**Tenth International Conference on Computational Fluid Dynamics (ICCFD10), Barcelona,Spain, July 9-13, 2018**

**ICCFD10-2018-186**

# Explicit local time stepping scheme for the unsteady simulation of turbulent flows

G. Jeanmasson*, I. Mary* and L. Mieussens*,**

Corresponding author: guillaume.jeanmasson@onera.fr

* ONERA, France.

** Université de Bordeaux, France.

**Abstract:** In this paper, two local time stepping schemes of order two and three in time are proposed. By construction, they are not mass conservative but a correction stage is added to make them conservative. These schemes are compared with some local time stepping schemes exisiting in the literature (schemes of Constantinescu and Sandu). The comparisons are carried out on various test cases. It reveals that our schemes can be more acurate and slightly faster than the schemes of Constantinesu and Sandu. Our third order local time stepping scheme is used to perform a simulation of airflow around an airfoil.

*Keywords:* Explicit local time stepping schemes, Runge-Kutta methods, Partitioned Runge-Kutta methods, Computational Fluid dynamics.

## 1 Introduction

Numerical simulations are widely used to study fluid mechanics problems. Simulations are based on the discretized form of the equation of fluid mechanics (Euler or Navier-Stokes equations). This discretized form is obtained by applying spacial and temporal integration methods. Finite volume approach is one of the most popular spacial integration method in CFD. For time integration, explicit Runge-Kutta methods are very efficient. However, with these methods the time step is strongly restricted by the numerical stability condition (CFL condition). This stability condition is defined in each cell and imposes that the time step is lower than the ratio between the cell volume and the magnitude of the wave speed in the cell. The time step for the entire domain is chosen to be lower than the smallest ratio in the mesh (generally imposed by the smallest cell in the mesh). Consequently, the time step may be much smaller than necessary in the other cells, which is not efficient. It is particularly true in most CFD meshes, which present some areas of mesh refinements where the stability condition is much more restrictive than in the other areas.

Implicit, unconditionally stable timestepping methods allows to use larger time steps, which reduces the computational time. However, parallel programming of implicit methods may be difficult. Moreover, implicit methods are less accurate than explicit methods.

The use of an explicit local time stepping scheme seems to be a good compromise between fully explicit and implicit methods. Indeed, an explicit local time stepping scheme allows a time step adaptation on the mesh to satisfy local stability conditions. Small time steps are used on cells which present the most severe CFL and larger time steps are applied on cells where CFL is less restrictive. This should result in a reduction of the computational cost, keeping the advantages of explicit methods: accuracy and good compatibility with parallel programming.

Two approaches are possible to construct explicit local time stepping schemes. In the first approach, each cell of the computational domain is integrated with its maximal allowable time step [1, 2, 3]. In the

second approach, each time step is associated to a group of cells [4, 5, 6, 7, 8, 9, 10, 11, 12]. The ratio between the different time steps may be an integer [4, 5, 6, 7, 10, 11, 12] or not [8, 9]. All the local time stepping schemes presented in this article are constructed by following this second approach. The local time stepping schemes seem to suffer of an important drawback: temporal high order is often incompatible with mass conservation [13]. In the 1980's, Osher and Sanders [5] proposed a locally varying time stepping scheme. It is conservative but only of order one in time. The schemes presented in [6, 13, 8, 9] are second order accurate but not mass-conservative. Note that [8, 9] can be higher order acurate in the case of linear spatial discretization. The scheme developped in [12] is second order acurate and seems mass-conservative but this property is not mentioned in the paper. In [7], a comparison is made between different multirate Runge-Kutta schemes and a multirate time stepping scheme of order three in time is proposed. However, the scheme is not mass-conservative. The strategy proposed by Constantinescu and Sandu [4] to construct local time stepping schemes seems interesting. Indeed, their strategy allows to obtain second order accurate and mass conservative local time stepping schemes. More over, the overall stability of their original local time stepping scheme proposed in [4] can be improved using a higher order Runge-Kutta base method [7, 10].

In this article, we propose two local time stepping schemes which are second order and third order accurate. Consistent solutions are calculated at the interface between domains of different time steps like in [6, 7, 11, 8, 9] to obtain the desired orders of acuracy. More over, a correction stage is added to make these schemes conservative. The aim of this article is to make a comparison between our strategy to construct mass conservative local time stepping schemes and the strategy of Constantinescu and Sandu.

In the first part of this article, we present some theoretical concepts for the construction of local time stepping schemes and we introduce the strategy used by Constantinescu and Sandu. In the second part, we introduce our approach to construct mass conservative local time stepping schemes of order two and three. The final part is dedicated to the comparison between the two strategies through several numerical test cases.

# 2 Local time stepping scheme and Partitionned Runge-Kutta methods

## 2.1 Semi-discretisation of a one-dimensional hyperbolic equation

We consider the one-dimensional hyperbolic equation :

$$\frac{\partial y(t,x)}{\partial t} + \frac{\partial f(y(t,x))}{\partial x} = 0, \tag{1}$$

with $y(0,x) = y^0(x)$, $x \in ]-\infty, +\infty[$ and $t > 0$. We also consider a 1D computational domain divided into cells of variable length $\Delta x_i$. Cell $i$ is bounded by points $x_{i-\frac{1}{2}}$ and $x_{i+\frac{1}{2}}$. The Finite Volume method is applied to (1). After space integration over cell $i$, we obtain :

$$\frac{\partial y_i}{\partial t} = -\frac{1}{\Delta x_i}(F_{i+\frac{1}{2}} - F_{i-\frac{1}{2}}), \tag{2}$$

where $y_i = \int_{x_{i-\frac{1}{2}}}^{x_{i+\frac{1}{2}}} y(t,x)dx$. The numerical flux at point $x_{i+\frac{1}{2}}$ is denoted by $F_{i+\frac{1}{2}}$ and reads $F_{i+\frac{1}{2}} = F(y_{i-1}, y_i, y_{i+1}, y_{i+2})$ in the case of a flux function with a four cell stencil, for instance. Now (2) can be viewed as an ordinary differential system that can be solved with an explicit Runge-Kutta (RK) method, as described in the following section. This is then so called the Method Of Lines (MOL).

## 2.2 Explicit Runge-Kutta methods

We consider an Ordinary Differential Equation :

$$\frac{dy}{dt} = f(t,y) \qquad y(t=0) = y^0. \tag{3}$$

The solution of (3) at time time $t^n$ is $y^n$. A s-step explicit Runge-Kutta method allows to compute $y^{n+1}$

(solution of (3) at time $t^{n+1} = t^n + \Delta t$) thanks to $y^n$ and $s-1$ intermediate values. In [14], a s-step explicit Runge-Kutta method is defined by the formulas :

$$y^{n+1} = y^n + \Delta t \sum_{i=1}^{s} b_i K_i, \tag{4}$$

$$K_i = f(t^n + c_i \Delta t, y^n + \Delta t \sum_{j=1}^{i-1} a_{ij} K_j) \tag{5}$$

$$= f(t^n + c_i \Delta t, y^{i-1}), \tag{6}$$

where $1 \leq i \leq s$ with $K_1 = f(t^n, y^n)$. The method is defined by its coefficients $A = \{a_{ij}\}$, $b = \{b_i\}$ and $c = \{c_i\}$ that can be represented in a Butcher tableau (see [14]) :

| c | A |
|---|---|
| | b |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| $c_1 = 0$ | 0 | | | | | | |
| $c_2$ | $a_{21}$ | | | | | | |
| $c_3$ | $a_{31}$ | $a_{32}$ | | | | | |
| . | . | | | | | | |
| . | . | | | | | | |
| . | . | | | | | | |
| $c_s$ | $a_{s1}$ | $a_{s2}$ | . | . | . | $a_{s,s-1}$ | |
| | $b_1$ | $b_2$ | . | . | . | $b_{s-1}$ | $b_s$ |

Table 1: Butcher tableau of an s-stage explicit Runge-Kutta method

All Runge-Kutta methods in this article satisfy the property of consistency :

$$c_i = \sum_{j=1}^{i-1} a_{ij}. \tag{7}$$

## 2.3 Partitioned Runge-Kutta methods

In [4] and [13], Partitioned Runge-Kutta (PRK) methods are applied to temporal multiscale problems.

Let us consider a 1D computational domain composed of $n$ cells (figure 1). The Finite Volume method applied to the one-dimensional hyperbolic equation (1) on the whole domain leads to the autonomous system of Ordinary Differential Equations

$$\frac{dy}{dt} = f(y), \qquad y \in R^n. \tag{8}$$

As shown in figure (1), the computational domain is divided into two subdomains which have different time scales. The first subdomain has a large time scale and contains cells from cell 1 to cell $i_0$ (large cells). The second subdomain has a short time scale and contains cells from cell $i_{0+1}$ to cell $n$ (small cells).

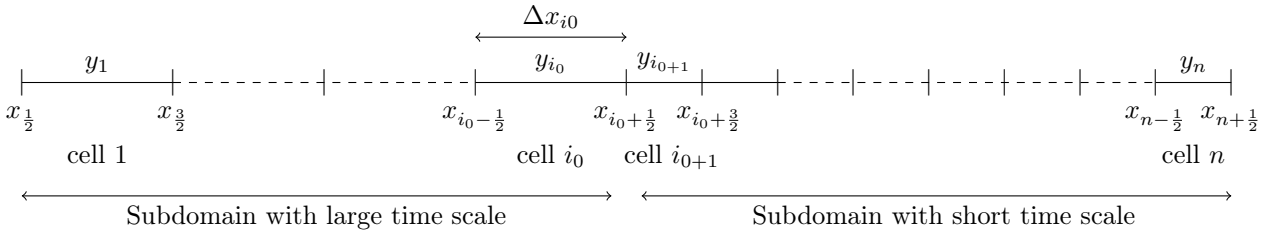


Figure 1: Partitionned computational domain

We denote by $Y_L = [y_1, y_2, ..., y_{i_0}]$ the vector of unknowns in the large time scale subdomain and by $Y_S = [y_{i_{0+1}}, y_{i_{0+2}}, ..., y_n]$ the vector of unknowns in the short time scale subdomain. System (8) can be rewritten using the variables $Y_L$ and $Y_S$ :

$$\begin{cases} \frac{dY_L}{dt} = f_L(Y_L, Y_S) \\ \frac{dY_S}{dt} = f_S(Y_L, Y_S) \end{cases} \tag{9}$$

The dependency between vectors $Y_L$ and $Y_S$ in each equation of (9) occurs at the interface between the subdomains. The two equations can be solved with different Runge-Kutta methods since $Y_L$ and $Y_S$ do not evolve with the same time scale. We denote by RKL the $s$-stage Runge-Kutta method for the Large time scale subdomain. It is used to solve the first equation and its coefficients are : $A^L$, $b^L$, $c^L$. Similarly, we denote by RKS the $s$-stage Runge-Kutta method for the Short time scale subdomain. It is used to solve the second equation and its coefficients are : $A^S$, $b^S$, $c^S$. According to [14], the PRK method using RKL and RKS is given by the following formulas:

$$Y_L^{n+1} = Y_L^n + \Delta t \sum_{i=1}^{s} b_i^L K_L^i \qquad Y_S^{n+1} = Y_S^n + \Delta t \sum_{i=1}^{s} b_i^S K_S^i$$

$$Y_L^i = Y_L^n + \Delta t \sum_{j=1}^{i-1} a_{ij}^L K_L^i \qquad Y_S^i = Y_S^n + \Delta t \sum_{j=1}^{i-1} a_{ij}^S K_S^i \tag{10}$$

$$K_L^i = f_L(Y_L^{i-1}, Y_S^{i-1}) \qquad K_S^i = f_S(Y_L^{i-1}, Y_S^{i-1})$$

with $K_L^1 = f_L(Y_L^n, Y_S^n)$ and $K_S^1 = f_S(Y_L^n, Y_S^n)$. We remind that $Y_L^n$ and $Y_S^n$ denote the vectors of unknowns at time $t^n$ in the large time scale subdomain and in the short time scale subdomain, respectively. $Y_L^{n+1}$ and $Y_S^{n+1}$ are the vectors of unknowns at time $t^n + \Delta t$.

