université de BORDEAUX

THÈSE de DOCTORAT

PRÉSENTÉE à

L'UNIVERSITÉ DE BORDEAUX

ÉCOLE DOCTORALE DE MATHÉMATIQUES ET D'INFORMATIQUE

Spécialité : Mathématiques Appliquées et Calcul Scientifique

Par **Pascal JACQ**

Méthodes numériques de type Volumes Finis sur maillages non structurés pour la résolution de la thermique anisotrope et des équations de Navier-Stokes compressibles.

Finite Volume methods on unstructured grids for solving anisotropic heat transfer and compressible Navier-Stokes equations.

Directeur de thèse : **Pierre-Henri Maire**
Directeur de thèse : **Rémi Abgrall**

Soutenue le : 9 juillet 2014

Après avis des rapporteurs :

Florian De Vuyst	Professeur, ENS Cachan
Boniface Nkonga	Professeur, Université de Nice

Devant la commission d'examen composée de :

Rémi Abgrall	Professeur, Université Zürich	Directeur
Jean-Paul Caltagirone	Professeur, Institut de Mécanique et d'Ingénierie	Président
Florian De Vuyst	Professeur, ENS Cachan	Rapporteur
Pierre-Henri Maire	Ingénieur-Chercheur, CEA; Prof. associé, IPB ..	Directeur
Boniface Nkonga	Professeur, Université de Nice	Rapporteur
Jean Roman	Professeur, Directeur Scientifique adjoint INRIA	Examinateur

A mes proches.

Remerciements

Je tiens tout d'abord à remercier mon directeur de thèse, Pierre-Henri Maire, pour ces 3 années de collaboration. J'ai beaucoup apprécié son côté pédagogique, sa façon bien à lui de présenter les problèmes qui paraissent soudain si simples. Il y a bien sûr eu des hauts et des bas durant ces années mais je tiens à le remercier de m'avoir poussé et aidé à arriver au bout de cette thèse.

Je tiens aussi à remercier chaleureusement mon autre directeur de thèse, Rémi Abgrall. C'est vrai que notre collaboration a été plus intense lorsque j'étais ingénieur dans l'équipe Bacchus de l'INRIA, que lorsque j'étais en thèse, mais c'est tout de même grâce à lui que j'ai pu rencontrer Pierre-Henri et effectuer ces travaux. Ses remarques et conseils m'ont été très utiles lors de la rédaction de ce manuscrit. Je tiens aussi à saluer sa pédagogie bien à lui et sa grande connaissance des vins Bretons.

Je tiens également à remercier Florian De Vuyst d'avoir accepté de rapporter cette thèse. Ses remarques et questions sur le manuscrit m'ont beaucoup apporté. Je tiens tout particulièrement à remercier mon autre rapporteur Boniface Nkonga qui m'a mis le pied à l'étrier il y a plus de 7 ans en me proposant un poste d'ingénieur à l'INRIA. A l'époque, à son plus grand désarroi, je ne voulais pas entendre parler de thèse. Après quelques années de réflexion, j'ai changé d'avis et je suis très heureux qu'il ait participé lui aussi à tout ça.

Merci beaucoup à Jean-Paul Caltagirone d'avoir accepté de présider ce jury quelques jours avant de devenir émérite. Merci aussi à Jean Roman d'avoir accepté de faire partie de ce jury même si l'aspect HPC n'était pas prédominant dans cette thèse.

Je tiens aussi à remercier chaleureusement Jean Claudel qui m'a beaucoup aidé pendant cette thèse, surtout pour la partie Mécanique des Fluides. Sans ses connaissances et son soutien je pense que les "carbuncles" et autres bizarreries numériques seraient venues à bout de ma santé mentale. J'ai aussi beaucoup apprécié sa gentillesse et son humour, aussi décalé que ses goûts pour la soupe. Enfin, j'ai aussi toujours aimé découvrir de nouvelles séries télé ou encore de discuter avec lui de sa passion pour les vélos couchés.

Je tiens bien évidement à remercier tous les collègues et amis qui m'ont fait passer des bons moments au travail et en dehors. Dans la catégorie thésards, ces camarades de galères, compagnons de pauses café et amoureux des happy hours, il y a eu François que je souhaiterais remercier pour avoir participé à la création de la 1ère version de ce désormais célèbre jeu vidéo "Punch" et pour toutes les autres conneries qui nous ont fait passer de bonnes journées au boulot. Cyril, toujours pret à se lancer avec moi dans la programmation d'un projet fou censé nous rendre riches (ou pas). Gaby, qui a eu le malheur de tomber dans mon bureau au début de son stage, désolé de t'avoir inculqué quelques mauvaises habitudes. Matthieu, mon dernier co-bureau au CEA toujours prêt pour une pause café ou pour me mettre ma raclée au squash.

Damien pour ses sites d'informations (9gag et compagnie), pour son côté jovial et sa passion pour les logiciels de repartitionnement.

Je tiens aussi à remercier les sportifs qui m'ont accompagné lorsque je me suis mis au running dans le club "trail et apéro". Merci à Orel et Fredo les coatchs de luxe, sportifs de l'extrême et membres permanents de l'équipe de France de vin rouge. Merci surtout à Abdou mon compagnon d'entrainement et de compétition avec qui on a parcouru de nombreux kilomètres, et qui m'a poussé à aller bien au-delà de ce que je pensais possible. Il faudra qu'un jour on fasse homologuer notre technique d'entrainement à handicap.

Au CEA je tiens à remercier Xavier pour son accueil dans l'équipe. Anne-Pascale pour sa gentillesse, son efficacité et pour les costumes et les photos des poulets de la médocaine. Jean-Jacques pour les exposés et pour ses innombrables connaissances. Olivier pour son côté geek avec ses 17 PC qui lui servent de chauffage l'hiver et pour le fait qu'il vienne manger avec les "jeunes" uniquement lorsqu'il y a des filles. Pierre qui le subit à temps plein. Céline pour sa bonne humeur, Isabelle pour sa gentillesse. Murielle et Agnès pour les pauses cafés. Jean-Marc pour ses questions python, Julien pour avoir supporté Gaby. Gérard et son côté boute-en-train. David pour sa gentillesse, David pour sa méchanceté. Thanh-ha pour son gâteau vert et son rire après les pots. Pierre pour son stage entre deux séances de voile.

A l'INRIA, il y a bien sûr Manu avec ses théories sur l'amour, qui m'a tout enseigné au babyfoot pour devenir champion du monde de l'INRIA. Mathieu, que je ne remercie pas pour m'avoir légué son appli du café mais plutôt pour son côté sociable, et pour les sorties karting où je lui mettais sa misère. Robin le champion de France de Tremulous qui a fini sa thèse malgré son emploi du temps restreint. Guillaume pour sa passion pour Kaamelott, son côté geek et pour la version ultime malheureusement perdue de FBx. Christelle et XL avec qui on a passé de bons moments avant que notre bureau ne brûle. Tous les anciens Nico, Adam, Jérémie, Stéphane, Algiane, Mario, Guilhem, Patator,...
Je te remercie toi aussi qui lis cette ligne sans être tombé plus haut sur ton nom, c'est surement un oubli et j'en suis désolé. N'hésite pas à me contacter pour toute réclamation.

Je tiens à remercier également mes parents pour avoir assisté à ma soutenance. C'était important pour moi qu'ils soient là, ça l'était pour eux aussi j'imagine. Je tiens à m'excuser une fois de plus d'avoir rédigé cette thèse en cette langue étrangère que sont pour eux les mathématiques (ah et aussi l'anglais...).

Je ne saurais comment remercier Audrey qui m'a porté et supporté pendant cette fin de thèse. Je ne serais sûrement pas arrivé au bout de ce manuscrit sans son soutien, ses encouragements, ses petits plats et tous les sacrifices auxquels elle a consenti. Pour tout ça et pour le reste, merci énormément.

Résumé

Méthodes numériques de type Volumes Finis sur maillages non structurés pour la résolution de la thermique anisotrope et des équations de Navier-Stokes compressibles

Lors de la rentrée atmosphérique nous sommes amenés à modéliser trois phénomènes physiques différents. Tout d'abord, l'écoulement autour du véhicule entrant dans l'atmosphère est hypersonique, il est caractérisé par la présence d'un choc fort et provoque un fort échauffement du véhicule. Nous modélisons l'écoulement par les équations de Navier-Stokes compressibles et l'échauffement du véhicule au moyen de la thermique anisotrope. De plus le véhicule est protégé par un bouclier thermique siège de réactions chimiques que l'on nomme communément ablation.

Dans le premier chapitre de cette thèse nous présentons le schéma numérique de diffusion CCLAD (Cell-Centered LAgrangian Diffusion) que nous utilisons pour résoudre la thermique anisotrope. Nous présentons l'extension en trois dimensions de ce schéma ainsi que sa parallélisation.
Nous continuons le manuscrit en abordant l'extension de ce schéma à une équation de diffusion tensorielle. Cette équation est obtenue en supprimant les termes convectifs de l'équation de quantité de mouvement des équations de Navier-Stokes. Nous verrons qu'une pénalisation doit être introduite afin de pouvoir inverser la loi constitutive et ainsi appliquer la méthodologie CCLAD. Nous présentons les propriétés numériques du schéma ainsi obtenu et effectuons des validations numériques.
Dans le dernier chapitre, nous présentons un schéma numérique de type Volumes Finis permettant de résoudre les équations de Navier-Stokes sur des maillages non-structurés obtenu en réutilisant les deux schémas de diffusion présentés précédemment.

Mots clés : Méthodes Volumes Finis, Maillages Non-Structurés, Thermique Anisotrope, Equations de Navier-Stokes compressibles, Calculs Hautes Performances

Abstract

Finite Volume methods on unstructured grids for solving anisotropic heat transfer and compressible Navier-Stokes equations

When studying the problem of atmospheric reentry we need to model three different physical phenomenons. First, the flow around the atmospheric reentry vehicle is hypersonic, it is characterized by the presence of a strong shock which leads to a rapid heating of the vehicle. We model the flow using the compressible Navier-Stokes equations and the heating of the vehicle is modeled with the anisotropic heat transfer equation. Furthermore the vehicle is protected by an heat shield, where thermochemical reactions, commonly named ablation, occurs.

In the first chapter of this thesis we introduce the numerical diffusion scheme CCLAD (Cell-Centered LAgrangian Diffusion) that we use to solve the anisotropic heat diffusion. We develop its non trivial extension to three-dimensional geometries and present its parallelization.
We continue this thesis by the presentation of the extension of this scheme to tensorial diffusion. This equation is obtained by suppressing the convective terms of the momentum equation of the Navier-Stokes equations. We show that we need to introduce a penalization term in order to be able to invert the constitutive law. The invertibility of the constitutive law allows us to apply the CCLAD methodology to this equation straightforwardly. We present the numerical properties of this scheme and show numerical validations.
In the last chapter, we present a Finite Volume scheme on unstructured grids that solves the compressible Navier-Stokes equations. This numerical scheme is mainly obtained by gathering the contributions of the two diffusion schemes we developed in the previous chapters.

Keywords : Finite Volume Methods, Unstructured Grids, Anisotropic Heat Transfer, Compressible Navier-Stokes Equations, High Performance Computing

Contents

Résumé

Ce manuscrit traite de la conception et de l'analyse de schémas numériques de type Volumes Finis novateurs pour la résolution de la thermique anisotrope et des équations de Navier-Stokes compressibles sur maillages non-structurés. Le contexte de cette étude est la simulation numérique de la rentrée atmosphérique, avec comme finalité l'étude du phénomène d'ablation qui occure dans les protections thermiques des véhicules de rentrée.

Figure 1: Exemple d'un véhicule de rentrée atmosphérique: IXV de l'ESA.

Pour modéliser la rentrée atmosphérique nous sommes amenés à prendre en compte trois phénomènes physiques différents. Tout d'abord, l'écoulement autour du véhicule entrant dans l'atmosphère est hypersonique, il est caractérisé par la présence d'un choc fort et provoque un fort échauffement du véhicule. Nous modélisons l'écoulement par les équations de Navier-Stokes compressibles et l'échauffement du véhicule au moyen de la thermique anisotrope. Les flux thermiques obtenus de part et d'autre de la paroi de l'objet sont ensuite les principaux ingrédients qui pilotent les nombreuses réactions chimiques se déroulant au sein du bouclier thermique et que l'on nomme communément ablation.

Schéma numérique de type Volumes Finis pour résoudre une équation de diffusion anisotrope sur des maillages non-structurés.

Nous nous intéressons tout d'abord à la modélisation de la diffusion thermique au sein du véhicule de rentrée. Les matériaux utilisés dans la construction de ce type de véhicule sont des matériaux composites. Ces matériaux présentent de fortes anisotropies, ce qui nous ammène à

traiter de la diffusion thermique anisotrope. Pour ce faire, nous présentons la construction du schéma numérique de diffusion CCLAD (Cell-Centered LAgrangian Diffusion).

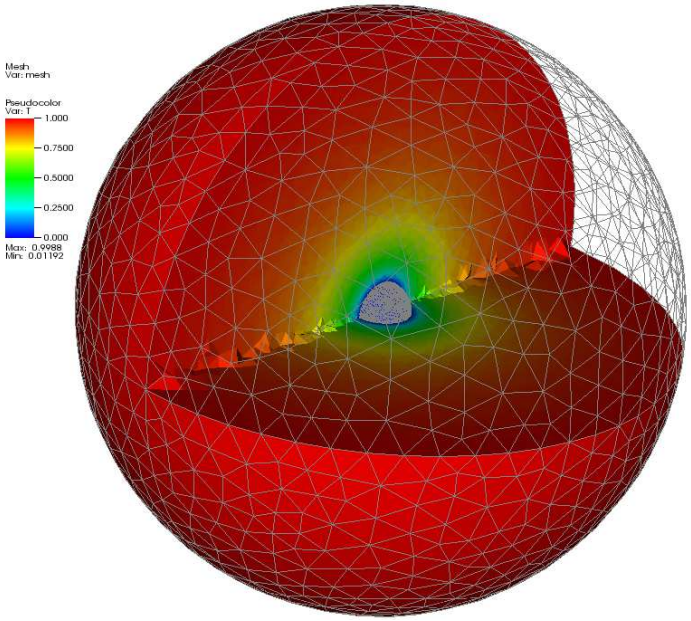


Figure 2: Exemple d'un calcul de diffusion sur un cas test 3D d'une sphère tronquée.

Après avoir présenté le schéma sous sa forme classique en deux dimensions d'espace, nous présentons son extension non triviale en trois dimensions. La caractéristique de ce schéma est le partitionnement de chaque volume de contrôle polyédrique en sous-cellules, ainsi que le partitionnement de chaque face de cellule en sous-faces. Nous définissons des variables auxiliaires au centre de chaque sous-face, afin de construire des flux, au moyen d'une formulation variationnelle locale. Ces variables auxilaires sont ensuite éliminées par l'inversion d'un système linéaire local aux noeuds, obtenu en écrivant les conditions de continuité de la température et des flux sur chaque sous-face. Ceci nous permet alors de construire une matrice globale de diffusion, creuse, faisant intervenir uniquement les degrés de libertés du schéma situés au barycentre des cellules.
La taille des maillages tridimensionnels étant bien plus conséquente que celle des maillages bidimensionnels, nous montrons qu'il est nécessaire de paralléliser le schéma. Du fait du caractère local du schéma CCLAD nous montrons que la parallélisation réalisée est efficace. Nous voyons que nous obtenons un speedup de 100 pour 128 coeurs.
Les propriétés mathématiques du schéma sont étudiées et nous montrons au moyen de nombreux cas tests numériques que le schéma obtenu est d'ordre 2 sur des maillages de quadrangles et d'hexaèdres déformés de manière régulière ainsi que sur des maillages de triangles et de tétraèdres.
Nous traitons aussi du cas particulier des pyramides en trois dimensions qui pose problème à l'écriture du schéma CCLAD autour du sommet principal entouré de 4 faces. Nous proposons alors une modification simple au schéma CCLAD pour ce type de sommet afin de pouvoir traiter ces éléments particuliers. Cette modification nous permet alors d'écrire le schéma CCLAD sur des maillages polyédriques quelconques.

Schéma numérique de type Volumes Finis pour résoudre une équation de diffusion tensorielle sur des maillages non-structurés.

Nous abordons ensuite l'extension du schéma CCLAD à une équation de diffusion tensorielle. Cette équation est obtenue en supprimant les termes convectifs de l'équation de quantité de mouvement des équations de Navier-Stokes.
Nous montrons que l'extension à la diffusion tensorielle n'est pas triviale. En effet nous mettons en évidence que la loi constitutive est non-inversible sur l'espace des tenseurs généraux du second ordre. Cette non inversibilité poserait problème lors de la phase d'élimination des variables auxiliaires. En s'appuyant sur les travaux d'Arnold-Falk en élasticité linéaire, nous proposons alors une pénalisation de la loi constitutive, en y ajoutant un tenseur à trace nulle, qui permet de rétablir l'inversibilité de la loi tout en laissant inchangée la divergence du tenseur. Ceci nous amène à formuler un problème équivalent qui nous permet d'appliquer alors naturellement la méthodologie CCLAD.
Nous présentons les propriétés mathématiques du schéma ainsi obtenu, et insistons sur le fait que la pénalisation apportée conserve la propriété de divergence nulle au niveau discret.
Nous montrons au moyen de nombreuses validations numériques que le schéma est d'ordre 2 sur des maillages de quadrangles déformés de manière régulière et sur des triangles.
Enfin, grâce à une méthodologie de programmation élégante, nous montrons que ce schéma bénéficie pleinement des développements informatiques effectués précédement, nottament en terme de parallélisation.

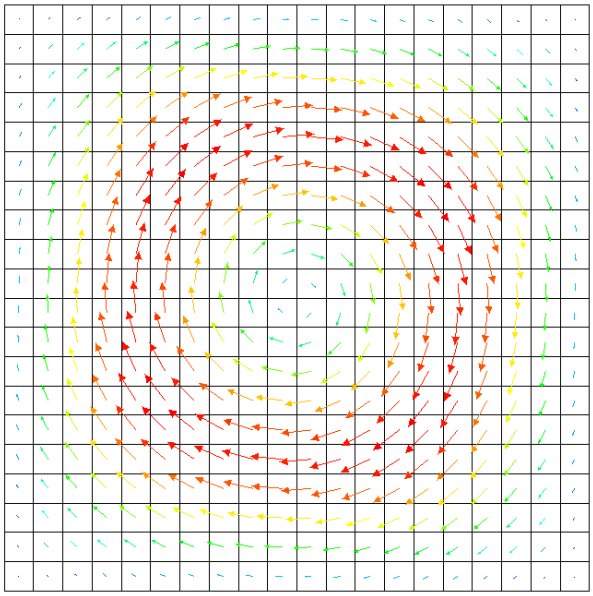


Figure 3: Exemple d'un calcul de diffusion tensoriel sur un cas test représentant un vortex.

Schéma numérique de type Volumes Finis pour résoudre les équations de la mécanique des fluides sur des maillages non-structurés.

