

High Order Semi-implicit Schemes for Time Dependent Partial Differential Equations

Sebastiano Boscarino¹ · Francis Filbet² ·
Giovanni Russo¹

Received: 28 April 2014 / Revised: 8 January 2016 / Accepted: 14 January 2016 /
Published online: 28 January 2016
© Springer Science+Business Media New York 2016

Abstract The main purpose of the paper is to show how to use implicit–explicit Runge–Kutta methods in a much more general context than usually found in the literature, obtaining very effective schemes for a large class of problems. This approach gives a great flexibility, and allows, in many cases the construction of simple linearly implicit schemes without any Newton’s iteration. This is obtained by identifying the (possibly linear) dependence on the unknown of the system which generates the stiffness. Only the stiff dependence is treated implicitly, then making the whole method much simpler than fully implicit ones. The resulting schemes are denoted as *semi-implicit* R–K. We adopt several semi-implicit R–K methods up to order three. We illustrate the effectiveness of the new approach with many applications to reaction–diffusion, convection diffusion and nonlinear diffusion system of equations.

Keywords IMEX schemes · Stiff problems · Time dependant partial differential equations

Mathematics Subject Classification Primary 82C40 · Secondary 65N08 · 65N35

1 Introduction

A well-known approach in the numerical solution of evolutionary problems in partial differential equations is the method of lines. In this approach a partial differential equation is

✉ Sebastiano Boscarino
boscarino@dmi.unict.it

Francis Filbet
francis.filbet@math.univ-toulouse.fr

Giovanni Russo
russo@dmi.unict.it

¹ Department of Mathematics and Computer Science, University of Catania, Via A.Doria 6, 95125 Catania, Italy

² Institut de Mathématiques de Toulouse, Université Paul Sabatier, Toulouse III & IUF, 118, Route de Narbonne, 31062 Toulouse Cedex, France

first discretized in space by finite difference or finite element techniques and converted into a system of ordinary differential equations (ODEs). In some cases the right hand side can be written as the sum of two terms, a stiff one and a non stiff one:

$$\begin{cases} \frac{du}{dt}(t) = F(t, u(t)) + \frac{1}{\varepsilon} G(t, u(t)), & \forall t \geq t_0, \\ u(t_0) = u_0, \end{cases} \quad (\mathcal{A})$$

where ε is a small parameter, which generates some stiffness in the system. We call such stiff problem of *additive type* and hereafter denoted by (\mathcal{A}) .

The development of numerical schemes for systems of stiff ODEs of the form (\mathcal{A}) attracted a lot of attention in the last decades. Systems of such form often arise from the discretization of partial differential equations, such as convection–diffusion equations and hyperbolic systems with relaxation. In previous works we considered the latter case which in recent years has been a very active field of research, due to its great impact on applied sciences. In fact, relaxation is important in many physical situations, for example it arises in discrete kinetic theory of rarefied gases, hydrodynamical models for semiconductors, linear and non-linear waves, viscoelasticity, traffic flows, shallow water [1–8].

Hopefully, when a problem with easily separable stiff and non-stiff components is considered, a combination of implicit and explicit Runge–Kutta methods can be used. The implicit method is used to treat the stiff component $G(t, u(t))/\varepsilon$ in a stable fashion while the non-stiff component $F(t, u(t))$ of the system is treated using the explicit scheme. These combined implicit/explicit (IMEX) schemes are already used for several problems, including convection–diffusion–reaction systems, hyperbolic systems with relaxation, collisional kinetic equations, and so on.

However not all systems containing stiff terms appear in partitioned or additive form, and therefore the use of standard IMEX schemes is not straightforward. In such cases one usually relies on fully implicit schemes.

In the context of ODEs, several authors usually call *semi-implicit RK methods* (in contrast to implicit RK methods) numerical schemes that require the solution of linear systems of equations for the computation of the numerical solution with no Newton iteration. A typical case is given by Rosenbrock schemes, [9], which are linearly implicit schemes, that do not make use of the particular structure of the system. Similarly, in the context of PDEs, some authors denote by *semi-implicit* additive schemes in which the two tableau correspond respectively to an explicit and an implicit scheme (see [10, 11]).

In other cases, semi-implicit schemes denote methods for the numerical solution of a problem of the form

$$\frac{du}{dt} = f(t, u),$$

obtained by adding and subtracting an approximation $g(t, u)$ of $f(t, u)$ which is more amenable for an implicit treatment:

$$\frac{du}{dt} = f(t, u) - g(t, u) + g(t, u).$$

Then the term $f(t, u) - g(t, u)$ is treated explicitly, while $g(t, u)$ is treated implicitly.

Examples of this type are given in [12], where the authors consider nonlinear hyperbolic systems containing fully nonlinear and stiff relaxation terms in the limit of arbitrary late times. The dynamics is asymptotically governed by effective systems which are of parabolic

type and may contain degenerate and/or fully nonlinear diffusion terms. Fully nonlinear relaxation terms can arise, for instance, in presence of *strong friction*, see for example in [13] and references therein. Furthermore, a general class of models of the same type were introduced by Kawashima and LeFloch (see for example [12]). For such problems in [12], the authors introduced a semi-implicit formulation based on implicit–explicit (IMEX) Runge–Kutta methods. Similarly in [14], the author introduced a semi-implicit method for computing the two models of motion by mean curvature and motion by surface diffusion which is stable for large time steps. In all such models a semi-implicit method is more effective than a fully implicit one. Other examples are given in [15], where the authors construct a very effective solver for the Boltzmann equation near the fluid dynamic regime. In this case f denotes the Boltzmann operator, while g denotes a suitable BGK approximation. A similar technique is adopted in [16, 17] in the context of Navier–Stokes equations.

In other cases the stiffness is associated to some variables. For example, if a system can be written in the *partitioned* form, hereafter denoted by (\mathcal{P}) ,

$$\begin{cases} \frac{dy}{dt}(t) = \mathcal{F}_1(t, y(t), z(t)), \\ \varepsilon \frac{dz}{dt}(t) = \mathcal{F}_2(t, y(t), z(t)), \end{cases} \quad (\mathcal{P})$$

then the stiffness is associated to variable z , and the corresponding equation will be treated implicitly, while the equation for y is treated explicitly. In other cases it is more convenient to associate the stiffness to a part of the right hand side, for example if a system has the *additive* form (\mathcal{A}) , in this case the term $F(t, u(t))$ is treated explicitly while $G(t, u(t))/\varepsilon$ is treated implicitly. It can be shown that the same system can be written in either form, however sometimes one of the two forms is more convenient.

Directly motivated by the above cases, we consider a more general class of problems of the form

$$\begin{cases} \frac{du}{dt}(t) = \mathcal{H}(t, u(t), u(t)/\varepsilon), \quad \forall t \geq t_0, \\ u(t_0) = u_0, \end{cases} \quad (\mathcal{G})$$

where the function $\mathcal{H}: \mathbb{R} \times \mathbb{R}^m \times \mathbb{R}^m \rightarrow \mathbb{R}^m$ is sufficiently differentiable and the right hand side has a stiff dependence only on the last argument, emphasized by the appearance on the small parameter ε in the denominator. We denote this class of problems as *generalized partitioned form* and hereafter denoted by (\mathcal{G}) . All the cases mentioned before belong to this more general class. In our paper we denote by *semi-implicit schemes* numerical methods which solve problems of the form (\mathcal{G}) in which the variable u appearing as the second argument of \mathcal{H} is treated explicitly, while u appearing as third argument is treated implicitly.

Remark 1.1 Note that the parameter ε in (\mathcal{G}) does not necessarily appear explicitly, but it just indicates some stiffness in the term. Sometime the stiffness is hidden, and is not explicitly expressed in terms of a small parameter ε . For example, in the case of diffusion terms $\varepsilon = \mathcal{O}(\Delta x^2)$, see [12, 18–20].

The main goal of the paper is to focus on the treatment of systems of the form (\mathcal{G}) , by using some IMEX schemes already presented in the literature. Furthermore, in order to simplify the expression of the formulas, we drop the dependence of the parameter ε in the second argument as mentioned before, keeping in mind that the dependence on the second argument

is stiff. Then in this paper we consider the general class of non-autonomous problems of the form

$$\begin{cases} \frac{du}{dt}(t) = \mathcal{H}(t, u(t), u(t)), \forall t \geq t_0, \\ u(t_0) = u_0, \end{cases} \tag{1}$$

The aim of this paper is to propose a new approach, based on IMEX Runge–Kutta methods, for the construction of semi-implicit schemes. The approach is somehow inspired by partitioned Runge–Kutta methods [21] and related to the additive Runge–Kutta methods of Zhong [11].

In particular, we show several examples of systems of the form \mathcal{G} that can be efficiently solved with the new approach. In order to apply this idea, one has to identify where the dependence of the unknown is stiff. We do not propose a general technique to automatically identify the stiff dependence in the function \mathcal{H} that defines the system, rather we try to give arguments that help identifying the stiff dependence in each of the examples that we present. The problem of identifying the stiff term in a complicated system of equations is not new, and in several practical cases it is not so straightforward. In the context of numerical methods for systems of ODE’s, the problem of automatic stiffness detection is treated, for example, in the classical book of Hairer and Wanner ([9], p. 21). In the case in which a decision has to be made about where a variable has to be treated explicitly and where implicitly is an even more delicate one, and the research on techniques that automatically identify, in a general system, which is the dependence that is responsible for the stiffness, and whether it can be effectively treated linearly is far beyond the scope of the present paper.

In the next section, we describe the general framework to construct semi-implicit Runge–Kutta schemes based on approach outlined before. Several schemes are proposed with different stability properties and order of accuracy. We next compare the numerical solutions with exact ones available in the literature for reaction–diffusion problem and nonlinear convection–diffusion equation. After this validation step, we perform several numerical computations to show the robustness of our approach (nonlinear Fokker–Planck equation, Hele–Shaw flow and surface diffusion flow).

2 Relation Between the Various Types of Systems

In this section we show the following hierarchical dependence.

Proposition 2.1 *Consider a system in partitioned form (\mathcal{A}). Then it can be written either in the additive form (\mathcal{A}) or in the generalized partitioned form (\mathcal{G}).*

This hierarchical dependence can be illustrated by the following Fig. 1.