In most explicit local time stepping schemes (for instance [4], [6], [5] and [7]), RKL and RKS methods are based on the same RK method. RKS method is obtained by applying twice successively the base method with the time step $\frac{\Delta t}{2}$. RKL method is composed of the base method with the time step $\Delta t$, and some additional values calculated at the interface between the two subdomains. With this strategy, an explicit local time stepping scheme with times steps $\Delta t$ and $\frac{\Delta t}{2}$ is obtained. To have a Partitioned Runge-Kutta method of order $p$ and therefore, a local time stepping scheme of order $p$, both RKL and RKS methods must be of order $p$ but this is not sufficient. The coefficients of the methods have to satisfy additional coupling conditions. Following the methodology of [14] we can derive the following conditions on the Runge-Kutta coefficients to obtain a PRK method of order two and three, respectively (tables 2 and 3).

| RKL of order 2 | RKS of order 2 | Coupling conditions |
|---|---|---|
| $\sum_{i=1}^{s} b_i^L = 1$ | $\sum_{i=1}^{s} b_i^S = 1$ | $\sum_{i=1}^{s} b_i^L c_i^S = \frac{1}{2}$ |
| $\sum_{i=1}^{s} b_i^L c_i^L = \frac{1}{2}$ | $\sum_{i=1}^{s} b_i^S c_i^S = \frac{1}{2}$ | $\sum_{i=1}^{s} b_i^S c_i^L = \frac{1}{2}$ |

Table 2: Conditions to obtain a second order accurate PRK method

| RKL of order 3 | RKS of order 3 | Coupling conditions | |
|---|---|---|---|
| $\sum_{i=1}^{s} b_i^L = 1$ | $\sum_{i=1}^{s} b_i^S = 1$ | $\sum_{i=1}^{s} b_i^L c_i^S = \frac{1}{2}$ | $\sum_{i=1}^{s} b_i^S c_i^L = \frac{1}{2}$ |
| $\sum_{i=1}^{s} b_i^L c_i^L = \frac{1}{2}$ | $\sum_{i=1}^{s} b_i^S c_i^S = \frac{1}{2}$ | $\sum_{i=1}^{s}\sum_{j=1}^{i-1} b_i^L a_{ij}^S c_j^L = \frac{1}{6}$ | $\sum_{i=1}^{s}\sum_{j=1}^{i-1} b_i^S a_{ij}^S c_j^L = \frac{1}{6}$ |
| $\sum_{i=1}^{s} b_i^L (c_i^L)^2 = \frac{1}{3}$ | $\sum_{i=1}^{s} b_i^S (c_i^S)^2 = \frac{1}{3}$ | $\sum_{i=1}^{s}\sum_{j=1}^{i-1} b_i^L a_{ij}^S c_j^S = \frac{1}{6}$ | $\sum_{i=1}^{s}\sum_{j=1}^{i-1} b_i^S a_{ij}^L c_j^L = \frac{1}{6}$ |
| $\sum_{i=1}^{s}\sum_{j=1}^{i-1} b_i^L a_{ij}^L c_j^L = \frac{1}{6}$ | $\sum_{i=1}^{s}\sum_{j=1}^{i-1} b_i^S a_{ij}^S c_j^S = \frac{1}{6}$ | $\sum_{i=1}^{s}\sum_{j=1}^{i-1} b_i^L a_{ij}^L c_j^S = \frac{1}{6}$ | $\sum_{i=1}^{s}\sum_{j=1}^{i-1} b_i^S a_{ij}^L c_j^S = \frac{1}{6}$ |
| | | $\sum_{i=1}^{s} b_i^L c_i^S c_i^L = \frac{1}{3}$ | $\sum_{i=1}^{s} b_i^S c_i^S c_i^L = \frac{1}{3}$ |

Table 3: Conditions to obtain a third order accurate PRK method

The number of conditions quickly increases with the order of accuracy which makes the implementation of high order local time stepping schemes difficult.

## 2.4 Constantinescu and Sandu strategy

In [4], Constantinescu and Sandu have developped a strategy, based on PRK methods, to obtain conservative and second order accurate local time stepping schemes. In this subsection, we briefly describe two schemes based on this strategy. The first one is the original scheme, based on a RK2 method [4]. The second one follows the same strategy but is based on a RK3 method (see [7, 10]). A comparison between these schemes and the schemes we propose (that are based on a different strategy, see part 3) is performed in part 4.

### 2.4.1 First scheme : original scheme of Constantinescu and Sandu

The original scheme developed by Constantinescu and Sandu is based on the second order Heun's method:

| | | |
|---|---|---|
| 0 | 0 | |
| 1 | 1 | 0 |
| | $\frac{1}{2}$ | $\frac{1}{2}$ |

Table 4: Base method for the original scheme of Constantinescu and Sandu

The RKS method (Table 6) is obtained by applying twice successively the base method with the time step $\frac{\Delta t}{2}$. The RKL method (Table 5) is obtained by applying twice successively the base method with the time step $\Delta t$.

| | | | | |
|---|---|---|---|---|
| 0 | 0 | | | |
| 1 | 1 | 0 | | |
| 0 | 0 | 0 | 0 | |
| 1 | 0 | 0 | 1 | 0 |
| | $\frac{1}{4}$ | $\frac{1}{4}$ | $\frac{1}{4}$ | $\frac{1}{4}$ |

Table 5: RKL method

| | | | | |
|---|---|---|---|---|
| 0 | 0 | | | |
| $\frac{1}{2}$ | $\frac{1}{2}$ | 0 | | |
| $\frac{1}{2}$ | $\frac{1}{4}$ | $\frac{1}{4}$ | 0 | |
| 1 | $\frac{1}{4}$ | $\frac{1}{4}$ | $\frac{1}{2}$ | 0 |
| | $\frac{1}{4}$ | $\frac{1}{4}$ | $\frac{1}{4}$ | $\frac{1}{4}$ |

Table 6: RKS method

Both RKL and RKS methods are second order accurate and we have $b_i^L = b_i^S$ for every $i$. These properties ensure that this local time stepping scheme is second order accurate [4]. The equality of coefficents $b_i$ ensures that the scheme is mass conservative [4]. In [4], the authors demonstrate that the RKL method described in Table 5 is only applied in a buffer zone: an area of variable length (it depends on the spatial stencil) adjacent to the interface. Outside this buffer zone, RKL reduces to the Heun method described in Table 4. Finally, our semi-discrete approximation (2) is solved by the algorithm shown in table 7, in which $D_i^k = \frac{1}{\Delta x}(F_{i+\frac{1}{2}}(y_i^k, y_{i+1}^k) - F_{i-\frac{1}{2}}(y_{i-1}^k, y_i^k))$.

| | Large-time-scale-subdomain | | Short-time-scale-subdomain |
|---|---|---|---|
| 1 | $y_i^1 = y_i^n - \Delta t D_i^n$ | $y_i^1 = y_i^n - \Delta t D_i^n$ | $y_i^1 = y_i^n - \frac{\Delta t}{2} D_i^n$ |
| 2 | | $y_i^2 = y_i^n$ | $y_i^2 = y_i^n - \frac{\Delta t}{4}[D_i^n + D_i^1]$ |
| 3 | | $y_i^3 = y_i^n - \Delta t D_i^2$ | $y_i^3 = y_i^2 - \frac{\Delta t}{2} D_i^2$ |
| 4 | $y_i^{n+1} = y_i^n - \frac{\Delta t}{2}[D_i^n + D_i^1]$ | $y_i^{n+1} = y_i^n - \frac{\Delta t}{4}[D_i^n + D_i^1 + D_i^2 + D_i^3]$ | $y_i^{n+1} = y_i^2 - \frac{\Delta t}{4}[D_i^2 + D_i^3]$ |
| | Buffer zone | | |

Table 7: Algorithm of the original scheme of Constantinescu and Sandu

We can notice that this scheme is internally non consistent because at the interface values estimated at different times are mixed in flux balance $D$. For instance, at the interface, $D_i^2$ contains values estimated at time $t^n$ ($y_i^2$ from the buffer zone) and values estimated at time $t^{n+\frac{1}{2}}$ ($y_i^2$ from the short time scale subdomain). Despite this inconsistency, the scheme is second order accurate and conservative.

For each cell in the buffer zone, four flux balances have to be calculated, therefore the computational cost is the same as in the short time scale subdomain. In our applications, we use a flux function with a four cell stencil. In this case, it can be proven that the size of the buffer zone is equal to four cells.

### 2.4.2 Second scheme

In [7] and [10], a RK3 method is used as the base method and the strategy of Constantinescu and Sandu is applied. Here, we extend this approach using a RK3 low-storage scheme [15] as the base method. Low-storage Runge-Kutta schemes require less storage than "classical" Runge-Kutta schemes. The RK3 low-storage method applied to (8) reads :

| | | |
|---|---|---|
| Step 1 | $K_1 = \Delta t f(t^n, y^n)$<br>$y^1 = y^n + \beta_1 K_1$ | $\beta_1 = \frac{1}{2}$ |
| Step 2 | $K_2 = \alpha_2 K_1 + \Delta t f(t^n + c_2 \Delta t, y_1)$<br>$y^2 = y^1 + \beta_2 K_2$ | $\alpha_2 = -0.6830127$<br>$c_2 = \frac{1}{2}$<br>$\beta_2 = 0.9106836$ |
| Step 3 | $K_3 = \alpha_3 K_2 + \Delta t f(t^n + c_3 \Delta t, y_2)$<br>$y^{n+1} = y^3 = y^2 + \beta_3 K_3$ | $\alpha_3 = -\frac{4}{3}$<br>$\beta_3 = 0.3660254$<br>$c_3 = 0.7886751$ |

Table 8: The RK3 low-storage scheme written in "low-storage form"

The values of the coefficients $\alpha$ and $\beta$ have been proposed by Lowery and Reynolds [16]. Successive values of $K$ and $y$ overwrite the previous ones so that at any stage only $2N$ storage locations are required. The same method can be written in the "classical Runge-Kutta form" and a Butcher tableau can be associated:

| | | |
|---|---|---|
| Step 1 | $K_1 = \Delta t f(t^n, y^n)$<br>$y^1 = y^n + a_{21}K_1$ | <br>$a_{21} = \beta_1$ |
| Step 2 | $K_2 = \Delta t f(t^n + c_2\Delta t, y_1)$<br>$y^2 = y^n + a_{31}K_1 + a_{32}K_2$ | $a_{31} = \beta_1 + \beta_2\alpha_2$<br>$a_{32} = \beta_2$<br>$c_2 = \frac{1}{2}$ |
| Step 3 | $K_3 = \Delta t f(t^n + c_3\Delta t, y_2)$<br>$y^{n+1} = y^3 = y^n + b_1K_1 + b_2K_2 + b_3K_3$ | $b_1 = a_{31} + \beta_3\alpha_2\alpha_3$<br>$b_2 = \beta_3 + \beta_3\alpha_2$<br>$b_3 = \beta_3$<br>$c_3 = 0.7886751$ |

| | | | |
|---|---|---|---|
| 0 | 0 | | |
| $c_2$ | $a_{21}$ | | |
| $c_3$ | $a_{31}$ | $a_{32}$ | |
| | $b_1$ | $b_2$ | $b_3$ |

Table 9: The RK3 low-storage scheme written in "classical form" and its associated Butcher tableau

In the following, we only use the "classical form" described in table 9 for our RK3 scheme. Indeed, its associated Butcher tableau is very useful for constructing and describing local time stepping schemes.