Nous présentons ensuite les équations de la mécanique des fluides. Nous écrivons tout d'abord les équations d'Euler qui réprésentent un fluide non visqueux et non conducteur thermiquement. Nous dérivons alors un schéma classique de type volume fini d'ordre 2 pour résoudre ces équations. Ce schéma utilise des solveurs de Riemann approchés pour définir les flux numériques,

et utilise une recontruction de type MUSCL assortie de limiteurs de pente.
Nous présentons des cas tests de validation et abordons le problème du “carbuncle” qui apparait dans les écoulements à fort Mach que nous souhaitons aborder. Nous proposons alors une méthodologie de type flux tournés qui permet d'erradiquer ce phénomène tout en conservant une grande précision numérique.

Nous présentons alors les équations de Navier-Stokes compressibles qui décrivent la mécanique des fluides. Grâce à une décomposition de ces équations, nous proposons de les résoudre en réutilisant l'ensemble des schémas numériques developpés au cours de cette thèse. Nous présentons les différentes matrices de passages nécessaires à l'intégration des contributions matricielles de chaque schéma au problème global.
Finalement, nous présentons des cas tests de validation et effectuons des comparaisons avec d'autres logiciels de mécanique des fluides développés par la NASA sur des cas tests de rentrée athmosphérique. Les résultats numériques obtenus sont très convainquants et permettent de valider le choix de nos différents schémas numériques.

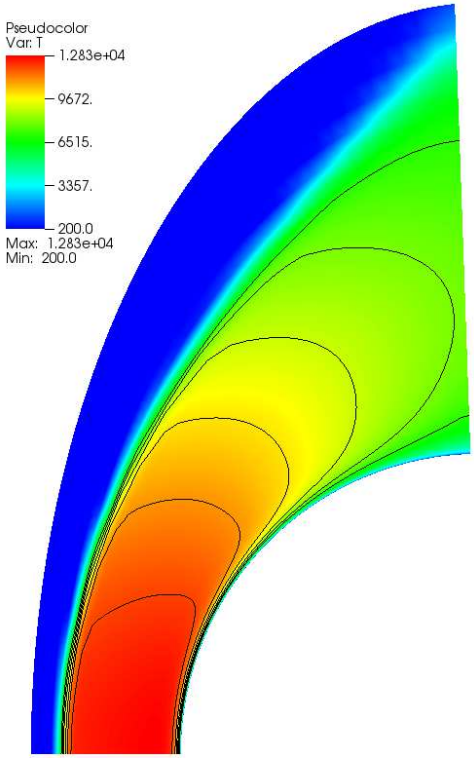
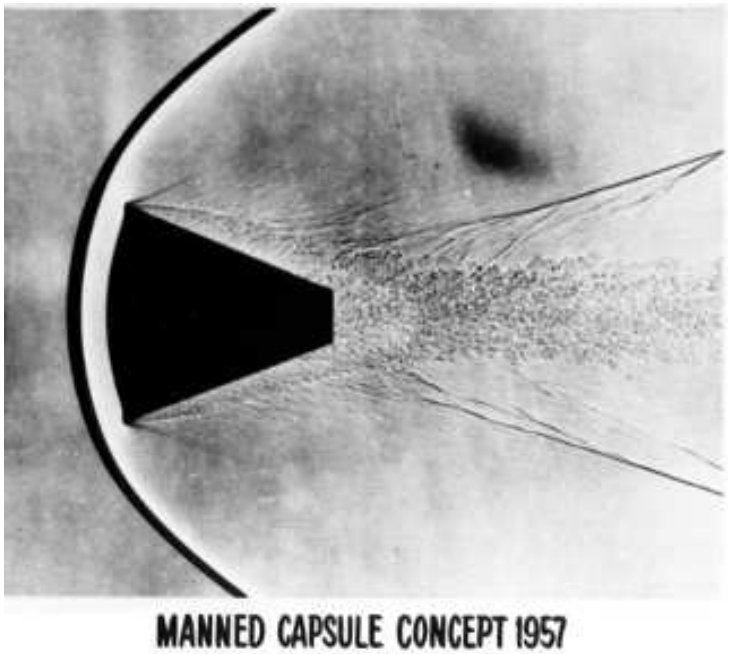


Figure 4: Exemple d'un calcul Navier-Stokes représentant la répartition de la température autour d'un cylindre à Mach 8.

Introduction

This document aims at presenting the numerical simulation of heat transfer in the domain of hypersonic ablation of thermal protection systems [28, 29]. In this context, one has to solve not only the compressible Navier-Stokes equations for the fluid flow but also the anisotropic heat equation for the solid materials which compose the thermal protection. These two models, *i.e.*, the Navier-Stokes equations and the heat equation, are strongly coupled by means of a surface ablation model which describes the removal of surface materials resulting from complex thermochemical reactions such as sublimation and oxydation.

Figure 5: Schlieren photography of an early reentry-vehicle concept from NASA. Taken from www.nasaimages.org.



In this work, we are considering the atmospheric reentry of human made vehicles. As pictured in Figure 6, these vehicles can take the form of space shuttles or space probes. These vehicles after a journey in space need to come back to earth, or to land on an other planet as what happened with the Curiosity rover that landed on Mars in 2012. When entering the planet atmosphere, these vehicles are moving at hypersonic speeds and need to be slowed down to land safely. For instance, the starting reentry velocity of Curiosity was $5800\,\mathrm{ms}^{-1}$, at the end of the reentry this velocity was reduced to $470\,\mathrm{ms}^{-1}$. At that time conventional parachutes were used to complete the landing. During the first phase of the reentry, also called aerobraking, the flow around the reentry vehicle is hypersonic. Figure 5 shows an interesting representation of an hypersonic flow around an early reentry vehicle concept from NASA. The Schlieren photography process

Figure 6: Artists view of reentry vehicles : from http://www.esa.int/spaceinimages/.

(a) Intermediate eXperimental Vehicle from ESA.

(b) Atmospheric Re-entry Demonstrator from ESA.

used, allows us to observe the variations of the density in the fluid. We can see clearly on this picture that a strong bow shock forms in front of the reentry-vehicle. This shock induces a large increase of the temperature of the flow in front of the vehicle. The reentry vehicle undergo extremely large heat fluxes which increases its temperature. For the Curiosity rover the heat shield experienced peak temperatures of up to $2090°C$. At a certain point the temperature is so high that chemical reactions occurs in the Thermal Protection System (TPS). These complex chemical reactions are mentioned in the following as ablation. The TPS are conceived with specific materials which when submitted to these extreme temperatures produces endothermic chemical reactions. This allows to decrease the temperature of the reentry vehicle which can remains intact until the final landing. All these phenomenons are pictured in Figure 7.

The book of Anderson [15] gives a great overview of the physical and mathematical fundamentals of hypersonic and high-temperature gas dynamics. It starts from the description of extremely simple, yet accurate, methods such as the Local Surface Inclination Method. These methods

Figure 7: Description of the physical phenomenons occurring during reentry.

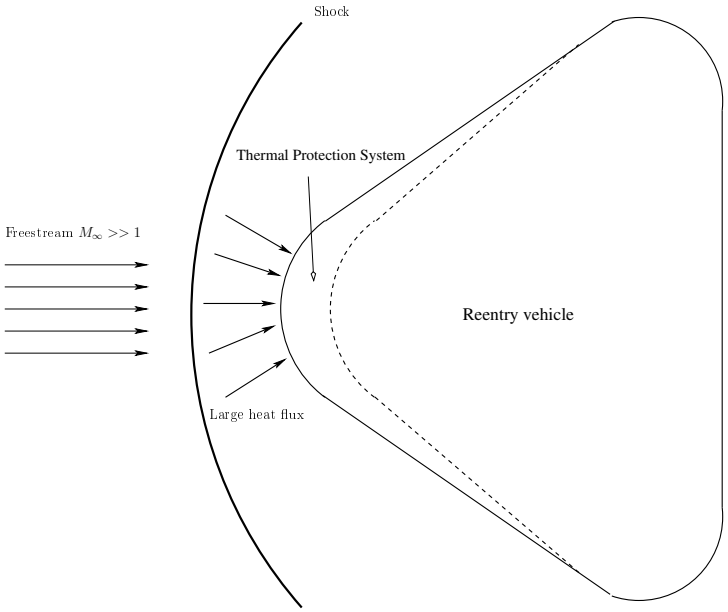
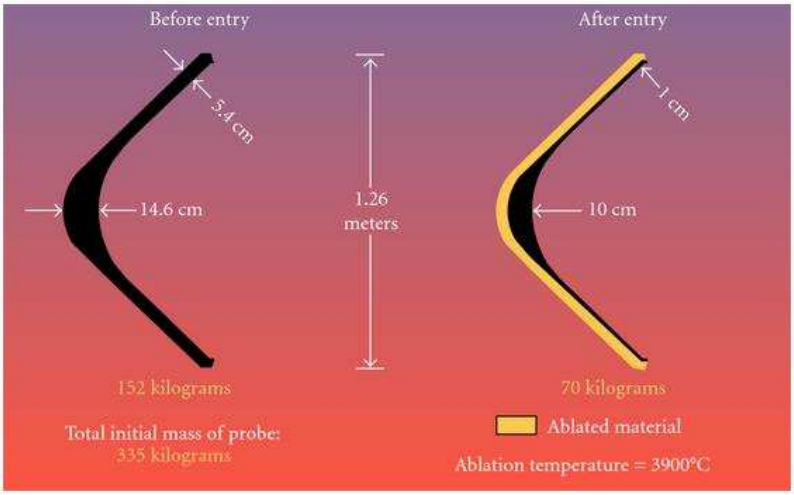


Figure 8: Galileo probe heat shield ablation : taken from www.nasaimages.org.



were used in the early days of hypersonic aerodynamics, in the 1950s, when very low computational power was available. He also presents in his book, the complex physical phenomenons that happens in the gas at very high-temperatures. These problems can now be modeled accurately with the help of modern high-order numerical methods and the huge computational power available on modern supercomputers.

One of the main purpose of numerical simulations when dealing with atmospheric reentry is the dimensioning of the Thermal Protection System of a reentry vehicle. For instance, in Figure 8, we display the description of the heat shield of the Galileo probe, before and after its reentry. We can observe that the mass of the heat shield was divided by two in the reentry phase. Yet, the shield is still ten centimeter thick around the leading edge. This is why more accurate simulations are needed. With the help of more accurate models, the design of the heat shield

could have been thinner around the leading edge. The shield would then have been lighter, which is a very important issue when sending an object to space.

The CEA-CESTA, where this PhD thesis took place, is very interested in the numerical simulation of atmospheric reentry. This thesis is one of the many that took place at the CEA-CESTA with a focus on different aspects of the modelization. We can first cite the thesis of Anthony Velghe [129] and Thibault Harribey [56] which were focused on the physical modeling of ablation at the microscopic level. Using very high-order numerical methods and Direct Numerical Simulations they were able to accurately model the effects of turbulence on the recession of ablative materials. In his thesis, Manuel Latige [80] studied the ablation phenomenon at a higher scale. He was interested by the study of the multiphasic ablation that happens when dealing with composite materials. These kind of materials are composed of fibers and matrices which reacts differently to the increase of temperature. These PhD thesis were devoted to a better understanding of the physical aspects of the ablation at microscopic and mesoscopic levels. From these studies accurate ablation models have been designed. This brings us back to our thesis, whose main purpose is to develop efficient parallel numerical methods on unstructured grids to contribute to the improvement of the numerical modeling of the global atmospheric reentry problem.

The remaining of this document is organized as follows. In Chapter 1, we present the construction of a Cell-Centered Finite Volume scheme for solving anisotropic diffusion equation on unstructured meshes. This Chapter follows the CCLAD methodology introduced by Maire and Breil [32, 90]. After recalling the two-dimensional version of the CCLAD scheme, we present a non trivial extension to the CCLAD scheme in three-dimensional geometries on unstructured meshes. Then, we discuss the properties of the obtained scheme. We also focus on the parallel implementation of this scheme. It is important to deal with this topic for two reasons. First in three dimensions the cost of the scheme is important due to the usage of large meshes needed for real life applications. Then, we think that, today, it is delusional to start the development of a numerical method that cannot handle the massively parallel computational trend which is happening in the world of High Performance Computing (HPC). Finally, the efficiency of the parallel implementation is discussed and the robustness and accuracy of the numerical scheme is studied with the help of numerous test cases.

In Chapter 2, we present the construction of a Cell-Centered Finite Volume scheme for solving tensorial diffusion on unstructured meshes. This tensorial diffusion equation corresponds to the viscous fluxes contained in the momentum equation of the compressible Navier-Stokes equations. The space discretization of this equation is presented as an extension to tensorial diffusion of the CCLAD scheme explained in Chapter 1. However, the construction of this scheme is not straightforward. After detailing the properties of the constitutive law, we remark that we have to introduce a divergence free term, following the work of Arnold [16], in order to renders it invertible over the space of generic second-order tensors. The invertibility of the constitutive law is an essential property for the construction of the numerical scheme. The CCLAD methodology is then successfully applied to this modified constitutive law, and allows us to build our numerical scheme. The accuracy and robustness are this numerical scheme is then assessed on a variety of numerical test cases.

Finally, Chapter 3 focuses on Computational Fluid Dynamics (CFD). We start by giving a description of the compressible Navier-Stokes equations and its properties. Then, we consider the specific case of a non viscous non heat conducting fluid, which leads us to write the Euler

equations. The properties of these equations are also discussed. We continue by presenting the construction of a Cell-Centered Finite Volume scheme to solve the Euler equations. We use the classical MUSCL approach along with the use of approximate Riemann solvers to build a second-order space discretization of these equations. Explicit and implicit time discretization are explained. This classical approach, serves as the foundation for the construction of our Cell-Centered Finite Volume scheme which solves the Navier-Stokes equations. The novelty of this scheme lies in the utilization of the numerical schemes developed in Chapter 1 and 2 in order to discretize the viscous terms of the equations. The accuracy and robustness of this numerical scheme are then assessed through the comparison between exact solutions or computational codes from NASA [1] on various validation tests cases.

A part of our work has been presented in two international conferences and led to the publication of one paper in an international journal. Another paper, devoted to the construction of a Cell-Centered Finite Volume scheme for the numerical simulation of the Navier-Stokes equations on unstructured grids, is currently in preparation.

Oral communications

- A second-order cell-centered finite volume scheme for anisotropic diffusion on three-dimensional unstructured meshes. *European Congress on Computational Methods in Applied Sciences and Engineering (ECCOMAS 2012)*, September 10-14, 2012, Vienna, Austria.
- A Finite Volume scheme on moving meshes for solving anisotropic diffusion with phase change. *Young Investigators Conference (YIC 2013)*, September 2-6, 2013, Talence, France.

Publications

- P. Jacq, P.-H. Maire and R. Abgrall. A Nominally Second-Order Cell-Centered finite volume scheme for simulating three-dimensional anisotropic diffusion equations on unstructured grids. *Communications in Computational Physics*, Volume 16 (2014), pp. 841-891.

Chapter 1

A Finite Volume scheme for solving anisotropic diffusion on unstructured grids

In this chapter, we describe a finite volume scheme to solve anisotropic diffusion equations on unstructured grids. This scheme called CCLAD for Cell-Centered LAgrangian Diffusion, was initially presented by Maire and Breil in [32] and [90]. It was developed to be used in the context of heat transfer within laser-heated plasma flows such as those obtained in the domain of direct drive Inertial Confinement Fusion [17]. We will show that this scheme is also well suited for other applications such as the numerical simulation of atmospheric reentry flows. Here, we propose not only a three-dimensional unstructured extension of this scheme but also a massively parallel implementation, which aims at dealing with real life applications. Moreover, we analyze the performances and the efficiency of this parallel algorithm.
We recall that we aim at solving the heat transfer occurring in the thermal protection system of a reentry vehicle. The thermal protection system consists of several distinct materials with discontinuous and possibly anisotropic conductivity tensors. Our scheme needs to be able to accurately take into account the interfaces between the different materials. Furthermore the geometry of such systems can by extremely complicated. These geometrical complexities can be efficiently treated by employing unstructured grids. This leads to the following requirements related to the diffusion scheme under consideration:

- It should be a finite volume scheme where the primary unknown, *i.e.*, the temperature is located at the cell center. Therefore the interfaces of the mesh cells match the interfaces of the materials.

- It should be a sufficiently accurate and robust scheme to cope with unstructured grids. Therefore we could handle complex geometries.

Before describing the main features of our finite volume scheme, let us briefly give an overview of the existing cell-centered diffusion scheme on unstructured grids. For a more detailed overview of the existing methods, the interested reader, should refer for instance to [89].
The simpler cell-centered finite volume is the so-called two-point flux approximation wherein the normal component of the heat flux at a cell interface is computed using the finite difference of the adjacent temperatures. It is well known that this method is consistent if and only if the computational grid is orthogonal with respect to the metrics induced by the symmetric positive definite conductivity tensor. This restriction renders this method inoperative for solving anisotropic diffusion problems on unstructured grids or distorted grids. It has motivated the

work of Aavatsmark and his co-authors to develop a class of finite volume schemes based on multi-point flux approximations (MPFA) for solving the elliptic flow equation encountered in the context of reservoir simulation, refer to [3, 4]. In this method, the flux is approximated by a multi-point expression based on transmissibility coefficients. These coefficients are computed using the point-wise continuity of the normal flux and the temperature across cell interfaces. The link between lowest-order mixed finite element and multi-point finite volume methods on simplicial meshes is investigated in [133]. The class of MPFA methods is characterized by cell-centered unknowns and a local stencil. The global diffusion matrix corresponding to this type of schemes on general 3D unstructured grids is in general non-symmetric. There are many variants of the MPFA methods which differ in the choices of geometrical points and control volumes employed to derive the multi-point flux approximation. For more details about this method and its properties, the interested reader might refer to [5, 54, 6, 98] and the references therein. It is also worth mentioning that the theoretical analysis of the MPFA O scheme for heterogeneous anisotropic diffusion problems on general meshes have been performed in [13]. In this paper, the introduction of an hybrid discrete variational formulation and of a sufficient local condition for coercivity, depending on the grid and on the conductivity tensor, allows to prove the convergence of the proposed numerical method.

The mimetic finite difference (MFD) methodology is an interesting alternative approach for solving anisotropic diffusion equations on general unstructured grids. This method mimics the essential underlying properties of the original continuum differential operators such as conservation laws, solution symmetries and the fundamental identities of vector and tensor calculus, refer to [114, 115, 73, 72, 84]. More precisely, the discrete flux operator is the negative adjoint of the discrete divergence in an inner scalar product weighted by the inverse of the conductivity tensor. The classical MFD methods employ one degree of freedom per element to approximate the temperature and one degree of freedom per mesh face to approximate the normal component of the heat flux. The continuity of temperature and of the normal component of the heat flux across cell interfaces allows to assemble a global linear system satisfied by face-based temperatures unknowns. The corresponding matrix is symmetric positive definite. This type of discretization is usually second-order accurate for the temperature unknown on unstructured polyhedral grids having degenerate and non convex polyhedra with flat faces [85]. In the case of grids with strongly curved faces the introduction of more than one flux per curved face is required to get the optimal convergence rate [33].