The first part of the Proposition, $\mathcal{P} \subset \mathcal{A}$, is self evident. To show that $\mathcal{P} \subset \mathcal{G}$ we consider the system (\mathcal{P}) and set $u = (u_1, u_2)$ with $u_1 = y$ and $u_2 = \varepsilon z$, then we have from (\mathcal{P})

$$\begin{cases} \frac{du_1}{dt}(t) = \mathcal{F}_1(t, u_1(t), u_2(t)/\varepsilon), \\ \frac{du_2}{dt}(t) = \mathcal{F}_2(t, u_1(t), u_2(t)/\varepsilon), \end{cases}$$

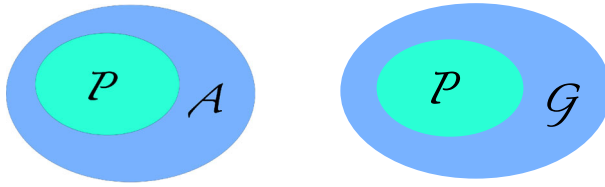


Fig. 1 Illustration of the hierarchical dependence of (P) , with respect to (A) and (G) approach

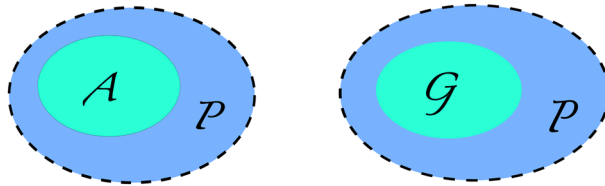


Fig. 2 Illustration of hierarchical dependence between partitioned approach (P) , additive approach (A) and generalised partitioned form (G) approach where the dashed lines represent the set of systems with doubled number of unknowns

that is, system (G) with $u = (u_1, u_2)$ and

$$\begin{pmatrix} \mathcal{F}_1(t, u_1(t), u_2(t)/\varepsilon) \\ \mathcal{F}_2(t, u_1(t), u_2(t)/\varepsilon) \end{pmatrix} = \mathcal{H}(t, u(t), u(t)/\varepsilon).$$

We note that in this two cases no duplications of unknowns is needed. Unfortunately there is no such hierarchical dependence between (A) and (G) systems. Note however that if we allow to double the number of unknowns, then we have the following dependence.

Proposition 2.2 *We have the following assertions*

- (1) *Consider a system in the additive form (A) . Then by doubling the number of unknowns, it can be written in the partitioned form (P) .*
- (2) *Consider a system in the generalized partitioned form (G) . Then by doubling the number of unknowns, it can be written in the partitioned form (P) .*

This hierarchical dependence can be illustrated by the following Fig. 2.

Part (1) of the Proposition is well known in the literature (see, for example, [22]). To prove (2), we consider a solution to (G) and set $v = u/\varepsilon$. Thus, we have

$$\begin{cases} \frac{du}{dt}(t) = \mathcal{H}(t, u(t), v(t)), \\ \varepsilon \frac{dv}{dt}(t) = \mathcal{H}(t, u(t), v(t)), \end{cases} \tag{2}$$

it is a particular case of system (P) . Note that these two last cases require duplication of unknowns.

Such inclusions are very important, since the analysis of the numerical schemes applied to one family of schemes is automatically valid also for schemes applied to the other two families. As far as we know, it is not possible to write system (A) in the form (G) or viceversa, without doubling the number of unknowns.

Nevertheless, we will see later (“Appendix”) that the computational cost and the number of evaluations of \mathcal{H} does not increase.

Thus, the formal equivalence among the various systems allows us to adopt techniques well known for additive or partitioned systems to more general cases.

A remark is in order at this point. A vast literature exists on the formal analysis of systems (\mathcal{P}), which are denoted as *singular perturbation systems*, [2, 9, 23–26]. However, system (2) is only formally a particular case of (\mathcal{P}), and the analysis developed for the former cannot be directly applied to this case, since now the two functions \mathcal{F}_1 and \mathcal{F}_2 are the same. Furthermore, a detailed asymptotic analysis of the schemes presented here goes beyond the scope of the paper, and will be considered in a future work for specific applications where the asymptotic behavior $\varepsilon \rightarrow 0$ is well understood at the continuous level [27, 28].

3 Numerical Methods for ODEs

In this section we review the concept of partitioned Runge–Kutta methods and derive a new class of semi-implicit R–K schemes, and we propose several schemes up to third order of accuracy, based on IMEX Runge–Kutta schemes already existing in the literature.

3.1 From Partitioned to Semi-implicit Runge–Kutta Methods

In the literature some interesting numerical methods do not belong to the classical class of implicit or explicit Runge–Kutta methods. They are called *partitioned* Runge–Kutta methods, [9, 21]. In order to present these methods we consider non-autonomous differential equations in the partitioned form,

$$\begin{cases} \frac{dy}{dt}(t) = \mathcal{F}_1(t, y(t), z(t)), \\ \frac{dz}{dt}(t) = \mathcal{F}_2(t, y(t), z(t)), \end{cases} \tag{3}$$

where $y(t)$ and $z(t)$ may be vectors of different dimensions and $y(t_0) = y_0, z(t_0) = z_0$ are the initial conditions.

The idea of the partitioned Runge–Kutta methods is to apply two different Runge–Kutta methods, i.e.

$$\begin{array}{c|c} \hat{c} & \hat{A} \\ \hline & \\ \hline \hat{b}^T & \\ \hline & \\ \hline c & A \\ \hline & \\ \hline & b^T \end{array} \tag{4}$$

where we treat the first variable y with the first method, $\hat{A}, \hat{b}^T = (\hat{b}_1, \dots, \hat{b}_s), \hat{c} = (\hat{c}_1, \dots, \hat{c}_s)$ and the second variable z with the second method, $A, b^T = (b_1, \dots, b_s), c = (c_1, \dots, c_s)$ under the usual assumption

$$\sum_j \hat{a}_{i,j} = \hat{c}_i, \quad \text{and} \quad \sum_j a_{ij} = c_i, \quad \text{for } 1 \leq i \leq s. \tag{5}$$

In other words, if we consider a numerical approximation (y^n, z^n) of (3) at time t^n , a partitioned Runge–Kutta method for the solution of (3) is given by

$$\begin{cases} k_i = \mathcal{F}_1 \left(t^n + \hat{c}_i \Delta t, y^n + \Delta t \sum_{j=1}^s \hat{a}_{ij} k_j, z^n + \Delta t \sum_{j=1}^s a_{ij} \ell_j \right), & 1 \leq i \leq s, \\ \ell_i = \mathcal{F}_2 \left(t^n + c_i \Delta t, y^n + \Delta t \sum_{j=1}^s \hat{a}_{ij} k_j, z^n + \Delta t \sum_{j=1}^s a_{ij} \ell_j \right), & 1 \leq i \leq s \end{cases} \tag{6}$$

and the numerical solution at the next time step is given by

$$\begin{cases} y^{n+1} = y^n + \Delta t \sum_{i=1}^s \hat{b}_i k_i, \\ z^{n+1} = z^n + \Delta t \sum_{i=1}^s b_i \ell_i. \end{cases} \tag{7}$$

We observe that we can rewrite system (1) as a partition one

$$\begin{cases} \frac{dy}{dt}(t) = \mathcal{H}(t, y(t), z(t)), \\ \frac{dz}{dt}(t) = \mathcal{H}(t, y(t), z(t)), \end{cases} \tag{8}$$

with initial conditions $y(t_0) = y_0, z(t_0) = y_0$. In such a case the solution of system (8) satisfies $z(t) = y(t)$ for any $t \geq t_0$ and is also a solution of Eq. (1). The system is a particular case of partitioned system in which $\mathcal{F}_1 = \mathcal{F}_2$ but with an additional computational cost since we double the number of variables. Applying the partitioned Runge–Kutta method (6)–(7) we have

$$\begin{cases} k_i = \mathcal{H}(t^n + \hat{c}_i \Delta t, Y_i, Z_i), & 1 \leq i \leq s, \\ \ell_i = \mathcal{H}(t^n + c_i \Delta t, Y_i, Z_i), & 1 \leq i \leq s, \end{cases} \tag{9}$$

with

$$\begin{cases} Y_i = y^n + \Delta t \sum_{j=1}^s \hat{a}_{i,j} k_j, & 1 \leq i \leq s, \\ Z_i = z^n + \Delta t \sum_{j=1}^s a_{ij} \ell_j, & 1 \leq i \leq s, \end{cases}$$

and the numerical solutions at the next time step are

$$\begin{cases} y^{n+1} = y^n + \Delta t \sum_{i=1}^s \hat{b}_i k_i, \\ z^{n+1} = z^n + \Delta t \sum_{i=1}^s b_i \ell_i. \end{cases}$$

In general, k_i and ℓ_i given by (9) for all $1 \leq i \leq s$ are different. However, there are two cases in which $k_i = \ell_i$, $i = 1, \dots, s$. The first one is when the system is autonomous, i.e. \mathcal{H} does not explicitly depend on time, and the second one is when $\hat{c}_i = c_i$, $i = 1, \dots, s$. In these two cases only one evaluation of \mathcal{H} is needed in (9), and only one set of stage fluxes is computed:

$$k_i = \mathcal{H}\left(y^n + \Delta t \sum_{j=1}^s \hat{a}_{i,j} k_j, z^n + \Delta t \sum_{j=1}^s a_{ij} k_j\right), \quad 1 \leq i \leq s.$$

From now on we assume that the system is autonomous. This restriction can be removed: even in the general case of a non-autonomous system, and $\hat{\mathbf{c}} \neq \mathbf{c}$, it is still possible to derive a scheme that does not require two sets of stage fluxes. The details are reported in the ‘‘Appendix’’.

Now we are ready to propose semi-implicit Runge–Kutta methods in order to solve autonomous problem of the form (8), i.e.

$$\begin{cases} \frac{dy}{dt} = \mathcal{H}(y, z), \\ \frac{dz}{dt} = \mathcal{H}(y, z), \end{cases} \tag{10}$$

with initial conditions $y(t_0) = y_0$, $z(t_0) = y_0$, where we treat the first variable y explicitly, and the second one z , implicitly.

From now on we shall adopt IMEX R–K schemes with $\mathbf{b} = \hat{\mathbf{b}}$. It is usual to consider diagonally implicit R–K (DIRK) schemes for the implicit part, [9]. In addition to be simpler to implement, this ensures that the terms involving the first argument of \mathcal{H} are indeed explicitly computed. The coefficients of the method are usually represented in a double Butcher tableau as (4).

Then a semi-implicit Runge–Kutta method is implemented as follows. First we set $z^n = y^n$ and compute the stage fluxes for $i = 1, \dots, s$, we set $Y_1 = \tilde{Z}_1 = y^n$ and

$$\begin{cases} Y_i = y^n + \Delta t \sum_{j=1}^{i-1} \hat{a}_{ij} k_j, & 2 \leq i \leq s, \\ \tilde{Z}_i = y^n + \Delta t \sum_{j=1}^{i-1} a_{ij} k_j, & 2 \leq i \leq s \\ k_i = \mathcal{H}\left(Y_i, \tilde{Z}_i + \Delta t a_{ii} k_i\right), & 1 \leq i \leq s, \end{cases} \tag{11}$$

and, finally update the numerical solution by

$$y^{n+1} = y^n + \Delta t \sum_{i=1}^s \hat{b}_i k_i, \tag{12}$$

and

$$z^{n+1} = y^n + \Delta t \sum_{i=1}^s b_i k_i. \tag{13}$$

We observe that because $\hat{b}_i = b_i$ for $i = 1, \dots, s$ then the numerical solutions are the same, i.e. if $z^0 = y^0$ then $z^n = y^n \forall n \geq 0$, therefore the duplication of the system is only apparent, since there is indeed only one set of numerical solution.

