J Sci Comput
DOI 10.1007/s10915-014-9879-0

Runge–Kutta Residual Distribution Schemes

Andrzej Warzyński · Matthew E. Hubbard · Mario Ricchiuto

Received: 11 July 2013 / Revised: 13 April 2014 / Accepted: 4 June 2014

Abstract We are concerned with the solution of time-dependent non-linear hyperbolic partial differential equations. We investigate the combination of residual distribution methods with a consistent mass matrix (discretisation in space) and a Runge–Kutta-type time-stepping (discretisation in time). The introduced non-linear blending procedure allows us to retain the explicit character of the time-stepping procedure. The resulting methods are second order accurate provided that both spatial and temporal approximations are. The proposed approach results in a global linear system that has to be solved at each time-step. An efficient way of solving this system is also proposed. To test and validate this new framework, we perform extensive numerical experiments on a wide variety of classical problems. An extensive numerical comparison of our approach with other multi-stage residual distribution schemes is also given.

Keywords Hyperbolic conservation laws · Time-dependent problems · Second order schemes · Residual distribution · Runge–Kutta time-stepping

1 Introduction

In this paper we study the numerical solution of hyperbolic partial differential equations. In the scalar case these equations take the following form:

$$\partial_t u + \nabla \cdot \mathbf{f}(u) = 0 \qquad \text{in } \Omega \times [0, T]. \tag{1}$$

A. Warzyński (✉) · M. E. Hubbard
School of Computing, University of Leeds, Leeds LS2 9JT, UK
e-mail: scaw@leeds.ac.uk

M. E. Hubbard
e-mail: m.e.hubbard@leeds.ac.uk

M. Ricchiuto
INRIA Bordeaux Sud-Ouest, 33405 Talence Cedex, France
e-mail: Mario.Ricchiuto@inria.fr

Published online: 26 June 2014

Ω is the spatial domain and T is the given final time. Equation (1) is equipped with an initial solution:

$$u(\mathbf{x}, t = 0) = u_0(\mathbf{x}) \qquad \mathbf{x} \in \Omega,$$

and boundary conditions defined on $\partial\Omega$ or a properly defined subset. The above equation models the evolution of the conserved unknown $u(\mathbf{x}, t)$, the flux of which is denoted here by $\mathbf{f}$. Systems of equations are introduced later in the text. The framework we shall design our schemes in is that of Residual Distribution ($\mathcal{RD}$).

The residual distribution framework (see [19] for a thorough overview) is widely recognised as an efficient and accurate way of discretising steady state hyperbolic PDEs. It was originally introduced by Roe [35] as an alternative to the frequently-used finite volume and finite difference methods. Its ability to perform genuinely multidimensional upwinding [4,20] enables the construction of approximations free of spurious oscillations, even in the vicinity of shocks in the solution. More important, it facilitates construction of methods which are capable of capturing the underlying physical processes incorporated within the mathematical model in a truly multidimensional manner. Extension to systems, although not straightforward, is well understood and covered in the literature. Usually it is performed with the aid of Roe's parameter vector in the case of the Euler equations [21,37] or, for instance, the CRD approach of Csík et al. [16] in more complex cases. In [4,25] it was demonstrated that when residual distribution methods do provide plausible solutions, these are usually more accurate than those obtained with the aid of more popular finite volume methods. However, the residual distribution framework lacks the robustness of the finite volume approach, in particular, when it comes to discretising time-dependent equations. Various techniques of extending residual distribution methods to transient problems exist, but none is considered fully satisfactory. The existing approaches can be grouped into two categories: the space-time and consistent mass matrix frameworks.

The *Space-Time* residual distribution framework [5,15,17,22,34] is very faithful to the $\mathcal{RD}$ and multidimensional upwinding spirit. Although it allows construction of discretisations with *all* the desired properties, those methods are subject to a CFL-type restriction on the time-step. This restriction is particularly disappointing when taking into account that they are, by construction, implicit. In the *two layer* variant [14] one couples two space-time slabs at a time and solves the equations simultaneously in both. The resulting system to be solved at each step is larger, but the construction removes from one of the layers the restriction on the time-step. In theory, this means that an arbitrarily large time-step can be used. For a full discussion see [15]. More recently, Hubbard and Ricchiuto [26] proposed to drive the height of one of the space-time slabs (and hence its associated time-step) to zero so that the scheme becomes discontinuous-in-time. The resulting formulation is simpler than the original whereas all of the properties are retained.

In this paper we focus on a different approach. The *Consistent Mass Matrix* formulation was inspired by the discovery of a close relation between the residual distribution and finite element frameworks [12]. This, quite naturally, led to the introduction of a mass matrix m_{ij} (see, in particular, [33] and [18]) and coupling in space of the time derivatives of the nodal values. It was the first successful attempt to construct second-order residual distribution schemes for time-dependent problems. This approach was implemented and investigated in a number of references, i.e. [5,18,33,34] or [11]. In all of these references the authors used multi-step methods to integrate the underlying PDE in time. It is usually argued that the major disadvantage of this approach is the fact that the resulting discretisations are implicit, i.e. the resulting linear system is not diagonal (even if explicit multi-step methods are utilised) and therefore expensive. In [7] Ricchiuto and Abgrall modified the consistent mass matrix framework by introducing the so-called shifted time operator. By combining it with a multi-stage Runge–Kutta method, they derived a genuinely explicit scheme. The resulting methods are indeed explicit, but the formulation is somewhat complicated and leaves open the question of constructing a genuinely positive method. Recently, the authors of this paper proposed to draw together the discontinuous-in-space residual distribution scheme [6,27] and the first order forward Euler time-stepping procedure. The resulting scheme [42] is explicit and positive, but so far only first order accurate.

In this contribution we focus on the consistent mass matrix formulation in its original form, i.e. continuous-in-time. To integrate the underlying PDE in time we use the second order TVD Runge–Kutta discretisation method due to Shu and Osher [38]. Although this time-stepping procedure is explicit, our methods are implicit in the sense that at each time-step (and each RK stage) a system of linear equations (regardless whether the PDE is linear or non-linear) has to be solved. This contrasts with the aforementioned space-time approximations, in which a system of nonlinear equations must be solved at each time-step, therefore reducing the computation per time-step. We investigate the resulting discretisations in terms of accuracy and positivity. The order of accuracy we obtain, i.e. two, corresponds to that demonstrated in [7]. However, by avoiding the shifted time operator we construct a formulation that is more straightforward. The goal of the new approach was to improve the accuracy of the explicit Runge–Kutta residual distribution framework of Ricchiuto and Abgrall [7]. The comparison of numerical results in this paper does indeed demonstrate that the new method gives significantly more accurate approximations on the same computational meshes, due to lower numerical diffusion, but only at the expense of requiring more computation per time-step and reduced stability in regions with strong shocks. The case in which the resulting system of equations is non-linear (i.e. genuinely non-linear schemes) will not be covered here. It is a subject of ongoing research.

This paper is structured as follows. In the following section we introduce the notation and basic assumptions. In Sect. 3 we introduce the Consistent Mass Matrix formulation and then move to the Runge–Kutta Residual Distribution ($\mathcal{RKRD}$) framework in Sect. 4. This is followed by the introduction of the explicit Runge–Kutta Residual Distribution framework of Ricchiuto and Abgrall [7] in Sect. 5. Extension to systems of non-linear equations is discussed in Sect. 6 and extensive numerical results are presented in Sect. 7.

2 The Notation

Although our approach extends to $\mathbb{R}^3$, for clarity and brevity we assume that the spatial domain Ω is embedded in $\mathbb{R}^2$. We also assume that Ω is subdivided into non-overlapping triangular elements, denoted by E, belonging to $\mathcal{T}_h$, such that:

$$\bigcup_{E\in\mathcal{T}_h} E = \Omega.$$

$\mathcal{D}_i$ will stand for the subset of triangles containing node $\mathbf{x}_i$. It is assumed that the temporal domain $[0, T]$ is discretized into a set of $N + 1$ discrete levels $\{t^n\}_{n=0,1...,N}$ such that:

$$t^0 = 0, \quad t^N = T, \quad t^n < t^{n+1} \quad \text{and} \quad \Delta t^n = t^{n+1} - t^n.$$

For each element $E \in \mathcal{T}_h$ and for each node $\mathbf{x}_i \in E$, ψ_i^E is defined as the linear Lagrange basis function associated with $\mathbf{x}_i$ respecting:

$$\psi_i^E(\mathbf{x}_j) = \delta_{ij} \quad \forall i, j \in \mathcal{T}_h, \qquad \sum_{j \in E} \psi_j^E = 1 \quad \forall E \in \mathcal{T}_h. \tag{2}$$

As long as it does not introduce any ambiguity, the superscript ^{E} will be omitted. At each time level t^n the approximate solution u_h^n is assumed to be globally continuous and linear within each element $E \in \mathcal{T}_h$, and is given by:

$$u_h^n(\mathbf{x}) = \sum_i \psi_i(\mathbf{x})\, u_i^n, \tag{3}$$

where $u_i^n = u_h^n(\mathbf{x}_i)$ are the nodal values of the approximate solution at time t^n. Having decided to focus on piece-wise linear approximations, we aim to design a scheme that is second order accurate.

Last, but not least, we introduce the cell residual $\phi^E(u)$:

$$\phi^E(u) = \int_E \nabla \cdot \mathbf{f}(u)\, d\Omega.$$

Cell residuals are one of the key ingredients of all residual distribution approximations. This will become apparent in the following sections.

3 The Consistent Mass Matrix Formulation

Cell residuals introduced in the previous section are used to construct the consistent mass matrix semi-discrete approximation of (1):

$$\sum_{E \in \mathcal{D}_i} \sum_{j \in E} m_{ij}^E \frac{du_j}{dt} + \sum_{E \in \mathcal{D}_i} \beta_i \phi^E = 0. \tag{4}$$

The distribution coefficients β_i $(i = 1, 2, 3)$ define the so-called distribution strategy. Note that each set of these coefficients leads to a distinct approximation. Four examples of such distributions are given in Sect. 4. Here we only mention that for conservation, in every element $E \in \mathcal{T}_h$, the following condition has to be satisfied:

$$\beta_1 + \beta_2 + \beta_3 = 1 \qquad \forall_{E \in \mathcal{T}_h}.$$

In the case of steady state problems for which piece-wise linear approximations are implemented, existence of $C \in \mathbb{R}$ such that:

$$\beta_i \leq C \qquad \forall E \in \mathcal{T}_h \quad \forall i \in E, \tag{5}$$

i.e. the boundedness of β_i, guarantees accuracy of order two (see [5] for details). The definition of the mass matrix, m_{ij}, that guarantees accuracy of order two in both space and time when transient problems are considered is not unique. Four different approaches are known, only one of which will be employed here. A thorough overview was given in [7] in which it was observed that their Formulation 2 (naming as in [7]) gives best (in terms of accuracy and stability) results. For each cell $E \in \mathcal{T}_h$ this mass matrix is defined as:

$$m_{ij}^E = \frac{|E|}{36}(3\delta_{ij} + 12\beta_i - 1) \tag{6}$$

with δ_{ij} Kronecker's delta. The consistency of this mass matrix with the distribution strategy follows from the dependency of m_{ij}^E on β_i. This formulation was originally derived by März [33]. Its construction is based on the analogy between the linearity preserving $\mathcal{RD}$, i.e. methods for which Condition (5) is satisfied, and the stabilized Galerkin finite element methods.

4 The Runge–Kutta Residual Distribution Framework

The Runge–Kutta Residual Distribution framework is derived by first integrating (1) in time using Runge–Kutta scheme. Here we choose the second order TVD Runge–Kutta time-stepping indicated by Shu and Osher [38], which leads to the following semi-discrete formulation:

$$\begin{cases} \frac{\delta u^1}{\Delta t} + \nabla \cdot \mathbf{f}(u^n) = 0, \\ \frac{\delta u^{n+1}}{\Delta t} + \frac{1}{2}\left(\nabla \cdot \mathbf{f}(u^n) + \nabla \cdot \mathbf{f}(u^1)\right) = 0, \end{cases} \tag{7}$$

in which $\delta u^k = u^k - u^n$ is the increment during the current Runge–Kutta stage and u^1 is the intermediate Runge–Kutta estimate approximating u at time $t = t^{n+1}$. Using the consistent mass matrix formulation (4) to integrate both stages in (7) in space leads to:

$$\begin{cases} \sum_{E \in \mathcal{D}_i} \sum_{j \in E} m_{ij}^E \frac{\delta u_j^1}{\Delta t} + \sum_{E \in \mathcal{D}_i} \beta_i \phi^E(u^n) = 0, \\ \sum_{E \in \mathcal{D}_i} \sum_{j \in E} m_{ij}^E \frac{\delta u_j^{n+1}}{\Delta t} + \sum_{E \in \mathcal{D}_i} \frac{1}{2} \beta_i \left(\phi^E(u^n) + \phi^E(u^1)\right) = 0. \end{cases} \tag{8}$$

Equation (8) defines two linear systems to be solved at each time-step. These systems can be written in a more compact form as:

$$\begin{cases} u^1 = u^n - \Delta t\, \mathbf{M}^{-1} \boldsymbol{\phi}^1, \\ u^{n+1} = u^1 - \Delta t\, \mathbf{M}^{-1} \boldsymbol{\phi}^2, \end{cases} \tag{9}$$

which is the form that was employed to carry out numerical experiments in Sect. 7. In Formulation (9) $\mathbf{M}$ is the global mass matrix, the entries of which are defined by Formula (6), and $\boldsymbol{\phi}^1$ and $\boldsymbol{\phi}^2$ are the vectors of signals each node has received, i.e.

$$\boldsymbol{\phi}_i^1 = \sum_{E \in \mathcal{D}_i} \beta_i \phi^E(u^n), \qquad \boldsymbol{\phi}_i^2 = \sum_{E \in \mathcal{D}_i} \frac{1}{2} \beta_i \left(\phi^E(u^n) + \phi^E(u^1)\right).$$

Formulation (9) (or, equivalently, (8)) defines the second order TVD Runge–Kutta Residual Distribution ($\mathcal{RKRD}$) framework. By choosing different time-stepping routines one obtains different frameworks. However, we believe that our choice is appropriate. The spatial accuracy of the residual distribution framework in the form considered here is at most two. Combining it with higher than second order time-stepping routine is unlikely to increase the accuracy. This was investigated in [7]. Among the second order Runge–Kutta methods the second order TVD method (7) is the most frequent choice, which brings us to Formulation (9). For brevity, hereafter this formulation will be referred to as the Runge–Kutta Residual Distribution ($\mathcal{RKRD}$) framework.

