Frédéric SICOT

Thesis for the degree Master of Science

High Order Schemes on Non-uniform Structured Meshes in a Finite-Volume Formulation

Application in Computational Fluid Dynamics

Mathematical Sciences
CHALMERS UNIVERSITY OF TECHNOLOGY
Göteborg Sweden 2006

CERFACS
Toulouse – France
Supervisor: Jean-François BOUSSUGE

Abstract

In this thesis, a new family of high-order implicit schemes is developed. They are designed to work on non-uniform grids by taking the metric of the meshes into account. They also have a better resolution than classical explicit schemes and so can properly represent a wider range of scales.

After the derivation of the schemes, they are tested in Matlab to easily check the expected properties. They are used to solve partial differential equations by mean of finite-volume method and showed a good behavior.

These schemes are then integrated in a Computational Fluid Dynamics (CFD) software co-developed by ONERA (The French National Aerospace Research Establishment), CERFACS (European Centre for Research and Advanced Training in Scientific Computation) and AIRBUS. This software, called *elsA* (french acronym meaning *Software Package for Aerodynamics Simulation*), solves compressible flows by mean of finite-volume method on multi-blocks structured grids. Different methods are developed to correctly handle the domain decomposition.

Contents

List of Figures

Introduction

In the solution of many physical problems, one is often confronted with flows possessing a wide range of space and time scales. In order to resolve all the scales of such flows, highly accurate methods, which can properly represent all the scales of the solution, are needed.

In recent years, so called compact (or PADÉ-type) schemes have gained increasing popularity in application such as Direct Numerical Simulation (DNS) or Large Eddy Simulation (LES). The work of LELE [8] contributed greatly to the diffusion of these schemes. Their main advantage is that they provide a better representation of the shorter length scales of the solution as compared to classical finite-difference and finite-volume schemes. This better representation of the shorter length scales can be attributed to the implicit nature of these schemes.

Originally, such schemes were defined in the finite-difference context, in one dimension and on a uniform grid. Then, GAMET et al. [4] take the non-uniformity of the grid into account.

The finite-difference approach is rather easy to implement but special attention must be paid to the conservation properties of the scheme, as it is not guaranteed. On the other hand, the finite-volume (FV) method is inherently conservative and is preferred by the CFD practitioners in industry: it is (almost) as easy to implement as finite-difference method and easier than the finite-element method. Unfortunately, only few papers dealing with compact schemes in the FV context can be found in the literature. To my knowledge, KOBAYASHI was the first to describe high-order PADÉ-type schemes in the FV context in [6].

While the other papers only deal with centered schemes, [1] describes upwind schemes derived from a main formula. By removing some coefficients, the authors are able to describe a family of schemes with different properties on a uniform grid. Upwind schemes are useful for correct boundary treatments.

All the previous papers developed 1D schemes. 2D and 3D computations are still feasible thanks to dimensional splitting: a multidimensional problem is simply split into a sequence of one-dimensional problems. This method works only on cartesian grids. In [7], LACOR et al. extend these schemes on arbitrary 2D grids (*i.e.* non-cartesian, non-uniform). They not only use the adjacent cells of the interface where the fluxes are computed, but also cells below and above. The derivation of such schemes is very complex and they can achieve only 2^{nd} order accuracy. Extending their method in 3D and higher accuracy (6^{th} order for instance) soon appears of very high difficulty and is furthermore useless in practice since it would require too much computation time and memory space.

So in this thesis, a new family of 1D schemes with a maximum accuracy of order 4 and 6 on non-uniform grid is developed. Based on the work by LELE and GAMET, the present schemes are derived using ideas of LACOR ([7] and [1]). Dimensional splitting is used so that the schemes works only on non-uniform cartesian grids. Nevertheless this is a very efficient approach as it is simple and relatively inexpensive way to extend to multidimensional problems compared to the way chosen in [7]. And even if the present

schemes are not design to work on arbitrary grids (*i.e.* non cartesian), a better accuracy than standard schemes is expected.

The new family is implemented in the CFD software *elsA* where the time and spatial discretizations are implemented separately. For time integration one can chose, for instance, an implicit scheme such as backward Euler or an explicit scheme such as Runge-Kutta. In this thesis, the focus is only on spatial discretization and so on semi-discretized equations (with the time derivative still appearing). The properties (order, stability...) of the global scheme in space and time depend, of course, of the time integration algorithm chosen by the user.

Chapter 1

Derivation of the schemes

Let's take the example of the advection equation:

$$\frac{\partial f(x,t)}{\partial t} + v\frac{\partial f(x,t)}{\partial x} = 0. \tag{1.1}$$

This equation describes the passive advection of some scalar field $f(x,t)$ carried along by a flow at constant speed v. It has an analytical solution: $f(x,t) = f_0(x - vt)$ where f_0 is the initial condition.

In the 1D mesh sketched in Fig. 1.1, there are 4 cells numbered from $i-1$ to $i+2$. The right interface (or cell face) of cell i is numbered $i + 1/2$. The width of cell i is denoted by h_i: $h_i = x_{i+1/2} - x_{i-1/2}$.

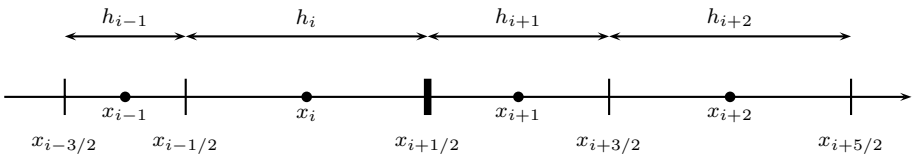


Figure 1.1: Non-uniform 1D mesh

In the FV approach, Eq. (1.1) is integrated over the mesh cells:

$$\frac{\partial}{\partial t}\int_{x_{i-1/2}}^{x_{i+1/2}} f(x)\,\mathrm{d}x + v(f_{i+1/2} - f_{i-1/2}) = 0, \quad \forall i. \tag{1.2}$$

This is still an exact relation, even if the mesh is non-uniform. Eq. (1.2) depends on the values of f at the cell faces. The present schemes aim to interpolate f at the cell-faces knowing the cell-center values. Before deriving them, one needs to choose whether pointwise values or cell-average values of f are used. The pointwise value f_i for instance, can be defined in the center of the cell i: $f_i = f(x_i)$. The cell-average value of cell i, denoted $\bar{f}_i$ can be defined as:

$$\bar{f}_i = \frac{1}{h_i}\int_{x_{i-1/2}}^{x_{i+1/2}} f(x)\,\mathrm{d}x. \tag{1.3}$$

In a pointwise approach, the spatial discretization scheme does not only require f to be approximated with high-order accuracy but also the integral (appearing after the time derivative). Whereas in a cell-average approach, Eq. (1.2) is simplified to:

$$\frac{\partial \bar{f}_i}{\partial t}h_i + v(f_{i+1/2} - f_{i-1/2}) = 0, \quad \forall i. \tag{1.4}$$

The integral does not have to be evaluated since its value is stored (it is the cell-average value). This approach seems therefore preferable as it is related to the integral form of conservation law. Note that the use of cell-average values was already advocated by KOBAYASHI [6].

Next the values $f_{i+1/2}$ and $f_{i-1/2}$ of f at the interfaces have to be evaluated. In the next sections, the present schemes will be compared with two classical schemes of order 2 (denoted CE2) and order 4 (denoted CE4). They are both explicit and read:

$$\begin{cases} f_{i+1/2} = \frac{1}{2}\left(\bar{f}_i + \bar{f}_{i+1}\right), & \text{(CE2)} \\ f_{i+1/2} = -\frac{1}{12}\left(\bar{f}_{i-1} + \bar{f}_{i+2}\right) + \frac{7}{12}\left(\bar{f}_i + \bar{f}_{i+1}\right), & \text{(CE4).} \end{cases} \tag{1.5}$$

CE2 simply computes the average of the two adjacent cells and CE4 has a stencil of four cells to reach 4$^{\text{th}}$ order accuracy. The given order is only true on uniform grid. When the mesh is non-uniform, an error term of order 1 appears. The present schemes are now developed to take into account the metric of the grid.

1.1 General scheme

Because of constraints imposed by the future implementation in *elsA*, two cells on each side of an interface are used, that is 4 cells total. Then for the implicit part, the system should not be more complex than a tridiagonal system as it would be computationally too expensive to solve.

For any scalar quantity u, the most general form of this kind of scheme about interface $i+1/2$ reads:

$$\beta\tilde{u}_{i-1/2} + \tilde{u}_{i+1/2} + \gamma\tilde{u}_{i+3/2} = a\bar{u}_{i-1} + b\bar{u}_i + c\bar{u}_{i+1} + d\bar{u}_{i+2}, \tag{1.6}$$

where $\bar{u}_k$ is the known cell-average value of u in cell k and $\tilde{u}$ is the computed approximation of u at the interfaces. The relations between the coefficients a, b, c, d and β, γ are derived by matching the Taylor series coefficients of various order (the full methodology is detailed in appendix A). The first unmatched coefficient determines the formal truncation error of Eq. (1.6). These constraints are:

- order 1:

$$1 + \beta + \gamma = a + b + c + d. \tag{1.7}$$

- order 2:

$$\gamma h_{i+1} - \beta h_i = \frac{1}{2}\Big[- a(2h_i + h_{i-1}) - bh_i + ch_{i+1} + d(2h_{i+1} + h_{i+2})\Big]. \tag{1.8}$$

- order 3:

$$\begin{aligned} \beta h_i^2 + \gamma h_{i+1}^2 = \frac{1}{3}\Bigg[& \frac{a}{h_{i-1}}\Big((h_i + h_{i-1})^3 - h_i^3\Big) + bh_i^2 \\ & + ch_{i+1}^2 + \frac{d}{h_{i+2}}\Big((h_{i+1} + h_{i+2})^3 - h_{i+1}^3\Big)\Bigg]. \end{aligned} \tag{1.9}$$

- order 4:

$$\begin{aligned} \gamma h_{i+1}^3 - \beta h_i^3 = \frac{1}{4}\Bigg[& \frac{a}{h_{i-1}}\Big(h_i^4 - (h_i + h_{i-1})^4\Big) - bh_i^3 \\ & + ch_{i+1}^3 + \frac{d}{h_{i+2}}\Big((h_{i+1} + h_{i+2})^4 - h_{i+1}^4\Big)\Bigg]. \end{aligned} \tag{1.10}$$

- order 5:

$$\beta h_i^4 + \gamma h_{i+1}^4 = \frac{1}{5}\left[\frac{a}{h_{i-1}}\left((h_i + h_{i-1})^5 - h_i^5\right) + b h_i^4 + c h_{i+1}^4 + \frac{d}{h_{i+2}}\left((h_{i+1} + h_{i+2})^5 - h_{i+1}^5\right)\right]. \tag{1.11}$$

- order 6:

$$\gamma h_{i+1}^5 - \beta h_i^5 = \frac{1}{6}\left[\frac{a}{h_{i-1}}\left(h_i^6 - (h_i + h_{i-1})^6\right) - b h_i^5 + c h_{i+1}^5 + \frac{d}{h_{i+2}}\left((h_{i+1} + h_{i+2})^6 - h_{i+1}^6\right)\right]. \tag{1.12}$$

As there are six coefficients to find, as many constraints can be satisfied and so 6^{th} order accuracy can be obtained. Eq. (1.6) has the same stencil as CE4 (four cells each). Actually the present schemes are often referred as compact since for the same stencil they have a higher order or, for the same order, can have a narrower stencil than explicit schemes.

The previous constraints Eq. (1.7)-(1.12) can be written in a matrix form easier to handle. The coefficients can be computed by solving $Ax = b$ where A is:

$$\begin{pmatrix}
1 & 1 & -1 & -1 & -1 & -1 \\
-h_i & h_{i+1} & h_i + \frac{h_{i-1}}{2} & \frac{h_i}{2} & -\frac{h_{i+1}}{2} & -h_{i+1} - \frac{h_{i+2}}{2} \\
h_i^2 & h_{i+1}^2 & \frac{1}{3h_{i-1}}\left(h_i^3 - (h_i + h_{i-1})^3\right) & -\frac{h_i^2}{3} & -\frac{h_{i+1}^2}{3} & \frac{1}{3h_{i+2}}\left(h_{i+1}^3 - (h_{i+1} + h_{i+2})^3\right) \\
-h_i^3 & h_{i+1}^3 & \frac{1}{4h_{i-1}}\left((h_i + h_{i-1})^4 - h_i^4\right) & \frac{h_i^3}{4} & -\frac{h_{i+1}^3}{4} & \frac{1}{4h_{i+2}}\left(h_{i+1}^4 - (h_{i+1} + h_{i+2})^4\right) \\
h_i^4 & h_{i+1}^4 & \frac{1}{5h_{i-1}}\left(h_i^5 - (h_i + h_{i-1})^5\right) & -\frac{h_i^4}{5} & -\frac{h_{i+1}^4}{5} & \frac{1}{5h_{i+2}}\left(h_{i+1}^5 - (h_{i+1} + h_{i+2})^5\right) \\
-h_i^5 & h_{i+1}^5 & \frac{1}{6h_{i-1}}\left((h_i + h_{i-1})^6 - h_i^6\right) & \frac{h_i^5}{6} & -\frac{h_{i+1}^5}{6} & \frac{1}{6h_{i+2}}\left(h_{i+1}^6 - (h_{i+1} + h_{i+2})^6\right)
\end{pmatrix} \tag{1.13}$$

The constraints are written by rows with increasing order of accuracy. The columns correspond to the coefficients. Indeed the unknown vector is $x = (\beta, \gamma, a, b, c, d)^\top$ and the right hand-side is given by $b = (-1, 0, 0, 0, 0, 0)^\top$. This approach is a powerful formalism as it enables us to easily compute the coefficients for all the family.

In the next sections, the different schemes are described. To identify them without ambiguity, the nomenclature developed in [1] is adopted: a scheme is denoted by four letters "IaEb". The I refers to the implicit part (left-hand side of Eq. (1.6)) and E to the explicit part (right-hand side of Eq. (1.6)). a and b can take either the letter C meaning centered or U for upwind. Then each part is followed by one or two numbers giving information on the stencil. They are detailed further on.

1.2 Centered schemes

If all these coefficients are non-zero, a so-called centered scheme is obtained. It is denoted by ICxECy where x and y are the number of points in, respectively, the left-hand side (lhs) and right-hand side (rhs) of Eq. (1.6). In our application, the scheme with highest accuracy is then called IC3EC4 (see appendix A.2.1). A centered scheme with a narrower stencil is IC3EC2 where $a = d = 0$ (see appendix A.2.2). This one is of order 4.

One important remark is that this scheme becomes 'symmetric' centered ($a = d$, $b = c$ and $\gamma = \beta$) when the mesh is uniform. With cells having the same size h, the linear system Eq. (1.13) is independent of h. The coefficients are:

$$\gamma = \beta = \frac{1}{3}, \quad a = d = \frac{1}{36}, \quad b = c = \frac{29}{36} \tag{1.14}$$

The order of a scheme is not the only point. One important characteristic is the resolution of the scheme. It gives information of how well a scheme can represent short length scales. This can be determined by the Fourier analysis of the error like described in [10] and [11]. The following analysis is derived on a uniform mesh. In this case, the scheme is symmetric centered and identical to its finite difference version.

For the purpose of Fourier analysis the variables are assumed to be periodic over the domain $[0, L]$. If there are N points, then $h = L/N$. The variables may be decomposed into their Fourier series:

$$u(x) = \sum_{k=-N/2}^{N/2} \hat{u}_k \exp\left(\frac{2i\pi kx}{L}\right), \tag{1.15}$$

where $i = \sqrt{-1}$. Since u is real-valued, the Fourier coefficients satisfy $\hat{u}_k = \hat{u}^*_{-k}$ where $*$ denotes the complex conjugate.

It is convenient to introduce a scaled wavenumber $\omega = 2\pi kh/L, \omega \in [0, \pi]$ and a scaled coordinate $s = x/h$. the Fourier modes in terms of these are simply $\exp(i\omega s)$. The exact translation n with respect to s generates a function with Fourier coefficients $\hat{u}^n_k = \hat{u}_k \exp(i\omega n)$. The interpolation error may be assessed by comparing the Fourier coefficients of $\tilde{u}$ obtained by the present schemes with exact Fourier coefficients. The transfer function is then:

$$T(\omega) = \frac{ae^{-3i\omega/2} + be^{-i\omega/2} + ce^{i\omega/2} + de^{3i\omega/2}}{\beta e^{-i\omega} + 1 + \gamma e^{i\omega}}. \tag{1.16}$$

The real part of the transfer function is called dispersion error and the imaginary part is called dissipative error.

Exact interpolation corresponds to $T(\omega) = 1$. The range of wavenumbers over which T is close to 1 within a specified error tolerance ε defines the set of well-resolved waves. A scheme that has a larger range is more accurate in spectral space and this definition of accuracy is not related to the formal accuracy defined in terms of truncation error. It should also be noted that the resolution depends only on the scheme and not on the number of points N used in the discretization.

In case of symmetric centered scheme, the transfer function becomes real-valued:

$$T(\omega) = \frac{2b\cos\frac{\omega}{2} + 2a\cos\frac{3\omega}{2}}{1 + 2\alpha\cos\omega}, \tag{1.17}$$

where a, b, α are computed with respect to the finite-difference constraints. So the symmetric centered schemes are not dissipative but only dispersive.

In Fig. 1.2, the transfer function is plotted for several schemes. The set of well-resolved waves for CE2 is only 9 % within an error tolerance of $\varepsilon = 10^{-2}$. The range for CE4 increases to 26 %. With the same order but a narrower stencil, IC3EC2 has 33 % of well-represented waves. So its resolution is better than CE4. This is due the implicitness of the scheme. Finally IC3EC4 has the best resolution with a percentage of 50 %. It has the highest accuracy, both in terms of truncation error and resolution.

If u is periodic then the relation Eq. (1.6) written for each interface can be solved together as a linear system of equations for the unknown values $\tilde{u}$. This linear system is cyclic tridiagonal. The general non-periodic case requires upwind schemes appropriate for the near-boundary interfaces. It is proved in [11] that boundary schemes can have up to two formal order less than the interior scheme without propagating errors. In the next section, 5^{th} order upwind schemes are derived having only one order less than IC3EC4.

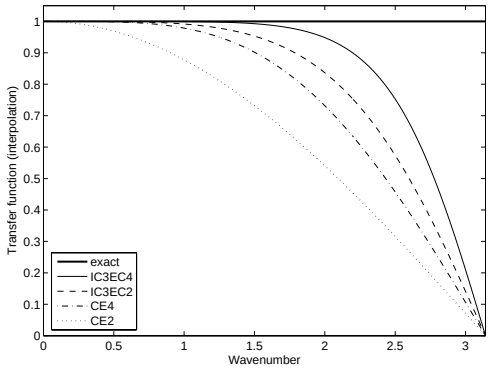


Figure 1.2: Fourier analysis of the error of centered schemes

1.3 Upwind schemes

By setting some of these coefficients to 0, upwind schemes are obtained. In case of ICEU or IUEC, at least one of the coefficients has to be zero and, at most, 5^{th} order accuracy can be achieved. In the case of IUEU, at least 2 coefficients must be 0 (at least one on each side) and therefore, the maximum order of accuracy is 4.

1.3.1 ICEU Schemes

These schemes are referred as ICxEUyz: x is the number of points in the left-hand side, y in the right-hand side and z is the position of $\bar{u}_i$ in the right-hand side. x and y are set to 3 to attain maximum accuracy (5^{th} order). The last constraint Eq. (1.12) is removed, which is equivalent to remove the last row of Eq. (1.13).

Then, the right-hand side is sided. Either a or d can be set to zero. This is equivalent to respectively remove the 3^{rd} or 6^{th} column of Eq. (1.13) and the corresponding terms in x and b. They are called IC3EU31 and IC3EU32.

