DESIGN AND IMPLEMENTATION OF PREDICTORS FOR ADDITIVE SEMI-IMPLICIT RUNGE-KUTTA METHODS

INMACULADA HIGUERAS*, JOSÉ MIGUEL MANTAS†, AND TEO ROLDÁN‡

Abstract. Space discretization of some time-dependent partial differential equations gives rise to stiff systems of ordinary differential equations. In this case, implicit methods should be used and therefore, in general, nonlinear systems must be solved. The solutions to these systems are approximated by iterative schemes and, in order to obtain an efficient code, good initializers should be used. Recently, a parallel code based on some Runge-Kutta and additive Runge-Kutta methods has been constructed, focusing especially on additive semi-implicit Runge-Kutta (ASIRK) methods. The aim of the present paper is to develop efficient initializers for these methods.

Key words. Stage value predictors, Additive Runge-Kutta methods, IMEX Runge-Kutta methods, ASIRK methods.

AMS subject classifications. 65L06, 65L05, 34-04, 65Y20, 65H10.

1. Introduction.

Space discretization of some time-dependent PDEs [11, 12, 14, 21, 22, 27, 29, 30] gives rise to stiff systems of ordinary differential equations in additive form

$$y' = f(y) + g(y)\,, \qquad y(t_0) = y_0\,, \tag{1.1}$$

where $f, g : \mathbb{R}^k \to \mathbb{R}^k$ are sufficiently smooth functions with different stiffness properties. In these cases, implicit methods should be used and therefore, in general, nonlinear systems must be solved.

For system (1.1), the numerical approximation at $t_{n+1} = t_n + h$ with an additive Runge-Kutta method with coefficients $(\mathcal{A}, \hat{\mathcal{A}}, b^t, \hat{b}^t)$, is given by

$$y_{n+1} = y_n + h\,(b^t \otimes I_k)\,F(Y_{n+1}) + h\,(\hat{b}^t \otimes I_k)\,G(Y_{n+1})\,, \tag{1.2}$$

where y_n is the previously computed solution at $t = t_n$, and Y_{n+1} is the vector of internal stages, $Y_{n+1} = (Y^t_{n+1,1}, \dots, Y^t_{n+1,s})^t \in \mathbb{R}^{sk}$, obtained from

$$Y_{n+1} = e \otimes y_n + h\,(\mathcal{A} \otimes I_k)\,F(Y_{n+1}) + h\,(\hat{\mathcal{A}} \otimes I_k)\,G(Y_{n+1})\,. \tag{1.3}$$

As usual, $e = (1, \dots, 1)^t \in \mathbb{R}^s$, $F(Y_{n+1}) = (f(Y_{n+1,1})^t, \dots, f(Y_{n+1,s})^t)^t$, $G(Y_{n+1}) = (g(Y_{n+1,1})^t, \dots, g(Y_{n+1,s})^t)^t$ and the symbol $\otimes$ denotes the Kronecker product. Following the Runge-Kutta notation, we will establish that $c = \mathcal{A}\,e$ and $\hat{c} = \hat{\mathcal{A}}\,e$.

Therefore, if the additive Runge-Kutta scheme is implicit and we are dealing with nonlinear problems, to compute y_{n+1} from (1.2) the nonlinear system (1.3) must be solved. In the context where additive ODEs (1.1) arise, nonlinear systems (1.3) are usually solved with an iterative method which requires a predictor that must be as accurate as possible in order to ensure a secure and fast convergence to the desired solution. These kind of predictors, also known as stage value predictors or starting

*Departamento de Ingeniería Matemática e Informática, Universidad Pública de Navarra. 31006 Pamplona, Spain. (higueras@unavarra.es).

†Departamento de Lenguajes y Sistemas Informáticos, Universidad de Granada. 18071 Granada, Spain. (jmmantas@ugr.es).

‡Departamento de Ingeniería Matemática e Informática, Universidad Pública de Navarra. 31006 Pamplona, Spain. (teo@unavarra.es).

algorithms, have been studied for different problems in [7, 13, 24, 26]. The numerical experiments done in these papers show that the use of high order predictors decreases the number of iterations and therefore the computational cost of the numerical integration is reduced. Furthermore, in some situations, convergence problems can be avoided with the use of efficient predictors.

Many techniques have been developed aiming at decreasing the overall computational cost of the numerical integration. For high dimension nonlinear problems, a slight improvement in the numerical solver (the computational implementation of which is straightforward) can give rise to a large saving in the computational cost. A fairly sophisticated technique recently developed in [16, 18, 19] consists in exploiting the parallelism in Runge-Kutta schemes (more precisely, RadauIIA method [6]) and additive Runge-Kutta methods (namely, the LRR(3,2,2) scheme and additive semi-implicit Runge-Kutta (ASIRK) methods studied e.g. in [15, 20] and [30] respectively). The aim of this paper is to design and implement predictors for the additive Runge-Kutta schemes considered in [18, 19], and show their relevance to decrease the overall computational time.

The rest of the paper is organized as follows. Section 2 is devoted to stage value predictors for additive Runge-Kutta methods; predictors for this kind of methods are reviewed and a direct application of the results from [7] allows us to deal with the LRR(3,2,2) method. In Section 3 we consider ASIRK methods; with some transformations, the material in [7] gives us predictors for these schemes. Finally, in Section 4, we present and illustrate some numerical experiments that show the efficiency of the new initializers constructed. The paper ends with some conclusions.

2. Predictors for additive Runge-Kutta methods.

In this section we consider general additive Runge-Kutta methods (1.2)-(1.3) such that at least one of the Runge-Kutta schemes is implicit. In this case, if we are dealing with nonlinear problems, the nonlinear system (1.3) must be solved. When this system is solved with an iterative scheme, a predictor for the internal stages, denoted on the following by $Y_{n+1}^{(0)}$, is needed.

In [7] initializers for additive and partitioned Runge-Kutta methods are considered and a detailed study is performed. The approach followed in that paper is an extension of the one done in [13, 24, 26] for Runge-Kutta methods. The numerical experiments in [7] show that a good predictor for the implicit method does not necessarily mean that it is a good one for the additive Runge-Kutta scheme. In order to construct efficient stage value predictors for additive Runge-Kutta methods, we must take into account both Runge-Kutta methods involved in the additive scheme.

In this paper, we assume that we already have a given step from t_{n-1} to $t_n = t_{n-1} + h_{\text{old}}$, and therefore y_{n-1}, y_n and Y_n are known values, and we are about to give a new step from t_n to $t_{n+1} = t_n + h$, with stepsize $h = r\, h_{\text{old}}$. Proceeding in this way, r is the stepsize ratio and we are dealing with the most general case of variable stepsize. In this context, we consider stage value predictors of the form

$$Y_{n+1}^{(0)} = b_0 \otimes y_{n-1} + (B \otimes I_k) Y_n \,, \tag{2.1}$$

where the vector b_0 and the matrix B are determined by imposing order conditions. Observe that predictor (2.1) has no additional cost because the vectors involved, namely y_{n-1} and Y_n, are already known from the previous time step.

The order ν of a predictor is defined as the largest integer which satisfies

$$\| Y_{n+1} - Y_{n+1}^{(0)} \| = \mathcal{O}(h^{\nu+1}) \,.$$

A detailed study, including general order conditions, can be seen in [7]. In particular, for additive RK schemes with coefficients $(\mathcal{A}, \hat{\mathcal{A}}, b^t, \hat{b}^t)$, it is proved that the stage value predictor (2.1) achieves order 2 if the following set of order conditions are fulfilled:

$$\begin{aligned}
\text{Consistency:}\quad & b_0 + B\,e = e\,,\\
\text{Order 1:}\quad & B\,c = e + r\,c\,,\\
& B\,\hat{c} = e + r\,\hat{c}\,,\\
\text{Order 2:}\quad & B\,\mathcal{A}\,c = eb^t c + r\mathcal{A}(e + rc)\,,\\
& B\,\mathcal{A}\,\hat{c} = eb^t\hat{c} + r\mathcal{A}(e + r\hat{c})\,,\\
& B\,\hat{\mathcal{A}}\,c = e\hat{b}^t c + r\hat{\mathcal{A}}(e + rc)\,,\\
& B\,\hat{\mathcal{A}}\,\hat{c} = e\hat{b}^t c + r\hat{\mathcal{A}}(e + r\hat{c})\,.
\end{aligned}$$

Observe that some order conditions involve the coefficients of both methods; consequently, together with the order conditions for each method, some coupling conditions are required. Unfortunately, the set of order conditions increases considerably with the order; for example 14 additional order conditions are required to obtain order 3. Therefore, in the general case, $c \neq \hat{c}$, it will be difficult to obtain high order stage value predictors.