Let us now introduce four examples of algorithms falling into the framework of $\mathcal{RKRD}$ methods. Each such scheme is constructed by first choosing a distribution strategy for cell residuals. This distribution determines the β coefficients which are then substituted into (8) to complete the construction. There is a number of well understood and established distribution techniques for cell residuals within the $\mathcal{RD}$ framework. An overview can be found in [19]. Here we will focus on the

- N, LDA, SU and BLEND

schemes leading to, respectively, the

- RKRD-N, RKRD-LDA, RKRD-SU and RKRD-BLEND

methods.

The RKRD-N scheme

Among linear and positive schemes, the N scheme of Roe [36] allows the largest time-step and has the smallest cross diffusion [39]. The N scheme is usually defined with the aid of the so-called flow sensors. For each vertex $i \in E$, the corresponding flow sensors are defined as:

$$k_i = -\frac{\mathbf{a}(u)\cdot\mathbf{n}_i}{2}|e_i|, \quad k_i^+ = \max(0, k_i), \quad k_i^- = \min(0, k_i), \tag{10}$$

in which $\mathbf{n}_i$ is the outward pointing unit normal vector to edge e_i (opposite the ith vertex). $|e_i|$ denotes the length of e_i and $\mathbf{a}(u) = \frac{\partial \mathbf{f}}{\partial u}$ is the flux Jacobian. We will also need the following quantities:

$$N = \left(\sum_{j\in E} k_j^+\right)^{-1}, \quad u_{in} = -\sum_{j\in E} N k_j^- u_j.$$

Finally, the distribution itself reads:

$$\beta_i^N \phi^E = k_i^+(u_i - u_{in}).$$

The N scheme is only first order accurate and hence the RKRD-N scheme cannot be expected to be more accurate than that. Recall that consistency of the mass matrix discussed in Sect. 3 is required to guarantee that the accuracy remains of order two when a linearity preserving scheme is applied to time-dependent problems. However, in the case of the N scheme the linearity preservation condition (i.e. Eq. (5)) is not satisfied and hence there is no need for such a consistency condition to be considered. Moreover, the equivalence between the stabilised Galerkin finite element and residual distribution methods was derived on the assumption that the distribution coefficients are bounded (linearity preservation). Again, the N scheme is not linearity preserving and hence Formulation (6) cannot be used. Instead, we replace the mass matrix in this case with the diagonal matrix:

$$m_{ij}^N = \delta_{ij}\frac{|E|}{3}.$$

Note that an identical formulation is obtained by row-lumping the high order mass matrix (6).

The RKRD-LDA scheme

The Low Diffusion A scheme of Roe [36], more often referred to as the LDA scheme, like the N scheme is linear, but not positive. However, this scheme has one nice property which the N scheme lacks, namely the linearity preservation. This means that in smooth regions, provided that the discrete representation of the data is linear, it is second order accurate (see [4] for details). The distribution coefficients for this scheme are given by:

$$\beta_i^{LDA} = \frac{k_i^+}{\sum_{j\in E} k_j^+} \geq 0.$$

The consistent mass matrix that guarantees second order of accuracy is defined by (6).

The RKRD-SU scheme

The SU distribution was inspired by the close link between the residual distribution and Galerkin finite element frameworks (in particular the Streamline Upwind Petrov Galerkin approach [10,29,30]). Its derivation can be found in [19]. The distribution coefficients are given by:

$$\beta_i^{SU} = \frac{1}{3} + k_i \tau,$$

in which τ is a scaling parameter, taken here as

$$\tau = \left(\sum_{j \in E} |k_j| \right)^{-1}.$$

This scheme is linear and linearity preserving, but not positive. As in the case of the LDA scheme, the consistent mass matrix is defined by (6).

The RKRD-BLEND scheme

Desire to construct methods which are simultaneously linearity preserving and positive brings the need to consider non-linear distributions. As a representative of this class of discretisations, the BLEND scheme will be now introduced. It is obtained by *blending* the two linear schemes presented so far, namely the positive N and the linearity preserving LDA schemes. The signals in this case are defined as:

$$\phi_i^{BLEND} = (1 - \theta(u_h))\, \phi_i^{LDA} + \theta(u_h) \phi_i^{N},$$

in which $\theta(u_h)$ is a blending coefficient. In this work we propose a blending parameter allowing retention of the explicit nature of the time-marching scheme. In particular, we propose to combine our $\mathcal{RKRD}$ framework with the time-shifted residual evaluation proposed in [7]. In every cell E this blending coefficient is defined as:

$$\theta^k(u_h) = \frac{\left| \overline{\Xi^{E(k)}} \right|}{\sum_{j \in E} \left| \overline{\Xi_j^{N(k)}} \right|}$$

where $k = 1, 2$ denotes Runge–Kutta stage and $\overline{\Xi^{E(k)}}$ the total shifted residual:

$$\overline{\Xi^{E(k)}} = \int_E \left(\overline{\delta u^k} + e^k \right) d\Omega.$$

$\overline{\delta u^k}$ is the so-called *shifted time-operator* introduced in [7]:

$$\overline{\delta u^k} = u^{k-1} - u^n \tag{11}$$

and e^1 and e^2 are the corresponding evolution operators:

$$e^1 = \nabla \cdot \mathbf{f}(u^n), \qquad e^2 = \frac{1}{2} \nabla \cdot \mathbf{f}(u^1) + \frac{1}{2} \nabla \cdot \mathbf{f}(u^n).$$

Finally, $\overline{\Xi_j^{N(k)}}$ is determined by signals sent by distributing the residuals with the aid of the N scheme and is defined as:

$$\overline{\Xi_j^{N(k)}} = \frac{|E|}{3} \frac{\overline{\delta u^k}}{\Delta t} + \beta_j^N \int_E e^k \, d\Omega.$$

The mass matrix for the BLEND scheme is defined as:

$$m_{ij}^{BLEND} = \theta(u_h) m_{ij}^{N} + (1 - \theta(u_h)) m_{ij}^{LDA}.$$

Note that the above formulation guarantees that the resulting system of equations is *linear*. Indeed, had $\theta^k(u_h)$ depended on u_h^{n+1} (or, to be more precise, on δu^k rather than on $\overline{\delta u^k}$), this would not have been the case and a system of *non-linear* equations would have been constructed instead. Therefore the RKRD-BLEND scheme is not genuinely non-linear. Our aim is to construct efficient schemes and hence the desire to avoid non-linear approximations.

5 The Explicit Runge–Kutta Residual Distribution Framework

The method presented in Sect. 4 is implicit in the sense that at every time-step two linear systems have to be solved. In [7] Ricchiuto et al. derived an approximation to that approach, namely the framework of explicit Runge–Kutta Residual Distribution methods, in which the resulting linear systems are diagonal. It is based on the observation that for every cell $E \in \mathcal{T}_h$ and set of distribution coefficients β_i, there exists a uniformly bounded and locally differentiable *bubble* function γ_i, such that $\sum_{i \in E} \gamma_i = 0$, and the following relation holds (cf. Eq. (4)):

$$\begin{aligned} &\sum_{E \in \mathcal{D}_i} \sum_{j \in E} m_{ij}^E \frac{du_j}{dt} + \sum_{E \in \mathcal{D}_i} \beta_i \phi^E \\ &\quad = \int\limits_E \psi_i \left(\frac{\partial u_h}{\partial t} + \nabla \cdot \mathbf{f}(u_h) \right) d\Omega + \int\limits_E \gamma_i \left(\frac{\partial u_h}{\partial t} + \nabla \cdot \mathbf{f}(u_h) \right) d\Omega. \end{aligned} \tag{12}$$

For a proof of this statement and examples of bubble functions satisfying the above refer to [7]. The Lagrange basis function ψ_i acts here as the Galerkin test function. As a consequence, every residual distribution discretization that fits into Formulation (4) can be rewritten as a sum of a finite element-type term and a stabilizing bubble function contribution. It follows immediately that the first stage in System (8) can be rewritten as

$$\begin{aligned} &\sum_{E \in \mathcal{D}_i} \sum_{j \in E} m_{ij}^E \frac{\delta u_j^1}{\Delta t} + \sum_{E \in \mathcal{D}_i} \beta_i \phi^E(u^n) \\ &\quad = \int\limits_E \psi_i \left(\frac{\delta u^1}{\Delta t} + \nabla \cdot \mathbf{f}(u^n) \right) d\Omega + \int\limits_E \gamma_i \left(\frac{\delta u^1}{\Delta t} + \nabla \cdot \mathbf{f}(u^n) \right) d\Omega. \end{aligned} \tag{13}$$

Similarly, the second stage in (8) can be rewritten as

$$\begin{aligned} &\sum_{E \in \mathcal{D}_i} \sum_{j \in E} m_{ij}^E \frac{\delta u_j^{n+1}}{\Delta t} + \sum_{E \in \mathcal{D}_i} \frac{1}{2} \beta_i \left(\phi^E(u^n) + \phi^E(u^1) \right) \\ &= \frac{1}{2} \int\limits_E \psi_i \left(\frac{\delta u^{n+1}}{\Delta t} + \nabla \cdot \mathbf{f}(u^1) \right) d\Omega + \frac{1}{2} \int\limits_E \gamma_i \left(\frac{\delta u^{n+1}}{\Delta t} + \nabla \cdot \mathbf{f}(u^1) \right) d\Omega \\ &\quad + \frac{1}{2} \int\limits_E \psi_i \left(\frac{\delta u^{n+1}}{\Delta t} + \nabla \cdot \mathbf{f}(u^n) \right) d\Omega + \frac{1}{2} \int\limits_E \gamma_i \left(\frac{\delta u^{n+1}}{\Delta t} + \nabla \cdot \mathbf{f}(u^n) \right) d\Omega. \end{aligned} \tag{14}$$

Recall that the above formulation leads to a global non-diagonal mass matrix that has to be solved at every stage of the Runge–Kutta time-stepping. In order to construct a *genuinely* explicit method, i.e. such that the mass matrix is diagonal, Ricchiuto and Abgrall introduced the so-called *shifted time-operator*,

$$\overline{\delta u^k} = u^{k-1} - u^n \, , \tag{15}$$

and substituted it into the right-hand-side of Eqs. (13)–(14), but only in the bubble function contribution. In the first stage (Eq. (13)), this leads to the approximation

$$\begin{aligned} &\sum_{E \in \mathcal{D}_i} \sum_{j \in E} m_{ij}^E \frac{\delta u_j^1}{\Delta t} + \sum_{E \in \mathcal{D}_i} \beta_i \phi^E(u^n) \\ &\approx \int_E \psi_i \left(\frac{\delta u^1}{\Delta t} + \nabla \cdot \mathbf{f}(u^n) \right) d\Omega + \int_E \gamma_i \left(\frac{\overline{\delta u^1}}{\Delta t} + \nabla \cdot \mathbf{f}(u^n) \right) d\Omega . \end{aligned} \tag{16}$$

A similar relation holds for the second stage, i.e. Eq. (14). The next steps involve mainly algebraic manipulations and are detailed in [7]. The final form of the scheme considered here, referred to in [7] as the globally lumped formulation, is given by

$$\begin{cases} |S_i| \frac{u_i^1 - u_i^n}{\Delta t} + \sum_{E \in \mathcal{D}_i} \beta_i \phi^E(u^n) = 0, \\ |S_i| \frac{u_i^{n+1} - u_i^1}{\Delta t} + \sum_{E \in \mathcal{D}_i} \beta_i \Phi^{RK}(u^n, u^1) = 0, \end{cases} \tag{17}$$

in which Φ^{RK} is the Runge–Kutta residual defined as

$$\Phi^{RK}(u^n, u^1) = \sum_{j \in E} m_{ij}^E \frac{u_j^1 - u_j^n}{\Delta t} + \frac{1}{2} \beta_i \left(\phi^E(u^n) + \phi^E(u^1) \right) .$$

If this formulation is rewritten in the form of Eq. (9) then

$$\begin{cases} u^1 = u^n - \Delta t \, \mathbf{M_{ex}}^{-1} \boldsymbol{\phi_{ex}}^1 , \\ u^{n+1} = u^1 - \Delta t \, \mathbf{M_{ex}}^{-1} \boldsymbol{\phi_{ex}}^2 , \end{cases} \tag{18}$$

in which $\mathbf{M_{ex}}$ is a diagonal matrix, the entries of which are given by: $(\mathbf{M_{ex}})_i = \frac{1}{|S_i|}$. The right-hand-side vectors $\boldsymbol{\phi_{ex}}^1$ and $\boldsymbol{\phi_{ex}}^2$ are defined as:

$$(\boldsymbol{\phi_{ex}}^1)_i = \sum_{E \in \mathcal{D}_i} \beta_i \phi^E(u^n), \qquad (\boldsymbol{\phi_{ex}}^2)_i = \sum_{E \in \mathcal{D}_i} \beta_i \Phi^{RK}(u^n, u^1) .$$

This contrasts with the matrix, $\mathbf{M}$, defining the (implicit) $\mathcal{RKRD}$ scheme (9), which is, in general, non-diagonal. Ricchiuto and Abgrall [7] prove that the above construction does not degrade the overall accuracy of the scheme. Their experimental investigation also demonstrates that the resulting discretization is second order accurate. This is confirmed by the numerical results presented in Sect. 7.

