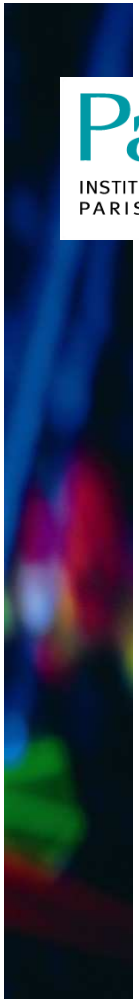
A new dynamic code architecture for CFD computations : application to the development of an overset-grid compact high-order solver for compressible aerodynamics

Pierre-Yves Outtier



ParisTech
INSTITUT DES SCIENCES ET TECHNOLOGIES
PARIS INSTITUTE OF TECHNOLOGY

ARTS ET MÉTIERS ParisTech

2014-ENAM-0029

École doctorale n° 432 : Science des Métiers de l'ingénieur

Doctorat ParisTech

T H È S E

pour obtenir le grade de docteur délivré par

l'École Nationale Supérieure d'Arts et Métiers

Spécialité " Mécanique "

présentée et soutenue publiquement par

Pierre-Yves OUTTIER

le 30 septembre 2014

A new dynamic code architecture for CFD computations: application to the development of an overset-grid compact high-order solver for compressible aerodynamics

Directeur de thèse : **Paola CINNELLA**

Jury

M. Christophe CORRE, Professeur, LMFA, Ecole Centrale de Lyon	Président
M. Éric LAMBALLAIS, Professeur, Institut P', CNRS - Université de Poitiers - ENSMA	Rapporteur
M. Christian TENAUD, Professeur, LIMSI, Université d'Orsay	Rapporteur
M. Christophe BENOIT, Ingénieur de Recherche, DSNA, ONERA	Examinateur
Mme Paola CINNELLA, Professeur, DynFluid, Università del Salento	Examinateur
M. Alain LERAT, Professeur émérite, DynFluid, Arts et Métiers ParisTech	Examinateur
M. Bertrand MICHEL, Ingénieur de Recherche, DSNA, ONERA	Examinateur
M. Michel VISONNEAU, DR CNRS, LHEA, Ecole Centrale de Nantes	Examinateur

T H È S E

Arts et Métiers ParisTech - Centre de Paris
DynFluid - Laboratoire de Dynamique des Fluides

"Men who have an excessive faith in their theories or in their ideas are not only poorly disposed to make discoveries but they also make very poor observations.

Claude Bernard (1813-1878),
Introduction à la Médecine expérimentale (Paris, 1865)

Remerciements

Je tiens en premier lieu à remercier ma directrice de thèse, le professeur Paola Cinnella, de m'avoir fourni l'opportunité de réaliser cette thèse. Depuis la recherche de financement jusqu'à la finalisation de mes travaux, je lui dois ma reconnaissance. J'ai apprécié tant la liberté dans l'orientation de mon travail que l'aide qu'elle m'a apporté. Le point le plus important pour moi est la collaboration qu'elle a permi entre le docteur Content et moi-même tout au long de mes années de thèse. J'ai enfin l'opportunité de remercier le docteur Content d'avoir partagé avec moi cette aventure de recherche. Que ce soient les cafés, les dealines, les discussions scientifiques ou musicales, les lignes de code, les equations, les latex, ... je suis heureux d'avoir fait équipe avec toi Cédric. Je tiens également à remercier le professeur Lerat de m'avoir accueilli moi et mes questions sur le métier de chercheur, de m'avoir dirigé vers le professeur Cinnella pour concrétiser mon projet de thèse, et tout simplement pour son accueil toujours sympathique. Enfin, je remercie tous les membres du laboratoire, permanent ou doctorant. J'ai passé de très bonnes années qui m'auront assurément permis de grandir.

Par ailleurs, mon meilleur allié pendant ces années a été Léontine. Mille mercis de m'avoir soutenu et accompagné dans ce projet, de m'avoir laissé m'épanouir dans mon travail, d'avoir aménagé une vie familiale, séparée de la vie professionnelle, tout simplement d'avoir été présente à mes côtés. Merci à mes parents, mon fère, ma soeur d'être toujours attentifs et attentionnés. Enfin, je remercie mes amis d'être présents et d'avoir feint de comprendre mes propos mal vulgarisés tentant d'expliquer la teneur de mes travaux.

Contents

Introduction

Context and motivation

Computational Fluid Dynamics (CFD) has become a major tool in aeronautical industry for the design of aircraft components, thanks to its increased predictive capabilities, computational affordability and the possibility of drastically reducing the use of costly experimental investigations. The vision of the Advisory Council for Aviation Research and Innovation in Europe (ACARE [1]) for the future directions of the European aeronautical industry up to 2020 [4] and 2050 [5] introduces extremely ambitious goals on energy consumption, pollutant emissions and perceived aircraft noise, namely, a reduction of carbon dioxide emissions by 75%, NOx emissions by 90%, and noise by 65% before 2050. The achievement of these goals induces in turn heavy demands on future product performance and requires step changes in aircraft technology, which cannot be achieved without introducing new layout principles and investigating unconventional layouts. The associated research activities can be fostered by an intensive use of numerical simulation, which is considered by ACARE as a key enable for future aircraft design.

Nowadays, several industrial CFD codes are routinely employed by European industries for aircraft design like, e.g. ONERA's code *elsA* [8], DRL's code TAU [3], Numeca's code FineTurbo [6], and many other. In fact, CFD can be considered as a mature tool for configurations at their design point in the flight envelop. However, a growing use of simulation capabilities is required in the future, especially for investigating configurations at off design conditions or highly loaded designs involving complicated flow features already at the nominal point. For this purpose, major developments are required to improve the predictive capabilities of CFD codes for taking into account complex flow phenomena like, e.g., those arising in unsteady and/or separated flows among other.

At the present stage of development, industrial codes are essentially based on robust nominally second-order methods, like the Jameson scheme [97] or Roe's scheme [151] with MUSCL extrapolation [181]. In practice, the effective order of accuracy is more often in between 1^{st} and 2^{nd}, or even lower, due to the use of irregular meshes. As a consequence, the application of industrial codes to the resolution of fine flow features requires the use of very fine meshes to compensate numerical errors introduced by the approximation scheme and not to improve the physical representation of flow phenomena. This is even more crucial when advanced simulation strategies like Large Eddy Simulation (LES) are used, because of the strong interactions of numerical errors with physical modeling.

For all these reasons, a demand exists for a new generation of industrial codes of increased accuracy. High-order numerical schemes are usually restricted to research applications, involving highly complex physical phenomena but simple geometries, and regular Cartesian or lowly deformed meshes. In order to target the transfer of knowledge on the use of high-order schemes for CFD computations from research laboratories toward more applied research in the industry, the European project IDIHOM [10] proposed to gather efforts of European partners among which

ONERA, DLR, Dassault, Airbus Defense and Space (ex Cassidian) and SNECMA as industrial supports. The objective of the project was to assess and improve the high-order numerical methods used for several years in an academic context and make them operable on more realistic flows involving complex industrial geometries.

The DynFluid Laboratory contributed to IDIHOM by further developing a family of high-order residual-based compact (RBC) schemes initially introduced by Lerat and Corre [112], with a twofold objective: on the one hand, to extend RBC schemes to unsteady flows by ensuring a stable and robust behavior while keeping optimal resolvability properties; on the other one, to improve the applicability of these schemes to complex geometrical configurations while preserving as much as possible their high accuracy. The first point was thoroughly investigated in [83]. Here, we focus more specifically on the second point.

In the quest for a new generation of high-accurate flow solvers, we were led to address the more general question of how to design a CFD code architecture that:

— Can take into account a variety of possibly geometrically complex configurations;
— Remains simple and modular enough to facilitate the introduction and testing of new ideas (numerical methods, models) with a minimal development effort, including "on the fly" tests;
— Use high-order numerical discretizations and advanced physical models.

This required some innovative choices in terms of programming languages, data structure and storage, and code architecture, which go beyond the mere development of a specific family of numerical schemes.

In the following of this introductory chapter we first briefly review domain discretization strategies well suited for complex geometries, and in particular overset grid strategies. Afterwards, we focus on high-order methods that can be seen as potential candidates for industrial simulations, with focus on residual based compact schemes. Subsequently, we focus on the design of suitable code architectures for CFD simulations. Finally, we describe the structure of the present dissertation.

CFD in complex geometries

In order to handle complex geometries, industrial codes are using either unstructured or multi-block structured grids. Many references have treated grid generation, see for instance [76, 177]. The choice of the meshing strategy, in turn, has a deep impact on the underlying discretization algorithm.

Unstructured algorithms were initially confined to the finite element community, and subsequently spread out to the finite volume community. Their main advantages are a greater flexibility in terms of representation of complex geometries, and the rapidity of the grid generation process thanks to the use of advancing-front methods (as Delaunay techniques [80, 76]). Moreover, unstructured meshes generally enable a discretization of the whole computational domain with a single grid block, which makes parallel balancing procedures easier and leads to superior parallel performance.

However, unstructured meshes induce an extra cost for the numerical solvers. One origin of the additional cost is related to the calculation and memory storage of mesh connectivity information. Even if suitable techniques for sorting data in memory exist, memory accesses are never as fast as quad elements organized as in a structured mesh. One other origin of the extra cost is that Delaunay techniques provide near isotropic meshes in boundary layers, leading to a larger number of elements in the computational domain. This could be avoided by using first

extrusion of a wall surface mesh before meshing the remainder of the domain or by using a structured grid layer close to the walls.

Among unstructured algorithms, we recall k-exact schemes, initially developed by Barth and Frederickson [25]. These belong to a general class of Godunov-type finite volume methods. They make use of a polynomial reconstruction of the solution at order k within each cell, plus high-order quadrature formulae for approximating flux integrals along cell faces. Other examples are given by continuous residual distribution schemes [17], and Discontinuous Galerkin schemes, discussed in more detail later on.

Structured grids, on the other hand, enable the use of structured approximation schemes like, e.g., finite difference schemes, but also structured finite volume algorithms, which take advantage of the prior knowledge of the neighboring points to simplify the algorithmic implementation, and reduce memory and CPU requirements. Moreover, structured grid methods are generally more accurate and efficient than unstructured ones for the representation of flow features aligned with mesh directions. Many structured-grid algorithms have been proposed though the years, mainly in a Finite-Difference framework even if work has also been done in a Finite-Volume framework. Some of them, and precisely the high-order ones, are reviewed later. Structured-grid methods are generally restricted to moderately complex geometries, because of the difficulty or impossibility of discretizing very complex computational domains with a single-block structured grid.

To handle complex geometries of interest for industrial applications by using structured grids, one possible strategy is to use multi-block methods based on a set of grids (or composite grid or overlapping grids) where multiple grids are allowed to overlap. Overlapping techniques consist in resolving the system of partial equations on multiple simple meshes which overlap. The values at the boundaries are interpolated from the interior of an overlapping neighboring mesh. The overlapping method has been introduced by Benek *et al.* [31], extended by Steger *et al.* [166] and Benek *et al.* [28]. Even if conservative interpolation techniques have been studied [35, 45, 185, 160, 75], it is commonly accepted to use non-conservative interpolations, which is acceptable at least for external aerodynamics. Indeed, obtaining high-order accuracy while maintaining the conservation property in multi-dimensional applications is not a straightforward task. Some recent work concerning high-order and conservative patched grid (i.e. multiblock grids with non conformal joins between interfaces common to two blocks) interface treatments can be found in [125].

The overset grid method has been recently used in conjunction with high-order schemes using large discretization stencils (for aeroacoustics, turbulent flow simulations, ...) for example in [192, 62, 64, 174, 70, 65, 46]. They have also been used in helicopter applications as the vortex-blade interaction studied by Saunier [158] along with mesh adaptation. In the literature, it is possible to find mesh generators and solvers dedicated to the overlapping strategy, let us cite among others Cassiopee [2, 137, 33] and Overture [90]. Concerning the limitations or current points of research, the efficient parallel treatment of overset grid approaches requires to be enhanced to enable its use for large-scale computations (e.g. large eddy simulations) especially when it involves moving grids and/or grid adaptation.

In this work, we develop an overset grid method in conjunction with a family of high-order residual-based compact schemes. The state-of-the-art of high-order methods for complex configurations is briefly reviewed in the following.

State-of-the-art of high-order methods

In the following, we briefly review the state-of-the-art of high-fidelity numerical methods for compressible aerodynamics, by distinguishing finite difference (FD), finite volume (FV) and finite element (FE) methods.

FD methods Very simple to implement, FD methods lead to very high computational efficiency, at least as far as simple geometries are concerned.

Even though some ideas may be traced back further, the starting point for FD schemes is the fundamental theoretical paper by Courant, Friedrichs and Lewy (1928)[59] on the solution of problems of mathematical physics by means of finite differences. First steps concerning the space discretization schemes in FD are due to Hildebrand (1956), Kopal (1961) or Collatz (1966) among others.

For FD schemes, high-accuracy is generally achieved by increasing the size of the discretization stencil. Moreover, most FD schemes are developed by treating derivatives in each space direction separately. This kind of scheme is called hereafter a Directional Non Compact (DNC) scheme.

In the field of DNC schemes, let us mention the family of schemes described for the first time in [113] and formally identical, in 1D, to high-order MUSCL FV schemes based on flux extrapolation. These schemes are called FE-MUSCL in this thesis, and can be seen as a generalization of the Roe scheme [151] to high-order accuracy. In the context of aeroacoustic simulations, Tam and Webb [176] and then Bogey and Bailly [37] proposed Dispersion-Relation-Preserving (DRP) schemes, which are optimized in the wavespace to maximize their resolvability, along with explicit filtering techniques [176, 37] to remove grid-to-grid oscillations.

Lele [111] contributed to develop high-order approximations of space derivatives based on the use of generalized Padé formulae. These compact schemes provide lower dissipation and dispersion errors than the non-compact approximations of the same order of accuracy but lead to the inversion of a mass matrix at each time step. Initially proposed by Abarbanel and Kumar [15], compact implicit schemes can be modified to move the implicitness off the space derivative onto the time derivative approximation. Residual-Based Compact (RBC) schemes use this procedure to construct a scheme based on the derivatives of the residual. RBC schemes were initially introduced by Lerat and Corre [112, 114], who designed a third-order accurate scheme for steady state compressible Euler equations. The dissipation of this scheme is also built on the residual and is upwinded through a characteristic time step [95]. The scheme can be coupled with explicit time integration method and is then conditionally stable, but is presented with a matrix-free implicit treatment. It has been extended to unsteady flow [57] in a dual time step framework. The general design for any odd orders, up to order seven, was described in [56] for inviscid flows and in [58] for advection/diffusion problems. The dissipation properties of the schemes have been studied in [115], leading to modifications of the discretization coefficients that ensure a stable and robust behavior for any flow conditions, and the spectral properties in [84]. The design principles and numerical properties of RBC schemes are discussed in more detail in Chapter 1.

FV methods Intrinsically conservative and, consequently, well suited for treating flows with discontinuities, the finite volume (FV) method is probably the most popular method for compressible simulations.

A large majority of industrial codes are based second-order FV schemes employed for the solution of Reynolds-Averaged Navier-Stokes equations (RANS). The first concepts of numerical

methods for compressible flows are found in the pioneering work of Godunov [81] and developments have been led at NASA Ames in the early seventies [123, 124, 150]. FV schemes are suitable for use on both structured and unstructured meshes and are classified according to the location of discrete unknowns (cell centers, cell vertices, ...). For lower order FV methods (first or second accuracy at most), the schemes differ one to another from the definition of the numerical flux at the interfaces from the unknowns. For higher-order methods (third-order and more), two strategies appear. On the one hand, the unknowns can represent cell-averaged properties. It avoids introducing quadrature formulae to evaluate volume and surface integrals, but require an implicit deconvolution step to relate cell-averaged values to face values [78, 105, 135, 107]: this procedure is restricted to uniform Cartesian meshes, and its application to non-uniform grids leads to a loss of accuracy. A more general procedure valid for non-uniform structured grids has been proposed in [107], with focus on compact schemes: thanks to this, it was possible to construct a compact scheme of fourth-order accuracy on Cartesian meshes and 3rd-order accuracy on distorted meshes. The implicit deconvolution step requires the solution of a linear system for each cell face, which may increase the computational cost considerably. In [87], a high-order piecewise polynomial reconstruction is used to increase the accuracy of MUSCL schemes on arbitrary unstructured grids. Face values are computed from cell-averaged quantities via a recursive correction technique involving approximations of the solution derivatives. Nevertheless, this approach requires to store additional connectivity information about the reconstruction stencil and to store or recompute at each time step reconstruction matrices. On the other hand, a second approach of FV schemes consists in considering pointwise values of the field variables in the control volume (e.g. values at cell centroids). When FV schemes on structured meshes are considered, and in an effort to minimize storage and computational cost requirements, a point-wise cell-centered FV formulation is a reasonable choice, following the general framework presented in [147, 146].

RBC schemes, initially developed in a FD framework, have been reformulated as FV schemes by using a straightforward extension of the numerical fluxes or by using a high-order FV extension to irregular structured meshes [89, 85]. The study of the dissipative properties of this FV version of RBC schemes has been lead by Grimich [83, 85]. Finally, an unstructured third-order FV RBC scheme has been developed by Corre and Du [55].

Finite elements Numerical frameworks based on Finite Elements are generally well established from a mathematical point of view. Based on the weak form of the conservation laws, the ability to achieve high-order accuracy rely on the choice of a compact support test function. The FE methods are well fitted to handle complex geometry using unstructured grids, but generally lead to a purely centered, non dissipative, discretization, unsuitable for fluid mechanics problems. To make FE method applicable to fluid flows, the addition of stabilization terms is then required.

In recent years, considerable attention has been received by Discontinuous Galerkin (DG) methods, Spectral Elements and Continuous Residual-Distribution methods.

DG methods combine features of the finite volume method and a discretization with standard (conforming) finite elements [52]. They are well suited for convection-diffusion problems, including the numerical simulation of compressible flows. The order of accuracy is chosen by the definition of the underlying polynomial basis. The solution is discontinuous at cell interfaces, where a Riemann solver is applied. DG can be used in conjunction with explicit temporal schemes because the mass matrix is block-diagonal and then easy to invert and can be stored. Based on the FE framework on unstructured grids, the method is quite costly because of the necessity of storing shape functions, but it is computationally compact and then suitable for

massive parallelization. The use of DG for RANS simulations have been discussed in Ref. [26] and for 3D turbulent flows in Ref. [106]. Yet, contact discontinuity and shock capturing ability of higher-order DG methods requires further research.

Concerning Spectral Element methods, various formulations exist, among which Spectral Volume (SV) and Spectral Difference (SD) methods. Similarly to DG methods, they are based on the use of element-wise discontinuous polynomials. Solution reconstruction is based on the introduction of inner degrees of freedom within each cell and thus it does not require information from neighboring cells. We refer to [170, 187, 189, 188, 171, 121, 186] for details on SV method. Spectral-Difference (SD) methods [120] are based on the FD framework with reconstruction within the cell on spectral degrees of freedom with use of staggered-grid method. They also use element-wise discontinuous reconstructions. The SD method is presented as easier to implement than the DG and SV methods, especially for high-order curved boundaries, because it does not involve surface or volume integrals.

Among continuous FE methods, we cite Residual-Distribution (RD) methods, the framework of which is based on both finite elements and finite volumes. RD schemes represent a very interesting alternative to DG schemes. While computationally compact and probably more flexible, DG schemes suffer from the serious drawback of a very fast growth of the number of degrees of freedom (DoFs) with the cell polynomial degree. In RD schemes the formulation remains local, as in DG, but the number of DoFs growths less quickly because the solution is assumed to be continuous. Another difference between RD and DG schemes is that, to date at least, the non oscillatory properties of the RD scheme in the case of discontinuous solutions are probably better understood than in the DG approach. Inspired from papers by Roe [152] and Ni [128], RD schemes rely on many contributions from different groups that enriched the framework (see [17] and references cited therein).

As a final remark on FE methods, since current simulation codes in the aeronautic industry rely on finite volume methods, the use of FE technique requires the development of completely new codes, in terms of implementation and parallelization techniques, since present codes are not flexible enough to allow the use of different discretization approach within the same architecture.

CFD code architectures

The main purpose of a CFD (Computational Fluid Dynamics) code is to gather numerical methods - for geometry processing, computation of physical properties, space and time discretization - in one entity and to make them accessible and inter-operable. Numerical codes are either dedicated to a single application or general purpose. In the first case, they are tailored and optimized for a single flow configuration and can hardly be applied to another one without significant modification; on the other hand, multi-purpose codes usually allow for solving any flow of interest, but require managing complex data structure and are often not easy to manipulate by researchers. Several open source codes mainly based on unstructured meshes are available (FreeFEM [11], OpenFoam[98, 12], Coolfluid [109, 142], Nek5000 [134], Code_Saturne [21]). These numerical codes are written mainly using compiled programming languages such as FORTRAN[18], C or C++[169]. The code *Kernel* is more or less static after its compilation and in the case of an object-oriented language, extensions are made via inheritance from generic classes.

Several CFD codes started using more flexible programming languages like the interpreted Python [9, 108] language. Python may be used to glue together portions of programs[191]. It is also used in ONERA's solver elsA[8], DLR's solver Tau [3] or, more recently, in Stanford University open source code SU2[133] to create a user-friendly interface to the underlying C++

structure, or by the US Army helicopter solver HELIOS [168] to create a software integration framework.

In this work, we develop a new CFD code, named DynHoLab (Dynamic High-order Laboratory), combining the interpreted, object-oriented, Python language, and the compiled, fast Fortran language. This enables fast development in a flexible, general and modular Python environment along with high CPU performance characteristic of Fortran language.

Scope of the thesis

This work is the opportunity for in-depth thinking on the architecture of CFD codes. The challenge that arises when dealing with non-trivial industrial applications lies in the capacity of research partners to handle complex computer programs with few human resources. A solution mixing Python and Fortran languages is proposed with details on the concepts at the basis of the code architecture, as well as study of the parallel performances. The numerical methods are validated on test-cases of increasing complexity, demonstrating at the same time the variety of physics and geometry currently achievable with DynHoLab.

Based on the computational framework designed, this work attempts to provide a way to handle complex geometries while increasing the order of accuracy of the numerical methods, with the underlying question of ensuring high overall efficiency of the numerical simulation. In order to apply high-order RBC schemes to complex geometries, the present strategy consists in a multi-domain implementation on overlapping structured meshes.

Structure of the report

The thesis is organized as follows:

— Chapter 1 reviews the conservation laws, physical model and numerical methods in use in this work.

— Chapter 2 is dedicated to the development of the DynHoLab code, including the code architecture, its parallel scalability study and its current status of development.

— Chapter 3 is devoted to the validation of numerical methods implemented in DynHoLab. For this purpose, we consider a series of test cases of increasing complexity. This includes steady and unsteady linear advection, a validation of the 2-D and 3-D RANS solver and the application to the implicit LES over 2-D periodic hills.

— Finally, Chapter 4 details the development of the overlapping strategy, its validation and the application of this framework to 2-D transonic flows over single and tandem configurations of NACA0012 airfoils.

Introduction (french)

Contexte et motivation

La Mécanique des Fluides Numérique (MFN) est devenue un outil de première importance dans l'industrie aéronautique, par exemple pour la conception des éléments de voilures, et ce grâce à sa capacité prédictive, son coût devenu abordable et la possibilité de réduire drastiquement le recourt à des expérimentations coûteuses. La vision du *Advisory Council for Aviation Research and Innovation in Europe* (ACARE [1]) concernant les futures directions pour l'industrie aérodynamique jusque 2020 [4] et 2050 [5] fait état de cibles extrêmement ambitieuses concernant les consommations d'énergie, les émissions de polluants et le bruit émis par les avions. Plus précisément, ce rapport mentionne une réduction des émissions de dioxyde de carbone de 75%, des émissions de NOx de 90% et de bruit de 65% avant 2050. L'atteinte d'une telle cible requière des évolutions conséquentes des technologies dans l'aéronautique, ce qui ne se fera pas sans innovation, qui plus est de rupture. La recherche associée à cet effort industriel fera assurément une utilisation intensive de la simulation numérique, qui est d'ailleurs considérée par ACARE comme une clef pour le développement des avions du future.

Les codes de MFN sont utilisés quotidiennement par l'industrie Européenne pour la conception des avions. Parmi ces solveurs, citons le code *elsA* de l'ONERA [8], le code TAU du DRL [3], le code FineTurbo de Numeca [6], et bien d'autres. On peut probablement considérer que la MFN est un outil mature pour son utilisation dans des configurations proches du point dans le domaine de vol pour lequel les méthodes de MFN ont été conçues. Cependant, une demande croissante des capacités de simulation est attendue dans les années futures, notamment pour l'étude de configurations différentes des conditions de calcul standard ou pour des conditions de forte charge mettant en œuvre des écoulements complexes, même au point nominal des méthodes de MFN. Pour répondre à ce besoin, des développements majeurs sont nécessaires pour améliorer les capacités prédictives des code de MFN pour prendre en compte des phénomènes d'écoulements complexes comme, par exemple, ceux intervenant en régime instationnaire et/ou pour des écoulements décollés, parmi tant d'autres.

Dans l'état actuel des choses, les code de calcul industriels sont essentiellement basés sur des schémas numériques robustes et nominalement du second ordre comme le schéma de Jameson [97] ou celui de Roe [151] lorsqu'il est utilisé avec une extrapolation des flux de type MUSCL [181]. En pratique, l'ordre effectif des ces schémas est plus souvent situé entre le premier et second ordre, et parfois moins, cela étant dû à l'utilisation de maillages irréguliers. Par conséquence, l'utilisation de codes industriels pour la résolution d'écoulement mettant en œuvre de petites échelles requière l'utilisation de maillage très raffinés pour compenser les erreurs numériques introduites par les schémas numériques et non pour représenter les phénomènes physiques sous-jacents. Cela devient encore plus important lorsque l'on envisage des stratégies de simulation avancées telles que la simulation des grandes échelles, ou Large Eddy Simulation (LES), cela à cause des interactions très fortes entre les erreurs numériques et la modélisation de la physique.

Pour toutes ces raisons, une demande existe pour une nouvelle génération de codes industriels d'une précision plus importante. Les schémas numérique d'ordre élevés sont habituellement réservés à des applications de recherche faisant intervenir des phénomènes physiques très complexes mais des géométries très simples et des maillages Cartésiens ou très peu déformés. Afin de favoriser le transfert des connaissances sur l'utilisation des méthodes d'ordre élevé pour le calcul en mécaniques des fluides numérique depuis les laboratoires de recherche vers la recherche appliquée dans l'industrie, le projet européen IDIHOM [10] a proposé de joindre les efforts des partenaires européens parmi lesquels l'ONERA, le DLR, Dassault et Airbus defense and space (ex Cassidian) et SNECMA comme soutiens industriels. L'objectif du projet est d'évaluer et d'améliorer les méthodes numérique d'ordre élevé utilisées depuis plusieurs années dans un contexte académique et de les rendre utilisables sur écoulements plus réaliste impliquant des géométries industrielles complexes.

Le laboratoire DynFluid a contribué au projet Européen IDIHOM en poursuivant le développement d'une famille de schémas numériques compacts d'ordre élevé basés sur le résidu (residual-based compact ou RBC) initialement introduits par Lerat et Corre [112]. La poursuite du développement de ces schémas s'est faite avec deux objectifs : d'un côté, étendre les schémas RBC aux écoulements instationnaires en assurant la stabilité et la robustesse des schémas tout en conservant des propriétés de résolvabilité optimales; d'un autre côté, améliorer l'utilisation de ces schémas en géométrie complexe tout en conservant autant que possible la précision d'ordre élevé. Le premier point a été examiné dans les détails dans la thèse de Karim Grimich [83]. Dans ce travail, on se concentre plus spécialement sur le deuxième point.

Dans la recherche d'une nouvelle génération de codes de mécanique des fluides de haute précision, nous sommes amenés à adresser la question plus générale de l'architecture d'un code de mécanique des fluides numérique aillant les propriétés suivantes :

— qu'il soit capable de géré des géométries complexes;
— qu'il reste simple et assez modulaire pour faciliter l'introduction et le test de nouvelles idées (méthodes numériques, modèles) en nécessitant un effort de développement minimal, incluant des tests à la volée;
— qu'il permette l'utilisation de discrétisations numériques d'ordre élevé et des modélisations physiques avancées.

Cette ambition requiert des choix innovants en terme de langage de programmation, de structure de données et de son stockage, de l'architecture de code, tout cela allant bien plus loin que la simple implémentation d'une famille de schémas numériques spécifiques.

Dans la suite de ce chapitre introductif, et dans un premier temps, nous passons rapidement en revue les méthodes de discrétisations spatiales bien adaptées aux géométries complexes, et en particulier les méthodes avec recouvrement de maillages. Ensuite, nous nous attardons sur les méthodes numériques d'ordre élevé qui sont des candidates potentielles pour des simulations numériques d'une complexité industrielle, avec une attention particulière pour les schémas compacts basés sur le résidu. Enfin, nous traiterons l'architecture de code adaptée au simulations numériques en mécanique des fluides. Nous terminerons en présentant la structure du présent manuscrit.

MFN en géométries complexes

De manière à traiter des géométries complexes, les codes industriels sont soit basés sur une stratégie de maillages non-structurés, soit sur des maillages multi-bloc de grilles structurées. De nombreuses références traitent la génération de maillage, on peut lire par exemple [76, 177]. Le

choix d'une méthode de maillage a, à son tour, un très grand impact sur les algorithmes de discrétisation sous-jacents.

Les algorithmes non-structurés ont été initialement confinés à la communauté de la méthode des éléments finis, et par la suite ont diffusé dans la communauté de la méthode des volumes finis. L'avantage principal de ces algorithmes est leur très grande flexibilité en terme de représentation de la géométrie complexe, et la rapidité de la génération du maillage grâce aux méthodes complètement automatiques (méthodes d'avancée de front, de Delaunay [80, 76]). De plus, les maillages non structurés permettent en général d'obtenir une discrétisation du domaine complet en une seule entité, ce qui rend l'équilibrage des calculs parallèles plus aisé et mène finalement à des performances parallèles globalement supérieures dans des configurations industrielles.

Cependant, l'utilisation de maillages non-structurés induit un coût supplémentaire pour le solveur numérique. Une origine de ce coût additionnel est reliée au calcul et au stockage de la connectivité du maillage. Même si des techniques adaptée à l'organisation des données en mémoire existent, l'accès mémoire ne peut pas être aussi rapide que pour des éléments organisés de manière structurée. Une autre origine du coût supplémentaire est la méthode de Delaunay qui produit des maillages quasi-isotrope dans les couches limites, menant à un nombre supérieur d'éléments dans le domaine de calcul. Cela peut être évité par l'utilisation dans une première phase de maillage d'algorithmes d'extrusion de surface depuis les paroi (menant à des maillages structurés étirés près des parois) avant de mailler le reste du domaine avec un algorithme générant un maillage non-structuré.

Parmi les schémas numérique adaptés aux maillages non-structuré, on rappelle les schémas k-exacts, initialement développés par Barth and Frederickson [25]. Ces schémas appartiennent à la classe générique des méthodes de volumes-finis de type Godunov. Ils font usage de reconstructions polynomiales de la solution à l'ordre k dans chaque cellule, plus des formules de quadrature d'ordre élevé pour l'approximation des intégrales de flux sur les faces des cellules. D'autres exemples de schémas adaptés aux maillages non structurés sont donnés par les schémas *continuous residual distribution* [17], et par les méthodes de Galerkin discontinues dont on discute plus en détail plus loin dans cette introduction.

D'un autre côté, les maillages structurés permettent l'utilisation de schémas numériques adaptés comme par exemple les schémas de différence finies, mais aussi les algorithmes de volumes finis structurés tirant parti de cette structure du maillage et de cette connaissance des points voisins du maillage, pour simplifier l'implémentation des algorithmes et réduire la mémoire et le temps CPU nécessaires. De plus, les méthodes basées sur des grilles structurées sont généralement plus précises et plus efficaces que celles dédiées aux maillages non-structurés pour des éléments caractéristiques des écoulements alignés avec les directions du maillage. Beaucoup d'algorithmes dédiés aux maillages structurés ont été proposés au cours des années, principalement pour la méthode de différences finies, même si un travail conséquent a été produit sur les méthodes de type volumes-finis. Quelques unes de ces méthodes et précisément des méthodes d'ordre élevé, sont passées en revue plus loin dans cette introduction. Les méthodes basées sur les grilles structurées sont généralement limitées aux géométries modérément complexes, en partie à cause des difficultés ou impossibilité pour générer des domaines de calcul complexes avec une seule grille structurée.

Pour pallier à cette difficulté et envisager l'utilisation de maillages structurés dans des configurations industrielles, une stratégie possible consiste en l'emploie de maillages multi-bloc basés sur un ensemble de grilles structurées que l'on autorise à se chevaucher. Les techniques de grilles recouvrantes consistent à la résolution du système d'équations aux dérivées partielles sur une multitude de grilles structurées simples qui se recouvrent partiellement. Les valeurs aux frontières des grilles sont interpolées depuis l'intérieur des grilles voisines. La méthode

de grilles recouvrante a été introduite par Benek *et al.* [31], étendue par Steger *et al.* [166] et ensuite Benek *et al.* [28]. Même sur les techniques d'interpolation conservative ont été étudiées [35, 45, 185, 160, 75], il est communément admis d'utiliser des interpolation non-conservatives, qui sont acceptables au moins pour l'aérodynamique externe. En effet, obtenir l'ordre élevé tout en maintenant la propriété de conservation dans un cadre multidimensionnel n'est pas une tache évidente. Des travaux récents sur l'ordre élevé et la conservativité dans les raccords de maillages non-conformes peuvent être trouvés dans [125].

La méthode de maillages recouvrants a été récemment utilisée en conjonction avec des schémas d'ordre élevé ayant des supports de discrétisation large (pour l'aéroacoustique, pour des écoulements turbulents, ...) par exemple dans [192, 62, 64, 174, 70, 65, 46]. Les maillages recouvrants ont été également utilisés pour des applications sur des hélicoptères comme l'interaction pale-tourbillon étudiée par Saunier [158] avec de l'adaptation de maillage. Dans la littérature, on trouve des générateurs de maillages recouvrants et des solveurs dédiés, citons parmi d'autres Cassiopee [2, 137, 33] et Overture [90]. Concernant les limitations ou points de recherche actuels, la performances du traitement parallèle requiert une attention particulière pour permettre l'usage de cette technique pour des calculs plus massivement parallèles comme la simulation des grandes échelles turbulentes, d'autant plus si ces calculs mettent en œuvres des grilles mobiles ou de l'adaptation de maillage.

Dans ce travail, on développe une méthode de grilles recouvrantes en conjonction avec une famille de schémas d'ordre élevé compacte et basé sur le résidu. L'état de l'art des méthodes d'ordre élevé pour des configurations géométriques complexes est brièvement passé en revue dans le suite de cette introduction.

État de l'art des méthodes d'ordre élevé

Dans la suite, on réalise une brève revue de l'état de l'art des méthodes numérique de haute fidélité pour l'aérodynamique compressible, en distinguant les méthodes de différences finies (FD), de volumes finis (FV) et des éléments finis (FE).

Méthode des différences finies Très simple à implémenter, les méthodes de différences finies permettent d'obtenir des codes de calculs très performants en terme de coût de calcul, dès lors que l'on ne considère que des géométries très simples. Même si certaines idées peuvent être retrouver plus tôt, le point de départ des méthodes de différences finies est l'article théorique fondamental de Courant, Friedrichs et Lewy (1928)[59] sur la solution de problèmes mathématiques appliqués à la physique. Les premiers pas concernant les schémas de discrétisation spatiale en différences finies peuvent être affiliés à Hildebrand (1956), Kopal (1961) ou Collatz (1966) parmi d'autres. Pour ces schémas, l'ordre élevé est généralement atteint en augmentant la taille du support de discrétisation. De plus, la plupart des schémas de différences finies sont développés pour traiter les dérivées dans chaque direction séparément. Ce type de schéma est appelé dans la suite de ce manuscrit schéma directionnel non compact (DNC). Dans l'ensemble des schémas DNC, on peut mentionner la famille des schémas décrit pour la première fois dans [113] et formellement identique, en 1D, aux schémas de volumes finis MUSCL d'ordre élevé et basés sur l'extrapolation de flux. Ces schémas sont appelés FE-MUSCL dans cette thèse, et peuvent être vu comme une généralisation du schéma de Roe [151] à l'ordre élevé.

Dans le contexte de la simulation aéroacoustique, Tam et Webb [176] puis Bogey et Bailly [37] ont proposé les schémas numériques Dispersion-Relation-Preserving (DRP), qui sont optimisés dans l'espace des nombres d'ondes de façon à maximiser leur résolvabilité. Ces schémas sont utilisés conjointement avec des techniques de filtrage [176, 37] pour supprimer les oscillations

maille à maille.

Lele [111] a contribué au développement des schémas d'ordre élevé pour l'approximation des dérivées spatiales basés sur l'utilisation d'opérateurs généralisés de Padé. Ces schémas compacts fournissent une dissipation plus faible et moins d'erreurs de dispersion que les schémas non-compact du même ordre de précision. Cependant, les schémas compacts nécessitent l'inversion d'une matrice de masse à chaque pas de temps.

Initialement proposés par Abarbanel et Kumar [15], les schémas compacts et implicites peuvent être modifiés pour déplacer le caractère implicite depuis les dérivées spatiales vers l'approximation de la dérivée temporelle. Les schémas compacts basé sur le résidu (RBC) utilisent cette procédure pour construire un schéma basé sur les dérivées du résidu. Les schémas RBC ont été initialement introduits par Lerat et Corre [112, 114], qui ont construit un schéma précis à l'ordre trois pour les équations d'Euler compressible en régime stationnaire. La dissipation de ce schéma est également construite sur le résidu et est décentrée vers l'amont à l'aide d'un pas de temps caractéristique [95]. Ce schéma peut être couplé avec une méthode d'intégration explicite en temps et est alors conditionnellement stable, mais est présenté avec un traitement implicite sans matrice. Il a été étendu aux écoulements instationnaires [57] à l'aide d'une méthode de pas de temps dual. La conception complète pour tout ordre de précision impaire, jusqu'à l'ordre 7 est décrite dans [56] pour des écoulements non-visqueux et dans [58] pour les problèmes d'advection/diffusion. Les propriétés dissipatives de ces schémas ont été étudiées dans [115], amenant à des modifications des coefficients de discrétisation de manière à assurer le caractère robuste et stable pour toutes les conditions d'écoulement, et les propriétés spectrales sont étudiées dans [84]. Les principes de construction et les propriétés numériques des schémas RBC sont discutées plus en détail dans le chapitre 1.

Méthodes des volumes finis Intrinsèquement conservative et, par conséquent, bien adapté au traitement des écoulements possédant des discontinuités, la méthode des volumes finis est probablement la plus populaire pour les simulations d'écoulements compressibles.

Une grande majorité des codes industriels est basé sur des schémas volumes-finis d'ordre deux pour la simulation des équations de Navier-Stokes moyennées (Reynolds-Averaged Navier-Stokes ou RANS). Les premiers concepts des méthodes numériques pour la simulation des écoulements compressibles peuvent être trouvés dans les travaux pionniers de Godunov [81] et les développements qui ont été menés à la NASA Ames dans le début des années soixante-dix [123, 124, 150]. Les schémas volumes-finis sont adaptés à l'utilisation de maillages structurés ou non-structurés et sont classés selon l'emplacement des inconnues (centre des cellules, centre des arrêtes, ...). Pour les méthodes volumes-finis d'ordre de précision bas (premier ou second ordre au plus), les schémas différent l'un de l'autre par la définition du flux numérique à l'interface depuis les inconnues. Pour les méthodes d'ordre élevé (ordre trois et plus), deux stratégies apparaissent. D'un côté, les inconnues peuvent représenter les moyennes sur les cellules. Cela évite d'introduire des formules de quadrature pour l'évaluation du volume et des intégrales de surface, mais requiert une étape de déconvolution implicite pour relier les moyennes des cellules aux valeurs sur les faces [78, 105, 135, 107]: cette procédure est restreinte au cas de maillage Cartésien uniformes, et son application à des grilles non-uniformes amène une perte de précision. Une méthode plus générale et valide pour des maillages structurés non-uniformes a été proposée dans [107], avec une attention aux schémas compacts: grâce à ce papier, il est possible de construire un schéma compact d'ordre de précision quatre sur des maillages Cartésien et d'ordre trois sur des maillages déformés. L'étape de déconvolution implicite nécessite la résolution d'un système linéaire pour chaque face de cellule, ce qui peut augmenter considérablement le coût de calcul. Dans [87], une reconstruction polynomiale par morceaux d'ordre élevée est utilisée pour augmenter l'ordre de

précision des schémas MUSCL sur des maillages arbitraires non-structurés. Les valeurs des faces sont calculées à partir des valeurs des moyennes sur les cellules via une technique de correction récursive impliquant des approximations des dérivées des inconnues. Cependant, cette approche requiert le stockage additionnel de la connectivité nécessaire pour la reconstruction du support de discrétisation et le stockage ou le calcul à chaque pas de temps des matrices de reconstruction.

D'un autre côté, une deuxième approche des schémas volumes-finis consiste à considérer des valeurs ponctuelles des variables inconnues dans le volume de contrôle (par exemple les valeurs au centre des cellules). Lorsqu'on considère les schémas volumes-finis sur maillages structurés, et dans un effort de limiter le stockage mémoire et le coût de calcul, une approche avec valeur ponctuelle au centre des cellules est un choix raisonnable, comme présenté dans [147, 146].

Les schémas RBC, initialement présentés dans le cadre de la méthode des différences finies ont été reformulés en schémas volumes-finis en utilisant une extension directe des flux numériques ou en utilisant une extension d'ordre élevée en volumes-finis pour les maillages structurés irréguliers [89, 85]. L'étude des propriétés dissipatives de cette extension au volumes-finis des schémas RBC a été menée par Grimich [83, 85]. Finalement, une méthode volume-finis d'ordre trois sur maillage non-structuré a été proposée par Corre and Du dans [55].

Méthode des éléments finis Le cadre numérique de la méthode des éléments finis est généralement bien établi d'un point de vue mathématique. Basé sur la forme faible des équations de conservation, leur capacité à atteindre un ordre élevé repose sur le choix des fonctions test à support compact. Les méthodes des éléments finis sont bien adaptées aux géométries complexes sur maillage non-structuré mais conduisent généralement à des discrétisations purement centrées et par conséquent non dissipatives, et donc non adaptées aux problèmes de mécanique des fluides. Pour rendre la méthode des éléments finis applicable à la simulation des écoulements de fluides, l'ajout des termes de stabilisation est alors nécessaire.

Au cours des dernières années, une attention considérable a été portée aux méthodes Galerkin discontinues (DG), aux méthodes d'éléments spectraux et aux méthodes continues de distribution du résidu (RD).

Les méthodes de Galerkin discontinues combinent les caractéristiques de la méthode des volumes finis et une discrétisation standard (conforme) des éléments finis [52]. Elles sont bien adaptés pour des problèmes de convection-diffusion, y compris la simulation numérique des écoulements compressibles. Le degré de précision est choisi par la définition de la base polynomiale sous-jacente. La solution est discontinue aux interfaces de cellule, où un solveur de Riemann est appliqué. Les méthodes DG peuvent être utilisées en conjonction avec des schémas temporels explicites parce que la matrice de masse est bloc-diagonale et donc facile à inverser et peut être stockée. Basé sur le cadre éléments-finis sur des grilles non structurées, la méthode est très coûteuse en raison de la nécessité de stocker les fonctions de forme, mais elle est compacte et donc adaptée à une parallélisation massive. L'utilisation de la DG pour les simulations RANS ont été discutés dans [26] et pour les écoulements turbulents 3D dans [106]. Pourtant, les discontinuités de contact et la capacité de capture des chocs de ces méthodes d'ordre élevé exige des avancées supplémentaires de la recherche.

Concernant les méthodes des éléments spectraux, diverses formulations existent, parmi lesquelles les volume spectraux (SV) et les différences spectrales (SD). De même que pour les méthodes DG, elles sont fondées sur l'utilisation de polynômes discontinus entre les éléments. La reconstruction de la solution est basée sur l'introduction de degrés de liberté interne dans chaque cellule, et donc ne requiert pas d'informations de cellules voisines. On se reporte à [170, 187, 189, 188, 171, 121, 186] pour les détails sur les méthodes SV.

Les méthodes de différences spectrales (SD) [120] sont basés sur le cadre FD avec une recon-

struction au sein de la cellule basée sur les degrés de liberté spectraux avec utilisation de grille décalée. Elles utilisent aussi des reconstructions discontinues entre éléments. La méthode SD est présentée comme plus facile à mettre en œuvre que les méthodes DG et SV, en particulier pour les frontières courbes représentées à l'ordre élevé, car elle ne comporte pas d'intégrales de surface ou de volume.

Parmi les méthodes éléments-finis continues, citons la méthode à distribution du résidu (RD), dont le cadre est basée sur les méthodes éléments finis et volumes finis. Les schémas numériques RD représentent une alternative très intéressante aux schémas DG. Bien que compactes et probablement plus souple, les méthodes DG présentent l'inconvénient grave d'une croissance très rapide du nombre de degrés de liberté avec le degré des polynômes des cellules. Dans les schémas RD, la formulation reste local comme dans la méthode DG, mais le nombre de degrés de libertés croit moins vite parce que la solution est supposé être continue. Une autre différence entre les méthodes RD et DG est que, à ce jour, au moins, les propriétés non oscillatoires des schémas RD dans le cas de solutions discontinues sont sans doute mieux comprise que dans l'approche DG. Inspirés de papiers de Roe [152] et Ni [128], les schémas RD reposent sur de nombreuses contributions de différents groupes qui ont enrichi la méthode (voir [17] et références citées dans ce papier).

Comme une dernière remarque sur les méthodes des éléments-finis, puisque les codes de simulation actuels dans l'industrie aéronautique reposent sur des méthodes de volumes finis, l'implémentation des éléments finis nécessite le développement de tout nouveaux codes, en termes de mise en œuvre et techniques de parallélisation, puisque les codes actuels ne sont pas assez souples pour permettre l'utilisation de l'approche de discrétisation différente dans la même architecture.

Architectures de code pour la simulation de la mécanique des fluides

Le but principal d'un code de MFN (mécanique des fluides numérique) est de rassembler des méthodes numériques - pour le traitement de la géométrie, calcul des propriétés physiques, discrétisation en temps et en espace - dans une entité et de les rendre accessibles et interopérables. Les codes de calcul sont soit dédiés à une seule application ou à usage général. Dans le premier cas, ils sont adaptés et optimisés pour une configuration d'écoulement simple et peuvent difficilement être appliqués à une autre configuration sans modifications importantes; dans l'autre cas, les codes multi-usages permettent généralement de résoudre tout type de configuration, mais exigent la manipulation de structures de données complexes et sont souvent difficiles à manipuler pour les chercheurs. Plusieurs codes open source basés principalement sur des maillages non structurés sont disponibles (FreeFEM [11], OpenFoam[98, 12], Coolfluid [109, 142], Nek5000 [134], Code_Saturne [21]). Ces codes numériques sont écrits principalement en utilisant des langages de programmation compilés tels que FORTRAN [18], C ou C++[169]. Le code it Kernel est plus ou moins statique après sa compilation et dans le cas d'un langage orienté objet, les extensions sont faites via l'héritage de classes génériques.