Another class of finite volume schemes for solving diffusion equations, with full tensor coefficients, on general grids has been initially proposed in [62] and generalized in [63]. This approach has been termed discrete duality finite volume (DDVF) [42] since it arises from the construction of discrete analogs of the divergence and flux operators which fulfill the discrete counterpart of vector calculus identities. The DDFV method requires to solve the diffusion equation not only over the primal grid but also over a dual grid. Namely, there are both cell-centered and vertex-centered unknowns. In addition, the construction of the dual grid in the case of a three-dimensional geometry is not unique. There are at least three different choices which lead to different variants of the three-dimensional DDFV schemes, see [64] and the references therein. The DDFV method described in [64] is characterized by a symmetric definite positive matrix and exhibits a numerical second-order accuracy for the temperature. Compared to a classical cell-centered finite volume scheme, this DDFV discretization necessitates twice as much degrees of freedom over hexahedral grids [65]. Let us point out that the use of such a method might be difficult in the perspective of solving coupled problems such as heat transfer and fluid flow.

The main feature of our finite volume scheme relies on the partition of each polyhedral cell of the computational domain into sub-cells and on the partition of each cell face into sub-faces, which are composed of triangular faces. There is one degree of freedom per element to approximate

the temperature unknown and one degree of freedom per sub-face to approximate the normal component of the heat flux across cell interfaces. For each cell, the sub-face normal fluxes impinging at a vertex are expressed with respect to the difference between sub-face temperatures and the cell-centered temperature. This approximation of the sub-face fluxes results from a local variational formulation written over each sub-cell. The sub-face temperatures, which are auxiliary unknowns, are locally eliminated by invoking the continuity of the temperature and the normal component of the heat flux across each cell interface. This elimination procedure of the sub-face temperatures in terms of the cell-centered temperatures surrounding a vertex is achieved by solving a local linear system of reasonable size at each vertex. Gathering the contribution of each vertex allows to construct easily the global sparse diffusion matrix. The node-based construction of our scheme provides a natural treatment of the boundary conditions. The scheme stencil is local and for a given cell consists of the cell itself and its node-based neighbors. Since the constitutive law of the heat flux has been approximated by means of a local variational formulation, the corresponding discrete diffusion operator inherits the positive definiteness property of the conductivity tensor. In addition, the semi-discrete version of the scheme is stable with respect to the discrete L^2 norm. For tetrahedral grids, the scheme preserves linear solutions with respect to the space variable and is characterized by a numerical second-order convergence rate for the temperature. For smooth distorted hexahedral grids its exhibits an accuracy which is almost of second-order. Let us point out that our formulation is similar to the local MFD discretization developed in [86] for simplicial grids.

The remainder of this chapter is organized as follows. In Section 1.1, we first give the problem statement introducing the governing equations, the notation and assumptions for deriving our finite volume scheme. This is followed by Section 1.2, which is devoted to the space discretization of the scheme in two dimension of space as presented in [89]. In fact this section recalls the space discretization of the original CCLAD scheme which will be the cornerstone of this chapter. In this section, we derive the sub-face fluxes approximation by means of a sub-cell-based variational formulation. We also describe the elimination of the sub-face temperatures in terms of the cell-centered unknowns to achieve the construction of the global discrete diffusion operator. In Section 1.3, we will show a first extension of the scheme that was developed in this thesis, namely the extension to three-dimensions of the scheme. This extension is not trivial, the geometry of the cells are more complex and many hypothesis used in the two dimensional schemes are not valid in three dimensions. We will show how to overcome these issues to build the scheme in three dimensions. Section 1.4 is devoted to the presentation of the main properties of the semi-discrete scheme and the boundary conditions implementation. The time discretization is briefly developed in Section 1.5. We describe an other extension developed in this thesis, the parallel implementation of the scheme, and its efficiency analysis in Section 1.6. Finally, the robustness and the accuracy of the scheme are assessed using various representative test cases in Section 1.8.

1.1 Governing equations

Our motivation is to describe a finite volume scheme that solves the anisotropic heat conduction equation on d-dimensional unstructured grids. Let us introduce the governing equations, notations and the assumptions required for the present work. Let $\mathcal{D}$ be an open set of the d-dimensional space $\mathfrak{R}^d$. Let $\boldsymbol{x}$ denotes the position vector of an arbitrary point inside the domain $\mathcal{D}$ and $t > 0$ the time. We aim at constructing a numerical scheme to solve the following

initial-boundary-value problem for the temperature $T = T(\boldsymbol{x}, t)$

$$\rho C_v \frac{\partial T}{\partial t} + \nabla \cdot \boldsymbol{q} = \rho r, \qquad (\boldsymbol{x}, t) \in \mathcal{D} \times [0, \mathfrak{T}] \tag{1.1a}$$

$$T(\boldsymbol{x}, 0) = T^0(\boldsymbol{x}), \qquad \boldsymbol{x} \in \mathcal{D} \tag{1.1b}$$

$$T(\boldsymbol{x}, t) = T^*(\boldsymbol{x}, t), \qquad \boldsymbol{x} \in \partial\mathcal{D}_D \tag{1.1c}$$

$$\boldsymbol{q}(\boldsymbol{x}, t) \cdot \boldsymbol{n} = q_N^*(\boldsymbol{x}, t), \qquad \boldsymbol{x} \in \partial\mathcal{D}_N \tag{1.1d}$$

$$\alpha T(\boldsymbol{x}, t) + \beta \boldsymbol{q}(\boldsymbol{x}, t) \cdot \boldsymbol{n} = q_R^*(\boldsymbol{x}, t). \qquad \boldsymbol{x} \in \partial\mathcal{D}_R \tag{1.1e}$$

Here, $\mathfrak{T} > 0$ denotes the final time, ρ is a positive real valued function, which stands for the mass density of the material. The source term, r, corresponds to the specific heat supplied to the material and C_v denotes the heat capacity at constant volume. We assume that ρ, C_v, and r are known functions. The initial condition is characterized by the initial temperature field T^0. Three types of boundary conditions are considered: Dirichlet, Neumann and Robin boundary conditions. They consist in specifying respectively the temperature, the flux and a combination of them. We introduce the partition $\partial\mathcal{D} = \partial\mathcal{D}_D \cup \partial\mathcal{D}_N \cup \partial\mathcal{D}_R$ of the boundary domain. Here, T^* and q_N^* denote respectively the prescribed temperature and flux for the Dirichlet and Neumann boundary conditions, whereas α, β and q_R^* are the parameters of the Robin boundary condition. The vector $\boldsymbol{q}$ denotes the heat flux and $\boldsymbol{n}$ is the outward unit normal to the boundary of the domain $\mathcal{D}$.

Eq. (1.1a) is a parabolic partial differential equation for the temperature T, where the conductive flux, $\boldsymbol{q}$, is defined according to the Fourier law

$$\boldsymbol{q} = -\mathbb{K}\nabla T. \tag{1.2}$$

The second-order tensor, $\mathbb{K}$, is the conductivity tensor. It is an intrinsic property of the material under consideration. We suppose that $\mathbb{K}$ is positive definite to ensure the model consistency with the Second Law of thermodynamics, i.e. $\boldsymbol{q} \cdot \nabla T \leq 0$. Namely, this property ensures that heat flux direction is opposite to temperature gradient. Let us point out that in the problems we are considering the conductivity tensor is always symmetric positive definite, *i.e.*, $\mathbb{K} = \mathbb{K}^t$, where the superscript t denotes transpose.

Comment 1: *The normal component of the heat flux at the interface between two distinct materials, labelled by 1 and 2, is continuous, that is*

$$(\mathbb{K}\nabla T)_1 \cdot \boldsymbol{n}_{12} = (\mathbb{K}\nabla T)_2 \cdot \boldsymbol{n}_{12},$$

where $\boldsymbol{n}_{12}$ is the unit normal to the interface. The temperature itself is also continuous.

In the next two sections we make the distinction between the two dimensional version and the three dimensional version of the scheme. In the following section we recall the two-dimensional version of the CCLAD scheme as originally presented by Breil and Maire in [90]. This section introduces the methodology and the key concepts behind the construction of most of the numerical schemes that will be used in this thesis.

1.2 Space discretization in two dimensions

1.2.1 Notations and assumptions

Having defined the problem we want to solve, let us introduce some notation necessary to develop the discretization scheme in two dimensions of space. Let $\cup_c \omega_c$ denotes a partition of the

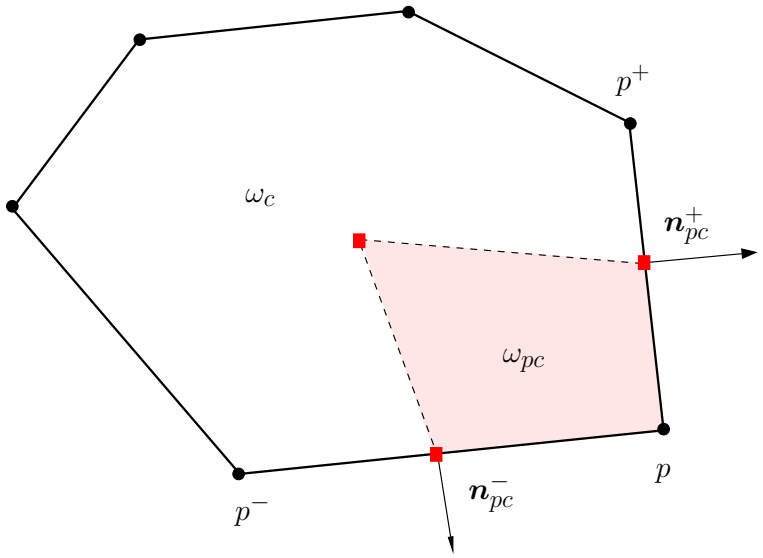


Figure 1.1: Notation related to polygonal cell ω_c and one of its sub-cell ω_{pc}.

computational domain $\mathcal{D}$ into polygonal cells ω_c. The counterclockwise ordered list of vertices (points) of cell c is denoted by $\mathcal{P}(c)$. In addition, p being a generic point, we define its position vector denoted as $\boldsymbol{x}_p$ and the set $\mathcal{C}(p)$ which contains all the cells surrounding point p. Being given $p \in \mathcal{P}(c)$, p^- and p^+ are the previous and next points with respect to p in the ordered list of vertices of cell c. Let ω_c be a generic polygonal cell, for each vertex $p \in \mathcal{P}(c)$, we define the sub-cell ω_{pc} by connecting the centroid of ω_c to the midpoints of edges $[p^-, p]$ and $[p, p^+]$ impinging at node p, refer to Figure 1.1. In two dimensions the sub-cell, as just defined, is always a quadrilateral regardless of the type of cells that compose the underlying grid. The boundaries of the cell ω_c and the sub-cell ω_{pc} are denoted respectively $\partial\omega_c$ and $\partial\omega_{pc}$. Finally, considering the intersection between the cell and sub-cell boundaries, we introduce half-edge geometric data. As the name implies, a half-edge is a half of an edge and is constructed by splitting an edge down its length. More precisely, we define the two half-edges related to point p and cell c as $\partial\omega_{pc}^- = \partial\omega_{pc} \cap [p^-, p]$ and $\partial\omega_{pc}^+ = \partial\omega_{pc} \cap [p, p^+]$. The unit outward normal and the length related to half-edge $\partial\omega_{pc}^\pm$ are denoted respectively $\boldsymbol{n}_{pc}^\pm$ and $l_{pc}^\pm$.

To proceed with the construction of numerical scheme, let us integrate (1.1) over ω_c and make use of the divergence formula. This leads to the weak form of the heat conduction equation

$$\frac{d}{dt}\int_{\omega_c} \rho C_v T(\boldsymbol{x},t)\,\mathrm{d}v + \int_{\partial\omega_c} \boldsymbol{q}\cdot\boldsymbol{n}\,\mathrm{d}s = \int_{\omega_c} \rho r(\boldsymbol{x},t)\mathrm{d}v, \tag{1.3}$$

where $\boldsymbol{n}$ denotes the unit outward normal to $\partial\omega_c$. We shall first discretize this equation with respect to the spatial variable $\boldsymbol{x}$. The physical data, ρ, C_v and r are supposed to be known functions with respect to space and time variables. We represent them using a piecewise constant approximation over each cell ω_c. The piecewise constant approximation of any variable will be denoted using subscript c. The tensor conductivity $\mathbb{K}$ space approximation is also constructed using a piecewise constant representation over each cell, which is denoted by $\mathbb{K}_c$. Concerning the unknown temperature field, the discretization method we are going to use is the finite volume method for which the finite dimensional space to which the approximate solution belongs is also the space of piecewise constant functions. Bearing this in mind, (1.3) rewrites

$$m_c C_{vc}\frac{d}{dt}T_c + \int_{\partial\omega_c} \boldsymbol{q}\cdot\boldsymbol{n}\,\mathrm{d}s = m_c r_c, \tag{1.4}$$

Here, m_c denotes the mass of the cell, that is, $m_c = \rho_c \mid \omega_c \mid$ where $\mid \omega_c \mid$ stands for the volume of the cell. Let us point out that $T_c = T_c(t)$ is nothing but the mean value of the temperature over ω_c

$$T_c(t) = \frac{1}{\mid \omega_c \mid} \int_{\omega_c} T(\boldsymbol{x}, t) \, \mathrm{d}v.$$

To define completely the space discretization it remains to discretize the surface integral in the above equation. To do so, let us introduce the following piecewise constant approximation of the normal heat flux over each half-edges

$$q_{pc}^{\pm} = \frac{1}{l_{pc}^{\pm}} \int_{\partial\omega_{pc}^{\pm}} \boldsymbol{q} \cdot \boldsymbol{n} \, \mathrm{d}s. \tag{1.5}$$

The scalar $q_{pc}^{\pm}$ stands for the half-edge normal flux related to the half-edge $\partial\omega_{pc}^{\pm}$. Knowing that $\partial\omega_c = \cup_{p\in\mathcal{P}(c)} \partial\omega_{pc}^{\pm}$, the semi-discretized heat conduction equation writes as

$$m_c C_{vc} \frac{d}{dt} T_c + \sum_{p\in\mathcal{P}(c)} l_{pc}^{-} q_{pc}^{-} + l_{pc}^{+} q_{pc}^{+} = m_c r_c. \tag{1.6}$$

We conclude this paragraph by introducing as auxiliary unknowns the half-edge temperatures $T_{pc}^{\pm}$ defined by

$$T_{pc}^{\pm} = \frac{1}{l_{pc}^{\pm}} \int_{\partial\omega_{pc}^{\pm}} T(\boldsymbol{x}, t) \, \mathrm{d}s. \tag{1.7}$$

In this equation, we have also assumed a piecewise constant approximation of the temperature field over each half-edge. These half-edge temperatures will be useful in constructing the numerical approximation of the heat flux.

Thanks to Comment 1, the piecewise constant approximations of the normal heat flux and temperature along each edge are defined such that these half-edge-based quantities are continuous across each edge. To exhibit these continuity conditions, let us consider two neighboring cells, denoted by subscripts c and d, which share a given edge, refer to Figure 1.2. This edge corresponds to the segment $[p, p^+]$, where p and p^+ are two consecutive points in the counterclockwise numbering attached to cell c. It also corresponds to the segment $[r^-, r]$, where r^- and r are two consecutive points in the counterclockwise numbering attached to cell d. Obviously, these four labels define the same edge and thus their corresponding points coincide, *i.e.*, $p \equiv r$, $p^+ \equiv r^-$. The sub-cell of cell c attached to point $p \equiv r$ is denoted ω_{pc}, whereas the sub-cell of cell d attached to point $r \equiv p$ is denoted ω_{rd}. This double notation, allows to define precisely the half-edge fluxes and temperatures at the half-edge corresponding to the intersection of the two previous sub-cells. Namely, viewed from sub-cell ω_{pc} (resp. ω_{rd}), the half-edge flux and temperature are denoted q_{pc}^{+} and T_{pc}^{+} (resp. q_{rd}^{-} and T_{rd}^{-}). Bearing this notation in mind, continuity conditions at the half-edge $(\omega_{pc} \cup \omega_{rd})$ for the half-edge fluxes and temperatures write explicitly as

$$q_{pc}^{+} + q_{rd}^{-} = 0, \tag{1.8a}$$

$$T_{pc}^{+} = T_{rd}^{-}. \tag{1.8b}$$

The continuity condition for the heat flux follows from the definition of the unit outward normals related to $\omega_{pc} \cup \omega_{rd}$, *i.e.*, $\boldsymbol{n}_{pc}^{+} = -\boldsymbol{n}_{rd}^{-}$.

To achieve the space discretization of (1.6), it remains to construct a consistent approxi-

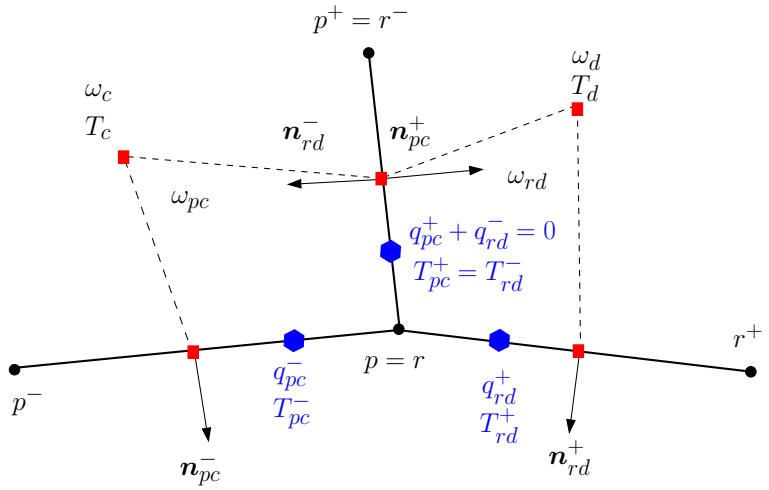


Figure 1.2: Continuity conditions for the half-edge fluxes and temperatures at a half-edge shared by two sub-cells attached to the same point. Labels c and d denote the indices of two neighboring cells. Labels p and r denote the indices of the same point relatively to the local numbering of points in cell c and d. The neighboring sub-cells are denoted by ω_{pc} and ω_{rd}. The half-edge fluxes, $q_{pc}^{\pm}$, $q_{rd}^{\pm}$ and temperatures, $T_{cp}^{\pm}$, $T_{rd}^{\pm}$ are displayed using blue color.

mation of the half-edge normal flux, that is, to define a numerical half-edge-based flux function $h_{pc}^{\pm}$ such that

$$q_{pc}^- = h_{pc}^-(T_{pc}^- - T_c, T_{pc}^+ - T_c), \quad q_{pc}^+ = h_{pc}^+(T_{pc}^- - T_c, T_{pc}^+ - T_c). \tag{1.9}$$

Here, $h_{pc}^{\pm}$ denotes a real valued function which is continuous with respect to its arguments. Let us note that we have expressed this function in terms of the temperature difference $T_c - T_{pc}^{\pm}$ since the heat flux is proportional to the temperature gradient. The next steps in the design of our finite volume scheme are the following:

- Construction of the half-edge numerical fluxes by means of a **local variational formulation over the sub-cell**.
- Elimination of the half-edge temperatures through the use of the **continuity condition** (1.8) **across sub-cell interface**.

These tasks are the main topics of the next section.

Before proceeding any further, we start by giving a useful and classical result concerning the representation of a vector in terms of its normal components. This result leads to the expression of the standard inner product of two vectors, which will be one the tools utilized to derive the sub-cell variational formulation. Here, we recall briefly the methodology which has been thoroughly exposed by Shashkov in [113, 93].