As in the case of the (implicit) $\mathcal{RKRD}$ framework, four different distribution strategies will be considered here, namely the LDA, SU, N and BLEND schemes. These will lead to, respectively, the explicit RKRD-LDA, explicit RKRD-SU, explicit RKRD-N and explicit RKRD-BLEND schemes. The distribution coefficients, β_i, and the blending parameter, θ, are defined as in Sect. 4. Note that the (implicit) RKRD-N and explicit RKRD-N schemes are, by construction, identical.

It should be pointed out that in [7] the authors, apart from Scheme (18), presented one more formulation of the explicit $\mathcal{RKRD}$ framework: the so-called selectively lumped explicit $\mathcal{RKRD}$ scheme. The two differ only slightly, the latter being somewhat more complicated and slightly less stable (based on experimental observations). Here only the globally lumped formulation will be considered as this document is only meant to give an overview rather than a complete review of possible alternatives. Moreover, as already pointed out, between the two the globally lumped formulation is more straightforward and gives better results.

6 Non-Linear Systems of Equations

To demonstrate the robustness of the computational frameworks presented in Sects. 4 and 5 we shall employ them to solve a system of non-linear hyperbolic PDEs, namely the Euler equations of fluid dynamics. We will also use this model problem to compare the two approaches when applied in more involved scenarios.

The underlying system of equations can be written in a vector form as:

$$\partial_t \mathbf{w} + \nabla \cdot \mathbf{F} = \mathbf{0} \tag{19}$$

in which $\mathbf{w}$ is the vector of conserved variables and $\mathbf{F} = (\mathbf{g}, \mathbf{h})$ are the conservative fluxes. In the two-dimensional setting, i.e. in $\mathbb{R}^2$, these are given by:

$$\mathbf{w} = \begin{pmatrix} \rho \\ \rho u \\ \rho v \\ E_{total} \end{pmatrix}, \quad \mathbf{g} = \begin{pmatrix} \rho u \\ \rho u^2 + p \\ \rho u v \\ u(p + E_{total}) \end{pmatrix} \quad \mathbf{h} = \begin{pmatrix} \rho v \\ \rho u v \\ \rho v^2 + p \\ v(p + E_{total}) \end{pmatrix}.$$

In the above ρ is the density and u and v are the x and y components of the velocity, respectively. The total energy E_{total} is related to the other quantities by a state equation which, for a perfect gas, takes the form:

$$E_{total} = \frac{p}{\gamma - 1} + \frac{1}{2} \rho \left(u^2 + v^2\right).$$

Here γ is the ratio of specific heats (the *Poisson adiabatic constant*) and p is the pressure. Only the case of air will be considered, that is $\gamma = 1.4$.

6.1 Conservative Linearisation

In the context of residual distribution methods, the Euler equations are most frequently solved under the assumption that the "parameter vector" of Roe [37]:

$$\mathbf{z} = \begin{pmatrix} z_1 \\ z_2 \\ z_3 \\ z_4 \end{pmatrix} = \sqrt{\rho} \begin{pmatrix} 1 \\ u \\ v \\ H \end{pmatrix},$$

varies linearly within each mesh cell. In the above H stands for the total enthalpy $H = \frac{E_{total} + p}{\rho}$. The parameter vector $\mathbf{z}$ and its linearity facilitate construction of a conservative linearisation. It is a very desirable feature when dealing with hyperbolic equations as it guarantees that the position and strength of non-linear discontinuities in the solution are approximated correctly.

By analogy with the scalar case, the cell residual, Φ^E, lies at the basis of all $\mathcal{RD}$ approximations of (19):

$$\Phi^E = \int_E \nabla \cdot \mathbf{F}(\mathbf{w}_h)\, d\Omega = \oint_{\partial E} \mathbf{F}(\mathbf{w}_h) \cdot \mathbf{n}\, d\Gamma. \tag{20}$$

$\mathbf{n}$ is the outward pointing unit normal vector. In order to derive a discrete system approximating (19), one has to find an efficient and accurate way of calculating (20). Evaluating it in terms of the parameter vector gives:

$$\Phi^E = \int_E \left(\frac{\partial \mathbf{g}}{\partial \mathbf{z}} \mathbf{z}_x + \frac{\partial \mathbf{h}}{\partial \mathbf{z}} \mathbf{z}_y \right) d\Omega. \tag{21}$$

Assuming that $\mathbf{z}$ is piece-wise linear (and hence both $\mathbf{z}_x$ and $\mathbf{z}_y$ are piece-wise constant), one can further expand (21) as:

$$\Phi^E = \left(\int_E \frac{\partial \mathbf{g}}{\partial \mathbf{z}} d\Omega \right) \mathbf{z}_x + \left(\int_E \frac{\partial \mathbf{h}}{\partial \mathbf{z}} d\Omega \right) \mathbf{z}_y. \tag{22}$$

From quadratic dependence of the numerical flux on $\mathbf{z}$ (and hence the linear dependence of the flux Jacobian on it), Φ_E can be evaluated exactly using a one point quadrature rule:

$$\Phi^E = |E| \left(\frac{\partial \mathbf{g}(\bar{\mathbf{z}})}{\partial \mathbf{z}} \mathbf{z}_x + \frac{\partial \mathbf{h}(\bar{\mathbf{z}})}{\partial \mathbf{z}} \mathbf{z}_y \right) \tag{23}$$

in which $\bar{\mathbf{z}}$ is taken as the average of the values of $\mathbf{z}$ at the vertices of the corresponding triangle E:

$$\bar{\mathbf{z}} = \frac{\mathbf{z}_1 + \mathbf{z}_2 + \mathbf{z}_3}{3}, \qquad \text{with } \mathbf{z}_i = \mathbf{z}(\mathbf{x}_i) \text{ and } \mathbf{x}_i \in E. \tag{24}$$

Within each cell E, the gradient of $\mathbf{z}$ is constant. Denoting by $\mathbf{n}_i$ the unit outward pointing normal to edge $e_i \in E$ (opposite the i^{th} vertex), it can be calculated using:

$$\nabla \mathbf{z} = -\frac{1}{2|E|} |e_i| \sum_{i=1}^{3} \mathbf{z}_i\, \mathbf{n}_i.$$

Equation (23), gives a very simple formula for evaluating cell residuals, but expressed in terms of Roe's parameter vector. A similar formula in terms of the conservative variables would be more practical and natural to work with. This can be achieved by first noting that:

$$\mathbf{z}_x = \frac{\partial \mathbf{z}}{\partial \mathbf{w}} \mathbf{w}_x, \qquad \mathbf{z}_y = \frac{\partial \mathbf{z}}{\partial \mathbf{w}} \mathbf{w}_y.$$

and then showing that the averaged gradient of $\mathbf{w}$:

$$\widehat{\mathbf{w}_x} = \frac{1}{|E|} \int_E \mathbf{w}_x\, d\Omega, \qquad \widehat{\mathbf{w}_y} = \frac{1}{|E|} \int_E \mathbf{w}_y\, d\Omega$$

can be evaluated as:

$$\widehat{\mathbf{w}_x} = \frac{1}{|E|}\int\limits_E \frac{\partial \mathbf{w}}{\partial \mathbf{z}}\mathbf{z}_x\, d\Omega = \frac{1}{|E|}\int\limits_E \frac{\partial \mathbf{w}}{\partial \mathbf{z}}\, d\Omega\, \mathbf{z}_x = \frac{\partial \mathbf{w}(\bar{\mathbf{z}})}{\partial \mathbf{z}}\mathbf{z}_x,$$

$$\widehat{\mathbf{w}_y} = \frac{1}{|E|}\int\limits_E \frac{\partial \mathbf{w}}{\partial \mathbf{z}}\mathbf{z}_y\, d\Omega = \frac{1}{|E|}\int\limits_E \frac{\partial \mathbf{w}}{\partial \mathbf{z}}\, d\Omega\, \mathbf{z}_y = \frac{\partial \mathbf{w}(\bar{\mathbf{z}})}{\partial \mathbf{z}}\mathbf{z}_y.$$

It now follows that (23) is equivalent to:

$$\Phi^E = |E|\left(\frac{\partial \mathbf{g}(\bar{\mathbf{z}})}{\partial \mathbf{w}}\widehat{\mathbf{w}_x} + \frac{\partial \mathbf{h}(\bar{\mathbf{z}})}{\partial \mathbf{w}}\widehat{\mathbf{w}_y}\right), \tag{25}$$

which is the formula that we used to get our results.

The linearisation process described above shows how to evaluate the cell residuals Φ^E exactly. This means the procedure outlined here is conservative as:

$$\sum_{E\in\Omega}\Phi^E = \sum_{E\in\Omega}\oint\limits_{\partial E}\mathbf{F}_h\cdot\mathbf{n}\,d\Gamma = \oint\limits_{\Omega}\mathbf{F}_h\cdot\mathbf{n}\,d\Gamma.$$

In other words, the discrete flux balance (summed up over the whole domain) reduces to boundary contributions, even though it is evaluated numerically.

A detailed description of how to evaluate the time derivative $\partial_t\mathbf{w}$ using the hypothesis of linear $\mathbf{z}$ can be found in [5]. Here we decided to use a mixed approach and to simplify the calculations by evaluating the time derivative using the hypothesis of linear $\mathbf{w}$ rather than $\mathbf{z}$.

6.2 Matrix Distribution Schemes

Conservative linearisation discussed in the previous section is simply a tool that is implemented to calculate cell residuals when the underlying system of PDEs being solved is the Euler equations. The next step is to distribute those residuals among the vertices of the given cell *and* degrees of freedom located at each of those vertices (four unknowns per vertex in the case of two-dimensional Euler equations). To this end we implemented the so called *matrix distribution* approach devised in [1,40,41]. We note that contrary to the previous section in which the parameter vector of Roe was presented, definitions presented here are independent of the underlying system of PDEs being discretized. The only condition is that the underlying system is hyperbolic.

Matrix distribution schemes are constructed by heuristically generalising their scalar counterparts to systems of equations. Only the matrix LDA, N, and BLEND schemes will be considered here, all of which are defined with the aid of matrix flow parameters. For every cell $E\in\mathcal{T}_h$ these are defined as (cf. Eq. (10)):

$$\mathbf{K}_j = -\frac{1}{2}\left(\mathbf{A}(\bar{\mathbf{w}}), \mathbf{B}(\bar{\mathbf{w}})\right)\,\mathbf{n}_j\,|e_j|,$$

with $\bar{\mathbf{w}}$ being the cell average of $\mathbf{w}$ (cf. Eq. (24)) and $\mathbf{A}$ and $\mathbf{B}$ defined as Jacobian matrices of the fluxes:

$$\mathbf{A} = \frac{\partial \mathbf{g}}{\partial \mathbf{w}}, \qquad \mathbf{B} = \frac{\partial \mathbf{h}}{\partial \mathbf{w}}. \tag{26}$$

Vector $\mathbf{n}_j$ is the unit normal to edge e_j (opposite the j^{th} vertex) pointing outward from cell E. $|e_j|$ denotes the length of e_j. Note that this definition is consistent with the definition of scalar flow sensors. Indeed, if $\mathbf{f}$ and u from Eq. (1) are substituted into (26) then the resulting quantity will be equal to the scalar flow sensor, k_i, introduced in Sect. 4.

Since the system is hyperbolic, the matrix flow sensor admits real eigenvalues and a complete set of right and left eigenvectors. In other words, it can be diagonalised:

$$\mathbf{K}_j = \mathbf{R}_j \boldsymbol{\Lambda}_j \mathbf{R}_j^{-1},$$

with $\mathbf{R}_j$ being composed of the right eigenvectors of $\mathbf{K}_j$ and $\boldsymbol{\Lambda}_j$ containing the corresponding eigenvalues on its diagonal and zero elsewhere. These matrices can be found in, for example, Sect. 4.3.2 of the monograph by Godlewski and Raviart [24]. The authors also give a very detailed presentation of the conservative linearisation for the two-dimensional Euler equations.

Let now $\lambda_1, \lambda_2, \lambda_3$ and λ_4 denote the non-zero entries of $\boldsymbol{\Lambda}_j$ (eigenvalues of $\mathbf{K}_j$). The following matrices based on $\boldsymbol{\Lambda}_j$:

$$\boldsymbol{\Lambda}_j^+ = \operatorname{diag}\{\max(0, \lambda_k)\}_{k=1}^4, \qquad \boldsymbol{\Lambda}_j^- = \operatorname{diag}\{\min(0, \lambda_k)\}_{k=1}^4,$$

and

$$\left|\boldsymbol{\Lambda}_j\right| = \operatorname{diag}|\lambda_k|_{k=1}^4 = \boldsymbol{\Lambda}_j^+ - \boldsymbol{\Lambda}_j^-,$$

can now be used to define:

$$\mathbf{K}_j^+ = \mathbf{R}_j \boldsymbol{\Lambda}_j^+ \mathbf{R}_j^{-1}, \qquad \mathbf{K}_j^- = \mathbf{R}_j \boldsymbol{\Lambda}_j^- \mathbf{R}_j^{-1}, \qquad \left|\mathbf{K}_j\right| = \mathbf{R}_j \left|\boldsymbol{\Lambda}_j\right| \mathbf{R}_j^{-1}.$$

The above definitions are, again, consistent with the corresponding ones in the scalar case, cf. Eq. (10). It is worth recalling that for all scalar residual distribution methods/frameworks considered here, the flow sensors are evaluated using only the *previous* (already calculated) solution. This guarantees that the resulting systems of equations are linear. Matrix flow sensors are consistent with their scalar counterparts and hence a similar property holds in the case considered here. We will now present particular matrix distribution schemes.