1.3.2 IUEC Schemes

In [1], this kind of schemes is described as being worthless because of a poor resolution. Nevertheless, they can be useful for the join boundary interfaces as described in chapter 4. Our goal is to compute u at the interfaces, both interior and boundary faces. Say there are n interfaces, there also have n relations like Eq. (1.6). Let's consider that there are 2 ghost cells added at each boundary, the mesh reads:

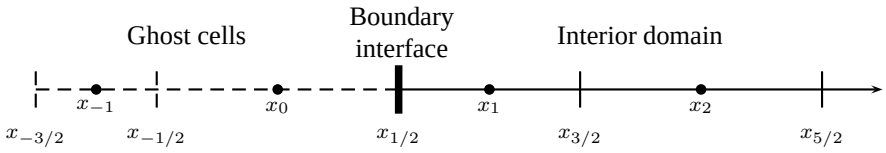


Figure 1.3: Mesh with ghost cells

These ghost cells unable us to use a centered scheme in the explicit part and Eq. (1.6) is written for the two boundaries:

$$\begin{aligned} \beta u_{1/2} + u_{1/2} + \gamma u_{3/2} &= a\bar{u}_{-1} + b\bar{u}_0 + c\bar{u}_1 + d\bar{u}_2 \\ \beta u_{n-3/2} + u_{n-1/2} + \gamma u_{n+1/2} &= a\bar{u}_{n-2} + b\bar{u}_{n-1} + c\bar{u}_n + d\bar{u}_{n+1} \end{aligned} \tag{1.18}$$

The boundary schemes require to know u between the 2 ghost cells ($-1/2$ and $n+1/2$). The n interfaces and the 2 interfaces between the ghost cells at the two ends leads to $n+2$

unknowns $\tilde{u}$, for n relations. The system is under-determined. This can be clearly shown with the following system:

$$\begin{pmatrix} \beta & 1 & \gamma & 0 & \cdots & 0 \\ 0 & \beta & 1 & \gamma & 0 & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots \\ 0 & \cdots & 0 & \beta & 1 & \gamma \end{pmatrix} \begin{pmatrix} \tilde{u}_{-1/2} \\ \tilde{u}_{1/2} \\ \tilde{u}_{3/2} \\ \vdots \\ \tilde{u}_{n-1/2} \\ \tilde{u}_{n+1/2} \end{pmatrix} = \begin{pmatrix} a\bar{u}_{-1} + b\bar{u}_0 + c\bar{u}_1 + d\bar{u}_2 \\ a\bar{u}_0 + b\bar{u}_1 + c\bar{u}_2 + d\bar{u}_3 \\ \vdots \\ a\bar{u}_{n-2} + b\bar{u}_{n-1} + c\bar{u}_n + d\bar{u}_{n+1} \end{pmatrix} \tag{1.19}$$

The matrix of the system is not square: there are 2 more columns than there are rows. Upwind schemes for the implicit part are needed to remove the first and the last column of the matrix.

These schemes are referred as IUxyECz: x is the number of points in the left-hand side, z in the right-hand side and y is the position of $\tilde{u}_{i+1/2}$:

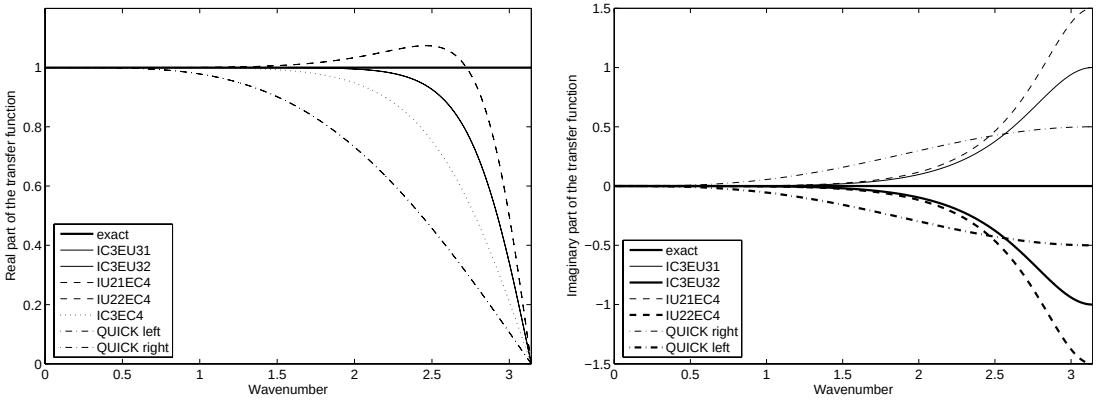
- IU21EC4 scheme: In this scheme, β is set to 0. It is useful for the left boundary. The details are given in appendix A.3.3.
- IU22EC4 scheme: In this scheme, γ is set to 0. It is useful for the right boundary. The details are given in appendix A.3.4.

The present schemes are compared with the QUICK (quadratic upstream interpolation for convective kinetics) scheme developed in [9]. This scheme is often used in CFD and reads:

$$\tilde{u}_{i+1/2} = -\frac{1}{8}u_{i-1} + \frac{6}{8}u_i + \frac{3}{8}u_{i+1}, \tag{1.20}$$

with respect to the flow direction (indices in i could be to the left or the right of the mesh).

The Fourier analysis gives Fig. 1.4. Concerning the dispersive error, even though the ICEU schemes are only 5^{th} order accurate, they have a better resolution than IC3EC4 with 68 % of well-resolved waves. As stated in [1], the IUEC schemes reveal the worst behavior. For some range, the waves are even amplified. The QUICK scheme has a poor representation with only 26 % of well-resolved waves.



(a) Real part of the transfer function

(b) Imaginary part of the transfer function

Figure 1.4: Fourier analysis of the error of upwind schemes

No matter how the schemes are sided (to the left or to the right) they have the same dispersive error, but an opposite dissipation error. This error becomes important for high

wavenumbers with a slightly advantage to the ICEU schemes. The QUICK scheme has an early differentiation with the exact interpolation but for high wavenumbers, the error becomes smaller.

1.3.3 IUEU Schemes

These schemes are referred as IUxyEUzw: x and z are the number of points in, respectively, the lhs and the rhs. y is the position of $\tilde{u}_{i+1/2}$ and w is the position of $\bar{u}_i$. These schemes are not used in our application, so they are not described any further.

1.4 Properties of the schemes

Numerical schemes for CFD have several properties. The order of the schemes has already been studied. It states how the error decreases when the mesh is refined. This is an important property but not the only one to consider. And actually it can be a source of error: on a given mesh a scheme can be less accurate than another one of lower order.

The resolution of the scheme which indicates its spectral behavior has also been studied. This can be developed by Fourier analysis of the scheme.

Other properties are needed for CFD, namely conservativeness, boundedness and transportiveness. They are all related to the inherent physics described by the solved equations.

1.4.1 Conservativeness

This property ensures global conservation of the fluid properties for the entire domain. This is automatically satisfied for finite-volume method.

1.4.2 Boundedness

This property is simple to understand: it states that in the absence of sources the internal values of a property u is bounded by its boundary values. Hence in a steady state conduction problem without sources and with boundary temperatures of 500 °C and 200°C, all interior values of T should be less than 500°C and greater than 200°C.

One necessary condition (but not sufficient) is that a cell-face value should not be larger or smaller than the cell values used to compute it. For example the classical centered 2$^{\text{nd}}$ order scheme (CE2) satisfies this requirement since the cell-face value is computed as the average of the two adjacent cell-values:

$$\tilde{u}_{i+1/2} = \frac{1}{2}(\bar{u}_i + \bar{u}_{i+1}) \tag{1.21}$$

But it can be easily shown that for the convection-diffusion equation, this scheme is unbounded as soon as the Peclet Number (ratio of the convection over the diffusion) is higher than 2. This is due to negative coefficients in the discretized equation where $\tilde{u}$ has been replaced by the rhs of Eq. (1.21). The boundedness of a scheme depends both on the solved equation and of the flow.

Unfortunately, a similar criteria on the present schemes can hardly be found since they are implicit and the coefficients are mesh dependent. We just have to keep in mind that they can be unbounded under certain flow conditions and mesh metrics.

1.4.3 Transportiveness

Unlike diffusion phenomena, convective phenomena influences exclusively in the flow direction so that a point only experiences effects due to changes at upstream location. In other

words, the transportiveness property is a measure of the ability of the scheme to recognize the direction of the flow.

Such a property is built into upwind schemes while the centered schemes are not transportive.

1.5 Solution of the scheme

In order to use the scheme, a tridiagonal system has to be solved. One of the most efficient algorithm is the Tri-Diagonal Matrix Algorithm (TDMA) also known as the Thomas algorithm. It is simply a Gaussian elimination without pivoting that takes into account the tridiagonal structure of the matrix. In appendix B, one can find the basic algorithm optimized for our application.

As there are no pivoting, it would be nice for numerical stability that the matrix of the system be diagonally dominant. This is equivalent to show that the scheme satisfies:

$$|\beta| + |\gamma| \leq 1 \tag{1.22}$$

It is possible to compute analytically $|\beta| + |\gamma|$ for some special meshes:

1. a uniform mesh leads to $\beta = \gamma = \frac{1}{3}$ and the matrix is diagonally dominant.
2. a mesh for a boundary layer like the half of Fig. 1.5: $h_i = \alpha h_{i-1}, h_{i+1} = \alpha h_i$, $h_{i+2} = \alpha h_{i+1}, \alpha > 0$. In Fig. 1.5, $\alpha = 1.1$:

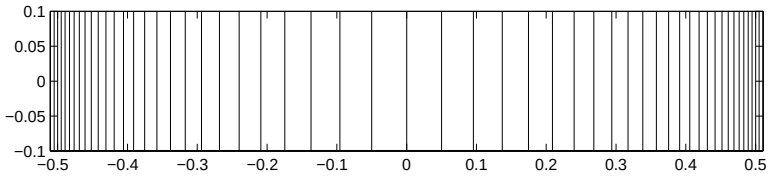


Figure 1.5: Boundary-Layer Mesh

In this case, the diagonal dominance criteria reads:

$$|\beta| + |\gamma| = \frac{1 + \alpha^4}{\alpha + \alpha^2 + \alpha^3}. \tag{1.23}$$

This equation is plot in Fig. 1.7(a). The upper limit of validity is $\alpha = 1.72$. As the maximum stretch α usually met in this kind of mesh is about 1.2, the criteria is satisfied.

3. a mesh with alternating bigger and smaller cells: $h_i = \alpha h_{i-1}, h_{i+1} = h_{i-1}, h_{i+2} = h_i$. In Fig. 1.6, $\alpha = 3$:

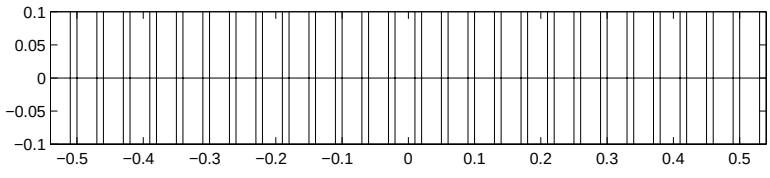
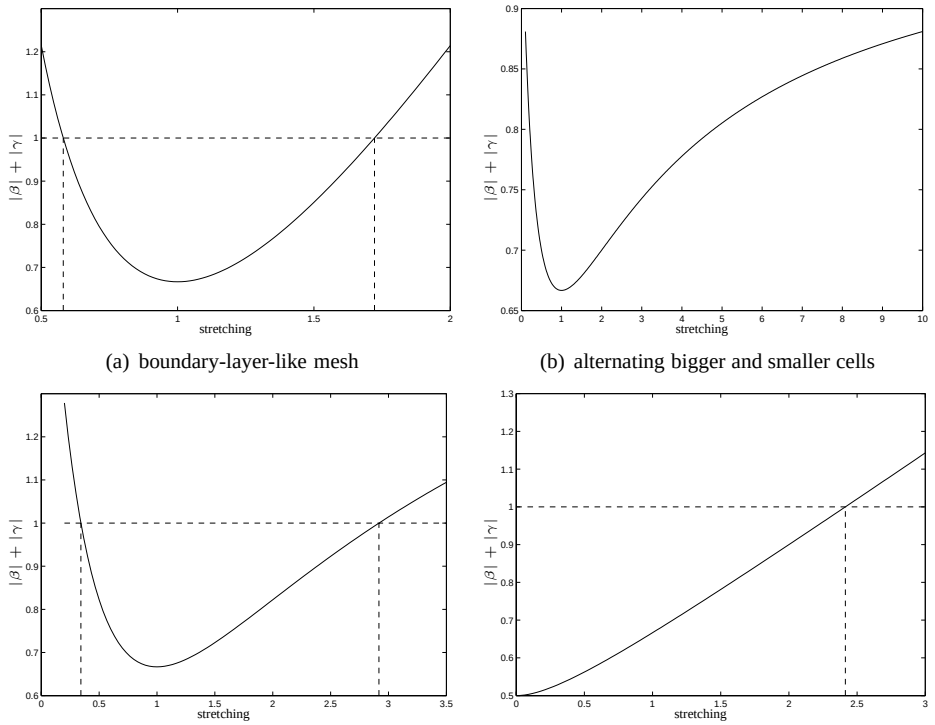


Figure 1.6: Alternating cells Mesh

The relation is now:

$$|\beta| + |\gamma| = \frac{\alpha}{2 + \alpha} + \frac{1}{1 + 2\alpha}. \tag{1.24}$$

Eq. (1.24) is plot in Fig. 1.7(b). Its upper limit is 1 when the stretch α is going toward infinity. So this kind of mesh always leads to a diagonal dominant matrix.



(a) boundary-layer-like mesh

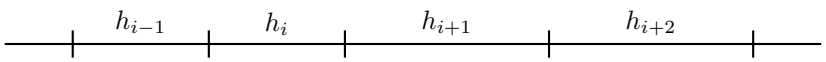
(b) alternating bigger and smaller cells

(c) case 4 mesh

(d) case 5 mesh

Figure 1.7: Validity of the diagonal dominant hypothesis

4. a mesh with parameters: $h_{i-1} = h_i, h_{i+1} = \alpha h_i, h_{i+2} = h_{i+1}$:

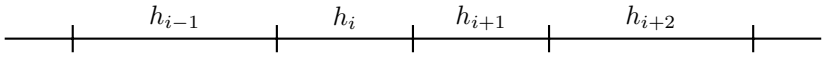


gives the following relation (see Fig. 1.7(c)):

$$|\beta| + |\gamma| = \frac{4(1 + 2\alpha + 2\alpha^3 + \alpha^4)}{(1 + \alpha)^2(2 + 5\alpha + 2\alpha^2)}. \tag{1.25}$$

The maximum stretch is 2.91. Such cases may appear especially on join boundary between mesh blocks. In such cases the stretch can unfortunately be higher, but theses cases are rare.

5. The final mesh considered is of the form: $h_i = \alpha h_{i-1}, h_{i+1} = h_i, h_{i+2} = h_{i-1}$:



In this situation, the equation is:

$$|\beta| + |\gamma| = \frac{(1 + \alpha)^2}{2 + 4\alpha}, \tag{1.26}$$

and the maximum stretch is 2.41 without minimum.

So in practical cases, one can assume that the matrix of the system is diagonally dominant. That allows the use of an algorithm without pivoting.

1.6 Fully upwind schemes

When the cells and interfaces used to compute an interface are only situated on one side of it, the scheme is said to be fully upwind. These schemes are useful for some boundary conditions. As IC3EC4 is of order 6, it is admitted that a 5^{th} order scheme on the boundary is sufficient. This requires 5 degrees of freedom. For example, the fully right upwind scheme reads:

$$\tilde{u}_{i+1/2} + \alpha\tilde{u}_{i+3/2} = a\bar{u}_{i+1} + b\bar{u}_{i+2} + c\bar{u}_{i+3} + d\bar{u}_{i+4}. \tag{1.27}$$

We use the following cells and interfaces:

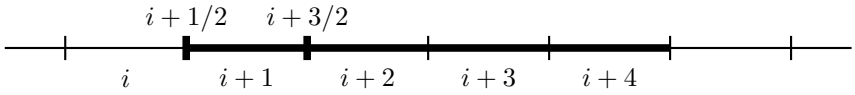


Figure 1.8: Cells and interfaces used in the fully right upwind scheme

and the fully left upwind scheme reads:

$$\alpha\tilde{u}_{i-1/2} + \tilde{u}_{i+1/2} = a\bar{u}_{i-3} + b\bar{u}_{i-2} + c\bar{u}_{i-1} + d\bar{u}_{i}. \tag{1.28}$$

We use the following cells and interfaces:

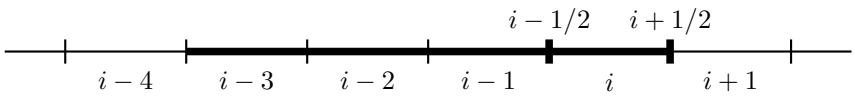


Figure 1.9: Cells and interfaces used in the fully left upwind scheme

The methodology of derivation is similar to the previous schemes.

Chapter 2

Derivation of a procedure for diffusive flux

The 1D diffusion equation reads:

$$\rho c_P \frac{dT}{dt} = \frac{d}{dx}\left(k\frac{dT}{dx}\right). \tag{2.1}$$

In the right hand-side of Eq. (2.1) appears the second-order derivative of T. When it is integrated over a cell, the first derivative of T at the interfaces appears:

$$\rho c_P \frac{d\overline{T}}{dt} h_i = \left(k\frac{dT}{dx}\right)_{i+1/2} - \left(k\frac{dT}{dx}\right)_{i-1/2}, \quad \forall i. \tag{2.2}$$

The present schemes are not able to directly compute the derivatives at cell faces. They just take the cell average value $\bar{u}$ of a scalar function u and interpolate it to get $\tilde{u}$ the approximation of u at the cell interfaces. In order to get the first derivative of u at the interfaces with the present schemes, one would like to know the cell average value of the derivative of u. Let's compute it:

$$\overline{\left(\frac{du}{dx}\right)}_i = \frac{1}{h_i}\int_{x_{i-1/2}}^{x_{i+1/2}} \frac{du}{dx} dx = \frac{1}{h_i}(u_{i+1/2} - u_{i-1/2}), \quad \forall i. \tag{2.3}$$

$\overline{u_x}$ can be computed with the cell interface value of u. So the following 3-steps procedure is proposed:

Step 1: compute the cell interface value $\tilde{u}$ of u by applying the present implicit schemes;

Step 2: compute the cell-average value $\overline{u_x}$ of the derivative u_x using Eq. (2.3) and the approximated cell-face values $\tilde{u}$;

Step 3: compute the cell interface value $\widetilde{u_x}$ of u_x using the same schemes on $\overline{u_x}$.

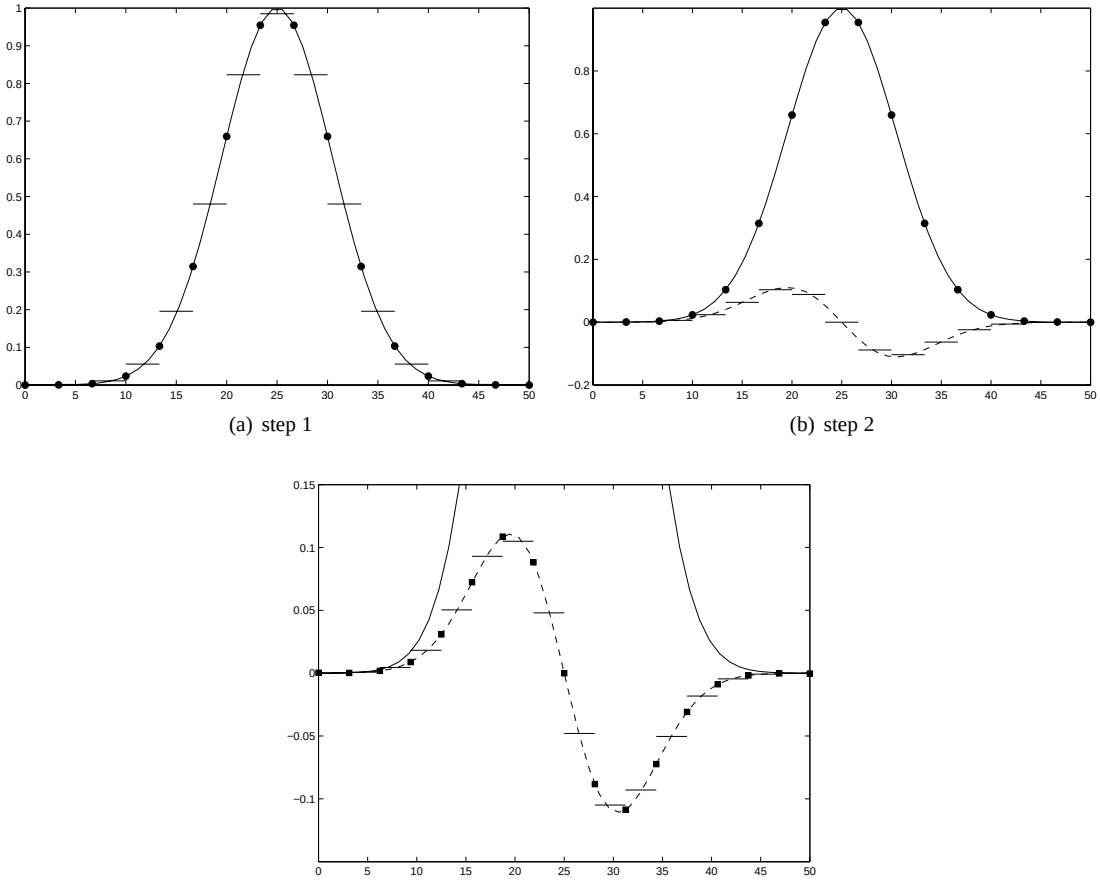
Two boundary conditions are usually associated to this equation: adiabatic condition where $\partial T/\partial x = 0$ is prescribed on the boundaries and isotherm boundaries where T is hold fixed. In both cases, the boundary interface that is not imposed have to be computed (T in case of adiabatic condition or $\partial T/\partial x$ for isotherm condition). The computation of these boundary interfaces is then handled by fully upwind schemes.