1.2.2 Expression of a vector in terms of its normal components

Let $\boldsymbol{\phi}$ be an arbitrary vector of the two-dimensional space $\Re^2$ and $\boldsymbol{\phi}_{pc}$ its piecewise constant approximation over the sub-cell ω_{pc}. Let $\phi_{pc}^{\pm}$ be the half-edge normal components of $\boldsymbol{\phi}_{pc}$, that is,

$$\boldsymbol{\phi}_{pc} \cdot \boldsymbol{n}_{pc}^- = \phi_{pc}^-,$$
$$\boldsymbol{\phi}_{pc} \cdot \boldsymbol{n}_{pc}^+ = \phi_{pc}^+.$$

Introducing the corner matrix $\mathbb{J}_{pc} = [\boldsymbol{n}_{pc}^-, \boldsymbol{n}_{pc}^+]$, the above 2×2 linear system rewrites

$$\mathbb{J}_{pc}^t \boldsymbol{\phi} = \begin{pmatrix} \phi_{pc}^- \\ \phi_{pc}^+ \end{pmatrix}.$$

Provided that $\boldsymbol{n}_{pc}^-$ and $\boldsymbol{n}_{pc}^+$ are not collinear, the above system has always a unique solution written under the form

$$\boldsymbol{\phi}_{pc} = \mathbb{J}_{pc}^{-t} \begin{pmatrix} \phi_{pc}^- \\ \phi_{pc}^+ \end{pmatrix}. \tag{1.10}$$

This equation allows to express any vector in terms of its normal components on two non-collinear unit vectors. This representation allows to compute the inner product of two vectors $\boldsymbol{\phi}_{pc}$ and $\boldsymbol{\psi}_{pc}$ as follows

$$\boldsymbol{\phi}_{pc} \cdot \boldsymbol{\psi}_{pc} = (\mathbb{J}_{pc}^t \mathbb{J}_{pc})^{-1} \begin{pmatrix} \psi_{pc}^- \\ \psi_{pc}^+ \end{pmatrix} \cdot \begin{pmatrix} \phi_{pc}^- \\ \phi_{pc}^+ \end{pmatrix}. \tag{1.11}$$

The 2×2 matrix $\mathsf{H}_{pc} = \mathbb{J}_{pc}^t \mathbb{J}_{pc}$ writes

$$\mathsf{H}_{pc} = \begin{pmatrix} \boldsymbol{n}_{pc}^- \cdot \boldsymbol{n}_{pc}^- & \boldsymbol{n}_{pc}^- \cdot \boldsymbol{n}_{pc}^+ \\ \boldsymbol{n}_{pc}^+ \cdot \boldsymbol{n}_{pc}^- & \boldsymbol{n}_{pc}^+ \cdot \boldsymbol{n}_{pc}^+ \end{pmatrix} = \begin{pmatrix} 1 & -\cos\theta_{pc} \\ -\cos\theta_{pc} & 1 \end{pmatrix}, \tag{1.12}$$

where θ_{pc} denotes the measure of the angle between the two half-edges of sub-cell ω_{pc} impinging at point p, refer to Figure 1.3. This matrix admits an inverse provided that $\theta_{pc} \neq 2k\pi$, where k is an integer. It means that we can not deal with cells having straight angles. Under this condition, H_{pc}^{-1} writes

$$\mathsf{H}_{pc}^{-1} = \frac{1}{\sin^2\theta_{pc}} \begin{pmatrix} 1 & \cos\theta_{pc} \\ \cos\theta_{pc} & 1 \end{pmatrix}.$$

This matrix, which is symmetric definite positive, represents the local metric tensor associated to the sub-cell ω_{pc}. Let us remark that we have recovered exactly the expressions initially derived in [93].

1.2.3 Sub-cell-based variational formulation

We construct an approximation of the half-edge fluxes by means of a local variational formulation written over the sub-cell ω_{pc}. Contrary to the classical cell-based variational formulation used in the context of Mimetic Finite Difference Method [72, 93, 85], the present sub-cell-based variational formulation leads to a local **explicit expression of the half-edge fluxes** in terms of the half-edge temperatures and the mean cell temperature. The local and explicit feature of the half-edge fluxes expression is of great importance, since it allows to construct a numerical scheme with only one unknown per cell.

Our starting point to derive the sub-cell based variational formulation consists in writing the partial differential equation satisfied by the heat flux. From the heat flux definition (1.2), it follows that $\boldsymbol{q}$ satisfies

$$\mathbb{K}^{-1}\boldsymbol{q} + \nabla T = \boldsymbol{0}. \tag{1.13}$$

Let us point out that the present approach is strongly linked to the mixed formulation utilized in the context of mixed finite element discretization [7, 86, 119]. Dot-multiplying the above equation by an arbitrary vector $\boldsymbol{\phi} \in \mathcal{D}$ and integrating over the cell ω_{pc} yields

$$\int_{\omega_{pc}} \boldsymbol{\phi} \cdot \mathbb{K}^{-1}\boldsymbol{q} \, \mathrm{d}v = -\int_{\omega_{pc}} \boldsymbol{\phi} \cdot \nabla T \, \mathrm{d}v, \quad \forall \boldsymbol{\phi} \in \mathcal{D}. \tag{1.14}$$

Integrating by part the right-hand side and applying the divergence formula lead to the following variational formulation

$$\int_{\omega_{pc}} \boldsymbol{\phi} \cdot \mathbb{K}^{-1}\boldsymbol{q}\,\mathrm{d}v = \int_{\omega_{pc}} T\nabla\cdot\boldsymbol{\phi}\,\mathrm{d}v - \int_{\partial\omega_{pc}} T\boldsymbol{\phi}\cdot\boldsymbol{n}\,\mathrm{d}s, \quad \forall\boldsymbol{\phi}\in\mathcal{D}. \tag{1.15}$$

This sub-cell-based variational formulation is the cornerstone to construct a local and explicit expression of the half-edge fluxes. Replacing T by its piecewise constant approximation, T_c, in the first integral of the right-hand side and applying the divergence formula to the remaining volume integral leads to

$$\int_{\omega_{pc}} \boldsymbol{\phi} \cdot \mathbb{K}^{-1}\boldsymbol{q}\,\mathrm{d}v = T_c\int_{\partial\omega_{pc}} \boldsymbol{\phi}\cdot\boldsymbol{n}\,\mathrm{d}s - \int_{\partial\omega_{pc}} T\boldsymbol{\phi}\cdot\boldsymbol{n}\,\mathrm{d}s, \quad \forall\boldsymbol{\phi}\in\mathcal{D}. \tag{1.16}$$

Observing that the sub-cell boundary, $\partial\omega_{pc}$, decomposes into the inner part $\underline{\partial\omega_{pc}} = \partial\omega_{pc}\cap\omega_c$ and the outer part $\overline{\partial\omega_{pc}} = \partial\omega_{pc}\cap\partial\omega_c$ allows to split the surface integrals of the right-hand side of (1.16) as follows

$$\int_{\omega_{pc}} \boldsymbol{\phi} \cdot \mathbb{K}^{-1}\boldsymbol{q}\,\mathrm{d}v = T_c\int_{\overline{\partial\omega_{pc}}} \boldsymbol{\phi}\cdot\boldsymbol{n}\,\mathrm{d}s + T_c\int_{\underline{\partial\omega_{pc}}} \boldsymbol{\phi}\cdot\boldsymbol{n}\,\mathrm{d}s - \int_{\overline{\partial\omega_{pc}}} T\boldsymbol{\phi}\cdot\boldsymbol{n}\,\mathrm{d}s - \int_{\underline{\partial\omega_{pc}}} T\boldsymbol{\phi}\cdot\boldsymbol{n}\,\mathrm{d}s. \tag{1.17}$$

We replace T by its piecewise constant approximation, T_c, in the fourth surface integral of the right-hand side, then noticing that the second integral is equal to the fourth one, transforms Eq. (1.17) into

$$\int_{\omega_{pc}} \boldsymbol{\phi} \cdot \mathbb{K}^{-1}\boldsymbol{q}\,\mathrm{d}v = T_c\int_{\overline{\partial\omega_{pc}}} \boldsymbol{\phi}\cdot\boldsymbol{n}\,\mathrm{d}s - \int_{\overline{\partial\omega_{pc}}} T\boldsymbol{\phi}\cdot\boldsymbol{n}\,\mathrm{d}s. \tag{1.18}$$

Comment 2: *A this point it is interesting to remark that the above sub-cell-based formulation is a sufficient condition to recover the classical cell-based variational formulation. This is due to the fact that the set of sub-cells is a partition of the cell, i.e.,*

$$\omega_c = \bigcup_{p\in\mathcal{P}(c)} \omega_{pc}, \quad \textit{and} \quad \partial\omega_c = \bigcup_{p\in\mathcal{P}(c)} \overline{\partial\omega_{pc}}.$$

Thus, summing (1.18) *over all the sub-cells of* ω_c *leads to*

$$\int_{\omega_c} \boldsymbol{\phi} \cdot \mathbb{K}^{-1}\boldsymbol{q}\,\mathrm{d}v = T_c\int_{\partial\omega_c} \boldsymbol{\phi}\cdot\boldsymbol{n}\,\mathrm{d}s - \int_{\partial\omega_c} T\boldsymbol{\phi}\cdot\boldsymbol{n}\,\mathrm{d}s.$$

This last equation corresponds to the cell-based variational formulation of the partial differential equation (1.13)*. This form is used in the context of Mimetic Finite Difference Method [72] to obtain a discretization of the heat flux. More precisely, it leads to a linear system satisfied by the half-edge fluxes. This results in a non explicit expression of the half-edge flux with respect to the half-edge temperatures and the cell-centered temperature [85], which leads to a finite volume discretization characterized by face-based and cell-based unknowns. In contrast to this approach, the sub-cell-based variational formulation yields a finite volume discretization with one unknown per cell.*

Returning to the sub-cell based variational formulation, we discretize the right-hand side of (1.18) by introducing the half-edge normal components of $\boldsymbol{\phi}$ and the piecewise constant approximation of the half-edge temperatures as follows

$$\int_{\omega_{pc}} \boldsymbol{\phi} \cdot \mathbb{K}^{-1}\boldsymbol{q}\,\mathrm{d}v = -[l^-_{pc}(T^-_{pc}-T_c)\phi^-_{pc} + l^+_{pc}(T^+_{pc}-T_c)\phi^+_{pc}]. \tag{1.19}$$

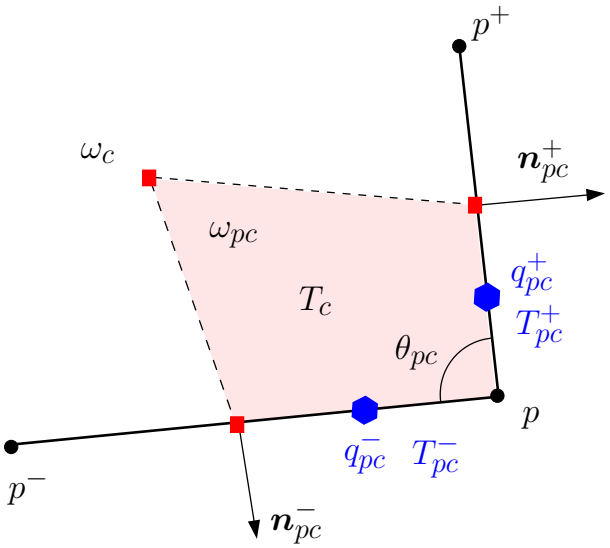


Figure 1.3: Fragment of a polygonal cell ω_c. Notation for the sub-cell ω_{pc}: The half-edge fluxes, $q_{pc}^{\pm}$, and temperatures, $T_{pc}^{\pm}$ are displayed using blue color.

Assuming a piecewise constant representation of the test function allows to compute the volume integral in the left-hand side thanks to the quadrature rule

$$\int_{\omega_{pc}} \boldsymbol{\phi} \cdot \mathbb{K}^{-1} \boldsymbol{q} \, \mathrm{d}v = w_{pc} \boldsymbol{\phi}_{pc} \cdot \mathbb{K}_c^{-1} \boldsymbol{q}_{pc}, \tag{1.20}$$

Here, $\mathbb{K}_c$ denotes the piecewise constant approximation of the conductivity tensor and $\boldsymbol{\phi}_{pc}$, $\boldsymbol{q}_{pc}$ are the piecewise constant approximations of vectors $\boldsymbol{\phi}$ and $\boldsymbol{q}$, refer to Figure 1.3. In addition, w_{pc} denotes some positive corner volume related to sub-cell ω_{pc}, which will be determined later.

Comment 3: *Note that the corner volumes associated to the same cell ω_c must satisfy the consistency condition*

$$\sum_{p \in \mathcal{P}(c)} w_{pc} = \mid \omega_c \mid . \tag{1.21}$$

Namely, the corner volumes of a cell sums to the volume of the cell. This is the minimal requirement to ensure that constant functions are exactly integrated using the above quadrature rule.

Now, combining (1.20) and (1.19) and using the expression of the vectors $\boldsymbol{q}$ and $\boldsymbol{\phi}$ in terms of their half-edge normal components leads to the following variational formulation

$$w_{pc} (\mathbb{J}_{pc}^t \mathbb{K}_c \mathbb{J}_{pc})^{-1} \begin{pmatrix} q_{pc}^- \\ q_{pc}^+ \end{pmatrix} \cdot \begin{pmatrix} \phi_{pc}^- \\ \phi_{pc}^+ \end{pmatrix} = - \begin{bmatrix} l_{pc}^-(T_{pc}^- - T_c) \\ l_{pc}^+(T_{pc}^+ - T_c) \end{bmatrix} \cdot \begin{pmatrix} \phi_{pc}^- \\ \phi_{pc}^+ \end{pmatrix} . \tag{1.22}$$

Knowing that this variational formulation must hold for any vector $\boldsymbol{\phi}_{pc}$, this implies

$$\begin{pmatrix} q_{pc}^- \\ q_{pc}^+ \end{pmatrix} = -\frac{1}{w_{pc}} (\mathbb{J}_{pc}^t \mathbb{K}_c \mathbb{J}_{pc}) \begin{bmatrix} l_{pc}^-(T_{pc}^- - T_c) \\ l_{pc}^+(T_{pc}^+ - T_c) \end{bmatrix} . \tag{1.23}$$

This equation constitutes the approximation of the half-edge normal fluxes over a sub-cell. This local approximation is coherent with expression of the constitutive law (1.2) in the sense that the numerical approximation of the heat flux is equal to a tensor times a numerical approximation

of the temperature gradient. This tensor can be viewed as an effective conductivity tensor associated to the sub-cell ω_{pc}. Thus, it is natural to set

$$\mathbb{K}_{pc} = \mathbb{J}_{pc}^t \mathbb{K}_c \mathbb{J}_{pc}. \tag{1.24}$$

Let us emphasize that this corner tensor inherits all the properties of the conductivity tensor $\mathbb{K}_c$. Namely, $\mathbb{K}_c$ being positive definite, $\mathbb{K}_{pc}$ is also positive definite. This comes from the fact that

$$\mathbb{K}_{pc}\boldsymbol{\phi} \cdot \boldsymbol{\phi} = \mathbb{K}_c(\mathbb{J}_{pc}\boldsymbol{\phi}) \cdot (\mathbb{J}_{pc}\boldsymbol{\phi}), \quad \forall \boldsymbol{\phi} \in \mathfrak{R}^2.$$

Using a similar argument, note that if $\mathbb{K}_c$ is symmetric, $\mathbb{K}_{pc}$ is also symmetric. Recalling that $\mathbb{J}_{pc} = [\boldsymbol{n}_{pc}^-, \boldsymbol{n}_{pc}^+]$, we readily obtain the expression of the corner tensor $\mathbb{K}_{pc}$ in terms of the unit normal $\boldsymbol{n}_{pc}^{\pm}$

$$\mathbb{K}_{pc} = \begin{pmatrix} \mathbb{K}_c\boldsymbol{n}_{pc}^- \cdot \boldsymbol{n}_{pc}^- & \mathbb{K}_c\boldsymbol{n}_{pc}^+ \cdot \boldsymbol{n}_{pc}^- \\ \mathbb{K}_c\boldsymbol{n}_{pc}^- \cdot \boldsymbol{n}_{pc}^+ & \mathbb{K}_c\boldsymbol{n}_{pc}^+ \cdot \boldsymbol{n}_{pc}^+ \end{pmatrix}. \tag{1.25}$$

Let us remark that in the isotropic case, *i.e*, $\mathbb{K}_c = \kappa_c \mathsf{I}_d$, the corner tensor collapses to

$$\mathbb{K}_{pc} = \kappa_c \mathsf{H}_{pc}, \tag{1.26}$$

where κ_c denotes the piecewise constant scalar conductivity over cell ω_c and H_{pc} is the second-order tensor defined by (1.12).

We conclude by claiming that a sub-cell-based variational formulation has allowed to construct the following numerical approximation of the half-edge normal fluxes

$$\begin{pmatrix} q_{pc}^- \\ q_{pc}^+ \end{pmatrix} = -\frac{1}{w_{pc}} \mathbb{K}_{pc} \begin{bmatrix} l_{pc}^-(T_{pc}^- - T_c) \\ l_{pc}^+(T_{pc}^+ - T_c) \end{bmatrix}. \tag{1.27}$$

Here, w_{pc} is a positive volume weight, which will be determined later, and the corner conductivity tensor, $\mathbb{K}_{pc}$ is expressed by (1.25).

Comment 4: *It is interesting to remark that the corner tensor $\mathbb{K}_{pc}$ is a linear function with respect to the piecewise constant approximation of the conductivity tensor $\mathbb{K}_c$. This follows directly from* (1.24)*. In addition, the corner tensor corresponding to the transpose of $\mathbb{K}_c$ is the transpose of $\mathbb{K}_{pc}$, i.e., $\mathbb{K}_{pc}(\mathbb{K}_c^t) = \mathbb{K}_{pc}^t(\mathbb{K}_c)$.*

1.2.4 Elimination of the half-edge temperatures

From (1.27), it appears that the numerical approximation of the half-edge fluxes at a corner depends on the difference between the mean cell temperature and the half-edge temperatures. The mean cell temperature is the primary unknown whereas the half-edge temperatures are auxiliary unknowns, which can be eliminated by means of continuity argument (1.8a). Namely, we use the fact that the half-edge normal fluxes are continuous across each half-edges impinging at a given point. This local elimination procedure, which will be described below, yields a linear system satisfied by the half-edge temperatures. We will show that this system admits always a unique solution which allows to express the half-edge temperatures in terms of the mean temperatures of the cells surrounding the point under consideration. Therefore, this local elimination procedure results in a finite volume discrete scheme with one unknown per cell.