The LDA scheme The split residuals for the matrix LDA scheme are defined as:

$$\boldsymbol{\phi}_i^{LDA} = \boldsymbol{B}_i^{LDA} \boldsymbol{\phi}^E, \qquad \boldsymbol{B}_i^{LDA} = \mathbf{K}_i^+ \mathbf{N}, \qquad \mathbf{N} = \left(\sum_{j \in E} \mathbf{K}_j^+\right)^{-1},$$

The existence of matrix product $\mathbf{K}_i^+ \mathbf{N}$ was proven in [2,4].

The N scheme The matrix N scheme is defined by:

$$\boldsymbol{\phi}_i^N = \mathbf{K}_i^+ (\mathbf{w}_i - \mathbf{w}_{in}), \qquad \mathbf{w}_{in} = -\mathbf{N} \sum_{j \in E} \mathbf{K}_j^- \mathbf{w}_j,$$

The existence of matrix $\mathbf{N}$ was proven in [2,4].

The BLEND scheme The matrix BLEND scheme is given by:

$$\boldsymbol{\phi}_i^{BLEND} = \boldsymbol{\Theta} \boldsymbol{\phi}_i^N + (\boldsymbol{I} - \boldsymbol{\Theta}) \boldsymbol{\phi}_i^{LDA},$$

with $\boldsymbol{I}$ the identity matrix. The entries of the non-linear blending matrix $\boldsymbol{\Theta}$ were computed using the following formula:

$$\boldsymbol{\Theta}_{k,k} = \frac{\left|\boldsymbol{\phi}_k^E\right|}{\sum_{i \in E}\left|\boldsymbol{\phi}_{i,k}^N\right|}. \tag{27}$$

In expression (27), index k refers to the k^{th} equation of the system, i.e. $\boldsymbol{\phi}_k^E$ and $\boldsymbol{\phi}_{i,k}^N$ are the k^{th} components of vectors $\boldsymbol{\phi}^E$ and $\boldsymbol{\phi}_i^N$, respectively [16]. Note that $\boldsymbol{\Theta}$ is a diagonal matrix. Depending on the problem being solved (smooth or exhibiting shocks), one is free to either give preference to the LDA scheme for smooth problems (set all the diagonal values to minimum), or to the N scheme for non-smooth problems (set all the diagonal values to maximum).

The mass matrix (6) for systems is derived by applying the procedure outlined in [33] to systems. Since at every vertex $i \in E$ there are four degrees of freedom, the mass matrix coefficient m_{ij}^E becomes a 4 × 4 matrix $\mathbf{M}_{ij}^E$ defined as:

$$\mathbf{M}_{ij}^E = \frac{|E|}{36}(3\delta_{ij}\boldsymbol{I} + 12\boldsymbol{B}_i^E - \boldsymbol{I}),$$

in which $\boldsymbol{B}_i^E$ is the corresponding distribution matrix and $\boldsymbol{I}$ is the identity matrix.

7 Numerical Results

In this section we will use a range of test cases, for a linear scalar equation, a non-linear scalar equation and a nonlinear system of equations, to demonstrate the accuracy and efficiency of the implicit $\mathcal{RKRD}$ schemes and compare them with the corresponding explicit schemes. Recall that we are using the terms implicit and explicit to distinguish between the two formulations, even though the Runge–Kutta time-stepping used is explicit in both cases.

We used two types of triangulations, i.e. structured (regular and isotropic) and unstructured, examples of which are illustrated in Fig. 1. Further details regarding the meshes are discussed when particular results are presented. The linear systems resulting from the implicit $\mathcal{RKRD}$ discretization were solved using PETSc [9] (see also the manual [8]) within which the ILU preconditioned GMRES solver was used. Since it gave good results, no other solver was implemented. To guarantee convergence, the relative tolerance in PETSc, i.e. the stopping criterion, was set to 10^{-8} in the case of scalar equations and to 10^{-5} for the Euler equations. Reducing it, i.e. setting to values lower than 10^{-5}, did not show any noticeable improvements (qualitative nor quantitative). However, in the case of the scalar equations the extra overhead related to setting a lower tolerance did not have a significant effect on the efficiency and we decided to run our experiments using the reduced value. The initial estimate was always set to zero.

In all computations the time-step Δt was calculated using the following formula:

$$\Delta t_i = \text{CFL}\,\frac{|S_i|}{\sum_{E \in \mathcal{D}_i}\alpha^E} \qquad \forall i \in \mathcal{T}_h.$$

The α^E coefficient is defined as:

$$\alpha^E = \frac{1}{2}\max_{j \in E}\left\|\frac{\partial \mathbf{f}(u_j)}{\partial u}\right\| h_E, \tag{28}$$

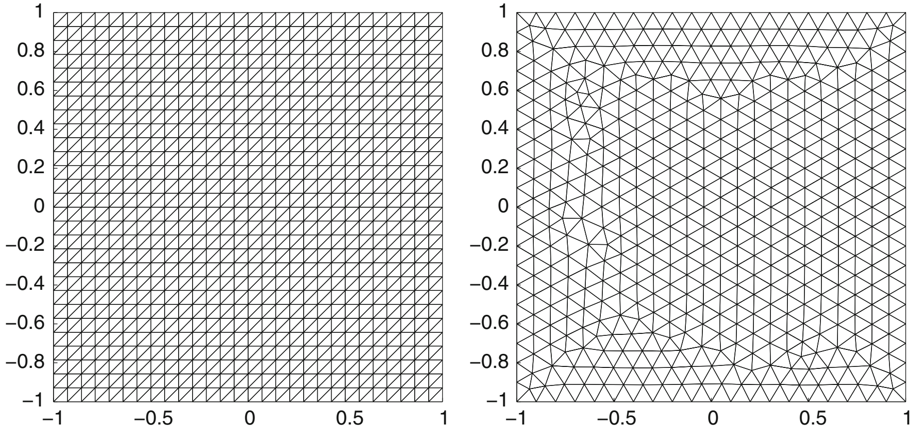


Fig. 1 Representative structured (*left*) and unstructured (*right*) grids used for transient problems

h_E stands for the reference length for element E in the scalar case, while for the Euler equations this coefficient was set to:

$$\alpha^E = \frac{1}{2} \max_{j \in E} \left(||\mathbf{u}_j|| + a_j \right).$$

h_E stands for the reference length for element E. The velocity vector $\mathbf{u}_j = (u_j, v_j)$ is evaluated at vertex $j \in E$ and the speed of sound a_j is given by:

$$a_j = \sqrt{\frac{\gamma p_j}{\rho_j}}. \tag{29}$$

The Courant-Friedrichs-Lewy (CFL) number was set to 0.9 in the case of scalar equations and between 0.1 and 0.9 in the case of the Euler equations. Precise values are given when the corresponding results are presented.

7.1 Scalar Equations

Three distinct scalar test problems were implemented. Test Cases A and B are linear equations with *smooth* initial conditions which were used to measure convergence rates. Test Case C is a non-linear equation with a piece-wise constant initial condition, the solution to which exhibits shocks and rarefaction waves. It was employed to investigate positivity. In all experiments, the final time was set as:

- $T = 1$ for Test Cases A and C;
- $T = \frac{\pi}{2}$ for Test Case B.

Test Case A: The *constant advection equation* given by

$$\partial_t u + \mathbf{a} \cdot \nabla u = 0 \qquad \text{on} \quad \Omega_t = \Omega \times [0, 1]$$

with $\Omega = [-1, 1] \times [-1, 1]$ and $\mathbf{a} = (1, 0)$. The exact solution to this problem (which was also used to specify the initial condition at $t = 0$) is given by

$$u(\mathbf{x}, t) = \begin{cases} z^5 \left(70z^4 - 315z^3 + 540z^2 - 420z + 126 \right) & \text{if } r < 0.4, \\ 0 & \text{otherwise} \end{cases}$$

in which $r = \sqrt{(x + 0.5 - t)^2 + y^2}$, $z = -\frac{r-0.4}{0.4}$ and $\mathbf{x} = (x, y)$. Note that this function is $C^4(\Omega)$ regular. The boundary conditions were set to

$$u(\mathbf{x}, t) = 0 \qquad \text{on } \partial\Omega.$$

Note that for structured grids the advection velocity given above is aligned with the mesh.

Test Case B: The rotational advection equation, given by:

$$\partial_t u + \mathbf{a} \cdot \nabla u = 0 \qquad \text{on} \quad \Omega_t = \Omega \times [0, \frac{\pi}{2}]$$

with $\Omega = [-1, 1] \times [-1, 1]$ and $\mathbf{a} = (-y, x)$. The exact solution to this problem (which was also used to specify the initial condition at $t = 0$) is given by

$$u(\mathbf{x}, t) = \begin{cases} z^5(70z^4 - 315z^3 + 540z^2 - 420z + 126) & \text{if } r < 0.4, \\ 0 & \text{otherwise} \end{cases}$$

where $r = \sqrt{(x - x_c)^2 + (y - y_c)^2}$ and

$$z = -\frac{r - 0.4}{0.4}, \quad x_c = \frac{1}{2}\cos\left(t - \frac{\pi}{2}\right), \quad y_c = \frac{1}{2}\cos\left(t - \frac{\pi}{2}\right).$$

The boundary conditions were set to:

$$u(\mathbf{x}, t) = 0 \qquad \text{on } \partial\Omega.$$

Contrary to Test Case A, here the advection velocity is generally not aligned with the mesh. This test case is used to make sure that results obtained for Test Case A are not biased by the direction of the flow.

Test Case C: The inviscid Burgers' equation is given by:

$$\partial_t u + \nabla \cdot \mathbf{f}(u) = 0 \qquad \text{on} \quad \Omega_t = \Omega \times [0, 1]$$

with $\mathbf{f} = (\frac{u^2}{2}, \frac{u^2}{2})$. As for Test Cases A and B, the spatial domain is a square: $\Omega = [-1, 1] \times [-1, 1]$. The initial condition was set to be piece-wise constant:

$$u(\mathbf{x}, 0) = \begin{cases} 1 & \text{if } \mathbf{x} \in [-0.6, -0.1] \times [-0.5, 0] \\ 0 & \text{otherwise} \end{cases}$$

The boundary conditions were set to:

$$u(\mathbf{x}, t) = 0 \qquad \text{on } \partial\Omega.$$

The solution to this problem is discontinuous and exhibits rarefaction and shock waves [31,32] and was used to test for positivity.

Linear equations (Test Case A and B) were solved on structured grids. To demonstrate robustness of the methods discussed here, in particular to show that they can be used with both structured and unstructured discretisations of the domain, an unstructured mesh with 26,054 elements (topology similar to that on the right of Fig. 1) was used in the case of the non-linear Burgers' equation.

The grid convergence analysis confirmed that the implicit/explicit RKRD-N scheme (recall that this scheme is the same in both frameworks) is only first order accurate, whereas both forms of RKRD-LDA, RKRD-SU and RKRD-BLEND scheme exhibit convergence of order

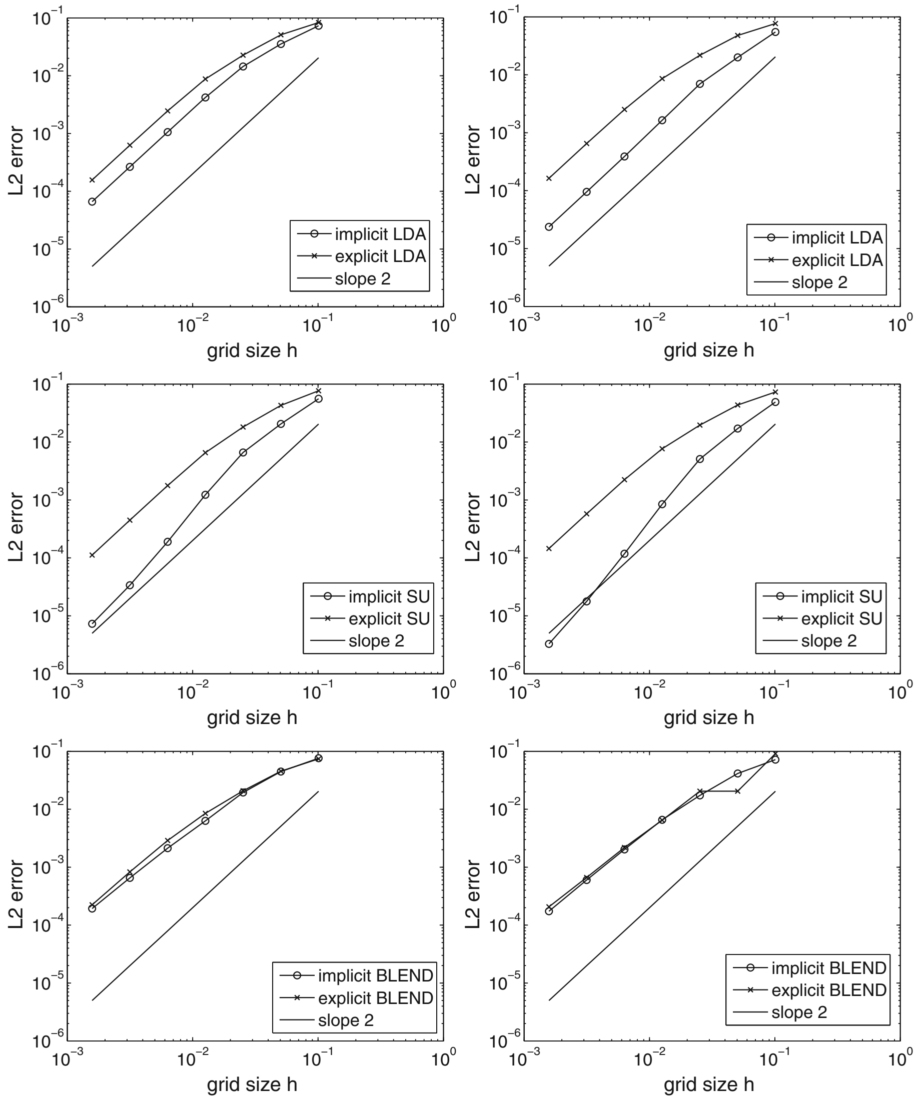
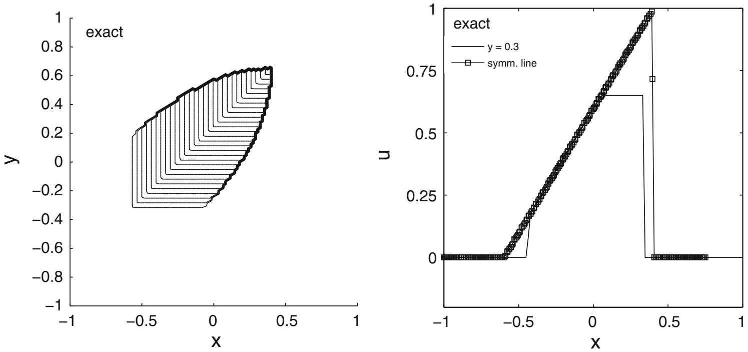