Plusieurs codes de mécanique des fluides numérique ont commencé à utiliser des langages de programmation plus souple comme le langage interprété Python [9, 108]. Le Python peut être utilisé pour assembler ensemble des morceaux de programmes [191]. Il est aussi utilisé dans le solveur elsA[8] de l'ONERA, le solveur Tau [3] du DLR ou, plus récemment, à l'université de Stanford, dans le code *open source* SU2[133] pour créer une interface utilisateur conviviale par dessus la structure C++ du programme, ou encore dans le solveur hélicoptère HELIOS [168] de l'US Army pour créer un environnement de couplage de solveurs numériques.

Dans ce travail, on développe un code de mécanique des fluide novateur, nommé DynHoLab (Dynamic High-order Laboratory), combinant le langage Python (interprété, object-oriented) et le langage Fortran (compilé, rapide). Cela permet d'allier des développements rapides dans un environnement flexible, généraliste et modulaire tout en tirant partie des performances CPU à l'aide du langage Fortran.

Cadre de la thèse

Ce travail est l'occasion de mener une réflexion approfondie sur l'architecture de codes CFD. Le défi qui se pose lorsqu'on traite des applications industrielles non-triviales réside dans la capacité des partenaires de recherche à gérer des programmes informatiques complexes avec peu de ressources humaines. Une solution alliant un mélange des langages Python et Fortran est proposé avec des détails sur les concepts à la base de l'architecture du code, ainsi que l'étude des performances parallèles. Les méthodes numériques sont validés sur des cas-tests de complexité croissante, démontrant en même temps la variété de la physique et la géométrie actuellement réalisable avec DynHoLab.

Basé sur le solveur conçu en première partie, ce travail tente également de fournir un moyen de gérer des géométries complexes tout en augmentant la précision des méthodes numériques, avec la question sous-jacente de garantir un niveau élevé d'efficacité globale de la simulation numérique. Afin d'appliquer les schémas RBC d'ordre élevé à des géométries complexes, la stratégie actuelle consiste en une mise en œuvre de techniques multi-domaines avec recouvrement de maillages structurés.

Structure de ce rapport

Le manuscrit est organisé de la manière suivante.

— Le chapitre 1 examine les lois de conservation, modèle physique et numérique méthodes utilisés dans ce travail.

— Le chapitre 2 est dédié au développement du code de calcul DynHoLab, incluant l'architecture du code, ses performances parallèles et un étude de scalabilité et son état de développement courant.

— Le chapitre 3 est dédié à la validation des méthodes numériques implémentées dans DynHoLab. A cette fin, on considère une série de cas-tests de complexité croissante. Cela inclue des cas d'advection linéaire stationnaires et instationnaires, une validation du solveur RANS 2-D et 3-D RANS and une application à de la LES implicite sur un cas de collines 2-D périodiques.

— Finalement, le Chapitre 4 détaille le développement de la stratégie de maillages recouvrants, sa validation et son application à des cas d'écoulements 2-D transsoniques sur un profil d'aile NACA0012 isolé ou de deux profils en positions décalées.

Chapter 1

Generalities: from the continuous problem to discrete approximations

"Knowledge is indivisible. When people grow wise in one direction, they are sure to make it easier for themselves to grow wise in other directions as well. On the other hand, when they split up knowledge, concentrate on their own field, and scorn and ignore other fields, they grow less wise – even in their own field."

Isaac Asimov (1920-1992), *The Roving Mind (1983)*

Contents

Résumé du chapitre

On s'attache, dans ce chapitre, à introduire le problème résolu. On commence par amener les lois de conservation, qui constituent le problème continu, sous forme de divergence et sous forme faible ou intégrale. Les modèles associés à ces lois sont ensuite détaillés. Dans une deuxième partie, on s'intéresse à la manière de discrétiser le domaine spatial et temporel et plus particulièrement à la manière d'approcher les dérivées. Finalement, on explicite la construction des schémas numériques utilisés dans cette thèse, à savoir les schémas RBC et FE-MUSCL pour des ordres de précision de 3 à 7. On y associe des méthodes numériques pour la convergence de problème stationnaires ou pour l'intégration en temps de problèmes instationnaires.

In this work, we consider the numerical approximation of some conservation laws of increasing complexity from the scalar linear problem to the Navier-Stokes equations. In this chapter, we first review the equations and models employed later on in applications. Then, we deal with the numerical methods used for both space and time discretizations and briefly discuss their properties.

1.1 Conservation laws: the continuous problem

The design of new numerical schemes for compressible flow simulations often starts with the study of simple conservation laws for which one has more information on the properties of the exact solution. In this section, we introduce conservation laws, which are a system of partial differential equations stating the conservation of some properties over a given region of space and time.

1.1.1 Problem statement

Conservation laws in divergence form

Let us consider the numerical approximation of solutions of the following system of conservation laws written in divergence form (also called conservative form)

$$\begin{cases} \dfrac{\partial w}{\partial t} + \nabla \cdot \mathcal{F}(w) = \mathcal{S}(x) & \text{on} \quad \Omega_T = \Omega \times [0, t_f] \subset \mathbb{R}^D \times \mathbb{R}^+ \\ w(x, t=0) = w_0(x) & \text{on} \quad \Omega \subset \mathbb{R}^D \\ w(x \in \partial\Omega, t) = w_{bc}(t) & \text{on} \quad \partial\Omega_T \subset \mathbb{R}^{D-1} \times \mathbb{R}^+ \end{cases} \tag{1.1}$$

where $w : \Omega_T \mapsto \mathbb{R}^m$ is an m-vector of conserved quantities, $x \in \mathbb{R}^D$ is a D-vector defining position in space, $\mathcal{F} : \mathbb{R}^m \mapsto \mathbb{R}^{m \times D}$ is the tensor of the conservative fluxes, $\mathcal{S} : \Omega \mapsto \mathbb{R}^m$ is an m-vector of source terms independent on w and $\Omega_T = \Omega \times [0, t_f] \subset \mathbb{R}^D \times \mathbb{R}^+$ is the space-time domain over which solutions are sought. When $\mathcal{S} = 0$, we get the homogeneous case. Note that the dependence of the state vector w on the space x and the time t is not written explicitly for the sake of clarity. System (1.1) is equipped with a set of boundary conditions on $\partial\Omega$ (or on properly defined portions of this set), and also with an initial condition. Let us denote the projections of the flux tensor function on the reference Cartesian frame coordinate vectors as follows:

$$\mathcal{F}(w) = (F_1(w), \ldots, F_D(w)) \qquad \text{on} \qquad \mathcal{B} = (e_1, \ldots, e_D). \tag{1.2}$$

We assume that the system of conservation laws (1.1) is hyperbolic, that is, for any given unit vector $n = (n_1, \ldots, n_D) \in \mathbb{R}^D$, the matrix

$$J(n, w) = \sum_{d=1}^{D} \frac{\partial F_d(w)}{\partial w} n_d \tag{1.3}$$

admits a complete set of real eigenvalues along with linearly independent eigenvectors. We denote by $\Lambda(n, w)$ the diagonal matrix of the eigenvalues of $J(n, w)$ and by $R(n, w)$ the matrix of its right eigenvectors :

$$J(n, w) = P(n, w)\Lambda(n, w)P(n, w)^{-1} \tag{1.4}$$

For more convenience, we also introduce the notation

$$\forall d \in [\![1, D]\!], \;\; J_d = \frac{\partial F_d(w)}{\partial w} \tag{1.5}$$

for the flux Jacobian matrices. Using the notation (1.5), the equation (1.1) can be rewritten in the quasi-linear form

$$\frac{\partial w}{\partial t} + \sum_{d=1}^{D} J_d \frac{\partial w}{\partial x_d} = \mathcal{S}(x)\,. \tag{1.6}$$

Weak form and integral form

The differential form of conservation laws (1.1) is only valid for sufficiently smooth functions. Discontinuous solutions can be found by solving the equations in integral form

$$\forall \Omega_s \subset \Omega, \quad \frac{\partial}{\partial t} \int_{\Omega_s} w \, \mathrm{d}\Omega + \oint_{\Gamma_s} \mathcal{F} \cdot \mathrm{d}s_{ext} = 0 \tag{1.7}$$

which states that the rate of change of w within the fixed domain Ω_s is equal to the flux of that quantity through its boundary Γ_s. This formulation leads naturally to the finite volume approach.

Another related approach which also removes the condition of smoothness imposed on w, consists in multiplying the differential equation by a smooth test-function φ and integrating by parts to move the derivatives off from w onto φ. The resulting integral equation is called the weak form of the conservation law,

$$\forall \varphi \in \mathcal{C}^1(\Omega_T), \quad \frac{\partial}{\partial t} \int_{\Omega} \varphi w \, \mathrm{d}\Omega - \int_{\Omega} \nabla\varphi \cdot \mathcal{F} \, \mathrm{d}\Omega + \oint_{\Gamma} \varphi \mathcal{F} \cdot \mathrm{d}s_{ext} = 0 \tag{1.8}$$

and is the basis of the finite element method. It can be shown that the integral form (1.7) and the weak form (1.8) are equivalent [119]. Solutions of these equations are called weak solutions and may contain any finite number of discontinuities. While classical solutions are also weak solutions of the problem, the weak form (1.8) enlarges the set of possible admissible solutions to the set of essentially bounded functions. However, additional constraints are needed to guarantee the uniqueness of the solution. Existence and uniqueness of the solutions of the continuous problem need (or imply) the existence of some vanishing dissipative phenomenon related to the concept of entropic weak solution or vanishing viscosity solution [19, 41]. Care must therefore be taken to ensure that the numerical scheme converges towards the correct, entropy-satisfying solution of the problem. It is well-known for instance that Roe's approximate Riemann solver violates this entropy condition and admits expansion shocks as steady solutions [151].

Across a discontinuity surface Σ propagating with a velocity c, the solution w undergoes a jump which satisfies the Rankine-Hugoniot relation

$$s[\![w]\!] = [\![\mathcal{F} \cdot m]\!] \tag{1.9}$$

where m is the normal to the discontinuity Σ, $s = c \cdot m$ represent its speed in the direction m and the brackets denote the jump in the quantity at the discontinuity.

1.1.2 Flow models

Both in the problems considered in this work and in the computational code created for their numerical solution, various conservation laws are considered: scalar advection equation, Burgers non-linear equation, Euler and Navier-Stokes system of equations. The derivation and properties of these conservation laws is well-known (see for instance [93]) and will not be recalled in this thesis. We only recall expression of the Navier-Stokes system of equations as a way to introduce the notations in use in this document.

Navier-Stokes equations

The Navier-Stokes equations arise from applying the Newton's second law to fluid motion, along with mass and energy conservation principles. We assume that the stress tensor is the sum of a pressure term (isotropic part of the Cauchy strain tensor) and the diffusive viscous tensor (anisotropic part of the tensor). The system of equations can be cast in the conservative form (1.1) and represent two scalar equations and one vector equation: the mass conservation (or continuity equation), the momentum conservation and the energy conservation. The fluxes are split in an inviscid part and a viscous part

$$\frac{\partial w}{\partial t} + \nabla(\mathcal{F}^E - \mathcal{F}^V) = \mathcal{S}(x)\,. \tag{1.10}$$

The state vector w is chosen to contain the conservative variables. The state vector and inviscid (Euler) fluxes read

$$w = \begin{bmatrix} \rho \\ \rho u_1 \\ \rho u_2 \\ \rho u_3 \\ \rho E \end{bmatrix},\ F_1^E(w) = \begin{bmatrix} \rho u_1 \\ \rho u_1^2 + p \\ \rho u_1 u_2 \\ \rho u_1 u_3 \\ \rho u_1 H \end{bmatrix},\ F_2^E(w) = \begin{bmatrix} \rho u_2 \\ \rho u_1 u_2 \\ \rho u_2^2 + p \\ \rho u_2 u_3 \\ \rho u_2 H \end{bmatrix},\ F_3^E(w) = \begin{bmatrix} \rho u_3 \\ \rho u_3 u_1 \\ \rho u_3 u_2 \\ \rho u_3^2 + p \\ \rho u_3 H \end{bmatrix} \tag{1.11}$$

with ρ the density, u_i the fluid velocity in the ith direction, $E = e + 1/2||u||^2$ the total energy, p the pressure and $H = h + 1/2||u||^2$ the total enthalpy. The dissipative fluxes in turn reads

$$\forall d \in \{1,2,3\}, \quad F_d^V(w) = \begin{bmatrix} 0 \\ \tau_{d1} \\ \tau_{d2} \\ \tau_{d3} \\ -\frac{\partial q}{\partial x_d} + u \cdot (\nabla \tau\ e_d) \end{bmatrix} \tag{1.12}$$

with u, τ, μ and q respectively the fluid velocity field, the viscous stress tensor, the dynamic viscosity and the heat flux. In order to close this system, some relations should be added for the expression of the pressure p, the stress tensor τ and the heat flux q. Let us now deal with the models in use in this work.

1.1.3 Constitutive relations

Equation of state

The fluids considered are perfect gases governed by the ideal gas law $p = \rho r T$ where r is the gas constant and T the temperature. The fluids are supposed to be calorically perfect, *i.e.* in thermodynamic equilibrium, not chemically reacting, and such that their internal energy e and enthalpy h are functions of temperature only and not of pressure. Even more, the isochoric and isobaric specific heats C_v and C_p are constant. We introduce the specific heat ratio $\gamma = C_p/C_v$. Since we deal with compressible flows, we introduce the speed of sound $c = (\partial p/\partial \rho)_s^{1/2}$, with s the entropy, which becomes for a perfect gas

$$c = \left(\frac{\gamma p}{\rho}\right)^{1/2} \tag{1.13}$$

Heat flux

The heat flux is governed by the Fourier's law and writes $q = -\kappa \nabla T$ with κ the fluid thermal conductivity. This can derived from the Prandtl number $P_r = \mu C_p/\kappa$.

Newtonian fluids

We restrict our scope to fluids for which the viscous stresses arising in the flow are proportional to strain rate. The slope is called dynamic viscosity of the Newtonian fluid and is denoted by μ. Under the Stokes hypothesis, the expansion viscosity (or bulk viscosity) is null and the Lamé parameter λ (or second Lamé parameter) is related to the viscosity by the relation $3\lambda + 2\mu = 0$. This is theoretically valid for a monoatomic gas or for $T = 0$ and is just an approximation for other gases. Then, denoting by $\epsilon = 1/2(\nabla u + (\nabla u)^T)$ the strain tensor, the stress tensor τ writes $\tau = 2\mu\epsilon + \lambda\, tr(\epsilon) I$ giving

$$\tau = \mu \left(\nabla u + (\nabla u)^T \right) - \frac{2}{3} \mu\, tr(\nabla u) I \tag{1.14}$$

with $tr(.)$ the trace operator and I the identity matrix.

Viscosity laws

Equation (1.14) has to be supplemented by a law relating the viscosity to the thermodynamic variables, and namely the temperature (dependency on the pressure being negligible in most practical cases). One possibility is to neglect dependency on the temperature and take a constant viscosity. For compressible flows, it is more appropriate to use the power law

$$\mu = \mu_{ref} \left(\frac{T}{T_{ref}} \right)^n , \tag{1.15}$$

or, more often, the Sutherland law [172]

$$\mu = \mu_{ref} \left(\frac{T}{T_{ref}} \right)^n \frac{T_{ref} + S_{Suth}}{T + S_{Suth}} \tag{1.16}$$

with, for both the power law and Sutherland law and for air at moderate temperatures and pressures, $\mu_{ref} = 1.716 \times 10^{-5}\ kg.m^{-1}.s^{-1}$, $T_{ref} = 273.15\ K$, $n = 2/3$, and $S_{Suth} = 110.4\ K$.

Turbulence modeling

This paragraph is not the place for a review of any approach to the turbulence simulation. A brief overview is provided in the introduction where the reader can find major references on the various formalisms. For further details on the characteristics of a turbulent flow, the reader is invited to refer to [118, 43, 27, 141, 92, 110]. As justified in the introduction of this document, direct numerical simulation (DNS) is not a realistic possibility in most cases of practical importance. In this work, the turbulence is regarded in the formalism of either the Reynolds-Averaged Navier-Stokes (RANS) turbulence models or the Large Eddy Simulation (LES).

As formalism for turbulence, let us consider, under the hypothesis of ergodicity [183], the Reynolds decomposition [143] applied to system (1.10-1.12). More exactly, the Reynolds decomposition is applied to the scalar variables p and ρ using the Reynolds average and the decomposition using the Favre average [71, 72] is used for the other variables. This results in

the appearance of two new unclosed terms – the Reynolds stress tensor τ_r and the turbulent heat flux q_t – that need to be modeled. The turbulence models used in this thesis follow the Boussinesq hypothesis [38, 39] stating that the momentum transfer caused by turbulent eddies can be modeled with an eddy viscosity. This is an analogy with how the momentum transfer caused by the molecular motion in a gas can be described by a molecular viscosity. For more details about the limitation of the eddy viscosity approximation see [22]. Under the Boussinesq hypothesis, the Reynolds stress tensor and turbulent heat flux write

$$\tau_r = -\frac{2}{3}\left(\rho k + \mu_t \nabla \cdot u\right) I + 2\mu_t \epsilon \tag{1.17}$$

$$q_t = -\frac{C_p \mu_t}{P_t} \nabla T \tag{1.18}$$

with μ_t and P_t being respectively the turbulent viscosity coefficient and the turbulent Prandtl number. The turbulent viscosity can be determined from an algebraic relationship [42, 24] or from local turbulent quantities, like the turbulent kinetic energy k, its dissipation rate ϵ or a turbulence length scale. The determination of these quantities requires the resolution of additional transport equations. Several alternative transport equation models are available (for an overview see [190, 22]). The Boussinesq approximation allows to write the instantaneous Navier-Stokes equations and the RANS equations under the same form, by introducing the following starred quantities

$$\begin{cases} E^* = e + \dfrac{1}{2}\|u\|^2 + k = E + k \\ p^* = p + \dfrac{2}{3}\rho k \\ \tau^* = \tau + \tau_r + \dfrac{2}{3}\rho k I = (1 + \dfrac{\mu_t}{\mu})\tau \\ q^* = q + q_t = (1 + \dfrac{\mu_t}{\mu}\dfrac{p}{P_t})q \end{cases} \tag{1.19}$$

with k the turbulent kinetic energy. For the computations presented in chapters 3 and 4, RANS equations are closed employing the two transport equations model of Menter, namely the $k-\omega$ SST model [126]. To solve the RANS system, one can choose between two strategies: the first choice consists in solving two uncoupled system of equations, *i.e.* solve the RANS equations with fixed turbulent quantities and then solve the equations of the turbulence model with fixed mean flow fields ; the second choice consists in solving the coupled system, *i.e.* consider the system composed by the RANS equations and the equations of the chosen turbulence model.

A second approach considered in this thesis is Large Eddy Simulation (LES) [164], which arises from low-pass filtering of the Navier-Stokes equations. Again, to account for compressibility effects, Favre proposed a density-weighted filtering operator, called Favre filtering [73], which, in the limit of incompressibility, becomes the standard filtering operator. Usually, the cut-off frequency is related to the characteristic mesh size employed for the LES simulation. Due to the non linearity of Navier-Stokes equations, the filtered equations involve several unclosed terms. Generally, these are collected into a subgrid stress tensor (SGS which represents the effect of the subgrid scales of motion on the resolved ones). The most common approach [79] consists in the inclusion of an explicit SGS modeling strategy and the design of an accurate enough numerical method so as to consider the numerical error on the space and time derivatives negligible. As an alternative, the implicit LES (or ILES) approach [86] makes use of the intrinsic numerical dissipation introduced by some discretization schemes [66, 67] or of selective explicit filtering techniques [37] to drain energy associated to the unresolved flow scales. For this purpose, it

is of paramount importance that numerical dissipation/filtering only acts on ill-resolved scales, without modifying the resolved ones. ILES modeling is employed in one application of chapter 3.

1.2 Discrete approximations

Hereafter, we first discuss the domain tesselation and provide the operators used to describe the derivative approximations. Then, the numerical schemes employed in the applications for both time and space are reviewed. For simplicity numerical schemes for the approximation of space derivatives are presented in the finite difference framework and for regular Cartesian grids. The practical implementation of the schemes on general curvilinear meshes is discussed in chapter 2.

1.2.1 Mesh geometry

This work restricts its scope to the use of structured meshes over the computational domain Ω. We will denote by $\mathcal{G}_h$ the grid, h being a reference element length, and we denote by E the generic element – or cell – of $\mathcal{G}_h$, $|E|$ being its area. For all grids, the following property is assumed to be verified:

$$\exists (C_1, C_2) \in \mathbb{R} \times \mathbb{R}, \quad 0 < C_1 \leq \sup_{E \in \mathcal{G}_h} \frac{h^D}{|E|} \leq C_2 < \infty \,. \tag{1.20}$$

This corresponds to the fact that no vanishing area elements are present, as well as no unreasonably stretched cells. In this chapter, for the description of the numerical schemes in the finite difference framework, the domain tesselation will be considered to be a uniform Cartesian mesh, that is, on a domain $\Omega \in \mathbb{R}^D$:

$$\forall d \in [\![1, D]\!], \ \forall i_d \in [\![1, I_d]\!], \ x_d^{i_d} = i_d \, \delta x_d \tag{1.21}$$

with I_d the number of nodes in the mesh direction d and with steps of the same order, say $O(h)$:

$$\forall d \in [\![1, D]\!], \ \delta x_d = O(h). \tag{1.22}$$

It should be noted that the mesh directions do not necessary match the reference Cartesian frame $\mathcal{B}$. Indeed, either a rotation or translation could be applied to the grid while keeping its Cartesian properties. For this reason and for the sake of clarity, we therefore use the index notation (i, j, k) to identify the position in a grid $\mathcal{G}_h$.

Considering a discretization $\tilde{u}$ on $\mathcal{G}_h$ of an unknown field u in $L^p(\Omega)$ with values in $\mathbb{R}^D$, let us define some basic operators for the expression of derivative approximations. We will denote by δ_d and μ_d the basic difference and average operators of characteristic distance h at the generic point (i, j, k)

$$\begin{aligned}
(\delta_1 \tilde{u})_{i+\frac{1}{2},j,k} &= \tilde{u}_{i+1,j,k} - \tilde{u}_{i,j,k} & (\mu_1 \tilde{u})_{i+\frac{1}{2},j,k} &= \frac{1}{2} (\tilde{u}_{i+1,j,k} + \tilde{u}_{i,j,k}) \\
(\delta_2 \tilde{u})_{i,j+\frac{1}{2},k} &= \tilde{u}_{i,j+1,k} - \tilde{u}_{i,j,k} & (\mu_2 \tilde{u})_{i,j+\frac{1}{2},k} &= \frac{1}{2} (\tilde{u}_{i,j+1,k} + \tilde{u}_{i,j,k}) \\
(\delta_3 \tilde{u})_{i,j,k+\frac{1}{2}} &= \tilde{u}_{i,j,k+1} - \tilde{u}_{i,j,k} & (\mu_3 \tilde{u})_{i,j,k+\frac{1}{2}} &= \frac{1}{2} (\tilde{u}_{i,j,k+1} + \tilde{u}_{i,j,k})
\end{aligned} \tag{1.23}$$

As the reader might not be familiar with this notation, we provide in appendix A some usual derivative approximations obtained by combinations of these operators, including directional non-compact approximations of the first derivative commonly used in the finite difference framework and directional compact approximations based on Padé fractions (both standard and staggered operators).

1.2.2 Conservative approximation of space derivatives

We consider that the unknowns are located at cell-centers, meaning that the semi-discrete approximation of the conservative form (1.1) of the conservation laws writes

$$\left(\frac{\partial w}{\partial t}\right)^n + \left[\frac{\delta_d H_d}{\delta x_d}\right]^n_{i,j,k} = 0 \tag{1.24}$$

where the Einstein summation convention is employed and H_d is the numerical flux evaluated at the center of faces thanks to the difference operator δ_d whereas the state vector $\tilde{w}$ is cell-centered. In this work, all schemes will be presented in the formalism of a sum of a centered approximation of the physical flux plus a numerical dissipation, *i.e.* $H_d = H_d^c + H_d^d$ where the subscript c and d stands for *centered* and *dissipative* part of the numerical flux.

1.2.3 Directional approximation of the space derivative

A classical way to approximate the fluxes F_d is to consider each mesh direction separately. Starting from the simplest centered approximation of the centered part of the numerical flux $H_d^c = \mu_d F_d + O(h^2)$, and then using some Taylor series expansions, it is possible to derive higher-order approximations of the first derivative by successive correction of the leading truncation error term. The standard high-order approximations of the first derivative in terms of the mean and difference operators (1.23) can be found in appendix A under the name *non-compact approximations*. For compressible flow simulations, it is essential to complete the numerical scheme by a dissipation H_d^d, which should also be high-order accurate not to lose the overall high-order accuracy of the approximation. In order to obtain dissipative approximations, we follow hereafter the approach presented for the first time in [116]. This results in the family of schemes described for the first time in [113], which can be seen as a generalization of the Roe scheme [151] to high-order accuracy

$$H_d = \mu_d F_d - \frac{1}{2}|Q_d|\delta_d\tilde{w} \qquad \text{(Roe scheme)} \tag{1.25}$$

$$H_d = (I - \frac{1}{6}\delta_d^2)\mu_d F_d \ + \ \frac{1}{12}|Q_d|\delta_d^3\tilde{w} \qquad \text{(order 3)} \tag{1.26}$$

$$H_d = (I - \frac{1}{6}\delta_d^2 + \frac{1}{30}\delta_d^4)\mu_d F_d \ + \ \frac{1}{60}|Q_d|\delta_d^5\tilde{w} \qquad \text{(order 5)} \tag{1.27}$$

$$H_d = (I - \frac{1}{6}\delta_d^2 + \frac{1}{30}\delta_d^4 - \frac{1}{140}\delta_d^6)\mu_d F_d \ + \ \frac{1}{280}|Q_d|\delta_d^7\tilde{w} \qquad \text{(order 7)} \tag{1.28}$$

with Q_d the Roe average in direction d, this last being established in [151] to ensure that $\delta_d F_d = |Q_d|\delta w$. Another approach to derive the same directional schemes in the finite volume framework is to use a MUSCL reconstruction to the physical fluxes [113]. For this reason, in the following, the preceding family of schemes is referred-to as the FE-MUSCL schemes – for flux-extrapolation MUSCL schemes.

Concerning the properties of the schemes, they are conservative but not total variation diminishing (TVD). In order to provide such a property to the schemes, one could think of introducing *e.g.* flux limiters [173]. A simpler and cheaper alternative consists in replacing the numerical dissipation of FE-MUSCL schemes by an artificial dissipation, for example based on a blending of second and higher derivatives as proposed by Jameson [97]. For higher orders, we adopt the numerical dissipation introduced by Kim and Lee [101]

$$(H_d^d) = \rho(J)\left[\epsilon_2\delta w + \epsilon_{2p}\delta^{2p-1}w\right]$$

1.2.4 Residual-Based Compact approximation of the space derivative

As an alternative to the approach presented in the preceding section to consider the mesh direction separately, intrinsically multidimensional discretizations can be obtained by approximating all space derivatives simultaneously. Specifically, we consider hereafter the family of Residual-Based Compact (RBC) schemes, first introduced by Lerat and Corre (2001).

Residual-based schemes

A Residual-Based (RB) scheme acts on the whole residual and can be written only in terms of approximations of the residual and its derivatives.

$$r = \frac{\partial w}{\partial t} + \nabla \cdot \mathcal{F} - \mathcal{S} \tag{1.29}$$

For steady flow computations, the residual reduces to the divergence of the flux and possibly source terms. For the study of the discretization of the convective term, we employ the notation $r^{\text{conv}} = \nabla \cdot \mathcal{F}$. Precisely, a RB scheme writes $\tilde{r}_0 = \Phi$, where $\tilde{r}_0$ is a centered approximation of the residual and Φ is a dissipation term.

Compact approximation of space derivatives

For smooth solutions, the simplest centered discretization of $\nabla \cdot \mathcal{F}$ is build by means of simple second-order accurate centered approximations of the first derivative ; this is called *simplest centered* in table (1.1). Through the correction of the leading truncation error term, the approximation can be improved. This is classically done using either a compact or non-compact formulation. We first consider the possibility of increasing accuracy up to the fourth order. With the non-compact formulation, the second-order error term is corrected explicitly (refer to appendix A) as it is done for FE-MUSCL schemes (see section 1.2.3). The expression and truncation error are given under the name *non-compact* in table (1.1). The price to pay to achieve, *e.g.* the fourth-order accuracy with non-compact approximations is to enlarge the stencil. Unlike the non-compact formulation, the Padé-compact discretization is obtained by correcting the second-order error term implicitly. For this compact approximation, the stencil is the same as the *simplest centered* one. Yet, it requires the resolution of an algebraic system at each step (see appendix A). More generally, following the notations of [115], the first derivatives of the inviscid fluxes are approximated at order $2p$ using the Padé fractions

$$\left(\frac{\partial F_d^E}{\partial x_d}\right)_{i,j,k} = \left(\frac{\overline{N_d}}{\overline{D_d}} \frac{\delta_d \mu_d}{\delta x_d} F_d^E\right)_{i,j,k} + O(\delta x_d^{2p}) \tag{1.30}$$

with $\overline{N_d}$ and $\overline{D_d}$ formal polynomials of the difference operator δ_d^2

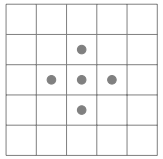
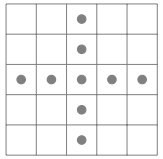
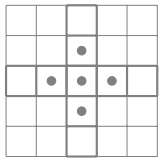
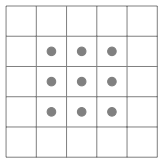
$$\begin{aligned} \overline{N_d} &= I + \bar{a}\delta_d^2 \\ \overline{D_d} &= I + \bar{b}\delta_d^2 + \bar{c}\delta_d^4 \end{aligned} \tag{1.31}$$

and $\bar{a}$, $\bar{b}$, $\bar{c}$ coefficients that are selected to achieve the required order of accuracy on a compact stencil.

Residual-based compact approximation

The *residual-based compact* discretization is obtained by using Padé operators to construct approximations of the main residual and the numerical dissipation term.

Table 1.1: Some discretizations of the convective part of the residual for the homogeneous system (1.1)

Discretisation	Order	Expression	Truncation error ($r = 0$)	Stencil ($D = 2$)
Simplest centred	2	$\tilde{r}^{\text{conv}} = \sum_d \frac{1}{\delta x_d} \delta_d \mu_d F_d$	$\epsilon = \frac{1}{6} \sum_d \delta x_d^2 \frac{\partial^3 F_d}{\partial x_d^3} + 0(h^4)$	
Non-compact	4	$\tilde{r}^{\text{conv}} = \sum_d \frac{1}{\delta x_d} \delta_d \mu_d \Big(I - \frac{\delta_d^2}{6} \Big) F_d$	$\epsilon = -\frac{1}{30} \sum_d \delta x_d^4 \frac{\partial^5 F_d}{\partial x_d^5} + 0(h^6)$	
Pade-compact	4	$\tilde{r}^{\text{conv}} = \sum_d \frac{1}{\delta x_d} \nabla_d^{Pade} F_d$ with $\nabla_d^{Pade} F_d = \frac{\delta_d \mu_d}{I + \frac{1}{6} \delta_d^2}$	$\epsilon = -\frac{1}{180} \sum_d \delta x_d^4 \frac{\partial^5 F_d}{\partial x_d^5} + 0(h^6)$	
RBC	4	$\tilde{r}^{\text{conv}} = \sum_d \sum_{\substack{f \\ f \neq d}} \frac{1}{\delta x_d} \delta_d \mu_d \Big(I + \frac{\delta_f^2}{6} \Big) F_d$	$\epsilon = -\frac{1}{180} \sum_d \delta x_d^4 \frac{\partial^5 F_d}{\partial x_d^5} + 0(h^6)$	

The effort is then reported on the unsteady term as shown in equation (1.32). Note that this complication disappears for steady computations for which the operator applied to the time derivative can be reduced to the identity without loss of accuracy. In the RBC language, the centered approximation of the residual is called the main residual and denoted by the subscript $\tilde{r}_0$. It is evaluated at cell centers. For 2-D inviscid computations, the approximation of the main residual writes

$$(\tilde{r}_0)_{i,j} = \left(\overline{D_1}\,\overline{D_2}\,\frac{\partial w}{\partial t} + \overline{D_2}\,\overline{N_1}\,\frac{\delta_1\mu_1}{\delta x_1}F_1^E + \overline{D_1}\,\overline{N_2}\,\frac{\delta_2\mu_2}{\delta x_2}F_2^E + \overline{D_1}\,\overline{D_2}\,\mathcal{S}\right)_{i,j} + O(h^{2p}) \tag{1.32}$$

with the coefficients given in table 1.2. Equation (1.32) is obtained by applying the Padé denominators $\overline{D_1 D_2}$ to the whole left-hand side of the equations. This introduces a second-order error term that vanishes since it only depends on the exact residual, which is null. The 3-D counterpart of (1.32) is derived in a similar way and can be found in [57]. For $2p = 4$ on a 3×3 stencil, all parameters are fixed, then leading to a unique scheme. For $2p = 6$, a family of schemes is found but the choice given in table 1.2 is the one retain in [56] for its suitability with the discretization of viscous terms. For $2p = 8$ on a 5×5 stencil, a unique scheme is found again. In the case of an unsteady computation, the scheme is intrinsically implicit and for the centered residual r_0, the mass matrix issued from $D_1 D_2$ should be inverted. Yet, the presence of a similar term in the dissipation makes the inversion much more complicated and it is preferred to solve the residual in a dual time framework (see section 1.2.5), as proposed in [57, 56]. Then, the time derivative is then treated as a source term.

Concerning the viscous terms, in this thesis we restrict to second-order accurate approximations, provided by the Padé fractions

$$\left(\frac{\partial F_d^V}{\partial x_d}\right)_{i,j,k} = \left(\frac{I}{\prod_f \overline{D_f}}\frac{\delta_d}{\delta x_d}F_d^V\right) + O(\delta x_d^2) \tag{1.33}$$

where the formal polynomial operators $\overline{D_d}$ are defined by (1.31) and the viscous fluxes are approximated at cell faces. For a two dimensional system, the main residual, *i.e.* the centered approximation of the system, writes

$$(\tilde{r}_0)_{i,j} = \left[\frac{\delta_1}{\delta x_1}\left(\overline{D_2}\,\overline{N_1}\mu_1 F_1^E - F_1^V\right) + \frac{\delta_2}{\delta x_2}\left(\overline{D_1}\,\overline{N_2}\mu_2 F_2^E - F_2^V\right) + \overline{D_1}\,\overline{D_2}\,\mathcal{S}\right]_{i,j} \tag{1.34}$$

Residual-based dissipation

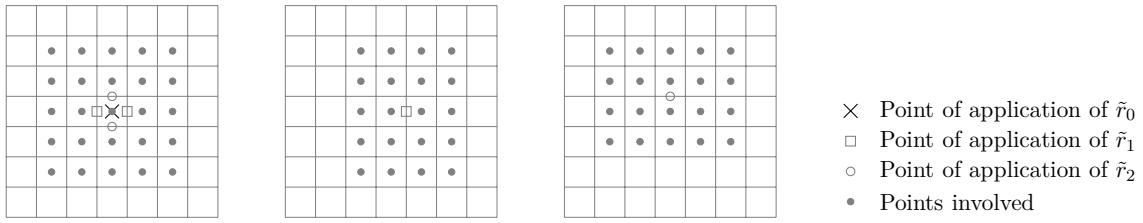
A centered numerical scheme cannot be used without a numerical dissipation to bring parabolicity for stability considerations. Usually, a dissipation of the same order of accuracy than the space operator discretization or higher is added. In the original paper introducing RBC schemes (of order two and three [112]), the choice is made to introduce a dissipative term of order one – to obtain a robust scheme – that vanishes at steady state, so that the dissipation operator becomes high-order accurate at steady state. This is achieved by designing a dissipation expressed only by means of approximations of the residual derivatives. The construction of the dissipation is based on previous work on second-order accurate schemes of the Lax-Wendroff type and on a so-called characteristic time-step matrix [95]. The resulting dissipative operator is parameter free and provides oscillation free solutions without limiters or any other correction for second- and third-order accurate schemes (see [117] for stability and shock capturing properties of RBC schemes). Furthermore, it is independent on the time step used to reach the steady state.

Equation (1.29) is completed by a dissipative term of the form

$$\tilde{\Phi} = \frac{1}{2}\sum_d \Big(\delta_d(\phi_d \tilde{r}_d)\Big) \tag{1.35}$$

with $\tilde{r}_d$ an approximation of the residual shifted to the face in the mesh direction d. In the RBC language, they are called mid-point residual and constructed similarly to the main residual. The preceding operator is consistent with a first-order term which vanishes when the residual goes to zero, plus a high-order term.

We have seen that the approximation of the first derivatives in the main residual are based on Padé fractions with an even number of operators (either difference and average operator). This leads to a so-called standard Padé operator approximating the derivative at the same type of topological elements as the degrees of freedom used for the approximation (see appendix A). We recall that the state vector is stored at cell-centers in this work. Since the mid-point residual are defined on cell faces, new operators should be considered. Two kinds of derivative approximation corresponding to two differentiation directions should be defined: we will denote them as shifted direction and cross direction (relative to the shifted direction). The main residual and mid-point residuals are sketched in figure 1.1. The size of the stencil of the main residual corresponds to the one of the fifth- and seventh-order accurate RBC schemes and the stencil of the mid-point residuals are the biggest centered stencils obtainable that lie in the main residual stencil. For the



(a) $\tilde{r}_0$ at (i, j) (b) $\tilde{r}_1$ at $(i + \frac{1}{2}, j)$ (c) $\tilde{r}_2$ at $(i, j + \frac{1}{2})$

Figure 1.1: Sketch of the stencils of high-order RBC schemes in 2-D

derivative approximation in the shifted direction, *e.g.* in direction $d = 1$ at point $(i + 1/2, j, k)$ or $d = 2$ at point $(i, j + 1/2, k)$, a staggered approximation of the first derivative should be introduced as

$$\frac{\partial F_d}{\partial x_d} = \frac{N_d^\delta}{D_d^\delta}\frac{\delta_d F_d}{\delta x_d} + O(\delta x_d^{2p-2}) \tag{1.36}$$

and for the cross direction, for instance the derivation of a field in direction $d = 1$ at point $(i, j + 1/2, k)$ or in direction $d = 2$ at point $(i + 1/2, j, k)$, the standard Padé fraction can be used. But to build a dissipation at order $2p - 1$, the shifted residual should be evaluated at order $2p - 2$. Then, new coefficients for such derivative approximation are introduced as (a, b, c) for the formal operators N_d, D_d with a definition similar to (1.31). To recover the information from the cell centers, an average operator for a smooth function f discretized on the mesh is defined as

$$\tilde{f} = \frac{N_d^\mu}{D_d^\mu}\mu_d f + O(\delta x_d^{2p-2}) \tag{1.37}$$

where N_d^δ, D_d^δ and D_d^μ are formal polynomials of second difference operators defined by

$$\begin{aligned} N_d^\delta &= I + a^\delta \delta_d^2\,, & D_d^\delta &= I + b^\delta \delta_d^2 + c^\delta \delta_d^4 \\ N_d^\mu &= I + a^\mu \delta_d^2\,, & D_d^\mu &= I + b^\mu \delta_d^2 + c^\mu \delta_d^4 \end{aligned} \tag{1.38}$$

In order to fit the stencil of the dissipation inside the stencil of the main residual while keeping the desired order of accuracy, we impose for each direction d the denominators to be equal: $D_d^\delta = D_d^\mu$. Finally, the same trick as for the main residual is used, that is to multiply the approximation of the residual by the denominators of the fraction operators so as to obtain in three dimensions of space for a steady computation of the homogeneous system

$$\begin{aligned}(\tilde{r}_1^{\text{steady}})_{i+\frac{1}{2},j,k} = \Bigg[& \frac{\delta_1}{\delta x_1}\Big(N_1^\delta D_2 D_3 F_1^E - \mu_1 F_1^{V_1}\Big) \\ & + \frac{\delta_2}{\delta x_2}\Big(N_1^\mu N_2 D_3 \mu_1 \mu_2 F_2^E - \mu_2 F_2^{V_1}\Big) \\ & + \frac{\delta_3}{\delta x_3}\Big(N_1^\mu N_3 D_2 \mu_1 \mu_3 F_3^E - \mu_3 F_3^{V_1}\Big) \Bigg]_{i+\frac{1}{2},j,k} + 0(h^{2p-2})\end{aligned} \tag{1.39}$$

where F^{V_d} are viscous fluxes F^V evaluated at the center of cell face in the mesh direction d, and similar expressions for $\tilde{r}_2$ and $\tilde{r}_3$ derived by circular permutations of the subscripts $\{1,2,3\}$, the flux components $\{F_1, F_2, F_3\}$ and the space steps $\{\delta x_1, \delta x_2, \delta x_3\}$. The sets of possible coefficients are given in [56] for steady flow problems. When it comes to unsteady simulations, the dissipative properties of RBC schemes are to be carely studied. In [84], the application of the so-called χ-criterion leads to following treatment of the time derivative to be added to the steady terms in the residual

$$(\tilde{r}_1)_{i+\frac{1}{2},j,k} = \left[\tilde{r}_1^{\text{steady}} + D_2 D_3 N_1^\mu \mu_1 \frac{\partial w}{\partial t}\right]_{i+\frac{1}{2},j,k} + 0(h^{2p-2}) \tag{1.40}$$

and again similar expressions for $\tilde{r}_2$ and $\tilde{r}_3$ derived by circular permutations.

Characteristic time-step matrix

The functions ϕ_d for all directions d should ensure the stability of the time-dependent scheme. Namely, they should be chosen so that the operator P is dissipative. The matrix functions ϕ_d have the same eigenvectors as J_d. Using notations of section 1.1.1, the matrices ϕ_d are defined as

$$\forall d \in [\![1, D]\!],\ \phi_d = R(x_d, w)\Lambda_{CTS}(x_d, w)R(x_d, w)^{-1} \tag{1.41}$$

with Λ_{CTS} (for characteristic time-step) a modified diagonal matrix of eigenvalues. Writing $\lambda_{CTS}^{(i)}$ its eigenvalues, $\lambda^{(i)}$ the eigenvalues of Λ, we obtain

$$\forall d \in [\![1, D]\!], \qquad \begin{cases} \lambda_{CTS,d}^{(i)} = sgn(\lambda_d^{(i)})\psi_d^{(i)} \\ \psi_d^{(i)} = \min\left(1,\ \min\limits_{q\neq d}\left(\dfrac{\delta x_q\,|\lambda_d^{(i)}|}{\delta x_d\, m(J_q)}\right)\right) \\ m(J_d) = \min\limits_i(|\lambda_d^{(i)}|) \end{cases} \tag{1.42}$$

where we use the notation $\lambda_d^{(i)}$ for $\lambda^{(i)}(x_d, w)$, the i-th eigenvalue of Λ in the direction d.

Coefficients of the Padé operators

We close the section on the discretization of the space derivatives and complete the RBC formulation with table 1.2 providing the coefficients to be used for the RBC schemes of order 3,

5 and 7, as well as summary expressions of the schemes in formula 1.43 to 1.48.

$$\begin{aligned} \overline{N_d} &= I + \bar{a}\delta_d^2\,, & \overline{D_d} &= I + \bar{b}\delta_d^2 + \bar{c}\delta_d^4\,, \\ N_d^\delta &= I + a^\delta\delta_d^2\,, & N_d^\mu &= I + a^\mu\delta_d^2\,, \\ N_d &= I + a\delta_d^2\,, & D_d &= I + b\delta_d^2 + c\delta_d^4 \end{aligned} \tag{1.43}$$

$$\left[\overline{D_1}\,\overline{D_2}\,\overline{D_3}\frac{\partial w}{\partial t} + \frac{\partial H_1}{\partial x_1} + \frac{\partial H_2}{\partial x_2} + \frac{\partial H_3}{\partial x_3} + \overline{D_1}\,\overline{D_2}\,\overline{D_3}S\right]_{i,j,k} = 0 \tag{1.44}$$

$$\begin{aligned} (H_1)_{i+\frac{1}{2},j,k} &= \overline{D_2}\,\overline{D_3}\,\overline{N_1}\mu_1 F_1^E - F_1^V + \frac{1}{2}\phi_1\Pi_1 \\ (H_2)_{i,j+\frac{1}{2},k} &= \overline{D_1}\,\overline{D_3}\,\overline{N_2}\mu_2 F_2^E - F_2^V + \frac{1}{2}\phi_2\Pi_2 \\ (H_3)_{i,j,k+\frac{1}{2}} &= \overline{D_1}\,\overline{D_2}\,\overline{N_3}\mu_3 F_3^E - F_3^V + \frac{1}{2}\phi_3\Pi_3 \end{aligned} \tag{1.45}$$

$$\begin{aligned} (\Pi_1)_{i+\frac{1}{2},j,k} = \Big[&\left(N_1^\delta D_2 D_3 F_1^E - \mu_1 F_1^{V_1}\right) + \left(N_1^\mu N_2 D_3 \mu_1\mu_2 F_2^E - \mu_2 F_2^{V_1}\right) \\ &+ \left(N_1^\mu N_3 D_2 \mu_1\mu_3 F_3^E - \mu_3 F_3^{V_1}\right) + D_2 D_3 N_1^\mu \mu_1 \frac{\partial w}{\partial t} \Big]_{i+\frac{1}{2},j,k} \end{aligned} \tag{1.46}$$

$$\begin{aligned} (\Pi_2)_{i,j+\frac{1}{2},k} = \Big[&\left(N_2^\mu N_1 D_3 \mu_1\mu_2 F_1^E - \mu_1 F_1^{V_2}\right) + \left(N_2^\delta D_1 D_3 F_2^E - \mu_2 F_2^{V_2}\right) \\ &+ \left(N_2^\mu N_3 D_1 \mu_2\mu_3 F_3^E - \mu_3 F_3^{V_2}\right) + D_1 D_3 N_2^\mu \mu_2 \frac{\partial w}{\partial t} \Big]_{i,j+\frac{1}{2},k} \end{aligned} \tag{1.47}$$

$$\begin{aligned} (\Pi_3)_{i,j,k+\frac{1}{2}} = \Big[&\left(N_3^\mu N_1 D_2 \mu_1\mu_3 F_1^E - \mu_1 F_1^{V_3}\right) + \left(N_3^\mu N_2 D_1 \mu_2\mu_3 F_2^E - \mu_2 F_2^{V_3}\right) \\ &+ \left(N_3^\delta D_1 D_2 F_3^E - \mu_3 F_3^{V_3}\right) + D_1 D_2 N_3^\mu \mu_3 \frac{\partial w}{\partial t} \Big]_{i,j,k+\frac{1}{2}} \end{aligned} \tag{1.48}$$

Table 1.2: Coefficients of the RBC scheme for the order 3, 5 and 7

Accuracy	$\overline{a}$	$\overline{b}$	$\overline{c}$	a^δ	a^μ	a	b	c
3^{th} order	0	1/6	0	0	0	0	1/6	0
5^{th} order	1/10	4/15	1/90	1/6	1/12	1/10	4/15	1/90
7^{th} order	5/42	2/7	1/70	11/60	1/10	5/42	2/7	1/70

1.2.5 Discretization of the time derivative

The temporal domain $[0, t_f]$ is subdivided by a sequence of $(M+1)$ discrete time levels $\{t_0 = 0, t_1, \ldots, t_n, t_{n+1}, \ldots, t_{M-1} = t_f\}$. The schemes we consider allow, knowing the solution up to a certain time t_n, the computation of the solution approximation at time t_{n+1}. We consider both steady and unsteady computations.