Following the strategy of Constantinescu and Sandu, the base method is duplicated to constuct RKL (table 10) and RKS (table 11) methods.

| | | | | | | |
|---|---|---|---|---|---|---|
| 0 | 0 | | | | | |
| $c_2$ | $a_{21}$ | | | | | |
| $c_3$ | $a_{31}$ | $a_{32}$ | | | | |
| 0 | 0 | 0 | 0 | | | |
| $c_2$ | 0 | 0 | 0 | $a_{21}$ | | |
| $c_3$ | 0 | 0 | 0 | $a_{31}$ | $a_{32}$ | |
| | $\frac{b_1}{2}$ | $\frac{b_2}{2}$ | $\frac{b_3}{2}$ | $\frac{b_1}{2}$ | $\frac{b_2}{2}$ | $\frac{b_3}{2}$ |

Table 10: RKL method

| | | | | | | |
|---|---|---|---|---|---|---|
| 0 | 0 | | | | | |
| $\frac{c_2}{2}$ | $\frac{a_{21}}{2}$ | | | | | |
| $\frac{c_3}{2}$ | $\frac{a_{31}}{2}$ | $\frac{a_{32}}{2}$ | | | | |
| $\frac{1}{2}$ | $\frac{b_1}{2}$ | $\frac{b_2}{2}$ | $\frac{b_3}{2}$ | | | |
| $\frac{1+c_2}{2}$ | $\frac{b_1}{2}$ | $\frac{b_2}{2}$ | $\frac{b_3}{2}$ | $\frac{a_{21}}{2}$ | | |
| $\frac{1+c_3}{2}$ | $\frac{b_1}{2}$ | $\frac{b_2}{2}$ | $\frac{b_3}{2}$ | $\frac{a_{31}}{2}$ | $\frac{a_{32}}{2}$ | |
| | $\frac{b_1}{2}$ | $\frac{b_2}{2}$ | $\frac{b_3}{2}$ | $\frac{b_1}{2}$ | $\frac{b_2}{2}$ | $\frac{b_3}{2}$ |

Table 11: RKS method

Even though both RKL and RKS methods are third order accurate, the local time stepping scheme is only second order accurate because some third order coupling conditions are not satisfied [4]. However, the local time stepping is mass conservative [10]. Like for the original scheme of Constantinescu and Sandu, there is a buffer zone at the interface, in the large time scale subdomain. Outside this buffer zone, the RKL method reduces to the base method described in table 9. Finally, our semi-discrete approximation (2) is solved by the algorithm shown in table 12, in which $D_i^k = \frac{1}{\Delta x}(F_{i+\frac{1}{2}}(y_i^k, y_{i+1}^k) - F_{i-\frac{1}{2}}(y_{i-1}^k, y_i^k))$.

| | Large-time-scale-subdomain | | Short-time-scale-subdomain |
|---|---|---|---|
| 1 | $y_i^1 = y_i^n - a_{21}\Delta t D_i^n$ | $y_i^1 = y_i^n - a_{21}\Delta t D_i^n$ | $y_i^1 = y_i^n - \frac{a_{21}}{2}\Delta t D_i^n$ |
| 2 | $y_i^2 = y_i^n - a_{31}\Delta t D_i^n$ $-a_{32}\Delta t D_i^1$ | $y_i^2 = y_i^n - a_{31}\Delta t D_i^n$ $-a_{32}\Delta t D_i^1$ | $y_i^2 = y_i^n - \frac{a_{31}}{2}\Delta t D_i^n$ $-\frac{a_{32}}{2}\Delta t D_i^1$ |
| 3 | | $y_i^3 = y_i^n$ | $y_i^3 = y_i^n - \frac{b_1}{2}\Delta t D_i^n$ $-\frac{b_2}{2}\Delta t D_i^1 - \frac{b_3}{2}\Delta t D_i^2$ |
| 4 | | $y_i^4 = y_i^n - a_{21}\Delta t D_i^3$ | $y_i^4 = y_i^3 - \frac{a_{21}}{2}\Delta t D_i^3$ |
| 5 | | $y_i^5 = y_i^n - a_{31}\Delta t D_i^3$ $-a_{32}\Delta t D_i^4$ | $y_i^5 = y_i^3 - \frac{a_{31}}{2}\Delta t D_i^3$ $-\frac{a_{32}}{2}\Delta t D_i^4$ |
| 6 | $y_i^{n+1} = y_i^n - b_1\Delta t D_i^n$ $-b_2\Delta t D_i^1 - b_3\Delta t D_i^2$ | $y_i^{n+1} = y_i^n - \frac{b_1}{2}\Delta t(D_i^n + D_i^3)$ $-\frac{b2}{2}\Delta t(D_i^1 + D_i^4) - \frac{b_3}{2}\Delta t(D_i^2 + D_i^5)$ | $y_i^{n+1} = y_i^3 - \frac{b_1}{2}\Delta t D_i^3$ $-\frac{b_2}{2}\Delta t D_i^4 - \frac{b_3}{2}\Delta t D_i^5$ |
| | Buffer zone | | |

Table 12: Algorithm of the second locally varying time stepping scheme

The algorithm is written in "classical form" but it can be easily implemented in the low storage form since these two formulations are equivalent.

In this case, with a flux function with a four cell stencil, the size of the buffer zone is equal to 6 cells.

# 3 Schemes proposed

In this section, we present a different approach to construct local time stepping schemes. Like in the strategy of Constantinescu and Sandu, we use the theory of PRK methods but here we guarantee the internal consistency of the scheme. This allows to obtain higher order accuracy, since we are able to design a third order scheme. Note that our schemes require a correction step to make them mass conservative.

## 3.1 Scheme of order 2

Again, we use the second order Heun's method (table 4) as the base method. It is applied twice with the step $\frac{\Delta t}{2}$ to obtain the RKS method (table 14). Our RKL method is composed of the base method and some additional values to perform the transition with the RKS method and achieve the desired order of accuracy. Thus, both RKL and RKS methods are of order 2. Now, we impose internal consistency by the relation $c_i^L = c_i^S$, for every $i$. Then, it can be easily seen that second order coupling conditions (see table 2) are satisfied. Note that the intermediate time coefficients of RKS are already fixed (we have $c^S = \{0; \frac{1}{2}; \frac{1}{2}; 1\}$). For RKL, we have $c_1^L = 0$ and $c_4^L = 1$, but $c_2^L$ and $c_3^L$ are still undetermined. Then the consistency property is obtained with $c_2^L = c_3^L = \frac{1}{2}$. This requires to define additional intermediate values at time $t^n + \frac{1}{2}\Delta t$ at the interface, in the large time scale subdomain. Finally, the coefficients $a_{ij}$ are determined to satisfy the consistency relation (7). The Butcher tableau of RKL and RKS methods read :

| | | | | |
|---|---|---|---|---|
| 0 | 0 | | | |
| $\frac{1}{2}$ | $\frac{1}{2}$ | 0 | | |
| $\frac{1}{2}$ | $\frac{1}{2}$ | 0 | 0 | |
| 1 | 1 | 0 | 0 | 0 |
| | $\frac{1}{2}$ | 0 | 0 | $\frac{1}{2}$ |

Table 13: RKL method

| | | | | |
|---|---|---|---|---|
| 0 | 0 | | | |
| $\frac{1}{2}$ | $\frac{1}{2}$ | 0 | | |
| $\frac{1}{2}$ | $\frac{1}{4}$ | $\frac{1}{4}$ | 0 | |
| 1 | $\frac{1}{4}$ | $\frac{1}{4}$ | $\frac{1}{2}$ | 0 |
| | $\frac{1}{4}$ | $\frac{1}{4}$ | $\frac{1}{4}$ | $\frac{1}{4}$ |

Table 14: RKS method

However, this local time stepping scheme defined by RKL and RKS is not mass conservative. It can be shown by applying RKL and RKS methods to (2) on cells $i_0$ and $i_{0+1}$, respectively. We remind that cell $i_0$ is the last cell of the large time scale subdomain while cell $i_{0+1}$ is the first cell of the short time scale subdomain (see figure 1). The application of RKL and RKS methods on cells $i_0$ and $i_{0+1}$ is shown in table 15. Note that to clarify the presentation, we assume a 2 cell stencil flux function: $F^k_{i+\frac{1}{2}} = F_{i+\frac{1}{2}}(y^k_i, y^k_{i+1})$.

| step | Solution on cell $i_0$ | Solution on cell $i_{0+1}$ |
|---|---|---|
| 1 | $y^1_{i0} = y^n_{i0} - \frac{\Delta t}{2\Delta x_{i0}}(F^n_{i0+\frac{1}{2}} - F^n_{i0-\frac{1}{2}})$ | $y^1_{i0+1} = y^n_{i0+1} - \frac{\Delta t}{2\Delta x_{i0+1}}(F^n_{i0+\frac{3}{2}} - F^n_{i0+\frac{1}{2}})$ |
| 2 | $y^2_{i0} = y^1_{i0}$ | $y^2_{i0+1} = y^n_{i0+1} - \frac{\Delta t}{4\Delta x_{i0+1}}(F^n_{i0+\frac{3}{2}} + F^1_{i0+\frac{3}{2}} - F^n_{i0+\frac{1}{2}} - F^1_{i0+\frac{1}{2}})$ |
| 3 | $y^3_{i0} = y^n_{i0} - \frac{\Delta t}{\Delta x_{i0}}(F^n_{i0+\frac{1}{2}} - F^n_{i0-\frac{1}{2}})$ | $y^3_{i0+1} = y^2_{i0+1} - \frac{\Delta t}{2\Delta x_{i0+1}}(F^2_{i0+\frac{3}{2}} - F^2_{i0+\frac{1}{2}})$ |
| 4 | $y^{n+1}_{i0} = y^n_{i0} - \frac{\Delta t}{2\Delta x_{i0}}(F^n_{i0+\frac{1}{2}} + F^1_{i0+\frac{1}{2}} - F^n_{i0-\frac{1}{2}} - F^1_{i0-\frac{1}{2}})$ | $y^{n+1}_{i0+1} = y^2_{i0+1} - \frac{\Delta t}{4\Delta x_{i0+1}}(F^2_{i0+\frac{3}{2}} + F^3_{i0+\frac{3}{2}} - F^2_{i0+\frac{1}{2}} - F^3_{i0+\frac{1}{2}})$ |

Table 15: RKL and RKS methods applied on cells $i_0$ and $i_{0+1}$

We analyse the expression of $y^{n+1}_{i0}$. We notice that, between $t^n$ and $t^n + \Delta t$, the Total Flux Lost by the large time scale subdomain through the interface $x_{i0+\frac{1}{2}}$, denoted by $TFL_{i0+\frac{1}{2}}$, is :

$$TFL_{i0+\frac{1}{2}} = \frac{1}{2}[F^n_{i0+\frac{1}{2}} + F^1_{i0+\frac{1}{2}}] \tag{11}$$

The same analysis can be made on the other side of the interface. The Total Flux Received by the short time scale subdomain through the interface $x_{i0+\frac{1}{2}}$, denoted by $TFR_{i0+\frac{1}{2}}$, is:

$$TFR_{i0+\frac{1}{2}} = \frac{1}{4}[F^n_{i0+\frac{1}{2}} + F^1_{i0+\frac{1}{2}} + F^2_{i0+\frac{1}{2}} + F^3_{i0+\frac{1}{2}}] \tag{12}$$

It is clear that total fluxes $TFL_{i0+\frac{1}{2}}$ and $TFR_{i0+\frac{1}{2}}$ are different and consequently the local time stepping scheme is not mass preserving at the interface between the subdomains.