However, if $c = \hat{c}$, many order conditions are equivalent and therefore, for those methods, it would be easier to construct high order stage value predictors. In this case, the set of order conditions is reduced to

$$\begin{aligned}
\text{Consistency:}\quad & b_0 + B\,e = e\,,\\
\text{Order 1:}\quad & B\,c = e + r\,c\,,\\
\text{Order 2:}\quad & B\,\mathcal{A}\,c = eb^t c + r\mathcal{A}(e + rc)\,,\\
& B\,\hat{\mathcal{A}}\,c = e\hat{b}^t c + r\hat{\mathcal{A}}(e + rc)\,,\\
\text{Order 3:}\quad & B\,\mathcal{A}\,c^2 = eb^t c^2 + r\mathcal{A}(e + rc)^2\,,\\
& B\,\mathcal{A}^2\,c = eb^t\mathcal{A}c + r\mathcal{A}eb^t c + r^2\mathcal{A}^2(e + rc)\,,\\
& B\,\mathcal{A}\hat{\mathcal{A}}\,c = eb^t\hat{\mathcal{A}}c + r\mathcal{A}e\hat{b}^t c + r^2\mathcal{A}\hat{\mathcal{A}}(e + rc)\,,\\
& B\,\hat{\mathcal{A}}\,c^2 = e\hat{b}^t c^2 + r\hat{\mathcal{A}}(e + rc)^2\,,\\
& B\,\hat{\mathcal{A}}\mathcal{A}\,c = e\hat{b}^t\mathcal{A}c + r\hat{\mathcal{A}}eb^t c + r^2\hat{\mathcal{A}}\mathcal{A}(e + rc)\,,\\
& B\,\hat{\mathcal{A}}^2\,c = e\hat{b}^t\hat{\mathcal{A}}c + r\hat{\mathcal{A}}e\hat{b}^t c + r^2\hat{\mathcal{A}}^2(e + rc)
\end{aligned} \tag{2.2}$$

where, as usual, given a vector $u \in \mathbb{R}^s$, the vector $u^k \in \mathbb{R}^s$ denotes the componentwise k-th power of vector u.

The theory developed in [7] is valid for additive Runge-Kutta methods and, in particular, for schemes $(\mathcal{A}, \hat{\mathcal{A}}, b^t, \hat{b}^t)$ containing an implicit and an explicit method. These approaches, known in the literature as IMplicit-EXplicit (IMEX) Runge-Kutta methods are used for additive ODEs of the form (1.1) where the function f contains the non-stiff part of the problem and the function g the stiff one. In this situation, an explicit method $(\mathcal{A}, b^t)$ can be used for f whereas an implicit method $(\hat{\mathcal{A}}, \hat{b}^t)$ is required for g [1, 11, 12, 20, 21, 22, 27].

As for Runge-Kutta methods, in many codes [19], a standard choice for stage value predictors is the most recent numerical approximation y_n, i.e.,

$$Y_{n+1}^{(0)} = e \otimes y_n \, . \tag{2.3}$$

It is straightforward to confirm that this predictor does not reach order 1 and therefore it is not a good predictor. However, better stage value predictors are possible. In the following example we show how to construct them for a method described in the literature.

Example 1. *In [20] an IMEX Runge-Kutta scheme, known as LRR(3,2,2), is studied. Its coefficients are given by*

$$\begin{array}{c|cccc} 0 & 0 & 0 & 0 & 0 \\ 1/2 & 1/2 & 0 & 0 & 0 \\ 1/3 & 1/3 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 \\ \hline & 0 & 1 & 0 & 0 \end{array} \qquad \begin{array}{c|cccc} 0 & 0 & 0 & 0 & 0 \\ 1/2 & 0 & 1/2 & 0 & 0 \\ 1/3 & 0 & 0 & 1/3 & 0 \\ 1 & 0 & 0 & 3/4 & 1/4 \\ \hline & 0 & 0 & 3/4 & 1/4 \end{array} \tag{2.4}$$

Observe that for this method, $c = \hat{c}$, and hence, up to order 3, the set of order conditions is simplified to (2.2). A simple analysis shows that it is not possible to obtain a third order predictor and that the best one consists of a family of order 2. In this case, the order conditions (2.2), up to order 2, may be written as

$$B\big[e \mid c \mid \mathcal{A}c \mid \hat{\mathcal{A}}c\big] = \big[e - b_0 \mid e + rc \mid eb^t c + r\mathcal{A}(e + rc) \mid e\hat{b}^t c + r\hat{\mathcal{A}}(e + rc)\big] \, .$$

For method (2.4) *with the above equation we obtain a family of predictors of order 2 given by*

$$B = \begin{pmatrix} -b_{01} & 0 & 0 & 1 \\ \frac{1}{2}(3r(r+1) - 2b_{02}) & r(3r+2) & -\frac{9}{2}r(r+1) & r+1 \\ \frac{2r^2}{3} + r - b_{03} & \frac{4}{3}r(r+1) & -r(2r+3) & \frac{2r}{3} + 1 \\ r(4r+3) - b_{04} & 4r(r+1) & -9r(r+1) & (r+1)^2 \end{pmatrix} .$$

In this family, the case $b_0 = 0$ corresponds to predictors obtained only from the internal stages, such as the ones used in [19],

$$Y_{n+1}^{(0)} = (B \otimes I_k) Y_n \, . \tag{2.5}$$

For this predictor the matrix B is given by

$$B = \begin{pmatrix} 0 & 0 & 0 & 1 \\ \frac{3}{2}r(r+1) & r(3r+2) & -\frac{9}{2}r(r+1) & r+1 \\ \frac{2r^2}{3} + r & \frac{4}{3}r(r+1) & -r(2r+3) & \frac{2r}{3} + 1 \\ r(4r+3) & 4r(r+1) & -9r(r+1) & (r+1)^2 \end{pmatrix} . \tag{2.6}$$

Observe that the first row in B gives that $Y_{n+1,1}^{(0)} = Y_{n,4} = y_n$. This agrees with the fact that the first stage is explicit and therefore it is not initialized.

3. Predictors for additive semi-implicit Runge Kutta methods. In this section we consider another class of additive methods studied in [30]. When applied to (1.1), an s-stage additive semi-implicit Runge-Kutta (ASIRK) method is given by

$$k_i = h\left(f(y_n + \sum_{j=1}^{i-1} b_{ij}k_j) + g(y_n + \sum_{j=1}^{i-1} c_{ij}k_j + a_i k_i)\right) \quad i = 1, \ldots, s\,, \tag{3.1}$$

$$y_{n+1} = y_n + \sum_{i=1}^{s} w_i k_i\,. \tag{3.2}$$

where a_i, b_{ij}, c_{ij} and w_j are the coefficients of the method. In [30], scheme (3.1)-(3.2) is called ASIRK-sA method to distinguish it from other versions of ASIRK schemes (ASIRK-sB and ASIRK-sC) based on Rosenbrock linearizations of Runge-Kutta schemes. Although ASIRK-sB and ASIRK-sC methods require less computational effort, stability requirements for some problems make ASIRK-sA methods necessary. Observe that in (3.1)-(3.2) the method is implicit in g and, consequently, if g is nonlinear a nonlinear system must be solved.

It is straightforward to see that an ASIRK method (3.1)-(3.2) can be formulated as

$$\begin{aligned} K_{n+1} &= h\,F(e \otimes y_n + (\mathcal{B} \otimes I_k)K_{n+1}) + h\,G(e \otimes y_n + (\mathcal{C} \otimes I_k)K_{n+1})\,, \\ y_{n+1} &= y_n + (w^t \otimes I_k)K_{n+1}\,, \end{aligned} \tag{3.3}$$

where F and G are defined in an analogous way to the functions F and G in (1.3). The matrices $\mathcal{B}$ and $\mathcal{C}$, and the vector w characterize the ASIRK method. Observe that matrix $\mathcal{B}$ is strictly lower triangular whereas matrix $\mathcal{C}$ is lower triangular. In this way, the nonstiff term f is treated explicitly whereas a diagonally implicit method is used for the stiff term g.

For example, in [30] the following 2-stage ASIRK-2A method

$$k_1 = h\left(f(y_n) + g(y_n + a_1 k_1)\right)\,, \tag{3.4}$$

$$k_2 = h\left(f(y_n + b_{21}k_1) + g(y_n + c_{21}k_1 + a_2 k_2)\right)\,, \tag{3.5}$$

$$y_{n+1} = y_n + w_1 k_1 + w_2 k_2\,. \tag{3.6}$$

is considered; for this scheme matrices $\mathcal{B}$ and $\mathcal{C}$ in (3.3) are given by

$$\mathcal{B} = \begin{pmatrix} 0 & 0 \\ b_{21} & 0 \end{pmatrix}\,, \qquad \mathcal{C} = \begin{pmatrix} a_1 & 0 \\ c_{21} & a_2 \end{pmatrix}\,.$$

This method, formulated as (3.3), is also considered in [19].

For solving the nonlinear system (3.3) with an iterative nonlinear solver, predictors for the internal derivatives K_{n+1} are required. From our experience with Runge-Kutta and additive Runge-Kutta schemes, we propose internal derivative predictors of the form

$$K_{n+1}^{(0)} = \mathfrak{b}_0 \otimes y_{n-1} + (\mathfrak{B} \otimes I_k)K_n\,. \tag{3.7}$$

The next step is to give conditions on vector $\mathfrak{b}_0$ and matrix $\mathfrak{B}$ in order to obtain a high order predictor. We could develop a theory similar to the one in [7] by means of Taylor expansions and rooted trees. However, instead of proceeding in this way, we will rewrite the ASIRK method as an additive Runge-Kutta scheme, in such a way that we can use the results obtained in [7].