This procedure is illustrated in Fig. 2.1 on the function given by Eq. (2.4) used in the test section on the domain $[0, 50]$:

$$u(x) = \exp\left(-\frac{1}{60}(x-25)^2\right). \tag{2.4}$$

As $u(x)$ admits no integral, $\bar{u}$ is numerically computed using the function `quadl` of Matlab (which implements a recursive adaptive Lobatto quadrature) within an error of 10^{-6}.

In subfigure (a), the approximation $\tilde{u}$ in circles is computed from the known values $\bar{u}$ in horizontal solid lines. The circles match $u(x)$. Step 2 is illustrated in subfigure (b) where $\overline{u_x}$ is plotted over the exact value u_x in dashed line. Step 3 is finally done by applying the same scheme again. As shown in (c), $\widetilde{u_x}$ (in squares) match pretty well u_x at the interfaces.



(a) step 1

(b) step 2

(c) step 3

Figure 2.1: Illustration of the procedure for diffusive flux

The scheme is actually applied twice. This can increase the error and it is then hard to tell the order of the procedure for diffusive term. The empirical order of the procedure will be computed in the test section.

Chapter 3

Numerical Tests

Basically the present schemes interpolate a function on cell-faces knowing its cell-average value. First, a simple interpolation problem is performed. The empirical order will be computed and compared with standard schemes. It will be interesting to observe their behavior on non-uniform meshes.

Once this basic tests are done, some usual partial differential equations such as convection problem in 1D or 2D are solved. The performances will be measured by comparison with other schemes (CE2 and CE4). Finally the procedure for diffusive flux is used to solve the heat equation.

3.1 1D interpolation

Let's consider a mesh with $n-1$ interior cells numbered from 1 to $n-1$. So there are n interfaces ($n-2$ interior and 2 boundaries). The mesh is surrounded by two "ghost" cells so that centered scheme in the explicit part can always be used (an intensive use of these ghost cells is described in the next sections). For the interior interfaces numbered from 2 to $n-1$, the centered scheme with highest accuracy IC3EC4 is used . For the boundaries, upwind schemes in the implicit part are used, namely IU21EC4 for the left boundary numbered 1 and IU22EC4 for the right boundary n. The matrix form of the schemes reads:

$$\begin{pmatrix} 1 & \gamma_1 & 0 & \cdots & 0 \\ \beta_2 & 1 & \gamma_2 & 0 & 0 \\ 0 & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \beta_{n-1} & 1 & \gamma_{n-1} \\ 0 & \dots & 0 & \beta_n & 1 \end{pmatrix} \begin{pmatrix} \tilde{u}_1 \\ \tilde{u}_2 \\ \vdots \\ \tilde{u}_{n-1} \\ \tilde{u}_n \end{pmatrix}$$
$$= \begin{pmatrix} a_1 & b_1 & c_1 & d_1 & 0 & \cdots & \cdots & 0 \\ 0 & a_2 & b_2 & c_2 & d_2 & 0 & \cdots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & a_{n-1} & b_{n-1} & c_{n-1} & d_{n-1} & 0 \\ 0 & \cdots & \cdots & 0 & a_n & b_n & c_n & d_n \end{pmatrix} \begin{pmatrix} \bar{u}_{-1} \\ \bar{u}_0 \\ \vdots \\ \bar{u}_n \\ \bar{u}_{n+1} \end{pmatrix} \quad (3.1)$$

where $\tilde{u}$ is the interpolated value of u at the interfaces. The coefficients

- $(\gamma_1, a_1, b_1, c_1, d_1)^\top$ are computed following Eq.(A.35)-(A.39),
- $(\beta_n, a_n, b_n, c_n, d_n)^\top$ are computed following Eq.(A.41)-(A.45).

- $(\beta_i, \gamma_i, a_i, b_i, c_i, d_i)^\top, 2 \leq i \leq n-1$ are computed following Eq.(A.11)-(A.16).

This system $A\tilde{u} = B\bar{u}$ is consistent:

- A is a matrix $n \times n$;
- there are n unknowns $\tilde{u}$;
- B is a matrix $n \times n+3$ and $n+3$ cells are used ($n-1$ interior cells and 4 ghost cells) so that $B\bar{u}$ is a vector $n \times 1$ which is consistent with the product $A\tilde{u}$

It is solved using the TDMA e.g. the Thomas algorithm. See appendix B for further details.

The schemes are now tested on a function with a known integral. A random 1D mesh is generated and the cell-average values of the function are computed. The approximated values $\tilde{u}$ at the interfaces are then compared to the true one. In Fig. 3.1 the function $f(x) = \sin(7x) + 3\cos(5x) + 10$ is in solid line and the vertical dashed lines show the random mesh. The horizontal solid lines are the cell-average value of f. The interpolated values of f are plotted in circle and they match the function at the interfaces despite a very distort mesh (alternation of very small and very big cells).

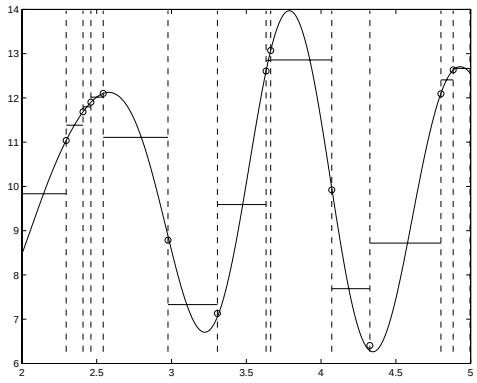
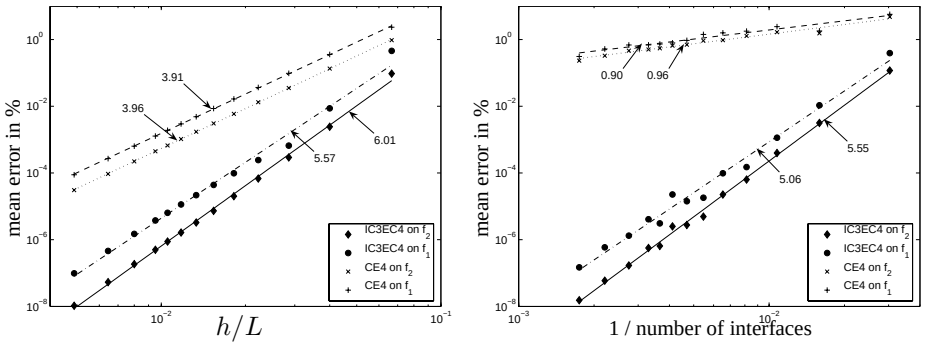


Figure 3.1: Example of the interpolation scheme

Now the empirical error is computed to check whether the scheme is 6^{th} order accurate. Therefore a uniform mesh is considered and the error is computed with respect to different mesh size in Fig. 3.2(a). Two functions are used: f_1 is the previous function f and $f_2(x) = e^{x+2}\sin(7x) + 1000$. The slope of the error is computed using least squares



(a) uniform mesh (b) non-uniform mesh

Figure 3.2: Slope of the error

method. The first function has an order of 6.01 and 5.57 for the second. This is rather close to the theoretical order of 6. The error is smaller than for the classical 4^{th} order centered scheme denoted CE4.

Furthermore, this scheme remains 6^{th} order on non-uniform meshes while CE4 becomes 1^{st} order accurate as displayed in Fig. 3.2(b). To obtain this figure, a uniform mesh is generated and then each interface is moved by a random slide. Then the error is plotted versus the number of interfaces. The scheme IC3EC4 has an order of 5.56 and 5.06 which is still high while the standard 4^{th} order central scheme becomes 1^{st} order accurate as expected. Of course, this test is the worst case: in real-world problems, the meshes are not so misshapen. But at least it shows that the new derived schemes are robust.

Now that the formal behavior of the scheme is checked, it is used to solve partial differential equations.

3.2 1D advection equation

Recall the advection equation:

$$\frac{\partial f(x,t)}{\partial t} + v\frac{\partial f(x,t)}{\partial x} = 0, \quad f(x,t=0) = f_0(x), \tag{3.2}$$

and its semi-discrete finite-volume formulation from Eq.(1.4):

$$\frac{\partial \bar{f}_i}{\partial t} = -\frac{v}{h_i}(f_{i+1/2} - f_{i-1/2}), \quad \forall i. \tag{3.3}$$

An ordinary differential equation in time is finally found. The time integration is achieved using a modified Runge-Kutta of order 4 (RK4, see appendix C for more details). For this equation, the Courant-Friedrichs-Lewy Condition (CFL condition) is equal to $v\Delta t/\Delta x$ and for RK4, CFL $< \sqrt{2}$ must be fulfilled.

This equation can be used to describe the propagation of a Gaussian wave. In this case, f_0 is:

$$f_0(x) = \frac{1}{\sqrt{2\pi}} e^{-\frac{x^2}{2}} \tag{3.4}$$

The present schemes Eq.(1.6) are designed to work on non-uniform meshes. Nevertheless, as the wave has to go all over the domain, a uniform mesh seems relevant. In consequence this scheme will become 'symmetric' centered: the coefficients are given by Eq.(1.14).

Periodic boundary conditions are imposed. It means that when the wave disappears on one side of the domain, it reappears on the other side.

In order to use the present schemes and to implement the boundary conditions, ghost cells are used: on each side of the domain, 2 ghost cells are added. Say that the interior cells are numbered from 1 to $n-1$. Then the left ghost cells are numbered -1 and 0, and the right ghost cells are numbered n and $n+1$. To implement the periodic boundary conditions, the following relation are set at each iteration:

$$\begin{cases} \bar{f}_n = \bar{f}_1, & \bar{f}_{n+1} = \bar{f}_2, \\ \bar{f}_{-1} = \bar{f}_{n-2}, & \bar{f}_0 = \bar{f}_{n-1}. \end{cases} \tag{3.5}$$

As in the previous section, the ghost cells enable us to use centered schemes in the explicit part to compute the boundary interfaces.

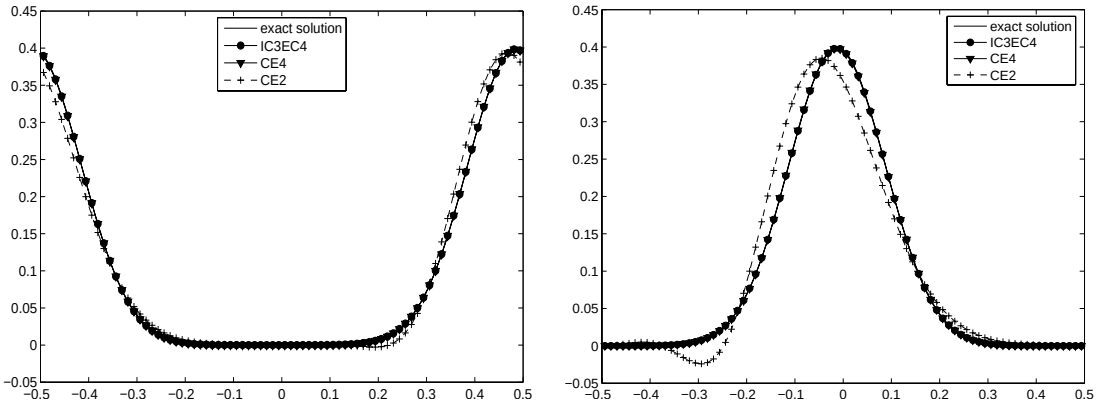
Finally, the advection equation is solved by algorithm 3.1 on the following page. This algorithm is now applied on several meshes to compare the behavior of the present schemes and the standard schemes. The Gaussian wave has a unit speed and is advected on the domain $[-1/2, 1/2]$.

Algorithm 3.1 Algorithm for solving the advection equation

Require: a uniform mesh of $n-1$ interior cells, v, $f_0(x)$, t_{max}, a CFL
  choose a time step Δt in accordance with the CFL
  compute $\bar{f}_0$ on each cell, then set $\bar{f}^0 = \bar{f}_0$
  compute the coefficients $(\beta, \gamma, a, b, c, d)$ with respect to the mesh and the scheme for each interface
  for each cell i **do** {time update}
    compute $\bar{f}^t$ using RK4 (algorithm C.1) and the update of the ghost cells
  end for

3.2.1 Uniform mesh

An example of run is exposed in Fig. 3.3. The exact solution is in solid lines. Three schemes are compared: the present scheme IC3EC4 in circles, the standard 2^{nd} order scheme (CE2) in plus signs and the standard 4^{th} order scheme (CE4) in triangles.



(a) $t = 2.5s$ (b) $t = 5s$

Figure 3.3: Propagation of a 1D Gaussian wave, $v = 1$, CFL=1.4

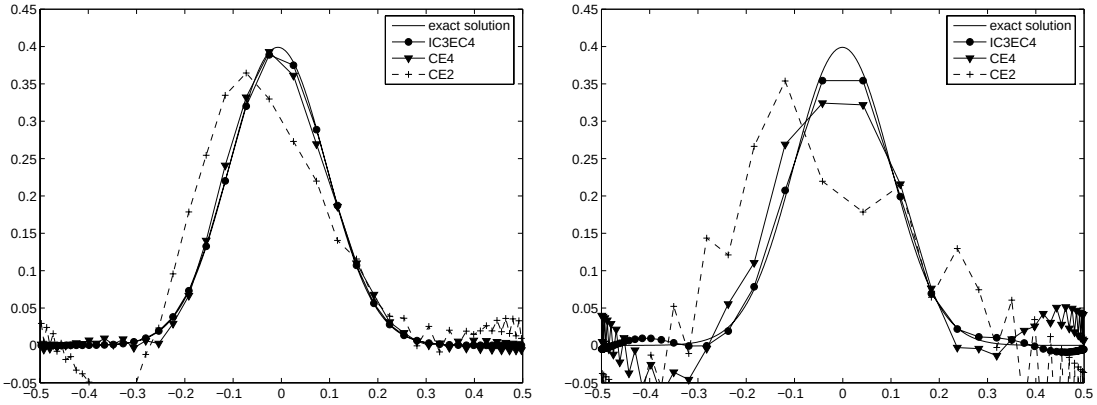
Even though the CFL is close to its upper limit, the first scheme matches the exact solution while CE2 shows some errors that grow with respect to time. CE4 is almost as good as the present scheme. This is because it has a higher accuracy than CE2 and because the mesh is uniform. In the next section, the schemes are tested on non-uniform meshes. The behavior of the present scheme should be better than the classical schemes.

3.2.2 Non-uniform meshes

For instance, consider a first mesh that alternates bigger and bigger cells with smaller and smaller cells. This kind of mesh is usually used for boundary layer flow where the need of high accuracy near the boundaries requires a finer mesh. It is sketched in Fig. 1.5 that satisfies the criteria for diagonally dominance of the scheme.

In case of a non-uniform mesh, the CFL is given by $v\Delta t / \min(\Delta x)$. The Gaussian wave is advected during a natural number of seconds so that its final state should be equal to its start. In Fig. 3.4, the IC3EC4 scheme clearly shows better results than CE2 and CE4. This behavior was expected since the latter schemes are not design to be used on non-uniform meshes.

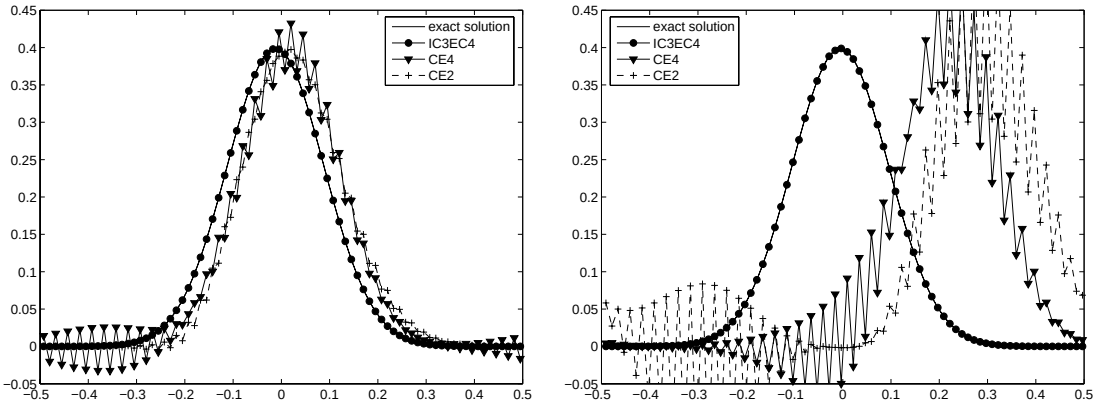
The previous mesh is not adapted to the problem of the advection of a Gaussian wave because there are only a few points in the middle of the domain. In order to better see the influence of the non-uniformity, the schemes are tested on a mesh that alternates bigger



(a) Aspect ratio=1.1 (b) Aspect ratio=1.2

Figure 3.4: Advection of a Gaussian wave on boundary layer mesh, $t = 2$ s

and smaller cells as in Fig. 1.6 where the aspect ratio is 3. There is a better distribution of the points in all the domain. The results appears in Fig. 3.5. Again the present scheme perfectly matches the true solution. The other schemes show important oscillations. For a small stretch, an error in the location of the peak is observed while for a higher stretch, the result is really far from the true solution.



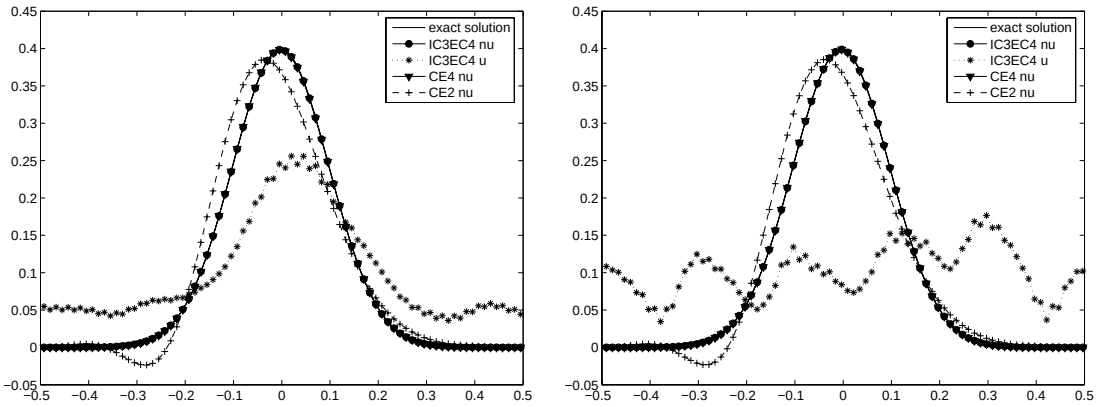
(a) Aspect ratio=1.5 (b) Aspect ratio=3

Figure 3.5: Advection of a Gaussian wave on alternating size mesh, $t = 2$ s

It may seem unfair to compare the present scheme to other schemes designed to be used on uniform mesh, IC3EC4 is now compared to the non-uniform version of CE2 and CE4. CE2$_{nu}$ is given by:

$$\tilde{u}_{i+1/2} = \frac{h_{i+1}}{h_i + h_{i+1}} \bar{u}_i + \frac{h_i}{h_i + h_{i+1}} \bar{u}_{i+1} \tag{3.6}$$

CE4$_{nu}$ can be easily derived as IC1CE4 by the procedure of appendix A. It is also interesting to observe the behavior of a uniform version of IC3EC4 and so use the coefficients Eq. (1.14). After 5 s, the results are plotted on Fig. 3.6. It is not surprising that the 2$^{\text{nd}}$ order scheme, even in its non-uniform version, gives poor results. CE4$_{nu}$ gives results almost as good as IC3EC4: even when the aspect ratio is high, its behavior is satisfying.



(a) aspect ratio $= 11/9$ (b) aspect ratio $= 3$

Figure 3.6: Advection on the alternating size mesh

What is more peculiar is that IC3EC4$_u$ gives really poor results. When the aspect ratio is 3, the shape of the wave is lost and even CE2$_{nu}$ is a lot better. It may be explained by the fact that the interfaces are coupled three by three. So if an error is committed, which is really likely to happen when IC3EC4$_u$ is applied on a non-uniform mesh, it is propagated to its two neighbors which in turn will propagate it to their neighbors and so on. The use of an implicit scheme can be dangerous since the error can propagate a lot faster.