We propose a correction step to make the scheme conservative. Following the notations given above, $y^{n+1}_{i0}$ reads :

$$y^{n+1}_{i0} = y^n_{i0} - \frac{\Delta t}{\Delta x_{i0}} TFL_{i0+\frac{1}{2}} + \frac{\Delta t}{2\Delta x_{i0}}(F^n_{i0-\frac{1}{2}} + F^1_{i0-\frac{1}{2}}) \tag{13}$$

This value is corrected by adding the total flux difference. This leads to the following expression for the corrected value $y^{n+1,c}_{i0}$:

$$y^{n+1,c}_{i0} = y^{n+1}_{i0} + \frac{\Delta t}{\Delta x_{i0}} TFL_{i0+\frac{1}{2}} - \frac{\Delta t}{\Delta x_{i0}} TFR_{i0+\frac{1}{2}} \tag{14}$$

We substitute $y^{n+1}_{i0}$ by its expression given by (13). The corrected value $y^{n+1,c}_{i0}$ now reads:

$$y^{n+1,c}_{i0} = y^n_{i0} - \frac{\Delta t}{\Delta x_{i0}} TFR_{i0+\frac{1}{2}} + \frac{\Delta t}{2\Delta x_{i0}}(F^n_{i0-\frac{1}{2}} + F^1_{i0-\frac{1}{2}}) \tag{15}$$

In (15), the total flux lost by the large time scale subdomain through the interface $x_{i0+\frac{1}{2}}$ now is exactly $TFR_{i0+\frac{1}{2}}$, defined in (12). Consequently, the corrected scheme is mass conservative.

The algorithm of this local time stepping applied to our semi-discrete approximation (2) is shown in table 16. In this algorithm, $D^k_i = \frac{1}{\Delta x}(F_{i+\frac{1}{2}}(y^k_i, y^k_{i+1}) - F_{i-\frac{1}{2}}(y^k_{i-1}, y^k_i))$.

| | Large-time-scale-subdomain | | Short-time-scale-subdomain |
|---|---|---|---|
| 1 | $y_i^3 = y_i^n - \Delta t D_i^n$ | $y_i^1 = y_i^n - \frac{\Delta t}{2} D_i^n$ | $y_i^1 = y_i^n - \frac{\Delta t}{2} D_i^n$ |
| 2 | | $y_i^2 = y_i^1$ | $y_i^2 = y_i^n - \frac{\Delta t}{4}[D_i^n + D_i^1]$ |
| 3 | | $y_i^3 = y_i^n - \Delta t D_i^n$ | $y_i^3 = y_i^2 - \frac{\Delta t}{2} D_i^2$ |
| 4 | $y_i^{n+1} = y_i^n - \frac{\Delta t}{2}[D_i^n + D_i^3]$ | $y_i^{n+1} = y_i^n - \frac{\Delta t}{2}[D_i^n + D_i^3]$ | $y_i^{n+1} = y_i^2 - \frac{\Delta t}{4}[D_i^2 + D_i^3]$ |
| 5 | | Correction step (last cell) | |

Buffer zone

Table 16: Algorithm of the proposed scheme of order 2

In our applications, with a four cell stencil flux function, the size of the buffer zone is equal to two cells, while it is equal to four cells for the original scheme of Constanintescu and Sandu. Moreover, only two flux evaluations ($D_i^n$ and $D_i^3$) are needed in the buffer zone whereas the original scheme of Constantinescu and Sandu requires four flux evaluations in this zone. Our new scheme, despite the additional count due to the correction step, seems to lead to a small computational speedup compared to the original scheme of Constantinescu and Sandu (see section 4).

## 3.2 Scheme of order 3

The base method used to construct this local time stepping scheme is the RK3 low storage method described in table 9. The construction of RKS method is performed with two successive applications of the base method with the time step $\frac{\Delta t}{2}$. Consequently, the RKS method has the following intermediate time coefficients: $c^S = \{0; \frac{c_2}{2}; \frac{c_3}{2}; \frac{1}{2}; \frac{1+c_2}{2}; \frac{1+c_3}{2}\}$. For the construction of RKL method, the base method is applied once with a time step $\Delta t$, which leads to the following intermediate time coefficients: $c^L = \{0; c_2 = \frac{1}{2}; c_3\}$. As for the first proposed scheme, we impose internal consistency by the relations $c_i^L = c_i^S$, for every $i$. Indeed, this property implies that some of the third order coupling conditions of table 3 are necessarily satisfied (namely that of lines 1,4,5). To satisfy internal consistency, an additional value at time $t + c_3\Delta t$ must be calculated in RKS method. In RKL method, additional values at times $t + \frac{c_2}{2}\Delta t$, $t + \frac{c_3}{2}\Delta t$, $t + \frac{1+c_2}{2}\Delta t$ and $t + \frac{1+c_3}{2}\Delta t$ must be computed. RKL and RKS methods are now two third order Runge-Kutta methods with 7 stages. Their intermediate time coefficients are the same and read: $\{ 0; \frac{c_2}{2} ; \frac{c_3}{2} ; \frac{1}{2} ; \frac{1+c_2}{2} ; c_3; \frac{1+c_3}{2} \}$. The Runge-Kutta coefficients for the calculation of additional values are denoted by $\alpha^L$ in RKL method and by $\alpha^S$ in RKS method. The Butcher tableaus of RKL and RKS methods are shown in tables 17 and 18, respectively.

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 0 | 0 | | | | | | |
| $\frac{c_2}{2}$ | $\alpha_1^L$ | | | | | | |
| $\frac{c_3}{2}$ | $\alpha_2^L$ | $\alpha_3^L$ | | | | | |
| $\frac{1}{2}$ | $a_{21}$ | 0 | 0 | | | | |
| $\frac{1+c_2}{2}$ | $\alpha_4^L$ | $\alpha_5^L$ | $\alpha_6^L$ | $\alpha_7^L$ | | | |
| $c_3$ | $a_{31}$ | 0 | 0 | $a_{32}$ | 0 | | |
| $\frac{1+c_3}{2}$ | $\alpha_8^L$ | $\alpha_9^L$ | $\alpha_{10}^L$ | $\alpha_{11}^L$ | $\alpha_{12}^L$ | $\alpha_{13}^L$ | |
| | $b_1$ | 0 | 0 | $b_2$ | 0 | $b_3$ | 0 |

Table 17: RKL method

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 0 | 0 | | | | | | |
| $\frac{c_2}{2}$ | $\frac{a_{21}}{2}$ | | | | | | |
| $\frac{c_3}{2}$ | $\frac{a_{31}}{2}$ | $\frac{a_{32}}{2}$ | | | | | |
| $\frac{1}{2}$ | $\frac{b_1}{2}$ | $\frac{b_2}{2}$ | $\frac{b_3}{2}$ | | | | |
| $\frac{1+c_2}{2}$ | $\frac{b_1}{2}$ | $\frac{b_2}{2}$ | $\frac{b_3}{2}$ | $\frac{a_{21}}{2}$ | | | |
| $c_3$ | $\alpha_1^S$ | $\alpha_2^S$ | $\alpha_3^S$ | $\alpha_4^S$ | $\alpha_5^S$ | | |
| $\frac{1+c_3}{2}$ | $\frac{b_1}{2}$ | $\frac{b_2}{2}$ | $\frac{b_3}{2}$ | $\frac{a_{31}}{2}$ | $\frac{a_{32}}{2}$ | 0 | |
| | $\frac{b_1}{2}$ | $\frac{b_2}{2}$ | $\frac{b_3}{2}$ | $\frac{b_1}{2}$ | $\frac{b_2}{2}$ | 0 | $\frac{b_3}{2}$ |

Table 18: RKS method

As explained below, the internal consistency implies that some of third order coupling conditions of table 3 are automatically satisfied. The 4 coupling conditions of lines 2 and 3 of table 3 are unsatisfied. Some of them are equivalent, therefore the remaining unsatisfied coupling coniditions read:

$$\sum_{i=1}^{s}\sum_{j=1}^{s} b_i^L a_{ij}^S c_j^S = \frac{1}{6}, \tag{16}$$

$$\sum_{i=1}^{s}\sum_{j=1}^{s} b_i^S a_{ij}^L c_j^L = \frac{1}{6}, \tag{17}$$

The coefficients for the computation of additional values in RKL method have to satisfy (17) and four relations given by the consistency property (7). This gives 5 relations that can be used to determine 5 unknowns only. The other eight coefficients can be set to any arbitrary value. A simple and natural choice is to set: $\alpha_3^L = \alpha_5^L = \alpha_6^L = \alpha_7^L = \alpha_9^L = \alpha_{10}^L = \alpha_{12}^L = \alpha_{13}^L = 0$. The resolution of the system leads to the following values for the other coefficients:

$$\begin{cases} \alpha_1^L = \frac{c_2}{2} \\ \alpha_2^L = \frac{c_3}{2} \\ \alpha_4^L = \frac{1+c_2}{2} \\ \alpha_{11}^L = \frac{2}{3b_3} \approx 1.821 \\ \alpha_8^L = \frac{1+c_3}{2} - \frac{2}{3b_3} \approx -0.9270 \end{cases} \tag{18}$$

A similar analysis leads to the following values for the coefficients dedicated to the calculation of additional values in RKS method:

$$\begin{cases} \alpha_2^S = (\frac{2}{3} - \frac{b_2}{2})\frac{1}{b_3} \approx 1.2440 \\ \alpha_1^S = c_3 - \alpha_2^S \approx -0.45534 \\ \alpha_3^S = 0 \\ \alpha_4^S = 0 \\ \alpha_5^S = 0 \end{cases} \tag{19}$$

As for our previous second order scheme, the local time stepping scheme defined by RKL and RKS methods described in tables 17 and 18 is not mass conservative. The strategy based on the Total Flux Lost and the Total Flux Received at the interface between subdomains can be applied to make the scheme conservative. The algorithm of this local time stepping scheme applied to solve (2) is shown in table 19. In this table, $D_i^k = \frac{1}{\Delta x_i}(F_{i+\frac{1}{2}}(y_i^k, y_{i+1}^k) - F_{i-\frac{1}{2}}(y_{i-1}^k, y_i^k))$.