As it has been pointed out above, formulation (3.3) corresponds to the use of internal derivatives K_{n+1}. We propose another formulation in terms of the internal stages Y_{n+1}, $\hat{Y}_{n+1}$:

$$Y_{n+1} = e \otimes y_n + (\mathcal{B} \otimes I_k)\left(h\,F(Y_{n+1}) + h\,G(\hat{Y}_{n+1})\right), \tag{3.8}$$

$$\hat{Y}_{n+1} = e \otimes y_n + (\mathcal{C} \otimes I_k)\left(h\,F(Y_{n+1}) + h\,G(\hat{Y}_{n+1})\right), \tag{3.9}$$

$$y_{n+1} = y_n + h\,(w^t \otimes I_k)\left(F(Y_{n+1}) + G(\hat{Y}_{n+1})\right). \tag{3.10}$$

This scheme is obtained from (3.3) by denoting $K_{n+1} = h\left(F(Y_{n+1}) + G(\hat{Y}_{n+1})\right)$. Formulation (3.8)-(3.10) corresponds to a $2s$-stage additive Runge-Kutta method (1.3) with block matrices:

$$\mathcal{A} = \begin{pmatrix} \mathcal{B} & 0 \\ \mathcal{C} & 0 \end{pmatrix}, \qquad \hat{\mathcal{A}} = \begin{pmatrix} 0 & \mathcal{B} \\ 0 & \mathcal{C} \end{pmatrix},$$

weight vectors $b^t = (w^t, 0)$, $\hat{b}^t = (0, w^t)$, and stage value vector $(Y_n, \hat{Y}_n)$. Recall that in this case, the $2s$-stage additive Runge-Kutta method satisfies $c = \hat{c}$. For example, the corresponding Butcher tableau for the ASIRK-2A method (3.4)-(3.6) written as a 4-stage additive Runge-Kutta method is

$$\begin{array}{c|cccc} 0 & 0 & 0 & 0 & 0 \\ b_{21} & b_{21} & 0 & 0 & 0 \\ a_1 & a_1 & 0 & 0 & 0 \\ c_{21}+a_2 & c_{21} & a_2 & 0 & 0 \\ \hline & w_1 & w_2 & 0 & 0 \end{array} \qquad \begin{array}{c|cccc} 0 & 0 & 0 & 0 & 0 \\ b_{21} & 0 & 0 & b_{21} & 0 \\ a_1 & 0 & 0 & a_1 & 0 \\ c_{21}+a_2 & 0 & 0 & c_{21} & a_2 \\ \hline & 0 & 0 & w_1 & w_2 \end{array}$$

Observe that, in general, with formulation (3.3) the nonlinear system to be solved has dimension $s \cdot k$, while in terms of the internal stages, the system (3.8)-(3.9) has dimension $2 \cdot s \cdot k$, and therefore from the dimension point of view, it is preferable to solve (3.3). However this is not a drawback because this transformation is simply a tool to use the known results.

Once we have an additive Runge-Kutta method, we can consider a stage value predictor of the form (2.1)

$$\begin{pmatrix} Y^{(0)}_{n+1} \\ \\ \hat{Y}^{(0)}_{n+1} \end{pmatrix} = \mathbb{b}_0 \otimes y_{n-1} + (\mathbb{B} \otimes I_k) \begin{pmatrix} Y_n \\ \\ \hat{Y}_n \end{pmatrix}, \tag{3.11}$$

where

$$\mathbb{b}_0 = \begin{pmatrix} b_0 \\ \hat{b}_0 \end{pmatrix} \qquad \text{and} \qquad \mathbb{B} = \begin{pmatrix} \mathbb{B}_{11} & \mathbb{B}_{12} \\ \mathbb{B}_{21} & \mathbb{B}_{22} \end{pmatrix}.$$

In particular, for the implicit internal stages $\hat{Y}^{(0)}_{n+1}$, we have the predictor

$$\hat{Y}^{(0)}_{n+1} = \hat{b}_0 \otimes y_{n-1} + (\mathbb{B}_{21} \otimes I_k)Y_n + (\mathbb{B}_{22} \otimes I_k)\hat{Y}_n\,. \tag{3.12}$$

Observe that although we have written a predictor for the vector $(Y_{n+1}, \hat{Y}_{n+1})$, the component Y_{n+1} is obtained in an explicit way (see (3.8)) and consequently we do not need to initialize it. Therefore we are only interested in conditions over the blocks $\mathbb{B}_{21}$ and $\mathbb{B}_{22}$.

Once we have expressed the ASIRK-sA method as an additive Runge-Kutta scheme, we can apply the theory developed in [7] summarized in Section 2. As $c = \hat{c}$, we can use (2.2) to obtain the conditions that $\mathbb{b}_0$ and $\mathbb{B}$ should satisfy to obtain a given order. If we use the blocks of matrix $\mathbb{B}$, then the conditions up to order 2 are given by:

$$\text{Consistency:}\quad \begin{pmatrix} b_0 \\ \hat{b}_0 \end{pmatrix} + \begin{pmatrix} \mathbb{B}_{11} & \mathbb{B}_{12} \\ \mathbb{B}_{21} & \mathbb{B}_{22} \end{pmatrix} e = e$$

$$\text{Order 1:}\quad \begin{pmatrix} \mathbb{B}_{11} & \mathbb{B}_{12} \\ \mathbb{B}_{21} & \mathbb{B}_{22} \end{pmatrix} \begin{pmatrix} \mathcal{B}e \\ \mathcal{C}e \end{pmatrix} = e + r \begin{pmatrix} \mathcal{B}e \\ \mathcal{C}e \end{pmatrix}$$

$$\text{Order 2:}\quad \begin{pmatrix} \mathbb{B}_{11} & \mathbb{B}_{12} \\ \mathbb{B}_{21} & \mathbb{B}_{22} \end{pmatrix} \begin{pmatrix} \mathcal{B} & 0 \\ \mathcal{C} & 0 \end{pmatrix} \begin{pmatrix} \mathcal{B}e \\ \mathcal{C}e \end{pmatrix} = eb^t \begin{pmatrix} \mathcal{B}e \\ \mathcal{C}e \end{pmatrix} + r \begin{pmatrix} \mathcal{B} & 0 \\ \mathcal{C} & 0 \end{pmatrix} \left(e + r \begin{pmatrix} \mathcal{B}e \\ \mathcal{C}e \end{pmatrix} \right)$$

$$\begin{pmatrix} \mathbb{B}_{11} & \mathbb{B}_{12} \\ \mathbb{B}_{21} & \mathbb{B}_{22} \end{pmatrix} \begin{pmatrix} 0 & \mathcal{B} \\ 0 & \mathcal{C} \end{pmatrix} \begin{pmatrix} \mathcal{B}e \\ \mathcal{C}e \end{pmatrix} = e\hat{b}^t \begin{pmatrix} \mathcal{B}e \\ \mathcal{C}e \end{pmatrix} + r \begin{pmatrix} 0 & \mathcal{B} \\ 0 & \mathcal{C} \end{pmatrix} \left(e + r \begin{pmatrix} \mathcal{B}e \\ \mathcal{C}e \end{pmatrix} \right)$$

From the second component, we easily obtain the set of order conditions for the internal stage predictor $\hat{Y}_{n+1}^{(0)}$,

$$\text{Consistency:}\quad \hat{b}_0 + \mathbb{B}_{21}e + \mathbb{B}_{22}e = e\,, \tag{3.13}$$

$$\text{Order 1:}\quad \mathbb{B}_{21}\mathcal{B}e + \mathbb{B}_{22}\mathcal{C}e = e + r\mathcal{C}e\,, \tag{3.14}$$

$$\text{Order 2:}\quad \mathbb{B}_{21}\mathcal{B}^2e + \mathbb{B}_{22}\mathcal{C}\mathcal{B}e = ew^t\mathcal{B}e + r\mathcal{C}(e + r\mathcal{B}e)\,, \tag{3.15}$$

$$\mathbb{B}_{21}\mathcal{B}\mathcal{C}e + \mathbb{B}_{22}\mathcal{C}^2e = ew^t\mathcal{C}e + r\mathcal{C}(e + r\mathcal{C}e)\,, \tag{3.16}$$

where we have used that for ASIRK-sA schemes $b^t = (w^t, 0)$ and $\hat{b}^t = (0, w^t)$.

Remember that our aim is the study of predictors of the form (3.7). The next step is to transfer the set of order conditions (3.13)-(3.16) for the stage value predictor $\hat{Y}_{n+1}^{(0)}$ in (3.12), to a set of order conditions for the internal derivative predictor $K_{n+1}^{(0)}$ in (3.7). To obtain this goal, we begin by studying the relationship between $\mathfrak{b}_0$, $\mathfrak{B}$ in (3.7), and $\mathbb{b}_0$, $\mathbb{B}$ in (3.11).

PROPOSITION 3.1. *We consider an internal derivative predictor of the form* (3.7) *for the ASIRK method* (3.3). *Then the coefficients of the vector* $\mathfrak{b}_0$ *and the matrix* $\mathfrak{B}$ *can be obtained in terms of the coefficients of the stage value predictor* (3.12) *as follows:*

$$\mathfrak{b}_0 = \mathcal{C}^{-1}\left(\hat{b}_0 + \mathbb{B}_{21}e + \mathbb{B}_{22}e - e\right), \quad \mathfrak{B} = \mathcal{C}^{-1}(\mathbb{B}_{21}\mathcal{B} + \mathbb{B}_{22}\mathcal{C} - ew^t)\,. \tag{3.17}$$

Proof. If we establish that $K_{n+1} = h\left(F(Y_{n+1}) + G(\hat{Y}_{n+1})\right)$, we can write (3.9) as

$$\hat{Y}_{n+1} = e \otimes y_n + (\mathcal{C} \otimes I_k)\, K_{n+1}\,,$$

and since the matrix $\mathcal{C}$ is regular, we have

$$K_{n+1} = (\mathcal{C}^{-1} \otimes I_k)\, (\hat{Y}_{n+1} - e \otimes y_n)\,.$$

Hence, given the stage value predictor $\hat{Y}^{(0)}_{n+1}$, we define the internal derivative predictor $K^{(0)}_{n+1}$ as

$$K^{(0)}_{n+1} = (\mathcal{C}^{-1} \otimes I_k)\,(\hat{Y}^{(0)}_{n+1} - e \otimes y_n)\,.$$

Equation (3.12) allows us to write

$$K^{(0)}_{n+1} = (\mathcal{C}^{-1} \otimes I_k)\,\big(\hat{b}_0 \otimes y_{n-1} + (\mathbb{B}_{21} \otimes I_k) Y_n + (\mathbb{B}_{22} \otimes I_k)\hat{Y}_n - e \otimes y_n\big)\,.$$

From equations (3.8)-(3.10), we replace Y_n, $\hat{Y}_n$ and y_n to obtain

$$\begin{aligned} K^{(0)}_{n+1} &= (\mathcal{C}^{-1} \otimes I_k)\,\big(\hat{b}_0 \otimes y_{n-1} + (\mathbb{B}_{21} \otimes I_k)(e \otimes y_{n-1} + (\mathcal{B} \otimes I_k)\,K_n) \\ &\qquad\qquad + (\mathbb{B}_{22} \otimes I_k)(e \otimes y_{n-1} + (\mathcal{C} \otimes I_k)\,K_n) \\ &\qquad\qquad - e \otimes (y_{n-1} + (w^t \otimes I_k)\,K_n)\big) \\ &= (\mathcal{C}^{-1} \otimes I_k)\,\big((\hat{b}_0 + \mathbb{B}_{21} e + \mathbb{B}_{22} e - e) \otimes y_{n-1} \\ &\qquad\qquad + ((\mathbb{B}_{21}\mathcal{B} + \mathbb{B}_{22}\mathcal{C} - e w^t) \otimes I_k)\,K_n\big)\,. \end{aligned}$$

Hence we get a predictor of the form (3.7) with $\mathfrak{b}_0$ and $\mathfrak{B}$ defined by (3.17).