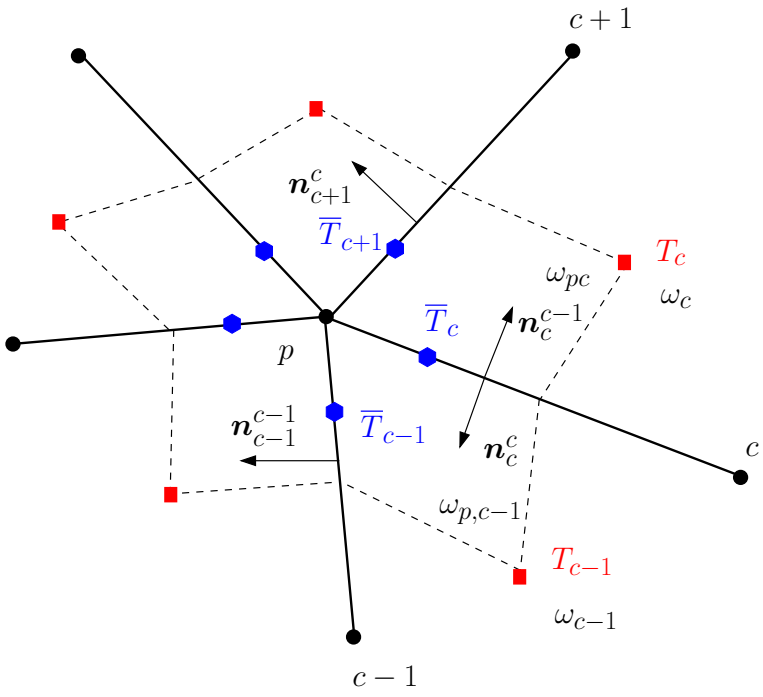


Figure 1.4: Notation for sub-cells surrounding point p.

Local notation around a point

To derive the local elimination procedure, we introduce some convenient notation. Let p denotes a generic point which is not located on the boundary $\partial\mathcal{D}$. The treatment of boundary points is postponed to Section 1.4.7, which is devoted to boundary conditions implementation. Let $\mathcal{C}(p)$ be the set of cells that surround point p. The edges impinging at point p are labelled using the subscript c ranging from 1 to $\mathfrak{C}_p$, where $\mathfrak{C}_p$ denotes the total number of cells surrounding point p. The cell (sub-cell) numbering follows the edge numbering, that is, cell ω_c (sub-cell ω_{pc}) is located between edges c and $c+1$, refer to Figure 1.4. The unit outward normal to cell ω_c at edge c is denoted by $\boldsymbol{n}_c^c$ whereas the unit outward normal to cell ω_c at edge $c+1$ is denoted by $\boldsymbol{n}_c^{c+1}$. Assuming the continuity of the half-edge temperatures leads to denote by $\overline{T}_c$ the unique half-edge temperature of the half-edge c impinging at point p. Note that we have omitted the dependency on point p in the indexing each time this is possible to avoid too heavy notion. With this notation, the expression of the half-edge fluxes (1.27) turns into

$$\begin{pmatrix} q_c^c \\ q_{c+1}^c \end{pmatrix} = -\frac{1}{w_{pc}} \mathbb{K}_{pc} \begin{bmatrix} l_c(\overline{T}_c - T_c) \\ l_{c+1}(\overline{T}_{c+1} - T_c) \end{bmatrix}, \quad \forall c \in \mathcal{C}(p). \tag{1.28}$$

Here, q_c^c (resp. q_{c+1}^c) denotes the half-edge normal flux at edge c (resp. $c+1$) viewed from cell c. In addition l_c denotes the half of the length of edge c. In these equations, we assume a periodic numbering around the point p. According to (1.25), the sub-cell conductivity tensor is defined as

$$\mathbb{K}_{pc} = \begin{pmatrix} \mathbb{K}_c \boldsymbol{n}_c^c \cdot \boldsymbol{n}_c^c & \mathbb{K}_c \boldsymbol{n}_{c+1}^c \cdot \boldsymbol{n}_c^c \\ \mathbb{K}_c \boldsymbol{n}_c^c \cdot \boldsymbol{n}_{c+1}^c & \mathbb{K}_c \boldsymbol{n}_{c+1}^c \cdot \boldsymbol{n}_{c+1}^c \end{pmatrix}, \quad \forall c \in \mathcal{C}(p), \tag{1.29}$$

where $\mathbb{K}_c$ is the piecewise constant approximation of the conductivity tensor in cell c. Combining (1.28) and (1.29) yields the explicit expressions

$$q_c^c = -\alpha_c [l_c(\mathbb{K}_c \boldsymbol{n}_c^c \cdot \boldsymbol{n}_c^c)(\overline{T}_c - T_c) + l_{c+1}(\mathbb{K}_c \boldsymbol{n}_{c+1}^c \cdot \boldsymbol{n}_c^c)(\overline{T}_{c+1} - T_c)], \tag{1.30a}$$

$$q_{c+1}^c = -\alpha_c [l_c(\mathbb{K}_c \boldsymbol{n}_c^c \cdot \boldsymbol{n}_{c+1}^c)(\overline{T}_c - T_c) + l_{c+1}(\mathbb{K}_c \boldsymbol{n}_{c+1}^c \cdot \boldsymbol{n}_{c+1}^c)(\overline{T}_{c+1} - T_c)], \tag{1.30b}$$

where we have introduced the inverse of the volume weight setting $\alpha_c = \frac{1}{w_{pc}}$. Shifting index c, *i.e.*, $c \to c-1$, in (1.30b) leads to the following expression for the half-edge normal flux at edge c viewed from cell $c-1$

$$q_c^{c-1} = -\alpha_{c-1}[l_{c-1}(\mathbb{K}_{c-1}\boldsymbol{n}_{c-1}^{c-1}\cdot\boldsymbol{n}_c^{c-1})(\overline{T}_{c-1}-T_{c-1}) + l_c(\mathbb{K}_{c-1}\boldsymbol{n}_c^{c-1}\cdot\boldsymbol{n}_c^{c-1})(\overline{T}_c - T_{c-1})]. \quad (1.31)$$

Linear system satisfied by the half-edge temperatures

Bearing this in mind, we are now in position to proceed with the elimination of the half-edge temperatures by writing the continuity of the half-edge normal fluxes at each edge c. This continuity condition at edge c reads as

$$l_c q_c^{c-1} + l_c q_c^c = 0, \quad \forall c \in \mathcal{C}(p). \quad (1.32)$$

Let us remark that this continuity condition provides $\mathfrak{C}_p$ equations for the $\mathfrak{C}_p$ auxiliary unknowns $\overline{T}_c$. Substituting (1.31) and (1.30a) into the continuity condition yields

$$\begin{aligned}&\alpha_{c-1}l_{c-1}l_c(\mathbb{K}_{c-1}\boldsymbol{n}_{c-1}^{c-1}\cdot\boldsymbol{n}_c^{c-1})\overline{T}_{c-1} + [\alpha_{c-1}l_c^2(\mathbb{K}_{c-1}\boldsymbol{n}_c^{c-1}\cdot\boldsymbol{n}_c^{c-1}) + \alpha_c l_c^2(\mathbb{K}_c\boldsymbol{n}_c^c\cdot\boldsymbol{n}_c^c)]\overline{T}_c + \alpha_c l_c l_{c+1}(\mathbb{K}_c\boldsymbol{n}_{c+1}^c\cdot\boldsymbol{n}_c^c)\overline{T}_{c+1}\\ &= \alpha_{c-1}l_c[l_{c-1}(\mathbb{K}_{c-1}\boldsymbol{n}_{c-1}^{c-1}\cdot\boldsymbol{n}_c^{c-1}) + l_c(\mathbb{K}_{c-1}\boldsymbol{n}_c^{c-1}\cdot\boldsymbol{n}_c^{c-1})]T_{c-1} + \alpha_c l_c[l_c(\mathbb{K}_c\boldsymbol{n}_c^c\cdot\boldsymbol{n}_c^c) + l_{c+1}(\mathbb{K}_c\boldsymbol{n}_{c+1}^c\cdot\boldsymbol{n}_c^c)]T_c.\end{aligned}$$

To write this equation under a more concise form, let us introduce $\boldsymbol{T} = (T_1, \ldots, T_{\mathfrak{C}_p})^t$ as the vector of the cell-centered temperatures around point p and $\overline{\boldsymbol{T}} = (\overline{T}_1, \ldots, \overline{T}_{\mathfrak{C}_p})^t$ as the vector of the half-edge temperatures around point p. The continuity condition (1.32) amounts to write that $\overline{\boldsymbol{T}}$ satisfies the following $\mathfrak{C}_p \times \mathfrak{C}_p$ linear system

$$\mathbb{N}\overline{\boldsymbol{T}} = \mathbb{S}\boldsymbol{T}. \quad (1.33)$$

Let us remark that $\mathbb{N}$ is a tridiagonal cyclic matrix. This cyclic form is natural consequence of the periodic numbering we have used in solving continuity equations (1.32). The non-zero terms corresponding to the cth row of this matrix write as

$$\begin{cases}\mathbb{N}_{c,c-1} = \alpha_{c-1}l_{c-1}l_c(\mathbb{K}_{c-1}\boldsymbol{n}_{c-1}^{c-1}\cdot\boldsymbol{n}_c^{c-1}),\\ \mathbb{N}_{c,c} = \alpha_{c-1}l_c^2(\mathbb{K}_{c-1}\boldsymbol{n}_c^{c-1}\cdot\boldsymbol{n}_c^{c-1}) + \alpha_c l_c^2(\mathbb{K}_c\boldsymbol{n}_c^c\cdot\boldsymbol{n}_c^c),\\ \mathbb{N}_{c,c+1} = \alpha_c l_c l_{c+1}(\mathbb{K}_c\boldsymbol{n}_{c+1}^c\cdot\boldsymbol{n}_c^c).\end{cases} \quad (1.34)$$

From the first equation it follows that

$$\mathbb{N}_{c+1,c} = \alpha_c l_c l_{c+1}(\mathbb{K}_c\boldsymbol{n}_c^c\cdot\boldsymbol{n}_{c+1}^c).$$

The comparison of this term with $\mathbb{N}_{c,c+1}$ shows that $\mathbb{N}$ is symmetric if and only if the conductivity tensor, $\mathbb{K}_c$ is also symmetric. Regarding $\mathbb{S}$, it is a bidiagonal cyclic matrix, the non-zero terms corresponding to the cth row are:

$$\begin{cases}\mathbb{S}_{c,c-1} = \alpha_{c-1}l_c[l_{c-1}(\mathbb{K}_{c-1}\boldsymbol{n}_{c-1}^{c-1}\cdot\boldsymbol{n}_c^{c-1}) + l_c(\mathbb{K}_{c-1}\boldsymbol{n}_c^{c-1}\cdot\boldsymbol{n}_c^{c-1})],\\ \mathbb{S}_{c,c} = \alpha_c l_c[l_c(\mathbb{K}_c\boldsymbol{n}_c^c\cdot\boldsymbol{n}_c^c) + l_{c+1}(\mathbb{K}_c\boldsymbol{n}_{c+1}^c\cdot\boldsymbol{n}_c^c)].\end{cases} \quad (1.35)$$

Comment 5: *At that point, we have nearly all the information needed to construct the numerical scheme. It remains to present how the boundary conditions are introduced in the formulation. This is explained in section 1.4.7. Solving the system* (1.33) *allows us to express the half-edge fluxes in terms of cell temperatures. We can then build the global linear system by summing all the contributions of the half-edge fluxes around each nodes. Before explaining how to proceed, we show how to build the three-dimensional version of these linear systems.*

1.3 Space discretization in three dimensions

We now present the space discretization in three dimensions. The methodology used is exactly the same as the one used in the previous section. We will focus on the specific ingredients that need to be introduced in order to deal with three dimensional geometries.

1.3.1 Additional notations

Let us introduce some additional notations that will be useful to develop the space discretization of problem (1.1) in a three-dimensional geometry. The domain $\mathcal{D}$ is now paved with non overlapping **polyhedral** cells, *i.e.*, $\mathcal{D} = \cup_c \omega_c$, where ω_c denotes a generic polyhedral cell. In two-dimensions the list of the counterclockwise ordered vertices belonging to a cell is sufficient to fully define a cell. Unfortunately this is not the case anymore in three-dimensional geometry. To complete the cell geometry description, we introduce the set $\mathcal{F}(c)$ as being the list of faces of cell c and the set $\mathcal{F}(p,c)$, which is the list of faces of cell c impinging at point p. We observe that the former set is linked to the latter by $\mathcal{F}(c) = \cup_{p\in\mathcal{P}(c)}\mathcal{F}(p,c)$. A generic face is denoted either by the index f or by $\partial\omega_c^f$.

Here, we consider a mesh composed of polyhedral cells. Namely, the term polyhedral cell stands for a volume enclosed by an arbitrary number of faces, each determined by an arbitrary number (3 or more) of vertices. If a face has four or more vertices, they can be non-coplanar, thus the face is not a plane and it is difficult to define its unit outward normal. To overcome this problem, we employ the decomposition of a polyhedral cell into elementary tetrahedra, as introduced by Burton in [35], to discretize the conservation laws of Lagrangian hydrodynamics onto polyhedral grids. According to Burton's terminology, these elementary tetrahedra are called iotas, since ι is the smallest letter in the Greek alphabet. Being given the polyhedral cell c, we consider the vertex $p \in \mathcal{P}(c)$ which belongs to the face $f \in \mathcal{F}(c)$ and the edge e, refer to Figure 1.5. The iota tetrahedron, $\mathcal{I}^{pfe}$, related to point p, face f and edge e, is constructed by connecting point p, the centroid of cell c, the centroid of face f and the midpoint of edge e as displayed in Figure 1.5. Further, we denote by $\boldsymbol{\mathcal{I}}^{pfec}$, the outward normal vector to the triangular face obtained by connecting the point p to the midpoint of edge e and the centroid of face f. Let us point out that $|\,\boldsymbol{\mathcal{I}}^{pfec}\,|$ is the area of the aforementioned triangular face.

We can define the decomposition of the polyhedral cell, ω_c, into sub-cells. The sub-cell, ω_{pc}, related to point p is obtained by gathering the iotas attached to point p as follows

$$\omega_{pc} = \bigcup_{f\in\mathcal{F}(p,c)} \bigcup_{e\in\mathcal{E}(p,f)} \mathcal{I}^{pfe},$$

where $\mathcal{E}(p,f)$ is the set of edges of face f impinging at point p. For the hexahedral cell displayed in Figure 1.5, the sub-cell ω_{pc} is made of 6 iotas since there are 3 faces impinging at point p and knowing that for each face there are two edges connected to point p. The volume of the sub-cell ω_{pc} is given by

$$|\,\omega_{pc}\,| = \sum_{f\in\mathcal{F}(p,c)} \sum_{e\in\mathcal{E}(p,f)} |\,\mathcal{I}^{pfe}\,|\,.$$

It is worth mentioning that the set of sub-cells, $\{\omega_{pc},\ \ p \in \mathcal{P}(c)\}$, is a partition of the polyhedral cell c and thus the cell volume is defined by

$$|\,\omega_c\,| = \sum_{p\in\mathcal{P}(c)} |\,\omega_{pc}\,|\,.$$

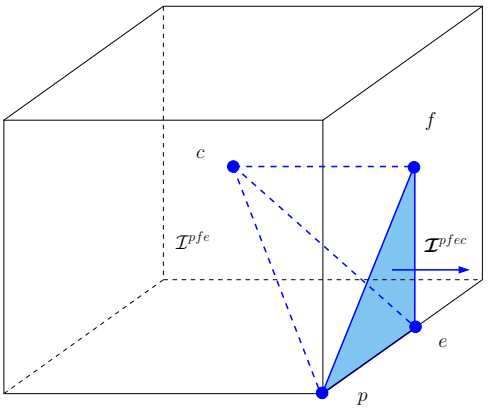


Figure 1.5: Definition of the iota cell $\mathcal{I}^{pfe}$ and the outward normal vector $\boldsymbol{\mathcal{I}}^{pfec}$ related to point p, face f and edge e in the hexahedral cell c.

The sub-face related to point p and face f is denoted by $\partial\omega_{pc}^{f}$ and defined as $\partial\omega_{pc}^{f} = \omega_{pc} \cap \partial\omega_{c}^{f}$. It consists of the union of the two outer triangular faces attached to the two iotas related to point p and face f, refer to Figure 1.6. The area and the unit outward normal corresponding to the sub-face $\partial\omega_{pc}^{f}$ are given by

$$A_{pc}^{f} = \mid \sum_{e \in \mathcal{E}(p,f)} \boldsymbol{\mathcal{I}}^{pfec} \mid, \quad \boldsymbol{n}_{pc}^{f} = \frac{1}{A_{pc}^{f}} \sum_{e \in \mathcal{E}(p,f)} \boldsymbol{\mathcal{I}}^{pfec}.$$

Let us point out that the set of sub-faces, $\{\partial\omega_{pc}^{f}, \quad p \in \mathcal{P}(c,f)\}$, where, $\mathcal{P}(c,f)$ is the set of points of cell c lying on face f, is a partition of the generic face f.

Now, we are in position to construct the space discretization of our diffusion problem. We recall that Eq. (1.4) still holds using the notations introduced in section 1.2.1, namely

$$m_c C_{vc} \frac{\mathsf{d}}{\mathsf{d}t} T_c + \int_{\partial\omega_c} \boldsymbol{q} \cdot \boldsymbol{n} \mathsf{d}s = m_c r_c,$$

To achieve the first step of the space discretization of (1.4), it remains to discretize the surface integral of the heat flux employing the partition of faces into sub-faces.

Knowing that $\partial\omega_c = \cup_{f \in \mathcal{F}(c)} \partial\omega_c^f$ the surface integral of the heat flux reads

$$\int_{\partial\omega_c} \boldsymbol{q} \cdot \boldsymbol{n} \, \mathsf{d}s = \sum_{f \in \mathcal{F}(c)} \int_{\partial\omega_c^f} \boldsymbol{q} \cdot \boldsymbol{n} \, \mathsf{d}s.$$

Now, recalling the partition of face f into sub-cells, *i.e.*, $\partial\omega_c^f = \cup_{p \in \mathcal{P}(c,f)} \omega_{pc}^f$, leads to write the above surface integral as

$$\begin{aligned} \int_{\partial\omega_c} \boldsymbol{q} \cdot \boldsymbol{n} \, \mathsf{d}s &= \sum_{f \in \mathcal{F}(c)} \sum_{p \in \mathcal{P}(c,f)} \int_{\omega_{pc}^f} \boldsymbol{q} \cdot \boldsymbol{n} \, \mathsf{d}s \\ &= \sum_{p \in \mathcal{P}(c)} \sum_{f \in \mathcal{F}(p,c)} \int_{\omega_{pc}^f} \boldsymbol{q} \cdot \boldsymbol{n} \, \mathsf{d}s. \end{aligned}$$

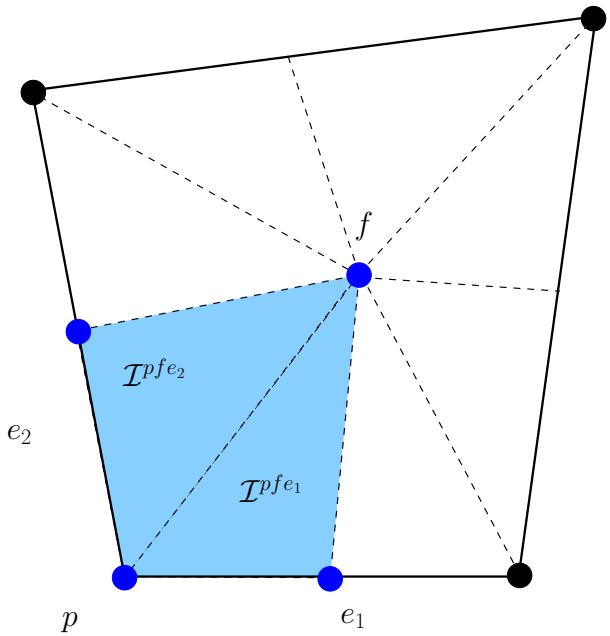


Figure 1.6: Generic quadrilateral face, f, related to the hexahedral cell ω_c. The sub-face, $\partial\omega_{pc}^f$, related to point p and face f is obtained by gathering the triangular faces corresponding to the iotas $\mathcal{I}^{pfe_1}$ and $\mathcal{I}^{pfe_2}$.