| | Large-time-scale-subdomain | | Short-time-scale-subdomain | |
|---|---|---|---|---|
| 1 | | $y_i^1 = y_i^n - \alpha_1^L \Delta t D_i^n$ | $y_i^1 = y_i^n - \frac{a_{21}\Delta t}{2} D_i^n$ | $y_i^1 = y_i^n - \frac{a_{21}\Delta t}{2} D_i^n$ |
| 2 | | $y_i^2 = y_i^n - \alpha_2^L \Delta t D_i^n$ | $y_i^2 = y_i^n - \frac{a_{31}\Delta t}{2} D_i^n$ $- \frac{a_{32}\Delta t}{2} D_i^1$ | $y_i^2 = y_i^n - \frac{a_{31}\Delta t}{2} D_i^n$ $- \frac{a_{32}\Delta t}{2} D_i^1$ |
| 3 | $y_i^3 = y_i^n - a_{21}\Delta t D_i^n$ | $y_i^3 = y_i^n - a_{21}\Delta t D_i^n$ | $y_i^3 = y_i^n - \frac{b_1\Delta t}{2} D_i^n$ $- \frac{b_2\Delta t}{2} D_i^1 - \frac{b_3\Delta t}{2} D_i^2$ | $y_i^3 = y_i^n - \frac{b_1\Delta t}{2} D_i^n$ $- \frac{b_2\Delta t}{2} D_i^1 - \frac{b_3\Delta t}{2} D_i^2$ |
| 4 | | $y_i^4 = y_i^n - \alpha_4^L \Delta t D_i^n$ $- a_{32}\Delta t D_i^3$ | $y_i^4 = y_i^3 - \frac{a_{21}\Delta t}{2} D_i^3$ | $y_i^4 = y_i^3 - \frac{a_{21}\Delta t}{2} D_i^3$ |
| 5 | $y_i^5 = y_i^n - a_{31}\Delta t D_i^n$ $- a_{32}\Delta t D_i^3$ | $y_i^5 = y_i^n - a_{31}\Delta t D_i^n$ $- a_{32}\Delta t D_i^3$ | $yi^5 = y_i^n - \alpha_1^S \Delta t D_i^n$ $- \alpha_2^S \Delta t D_i^1$ | |
| 6 | | $y_i^6 = y_i^n - \alpha_8^L \Delta t D_i^n$ $- \alpha_{11}^L \Delta t D_i^4$ | $y_i^6 = y_i^3 - \frac{a_{31}\Delta t}{2} D_i^3$ $- \frac{a_{32}\Delta t}{2} D_i^4$ | $y_i^6 = y_i^3 - \frac{a_{31}\Delta t}{2} D_i^3$ $- \frac{a_{32}\Delta t}{2} D_i^4$ |
| 7 | $y_i^{n+1} = y_i^n - b_1\Delta t D_i^n$ $- b_2\Delta t D_i^3 - b_3\Delta t D_i^5$ | $y_i^{n+1} = y_i^5 - b_1\Delta t D_i^n$ $- b_2\Delta t D_i^3 - b_3\Delta t D_i^5$ | $y_i^{n+1} = y_i^3 - \frac{b_1\Delta t}{2} D_i^3$ $- \frac{b_2\Delta t}{2} D_i^4 - \frac{b_3\Delta t}{2} D_i^6$ | $y_i^{n+1} = y_i^3 - \frac{b_1\Delta t}{2} D_i^3$ $- \frac{b_2\Delta t}{2} D_i^4 - \frac{b_3\Delta t}{2} D_i^6$ |
| 8 | | Correction step | | |
| | | Buffer zone | Buffer zone | |

Table 19: Algorithm of the second proposed scheme

This algorithm is written in the Runge-Kutta "classical form" but it can be implemented in the "low-storage form" because these forms are equivalent. In this scheme, there are two buffer zones because both RKL and RKS methods provide one or several additional values. The size of each buffer zone is equal to two cells in our applications (flux function with a four cell stencil). We remind that for the scheme of Constantinescu and Sandu based on RK3, there is one buffer zone of six cells. Our new scheme, despite the additional count due to the correction step, seems to lead to a small computational speedup compared to the scheme of Constantinescu and Sandu based on RK3 method (see section 4).

# 4 Numerical tests

In this section, some numerical test cases are used to perform comparisons between our schemes and the schemes developed with the strategy of Constantinescu and Sandu. The schemes based on RK2 and RK3 wich follow the strategy of Constantinescu and Sandu are denoted by CSRK2 and CSRK3, respectively. The schemes that we propose based on RK2 and RK3 are denoted by Prop. RK2 and Prop. RK3, respectively.

## 4.1 Two-dimensional vortex advection

This test case is based on 2D compressible Euler equations :

$$\begin{cases} \partial_t \rho + \partial_{x_j}(\rho u_j) = 0 \\ \partial_t \rho + \partial_{x_j}(\rho u_i u_j + p\delta_{ij}) = 0 \\ \partial_t(\rho e) + \partial_{x_j}(\rho e u_j + p u_j) = 0 \end{cases} \tag{20}$$

where $\rho$, $u_1$, $u_2$, $p$ are the fluid density, the fluid velocity in directions $\vec{x}$ and $\vec{y}$ and the fluid pressure, respectively. Total energy is denoted by $e$ and reads : $e = \frac{1}{\gamma - 1}\frac{p}{\rho} + \frac{u_1^2 + u_2^2}{2}$, with $\gamma \approx 1.40$.

The initial state is a 2D uniform flow ($\rho = 1, u_1 = 1, u_2 = 0, p = 1$) with the superposition of perturbations ($\delta\rho, \delta u_1, \delta u_2, \delta p$) which simulate a vortex. The perturbations read:

$$\begin{cases} \delta\rho = (1 + \delta T)^{\frac{1}{\gamma-1}} \\ \delta u_1 = \frac{\epsilon}{2\pi} e^{0.5(1-r^2)}(y_c - y) \\ \delta u_2 = \frac{\epsilon}{2\pi} e^{0.5(1-r^2)}(x - x_c) \\ \delta p = (1 + \delta T)^{\frac{\gamma}{\gamma-1}} \end{cases} \tag{21}$$

where $\delta T = -\frac{(\gamma-1)\epsilon^2}{8\gamma\pi^2} e^{1-r^2}$, is the temperature perturbation around the value $T = 1$. Note that the corresponding initial value and perturbation for the entropy are $S = 1$ and $\delta S = 0$, respectively. The coordinates $(x_c, y_c)$ are the coordinates of the the vortex center, $r = \sqrt{(x - x_c)^2 + (y - y_c)^2}$ is the radius of the vortex and $\epsilon = 5$ is the vortex strength. The initial solution of this problem reads:

$$\begin{cases} \rho^0 = 1 + \delta\rho \\ u_1^0 = 1 + \delta u_1 \\ u_2^0 = 0 + \delta u_2 \\ p^0 = 1 + \delta p \end{cases} \tag{22}$$

The solution of this problem is a simple advection of the vortex in the $\vec{x}$ direction. For more information about this test case, see [17].

The computational domain is rectangular (dimensions : $30 \times 20$) with periodic boundary conditions. It is divided into three zones. In the first and last zones, the grid spacings in directions $\vec{x}$ and $\vec{y}$ are $\Delta x = \Delta y = 0.05$. In the middle zone, the grid spacing in the $\vec{x}$ direction is refined by a factor 2 ($\Delta x = 0.025$ and $\Delta y = 0.05$). Thus, the time step $\frac{\Delta t}{2}$ is used in this zone and the time step $\Delta t$ is applied in the first and last zones. The mesh and the density $\rho$ at initial state are shown in figure 2. Figure 3 also shows the density profile at initial state and at $y = 10$. The spacial scheme used for the Euler flux discretization is a hybrid centred/upwind version of the scheme AUSM+(P) proposed by Mary and Sagaut [18].

Figure 4 shows the logarithm ($log_{10}$) of the $L_2$ error against the logarithm of the time step $\Delta t$ after the simulation time 14.5, for each local time stepping scheme. The $L_2$ error is the difference (computed in $L_2$ norm) between the density obtained with the local time stepping schemes and the density computed with a reference scheme. The reference scheme is the classical RK4 explicit method used with the reference time step $\Delta t_{ref} = 5.10^{-4}$ ($CFL = 0.04$). For each scheme, we make four simulations in which the time step for the first and last zones are: $2\Delta t_{ref}$, $4\Delta t_{ref}$, $8\Delta t_{ref}$ and $16\Delta t_{ref}$, respectively. In this figure, each mark's ordinate corresponds to the $L_2$ error and the abscissa is the time step used for the first and last zones. Note that the slope of each curve is in accordance with the theoretical temporal order of the local time stepping schemes. We can notice that the $L_2$ errors of the original scheme of Constantinescu and Sandu and our proposed scheme based on RK2 are the same. Our proposed scheme based on RK3 has the lowest $L_2$ error.

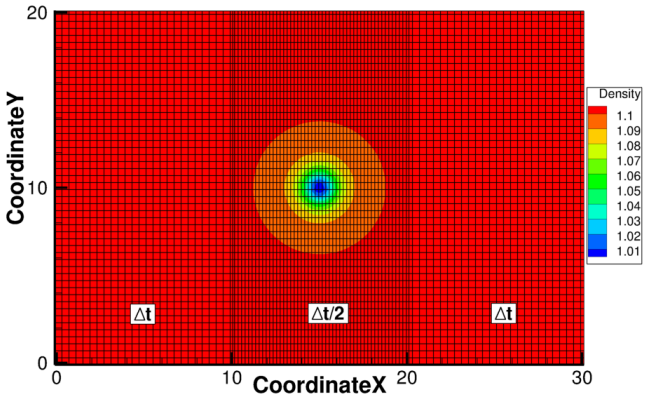


Figure 2: Variable density and mesh at initial state in 2D

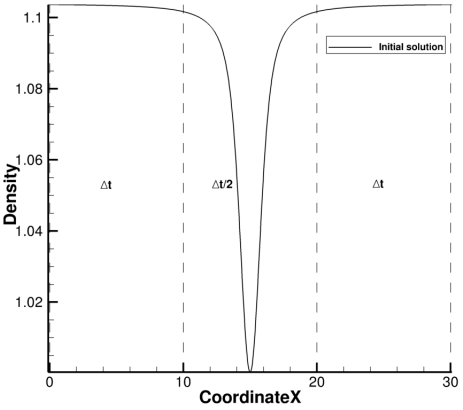


Figure 3: Variable density at initial state and at $y = 10$

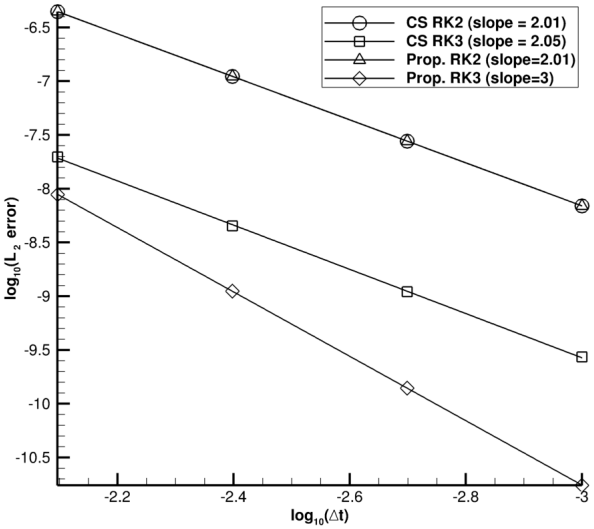


Figure 4: Logarithm ($log_{10}$) of the $L_2$ errors against the logarithm of the time step $\Delta t$ for each local time stepping scheme

Other characteristics of each scheme are studied. They are listed in the table 20 below. For this study, the computation is performed to achieve the simulation time 43.5. At this time, the vortex has returned approximately to its initial position (we remind that we are using periodic boundary conditions). Each scheme has been used with its maximum allowable CFL(its maximum allowable time step), shown in the row "CFLmax ($\Delta t_{max}$)" of table 20. The row "speedup" is the relative difference between the computational time of the classical RK2 explicit scheme used with its maximum allowable time step and the computational time of the local time stepping schemes. Finally, the row "Conservation defect" is the relative difference between the total fluid mass in the computational domain at final time $m(t_f)$ and at initial time $m(t_0)$ :

$$\text{Conservation defect} = \frac{m(t_f) - m(t_0)}{m(t_0)}$$

| | CFLmax($\Delta t_{max}$) | Speedup | Conservation deffect |
|---|---|---|---|
| CS RK2 | 0.25 ($\Delta t_{max} = 0.007$) | 21% | 3.4 $10^{-16}$ |
| Prop. RK2 | 0.25 ($\Delta t_{max} = 0.007$) | 25% | 1.7 $10^{-16}$ |
| CS RK3 | 1.1 ($\Delta t_{max} = 0.029$) | 73% | 1.7 $10^{-16}$ |
| Prop. RK3 | 1.1 ($\Delta t_{max} = 0.029$) | 74% | 9.1 $10^{-16}$ |