■

We are in a position to give the set of order conditions for the internal derivative vector (3.7).

PROPOSITION 3.2. *The set of order conditions up to order 2 for predictor* (3.7) *are*

$$\begin{aligned} &\textit{Consistency:} && \mathfrak{b}_0 = 0\,, \\ &\textit{Order 1:} && \mathfrak{B}\, e = r\, e\,, && (3.18) \\ &\textit{Order 2:} && \mathfrak{B}\,\mathcal{B}\, e = r\,(e + r\mathcal{B}e)\,, && (3.19) \\ & && \mathfrak{B}\,\mathcal{C}\, e = r\,(e + r\mathcal{C}e)\,. && (3.20) \end{aligned}$$

Proof. From the consistency condition (3.13), the first equation in (3.17) remains $\mathfrak{b}_0 = 0$. Multiplying matrix $\mathfrak{B}$ in equation (3.17) by e, $\mathcal{B}e$ and $\mathcal{C}e$ and using (3.14), (3.15) and (3.16) respectively, we obtain the order conditions (3.18)-(3.20) for predictor (3.7).

■

The above result allows us to study and construct internal derivative predictors for ASIRK methods.

EXAMPLE 2. *We consider the 2-stage ASIRK-2A method (3.4)-(3.6) from [30]. An analysis of this scheme gives that order 2 is not possible because* (3.19)-(3.20) *cannot simultaneously be satisfied. If we impose the first order condition and the second order condition (3.19), we obtain the matrix* $\mathfrak{B}$ *given by*

$$\mathfrak{B} = r\,\big[e \mid e + r\mathcal{B}e\big]\,\big[e \mid \mathcal{B}e\big]^{-1} = r \begin{pmatrix} 1 - \frac{1}{b_{21}} & \frac{1}{b_{21}} \\ 1 - \frac{1}{b_{21}} - r & \frac{1}{b_{21}} + r \end{pmatrix}\,.$$

On the other hand, the first order condition and the second order condition (3.20) gives

$$\mathfrak{B} = r\left[e \mid e + r\mathcal{C}e\right]\left[e \mid \mathcal{C}e\right]^{-1} = \begin{pmatrix} -\frac{r(-a_2-c_{21}+a_1r+1)}{-a_1+a_2+c_{21}} & -\frac{(a_1(r-1)+1)r}{a_1-a_2-c_{21}} \\ \frac{(a_2(r-1)+c_{21}(r-1)+1)r}{a_1-a_2-c_{21}} & \frac{r(a_1-a_2r-c_{21}r-1)}{a_1-a_2-c_{21}} \end{pmatrix}.$$

In both cases, we obtain a first order internal derivative predictor. In particular, if we consider the set of parameters derived in [30] that give a second-order additive semi-implicit Runge-Kutta method:

$$w_1 = \frac{1}{2}, \qquad w_2 = \frac{1}{2}, \qquad b_{21} = 1, \qquad a_1 = \frac{1}{4}, \qquad a_2 = \frac{1}{3}, \qquad c_{21} = \frac{5}{12},$$

we get

$$\mathfrak{B} = \begin{pmatrix} 0 & r \\ -r^2 & r(1+r) \end{pmatrix} \quad \text{and} \quad \mathfrak{B} = \begin{pmatrix} -\frac{1}{2}r(r+1) & \frac{1}{2}r(r+3) \\ -\frac{1}{2}r(3r+1) & \frac{3}{2}r(r+1) \end{pmatrix}$$

respectively.

Example 3. *We study and construct predictors for the ASIRK-3A scheme constructed in [30] and considered in [18]. For this method, the matrices $\mathcal{B}$, $\mathcal{C}$, and the vector w are given by*

$$\mathcal{B} = \begin{pmatrix} 0 & 0 & 0 \\ b_{21} & 0 & 0 \\ b_{31} & b_{32} & 0 \end{pmatrix}, \quad \mathcal{C} = \begin{pmatrix} a_1 & 0 & 0 \\ c_{21} & a_2 & 0 \\ c_{31} & c_{32} & a_3 \end{pmatrix}, \quad w = \begin{pmatrix} w_1 \\ w_2 \\ w_3 \end{pmatrix},$$

where

$$b_{21} = \tfrac{8}{7}, \quad b_{31} = \tfrac{71}{252}, \quad b_{32} = \tfrac{7}{36},$$
$$c_{21} = 0.3067269871935408, \; c_{31} = 0.45, \quad c_{32} = -0.2631108321468882,$$
$$a_1 = 0.4855612330925677, \; a_2 = 0.9511295466999914, \; a_3 = 0.1892078709825326,$$
$$w_1 = \tfrac{1}{8}, \quad w_2 = \tfrac{1}{8}, \quad w_3 = \tfrac{3}{4}.$$

For this method it is possible to obtain an optimum predictor of order 2. According to (3.18)-(3.20), it is given by

$$\mathfrak{B} = r\left[e \mid e + r\mathcal{B}e \mid e + r\mathcal{C}e\right]\left[e \mid \mathcal{B}e \mid \mathcal{C}e\right]^{-1} = \begin{pmatrix} \mathfrak{b}_{11} & \mathfrak{b}_{12} & \mathfrak{b}_{13} \\ \mathfrak{b}_{21} & \mathfrak{b}_{22} & \mathfrak{b}_{23} \\ \mathfrak{b}_{31} & \mathfrak{b}_{32} & \mathfrak{b}_{33} \end{pmatrix}, \tag{3.21}$$

where

$$\begin{aligned}
\mathfrak{b}_{11} &= (0.6567918590808737r - 0.09320814091911545)r\,,\\
\mathfrak{b}_{12} &= (0.4691370422006241r + 0.719137042200632)r\,,\\
\mathfrak{b}_{13} &= (0.374071098718483 - 1.125928901281498r)r\,,\\
\mathfrak{b}_{21} &= (-0.34320814091912655r - 0.09320814091911545)r\,,\\
\mathfrak{b}_{22} &= (1.469137042200624r + 0.719137042200632)r\,,\\
\mathfrak{b}_{23} &= (0.374071098718483 - 1.125928901281498r)r\,,\\
\mathfrak{b}_{31} &= (-0.34320814091912644r - 0.09320814091911545)r\,,\\
\mathfrak{b}_{32} &= (0.469137042200624r + 0.719137042200632)r\,,\\
\mathfrak{b}_{33} &= (0.3740710987184832 - 0.1259289012814977r)r\,.
\end{aligned}$$

4. Numerical experiments. In this section we use the stage value predictors constructed above. Our aim is to study the influence of the stage value predictors on the computational efficiency when the LRR(3,2,2) and ASIRK-3A numerical schemes are implemented.

4.1. Implementation of the solvers. We have used efficient mixed Fortran and C implementations of both numerical schemes which use standard routines of the BLAS linear algebra libraries [2] and a C implementation of the restarted generalized minimum residual method (restarted GMRES) [25].

When the LRR(3,2,2) scheme (or the ASIRK-3A scheme) is applied to a system of the form (1.1), a k-dimensional nonlinear system arises for each Runge-Kutta stage and time step. These nonlinear systems are usually solved by the modified Newton method. The i-th Newton iterative process ($i = 1, \ldots, s$) of the n-th time step computes an approximation to the internal stage vector $Y_{n+1,i} \in \mathbb{R}^k$ (for the ASIRK-3A, $Y_{n+1,i}$ would be the internal derivative $K_{n+1,i}$) by performing $m_{n,i}$ loops ($m_{n,i}$ is determined using a particular stopping criterion). At the v-th loop of the Newton process, a linear system with the form

$$M_{n+1,i} \cdot \Delta Y^{(v-1)}_{n+1,i} = R^{(v-1)}_{n+1,i}, \qquad v = 1, \ldots, m_{n,i} \tag{4.1}$$

must be solved (see [18]), where the coefficient matrix $M_{n+1,i} \in \mathbb{R}^{k\times k}$ depends on an evaluation of the Jacobian of the function g (and exhibits the same structure) and $R^{(v-1)}_{n+1,i}$ is the residual vector of the v-th loop.