Steady computations

When the goal of a computation is to reach a steady state, the best way is not to use a physical discretization of the time derivative but instead to solve the semi-discretized problem (meaning a problem in which only the space derivatives are discretized). Then, in order to solve this semi-discretized problem, either a time-marching method or a direct resolution can be used. Concerning time-marching method, a backward Euler solver is used to converge the numerical solution using local computation of the time-step under the CFL constraint (see further below). As a consequence, we focus our attention on the generic space-time slab $\Omega \times [t_n, t_{n+1}]$. The time-width of the slab is given by the time-step $\Delta t = t_{n+1} - t_n$. The corresponding approximation of the state vector increment is denoted by $(\Delta\tilde{w})^n$. To speed-up convergence, an implicit treatment is employed. In this work, several different choices for left-hand side (LHS) operator are considered. First, a matrix-free algorithm – which can be found in [194, 122, 54] – can be used. This is achieved by replacing H_d^n with H_d^{n+1} in equation (1.24) and leads to

$$(\Delta\tilde{w})_i^n + \left[\sigma_d \delta_d \Delta H_d\right]_j^n = \Delta w^{exp} \tag{1.49}$$

with $\Delta w^{exp} = -(\delta_d H_d/\delta x_d)_{i,j,k}^n$ the explicit operator and $\sigma_d = \Delta t/\delta x_d$. This matrix-free algorithm can be considered as the use of an inviscid numerical flux which is not consistent in the general case with the one use in the explicit operator. This inviscid numerical flux is chosen very simple to simplify the implicit treatment. It is chosen as the Rusanov flux $(H_d)_{i+1/2} = (\mu_d F_d^E - 1/2\rho_d^E \delta_d w)_{i+1/2}$ and using the following approximation of the inviscid numerical fluxes

$$H_d^E = H_d^E(w_L, w_R) \quad \Rightarrow \quad (\Delta H_d^E)^n \approx \Big(\frac{\partial H^E}{\partial w_L}\Big)\Delta w_L^n + \Big(\frac{\partial H^E}{\partial w_R}\Big)\Delta w_R^n \tag{1.50}$$

and using the approximation of the viscous fluxes

$$(\Delta F^{V_d})^n = \Big(\frac{\partial F^{V_d}}{\partial w_{x_d}}\Big)\frac{\delta}{\delta x_d}(\Delta w^n) = \frac{(J^{V_d})^n}{\delta x_d}\delta\Delta w^n = \frac{(\rho^{V_d})^n I}{\delta x_d}\delta\Delta w^n \tag{1.51}$$

we can rewrite the implicit treatment as a first order implicit stage

$$D\Delta w_i^n + C_d^+ \Delta w_{i+e_d}^n + C_d^- \Delta w_{i-e_d}^n = \Delta w_i^{exp} - \sigma_d(\mu_d J_d^E \delta_d \Delta w^n)_i \tag{1.52}$$

where $e_d = (\delta_i^d)_{i=1,D}$ and $\sigma_d = \Delta t/\delta x_d$ and with

$$\begin{cases} C_d^+ = \sigma_d\Big[(\dfrac{\partial H_d}{\partial w}) + \dfrac{A_d^V}{\delta x_d}\Big] = \Big[\dfrac{1}{2}\sigma_d\rho_d^E I + \dfrac{\sigma_d\rho_d^V}{\delta x_d} I\Big]_{i+e_d/2} \\ C_d^- = \sigma_d\Big[(\dfrac{\partial H_d}{\partial w}) + \dfrac{A_d^V}{\delta x_d}\Big] = \Big[\dfrac{1}{2}\sigma_d\rho_d^E I + \dfrac{\sigma_d\rho_d^V}{\delta x_d} I\Big]_{i-e_d/2} \\ D = I + C_d^+ + C_d^- \end{cases} \tag{1.53}$$

Finally, by solving the matrix-free implicit operator by a point Jacobi treatment, we write

$$\begin{aligned} \Delta w_i^{(l+1)} = D_i^{-1}\Bigg[& \Delta w_i^{exp} - \sigma\delta\mu\Delta(F^E)_i^{(l)} \\ & + \Big(\frac{1}{2}(\sigma_d\rho_d^E)_{i+e_d/2} + (\frac{\sigma_d\rho^v}{\delta x_d})_{i+1/2}\Big)\Delta w_{i+1}^{(l)} \\ & + \Big(\frac{1}{2}(\sigma_d\rho_d^E)_{i-e_d/2} + (\frac{\sigma_d\rho^v}{\delta x_d})_{i-1/2}\Big)\Delta w_{i-1}^{(l)}\Bigg] \end{aligned} \tag{1.54}$$

with the diagonal

$$D_i^{-1} = [1 + \frac{1}{2}(\sigma_d \rho_d^E)_{i-1/2} + \frac{1}{2}(\sigma_d \rho_d^E)_{i+1/2} + (\frac{\sigma_d \rho_d^V}{\delta x_d})_{i-1/2} + (\frac{\sigma_d \rho_d^V}{\delta x_d})_{i+1/2}]^{-1} I.$$

This implicit treatment is then "matrix-free", the matrix D being a diagonal matrix and $C_d^{\pm}$ being scalars depending on the Jacobian spectral radii. This feature provide a low storage algorithm which is usable for complex configurations requiring many degrees of freedom. The work of Corre on the matrix-free implicit treatment [54] indicates that the Jacobi algorithm requires typically five to eight iterations for a third order scheme and around fifteen iterations for fifth- and seventh-order schemes. This is due to the difference of numerical schemes used in the implicit and explicit operators.

A scalar version of the LU-SSOR [193] algorithm is also considered for the implicit treatment. It present many similarities with the matrix-free algorithm and then is not given in details in this chapter.

Let us give some key points on a Jacobian-free Newton-Krylov [104] algorithm. Jacobian-free Newton–Krylov (JFNK) methods are iterative methods consisting of a combination of Newton-type methods for convergent solution of nonlinear equations and Krylov subspace methods for solving the Newton correction equations. The link between the two methods is the Jacobian-vector product, which may be evaluated in an approximate way without forming and storing the elements of the true Jacobian. As mentioned before, an overview of this family of methods can be found in [104]. The purpose of this section is to summarize the main features of these methods. The system defined by (1.24) can be written as:

$$N(w) = 0 \tag{1.55}$$

The Newton iteration for (1.55) derives from a multivariate Taylor expansion about a current point w^n:

$$N(w^{n+1}) = N(w^n) + J(w^{n+1} - w^n) + O(J^2) \tag{1.56}$$

where $J = \left(\frac{\partial N}{\partial w}\right)^n$ refers to the Jacobian matrix of N. Setting the right-hand side to zero and neglecting higher-order terms yields a strict Newton method, and the solution is found by iterating over a sequence of linear systems

$$\forall n \in [\![0, M-1]\!], \quad \begin{cases} J\delta w = -N(w^n) \\ w^{n+1} = w^n + \delta w \end{cases} \tag{1.57}$$

given an initial tentative solution w^0, with $N(w)$ the vector-valued function of nonlinear residuals, w the state vector and n is the iteration index. The Newton iteration is terminated when the required drop in the norm of the nonlinear residual has been reached

$$\frac{\|N(w^n)\|}{\|N(w^0)\|} < \text{tolerance} \tag{1.58}$$

Since the use of an iterative technique to solve equation (1.57) does not require the exact solution of the linear system, the resulting algorithm is categorized as an "inexact" Newton's method [63]. A simple inexact method results in the following convergence criteria on each linear iteration:

$$\|J^n \delta w^n + N(w^n)\| < \gamma \|N(w^n)\| \tag{1.59}$$

where the forcing term γ is a constant smaller than unity. For details on the forcing term see Reference [69].

A Krylov subspace method, as GMRES [156] is used to solve equation (1.57). Let δw_0 be an initial guess, r_0 the initial linear residual defined by $r_0 = -N(w) - J\delta w_0$ and j the Krylov iteration index. δw_j is drawn from the subspace spanned by the Krylov vectors, $\{r_0, Jr_0, J^2 r_0, \ldots, J^{j-1} r_0\}$, and can be written as:

$$\delta w_j = \delta w_0 + \Sigma_i \beta_i J^i r_0 \tag{1.60}$$

where the β_i minimize the residual. As we can see in equation (1.60), Krylov subspace method requires the action of the Jacobian only in the form of matrix-vector products which may be approximated by:

$$Jv \approx \frac{N(w + \epsilon v) - N(w)}{\epsilon} \tag{1.61}$$

Note that, as pointed out by Heng-Bin An *et al.* [20], the use of first order finite difference approximation for the Jacobian-vector product is at least as good as second order finite difference approximation on the Jacobian itself. Finally, as pointed out in Reference [104], JFNK is feasible only through effective preconditioning. We choose to use a scalar version of LU-SSOR algorithm [193] as preconditioner.

For direct resolution of the problem, a numerical evaluation of the Jacobian can be performed as in the Jacobian-free Newton-Krylov algorithm, this being wrapped inside a Newton solver.

Unsteady computations

For fast unsteady problems, it is advisable to use an explicit discretization of the time derivative. A good candidate is an explicit Runge-Kutta scheme. The most famous is the fourth-order accurate scheme proposed by Runge (1895) and Kutta (1901). Its stability domain depends partly on the order condition since the amplification factor for an advection problem writes $G = e^{-ika\Delta t}$ with k the wave number and a the advection velocity. An improvement is to consider a linear low storage version of Hu *et al.* [94]. We also refer to the low-storage nonlinear version of Berland [36]. Obviously an explicit discretization scheme of the time derivative is only possible for explicit schemes. This is the case of the family of FE-MUSCL schemes described in Section 1.2.3. On the contrary, RBC schemes are intrinsically implicit due to the mass matrix formed by the denominators of the Padé operators. For these implicit schemes, explicit treatments by predictor-correction process are still under development but can be found in Reference [148].

For slow unsteady problems, an implicit treatment is preferred allowing for the use a greater time-steps with respect to the physical CFL condition. A Linear Multi-step Method (LMM) of second-order accuracy, writing

$$\left(\frac{\partial w}{\partial t}\right)^{n+1} = \frac{1}{2\Delta t}(3w^{n+1} - 4w^n + w^{n-1}) + O(\Delta t^2) \tag{1.62}$$

which is A-stable, is used to discretize the time derivative. The implicit system is solved by either a dual time stepping technique or inner Newton sub-iterations. The dual time system writes

$$\frac{\Delta w^{n,m}}{\Delta \tau} + \frac{\frac{3}{2}(w^{n,m} - w^n) - \frac{1}{2}\Delta w^{n-1}}{\delta t} + \left[\frac{\delta_d H_d}{\delta x_d}\right]^{n,m} = 0 \tag{1.63}$$

where m is the pseudo-iteration (on dual-time) counter, n is the physical iteration counter, $\Delta w^{n,m} = w^{n,m+1} - w^{n,m}$ and $\Delta w^{n-1} = w^n - w^{n-1}$. This leads to the implicit system

$$\frac{\Delta w^{n,m}}{\Delta \tau} + \frac{3}{2}\frac{\Delta w^{n,m}}{\delta t} + \left[\frac{\delta_d(\Delta H_d)}{\delta x_d}\right]^{n,m} = \Delta w^{exp} \tag{1.64}$$

with the explicit stage written

$$\Delta w^{exp} = -\left[\frac{\delta_d H_d}{\delta x_d}\right]^{n,m} - \frac{\frac{3}{2}(w^{n,m} - w^n) - \frac{1}{2}\Delta w^{n-1}}{\delta t} \tag{1.65}$$

This implicit system is solved by one of the implicit techniques described for steady computations with respect to the dual time. Another way to solve the dual-time system is to employ the Gear's Method consisting in Newton sub-iterations. The Newton method solve $\Gamma(w) = 0$. In our flow simulations, we consider

$$\Gamma(w^n) = \frac{\partial w^n}{\partial t} - \Delta w^{exp}$$

and discretizing the time derivative by the second-order accurate LMM approximation, we obtain after linearization

$$\Big(\frac{3}{2}I + J_{\text{Rhs}}\Big)\Delta w^n = -\Gamma(w^n)$$

with J_{Rhs} the Jacobian of the right-hand side of the system at time n, and I the identity.

Stability criteria

For explicit schemes, the stability study requires that the time step Δt does verify two criteria for the Navier-Stokes equations. The first criteria is governed by the convective terms of the equations and it is named after Courant, Friedrichs and Lewy (hereinafter CFL). This stability condition imposes that the distance covered during time Δt by a perturbation (fluid element and acoustic) propagating at speed $(u_d \pm c)$ be less than the distance between two points of the mesh [60]. A second criteria should be complied to ensure the convergence. It concerns the time-space discretization of the viscous fluxes and is due to the viscosity and thermal-conduction effects. Given a characteristic distance h in a cell (i, j, k), these criteria can be written

$$\Delta t = \text{CFL}\ \min(\Delta t_{\text{euler}}, \Delta t_{\text{viscous}}) \tag{1.66}$$

with

$$\Delta t_{\text{euler}} = \frac{h}{\|u\| + c}\,, \qquad \Delta t_{\text{viscous}} = \frac{h^2\, P_r\, Re\, \rho}{2\,\gamma\,\mu} \tag{1.67}$$

1.3 Chapter summary

We have introduced the mathematical problem at the core of this work with the physical models for compressible flows that close the system for both inviscid and viscous formulations. This was the place to introduce the notation in use throughout this document. The discretization on structured mesh with storage of the state vector at cell-centers was defined. The conservative formalism is adopted for the design of numerical approximations of the space derivatives. A family of directional non-compact schemes denoted FE-MUSCL was presented with a range of order of accuracy from 3 to 7. Then some details on the construction of the family of RBC schemes was provided. These schemes are based on compact approximations of the derivatives, on a residual-based dissipation and a characteristic-time step matrix to ensure the dissipation of the scheme in all flow conditions. Finally, we review the numerical methods in use for either the convergence of steady problems or the time integration of unsteady flows.

1.4 Résumé du chapitre (french)

Nous avons introduit le problème mathématique au coeur de ce travail, accompagné des modèles physiques pour les écoulements compressibles de manière à fermer le système d'équations à résoudre, que ce soit pour une formulation visqueuse ou non-visqueuse. Ce premier chapitre est aussi le lieu pour introduire les notations utilisées tout au long de ce document. La discrétisation sur maillage structuré avec stockage du vecteur d'état au centre des cellules a été présenté. Le fomalisme des schémas numériques sous forme conservative est adopté pour la conception des approximations des dérivées spatiales. Une classe de schémas non-compacts et directionnels dénommé FE-MUSCL a été rappelé pour des ordres de précision allant de 3 à 7. Puis, le principe de construction de la famille des schémas RBC a été présenté. Ces schémas numériques sont basés sur une approximation compacte des dérivées spatiales, sur une dissipation basée sur le résidu et une matrice à pas de temps caractéristique de manière à assurer des propriétés dissipatives des schémas dans toutes les conditions d'écoulement. Finalement, le chapitre se termine par une brève revue des méthodes utilisée pour la convergence des problèmes stationnaires ou pour l'intégration en temps des problèmes instationnaires.

Chapter 2

A new dynamic code architecture for CFD computations – DynHoLab code

"Computer science is no more about computers than astronomy is about telescopes."

Edsger Dijkstra (1930 - 2002)

Contents

Introduction (french)

L'objectif de ce chapitre est de fournir un cadre général pour développer facilement et rapidement ainsi que tester de nouveaux modèles et algorithmes numériques pour la mécanique des fluides numérique : ceux-ci comprennent des schémas de discrétisation, des conditions aux limites, des modèles de gaz réels, des modèles de turbulence. La principale caractéristique de ce code est un environnement de calcul indépendant des modèles spécifiques ou des méthodes numériques utilisées pour résoudre le problème.

Une exigence essentielle pour ce programme est qu'il doit être conçu de manière à faire face aux évolutions inévitables dans les modèles physiques et méthodes numériques avec un effort de développement minimal. Pour ce faire, nous nous appuyons sur une structure de données unique qui est le seul flux en entrée et sortie de chaque méthode du code. Cela évite la création d'interfaces de méthodes qui sont congelés ou du moins difficiles à modifier sans interventions massives et intrusives dans le code.

Le nouveau code devrait donc permettre des développements rapides grâce aux objets de haut niveau issu du paradigme orienté objet. Les nouvelles fonctionnalités devraient être ajoutées au programme sans qu'il soit nécessaire de mettre en œuvre des méthodes directement dans le noyau de code.

En bref, le but de ce travail est de développer un outil interopérable et évolutif (même au moment de l'exécution, "à la volée"), de développer et de tester des modèles physiques et des stratégies numériques. Dans ce sens, le code représente une "laboratoire virtuel" pour de nouveaux développements théoriques et numériques. Pour atteindre ces objectifs, nous choisissons d'utiliser le langage Python pour assurer le développement rapide de la structure du code et l'interopérabilité des modules, et le langage Fortran pour ses performances en calcul intensif et CGNS comme procédé de stockage de données, la structure de données étant séparé du code.

Dans la suite de ce chapitre, nous décrivons la philosophie sous-jacente à l'architecture de ce projet de code, ainsi que ses caractéristiques les plus importantes. Ensuite, on décrit l'état du code de calcul en termes de modèles et de méthodes qui sont actuellement disponibles. Certaines applications numériques illustrant les capacités de code sont présentés au chapitre 3. La plupart du contenu de ce chapitre a été publié dans Réf [130].

Introduction

The objective of this chapter is to provide a general framework to easily and quickly develop and test new models and numerical algorithms for CFD: these include discretization schemes, boundary conditions, real-gas models, turbulence models. The main feature of the code is a computational environment independent on the specific models or numerical methods used to solve the problem.

One crucial requirement for this code is that it should be designed in such a way as to face unavoidable evolutions in CFD models and numerical methods with a minimal development effort. For this purpose, we rely on an unique data structure that is given as input and output to each single method of the code. This avoids the creation of method interfaces that are frozen or at least difficult to modify without massive and intrusive code modifications.

The new code, then, should enable fast development and modification along with high level objects and an object-oriented paradigm. New features should be added to the program without any need for implementing methods directly in the code kernel.

In short, the aim of this work is to develop an inter-operable and evolving tool (even at run time, “on the fly”), to develop and test physical models and numerical strategies. In this sense, the code represents a virtual “laboratory” for new theoretical and numerical developments. To achieve these objectives, we choose to use Python language for fast development of the code structure and module interoperability, Fortran for computation performance and CGNS as data storage method, data structure being separated from the code.

In the following of this Chapter, we describe the underlying philosophy of the proposed code architecture, as well as its most important features and then provide the status of the associated computational code in terms of models and methods that are presently available. Some numerical applications illustrating the code capabilities are presented in Chapter 3. Most

of the contents of this Chapter have been published in Ref [130].

2.1 Development of a dynamic code architecture: the DynHoLab code

This section provides an overview of the code architecture. This makes use of two basic concepts of Python programming: *modules*, *i.e.* files containing methods, and *packages*, *i.e.* directories containing at least one module named `__init__.py`.

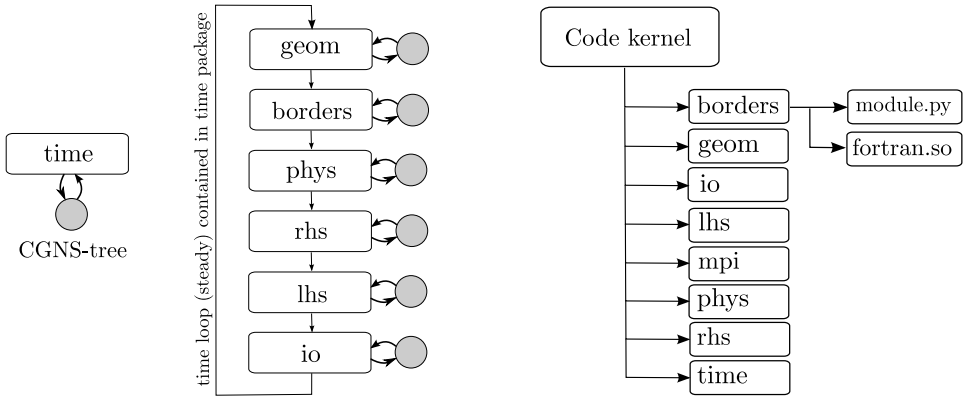
2.1.1 Programming languages

DynHoLab is coded using a combination of a compiled and type-safe language with an interpreted and dynamically typed language: this way, it benefits from the fast and safe execution of the first one, along with easy and fast development capabilities thanks to the second one. Because FORTRAN is traditionally used among the numerical computation community, it has been chosen as the compiled language. The second language could have been either Java or Python. The Python community is growing fast in the numerical simulation field. Thanks to projects like *numpy* [180] (numerical libraries), *scipy* [100] (mathematics, science and engineering libraries), *matplotlib* [96] (plotting library), *sage* [167] (mathematics software) and many others, Python is a very good candidate to develop a CFD kernel. It is free and open-source, and several libraries and design patterns are made available thanks to its community-based development model.

One of the drawbacks of Python is its execution time that is obviously longer than that of compiled languages. Thus, the computational code has to be designed so as to migrate time consuming operations to FORTRAN routines while keeping high level programming in the Python part.

2.1.2 Design of a code for CFD computations

In order to illustrate the basic concepts of DynHoLab code architecture, we consider, to fix ideas, a typical iteration loop used to solve a steady CFD problem, represented in Figure 2.1(a). In general, the iterative procedure can be seen as a loop in time, contained either in the main



(a) Sketch of a typical steady computation loop (b) Organisation in packages and modules

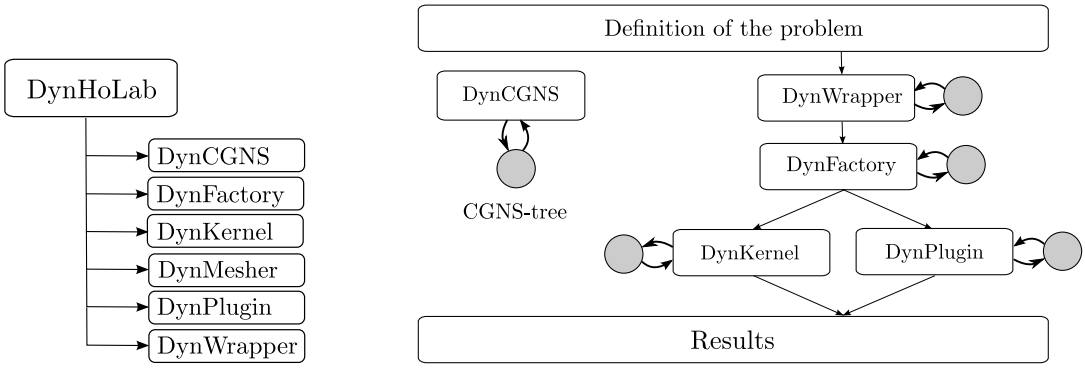
Figure 2.1: Code ingredients: kernel, packages, modules and data-structure

program or in a package. The time-loop makes use of other code parts, or packages, to perform all the needed computations. Each package called by the time-loop communicates data to a data-storage structure organized under the form of a CGNS tree, as discussed in detail in Section 2.1.3. The interest of organizing the code under the form of Python packages is an excellent code modularity; moreover, every code component may be imported by other programs to be re-used. A main file governs the execution and schedules package calls. Each package is composed of both Python modules (*i.e.* files) and dynamic libraries containing time-consuming methods. These are obtained from Fortran routines that are wrapped in Python by using the wrapper f2py [7]. All Fortran dynamic libraries required for a given operation are gathered together in the corresponding package (see *e.g.* Fig. 2.1(b)). Each sub-package corresponds to a class that manages a list of methods for either the initialization or the production phase (sometimes both). The code can be easily updated with new features by adding new Python sub-packages.

Going back to the preceding example, sub-packages required for the steady CFD computation are:

— geom: gathers methods related to geometry (compute volumes, normals, distances to walls...)
— borders: gathers methods related to the application of boundary conditions and mesh joins
— phys: gathers methods related to physics for different systems of equations: convection and diffusion of physical fluxes, sources terms and related sub-models (equations of state, turbulence models, ...)
— rhs: gathers methods related to the computation of the right-hand-side of the equation (compute the explicit increment)
— lhs: gathers methods related to the computation of the left-hand-side of the equation (compute the implicit increment)
— io: gathers methods related to inputs and outputs.
— time: gathers methods related to the time (time step computation) and the time solvers (steady, unsteady).

The macroscopic code architecture, in terms of packages and sub-packages, is represented schematically in Figure 2.2(a). It is based on several macro-packages, each one composed by sub-packages. The articulation among different macro-packages is depicted in Figure 2.2(b). Problem definition is input to the code through some user interface, then the problem is wrapped



(a) Organisation of DynHoLab in packages and sub-packages

(b) Global architecture of DynHoLab

Figure 2.2: Global architecture of the code

to a data-structure chosen as a CGNS-tree containing all needed information to run the computation. The wrapping procedure is carried out by the package *DynWrapper* but all the required data is stored in the CGNS-tree. The CGNS implementation is located in package *DynCGNS*. This package provides methods to organize the data, to access elements in the CGNS-tree and other utilities. *DynCGNS* is independent and separated from the implementation of numerical methods. However it is accessed and used by every other subpackage in the code. After the wrapping step, the methods required for the specific problem are found and stack in lists of methods. These can belong either to the code kernel *DynKernel* or to a newly added plug-in contained in a separate package, named *DynPlugin*, that is discussed later. This work is achieved by the package *DynFactory*. This factory is designed to easily integrate any plug-in. The lists of methods created in the factory are themselves stored in the CGNS-tree in order to avoid the storage inside the factory. At this point of the execution, the data-structure contains all needed information to be executed. The execution runs through the code *Kernel* and potential plug-ins required for the calculation.

2.1.3 Data storage

An important concept used by the code is separation of data storage from the different methods required by the solution algorithm (e.g. calculation of the physical flux function, calculation of the explicit increment, system solution, solution update...). The data-structure flows through different methods (see Figure 2.3), is modified within them, and flows out.

This concept is somewhat opposed to the use of the Object-Oriented (OO) paradigm. The latter is often used to provide data encapsulation, arguing that it breaks the complexity by grouping data and functions together and protecting the data from accidental corruption. The main drawback of a code design based on the OO paradigm is that it provides a static public interface that cannot tackle future evolutions of the methods easily. By creating a separated data-structure on which each method acts, the need for a public interface is suppressed since the data-structure is used as both input and output of every single method. This is what we call the concept of "flux". Moreover, thanks to this inter-operable structure, any other program sharing the same data-structure can be glued to the CFD solver in order to solve a more complex problem (e.g. pre- and post-treatment, uncertainty quantification, shape optimization, fluid-structure interactions, ...).

The data-structure chosen for DynHoLab is a CGNS-tree, implemented through two classes (*tree* and *node*) and providing a full hierarchical structure to store the data. Since this data-structure has been developed by a consortium (ADAPCO, ANSYS, Boeing, NASA, ONERA, Rolls-Royce, Standford University, Tecplot, US Air Force,...), it is being adopted by more and more computer codes and related programs. The use of CGNS in the code is inspired of one of the implementations of the CGNS standard [140, 153] written in python language [139].

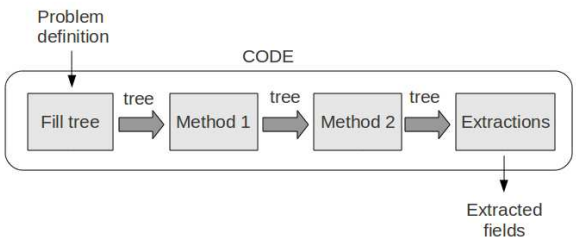


Figure 2.3: Sketch of a code using a tree data-structure as a flux

2.1.4 Concept of plug-in

One of the requirements underlying the design of DynHoLab is to be able to add a new feature without modifying the code *Kernel*. To this purpose, the concept of plug-in and hook is used.

Figure 2.4 shows that a sub-package is composed by a class and a list of methods. These methods are not encapsulated in the class and they act directly on the CGNS-tree as explained in the previous section. The class is only composed of two methods related to initialization and production. Each of this two methods of the class is linked to a list of methods prepared beforehand. The methods used within these two lists can be either located in the sub-package itself, or can be a plug-in that follows the DynHoLab frame for plug-ins.

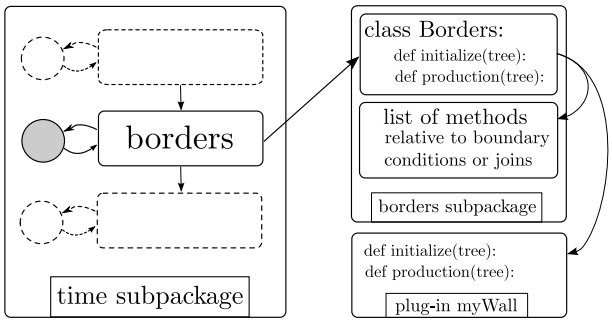


Figure 2.4: Sketch representing the architecture of a subpackage. Methods related to the initialization and production phase of a feature can either be located in the code *Kernel* or in a plug-in

To fix ideas, consider for instance the class *Borders* contained in *DynKernel* (fig. 2.4). This contains a list of methods corresponding to the boundary conditions available in the minimal code delivery. One can imagine of coding a new boundary condition, called *myWall*, and to plug it to the code. It is then possible to develop a numerical method acting on the CGNS-tree without coding in the *Kernel* itself (example of the boundary condition *myWall* on Figure 2.4). Using this feature, only fundamental numerical methods have been implemented in the code kernel. The idea is that the code *Kernel* contains a minimum of packages required to run basic CFD computations, and that more specific models or numerical methods are always implemented as plug-ins. Moreover it is an easy way to share numerical methods between collaborators. The user is then free to add any method, even dedicated to a single test case.

2.1.5 Parallel computations

The code has also been developed to enable parallel computing. At this stage of development, this includes two levels of parallelization. The first one is a multiprocessing feature for parametric computations. This needs almost no communication between processors. This kind of capability can be used, *e.g.* for non-intrusive uncertainty quantification or shape optimization [91]. The second level of parallelization is achieved through a MPI implementation using domain decomposition. This ability is based on the existing multi-block implementation of the CFD kernel. The execution path is represented on Figure 2.5. During the initialization of the problem, one processor plays the role of master. The master processor reads the definition of the problem, cuts the computational domain, the boundary conditions among blocks, and creates new conformal joins resulting of the domain decomposition. The mesh is not loaded in memory during this step. The balancing is achieved through the mesh index management. At the end

of the initialization process, sub-domains are stored as new CGNS zones (*i.e.* a CGNS node representing the domain considered that have child nodes containing all needed data for the computation) that are subsequently sent to each processor. From this point of the execution on, all processors play the same role and run the numerical initialization and the computation. Note that *io* sub-package of *DynKernel* does not yet handle parallel writing in a single file.

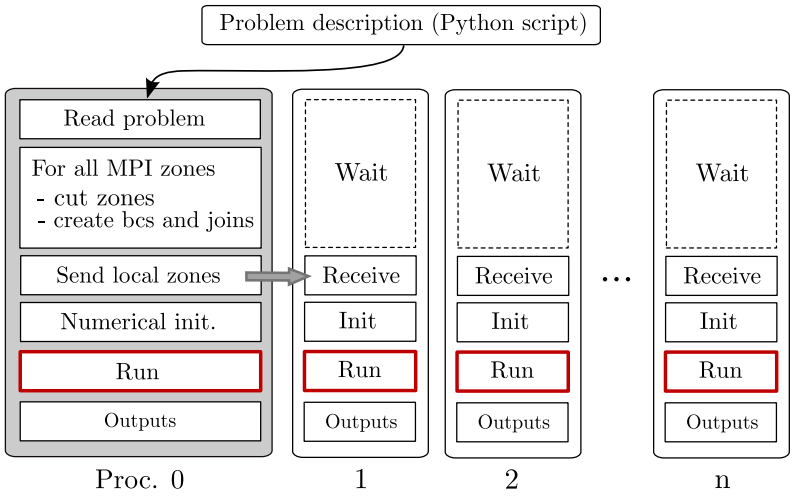


Figure 2.5: Sketch of a MPI computation on DynHoLab

Scalability study

A common way to assess the performance of a code in high-performance computing (HPC) is to measure the scalability (also referred to as the scaling efficiency) of the application. This consist in computing the turnaround time of the application as a measure indicating how efficient the application is when using increasing numbers of parallel processing elements. There are commonly two ways to characterize the parallel performance of a code, depending on the configuration of the computation, that is whether it is CPU-bounded or memory-bounded. These two terms are defined hereafter. In the case of a CPU-bounded computation, it is more pertinent to study the strong scalability of the application and in the case of a memory-bounded computation, the weak scalability.

Weak scalability Let us consider the weak scaling first. We consider for instance a problem that would not fit in the RAM of a single node. A way to overcome this problem is to divide the problem and execute it on multiple processing elements. This type of measurement is justified for programs that take a lot of memory or other system resources (something that is memory-bound). The weak scalability should inform on the minimum size of workload to be assigned to each processing elements to get a constant turn around time. For the study, the problem size assigned to each processing element stays constant and additional elements are used to solve a larger total problem. In the case of weak scaling, linear scaling is achieved if the run time stays constant while the workload is increased in direct proportion to the number of processors. Most programs running in this mode should scale well to larger core counts as they typically employ nearest-neighbor communication patterns where the communication overhead is relatively constant regardless of the number of processes used. However, some exceptions include algorithms that employ heavy use of global communication patterns, eg. FFTs and transposes. If the amount of time to complete a work unit with 1 processing element is t1, and

the amount of time to complete N of the same work units with N processing elements is tN, the weak scaling efficiency (as a percentage of linear) is given as

$$E_{weak} = \frac{t_1}{t_N} * 100\% \tag{2.1}$$

A characterization of DynHoLab code using weak scaling study is presented on figure 2.6(a), 2.6(c), 2.6(e). This study is lead with a MUSCL-type scheme [49, 113] of third order of accuracy in three dimensions of space. This correspond to the cheapest scheme implemented (among high-order schemes) which correspond to the worst case for a scalability study. On figure 2.6(c), we observe that a minimum amount of processing units are required to reach a plateau. This could be explained by the fact that when using very few processing units, not all faces of blocks are concerned by parallel communication. We conclude on the weak scaling limit being around 50^3 degrees of freedom per block for DynHoLab code.

Strong scalability Let us now consider the strong scaling. This scaling is to be considered when the limitation for a bigger problem to be considered is the cpu time, for programs that take a long time to run. The problem is said to be cpu-bounded. We then fix the problem size but increase the number of processing elements. The goal in this case is to find an equilibrium that allows the computation to complete in a reasonable amount of time, yet no wasting time in parallel communications. We search for a program to scale linearly, that is for the speedup to be equal to the number of processing elements used (N). The speedup is measured in terms of work units completed per unit time. In general, it is harder to achieve good strong-scaling at larger process counts since the communication overhead for many/most algorithms increases in proportion to the number of processes used. If the amount of time to complete a work unit with 1 processing element is t_1, and the amount of time to complete the same unit of work with N processing elements is t_N, the strong scaling efficiency (as a percentage of linear) is given as

$$E_{strong} = \frac{t_1}{N * t_N} * 100\% \tag{2.2}$$

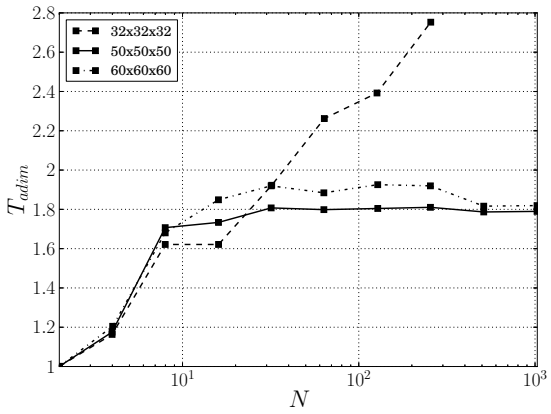
The strong scalability study of DynHoLab is presented on figures 2.6(b), 2.6(d) and 2.6(f). This study is lead in three dimensions of space with the same scheme used for the weak scalability study. It shows a satisfactory behavior of the code for both few and large number of processing units.

2.2 Present status of the DynHoLab code

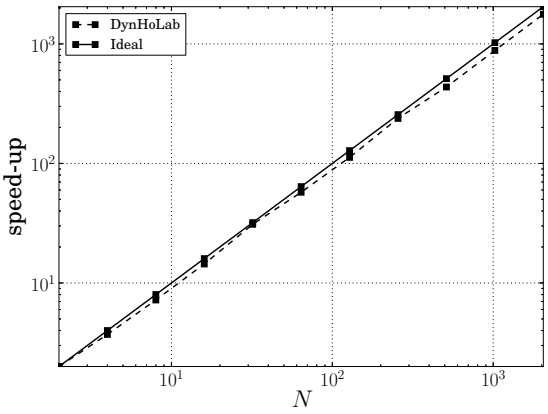
In this section we describe the flow models, approximation schemes and solvers presently available in DynHoLab and implemented in the framework of this thesis. Some other functionalities are under development in the context of parallel thesis/projects and will be discussed in the conclusions.

2.2.1 Flow models

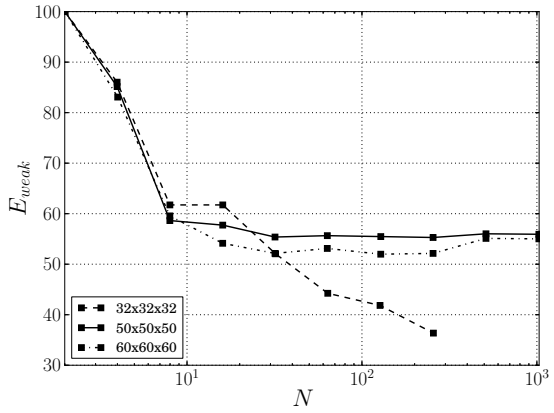
DynHoLab can handle different governing equations both in two and three dimensions of space. To this purpose, it is sufficient to specify the nature of the unknowns w, of the physical fluxes F and source terms S of Eq. (1.1). For instance, by specifying $w \in \mathbb{R}$, $F = aw$ with $a \in \mathbb{R}$ and $S = 0$ one recovers a linear scalar advection equation. In the following, we present numerical results for 3-D linear advection and for 2-D and 3-D compressible turbulent flows obtained just



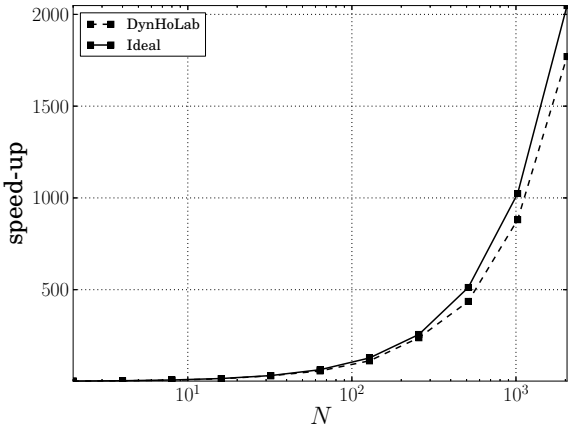
(a) Weak scalability: Time per iteration per cell as a function of the processing units



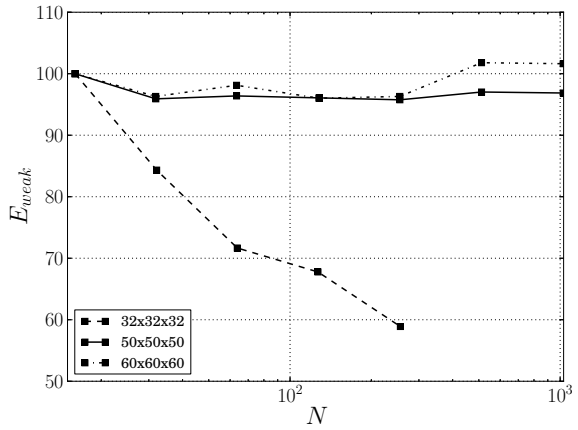
(b) Strong scalability: Speed-up as a function of the processing units (log-log)



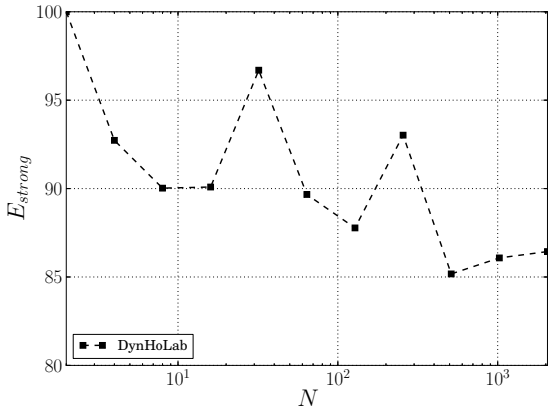
(c) Weak scalability: Efficiency normalized with respect to 2 processing units



(d) Strong scalability: Speed-up as a function of the processing units (log-normal)



(e) Weak scalability: Efficiency normalized with respect to 16 processing units



(f) Strong scalability: Efficiency as a fonction of the processing units

Figure 2.6: Scalability of DynHoLab

by changing the specifications for w, F and S in the code. In the case of the Navier-Stokes equations, the general equations are supplemented by several sub-models. Namely, one needs to

specify an equation of state describing the gas thermodynamic behavior and constitutive laws for the viscous stresses and the heat flux. For turbulent flows, these also include the contributions of the Reynolds stresses and of the turbulent heat flux, if a Reynolds-Averaged approach for turbulence modeling is selected.

Precisely, in the applications shown in next chapters, we make use of the ideal gas equation of state as the thermodynamic model, of the constitutive law for viscous stresses in Newtonian fluids, of Fourier's law for heat flux, and of Menter's k-ω-SST turbulence model [126]. In DynHoLab, it is possible to solve the RANS equations both as a coupled and an uncoupled system. Details of the implementation used for the turbulence equations has been published in Ref. [53]. Note that it is possible to choose and implement any other equation of state or turbulence model in use without changing anything of the code structure.

2.2.2 Space discretisations

Equation (1.1) is discretized in space on a set of m structured or unstructured grids by means of some approximation technique, to give

$$w_t + \mathcal{R}(w) = 0 \tag{2.3}$$

where $\mathcal{R}$ is a space approximation operator. $\mathcal{R}$ may be obtained by applying any approximation method in use, including finite differences, finite volumes or finite elements. The semi-discrete equation (2.3) represents a set of N ordinary differential equations, N depending on m, and on the number of degrees of freedom, control volumes or grid points contained in each grid.

At the present stage of development, the DynHoLab code uses a finite volume methodology on multi-block structured meshes. The numerical fluxes may be approximated by means of different numerical schemes, including the first-order accurate Roe scheme [151], the second-order Jameson [97] scheme, as well as a class of directional high-order upwind schemes [49, 113], referred-to as FE-MUSCL (see Section 1.2.3), and the Residual-Based Compact (RBC) schemes [112, 56] described in Section 1.2.4. For both FE-MUSL and RBC schemes, we implemented different orders of accuracy, ranging from 3 to 7. Also the dispersion-relation preserving (DRP) schemes [37] on 9, 11 and 13 points with their selective filters are presently implemented.

2.2.3 Time stepping schemes

Concerning time-stepping methods, all the schemes presented in Section 1.2.5 are currently implemented in DynHoLab. To sum-up, for steady computations, a backward Euler solver is used to converge the numerical solution. To speed-up convergence, several implicit treatments are included: a matrix-free [54] algorithm, a LU-SSOR [193] algorithm and a Jacobian-free Newton-Krylov [104] algorithm. For unsteady computations, it is possible to choose between an explicit Runge-Kutta scheme (the low storage version of Hu *et al.* [94], an optimized version in the wave space [37], and the low-storage non-linear version of Berland [36]) and a second-order backward linear multi-step scheme (Gear's scheme), solved by either a dual time stepping technique or by inner Newton sub-iterations.

2.2.4 Multiblock grid strategy

Based on the data storage tree and on the functional architecture of the code, DynHoLab is constructed intrinsically as a multi-block solver. Specifically, the code handle both conformal and overlapping multiblock meshes. This approach serves several goals:

— it enables the computation of complex geometries,

— it can be used as a basic domain decomposition strategy for parallel computations,
— it may be used to cope with moving bodies,
— it may be used for mesh adaptation.

Details of the treatment of conformal joins is described in Appendix B. Concerning the overlapping strategy, a specific chapter is dedicated to its management and implementation. We invite the reader to refer to chapter 4 for a complete discussion on this subject.

2.3 Chapter summary

A new numerical environment for CFD computations, written in Python and FORTRAN languages, was presented. This numerical framework is designed to allow the use of finite differences, finite volumes or finite elements spatial discretizations. The approximation strategy currently implemented in the code is a multiblock structured finite-volume framework. The code's architecture uses the concept of separation of data storage from the numerical methods implementation. The data structure follows the CGNS standard, implemented using Python language. The parallel implementation and scalability study was detailed. The code allows taking into account very different flow physics within the same structure. Validations and computational results are shown in the next chapter.

This code intent to provide a easy and versatile framework for research. It has been developed in collaboration with Dr Content, research engineer at the DynFluid Laboratory. As an evidence of the implanting of this work, two theses which are being conducted in the laboratory uses the DynHoLab code for their new developments or numerical simulations. The first one concerns the study of dense gas and lead to the generalization of the physics module (precisely the equation of state) during its first year. The second one is interested in uncertainty quantification on ORC turbines. A third thesis, lead at ONERA and dealing with high-order finite-volume schemes as well as no-match joins for coincident non-overlapping grids, is conducted within the DynHoLab code. These developments are created as plug-ins to the code making collaboration easier.

2.4 Résumé du chapitre (french)

Un nouvel environnement numérique de travail pour les calculs en MFN, implémenté en Python et FORTRAN, a été présenté. Cet environnement de calcul est conçu pour permettre l'utilisation indifférente des méthodes de différences finies, volumes finis ou éléments finis pour la discrétisation spatiales du problème abordé. La stratégie actuellement implémentée dans le code est une stratégie volumes finis sur maillages structurés multi-blocs. L'architecture du code de calcul utilise le concept de séparation des algorithmes numériques du stockage des données. La structure de donnée suit la norme CGNS mais localement implémentée en Python. L'implémentation parallèle et l'étude des performances en terme de scalabilité du code a été détaillé. Le code permet de prendre en compte des physiques très différentes dans la même structure. Les validations et résultats numériques sont présentés dans le chapitre suivant.