Remark 3.1 (Embedded methods). We can select a different vector of weights for the z variable, say the $\tilde{b} \neq b = \hat{b}$. Such a vector will provide a lower order approximation of the solution for z . Consider now a scheme in which, at the beginning of the time step, we set $z^n = y^n$. Such a scheme would produce the same sequence y^n as scheme (11, 12), and would allow an automatic time step control. This procedure is commonly used in the context of numerical methods for ODEs [21]. Note that using \tilde{b} for the y variable, advancing with the z variable and setting $y_n = z_n$ at the beginning of each time step would give *exactly the same scheme*. In practice one advances the numerical solution with the more accurate one and uses the other variable to estimate the error. We shall not implement any time step control in the present paper.

Remark 3.2 We note that this new approach includes Zhong’s method [11] for autonomous systems. The theory developed in [11] for additive semi-implicit Runge–Kutta methods can be extended in a straightforward manner to the semi-implicit Runge–Kutta methods, (11), (12) and (13). In fact, by setting $\mathcal{H}(y, y) = f(y) + g(y)$ we obtain by (11):

$$\begin{aligned}
 k_i &= \mathcal{H} \left(y^n + \sum_{j=1}^{j-1} \hat{a}_{ij} k_j, y^n + \sum_{j=1}^{j-1} a_{ij} k_j + a_{ii} k_i \right), \\
 &= f \left(y^n + \sum_{j=1}^{j-1} \hat{a}_{ij} k_j \right) + g \left(y^n + \sum_{j=1}^{j-1} a_{ij} k_j + a_{ii} k_i \right), \tag{14}
 \end{aligned}$$

for $i = 1, \dots, s$. In particular, for updating the numerical solution, Zhong’s methods use only one set of weights, i.e. b_i for all $i = 1, \dots, s$, and, in our case, this means that we have only one set of unknowns, namely $y^n = z^n$. From Eq. (12), we have for numerical solution

$$y^{n+1} = y^n + \sum_{i=1}^s b_i k_i, \tag{15}$$

therefore (14) and (15) represent exactly the additive semi-implicit Runge–Kutta methods proposed by Zhong, [11], applied in a more general context, and therefore they have to satisfy more order conditions than in Zhong’s case (see next section).

In the following we propose different types of semi-implicit Runge–Kutta methods and verify that the order conditions are the same as the ones satisfied by the explicit and implicit Runge–Kutta schemes.

3.2 Classification of IMEX Runge–Kutta Schemes

Next we give a classification of such schemes and recall the order conditions to obtain second and third order accuracy in time. Finally we list several second and third order IMEX R–K schemes presented in the literature, [6,7] that we will use for our semi-implicit framework (11)–(13).

IMEX Runge–Kutta schemes presented in the literature can be classified in three different types characterized by the structure of the matrix $A = (a_{ij})_{i,j=1}^s$ of the implicit scheme. Following [23], we will rely on the following notions [2,7,22].

Definition 3.3 An IMEX Runge–Kutta method is said to be of **type A** [7] if the matrix $A \in \mathbb{R}^{s \times s}$ is invertible. It is said to be of **type CK** [2] if the matrix $A \in \mathbb{R}^{s \times s}$ can be written in the form

$$A = \begin{pmatrix} 0 & 0 \\ a & \mathcal{A} \end{pmatrix},$$

in which the matrix $\mathcal{A} \in \mathbb{R}^{(s-1) \times (s-1)}$ invertible. Finally, it is said to be of **type ARS** [22] if it is a special case of the type CK with the vector $a = 0$.

Schemes of type CK are very attractive since they allow some simplifying assumptions, that make order conditions easier to treat, therefore permitting the construction of higher order IMEX Runge–Kutta schemes. On the other hand, schemes of type A are more amenable to a theoretical analysis, since the matrix A of the implicit scheme is invertible.

3.3 Order Conditions and Numerical Schemes

Runge–Kutta methods (11)–(13) are obtained from the semi-implicit schemes (6)–(7). Thus, the order conditions for (11)–(13) are a direct consequence of the classical order conditions computed for partitioned Runge–Kutta methods, [21, 29]. Here we recall some known results for IMEX R–K schemes presented in [2, 6, 7]. In particular the order conditions up to order 3 can be simplified if we set $\hat{b}_i = b_i$ for $i = 1, \dots, s$. Then using the previous notation for the explicit and implicit part and assuming (4) and (5), we have for a method of order 2

$$\sum_i b_i = 1, \quad \sum_i b_i c_i = 1/2, \quad \sum_i b_i \hat{c}_i = 1/2. \tag{16}$$

and for a method of order 3

$$\sum_i b_i c_i^2 = 1/3, \quad \sum_{i,j} b_i a_{ij} c_j = 1/6, \tag{17}$$

$$\sum_i b_i \hat{c}_i^2 = 1/3, \quad \sum_{i,j} b_i \hat{a}_{ij} \hat{c}_j = 1/6, \tag{18}$$

$$\sum_i b_i \hat{c}_i c_i = 1/3, \quad \sum_{i,j} b_i a_{ij} \hat{c}_j = 1/6, \quad \sum_{i,j} b_i \hat{a}_{ij} c_j = 1/6. \tag{19}$$

The general conditions in case $\hat{b} \neq b$ can be found in [7].

We recall some well know definitions that we shall adopt in the paper.

Definition 3.4 An implicit R–K method is called stiffly accurate if $b^T = e_s^T A$ with $e_s^T = (0, \dots, 0, 1)$, i.e. methods for which the numerical solution is identical to the last internal stage (see [9], p. 45).

Remark 3.5 This property is important for the L-stability of the implicit part of the method, i.e. an A-stable implicit method stiffly accurate is also L-stable, (see [9], p. 44–45).

Note that the coefficients of the implicit part of some IMEX RK schemes presented later in this paper satisfy the property given in the Definition 3.4.

Now, we first consider second order schemes with two stages that satisfy the set of order conditions (16). For practical reasons, in order to simplify the computations of the coefficients,

we consider *singly diagonally implicit Runge–Kutta* (SDIRK) schemes, [9], for the implicit part, i.e. $a_{ii} = \gamma$, for $i = 1, \dots, s$. The Butcher tableau takes then the following form

$$\begin{array}{c|cc} 0 & 0 & 0 & \gamma & \gamma & 0 \\ \hat{c} & \hat{c} & 0 & c & c - \gamma & \gamma \\ \hline & b_1 & b_2 & & b_1 & b_2 \end{array} \tag{20}$$

with the following coefficients:

$$b_1 = 1 - b_2, \quad \hat{c} = 1/(2b_2), \quad c = (1/2 - \gamma(1 - b_2))/b_2, \tag{21}$$

where $b_2 \neq 0$ and $\gamma \in \mathbb{R}$ free parameters. If we require that the implicit part of the scheme is *A*-stable, then we consider $\gamma \geq 1/4$, [9].

We list below the second order schemes that we shall use in the paper.

3.3.1 The Second Order Semi-implicit Runge–Kutta Scheme: H-SDIRK2(2,2,2)

A first example of scheme satisfying the second order conditions given in (16) is $b_2 = \gamma = 1/2$, which yields the following table

$$\begin{array}{c|cc} 0 & 0 & 0 & 1/2 & 1/2 & 0 \\ 1 & 1 & 0 & 1/2 & 0 & 1/2 \\ \hline & 1/2 & 1/2 & & 1/2 & 1/2 \end{array} \tag{22}$$

This scheme is the combination of Heun method (explicit part) and an *A*-stable second order singly diagonal implicit Runge–Kutta SDIRK method (implicit part), hence we call it H-SDIRK2(2,2,2).

3.3.2 The Stiffly Accurate Semi-implicit Runge–Kutta Scheme: LSDIRK2(2,2,2)

Another choice for the coefficients (21) is $b_2 = \gamma, c = 1$, where γ is chosen as the smallest root of the polynomial $\gamma^2 - 2\gamma + 1/2 = 0$, i.e. $\gamma = 1 - 1/\sqrt{2}$ and $\hat{c} = 1/(2\gamma)$. This gives

$$\begin{array}{c|cc} 0 & 0 & 0 & \gamma & \gamma & 0 \\ \hat{c} & \hat{c} & 0 & 1 & 1 - \gamma & \gamma \\ \hline & 1 - \gamma & \gamma & & 1 - \gamma & \gamma \end{array} \tag{23}$$

This scheme is the combination of Runge–Kutta method (explicit part) and an *L*-stable second order SDIRK method in the implicit part. Thus, we call it LSDIRK2(2,2,2).

3.3.3 The IMEX-SSP2(2,2,2) L-stable Scheme: H-LDIRKp(2,2,2)

We choose $b_2 = 1/2, \hat{c} = 1$, i.e. the corresponding Butcher tableau is given by

$$\begin{array}{c|cc} 0 & 0 & 0 & \gamma & \gamma & 0 \\ 1 & 1 & 0 & 1 - \gamma & 1 - 2\gamma & \gamma \\ \hline & 1/2 & 1/2 & & 1/2 & 1/2 \end{array} \tag{24}$$

This scheme, introduced in [6], is based on Heun method coupled with an *L*-stable SDIRK Runge–Kutta. We call it H-LDIRKp(2,2,2). The implicit part has order $p = 2$ with $\gamma = 1 - 1/\sqrt{2}$ and order $p = 3$ with $\gamma = (3 + 3\sqrt{6})/3$. The version for $p = 2$ also has the *strongly stability preserving* (SSP) property, see [6]. The choice of γ for $p = 3$ guarantees that the implicit part is a third-order DIRK scheme with the best dampening properties [21].

To conclude this subsection devoted to two stages second order schemes, let us give a second order scheme, which is not in the class of singly diagonally implicit Runge–Kutta schemes.

3.3.4 The Second Order Semi-implicit Runge–Kutta Scheme of Type CK: H-CN(2,2,2)

A variant of the previous scheme satisfying the second order conditions (16) is given by the following table

$$\begin{array}{c|cc} 0 & 0 & 0 \\ \hline 1 & 1 & 0 \\ \hline & 1/2 & 1/2 \end{array} \quad \begin{array}{c|cc} 0 & 0 & 0 \\ \hline 1 & 1/2 & 1/2 \\ \hline & 1/2 & 1/2 \end{array} \tag{25}$$

This scheme is the combination of Heun method (explicit part) and a trapezoidal rule Crank Nicolson (implicit part) which is A-stable second order implicit, hence we call it H-CN(2,2,2). It may be a natural choice when dealing with convection–diffusion equation, since the Heun method is an SSP explicit RK [30], and the trapezoidal rule (also known as Crank–Nicolson) is an A-stable method, widely used for diffusion problems.

Finally, we also propose a three stage, second order IMEX scheme.