The coefficient matrix $M_{n+1,i}$ of (4.1) is usually sparse. Since iterative methods are particularly well-suited to deal with these types of coefficient matrices, the restarted generalized minimum residual iterative method (restarted GMRES) [25] with no preconditioning and 10 iterations before restarting (GMRES(10)) has been implemented. As the test problems considered lead to sparse banded coefficient matrices, we used an implementation of the GMRES(10) scheme which applies the banded matrix-vector product by using a compact storage for banded matrices.

The Jacobian evaluation is performed once for each Runge-Kutta stage and before the Newton iteration process while each Newton loop includes the iterative solution of system (4.1). To determine the end of the Newton iteration process, we used a strategy similar to that employed in [3, 4]. The user must provide the absolute and relative tolerances *atol* and *rtol*, and the stopping criterion for the Newton iteration keeps the local error of each component of the solution vector $y_{n+1}(l)$ $(l = 1, ..., k)$ below $atol{+}rtol{\cdot}y_{n+1}(l)$. This strategy, assuming there is not convergence problems, depends on the value of $\Delta Y_{n+1,i}^{(v-1)}$ obtained from solving the system (4.1) and is briefly shown in the Algorithm 1. This algorithm describes the computations performed at the vth loop of the i-th stage at the n-th integration step to compute the logical variable *convergence* (initially false) which determines the number of Newton iterations $m_{n,i}$.

In Algorithm 1, v is a global variable of the Newton solver which stores the number of iterations. Therefore, if *convergence* is equal to *true* after executing the algorithm, then v stores the total number of iterations $m_{n,i}$. The parameter *eps* is the machine precision (we used 10^{-16} in the experiments) and the norm $\|\cdot\|_{scal}$, at the n-th time step, is defined by

$$\|x\|_{scal} = \sqrt{\frac{1}{k}\sum_{l=1}^{k}\left(\frac{x(l)}{atol + rtol\cdot|y_n(l)|}\right)^2}, \qquad \text{for all } x \in \mathbb{R}^k.$$

The variable α of Algorithm 1 is also a global variable of the Newton process which stores an estimate of the convergence rate for the Newton scheme. At the v-th Newton iteration, with $v > 1$, this variable is computed from the norm of the vectors $\Delta Y_{n+1,i}^{(v-1)}$ and $\Delta Y_{n+1,i}^{(v-2)}$, obtained in the current and the previous loop of the Newton method, respectively (see [4, 6] for more details). This variable is used to compute the value of *convergence* in each Newton loop. When $v = 1$, a new value of *convergence* can be also obtained if α was modified in the previous Runge-Kutta stage (in other case, the value of *convergence* does not change and maintains the value $false$).

4.2. The test problems. The implementations of both schemes have been used to integrate two different test problems which are briefly described below.
Problem 1. The first problem we have considered is the *BRELAX* problem from [10, 17]. In this example we perform a time integration of a hydrodynamical model of the Boltzmann equation for rarefied gases in 1D . This model is based on a system of 5 PDEs which can be written in the following form,

$$\begin{pmatrix} U_t \\ V_t \end{pmatrix} = \begin{pmatrix} -F^1(U)_x - F^2(U,V)_x \\ -G(U,V,U_x,V_x) + D(U,V,U_x,V_x)_x \end{pmatrix}, \tag{4.2}$$

where

$$U = \begin{pmatrix} \rho \\ m = \rho u \\ z = \frac{1}{2}\rho u^2 + \frac{3}{2}p \end{pmatrix}, \quad V = \begin{pmatrix} \sigma \\ q \end{pmatrix}, \quad F^1(U) = \begin{pmatrix} \rho u \\ \rho u^2 + p \\ \frac{1}{2}\rho u^3 + \frac{5}{2}up \end{pmatrix},$$

$$F^2(U,V) = \begin{pmatrix} 0 \\ \sigma \\ \sigma u + q \end{pmatrix}, \quad D(U,V,U_x,V_x)_x = \begin{pmatrix} \dfrac{p\hat{\omega_4}}{\mu\omega_2}\left[\dfrac{\mu^3\theta}{\mathcal{M}R\rho^2}\left(\dfrac{\sigma}{\mu\theta}\right)_x\right]_x \\ \dfrac{p\hat{\theta_4}}{\mu\theta_2}\left[\dfrac{\mu^3\theta}{\rho^2}\left(\dfrac{q}{R\mu\theta^2}\right)_x\right]_x \end{pmatrix},$$

$$G(U,V,U_x,V_x) = \begin{pmatrix} u\sigma_x - \frac{4}{3}\sigma u_x + \frac{2p}{\omega_2\mu}\left(\sigma + \frac{4}{3}\mu u_x - \sigma_2 - \sigma_3\right) \\ uq_x - qu_x + \frac{3\mathcal{M}p}{2\theta_2\mu}\left(q + \frac{3}{2}\mu\mathcal{M}R\,\theta_x - q_2 - q_3\right) \end{pmatrix}.$$

Algorithm 1: Stopping Criterion for the modified Newton process, v-th loop for the ith stage of the n-th time step

```
Result: $convergence$, $\alpha$
if $\|\Delta Y_{n+1,i}^{(v-1)}\|_{scal} < 100 \cdot eps \cdot \|y_n\|_{scal}$ then $convergence = true$;
else
    if $(v = 1)$ and ($\alpha$ is available from a previous stage) then
        $convergence = \left(\|\Delta Y_{n+1,i}^{(v-1)}\|_{scal}\frac{\alpha}{1-\alpha} < 10^{-3}\right)$;  $\alpha = \sqrt{\alpha}$ ;
    else if $(v = 2)$ then
        $\alpha = \frac{\|\Delta Y_{n+1,i}^{(1)}\|_{scal}}{\|\Delta Y_{n+1,i}^{(0)}\|_{scal}}$;
        $convergence = \left(\|\Delta Y_{n+1,i}^{(v-1)}\|_{scal}\frac{\alpha}{1-\alpha} < 0.03\right)$;
    else if $(v > 2)$ then
        $\alpha = \sqrt{\alpha \cdot \frac{\|\Delta Y_{n+1,i}^{(v-1)}\|_{scal}}{\|\Delta Y_{n+1,i}^{(v-2)}\|_{scal}}}$;
        $convergence = \left(\|\Delta Y_{n+1,i}^{(v-1)}\|_{scal}\frac{\alpha}{1-\alpha} < 0.03\right)$;
end
if not convergence then $v = v + 1$;
```

were the terms σ_2, σ_3, q_2 and q_3 are defined in [17]. In this system, we have five independent variables ρ (mass density), $m = \rho u$ (momentum, where u is the macroscopic velocity), $z = \frac{1}{2}\rho u^2 + \frac{3}{2}p$ (total energy, where p is the normal pressure), σ (pressure deviation tensor) and q (heat flux vector). Here θ is the temperature ($\theta = \frac{p}{R\rho}$) and μ is the viscosity (here $\mu = \theta$). $\mathcal{M}$ denotes the Maxwell number (also called the Eucken factor) which is a constant characteristic of a given gas ($\mathcal{M} = k(\mu c_v)^{-1}$, where k is the heat conductivity of the gas and c_v is the specific heat at constant volume) and has an approximated value of 2.5 for most monatomic natural gases. R is the perfect gas constant ($R = 1$). The symbols ω_2, θ_2, $\hat{\omega}_4$ and $\hat{\theta}_4$ denote coefficients of the relaxed Burnett system (see [10, 17]) and have the following approximated values:

$$\omega_2 = 2, \quad \theta_2 = 45/8, \quad \hat{\omega}_4 = 20, \quad \hat{\theta}_4 = 25.$$

A spatial discretization of this system is proposed in [17]. The physical space is truncated to the finite region $[0, 250]$ and we have subdivided this interval in $N - 1$ subintervals of length $\Delta x = \frac{250}{N-1}$. The discretization is based on combining first order upwind relaxation schemes [9] for the conservative part $(F^1(U)_x)$ and standard second order central differences for the non conservative part $(F^2(U, V)_x$, $G(U, V, U_x, V_x)$ and $D(U, V, U_x, V_x)_x)$.

The resulting system has $5N$ ODEs when the 1D space is discretized by using N grid points. The stiff and nonstiff terms in (4.2) have the following form:

$$\mathbf{f}(U, V) = (-F^1(U)_x - F^2(U, V)_x, 0)^T, \quad \mathbf{g}(U, V) = (0, D(U, V, U_x, V_x)_x - G(U, V, U_x, V_x))^T.$$

If we maintain the same spatial discretization and arrangement which is proposed in [17], the subsystem associated to **g** has a banded structure and the Jacobian of **g** has 9 subdiagonals and 5 superdiagonals.

The initial vector $y_0 = y(0)$ of the differential system captures the state before a one-dimensional shock profile with Mach number 10. The gas is initially at the upstream equilibrium state in the left half-space and in the downstream equilibrium state in the right-half space. The two states are smoothly connected with an hyperbolic tangent function (see [17]).

We performed the time integration from $t = 0$ through $t = 10^{-3}$ by using 20 time steps with fixed size $h = 5 \cdot 10^{-5}$. We performed experiments for different values of the parameter N. For this problem, we used a relative error $rtol = 10^{-16}$ and the absolute error $atol = 10^{-16}$ in order to stop the Newton process.