Ce code tente de fournir un environnement de recherche facile et souple. Il a été développé en collaboration avec le Dr Content, ingénieur de recherche au laboratoire DynFluid. En guise de démonstration de l'implantation de ce travail, deux thèses actuellement conduites dans le laboratoire utilisent le code DynHoLab pour leurs développements ou pour leurs simulations numériques. La première concerne l'étude de gaz denses et mène à la généralisation du module de physique (précisément l'équation d'état) pendant sa première année. La seconde s'intéresse à la quantification des incertitudes dans les turbines ORC. Une troisième thèse, menée à l'ONERA et traitant des schémas de volumes finis d'ordre élevé et de raccord d'ordre élevé de maillage

non-coincidant est également conduite dans le code DynHoLab. Les développement sont tous créés comme plug-ins du code rendant la collaboration plus aisée.

Chapter 3

Preliminary validations on conformal meshes

Contents

Introduction (french)

Le présent chapitre est consacré à la validation de méthodes numériques mis en œuvre dans DynHoLab. A cet effet, on considère une série de cas tests de complexité croissante. Un problème linéaire scalaire est utilisée pour vérifier les ordres de précision des schémas numériques. Les propriétés de capture de choc ainsi que l'efficacité des traitements implicites, sont ensuite étudiés pour un problème scalaire non linéaire (équation de Bürgers).

Ensuite, la performance des schémas d'ordre élevé est évaluée grâce à des comparaisons avec d'autres codes CFD sur différents écoulements autour de profils NACA0012. Puis, le solveur RANS est validé pour un écoulement turbulent sur une plaque plane 2-D, un écoulement transsonique sur l'aile NACA64A10, et un écoulement sur l'aile transsonique ONERA M6. Enfin, une simulation d'écoulement ILES sur une colline périodique 2-D est réalisée pour étudier les capacités de notre solveur face à un cas 3-D instationnaire complexe?

Introduction

The present chapter is devoted to the validation of numerical methods implemented in DynHoLab. For this purpose, we consider a series of test cases of increasing complexity. A linear scalar problem is used to check the orders of accuracy of the numerical schemes. Shock capturing properties, as well as the efficiency of the implicit treatments, are then investigated for a nonlinear scalar problem (Bürgers equation). Then, the performance of high-order schemes is assessed through comparisons with other CFD codes on various flows over the NACA0012 airfoil. Afterwards, the RANS solver is validated for a 2-D turbulent flow over a flat plate, a transonic flow over the NACA64A10 airfoil, and a transonic flow over the ONERA M6 wing. Finally, an ILES simulation of the flow past a periodic 2-D hill is carried out to investigate the capabilities of our solver for complex 3-D unsteady flows.

3.1 Linear advection problems

In this Section, we present some numerical validations for both steady and unsteady advection test cases. This allow to verify the orders of accuracy and error levels of the numerical schemes implemented in DynHoLab, with focus on the FE-MUSCL and RBC schemes.

3.1.1 Helicoidal advection of a Gaussian pulse

The first test case is a steady scalar problem, namely, 3-D helicoidal linear advection of a Gaussian pulse, taken from references [114, 56]. Specifically, we solve the following partial-differential equation

$$\frac{\partial u}{\partial t} + z\frac{\partial u}{\partial x} + \frac{1}{10}\frac{\partial u}{\partial y} - x\frac{\partial u}{\partial z} = 0, \tag{3.1}$$

on the domain defined by $(x, y, z) \in]-1; 0[\times]0; 1[\times]0; 1[$, with the following initial and boundary conditions

$$\begin{cases} u(x,y,z,0) = 0,\ x \in]-1,0[,\ y \in]0,1[,\ z \in]0,1[\\ u(x,y,0,t) = \exp\big(-50((x+\frac{1}{2})^2 + (y+\frac{1}{2})^2)\big), \quad x \in [-1,0],\ y \in [0,1],\ t \geqslant 0 \\ u(-1,y,z,t) = 0,\ y \in [0,1],\ z \in [0,1],\ t \geqslant 0 \\ u(x,0,z,t) = 0,\ x \in [-1,0],\ z \in [0,1],\ t \geqslant 0 \\ u(x,1,z,t) = 0,\ x \in [-1,0],\ z \in [0,1],\ t \geqslant 0 \\ u(x,y,1,t) = 0,\ x \in [-1,0],\ y \in [0,1],\ t \geqslant 0 \end{cases} \tag{3.2}$$

Equation (3.1) is a PDE of the form (1.1), with $n = 1$, $w = u$, $d = 3$ and $F = (zu, 1/10u, -xu)$. The initial field set on the $z = 0$ plane is both rotated around and advected along the y-axis. The exact steady solution for the scalar w is constant on its helicoidal characteristics. Steady solutions are obtained from an initial uniform field. The computational domain is discretized by regular Cartesian meshes of increasing density. Figure 3.1(b) shows iso-values of the solution obtained with a fifth-order RBC scheme on a 60^3 grid. Computations were performed using a series of Cartesian meshes and using both FE-MUSCL schemes and RBC schemes of several orders of accuracy. The Cartesian meshes used for the convergence of the error have size $(20^3, 30^3, 40^3, 50^3, 60^3, 70^3)$ expressed as numbers of cells. Since we use regular Cartesian grids for this steady problem, we expect to recover the nominal convergence order upon mesh refinement. Figure 3.1(a) shows that the L_2-norm of the error with respect to the exact solution converges with a slope that is found equal to the nominal convergence order, which validates the present implementation of the spatial discretization.

In order to investigate the impact of the discretization of the right-hand side of the equation on the convergence properties of the implicit schemes, we plot in Figures 3.1(c) and 3.1(d) the convergence of the residual and the convergence of the error with respect to the analytical solution as a function of the iterations. These computations are led with a 80^3 grid. The inversion of the left-hand side of the equation is achieved by a scalar LU-SSOR method with a CFL number of 100. FE-MUSCL schemes show a faster convergence of the residual compared to the RBC schemes (beware of the x-axis that is a log scale for the RBC scheme plot). Despite their first order dissipation during the transient phase, RBC schemes need a greater convergence of the residual to reach convergence of the error. The implicit numerical method used is a scalar version of the LU-SSOR algorithm (please refer to Chapter 1).

As a way to characterize and compare the efficiency of the schemes, we plot in Figure 3.1(e) the error obtained at steady state as a function of the work unit (made non-machine dependent

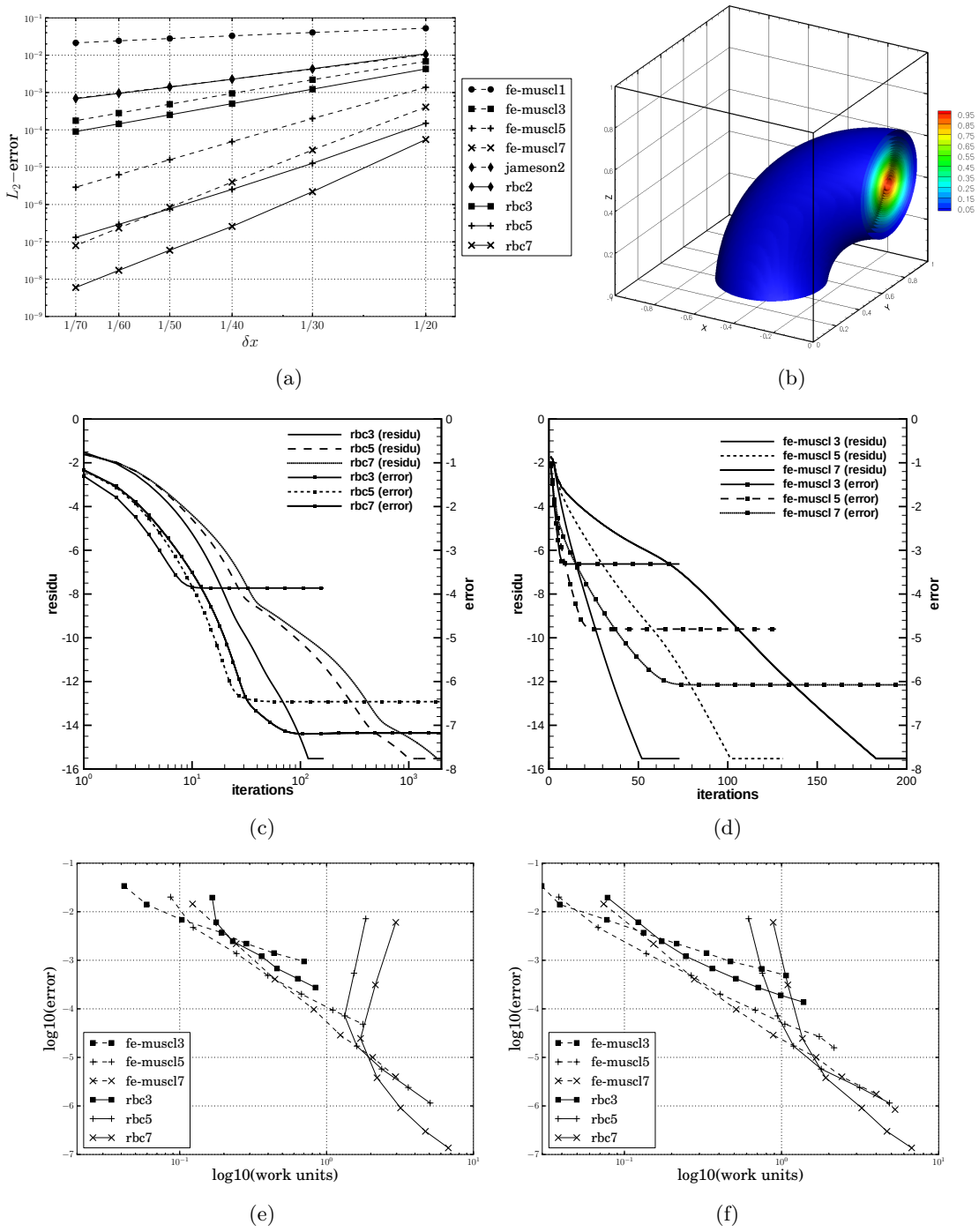


Figure 3.1: 3-D helicoidal advection (a) Convergence of the L_2-norm of the error upon mesh refinement, (b) iso-values of u obtained with 5^{th} order RBC scheme on a 60^3 grid, (c) Convergence of the error and residual for RBC schemes on a 80^3 grid, (d) Convergence of the error and residual for FE-MUSCL schemes on a 80^3 grid, (e) Efficiency of the schemes for a drop of 10 order of magnitude of the residual, (f) Efficiency of the schemes for converged errors.

with the TauBench tool). The work unit is computed for a drop of 10 orders of magnitude of the residual. On this plot, each point corresponds to one computation. A series of grid starting from $(10+n)^3$ cells is employed with $n \in 0, 5, 10, 15, ...$ By looking for a given level of error, it is

easy to compare the cost of the schemes in terms of CPU time. And when fixing the work units, we compare the error level given by the schemes. On very coarse grids, the residual used in RBC discretizations is slower to converge than for more refined grids. To get a better evaluation of the cost of the schemes, we plot in Figure 3.1(f) the same errors as a function of the work units required to reach convergence of the error. It can be seen that the FE-MUSCL schemes are more efficient on coarse grids than the RBC scheme of the same order of accuracy. In order to obtained error levels smaller than $10^{-2.5}$, the RBC scheme of order 3 should be preferred to the FE-MUSCL scheme of the same order of accuracy. We get a similar situation with FE-MUSCL schemes of order 5 and 7: The order 5 should be preferred for error levels bigger than $10^{-3.5}$ and order 7 for lower error levels. For a little turnaround time, RBC schemes of order 5 and 7 should not be employed. For coarse meshes, FE-MUSCL scheme of order 7 seems to show the same behavior as the RBC scheme of order 5.

3.1.2 Circular advection of a hump

Circular advection of a hump

The second test-case is a linear unsteady problem, and precisely the linear advection, with rotation around point $(x, y) = (0, 0)$, of a smooth hump over the square domain $[-1, 1]^2$, expressed as the following initial value problem [61, 56]

$$\begin{cases} \forall (x,y) \in [-1,1]^2, \quad \dfrac{\partial w}{\partial t} - 2\pi y \dfrac{\partial w}{\partial x} + 2\pi x \dfrac{\partial w}{\partial y} = 0 \\ \forall r \leq 0.25, \quad w(x,y,0) = \cos^2(2\pi r) \\ \forall r > 0.25, \quad w(x,y,0) = 0 \end{cases} \tag{3.3}$$

The hump is initialized at $t = 0$ at point $(-0.5, 0)$. The results presented on figures 3.2(a) to 3.2(d) correspond to simulations up to time $t = 1$, when the exact solution of the problem returns to its initial position. All the schemes are applied on a 30×30 grid and time integration is based on the second-order Gear scheme, solved by Newton sub-iterations with a physical time-step $\Delta t = 0.0025$. At each physical iteration, the L_2-norm of the sub-iteration residual normalized by its initial value is driven down to 10^{-4}. All boundary values are obtained from extrapolation of the interior values.

Figure 3.2(a) shows the shape and iso-contours of the solution computed with the seventh-order RBC scheme at four positions. Figure 3.2(b) presents a comparison of solutions computed by the third- and seventh-order RBC schemes. Clearly, the seventh order scheme improves simulation results, especially in terms of dispersion. Figure 3.2(d), which displays the time evolution of the maximum value of the solution, illustrates the lower dissipation introduced by the high-order schemes upon low-order ones. In the exact solution, the maximum value remains constant in time. It can be highlighted that RBC schemes are always less dissipative than the FE-MUSCL schemes of the same order of accuracy. On Figure 3.2(c), it can be observed that the directional approach of the implicit stage of the scheme implies a higher number of sub-iterations when the direction of the flow is diagonal. To check the spatial order of accuracy of the schemes for an unsteady problem, we compute the error with respect to the exact solution at time $t = 0.05$ for solutions computed on a series of increasingly fine grids. The time steps are chosen as $\Delta t \propto \delta x^{o/2}$ with o the expected spatial order of accuracy, to rule out the second-order error due to the time approximation scheme. The errors are plotted in Figure 3.2(f) versus the work units required for each simulation. It can be noticed that RBC schemes are more expensive than FE-MUSCL schemes on coarse grids but provide better efficiencies on fine grids.

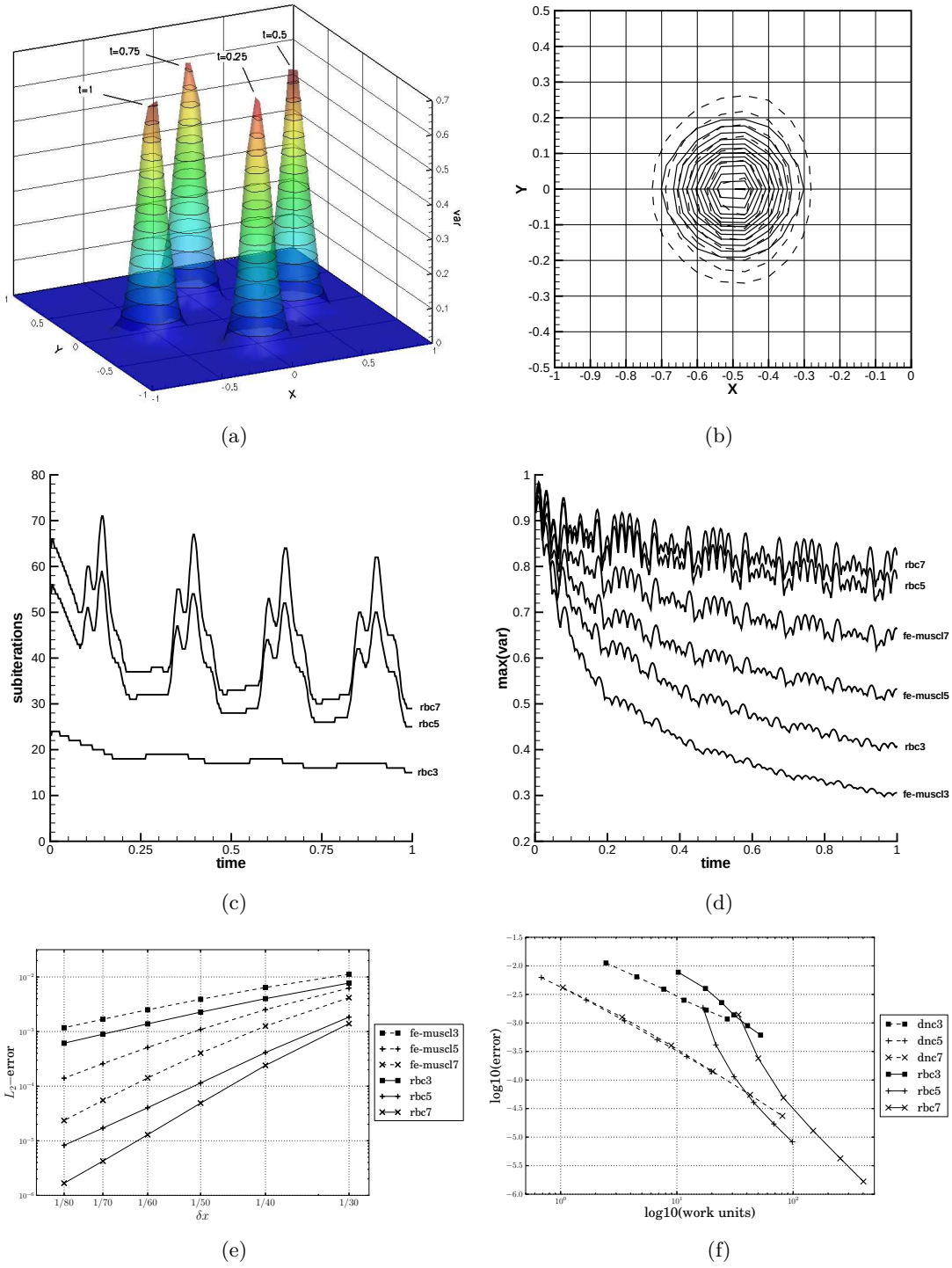


Figure 3.2: Circular advection of a hump, (a) Shape and iso-values (from 0.05 to 0.6 by 0.05) of the hump at four positions computed with RBC7, (b) Iso-values of the hump at t=1 computed with RBC3 (dashed) and RBC7 (solid), (c) Evolution of the number of sub-iterations, (d) Evolution of the maximum value of the hump, (e) Convergence of the L_2-norm of the error upon mesh refinement, (f) Efficiency of the schemes plotted as the error upon work units required.

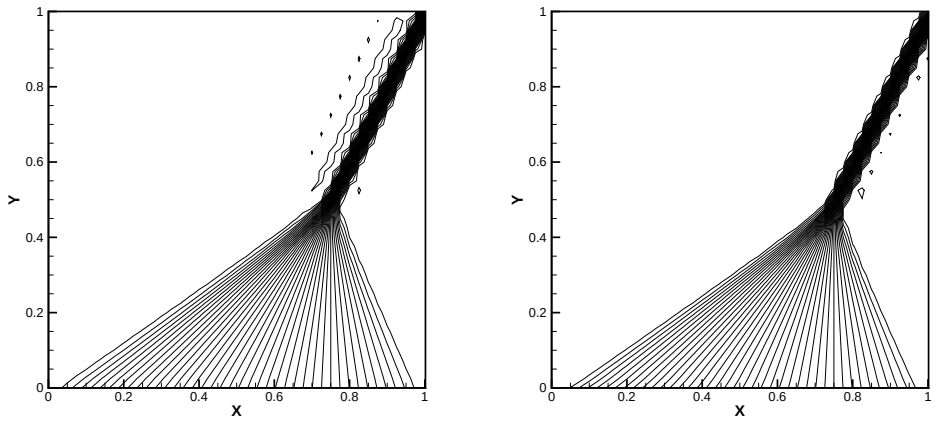
3.1.3 Steady inviscid solution of the 2-D Bürgers equation

Next, we solve the 2-D non-linear Bürgers equation over the square domain $[0,1] \times [0,1]$

$$\begin{cases} \forall (x,y) \in]0,1[^2, & \dfrac{\partial w}{\partial t} + \dfrac{\partial w^2/2}{\partial x} + \dfrac{\partial w}{\partial y} = 0 \\ \forall (x,y) \in [0,1]^2, & w(x,y,0) = (1-x)w_l + xw_r \\ \forall x \in [0,1], t \geq 0, & w(x,0,t) = (1-x)w_l + xw_r \\ \forall y \in [0,1], t \geq 0, & w(0,y,t) = w_l > 0 \\ \forall y \in [0,1], t \geq 0, & w(1,y,t) = w_r < 0 \end{cases} \tag{3.4}$$

The values of $w_l = 1.5$ and $w_r = -0.5$ lead to the formation of a shock wave. The computations are conducted on a 40×40 mesh with FE-MUSCL schemes and RBC schemes. No shock correction is applied in either case. Contours of the solution obtained with RBC7 and FE-MUSCL7 are plotted respectively in Figure 3.3(a) and 3.3(b). Comparing these two plots, it can be conclude that the FE-MUSCL scheme oscillates more than the RBC schemes. This is confirmed by the 1-D cuts at $y = 0.75$ shown in Figs. 3.4(a) and 3.4(b).

Steady solutions have been obtained by using different implicit stages presented in chapter 1, the scalar LU-SSOR and JFNK algorithms. For JFNK, we consider a restarted version of GMRES [23] as the inner Krylov subspace algorithm, preconditioned with a LU-SSOR inversion. The matrix-vector product is evaluated with a first-order forward scheme. The tolerance of the inexact Newton algorithm is chosen as in [68] with an initial value of 0.9. The difference of efficiency between the two methods should be reduced by working on the optimization of the implementation of the JFNK algorithm. Figures 3.4(c) and 3.4(d) shows the convergence history of the L_2-norm of the residual during the iteration process. The number of iterations of the JFNK algorithm required to achieve a drop of ten decades of the residual is far smaller than those required by the scalar LU-SSOR algorithm. The work units required by both algorithms in conjunction with different RBC schemes are reported in Tab. 3.1. The values are normalized with respect to an RBC7 computation with LU-SSOR. Despite the considerably lower number of iterations to converge , the CPU cost of JFNK is higher that that of LU-SSOR. A more optimized implementation of JFNK could improve this result and will be studied in the future.

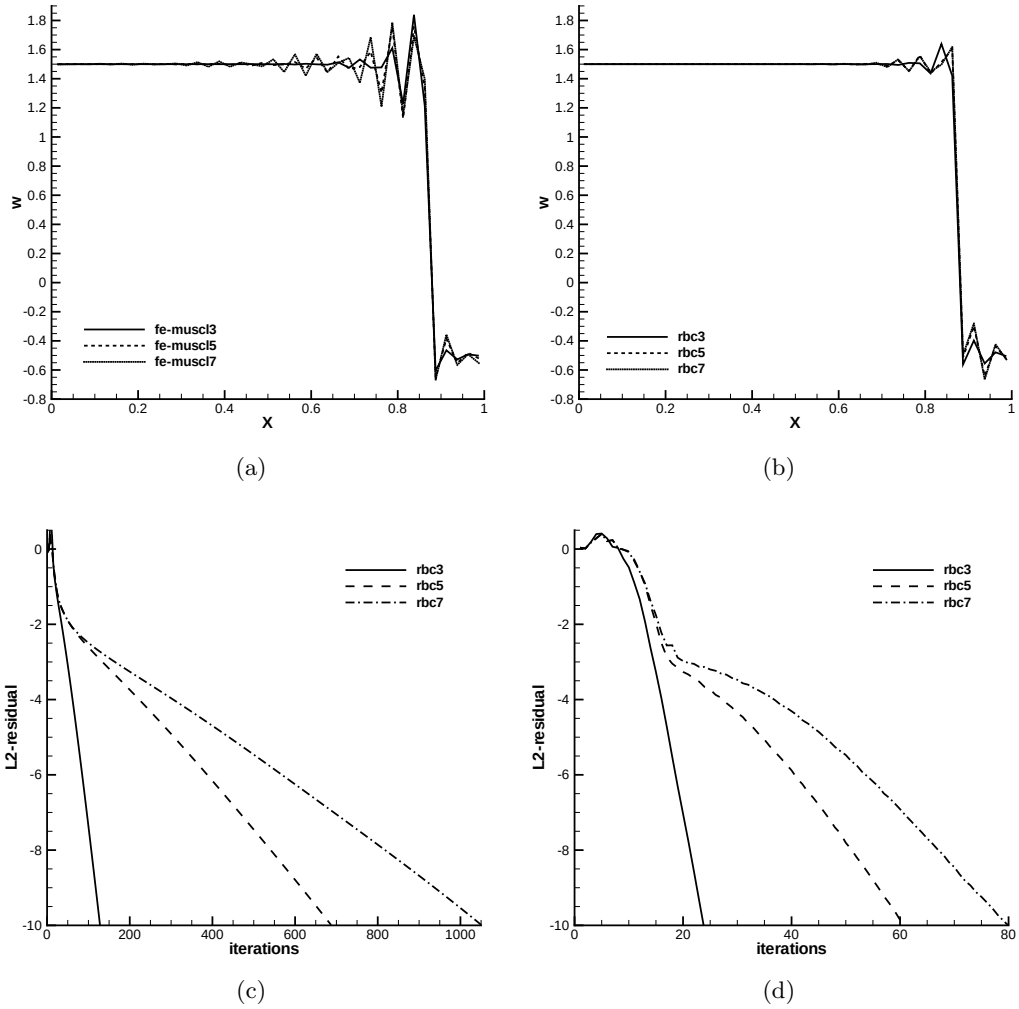


(a) (b)

Figure 3.3: 2-D inviscid Bürgers equation, (a) Contours obtained with FE-MUSCL7, (b) Contours obtained with RBC7.

Table 3.1: Inviscid Bürgers equation: work units required by the scalar LU-SSOR algorithm and JFNK algorithm to achieve a drop of ten decades of the L_2-norm of the residual.

	LU-SSOR	JFNK
RBC3	0.12	0.23
RBC5	0.63	0.85
RBC7	1.00	1.13



(a) (b)

(c) (d)

Figure 3.4: 2-D inviscid Bürgers equation, (a) Section at $y = 0.75$ with FE-MUSCL schemes, (b) Section at $y = 0.75$ with RBC schemes, (c) Convergence of the L_2-norm of the residual with LU-SSOR, (d) Convergence of the L_2-norm of the residual with JFNK.

3.2 Compressible flow over the NACA0012 airfoil

The test-cases considered in this section are taken from the 1st and 2nd International Workshop on High-Order CFD Methods [13, 14]. They are named cases 1.3 a, b and c in these references and represent different compressible flow regimes around a NACA0012 airfoil. The airfoil geometry is the NACA-0012 airfoil defined by the following equation

$$\forall x \in [0,1], \quad y = \pm 0.6(0.2969\sqrt{x} - 0.1260x - 0.3516x^2 + 0.2843x^3 - 0.1015x^4). \tag{3.5}$$

The airfoil defined using this equation has a finite trailing edge 0.252%. We modify the x^4 coefficient in the previous equation such that the trailing edge has a zero thickness. The geometry profile equation is taken as

$$\forall x \in [0,1], \quad y = \pm 0.6(0.2969\sqrt{x} - 0.1260x - 0.3516x^2 + 0.2843x^3 - 0.1036x^4). \tag{3.6}$$

The structured meshes in use are described in details in each subsection. This problem has been often used for testing high-order methods for the computation of external flows with a high-order curved boundary representation. In this work, we use high-order finite-volume methods with no high-order representation of the boundary. Both inviscid and viscous, subsonic and transonic flow conditions will be simulated. For all cases, we compute the lift and drag coefficients for several choices of the spatial order of accuracy and grid resolution. Looking back to the results of the workshops, we can say that this exercise consists in comparing the speed of convergence to the reference solution provided by the code. Indeed, the reference values for lift and drag coefficients of the participants agree only up to the third or fourth decimal, but each participant provides convergence errors until seven decimals or more. This means that using the reference values of another participant would change completely the results concerning convergence of lift and drag coefficient and probably exhibit an asymptote around an error level of the order of magnitude of the uncertainty about the reference value. In the following, all calculations are carried out by using LU-SSOR for the implicit treatment.

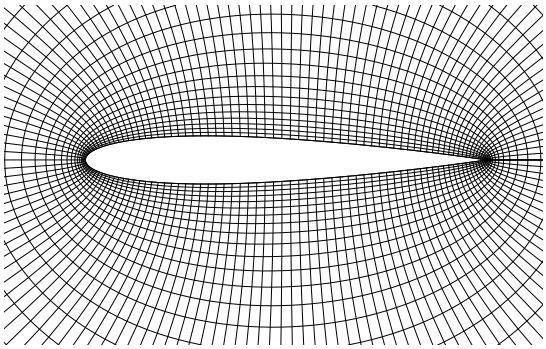
3.2.1 Subsonic inviscid

The governing equations considered are the 2-D Euler equations with a constant ratio of specific heats $\gamma = 1.4$. Concerning the flow conditions, we consider the subsonic inviscid flow at $Mach = 0.5$ and $\alpha = 2^\circ$ angle of attack. We set the appropriate total temperature, total pressure, and flow angle at the inflow and the static pressure at the outflow. For these computations, we first lead a study with a far field defined as a circle, centered at the airfoil mid chord with a radius of 1000 chords as specified in the requirements in the second workshop on high-order methods. The meshes are O-grids created with the elliptic solver contained in Ogen [90]. The meshes are described in Table 3.2 and we provide partial view of grid 3 in Fig. 3.5(a). Our reference values for this test case (RBC3 computations on the finest grid, far field at 50000 chords, see below) are:

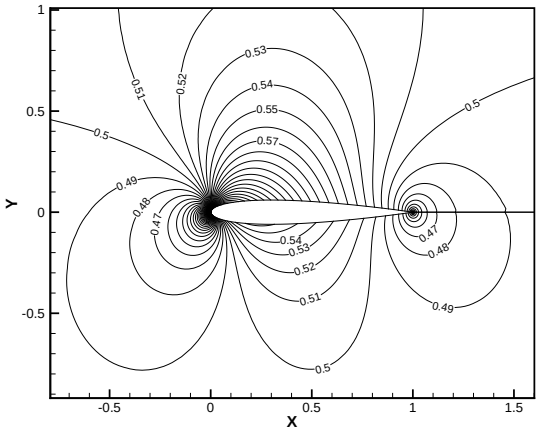
$$C_l = 0.28647753, \qquad C_d = 0.$$

Fig. 3.5 shows typical pressure coefficient and Mach number contours around the airfoil, obtained with scheme RBC5. The pressure coefficient is computed as

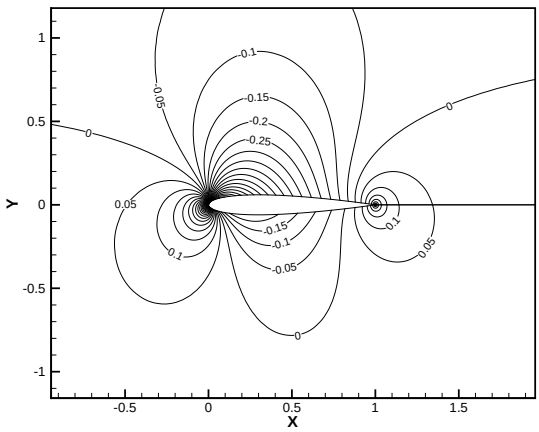
$$C_p = \frac{p - p_\infty}{\frac{1}{2}\gamma p_\infty M_\infty^2} \tag{3.7}$$



(a) O-grid 3 used for inviscid computations (partial view)



(b) iso-lines of the Mach number



(c) iso-lines of the pressure coefficient

Figure 3.5: Subsonic inviscid flow over a NACA-0012 airfoil at $Mach = 0.5$ and $\alpha = 2^\circ$ using RBC5 on mesh 4

Table 3.2: Description of the series of mesh for inviscid computations of NACA-0012

Mesh	Number of cells	Far-Field location (chord unit)
1	702	1000
2	1683	1000
3	4545	1000
4	8305	1000
5	13266	1000
6	25886	1000
7	42907	1000
8	88347	1000

In Fig. 3.6 we provide the convergence of the absolute value of the lift coefficient with respect to the reference solution, as a function of grid refinement. This is computed as $1/\sqrt{ndof}$ with $ndof$ the number of degrees of freedom, which is equal to the number of cells for our method. The left plot in Fig. 3.6 shows the study lead with RBC schemes using the series of mesh described in Table 3.2. We observe a convergence of the lift coefficient with mesh refinement. The nominal

order of accuracy has only a moderate impact on the convergence speed. The reasons for this are the following: - all solutions were compared to a reference value obtained by running RBC3 on the finest mesh with far field at 50000 chords, which is probably not appropriate for RBC5 and RBC7; - the boundary conditions are only second order accurate for all schemes - we not take into account mesh deformations, which decreases the formal order of accuracy. Nevertheless, we do observe a quite significant reduction of the error levels for the lift coefficient, for a given number of DOF, when increasing the scheme order. Our results are compared to those produced with the SBP-SAT method and presented by the Linköping University at the 2nd workshop on high-order methods. These results were obtained with a far-field located at 100 chords. In order to investigate the impact of the far-field location, we provide in the left plot of Fig. 3.6 a convergence of the lift coefficient with the far-field location using the RBC scheme of order three and based on three meshes of Table 3.2. These meshes have a far-field located at x chords with x taken in the set $\{50, 100, 200, 400, 800, 1500, 3000, 10000, 50000\}$. In this second plot, we observe that RBC3 has a similar convergence behavior as the SBP-SAT schemes for a far-field located at 100 chords. Specifically, error levels stagnate around values of the order of 10^{-2}-10^{-3}. On the other hand, we remark that the far-field has an extremely significant impact when looking at a convergence levels up to five decimals, much greater than the impact of the inner scheme. A very large grid, with a far-field boundary located at more than 10000 chords is required to achieve convergence levels of the order of 6 decimal digits. In Fig. 3.7, we compare the RBC3 results (solutions converged with respect to the far-field location, several grid densities) with the values presented for Tau and DG methods by the DLR at the 1st and 2nd workshop on high-order methods. The lift coefficient levels obtained with RBC3 for a converged far-field location are comparable to the ones obtained with the discontinuous Garlerkin method of the same order of accuracy (DG P2). Finally, Fig. 3.8(a) compares convergence of aerodynamic coefficients predicted by RBC schemes on a 1000-chord mesh with the TAU code. It can be seen that RBC schemes allow a significant improvement of convergence levels for a given computational cost. Nevertheless, the use of RBC schemes of high order does not really lead to any improvement for this case. This is due to the previous reasons, plus the fact that we use the same implicit operator for all schemes, which is not optimal to ensure quick convergence at higher orders.

3.2.2 Transonic inviscid

We consider the inviscid transonic flow over a NACA0012 at $Mach = 0.8$ and $\alpha = 1.25^\circ$ angle of attack. Our reference values for this test-case (obtained with RBC3 on the finest grid with far field at 50000 chords) are:

$$C_l = 0.3529651, \qquad C_d = 0.0226556$$

Typical convergence histories in terms of the aerodynamic coefficients are shown in Fig. 3.9. Fig. 3.10 provides an overview of the Mach number and pressure coefficient contours for this case, characterized by a shock wave at the upper surface. RBC schemes provide sharp and non oscillatory shocks without any need for limiters or artificial viscosity, as shown in 1-D by Lerat [117]. Fig. 3.11 illustrates the convergence of the aerodynamic coefficients to the reference values as a function of mesh refinement. The use of high order schemes does not improve substantially the lift coefficient with respect to standard method as that of code TAU. RBC schemes of higher order tend to converge somewhat more quickly than RBC schemes of low order. On coarse grids, error levels for RBC schemes are higher than those of SBP-SAT schemes, but the trend is inverted for finer grids, since the error of SBP-SAT tends to stagnate. In terms of work units (Fig. 3.12), RBC schemes are more efficient than TAU on the drag coefficient, since it provides a given convergence level for a lesser amount of computational time.

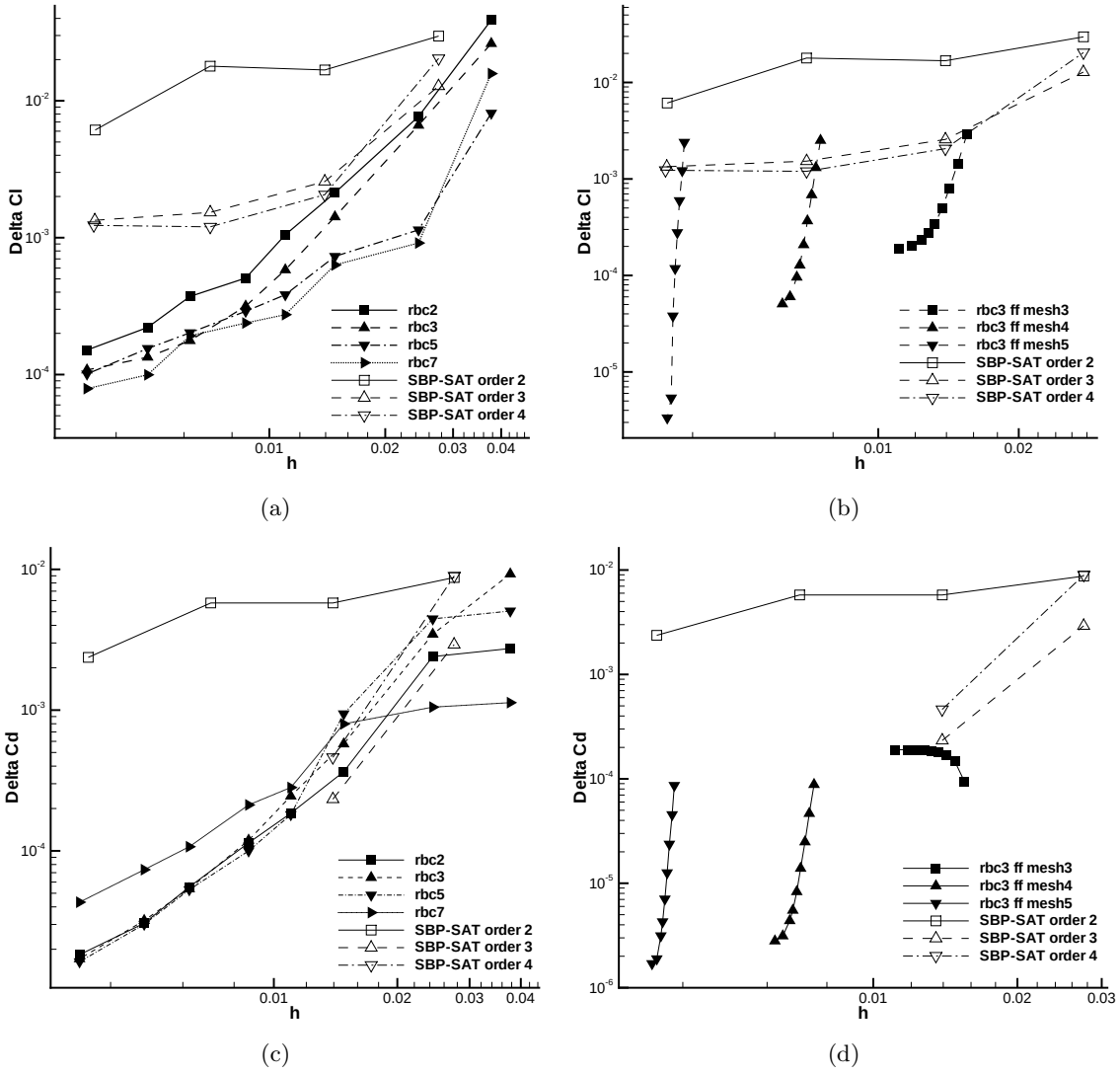
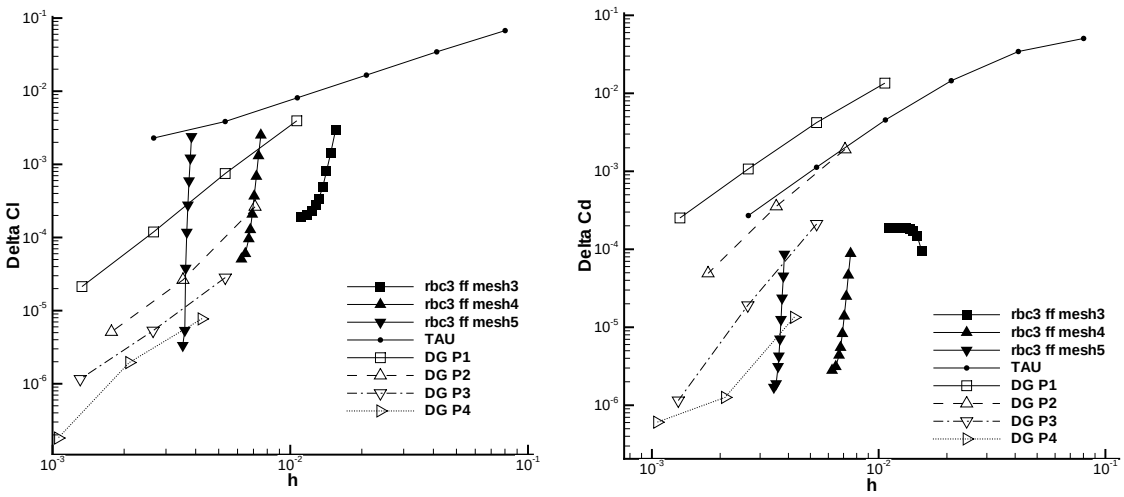


Figure 3.6: Convergence of the lift coefficient (top) and of the drag coefficient (bottom) as a function of the grid refinement (right: comparison between RBC schemes and SBP-SAT methods; RBC results obtained on a 1000-chord grid, SBP-SAT with a 100-chord grid, left: error convergence with respect to the far-field location for scheme RBC3)

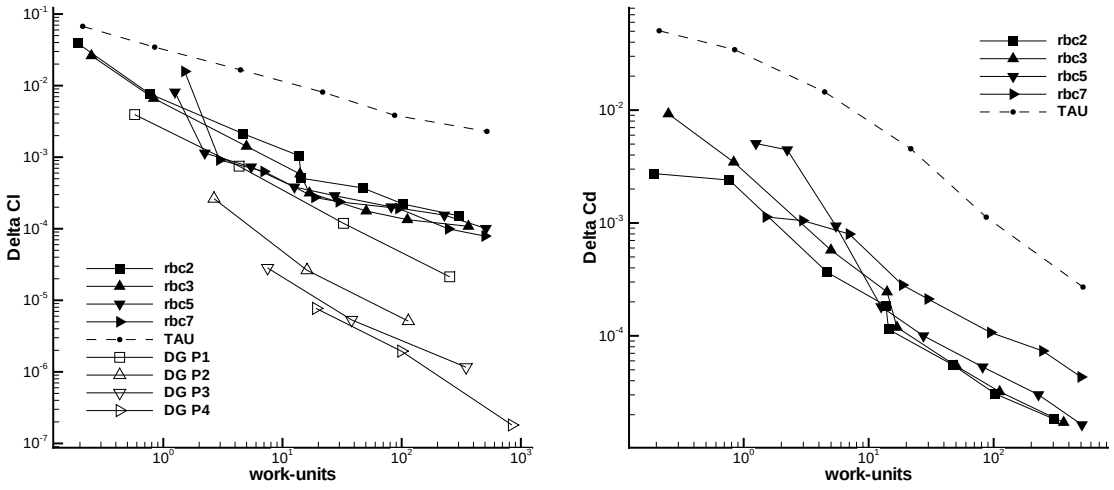
3.2.3 Subsonic viscous

The governing equations considered are the 2-D Navier-Stokes equations with a constant ratio of specific heats 1.4 and Prandtl number of 0.72. Concerning the flow conditions, we consider a subsonic laminar flow with $Mach = 0.5$, angle of attack $\alpha = 0^\circ$, Reynolds number $Re = 5000$ and $Pr = 0.72$ (adiabatic wall). The dynamic viscosity is assumed to be constant. Such a flow past an aerodynamic profile is entirely defined by the far field Mach number and Reynolds number - based on the chord length - and the angle of attack. For the far field representation, 1-D non-reflexion boundary conditions are employed [178, 179]. On the airfoil surface, a no slip adiabatic wall condition is imposed. We choose a steady configuration which was a test-case of the GAMM workshop [161] of 1987 on compressible viscous flow around a NACA0012 airfoil. The results presented in Table 3.3 are comparable to those of the GAMM workshop but those



(a) (b)

Figure 3.7: Convergence of the lift coefficient as a function of the grid refinement, comparison between RBC, Tau and DG methods (for RBC, for each grid refinement level we change the far-field location)



(a) (b)

Figure 3.8: Convergence of the lift and drag coefficients as a function of work units: comparison between RBC, Tau and DG methods (RBC solutions obtained on 1000-chord grids)

last were given with only two significant digit. Then, we do not plot any convergence of the drag coefficient as for the preceding cases. Fig. 3.15 provides an overview of the Mach number and pressure coefficient contours for this case, and Fig. 3.16 show the recirculation bubble at the trailing edge of the airfoil. The separation point is located around $0.79 - 0.81\%$ of the chord. When using a coarse grid or a dissipative scheme, the separation point moves towards the trailing edge, leading to a smaller recirculation bubble.