3.3.5 The Stiffly Accurate IMEX-SSP2(3,3,2) L-stable Scheme: SSP-LDIRK2(3,3,2)

The last second order scheme is obtained by combining a three-stage, second-order SSP scheme with an L-stable, second-order DIRK scheme, and is denoted by SSP-LDIRK2(3,3,2). This scheme is given by

$$\begin{array}{c|cccc} 0 & 0 & 0 & 0 \\ \hline 1/2 & 1/2 & 0 & 0 \\ \hline 1 & 1/2 & 1/2 & 0 \\ \hline & 1/3 & 1/3 & 1/3 \end{array} \quad \begin{array}{c|ccc} 1/4 & 1/4 & 0 & 0 \\ \hline 1/4 & 0 & 1/4 & 0 \\ \hline 1 & 1/3 & 1/3 & 1/3 \\ \hline & 1/3 & 1/3 & 1/3 \end{array} \tag{26}$$

3.3.6 The IMEX-SSP3(4,3,3) L-stable Scheme: SSP-LDIRK3(4,3,3)

As third semi-implicit Runge–Kutta methods of the type (11)–(13), that satisfies the set of order conditions (17), (18) and (19), a possible choice is given by the IMEX-SSP3(4,3,3) L-stable scheme with $\hat{b}_i = b_i$ for $i = 1, \dots, s$, i.e.

$$\begin{array}{c|cccc} 0 & 0 & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & 0 & 0 \\ \hline 1 & 0 & 1 & 0 & 0 \\ \hline 1/2 & 0 & 1/4 & 1/4 & 0 \\ \hline & 0 & 1/6 & 1/6 & 2/3 \end{array} \quad \begin{array}{c|cccc} \alpha & \alpha & 0 & 0 & 0 \\ \hline 0 & -\alpha & \alpha & 0 & 0 \\ \hline 1 & 0 & 1 - \alpha & \alpha & 0 \\ \hline 1/2 & \beta & \eta & 1/2 - \beta - \eta - \alpha & \alpha \\ \hline & 0 & 1/6 & 1/6 & 2/3 \end{array} \tag{27}$$

with $\alpha = 0.24169426078821$, $\beta = \alpha/4$ and $\eta = 0.12915286960590$. We call this scheme SSP-LDIRK3(4,3,3). For this particular choice, let us observe that the number of evaluations of the right hand side \mathcal{H} is still reasonable since the coefficients $c_i = \hat{c}_i$ for $2 \leq i \leq 4$ and only c_1 differs from \hat{c}_1 .

We remark that schemes H-LDIRK2(2,2,2), SSP-LDIRK2(3,3,2), and SSP-LDIRK3(4,3,3) were introduced in the context of hyperbolic systems with stiff relaxation in [6].

Remark 3.6 Even though IMEX R–K methods can be considered as additive R–K methods ([2, 22, 31]), we note that the coefficients in the Zhong form (14), (15) are not the standard coefficients of an IMEX R–K method. It is possible to show, by inspection, that Eqs. (14), (15) can be written as an IMEX R–K method. Let

$$\hat{A} = \begin{pmatrix} 0 & 0 & & \\ \hat{a}_{21} & 0 & & \\ \vdots & \vdots & \ddots & \\ a_{s1} & \cdots & \cdots & a_{ss-1} \end{pmatrix}, \quad A = \begin{pmatrix} a_{11} & 0 & & \\ a_{21} & a_{22} & & \\ \vdots & \vdots & \ddots & \\ a_{s1} & \cdots & \cdots & a_{ss} \end{pmatrix}, \quad b = \begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_s \end{pmatrix}$$

be the coefficients of the scheme (14), (15), with s stages. Then we write explicitly

$$y^{n+1} = y^n + \sum_{i=1}^s b_i \left(f(\hat{Y}_i) + g(Y_i) \right) \tag{28}$$

with

$$\begin{cases} \hat{Y}_1 = y^n, \\ Y_1 = y^n + \Delta t a_{11} \left(f(\hat{Y}_1) + g(Y_1) \right), \\ \hat{Y}_2 = y^n + \Delta t \hat{a}_{21} \left(f(\hat{Y}_1) + g(Y_1) \right), \\ Y_2 = y^n + \Delta t a_{21} \left(f(\hat{Y}_1) + g(Y_1) \right) + \Delta t a_{22} \left(f(\hat{Y}_2) + g(Y_2) \right), \\ \dots \\ \hat{Y}_s = y^n + \Delta t \hat{a}_{s1} \left(f(\hat{Y}_1) + g(Y_1) \right) + \dots + \Delta t \hat{a}_{ss-1} \left(f(\hat{Y}_{s-1}) + g(Y_{s-1}) \right), \\ Y_s = y^n + \Delta t a_{s1} \left(f(\hat{Y}_1) + g(Y_1) \right) + \dots + \Delta t a_{ss} \left(f(\hat{Y}_{ss}) + g(Y_{ss}) \right). \end{cases} \tag{29}$$

and the Butcher tableau is

0	0
a_{11} 0	0 a_{11}
\hat{a}_{21} 0 0	0 \hat{a}_{21} 0
a_{21} 0 a_{22} 0	0 a_{21} 0 a_{22}
\vdots \vdots \vdots \vdots \ddots	\vdots \vdots \vdots \vdots \ddots
a_{s1} 0 a_{s2} 0 \cdots a_{ss} 0	0 a_{s1} 0 a_{s2} \cdots a_{ss}
b_1 0 b_2 0 \cdots b_s 0	0 b_1 0 b_2 \cdots b_s

Using the Kronecker product, this can be written in a more compact form as

$$\begin{array}{c|c} \mathbf{c} & \hat{\mathcal{A}} \\ \hline & \hat{\mathbf{b}}^T \end{array} \quad \begin{array}{c|c} \mathbf{c} & \mathcal{A} \\ \hline & \mathbf{b}^T \end{array}$$

with

$$\begin{aligned} \hat{\mathbf{b}}^T &= b^T \otimes (1, 0), \quad \mathbf{b}^T = b^T \otimes (0, 1) \\ \hat{\mathcal{A}} &= (\hat{A} \otimes \hat{B}_1) + (A \otimes \hat{B}_2), \quad \mathcal{A} = (\hat{A} \otimes B_1) + (A \otimes B_2), \\ \hat{B}_1 &= \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix}, \quad \hat{B}_2 = \begin{pmatrix} 0 & 0 \\ 1 & 0 \end{pmatrix}, \end{aligned}$$

and

$$B_1 = \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix}, \quad B_2 = \begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix},$$

i.e., a double Butcher tableau (4) of scheme (28-29) with $\hat{c} = c = \hat{A}e = Ae$, and $e = (1, \dots, 1)^T$ the unit vector of length $2s$. Note that this is an IMEX scheme with a double Butcher tableau as (4). From this, we can derive the set of order conditions for scheme of the form (14)–(15) in terms of matrix \hat{A} , A and vectors \hat{b} , b , from the order condition of partitioned Runge–Kutta schemes with the same value of c and $\hat{b} \neq b$. A similar analysis, with a slightly different notation, was already performed in [32]. Below we list these equations up to order $p = 3$,

$$\begin{aligned} \hat{b}^T e &= 1, \quad b^T e = 1, \quad (\text{first order}), \\ \hat{b}^T c &= 1/2, \quad b^T c = 1/2, \quad (\text{second order}), \\ \hat{b}^T c^2 &= 1/3, \quad b^T c^2 = 1/3, \quad (\text{third order}) \\ b^T Ac &= 1/6, \quad b^T \hat{A}c = 1/6, \quad \hat{b}^T Ac = 1/6, \quad \hat{b}^T \hat{A}c = 1/6. \end{aligned}$$

Explicit computation of these conditions give

$$\begin{aligned} \sum_i b_i &= 1, \quad \sum_i b_i \hat{c}_i = 1/2, \quad \sum_i b_i c_i = 1/2, \quad \sum_i b_i c_i^2 = 1/3, \quad \sum_i b_i \hat{c}_i^2 = 1/3, \\ \sum_i b_i a_{ij} c_j &= 1/6, \quad \sum_i b_i \hat{a}_{ij} \hat{c}_j = 1/6, \quad \sum_i b_i a_{ij} \hat{c}_j = 1/6, \quad \sum_i b_i \hat{a}_{ij} c_i = 1/6. \end{aligned}$$

which are a subset of the classical seven third order conditions (16–19) for IMEX schemes of order three with $\hat{b} = b$ and $\hat{c} \neq c$. The missing condition is the one corresponding to the mixed derivative $\partial_{yz}^2 \mathcal{H}(y, z)$, which of course is zero for $\mathcal{H}(y, z) = f(y) + g(z)$. We observe that the third order conditions are only five in the case of Zhong [11]. This discrepancy is clearly explained in the paper [32].

4 Applications

Semi-implicit schemes present a particular interest for nonlinear problems where stiff terms can be treated efficiently by a linear solver rather than by a nonlinear one. For instance, our semi-implicit method is well suited to semi-linear PDEs: for $p \in \mathbb{N}, t \geq 0$ and in $x \in \Omega \subset \mathbb{R}^n$, we consider the differential equation of order $p + 1$ by

$$\frac{\partial u}{\partial t} + F(u, \nabla u, \dots, \nabla^p u) + \sum_{|k|=p+1} A_{k_1, \dots, k_n} \frac{\partial^{p+1}}{\partial x_1^{k_1} \dots \partial x_n^{k_n}} u = 0.$$

Of course for this problem classical IMEX schemes are also adapted. However for quasi-linear problems where the function A depends on $u, \dots, \nabla^p u$, semi-implicit methods are particularly promising since only the term of order $p + 1$ can be treated implicitly. Tests 4 and 5 below belong to this category.

Furthermore, when the highest order term of order $p + 1$ is negligible, the term of order p can be eventually treated with a semi-implicit solver (convection-dominated diffusion problems, see Tests 2 and 3 below).

Another kind of applications concerns relaxation problems with nonlinear relaxation systems [33]

$$\begin{cases} \frac{\partial u}{\partial t} + \operatorname{div} v = 0 \\ \frac{\partial v}{\partial t} + A \nabla u = \frac{G(u, v)}{\varepsilon} (F(u) - v), \end{cases}$$

with $G(u, v) \leq \alpha$, with α a positive real constant. The nonlinear term $G(u, v)$ can be treated explicitly while the term $(F(u) - v)$ is treated implicitly without any iterative solvers.

In this section we present several numerical tests for nonlinear PDEs for reaction–diffusion systems and nonlinear convection–diffusion equation for which we verify the order of accuracy and stability issues with respect to the CFL condition. Then, we treat a nonlinear Fokker–Planck equation to investigate the long time behavior of the numerical solution obtained from (11)–(13). Finally we complete this section with numerical tests on Hele–Shaw flow and surface diffusion flow.

