, G., Hydon, P.E.: A new technique for preserving conservation laws. Found. Comput. Math. **22**(2), 477–506 (2022). https://doi.org/10.1007/s10208-021-09511-1
27. Frasca-Caccia, G., Hydon, P.E.: Numerical preservation of multiple local conservation laws. Appl. Math. Comput. **403**, 126203 (2021). https://doi.org/10.1016/j.amc.2021.126203
28. Frasca-Caccia, G., Hydon, P.E.: Simple bespoke preservation of two conservation laws. IMA J. Numer. Anal. **40**(2), 1294–1329 (2020). https://doi.org/10.1093/imanum/dry087
29. Frasca-Caccia, G., Singh, P.: Optimal parameters for numerical solvers of PDEs. J. Sci. Comput. **97**(1), 11 (2023). https://doi.org/10.1007/s10915-023-02324-0
30. Frasca-Caccia, G., Valentino, C., Colace, F., Conte, D.: An overview of differential models for corrosion of cultural heritage artefacts. Math. Model. Nat. Phenom. **18**, 27 (2023). https://doi.org/10.1051/mmnp/2023031
31. Frittelli, M., Sgura, I.: Matrix-oriented FEM formulation for reaction-diffusion PDEs on a large class of 2D domains. Appl. Numer. Math. **200**, 286–308 (2024). https://doi.org/10.1016/j.apnum.2023.07.010
32. González-Pinto, S., Hernández-Abreu, D., Pagano, G., Pérez-Rodríguez, S.: Generalized TASE-RK methods for stiff problems. Appl. Numer. Math. **188**, 129–145 (2023). https://doi.org/10.1016/j.apnum.2023.03.007
33. Hairer, E., Wanner, G.: Solving ordinary differential equations II - Stiff and Differential Algebraic Problems. Springer Series in Computational Mathematics (14), Springer, Berlin (1996). https://doi.org/10.1007/978-3-642-05221-7
34. Hundsdorfer, W., Verwer, J.: Numerical Solution of Time-Dependent Advection Diffusion-Reaction Equations. Springer (2007). https://doi.org/10.1007/9783-662-09017-6