Problem 2. We refer to this second problem as the *Diode* problem. This problem was introduced in [28] and describes the electron transport in a one-dimensional silicon $n^+ - n - n^+$ diode of length $L = 0.6\mu m$, where the interaction with the semiconductor crystal is taken into account by an effective relaxation-time operator. The underlying mathematical model is described in detail in [5] and can be written as:

$$\begin{aligned} \frac{\partial f}{\partial t} &= -F(x,v,t) + G(x,v,t), \\ F(x,v,t) &= v\frac{\partial f}{\partial x} - \frac{e}{m}E\frac{\partial f}{\partial v}, \quad G(x,v,t) = \frac{1}{\tau(x,t)}\left[M(v)\rho(x,t) - f\right], \end{aligned} \tag{4.3}$$

where $f = f(x,v,t)$ is the scaled probability density function, x the spatial variable ($x \in [0,L]$), t the time variable, v the velocity in phase space ($v \in [-3.5, 3.5]$), m the effective electron mass, e the electron charge unit, and

$$M(v) = \frac{1}{\sqrt{2\pi\theta}}e^{-v^2/2\theta}, \qquad \text{with } \theta = \frac{k_b T_0}{m},$$

being k_b the Boltzmann constant and T_0 the lattice temperature ($T_0 = 300K$).

The term $\rho(x,t)$ is the electron concentration which can be obtained as follows:

$$\rho(x,t) = \int_{-\infty}^{\infty} f(x,v,t)dv. \tag{4.4}$$

The term E is the electric field which can be obtained by solving the coupled Potential Equation:

$$E(x,t) = -\frac{\partial \phi}{\partial x}, \qquad \epsilon_0 \frac{\partial^2 \phi}{\partial x^2} = e(\rho(x,t) - C(x)), \tag{4.5}$$

where ϵ_0 is the dielectric constant, $C(x)$ is the doping profile and the following boundary conditions are used for the potential ϕ : $\phi(0,t) = 0\ V, \quad \phi(L,t) = 0.7\ V$.

Here, the relaxation parameter $\tau(x,t)$ is given by:

$$\tau(x,t) = \frac{m}{e}\frac{2\mu_0}{1 + \sqrt{1 + 4((\mu_0/v_d)E)^2}} \ .$$

The initial conditions are given by $f(x,v,0) = M(v)C(x)$. The particular doping profile $C(x)$ for the device together with the value of the physical contants $\mu_0, v_d, k_b, e, \epsilon_0$ and m are given in [28].

In order to control spurious oscillations, the numerical simulation of this model is based on the method of lines with high order non oscillatory finite differences reconstructions of the derivatives of the density function $f(x,v,t)$ [8]. In order to guarantee the precision, the computational domain (directions x and v) is discretized by taking an uniform mesh in each direction. We used a mesh with dimensions $(N_x+1)\times(N_v+1)$ in the numerical resolutions. The points coordinates of the mesh are calculated as we indicate below:

$$x_i = i\frac{L}{N_x-1}, \quad i = 0,\dots,N_x, \qquad v_j = -3.5 + j\frac{7}{N_v}, \quad j = 0,\dots,N_v.$$

Numerically, we simulate the equation (4.3) using a conservative fifth order finite difference Weighted Essentially Non Oscillatory (WENO5) scheme [8] to approximate the derivatives of f in the term $F(x,u,t)$ (see equation (4.3)). The value of the electric field E at each grid point x_i is computed by solving equation (4.5). To do that, we have used central finite differences to approximate the derivatives and the value of ϕ at each grid point is obtained by solving a tridiagonal linear system.

The concentration ρ is numerically evaluated at each grid point by using the midpoint quadrature formula (see equation (4.4)).

In order to apply the WENO5 scheme, we use the following boundary conditions:

- For all $i = 0,\dots,3, \quad j = 1,\dots,N_v-1$:

$$\begin{array}{lll} \text{If } v_j > 0 \text{ then:} & f_{-i,j} = 5\cdot 10^5 \cdot M(v_j), & f_{N_x+i,j} = f_{N_x-1,j}, \\ \text{In other case:} & f_{-i,j} = f_{1,j}, & f_{N_x+i,j} = 5\cdot 10^5\cdot M(v_j). \end{array}$$

- For all $j = 0,\dots,3, \quad i = 1,\dots,N_x-1$: $\quad f_{i,-j} = f_{i,1}, \qquad f_{i,N_v+j} = f_{i,N_v-1}$.

As a result, we obtain a stiff semi-discrete system of $(N_x-1)(N_v-1)$ ODEs, where the term G must be treated implicitly. In order to perform the time integration using an additive scheme, we write the ODE system in the following form:

$$\frac{\partial f_{i,j}}{\partial t} = \mathbf{f}(f_{i,j}) + \mathbf{g}(f_{i,j}), \quad i = 1,\dots,N_x-1, \quad j = 1,\dots,N_v-1, \tag{4.6}$$
$$\mathbf{f}(f_{i,j}) = \{-F\}_{i,j}, \qquad \mathbf{g}(f_{i,j}) = \{G\}_{i,j},$$

where $f_{i,j}$ denotes $f(x_i, v_j, t)$. To construct the system of ODEs, we have used the following arrangement of variables:

$$f_{1,1},\dots,f_{N_x-1,1}, f_{1,2},\dots,f_{N_x-1,2},\dots\dots, f_{1,N_v-1},\dots,f_{N_x-1,N_v-1}.$$

In this way, the Jacobian of $\mathbf{g}$ exhibits a banded structure with $\lceil N_v/2\rceil$ subdiagonals and $\lceil N_v/2\rceil$ superdiagonals.

We performed experiments of time integration from $t=0$ through $t = 10^{-2}$ by using 10 time steps with fixed size $h = 10^{-3}$ for different values of N_x and N_v. For this problem, we used a relative error $rtol = 10^{-14}$ and the absolute error $atol = 10^{-11}$ to establish the stopping criterion of the Newton process.

4.3. Numerical results. The numerical experiments were performed on an AMD Opteron processor 2.2Ghz, running Linux. We used the gnu fortran and C compilers with the compiler choices recommended by AMD for scientific codes. We also used versions of the BLAS library optimized for the AMD Opteron architecture.

TABLE 4.1
Results with LRR(3,2,2) scheme for the BRELAX problem

N	Trivial Predictor			Proposed Predictor		
	N_{its}	T_{Newton}	T_{total}	N_{its}	T_{Newton}	T_{total}
200	3.00	0.3800	0.9561	2.05	0.2880	0.8201
400	3.00	0.7520	1.7881	2.05	0.5200	1.5441
800	3.00	1.6321	3.8682	2.05	1.0601	3.1402
1200	3.00	2.4882	5.8084	2.05	1.7401	5.0043
1400	3.00	2.7522	6.4244	2.05	2.0081	5.8844
1600	3.00	3.3402	7.7285	2.05	2.2841	6.7764
1800	3.00	3.7122	8.5605	2.05	2.4882	7.2205
2000	3.00	4.1003	9.5686	2.05	2.7842	8.0045
4000	3.00	9.1686	20.3613	2.05	6.1204	16.5850
5000	3.00	11.6047	25.4176	2.05	8.1205	21.1973

The evaluation of the Jacobian matrix of the function g at a point is approximated by finite differences for both the *BRELAX* problem and the *Diode* problem.

To implement the ASIRK-3A scheme, we approximated the evaluation of the Jacobian at each time step as it is done in [18]. In this way, it is only necessary to perform one evaluation of the Jacobian for each time step.

As the aim of these numerical experiments is to show the influence of the predictors on the computational efficiency of the code, we performed several implementations of the LRR(3,2,2) and ASIRK-3A schemes using different stage value predictors, and measured the average number of Newton iterations (N_{its}), the time taken to complete the Newton iterations (T_{Newton}) and the total execution time (T_{total}) (see Tables 4.1, 4.2, 4.3 and 4.4).

For the LRR(3,2,2) scheme, we compare the efficiency of the solver when the low cost trivial predictor defined in (2.3) is used, with the case in which the second order predictor of the form (2.5) with B defined by (2.6).

For the ASIRK-3A scheme, we compare the low cost trivial predictor defined as:

$$K_{n+1}^{(0)} = e \otimes K_{n,3}$$

with the predictor of the form (3.7) with $\mathfrak{b}_0 = 0$ and the matrix $\mathfrak{B}$ given by (3.21).

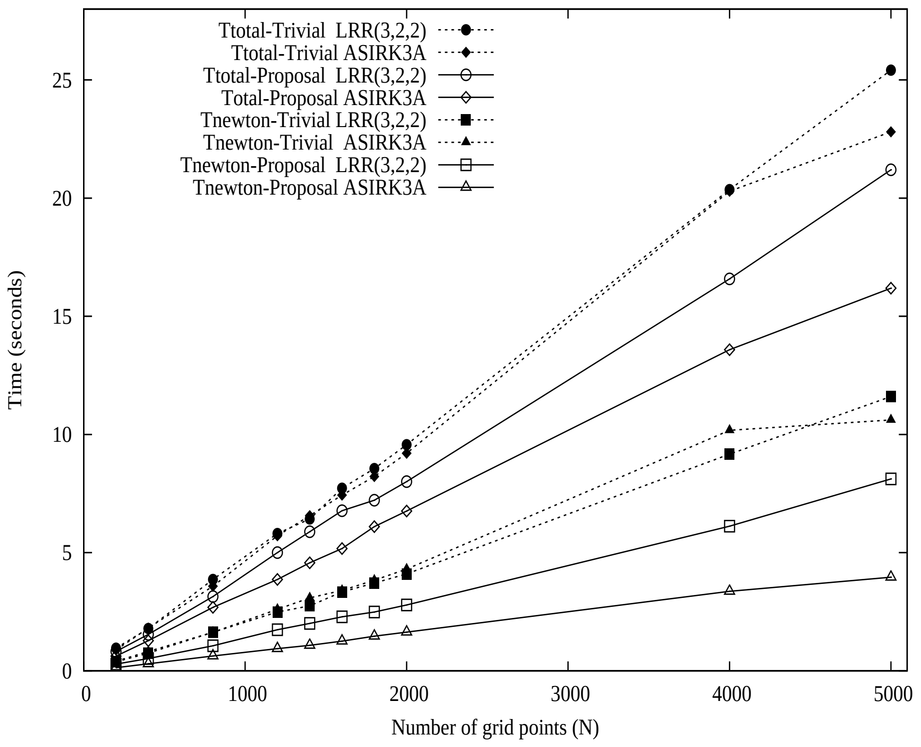
Tables 4.1 and 4.2 show the numerical results for the *BRELAX* problem with the LRR(3,2,2) method and the ASIRK-3A schemes respectively. The time results which appear in both Tables are graphically described in Figure 4.1 and a comparison of the average number of Newton iterations obtained by using each predictor is illustrated in Figure 4.3. It can be observed that for this problem, the use of the proposed predictor allows us to reduce considerably the average number of iterations. The impact on the total execution time is not too great because the Jacobian evaluation (obtained by finite differences) are outside the Newton iteration and this computing phase is highly significant in the overall execution time (for this particular problem). However, the cost of the Jacobian evaluation could be reduced by computing it analytically or by using the information known about the problem and consequently the impact of this phase on the overall execution time would not have been so great.