Fig. 2 Grid convergence in the L^2 norm for the implicit and explicit RKRD-LDA (*top*), RKRD-SU (*middle*) and RKRD-BLEND (*bottom*) schemes. The *errors* shown are for Test Cases A (*left*) and B (*right*)

two. The behaviour of the error in the second order schemes as the mesh is refined is presented in Fig. 2. The LDA and SU schemes gave best results, the SU scheme being noticeably more accurate than the LDA scheme. The BLEND scheme is slightly less accurate than both of them. This is most likely due to its use of a non-linear combination of first and second order schemes. In terms of accuracy on a given mesh, the implicit RKRD scheme outperformed its explicit counterpart. The difference is relatively small in the case of the non-linear BLEND scheme, but becomes more noticable when moving to the linear distributions. In the case of the SU scheme it was over an order of magnitude.

These experiments were carried out on a set of regular triangular meshes (topology as on the left of Fig. 1) with the coarsest mesh of a 14 × 14 regular grid refined uniformly 6 times by



u_{min}	u_{max}
0.0	1.0

Fig. 3 2d Burgers' equation: the analytical solution. *Left*: contours at time $t = 1$. *Middle*: solution along the line $y = 0.3$ and along the symmetry line, $y = x + 0.1$. *Right*: minimum and maximum values of the solution

a factor 2 in each direction. The accuracy was monitored by the convergence of the L^2 norm of error at the final time of the simulation with respect to the exact solution. The behaviour of the L^1 and L^∞ norms was qualitatively and quantitatively very similar. Carrying out a similar grid refinement study on unstructured meshes led to qualitatively similar results.

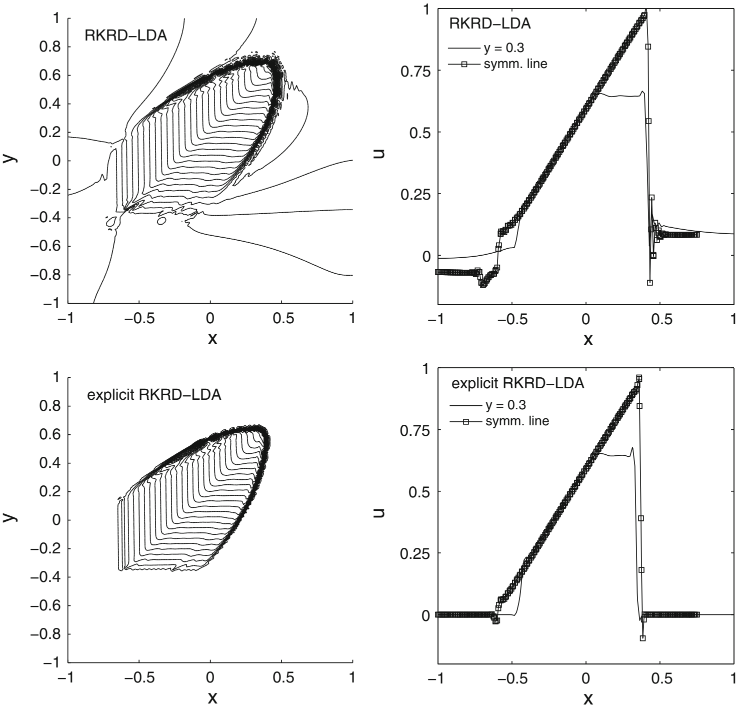
Concerning Test Case C, i.e. Burgers' equation, Fig. 3 shows the contours and cross sections (along the symmetry line $y - x = 0.1$ and $y = 0.3$) of the exact solution [43]. The maximum and minimum values of the profile are written next to the plots. Similar plots and quantities are given for the approximate solutions obtained with the second order schemes in both $\mathcal{RKRD}$ frameworks, see Figs. 4, 5 and 6.

As expected, the N scheme gave a solution free of spurious oscillations (it is positive), though more diffusive than schemes for which results are shown. The solution obtained with the aid of the LDA scheme, Fig. 4, exhibits oscillations near discontinuities (again, as expected). Compared to the explicit $\mathcal{RKRD}$ approach, these oscillations are much more pronounced when the implicit $\mathcal{RKRD}$ framework is used, suggesting a reduction in the numerical dissipation incurred. Other schemes showed far less sensitivity to whether the $\mathcal{RKRD}$ discretization was explicit or implicit. The implicit and explicit RKRD-BLEND schemes performed much better than the implicit and explicit RKRD-LDA schemes, respectively, in the sense that blending smooths the solutions and the resulting approximations have smaller under/over-shoots. Although less diffusive then the N scheme, The BLEND scheme is less diffusive than the N scheme, but not completely oscillation-free. To summarise, the BLEND scheme gives the best trade-off between being oscillation-free and second order accurate. In terms of accuracy the implicit framework is more accurate, but more prone to numerical oscillations than its explicit counterpart.

Finally, we want to comment on scaling and performance of the linear solver that was applied to solve linear systems resulting from the implicit $\mathcal{RKRD}$ discretization. As mentioned earlier, only GMRES preconditioned with ILU was used. To guarantee convergence, the linear solver was set to iterate until the relative tolerance,

$$r_{tol} = \frac{||r||_{l^2}}{||b||_{l^2}},$$

reached 10^{-8}. In the above, r is the current residual and b is the right-hand-side vector (since the initial estimate was set to zero, b is also the initial residual). For all test cases and for all schemes the linear solver converged, on average, in less than 10 iterations, with the final



u_{min}	u_{max}
-0.181	1.041

u_{min}	u_{max}
-0.141	0.961

Fig. 4 2d Burgers' equation: implicit (*top*) and explicit (*bottom*) RKRD-LDA schemes. *Left*: contours at time $t = 1$. *Middle*: solution along line $y = 0.3$ and along the symmetry line, $y = x + 0.1$. *Right*: minimum and maximum values of the solution

residual equal to roughly 10^{-11}. Some sample results are given in Table 1. The behaviour of the iterative solver when the BLEND scheme is used may seem odd as the number of iterations needed for convergence for the first and the second stage of the Runge–Kutta time-stepping differs by around 100 %. This is due to the fact that during the first stage the blending parameter picks the first order N scheme most of the time and the system of equations is very close to a diagonal matrix. The opposite situation is taking place during the second stage.

7.2 The Euler Equations

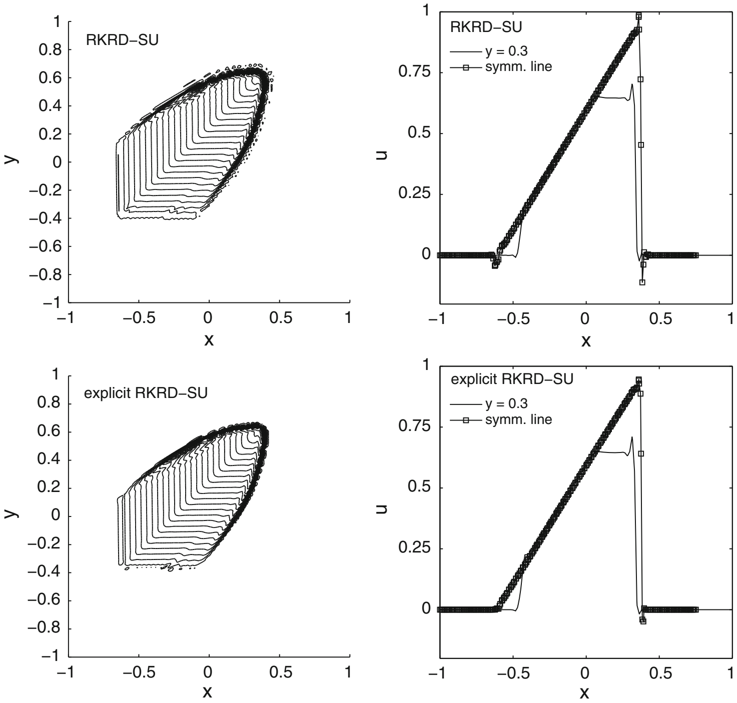
In the case of the Euler equations three distinct test problems were implemented:

- Double Mach Reflection (the solution exhibits strong shocks);
- Mach 3 Flow Over a Step (the solution exhibits strong shocks);
- Advection of a Vortex (the analytic solution is C^2 regular).

Shocks appearing in the Double Mach Reflection and Mach 3 test cases were too strong for the LDA scheme to cope with. This being the case, only the RKRD-BLEND scheme was considered in these cases.

Double Mach Reflection

This problem was originally introduced by Woodward et al. in [44]. It constitutes a very severe test for the robustness of schemes designed to compute discontinuous flows. The flow



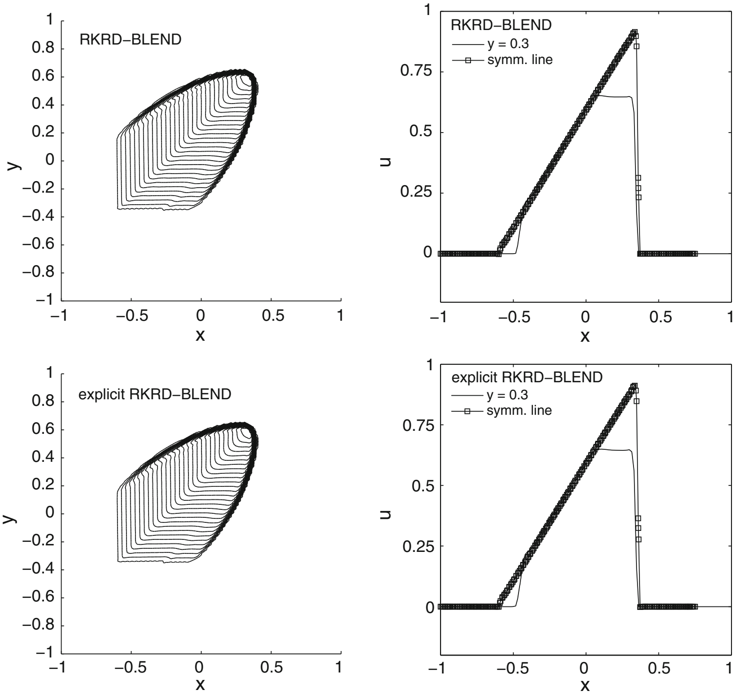
u_{min}	u_{max}
-0.167	1.058

u_{min}	u_{max}
-0.140	1.017

Fig. 5 2d Burgers' equation: implicit (*top*) and explicit (*bottom*) RKRD-SU schemes. *Left*: contours at time $t = 1$. *Middle*: solution along line $y = 0.3$ and along the symmetry line, $y = x + 0.1$. *Right*: minimum and maximum values of the solution

consists of the interaction of a planar right-moving Mach 10 shock with a 30° ramp. In the frame of reference used, the x axis is aligned with the ramp. The computational domain is the rectangle $[0, 4] \times [0, 1]$, with the ramp starting at $x = \frac{1}{6}$ and stretching till the right-hand-side corner of the domain ($x = 4$, $y = 0$). The simulations were run until time $T = 0.2$ on unstructured meshes with topology similar to that in Fig. 7. At the initial state, the shock forms a 60° angle with the x axis. See Fig. 8 for the geometry and initial values of the solution. The CFL number was set to 0.9.