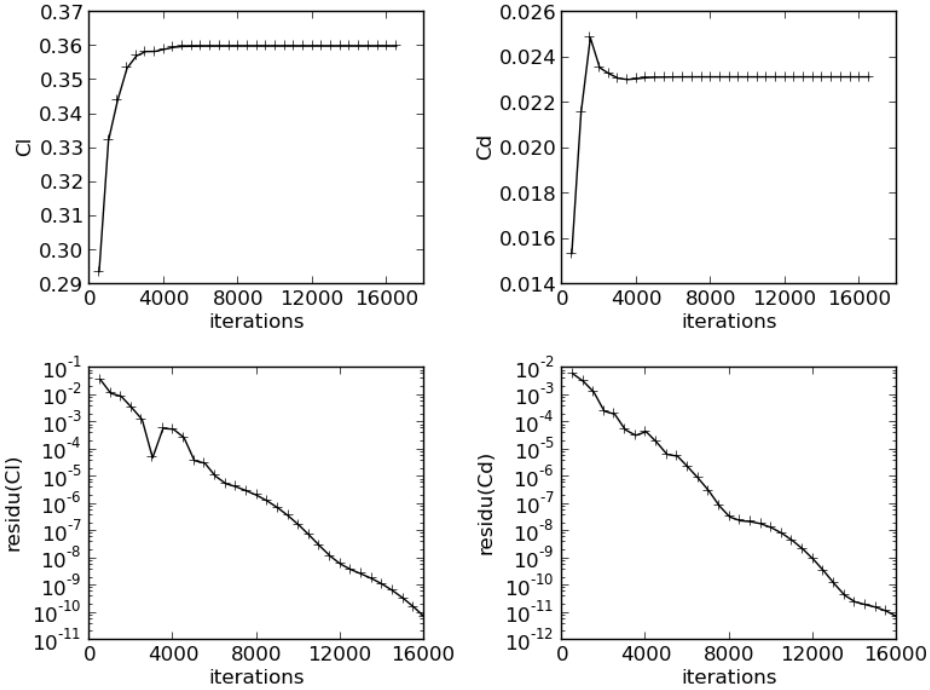
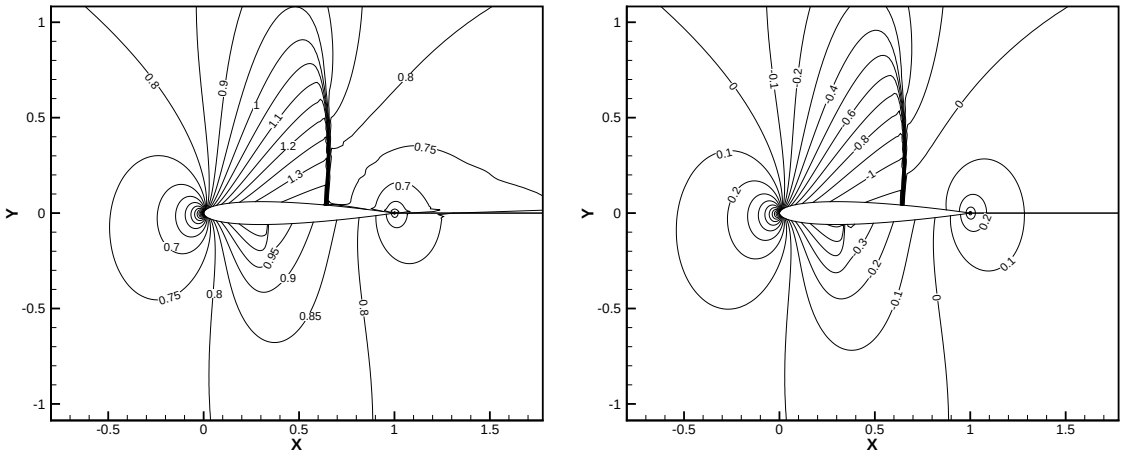


Figure 3.9: Convergence of the lift and drag coefficients for the transonic inviscid flow over the NACA-0012 airfoil using RBC3 on grid 6

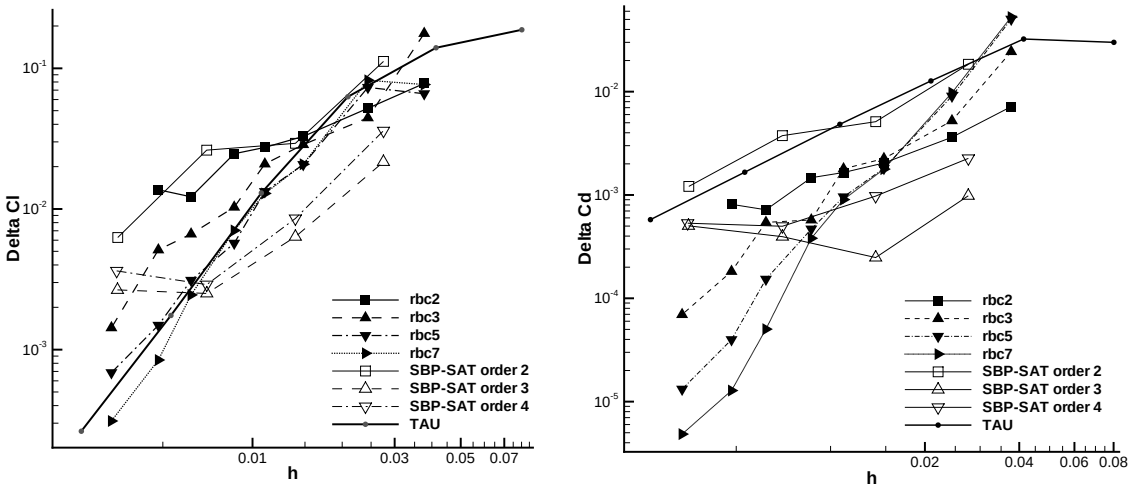


(a) (b)

Figure 3.10: Transonic inviscid flow over a NACA-0012 airfoil at $Mach = 0.8$ and $\alpha = 1.25^\circ$ using RBC3 on mesh 6 (left: iso-lines of the Mach number, right: iso-lines of the pressure)

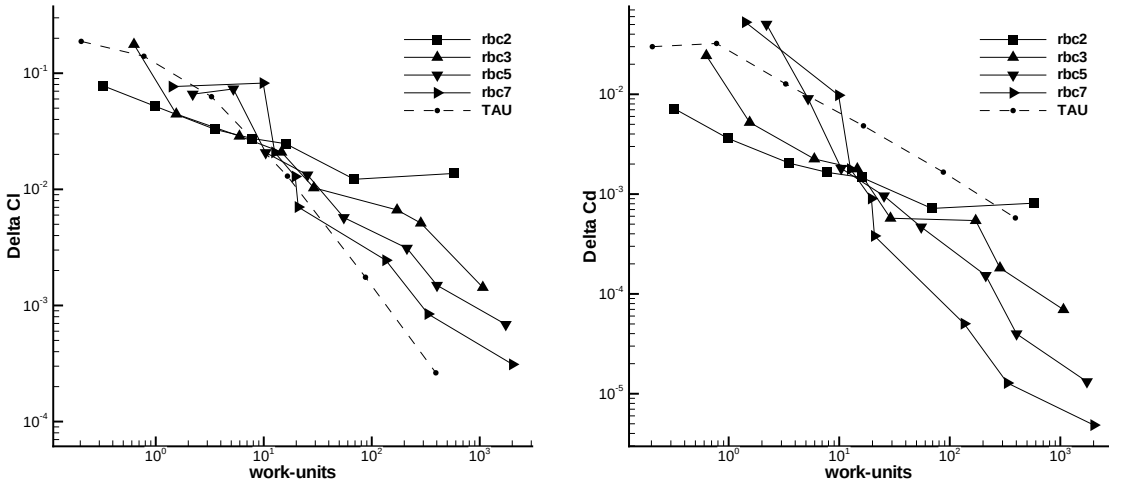
Table 3.3: Subsonic viscous flow over a NACA0012 airfoil

Scheme	Mesh	im	jm	h	C_d	max(c_f)	decol
RBC3	1	67	35	0.02065041	0.05410431	0.14426545	0.81193048
	2	134	69	0.01039975	0.05420491	0.14499917	0.79897599
	3	267	137	0.00522858	0.05451909	0.14585844	0.80879807
RBC5	1	67	35	0.02065041			
	2	134	69	0.01039975	0.05407106	0.05407106	0.78540246
	3	267	137	0.00522858	0.05449134	0.05449134	0.80558254
RBC7	1	67	35	0.02065041			
	2	134	69	0.01039975	0.05406115	0.05406115	0.78540246
	3	267	137	0.00522858	0.05450876	0.05450876	0.80593048



(a) (b)

Figure 3.11: Convergence of the lift coefficient as a function of the grid refinement, comparison between RBC, Tau and DG methods.



(a) (b)

Figure 3.12: Convergence of the lift and drag coefficients as a function of work units: comparison between RBC, Tau and DG methods (RBC solutions obtained on 1000-chord grids).

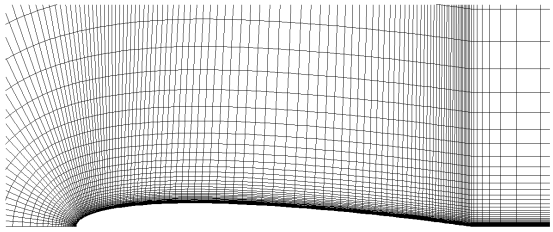


Figure 3.13: Mesh used for the viscous flow past the NACA0012 airfoil (grid 2)

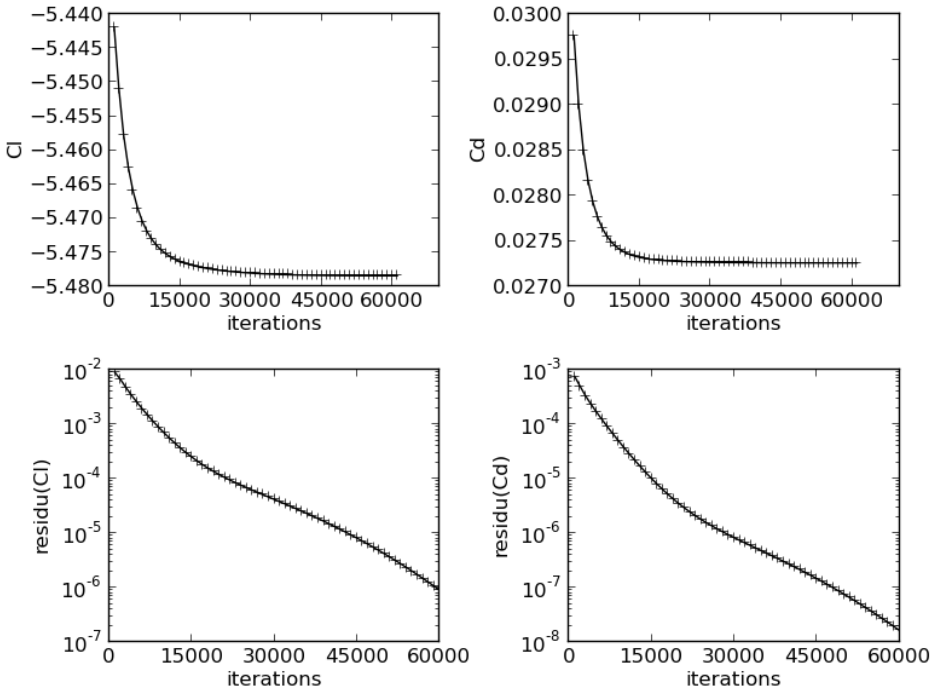
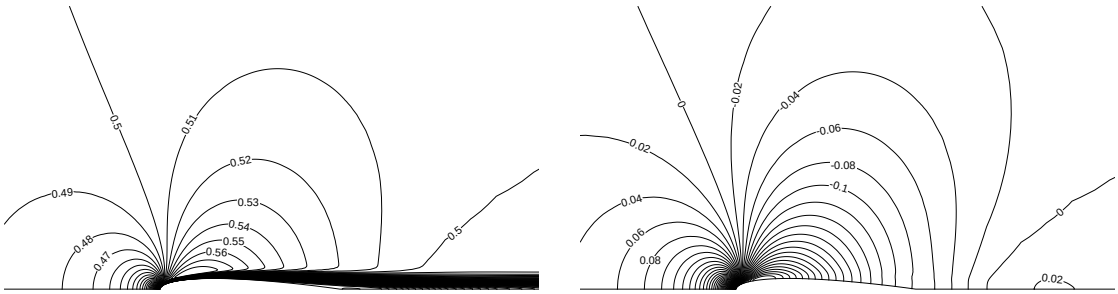


Figure 3.14: Convergence of the lift and drag coefficients for the subsonic viscous flow over the NACA-0012 airfoil using RBC5 on grid 2



(a) (b)

Figure 3.15: Subsonic viscous flow over a NACA-0012 airfoil at Mach=0.5 and Reynolds=5000 using RBC5 on mesh 2 (left: iso-lines of the Mach number, right: iso-lines of the pressure)

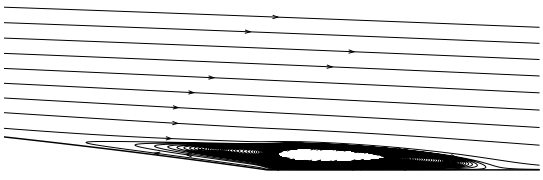


Figure 3.16: Subsonic viscous flow over a NACA-0012 airfoil at Mach=0.5 and Reynolds=5000 using RBC5 on mesh 2, streamlines: zoom on the recirculation bubble at the trailing edge of the airfoil

3.3 Validation of the RANS solver

The following test-cases result from a collaboration with Dr Content. The two first test-cases are extracted from [53] and the third one is extracted from [130].

3.3.1 Turbulent flat plate

To check the implementations of the RANS system(s) in DynHoLab, we consider the turbulent flat plate verification case provided on the NASA Turbulence Modeling Resource website [154]. DynHoLab results are compared to the solutions obtained with two independent compressible RANS codes, CFL3D [155] and FUN3D [103]. CFL3D is a cell-centered structured-grid code, and FUN3D is a node-centered unstructured-grid code. Both codes use Roe's Flux Difference Splitting and a UMUSCL upwind approach. As the grid is sufficiently refined, the results should approach the same approximation (if the flow conditions and boundary conditions are the same). We thus consider the finest 545×385 grid ($y_+ = 0.1$) over the entire plate to perform our comparisons.

The turbulent flat plate case was run at $M = 0.2$, at a Reynolds number of $Re_L = 5 \cdot 10^6$ based on a unit length L. The solid wall of the grid extended from $x = 0$ to $x = 2$, i.e. from $Re_x = 0$ to $Re_x = 10 \cdot 10^6$. At the far-field, a turbulent intensity Tu equal to $5 \cdot 10^{-3}$ and a turbulent Reynolds number equal to $1 \cdot 10^{-2}$ were imposed. Since for this case the maximum boundary layer thickness is about 0.03 L, the grid extended from $y = 0$ at the wall to $y = L$ is enough to avoid influence of the upper boundary condition on the developing boundary layer. Three grids of increasing density are considered: 545×385, 273×193 and 137×97. Each coarser grid is exactly every-other-point of the next finer grid. The grid are stretched in the wall-normal direction, and also clustered near the plate leading edge. The finest grid has minimum spacing at the wall of $y = 5 \cdot 10^{-7}$, giving an approximate average $y_+ = 0.15$ over the plate at the Reynolds number run. Even the coarsest grid has a quite fine wall-normal spacing, giving an approximate average $y_+ = 1.0$ over the plate.

Figure 3.17 compares the convergence of the drag coefficient for both uncoupled and coupled RANS computations. The numerical fluxes are evaluated by the third-order accurate FE-MUSCL scheme and the steady state is reached with the JFNK and the LU-SSOR algorithm. LU-SSOR computations have been made with a CFL ramp from 0.01 to 10 between the 5000^{th} and the 10000^{th} iteration. As expected, the JFNK algorithm ensures a better convergence rate

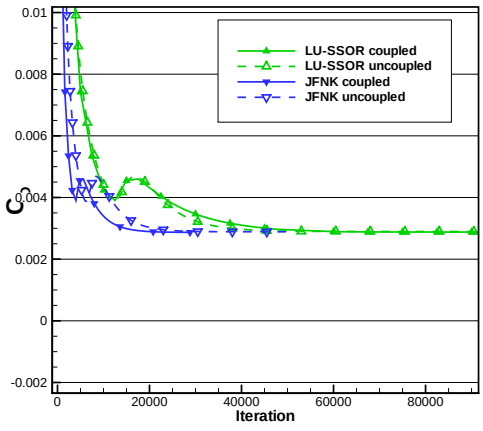
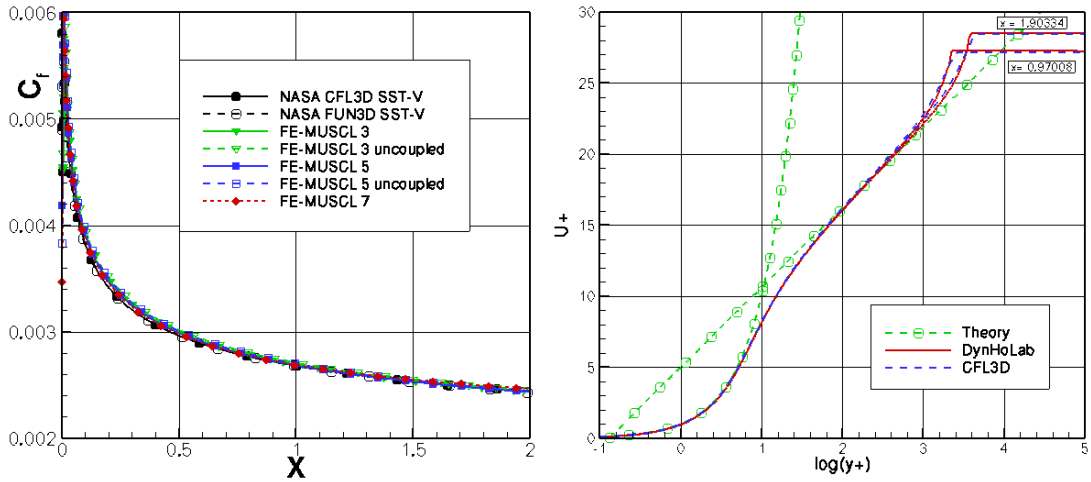


Figure 3.17: Convergence of the Drag coefficient for the third-order accurate FE-MUSCL scheme

than LU-SSOR in term of iterations. Namely, about 20000 iterations are required to converge

the drag coefficient within an error of 1%, whereas 46000 iterations are required by LU-SSOR to achieve the same convergence criterion. Also note that the coupled approach speeds up convergence, especially when used in conjunction with the JFNK method. Precisely, a gain in required number of iterations of 24% is observed over the uncoupled computation with the same approach. Nevertheless, due to the choice of a basic version of GMRES (*i.e.* without restart, Ref [104]), the CPU cost of a JFNK iteration is more than five times larger than a LU-SSOR iteration. In the following, the LU-SSOR algorithm is used.

The surface skin friction coefficient and the mean velocity profiles from CFL3D, FUN3D and DynHoLab codes are plotted in figures 3.18(a) and 3.18(b). All codes yield nearly identical results over most of the plate.
Since the flow over a flat plate (without pressure gradient) is in turbulence equilibrium, it is not surprising to obtain results for uncoupled and coupled RANS simulations fairly close to each other and also close to the law-of-the-wall theory. Figures 3.19(a) and 3.19(b) shown turbulent

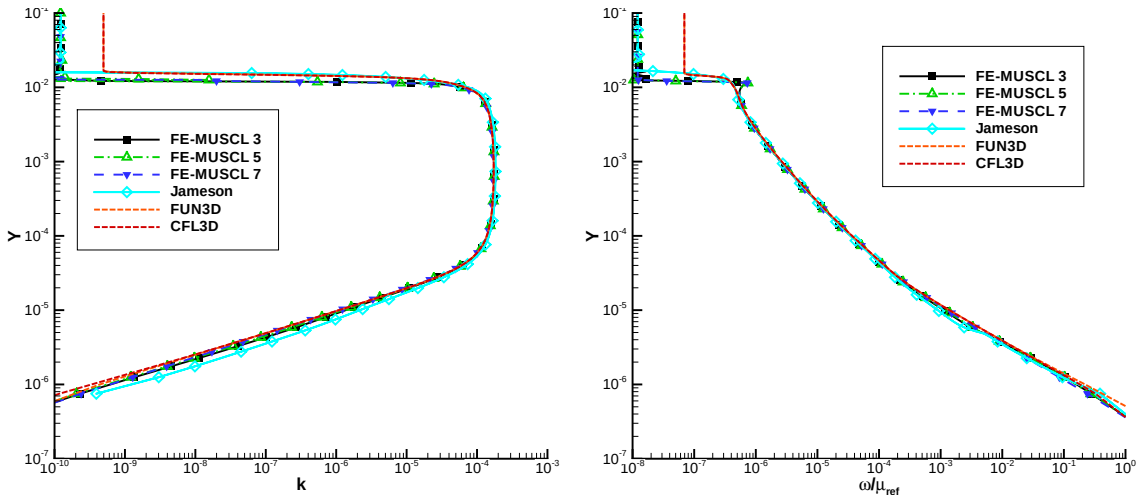


(a) $C_f = f(x)$ (b) $U+ = f(\log(y+))$

Figure 3.18: (a): Skin friction coefficient obtained by CFL3D, FUN3D and FE-MUSCL 3, 5 & 7. (b): Mean velocity profiles at $x = 0.97008$ and $x = 1.90334$ obtained by CFL3D and DynHoLab are compared to law-of-the-wall theory

quantities profiles at $x = 0.97$ from CFL3D, FUN3D and DynHoLab codes. In the boundary layer, these computations yield to the same predictions. This is not the case in the free-stream were DynHoLab predicts lower level of turbulent quantities. This last observation is due to the initialization of the turbulence level in the far-field which is different in the case of the computation led with DynHoLab.

Figure 3.20 compares computed drag coefficient for different schemes and different grid densities. Note that in this particular flat plate case, when looking at the total integrated drag coefficient on the plate, formal order-property convergence may not be generally achievable. This is because the skin friction (which is the only contributor to the drag in this case) is singular at the leading edge. The discrepancies between results obtained with FE-MUSCL schemes and those obtained with Jameson scheme are due to the extra dissipation in the design of the last scheme.



(a) k profile (b) ω profile

Figure 3.19: SST turbulent quantities profiles at $x = 0.97$ obtained by CFL3D, FUN3D and FE-MUSCL 3, 5 & 7. (a): k profil (b):ω profile

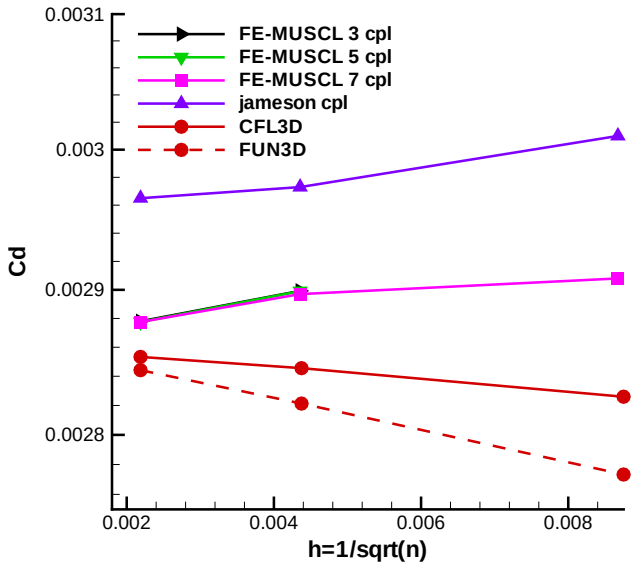


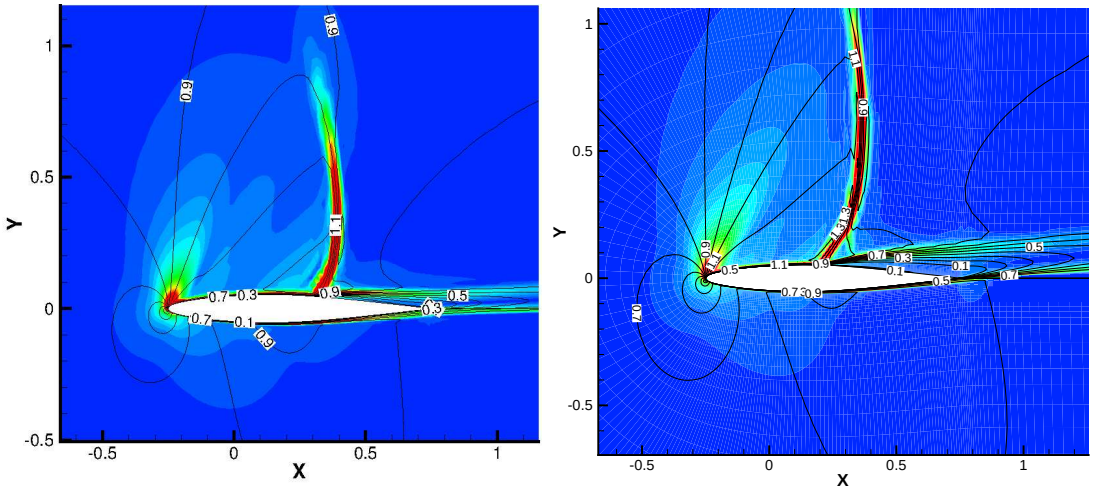
Figure 3.20: Grid convergence of the Drag coefficient for FE-MUSCL 3, 5 & 7 on the coupled equations

3.3.2 Transonic flow over a NACA 64A10

Flows over the NACA 64A10 at turbulent transonic conditions [99] are chosen to point out the discrepancies between the coupled and uncoupled approaches.
These configurations include an interaction between shock-wave and the boundary layer. The severity of the interaction has been modified in Reference [99] by varying the angle of attack at a fixed free-stream Mach number of 0.8 and a Reynolds number based on the free-stream properties and the airfoil chord equal to $Re_c = 2 \cdot 10^6$. Two of the three angles of attack in-

vestigated by Johnson *et al.* ($\alpha = 3.5^o$ and $\alpha = 6.2^o$) have been considered, corresponding to a shock-wave/boundary layer interaction of medium strength with mild separation, and an interaction of sufficient strength to produce a shock-induced stall situation.

The computational domain is discretized by a C-mesh with 311x75 nodes. The far-field boundary is located at 8.5 chords. The height of the first cell close to the wall is such that $y+ \approx 2$. At the far-field, a turbulent intensity Tu equal to $5 \cdot 10^{-3}$ and a turbulent Reynolds number equal to $1 \cdot 10^{-2}$ were imposed.
In figure 3.21, Schlieren visualizations and Mach number contours of the two configurations are compared to Mach contour from Johnson experiment [99]. Results were obtained with the FE-MUSCL 3 scheme and the coupled approach.



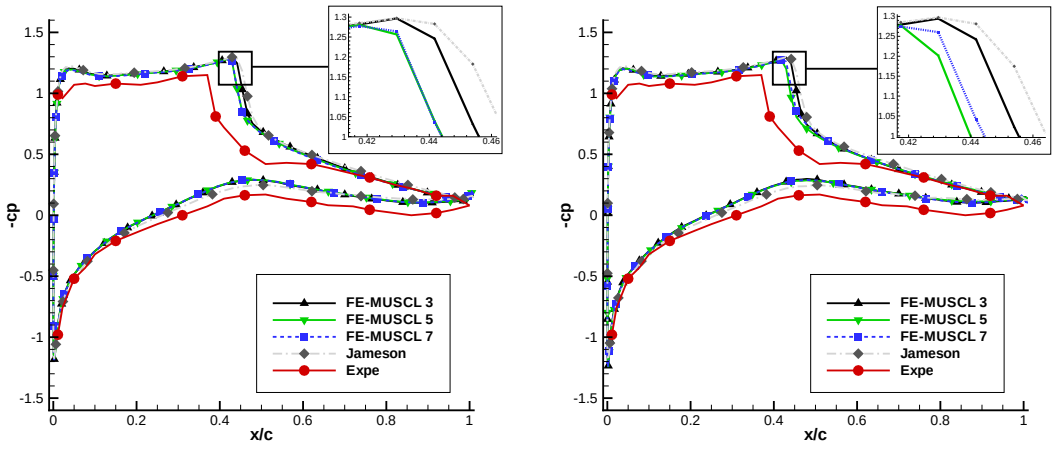
(a) $\alpha = 3.5^o$ (b) $\alpha = 6.2^o$

Figure 3.21: Schlieren visualizations and Mach contour on the turbulent flow over the NACA 64A10 from DynHoLab. Results obtained for a)the $\alpha = 3.5^o$ configuration, b) the $\alpha = 6.2^o$ configuration.

The Mach contours given by CFD computations are fairly close to those reported by Johnson *et al.*. Especially, the CFD computations are successful in predicting the very dominant forward curvature in the shock wave near the airfoil. To be more quantitative we report on figure 3.22 the surface pressure comparison between Johnson measurements and coupled and uncoupled RANS computations for the configuration $\alpha = 6.2^o$.

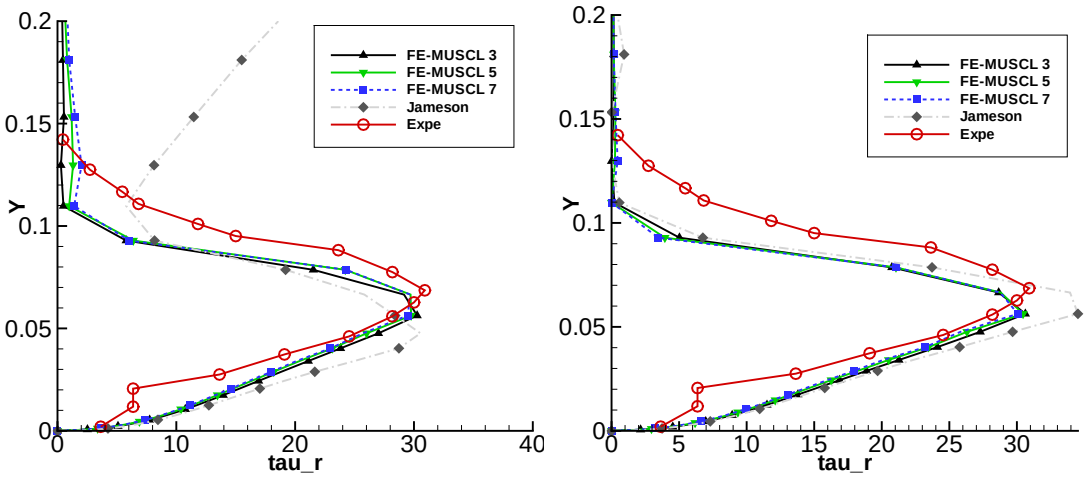
The separation point is predicted with a small delay by all computations between $x/c = 0.42$ and $x/c = 0.44$ while it occurs at $x/c = 0.37$ in experiment. The use of high order schemes to compute the convective terms slightly improves predictions. This might be due to the lowest amount of numerical dissipation introduced in high-order schemes, leading to a slightly stronger shock. Note that the coupled and the uncoupled approach yield to the same pressure surface distribution, even after the separation point where non-equilibrium effects occurs.

Figures 3.23 and 3.24(a), 3.24(b) compare the non-dimensional Reynolds shear stress profile at two chord-wise position in the separated flow region given by coupled and uncoupled approaches with the one reported in Johnson *et al* [99]. For FE-MUSCL computations the Reynolds shear stress predicted is nearly the same regardless of the order of accuracy. This is not the case when Jameson scheme is used. Actually, in Jameson coupled computation, an over production of the turbulent kinetic energy at the edge of the separated flow region occurs. The



(a) coupled computation (b) uncoupled computation

Figure 3.22: Comparison between Johnson measurements, coupled and uncoupled RANS computations, $\alpha = 6.2^o$.



(a) coupled (b) uncoupled

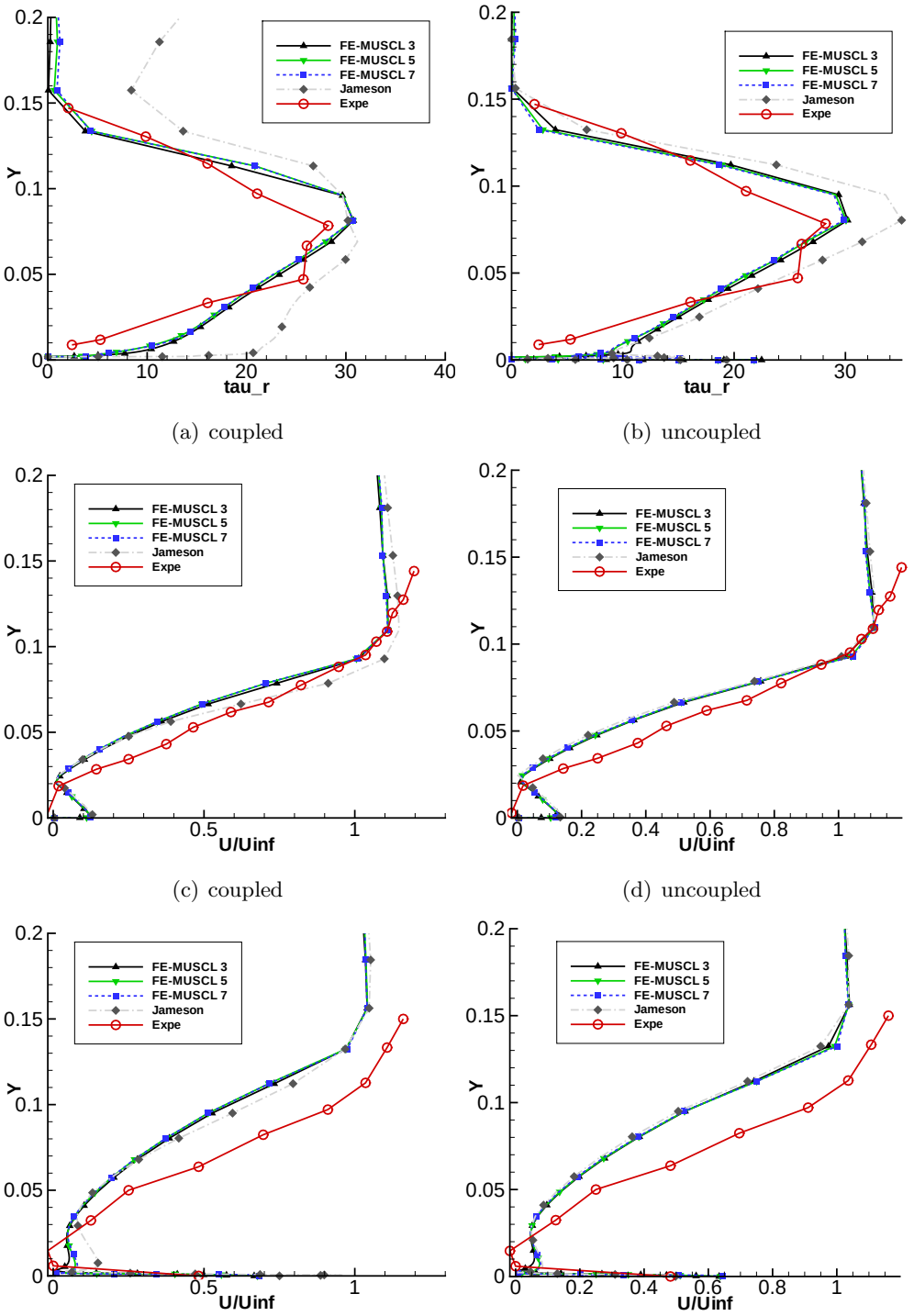
Figure 3.23: Comparison of the Reynolds shear stress at $x/c = 0.82$. Results obtained for a) coupled computation, b) uncoupled computation

over production of k in Jameson coupled computation slightly deforms the velocity profile as shown Figures 3.24(c), 3.24(d), 3.24(e) and 3.24(f).

3.3.3 ONERA M6 wing

The final numerical application of this section on the validation of the RANS solver is a 3-D transonic turbulent flow calculation over the ONERA M6 wing. The ONERA M6 wing is a classic CFD validation case for external flows because of its simple geometry combined with complexities of transonic flow (i.e. local supersonic flow, shocks, and turbulent boundary layers separation) and it is included in the NASA validation database [182].

In the presented computation, we use the structured mesh provided by the NASA website [182], the characteristics of which are given in table 3.4. Concerning the boundary conditions,



(a) coupled (b) uncoupled

(c) coupled (d) uncoupled

(e) coupled (f) uncoupled

Figure 3.24: Top: Comparison of the Reynolds shear stress at $x/c = 1.02$. Results obtained for a) coupled computation, b) uncoupled computation, Middle: Comparison of the mean velocity profile at $x/c = 0.82$. Results obtained for a) coupled computation, b) uncoupled computation, Bottom: Comparison of the mean velocity profile at $x/c = 1.02$. Results obtained for a) coupled computation, b) uncoupled computation

a no-reflexion condition is used for the far-field, an adiabatic no-slip wall condition is imposed at the wall surface and a symmetry plane boundary condition is used for the wing root plane. Here, we simulate flow conditions such that the Reynolds number based on the free-stream conditions and wing mean chord is equal to $Re = 11.72 \times 10^6$, the Mach number $M_\infty = 0.8395$, and the angle of attack is $\alpha = 3.06^\circ$. The results shown on Figure 3.26, 3.27 and 3.28 are computed with uncoupled RANS equations using the $k\omega$-SST model and the third-order FE-MUSCL scheme on both the mean flow equations and the turbulent transport equations. Figures 3.27 and 3.28 show the pressure coefficient in the cross sections of the wing. Results are in good agreement with experimental data by Schmitt and Charpin [159].

Table 3.4: Description of the M6 wing mesh provided by the NASA website [182]

zone	dimension	total grid points
block 1	$25 \times 49 \times 33$	40425
block 2	$73 \times 49 \times 33$	118041
block 3	$73 \times 49 \times 33$	118041
block 4	$25 \times 49 \times 33$	40425

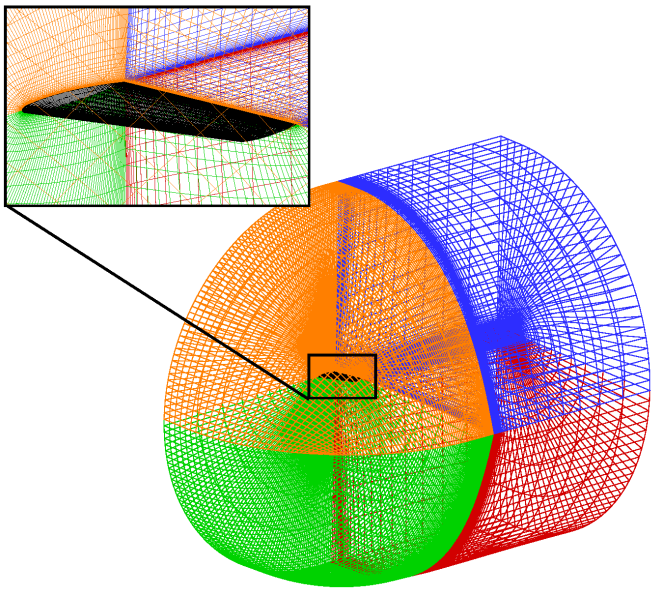


Figure 3.25: The NASA mesh of the ONERA M6 wing, composed of four structured blocks.

3.4 Implicit Large Eddy Simulation of the flow over 2-D periodic hills

A very large implicit large eddy simulation (ILES) of the flow over a periodic 2D-Hill was carried out using the RBC scheme of 3rd-order accuracy. The computations were conducted for an average Mach number of 0.1 and an average Reynolds number of 10595. The flow is driven by a forcing term counteracting the drag force exerted on the channel walls. The forcing term is updated at each time step as a function of the instantaneous space-averaged mass flow. The wall temperature is imposed.

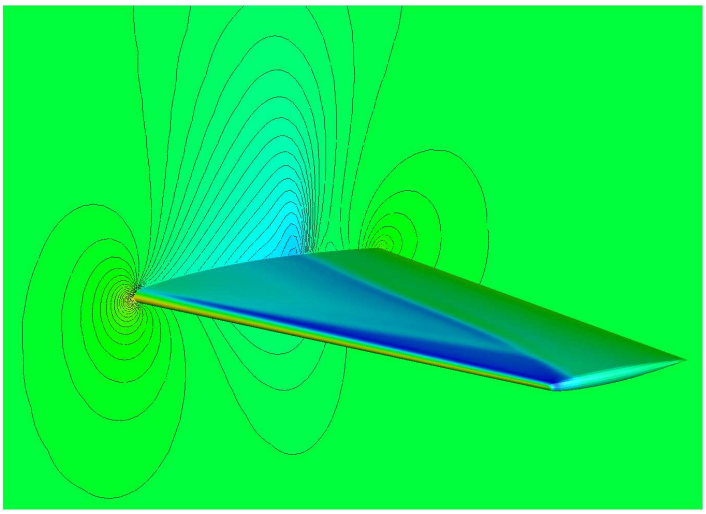
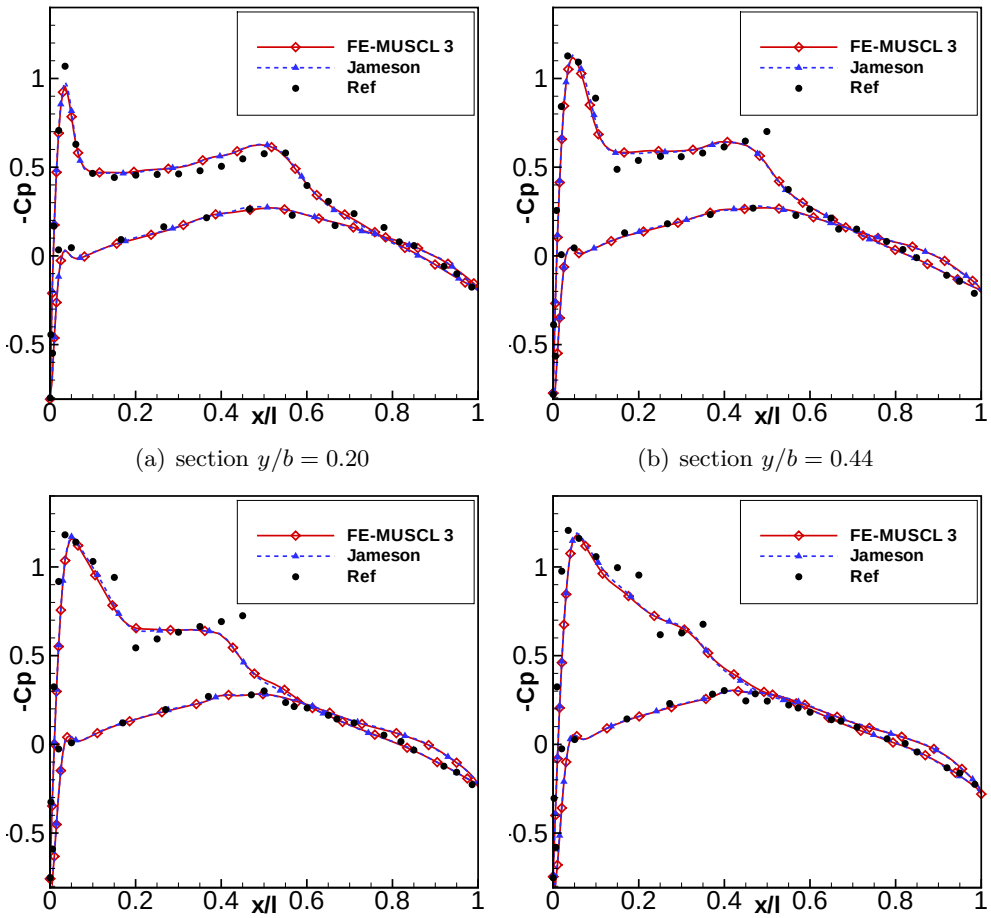


Figure 3.26: Computation of the flow over the ONERA M6 wing at $Re = 11.72 \times 10^6$, Mach number $M = 0.8395$ and the angle of attack $\alpha = 3.06^\circ$. Third-order FE-MUSCL scheme.



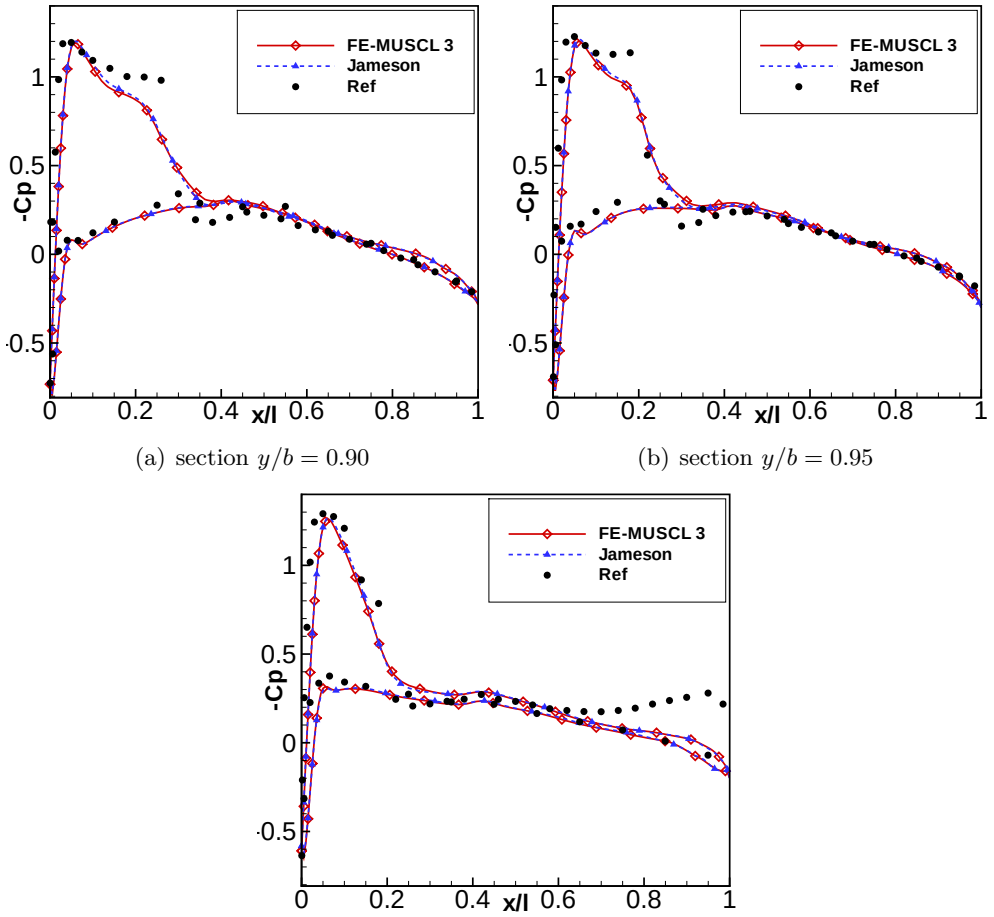
(a) section $y/b = 0.20$

(b) section $y/b = 0.44$

(c) section $y/b = 0.65$

(d) section $y/b = 0.80$

Figure 3.27: Plots of the pressure coefficient on the surface of the ONERA M6 wing.



(a) section $y/b = 0.90$

(b) section $y/b = 0.95$

(c) section $y/b = 0.99$

Figure 3.28: Plots of the pressure coefficient on the surface of the ONERA M6 wing.

The calculations were carried out on a structured grid made of $64 \times 32 \times 32$ cells, corresponding to the coarsest linear grid used for the 2nd workshop on High-Order Methods [14].

The calculations were initialized with a uniform field. The simulation took about 50 convective periods (based on the streamwise length L of the computational domain and the bulk velocity at the throat section U_b, a convective period being given by $T = L/U_b$) to reach a statistically steady state. Then, statistics were collected over 9 additional periods. Calculations were carried out by using a constant time step, taken equal to $10^{-2}T$. Fig. 3.29 shows the averaged fields of the streamwise velocity $<u>$, as well as mean streamlines. Fig. 3.30 and 3.31 display the average streamwise velocity and shear Reynolds stress profiles at different streamwise locations. Present results are compared to the reference LES of Breuer *et al.* [40], obtained on a grid of about 13 million cells by using a second-order centred finite volume scheme, to the experimental data by Rapp [40] and to results obtained by using a second-order accurate central difference scheme with second-order filtering (denoted FDo2-SFo2). Present results show that the third-order RBC scheme provides results in reasonably good agreement with the reference data, at least for first-order statistics, despite the extremely coarse grid in use. The RBC solution represents a dramatic improvement over the solution provided by the standard second-order scheme, which does not even capture the correct trends, since numerical errors introduced by the filter tend to laminarize the flow. This is confirmed by the fact that Reynolds stresses for

this scheme are virtually null, so that they are not represented on Fig. 3.31.