The numerical results for the *Diode* problem are given in Tables 4.3 and 4.4. The time results shown in both tables are graphically represented in Figure 4.2. It can be observed that for the *Diode problem*, the reduction in N_{its} is greater than for

TABLE 4.2
Results with ASIRK3A scheme for the BRELAX problem

N	Trivial Predictor			Proposed Predictor		
	N_{its}	T_{Newton}	T_{total}	N_{its}	T_{Newton}	T_{total}
200	3.00	0.4080	0.8841	1.00	0.1320	0.6200
400	3.00	0.8201	1.8081	1.12	0.3000	1.2761
800	3.00	1.6321	3.5802	1.12	0.6280	2.6842
1200	3.00	2.6002	5.7164	1.12	0.9401	3.8682
1400	3.00	3.0842	6.5644	1.12	1.0801	4.5683
1600	3.00	3.4002	7.4365	1.12	1.2561	5.1723
1800	3.00	3.8322	8.2245	1.12	1.4681	6.1044
2000	3.00	4.3043	9.2046	1.12	1.6401	6.7604
4000	3.17	10.1806	20.2893	1.00	3.3682	13.5848
5000	2.72	10.6127	22.8094	1.00	3.9642	16.1930

FIG. 4.1. *Execution times for BRELAX solvers with different meshes*



the *BRELAX* problem and, since the Jacobian evaluation is relatively cheap in this problem, the impact of the reduction on the overall execution time is very big.

Using the considered test problems, several numerical experiments were also performed to show the effects of the method/predictor used on the accuracy of the numerical results. The abovementioned values of *rtol* and *atol* for each problem have been also used in these experiments.

For the *BRELAX* problem, we performed the time integration for the interval

Table 4.3
Results with LRR(3,2,2) scheme for the Diode problem

$N_x \times N_v$	Trivial Predictor			Proposed Predictor		
	N_{its}	T_{Newton}	T_{total}	N_{its}	T_{Newton}	T_{total}
50×50	3.93	0.3960	0.4280	3.07	0.2920	0.3200
50×100	4.50	2.4402	2.5522	1.67	0.9401	1.0241
75×100	4.70	4.1283	4.2803	1.47	1.2641	1.4281
100×150	4.47	10.3486	10.7887	1.53	3.5362	3.9682
125×150	4.67	13.5048	14.0569	1.47	4.2483	4.7883
150×150	4.60	15.9690	16.6330	1.33	4.6483	5.3043
150×200	4.93	30.2419	31.5580	1.43	8.6925	9.8646
175×175	4.83	26.8457	27.9137	1.43	7.9685	9.0086
175×200	4.70	33.3701	34.7702	1.33	9.5166	10.8727
200×200	4.67	37.9304	39.5345	1.33	10.8367	12.4688

Table 4.4
Results with ASIRK3A scheme for the Diode problem

$N_x \times N_v$	Trivial Predictor			Proposed Predictor		
	N_{its}	T_{Newton}	T_{total}	N_{its}	T_{Newton}	T_{total}
50×50	4.07	0.4000	0.4240	1.30	0.1440	0.1800
50×100	4.40	2.4442	2.5442	1.23	0.7121	0.8000
75×100	4.47	3.9602	4.0923	1.17	1.0401	1.1801
100×150	4.43	10.3806	10.8167	1.17	2.7402	3.1642
125×150	4.47	13.1168	13.6729	1.17	3.4482	3.9842
150×150	4.43	15.6970	16.3450	1.17	4.1243	4.7643
150×200	5.07	31.1339	32.2700	1.17	7.1764	8.3045
175×175	5.10	28.5818	29.6379	1.47	8.2365	9.2366
175×200	4.50	32.2820	33.6381	1.93	13.8649	15.1889
200×200	4.57	37.4863	39.0544	2.57	21.0493	22.6094

$[0, 1.0]$ with $N = 200$ mesh points, using both solvers with different stage value predictors and 20000 time steps with fixed size $h = 5 \cdot 10^{-5}$. Figure 4.4 shows a graphical representation of the mass density at $t = 1.0$.

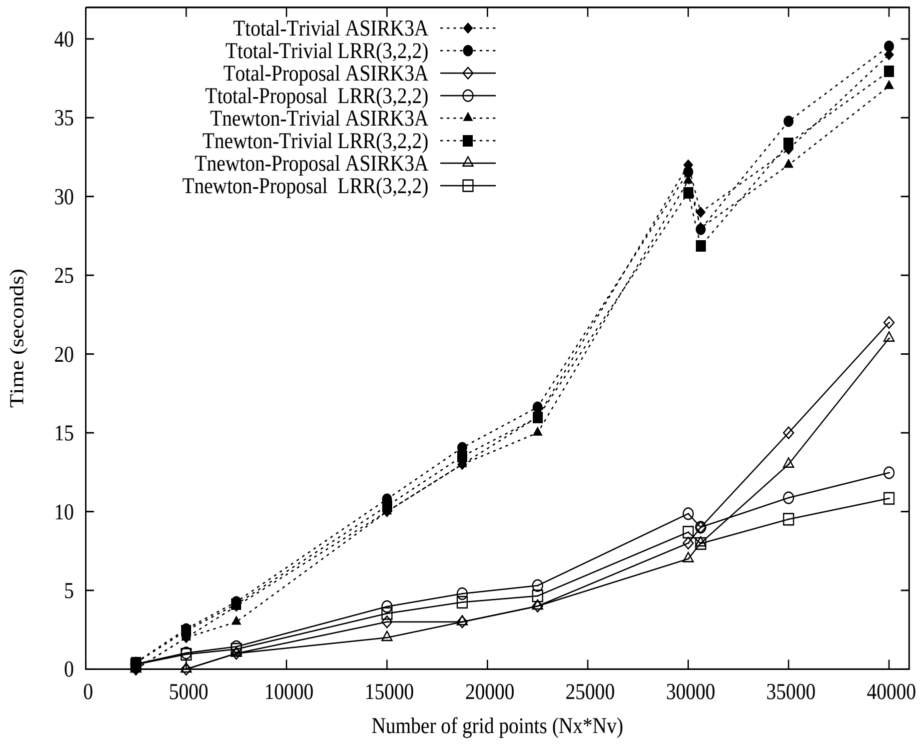
For the *Diode* problem, we performed the time integration for the interval $[0, 1.0]$, with $N_x = N_v = 150$, using both solvers with different stage value predictors and 1000 time steps with fixed size $h = 10^{-3}$. Figure 4.5 shows a graphical representation of the electron density $(\rho(x, t))$ at $t = 1.0$.

In all the figures, the results given by all the solvers, are almost indistinguishable. Therefore, these experiments do not reveal differences between the approximations obtained with different starting algorithms despite of the solutions computed using the proposed predictors are obtained in smaller runtimes.

5. Conclusions.

In this paper we have studied predictors for additive Runge-Kutta and ASIRK schemes. We have shown how to construct higher order predictors without additional cost and which require very little implementation effort. The numerical experiments carried out show that the use of good predictors can reduce considerably the execution time of the Newton iterations and therefore the overall execution time is diminished.

FIG. 4.2. *Execution times for Diode solvers with different meshes*



Acknowledgements. J. Mantas acknowledges partial support from DGI-MEC project MTM2008-06349-C03-03 and from project TIN2004-07672-C03-02. T. Roldán and I. Higueras acknowledge support from DGI-MEC project MTM2005-03894.