We monitor L_1 and L_∞ norms of the error, defined as:

$$\begin{cases} \varepsilon_\infty = \max_{0 \leq n \leq N_T} \max_{i,j} \|\omega_{i,j}^n - \omega(t^n, x_i, y_j)\|, \\ \varepsilon_1 = \max_{0 \leq n \leq N_T} \sum_{i,j} \Delta x \Delta y \|\omega_{i,j}^n - \omega(t^n, x_i, y_j)\|. \end{cases}$$

where $\omega(t, x, y)$ denotes the exact solution, while $\omega_{i,j}^n$ indicates its numerical approximation at (t^n, x_i, y_j) . For space discretization we will apply basic fourth order discretization with central finite difference for first derivative

$$\nabla_h \omega_i = \frac{-\omega_{i+2} + 8\omega_{i+1} - 8\omega_{i-1} + \omega_{i-2}}{12h}$$

where h is the space step, and for the second derivative is discretized using a fourth order central finite difference scheme as well

$$\nabla_h^2 \omega_i = \frac{-\omega_{i+2} + 16\omega_{i+1} - 30\omega_i + 16\omega_{i-1} - \omega_{i-2}}{12h^2}.$$

For all numerical tests below, linear systems have been solved by a simple iterative C++ solver, that is, a successive over-relaxation (SOR(ω)) method even if more efficient solvers may be applied. This approach allows a comparison between various semi-implicit schemes, but of course it is not suitable to compare the efficiency of implicit and fully explicit methods. The use of a more efficient solver, such as for example a multigrid, would be desirable, but this goes beyond the scope of the present paper. The runs are performed on 2.8 GHz Intel core i7 machine.

4.1 Test 1: Reaction–Diffusion Problem

We first consider a very simple reaction–diffusion system with nonlinear source for which there are explicit solutions.

To demonstrate the optimal accuracy of the semi-implicit method in various norms, we consider the reaction–diffusion system problem [34] together with periodic boundary conditions: $\omega = (\omega_1, \omega_2) : \mathbb{R}^+ \times (0, 2\pi)^2 \mapsto \mathbb{R}^2$

$$\begin{cases} \frac{\partial \omega_1}{\partial t} = \Delta \omega_1 - \alpha_1(t) \omega_1^2 + \frac{9}{2} \omega_1 + \omega_2 + f(t), & t \geq 0, \quad (x, y) \in (0, 2\pi)^2, \\ \frac{\partial \omega_2}{\partial t} = \Delta \omega_2 + \frac{7}{2} \omega_2, & t \geq 0, \quad (x, y) \in (0, 2\pi)^2, \end{cases} \tag{30}$$

with $\alpha(t) = 2e^{t/2}$ and $f(t) = -2e^{-t/2}$. The initial conditions are extracted from the exact solutions

$$\begin{cases} \omega_1(t, x, y) = e^{-t/2} (1 + \cos(x)), \\ \omega_2(t, x, y) = e^{-t/2} \cos(2x). \end{cases}$$

To apply our semi-implicit scheme (11)–(13) we rewrite this PDE in the form (G) with $u = (u_1, u_2)$ the component treated explicitly, $v = (v_1, v_2)$ the component treated implicitly and

$$\mathcal{H}(t, u, v) = \begin{pmatrix} \Delta v_1 - \alpha(t) u_1 v_1 + \frac{9u_1}{2} + v_2 + f(t) \\ \Delta v_2 + \frac{7v_2}{2} \end{pmatrix}.$$

Since the Δ operator induces some stiffness it is treated implicitly whereas reaction terms are treated according to the sign of the reaction term and are linearized in order to avoid the numerical solution of a fully nonlinear problem. Concerning the spatial discretization, we simply apply a fourth order central finite difference method to the Δ operator. A fourth order accurate scheme for spatial derivatives is applied in order to bring out the order of accuracy of the second and third order time discretization.

To estimate the order of accuracy of the schemes we compute a numerical approximation and refine the time step Δt according to the space step $\Delta x = \Delta y$ in such a way the CFL condition associated to the diffusion operator is violated, that is, we apply an hyperbolic CFL condition where we refine the time step and the space step simultaneously

$$\lambda = \frac{2\Delta t}{\Delta x},$$

with $\lambda = 1$.

Obviously, for a fully explicit scheme like the Runge–Kutta method, this condition would lead to some instabilities of the numerical solution since a parabolic CFL is necessary.

The semi-implicit schemes are expected to be stable even for large time step when the parabolic CFL condition is not satisfied.

Furthermore, the PDE system (30) is non autonomous since the source term depends on time, hence the two solutions given by the explicit coefficient $(\hat{b}_i, \hat{c}_i)_{1 \leq i \leq s}$ in (12) and by the implicit coefficient $(b_i, c_i)_{1 \leq i \leq s}$ in (13) are different and we want to explore numerically the two choices.

Absolute error in L^∞ norms at time $T = 2$ are shown in Fig. 3 for all the schemes presented in the previous section. As expected the order of accuracy is satisfied for all second and third order schemes. The results obtained with the third order SSP-DIRK3(4,3,3) scheme (27) are much more accurate than the others obtained from second order schemes. Moreover, let us also emphasize that comparing the amplitude of the L^∞ error norm obtained with the second order schemes, the SSP-LDIRK2(3,3,2) scheme (26) with three stages is much more accurate than the others. Among the second order schemes with two stages the H-CN scheme (25) is the most accurate for this test. However, when we take into account the CPU time (see

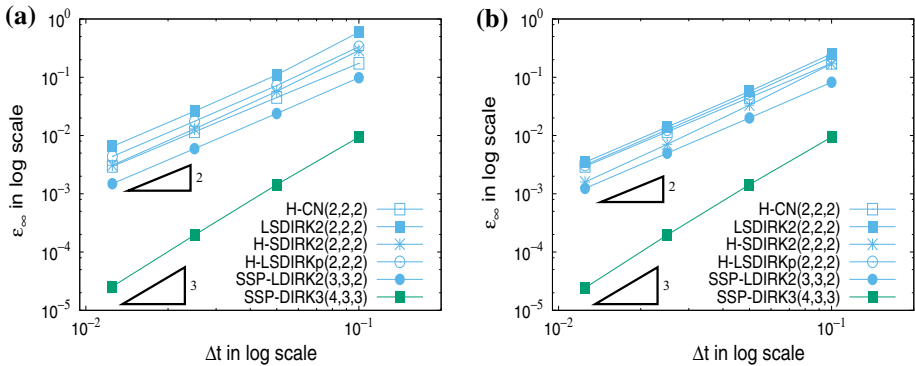


Fig. 3 Test 1: reaction–diffusion problem: L^∞ error norm when the last step is performed using the coefficients obtained from the (a) explicit scheme (12) and (b) and the implicit scheme (13) for the IMEX schemes described in the previous section. a y^{n+1} in (12). b z^{n+1} in (13)

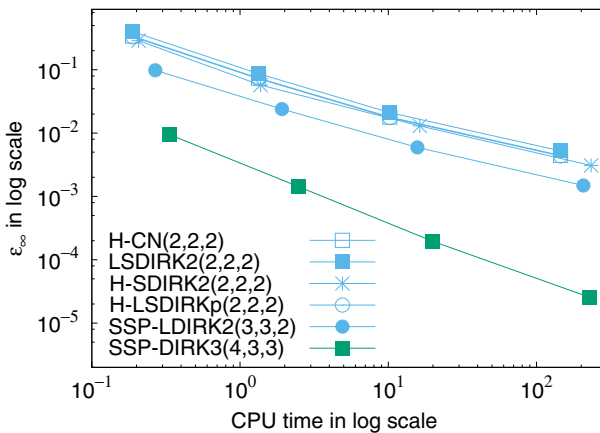


Fig. 4 Test 1: reaction–diffusion problem: L^∞ error norm with respect to the number of operations obtained from the IMEX schemes described in the previous section

Fig. 4), most of the second order schemes are equivalent, whereas the third order scheme is clearly advantageous for this test.

Finally, we only observe a slight difference between the two solutions y^{n+1} from (12) and z^{n+1} from (13), even if it seems that for this numerical test, the amplitude of the error is smaller when we use the solution given by the implicit stencil (13).

4.2 Test 2: Nonlinear Convection–Diffusion Equation

We consider the following nonlinear convection diffusion equation on the whole space \mathbb{R}^2 and apply a fourth order central finite difference scheme for the first and second spatial derivatives

$$\begin{cases} \frac{\partial \omega}{\partial t} + [V + \mu \nabla \log(\omega)] \cdot \nabla \omega - \mu \Delta \omega = 0, & (t, x) \in \mathbb{R}^+ \times \mathbb{R}^2, \\ \omega_0(t = 0) = e^{-\|x\|^2/2}, \end{cases}$$

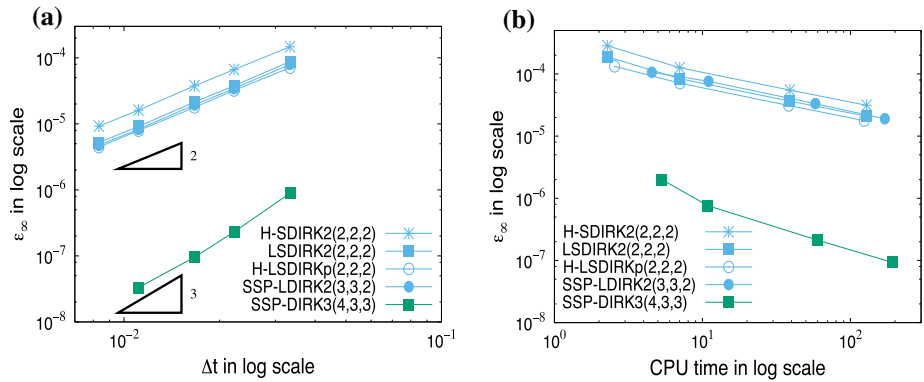


Fig. 5 Test 2: nonlinear convection–diffusion problem: **(a)** L^∞ error norm for the second order schemes (22), (23), (24), (25) and (26) and the third order SSP-DIRK3(4,3,3) L-stable scheme (27). **a** L^∞ error norm with respect to Δt . **b** L^∞ error norm with respect to the number of operations

where $V = {}^t(1, 1)$, $\mu = 0.5$. The exact solution is given by

$$\omega(t, x) = \frac{1}{\sqrt{4\mu t + 1}} e^{-\frac{\|x-Vt\|^2}{8\mu t + 2}}, \quad t \geq 0, \quad x \in \mathbb{R}^2.$$

After the space discretization, we apply our semi-implicit scheme (11)–(13) by writing the system of ODEs in the form (G) with u the component treated explicitly, v the component treated implicitly and

$$\mathcal{H}(t, u, v) = -(V + \mu \nabla \log(u)) \cdot \nabla v + \mu \Delta v.$$

We treat both the convection and diffusion implicitly (so that we do not have to worry in case the convection term dominates) but we only deal with a linear system at each time step. The computational domain in space is $(-10, 10)^2$ and the final time is $T = 0.5$. As in the previous case, the space step is chosen sufficiently small to neglect the influence of the space discretization and the time step Δt is taken proportional to Δx such that $\Delta t = \lambda \Delta x$, with $\lambda = 1$. Therefore, the classical CFL condition for convection diffusion problem $\Delta t = O(\Delta x^2)$ is not verified.