3.3 2D advection equation

The 2D advection equation is given by:

$$\frac{\partial f(x,y,t)}{\partial t} + v_x \frac{\partial f(x,y,t)}{\partial x} + v_y \frac{\partial f(x,y,t)}{\partial y} = 0 \tag{3.7}$$

The semi-discrete formulation reads:

$$\frac{\partial \bar{f}_{i,j}}{\partial t} = -\frac{1}{A_{i,j}} \left[v_x \Delta y_{i,j} (f_{i+1/2,j} - f_{i-1/2,j}) + v_y \Delta x_{i,j} (f_{i,j+1/2} - f_{i,j-1/2}) \right] \tag{3.8}$$

where $\Delta x_{i,j}$ is the width of the cell (i,j) and $\Delta y_{i,j}$ is its height. $A_{i,j}$ is the area of the

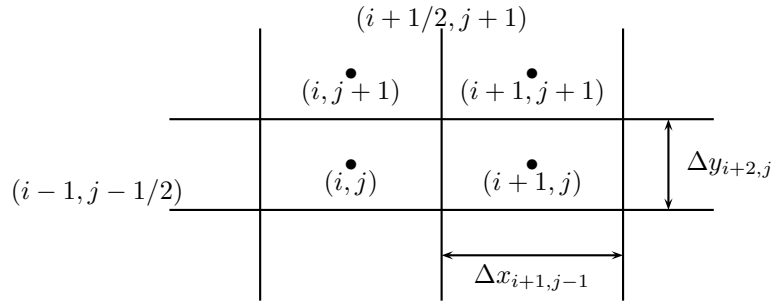


Figure 3.7: 2D structured (cartesian) grid

cell ($A_{i,j} = \Delta x_{i,j} \Delta y_{i,j}$ in case of cartesian mesh). $\bar{f}$ is the cell average value of f:

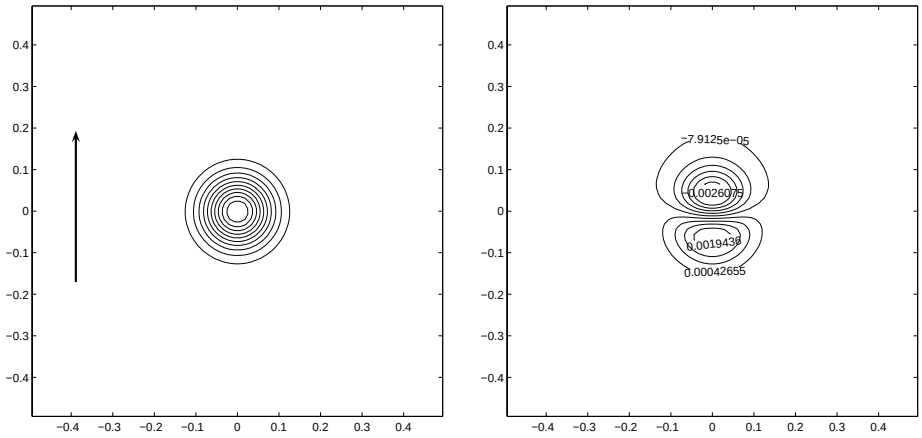
$$\bar{f}_{i,j} = \frac{1}{A_{i,j}} \int_{y_{i,j-1/2}}^{y_{i,j+1/2}} \int_{x_{i-1/2,j}}^{x_{i+1/2,j}} f(x,y)\,\mathrm{d}x\,\mathrm{d}y \tag{3.9}$$

3.3.1 Uniform mesh

A 2D Gaussian wave of the following shape:

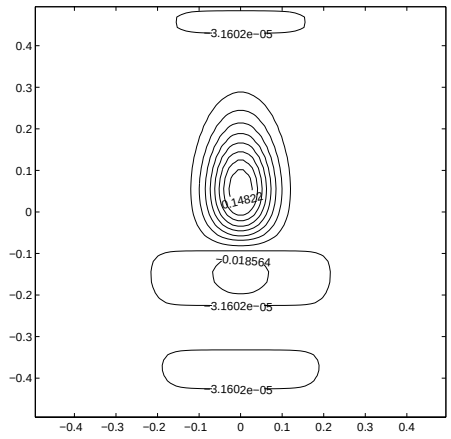
$$f(x, y, 0) = \frac{1}{4} \exp\left[-154.037(x^2 + y^2)\right] \tag{3.10}$$

is propagated in a domain $-0.5 < x < 0.5$, $-0.5 < y < 0.5$. In a first attempt, the speeds $v_x = 0$ and $v_y = 1$ are used. After 5 s, the results can be found in Fig. 3.8. To obtain these figures, a 80×80 cartesian uniform grid is used. Time integration is performed through a six-stage RK scheme with a very low time step to minimize the error due to time discretization. The computed solution can easily be compared to the exact solution to get the error. CE2 shows important errors both in terms of the location of the peak (which should be in the middle of the domain) and symmetry (the computed solution is stretched is the direction of the movement of the wave). For the classical 4^{th} order scheme (not shown here), the peak is well located but shows a little dissymmetry. The solution given by the IC3EC4 is in excellent agreement with the exact solution (circular isolines).

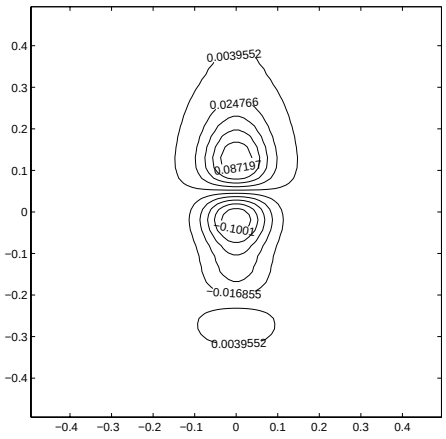


(a) IC3EC4 – Computed solution

(b) IC3EC4 – error



(c) CE2 – Computed solution



(d) CE2 – error

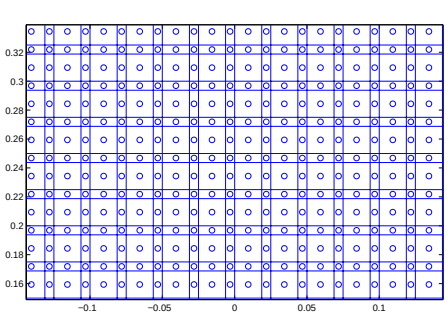
Figure 3.8: Advection of a 2D Gaussian wave

The amplitude of the peak is not well conserved by the explicit schemes. While the true value is 0.25, the CE2's peak is 0.1668, and CE4 gives 0.1695. The implicit IC3EC4 scheme gives a peak of 0.2465 which is rather close to the true solution.

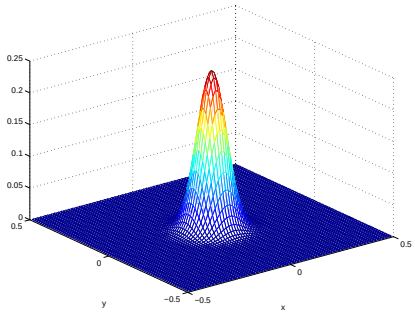
3.3.2 Non-uniform meshes

Cartesian Grid

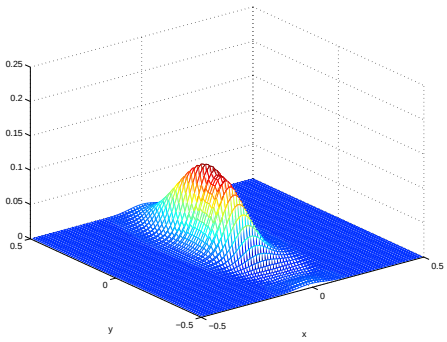
The schemes are now tested on a very stretched grid. Each direction is made of alternating bigger and smaller cells by an aspect ratio of 3. A part of the mesh can be observed in Fig. 3.9(a). The wave is advected during 10 seconds with a unit speed in the y-direction. The present schemes show a good solution in 3.9(b) with a maximum error of 1.5 %. The shape of the wave and its peak are not conserved by the two other schemes. They also show a stretching in the direction of the movement. Curiously, the worst scheme is CE4. Not only it gives a bad representation of the solution at $t = 10$ s, but it has advected during 14 periods instead of 10. The convection speed is higher. It is just due to the larger stencil of CE4. It expects to have 4 cells of equal size and instead it gets 4 stretched cells which is probably worse than for CE2 which gets two stretched cells. Nevertheless, the latter runs for 12 periods.



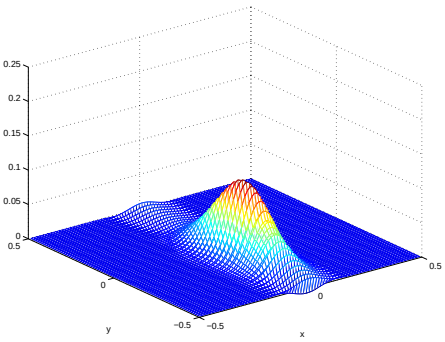
(a) Part of the stretched mesh



(b) IC3EC4



(c) CE2



(d) CE4

Figure 3.9: Advection on a 2D non-uniform cartesian mesh

The same kind of results as with the uniform mesh are retrieved but the error is intensified for the standard schemes. That is the reason why for this example, 3D plots are shown instead of contour plot. The 3D plots emphasize the fact that the peak is not conserved.

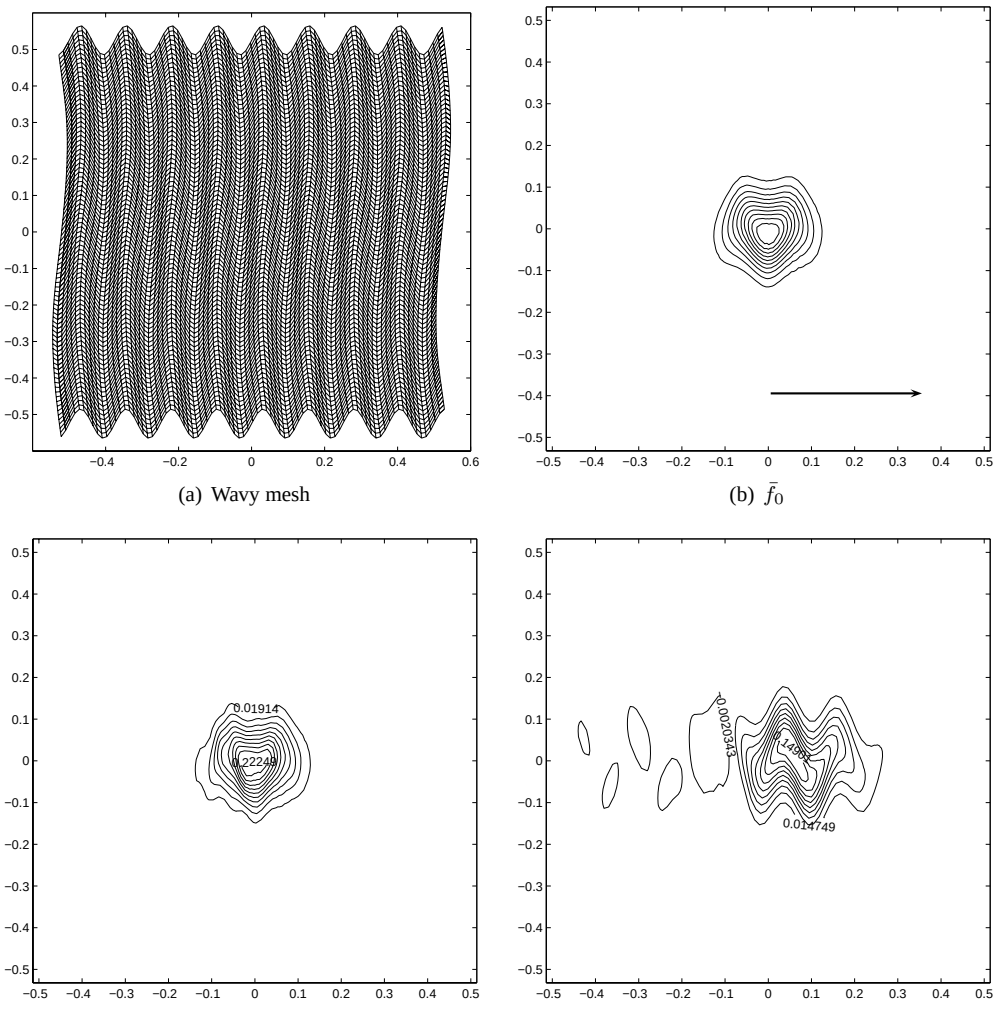
In case of classical schemes, numerical waves appears as described in [11]. While the physical wave is advected, spurious waves are generated by the spatial discretization. These waves have high frequencies, a smaller amplitude than the physical wave, the same speed but an opposite direction. These wave packets, also called wiggles, are a phenomenon to avoid because their interaction with the physical wave may be unpredictable. They can appear for any non-dissipative scheme which is the case of all centered schemes that have symmetric coefficients. IC3EC4 is symmetric centered only on uniform meshes (see section 1.2). So the present schemes are robust against such spurious waves.

Non-Cartesian Grid The schemes are now tested on a very irregular grid. To obtain Fig. 3.10(a), each coordinate $(\bar{x}_{i,j}, \bar{y}_{i,j})$ of the uniform mesh has been moved according to:

$$\begin{cases} x_{i,j} = \bar{x}_{i,j} + 0.02\sin(2\pi\bar{y}_{i,j}) \\ y_{i,j} = \bar{y}_{i,j} + 0.04\sin(16\pi\bar{x}_{i,j}) \end{cases} \tag{3.11}$$

This mesh is not cartesian and the present schemes are not designed to work on such a grid. Anyway, it is interesting to observe their behavior.

This time the convection speeds are $v_x = 1$ and $v_y = 0$. As the wave has a small radius, the isolines of the initial condition follow the mesh lines on Fig. 3.10(b) instead of concentric circles. But it doesn't matter since the result after advection can be compared to $\bar{f}_0$. In Fig. 3.10(c-d), the results of two schemes are compared : the implicit scheme gives pretty good results and preserve the peak amplitude. Only some isolines are a little bit distorted compared to $\bar{f}_0$. On the other hand, the solution given by CE2 is really influenced by the mesh: the isolines are also wavy and the solution is stretched in the direction of the movement.



(a) Wavy mesh (b) $\bar{f}_0$

(c) IC3EC4 after 2 s (d) CE2 after 2 s

Figure 3.10: 2D Advection on a wavy mesh

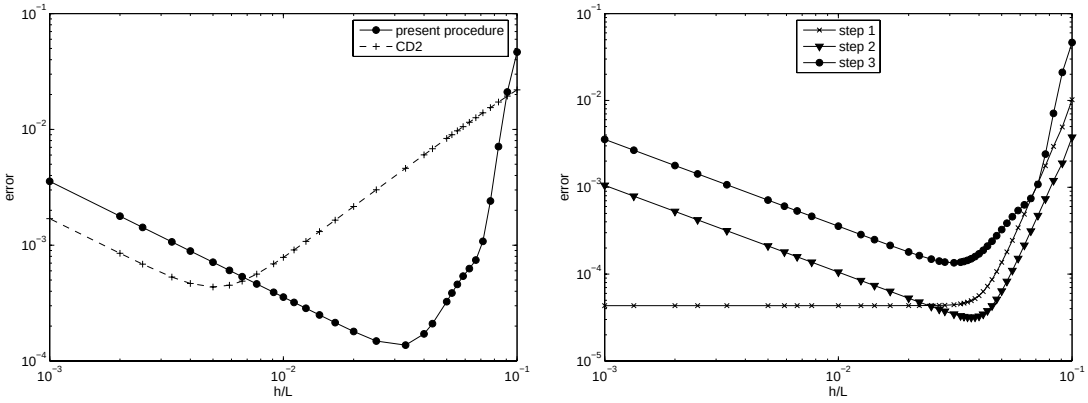
3.4 Test of the procedure for diffusive flux

3.4.1 Empirical order

Let's consider the function f of Eq.(2.4) and compute the empirical error knowing its true derivative. The present procedure is compared to the classical central 2 [nd] order difference scheme (CD2) which reads:

$$\overline{u'}_{i+1/2} = \frac{\bar{u}_{i+1} - \bar{u}_i}{x_{i+1} - x_i} \tag{3.12}$$

The results are given in Fig. 3.11(a). For the highest h, the present procedure seems to have a higher order since the slope of the error is steeper. Surprisingly, the error starts to increase for smaller h. CD2 has the same behavior but for even smaller cell size.



(a) Comparison with classical difference scheme

(b) Error in the 3 steps

Figure 3.11: Empirical error of diffusive flux procedure

In order to investigate the reason of this increasing error for small h, the error for each step is plotted on sub-figure (b). The first one (computation of $\tilde{u}$) reaches a lower bound for small value of h which implies a higher error on $\overline{u_x}$ and in the same order on $\widetilde{u_x}$. This can be explained easily since step 2 is computed as:

$$\overline{u_x} = \frac{\tilde{u}_{i+1/2} - \tilde{u}_{i-1/2}}{h_i}. \tag{3.13}$$

$\tilde{u}$ being computed with a bounded accuracy, the numerator of Eq. (3.13) is also lower bounded while the denominator h continue to decrease leading to a higher error proportional to $1/h$. Actually, the slope in this region is equal to 1. The same thing happens to CD2, as $\bar{u}$ is bounded, the numerator of Eq.(3.12) is lower bounded and its denominator decreases.

This bound is actually due to error tolerance on Matlab function `quadl` with which is computed $\bar{u}$. This tolerance is set to 10^{-6} by default. As the error on $\bar{u}$ can not go under this tolerance, $\tilde{u}$ is approximated with a lower bound and has consequences on the remaining steps.

Considering the highest h, the first two steps are perfectly parallel and have an order very close to 6. The third step does not present a straight line but its order is also very high, between 5 and 6 depending on the lower bound threshold used on h/L.

3.4.2 1D diffusion equation

The diffusion equation Eq.(2.1) is also called the heat equation. In this case, ρ is the density, c_P is the heat capacity and k is the thermal conductivity.

Problem

A laterally insulated bar of length L initially has some temperature distribution given by $f(x)$. The ends at $x = 0$ and $x = L$ are held fixed at $0°\text{C}$ for all time. The temperature profile versus time is now computed, $T(x,t)$, given that

$$T(x, t = 0) = f(x) = \exp\left(-\frac{1}{60}(x - 25)^2\right), \tag{3.14}$$

with $\frac{k}{\rho c_P} = 2$ cm^2.s^{-1} and $L = 50$ cm.

In order to check the computed solution, the true solution is given by mean of Fourier series:

$$T(x,t) = \sum_{n=1}^{\infty} D_n \sin\left(\frac{n\pi x}{L}\right) \exp\left(-\frac{n^2\pi^2 t}{L^2}\right), \tag{3.15}$$

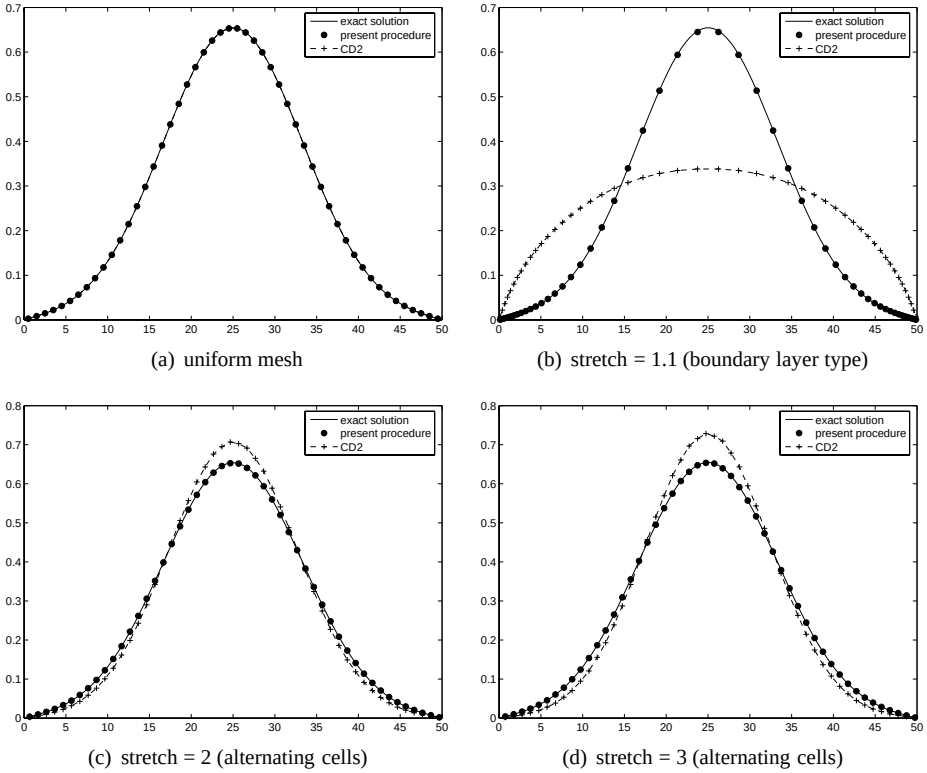
where:

$$D_n = \frac{2}{L}\int_0^L f(x) \sin\left(\frac{n\pi x}{L}\right) \, \mathrm{d}x. \tag{3.16}$$

Numerical Solution

For the first step, as the value of the temperature is known at the boundaries, only the interior interfaces have to be computed with upwind schemes for the first interior faces. Namely IC3EU31 for the left side and IC3EU32 for the right side. To compute the derivatives (3$^{\text{rd}}$ step), fully upwind schemes are used at the boundaries.