For this test case it is customary to plot contours of the density field. These are presented in Fig. 9. Only the region between $x = 0$ and $x = 3$ is displayed, although the grid continues to $x = 4$. The air ahead of the shock remains undisturbed and the shorter domain makes the presentation clearer. All the considered schemes successfully captured the interaction between the shock and the ramp (see [7,13] and [44] for reference results). As expected, refining the mesh increased the resolution and the accuracy with which that interaction was resolved. In all cases, the BLEND scheme gave a solution exhibiting higher resolution and thus capturing the shocks more accurately than the N scheme (results not shown). The coarsest mesh was insufficient to capture the contact emanating from the triple point and refining it led to a significant improvement. The explicit RKRD-BLEND and implicit RKRD-BLEND schemes, gave comparable results, shown in Fig. 9, however the one obtained with the aid of the explicit RKRD-BLEND scheme is of noticeably higher resolution. This is probably due to the fact that in the case of the implicit RKRD-BLEND scheme values on the diagonal of the blending matrix $\mathbf{\Theta}$ (cf. Eq. (27)) were set



u_{min}	u_{max}
-0.017	0.915

u_{min}	u_{max}
-0.011	0.914

Fig. 6 2d Burgers' equation: implicit (*top*) and explicit (*bottom*) RKRD-BLEND schemes. *Left*: contours at time $t = 1$. *Middle*: solution along line $y = 0.3$ and along the symmetry line, $y = x + 0.1$. *Right*: minimum and maximum values of the solution

Table 1 Performance of the GMRES solver when applied to the linear systems resulting from the second order $\mathcal{RKRD}$ discretisations (Test Case B)

		6,272	25,088	100,352	401,408	1,605,632
LDA	GMRES iter.	8.52/8.52	7.95/7.95	7.76/7.76	7.74/7.74	7.63/7.63
	$\|\|r_F\|\|_2$	1.39e−10	1.9e−11	1.92e−11	4.8e−12	6.55e−12
BLEND	GMRES iter.	4.30/7.56	4.44/8.21	4.33/8.68	3.29/7.82	4.27/8.57
	$\|\|r_F\|\|_2$	9.84e−11	2.09e−11	2.18e−11	1.03e−11	1.03e−11
SU	GMRES iter.	7.76/7.78	6.41/6.41	6/6	5.88/5.88	5.87/5.87
	$\|\|r_F\|\|_2$	2.4e−11	8.78e−11	8.61e−12	1.59e−12	5.13e−13

The table shows the average number of iterations it took to reach the stopping criterion during the first/second stage of the Runge–Kutta time-stepping and the l_2 norm of the final residual (when GMRES converged at the final time-step) at the second stage of the Runge–Kutta time-stepping (denoted by $||r_F||_2$). Results are given for the meshes used earlier in the grid convergence analysis (with 6,272, 25,088, 100,352, 401,408 and 1,605,632 elements, cf. top row of the table)

to the maximum value (i.e. the preference was given to the first order N scheme). Otherwise, instabilities would stop the algorithm from completing the simulation. The results in Fig. 9 are comparable with those obtained in [13] and [44] on meshes with similar resolution.

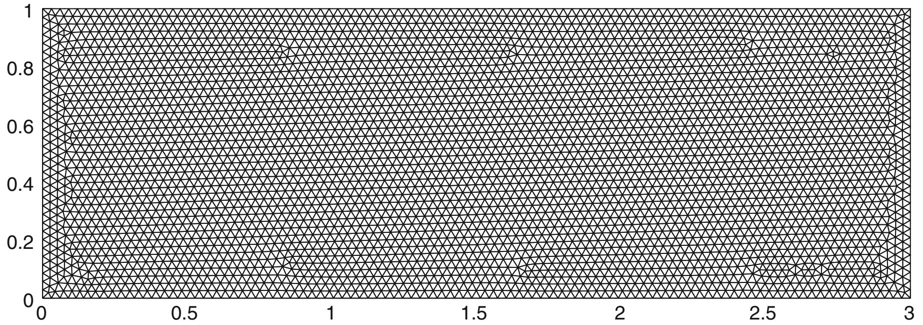


Fig. 7 The coarsest grid for the Double Mach Reflection test case, 7,865 cells

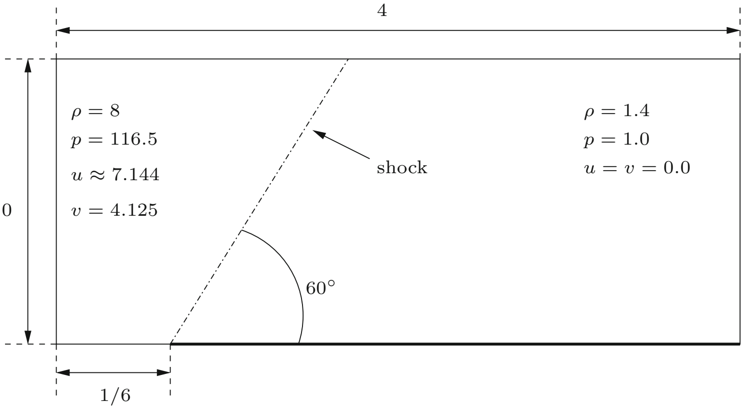


Fig. 8 The geometry and initial condition for the Double Mach Reflection test case

Mach 3 Flow Over a Step This test was originally introduced in the paper by Emery [23] and more recently reviewed by Woodward et al. in [44]. The problem begins with uniform Mach 3 flow in a wind tunnel containing a step. The wind tunnel is 1 length unit wide and 3 length units long. The step is 0.2 length units high and is located 0.6 length units from the left-hand end of the tunnel (see Fig. 10 for the geometry and the initial condition). The inflow and outflow conditions are prescribed at the left and right boundaries ($y = 0.0$ and $y = 3.0$), respectively. The exit boundary condition has no effect on the flow, because the exit velocity is always supersonic. Initially the wind tunnel is filled with a gas, which everywhere has density 1.4, pressure 1.0, and velocity $(3, 0)$. Gas with this density, pressure, and velocity is continually fed in from the left-hand boundary. Along the walls of the tunnel reflecting boundary conditions are applied. The corner of the step is the centre of a rarefaction fan and hence is a singular point of the flow. Following Woodward and Colella [44], in order to minimize numerical errors generated at this singularity, additional boundary conditions near the corner of the step were prescribed. For every boundary cell E located behind the step and such that, $\forall x \in E,\ 0.6 \leq x \leq 0.6125$, all the variables were reset to their initial values. This condition is based on the assumption of a nearly steady flow in the region near the corner. The simulations were carried out on an unstructured mesh with 71,080 nodes with the reference length set to approximately $\frac{1}{80}$ at the beginning and the end of the domain and $\frac{1}{1{,}000}$ at the corner of the step. The zoom of the mesh near the singularity point is illustrated in Fig. 11.

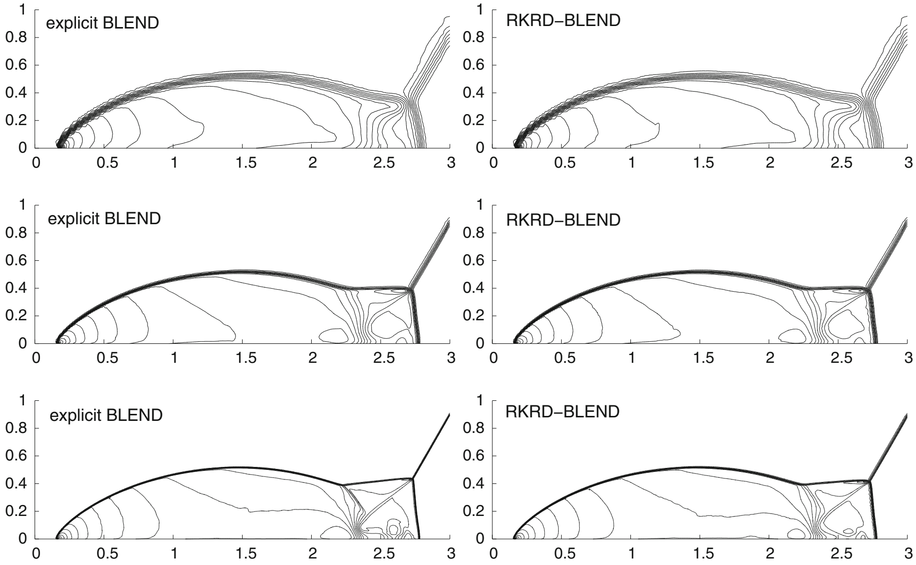


Fig. 9 Double Mach reflection: density contours for the explicit (*left*) and implicit (*right*) RKRD-BLEND schemes for meshes of 7,865 cells, for which the mesh referenbce size is $h = 1/30$ (*top*), 55,927 cells, for which the reference mesh size is $h = 1/80$ (*middle*) and 278,141 cells, for which the reference mesh size is $h = 1/190$ (*bottom*)

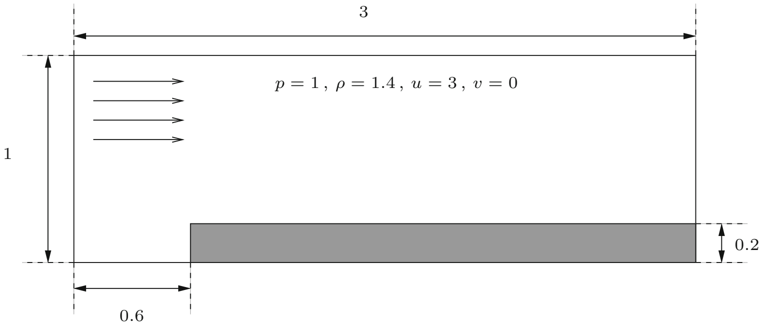


Fig. 10 Geometry and the initial condition for the Mach 3 test case

The CFL number was set to 0.8 for the explicit framework and 0.5 for the implicit approach. In the latter case, when the CFL was set to 0.8, negative densities appeared in the numerical solution at $t \approx 2.35$ since the implicit scheme generates slightly lower numerical dissipation (as noted when approximating the scalar equations).

Density contours at times $t = 0.5$, $t = 1.5$ and $t = 4.0$ obtained with the aid of explicit RKRD-BLEND and implicit RKRD-BLEND schemes are plotted in Fig. 12. All the figures show a sharp resolution of the shocks and are comparable to results that one can find in the literature obtained on meshes with similar resolution (see, for instance, [7,26] and [15]). The implicit RKRD-BLEND scheme captured the Mach stem more accurately, see Fig. 12. In both the implicit and explicit case the values on the diagonal of the blending matrix $\boldsymbol{\Theta}$ (cf. Eq. (27)) were set to maximum (i.e. the preference was given to the first order N scheme). Otherwise, instabilities close to the corner of the step would prevent the algorithms from completing the simulations.

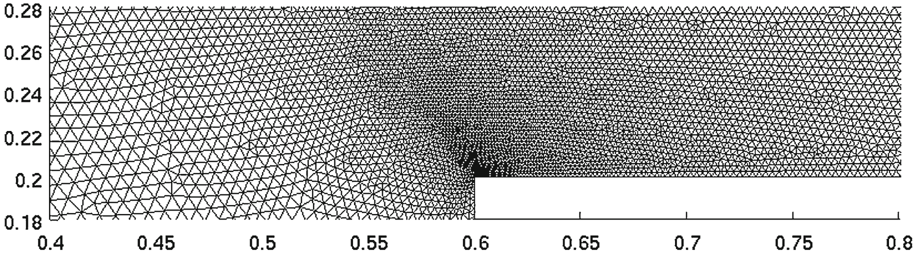


Fig. 11 The zoom of the grid used for the Mach 3 Flow Over a Step test case near the singularity point

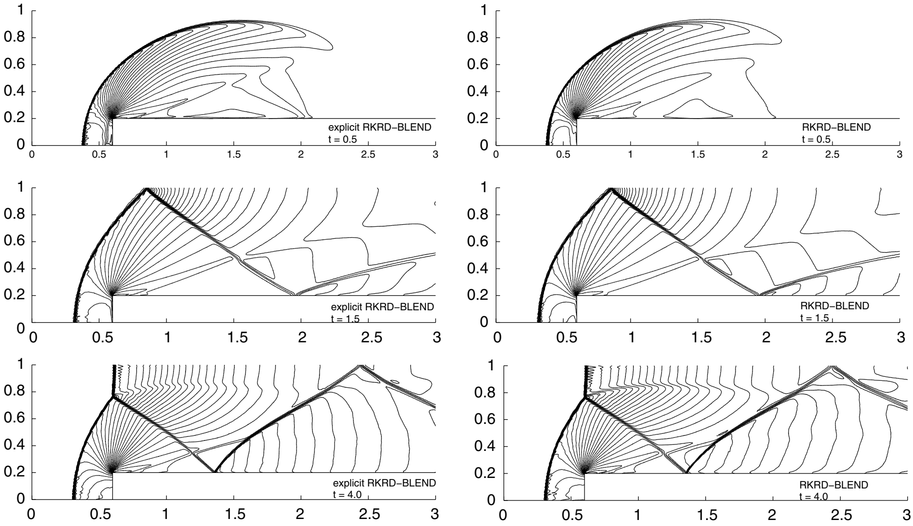


Fig. 12 Mach 3 flow over a step: density contours for the explicit (*left*) and implicit (*right*) RKRD-BLEND schemes at time $t = 0.5$ (*top*), $t = 1.5$ (*middle*) and $t = 4.0$ (*bottom*)

Advection of a Vortex The following problem was originally introduced in [22]. Its main appeal is the fact that the exact solution to this test case is known. The problem was solved on a rectangular domain $[0, 2] \times [0, 1]$ with an inflow boundary on its left side ($x = 0.0$), outflow at the right end of the domain ($x = 2$) and solid wall boundary conditions at the bottom and the top. The density for this test was constant and set to $\rho = 1.4$ throughout the domain. The centre of the vortex, (x_c, y_c), was initially set to $(0.5, 0.5)$ and was then advected during the simulation with the mean stream velocity $\mathbf{v}_m = (6, 0)$. The flow velocity was given by the mean $\mathbf{v}_m$ and the circumferential perturbation, i.e. $\mathbf{v} = \mathbf{v}_m + \mathbf{v}_p$, with:

$$\mathbf{v}_p = \begin{cases} 15\ (\cos(4\pi r) + 1)\ (-y, x) & \text{for } r < 0.25, \\ (0, 0) & \text{otherwise,} \end{cases}$$

with $r = \sqrt{(x - x_c)^2 + (y - y_c)^2}$. The pressure, similarly to the velocity vector, was given by its mean value $p_m = 100$ plus perturbation, i.e. $p = p_m + p_p$:

$$p_p = \begin{cases} \Delta p + C & \text{for } r < 0.25, \\ 0 & \text{otherwise,} \end{cases}$$