In Fig. 5 we present the numerical error for the L^∞ norm for the semi-implicit schemes described in the previous section and still verify the correct order of accuracy. We also present the error norm with respect to the CPU time. Once more, the third order SSP-DIRK3(4,3,3) scheme (27) with four stages gives much more accurate results than second order schemes with several order of magnitude.

4.3 Test 3: Nonlinear Fokker–Planck Equations for Fermions and Bosons

In [35,36], a nonlinear Fokker–Planck type equation modelling the relaxation of fermion and boson gases is studied. This equation has a linear diffusion and a nonlinear convection term:

$$\begin{cases} \frac{\partial \omega}{\partial t} = \operatorname{div}(x(1 + k\omega)\omega + \nabla \omega), & x \in \mathbb{R}^d, \quad t > 0, \\ \omega(x, 0) = \omega_0(x), \end{cases} \tag{31}$$

with $k = 1$ in the boson case and $k = -1$ in the fermion case. For this equation, the explicit solution is not known except steady states, but there are several works devoted to the long time behavior based on the knowledge of the qualitative behavior of the entropy functional. The long time behavior of this model has been rigorously investigated quite recently in [35] via an entropy-dissipation approach. More precisely, the stationary solution of (31) is given by the Fermi-Dirac ($k = -1$) and Bose-Einstein ($k = 1$) distributions:

$$\omega^{eq}(x) = \frac{1}{\beta e^{\frac{|x|^2}{2} - k}}, \tag{32}$$

where $\beta \geq 0$ is such that ω^{eq} has the same mass as the initial data ω_0 . For this equation, there exists an entropy functional given by

$$\mathcal{E}(t) := \int_{\mathbb{R}^d} \left(\frac{|x|^2}{2} \omega + \omega \log(\omega) - k(1 + k\omega) \log(1 + k\omega) \right) dx, \tag{33}$$

such that

$$\frac{d}{dt} \mathcal{E}_\omega(t) = -\mathcal{I}(t),$$

where the entropy dissipation $\mathcal{I}(t)$ is defined by

$$\mathcal{I}(t) := \int_{\mathbb{R}^d} \omega(1 + k\omega) \left| \nabla \left(\frac{|x|^2}{2} + \log \left(\frac{\omega}{1 + k\omega} \right) \right) \right|^2 dx. \tag{34}$$

Then decay rates towards equilibrium are given in [35, 36] for fermion case in any dimension and for 1D boson case by relating the entropy and its dissipation. Here we want to investigate the long time behavior of the numerical solution using different semi-implicit solvers with large time step and compare the numerical solution with the one obtained with an explicit method with a small time step [37].

To apply our semi-implicit scheme, we treat the linear diffusion term implicitly, and the non-linear convective term semi-implicitly. We rewrite this PDE in the form (G) with u the component treated explicitly, v the component treated implicitly and

$$\mathcal{H}(t, u, v) = \operatorname{div}(x(1 + ku)v + \nabla v) = \operatorname{div}(x(1 + ku)v) + \Delta v$$

and we apply a fourth order spatial discretization for the convective and diffusive components.

We consider the nonlinear Fokker-Planck Eq. (31) for fermions ($k = -1$) in 2D. The initial condition is chosen as

$$\omega_0(x) = \frac{1}{2\pi} |x|^2 \exp\left(-\frac{|x|^2}{2}\right), \quad x \in \mathbb{R}^2,$$

and the computational domain is $(-10, 10)^2$ with the space step $\Delta x = 0.1$.

Evolution of the discrete relative entropy $\mathcal{E}(t^n) = \mathcal{E}_\omega(t^n) - \mathcal{E}_{\omega^{eq}}$ and its dissipation $\mathcal{I}(t^n)$, given in (34), is presented in Fig. 6. This is obtained at the top by second order schemes, i.e. classical second order explicit Runge-Kutta scheme and H-LDIRKp(2,2,2) (24), and at the bottom by third order schemes, i.e. classical third order explicit Runge-Kutta scheme and SSP-DIRK3(4,3,3) (27).

We observe exponential decay rate of these quantities, which is in agreement with the result proved by Carrillo et al. [35] and the numerical results proposed in [37]. Classical Runge-Kutta schemes are subject to a parabolic condition whereas semi-implicit schemes can be used with a large time step without affecting the accuracy even for large time asymptotics.

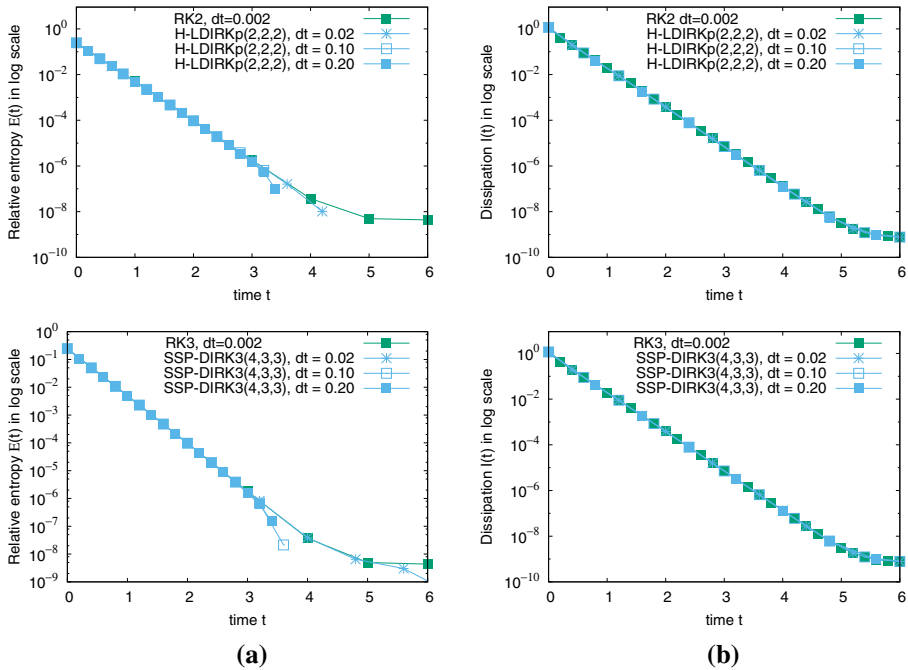


Fig. 6 Test 3: Fokker–Planck equation, **(a)** evolution of the relative entropy $\mathcal{E}_\Delta(t^n)$ and **(b)** the dissipation $\mathcal{I}_\Delta(t^n)$ for the second order explicit Runge–Kutta scheme and for the H-LDIRKp(2,2,2) scheme (24) (top) and for the third order explicit third order Runge–Kutta scheme and SSP-DIRK3(4,3,3) scheme (27) (bottom). **a** relative entropy functional $\mathcal{E}(t)$. **b** entropy dissipation $\mathcal{I}(t)$

4.4 Test 4: Hele–Shaw Flow

In this section we consider a fourth order nonlinear degenerate diffusion equation in one space dimension called the Hele–Shaw cell [38,39]

$$\frac{\partial \omega}{\partial t} + \frac{\partial}{\partial x} \left(\omega \frac{\partial^3 \omega}{\partial x^3} \right) = 0, \quad x \in \mathbb{R}, \quad t \geq 0, \tag{35}$$

with $\omega(x, t = 0) = \omega_0(x) \geq 0$. This example belongs to the first category of semi-linear equations recalled at the beginning of this section, and only the unknown appearing in the highest derivative is treated implicitly.

One of the remarkable features of Eq. (35) is that its nonlinearity guarantees the nonnegativity preserving property of the solution [40] and the conservation of mass

$$\int_{\mathbb{R}} \omega(t, x) dx = \int_{\mathbb{R}} \omega_0(x) dx.$$

Moreover there is dissipation of surface-tension energy, that is,

$$\frac{d}{dt} \int_{\mathbb{R}} \left| \frac{\partial \omega}{\partial x} \right|^2 dx = - \int_{\mathbb{R}} \omega \left| \frac{\partial^3 \omega}{\partial x^3} \right|^2 dx,$$

and dissipation of an entropy which highlights similarities with the Boltzmann equation

$$\frac{d}{dt} \int_{\mathbb{R}} \omega \log(\omega) dx = - \int_{\mathbb{R}} \left| \frac{\partial^2 \omega}{\partial x^2} \right|^2 dx.$$

On the one hand, we compare the numerical results obtained with our numerical approximation with the similarity property of monotonicity in time of solution

$$\omega(t, x) = \frac{1}{120(t + \tau)^{1/5}} \left[r^2 - \frac{x^2}{(t + \tau)^{2/5}} \right]_+^2,$$

where $[\cdot]_+$ denotes the positive part. We have chosen $r = 2$, $\tau = 4^{-5}$ and $x \in (-2, 2)$. This solution is only $\omega \in C^1(\mathbb{R} \times \mathbb{R})$ but the second derivative in space is discontinuous, therefore we cannot expect high order accuracy. Exact and numerical solutions at various times are reported in Fig. 8.

On the other hand, we consider the same problem with a given source term

$$f(\tau, x) = \frac{1}{8\tau^4} \exp\left(-\frac{x^2}{4\tau}\right) \left(2x^2 \tau^2 + (x^4 + 6\tau^2 - 9x^2\tau) \exp\left(-\frac{x^2}{4\tau}\right) \right),$$

with $\tau = t + 1$ such that the exact solution is smooth and given by $\omega_{exact}(t, x) = \exp(-x^2/4(t + 1))$.

For the time discretization we apply the scheme (27) by writing the system of ODEs in the form (G) with u the component treated explicitly and the v component treated implicitly:

$$\mathcal{H}(t, u, v) = -\frac{\partial}{\partial x} \left(u \frac{\partial^3 v}{\partial x^3} \right) + f(t + 1, x).$$

Concerning the space discretization, we apply a second order centred finite difference scheme for the space discretization

$$\mathcal{H}_{\Delta}(t, u_i, v_i) = -\frac{\mathcal{F}_{i+1/2} - \mathcal{F}_{i-1/2}}{\Delta x} + f(t + 1, x_i),$$

with

$$\mathcal{F}_{i+1/2} = u_{i+1/2} \frac{v_{i+2} - 3v_i + 3v_{i-1} - v_{i-2}}{\Delta x^3},$$

with $u_{i+1/2} = (u_i + u_{i+1})/2$. The time step is chosen as previously such that Δt is proportional to the space step Δx . In this way the stability condition associated to an hypothetical fully explicit time discretization for this problem, i.e. $\Delta t \leq C \Delta x^4$, is strongly violated.