The present procedure is compared to CD2 on Fig 3.12. They both gives good results when the mesh is uniform but as soon as the mesh is stretched (as for the 1D advection) CD2 gives really bad results.



(a) uniform mesh (b) stretch = 1.1 (boundary layer type)

(c) stretch = 2 (alternating cells) (d) stretch = 3 (alternating cells)

Figure 3.12: Numerical solution of heat equation, after 20 s

Chapter 4

Multi-Domain Treatment

For complex geometries, a mesh can be be divided in several blocks. As *elsA* is a parallel software, the can be allocated to several processors. And as they can be of very different sizes, each processor solves a subset of blocks. If a processor handles several blocks, they are solved sequentially in one way and communicate by ghost cells that overlap the cells of the neighbors blocks. A boundary between two blocks is sketched in Fig. 4.1. To give the illusion of a unique mesh, each block is surrounded by two ghost cells (in dashed lines). And after each iteration on the blocks, they exchange the cell-average values of their end cells (dotted lines).

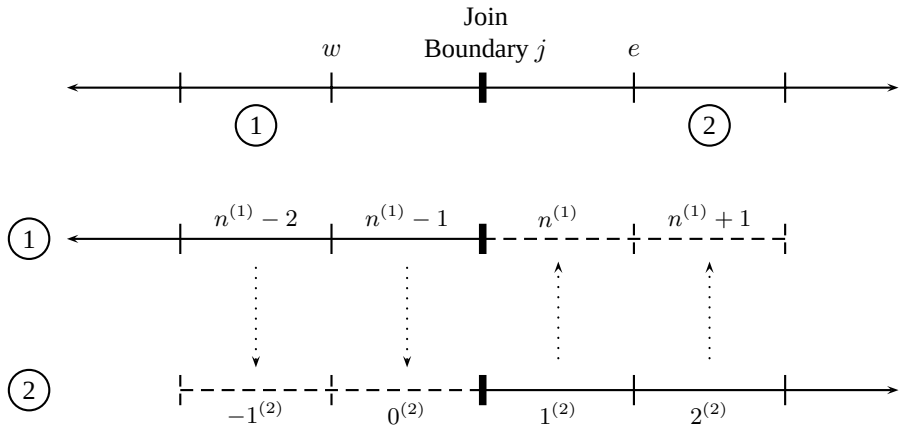


Figure 4.1: A Join Boundary

This strategy works well when the spatial discretization scheme is explicit but is inadequate for implicit schemes. The problem arises for the join boundary. As it is part of the interior domain of the global mesh, it should be computed with the present implicit centered scheme IC3EC4:

1. for block 1, IC3EC4 reads:

$$\beta w + j + \gamma e = a\bar{u}_{n^{(1)}-2} + b\bar{u}_{n^{(1)}-1} + c\bar{u}_{n^{(1)}} + d\bar{u}_{n^{(1)}+1} \tag{4.1}$$

2. for block 2, IC3EC4 reads:

$$\beta w + j + \gamma e = a\bar{u}_{-1^{(2)}} + b\bar{u}_{0^{(2)}} + c\bar{u}_{1^{(2)}} + d\bar{u}_{2^{(2)}} \tag{4.2}$$

Interfaces w and e also depend on the join boundary value to be computed. A matrix formulation of the problem is sketched in Fig. 4.2 for a 3-blocks mesh:

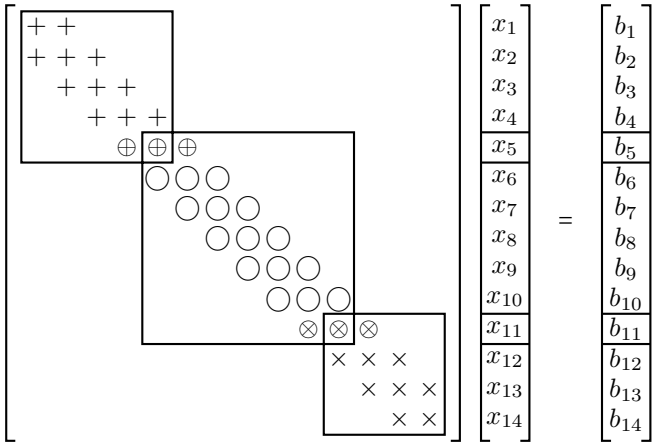


Figure 4.2: Matrix formulation of the multi-domain problem

where $+$ is only known by block 1, $\oplus$ corresponds to the join boundary between blocks 1 and 2, $\bigcirc$ is only known by block 2, $\otimes$ corresponds to the join boundary between blocks 2 and 3, $\times$ is only known by block 3. The join boundaries x_5 and x_{11} are computed in both blocks and the previous system can be transformed in the following equivalent pentadiagonal system:

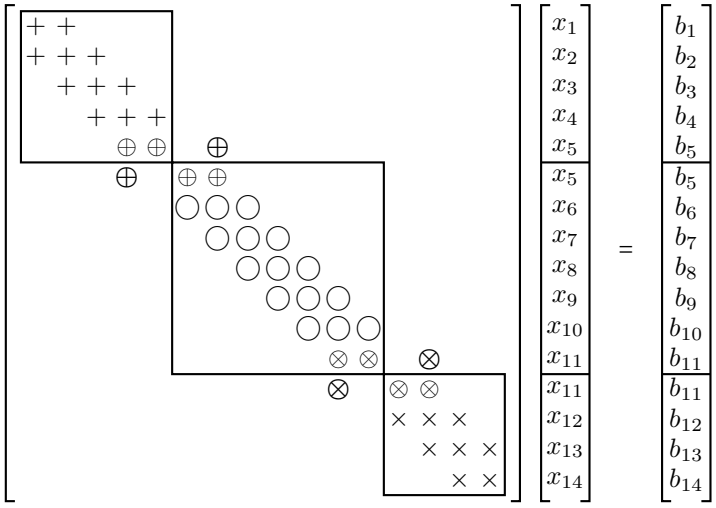


Figure 4.3: Pentadiagonal formulation of the multi-domain problem

The coupling between the blocks is done by the off diagonal elements in bold symbols. There exists tens of algorithms that solve banded systems in parallel and, for instance, ScaLaPack banded solvers could be called. This would work only and only if each processor handles one blocks. This is unfortunately not the case. When a processor solve several blocks, the first one could make the call to the solver routine as well as the first blocks handled by the other processors. But at one time, the blocks need to share informations (the elements of the Schur matrix in case of the ScaLaPack Algorithm, see [3]) in order to accomplish their computation. But as the other blocks of a processor have not been reached

(the processor is solving the first one), they can not call the routine and the computation is blocked.

Not only the blocks are solved sequentially but this is done in one way: when a processor finished the computation in the first block, he then goes to the second block and never come back to the first one. It means that a forward and backward strategy like the Thomas algorithm is not usable.

With all these constraints, the system is impossible to solve: consider that the first two blocks of Fig. 4.3 are solved by the same processor. The first blocks depends on 6 unknowns $x_1, \ldots, x_6$ but has only 5 equations, the system is under-determined. It needs block 2 (and also block 3 solved by another processor). But *elsA* will not allow block 2 to give back informations to block 1 leading to the impossibility of solving the system because of under-determined blocks. One could argue that *elsA* just have to be adapted, but actually the loop on the blocks in really on top of the code and there is no way it could be changed. Too much code would be impacted.

The only remaining solution is to decouple the blocks by removing the off diagonal elements in Fig. 4.3. The blocks are then completely independent and can be computed both in parallel or sequentially. There are two manners to achieve this.

4.1 Non-conservative strategy

The problem comes from the join boundary depending on w and e. As a solution, upwind schemes (see section 1.3) can be used in the implicit part, namely IU22EC4 when the join boundary is the last interface of a block and IU21EC4 when the join boundary is the first interface of a block. This strategy leads to uncoupled tridiagonal systems. The blocks are only linked in the explicit part of the scheme by the ghost cells (that changes the right-hand side b). The structure of the system is now:

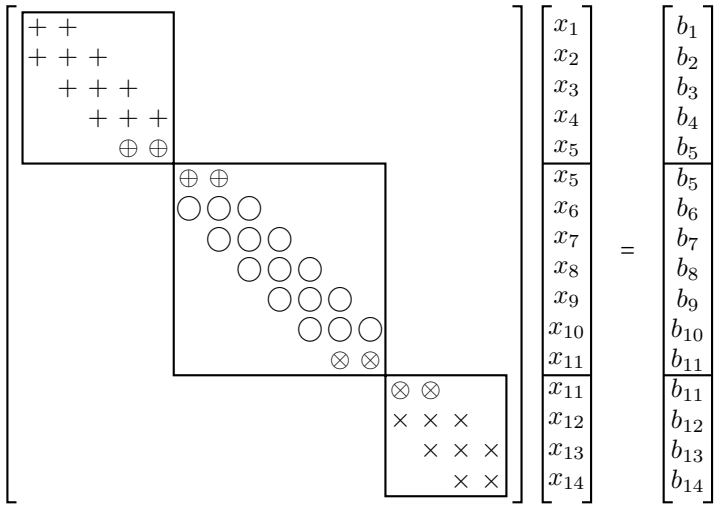
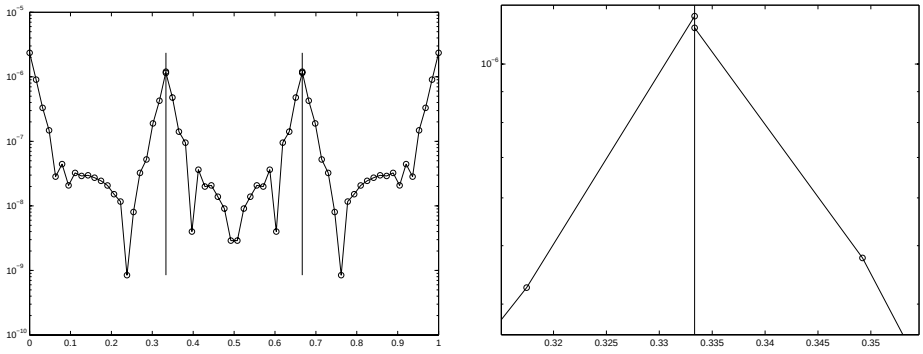


Figure 4.4: Matrix formulation of the non-conservative strategy

As the join boundary is computed by two different schemes in the adjacent blocks, its value is different in the two blocks. The upwind schemes are 5^{th} order accurate, so that the difference is not so high but it can actually be problem dependent: it would not be difficult to find a right hand side b that screws up the solution. Physically, the use of such an inexact solver leads to the loss of conservativeness of the whole scheme.

This solution is tested to observe its behavior. The function $f(x) = \sin(4\pi x)$ is divided in 3 blocks of 21 cells on the domain $[0, 1]$. The schemes are applied and the difference

between the values at the interfaces and the true values is computed on Fig. 4.5(a). The error of the solution at the interfaces if of order $1.2.10^{-6}$ when the magnitude of f is 0.8660. The relative error is very low. The solution is very satisfying.



(a) log of the absolute error (b) Zoom on the first join boundary

Figure 4.5: Error of the non-conservative strategy

In Fig. 4.5(b), the value at the first join boundary is different for the two blocks. But the difference is only $5.2.10^{-8}$. In this case there is a quasi-conservativeness which is acceptable.

4.2 Use of an explicit scheme

The simplest solution would be to compute the join boundaries explicitly leading to the following matrix structure:

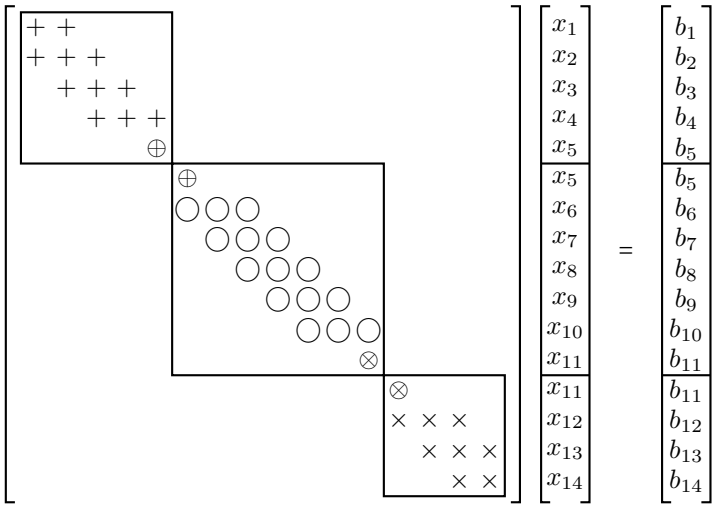


Figure 4.6: Matrix formulation of the explicit scheme strategy

A join boundary shares the same right-hand side in both blocks computed by mean of ghost cells and so this strategy is conservative.

As there are two ghost cells on each blocks, there are 4 common cells and so can achieve at most a 4^{th} order accurate scheme. To get the highest accuracy, CE4$_{nu}$ (e.g. IC1EC4) is used. It takes the metrics of the mesh into account. This scheme has already been used in Fig. 3.6 where it showed a very good behavior.

4.3 Comparison of the strategies

Two strategies have been developed: the first one has the highest order of accuracy but is not conservative. Furthermore the schemes sided in the implicit part (IUEC family) have a bad resolution (see Fig. 1.4). On the other hand the use of explicit schemes for join boundaries maintains the conservativeness. Being limited by the number of ghost cells available, the non-uniform version of CE4 is the highest order scheme available.

If the interior scheme is of order 4 (IC3EC2), then the second strategy is very satisfying: it is conservative, the order for interior domain and join boundaries is both 4 and the resolution is almost identical (see Fig. 1.2). But if the interior scheme is of order 6, then this strategy leads to join boundaries being computed with 2 order less than the interior scheme which might provoke errors. On the other hand, the use of the IUEC family for join schemes (1st strategy) has only one order less than interior domain but without the guarantee of conservativeness.

Both solution will be tested in the next section, directly in *elsA*.

Chapter 5

Application in Computational Fluid Dynamics

The present schemes and strategies have been implemented in the CFD software *elsA* [1]. This software aims at solving CFD problems for compressible flows around complex various geometries such as airplanes, turbomachinery, helicopters, missiles, rocket launchers... It was started in 1997 by ONERA (The French National Aerospace Research Establishment) and is now co-developed with CERFACS and AIRBUS. *elsA* is an object-oriented software written in C++ with core computation routines coded in Fortran.

5.1 Implementation in *elsA*

The Navier-Stokes equation in conservative form reads:

$$\frac{\partial W}{\partial t} + \frac{\partial F_x}{\partial x} + \frac{\partial F_y}{\partial y} + \frac{\partial F_z}{\partial z} = \mathcal{F}, \tag{5.1}$$

where W is the vector of conservative variables and F the convective fluxes:

$$W = \begin{pmatrix} \rho \\ \rho u \\ \rho v \\ \rho w \\ \rho E \end{pmatrix}, F_x = \begin{pmatrix} \rho u \\ \rho u^2 + p \\ \rho uv \\ \rho uw \\ (\rho E + p)u \end{pmatrix}, F_y = \begin{pmatrix} \rho v \\ \rho uv \\ \rho v^2 + p \\ \rho vw \\ (\rho E + p)v \end{pmatrix}, F_z = \begin{pmatrix} \rho w \\ \rho uw \\ \rho vw \\ \rho w^2 + p \\ (\rho E + p)w \end{pmatrix}, \tag{5.2}$$

ρ is the density of the fluid, (u, v, w) the speed vector, p the pressure and E the internal energy. The first element of these vectors corresponds to the continuity equation, the next three to the momentum equations and the last one to the energy equation. For turbulent flows computation based on RANS (Reynolds Average Navier-Stokes) up to four variables can be added to the five previous terms.

The present schemes are implemented in *elsA* to compute the fluxes through the interfaces of a cell. There are several ways to do this (see [2]), two of them are implemented in *elsA*:

- a divergence form: knowing $\overline{W}$ at the center of the cells (the bar $\bar{\cdot}$ still denotes the cell average value of a variable), $\overline{F_x}, \overline{F_y}$ and $\overline{F_z}$ are computed. By applying the scheme on each element of these vectors, one gets $\widetilde{F_x}, \widetilde{F_y}$ and $\widetilde{F_z}$. This method requires the interpolation of three vectors. The second way is computationally cheaper;

[1] Software Package for Aerodynamics Simulation, see `http://elsa.onera.fr`

- a skew symmetric form: the schemes are directly applied on $\overline{W}$ to get $\widetilde{W}$. Then $\widetilde{F_x}, \widetilde{F_y}$ and $\widetilde{F_z}$ are explicitly computed.

Both ways are implemented and it is the responsibility of the user to choose between them. Once $\widetilde{F_x}, \widetilde{F_y}$ and $\widetilde{F_z}$ are computed, the total flux passing through an interface between two cells is given by the relation:

$$F = \widetilde{F_x} S_x + \widetilde{F_y} S_y + \widetilde{F_z} S_z \tag{5.3}$$

where S_x is the surface of the interface projected on direction x.

The "cost" of a computation with the present schemes can be estimated. Let's consider a mesh of $n = i_m \times j_m \times k_m$ cells and $3n$ interfaces. n can eventually be very high, several millions for instance. The storage of the six coefficients for all the interfaces requires $18n$ memory words. Dimensional splitting is used, so that each direction can be handled separately. If the flow is laminar and the fluxes are computed in a skew symmetric form, the following systems have to be solved:

- $5 \times j_m \times k_m$ tridiagonal systems of size $i_m + 1$;
- $5 \times i_m \times k_m$ tridiagonal systems of size $j_m + 1$;
- $5 \times i_m \times j_m$ tridiagonal systems of size $k_m + 1$.

For divergence form, the number of systems to solve would be three times higher. So all these computations are expensive both in term of memory space and processor time.

To validate the implementation and the schemes, 2D computations are performed in order to check the dimensional splitting. The behavior on non-uniform grids is also tested. Finally, the domain decomposition strategies developed in the previous section are compared.

5.2 Convection of a vortex

The test corresponds to a vortex convected downstream by a uniform flow of velocity U_∞. The initial vortex is given by the stream function:

$$\Psi(x, y) = \Gamma e^{-\frac{(x-x_c)^2+(y-y_c)^2}{2R_c^2}}, \tag{5.4}$$

where Γ is the vortex strength, (x_c, y_c) is the location of the vortex center and R_c controls the size of the vortex. The resulting velocity distribution is obtained through the velocity stream function relationship:

$$u = U_\infty + \frac{\partial \Psi}{\partial y}, \quad v = -\frac{\partial \Psi}{\partial x}, \quad \Omega = \frac{\partial v}{\partial x} - \frac{\partial u}{\partial y}, \tag{5.5}$$

where Ω is the vorticity. Γ is chosen so that the speed of the vortex is much more lower than the mean flow speed to test the resolution of the schemes. The associated pressure variation follows the radial momentum equation:

$$p - p_\infty = -\frac{\Gamma^2}{2\rho R_c^2} e^{-\frac{(x-x_c)^2+(y-y_c)^2}{R_c^2}}. \tag{5.6}$$

A domain $[0, 2L] \times [-L, L]$ with $(x_c, y_c) = (L, 0)$ is considered. The boundaries in x-direction are periodic and y boundaries are slip walls. The vortex is convected by the flow during a natural number of periods so that the final and initial state should be equivalent.

As another diagnostic tool, the enstrophy is defined as one-half the square of the vorticity. This is an indicator of the shape of the vortex.

In the following computations, IC3EC4 is used for interior domain. The results are compared with the Jameson scheme already implemented in *elsA*. This scheme is 2 nd order accurate and is not design to work on non-uniform meshes. The results should be clearly in favor of the present schemes.