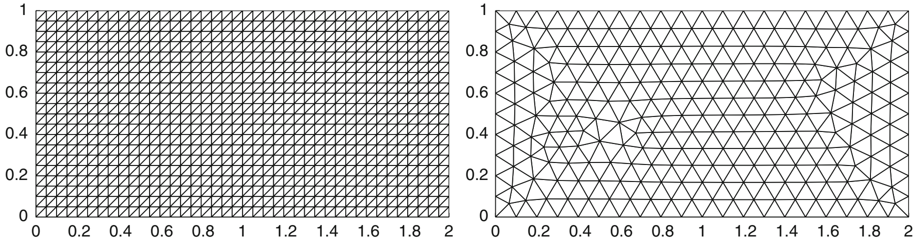


Fig. 13 The coarsest structured (*left*) and unstructured (*right*) grid used in the grid convergence analysis for the Advection of a Vortex test case

Table 2 The minimum and maximum values of the pressure obtained with the aid of the LDA, N and BLEND schemes using the explicit (ex) and implicit (im) $\mathcal{RKRD}$ frameworks

Scheme	N	ex BLEND	im BLEND	ex LDA	im LDA	exact
p_{min}	98.77133	94.11941	93.5180	93.06300	92.90018	93.213
p_{max}	100.1191	100.1159	100.0004	100.0766	100.0803	100

with $\Delta p + C$ defined so that the solution is C^2 regular:

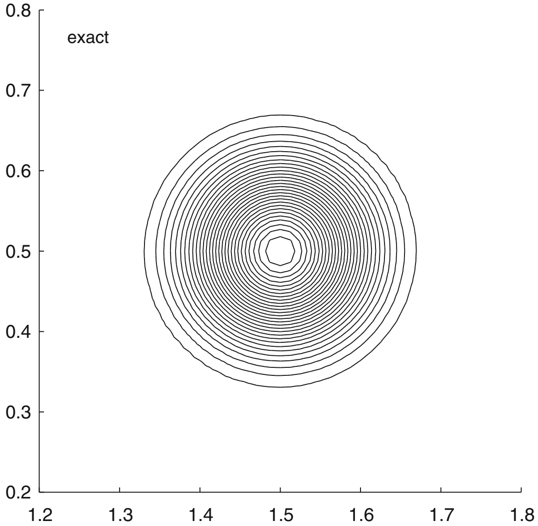
$$\Delta p = \frac{15^2\rho}{(4\pi)^2}\left(2\cos(4\pi r) + 8\pi r \sin(4\pi r) + \frac{\cos(8\pi r)}{8} + \pi r \sin(8\pi r) + 12\pi^2 r^2\right).$$

The regularity is guaranteed by choosing C such that $p|_{r=0.25} = p_m = 100$. With the above setup the maximal Mach number in the domain is $M = 0.8$. The simulation was run until time $T = \frac{1}{6}$.

The first set of experiments was carried out on a structured mesh with topology as in Fig. 13 with 161×81 nodes. The computations were performed with $CFL = 0.8$. In Table 2 values of the maximum and the minimum values of the pressure obtained are given. Isolines of the pressure inside and in the close vicinity of the vortex are shown in Fig. 15. The N scheme was also used and gave the most smeared out and the least accurate results of all the schemes (and so are not shown here). The minimum value of the solution in this case is much higher than the exact one. The solutions produced with the explicit RKRD-BLEND scheme was much better in this respect, however noticeably worse than those obtained with the (implicit and explicit) RKRD-LDA and implicit RKRD-BLEND schemes. The solution obtained with the implicit RKRD-BLEND scheme most closely resembles the exact solution, shown in Fig. 14. No clear superiority of either the explicit or implicit frameworks was noticed, though the implicit RKRD-BLEND scheme gave a somewhat smoother solution than its explicit counterpart. It should be noted, though, that in this section the implicit RKRD-BLEND scheme was set in such a way that the preference was given to the LDA scheme (cf. Sect. 6.2). In [22] similar experiments for this test problem were carried out (i.e. investigation of contour plots and the maximum/minimum values of the numerical solutions). Values presented in Table 2 show similar behaviour, but contour plots presented here (in particular those obtained with the aid of explicit and implicit RKRD-LDA schemes and implicit RKRD-BLEND) are much more faithful to the exact solution than those presented in the literature [22].

The grid convergence analysis was performed to investigate the order of accuracy of the implicit/explicit RKRD-LDA and RKRD-BLEND schemes. Errors were measured by means of the usual L^∞ norm and the L^2 and L^1 norms of the relative pressure error:

Fig. 14 Travelling Vortex: pressure contours for the exact solution, 25,600 cells



$$\epsilon_p = \left| \frac{p^{exact} - p^{approx}}{p_m} \right|,$$

in which p^{exact} and p^{approx} are the values of the analytical and numerical (approximate) pressure, respectively. Instead of calculating the error in the whole domain, only nodes inside and in the close vicinity of the vortex, i.e. nodes for which:

$$r = \sqrt{(x - x_c)^2 + (y - y_c)^2} \leq 0.35$$

were considered. This approach guaranteed that there was no interference between boundary and interior nodes and that the imposition of boundary conditions did not affect the results. The experiments were performed on a set of structured and unstructured meshes (topology as in Fig. 13), for which the reference length was varied from approximately $\frac{1}{10}$ to $\frac{1}{160}$ in the case of unstructured meshes and from $\frac{1}{20}$ to $\frac{1}{320}$ in the case of structured grids. The CFL number was reduced to 0.4 and 0.1 for the explicit and implicit $\mathcal{RKRD}$ frameworks, respectively. Recall that it was set to 0.8 to produce the contour plots, i.e. Fig. 15. Such a modification was necessary in order to demonstrate the accuracy for the coarsest meshes and to obtain results exhibiting second order convergence. The simulations were run until time $T = 0.08$ rather than $T = \frac{1}{6}$ (i.e. making the vortex travel from $(0.5, 0.5)$ to $(0.98, 0.5)$ instead of $(1.5, 0.5)$ and reducing the influence of the boundaries). The results for both the explicit and implicit $\mathcal{RKRD}$ frameworks on unstructured meshes are presented in Fig. 16. The results obtained on structured meshes are similar. In all cases second order accuracy is clearly seen in the L^2 norm of the error. It is only partially observed in the L^∞ norm, which is less forgiving. In [7] the authors presented errors only in the L^2 norm (comparable to those obtained here) claiming that the behaviour of their schemes (i.e. the explicit $\mathcal{RKRD}$ framework) is quantitatively and qualitatively very similar in all three norms considered: L^2, L^1 and L^∞. However, the configuration they used was somewhat different, i.e. periodic boundary conditions and a shorter domain were used. Both the implicit and explicit RKRD-LDA scheme exhibited a small drop in the order of accuracy when moving to the finest meshes. For all the experiments except those on the coarsest meshes, the implicit framework was more accurate then its explicit counterpart.

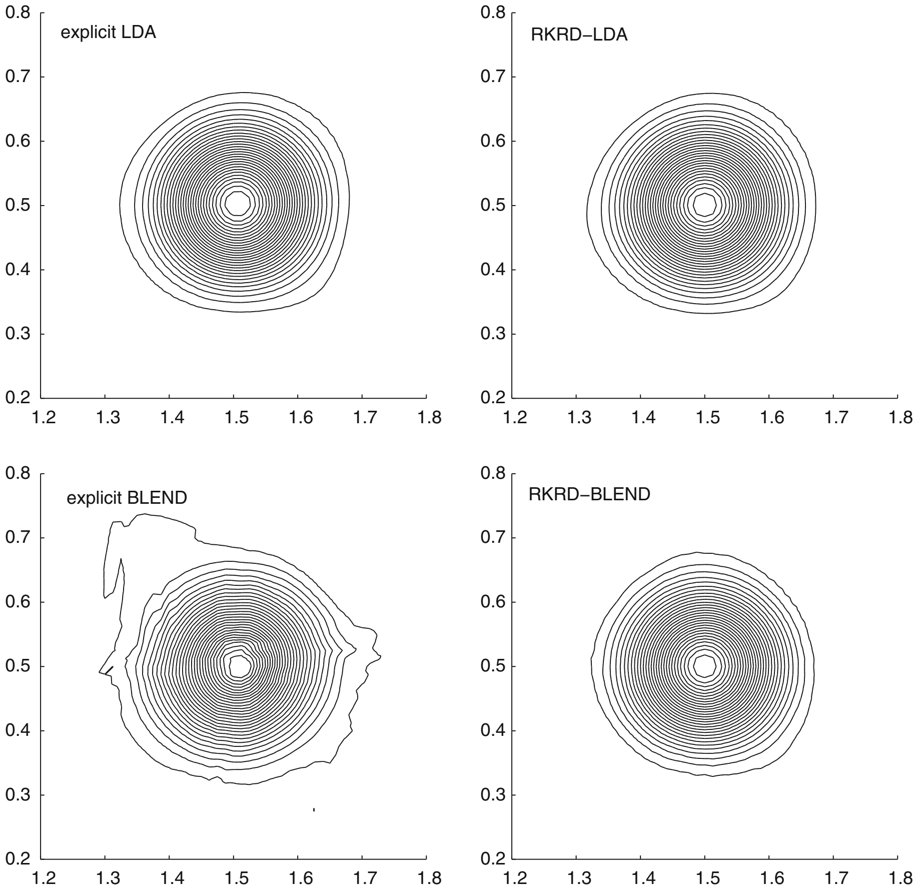


Fig. 15 Travelling vortex: pressure contours, on a mesh of 25,600 cells, for the explicit (*left*) and implicit (*right*) RKRD-LDA (*top*) and RKRD-BLEND (*bottom*) schemes

To investigate the overhead related to solving linear systems (and using PETSc) in the case of the implicit framework, selected execution times for the implicit and explicit frameworks are presented in Table 3. The *clock()* function from the C Programming Language Standard Library was used. The overhead that the implicit framework introduced is strictly related to solving two linear systems at each time-step: one at each stage of the Runge–Kutta time-stepping. This includes allocating the memory for the linear system, M (cf. Formulation (9)), assembling it and finally inverting to get the solution at the next time-step. All of these tasks were performed in one update procedure, all other parts of the code being shared between the implicit and explicit frameworks. In the case of the explicit $\mathcal{RKRD}$ framework, the linear system M_{ex} (cf. Formulation (18)) is diagonal and one can immediately calculate the solution at the new time-step, i.e. $\mathbf{w}^{n+1}$, based on the solution at the current time, i.e. $\mathbf{w}^{n}$. Table 3 contains two sets of times (evaluated on five consecutively refined meshes) for the implicit and explicit RKRD-LDA schemes, chosen to represent the implicit and explicit $\mathcal{RKRD}$ frameworks, respectively. The first value (Time 1) represents the amount of time (in seconds) taken for one time-step (two Runge–Kutta stages), i.e. the whole process of calculating $\mathbf{w}^{n+1}$ based on $\mathbf{w}^{n}$. The second value (Time 2) represents the time taken for one update procedure

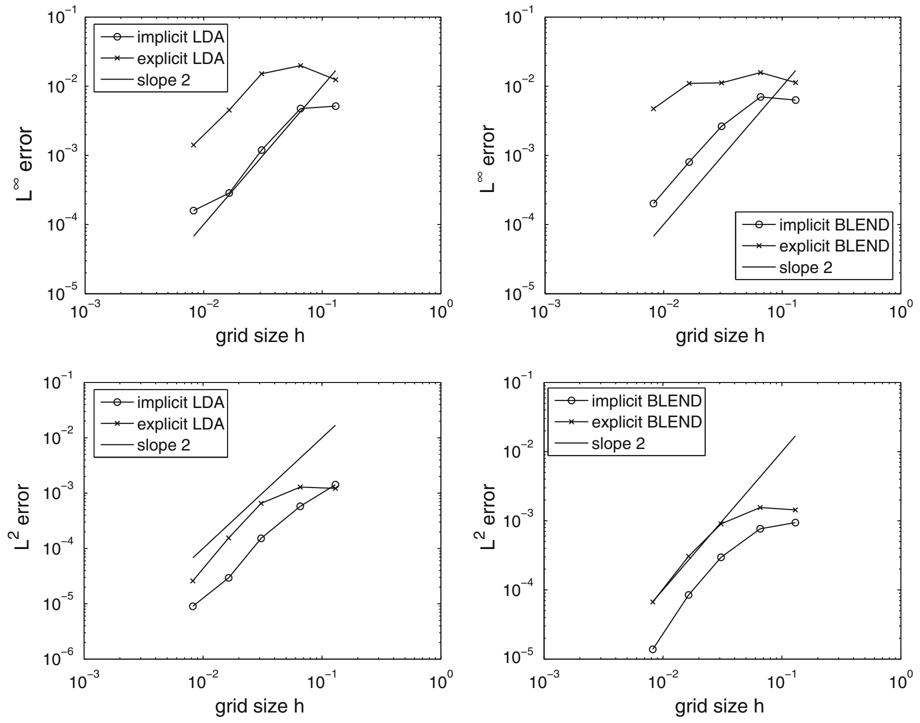


Fig. 16 Grid convergence for the implicit and explicit RKRD schemes ($CFL = 0.1$) schemes for the Advection of a Vortex test case. *Errors* calculated within a sub-domain surrounding the vortex at $T = 0.08$. Plots shown for the RKRD-LDA (*left*) and RKRD-BLEND (*right*) schemes with errors in the L^∞ (*top*) and L^2 (*bottom*) norms

Table 3 Performance of the implicit (im) and explicit (ex) RKRD-LDA methods when applied to the Advection of a Vortex test case

#Cells		474	1,856	7,374	29,656	118,522
im LDA	GMRES iter.	9.90/9.90	9.85/9.85	9.98/9.98	7.16/7.16	10.41/10.41
	L^2 Error	1.4219e−03	5.7804e−04	1.5207e−04	2.9426e−05	9.0165e−06
	Time 1	0.120e−2	4.709e−2	1.946e−1	7.359e−1	3.305
	Time 2	4.662e−3	1.818e−2	7.592e−2	2.822e−1	1.309
ex LDA	L^2 Error	1.2164e−03	1.2843e−03	6.5866e−04	1.5420e−04	2.5926e−05
	Time 1	2.628e−03	1.087e−02	4.3921e−02	1.760e−01	7.072e−01
	Time 2	0.0	1.7e−5	1.720e−04	5.08e−04	1.885e−03

The table shows (1) the average number of iterations taken to reach the stopping criterion during the first/second stage of the Runge–Kutta time-stepping (the implicit scheme only), (2) L^2 errors and (3) the amount of time (in seconds) for: one time-step (both stages, Time 1) and the update procedure (setting and solving the linear system for one stage, Time 2). Results are given for the unstructured meshes used in the grid convergence analysis (with 474, 1,856, 7,374, 29,656 and 118,522 cells, cf. top row of the table and Fig. 16)

(within one Runge–Kutta stage), that is creating and inverting the mass matrix, M, and then using it and $\mathbf{w}^n$ to calculate $\mathbf{w}^{n+1}$. In the case of the explicit $\mathcal{RKRD}$ framework the mass matrix is diagonal and hence its assembly and inversion is simple. The implicit RKRD-LDA scheme takes, on average, approximately four times longer to obtain the desired solution. For both schemes the execution time increases by a factor of four when the mesh is refined. We emphasise that the above is not an attempt to perform a thorough profiling or comparison of the implicit and explicit frameworks. Results from Table 3 are presented here to draw a general picture and to serve as guidance when considering these schemes in future. The implementation has not been optimised and the tolerance specified to signify convergence of the GMRES iteration in PETSc could be relaxed without significant impact on the accuracy of the results. Also, more efficient techniques than GMRES might be considered, *e.g.* a direct solver might be faster for 2D simulations on these meshes. The investigation of such issues is also the subject of future work.