Table 20: Comparison of some indicators for the local time stepping schemes

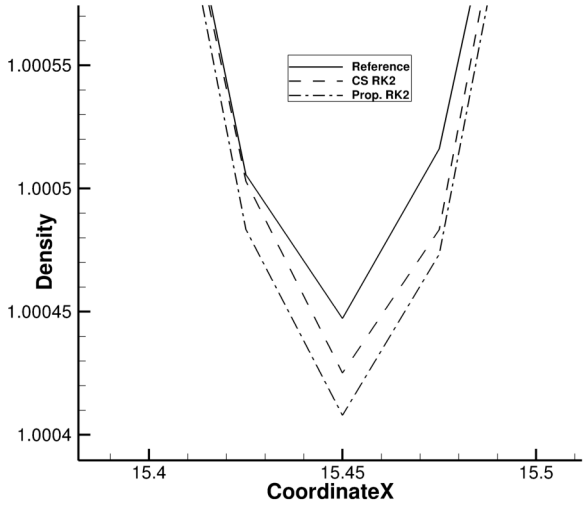


Figure 5: "Zoom" on the minimum of Density for schemes Prop.RK2 and CS RK2

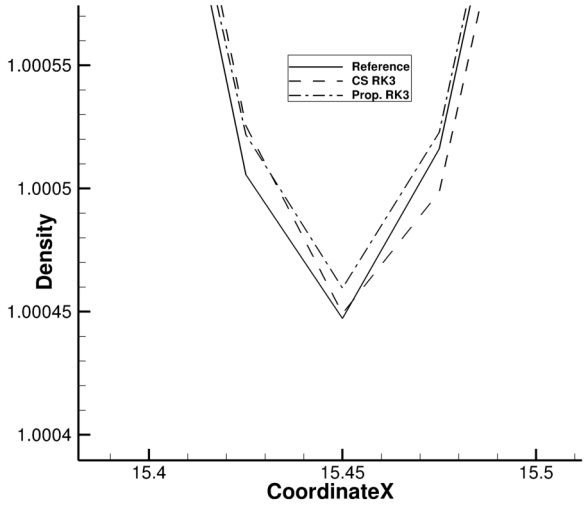


Figure 6: "Zoom" on the minimum of Density for schemes Prop.RK3 and CS RK3

Table 20 shows that our proposed local time stepping schemes have equivalent or slight better speedup than schemes of Constantinescu and Sandu based on the same method. Our proposed scheme based on RK3 has a slightly larger conservation defect than the other schemes. The local time stepping schemes based on RK3 method are more stable than the local time stepping schemes based on RK2 method. This better stability allows to use larger time steps, which significantly improves the speedup.

Finally, figures 5 and 6 show the minimum of the density profile along $y = 10$ for each local time stepping scheme and for the reference solution, at the final time 43.5. The reference solution mentioned in this figure is obtained with a classical RK4 explicit scheme with $\Delta t = 0.0035$. These figures show that solutions obtained with the local time stepping schemes based on RK3 method are closer to the reference solution, already as shown with the $L_2$ error curves (figure 4).

## 4.2 Sod shock tube

This test case is based on compressible 2D Euler equations (20). The initial state is given by :

- $\rho^0(x) = 1$, $p^0(x) = 1$ for $x \leq 9$,
- $\rho^0(x) = 0.125$, $p^0(x) = 0.1$ for $x > 9$,

while the initial velocity is zero in the whole domain.

The computational domain is rectangular ($18 \times 0.6$). It is divided into three zones of length 6. In the first and last zones, the grid spacings in the $\vec{x}$ direction ($\Delta x$) and in the $\vec{y}$ direction ($\Delta y$) are: $\Delta x = \Delta y = 0.03$. In the middle zone, $\Delta x = 0.015$ and $\Delta y = 0.03$. Thus, the time step $\frac{\Delta t}{2}$ is used in the middle zone and the

time step $\Delta t$ is applied in the first and last zones. The different zones and the initial state for the density profile at $y = 0.3$ are shown in figure (7). The spacial scheme used for this test case is the Roe scheme [19] with the minmod flux limiter [20].

Figure 8 shows the density profile at $y = 0.3$ for each local time stepping scheme and for the reference solution at the final time 4.96. The reference solution mentioned in this figure is obtained by using the classical RK4 explicit scheme with the time step $\Delta t = 0.001$ ($CFL = 0.14$). The local time stepping schemes are used with their maximal allowable time step shown in table 21 below. Figure 8 shows a very good agreement between the reference solution and the solutions obtained with the local time stepping schemes.

Figures 9 and 10 present a "zoom" on the shock area for the density at final time. These figures show that the solutions obtained with schemes CS RK3 and Prop. RK3 are closer to the reference solution than solutions obtained with schemes CS RK2 and Prop. RK2.

Finally, table 21 also shows the speedup as compared to a classical RK2 explicit scheme used with its maximal allowable time step. Our local time stepping schemes have an equivalent or a slight better speedup than the schemes of Constantinescu and Sandu based on the same RK method.

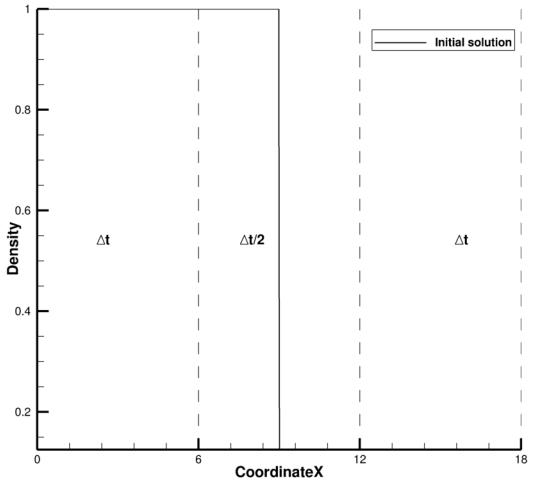


Figure 7: Density at initial time and the different zones with their time step

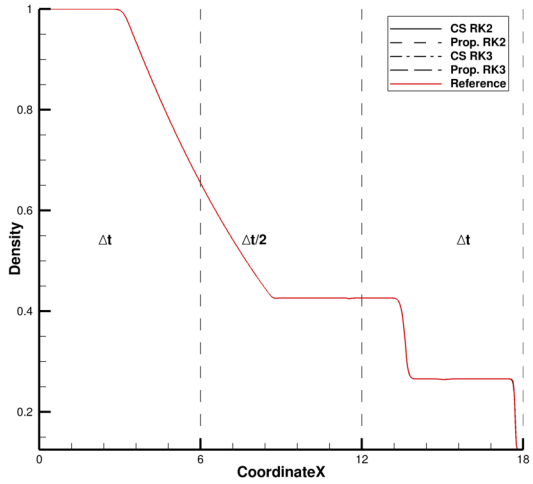


Figure 8: Density at final time

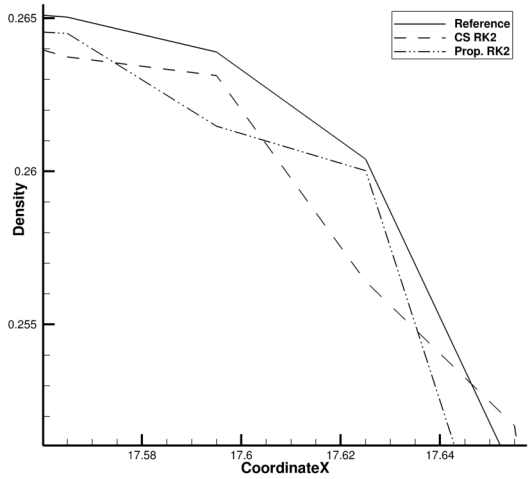


Figure 9: Density at final time in the shock area for the schemes CS RK2 and Prop. RK2

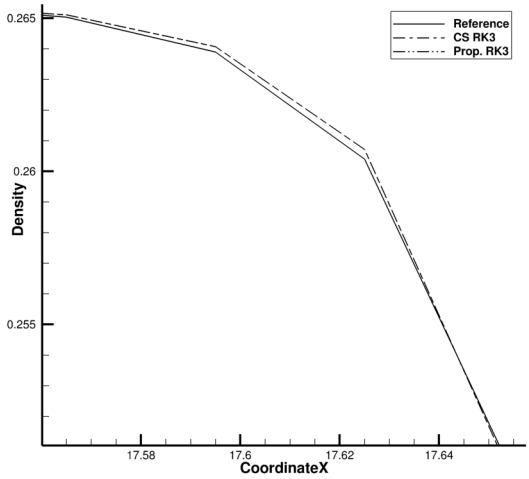


Figure 10: Density at final time in the shock area for the schemes CS RK3 and Prop. RK3

| | CFLmax ($\Delta t_{max}$) | Speedup |
|---|---|---|
| CS RK2 | 0.75 ($\Delta t_{max} = 0.01$) | 19% |
| Prop. RK2 | 0.75 ($\Delta t_{max} = 0.01$) | 19% |
| CS RK3 | 0.92 ($\Delta t_{max} = 0.0125$) | 26% |
| Prop. RK3 | 0.92 ($\Delta t_{max} = 0.0125$) | 29% |

Table 21: Comparison of the maximum CFL and the speedup of the local time stepping schemes for the Sod shock tube.

### 4.3 Taylor-Green Vortex

This test case is based on 3D Navier-Stokes equations :

$$\begin{cases} \partial_t \rho + \partial_{x_j}(\rho u_j) = 0 \\ \partial_t \rho + \partial_{x_j}(\rho u_i u_j + p\delta_{ij} - \tau_{ij}) = 0 \\ \partial_t(\rho e) + \partial_{x_j}(\rho e u_j + p u_j - \tau_{jk} u_k + q_j) = 0 \end{cases} \tag{23}$$

where $\rho$ is the fluid density, $u_i$ are the fluid velocity components, $e$ is the total specific energy of the fluid, $p$ is the fluid pressure and $q_j$ is the heat flux. The fluid follows the law of perfect gases. The tensor of viscous constraints $\tau_{ij}$ follows the law of Newtonian fluids.

The computational domain is a cube of dimension $2\pi L_0$ ($L_0 = 1$ is a reference length), with periodic boundary conditions. The initial conditions for variables $u_1$, $u_2$, $u_3$, $p$ and $\rho$ read :

$$\begin{cases} u_1^0 = U_0 \sin(x/L_0)\cos(y/L_0)\cos(z/L_0) \\ u_2^0 = -U_0 \cos(x/L_0)\sin(y/L_0)\cos(z/L_0) \\ u_3^0 = 0 \\ p^0 = P_0 + \rho_0 U_0^2/16(\cos(x/L_0) + \cos(y/L_0))(\cos(2z/L_0) + 2) \\ \rho^0 = p/(RT_0) \end{cases} \tag{24}$$

where $P_0 = 101183Pa$, $\rho_0 = 1.2kg.m^{-3}$, $T_0 = 294K$ and $U_0 = 34.38m.s^{-1}$. At the initial state, the computational domain contains several vortices. The spacial scheme used for the Euler flux discretization is a hybrid centred/upwind version of the scheme AUSM+(P) proposed by Mary and Sagaut [18]. Viscous fluxes are discretized with a classical second order centered scheme. The simulation is performed during a period of $16L_0/U_0$. At the final state, numerous turbulent scales are developped.