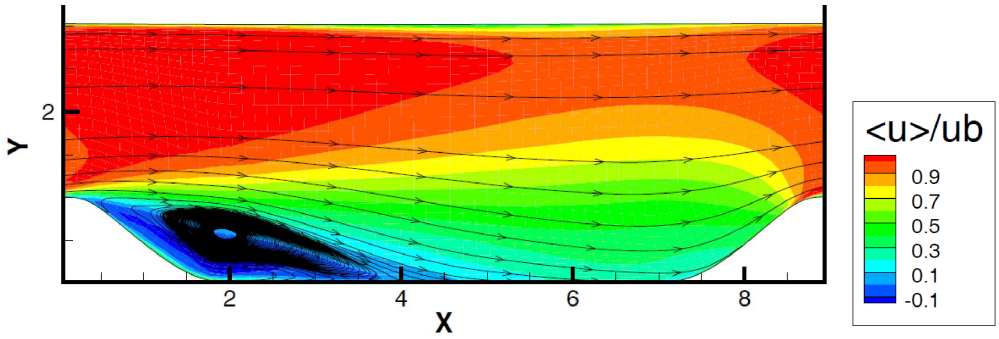


Figure 3.29: ILES of the flow over a periodi 2D Hill. Mean steamwise velocity contours and mean streamlines

3.5 Chapter summary

In this chapter we validated the different numerical methods implemented within the DynHoLab code against a broad variety of test case, which also proves the versatility of the code. Namely, very different flow models could be solved with the same code, going from well linear advection, Bürgers equation, Euler, Navier-Stokes and RANS systems of equations. Both numerical results and code performance were presented and compared on two- and three-dimensional simulations with others codes and reference simulations or experiments.

Concerning the high-order schemes, their implementation was validated in terms of observed orders of accuracy and error constants. Their contribution to a better prevision of the aerodynamic performance and their improved resolvability of turbulent scales are comforted. Regarding the robustness of high-order schemes, it should be noted that their correct employment is ensured only on fine and regular grids if no correction is applied concerning the fast mesh variation. This sensitivity to the mesh should be less present for compact schemes. However, these last require some attention concerning the boundary conditions. Some results presented could be probably improved by the development of high-order boundary conditions compatible with the use of compact schemes. It is worse noting that some flow configurations commonly considered as steady are found to be unsteady when the dissipation brings by the numerical scheme is reduced, as it is the case when high-order schemes are employed. High-order schemes can be better employed for unsteady flows in which the resolvability of fine scales are of primary importance for the prediction of high-order moments. Finally, the time integration of RBC schemes should be enhanced to treat fast varying flows with an high-order explicit treatment.

3.6 Résummé du chapitre (french)

Dans ce chapitre, nous avons validé les différentes méthodes numériques mises en œuvre dans le code de DynHoLab contre une large variété de cas de test, ce qui a également prouvé la polyvalence du code. Plus précisément, les modèles d'écoulement très différents pourraient être résolus avec le même code, allant de l'advection linéaire, l'équation de Bürgers, les systèmes d'équations d'Euler, Navier-Stokes et RANS. A la fois les résultats numériques et la performance du code ont été présentés et comparés sur des simulations en dimensions deux et trois avec d'autres codes et des simulations de référence ou des expériences.

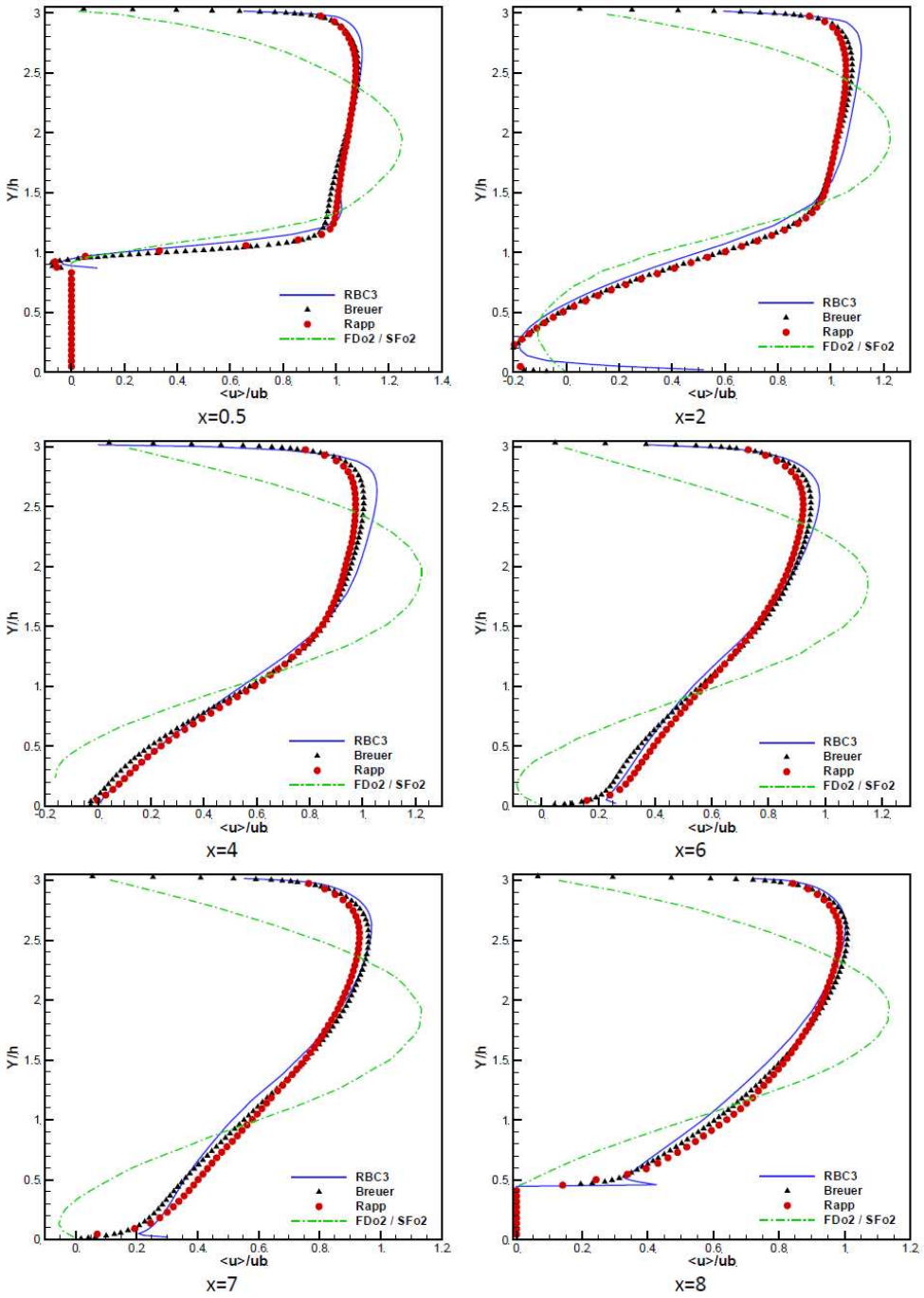


Figure 3.30: ILES of the flow over a periodi 2D Hill. Mean steamwise velocity profiles

Concernant les schémas d'ordre élevé, leur mise en œuvre a été validé en termes de d'ordre de précision observé et des constantes d'erreur. Leur contribution à une meilleure prévision de la performance aérodynamique et l'amélioration de leur résolvabilité des échelles turbulentes sont confortés. En ce qui concerne la robustesse des schémas d'ordre élevé, il convient de noter que leur emploi correcte est assurée uniquement sur les grilles fines et régulières si aucune correction est appliquée concernant la variation de géométrie rapide. Cette sensibilité à la maille

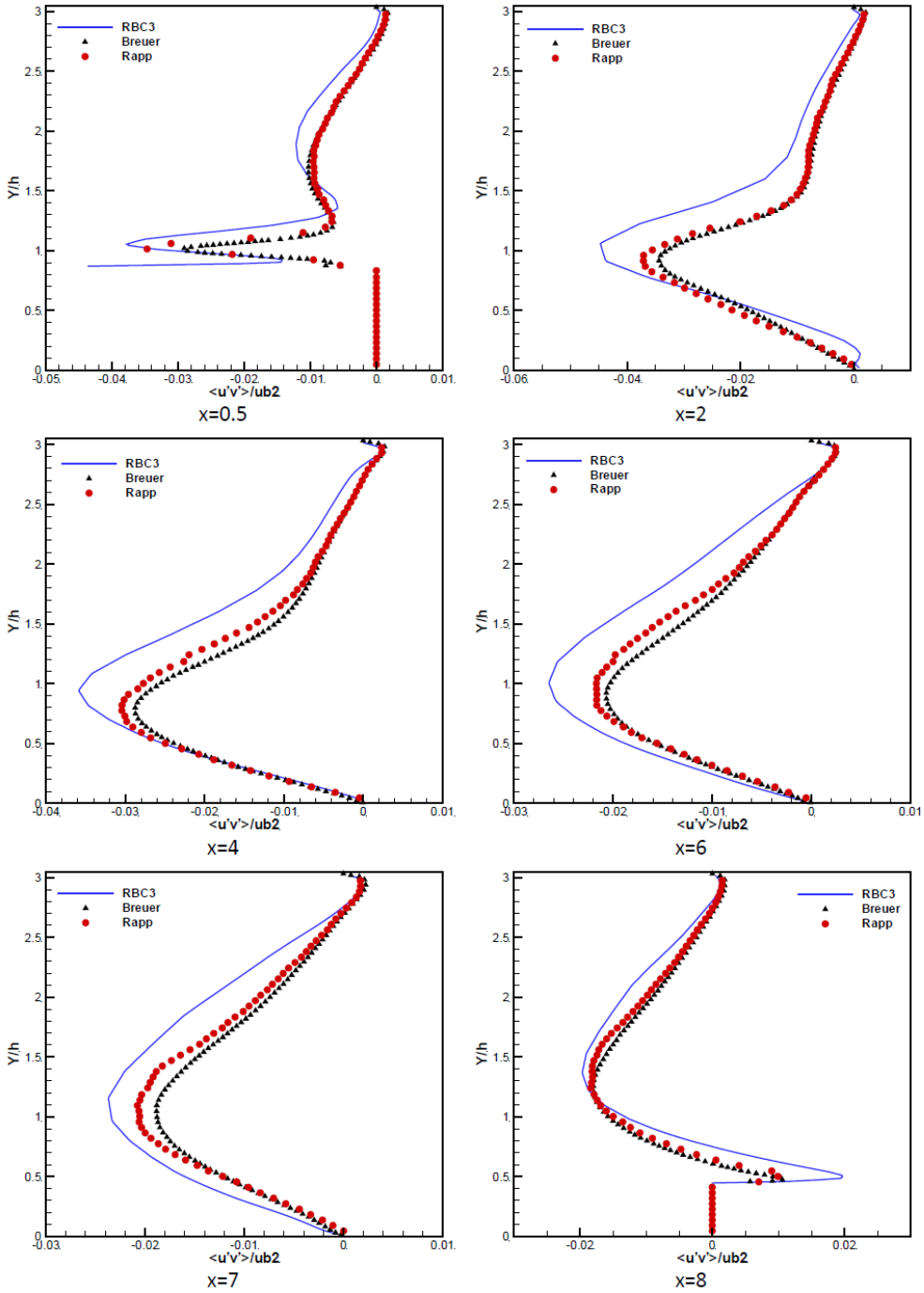


Figure 3.31: ILES of the flow over a periodi 2D Hill. Reynolds shear stress profiles

devrait être moins présente pour les schémas compacts. Cependant, ces derniers nécessitent une certaine attention sur les conditions aux limites. Certains résultats présentés peuvent être améliorés sans doute par le développement de conditions aux limites d'ordre élevé compatibles avec l'utilisation de schémas compacts. Il est intéressant de noter que certaines configurations d'écoulement généralement considérées comme stables se sont révélées être instable lorsque la dissipation apporte par le schéma numérique est réduite, comme il est le cas lorsque les schémas

d'ordre élevé sont employés. Les schémas d'ordre élevé peuvent être mieux mis à profit pour les écoulements instationnaires dans lequel la résolvabilité des échelles fines est de première importance pour la prédiction de moments d'ordre élevé. Enfin, l'intégration en temps des schémas RBC doit être améliorée afin de traiter les écoulements variant rapidement en temps avec des algorithmes explicites d'ordre élevé.

Chapter 4

High-order schemes for complex geometries on overlapping grids

Contents

Introduction (french)

Afin de simuler des configurations géométriquement complexes, les schémas RBC sont étendues à la méthode des volumes finis. Cette extension peut être faite en utilisant directement des flux numériques issus du cadre des différences finies. Les inconnues du problème peuvent être considérées comme valeurs ponctuelles de la solution dans les cellules du maillage. Pour les extensions directes de schémas aux différences finies à le cadre de volume fini, l'ordre formel de précision des schémas n'est pas conservé pour des maillages curvilignes en général. Cependant, l'ordre élevé peut encore être récupéré sur des grilles régulières cartésiennes, où le schéma de volume fini dégénère à celui d'ordre élevé issu des différences finies appliqué au maillage dual (maillage formé par les centres de cellules), ou sur des maillages faiblement déformées, à savoir des maillages avec des étirements et des défauts d'asymétrie du même ordre que le schéma utilisé (voir par exemple [144, 50, 145, 88] et les références citées). Dans une tentative de préserver l'ordre de précision élevé sur des maillages curvilignes quelconques, des extensions aux volumes finis d'ordre élevé des schémas RBC pour les régimes stationnaire et instationnaire ont été explicité dans [88, 85]. Pourtant, ces prolongations sont limitées à l'ordre trois sur des grilles

modérément déformées et la précision obtenue sur des maillages très déformés reste le second ordre. En outre, de telles formulations volumes finis d'ordre élevé impliquent un surcoût de calcul (en termes de temps CPU et la consommation de mémoire), en raison de l'utilisation de coefficients de discrétisation pondérés qui doivent être stockées et/ou recalculée à chaque pas de temps en cas de déformation des maillages. Dans le présent travail, afin d'étendre l'utilisation des schémas RBC d'ordre élevé à des géométries complexes tout en utilisant seulement des grilles de maillage cartésiennes ou faiblement déformées, nous étudions une méthodologie de maillages recouvrants. Dans cette approche, le domaine de calcul (éventuellement complexe) est décomposé en une série de sous-domaines topologiquement simples qui sont autorisés à se chevaucher les uns les autres.

Les méthodes de recouvrement de maillages ont été introduites par Benek et al. [30, 29, 165] pour des calculs aérodynamiques. La méthode a été généralisée par Chesshire et Henshaw [44, 45]. Des contributions importantes ont été apportées à l'ONERA par Benoît, Saunier, Péron et al.[157, 158, 33, 138] ainsi que d'autres développements en parallèle avec des travaux de PhD [136], où les maillages recouvrant sont utilisés conjointement avec des schémas directionnels non-compacts avec des ordres de précision nominaux allant de 3 à 5 et des interpolations de Lagrange d'ordre trois au plus. Des applications de maillages recouvrants avec des schémas centrés compacts d'ordre élevé et des filtres d'ordre élevé compacts peuvent être trouvées dans [162, 149]. Enfin, les méthodes de grilles recouvrantes ont également été étendues à des simulations aéroacoustiques avec des schémas à préservation de la relation de dispersion (DRP) et des interpolations d'ordre élevé optimisées dans l'espace de Fourier pour des simulations en aéroacoustique dans Réf. [48, 47].

Dans ce chapitre, nous étendons les schémas RBC d'ordre élevé (pour les ordres 3, 5 et 7) à des configurations complexes en aérodynamique en élaborant une stratégie de grilles recouvrantes avec des interpolations de Lagrange d'ordre élevé. L'intérêt des stratégies de maillage recouvrants est de permettre l'utilisation de maillages de type Chimère, incluant des grilles Cartésiennes, où les schémas d'ordre élevé peuvent être efficacement implémentés en utilisant la formulation différence finie tout en préservant leur précision nominale. A cause du support multidimensionnelle des schémas RBC, un soin particulier doit être apporté au traitement des interpolations des points situés dans les coins des grilles du maillage. Le domaine de calcul est discrétisé par un ensemble de grilles qui se chevauchent, chacune équipée d'une ou de plusieurs couches de cellules fantômes, selon le support de discrétisation du schéma utilisé. Ces cellules fantômes sont utilisés pour récupérer des informations à partir de grilles connexes. Les variables de champ sont interpolées à l'aide de polynômes de Lagrange d'ordre élevé, avec le même degré de précision que le schéma interne.

La méthode de la grilles recouvrantes est développé dans le code DynHoLab présenté dans le chapitre 2 [131]. Plusieurs applications numériques sont utilisés pour démontrer l'exactitude et l'efficacité de l'approche proposée et son potentiel pour des géométries complexes. Le présent travail a été présenté à la conférence 32nd AIAA Applied Aerodynamic Conference 2014 [129].

4.1 Introduction

In order to simulate geometrically complex configurations, RBC schemes are extended to a finite volume framework. This extension can be done straightforwardly by using numerical fluxes derived in the finite difference framework. Problem unknowns can be seen as pointwise values of the solution of mesh cells. For direct extensions of finite difference schemes to the finite volume framework, the formal order of accuracy of the scheme is not preserved for general curvilinear meshes; however, high-accuracy can still be recovered on regular Cartesian grids,

where the finite volume scheme degenerates to the high-order finite difference scheme applied to the dual grid (grid formed by the cell centers), or on weakly deformed grids, i.e. grids with stretching and skewness defects of the same order of the scheme in use (see for instance [144, 50, 145, 88] and references cited therein). In an attempt to preserve high-order accuracy on general curvilinear grids, high-accurate finite volume extensions of RBC schemes for steady and unsteady flows have been derived in [88, 85]. Yet, these extensions are limited to third order accuracy on moderately distorted grids and to second-order accuracy only on highly deformed grids. Moreover, such high-accurate finite volume formulations imply a computational overload (in terms of CPU time and memory consumption), due to the use of weighted discretization coefficients that need to be stored and/or recomputed at each time step in case of deforming grids. In the present work, in order to extend the use of finite difference high-order RBC schemes to complex geometries while only Cartesian or weakly deformed grids, we investigate an overset grid framework. In this approach, the (possibly complex) computational domain is decomposed in a series of topologically simple sub-domains that are allowed to overlap each other.

Overset methods were first introduced by Benek et al. [30, 29, 165] for aerodynamic computations. The method has been generalized by Chesshire and Henshaw [44, 45]. Significant contributions were made at ONERA by Benoît, Saunier, Péron et al. [157, 158, 33, 138] and further developed in a parallel PHD work [136], where overset grids are used along with directional non compact schemes with nominal order of accuracy 3 or 5 and Lagrange interpolations of third-order at most. Applications of overset grids in conjunction with high-order centered compact schemes and high-order compact filters can be found in [162, 149]. Finally, overset grid methods where also extended to aeroacoustic simulations along with dispersion relation preserving (DRP) schemes and high-order interpolations optimized in Fourier's space for aeroacoustics simulations in Ref. [48, 47].

In this chapter we extend RBC schemes of high order (order 3, 5 and 7) to complex configurations in aerodynamics by developing an overset grid strategy with high-order Lagrange interpolations. The interest of overlapping grid strategies is that they may enable the use of Chimera type meshes, including Cartesian blocks, where high-order schemes can be efficiently implemented using the finite difference formulation while preserving their nominal accuracy. Because of the multidimensional stencil of RBC schemes, care must be taken to interpolate grid points situated at the corners of mesh blocks correctly. The computational domain is discretized by a set of overlapping meshes, each one equipped with one or more layers of ghost cells, according to the discretization stencil of the scheme in use. Such ghost cells are used to fetch information from related grids. The field variables are interpolated using high-order Lagrange polynomials, with the same order of accuracy as the internal scheme.

The overset grid method is developed within the DynHoLab code presented in Chapter 2 [131]. Several numerical applications are used to demonstrate the accuracy and efficiency of the proposed approach and its potential for complex geometries. The present work has been presented at the 32nd AIAA Applied Aerodynamic Conference 2014 [129].

4.2 Overset grid strategy

Figure 4.1, taken from [48], illustrates the concept of overlapping between two 1D grids. For the sake of clarity, the two grids have the same constant increment δx and are shifted by half a grid spacing. However this configuration can be generalized to any multidimensional grids. Grid 1 (on top) ended at index n, and grid 2 (bottom) starts at index 1. In order to allow a two-way communication between the grids for the propagation of information coming from grid left to right or right to the left, interpolation should be employed. The communication involves

interpolations from interior points (white points) of grid 1 toward interpolated points (black points) of grid 2. White points are the donor points, whereas black points are receiver points, sometimes referred to as ghost points. The number of ghost points is fixed by the width of the discretization stencil. Since Ref.[48] deals with aeroacoustics simulations using DRP schemes on eleven points, *i.e.* an eleven-point centered finite difference scheme, five ghost are added at the right of grid 1 and five ghost points on the left of grid 2 on the example of Fig. 4.1 to allow a two-way communication. The region between the first interpolation point of grid 2 and the last interpolation point of grid 1 is the overlapping zone. Extra costs due to interpolation operations and to the size of the overlap should be minimized. The overlapping zone should then be reduced to the minimum size required. This size depends on the interpolation used, its order of accuracy and whether it is explicit or implicit. When a donor point can also be a receiver point, the interpolation is said implicit. That means that at least one of the values used to perform the interpolation is an unknown variable to be interpolated on the other grid. The two-sided interpolation processes are coupled and require the solution of a linear system of equations, which can be expensive [44]. Nevertheless, an implicit interpolation allows a reduced overlapping zone, and becomes pertinent for complex geometries, where the gap between two bodies is small for instance. In this thesis, we consider the use of overlapping structured grids in conjunction with centered high-order Lagrangian explicit interpolation. In order to use at

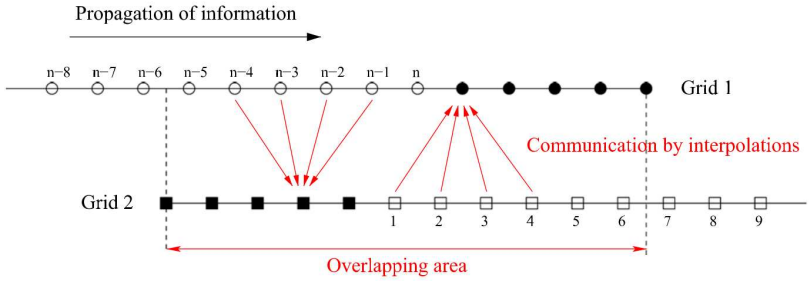


Figure 4.1: 1-D example of two overlapping grids. Each grid is both receiver and donor of information. The black points are the interpolation points, called ghost points. The white points are interior points on which derivatives are computed on a centered eleven-point stencil. Thus, five ghost points in the overlapping area are needed. The four arrows symbolize an explicit interpolation from a 4-point stencil. (Taken from [48])

best the overset capabilities and to apply them to various – directional and compact – scheme stencils, we develop a complete overset mesh generator called DynMesher plug-in. An example of structured overset mesh composed by two Cartesian grids with different refinement levels, connected by a polar grid, is provided in Figure 4.2.

4.2.1 Overset mesh generation

The generation of an overset mesh requires the following steps: 1) the generation and ordering of component grids; 2) the computation of the connectivity between component grids; 3) the assembly of the component grids; 4) the determination of the overlap regions and determination of the overset joins and holes (or blanked regions).

We define an overset mesh in D dimension(s) as a mesh composed of M ordered component grids $\{G_m\}_{m\in[1,M]}$ of the spatial domain $\Omega \subset \mathbb{R}^D$. The ordinal of the grid defines its priority in the overset assembly algorithm. For all analytical developments concerning numerical schemes and interpolation, we define a computational space in which all the grids will be considered to

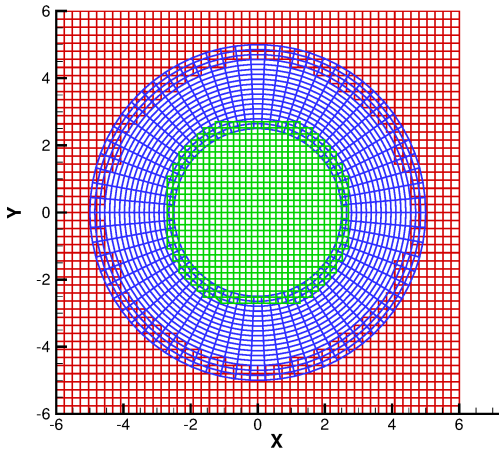


Figure 4.2: Example of overset mesh generated with the DynMesher code

be uniform Cartesian meshes, that is, for instance, on a domain $\Omega \in \mathbb{R}^3$

$$\begin{cases} \xi_{ijk} = \xi_i = i\,\delta\xi = i \\ \eta_{ijk} = \eta_j = j\,\delta\eta = j \\ \zeta_{ijk} = \zeta_k = k\,\delta\zeta = k \end{cases} \tag{4.1}$$

considering that $\delta\xi = \delta\eta = \delta\zeta = 1$. Otherwise, the grids are defined in the physical space, in which the 3-D coordinates will be denoted (x, y, z). In this physical space, the grid steps are considered to be of the same order, say $O(h)$. Furthermore, denoting by E the generic cell of G_m, $|E|$ being its area, we assume

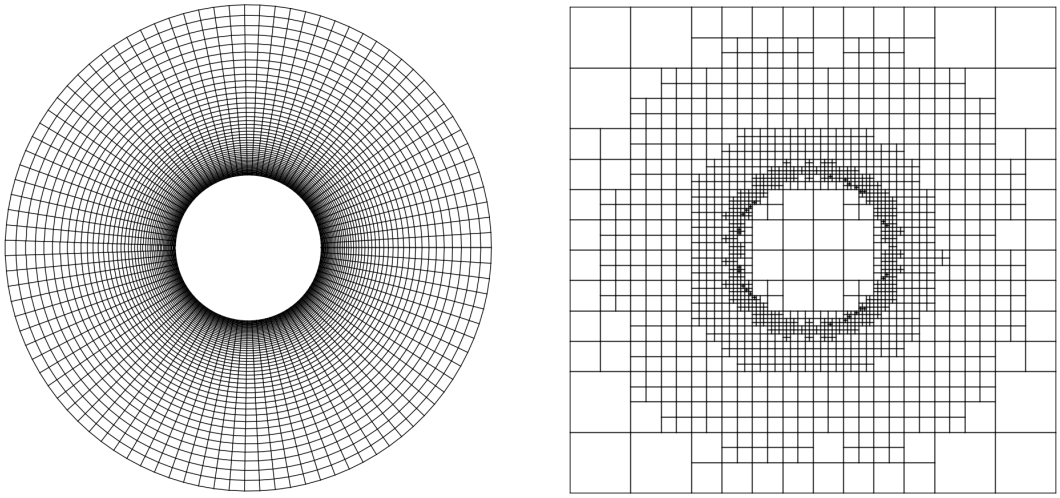
$$\exists\ (C_1, C_2) \in \mathbb{R}^+ \times \mathbb{R}^+, 0 < C_1 \leq \sup_{E \in G_m} \frac{h^D}{|E|} \leq C_2 < \infty \tag{4.2}$$

which corresponds to the fact that no vanishing area elements are present, as well as no unreasonably stretched cells.

One of the most time consuming tasks of the assembly algorithm is the determination of the connectivity of points between component grids. Considering a point $P \in G_m$, this step consists in finding whether or not P lies in the grid $G_{m'}$ and if it does, to find the base point $Q \in G_{m'}$ of the cell of $G_{m'}$ in which lies the point P. In 1-D, we then define the connectivity $c_{m'}(P)$ of the point $P \in G_m$ in the grid $G_{m'}$ as

$$c_{m'}(p) = i \quad \text{if} \quad x_i \leq x_P \leq x_{i+1}$$

If working on Cartesian meshes or working in the reference space of a grid known by its coordinate transformation, the search algorithm could consists in a dichotomy procedure in each direction of space. When working on general structured meshes, and in order not to apply a simple brute force algorithm, we precondition the search by a wrapping quadtree in 2-D and octree in 3-D [74, 34]. This roughly correspond to a dichotomy algorithm in each direction. The storage and algorithms relative to k-D trees is not detailed in this report. We invite the reader to refer to Refs. [74, 34]. Let us focus on the 2-D configuration represented in Figure 4.3. The quadtree is initialized by a first Cartesian bounding box wrapping grid $G_{m'}$. By looping on points of grid $G_{m'}$, the quadtree is filled with the index of the cells lying in the quadrants under a constrain of say N points per quadrant. To determine whether or not a cell lie in a quadrant, we compute

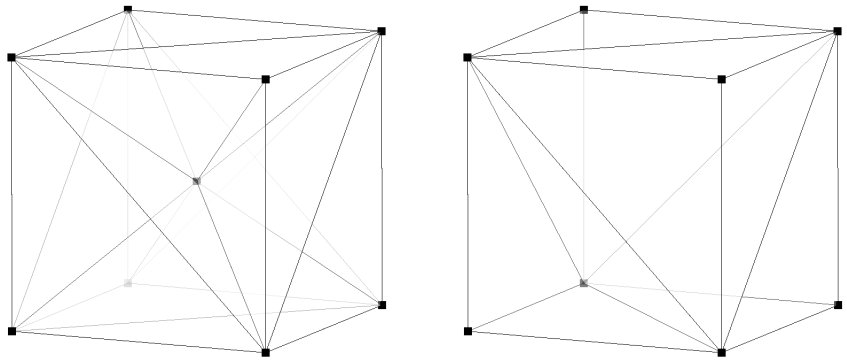


(a) Cylindrical mesh (b) Quadtree preconditioning

Figure 4.3: Cylindrical mesh preconditioned by a quadtree data-structure (for the example, the number of cells per quadrant is set to 15)

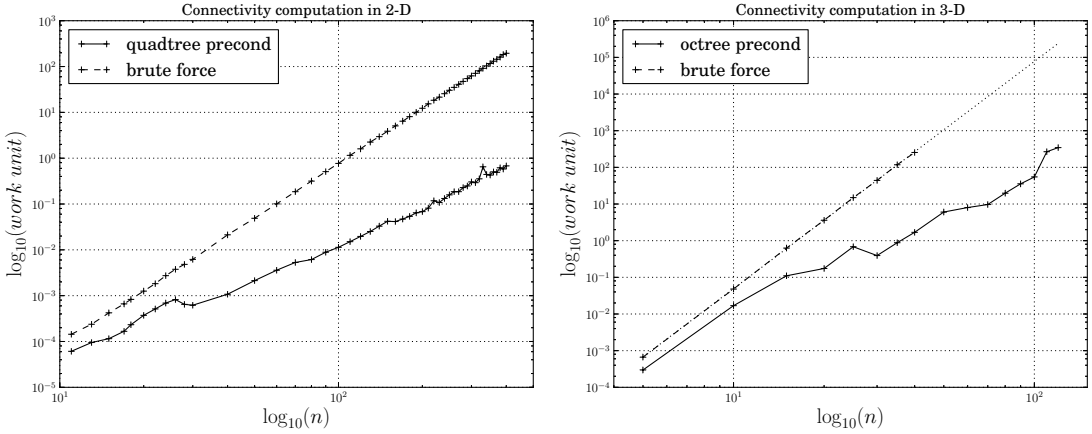
the Cartesian wrapping box of the cell and add the cell to each quadrant intersecting the cell wrapping box. This procedure then consists in preconditioning the donor grid $G_{m'}$ to accelerate the search. As a second step, this preconditioning tree is used inside the loop on points P of grid G_m. The work is reduced firstly to the determination of the quadrant in which the points P lie and secondly to a loop on cells of grid $G_{m'}$ lying in or intersecting the quadrant. For each cell of grid $G_{m'}$, a condition for the point P to belong to the cell should be employed. In 1-D, it is obvious but the 2-D and 3-D cases should not be underestimated. In 2-D, we determine if the point P is in the upper-right (positive-positive) quadrant of the basis formed by two opposite pairs of vectors. In a cell ABCD (closed cell with points in this order), the two pairs of vectors could be for instance (AB, AD) and (CD, CB). In 3-D, the problem is much more complicated. In order to simplify the condition, it is easier to work on a tetrahedron in which the condition for belonging is to be on the positive-positive-positive octant of a local base formed by three vectors of the tetrahedron and asking the sum of the local coordinates to be lower than the unity. The problem left is the filling of a hexahedron using tetrahedra. Trying to answer to the eighteenth problem of Hilbert, the work of Goldberg [82] give the five known space-filling tetrahedra when mirror images are not allowed. Two of them give a hexahedron space filler. The first one is easy to represent in space: it consists in cutting each face of the hexahedron in two triangles and form tetrahedra with a point inside the hexahedron (for instance the barycenter), this leads to twelve tetrahedra (see Fig. 4.4(a)). The second one consists in a space filling of the hexahedron using five tetrahedra. The theory is much more complicated and can be found in Ref. [82]. It is represented in Figure 4.4(b).

To evaluate the gain brought by this preconditioning procedure, let us determine the number of operations in d dimensions of space. A brute-force algorithm would require d loops to sweep all the cells of grid G_m around d loops to sweep the cells of grid $G_{m'}$. By using n points in each directions of both grids, we reach $0(n^{2d})$ tests of belonging in d dimensions of space, each requiring two conditional statements in 2-D and at most ten conditional statements in 3-D. This brute-force algorithm leads to $0(n^6)$ statements in 3-D. Using the preconditioning procedure, the algorithm require d loops to sweep all the cells of grid G_m around a finding procedure in the tree



(a) (b)

Figure 4.4: Two tetrahedron space fillers of hexahedron, (a) using twelve tetrahedra, (b) using five tetrahedra



(a) (b)

Figure 4.5: Comparison between brute-force and preconditioned algorithms

(equal to $d\log_2(n)$) in d dimensions and then, in the worst case, N tests (N being the number of cells stored in a quadrant or octant of the tree). This preconditioned algorithm leads to $0(\log_2(n)n^3)$ conditional statements in 3-D. To conclude, the determination of the connectivity of point P in grid $G_{m'}$ is of the order of d-times 1-D connectivity. Considering the determination of the connectivity of a 100^3 grid in a 100^3 grid, the preconditioned algorithm is about 10^6 times faster. The number N of cells stored in each quadrant or octant of the k-d tree should be adjusted on each computer (depending on the size of the RAM) to minimize the turn around time of the total algorithm (preconditioning and search procedure).

From the connectivity information, grids should be assembled to form an overlapping mesh. The size of the overlap depends on the interpolation stencil and the space discretization stencil for inside points. In this work, we use an assembly algorithm inspired from Ref. [44] with special care to treat correctly the corners of the grids for compact schemes. The algorithm consists in the following steps:

Initialization To each grid is attached a flag array. It represents the status of points (or cell-centers), whether they are interpolated ($flag < 1$), discretized ($flag > 1$), exterior (also named blanked, $flag = 0$) or boundary conditions ($flag = \pm 1$). It is initialized to

$(N_G + 2)$ with N_G the number of grids.

Application of boundary conditions The flag is set to the value -1 for non-permeable boundary conditions (*e.g.* wall), $+1$ for permeable boundary conditions (*e.g.* inflow, outflow, non-reflexion).

Contamination of walls In this step, we visit all points marked as non-permeable boundary condition and mark points of other grids that lie close to these points.

Blanking Then, to determine the blanked regions in each grid, the blanked points are used to determine a closed shape and any points of the grid that are within this shape are marked as blanked points.

Detect related grid This step consists in finding which grid, if any, each point on each grid can be interpolated from. Starting from the highest numbered grid and working down to the lowest numbered grid, each grid point on G_i is examined to find the highest grid G_j with $j > i$, if any, from which it can be interpolated. If the point cannot be interpolated from a higher grid and is not a valid discretization point then we find the highest G_j with $j < i$, if any, from which it can be interpolated. If the point cannot be interpolated from a lower grid either, it is marked as an exterior point. At the end of this step every point is marked either as an exterior point, as an interior point, or as an interpolated point with the flag indicated the ordinal of the donor grid.

Interpolation toward lower grids Starting for the highest numbered grid, we consider points with the flag strictly inferior to the ordinal. For these points, we mark with opposite value the connected point in the related grid.

Interpolation toward higher grids First, we should detect a deficit of interpolation points. Starting from the lowest grid to the highest one, we verify that the number of cells between two communication points on a grid should be higher than four times the number of ghost-cells needed in the problem. The number of ghost-cells is determined from the stencil size (ghost = (stencil-1)/2). Secondly, we extend negative values to the size of the interpolation stencil. Thirdly, we transform the negative flags to the ordinal value. Fourthly, we consider the points with a flag strictly superior to the ordinal. If these points contain a flag equal to the ordinal in their stencil, we mark them to their opposite value. Else, we mark them to zero (exterior points or blanked points). Fifthly, we consider points with negative flags and we check that the corresponding points from which they are interpolated are discretization points. This is done to avoid implicit interpolation.

All details are not indicated here. For example, special treatment is considered when a boundary condition is shared between multiple grids.

4.2.2 Interpolations

We consider the use of interpolation for the communication between overlapping grids. Stability issues can arise when considering non-centered stencils [65], or extrapolation [175]. In the following, only centered interpolations with an even number of stencil points are considered, so that no stability issue arises. Another issue is the conservative character of the interpolation. Conservative interpolation schemes [35, 45, 184] are difficult to generalized to high-order multidimensional applications. For external aerodynamics problems and/or smooth problems, it is commonly accepted to use non-conservative interpolations.

In this section, we deal for simplicity with 2-D structured grids. In the algorithm used to assemble grids and create a composite mesh, the grids are ordered from the grid with lowest priority to the one with highest priority. Let us deal with a grid 1 and a grid 2. Grid 1 is then

the "back" grid and grid 2 is the "shape" grid. The grids are assembled so that the boundary points of grid 2 are interpolated from grid 1 and some points of grid 1 are interpolated from grid 2 to make the communication two-way. Depending on where the function to be interpolated is defined, the points considered could be one of: the vertices of the grids, the cell-centers of the grids (or dual mesh vertices), the face centers of the grids. To illustrate the interpolation procedure, we consider for example a warped cell $P_1P_2P_3P_4$ of grid 2 and a point P of the grid 1 (see Figure 4.6). The aim is to interpolate a function f that is known at vertex of the cell $P_1P_2P_3P_4$ to the point P. All strategies presented hereafter use the canonical polynomial base

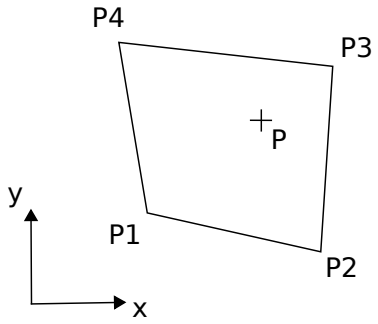


Figure 4.6: Sketch of the interpolation problem

for the interpolation method resulting in a Lagrangian interpolation of the desired order N.

$$f_P(x,y) = \sum_{r,t=0,N} c_{r,t} x^r y^t \tag{4.3}$$

The order depends on the interpolation stencil. To achieve such an interpolation, we use transfinite interpolation where the mesh is not distorted and we use the iso-parametric mapping method where the transfinite interpolation fails. The iso-parametric mapping was introduced in Ref. [32], extended to high-order in Ref. [163] and used for aeroacoustics simulations for instance in Ref. [48].

Using the previously defined notations, we suppose to know the values of the function f at points P_1, P_2, P_3, P_4. In the curvilinear space, the four points have coordinates

$$P_1 = (x_1, y_1),\ P_2 = (x_2, y_2),\ P_3 = (x_3, y_3),\ P_4 = (x_4, y_4).$$

We want to interpolate the value of a function f at point P. If we have

$$x_1 = x_4, \quad x_2 = x_3, \quad y_1 = y_2, \quad y_3 = y_4 \tag{4.4}$$

then the interpolation can be seen as the combination of two linear interpolations:

$$\begin{aligned} f(x,y) = \Big[(x_2 - x_1)(y_2 - y_1)\Big]^{-1} \cdot \Big(& f(P_1)\cdot(x_2 - x)(y_2 - y) + f(P_2)\cdot(x - x_1)(y_2 - y) \\ & + f(P_3)\cdot(x_2 - x)(y - y_1) + f(P_4)\cdot(x - x_1)(y - y_1) \Big) \end{aligned} \tag{4.5}$$

When dealing with curvilinear meshes, the configuration stated above is only a particular case. Let us consider two spaces, the curvilinear space (or physical space) where the mesh is defined, and the reference space. For a bilinear interpolation, in the reference space, the cell studied is a square and its points have coordinates

$$P_1 = (0,0),\ P_2 = (1,0),\ P_3 = (1,1),\ P_4 = (0,1).$$

Thanks to such a transformation, the interpolation can be carried out on the unit square $(\xi, \eta) \in [0,1]^2$ where the interpolation writes:

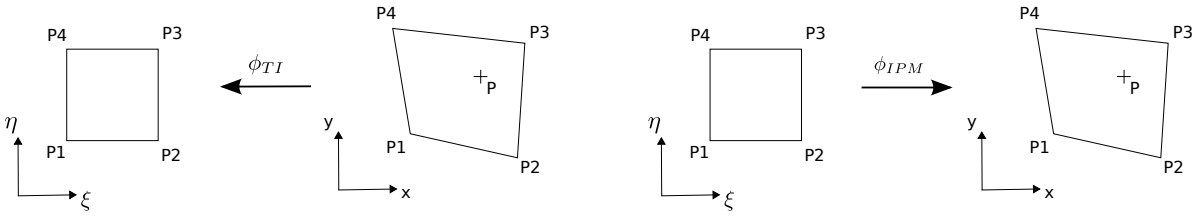
$$f(\xi,\eta) = \begin{pmatrix} 1-\xi & \xi \end{pmatrix} \begin{pmatrix} f(0,0) & f(1,0) \\ f(0,1) & f(1,1) \end{pmatrix} \begin{pmatrix} 1-\eta \\ \eta \end{pmatrix} \tag{4.6}$$

In order to be able to interpolate in the reference element, we find the transformation of the curvilinear stencil onto the reference element. This consists in solving the system

$$\forall (i,j) \in [0,N]^2, (\xi_{ij}, \eta_{ij}) = \Big(\sum_{r,t=0,N} a_{rt}^{(x)} x_{ij}^r y_{ij}^t, \sum_{r,t=0,N} a_{rt}^{(y)} x_{ij}^r y_{ij}^t \Big) \tag{4.7}$$

with (x_{ij}, y_{ij}) the coordinates of the points of the stencil in the curvilinear space and (ξ_{ij}, η_{ij}) the coordinates of the points of the stencil in the reference space (the unit square for a bilinear interpolation). The system can be written $F = M \cdot I$ and the matrix M has an associated application ϕ which transform the interpolation stencil in the curvilinear space on the interpolation stencil in the reference space. Using this transformation, we can get the position of the point P in the reference element and apply the Lagrange interpolation. Writing the Lagrange interpolation $\mathcal{L}$ and using $\phi_{TI}(P) = Q$, we have $\mathcal{L}(P) = \mathcal{L}(\phi_{TI}^{-1}(Q))$ and $I = M^{-1}F$. In summary, this method consists in finding a transformation using polynomial interpolations from the curvilinear domain on a reference Cartesian space. The interpolations are computed directly in the curvilinear space using a matrix inversion method (see Figure 4.7(a)). The size of the matrix to be inversed depend on the size of the stencil and thus on the order of the interpolation. One of the drawbacks of this method is that, depending on the regularity of the mesh and the size of the interpolation stencil, the matrix could be ill-conditioned. For example, this is the case for a bilinear interpolation when $x_1 = x_3$ and $y_2 = y_4$. A first solution to get out of this situation is to apply a rotation when the conditioning number of the matrix is too high.

A satisfactory method for all configurations is the iso-parametric mapping method. We still consider two spaces, the reference space and the curvilinear space. In this method, the transformation searched is the other way, going from the reference space onto the curvilinear one (see Figure 4.7(b)). This consists in solving the system



(a) transfinite interpolation

(b) Iso-parametric mapping

Figure 4.7: Principles of the transfinite interpolation and iso-parametric mapping

$$\forall (i,j) \in [0,N]^2, (x_{ij}, y_{ij}) = \Big(\sum_{r,t=0,N} a_{rt}^{(x)} \xi_{ij}^r \eta_{ij}^t, \sum_{r,t=0,N} a_{rt}^{(y)} \xi_{ij}^r \eta_{ij}^t \Big) \tag{4.8}$$

with (x_{ij}, y_{ij}) the coordinates of the points of the stencil in the curvilinear space and (ξ_{ij}, η_{ij}) the coordinates of the points of the stencil in the reference space (the unit square for a bilinear interpolation). The problem is then to find the coordinates of the point P in the reference space. This is done using a Newton method. As done in the previous section, we introduce the application ϕ_{IPM} associated with the matrix M of the system (4.8). We have $\phi_{IPM}(Q) = P$.

Writing $X = (x, y)$ the coordinates of the point P in the curvilinear space and $\Xi = (\xi, \eta)$ the coordinates of the point Q in the reference space, we have $\phi_{IPM}(\Xi) = X$ and the newton method writes

$$\mathcal{F}(X, \Xi) = \phi_{IPM}(\Xi) - X = 0 \tag{4.9}$$

which can also be written:

$$\Xi^{n+1} = \Xi^n - \left(\frac{(\phi_{IPM})_i}{\Xi_j} \right)^{-1} \mathcal{F}(X, \Xi^n) \tag{4.10}$$

with the Jacobian matrix M_{ij} invertible as long as the mapping is one-to-one (by construction), the cell is not degenerate, and the cell is convex. Typically, the initial guess $\Xi_0 = (1/2, 1/2)$ and the newton is converged in ~ 5 steps. Once a converged value of Ξ has been obtained, the desired interpolated value at P is evaluated by computing $\mathcal{L}(\Xi)$.

4.2.3 Hole cutting treatment in the implicit algorithm

A feature of primary importance in the overset strategy is the ability to handle complex geometries and then hole cutting in background grids. When dealing with explicit schemes, hole cutting should be taken into account in the Cartesian solver to prevent the application of the numerical scheme in use in the interior domain inside the blanked regions. In the case of steady computations or unsteady computations of slowly varying flows, implicit schemes are preferred to explicit ones, and a hole cutting procedure should be set up in the implicit stage of the CFD code. For the use of RBC schemes of high-order accuracy, we develop hereafter the management of hole cutting management for scalar implicit algorithms.