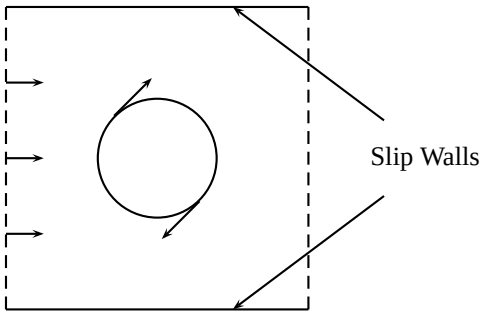
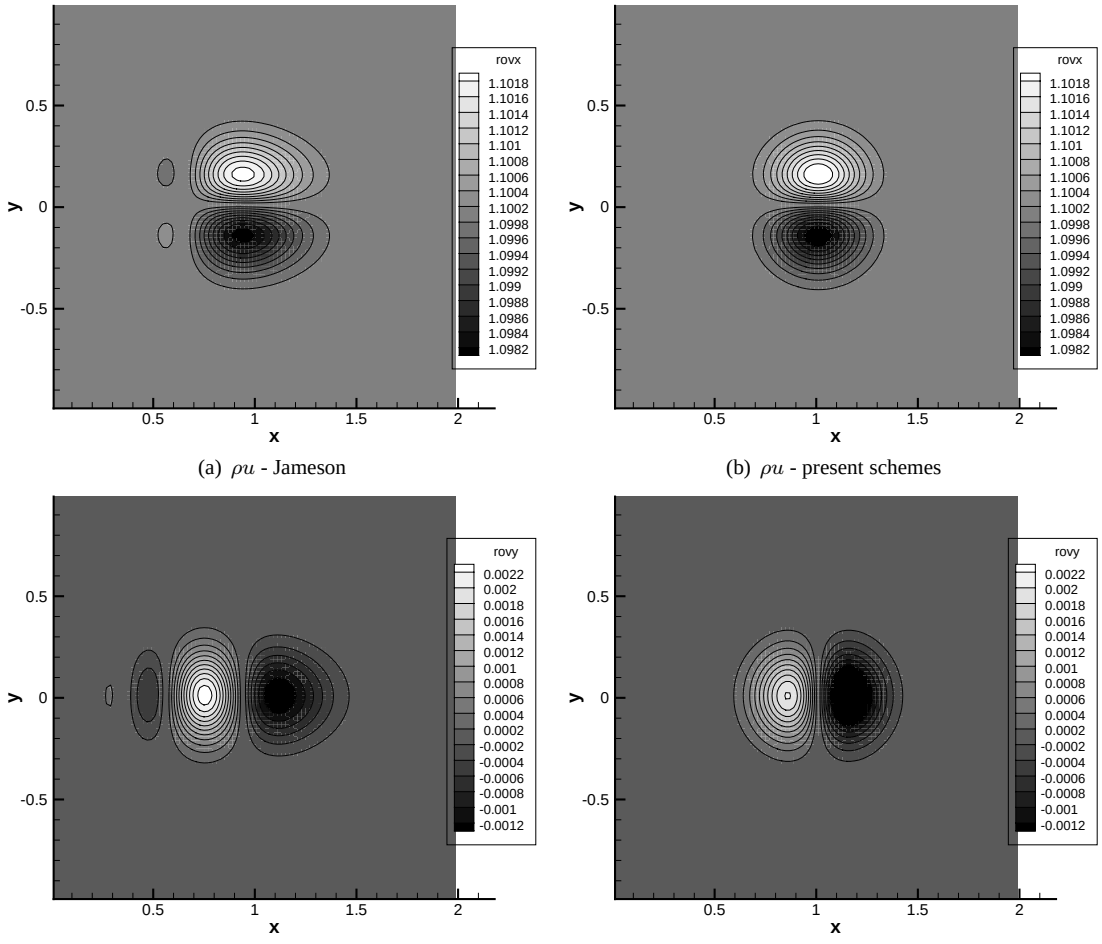


Figure 5.1: Domain of computation for the vortex convection

5.2.1 Euler Computation

After 5 periods of convection, the results are shown in Fig. 5.2. The present schemes give a very good solution that perfectly matches the initial condition. The Jameson scheme shows important deformations in the direction of the flow with the creation of counter-rotating vortices behind the original vortex.



(a) ρu - Jameson

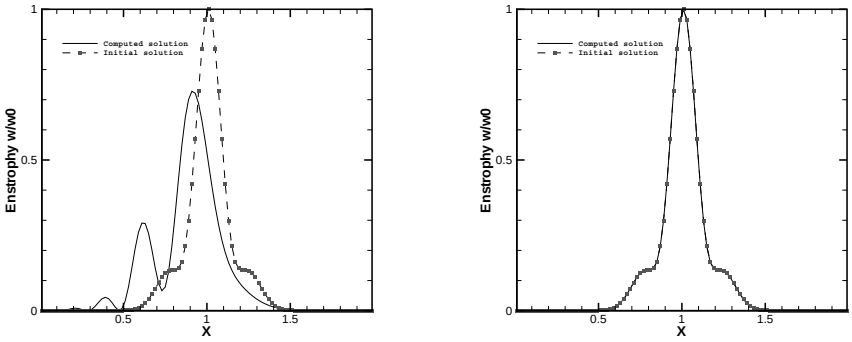
(b) ρu - present schemes

(c) ρv - Jameson

(d) ρv - present schemes

Figure 5.2: ρu and ρv after 5 periods of Euler computation

Enstrophy is a conservative quantity in two-dimensional inviscid flow. In Fig. 5.3, the initial enstrophy is in dashed line. After 5 periods, the Jameson scheme gives a really poor approximation of the solution while the present schemes solution is coincident with the initial condition.

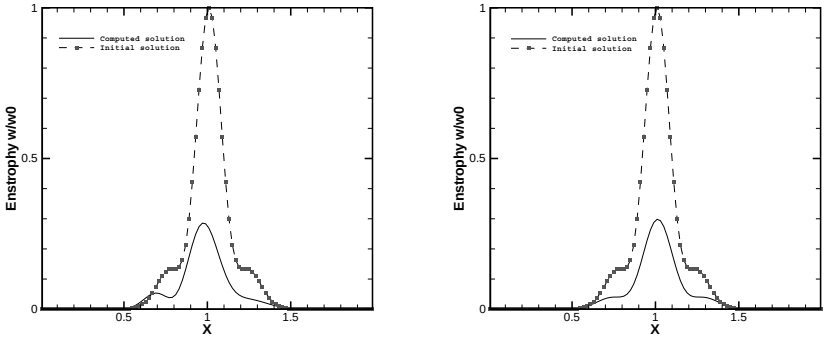


(a) Jameson (b) present schemes

Figure 5.3: Enstrophy after 5 periods of Euler computation

5.2.2 Navier-Stokes Computations

The Navier-Stokes equations take into account the viscosity of the fluid. So the curve of enstrophy should be damped by the diffusive flux:



(a) Jameson (b) present schemes

Figure 5.4: Enstrophy after 2 periods of Navier-Stokes computation

With the present schemes, the curve is damped but the shape is conserved. The Jameson scheme create a little dissymetry.

5.2.3 Non-Uniform grid

The vortex is now convected on the mesh Fig. 5.5. While the mesh is finner in the center of the grid where the vortex is, alternating cells appears the x-direction with an aspect ratio of 2.

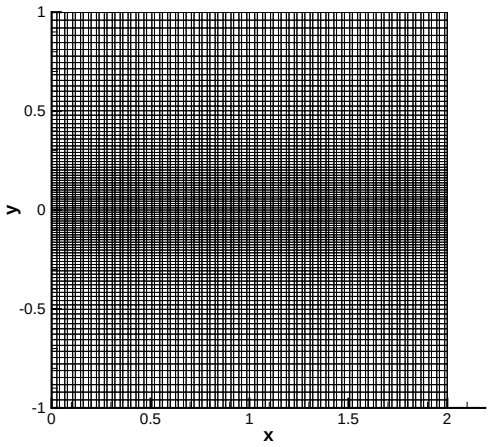
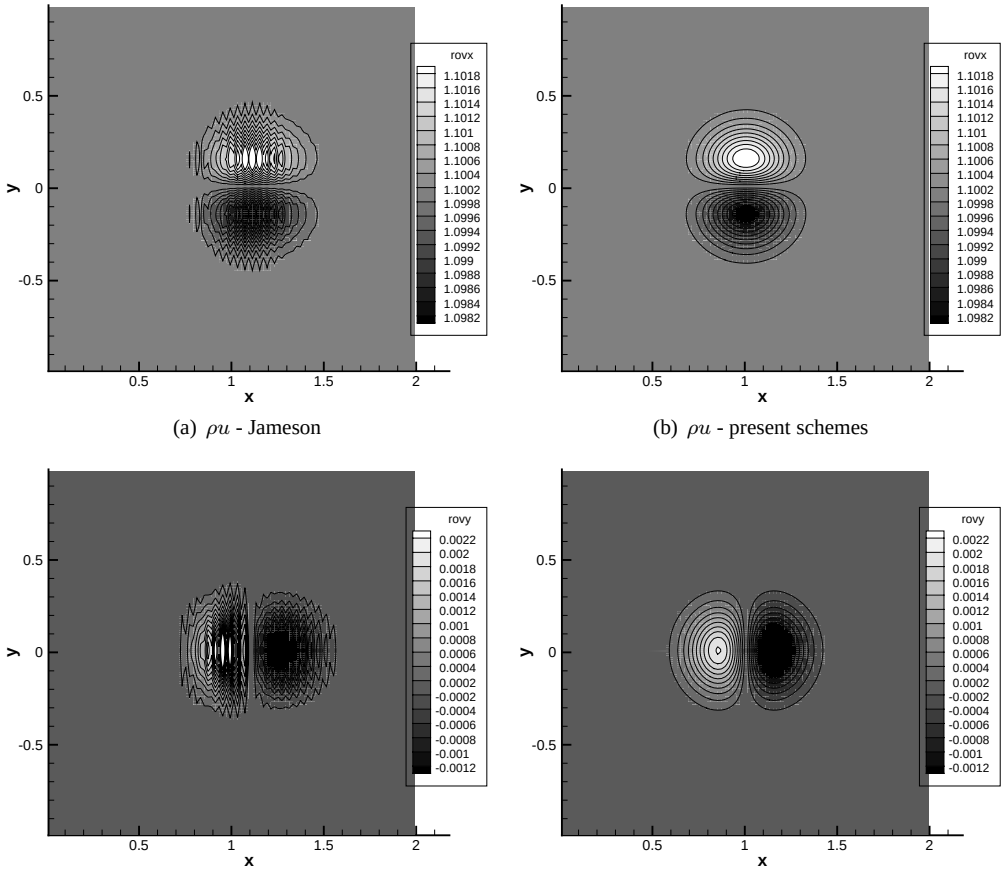


Figure 5.5: Non-uniform mesh for vortex convection

In Fig. 5.6, the results are presented after one period. For Jameson, the contour lines are very distorted because of the mesh and the convection speed is wrong since the location of the vortex is shifted to the right. The present schemes give a good result.



(a) ρu - Jameson

(b) ρu - present schemes

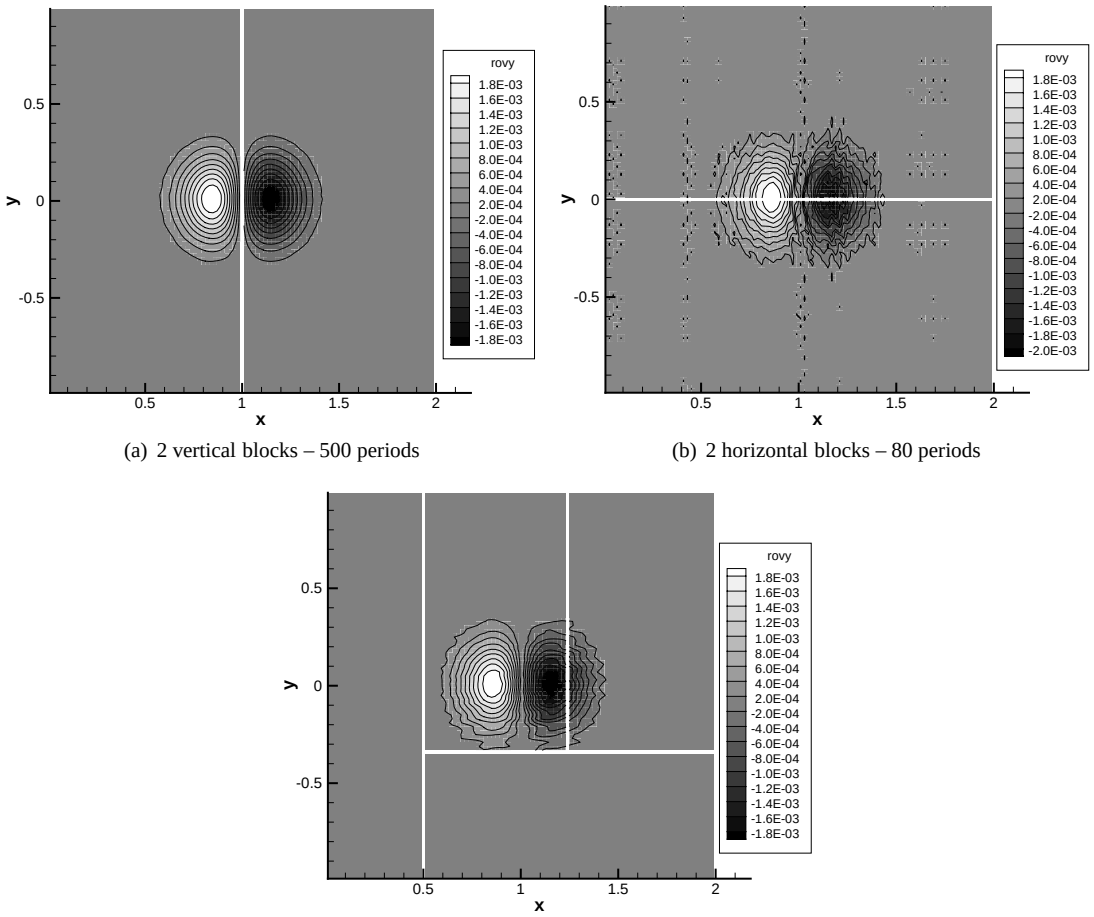
(c) ρv - Jameson

(d) ρv - present schemes

Figure 5.6: ρu and ρv after 1 period on non-uniform grid

5.2.4 Multi-blocks Computations

The uniform domain is now separated into several blocks. The non-conservative strategy is first tested on three cases in Fig 5.7. In (a), the domain is split into two vertical blocks, after 500 periods, the computed solution is really good compared to the initial condition. On (b), the domain is also split into two blocks, but horizontally. As the vortex is now always on two blocks, the effects of the non-conservativeness are clearly visible after 80 periods. And on the 4 blocks domain of subfigure (c), these effects are sensible after 20 periods.



(a) 2 vertical blocks – 500 periods

(b) 2 horizontal blocks – 80 periods

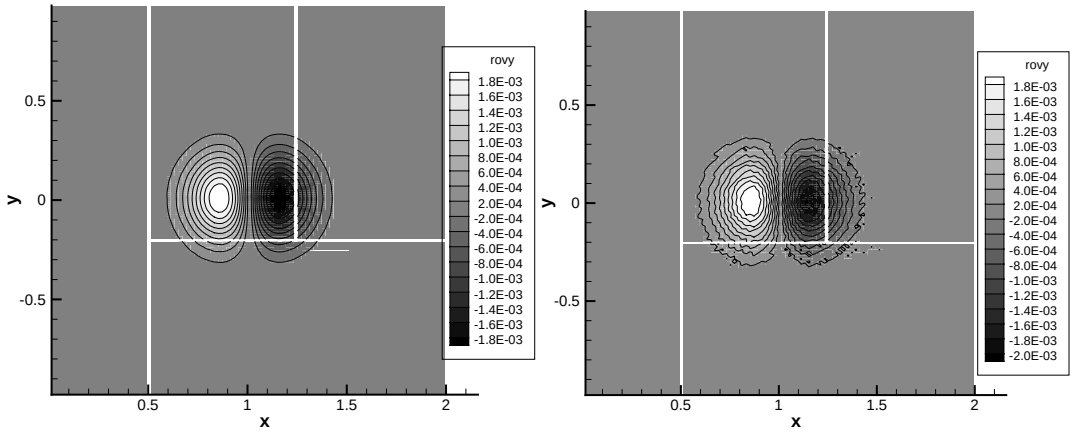
(c) 4 blocks – 20 periods

Figure 5.7: ρv – Results of multi-blocks computations

With the grid shown in Fig 5.5, the results are better. Even if the x direction has alternating size cells, the refinement around $y = 0$ allow the IUEC schemes to be more precise and the effect of non-conservativeness less sensitive as shown in Fig. 5.8 on the 4 blocks configuration. For the same number of periods and blocks, Fig. 5.8(a) is a lot better than Fig. 5.7(c). The effects of non-conservativeness are sensitive after 32 periods.

On the same times, the Jameson scheme gives worse solution, but even though this situation is clearly not satisfying. The non-conservativeness of the global scheme is a problem.

The conservative strategy is now tested for comparison. By using the explicit scheme CE4$_{nu}$ on join boundaries, the results are a lot better as shown in Fig. 5.9 obtained after 100 periods. Even though the results are slightly less accurate than in a one-block configuration where all the interior domain is computed with IC3EC4, the results are clearly



(a) 20 periods (b) 32 periods

Figure 5.8: 4 blocks computation on non-uniform grid

better than with the non-conservative strategy since the computed solution is very close to the initial condition after 100 periods.

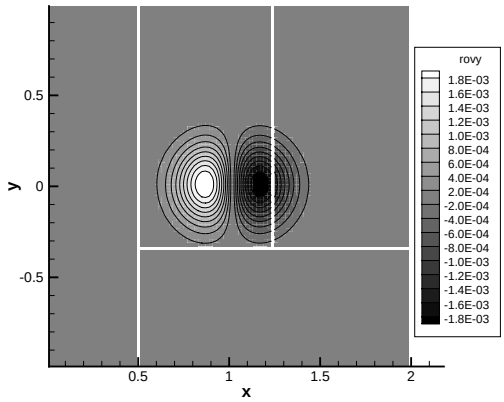


Figure 5.9: 4 blocks computation with explicit scheme on join boundaries (100 periods)

On the figure 5.10, the initial ρv (solid line) for $y = 0$ is compared to the 4-blocks configurations after 100 periods of convection in dashed line. The interior scheme IC3EC4 is of order 6, and CE4$_{nu}$ is used on join boundaries like Fig. 5.9. Even if the join treatment has 2 order less than the interior scheme, the computed solution approximate very well the initial solution so that the loss of accuracy is really small.

The conservativeness and the resolution of the join treatment appear to be a more important property than its order and formal truncation error.

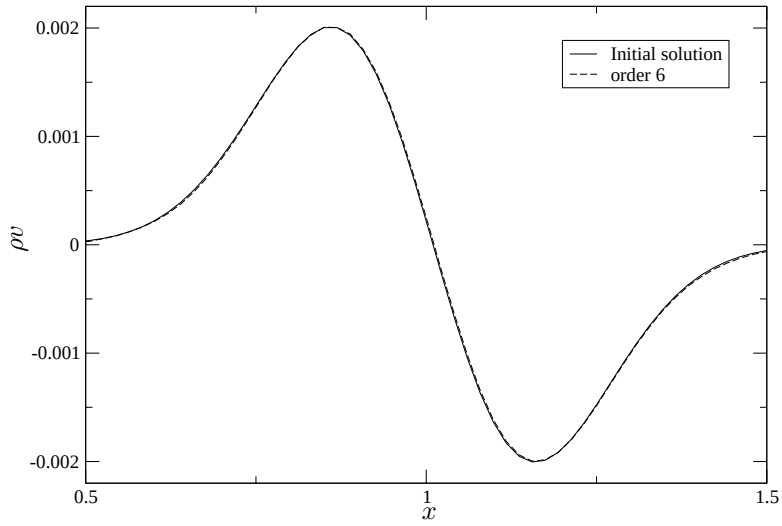


Figure 5.10: ρv at $y = 0$

Conclusion

A family of high-order implicit schemes has been developed: centered schemes of order 4 and 6 used in the interior domain and various upwind schemes used for the boundaries. They can directly be applied on convective terms and a procedure for diffusive fluxes has been derived.

These schemes and procedure have been deeply tested on cases usually used in the literature. These tests have been carried out in Matlab for its simplicity of use and speed of development. They showed a good behavior.

Then, they have been implemented in a Computational Fluid Dynamics software to compute the fluxes of the Navier-Stokes equations. This step required an intensive phase of coding and the development of methods for multi-domain treatment mixing parallel and sequential computations. The schemes have been tested on a vortex convection problem on uniform and non-uniform grids. The present schemes clearly show a better representation of the solution than the classical Jameson scheme. For multi-blocks configurations, the conservativeness of the join treatment appears to be a more important property than its formal order of accuracy. Nevertheless, the latter is still higher than classical schemes.