FIG. 4.3. *Average number of iterations for each solver when several predictors are used*

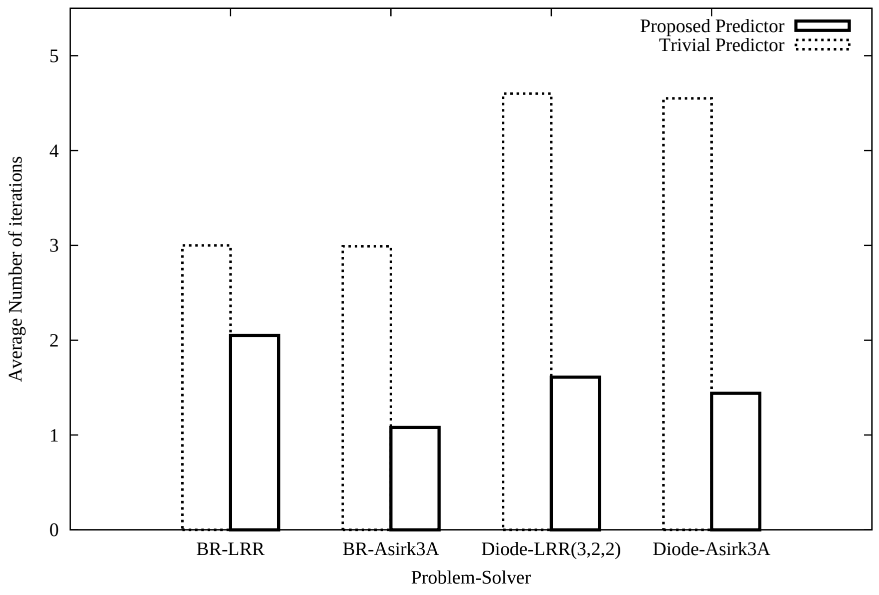


FIG. 4.4. *Numerical solution obtained with the BRELAX solvers for $N = 200$ and $t = 1.0$*

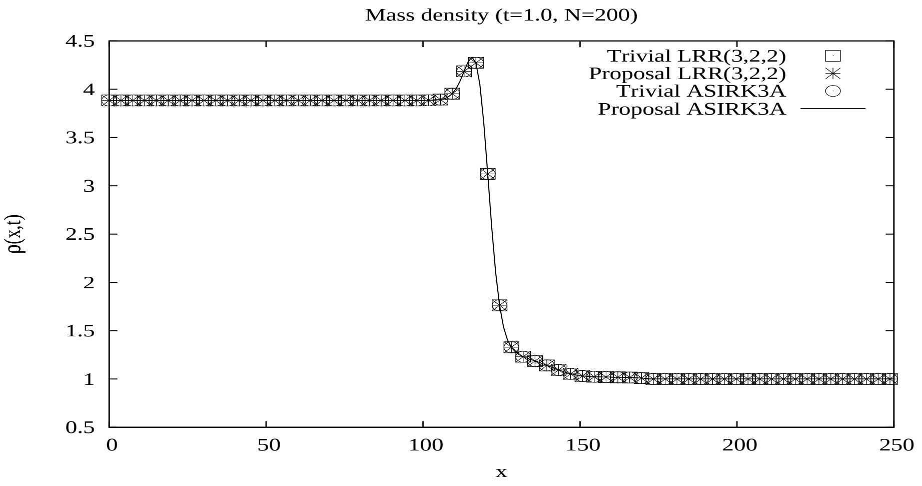
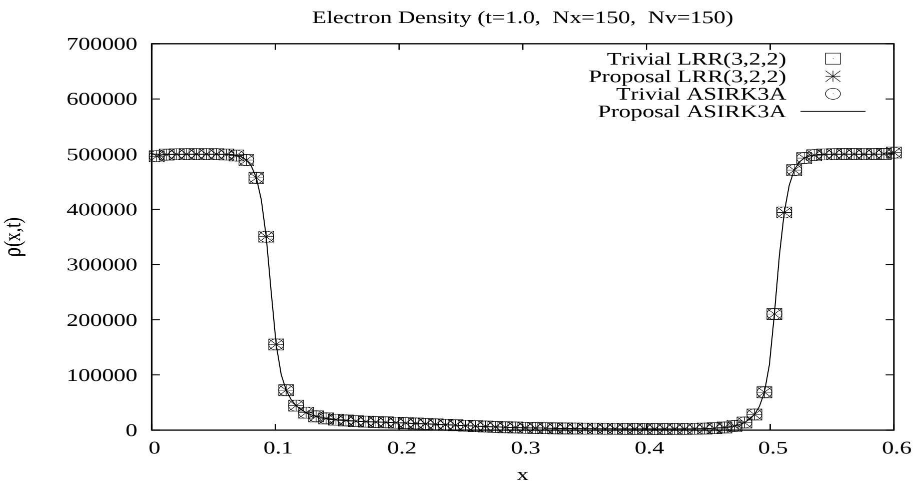


FIG. 4.5. *Numerical solution obtained with the Diode solvers for $N_x = 150$, $N_v = 150$ and $t = 1.0$*



REFERENCES

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

[2] J. J. Dongarra, J. Du Croz, S. Hammarling and R. J. Hanson, *An extended set of* FORTRAN *basic linear algebra subroutines.* ACM Trans. Math. Software, 14 (1988), pp. 1–17.

[3] Claus Bendtsen, *Parallel Software for stiff ODEs.* PhD thesis, 1996.

[4] Claus Bendtsen, *ParSODES. A parallel Stiff ODE Solver. Version 1.0.* User Guide (1996).

[5] C. Cercignani, I. Gamba, J. Jerome and C. W. Shu , *Device benchmarks comparisons via kinetic, hydrodynamic and high-field models.* Comput. Methods Appl. Mech. Engrg., 181 (2000), pp. 381–392.

[6] E. Hairer and G. Wanner, *Solving Ordinary Differential Equations II.* Springer, 1996.

[7] I. Higueras and T. Roldán, *Stage value predictors for additive and partitioned Runge–Kutta methods.* Appl. Numer. Math. 56, 1 (2006), pp. 1–18.

[8] G. Jiang, and C.-W. Shu, *Efficient Implementation of Weighted ENO Schemes.* J. Comp. Phys., 126, (1996), pp. 202-228.

[9] S. Jin and Z. P. Xin, *The relaxation schemes for systems of conservation laws in arbitrary space dimensions.* Comm. on Pure and Appl. Math. 48 (1995), pp. 235–276

[10] S. Jin, L. Pareschi and M. Slemrod, *A Relaxation Scheme for Solving the Boltzmann Equation Based on the Chapman-Enskog Expansion.* Acta Math. Appl. Sin. Engl. Ser. 18 (2002), pp. 37-62

[11] A. Kanevsky, M. H. Carpenter, D. Gottlieb and J. S. Hesthaven, *Application of implicit-explicit high order Runge-Kutta methods to discontinuous-Galerkin schemes.* J. Comput. Phys. 225 (2007), pp. 1753–1781.

[12] C. A. Kennedy and M. H. Carpenter, *Additive Runge-Kutta schemes for convection-diffusion-reaction equations.* Appl. Numer. Math. 44, 1-2 (2003), pp. 139–181.

[13] M. Laburta, *Starting algorithms for* IRK *methods.* J. Comput. Appl. Math. 83, (1997), pp. 269–288.

[14] E. Lindblad, D.M. Valiev, B. Müller, J. Rantakokko, P. Lötstedt and M.A. Liberman, *Implicit-Explicit Runge-Kutta method for combustion simulation.* ECCOMAS CFD, TU Delft, The Netherlands, 5-8 September (2006).

[15] S.F. Liotta, V. Romano and G. Russo, *Central schemes for balance laws of relaxation type.* SIAM J. Numer. Anal. 38, 4 (2000), pp. 1337–1356.

[16] J.M. Mantas, J. Ortega and J. Carrillo, *Component-Based Derivation of a Stiff ODE Solver implemented on a PC Cluster.* Internat. J. Parallel Progr. 30 (2002), pp. 99-148.

[17] J. M. Mantas, L. Pareschi, J. Carrillo and J. Ortega, *Parallel Integration of Hydrodynamical Approximations of the Boltzmann Equation for rarefied gases on a Cluster of Computers.* J. Comput. Methods Sci. Eng., 4 (2004), pp. 33-41.

[18] J. M. Mantas, P. Gonzalez and J. Carrillo, *Parallelization of Implicit-Explicit Runge-Kutta Methods for Cluster of PCs.* Lecture Notes in Comput. Sci., 3648, (2005), 815–825.

[19] J. M. Mantas, *Desarrollo Basado en Componentes de Resolutores de Ecuaciones Diferenciales para Multicomputadores.* PhD thesis, Universidad de Granada, Granada., 2003.

[20] L. Pareschi and G. Russo, *Implicit-Explicit Runge-Kutta schemes for stiff systems of differential equations.* Recent Trends in Numerical Analysis, pp. 269–289, Adv. Theory Comput. Math. 3, Nova Sci. Publ., Huntington, NY, 2001.

[21] L. Pareschi and G. Russo, *Implicit-Explicit Runge-Kutta Schemes and Applications to Hyperbolic Systems with Relaxation.* J. Sci. Comput., 25, 1 (2005), pp. 129–155.

[22] S. Pieraccini and G. Puppo, *Implicit-Explicit Schemes for BGK Kinetic Equations.* J. Sci. Comput., 32, 1 (2007), pp. 1–28.

[23] T. Roldán, *Implicit Runge–Kutta methods for DAEs: starting algorithms.* PhD thesis, Universidad Pública de Navarra, 2000.

[24] T. Roldan and I. Higueras, *IRK methods for DAE: starting algorithms.* J. Comput. Appl. Math., 111, 1 (1999), pp. 77–92.

[25] Y. Saad, *Iterative Methods for Sparse Linear Systems.* PWS Publishing Co., 1996.

[26] J. Sand, *Methods for starting iterations schemes for implicit Ruge–Kutta formulae.* Department of Computer Science, University Copenhagen, Denmark (1989).

[27] M. Seaïd, *Non-oscillatory relaxation methods for the shallow-water equations in one and two space dimensions.* Internat. J. Numer. Methods Fluids 46 (2004) pp. 457–484.

[28] J. A. Carrillo and F. Vecil, *Non oscillatory interpolation methods applied to Vlasov models.* SIAM J. Sci. Comput., 29, 1 (2007), pp. 1179–1206.

[29] J.G. Verwer and B.P. Sommeijer, *An Implicit-Explicit Runge–Kutta–Chebyshev Scheme for Diffusion-Reaction Equations.* SIAM J. Sci. Comput. 25 (2004), pp. 1824–1835.

[30] X. Zhong, *Additive Semi-Implicit Runge-Kutta Methods for Computing High-Speed Nonequilibrium Reactive Flows*. J. Comput. Phys., 128, 1 (1996), pp. 19–31.