To fix ideas, we consider a 1-D system of size N_x and the use of a three point scheme leading to the notation

$$M \Delta w^n = -\Delta t R(w^n) \tag{4.11}$$

with M a matrix of size $N_x \times N_x$ and $\Delta w^n = w^{n+1} - w^n$ the state vector increment. The hole cutting operation consists in nullifying the extra-diagonal terms in the blanked region enlarged of a rind of cells – which number depends on the stencil of the inner numerical scheme in use – and also in replacing the diagonal terms by identity, as presented in the following matrix

$$M = \left[\begin{array}{ccccccccccc}
\ddots & \ddots & \ddots & & & & & & & & \\
 & c_x^- & c_0 & c_x^+ & & & & & & & \\
 & & c_x^- & c_0 & c_x^+ & & & & & & \\
\hline
 & & & 0 & I & 0 & & & & & \\
 & & & & 0 & I & 0 & & & & \\
\hline
 & & & & & c_x^- & c_0 & c_x^+ & & & \\
 & & & & & & c_x^- & c_0 & c_x^+ & & \\
 & & & & & & & \ddots & \ddots & \ddots &
\end{array}\right]$$

in which the blanked region augmented by ghost-cells correspond to the lines of the matrix contained between the solid lines. In 2-D, the matrix M contains five diagonals and the lines concerned by the hole cutting are localized around the lines $i + (N_x - 1)j$. In addition to the modification of the matrix M, the explicit increment $-\Delta t R(w^n)$ corresponding to the rind of cells created around the blanked region should be interpolated from the related blocks thanks to the overset assembly. This last operation take into account the volume of cells in the case of a finite-volume framework.

4.2.4 Implementation details

As said previously, a complete mesh generator and mesh assembly problem has been created as a plug-in for the DynHoLab code. This Python package is named DynMesher and is composed by following sub-packages:

- *composite*: responsible for the assembly of composite meshes, it manages the connectivity computation, overlapping, conformal connection and non-conformal connection between grids, grid-to-grid interpolation
- *geometry*: contain functions for lengths, mesh momentums, dual-mesh, projections, check direct or indirect mesh, intersections.
- *interpolation*: from the connectivity information, this sub-package is responsible for the computation of the interpolation coefficients (transfinite interpolation and iso-parametric mapping)
- *plot*: manages special plots of composite grids (with blanked regions, with interpolation cells) writing Tecplot output files or using matplotlib to create plots.
- *structured*: creates simple structured grids or modify them (Cartesian, cylinder, multi-block sphere, transfinite interpolation, Coons Patch, extrusion, stretching functions, translation, rotation, permutation, symmetry)
- *unstructured*: only converts structured grids to unstructured ones for overlapping meshes rendering with blanked regions.

Along with these sub-packages, there are simple modules to manage windows of mesh or to fill automatically the CGNS-tree (please refer to chapter 2).

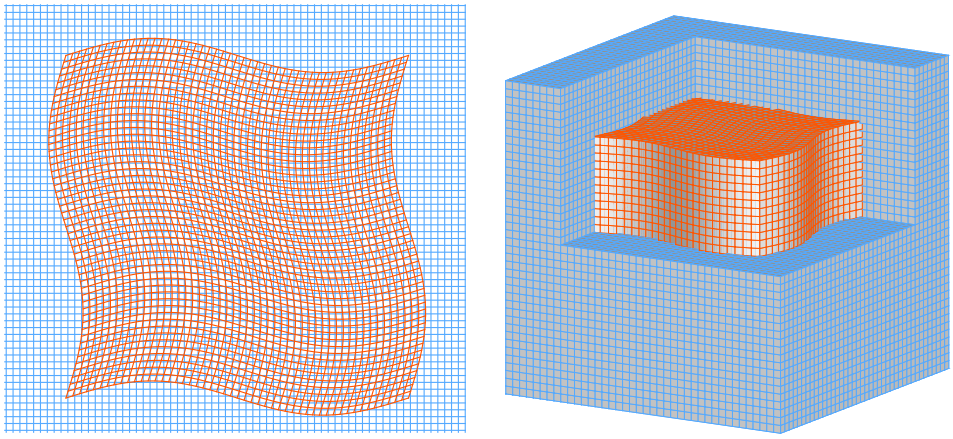
4.3 Numerical experiments

In order to validate the overset strategy with RBC schemes, we first provide 2-D and 3-D grid-to-grid interpolation validations, followed by a 3-D scalar advection test-case. Then, to demonstrate the capabilities of this implementation, we simulate linear advection of Gaussian hump. This case is used to illustrate dynamic creation/destruction of grids employed to follow the hump. Finally, we apply the overset algorithm to the computation of inviscid transonic flows over single or tandem NACA0012 airfoils.

4.3.1 Grid-to-grid interpolation

As a first test case, we consider the interpolation of data generated by some multi-dimensional analytic functions from one grid onto another one. The case was used in Ref. [163] to validate their own implementation of the interpolation. The aim is to isolate the interpolations to validate their implementation, without the influence of the other numerical methods such as temporal integration or spatial differentiation.

The grids used for this test case are presented in Figure 4.8. Both in the 2- and 3-dimensional case, the pair of grids consists in an curvilinear foreground grid overlapping a Cartesian background grid. A known analytical function is initialized on the background grid and then interpolated on the foreground grid. The analytical function can also be initialized on the foreground grid so as to study the error of interpolation on the foreground grid.



(a) (b)

Figure 4.8: Meshes used for the grid-to-grid interpolation test-case (a) 2-D grids (b) 3-D grids

The analytical functions considered are:

$$f1(x,y) = \cos(\omega x)\ \sin(\omega y) \tag{4.12}$$

$$f2(x,y) = (10.02)^2\ x\ \exp\left(\frac{-\sqrt{x^2+y^2}}{2}\right) \tag{4.13}$$

$$f3(x,y) = \frac{27\ x}{x^4+9(y^2+9)}\cos\left(\frac{20x}{y+3}\right)\cosh\left(\frac{x}{3}\right)\sinh\left(\frac{y}{3}\right) \tag{4.14}$$

$$f4(x,y,y) = \cos(\omega x)\ \sin(\omega y)\ \cos(\omega z) \tag{4.15}$$

with $\omega = \pi/2$. The explicit centered Lagrangian interpolations are used as described in Section 4.2.2. The accuracy of these interpolations are shown in Figure 4.9 where we plot the L_2-error norm as a function of the grid spacing. The orders of accuracy found numerically are very close to the formal order of the methods.

4.3.2 Helicoidal advection of a Gaussian pulse

We consider again the test case of Sec. 3.1.1. Computations are performed using a series of composite Cartesian meshes assembled using the DynMesher code and using RBC schemes [112, 56] of several accuracies. The composite meshes used for the study contain two Cartesian blocks with sizes $(26^3, 13^3)$, $(32^3, 16^3)$, $(48^3, 24^3)$, $(64^3, 32^3)$, expressed as numbers of cells. An example of composite structured mesh is shown on Figure 4.10(a) on which the masked cells of the background grid are plotted. Since we use regular Cartesian grids for this steady problem, we expect to recover the nominal convergence order upon mesh refinement, if numerical interpolations between neighboring domains are of the same order of accuracy of the inner point scheme. Figure 4.11 shows that the L_2-norm of the error with respect to the exact solution converges with a slope that is found equal to the nominal convergence order, which validates the present implementation.

4.3.3 Circular advection of a hump

Then, we consider again the test case previously discussed in Sec. 3.1.2. Results presented on figure 4.12(a) correspond to simulations up to time $t = 1$, when the exact solution of the

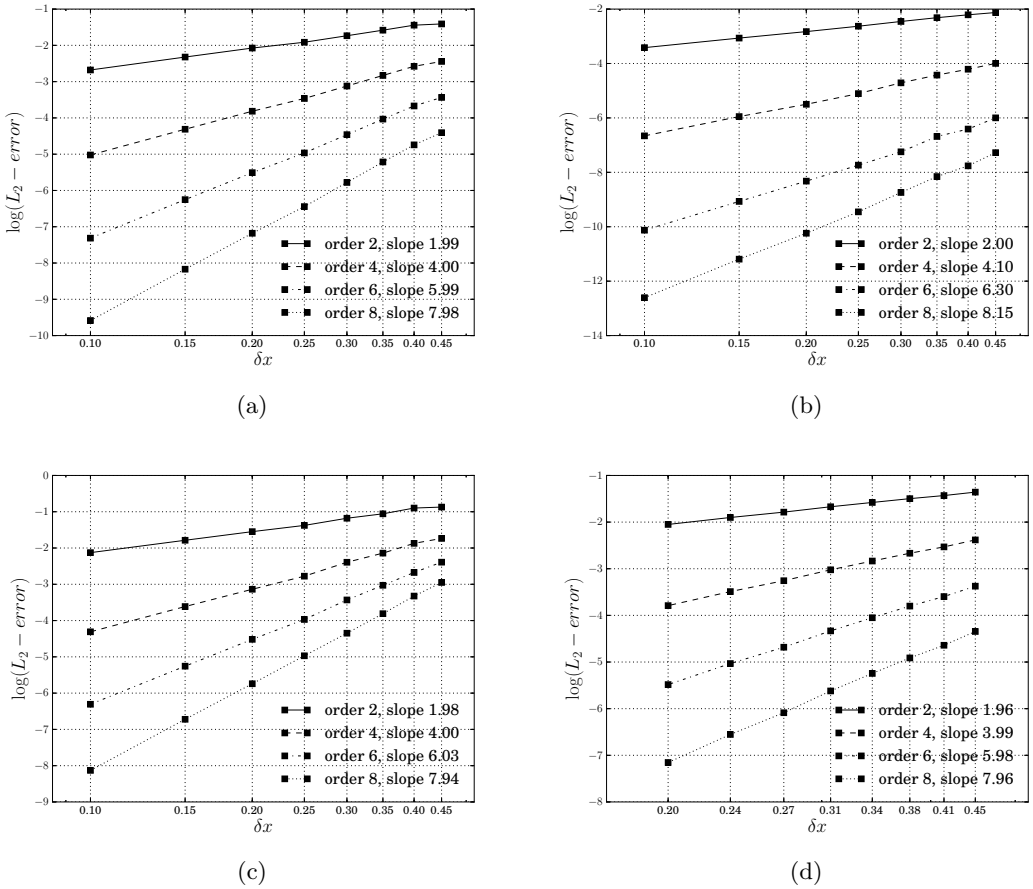
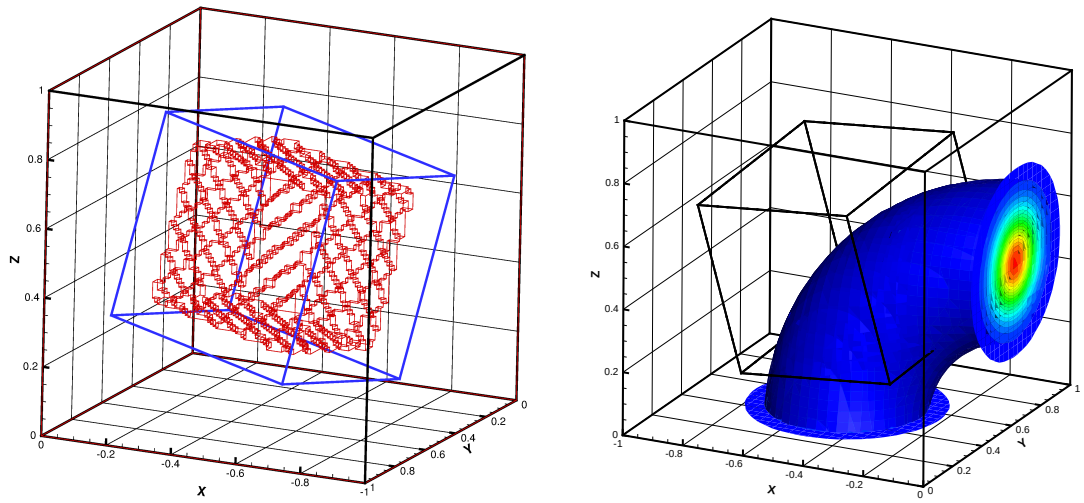


Figure 4.9: Study of the grid convergence of interpolations of varying order of accuracy for various analytical functions: (a) function f_1 (b) function f_2 (c) function f_3 (d) function f_4

problem returns to its initial position. All the schemes are applied on a composite overset grid composed by a $N \times N$ grid of size $[x_h - 0.4, x_h + 0.4] \times [y_h - 0.4, y_h + 0.4]$ with $N \in \{12, 16, 20, 24, 28, 32\}$ and (x_h, y_h) the coordinates of the maximum location of the hump. The number of cells of this fine grid corresponds to the same space discretization as the single-block counterparts presented in chapter 3, that is $N \in \{30, 40, 50, 60, 70, 80\}$ on a single-block. The fine grid is updated every $\delta t = 0.02$ and it overlaps a background grid coarser by a factor of 1.9, leading to $N \in \{16, 21, 26, 31, 37, 42\}$ for the background grid discretization. The time derivative is approximated by a gear scheme solved by Newton sub-iterations with a physical time-step $\Delta t = 0.0025$. At each physical iteration, the L_2-norm of the sub-iteration residual normalized by its initial value is driven down to 10^{-4}. All boundary values are obtained from extrapolation of the interior values.

Figure 4.12(a) shows the shape and iso-contours of the hump computed with the seventh-order RBC scheme at four positions. To check the spatial order of accuracy of the schemes for an unsteady problem, we compute the error with respect to the exact solution at time $t = 0.05$ for solutions computed on a series of increasingly fine grids. The time steps are chosen as $\Delta t \propto \delta x^{o/2}$ with o the expected spatial order of accuracy, to rule out the second-order error due to the time approximation scheme. Let us notice that, even if the code evaluates the scheme for the cells of the background grid overlapped by the fine grid (blanked cells), the total number of cells is lower in the overlapping version of this test-case. Moreover, the assembly algorithm has negligible turn



(a) (b)

Figure 4.10: 3-D helicoidal advection (a) example of overset Cartesian mesh used for the error grid convergence study, (b) isovalues of u obtained with 5^{th} order RBC scheme on a $(32^3, 16^3)$ composite structured cell grid

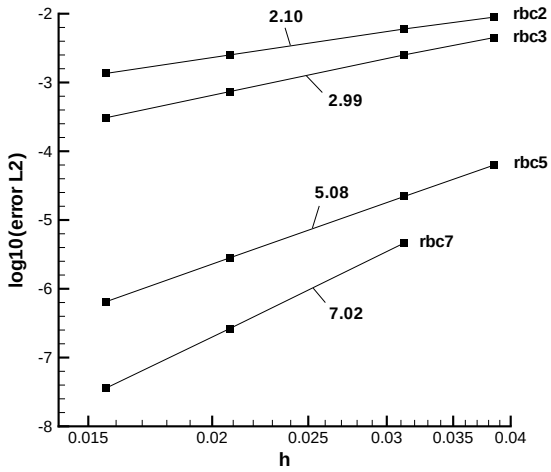
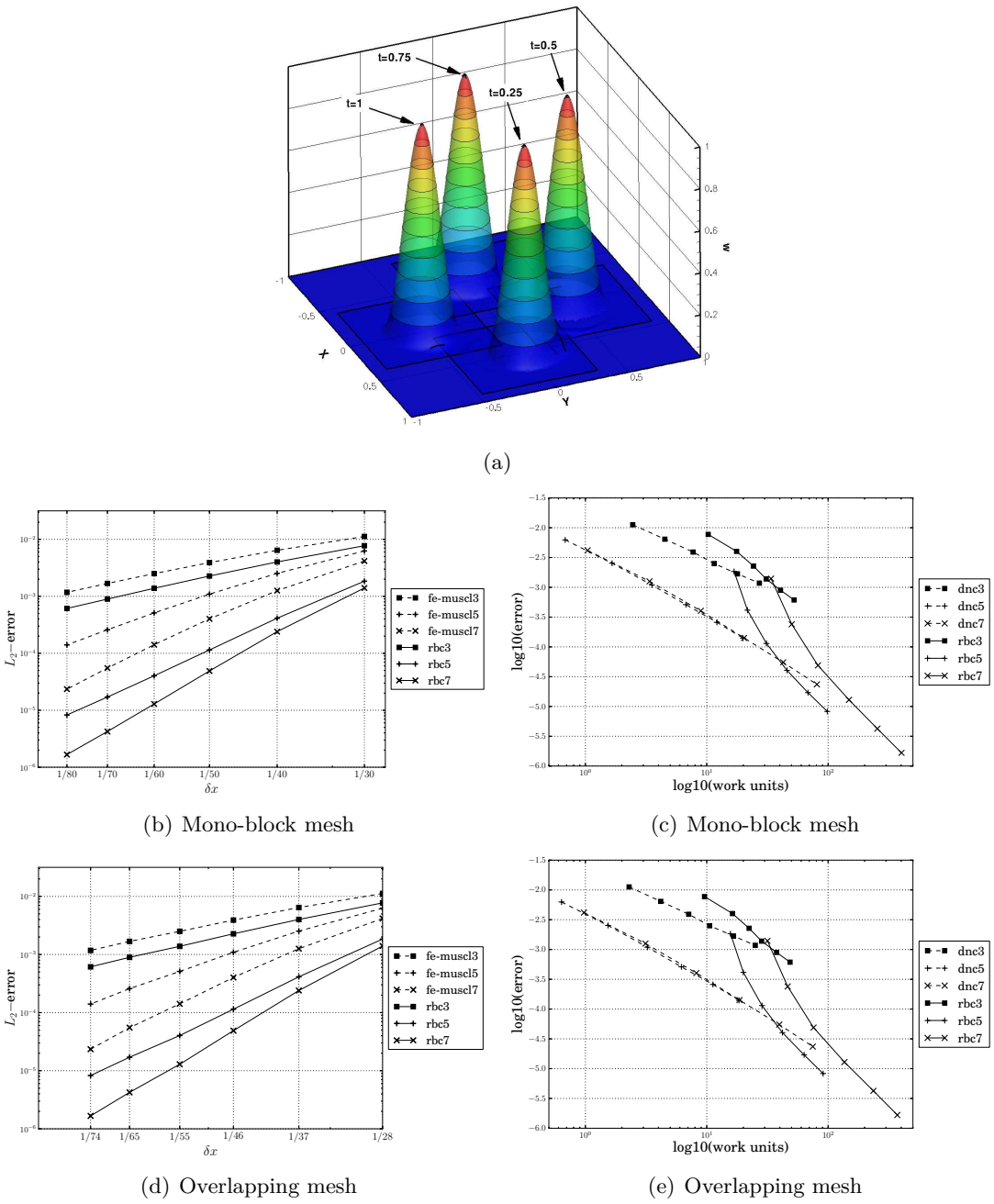


Figure 4.11: Study of the L_2 error as a function of the grid spacing for RBC schemes of 2^{nd}, 3^{rd} and 5^{th} order of accuracy

around time compared to time space and time discretization schemes. This is confirmed by the results in terms of efficiency of the strategy. On Figure 4.12(e), the convergence of the L_2-error and CPU time are given for both the mono-block and overlapping test-cases. The L_2-errors are nearly the same on both cases and the efficiency is reduced a little.

This test-case demonstrates the potential of this method for simulations using dynamic creation/destruction of grids such as AMR methods and validate the implementation in terms of memory management.



(a)

(b) Mono-block mesh

(c) Mono-block mesh

(d) Overlapping mesh

(e) Overlapping mesh

Figure 4.12: Circular advection of a hump, (a) Shape and iso-contours (from 0.05 to 0.6 by 0.05) of the hump at four positions computed with RBC7, (b) Convergence of the L_2-norm of the error upon mesh refinement on mono-block mesh, (c) Efficiency on mono-block mesh, (d) Convergence of the L_2-norm of the error upon mesh refinement on overlapping mesh, (e) Efficiency on overlapping mesh,

4.3.4 Inviscid transonic flow over a NACA0012

This test-case is the inviscid two-dimensional flows over a NACA0012 at transonic conditions $M = 0.8$ and $\alpha = 2.25^0$. The simulations are carried out using an overset grid composed by a combination of Cartesian and curvilinear grids (see Fig. 4.13). Precisely, the grid results

from the superposition of four Cartesian grids of 200×90, 140×140, 140×140, and 140×140 cells, an O-shaped body grid of 500×20 cells, and a background polar grid of 200×36 cells. A blanking algorithm is used to remove from the computation cells that are overlapped by grids with a higher priority level, so that in practice the solution is computed by using 75647 degrees of freedom. In the following, the overset grid solution is compared to that obtained on a single-block O-shaped grid made of 601×147 grid points (88347 degrees of freedom)

Figure 4.14a illustrates the computed isolines of the Mach number using an RBC scheme of 3rd-order accuracy. Shocks are captured in a sharp and almost non-oscillatory way. Present results are found to be in general good agreement with those collected at the first international workshop on High-Order Methods [13], and are superposed to those obtained for a single-block computation. This is confirmed by inspection of the Mach number distribution along the airfoil wall (see Fig. 4.14b) and of the aerodynamic coefficients, equal to $C_l = 0.3519$ and $C_d = 0.2265 \times 10^{-1}$ respectively. Compared to single block computations, the overlapping strategy allows a more efficient distribution of the degrees of freedom, and thus a better overall accuracy.

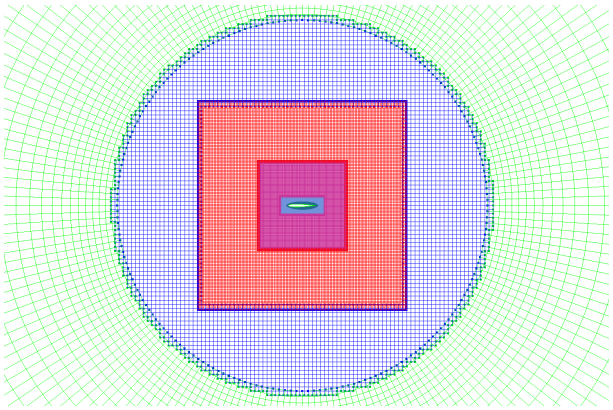
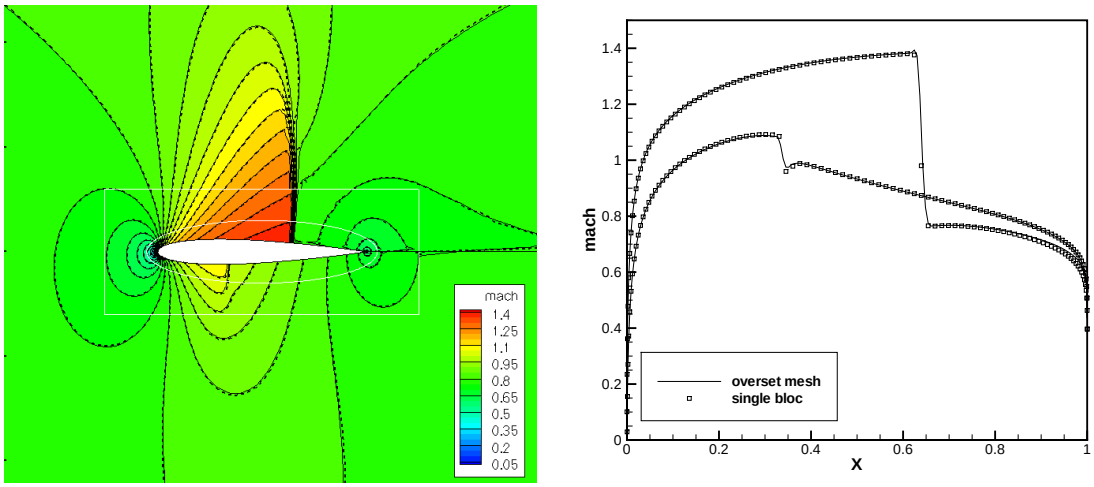


Figure 4.13: Detail of the overset grid used to compute transonic flow over a NACA0012.

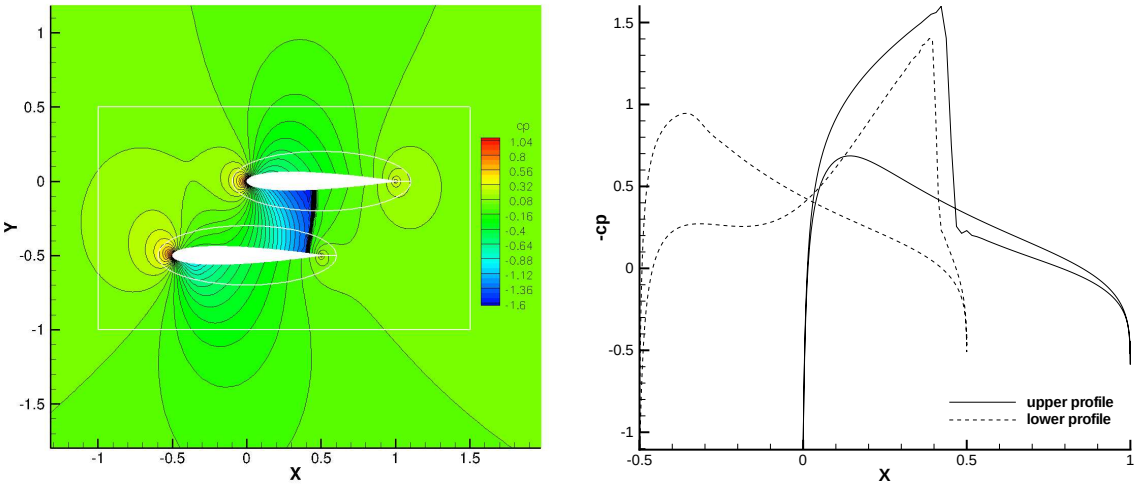


(a) (b)

Figure 4.14: Transonic flow over a NACA0012: comparison of an overset grid and a single-block grid solution. a) Mach isolines; b) wall distributions of the Mach number.

4.3.5 Flow past a tandem of airfoils

As a way to demonstrate the application of the present strategy to complex configuration, we compute a transonic flow around a two-element airfoil formed by two parallel NACA0012 airfoils. Concerning the geometric configuration, the airfoils are separated by half a chord in both directions. The flow conditions are characterized by a infinite Mach number of 0.7 and zero angle of attack. We refer to Ref. [116, 51, 127] for previous computations of the same case. The geometry is approximated by two elliptic meshes for the NACA0012 airfoils that overlap a series of overlapped Cartesian boxes. The total number of degrees of freedom is 274265 with a far-field located at 50 chords. The computation is carried out with the implicit version of the RBC solver of third order accuracy with a CFL number of 100 and requires 4000 iterations to converge the L_2-norm of the residual up to 10^{-10}. Contours of the pressure coefficient are displayed in Figure 4.15 as well as the wall distribution along the two airfoils. The lift and drag coefficients for the lower airfoil are respectively $c_l = 0.04796$ and $c_d = 0.04412$ and for the upper airfoil we get $c_l = -0.23944$ and $c_d = -0.01238$.



(a) (b)

Figure 4.15: Transonic flow over a two-element airfoil, (a) isolines of the pressure coefficient, (b) wall distributions of the pressure coefficient.

4.4 Chapter summary

In order to enable the handling of more complex geometrical configurations with RBC schemes of high-order accuracy, we have investigated the use of an overset grid framework. The overlapping strategy has been exposed with detailed attention on the algorithms used to obtain the connectivity between grids and on the computation of the interpolation coefficients. Finally, selected test-cases have been employed to validate the strategy and implementation and to demonstrate the potential of the method with for instance the dynamic creation/destruction of component grids. The last two-element airfoil configuration is usually not easy to compute using structured grids or require a significant effort to create the mesh. The overset strategy provide a satisfying turn around time of the global simulation – mesh generation and CFD computation – and is compatible with the use of high-order schemes and in particular RBC schemes despite their implicit nature when dealing with unsteady simulations.

4.5 Résumé de chapitre (french)

Afin de permettre la simulation sur des configurations géométriques complexes avec les schémas RBC d'ordre élevé, nous avons étudié l'utilisation d'une statégie de grilles recouvrantes. La stratégie de recouvrement a été exposé avec une attention détaillée sur les algorithmes utilisés pour obtenir la connectivité entre les grilles et sur le calcul des coefficients d'interpolation. Enfin, des cas-tests sélectionnés ont été utilisées pour valider la stratégie et la mise en oeuvre et démontrer le potentiel de la méthode, par exemple, avec la création/destruction dynamique de grilles recouvrantes composant le maillage. La dernière configuration avec deux profils d'ailes est généralement assez difficile à calculer en utilisant des maillages structurés ou exige un effort important pour créer le maillage. La stratégie de maillages recouvrants fourni un temps de restitution global très satisfaisant - génération de maillage et calcul MFN - et est compatible avec l'utilisation de schémas d'ordre élevé et en particulier les schémas RBC en dépit de leur caractère implicite lorsqu'il s'agit de simulations instationnaires.

Conclusion

As we discussed in the introduction, the CFD community have to face the increasing demand for more accurate solvers. The high-order methods being identified as one means of obtaining a higher accuracy in computer codes, one could consider their direct implementation within industrial codes. This is not a straightforward task, because of architectural constraints intrinsic to most existing CFD solvers. In this work, we have provided a reflection on the design and development of a computer code, the architecture of which would allow a better segmentation of the algorithms. Our solution is based on lists of functions acting on a structure of data, unlike most existing industrial codes usually based on the object-oriented paradigm. The advantages of such an architecture are undoubtedly the agility conferred on the code, enabling the treatment of different physical models as well as varied geometries, in conjunction with considerable simplicity, allowing a computer science novice to easily use the code for his research. Moreover, the right levels of both segmentation and generality of the code ensures a reduced maintenance by avoiding the multiplication of functions having the same goal (boundary conditions, calculation of metrics, implicit algorithms, ...). In conjunction with systematic tests and versioning, the structure chosen for the code makes possible the development of a complete CFD toolbox with very limited human resources. The programming languages selected for writing DynHoLab have also allowed its fast development along with a unified syntax, a documentation included in the code and local unit tests in Python modules. Parallel performance of this new numerical framework has been tested on national computer center resources and was demonstrated to provide a satisfactory scaling for structured tesselations of structured grids. As detailed in the summary of chapter 2, this work is the basis for other PhD theses currently ongoing in the DynFluid laboratory.

A second aspect treated in this thesis is the extension of high-order schemes to complex geometries, with focus on a family of residual-based compact schemes on structured grids. These were systematically compared to high-order directional non compact FE-MUSCL schemes. A series of validations going from the order of accuracy of the schemes on linear test cases, comparisons with other solvers on both subsonic and transonic flows over airfoils, validation of the functions forming the RANS solver, as well as the application to the turbulent ILES simulation of the flow past periodic hills show the extent of applications that can be treated with various numerical methods using DynHoLab code. Moreover, throughout the different validations and assessments proposed in the thesis, we showed that residual-based compact schemes exhibit better properties in terms of resolvability than non compact schemes of the same order of accuracy. On the development of high-order methods, specific points reviewed hereafter in the perspectives would require further improvements.

The specific strategy considered in this thesis for the extension of RBC schemes to complex geometries is their use in conjunction with a structured overlapping grid strategy. This allows to maximize the regions of the computational domain where the schemes are applied at their nominal order of accuracy. Depending on the discretization and interpolation stencils to be retained in the numerical strategy, existing assembly algorithms could not be able to treat the specific case of high-order RBC schemes. For this reason, a new mesh generator has been developed

as a plug-in of DynHoLab. Algorithms required for calculating connectivity information among different overlapping domains have been reworked to improve their computational efficiency.

Perspectives

Concerning the DynHoLab code, the best validation would be its longevity with a development community not limited to the DynFluid laboratory. Specifically on the code, further developments should be led on the parallelization in order to be able to balance the load on unstructured tesselations of structured grids, or even on meshes including overlapping grids. It would also be wise to replace the existing package responsible for the management of the python/CGNS tree data-structure (DynCGNS) by the one developed for the CGNS consortium by Poinot [139]. This replacement would enable direct interactions with tools such as Cassiopee [2]. Such code assemblies would be easy with any code sharing this Python/CGNS data-structure paving the way for coupled resolutions as fluid/structure interactions, or allow mesh adaption, rotating elements in overlapping meshes, ... About overlapping grids, the existing 3-D implementation in DynHoLab should be demonstrated. The current FV solver could be simplified to degenerate on the FD solver to minimize the CPU and memory usage in Cartesian grids. Also the solver could be optimized, rewriting loops on arrays, not to compute blanked dof. Another powerful strategy to be targeted in the future is the use of hybrid meshes having both structured and non-structured elements. This would enable intrinsic conservation in grid connections and thus could be used for internal flows such as turbomachinery. Concerning the current status of the code, developments are carried out on the physics sub-package to generalize the code to any equation of state. Other contributions concerning numerical schemes or pre- and post-processing are added as plug-in and thus non-intrusive in the code kernel.

On high-order schemes, further improvements can be expected through additional research on the treatment of complex geometries, boundary conditions, and implicit algorithms. In the specific case of RBC schemes, in order to improve accuracy and efficiency for fast unsteady flow computations, a reasonable choice would be a Runge-Kutta time integration scheme. Nevertheless, direct coupling of RBC schemes with Rung-Kutta (RK) schemes would lead to the inversion of broad-band matrices at each RK stage, because of the presence of the time derivative in the RBC dissipation operator. A good option could be that of writing the dissipation at a preceding RK stage, while modifying RK coefficients to ensure the right order of accuracy in time. This kind of approach was studied by Abgrall and Ricchiuto [16] for Residual Distribution schemes. With these advances, one would consider using the code for LES simulations on few thousand processors.

Conclusion (french)

Comme nous l'avons indiqué dans l'introduction, la communauté de la mécanique des fluides numérique (MFN) doit faire face à une demande croissante pour des solveurs plus précis. Les méthodes d'ordre élevé étant identifiés comme un moyen d'obtenir une plus grande précision dans les codes informatiques, on pourrait envisager leur mise en œuvre directe dans des codes industriels. Ce n'est pas une tâche simple, en raison de contraintes architecturales intrinsèques à la plupart des solveurs de MFN existants. Dans ce travail, nous avons fourni une réflexion sur la conception et le développement d'un code informatique, dont l'architecture permettrait une meilleure segmentation des algorithmes. Notre solution est basée sur des listes de fonctions agissant sur une structure de données, contrairement à la plupart des codes industriels existants, généralement basés sur le paradigme orienté objet. Les avantages d'une telle architecture sont sans aucun doute l'agilité conférée sur le code, permettant le traitement de différents modèles physiques, ainsi que des géométries variées, en conjonction avec une simplicité considérable, ce qui permet à un novice en informatique de pouvoir utiliser facilement le code pour ses recherches. De plus, les bons degrés de segmentation et de généralité du code assure une maintenance réduite en évitant la multiplication des fonctions ayant le même objectif (conditions aux limites, le calcul des métriques, des algorithmes implicites, ...). En conjonction avec des tests systématiques et la gestion de versions du code, la structure choisie pour le code a rendu possible le développement très rapide d'une véritable boîte à outils assez complète pour la MFN avec des ressources humaines très limitées. Les langages de programmation sélectionnés pour écrire DynHoLab ont également permis son développement rapide avec une syntaxe unifiée, une documentation incluse dans le code et des tests unitaires locaux inclus dans les modules Python. Les performances en parallèle de ce nouveau code ont été testées sur les ressources du centre de calcul national et ont démontré le passage à l'échelle satisfaisant pour des maillages multi-blocs structurés des grilles structurées. Comme indiqué dans le résumé du chapitre 2, ce travail est la base pour d'autres thèses de doctorat en cours actuellement dans le laboratoire de DynFluid.

Un deuxième aspect traité dans cette thèse est l'extension des schémas d'ordre élevé à la géométrie complexe, en mettant l'accent sur une famille de schémas numériques compacts basés sur le résidu sur des maillages structurés. Ceux-ci ont été systématiquement comparés à des schémas numériques d'ordre équivalent mais non compacts et directionnels (FE-MUSCL). Une série de validations allant de l'ordre de précision des schémas sur des cas linéaires, des comparaisons avec d'autres solveurs sur des écoulements subsoniques et transsoniques autour de profils, la validation des fonctions formant le solveur RANS, ainsi que l'application à la simulation ILES turbulentes de l'écoulement sur des collines périodiques montre l'étendue des applications qui peuvent être traitées avec différentes méthodes numériques en utilisant le code DynHoLab. En outre, tout au long des différentes validations proposées dans la thèse, nous avons montré que les schémas compacts basé sur le résidu présentent de meilleures propriétés en termes de résolvabilité que les schémas non compacts du même ordre de précision. Sur le développement de méthodes d'ordre élevé, des points spécifiques examinés ci-après dans les perspectives exigerait de nouvelles améliorations.

La stratégie spécifique considéré dans cette thèse pour l'extension des schémas RBC à des géométries complexes est leur utilisation en conjonction avec une stratégie de grilles structurées recouvrantes. Ceci permet d'optimiser les zones du domaine de calcul où les schémas sont appliqués avec leur ordre de précision nominal. En fonction du support de discrétisation et d'interpolation retenu dans la stratégie numérique, les algorithmes d'assemblage de grilles existants n'étaient pas en mesure de traiter le cas spécifique des schémas RBC d'ordre élevé. Pour cette raison, un nouveau générateur de maillage a été développé comme un plug-in de DynHoLab. Les algorithmes nécessaires pour le calcul des connectivités entre les différentes grilles qui se chevauchent ont été retravaillées afin d'améliorer leur efficacité de calcul.

Perspectives

En ce qui concerne le code de DynHoLab, la meilleure validation serait sa longévité avec une communauté de développement ne se limitant pas au laboratoire DynFluid. Plus précisément sur le code, de nouveaux développements devraient être menés sur la parallélisation afin d'être en mesure d'équilibrer la charge sur des pavages non structurées des grilles structurées, ou même sur des maillages incluant des grilles recouvrantes. Il serait également judicieux de remplacer le package existant responsable de la gestion du la structure d'arbre python/CGNS (DynCGNS) par celui développé pour le consortium CGNS par Poinot [139]. Ce remplacement permettrait des interactions directes avec des outils tels que Cassiopée [2]. Ces assemblages de code serait facile avec tout code partageant cette structure de donnée Python/CGNS et cela ouvre la voie à des résolutions couplés que les interactions fluide/structure, ou permettre de l'adaptation de maillage des éléments rotatifs dans les maillages recouvrants, ... A propos de grilles recouvrantes, la mise en œuvre 3-D existante dans DynHoLab devrait être démontrée. Le solveur volume-finis actuel pourrait être simplifié pour dégénérer sur le solveur de différences-finis afin de minimiser la consommation CPU et de mémoire dans les grilles Cartésiennes. De plus, le solveur pourrait être optimisé, en passant par la réécriture des boucles sur les tableaux, pour ne pas calculer les degrés de liberté neutralisés. Une autre stratégie puissante pouvant être testée à l'avenir est l'utilisation de maillages hybrides ayant des éléments à la fois structurées et non-structurées. Cela permettrait d'obtenir une conservativité intrinsèque dans les raccordements de maillages et la stratégie globale pourrait alors être utilisé pour les écoulements internes tels que les turbomachines. En ce qui concerne l'état actuel du code, des développements sont réalisés sur le module de physique afin de généraliser le code à toute équation d'état. Les autres contributions concernent des schémas numériques ou des pré- et post-traitement sont ajoutés comme plug-in et donc non intrusive dans le noyau de code.

Sur les schémas d'ordre élevé, d'autres améliorations sont attendues grâce à des recherches supplémentaires sur le traitement des géométries complexes, des conditions aux limites et les algorithmes implicites. Dans le cas spécifique des schémas RBC, afin d'améliorer la précision et l'efficacité des calculs instationnaires rapides, un choix raisonnable serait un schéma en temps de type Runge-Kutta. Néanmoins, le couplage direct des schémas RBC avec les schémas de Rung-Kutta (RK) conduirait à l'inversion de matrices à large bande à chaque pas, en raison de la présence de la dérivée temporelle dans l'opérateur de dissipation du schéma. Une bonne option pourrait être l'écriture d'une dissipation au pas de RK précédent, tout en modifiant les coefficients du schéma RK pour assurer le bon ordre de précision en temps. Ce type d'approche a été étudiée par Abgrall et Ricchiuto [16] pour les schémas Residual Distribution. Grâce à ces avancées, on pourrait envisager d'utiliser le code pour Les simulations sur quelques milliers de processeurs.

Appendix A

Discrete operators

Let us consider a discretization f_i on a uniform mesh $x_i = i\,\delta x = i\,h$ of an unknown field f in $L^p(\mathbb{R})$ with values in $\mathbb{R}$. We introduce some basic operators for the expression of derivative approximations of f. The succesive derivative of f are written $f^{(n)}$. We will denote by δ and μ the basic difference and average linear operators of characteristic distance $h = \delta x$ at the generic point i.

$$\begin{aligned} (\delta f)_{i+\frac{1}{2}} &= f_{i+1} - f_i \\ (\mu f)_{i+\frac{1}{2}} &= \frac{1}{2}\,(f_{i+1} + f_i) \end{aligned} \tag{A.1}$$

Denoting by the exponent $\delta^2 = \delta \circ \delta = \delta(\delta\,.)$ the combination of the operators, we first observe the behavior of the operators when forming some combinations of them. On the difference operator, we note that an even number of δ produces a symmetric operator corresponding to some approximation of an even derivative.

$$\delta^2 f_i = f_{i+1} - 2\,f_i + f_{i-1}$$
$$\delta^4 f_i = f_{i+2} - 4\,f_{i+1} + 6\,f_i - 4\,f_{i-1} + f_{i-2}$$
$$\delta^6 f_i = f_{i+3} - 6\,f_{i+2} + 15\,f_{i+1} - 20\,f_i + 15\,f_{i-1} - 6\,f_{i-2} + f_{i-3}$$

Using the average operator, it is permitted for second-order accurate filter to be designed.

$$\mu^2 f_i = \frac{1}{4}\,(f_{i+1} + 2\,f_i + f_{i-1})$$
$$\mu^4 f_i = \frac{1}{16}\,(f_{i+2} + 4\,f_{i+1} + 6\,f_i + 4\,f_{i-1} + f_{i-2})$$
$$\mu^6 f_i = \frac{1}{64}\,(f_{i+3} + 6\,f_{i+2} + 15\,f_{i+1} + 20\,f_i + 15\,f_{i-1} + 6\,f_{i-2} + f_{i-3})$$

And by forming an odd combination of difference operators, we produce an antisymmetric operator corresponding to the approximation of an odd derivative.

$$\delta\mu f_i = \frac{1}{2}(f_{i+1} - f_{i-1})$$
$$\delta^3\mu f_i = \frac{1}{2}(f_{i+2} - 2f_{i+1} + 2f_{i-1} - f_{i-2})$$
$$\delta^5\mu f_i = \frac{1}{2}(f_{i+3} - 4f_{i+2} + 5f_{i+1} - 5f_{i-1} + 4f_{i-2} - f_{i-3})$$
$$\delta^7\mu f_i = \frac{1}{2}(f_{i+4} - 6f_{i+3} + 14f_{i+2} - 14f_{i+1} + 14f_{i-1} - 14f_{i-2} + 6f_{i-3} - f_{i-4})$$

We observe that the coefficients of δ^n and μ^n operators are computed with Pascal's triangle as

$$\delta^n f_i = \sum_{k=0}^{n} (-1)^k \binom{n}{k} f_{i+k-\frac{n}{2}}\,, \qquad \mu^n f_i = \sum_{k=0}^{n} \frac{\binom{n}{k}}{\sum_l \binom{n}{l}} f_{i+k-\frac{n}{2}} \tag{A.2}$$

with $\binom{n}{k} = \frac{n!}{k!(n-k)!}$, giving the general formula

$$\mu^m \delta^n f_i = \sum_{l=0}^{m} \frac{\binom{m}{l}}{\sum_p \binom{n}{p}} \sum_{k=0}^{n} (-1)^k \binom{n}{k} f_{i+k+l-\frac{n}{2}-\frac{m}{2}} \tag{A.3}$$

For the design of high-order approximations of derivatives, it is worth noting that an even and odd number of difference operator δ – associate to an average operator to back up to discretization points – produces respectiviely a symmetric and antisymmetric operator corresponding to an approximation of respectively an even and odd derivative. The general formula writes

$$f_i^{(2n)} = \frac{1}{h^{2n}}\, \delta^{2n} f_i + O(h^2) \tag{A.4}$$

$$f_i^{(2n+1)} = \frac{1}{h^{2n+1}}\, \mu\delta^{2n+1} f_i + O(h^2) \tag{A.5}$$

A.1 Non-compact approximations

Combining the Taylor series of f at points $(i+1)$ and $(i-1)$, one could easily obtain a centered approximation of the first derivative on three points with a truncation error of order two (see formula (A.6) with coefficients provided in table A.1). Inspired from papers on RBC schemes [112], let us adopt the general notation

$$\forall n \geq 0, \quad f_i^{(1)} = \frac{1}{h} \sum_{j=0}^{n} a_{2j} \delta^{2j+1} \mu f_i \tag{A.6}$$

with the convention for the identity matrix $\delta^0 = I$ and $a_0 = 1$. By correcting the leading error term with equation (A.5), it is then easy to obtain an approximation of the first derivative on five points with a truncation error of order four and so on for higher approximations. Since the approximation is centered, it is zero dissipative and with coefficients provided in table A.1, the approximation (A.6) verify $f_i^{(1)} = O(h^{2n+2})$. As the reader may not be familiar with the above notations, we provide the same non-compact approximations of the first derivative using the generic notation

$$\forall n \geq 0, \quad f_i^{(1)} = \sum_{j=1}^{n+1} \alpha_j \frac{f_{i+j} - f_{i-j}}{2jh} \tag{A.7}$$

which is equivalent to the formula (A.6) with the coefficients given in table A.1. The order constraints are derived at the next section in a more general framework including compact schemes. This usually leads to N^{th}-order approximation on $N+1$ points. To get a closed system, this correspond to $n = (N-2)/2$. The second-order accurate three-point formula is used in the centered part of the Jameson scheme [97] and rewriting the Roe scheme [151] as a sum of a centered part and a dissipation, we find again this three-point formula for the centered part. Higher-order accurate schemes can be found in section 1.2.3 under the denomination FE-MUSCL schemes.

A.2 Compact approximations

The notion of rational approximations of function was introduced by Frobenius [77] but named after Henri Padé [132]. The Padé approximant give less dispersion than the corresponding approximation obtained by truncation of the Taylor series expansion (see the non-compact approximations of the previous section). Another advantage is the narrower stencil of the Padé approximant compared to the same-order non-compact formula. However, this operator requires the inversion of a system if it is used directly. One of the trick of RBC schemes [112] is to avoid this inversion by multiplying the global scheme by the denominators (see section 1.2.4). As previously, we introduce a notation using the mean and average operators (A.1) for the Padé approximation of the first derivative

$$\forall m \geq 0, \forall n \geq 0, \quad \sum_{j=0}^{m} b_{2j} \delta^{2j} f_i^{(1)} = \frac{1}{h} \sum_{j=0}^{n} a_{2j} \delta^{2j+1} \mu f_i \tag{A.8}$$

with $b_0 = a_0 = 1$ and $\delta^0 = I$ the identity and the corresponding operator with the notation used by Lele [111]

$$\forall m \geq 0, \forall n \geq 0, \quad \sum_{j=0}^{m} \beta_j (f_{i+j}^{(1)} + f_{i-j}^{(1)}) = \sum_{j=1}^{n+1} \alpha_j \frac{f_{i+j} - f_{i-j}}{2jh} \tag{A.9}$$

with $\beta_0 = 1/2$ for the uniqueness of coefficients. The constraints on the coefficients are derived by matching the Taylor series coefficients. To get a closed system for order N, one should verify $N = 2(n + m) + 2$. The truncation error is formally obtained from the first unmatched coefficients. Since the approximation is centered, the constraints are obtained for even order of accuracy. Applying this on equation (A.8) give the constraints

$$\begin{cases} b_2 - a_2 = \dfrac{1}{6} & (4^{th} \text{ order}) \\ \dfrac{1}{6} b_2 - b_4 + a_4 = \dfrac{1}{30} & (6^{th} \text{ order}) \\ \dfrac{1}{30} b_2 - \dfrac{1}{6} b_4 + b_6 - a_6 = \dfrac{1}{140} & (8^{th} \text{ order}) \end{cases} \tag{A.10}$$

and the constraints obtained for equation (A.9) are

$$\begin{cases} 1 + 2 \displaystyle\sum_{j=1}^{m} \beta_j = \sum_{j=1}^{n+1} \alpha_j & (2^{nd} \text{ order}) \\ 2\,(N-1) \displaystyle\sum_{j=1}^{m} j^{N-2} \beta_j = \sum_{j=1}^{n+1} j^{N-2} \alpha_j & (\text{even } N^{th} \text{ order}, N \geq 4) \end{cases} \tag{A.11}$$

The coefficients for formula (A.8) and (A.9) are provided in table A.2 for fourth-, sixth- and eighth-order approximations of the first derivative, with respectively one, two and three remarkable Padé fractions going from tridiagonal to heptadiagonal schemes.