Acknowledgments

This thesis is submitted as part of the requirements of the International Master's Programme in Engineering Mathematics at the Department of Mathematics, Chalmers University of Technology.

It has been carried out at CERFACS (European Center for Research and Advanced Training in Scientific Computing) in Toulouse (France) in co-operation with UTC (Compiègne University of Technology). I was a member of the CFD (Computational Fluid Dynamics) team dedicated to aerodynamic simulation.

I would like to thank many people who helped me and supported me during these six months:

- my supervisor Jean-François BOUSSUGE for always being so helpful despite his busy schedule;
- Guilhem CHEVALIER and Thierry POINSOT, leaders of the CFD team who welcomed me in their research team;
- Serge GRATTON, senior researcher at the parallel algorithm team for his kindness and help;
- Marc MONTAGNAC, *elsA* guru, who patiently answered all my questions about the implementation of the present schemes;
- Guillaume PUIGT and Jean-Phillipe BOIN for carefully reading this thesis and their very useful comments;
- all the people involved in the CFD team, especially my office mates, Yann, Florian and Julien. The atmosphere is great and it was a daily pleasure to work with them;
- the computer support group members who solved all my (numerous) computer problems very quickly.

I also want to express my special gratitude to the people at the Department of Mathematics, especially Ivar GUSTAFSSON, for their great teaching.

Toulouse, February 2006
Frédéric Sicot

Appendix A

Details of the derivation of the schemes

A.1 General scheme

u is expanded into its Taylor series about point $x_{i+1/2}$:

$$\begin{aligned} u(x) =& u_{i+1/2} + u'_{i+1/2}(x - x_{i+1/2}) + \frac{1}{2}u''_{i+1/2}(x - x_{i+1/2})^2 \\ &+ \frac{1}{3!}u'''_{i+1/2}(x - x_{i+1/2})^3 + \frac{1}{4!}u^{(4)}_{i+1/2}(x - x_{i+1/2})^4 \\ &+ \frac{1}{5!}u^{(5)}_{i+1/2}(x - x_{i+1/2})^5 + O(h^6) \end{aligned} \tag{A.1}$$

A.1.1 Left Hand Side

Setting $h_k = x_{k+1/2} - x_{k-1/2}$, the computation of the lhs gives:

$$\begin{aligned} u_{i-1/2} &= u_{i+1/2} - h_i u'_{i+1/2} + \frac{1}{2}h_i^2 u''_{i+1/2} - \frac{1}{3!}h_i^3 u'''_{i+1/2} + \frac{1}{4!}h_i^4 u^{(4)}_{i+1/2} \\ &\quad -\frac{1}{5!}h_i^5 u^{(5)}_{i+1/2} \end{aligned} \tag{A.2}$$

$$\begin{aligned} u_{i+3/2} &= u_{i+1/2} + h_{i+1} u'_{i+1/2} + \frac{1}{2}h_{i+1}^2 u''_{i+1/2} + \frac{1}{3!}h_{i+1}^3 u'''_{i+1/2} + \frac{1}{4!}h_{i+1}^4 u^{(4)}_{i+1/2} \\ &\quad +\frac{1}{5!}h_{i+1}^5 u^{(5)}_{i+1/2} \end{aligned} \tag{A.3}$$

So that the lhs of (1.6) becomes:

$$\begin{aligned} lhs =& u_{i+1/2}(1 + \beta + \gamma) + u'_{i+1/2}\left(\gamma h_{i+1} - \beta h_i\right) + u''_{i+1/2}\left(\frac{\beta h_i^2 + \gamma h_{i+1}^2}{2}\right) \\ &+ u'''_{i+1/2}\left(\frac{\gamma h_{i+1}^3 - \beta h_i^3}{3!}\right) + u^{(4)}_{i+1/2}\left(\frac{\beta h_i^4 + \gamma h_{i+1}^4}{4!}\right) \\ &+ u^{(5)}_{i+1/2}\left(\frac{\gamma h_{i+1}^5 - \beta h_i^5}{5!}\right) \end{aligned} \tag{A.4}$$

A.1.2 Right Hand Side

To compute $\bar{u}$, (A.1) is integrated over the different cells:

$$\begin{aligned}
\bar{u}_{i-1} =& \frac{1}{x_{i-1/2} - x_{i-3/2}} \int_{x_{i-3/2}}^{x_{i-1/2}} u(x)\,\mathrm{d}x = \frac{1}{h_{i-1}} \int_{x_{i+1/2}-h_i-h_{i-1}}^{x_{i+1/2}-h_i} u(x)\,\mathrm{d}x \\
=& u_{i+1/2} + u'_{i+1/2}\left[-\frac{1}{2}(2h_i + h_{i-1})\right] \\
&+ u''_{i+1/2}\left[\frac{1}{3!h_{i-1}}\left((h_i + h_{i-1})^3 - h_i^3\right)\right] \\
&+ u'''_{i+1/2}\left[\frac{1}{4!h_{i-1}}\left(h_i^4 - (h_i + h_{i-1})^4\right)\right] \\
&+ u^{(4)}_{i+1/2}\left[\frac{1}{5!h_{i-1}}\left((h_i + h_{i-1})^5 - h_i^5\right)\right] \\
&+ u^{(5)}_{i+1/2}\left[\frac{1}{6!h_{i-1}}\left(h_i^6 - (h_i + h_{i-1})^6\right)\right]
\end{aligned} \tag{A.5}$$

$$\begin{aligned}
\bar{u}_i =& \frac{1}{x_{i+1/2} - x_{i-1/2}} \int_{x_{i-1/2}}^{x_{i+1/2}} u(x)\,\mathrm{d}x = \frac{1}{h_i} \int_{x_{i+1/2}-h_i}^{x_{i+1/2}} u(x)\,\mathrm{d}x \\
=& u_{i+1/2} + u'_{i+1/2}\left[-\frac{h_i}{2}\right] + u''_{i+1/2}\left[\frac{h_i^2}{3!}\right] + u'''_{i+1/2}\left[-\frac{h_i^3}{4!}\right] \\
&+ u^{(4)}_{i+1/2}\left[\frac{h_i^4}{5!}\right] + u^{(5)}_{i+1/2}\left[-\frac{h_i^5}{6!}\right]
\end{aligned} \tag{A.6}$$

$$\begin{aligned}
\bar{u}_{i+1} =& \frac{1}{x_{i+3/2} - x_{i+1/2}} \int_{x_{i+1/2}}^{x_{i+3/2}} u(x)\,\mathrm{d}x = \frac{1}{h_{i+1}} \int_{x_{i+1/2}}^{x_{i+1/2}+h_{i+1}} u(x)\,\mathrm{d}x \\
=& u_{i+1/2} + u'_{i+1/2}\left[\frac{h_{i+1}}{2}\right] + u''_{i+1/2}\left[\frac{h_{i+1}^2}{3!}\right] + u'''_{i+1/2}\left[\frac{h_{i+1}^3}{4!}\right] \\
&+ u^{(4)}_{i+1/2}\left[\frac{h_{i+1}^4}{5!}\right] + u^{(5)}_{i+1/2}\left[\frac{h_{i+1}^5}{6!}\right]
\end{aligned} \tag{A.7}$$

$$\begin{aligned}
\bar{u}_{i+2} =& \frac{1}{x_{i+5/2} - x_{i+3/2}} \int_{x_{i+3/2}}^{x_{i+5/2}} u(x)\,\mathrm{d}x = \frac{1}{h_{i+2}} \int_{x_{i+1/2}+h_{i+1}}^{x_{i+1/2}+h_{i+1}+h_{i+2}} u(x)\,\mathrm{d}x \\
=& u_{i+1/2} + u'_{i+1/2}\left[\frac{1}{2}(2h_{i+1} + h_{i+2})\right] + u''_{i+1/2}\left[\frac{1}{3!h_{i+2}}\left((h_{i+1} + h_{i+2})^3 - h_{i+1}^3\right)\right] \\
&+ u'''_{i+1/2}\left[\frac{1}{4!h_{i+2}}\left((h_{i+1} + h_{i+2})^4 - h_{i+1}^4\right)\right] + u^{(4)}_{i+1/2}\left[\frac{1}{5!h_{i+2}}\left((h_{i+1} + h_{i+2})^5 - h_{i+1}^5\right)\right] \\
&+ u^{(5)}_{i+1/2}\left[\frac{1}{6!h_{i+2}}\left((h_{i+1} + h_{i+2})^6 - h_{i+1}^6\right)\right]
\end{aligned} \tag{A.8}$$

So that the rhs of (1.6) becomes:

$$
\begin{aligned}
rhs =& \frac{1}{h_{i+2}} \int_{x_{i+3/2}}^{x_{i+5/2}} u(x)\,\mathrm{d}x \\
=& u_{i+1/2}(a+b+c+d) \\
&+ u'_{i+1/2}\frac{1}{2}\Big[-a(2h_i+h_{i-1}) - bh_i + ch_{i+1} + d(2h_{i+1}+h_{i+2})\Big] \\
&+ u''_{i+1/2}\frac{1}{3!}\left[a\frac{1}{h_{i-1}}\Big((h_i+h_{i-1})^3 - h_i^3\Big) + bh_i^2 + ch_{i+1}^2 + d\frac{1}{h_{i+2}}\Big((h_{i+1}+h_{i+2})^3 - h_{i+1}^3\Big)\right] \\
&+ u'''_{i+1/2}\frac{1}{4!}\left[a\frac{1}{h_{i-1}}\Big(h_i^4 - (h_i+h_{i-1})^4\Big) - bh_i^3 + ch_{i+1}^3 + d\frac{1}{h_{i+2}}\Big((h_{i+1}+h_{i+2})^4 - h_{i+1}^4\Big)\right] \\
&+ u^{(4)}_{i+1/2}\frac{1}{5!}\left[a\frac{1}{h_{i-1}}\Big((h_i+h_{i-1})^5 - h_i^5\Big) + bh_i^4 + ch_{i+1}^4 + d\frac{1}{h_{i+2}}\Big((h_{i+1}+h_{i+2})^5 - h_{i+1}^5\Big)\right] \\
&+ u^{(5)}_{i+1/2}\frac{1}{6!}\left[a\frac{1}{h_{i-1}}\Big(h_i^6 - (h_i+h_{i-1})^6\Big) - bh_i^5 + ch_{i+1}^5 + d\frac{1}{h_{i+2}}\Big((h_{i+1}+h_{i+2})^6 - h_{i+1}^6\Big)\right]
\end{aligned}
\tag{A.9}
$$

A.1.3 Matrix formulation

By matching the terms of each derivatives in (A.4) and (A.9), one finds the constraints (1.7)-(1.12). An easier way to describe these constraints is to find the solution of the linear system $Ax = b$ where A is:

$$
\begin{pmatrix}
1 & 1 & -1 & -1 & -1 & -1 \\
-h_i & h_{i+1} & h_i + \frac{h_{i-1}}{2} & \frac{h_i}{2} & -\frac{h_{i+1}}{2} & -h_{i+1} - \frac{h_{i+2}}{2} \\
h_i^2 & h_{i+1}^2 & \frac{1}{3h_{i-1}}\left(h_i^3 - (h_i+h_{i-1})^3\right) & -\frac{h_i^2}{3} & -\frac{h_{i+1}^2}{3} & \frac{1}{3h_{i+2}}\left(h_{i+1}^3 - (h_{i+1}+h_{i+2})^3\right) \\
-h_i^3 & h_{i+1}^3 & \frac{1}{4h_{i-1}}\left((h_i+h_{i-1})^4 - h_i^4\right) & \frac{h_i^3}{4} & -\frac{h_{i+1}^3}{4} & \frac{1}{4h_{i+2}}\left(h_{i+1}^4 - (h_{i+1}+h_{i+2})^4\right) \\
h_i^4 & h_{i+1}^4 & \frac{1}{5h_{i-1}}\left(h_i^5 - (h_i+h_{i-1})^5\right) & -\frac{h_i^4}{5} & -\frac{h_{i+1}^4}{5} & \frac{1}{5h_{i+2}}\left(h_{i+1}^5 - (h_{i+1}+h_{i+2})^5\right) \\
-h_i^5 & h_{i+1}^5 & \frac{1}{6h_{i-1}}\left((h_i+h_{i-1})^6 - h_i^6\right) & \frac{h_i^5}{6} & -\frac{h_{i+1}^5}{6} & \frac{1}{6h_{i+2}}\left(h_{i+1}^6 - (h_{i+1}+h_{i+2})^6\right)
\end{pmatrix}
\tag{A.10}
$$

The constraints are written by rows with increasing order of accuracy. The columns correspond to the coefficients. Indeed the unknwon vector is $x = (\beta, \gamma, a, b, c, d)^\top$ and the right hand-side is given by $b = (-1, 0, 0, 0, 0, 0)^\top$. This approach enables us to easily compute the coefficients of all the family: for the other schemes, the columns corresponding to zero coefficients are removed. And as the order of accuracy decreases, the last rows of the matrix are also removed, the same number as removed coefficients.

A.2 Centered Scheme

A.2.1 Coefficients of the IC3EC4 scheme

The coefficients are given by (A.11)-(A.16):

$$
\beta = \frac{h_{i+1}^2\,(h_{i-1}+h_i)\,(h_{i+1}+h_{i+2})}{h_{i-1}\,(h_i+h_{i+1})^2\,(h_i+h_{i+1}+h_{i+2})} \tag{A.11}
$$

$$\gamma = \frac{h_i^2 \left(h_{i-1} + h_i\right) \left(h_{i+1} + h_{i+2}\right)}{\left(h_i + h_{i+1}\right)^2 \left(h_{i-1} + h_i + h_{i+1}\right) h_{i+2}} \tag{A.12}$$

$$a = \frac{h_i^2 h_{i+1}^2 \left(h_{i+1} + h_{i+2}\right)}{h_{i-1} \left(h_{i-1} + h_i\right) \left(h_{i-1} + h_i + h_{i+1}\right)^2 \left(h_{i-1} + h_i + h_{i+1} + h_{i+2}\right)} \tag{A.13}$$

$$\begin{aligned} b = & \frac{h_{i+1}^2}{h_{i-1}^2} \left(\frac{2h_{i-1}h_i \left(h_{i-1} + h_i\right)}{\left(h_i + h_{i+1}\right)^3} + \frac{\left(2h_{i-1} - h_i\right) \left(h_{i-1} + h_i\right)}{\left(h_i + h_{i+1}\right)^2} \right. \\ & + \frac{h_i^3}{\left(h_{i-1} + h_i\right) \left(h_{i-1} + h_i + h_{i+1}\right)^2} - \frac{h_{i-1} h_i^2 \left(h_{i-1} + h_i\right)}{\left(h_i + h_{i+1}\right)^2 \left(h_i + h_{i+1} + h_{i+2}\right)^2} \\ & + \frac{h_i \left(h_{i-1} + h_i\right) \left(h_i \left(h_i + h_{i+1}\right) - h_{i-1} \left(3h_i + h_{i+1}\right)\right)}{\left(h_i + h_{i+1}\right)^3 \left(h_i + h_{i+1} + h_{i+2}\right)} \\ & \left. - \frac{h_i^3}{\left(h_{i-1} + h_i + h_{i+1}\right)^2 \left(h_{i-1} + h_i + h_{i+1} + h_{i+2}\right)} \right) \end{aligned} \tag{A.14}$$

$$\begin{aligned} c = & \frac{h_i^2 \left(h_{i-1} + h_i\right) \left(2h_{i-1} \left(h_i + 2h_{i+1}\right) + \left(h_i + h_{i+1}\right) \left(2h_i + 5h_{i+1}\right)\right)}{\left(h_i + h_{i+1}\right)^3 \left(h_{i-1} + h_i + h_{i+1}\right)^2} \\ & + \frac{h_{i+1}}{h_{i+1} + h_{i+2}} - \frac{h_i \left(h_{i-1} + h_i\right) h_{i+1}^3}{h_{i-1} \left(h_i + h_{i+1}\right)^2 \left(h_i + h_{i+1} + h_{i+2}\right)^2} \\ & - \frac{\left(h_{i-1} + h_i\right) h_{i+1}^3 \left(-h_i \left(h_i + h_{i+1}\right) + h_{i-1} \left(3h_i + h_{i+1}\right)\right)}{h_{i-1}^2 \left(h_i + h_{i+1}\right)^3 \left(h_i + h_{i+1} + h_{i+2}\right)} \\ & - \frac{h_i^2 h_{i+1}^3}{h_{i-1}^2 \left(h_{i-1} + h_i + h_{i+1}\right)^2 \left(h_{i-1} + h_i + h_{i+1} + h_{i+2}\right)} \end{aligned} \tag{A.15}$$

$$d = \frac{h_i^2 h_{i+1}^2 \left(h_{i-1} + h_i\right)}{h_{i+2} \left(h_{i+1} + h_{i+2}\right) \left(h_i + h_{i+1} + h_{i+2}\right)^2 \left(h_{i-1} + h_i + h_{i+1} + h_{i+2}\right)} \tag{A.16}$$

A.2.2 Coefficients of the IC3EC2 scheme

This scheme is of order 4. As $a = d = 0$, the columns 3 and 6 of A are removed as well as the last 2 rows so that the system becomes:

$$\begin{pmatrix} 1 & 1 & -1 & -1 \\ -h_i & h_{i+1} & \frac{h_i}{2} & -\frac{h_{i+1}}{2} \\ h_i^2 & h_{i+1}^2 & -\frac{h_i^2}{3} & -\frac{h_{i+1}^2}{3} \\ -h_i^3 & h_{i+1}^3 & \frac{h_i^3}{4} & -\frac{h_{i+1}^3}{4} \end{pmatrix} \begin{pmatrix} \beta \\ \gamma \\ c \\ d \end{pmatrix} = \begin{pmatrix} -1 \\ 0 \\ 0 \\ 0 \end{pmatrix} \tag{A.17}$$

The coefficients are given by:

$$\beta = \frac{h_{i+1}^2}{(h_i + h_{i+1})^2} \tag{A.18}$$

$$\gamma = \frac{h_i^2}{(h_i + h_{i+1})^2} \tag{A.19}$$

$$b = \frac{2h_{i+1}^2(2h_i + h_{i+1})}{(h_i + h_{i+1})^3} \tag{A.20}$$

$$c = \frac{2h_i^2(h_i + 2h_{i+1})}{(h_i + h_{i+1})^3} \tag{A.21}$$

A.3 Upwind schemes

A.3.1 Coefficients of the IC3EU31 scheme

As $a = 0$, the last row and third column of A are removed so that the system becomes:

$$\begin{pmatrix} 1 & 1 & -1 & -1 & -1 \\ -h_i & h_{i+1} & \frac{h_i}{2} & -\frac{h_{i+1}}{2} & -h_{i+1} - \frac{h_{i+2}}{2} \\ h_i^2 & h_{i+1}^2 & -\frac{h_i^2}{3} & -\frac{h_{i+1}^2}{3} & \frac{1}{3h_{i+2}}\left(h_{i+1}^3 - (h_{i+1} + h_{i+2})^3\right) \\ -h_i^3 & h_{i+1}^3 & \frac{h_i^3}{4} & -\frac{h_{i+1}^3}{4} & \frac{1}{4h_{i+2}}\left(h_{i+1}^4 - (h_{i+1} + h_{i+2})^4\right) \\ h_i^4 & h_{i+1}^4 & -\frac{h_i^4}{5} & -\frac{h_{i+1}^4}{5} & \frac{1}{5h_{i+2}}\left(h_{i+1}^5 - (h_{i+1} + h_{i+2})^5\right) \end{pmatrix} \begin{pmatrix} \beta \\ \gamma \\ b \\ c \\ d \end{pmatrix} = \begin{pmatrix} -1 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix} \tag{A.22}$$

The solution is given by:

$$\beta = \frac{h_{i+1}^2(h_{i+1} + h_{i+2})}{(h_i + h_{i+1})^2(h_i + h_{i+1} + h_{i+2})} \tag{A.23}$$

$$\gamma = \frac{h_i^2(h_{i+1} + h_{i+2})}{(h_i + h_{i+1})^2 h_{i+2}} \tag{A.24}$$

$$b = \frac{h_{i+1}^2(h_{i+1} + h_{i+2})(5h_i^2 + 2h_{i+1}(h_{i+1} + h_{i+2}) + h_i(7h_{i+1} + 4h_{i+2}))}{(h_i + h_{i+1})^3(h_i + h_{i+1} + h_{i+2})^2} \tag{A.25}$$

$$\begin{aligned} c = & \frac{2h_i^2(h_i + 2h_{i+1})}{(h_i + h_{i+1})^3} + \frac{h_{i+1}}{h_{i+1} + h_{i+2}} - \frac{h_i h_{i+1}^3}{(h_i + h_{i+1})^2(h_i + h_{i+1} + h_{i+2})^2} \\ & - \frac{h_{i+1}^3(3h_i + h_{i+1})}{(h_i + h_{i+1})^3(h_i + h_{i+1} + h_{i+2})} \end{aligned} \tag{A.26}$$

$$d = \frac{h_i^2 h_{i+1}^2}{h_{i+2}(h_{i+1} + h_{i+2})(h_i + h_{i+1} + h_{i+2})^2} \tag{A.27}$$