Here, we have interchanged the order of the double summation to finally get a global summation over the points of cell c and a local summation over the faces impinging at point p. Let us denote by q_{pc}^f the piecewise constant representation of the normal component of the heat flux over sub-face $\partial\omega_{pc}^f$

$$q_{pc}^f = \frac{1}{A_{pc}^f} \int_{\partial\omega_{pc}^f} \boldsymbol{q} \cdot \boldsymbol{n} \, \mathsf{d}s. \tag{1.36}$$

Gathering the above results, Eq. (1.4) turns into

$$m_c C_{vc} \frac{\mathsf{d}}{\mathsf{d}t} T_c + \sum_{p \in \mathcal{P}(c)} \sum_{f \in \mathcal{F}(p,c)} A_{pc}^f q_{pc}^f = m_c r_c. \tag{1.37}$$

To conclude this paragraph we introduce the sub-face temperature, which will be useful in the description of our scheme as auxiliary unknown

$$T_{pc}^f = \frac{1}{A_{pc}^f} \int_{\partial\omega_{pc}^f} T(\boldsymbol{x}, t) \, \mathsf{d}s. \tag{1.38}$$

In writing this equation, we also assumed a piecewise constant approximation of the temperature field over each sub-face.

Let us write down the continuity conditions, in terms of sub-face fluxes and sub-face temperatures. To this end, we consider two neighboring cells denoted by c and d sharing a face and a point. The face is denoted by f in the local list of faces of cell c and g in the local list of faces in cell d. Regarding the common point, it is denoted by p in the local numbering of cell c and r in the local numbering of cell d. In what follows, we shall consider the sub-cells ω_{pc} and ω_{rd} sharing the sub-face $\partial\omega_{pc}^f \equiv \partial\omega_{rd}^g$, which is displayed in Figure 1.7. For sake of simplicity, we have only plotted the common sub-face shared by the two sub-cells ω_{pc} and ω_{rd}. When viewed

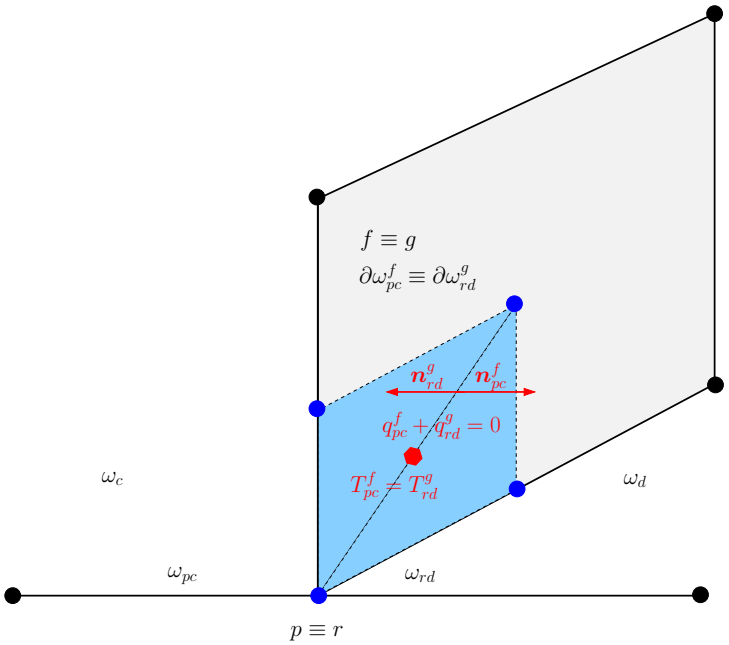


Figure 1.7: Continuity conditions for the sub-face fluxes and temperature on a sub-face shared by two sub-cells attached to the same point. Fragment of a polyhedral grid: quadrilateral face shared by hexahedral cells c and d. Labels p and r denote the indices of the same point relatively to the local numbering of points in cell c and d. The neighboring sub-cells are denoted by ω_{pc} and ω_{rd}. They share the sub-face $\partial\omega_{pc}^f \equiv \partial\omega_{rd}^g$, which has been colored in blue.

from sub-cell ω_{pc} the sub-face temperature and the sub-face flux are denoted by T_{pc}^f and q_{pc}^f, whereas viewed from sub-cell ω_{rd} they are denoted respectively by T_{rd}^g and q_{rd}^g. Using the above notations and recalling that the unit outward normals satisfy $\boldsymbol{n}_{pc}^f = -\boldsymbol{n}_{rd}^g$ leads to write the continuity conditions for the temperatures and the heat flux as

$$A_{pc}^f q_{pc}^f + A_{rd}^g q_{rd}^g = 0, \tag{1.39a}$$

$$T_{pc}^f = T_{rd}^g. \tag{1.39b}$$

To achieve the space discretization of (1.37), it remains to construct an approximation of the sub-face normal flux, that is, to define a numeric sub-face flux function h_{pc}^f such that:

$$q_{pc}^f = h_{pc}^f(T_{pc}^1 - T_c, \ldots, T_{pc}^k - T_c, \ldots, T_{pc}^{\mathfrak{F}_{pc}} - T_c), \quad \forall f \in \mathcal{F}(p,c), \tag{1.40}$$

where $\mathfrak{F}_{pc}$ denotes the number of faces of cell c impinging at point p, that is $\mathfrak{F}_{pc} = |\mathcal{F}(p,c)|$.

To write our scheme we are going to define an approximation of the sub-face numerical fluxes in terms of sub-face temperatures and cell-centered temperatures. We shall then eliminate the sub-face temperatures using the continuity conditions (1.39) across the sub-faces interfaces. This is the topic of the next section.

1.3.2 Expression of a vector in terms of its normal components

Here, we describe the methodology to recover a three-dimensional vector at each vertex of a polyhedron from the normal components related to the sub-faces impinging at each vertex.

Let $\boldsymbol{\phi}$ be an arbitrary vector of the three-dimensional space $\Re^3$ and $\boldsymbol{\phi}_{pc}$ its piecewise constant approximation over the sub-cell ω_{pc}. Let ϕ^f_{pc} be the sub-face normal components of $\boldsymbol{\phi}_{pc}$ defined by

$$\boldsymbol{\phi}_{pc} \cdot \boldsymbol{n}^f_{pc} = \phi^f_{pc}, \quad \forall f \in \mathcal{F}(p,c),$$

where $\mathcal{F}(p,c)$ is the set of sub-faces belonging to cell c and impinging at point p. The above linear system is characterized by 3 unknowns, *i.e.*, the Cartesian components of the vector $\boldsymbol{\phi}_{pc}$ and $\mathfrak{F}_{pc} =\mid \mathcal{F}(p,c) \mid$ equations. This system is properly defined provided that $\mathfrak{F}_{pc} = 3$. Namely, the number of faces of cell c, impinging at point p must be strictly equal to 3. In what follows, we assume that the polyhedral cells we are working with are characterized by $\mathfrak{F}_{pc} = 3$. Let us remark that this restriction allows us to cope with tetrahedron, hexahedron and prism. The extension to the case $\mathfrak{F}_{pc} > 3$ is investigated in appendix A where we study the particular case of pyramids for which $\mathfrak{F}_{pc} = 4$ at one vertex. This particular case is usually sufficient for handling three-dimensional industrial meshes.

Bearing this assumption in mind, let us introduce the corner matrix $\mathbb{J}_{pc} = [\boldsymbol{n}^1_{pc}, \boldsymbol{n}^2_{pc}, \boldsymbol{n}^3_{pc}]$ to rewrite the above 3×3 linear system as

$$\mathbb{J}^t_{pc} \boldsymbol{\phi}_{pc} = \begin{pmatrix} \phi^1_{pc} \\ \phi^2_{pc} \\ \phi^3_{pc} \end{pmatrix},$$

where the superscript t denotes the transpose matrix. Granted that the vectors $\boldsymbol{n}^f_{pc}$, for $f = 1 \dots 3$, are not co-linear, the above linear system has always a unique solution, which reads

$$\boldsymbol{\phi}_{pc} = \mathbb{J}^{-t}_{pc} \begin{pmatrix} \phi^1_{pc} \\ \phi^2_{pc} \\ \phi^3_{pc} \end{pmatrix}. \tag{1.41}$$

This equation allows to express any vector in terms of its normal components on the local basis $\{\boldsymbol{n}^1_{pc}, \boldsymbol{n}^2_{pc}, \boldsymbol{n}^3_{pc}\}$. This representation provides the computation of the inner product of two vectors $\boldsymbol{\phi}_{pc}$ and $\boldsymbol{\psi}_{pc}$ as follows

$$\boldsymbol{\phi}_{pc} \cdot \boldsymbol{\psi}_{pc} = \left(\mathbb{J}^t_{pc} \mathbb{J}_{pc}\right)^{-1} \begin{pmatrix} \psi^1_{pc} \\ \psi^2_{pc} \\ \psi^3_{pc} \end{pmatrix} \cdot \begin{pmatrix} \phi^1_{pc} \\ \phi^2_{pc} \\ \phi^3_{pc} \end{pmatrix}. \tag{1.42}$$

A straightforward computation shows that the 3×3 matrix $\mathbb{H}_{pc} = \mathbb{J}^t_{pc} \mathbb{J}_{pc}$ is expressed in terms of the dot products of the basis vectors

$$(\mathbb{H}_{pc})_{ij} = \boldsymbol{n}^j_{pc} \cdot \boldsymbol{n}^i_{pc}.$$

This matrix is symmetric positive definite and represents the local metric tensor associated to the sub-cell ω_{pc}.

Comment 6: *Let us remark that the problem of finding the expression of a vector in terms of its normal components always admits a unique solution in the two-dimensional case (when not co-linear) since the number of faces of cell c impinging at point p is always equal to two.*

1.3.3 Sub-cell-based variational formulation

We recall the main result obtained from the sub-cell-based variational formulation in section 1.2.3. After integrating Fourier Law on a sub-cell ω_{pc} and applying some approximations,

the last result we obtained before developing the terms using the two-dimensional notations was Eq. (1.18):

$$\int_{\omega_{pc}} \boldsymbol{\phi} \cdot \mathbb{K}^{-1} \boldsymbol{q} dv = T_c \int_{\overline{\partial\omega_{pc}}} \boldsymbol{\phi} \cdot \boldsymbol{n} ds - \int_{\overline{\partial\omega_{pc}}} T\boldsymbol{\phi} \cdot \boldsymbol{n} ds.$$

We pursue the study of the sub-cell-based variational formulation discretizing the right-hand side of this equation using the three-dimensional notations. First, we recall that the outer boundary of sub-cell ω_{pc} decomposes into sub-faces as

$$\overline{\partial\omega_{pc}} = \bigcup_{f \in \mathcal{F}(p,c)} \partial\omega_{pc}^f,$$

where $\partial\omega_{pc}^f$ is the sub-face associated to point p and face f in cell c, and $\mathcal{F}(p,c)$ is the set of faces of cell c impinging at point p. Utilizing the above partition allows to rewrite the right-hand side of (1.18) as

$$\int_{\omega_{pc}} \boldsymbol{\phi} \cdot \mathbb{K}^{-1} \boldsymbol{q} dv = T_c \sum_{f \in \mathcal{F}(p,c)} \int_{\partial\omega_{pc}^f} \boldsymbol{\phi} \cdot \boldsymbol{n} ds - \sum_{f \in \mathcal{F}(p,c)} \int_{\partial\omega_{pc}^f} T\boldsymbol{\phi} \cdot \boldsymbol{n} ds. \tag{1.43}$$

Introducing the sub-face temperature, T_{pc}^f, given by (1.38) and the sub-face approximation of vector $\boldsymbol{\phi}$ defined by $\phi_{pc}^f = \frac{1}{A_{pc}^f} \int_{\partial\omega_{pc}^f} \boldsymbol{\phi} \cdot \boldsymbol{n}\, \mathtt{d}s$, where A_{pc}^f is the area of the sub-face, leads to rewrite the above equation as follows

$$\int_{\omega_{pc}} \boldsymbol{\phi} \cdot \mathbb{K}^{-1} \boldsymbol{q} dv = - \sum_{f \in \mathcal{F}(p,c)} A_{pc}^f (T_{pc}^f - T_c) \phi_{pc}^f. \tag{1.44}$$

Finally, assuming a piecewise constant representation of the test function $\boldsymbol{\phi}$ allows to compute the volume integral in the left-hand side thanks to the quadrature rule

$$\int_{\omega_{pc}} \boldsymbol{\phi} \cdot \mathbb{K}^{-1} \boldsymbol{q} dv = w_{pc} \boldsymbol{\phi}_{pc} \cdot \mathbb{K}_c^{-1} \boldsymbol{q}_{pc}. \tag{1.45}$$

Here, $\mathbb{K}_c$ is the piecewise constant approximation of the conductivity tensor, $\boldsymbol{\phi}_{pc}$ and $\boldsymbol{q}_{pc}$ are the piecewise constant approximation of the vectors $\boldsymbol{\phi}$ and $\boldsymbol{q}$. The corner volume w_{pc} must satisfy the consistency condition (1.21).
Expressing the vectors $\boldsymbol{q}_{pc}$ and $\boldsymbol{\phi}_{pc}$ in terms of their normal components by means of (1.41) allows to write the right-hand side of (1.45) as

$$w_{pc} \boldsymbol{\phi}_{pc} \cdot \mathbb{K}_c^{-1} \boldsymbol{q}_{pc} = w_{pc} \left(\mathbb{J}_{pc}^t \mathbb{K}_c \mathbb{J}_{pc} \right)^{-1} \begin{pmatrix} q_{pc}^1 \\ q_{pc}^2 \\ q_{pc}^3 \end{pmatrix} \cdot \begin{pmatrix} \phi_{pc}^1 \\ \phi_{pc}^2 \\ \phi_{pc}^3 \end{pmatrix}, \tag{1.46}$$

where $\mathbb{J}_{pc}$ is the corner matrix defined by $\mathbb{J}_{pc} = [\boldsymbol{n}_{pc}^1, \boldsymbol{n}_{pc}^2, \boldsymbol{n}_{pc}^3]$. Recalling that $|\mathcal{F}(p,c)| = 3$ leads to rewrite the right-hand side of (1.44) as

$$- \sum_{f \in \mathcal{F}(p,c)} A_{pc}^f (T_{pc}^f - T_c) \phi_{pc}^f = - \begin{pmatrix} A_{pc}^1 (T_{pc}^1 - T_c) \\ A_{pc}^2 (T_{pc}^2 - T_c) \\ A_{pc}^3 (T_{pc}^3 - T_c) \end{pmatrix} \cdot \begin{pmatrix} \phi_{pc}^1 \\ \phi_{pc}^2 \\ \phi_{pc}^3 \end{pmatrix}. \tag{1.47}$$

Finally, combining (1.46) and (1.47), the sub-cell variational formulation becomes

$$w_{pc} \left(\mathbb{J}_{pc}^t \mathbb{K}_c \mathbb{J}_{pc} \right)^{-1} \begin{pmatrix} q_{pc}^1 \\ q_{pc}^2 \\ q_{pc}^3 \end{pmatrix} \cdot \begin{pmatrix} \phi_{pc}^1 \\ \phi_{pc}^2 \\ \phi_{pc}^3 \end{pmatrix} = - \begin{pmatrix} A_{pc}^1 (T_{pc}^1 - T_c) \\ A_{pc}^2 (T_{pc}^2 - T_c) \\ A_{pc}^3 (T_{pc}^3 - T_c) \end{pmatrix} \cdot \begin{pmatrix} \phi_{pc}^1 \\ \phi_{pc}^2 \\ \phi_{pc}^3 \end{pmatrix}. \tag{1.48}$$

Knowing that this equation must hold for any vector $\boldsymbol{\phi}_{pc}$, we obtain

$$\begin{pmatrix} q^1_{pc} \\ q^2_{pc} \\ q^3_{pc} \end{pmatrix} = -\frac{1}{w_{pc}} (\mathbb{J}^t_{pc} \mathbb{K}_c \mathbb{J}_{pc}) \begin{pmatrix} A^1_{pc}(T^1_{pc} - T_c) \\ A^2_{pc}(T^2_{pc} - T_c) \\ A^3_{pc}(T^3_{pc} - T_c) \end{pmatrix}. \tag{1.49}$$

This equation constitutes the approximation of the sub-face normal fluxes. This local approximation is compatible with the expression of the constitutive law (1.2) in the sense that the discrete approximation of the heat flux is equal to a tensor times the approximation of the temperature gradient. This tensor can be viewed as an effective conductivity tensor associated to the sub-cell ω_{pc}. As in the two-dimensional scheme, it is natural to set

$$\mathbb{K}_{pc} = \mathbb{J}^t_{pc} \mathbb{K}_c \mathbb{J}_{pc}.$$

Let us emphasize that this corner tensor inherits all the properties of the conductivity tensor $\mathbb{K}_c$. Namely, $\mathbb{K}_c$ being symmetric positive definite, $\mathbb{K}_{pc}$ is also symmetric positive definite. Recalling that $\mathbb{J}_{pc} = [\boldsymbol{n}^1_{pc}, \boldsymbol{n}^2_{pc}, \boldsymbol{n}^3_{pc}]$, we readily obtain the expression of the entries of the corner tensor, $\mathbb{K}_{pc}$, in terms of the unit normals $\boldsymbol{n}^f_{pc}$ for $f = 1 \dots 3$ and the cell conductivity $\mathbb{K}_c$

$$(\boldsymbol{K_{pc}})_{fg} = (\boldsymbol{K_c} \boldsymbol{n}^f_{pc}) \cdot \boldsymbol{n}^g_{pc}.$$

Finally, the sub-face flux approximation for the sub-face f is written under the compact form

$$q^f_{pc} = -\alpha_{pc} \sum_{g=1}^{3} (\mathbb{K}_{pc})_{fg} A^g_{pc} (T^g_{pc} - T_c), \tag{1.50}$$

where $\alpha_{pc} = \frac{1}{w_{pc}}$.

Comment 7: *This result is equivalent to the one obtained in two dimensions in Eq.* (1.27). *To obtain it we can replace the superscript f and g in Eq.* (1.50) *by the superscript $+$ and $-$, the third row and third column of the tensor $\mathbb{K}_{pc}$ being equal to zero.*

1.3.4 Elimination of the sub-face temperatures

Having defined the flux approximation in terms of the difference between the cell and the sub-face temperatures, we shall express the sub-face temperatures in terms of the cell temperatures of the cells c surrounding a specific point p, using the continuity conditions of the normal heat flux at cell interfaces. In order to have a simpler expression of the equations we are going to introduce some new local notations. First of all, in this paragraph we are dealing with quantities located around a point p, so in all the notations we will omit to specify the subscript p. For each face f in the list $\mathcal{F}(p)$ of the faces impinging at the node p we associate two tuples (c, i) and (d, j) which identify the neighboring cells c and d of the face f and their local numbering i (resp. j) in the subset $\mathcal{F}(p, c)$ (resp. $\mathcal{F}(p, d)$) of $\mathcal{F}(p)$. With this notation a sub-face temperature T^i_{pc} is denoted by $\bar{T}^i_c$ and using the continuity condition on the temperature is equal to T^j_{pd} which is denoted $\bar{T}^j_d$ and can also be simply denoted by $\bar{T}^f$. The bar notation help us to make the difference between the cell centered unknown and the sub-face unknown. Similarly the area of the sub-face f can be indifferently noted A^i_c, A^j_d or A^f. The local conductivity tensor $\mathbb{K}_{pc}$ will now be denoted by $\mathbb{K}^c$ so its components $(\mathbb{K}_{pc})_{ij}$ can be written $\mathbb{K}^c_{ij}$.