Experiments were performed on a Desktop PC equipped with an HT Intel Xeon core and twelve gigabytes of operating memory. All presented execution times are averages calculated during the corresponding simulation (the Advection of a Vortex test case, simulation run until time $T = 0.08$). The results show that the implicit RKRD-LDA, even though in most cases it is more accurate than its explicit counterpart, is relatively slow compared to the explicit RKRD-LDA scheme. A more thorough and extensive study would require optimisation of both implementations.

8 Conclusions

In this paper we introduced a new class of numerical approximations to time-dependent hyperbolic PDEs, namely the framework of Runge–Kutta residual distribution methods. The proposed framework facilitates construction of second order accurate schemes and this was confirmed experimentally. The non-linear RKRD-BLEND scheme, although not completely oscillation-free, gave very encouraging results in terms of monotonicity. In particular it coped well with all the severe test cases based on the system of non-linear Euler equations.

A thorough comparison with the explicit $\mathcal{RKRD}$ schemes of Ricchiuto and Abgrall [7] was carried out, in which it was shown that the new framework gives significantly more accurate results on the same meshes. However, this is counterbalanced by the additional work required to invert the mass matrices at each time-step and the reduced stability of the schemes in the vicinity of strong shocks. Whether the new schemes are more efficient or not depends on the magnitude of the improvement in accuracy and is problem-dependent.

Future work will include incorporating discontinuous-in-space data representation following the methodology of [27,28] and Abgrall [6]. This will aid the construction of a localised approximation for which there will be no need to solve a global linear system. Another possibility that will be considered and which is expected to lead to a construction of a second order accurate and *positive* scheme is a genuinely non-linear RKRD-BLEND scheme. Recall that in this paper the blending parameter was designed in such a way that the resulting discrete system of equations was linear. By substituting δu^k instead of $\overline{\delta u^k}$ into the definition one possibly will get a positive scheme.

Finally, the schemes presented in this paper are, at most, second order accurate. In principle, it should be possible to extend these approaches to higher orders of accuracy using strategies that have been applied to other types of residual distribution scheme. For the time-stepping, higher order Runge–Kutta schemes can be used [38] as they have been within the explicit $\mathcal{RKRD}$ framework [7]. A higher order spatial discretisation simply requires the application of a linearity preserving distribution scheme to a residual evaluated from a higher order polynomial interpolant. Such an interpolant can be defined by sub-dividing each element uniformly into sub-elements, in the manner of P^k Lagrange elements [3], or reconstructing higher order derivative information at the element vertices [11]. The higher order spatial representation incurs a formal loss of positivity, so the fact that the linearised nature of the mass matrices also removes this property would be of less importance in such situations. The construction and application of robust, higher than second order, residual distribution schemes of all types is a subject of ongoing work.

Acknowledgments The first author would like to acknowledge EPSRC who funded this work under grant number EP/G003645/1.

References

1. Abgrall, R., Deconinck, H., Sermeus, K.: Status of multidimensional upwind residual distribution schemes and applications in aeronautics. In: AIAA Paper 2000–2328, Fluids 2000/Denver (2000)
2. Abgrall, R., Mezine, M.: Residual distribution schemes for steady problems. In: Computational Fluid Dynamics, VKI LS 2003–05. von Karman Institute for Fluid Dynamics (2003)
3. Abgrall, R., Roe, P.L.: High order fluctuation schemes on triangular meshes. J. Sci. Comput. **19**(1–3), 3–36 (2003). Special issue in honor of the sixtieth birthday of Stanley Osher
4. Abgrall, R.: Toward the ultimate conservative scheme: following the quest. J. Comput. Phys. **167**(2), 277–315 (2001)
5. Abgrall, R., Mezine, M.: Construction of second order accurate monotone and stable residual distribution schemes for unsteady flow problems. J. Comput. Phys. **188**(1), 16–55 (2003)
6. Abgrall, R.: A residual distribution method using discontinuous elements for the computation of possibly non smooth flows. Adv. Appl. Math. Mech. **2**(1), 32–44 (2010)
7. Abgrall, Ricchiuto M.: Explicit Runge–Kutta residual distribution schemes for time dependent problems: second order case. J. Comput. Phys. **229**(16), 5653–5691 (2010)
8. Balay, S., Brown, J., Buschelman, K., Eijkhout, V., Gropp, W.D., Kaushik, D., Knepley, M.G., McInnes, L.C., Smith, B.F., Zhang, H.: PETSc Users Manual. Tech. Rep. ANL-95/11 - Revision 3.3, Argonne National Laboratory (2012)
9. Balay, S., Brown, J., Buschelman, K., Gropp, W.D., Kaushik, D., Knepley, M.G., Curfman McInnes, L., Smith, B.F., Zhang, H.: PETSc Web page (2012). http://www.mcs.anl.gov/petsc
10. Brooks, A.N., Hughes, T.J.R.: Streamline upwind/Petrov-Galerkin formulations for convection dominated flows with particular emphasis on the incompressible Navier–Stokes equations. Comput. Methods Appl. Mech. Eng. **32**(1–3), 199–259 (1982)
11. Caraeni, D., Fuchs, L.: Compact third-order multidimensional upwind scheme for Navier–Stokes simulations. Theor. Comput. Fluid Dyn. **15**(6), 373–401 (2002)
12. Carette, J.C., Deconinck, H., Paillère, H., Roe, P.L.: Multidimensional upwinding: its relation to finite elements. Int. J. Numer. Methods Fluids **20**(8–9), 935–955 (1995)
13. Cockburn, B., Shu, C.W.: The Runge–Kutta discontinuous Galerkin method for conservation laws. V. Multidimensional systems. J. Comput. Phys. **141**(2), 199–224 (1998)
14. Csík, Á., Deconinck, H., Ricchiuto, M., Poedts, S.: Space-time residual distribution schemes for hyperbolic conservation laws. In: 15th AIAA Computational Fluid Dynamics Conference, Anaheim, CA, USA (2001)
15. Csík, Á., Deconinck, H.: Space-time residual distribution schemes for hyperbolic conservation laws on unstructured linear finite elements. Int. J. Numer. Methods Fluids **40**(3–4), 573–581 (2002)
16. Csík, Á., Ricchiuto, M., Deconinck, H.: A conservative formulation of the multidimensional upwind residual distribution schemes for general nonlinear conservation laws. J. Comput. Phys. **179**(1), 286–312 (2002)
17. Csík, Á., Deconinck, H., Ricchiuto, M.: Residual distribution for general time-dependent conservation laws. J. Comput. Phys. **209**(1), 249–289 (2005)
18. Deconinck, H., Ferrante, A.: Solution of the Unsteady Euler Equations using Residual Distribution and Flux Corrected Transport. Tech. Rep. 97–08, von Karman Institute for Fluid Dynamics (1997)
19. Deconinck, H., Ricchiuto, M.: Residual distribution schemes: foundations and analysis. In: Encyclopedia of Computational Mechanics, vol. 3. Wiley, London (2007)
20. Deconinck, H.: Upwind methods and multidimensional splittings for the Euler equations. In: Computational Fluid Dynamics, VKI LS 1991–01. von Karman Institute for Fluid Dynamics (1991)

21. Deconinck, H., Roe, P.L., Struijs, R.: A multidimensional generalization of Roe's flux difference splitter for the Euler equations. Comput. Fluids **22**(2–3), 215–222 (1993)
22. Dobeš, J., Deconinck, H.: Second order blended multidimensional upwind residual distribution scheme for steady and unsteady computations. J. Comput. Appl. Math. **215**(2), 378–389 (2008)
23. Emery, A.F.: An evaluation of several differencing methods for inviscid fluid flow problems. J. Comput. Phys. **2**, 306–331 (1968)
24. Godlewski, E., Raviart, P.A.: Numerical Approximation of Hyperbolic Systems of Conservation Laws, Applied Mathematical Sciences, vol. 118. Springer, New York (1996)
25. Guzik, S.M.J., Groth, C.P.T.: Comparison of solution accuracy of multidimensional residual distribution and Godunov-type finite-volume methods. Int. J. Comput. Fluid Dyn. **22**(1–2), 61–83 (2008)
26. Hubbard, M., Ricchiuto, M.: Discontinuous fluctuation distribution: a route to unconditional positivity and high order accuracy. In: ICFD 2010 International Conference on Fluid Dynamics, Reading (UK) (2010)
27. Hubbard, M.E.: Discontinuous fluctuation distribution. J. Comput. Phys. **227**(24), 10125–10147 (2008)
28. Hubbard, M.E.: A framework for discontinuous fluctuation distribution. Int. J. Numer. Methods Fluids **56**(8), 1305–1311 (2008)
29. Hughes, T.J.R., Mallet, M.: A new finite element formulation for CFD III: the generalized streamline operator for multidimensional advective-diffusive systems. Comput. Methods Appl. Mech. Eng. **58**, 305–328 (1986)
30. Johnson, C.: Numerical Solution of Partial Differential Equations by the Finite Element Method. Dover Publications Inc., Mineola, NY (2009). Reprint of the 1987 edition
31. LeVeque, R.J.: Numerical Methods for Conservation Laws. Lectures in Mathematics ETH Zürich, 2nd edn. Birkhäuser Verlag, Basel (1992)
32. LeVeque, R.J.: Finite Volume Methods for Hyperbolic Problems. Cambridge Texts in Applied Mathematics. Cambridge University Press, Cambridge (2002)
33. März, J., Degrez, G.: Improving time accuracy of residual distribution schemes. Tech. Rep. 96–17, von Karman Institute for Fluid Dynamics (1996)
34. De Palma, P., Pascazio, G., Rossiello, G., Napolitano, M.: A second-order-accurate monotone implicit fluctuation splitting scheme for unsteady problems. J. Comput. Phys. **208**(1), 1–33 (2005)
35. Roe, P.L.: Fluctuations and signals—a framework for numerical evolution problems. In: Numerical Methods for Fluid Dynamics, pp. 219–257. Academic Press, London (1982)
36. Roe, P.L.: Linear advection schemes on triangular meshes. Tech. Rep. Technical Report CoA 8720, Cranfield Institute of Technology (1987)
37. Roe, P.L.: Approximate Riemann solvers, parameter vectors, and difference schemes. J. Comput. Phys. **43**(2), 357–372 (1981)
38. Shu, C.W., Osher, S.: Efficient implementation of essentially nonoscillatory shock-capturing schemes. J. Comput. Phys. **77**(2), 439–471 (1988)
39. Struijs, R., Deconinck, H., Roe, P.L.: Fluctuation splitting schemes for the 2D Euler equations. In: Computational Fluid Dynamics, VKI LS 1991–01. von Karman Institute for Fluid Dynamics (1991)
40. van der Weide, E., Deconinck, H., Issman, E., Degrez, G.: A parallel, implicit, multi-dimensional upwind, residual distribution method for the Navier-Stokes equations on unstructured grids. Comput. Mech. **23**(2), 199–208 (1999)
41. van der Weide, E., Deconinck, H.: Positive matrix distribution schemes for hyperbolic systems, with application to the Euler equations. In: Desideri, J., Hirsch, C., LeTallec, P., Pandolfi, M., Periaux, J. (eds.) Computational Fluid Dynamics '96, pp. 747–753 (1996)
42. Warzyński, A., Hubbard, M.E., Ricchiuto, M.: Discontinuous residual distribution schemes for time-dependent problems. In: Li J., Yang, H.T., Machorro, E. (ed.) Recent Advances In Scientific Computing and Applications, Contemporary Mathematics, vol. 586, pp. 375–382 (2013)
43. Warzyński, A.: Runge-kutta residual distribution schemes. Ph.D. thesis, University of Leeds (2013)
44. Woodward, P., Colella, P.: The numerical simulation of two-dimensional fluid flow with strong shocks. J. Comput. Phys. **54**(1), 115–173 (1984)