The numerical error in L^1 and L^∞ for both test cases are reported in Fig. 7 at the final time $t = 0.35$. We observe a rate of convergence about 1.6 for both L^1 and L^∞ norms for the non smooth solution and second order accuracy for the smooth solution.

Of course for these large time steps, the numerical scheme does not preserve positivity, but only some small spurious oscillations occur for short times and then they are damped after several time iteration thanks to the diffusion process (see Fig. 8).

4.5 Test 5: Surface Diffusion Flow

In this section, we consider the surface diffusion of graphs [41]

$$\frac{\partial \omega}{\partial t} + \operatorname{div} S(\omega) = 0, \quad x \in \mathbb{R}^2, \quad t \geq 0,$$

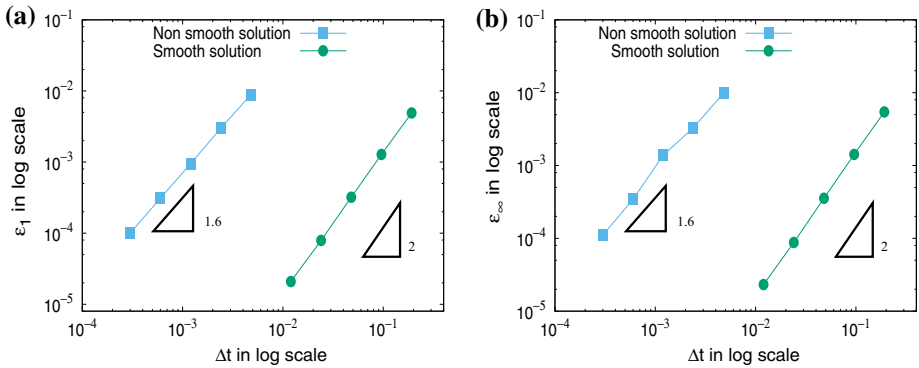


Fig. 7 Test 4: Hele–Shaw flow: **a** L^1 error norm and **b** L^∞ error norm for the SSP-LDIRK2(3,3,2) L-stable scheme (24)

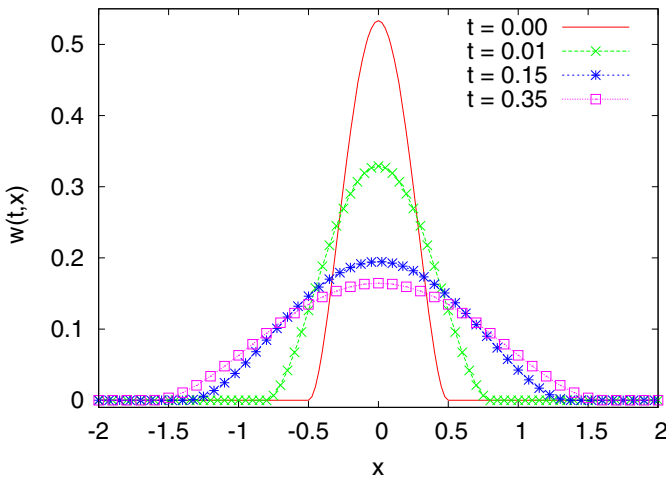


Fig. 8 Test 4: Hele–Shaw flow: time evolution of the numerical solution for the SSP-LDIRK3(4,3,3) L-stable scheme (27) for $t = 0, 0.01, 0.15$ and 0.35

where the nonlinear differential operator S is given by

$$S(\omega) := \left(Q(\omega) \left(I - \frac{\nabla\omega \otimes \nabla\omega}{Q^2(\omega)} \right) \nabla N(\omega) \right),$$

where Q is the area element

$$Q(\omega) = \sqrt{1 + |\nabla\omega|^2}$$

and N is the mean curvature of the domain boundary Γ

$$N(\omega) := \left(\frac{\nabla\omega}{Q(\omega)} \right). \tag{36}$$

The surface diffusion equation models the diffusion of mass within the bounding surface of a solid body, where $V = \Delta_\Gamma N(\omega)$ is the normal velocity of the evolving surface Γ ,

$$V = -\frac{1}{Q(u)} \frac{\partial u}{\partial t},$$

and Δ_Γ denotes the Laplace–Beltrami operator [41].

There are many applications of these models, such as body shape dynamics, surface construction, computer data processing or image processing. This equation is a highly nonlinear fourth-order PDE. The higher order differential operators and additional nonlinearities for these kind of problems are difficult to analyze and to simulate numerically due to the stiffness of order Δx^4 , where Δx is the space step [27,42]. We will apply our stable high order accurate methods based on semi-implicit time discretizations. Moreover, we will compare our time discretization with the one proposed by Smereka [14] or in [15], where the operator S is split in two parts

$$S(\omega) = \underbrace{S(\omega) - \beta \Delta^2 \omega}_{\text{less stiff part}} + \underbrace{\beta \Delta^2 \omega}_{\text{stiff, dissipative part}},$$

where β is a free parameter to be determined and in [14] it is chosen as $\beta = 2$. The first part is then treated explicitly whereas the stiff and dissipative part is treated implicitly. This splitting technique is very effective to stabilize numerical schemes but it may affect the numerical accuracy.

With our approach there is no need to add and subtract terms, because the system is automatically stabilized by the proper choice of the variable that will be implicitly treated, i.e. the numerator in the function $N(\omega)$, Eq. (36).

The solution of the surface diffusion of graphs verifies

$$\frac{1}{2} \frac{d}{dt} \int_\Omega \omega^2 dx + \int_\omega N^2(\omega) dx = 0,$$

giving L^2 stability.

We consider numerical solutions of the two-dimensional surface diffusion of graphs equation with the initial condition

$$\omega_0(x) = \frac{1}{2\pi T} \exp\left(-\frac{|x|^2}{2T}\right).$$

The computational domain is $(-10, 10)^2$ and we use a second order central finite difference scheme together with the second order H-LDIRKp(2,2,2) scheme (24) with

$$\mathcal{H}(u, v) := \left(Q(u) \left(I - \frac{\nabla u \otimes \nabla u}{Q^2(u)} \right) \nabla \mathcal{N}(u, v) \right),$$

and \mathcal{N}

$$\mathcal{N}(u, v) := \left(\frac{\nabla v}{Q(u)} \right).$$

We present in Fig. 9 the time evolution of the L^2 norm of the numerical solution and its dissipation:

$$\frac{d}{dt} \mathcal{E}(\omega) = -\mathcal{I}(t),$$

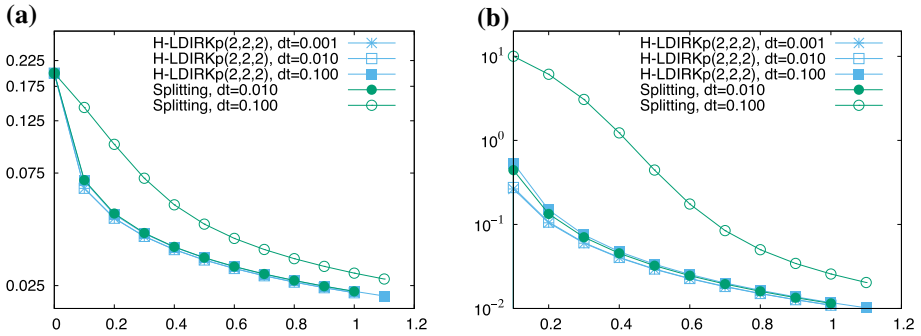


Fig. 9 Test 5: surface diffusion flow, (a) evolution of the L^2 norm and (b) the dissipation $\mathcal{I}_\Delta(t^n)$ for second order H-LDIRKp(2,2,2) L-stable scheme (24) and the one proposed in [14] based on a splitting technique in log scale. **a** functional $\mathcal{E}(t)$. **b** dissipation $\mathcal{I}(t)$

where the functional $\mathcal{E}(\omega)$ and the dissipation $\mathcal{I}(t)$ are defined by

$$\mathcal{E}(\omega) = \int_{\Omega} \omega^2(t, x) dx, \quad \mathcal{I}(t) = \int_{\Omega} N^2(\omega(t, x)) dx.$$

The results show that our second order numerical scheme (24) is stable and accurate for large time steps whereas the one based on the splitting technique given in [14] is stable but less accurate for large time step $\Delta t = 0.1$. These numerical simulations illustrate the efficiency of our approach based on semi-implicit numerical schemes.

5 Conclusions

In this paper we show that classical IMEX schemes can be adopted in a much wider context than the one they were originally introduced for. Indeed, in several contexts, the stiffness of the problem is essentially linear, and therefore IMEX schemes can be successfully adopted by treating the linear stiff part implicitly. Note that this approach is not equivalent to a linearization of the problem, and in several cases it is much simpler to apply than linearization. The overall accuracy of the scheme is automatically guaranteed by the standard order conditions for IMEX schemes. Many practical examples showing how to apply IMEX schemes in this new context are presented. We believe that this new approach can be successfully applied in many other contexts well beyond the ones presented in the paper.

Acknowledgments Francis Filbet is partially supported by the European Research Council ERC Starting Grant 2009, Project 239983-*NuSiKiMo* and the French ANR project STAB. Giovanni Russo and Sebastiano Boscarino have been partially supported by Italian PRIN 2009 project “Innovative numerical methods for hyperbolic problems with application to fluid dynamics, kinetic theory, and computational biology”, Prot. No. 2009588FHJ.

Appendix

Here we prove that it is possible to apply the IMEX schemes to the general system (8) with no doubling of the stage fluxes. We start observing that by choosing $\hat{y} = (t, u)$ and $z = u$, we can rewrite system (1) as

$$\begin{cases} \frac{d\hat{y}}{dt}(t) = \begin{pmatrix} 1 \\ \mathcal{H}(\hat{y}(t), z(t)) \end{pmatrix}, \\ \frac{dz}{dt}(t) = \mathcal{H}(\hat{y}(t), z(t)), \end{cases} \tag{37}$$

with initial conditions $\hat{y}(t_0) = (t_0, u_0), z(t_0) = u_0$.

In this way, system (37) for (\hat{y}, z) is a particular case of an autonomous partitioned system in which $\mathcal{F}_1 = (1, \mathcal{H})$ and $\mathcal{F}_2 = \mathcal{H}$ but apparently with an additional computational cost since we double the number of variables. Now, we apply the partitioned Runge–Kutta method (6)–(7) to (37) obtaining

$$\begin{cases} \hat{k}_i = \mathcal{H}(\hat{Y}_i, Z_i), & 1 \leq i \leq s, \\ \ell_i = \mathcal{H}(\hat{Y}_i, Z_i), & 1 \leq i \leq s, \end{cases}$$

with

$$\begin{cases} \hat{Y}_i = \hat{y}^n + \Delta t \sum_{j=1}^s \hat{a}_{i,j} \hat{k}_j, & 1 \leq i \leq s, \\ Z_i = z^n + \Delta t \sum_{j=1}^s a_{ij} \ell_j, & 1 \leq i \leq s, \end{cases}$$

Using the assumption (5), that is, $\sum_j \hat{a}_{i,j} = \hat{c}_i$, we obtain that the first component of \hat{Y} at the stage i is equal to $t^n + \hat{c}_i \Delta t$, and therefore, under the consistency condition $\sum_i \hat{b}_i = 1$, the first component of \hat{y} at time t^{n+1} is equal to time t^{n+1} .