We would like to check that the two notations are equivalent. At first glance, the Lele notation contains one more parameter than the RBC notation. It can also be seen that the constraint relations start at second order of accuracy in the Lele's notations and at fourth order of accuracy in the RBC notations. In fact, the RBC notation is already second order accurate with all its parameters being zero. The constraint relation of second order in the Lele notation should be taken into account to search a relation between the two system of notations. Starting from formula (A.8), one can replace the difference and average operators by their expression

(A.2). Then, switching the sums and using some relations on binomial coefficients provide the relation between the two system of notation by identification

$$\alpha_l = \frac{1}{\mathcal{A}}\left[1 - \sum_{j=l}^{n} a_{2j}(-1)^{j+l}\left[\binom{2j+1}{j+l} - \binom{2j+1}{j+l+1}\right]\right] \quad , \quad \beta_l = \frac{1}{\mathcal{A}}\sum_{j=l}^{m} b_{2j}(-1)^{j+l}\binom{2j}{j+l} \tag{A.12}$$

with $\mathcal{A} = \sum_{j=0}^{m} b_{2j}(-1)^j \binom{2j}{j} \neq 0$.

These Padé fractions can be found for instance in the work of Lele [111] with its well-known spectral-like scheme or in [149] for LES simulations. The coefficients of the compact approximations were optimized by Kim [102].

A.2.1 Fourier analysis of errors

Formal truncation error (order of accuracy and constant) of both non-compact and compact schemes are given in tables A.1 and A.2. The Fourier analysis of errors of differencing schemes is a classical technique. It provides an effective way to quantify the resolution characteristics of the approximations. The function f is supposed to be periodic over the domain $[0, L]$ discretized in N subdomains $h = L/N$. We introduce the decomposition of the function f into its Fourier coefficients

$$f(x) = \sum_{k=-N/2}^{N/2} \hat{f}_k \exp\left(\frac{2\pi i k x}{L}\right) \tag{A.13}$$

with $i^2 = -1$. The scaled wave number $k\delta x$ is defined on the domain $[0, \pi]$. The Fourier symbol of formula (A.9) defines an effective wave number $\tilde{k}$ of the discretization

$$\tilde{k}\delta x = \frac{\sum_{j=1}^{n+1}(\alpha_j/j)\sin(jk\delta x)}{1 + \sum_{j=1}^{m} 2\beta_j \cos(2jk\delta x)} \tag{A.14}$$

which allow the comparison of the dispersive errors compared to the exact differencing scheme and is plotted for various non-compact and compact approximations in figure A.1. On the left plot of this figure, we observe that the Padé fractions provide a better resolution that the non-conpact approximations of the same order of accuracy. Using the right plot, we can determine the resolvability of the schemes stating a level of error. For instance, requirering a dispersive error of 10^{-5}, only the three Padé fractions of order 8 can resolve a wave with less than $\lambda = 2\pi/k = 8\delta x$, meaning eight points per wave length, which correspond to $\pi/4 = k\delta x$. We remind that the finite differences present an indetermination at $k\delta x = \pi$ and are intrinsically unable to damp the grid to grid oscillations (or $\lambda = 2\delta x$ waves). A dissipation or filter should be added to these centered discretizations.

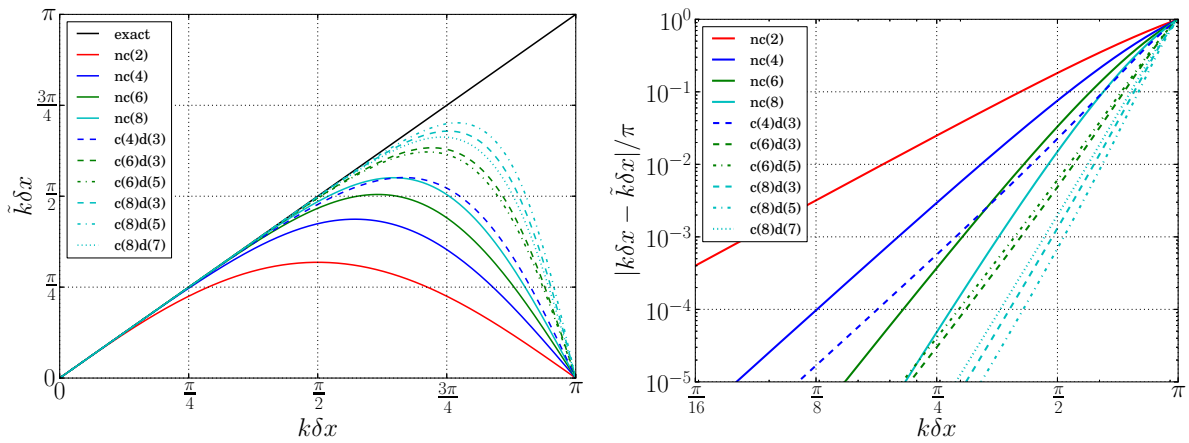


Figure A.1: Caracterisation of first derivative discretisations in the wave space. (nc(i): non-compact schemes of order i, c(i)d(j) compact schemes of order i with j diagonal)

Table A.1: Coefficients of the non-compact approximations of the first derivative for various order of accuracy for both formula (A.6) and (A.7) with leading error coefficient (lec)

Order	a_2	a_4	a_6	α_1	α_2	α_3	α_4	lec
2				1				1/6
4	-1/6			4/3	-1/3			-1/30
6	-1/6	1/30		3/2	-3/5	1/10		1/140
8	-1/6	1/30	-1/140	8/5	-4/5	8/35	-1/35	-1/630

Table A.2: Coefficients of the compact approximations of the first derivative for various order of accuracy for both formula (A.8) and (A.9) with leading error coefficient (lec)

Order	Diag	b_2	b_4	b_6	a_2	a_4	β_1	β_2	β_3	α_1	α_2	α_3	lec
4	3	1/6					1/4			3/2			-1/180
6	3	1/5			1/30		1/3			14/9	1/9		1/2100
6	5	1/6	-1/180				17/57	-1/114		30/19			1/1512
8	3	3/14			1/21	-1/420	3/8			25/16	1/5	-1/80	-1/17 640
8	5	2/7	1/70		5/42		4/9	1/36		40/27	25/54		-1/44 100
8	7	1/6	-1/180	1/1512			1503/4688	-9/586	5/4688	945/586			-23/226 800

Appendix B

Treatment of matching joins

B.1 Treatment of domain connectivity

We consider three types of joins: the matching join for which points of facing grids are coincident, the non-matching join (or non-conformal join) for which only the boundary shape coincide but not grid points, the overlapping joins for which an overlap region exists between facing grids.

Matching joins

Matching joins present the easiest treatment. Considering two 3-D grids $G^{(1)}$ and $G^{(2)}$, we define a join with two communication windows

$$((i_1^{(1)}, j_1^{(1)}, k_1^{(1)}), (i_2^{(1)}, j_2^{(1)}, k_2^{(1)})) \quad \text{and} \quad ((i_1^{(2)}, j_1^{(2)}, k_1^{(2)}), (i_2^{(2)}, j_2^{(2)}, k_2^{(2)}))$$

and a matrix of transformation of the axes defined as follows: each element shows the image of the adjacent zone's face of a positive index increment in the current zone's face. The first element is the image of a positive increment in i; the second element is the image of a positive increment in j; the third element (in 3-D) is the image of a positive increment in k on the current zone's face. For 3-D, the transformation matrix T is constructed from the notation $T = (\pm a, \pm b, \pm c)$ as follows:

$$T = \begin{pmatrix} sgn(a)del(a-1) & sgn(b)del(b-1) & sgn(c)del(c-1) \\ sgn(a)del(a-2) & sgn(b)del(b-2) & sgn(c)del(c-2) \\ sgn(a)del(a-3) & sgn(b)del(b-3) & sgn(c)del(c-3) \end{pmatrix}$$

with $sgn(x) = +1$ if $x \geq 0$, and -1 if $x < 0$, and $del(x-y) = +1$ if $|x| = |y|$, and 0 otherwise. For example, $T = (-2, +3, +1)$ gives the transformation matrix

$$T = \begin{pmatrix} 0 & 0 & +1 \\ -1 & 0 & 0 \\ 0 & +1 & 0 \end{pmatrix}$$

Then, using ghost points, the principle is easy and can be seen on figure (B.1)

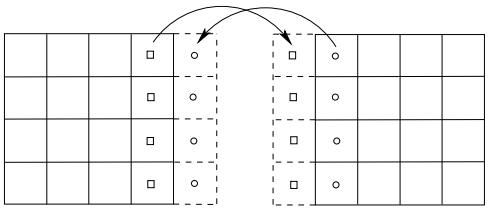


Figure B.1: Example of a 2-D match join: use of dashed ghost-cells for the communication between grids.

Bibliography

[1] Advisory council for aviation research and innovation in europe (acare), http://www.acare4europe.com.

[2] Cassiopée, url: http://elsa.onera.fr/cassiopee/userguide.html.

[3] The dlr tau website, url: http://tau.dlr.de/startseite/.

[4] European commission, "acare strategic reaserch agenda", 2004.

[5] European committion, "acare flightpath 2050. europe's vision for aviation", 2011.

[6] Fineturbo website, url: http://www.numeca.com/en/products/finetmturbo.

[7] Fortran to python wrapper. `http://cens.ioc.ee/projects/f2py2e/`.

[8] http://elsa.onera.fr/.

[9] http://www.python.org/.

[10] Idihom, url: http://www.idihom.de.

[11] O pironneau, f hecht, al hyaric, k ohtsuka - url: http://www. freefem. org, 2006.

[12] Openfoam project web pages, http://www.openfoam.org.

[13] 1st international workshop on high-order cfd methods, at the 50th aiaa aerospace sciences meeting, nashville, tennessee, January 7-8, 2012.

[14] 2nd international workshop on high-order cfd methods, May 27 - 28, 2013 at NH Hotel Köln City.

[15] S. Abarbanel and A. Kumar. Compact high-order schemes for the euler equations. *Journal of Scientific Computing*, 3:275, 1988.

[16] Rémi Abgrall, D. De Santis, and Mario Ricchiuto. High Order Residual Distribution Scheme for the RANS Equations. In *ICCFD7-2802*, pages 1–25, 2012.

[17] Rémi Abgrall. Residual distribution schemes: Current status and future trends. *Computers and Fluids*, 35:641–669, 2006.

[18] Adams, Brainerd, Hendrickson, Maine, Martin, and Smith. *The Fortran 2003 Handbook*. Springer, 2009.

[19] R.A. Adams. *Sobolev spaces*. Academic Press, 1978.

[20] H.-B. An, J. Wen, and T. Feng. On finite difference approximation of a matrix-vector product in the jacobian-free newton-krylov method. *Journal of Computational and Applied Mathematics*, 236:1399–1409, 2011.

[21] Frédéric Archambeau, Namane Méchitoua, and Marc Sakiz. Code_saturne: a finite volume code for the computation of turbulent incompressible flows - industrial applications. *International Journal on Finite Volumes*, 1, 2004.

[22] B. Aupoix. Introduction to turbulence modelling. from mixing length to reynolds stress models. *VKI Lecture Series*, 22-26 March 2004.

[23] A.H. Baker, E.R. Jessup, and T. Manteuffel. *SIAM J. Matrix Anal. Appl. 26, 962*, 2005.

[24] B. Baldwin and T. Lomax. Thin layer approximation and algebraic model for separated turbulent flows. *AIAA Paper*, AIAA-78-257, 1978.

[25] T.J. Barth and P.O. Frederichson. High-order solution of the euler equation on unstructured grids using quadratic reconstruction. *AIAA 90-0013*, 1990.

[26] F. Bassi and S. Rebay. *A high-order discontinuous Galerkin method for compressible turbulent flows*. Springer, 1999.

[27] G.K. Batchelor. *The theory of Homogeneous Turbulence*. Cambridge University Press, Cambridge, 1953.

[28] J. A. Benek, P. G. Buning, and J. L. Steger. A 3-d chimera grid embedding technique. In *AIAA 7th Computational Fluid Dynamics Conference*, number AIAA-85-1523, 1985.

[29] J. A. Benek, P. G. Buning, and J. L. Steger. A 3-D Chimera Grid Embedding Technique. In *AIAA 7th Computational Fluid Dynamics Conference*, pages AIAA–85–1523, 1985.

[30] J. A. Benek, J. L. Steger, and F. C. Dougherty. A flexible grid embedding technique with application to the Euler equations. In *AIAA 6th Computational Fluid Dynamics Conference*, pages AIAA–83–1944, 1983.

[31] J. A. Benek, J. L. Steger, and F. C. Dougherty. A flexible grid embedding technique with application to the euler equations. In *AIAA 6th Computational Fluid Dynamics Conference*, number AIAA-83-1944, 1983.

[32] J. A. Benek, J. L. Steger, F. C. Dougherty, and P. G. Buning. Chimera: A Grid-Embedding Technique. Technical report, NASA, 1986.

[33] C. Benoit and S. Peron. Automatic structured mesh generation around two-dimensional bodies defined by polylines or polyc1 curves. *Computers and Fluids*, 61:64–76, 2012.

[34] J. L. Bentley. Multidimensional binary search trees in database applications. *IEEE Transactions on Sofware Engineering*, VOL. SE-5, N. 4:333–340, July 1979.

[35] Marsha J. Berger. On conservation at grid interfaces. *Society for Industrial and Applied Mathematics*, 24, No. 5:967–983, 1987.

[36] Julien Berland, Christophe Bogey, and Christophe Bailly. Low-dissipation and low-dispersion fourth-order Runge–Kutta algorithm. *Computers & Fluids*, 35(10):1459–1463, December 2006.

[37] Christophe Bogey and Christophe Bailly. A family of low dispersive and low dissipative explicit schemes for flow and noise computations. *Journal of Computational Physics*, 194(1):194–214, February 2004.

[38] J.-P. Boussinesq. *Essai sur la théorie des eaux courantes*. Mémoire des savants étrangers, Ac. Sc. Paris, xxviii édition, 1877.

[39] J.-P. Boussinesq. *Théorie de l'écoulement tourbillonnant et tumultueux des liquides dans des lits rectilignes á grande section*. Gauthier-Villars, Paris, tome i et ii, 1897.

[40] M. Breuer, N. Peller, C. Rapp, and M. Manhart. Flow over periodic hills, numerical and experimental study over a wide range of reynolds numbers. *Computers and Fluids*, 38:433–457, 2009.

[41] H. Brezis. *Analyse Fonctionelle: Théorie et Applications*. Masson, Paris, 1984.

[42] T Cebeci and A. Smith. Analysis of turbulent boundary layers. *Applied Mathematics and Mechanics*, 15, 1974.

[43] P. Chassaing. *Turbulence en mécaniques des fluides*. Cépaduès-éditions, Toulouse, 2000.

[44] G. Chesshire and W. D. Henshaw. Composite overlapping meshes for the solution of partial differential equations. *Journal of Computational Physics*, 90:1–64, 1990.

[45] G. Chesshire and W. D. Henshaw. A scheme for conservative interpolation on overlapping grids. *Society for Industrial and Applied Mathematics*, 14, No. 4:819–845, 1994.

[46] J. Chicheportiche and X. Gloerfelt. Study of interpolation methods for high-accuracy computations on overlapping grids. *Computers and Fluids*, 68:112–133, 2012.

[47] Jérèmie Chicheportiche. *Calcul direct du rayonnement acoustique généré par une cavité cylindrique sous une aile d'avion*. PhD thesis, ENSAM Paris, 2011.

[48] Jérèmie Chicheportiche and Xavier Gloerfelt. Study of interpolation methods for high-accuracy computations on overlapping grids. *Computers & Fluids*, 68(2012):112–133, September 2012.

[49] P. Cinnella and A. Lerat. A study of turbulent compressible flows over oscillating airfoils by a high-order accurate numerical scheme. *Computational Fluid Dynamics Journal*, Special issue 9:257–271, 2001.

[50] Paola Cinnella. *Simulation d'écoulements compressibles turbulents autour de profils oscillants par une méthode numérique de haute précision*. PhD thesis, Arts et Métiers ParisTech, June 1999.

[51] D. K. Clarke, M. D. Salas, and Hassan H. A. *AIAA Journal*, 24-353, 1986.

[52] C. Cockburn, G. Karniadakis, and C.-W. Shu. *The development of discontinuous Galerkin methods*. Discontinuous Galerkin Methods, volume 11, 3-50. Springer, 1999.

[53] C. Content, P.-Y. Outtier, and P. Cinnella. Coupled/uncoupled solutions of rans equations using a jacobian-free newton-krylov method. In *21st AIAA Computational Fluid Dynamics Conference*, 24-27 June 2013, San Diego, California.

[54] C. Corre. *Contribution à la simulation et à l'analyse des écoulements compressibles*. PhD thesis, Habilitation à Diriger des Recherches, ENSAM Paris, 2004.

[55] C. Corre and X. Du. A residual-based scheme for computing compressible flows on unstructured grids. *Computers and Fluids*, 38:1338–1347, 2009.

[56] C. Corre, F. Falissard, and A. Lerat. High-order residual-based compact schemes for compressible inviscid flows. *Computers and fluids*, 36, pages 1567-1582, 2007.

[57] C. Corre, G. Hans, and A. Lerat. A residual-based compact schemes for the unsteady compressible navier-stokes equations. *Computers and fluids*, 34, pages 561-580, 2005.

[58] C. Corre and A. Lerat. High-order residual-based compact schemes for advection-diffusion problems. *Computers and fluids*, 37, pages 505-519, 2008.

[59] R. Courant, K. Friedrichs, and H. Lewy. Über die partiellen Differenzengleichungen der mathematischen Physik. *Mathematische Annalen*, 100:32–74, 1928.

[60] R. Courant, K. Friedrichs, and H. Lewy. über die partiellen differenzengleichungen der mathematischen physik. *Math. Ann.*, 100:32–74, 1928.

[61] P De Palma, G. Pascazio, G. Rossiello, and M. Napolitano. Implicit third-order accurate residual distribution schemes for unsteady hyperbolic problems. *VKI LS 2006-01, CFD high-order discretization methods*, 14-18 November 2005.

[62] J. W. Delfs. An overlapping grid technique for high resolution caa schemes for complex geometries. *AIAA Paper 2001-2199*, 2001.

[63] R. *et al.* Dembo. Inexact newton methods. *SIAM J. Num. Anal.*, 19:400–408, 1982.

[64] G Desquesnes, M. Terracol, E Manoha, and P Sagaut. On the use of high-order overlapping grid method for coupling cfd/caa. *Journal of Computational Physics*, 220:355–382, 2006.

[65] D. Desvigne, O. Marsden, C. Bogey, and C Bailly. Development of noncentered wavenumber-based optimized interpolation schemes with amplification control for overlapping grids. *SIAM J. SCI. COMPUT.*, 32:2074–2098.

[66] D. Drikakis. Advances in turbulent flow computations using high-resolution methods. *Progress in Aerospace Sciences*, 39:405–424, 2003.

[67] D. Drikakis, F. Grinstein, and D. Youngs. On the computation of instabilities and symmetry-breaking in fluid mechanics. *Progress in Aerospace Sciences*, 41:609–641, 2005.

[68] Eisenstat and Walker. Choosing the forcing terms in an inexact newton method. *SIAM J.of Sc.Comp.*, 1996.

[69] S.C. Eisenstat and H.F. Walker. Choosing the forcing terms in a inexact newton method. *SIAM J. Sci. Stat. Comput.*, 17:16–32, 1996.

[70] T. Emmert, P Lafon, and C. Bailly. Computation of aeroacoustic phenomena in subsonic and transonic ducted flow. In *13th AIAA/CEAS AeroAcoustics Conference*, volume AIAA 2007-3429, Rome, Italy, 21-23 May 2007.

[71] A. Favre. Equations des gaz turbulents compressibles. i. forme générales. *J. Méc.*, 4:361–390, 1965.

[72] A. Favre. Equations des gaz turbulents compressibles. ii. méthodes des vitesses moyennes ; méthodes des vitesses moyennes pondérées par la masse volumique. *J. Méc.*, 4:391–421, 1965.

[73] Alexandre Favre. Turbulence: Space-time statistical properties and behavior in supersonic flows. *Physics of Fluids (1958-1988)*, 26(10):2851–2863, 1983.

[74] R.A. Finkel and J.L. Bentley. Quad trees a data structure for retrieval on composite keys. *Acta Informatica*, 4(1):1–9, 1974.

[75] A. Fosso, H. Deniau, F. Sicot, and P. Sagaut. Curvilinear finite-volume schemes using high-order compact interpolation. *J. Comput. Phys.*, 229, 5090-5122, 2010.

[76] P. Frey and P.L. George. *Mesh Generation.* ISTE. Wiley, 2010.

[77] G. Frobenius. Ueber relationen zwischem den näherungsbrüchen von potenzreihen. *Journal für die reine und angewandte Mathematik (Crelle's Journal)*, Issue 90:1–17, 1881.

[78] D. V. Gaitonde and J.S. Shang. Optimized compact-difference-based finite volume schemes for linear wave phenomena. *Journal of Computational Physics*, 138:617–643, 1997.

[79] E. Garnier, N. Adams, and P. Sagaut. *Large-eddy simulation for compressible flows.* Spinger, 2009.

[80] P.L. George and H. Borouchaki. *Delaunay Triangulation and Meshing: Application to Finite Elements.* Butterworth-Heinemann, 1998.

[81] S.K. Godunov. A finite difference method for the numerical computation of discontinuous solutions of the equations of fluid dynamics. *Mat Sbornik*, 47:271–90, 1959.

[82] Michael Goldberg. Three infinite families of tetrahedral space-fillers. *Journal of Combinatorial Theory, Series A*, 16(3):348 – 354, 1974.

[83] K. Grimich. *High-order residual based compact schemes for unsteady Compressible flows. Application to scale resolving simulations.* PhD thesis, Arts et Métiers ParisTech, Paris, 2013.

[84] K. Grimich, P. Cinnella, and A. Lerat. Spectral properties of high-order residual-based compact schemes for unsteady compressible flows. *Journal of Computational Physics*, 252(0):142 – 162, 2013.

[85] K. Grimich, B. Michel, P. Cinnella, and A. Lerat. An accurate finite-volume formulation of a Residual-Based Compact scheme for unsteady compressible flows. *Computers & Fluids*, 92:93–112, 2014.

[86] F. Grinstein, L. G. Margolin, and W. J. Rider. *Implicit large-eddy simulation: computing turbulent fluid dynamics*. Cambridge University Press, 2007.

[87] F. Haider, P. Brenner, B. Courbet, and J.-P. Croisille. Efficient implementation of high order reconstruction in finite volume methods. In *Finite Volumes for Complex Applications VI Problems & Perspectives*, volume 4, pages 553–560, 2011. Springer Proceedings in Mathematics.

[88] Grégoire Hans. *Schémas numériques compacts basés sur le résidu en maillage irrégulier pour les équations de Navier-Stokes en compressible. PhD thesis, Arts et Métiers ParisTech*. PhD thesis, 2002.

[89] G. Hanss. *Schémas numériques compacts basés sur le résidu en maillage irrégulier pour les équations de Navier-Stokes en compressible*. PhD thesis, Arts et Métiers ParisTech, Paris, 2002.

[90] B. Henshaw, K. Chand, and D. Quinlan. Overture.

[91] Samuel J. Hercus and Paola Cinnella. Robust Shape Optimization of Uncertain Dense Gas Flows Through a Plane Turbine Cascade. In *ASME-JSME-KSME 2011 Joint Fluids Engineering Conference, AJK2011-FED*, pages 1739–1749. Asme, 2011.

[92] J.O. Hinze. *Turbulence*. McGraw-hill, New York, 1953.

[93] Charles Hirsch. *Numerical computation of internal and external flows*, volume 1: Fundementals of Numerical Discretisation. Wiley Interscience, 1988.

[94] F.Q. Hu, M.Y. Hussaini, and J.L. Manthey. Low-dissipation and low-dispersion runge–kutta schemes for computational acoustics. *Journal of Computational Physics*, 124(1):177 – 191, 1996.

[95] Y. Huang and A. Lerat. Second-order upwinding through a characteristic time-step matrix for compressible flow calculations. *Journal of Computational Physics*, 142:445–472, 1998.

[96] John D. Hunter. Matplotlib: A 2d graphics environment. *Computing In Science & Engineering*, 9(3):90–95, May-Jun 2007.

[97] A. Jameson, W. Schmidt, and E. Turkel. Numerical solutions of the euler equations by finite volume methods using runge-kutta time stepping. *AIAA paper*, 81-1259, 1981.

[98] H. Jasak, A. Jemcov, and Z. Tukovic. Openfoam: A c++ library for complex physics simulations. In *International Workshop on Coupled Methods in Numerical Dynamics*, IUC, Dubrovnik, Croatia, September 19^{th}-21^{th} 2007.

[99] D.A. Johnson and D. Bachelo. Transonic flow past a symmetrical airfoil - inviscid and turbulent flow properties. *AIAA Journal*, 18(1):16–24, 1979.

[100] Eric Jones, Travis Oliphant, Pearu Peterson, et al. SciPy: Open source scientific tools for Python, 2001–.

[101] J. W. Kim and D. J. Lee. Adaptive Nonlinear Artificial Dissipation Model for Computational Aeroacoustics. *AIAA Journal*, 39:810–818, May 2001.

[102] Jae W. Kim and Duck J. Lee. Optimized compact finite difference schemes with maximum resolution. *AIAA Journal*, 34(5):887–893, May 1996.

[103] Bil Kleb. FUN3D fully unstructered Navier-Stokes, 2008–. `http://fun3d.larc.nasa.gov/`.

[104] D. A. Knoll and D. E. Keyes. Jacobian-free newton-krylov methods: a survey of approches and applications. *Journal of Computational Physics*, 193:357–397, 2004.

[105] M.H. Kobayashi. On a class of padé finite volume methods. *Journal of Computational Physics*, 156:137–180, 1999.

[106] N. Kroll, H. Bieler, H. Decononck, V. Couallier, H. Van der Ven, and K. Sorensen. *ADIGMA – A European Initiative on the Development of Adaptive Higher-Order Variational Methods for Aerospace Applications*. Notes on Numerical Fluid Mechanics and Multidisciplinary Design. Springer, 2010.

[107] C. Lacor, S. Smirnov, and M. Baelmans. A finite volume formulation of compact central schemes on arbitrary structured meshes. *Journal of Computational Physics*, 198:535–566, 2004.

[108] Hans Petter Langtangen. *Python Scripting for Computational Science*. Springer, third edition, 2009.

[109] A. Lani, T. Quintino, D. Kimpe, H. Deconinck, S. Vandewalle, and S. Poedts. The coolfluid framework: Design solutions for high performance object oriented scientific computing software. In *Lecture Notes in Computer Science*, volume 3514, pages 279–286, 2005. Cited By (since 1996): 6.

[110] S. K. Lele. Compressibility effects on turbulence. *Annual Review Fluid Mechanics*, 26:211–254, 1994.

[111] Sanjiva K. Lele. Compact finite difference schemes with spectral-like resolution. *Journal of Computational Physics*, 103(1):16–42, November 1992.

[112] A. Lerat and C. Corre. A residual-based compact scheme for the compressible navier-stokes equations. *Journal of Computational Physics*, 170, pages 642-675, 2001.

[113] A. Lerat and C. Corre. *Approximations d'ordre élevé pour les écoulements compressibles*. Ecole de Printemps de mécanique des fluides, Fréjus, France, 2003.

[114] A. Lerat, C. Corre, and G. Hans. Efficient high-order schemes on non-uniform meshes for multi-d compressible flows. In *Frontiers of Computational Fluid Dynamics*, 2002.

[115] A. Lerat, K. Grimich, and P. Cinnella. On the design of high order residual-based dissipation for unsteady compressible flows. *Journal of Computational Physics*, 235:32–51, 2013.

[116] A. Lerat and A. Rezgui. Schémas dissipatifs précis à l'ordre trois pour les systèmes hyperboliques = dissipative third-order accurate schemes for hyperbolic systems. *Comptes rendus de l'Académie des sciences. Série II, Mécanique, physique, chimie, astronomie*, 323, n°6:397–403, 1996.

[117] Alain Lerat. Steady discrete shocks of high-order {RBC} schemes. *Journal of Computational Physics*, 252(0):350 – 364, 2013.

[118] M. Lesieur. *Turbulence in Fluids*. Martinus Nijhoff Publishers, Dordrecht, 1987.

[119] R.J. Leveque. *Numerical Methods for Conservations Laws*. Birkhäuser Verlag, 1990.

[120] Y. Liu, M. Vinokur, and Z.J. Wang. Discontinuous spectral difference method for conservation laws on unstructured grids. *Journal of Computational Physics*, 216:780–801, 2006.

[121] Y. Liu, M. Vinokur, and Z.J. Wang. Spectral (finite) volume method for conservation laws on unstructured grids v: Extension to three-dimensional systems. *Journal of Computational Physics*, 212:454–472, 2006.

[122] H. Luo, J.D. Baum, and R. Löhner. Matrix-free implicit method for computing unsteady flows on unstructured grids. *Computers and Fluids*, 30:137–159, 2001.

[123] R.W. MacCormack and A.J. Paullay. Compuational effinciency achieved by time-splitting of finite-difference operators. *AIAA Paper*, pages 72–154, 1972.

[124] R.W. MacCormack, A.W. Rizzi, and M. Inouye. *Steady supersonic flowfields with embendded subsonic regions.*, pages 424–47. Academic Press, 1976.

[125] B. Maugars, B. Michel, and P. Cinnella. High-order and conservative method for patched grid interfaces. In *32nd AIAA Applied Aerodynamics Conference*, Atlanta, USA, 2014.

[126] F.R. Menter. Two-equation eddy-viscosity turbulence models for engineering applications. *AIAA Journal*, 32(8):1598–1605, August 1994.

[127] K. Morinishi. *Computers & Fluids*, 21-331, 1992.

[128] R.H. Ni. A multiple grid scheme for solving the euler equations. *AIAA Journal*, 20:1565–1571, 1981.

[129] P.-Y. Outtier, C. Content, and P. Cinnella. High-order residual-based compact schemes for compressible flows on overset grids. In *32nd AIAA Applied Aerodynamics Conference*, AIAA Aviation. American Institute of Aeronautics and Astronautics, June 2014.

[130] P.-Y. Outtier, C. Content, P. Cinnella, and B. Michel. The high-order dynamic computational laboratory for cfd research and applications. In *21st AIAA Computational Fluid Dynamics Conference*, 24-27 June 2013, San Diego, California.

[131] Pierre-Yves Outtier, Cedric Content, Paola Cinnella, and Bertrand Michel. The high-order dynamic computational laboratory for CFD research and applications. In *21st AIAA Computational Fluid Dynamics Conference*, Fluid Dynamics and Co-located Conferences. American Institute of Aeronautics and Astronautics, June 2013.

[132] Henri Padé. *Sur la répresentation approchée d'une fonction par des fractions rationelles.* PhD thesis, Annales scientifiques l'École Normale Supérieure, 3ème série, tome 9, 1892, p3-93 (supplément).

[133] Francisco Palacios, Michael R Colonno, Aniket C Aranake, Alejandro Campos, Sean R Copeland, Thomas D Economon, Amrita K Lonkar, Trent W Lukaczyk, Thomas W R Taylor, and Juan J Alonso. Stanford University Unstructured (SU 2): An open-source integrated computational environment for multi-physics simulation and design. *AIAA Journal*, 2013-0287:1–60, 2013.

[134] James W. Lottes Paul F. Fischer and Stefan G. Kerkemeier. nek5000 Web page, 2008. http://nek5000.mcs.anl.gov.

[135] J.M.C. Pereira, M.H. Kobayashi, and J.C.F. Pereira. A fourt-order accurate finite volume compact method for incompressible navier–stokes solutions. *Journal of Computational Physics*, 167:217–243, 2001.

[136] S. Peron. *Méthode d'assemblage de maillages recouvrants pour des simulations autour de géometries complexes en aérodynamique compressible.* PhD thesis, Arts et Métiers ParisTech, 2014.

[137] S. Peron and C Benoit. Automatic off-body overset adaptive cartesian mesh method based on an octree approach. *Journal of Computational Physics*, 232:153–173, 2013.

[138] Stéphanie Péron and Christophe Benoit. Automatic off-body overset adaptive Cartesian mesh method based on an octree approach. *Journal of Computational Physics*, 232(1):153–173, January 2013.

[139] M. Poinot. Development of python/cgns solutions, http://pycgns.sourceforge.net/.

[140] D. Poirier, S. R. Allmaras, D. R. McCarthy, M. F. Smith, and F. Y. Enomoto. The cgns system. In *AIAA Paper 98-3007*, 1998.

[141] C. D. Pope. *Turbulent Flows*. Cambridge University Press, 2000.

[142] T. Quintino and H. Deconinck. *COOLFluiD - A collaborative simulation environment for research in aerodynamics*, volume 113 of *Notes on Numerical Fluid Mechanics*. 2010.

[143] O. Reynolds. On the dynamical theory of incompressible viscous fluids and the determination of the criterion. *Phil. Trans. of the Roy. Soc. of London*, 186:123–164, 1895.

[144] A. Rezgui. *Schémas numériques compacts et non compacts d'ordre trois pour le calcul d'écoulements compresibles*. PhD thesis, Arts et Métiers ParisTech, October 1997.

[145] A. Rezgui, P. Cinnella, and A. Lerat. Third-order accurate finite volume schemes for euler computations on curvilinear mesches. *Computers and fluids*, 30:875–901, 2001.

[146] A. Rezgui, P. Cinnella, and A. Lerat. Third-order finite volume schemes for euler computations on curvilinear meshes. *Computers and Fluids*, 30:875–901, 2001.

[147] Ali Rezgui. An analysis of accuracy and convergence of finite volume methods. *Computational Fluid Dynamics Journal*, 8(3):369–377, 1999.

[148] Mario Ricchiuto and Rémi Abgrall. Explicit Runge–Kutta residual distribution schemes for time dependent problems: Second order case. *Journal of Computational Physics*, 229(16):5653–5691, August 2010.

[149] D. P. Rizzetta, M. R. Visbal, and P. E. Morgan. A high-order compact finite-difference scheme for large-eddy simulation of active flow control. *Progress in Aerospace Sciences*, 44:397–426, 2008.

[150] A.W. Rizzi and M. Inouye. A time-split finite-volume technique for three-dimensional blunt-body flow. *AIAA Journal*, 11:1478–85, 1973.

[151] P. L. Roe. Approximate riemann solvers, parameter vectors, and difference schemes. *Journal of Computational Physics*, 43:357–372, 1981.

[152] P.L. Roe. Characteristic-based schemes for the euler equations. *Annual Review of Fluid Mechanics*, 18:337–365, 1986.

[153] C. Rumsey, B. Wedan, T. Hauser, and M. Poinot. Recent updates to the cfd general notation system (cgns). In *AIAA Paper 2012-1264*, 2012.

[154] C.L. Rumsey. 2d zero pressure gradient flat plate verification case, 2011. `http://turbmodels.larc.nasa.gov/flatplate_sst.html`.

[155] C.L. Rumsey, R.T. Biedron, and R.E. Bartels. CFL3D Version 6.6, 2006–. `http://cfl3d.larc.nasa.gov/`.

[156] Y. Saad and M. H. Schultz. GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems. *SIAM J. Sci. Stat. Comput.*, 7(3):856–869, July 1986.

[157] O. Saunier, C. Benoît, G. Jeanfaivre, and A. Lerat. Third-order cartesien overset mesh adaptation method for solving steady compressible flows. *J. Num. Meth. Fluids*, 57, 811-838, 2008.

[158] Olivier Saunier. *Méthode d'adaptation de maillages cartésiens basée sur des schémas d'ordre élevé pour les équations d'euler d'un fluide compressible. Application aux pales de rotor d'hélicoptère.* PhD thesis, Arts et Métiers ParisTech, 2008.

[159] V. Schmitt and F. Charpin. Pressure distributions on the onera-m6-wing at transonic mach numbers. *Experimental Data Base for Computer Program Assessment. Report of the Fluid Dynamics Panel Working Group 04*, AGARD AR 138, May 1979.

[160] K. Sebastian and C.W. Shu. Multidomain weno finite difference method with interpolation at subdomain interfaces. *J. Scient. Comput.*, 19:406–438, 2003.

[161] Yves Secretan, Gouri Dhatt, and Dinh Nguyen. Compressible viscous flow around a naca-0012 airfoil. In MarieOdile Bristeau, Roland Glowinski, Jacques Periaux, and Henri Viviand, editors, *Numerical Simulation of Compressible Navier-Stokes Flows*, volume 18 of *Notes on Numerical Fluid Mechanics*, pages 219–236. Vieweg+Teubner Verlag, 1987.

[162] S. E. Sherer and R. Visbal, M. Multi-resolution implicit large eddy simulations using a high-order overset-grid approach. *International Journal for Numerical Methods in Fluids*, 55:455–482, 2007.

[163] Scott E. Sherer and James N. Scott. High-order compact finite-difference methods on general overset grids. *Journal of Computational Physics*, 210(2):459–496, December 2005.

[164] J. Smagorinsky. General Circulation Experiments with the Primitive Equations. *Monthly Weather Review*, 91:99, 1963.

[165] J. L. Steger, F. C. Dougherty, and J. A. Benek. A Chimera Grid Scheme. In ASME FED K. N. Ghia, editor, *Advances in grid Generation*, page vol 5, 1985.

[166] J. L. Steger, F. C. Dougherty, and J. A. Benek. A chimera grid scheme. In ASME FED K. N. Ghia, editor, *Advances in grid Generation*, volume 5, 1985.

[167] W. A. Stein et al. *Sage Mathematics Software (Version x.y.z)*. The Sage Development Team, YYYY. `http://www.sagemath.org`.

[168] Roger C. Strawn. Software design strategies for multidisciplinary computational fluid dynamics. In *Seventh International Conference on Computational Fluid Dynamics (ICCFD7), Big Island, Hawaii*, July 9-13, 2012.

[169] B. Stroustrup. *The C++ Programming Language*. third edition edition, 1997.

[170] Y. Sun, Z.J. Wang, and Y. Liu. Spectral (finite) volume method for conservation laws on unstructured grids. *Journal of Computational Physics*, 178:210–251, 2002.

[171] Y. Sun, Z.J. Wang, and Y. Liu. Spectral (finite) volume method for conservation laws on unstructured grids iii: Extension to viscous flow. *Journal of Computational Physics*, 215:41–58, 2006.

[172] W Sutherland. The viscosity of gases and molecular force. *Philosophical Magazine*, S. 5, 36:507–531, 1893.

[173] P.K. Sweby. High resolution schemes using flux-limiters for hyperbolic conservation laws. *SIAM J. Num. Anal.*, 21:995–1011, 1984.

[174] C.K.W. Tam and F.Q. Hu. An optimized mutli-dimensional interpolation scheme for computational aeroacoustics applications using overset grids. *AIAA paper*, 2812, 2004.

[175] C.K.W. Tam and K. Kurbatskii. A wavenumber based extrapolation and interpolation method for use in conjunction with high-order finite difference schemes. *Journal of Computational Physics*, 157:588–617, 2000.

[176] C.K.W. Tam and J.C. Webb. Dispersion-relation-preserving finite difference schemes for computational acoutics. *Journal of Computational Physics*, 107:262–281, 1993.

[177] J. F. Thompson. *Numerical Grid Generation*. Elsevier, New York, 1982.

[178] Kevin W Thompson. Time dependent boundary conditions for hyperbolic systems. *Journal of Computational Physics*, 68(1):1 – 24, 1987.

[179] Kevin W Thompson. Time-dependent boundary conditions for hyperbolic systems, ii. *Journal of Computational Physics*, 89(2):439 – 461, 1990.

[180] S. Van Der Walt, S. C. Colbert, and G. Varoquaux. The numpy array: A structure for efficient numerical computation. *Computing in Science and Engineering*, 13(2):22–30, 2011.

[181] B. Van Leer. Towards the ultimate conservative difference scheme. v : A second order sequel to godunov's method. *Journal of Computational Physics*, 32:101–136, 1979.

[182] NPARC Alliance Verification and Validation Archive. `http://www.grc.nasa.gov/WWW/wind/valid/archive.html`.

[183] P. Walters. *An Introduction to Ergodic Theory*. Springer, 1982.

[184] Z. J. Wang. A fully conservative interface algorithm for overlapped grids. *Journal of Computational Physics*, 122:1–11, 1995.

[185] Z. J. Wang. A fully conservative interface algorithm for overlapped grids. *Journal of Computational Physics*, 122:96–106, 1995.

[186] Z.J. Wang. High-order methods for the euler and navierâstokes equations on unstructured grids. *Progress in Aerospace Sciences*, 43:1–41, 2007.

[187] Z.J. Wang and Y. Liu. Spectral (finite) volume method for conservation laws on unstructured grids ii: Extension to two-dimensional scalar equation. *Journal of Computational Physics*, 179:655–697, 2002.

[188] Z.J. Wang and Y. Liu. Spectral (finite) volume method for conservation laws on unstructured grids iii: extension to two-dimensional systems. *Journal of Computational Physics*, 194:716–741, 2004.

[189] Z.J. Wang and Y. Liu. Spectral (finite) volume method for conservation laws on unstructured grids iii: One-dimensional systems and partition optimization. *Journal of Scientific Computing*, 20:137–157, 2004.

[190] D.C. Wilcox. Turbulence modelling for CFD, DCW industries,inc., la Canada, California. 1993.

[191] Andrew M. Wissink, Jayanarayanan Sitaraman, Venkateswaren Sankaran, Dimitri J. Mavriplis, and Thomas H. Pulliam. A multi-code python-based infrastructure for overset cfd with adaptative cartesian grids. *aiaa*, 2008.

[192] J. Yin and Delfs. Sound generation from gust-airfoil interaction using caa-chimera method. *AIAA paper*, 2136, 2001.

[193] Seokkwan Yoon and Antony Jameson. An lu-ssor scheme for the euler and navier-stokes equations. In *AIAA-087-600*, 1987.

[194] Y. Zhao. Computation of complex turbulent flow using matrix-free implicit dual time-stepping scheme and lrn turbulence model on unstructured grids,. *Computers and Fluids*, 33:119–136, 2004.

Architecture novatrice de code dynamique :
application au développement d'un solveur compact d'ordre élevé pour l'aérodynamique compressible dans des maillages recouvrants.

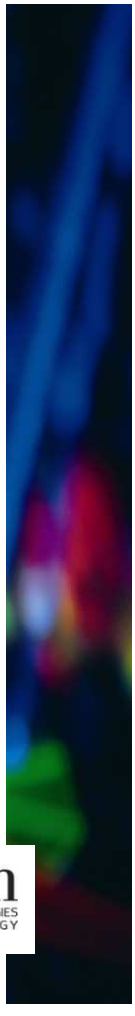
RESUME : L'utilisation de schémas numériques d'ordre élevé est généralement restreinte à des applications de recherche mettant en jeu des phénomènes physiques complexes mais des géométries simples à l'aide de maillages Cartésiens ou faiblement déformés. Il existe une demande pour une nouvelle génération de codes industriels ayant une précision accrue. Dans ce travail, nous avons été amenés à répondre à la question générale sur la façon de concevoir l'architecture d'un code CFD pouvant prendre en compte une variété de configurations géométriquement complexes, restant assez simple et permettant facilement l'implantation de nouvelles idées (méthodes numériques, modèles) avec un effort de développement minimal, et utilisant des schémas numériques d'ordre élevé ainsi que des modèles physiques avancés. Cela a nécessité des choix innovants en termes de langages de programmation, structure de données et stockage, et sur l'architecture de code, choix qui vont au-delà du simple développement d'une famille spécifique de schémas numériques. Une solution (code DynHoLab) alliant les langages Python et Fortran est proposé avec les détails sur les concepts à la base de l'architecture du code. Les méthodes numériques implantées dans le code sont validées sur des cas-tests de complexité croissante, démontrant en passant la variété des physiques et géométries actuellement réalisables avec DynHoLab. Puis, basé sur ce code, ce travail présente un moyen de gérer des géométries complexes tout en augmentant le degré de précision des méthodes numériques. Afin d'appliquer les régimes d'ordre élevé RBC à des géométries complexes, la stratégie actuelle consiste en une mise en œuvre multi-domaine sur des maillages recouvrants.

Mots clés : aérodynamique compressible, ordre élevé, maillages recouvrants, schéma compact base sur le résidu, méthodes numériques.

A new dynamic code architecture for CFD computations:
application to the development of an overset-grid compact high-order solver for compressible aerodynamics.

ABSTRACT : High-order numerical schemes are usually restricted to research applications, involving highly complex physical phenomena but simple geometries, and regular Cartesian or lowly deformed meshes. A demand exists for a new generation of industrial codes of increased accuracy. In this work, we were led to address the general question of how to design a CFD code architecture that: can take into account a variety of possibly geometrically complex configurations; remains simple and modular enough to facilitate the introduction and testing of new ideas (numerical methods, models) with a minimal development effort; use high-order numerical discretizations and advanced physical models. This required some innovative choices in terms of programming languages, data structure and storage, and code architecture, which go beyond the mere development of a specific family of numerical schemes. A solution mixing Python and Fortran languages is proposed with details on the concepts at the basis of the code architecture. The numerical methods are validated on test-cases of increasing complexity, demonstrating at the same time the variety of physics and geometry currently achievable with DynHoLab. Then, based on the computational framework designed, this work presents a way to handle complex geometries while increasing the order of accuracy of the numerical methods. In order to apply high-order RBC schemes to complex geometries, the present strategy consists in a multi-domain implementation on overlapping structured meshes.

Keywords : compressible aerodynamics, high-order, overlapping grids, residual-based compact scheme, CFD.



ARTS ET MÉTIERS ParisTech

ParisTech
INSTITUT DES SCIENCES ET TECHNOLOGIES
PARIS INSTITUTE OF TECHNOLOGY