A.3.2 Coefficients of the IC3EU32 scheme

As $d = 0$, the last row and column of A are removed so that the system becomes:

$$\begin{pmatrix} 1 & 1 & -1 & -1 & -1 \\ -h_i & h_{i+1} & h_i + \frac{h_{i-1}}{2} & \frac{h_i}{2} & -\frac{h_{i+1}}{2} \\ h_i^2 & h_{i+1}^2 & \frac{1}{3h_{i-1}}\left(h_i^3 - (h_i + h_{i-1})^3\right) & -\frac{h_i^2}{3} & -\frac{h_{i+1}^2}{3} \\ -h_i^3 & h_{i+1}^3 & \frac{1}{4h_{i-1}}\left((h_i + h_{i-1})^4 - h_i^4\right) & \frac{h_i^3}{4} & -\frac{h_{i+1}^3}{4} \\ h_i^4 & h_{i+1}^4 & \frac{1}{5h_{i-1}}\left(h_i^5 - (h_i + h_{i-1})^5\right) & -\frac{h_i^4}{5} & -\frac{h_{i+1}^4}{5} \end{pmatrix} \begin{pmatrix} \beta \\ \gamma \\ a \\ b \\ c \end{pmatrix} = \begin{pmatrix} -1 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix} \tag{A.28}$$

The solution is given by:

$$\beta = \frac{(h_{i-1} + h_i)\, h_{i+1}^2}{h_{i-1}\,(h_i + h_{i+1})^2} \tag{A.29}$$

$$\gamma = \frac{h_i^2\,(h_{i-1} + h_i)}{(h_i + h_{i+1})^2\,(h_{i-1} + h_i + h_{i+1})} \tag{A.30}$$

$$a = \frac{h_i^2 h_{i+1}^2}{h_{i-1}\,(h_{i-1} + h_i)\,(h_{i-1} + h_i + h_{i+1})^2} \tag{A.31}$$

$$\begin{aligned} b = & \frac{h_{i+1}^2}{(h_{i-1} + h_i)\,(h_i + h_{i+1})^3\,(h_{i-1} + h_i + h_{i+1})^2}\Big(2h_{i-1}^3\,(2h_i + h_{i+1}) \\ & + 2h_{i-1}\,(2h_i + h_{i+1})\left(5h_i^2 + 5h_i h_{i+1} + h_{i+1}^2\right) \\ & + h_i\,(h_i + h_{i+1})\left(10h_i^2 + 10h_i h_{i+1} + 3h_{i+1}^2\right) \\ & + h_{i-1}^2\left(15h_i^2 + 15h_i h_{i+1} + 4h_{i+1}^2\right)\Big) \end{aligned} \tag{A.32}$$

$$c = \frac{h_i^2\,(h_{i-1} + h_i)\,(2h_{i-1}\,(h_i + 2h_{i+1}) + (h_i + h_{i+1})\,(2h_i + 5h_{i+1}))}{(h_i + h_{i+1})^3\,(h_{i-1} + h_i + h_{i+1})^2} \tag{A.33}$$

A.3.3 Coefficients of the IU21EC4 scheme

$\beta = 0$ and the system becomes:

$$\begin{pmatrix} 1 & -1 & -1 & -1 & -1 \\ h_{i+1} & h_i + \frac{h_{i-1}}{2} & \frac{h_i}{2} & -\frac{h_{i+1}}{2} & -h_{i+1} - \frac{h_{i+2}}{2} \\ h_{i+1}^2 & \frac{1}{3h_{i-1}}\left(h_i^3 - (h_i + h_{i-1})^3\right) & -\frac{h_i^2}{3} & -\frac{h_{i+1}^2}{3} & \frac{1}{3h_{i+2}}\left(h_{i+1}^3 - (h_{i+1} + h_{i+2})^3\right) \\ h_{i+1}^3 & \frac{1}{4h_{i-1}}\left((h_i + h_{i-1})^4 - h_i^4\right) & \frac{h_i^3}{4} & -\frac{h_{i+1}^3}{4} & \frac{1}{4h_{i+2}}\left(h_{i+1}^4 - (h_{i+1} + h_{i+2})^4\right) \\ h_{i+1}^4 & \frac{1}{5h_{i-1}}\left(h_i^5 - (h_i + h_{i-1})^5\right) & -\frac{h_i^4}{5} & -\frac{h_{i+1}^4}{5} & \frac{1}{5h_{i+2}}\left(h_{i+1}^5 - (h_{i+1} + h_{i+2})^5\right) \end{pmatrix} \tag{A.34}$$

The coefficients are:

$$\gamma = \frac{h_i \left(h_{i-1} + h_i\right) \left(h_{i+1} + h_{i+2}\right)}{\left(h_i + h_{i+1}\right) \left(h_{i-1} + h_i + h_{i+1}\right) h_{i+2}} \tag{A.35}$$

$$a = -\frac{h_i h_{i+1}^2 \left(h_{i+1} + h_{i+2}\right)}{\left(h_{i-1} + h_i\right) \left(h_{i-1} + h_i + h_{i+1}\right)^2 \left(h_{i-1} + h_i + h_{i+1} + h_{i+2}\right)} \tag{A.36}$$

$$\begin{aligned} b = & \frac{h_{i+1}^2}{h_{i-1}} \left(\frac{h_{i-1} + h_i}{\left(h_i + h_{i+1}\right)^2} - \frac{h_i^2}{\left(h_{i-1} + h_i\right) \left(h_{i-1} + h_i + h_{i+1}\right)^2} \right. \\ & - \frac{h_i \left(h_{i-1} + h_i\right)}{\left(h_i + h_{i+1}\right)^2 \left(h_i + h_{i+1} + h_{i+2}\right)} \\ & \left. + \frac{h_i^2}{\left(h_{i-1} + h_i + h_{i+1}\right)^2 \left(h_{i-1} + h_i + h_{i+1} + h_{i+2}\right)} \right) \end{aligned} \tag{A.37}$$

$$\begin{aligned} c = & \frac{h_i \left(h_{i-1} + h_i\right) \left(2 \left(h_i + h_{i+1}\right) \left(h_i + 2h_{i+1}\right) + h_{i-1} \left(2h_i + 3h_{i+1}\right)\right)}{\left(h_i + h_{i+1}\right)^2 \left(h_{i-1} + h_i + h_{i+1}\right)^2} \\ & + \frac{h_{i+1}}{h_{i+1} + h_{i+2}} - \frac{\left(h_{i-1} + h_i\right) h_{i+1}^3}{h_{i-1} \left(h_i + h_{i+1}\right)^2 \left(h_i + h_{i+1} + h_{i+2}\right)} \\ & + \frac{h_i h_{i+1}^3}{h_{i-1} \left(h_{i-1} + h_i + h_{i+1}\right)^2 \left(h_{i-1} + h_i + h_{i+1} + h_{i+2}\right)} \end{aligned} \tag{A.38}$$

$$d = \frac{h_i \left(h_{i-1} + h_i\right) h_{i+1}^2}{h_{i+2} \left(h_{i+1} + h_{i+2}\right) \left(h_i + h_{i+1} + h_{i+2}\right) \left(h_{i-1} + h_i + h_{i+1} + h_{i+2}\right)} \tag{A.39}$$

A.3.4 Coefficients of the IU22EC4 scheme

$\gamma = 0$ and the system becomes:

$$\begin{pmatrix} 1 & -1 & -1 & -1 & -1 \\ -h_i & h_i + \frac{h_{i-1}}{2} & \frac{h_i}{2} & -\frac{h_{i+1}}{2} & -h_{i+1} - \frac{h_{i+2}}{2} \\ h_i^2 & \frac{1}{3h_{i-1}} \left(h_i^3 - \left(h_i + h_{i-1}\right)^3\right) & -\frac{h_i^2}{3} & -\frac{h_{i+1}^2}{3} & \frac{1}{3h_{i+2}} \left(h_{i+1}^3 - \left(h_{i+1} + h_{i+2}\right)^3\right) \\ -h_i^3 & \frac{1}{4h_{i-1}} \left(\left(h_i + h_{i-1}\right)^4 - h_i^4\right) & \frac{h_i^3}{4} & -\frac{h_{i+1}^3}{4} & \frac{1}{4h_{i+2}} \left(h_{i+1}^4 - \left(h_{i+1} + h_{i+2}\right)^4\right) \\ h_i^4 & \frac{1}{5h_{i-1}} \left(h_i^5 - \left(h_i + h_{i-1}\right)^5\right) & -\frac{h_i^4}{5} & -\frac{h_{i+1}^4}{5} & \frac{1}{5h_{i+2}} \left(h_{i+1}^5 - \left(h_{i+1} + h_{i+2}\right)^5\right) \end{pmatrix} \tag{A.40}$$

The coefficients are:

$$\beta = \frac{\left(h_{i-1} + h_i\right) h_{i+1} \left(h_{i+1} + h_{i+2}\right)}{h_{i-1} \left(h_i + h_{i+1}\right) \left(h_i + h_{i+1} + h_{i+2}\right)} \tag{A.41}$$

$$a = \frac{h_i^2 h_{i+1} \left(h_{i+1} + h_{i+2}\right)}{h_{i-1} \left(h_{i-1} + h_i\right) \left(h_{i-1} + h_i + h_{i+1}\right) \left(h_{i-1} + h_i + h_{i+1} + h_{i+2}\right)} \tag{A.42}$$

$$
\begin{aligned}
b =& h_{i+1}\left(\frac{2h_{i-1}\left(2h_i+h_{i+1}\right)^2+h_{i-1}^2\left(3h_i+2h_{i+1}\right)+h_i\left(6h_i^2+8h_ih_{i+1}+3h_{i+1}^2\right)}{\left(h_{i-1}+h_i\right)\left(h_i+h_{i+1}\right)^2\left(h_{i-1}+h_i+h_{i+1}\right)}\right.\\
&-\frac{h_i^2\left(h_{i-1}+h_i\right)}{h_{i-1}\left(h_i+h_{i+1}\right)\left(h_i+h_{i+1}+h_{i+2}\right)^2}\\
&+\frac{h_i\left(h_{i-1}+h_i\right)\left(h_i\left(h_i+h_{i+1}\right)-h_{i-1}\left(2h_i+h_{i+1}\right)\right)}{h_{i-1}^2\left(h_i+h_{i+1}\right)^2\left(h_i+h_{i+1}+h_{i+2}\right)}\\
&\left.-\frac{h_i^3}{h_{i-1}^2\left(h_{i-1}+h_i+h_{i+1}\right)\left(h_{i-1}+h_i+h_{i+1}+h_{i+2}\right)}\right)
\end{aligned}
\tag{A.43}
$$

$$
\begin{aligned}
c =& \frac{h_i^2\left(h_{i-1}+h_i\right)}{\left(h_i+h_{i+1}\right)^2\left(h_{i-1}+h_i+h_{i+1}\right)\left(h_{i+1}+h_{i+2}\right)\left(h_i+h_{i+1}+h_{i+2}\right)^2\left(h_{i-1}+h_i+h_{i+1}+h_{i+2}\right)}\\
&\left(h_{i+1}\left(h_i+h_{i+1}\right)^2\left(2h_i+5h_{i+1}\right)+\left(h_i+h_{i+1}\right)\left(h_i^2+8h_ih_{i+1}+10h_{i+1}^2\right)h_{i+2}\right.\\
&+\left(3h_i^2+12h_ih_{i+1}+10h_{i+1}^2\right)h_{i+2}^2+\left(3h_i+5h_{i+1}\right)h_{i+2}^3+h_{i+2}^4\\
&\left.+h_{i-1}\left(h_i+2h_{i+1}+h_{i+2}\right)\left(2h_{i+1}\left(h_i+h_{i+1}\right)+\left(h_i+2h_{i+1}\right)h_{i+2}+h_{i+2}^2\right)\right)
\end{aligned}
\tag{A.44}
$$

$$
d = -\frac{h_i^2\left(h_{i-1}+h_i\right)h_{i+1}}{\left(h_{i+1}+h_{i+2}\right)\left(h_i+h_{i+1}+h_{i+2}\right)^2\left(h_{i-1}+h_i+h_{i+1}+h_{i+2}\right)} \tag{A.45}
$$

Appendix B

TDMA

The present schemes requires to solve the system $A\tilde{u} = B\bar{u}$ described in (3.1). Instead of using $i/2$ indices for the interfaces, they are numbered from 1 to n so that an interior node i (numbered from 1 to $n-1$) has left interface i and right interfaces $i+1$. For simplicity, $d = B\bar{u}$ is set.

The TDMA (Tri-Diagonal Matrix Algorithm) has 2 steps:

1. the first consists to forward sweep the matrix in order to obtain a matrix with two diagonals, and so that the main diagonal is only made of 1. The algorithm starts by setting:

$$P_1 = \gamma_1, \quad Q_1 = d_1 \tag{B.1}$$

Then β are removed in the subdiagonal and the main diagonal is made of 1 by computing:

$$P_i = \frac{\gamma_i}{1 - \beta_i P_{i-1}}, \quad Q_i = \frac{d_i - \beta_i Q_{i-1}}{1 - \beta_i P_{i-1}}, \quad i = 1 \dots n-1 \tag{B.2}$$

For the last interface, one gets:

$$Q_n = \frac{d_n - \beta_n Q_{n-1}}{1 - \beta_n P_{n-1}} \tag{B.3}$$

And (3.1) becomes:

$$\begin{pmatrix} 1 & P_1 & & & \\ & 1 & P_2 & & \\ & & \ddots & \ddots & \\ & & & 1 & P_{n-1} \\ & & & & 1 \end{pmatrix} \begin{pmatrix} \tilde{u}_1 \\ u_2 \\ \vdots \\ \tilde{u}_{n-1} \\ \tilde{u}_n \end{pmatrix} = \begin{pmatrix} Q_1 \\ Q_2 \\ \vdots \\ Q_{n-1} \\ Q_n \end{pmatrix} \tag{B.4}$$

2. the second step is to backward sweep:

$$\begin{cases} \tilde{u}_n = Q_n \\ \tilde{u}_i = Q_i - P_i \tilde{u}_{i+1}, \quad i = n-1, \dots, 1 \end{cases} \tag{B.5}$$

These steps are described in algorithm B.1. This basic algorithm can easily be improved both in term of memory and processor time: in line 6 of B.1,an array can be avoided by saving Q_i in d_i since the rhs d_i is not needed in the next iterations. By saving d in $\tilde{u}$ and

Algorithm B.1 TDMA

Require: the implicit coefficients β, γ, the rhs $d = B\bar{u}$

1: *forward sweep*
2: $P_1 = \gamma_1$
3: $Q_1 = d_1$
4: **for** $i = 2$ to $n - 1$ **do**
5: $\quad P_i = \dfrac{\gamma_i}{1 - \beta_i P_{i-1}}$
6: $\quad Q_i = \dfrac{d_i - \beta_i Q_{i-1}}{1 - \beta_i P_{i-1}}$
7: **end for**
8: *backward sweep*
9: $\tilde{u}_n = \dfrac{d_n - \beta_n Q_{n-1}}{1 - \beta_n P_{n-1}}$
10: **for** $i = n - 1$ to 1 **do**
11: $\quad \tilde{u}_i = Q_i - P_i \tilde{u}_{i+1}$
12: **end for**

Ensure: $\tilde{u}$ the interpolated value of u at the interfaces

then Q in $\tilde{u}$(in line 11, $\tilde{u}_i$ only requires Q_i), d and Q are not stored explicitly in a new array.

The vector P only depends on the coefficients β and γ and so is constant through all the computations (if the mesh is not moving). So it can be computed only once. And the coefficients γ are not directly used in the further computations, so P can be stored in γ saving one more array. And finally, the denominator of P and Q is the same and constant, so it can also be computed only once. This require one additional array to store it. The final algorithm is B.2.

Algorithm B.2 Memory-saving TDMA

Require: the coefficients $\beta, \gamma, a, b, c, d$, $\bar{u}$ the cell-average value of u

> Pre-Processing (done only once)
for $i = 2$ to n **do**
$\quad m_i = 1 - \beta_i \gamma_{i-1}$
$\quad \gamma_i = \dfrac{\gamma_i}{m_i}$
end for
> then for each system to solve
compute the rhs (saved in $\tilde{u}$)
for $i = 1$ to n **do**
$\quad \tilde{u}_i = a_i \bar{u}_{i-2} + b_i \bar{u}_{i-1} + c_i \bar{u}_i + d_i \bar{u}_{i+1}$
end for
forward sweep (Q is also saved in $\tilde{u}$, replacing the rhs)
for $i = 2$ to n **do**
$\quad \tilde{u}_i = \dfrac{1}{m_i}(\tilde{u}_i - \beta_i \tilde{u}_{i-1})$
end for
backward sweep
for $i = n - 1$ to 1 **do**
$\quad \tilde{u}_i = \tilde{u}_i - \gamma_i \tilde{u}_{i+1}$
end for

Ensure: $\tilde{u}$ the interpolated value of u at the interfaces

Appendix C

Modified Runge-Kutta Algorithm

The following algorithm is described in [5]. It is used to solve the ordinary differential equation coming from the semi-descetized equations:

$$\frac{\partial \bar{f}_i}{\partial t} = -\frac{v}{h_i}(f_{i+1/2} - f_{i-1/2}) \tag{C.1}$$

The Runge-Kutta formula takes $\bar{f}_i^n$ and t_n and calculates an approximation for $\bar{f}_i^{n+1}$ at a brief time later, $t_n + \Delta t$. It uses a weighted average of approximated values of the right hand-side at several times within the interval $[t_n, t_n + \Delta t]$. The algorithm is C.1.

Algorithm C.1 Modified Runge-Kutta of order p (RKp)

Require: $\bar{f}$ known at times t_n, a time step Δt
  $\bar{f}^{(0)} = \bar{f}_i^n$
  for $k = 1$ to p **do**
    Compute the flux $\tilde{f}^{(k-1)}$ using the algorithm B.2
    $C_k = \frac{1}{p-k+1}$
    $\bar{f}_i^{(k)} = \bar{f}_i^n - C_k \Delta t \frac{v}{h_i}\left(\tilde{f}_{i+1/2}^{(k-1)} - \tilde{f}_{i-1/2}^{(k-1)}\right), \forall i$
  end for
  $\bar{f}^{n+1} = \bar{f}^{(p)}$
Ensure: $\bar{f}^{n+1}$ known at time $t_n + \Delta t$

Bibliography

[1] Tim BROECKHOVEN, Sergey SMIRNOV, Jan RAMBOER, and Chris LACOR. Finite volume formulation of compact upwind and central schemes with artificial selective damping. *Journal of Scientific Computing*, 21(3):341–367, December 2004.

[2] Fotini Katopodes CHOW and Parviz MOIN. A further study of numerical errors in large-eddy simulations. *Journal of Computational Physics*, 184:366–380, 2003.

[3] A. CLEARY and J. DONGARRA. Implementation in scalapack of divide-and-conquer algorithms for banded and tridiagonal linear systems.

[4] L. GAMET, F. DUCROS, F. NICOUD, and T. POINSOT. Compact finite-difference schemes on non-uniform meshes. application to direct numerical simulation of compressible flow. *International Journal for Numerical methods in Fluids*, 29:159–191, 1999.

[5] Klaus A. HOFFMANN and Steve T. CHIANG. *Computational Fluid Dynamics*, volume I. Engineering Education System, 4th edition, August 2000.

[6] Marcelo H. KOBAYASHI. On a class of padé finite volume methods. *Journal of Computational Physics*, 156(1):137–180, 1999.

[7] Chris LACOR, Sergey SMIRNOV, and Martine BAELMANS. A finite volume formulation of compact central schemes on arbitrary structured grids. *Journal of Computational Physics*, 198(2):535–566, 2004.

[8] Sanjiva K. LELE. Compact finite difference schemes with spectral-like resolution. *Journal of Computational Physics*, 103:16–42, 1992.

[9] B. P. LEONARD. A stable and accurate convective modelling procedure based on quadratic upstream interpolation. *Computational methods applied in mechanical engineering*, 19:59–98, 1979.

[10] T.K. SENGUPTA, G. GANERIWAL, and S. DE. Analysis of central and upwind compact schemes. *Journal of Computational Physics*, 192:677–694, 2003.

[11] Robert VICHNEVETSKY and John B. BOWLES. *Fourier Analysis of Numerical Approximations of Hyperbolic Equations*. SIAM, 1982.