The computational domain is divided into three zones of equal length, as shown in figure 11. In the first and last zones, the grid spacings in the $\vec{x}$ direction ($\Delta x$), $\vec{y}$ direction ($\Delta y$) and $\vec{z}$ direction ($\Delta z$)are $\Delta x = \Delta y = \Delta z = 0.05$, respectively. In the middle zone, the mesh is refined by a factor two in the $\vec{x}$ direction ($\Delta x = 0.025$, $\Delta y = \Delta z = 0.05$). In this zone, the time step $\frac{\Delta t}{2}$ is used while the time step $\Delta t$ is applied in the other zones. For each scheme, the enstrophy $\epsilon$ is computed at different times of the simulation. This variable reads :

$$\epsilon = \frac{L_0}{U_0^2 \rho_0} \int_\Omega \rho \vec{w}.\vec{w} d\Omega, \tag{25}$$

where $\vec{w}$ denotes the vorticity, defined by : $\vec{w} = \vec{\nabla} \wedge \vec{V}$ (where $\vec{V} = (u_1, u_2, u_3)$). The enstrophy has a high sensibility to the numerical method. Figure 12 shows the evolution of the enstrophy for each scheme. The reference solution mentioned in this figure is obtained with a classical RK4 explicit scheme used with

the time step $\Delta t_{ref} = 0.00015$ ($CFL = 0.03$). Each local time stepping is used with its maximal allowable time step, shown in the first row of table 22. A good agreement is observed between the reference solution and the solutions obtained with the local time stepping schemes.

Table 22 also shows the speedup obtained with the local time stepping schemes as compared to a classical RK2 explicit scheme, and the conservation defect for each scheme. Our local time stepping schemes are slightly faster than the schemes based on the strategy of Constantinescu and Sandu. Moreover, the conservation defects obtained with our schemes are very close to the ones calculated with the schemes of Constantinescu and Sandu.

Finally, figure 13 shows the logarithm of $L_2$ error as a function of the logarithm of the time step $\Delta t$ for each local time stepping scheme. The calculation of the $L_2$ error is carried out with respect to the density obtained with the reference solution. In this figure, each mark is associated to the use of the time steps : $2\Delta t_{ref}, 4\Delta t_{ref}$ and $8\Delta t_{ref}$, for the first and last zones. The slope of each curve is in accordance with the theoretical temporal order of each local time stepping scheme. Our proposed scheme based on RK3 has the lowest $L_2$ error.

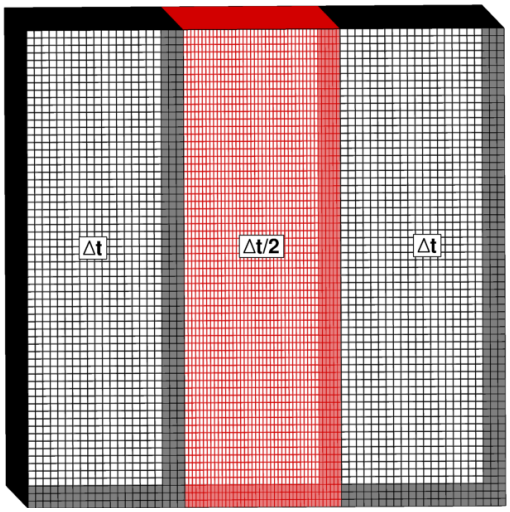


Figure 11: Mesh used for the Taylor-Green Vortex test case

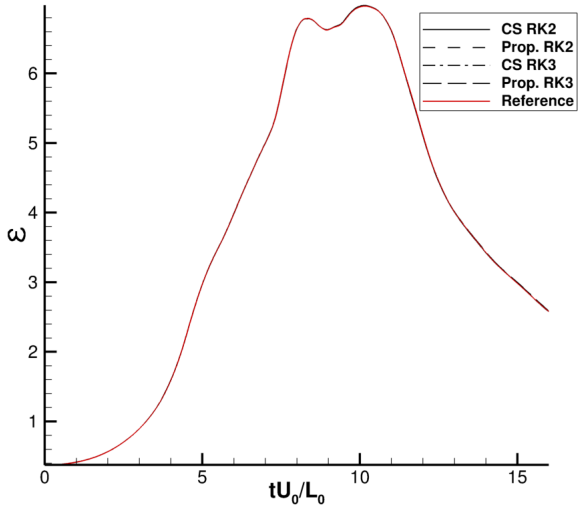


Figure 12: Evolution of enstrophy for each scheme

| | CFLmax($\Delta t_{max}$) | Speedup | Conservation defect |
|---|---|---|---|
| CS RK2 | 0.3 ($\Delta t_{max} = 0.0014$) | 20% | $8,1.10^{-12}$ |
| Prop. RK2 | 0.3 ($\Delta t_{max} = 0.0014$) | 27% | $7,9.10^{-12}$ |
| CS RK3 | 0.9 ($\Delta t_{max} = 0.004$) | 57% | $7,5.10^{-12}$ |
| Prop. RK3 | 0.9 ($\Delta t_{max} = 0.004$) | 59% | $8,5.10^{-12}$ |

Table 22: Taylor-Green Vortex: comparison of some indicators ofr the different local time stepping schemes

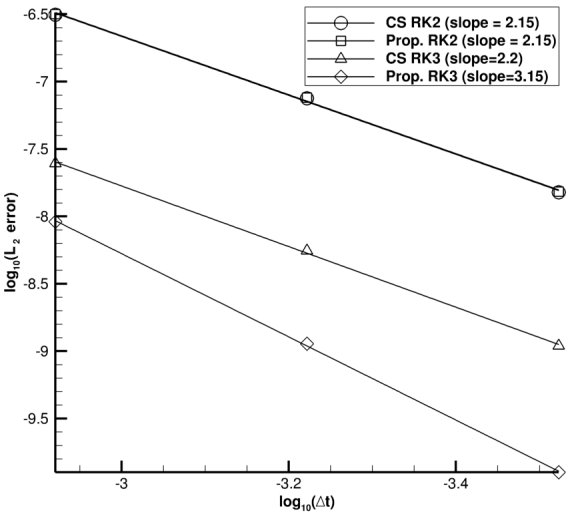


Figure 13: Logarithm ($log_{10}$) of the $L_2$ errors as functions of the logarithm of the time step $\Delta t$ for each local time stepping scheme

## 4.4 Turbulent flow around an airfoil

The previous test cases show that our third order local time stepping scheme is the most accurate. Moreover, it is slightly faster than the other local time stepping schemes presented in this article. Consequently, our third order local time steeping scheme (Prop. RK3) is chosen to compute a more complex test case.

The test case is based on 3D Navier-Stokes equations (24). The airfoil is the SD7003 model. For the computation, the Mach number $M$ and the Reynolds number $Re$ are set to $M = 0.1$ and $Re = 60000$, respectively. The value of the angle of attacks $\alpha$ is: $\alpha = 8°$. The computational domain is a 3D structured mesh with approximately 90 million cells. Figure 14 shows a section (in the xy map) of the mesh, with a reduced number of cells. The outer boundary conditions are set to "subsonic outflow". The airfoil surface is an isothermal wall, and periodic boundary conditions are set in the $\vec{z}$ direction. With the initial solution, the CFL number is calculated in each cell. Thanks to the repartition of the CFL number, the mesh is divided into zones of different time steps. The repartition of the different time steps is shown in figure 15. The smallest time steps ($\frac{\Delta t}{2}$, $\frac{\Delta t}{4}$, $\frac{\Delta t}{8}$) are localized close to the airfoil. The maximum time step $\Delta t$ is set to $\Delta t = 1.10^{-4}$. The spacial scheme used for convective fluxes discretization is a hybrid centred/upwind version of the scheme AUSM+(P) [18]. Viscous fluxes discretization is carried out with a spacial centered scheme. We run $5.10^4$ time iterations to achieve the final time $t_f = 5$. 5000 Statistical samples are accumulated during the simulation.

A simulation with a classical explicit scheme (RK3) is also performed. The time step used for this simulation is $\Delta t = 1.25.10^{-5}$. We run $4.10^5$ time iterations to achieve the final time $t_f = 5$. 40 0000 Statistical samples are accumulated during the simulation.

The Q-criterion $Q$ (that is an indicator of turbulent structures) is computed for the local time stepping scheme. The Q-criterion is defined by the formula:

$$Q = \frac{1}{2}\left[(tr(\nabla u))^2 - tr(\nabla u.\nabla u)\right], \tag{26}$$

where $u = (u_1, u_2, u_3)$, is the vector of the fluid velocity components. Figure 16 shows the iso-surface of $Q = 1.5$, coloured with the fluid velocity magnitude.

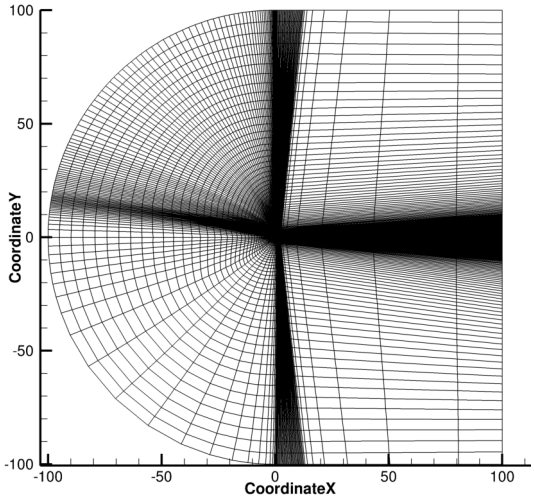


Figure 14: Mesh used for the computation of a flow over a SD7003 Wing

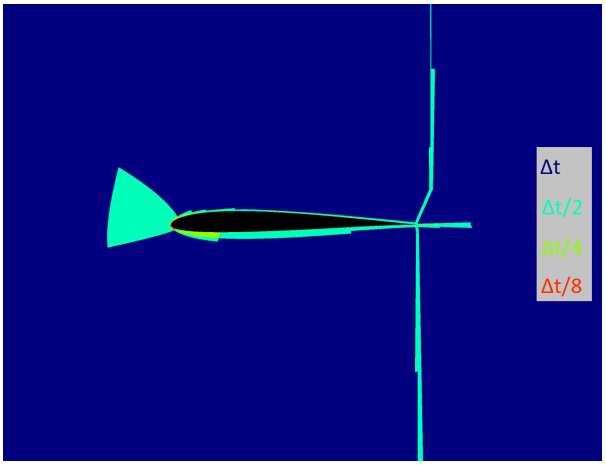


Figure 15: Repartition of the different time steps over the mesh

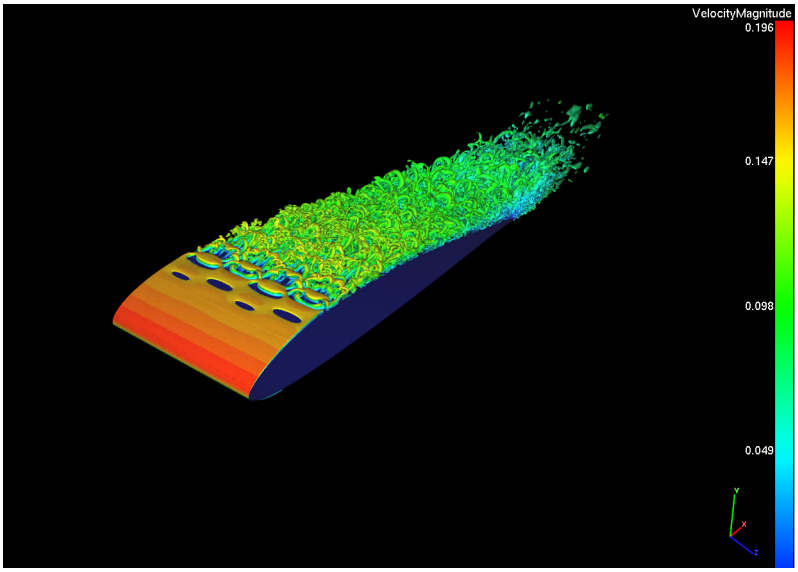


Figure 16: Iso-surface of $Q = 1.5$ coloured with the fluid velocity magnitude

The transition occurs in a laminar separation bubble. After reattachment, a 3D turbulent flow is observed. The mean x-velocity $< u >$ and the mean Reynolds stresses $< \rho u^{'} v^{'} >$ are computed in the transition, reattachment and mid-chord zones for each scheme. The curves are shown in figures 18 and 17, respectively.