Using this notation, Eq. (1.50), which defines the heat flux approximation, rewrites

$$q^i_c = -\alpha_c \sum_{k=1}^{3} \mathbb{K}^c_{ik} A^k_c (\bar{T}^k_c - T_c), \tag{1.51}$$

where α_c is the inverse of the volume weight. The continuity condition of the sub-face fluxes across the face $f \equiv (c,i) \equiv (d,j)$ reads

$$A_c^i q_c^i + A_d^j q_d^j = 0.$$

Replacing the sub-face fluxes by their approximation (1.51) into the above equation yields

$$-\alpha_c A_c^i \sum_{k=1}^{3} \mathbb{K}_{ik}^c A_c^k (\bar{T}_c^k - T_c) - \alpha_d A_d^j \sum_{k=1}^{3} \mathbb{K}_{jk}^d A_d^k (\bar{T}_d^k - T_d) = 0.$$

Let us point out that this equation holds for all the faces f impinging at node p, *i.e.* for all $f \in \mathcal{F}(p)$. Denoting $\mathfrak{F}_p = |\mathcal{F}(p)|$ the number of faces impinging at node p, the set of all the above equations forms a $\mathfrak{F}_p \times \mathfrak{F}_p$ linear system, which writes under the compact form

$$\mathbb{N}\bar{\boldsymbol{T}} = \mathbb{S}\boldsymbol{T}. \tag{1.52}$$

Here, the matrix $\mathbb{N}$ is a $\mathfrak{F}_p \times \mathfrak{F}_p$ square matrix and $\bar{\boldsymbol{T}} \in \mathfrak{R}^{\mathfrak{F}_p}$ is the vector of sub-face temperatures. Denoting $\mathfrak{C}_p = |\mathcal{C}(p)|$ the number of cells surrounding node p, the matrix $\mathbb{S}$ is a $\mathfrak{F}_p \times \mathfrak{C}_p$ rectangular matrix and vector $\boldsymbol{T} \in \mathfrak{R}^{\mathfrak{C}_p}$ is the vector of cell temperatures. The matrix $\mathbb{N}$ has five non-zero terms on each lines, its diagonal part writes

$$\mathbb{N}_{ff} = \alpha_c A_c^i \mathbb{K}_{ii}^c A_c^i + \alpha_d A_d^j \mathbb{K}_{jj}^d A_d^j.$$

Regarding its extra-diagonal parts, two terms come from the contribution of the sub-cell ω_{pc}. Let g be a generic face of cell c impinging at point p characterized by the index k in the local numbering , *i.e.*, $g \equiv (c,k)$, then the extra-diagonal entries related to cell c and faces i and k write

$$\mathbb{N}_{fg} = \alpha_c A_c^i \mathbb{K}_{ik}^c A_c^k, \quad \text{for} \quad k \in [1,3] \quad \text{and} \quad k \neq i.$$

The two remaining terms come from the sub-cell ω_{pd}. Let g be a generic face of cell d impinging at point p characterized by the index k in the local numbering , *i.e.*, $g \equiv (d,k)$, then the extra-diagonal entries related to cell d and faces j and k write

$$\mathbb{N}_{fg} = \alpha_d A_d^j \mathbb{K}_{jk}^d A_d^k, \quad \text{for} \quad k \in [1,3] \quad \text{and} \quad k \neq j$$

Let us remark that the matrix $\mathbb{N}$ has a symmetric structure, for $g \equiv (c,k)$, $f \equiv (c,i)$ we have $\mathbb{N}_{gf} = \alpha_c A_c^k \mathbb{K}_{ki}^c A_c^i$ and for $g \equiv (d,k)$, $f \equiv (d,j)$ we have $\mathbb{N}_{gf} = \alpha_d A_d^k \mathbb{K}_{kj}^d A_d^j$. We also note that $\mathbb{N}$ is symmetric if and only if $\mathbb{K}^c$ (resp. $\mathbb{K}^d$) is symmetric.

Finally, the matrix $\mathbb{S}$ has two non-zero terms on each row, one term for each neighboring cell c and d of the face f

$$\mathbb{S}_{fc} = \alpha_c \sum_{k=1}^{3} A_c^i \mathbb{K}_{ik}^c A_c^k,$$

$$\mathbb{S}_{fd} = \alpha_d \sum_{k=1}^{3} A_d^j \mathbb{K}_{jk}^d A_d^k.$$

Comment 8: *While in two dimensions the structure of the $\mathbb{N}$ matrix was tridiagonal cyclic, it is not the case anymore in three dimensions. This is one drawback of the scheme in three-dimensions, the inversion of this matrix and thus the cost of the construction of the global system is larger than in two-dimensions. The size of these local systems remains small regarding the size of the global system as presented in Table 1.1.*

In the next section, we investigate the properties of the matrices $\mathbb{N}$ and $\mathbb{S}$. This allows to write an explicit formulation of the sub-faces fluxes in terms of cell temperature. We then discuss the properties of the resulting scheme.

1.4 Construction and properties of the semi-discrete scheme

In this section, we start by stating the properties of the matrices $\mathbb{N}$ and $\mathbb{S}$ defined earlier. This lead us to the construction of the final form of the numerical scheme. We continue by describing some interesting properties that characterize our finite volume semi-discrete scheme. First we show that the fundamental inequality $\boldsymbol{q}\cdot\nabla T \leq 0$ is satisfied at the discrete level. Then, we show that the scheme is characterized by a positive semi-definite global diffusion matrix. In a third paragraph, we demonstrate the L^2-stability of the space discretization. In the fourth paragraph, we present how the boundary conditions are implemented and integrated in the global system. Finally, in the last paragraph we present a way of computing the volume weight ω_{pc} for various kind of cells. From now on, we will use the notations of the three-dimensional version of the scheme, which are more generic, but the results still holds in two-dimensions of space.

1.4.1 Properties of the matrices $\mathbb{N}$ and $\mathbb{S}$

The main motivation of this paragraph is to demonstrate the invertibility of the matrix $\mathbb{N}$ to ensure that the linear system (1.52) that solves the sub-face temperatures in terms of the cell temperatures admits always a unique solution. To this end, let us show that $\mathbb{N}$ is a positive-definite matrix. First, we introduce the matrix $\mathbb{L}^c$ of size $3 \times \mathfrak{F}_p$ defined by

$$\mathbb{L}^c_{ij} = \begin{cases} 1 & \text{if } j \equiv (c,i), \\ 0 & \text{elsewhere.} \end{cases}$$

Here, $\mathbb{L}^c$ is the rectangular matrix which associates the sub-face of cell c in its local numbering to its numbering around point p. We define the diagonal matrix $\mathbb{A}$ of size $\mathfrak{F}_p \times \mathfrak{F}_p$, which contains the area of the sub-faces, namely $\mathbb{A}_{ff} = A^c_i$ for the face $f \equiv (c,i)$. We define

$$\mathbb{A}^c = \mathbb{L}^c\mathbb{A},$$

the matrix which relates the area of sub-face of cell c in its local numbering to its numbering around the point p. Employing this notation, it is straightforward to show that matrix $\mathbb{N}$ writes

$$\mathbb{N} = \sum_{c\in\mathcal{C}(p)} \alpha_c \left(\mathbb{A}^c\right)^t \mathbb{K}^c\mathbb{A}^c.$$

We are going to show that $\mathbb{N}\bar{\boldsymbol{T}}\cdot\bar{\boldsymbol{T}} > 0$, for all $\bar{\boldsymbol{T}} \in \mathfrak{R}^{\mathfrak{F}_p}$. To this end, we compute $\mathbb{N}\bar{\boldsymbol{T}}\cdot\bar{\boldsymbol{T}}$ employing the above decomposition of $\mathbb{N}$

$$\begin{aligned} \mathbb{N}\bar{\boldsymbol{T}}\cdot\bar{\boldsymbol{T}} &= \sum_{c\in\mathcal{C}(p)} \alpha_c \left(\mathbb{A}^c\right)^t \mathbb{K}^c\mathbb{A}^c\bar{\boldsymbol{T}}\cdot\bar{\boldsymbol{T}} \\ &= \sum_{c\in\mathcal{C}(p)} \alpha_c\mathbb{K}^c(\mathbb{A}^c\bar{\boldsymbol{T}})\cdot(\mathbb{A}^c\bar{\boldsymbol{T}}). \end{aligned}$$

Recalling that α_c is non-negative and $\mathbb{K}^c$ is positive definite ensures that the right-hand side of the above equation is always non-negative, which ends the proof. Thus, matrix $\mathbb{N}$ is invertible and the sub-face temperatures are expressed in terms of the cell temperatures by means of the relation

$$\bar{\boldsymbol{T}} = \left(\mathbb{N}^{-1}\mathbb{S}\right)\boldsymbol{T}. \tag{1.53}$$

Further, if the cell temperature field is uniform, then the sub-face temperatures are also uniform and share the same constant value. This property follows from the relation satisfied by the matrices $\mathbb{N}$ and $\mathbb{S}$

$$\left(\mathbb{N}^{-1}\mathbb{S}\right)\mathbf{1}_{\mathfrak{C}_p} = \mathbf{1}_{\mathfrak{F}_p}, \tag{1.54}$$

Here, $\mathbf{1}_n$, where n is an integer, is the vector of size n, whose entries are equal to 1. To demonstrate the above relation, we show that $\mathbb{S}\mathbf{1}_{\mathfrak{C}_p} = \mathbb{N}\mathbf{1}_{\mathfrak{F}_p}$ by developing respectively the left and the right-hand side of this equality. Substituting the non-zero entries of matrix $\mathbb{S}$ leads to write the left-hand side

$$\begin{aligned}\left(\mathbb{S}\mathbf{1}_{\mathfrak{C}_p}\right)_f =& \mathbb{S}_{fc} + \mathbb{S}_{fd} \\ =& \alpha_c \sum_{k=1}^{3} A_c^i \mathbb{K}_{ik}^c A_c^k + \alpha_d \sum_{k=1}^{3} A_d^j \mathbb{K}_{jk}^d A_d^k. \end{aligned} \tag{1.55}$$

Replacing the non-zero entries of matrix $\mathbb{N}$ allows to express the right-hand side as

$$\left(\mathbb{N}\mathbf{1}_{\mathfrak{F}_p}\right)_f = \alpha_c A_c^i \mathbb{K}_{ii}^c A_c^i + \alpha_d A_d^j \mathbb{K}_{jj}^d A_d^j + \sum_{k=1,k\neq i}^{3} \alpha_c A_c^i \mathbb{K}_{ik}^c A_c^k + \sum_{k=1,k\neq j}^{3} \alpha_d A_d^j \mathbb{K}_{jk}^d A_d^k.$$

Gathering the common terms in the above equation yields

$$\left(\mathbb{N}\mathbf{1}_{\mathfrak{F}_p}\right)_f = \alpha_c \sum_{k=1}^{3} A_c^i \mathbb{K}_{ik}^c A_c^k + \alpha_d \sum_{k=1}^{3} A_d^j \mathbb{K}_{jk}^d A_d^k. \tag{1.56}$$

The comparison between (1.55) and (1.56) shows that for all $f \in \mathcal{F}(p)$, $\left(\mathbb{N}^{-1}\mathbb{S}\right)\mathbf{1}_{\mathfrak{C}_p} = \mathbf{1}_{\mathfrak{F}_p}$, which ends the proof.

After having expressed the half-edge temperatures in terms of the mean cell temperatures, we are now in position to achieve the construction of the scheme by gathering the previous results.

1.4.2 Local diffusion matrix at a generic point

In this paragraph, we achieve the space discretization of the diffusion equation gathering the results obtained in the previous sections. We start by recalling the semi-discrete version of the diffusion equation (1.37)

$$m_c C_{vc} \frac{\mathsf{d}}{\mathsf{d}t} T_c + \sum_{p\in\mathcal{P}(c)} \sum_{f\in\mathcal{F}(p,c)} A_{pc}^f q_{pc}^f = m_c r_c.$$

We define the contribution of the sub-cell ω_{pc} to the diffusion flux as

$$Q_{pc} = \sum_{f\in\mathcal{F}(p,c)} A_{pc}^f q_{pc}^f.$$

Using the local numbering of the sub-faces surrounding point p yields to rewrite the above expression as

$$Q_{pc} = \sum_{k=1}^{3} A_c^k q_c^k.$$

Now, we replace the normal flux by its corresponding expression (1.51) to get

$$Q_{pc} = -\sum_{k=1}^{3} A_c^k \left[\alpha_c \sum_{i=1}^{3} \mathbb{K}_{ki}^c A_c^i (\bar{T}_c^i - T_c)\right].$$

Interchanging the order of the summations in the right-hand side yields

$$Q_{pc} = -\sum_{i=1}^{3}\left[\alpha_c \sum_{k=1}^{3}(A_c^i \mathbb{K}_{ki}^c A_c^k)\right](\bar{T}_c^i - T_c).$$

To obtain a more compact form of Q_{pc}, we define the matrix $\widetilde{\mathbb{S}}$ whose entries write $\widetilde{\mathbb{S}}_{fc} = \alpha_c \sum_{k=1}^{3}(A_c^i \mathbb{K}_{ki}^c A_c^k)$, where $f \equiv (c,i)$. Employing this notation, the sub-cell contribution to the diffusion flux reads

$$Q_{pc} = -\sum_{f\in\mathcal{F}(p)} \widetilde{\mathbb{S}}_{cf}^t(\bar{T}^f - T_c).$$

Eliminating the sub-face temperatures by means of (1.53) and using the property (1.54) leads to

$$Q_{pc} = -\sum_{d\in\mathcal{C}(p)} \mathbb{G}_{cd}^p(T_d - T_c), \tag{1.57}$$

where $\mathbb{G}^p$ is a $\mathfrak{C}_p \times \mathfrak{C}_p$ matrix defined at point p by

$$\mathbb{G}^p = \widetilde{\mathbb{S}}^t \mathbb{N}^{-1} \mathbb{S}. \tag{1.58}$$

Let us point out that the entries of $\mathbb{G}^p$ have the physical dimension of a conductivity. Thus, it can be viewed as the effective conductivity tensor at point p. More precisely, it follows from (1.57) that the entry $\mathbb{G}_{cd}^p$ stands for the effective conductivity between cells c and d through the point p. This node-based effective conductivity tensor will be the cornerstone to assemble the global diffusion matrix over the computational grid.

Comment 9: *If the conductivity tensor $\mathbb{K}$ is symmetric, it is straightforward to show that $\widetilde{\mathbb{S}} = \mathbb{S}$. We claim that $\mathbb{G}^p$ is symmetric positive definite provided that the conductivity tensor $\mathbb{K}$ is itself symmetric positive definite. To prove this result, it is sufficient to observe that*

$$\begin{aligned}\mathbb{G}^p\boldsymbol{T}\cdot\boldsymbol{T} =& (\widetilde{\mathbb{S}}^t\mathbb{N}^{-1}\mathbb{S})\boldsymbol{T}\cdot\boldsymbol{T}\\ =& \mathbb{N}^{-1}(\mathbb{S}\boldsymbol{T})\cdot(\widetilde{\mathbb{S}}\boldsymbol{T}),\end{aligned}$$

where $\boldsymbol{T} \in \mathfrak{R}^{\mathfrak{C}_p}$ is the vector of cell temperatures. Since $\mathbb{K}$ is symmetric, one deduces that $\widetilde{\mathbb{S}} = \mathbb{S}$, in addition $\mathbb{N}$ is symmetric positive definite, which ends the proof.

1.4.3 Construction of the global diffusion matrix

Taking into account the previous results, the semi-discrete scheme over cell c reads

$$m_c C_{vc}\frac{\mathsf{d}}{\mathsf{d}t}T_c - \sum_{p\in\mathcal{P}(c)}\sum_{d\in\mathcal{C}(p)} \mathbb{G}_{cd}^p(T_d - T_c) = m_c r_c, \tag{1.59}$$

where $\mathcal{P}(c)$ is the set of points of cell c and $\mathcal{C}(p)$ is the set of cells surrounding the point p. This equation allows to construct the generic entries of the global diffusion matrix, $\mathbb{D}$, as follows

$$\mathbb{D}_{cc} = \sum_{p\in\mathcal{P}(c)}\sum_{d\in\mathcal{C}(p)} \mathbb{G}_{cd}^p, \tag{1.60a}$$

$$\mathbb{D}_{cd} = -\sum_{p\in\mathcal{P}(c)} \mathbb{G}_{cd}^p, c \neq d. \tag{1.60b}$$

If $\mathfrak{C}_{\mathcal{D}}$ denotes the total number of cells composing the computational grid, then matrix $\mathbb{D}$ is a $\mathfrak{C}_{\mathcal{D}} \times \mathfrak{C}_{\mathcal{D}}$ square matrix. The vector of cell-centered temperatures, $\boldsymbol{\mathcal{T}} \in \mathfrak{R}^{\mathfrak{C}_{\mathcal{D}}}$, is the solution of the system of differential equations

$$\mathbb{M}\mathbb{C}_v \frac{\mathsf{d}}{\mathsf{d}t}\boldsymbol{\mathcal{T}} + \mathbb{D}\boldsymbol{\mathcal{T}} = \mathbb{M}\boldsymbol{\mathcal{R}}. \tag{1.61}$$

Here, $\boldsymbol{\mathcal{R}} \in \mathfrak{R}^{\mathfrak{C}_{\mathcal{D}}}$ is the source term vector, $\mathbb{M}$ and $\mathbb{C}_v$ are the diagonal matrices whose entries are respectively the cell mass m_c and the cell heat capacity C_{vc}.

1.4.4 Fundamental inequality at the discrete level

In this paragraph we demonstrate that the discrete approximation of the sub-face normal fluxes (1.50) satisfies a discrete version of the fundamental inequality which follows from the Second Law of thermodynamics: $\boldsymbol{q} \cdot \nabla T \leq 0$.