Replacing \hat{y} by (t, y) , we have

$$\begin{cases} k_i = \mathcal{H}(t^n + \hat{c}_i \Delta t, Y_i, Z_i), & 1 \leq i \leq s, \\ \ell_i = \mathcal{H}(t^n + \hat{c}_i \Delta t, Y_i, Z_i), & 1 \leq i \leq s, \end{cases} \tag{38}$$

with

$$\begin{cases} Y_i = y^n + \Delta t \sum_{j=1}^s \hat{a}_{i,j} k_j, & 1 \leq i \leq s, \\ Z_i = z^n + \Delta t \sum_{j=1}^s a_{ij} \ell_j, & 1 \leq i \leq s, \end{cases}$$

Equation (38) already shows that $k_i = \ell_i, i = 1, \dots, s$. The numerical solutions at the next time step are

$$\begin{cases} y^{n+1} = y^n + \Delta t \sum_{i=1}^s \hat{b}_i k_i, \\ z^{n+1} = z^n + \Delta t \sum_{i=1}^s b_i k_i. \end{cases}$$

At this stage let us address several issues: number of evaluations, storage, order of accuracy and embedded methods.

Remark 5.1 Concerning the number of evaluations of \mathcal{H} , we observe that by writing (\mathcal{G}) as an autonomous partitioned system, we have $\ell_i = k_i$ for all $1 \leq i \leq s$. Therefore only one evaluation of \mathcal{H} is needed in (38), and only one set of stage fluxes is computed. Note that in (37), we could also choose $y = u$ and $\hat{z} = (t, u)$ and therefore use the $(c_i)_i$ coefficients for time stages.

References

1. Cafilisch, R.E., Jin, S., Russo, G.: Uniformly accurate schemes for hyperbolic systems with relaxation. *SIAM J. Numer. Anal.* **34**(1), 246–281 (2001)
2. Carpenter, M.H., Kennedy, C.A.: Additive Runge–Kutta schemes for convection–diffusion–reaction equations. *Appl. Numer. Math.* **44**, 139–181 (2003)
3. Chen, C.Q., Levermore, C.D., Liu, T.P.: Hyperbolic conservation laws with relaxation terms and entropy. *Commun. Pure Appl. Math.* **47**, 787–830 (1994)
4. Durran, D.R., Blossey, P.N.: Implicit–explicit multistep methods for fast-wave–slow-wave problems. *Mon. Wea. Rev.* **140**, 1307–1325 (2012)
5. Jin, S.: Runge–Kutta methods for hyperbolic conservation laws with stiff relaxation terms. *J. Comput. Phys.* **122**, 51–67 (1995)
6. Pareschi, L., Russo, G.: Implicit–explicit Runge–Kutta schemes for stiff systems of differential equations. In: Trigiante, D. (ed.) *Recent Trends in Numerical Analysis, Advances Theory Computational Mathematics*, vol. 3, pp. 269–288. Nova Sci. Publ., Huntington, NY (2001)
7. Pareschi, L., Russo, G.: Implicit–explicit Runge–Kutta schemes and applications to hyperbolic systems with relaxations. *J. Sci. Comput.* **25**, 129–155 (2005)
8. Weller, H., Lock, S.J., Wood, N.: Runge–Kutta IMEX schemes for the horizontally explicit/vertically implicit (HEVI) solution of wave equations. *J. Comput. Phys.* **252**, 365–381 (2013)
9. Hairer, E., Wanner, G.: *Solving ordinary differential equation. II. Stiff and differential algebraic problems*, Springer Series in Computational Mathematics, 14, Springer (second revised edition 1996), corrected second printing 2002
10. Higuera, I., Mantas, J.M., Roldán, T.: Design and implementation of predictors for additive semi-implicit Runge–Kutta methods. *SIAM J. Sci. Comput.* **31**(3), 2131–2150 (2009)
11. Zhong, X.: Additive semi-implicit Runge–Kutta methods for computing high-speed non equilibrium reactive flows. *J. Comput. Phys.* **128**, 19–31 (1996)
12. Boscarino, S., LeFloch, P.-G., Russo, G.: High-order asymptotic-preserving methods for fully non linear relaxation problems. *SIAM J. Sci. Comput.* **36**, A377–A395 (2014)
13. Berthon, C., LeFloch, P.-G., Turpault, R.: Late-time relaxation limits of nonlinear hyperbolic systems. A general framework. *Math. Comput.* **82**, 831–860 (2012)
14. Smreka, P.: Semi-implicit level set methods for curvature and surface diffusion motion. *J. Sci. Comput.* **19**(1–3), 439–456 (2003)
15. Filbet, F., Jin, S.: A class of asymptotic-preserving schemes for kinetic equations and related problems with stiff sources. *J. Comput. Phys.* **229**, 7625–7648 (2010)
16. Giraldo, F.X., Restelli, M., Läuter, M.: Semi-implicit formulations of the Navier–Stokes equations: application to nonhydrostatic atmospheric modeling. *SIAM J. Sci. Comput.* **32**(6), 3394–3425 (2010)
17. Giraldo, F.X., Kelly, J.F., Costantinescu, E.M.: Implicit–explicit formulations of a three-dimensional nonhydrostatic unified model of the atmosphere (NUMA). *SIAM J. Sci. Comput.* **35**(5), B1162–B1194 (2013)
18. Araújo, A., Barbeiro, S., Serranho, P.: Stability of finite difference schemes for complex diffusion processes. *SIAM J. Numer. Anal.* **50**, 1284–1296 (2012)
19. Boscarino, S., Russo, G.: High-order asymptotic-preserving methods for nonlinear relaxation from hyperbolic systems to convection–diffusion equations. In: *Submitted to Proceedings, High Order Nonlinear Numerical Methods for Evolutionary PDEs (HONOM 2013)*, March 18–22, Bordeaux (2013)
20. Boscarino, S., Bürger, R., Mulet, P., Russo, G., Villada, L.M.: Linearly implicit IMEX Runge–Kutta methods for a class of degenerate convection–diffusion problems. *SIAM J. Sci. Comput.* **37**, B305–B331 (2015)
21. Hairer, E., Norsett, S.P., Wanner, G.: *Solving Ordinary Differential Equation. I. Non-stiff Problems*, Springer Series in Computational Mathematics, vol. 8. Springer, Berlin (1993)
22. Ascher, U., Ruuth, S., Spiteri, R.J.: Implicit–explicit Runge–Kutta methods for time dependent partial differential equations. *Appl. Numer. Math.* **25**, 151–167 (1997)

23. Boscarino, S.: Error analysis of IMEX Runge–Kutta methods derived from differential algebraic systems. *SIAM J. Numer. Anal.* **45**, 1600–1621 (2007)
24. Boscarino, S.: On an accurate third order implicit–explicit Runge–Kutta method for stiff problems. *Appl. Numer. Math.* **59**, 1515–1528 (2009)
25. Boscarino, S., Russo, G.: Flux-explicit IMEX Runge–Kutta schemes for hyperbolic to parabolic relaxation problems. *SIAM J. Numer. Anal.* **51**, 163–190 (2013)
26. Hairer, E., Lubich, C., Roche, M.: Error of Runge–Kutta methods for stiff problems studied via differential algebraic equations. *BIT Numer. Math.* **28**(3), 678–700 (1988)
27. Filbet, F., Guo, R., Xu, Y.: Efficient high order semi-implicit time discretization and local discontinuous Galerkin methods for highly nonlinear PDEs. Preprint (2015)
28. Filbet, F., Rodrigues, L.M.: Asymptotically stable particle-in-cell methods for the Vlasov–Poisson system with a strong external magnetic field. Preprint (2015)
29. Hairer, E., Lubich, C., Wanner, G.: *Geometric Numerical Integration. Structure-Preserving Algorithms for Ordinary Differential Equations*, Springer Series in Computational Mathematics, vol. 31, 2nd edn. Springer, Berlin (2006)
30. Gottlieb, S., Shu, C.W., Tadmor, E.: Strong stability-preserving high-order time discretization methods. *SIAM Rev.* **43**, 89–112 (2001)
31. Boscarino, S., Russo, G.: On a class of uniformly accurate IMEX Runge–Kutta schemes and application to hyperbolic systems with relaxation. *SIAM J. Sci. Comput.* **31**, 1926–1945 (2009)
32. Higuera, I., Roldán, T.: Construction of additive semi-implicit Runge–Kutta methods with low-storage requirements. *J. Sci. Comput.* (2015). doi:[10.1007/s10915-015-0116-2](https://doi.org/10.1007/s10915-015-0116-2)
33. Filbet, F., Rambaud, A.: Analysis of an asymptotic preserving scheme for relaxation systems. *ESAIM Math. Model. Numer. Anal.* **47**, 609–633 (2013)
34. Zhang, K., Wong, J.C.F., Zhang, R.: Second-order implicit–explicit scheme for the Gray–Scott model. *J. Comput. Appl. Math.* **213**, 559–581 (2008)
35. Carrillo, J.A., Laurençot, Ph, Rosado, J.: Fermi–Dirac–Fokker–Planck equation: well-posedness and long-time asymptotics. *J. Differ. Equ.* **247**, 2209–2234 (2009)
36. Carrillo, J.A., Rosado, J., Salvarani, F.: 1D nonlinear Fokker–Planck equations for fermions and bosons. *Appl. Math. Lett.* **21**, 148–154 (2008)
37. Bessemoulin-Chatard, M., Filbet, F.: A finite volume scheme for nonlinear degenerate parabolic equations. *SIAM J. Sci. Comput.* **34**, 559–583 (2012)
38. Bertozzi, A.L.: The mathematics of moving contact lines in thin liquid films. *Not. AMS* **45**, 689–697 (1998)
39. Pareschi, L., Russo, G., Toscani, G.: A kinetic approximation to Hele–Shaw flow. *C. R. Math., Acad. Sci. Paris, Ser. I* **338**(2), 178–182 (2004)
40. Bertozzi, A.L., Pugh, M.: The lubrication approximation for thin viscous films: regularity and long-time behavior of weak solutions. *Commun. Pure Appl. Math.* **XLIX**, 85–123 (1996)
41. Deckelnick, K., Dziuk, G., Elliott, C.M.: Computation of geometric partial differential equations and mean curvature flow. *Acta Numer.* **14**, 139–232 (2005)
42. Xu, Y., Shu, C.W.: Local discontinuous Galerkin method for surface diffusion and Willmore flow of graphs. *J. Sci. Comput.* **40**, 375–390 (2009)