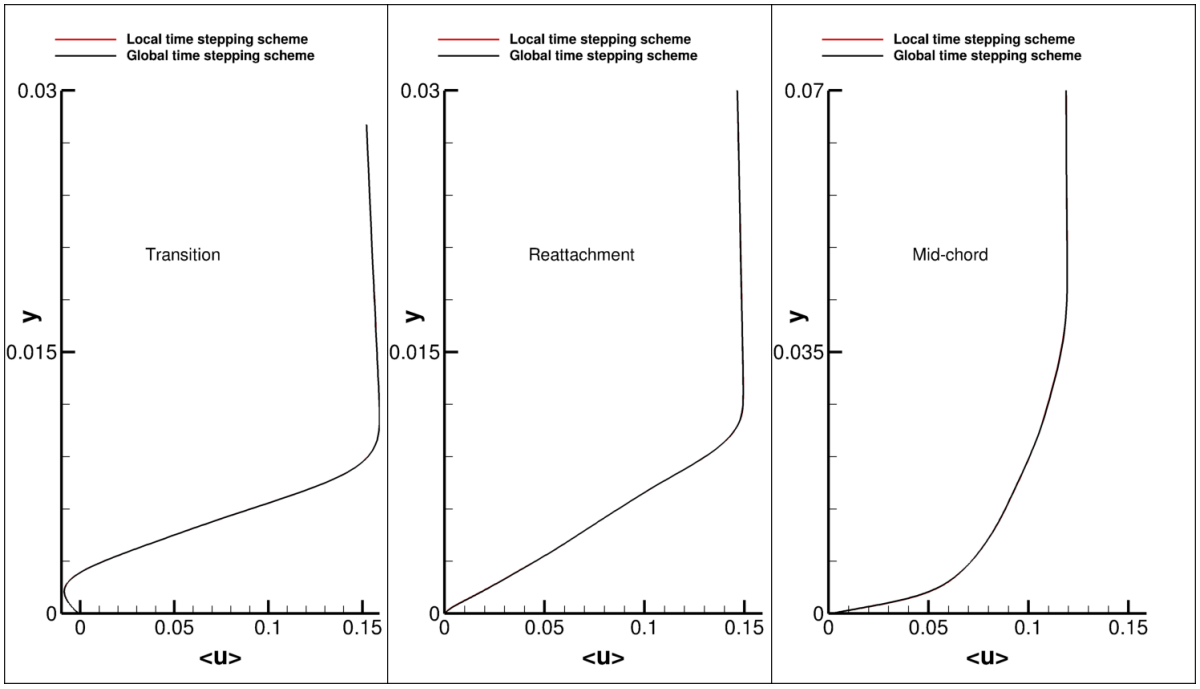


Figure 17: Mean x-velocity in the transition, reattachement and mid-chord zones

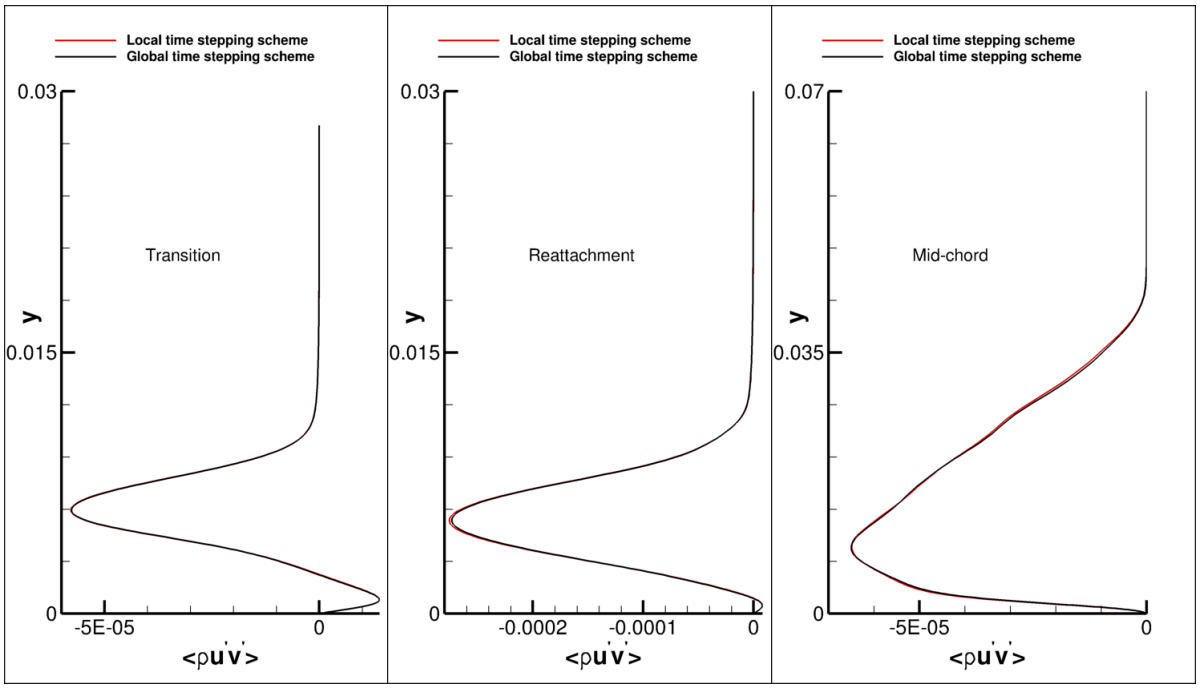


Figure 18: Mean Reynolds stresses in the transition, reattachement and mid-chord zones

A perfect agreement is observed between the local time stepping scheme and the global time stepping scheme. Note that our local time stepping turns out to be about three times as fast as the classical explicit scheme on this test case.

# 5 Conclusion

The diversity of spatial scales in fluid mechanics problems leads to the use of several cell sizes in CFD meshes. It is accompanied by a wide range of numerical stability conditions for explicit time integration methods. Local time stepping schemes are well suited to this problems and allow to obtain computational speedup as compared to classical explicit schemes. However, the construction of local time stepping schemes requires a particular attention to ensure temporal high order and mass conservation. Constantinescu and Sandu [4] have developped an interesting strategy which allows to construct mass conservative and second order accurate local time stepping schemes. Their strategy is based on the theory of Partitionned Runge-Kutta (PRK) methods. We also use the theory of PRK methods as well as consistent interpolations to develop two local time stepping schemes of order two and three. Our schemes are made conservative with an additional correction step. Our numerical tests reveal that our schemes are equivalent or more accurate than the schemes of [4], and slightly faster. Our strategy seems to be a good approach to construct efficient, accurate and conservative local time stepping schemes. Parallel implementation of our local time stepping schemes is a topic that we have not work on yet. A well balanced parrallelization strategy is necessary to perform large scale simulations. The implementation of an efficient parallelization strategy for our local time stepping schemes is the next step of our work.

# Acknowledgement

This work has been supported by ONERA (Office Nationale d'Etudes et de Recherches Aérospatiales) and DGA (Direction Générale de l'Armement)

# References


[1] T. Unfer. An asynchronous framework for the simulation of the plasma/flow interaction. *Journal of Computational Physics*, 2013.

[2] A. Toumi. Asynchronous numerical scheme for modeling hyperbolic systems. *C. R. Acad. Sci. Paris, Ser I*, 2015.

[3] V. Semiletov and S. Karabasov. Cabaret scheme for computational aero acoustics: Extension to asynchronous time stepping and 3d flow modelling. *International Journal of Aeroacoustics*, 2014.

[4] E. M. Constantinescu and A. Sandu. Multirate timestepping methods for hyperbolic conservation laws. *Journal of Scientific Computing , Vol 33, pages 239-278*, 2007.

[5] S. Osher and R. Sanders. Numerical approximations to nonlinear conservation laws with locally varying time and space grids. *Mathematics of computation, Vol 41, Number 164*, October 1983.

[6] H. Tang and G. Warnecke. A class of high resolution schemes for hyperbolic conservation laws and convection-diffusion equations with varying time and space grids. *Journal of Scientific Computing*, 2005.

[7] M.Schlegel, O.Knoth, M.Arnold, and R.Wolke. Multirate runge-kutta schemes for advection equations. *Journal of computational and applied mathematics*, 2009.

[8] L. Liu, X. Li, and F.Q. Hu. Nonuniform time-step runge-kutta discontinuous galerkin method for computational aeroacoustics. *Journal of Computational Physics*, 2010.

[9] L. Liu, X. Li, and F.Q. Hu. Nonuniform time-step explicit runge-kutta scheme for high-order finite difference method. *Computers and Fluids*, 2014.

[10] B. Seny, J. Lambrechts, R. Comblen, V. Legat, and J.-F. Remacle. Multirate time stepping for accelerating explicit discontinuous galerkin computations with application to geophysical flows. *International Journal For Numerical Mehtods In Fluids*, 2007.

[11] V. Savcenco, W. Hundsdorfer, and J. G. Verwer. A multirate time stepping strategy for stiff ordinary differential equations. *BIT numerical mathematics*, 2006.

[12] C. Dawson and R. Kirby. High resolution schemes for conservation laws with locally varying time steps. *Society for Industrial and Applied Mathematics*, 2001.

[13] W. Hundsdorfer, A. Mozartova, and V. Savcenco. Analysis of explicit multirate and partitioned runge-kutta schemes for conservation laws. *Report of Centrum voor Wiskunde en Informatica*, 2007.

[14] E.Hairer, G.Wanner, and S.P. Norsett. *Solving ordinary differential equations I : Nonstiff Problems.* Springer, Berlin, 1993.

[15] J.H. Williamson. Low-storage runge-kutta schemes. *Journal of computational physics*, 1980.

[16] P.S. Lowery and W. C. Reynolds. Numerical simulation of a spatially-developping, forced, plane mixing layer. *Report of the Stanford University, TF26*, 1986.

[17] C. Hu and C. W. Shu. Weighted essentially non-oscillatory schemes on triangular meshes. *Journal of Computational Physics*, 1999.

[18] I. Mary and P. Sagaut. Large eddy simulation of flow around an airfoil near stall. *AIAA Journal*, 2002.

[19] P. L. Roe. Approximate riemann solvers, parameter vectors and difference schemes. *Journal of computational physics*, 1981.

[20] P. L. Roe. Characteristic-based schemes for the euler equations. *Annu. Rev. Fluid Mech.*, 1986.