The discrete counterpart of the fundamental inequality states that for the sub-faces fluxes defined according to (1.50) the following inequality holds

$$\sum_{c \in \mathcal{C}(p)} \left(\sum_{f \in \mathcal{F}(p,c)} A_{pc}^f q_{pc}^f \right) T_c \geq 0. \tag{1.62}$$

To demonstrate this result, let us introduce, I_p, the nodal quantity defined by

$$I_p = \sum_{c \in \mathcal{C}(p)} \left(\sum_{f \in \mathcal{F}(p,c)} A_{pc}^f q_{pc}^f \right) T_c. \tag{1.63}$$

We prove that I_p is always positive using the sub-cell variational formulation. Imposing $\boldsymbol{\phi} = \boldsymbol{q}$ in (1.48) yields

$$w_{pc}\mathbb{K}_{pc}^{-1} \begin{pmatrix} q_{pc}^1 \\ q_{pc}^2 \\ q_{pc}^3 \end{pmatrix} \cdot \begin{pmatrix} q_{pc}^1 \\ q_{pc}^2 \\ q_{pc}^3 \end{pmatrix} = - \begin{pmatrix} A_{pc}^1 (T_{pc}^1 - T_c) \\ A_{pc}^2 (T_{pc}^2 - T_c) \\ A_{pc}^3 (T_{pc}^3 - T_c) \end{pmatrix} \cdot \begin{pmatrix} q_{pc}^1 \\ q_{pc}^2 \\ q_{pc}^3 \end{pmatrix}. \tag{1.64}$$

Now, rearranging the right-hand side leads to

$$w_{pc}\mathbb{K}_{pc}^{-1} \begin{pmatrix} q_{pc}^1 \\ q_{pc}^2 \\ q_{pc}^3 \end{pmatrix} \cdot \begin{pmatrix} q_{pc}^1 \\ q_{pc}^2 \\ q_{pc}^3 \end{pmatrix} = \left(\sum_{i=1}^{3} A_{pc}^i q_{pc}^i \right) T_c - \sum_{i=1}^{3} A_{pc}^i q_{pc}^i T_{pc}^i. \tag{1.65}$$

We notice that the left hand-side of (1.65) is always non-negative since $\mathbb{K}_{pc}$ is positive definite. Summing the equation (1.65) over all cells surrounding p yields

$$\sum_{c \in \mathcal{C}(p)} w_{pc}\mathbb{K}_{pc}^{-1} \begin{pmatrix} q_{pc}^1 \\ q_{pc}^2 \\ q_{pc}^3 \end{pmatrix} \cdot \begin{pmatrix} q_{pc}^1 \\ q_{pc}^2 \\ q_{pc}^3 \end{pmatrix} = \sum_{c \in \mathcal{C}(p)} \left(\sum_{i=1}^{3} A_{pc}^i q_{pc}^i \right) T_c - \sum_{c \in \mathcal{C}(p)} \left(\sum_{i=1}^{3} A_{pc}^i q_{pc}^i T_{pc}^i \right). \tag{1.66}$$

Due to the continuity condition of the sub-face temperatures, the second term of the right-hand side is equal to zero. Finally, Eq. (1.66) becomes

$$I_p = \sum_{c \in \mathcal{C}(p)} w_{pc}\mathbb{K}_{pc}^{-1} \begin{pmatrix} q_{pc}^1 \\ q_{pc}^2 \\ q_{pc}^3 \end{pmatrix} \cdot \begin{pmatrix} q_{pc}^1 \\ q_{pc}^2 \\ q_{pc}^3 \end{pmatrix} \geq 0, \tag{1.67}$$

which ends the proof.

Comment 10: *Inequality* (1.67) *is not only the discrete counterpart of the Second Law of thermodynamics but also the cornerstone to demonstrate the L^2-stability of the semi-discrete formulation of our finite volume scheme as we shall see in Paragraph 1.4.6.*

1.4.5 Positive semi-definiteness of the global diffusion matrix

In the following sections, we consider that the solution has a compact stencil so we can safely ignore the contribution of the boundary conditions. The integration of the boundary conditions into the scheme are presented in section 1.4.7.

We demonstrate that the global diffusion matrix, $\mathbb{D}$, is positive semi-definite, that is for all $\boldsymbol{\mathcal{T}} \in \mathfrak{R}^{\mathsf{C}_\mathcal{D}}$

$$\mathbb{D}\boldsymbol{\mathcal{T}} \cdot \boldsymbol{\mathcal{T}} \geq 0. \tag{1.68}$$

To prove this results, let us write the c-th entry of vector $\mathbb{D}\boldsymbol{\mathcal{T}}$

$$\begin{aligned}(\mathbb{D}\boldsymbol{\mathcal{T}})_c &= \sum_{p\in\mathcal{P}(c)} Q_{pc} \\ &= \sum_{p\in\mathcal{P}(c)} \sum_{f\in\mathcal{F}(p,c)} A_{pc}^f q_{pc}^f.\end{aligned}$$

Employing the above expression, the left-hand side of (1.68) reads

$$\mathbb{D}\boldsymbol{\mathcal{T}} \cdot \boldsymbol{\mathcal{T}} = \sum_{c=1}^{\mathsf{C}_\mathcal{D}} \sum_{p\in\mathcal{P}(c)} \sum_{f\in\mathcal{F}(p,c)} A_{pc}^f q_{pc}^f T_c.$$

Interchanging the order of summation lead to

$$\begin{aligned}\mathbb{D}\boldsymbol{\mathcal{T}} \cdot \boldsymbol{\mathcal{T}} &= \sum_{p=1}^{\mathsf{P}_\mathcal{D}} \sum_{c\in\mathcal{C}(p)} \sum_{f\in\mathcal{F}(p,c)} A_{pc}^f q_{pc}^f T_c \\ &= \sum_{p=1}^{\mathsf{P}_\mathcal{D}} I_p.\end{aligned}$$

Here, $\mathsf{P}_\mathcal{D}$ is the total number of nodes of the computational grid and $I_p = \sum_{c\in\mathcal{C}(p)} \sum_{f\in\mathcal{F}(p,c)} A_{pc}^f q_{pc}^f T_c$ has been already defined by Eq. (1.63). Due to the fundamental inequality satisfied by the discrete sub-face normal flux approximation, refer to Paragraph 1.4.4, I_p is always positive, which ends the proof.

1.4.6 L^2-stability of the semi-discrete scheme

In this paragraph, we prove the stability of our semi-discrete scheme in the absence of source term ($r = 0$) with respect to the discrete L^2 weighted norm defined by

$$\|\boldsymbol{\mathcal{T}}\|_{w2}^2 = \sum_{c=1}^{\mathsf{C}_\mathcal{D}} m_c C_{cv} T_c^2, \tag{1.69}$$

where $\mathsf{C}_\mathcal{D}$ is the total number of cells of the computational domain $\mathcal{D}$. In the absence of the source term, the semi-discrete scheme reads

$$\mathbb{M}\mathbb{C}_v \frac{\mathsf{d}\boldsymbol{\mathcal{T}}}{\mathsf{d}t} + \mathbb{D}\boldsymbol{\mathcal{T}} = \mathbf{0},$$

Dot-multiplying the above equation by $\boldsymbol{\mathcal{T}} \in \mathfrak{R}^{\mathsf{C}_\mathcal{D}}$ yields

$$\mathbb{MC}_v \frac{\mathsf{d}\boldsymbol{\mathcal{T}}}{\mathsf{d}t} \cdot \boldsymbol{\mathcal{T}} + \mathbb{D}\boldsymbol{\mathcal{T}} \cdot \boldsymbol{\mathcal{T}} = 0.$$

Assuming that the mass density and the heat capacity do not depend on time, the above equation turns into

$$\frac{\mathsf{d}}{\mathsf{d}t}(\frac{1}{2}\mathbb{MC}_v\boldsymbol{\mathcal{T}} \cdot \boldsymbol{\mathcal{T}}) = -\mathbb{D}\boldsymbol{\mathcal{T}} \cdot \boldsymbol{\mathcal{T}}.$$

Recalling that the global diffusion matrix, $\mathbb{D}$, is positive semi-definite and employing the definition of the discrete L^2 norm (1.69) leads to the inequality

$$\frac{\mathsf{d}}{\mathsf{d}t}\left(\|\boldsymbol{\mathcal{T}}\|_{w2}^2\right) \leq 0. \tag{1.70}$$

Here, we have ignored the contributions of the boundary terms assuming for instance periodic or homogeneous Neumann boundary conditions. This inequality shows that the L^2 norm of the semi-discrete solution remains bounded by the L^2 norm of the initial data. This implies the L^2-stability of our semi-discrete finite volume scheme.

1.4.7 Boundary conditions

In this paragraph, we present a generic methodology to implement the boundary conditions, which is crucial when dealing with real-word applications. It is worth mentioning that the boundary terms discretization is derived in a consistent manner with the scheme construction. To take into account the boundary terms, let us write the linear system linking the sub-face temperatures with the cell temperatures under the form

$$\mathbb{N}\bar{\boldsymbol{\mathcal{T}}} = \mathbb{S}\boldsymbol{\mathcal{T}} + \boldsymbol{\mathcal{B}}, \tag{1.71}$$

where the extra term $\boldsymbol{\mathcal{B}}$ is the vector containing the boundary conditions contribution, which shall be defined in the next paragraphs.

Let us consider a sub-face f located on the boundary of the domain, in the next paragraphs, we describe the modifications to bring to the matrices and boundary vector, depending on the boundary conditions types under consideration.

Dirichlet boundary condition

On the boundary sub-face $f \equiv (c, i)$, the temperature $\bar{T}^*$ is imposed, we have $\bar{T}_c^i = \bar{T}^f = \bar{T}^*$. We multiply this equation by A_c^i, thus $A_c^i\bar{T}^f = A_c^i\bar{T}^*$. Let us write this equation under the system form (1.71). The diagonal term of the fth line of the system writes

$$\mathbb{N}_{ff} = A_c^i.$$

The corresponding extra-diagonal term is given by

$$\mathbb{N}_{fg} = 0, \ \ \forall\, g \neq f.$$

Regarding the matrix $\mathbb{S}$, we obtain

$$\mathbb{S}_{fg} = 0, \ \ \forall\, g.$$

Finally, the fth component of the vector $\boldsymbol{\mathcal{B}}$ reads

$$\boldsymbol{\mathcal{B}}_f = A_c^i\bar{T}^*.$$

Neumann boundary condition

On the boundary sub-face $f \equiv (c,i)$ the normal flux q^* is prescribed, hence the continuity condition rewrites

$$q_c^i = q^*. \tag{1.72}$$

Multiplying this equation by A_c^i and replacing q_c^i by its expression (1.51) yields

$$-\alpha_c A_c^i \sum_{k=1}^{3} \boldsymbol{K}_{ik}^c A_c^k (\bar{T}_c^k - T_c) = A_c^i q^*. \tag{1.73}$$

The diagonal term of the fth line of matrix $\mathbb{N}$ reads

$$\mathbb{N}_{ff} = \alpha_c A_c^i \boldsymbol{K}_{ii}^c A_c^i.$$

There are two non-zero extra-diagonal terms that come from the contribution of the sub-cell c. If we note $g \equiv (c,k)$, for $k \neq i$, these two terms write under the form

$$\mathbb{N}_{fg} = \alpha_c A_c^i \boldsymbol{K}_{ik}^c A_c^k.$$

The matrix $\mathbb{S}$ has only one non-zero term its fth line

$$\mathbb{S}_{fc} = \alpha_c \sum_{k=1}^{3} A_c^i \boldsymbol{K}_{ik}^c A_c^k.$$

Finally, the fth component of vector $\boldsymbol{B}$ is $\boldsymbol{B}_f = -A_c^i q^*$.

Robin boundary condition

On the boundary sub-face $f \equiv (c,i)$, the condition $\alpha \bar{T}_c^i + \beta q_c^i = q_R^*$ is prescribed. Let us multiply this equation by A_c^i and replace q_c^i by its expression (1.51) to obtain

$$\alpha A_c^i \bar{T}_c^i - \beta \alpha_c A_c^i \sum_{k=1}^{3} \boldsymbol{K}_{ik}^c A_c^k (\bar{T}_c^k - T_c) = A_c^i q_R^* \tag{1.74}$$

The diagonal term of matrix $\mathbb{N}$ reads

$$\mathbb{N}_{ff} = \beta \alpha_c A_c^i \boldsymbol{K}_{ii}^c A_c^i - \alpha A_c^i.$$

This matrix has once again two non-zero extra-diagonal terms coming from the contribution of the sub-cell c. Denoting $g \equiv (c,k)$, for $k \neq i$, these two non-zero terms write

$$\mathbb{N}_{fg} = \beta \alpha_c A_c^i \boldsymbol{K}_{ik}^c A_c^k.$$

The non-zero term of the fth line of matrix $\mathbb{S}$ is given by

$$\mathbb{S}_{fc} = \beta \alpha_c \sum_{k=1}^{3} A_c^i \boldsymbol{K}_{ik}^c A_c^k.$$

Finally, the fth component of vector $\boldsymbol{B}$ is $\boldsymbol{B}_f = -A_c^i q_R^*$.

Let us remark that the Dirichlet boundary condition is recovered for $\alpha = 1$, $\beta = 0$ and $q_R^* = T^*$ whereas, the Neumann boundary condition corresponds to the case $\alpha = 0$, $\beta = 1$ and $q_R^* = q^*$.

Table 1.1: Statistics about the size of the local node-based systems for a sequence of refined tetrahedral grids.

Number of nodes	Size of the global system	Minimum size	Maximum size	Mean size
189	902	7	66	21
1219	6623	7	96	27
2447	13549	7	102	28
6543	37648	5	90	30

Contribution to the global diffusion matrix

We achieve the discretization of the boundary conditions by listing the modifications that we have to take into account in the assembling of the global diffusion matrix. Solving the local system (1.71), which relates the sub-face temperatures and the cell temperatures, yields the following expression of the sub-face temperature vector

$$\bar{\boldsymbol{T}} = \mathbb{N}^{-1}\mathbb{S}\boldsymbol{T} + \mathbb{N}^{-1}\boldsymbol{B}, \tag{1.75}$$

where the modifications inherent to matrices $\mathbb{N}$, $\mathbb{S}$ and vector $\boldsymbol{B}$ have been detailed in the previous paragraphs. The above expression of the sub-face temperature vector, $\bar{\boldsymbol{T}}$, turns the contribution of the sub-cell ω_{pc} to the diffusion flux, Q_{pc}, into

$$Q_{pc} = -\sum_{d\in\mathcal{C}(p)} \mathbb{G}^p_{cd}(T_d - T_c) - \left(\widetilde{\mathbb{S}}^t\mathbb{N}^{-1}\boldsymbol{B}\right)_c, \tag{1.76}$$

where the effective conductivity tensor, $\mathbb{G}^p$, is defined by (1.58). Finally, the global linear system corresponding to our finite volume scheme becomes

$$\mathbb{M}\mathbb{C}_v\frac{\mathsf{d}\boldsymbol{\mathcal{T}}}{\mathsf{d}t} + \mathbb{D}\boldsymbol{\mathcal{T}} = \mathbb{M}\boldsymbol{\mathcal{R}} + \boldsymbol{\Sigma}, \tag{1.77}$$

where $\boldsymbol{\Sigma}$ is the vector containing the boundary condition contributions, whose the cth entry is given by $\boldsymbol{\Sigma}_c = \left(\widetilde{\mathbb{S}}^t\mathbb{N}^{-1}\boldsymbol{B}\right)_c$. The definition of the other matrices and vectors of the above system remain unchanged.

Size of the local node-based systems

In order to build the global system we need to solve local nodal systems. The size of these systems, which remains small compared to the size of the global linear system, depends on the number of faces impinging at a node. For instance, in a Cartesian structured grid the size of these systems is constant and equal to 12x12 since at a given node the number of impinging faces is equal to 12. In an unstructured tetrahedral grid the size of these systems may vary a lot. To illustrate this point, we have counted the minimum and maximum size of these local node-based linear systems for a sequence of refined tetrahedral grids of a truncated sphere, refer to Figure 1.19(f). We observe in Table 1.1 that the size of these systems remains small compared to the size of the global system.

1.4.8 Volume weight ω_{pc} computation

Two-dimensional geometry

In this paragraph, we aim at deriving practical formulas to compute the volume weight, w_{pc}, present in the flux approximation (1.27). To begin with, let us consider a triangular cell, ω_c,

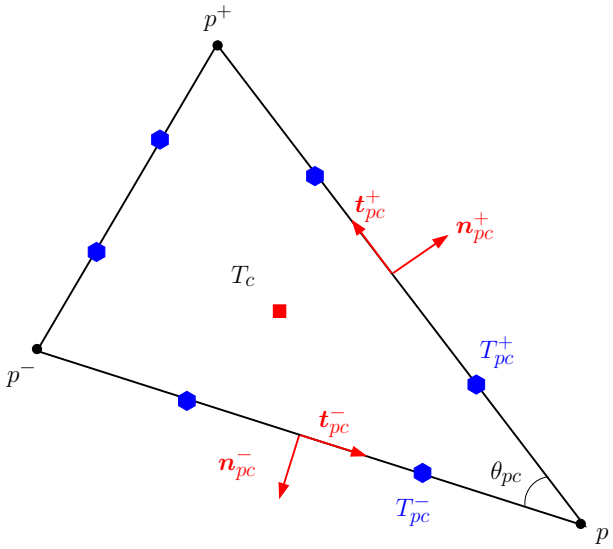


Figure 1.8: Notation for a triangular cell. Half-edge degrees of freedom are displayed in blue color.

characterized by its counterclockwise ordered vertices p^-, p and p^+, refer to Figure 1.8. **We state that the flux approximation** (1.27) **preserves linear fields over triangular cells provided that the volume weight is such that**

$$w^{\text{tri}}_{pc} = \frac{1}{3} \mid \omega_c \mid . \tag{1.78}$$

To prove this result, let us consider $T_h = T_h(\boldsymbol{x})$ a piecewise linear approximation of the temperature field, *i.e.*,

$$T_h(\boldsymbol{x}) = T_c + (\boldsymbol{\nabla} T)_c \cdot (\boldsymbol{x} - \boldsymbol{x}_c), \quad \forall \boldsymbol{x} \in \omega_c. \tag{1.79}$$

Here, $\boldsymbol{x}_c = \frac{1}{3}(\boldsymbol{x}_{p^-} + \boldsymbol{x}_p + \boldsymbol{x}_{p^+})$ is the centroid of ω_c and $T_c = T_h(\boldsymbol{x}_c)$ denotes the mean temperature of the cell. In addition, $(\boldsymbol{\nabla} T)_c$ corresponds to the uniform temperature gradient of the cell. Using the piecewise constant approximation of the conductivity tensor, K_c, this gradient is rewritten $(\boldsymbol{\nabla} T)_c = -\mathsf{K}_c^{-1} \boldsymbol{q}_c$, where $\boldsymbol{q}_c$ is the piecewise constant approximation of the flux. With this notation, (1.79) transforms into

$$T_h(\boldsymbol{x}) = T_c - \mathsf{K}_c^{-1} \boldsymbol{q}_c \cdot (\boldsymbol{x} - \boldsymbol{x}_c), \quad \forall \boldsymbol{x} \in \omega_c. \tag{1.80}$$

Expressing the two vectors $\boldsymbol{q}_c$ and $(\boldsymbol{x} - \boldsymbol{x}_c)$ in terms of their half-edge normal components by means of (1.10) yields

$$T_h(\boldsymbol{x}) = T_c - \mathsf{K}_{pc}^{-1} \begin{pmatrix} q^-_{pc} \\ q^+_{pc} \end{pmatrix} \cdot \begin{bmatrix} (\boldsymbol{x} - \boldsymbol{x}_c)^-_{pc} \\ (\boldsymbol{x} - \boldsymbol{x}_c)^+_{pc} \end{bmatrix}, \quad \forall \boldsymbol{x} \in \omega_c, \tag{1.81}$$

where $\mathsf{K}_{pc} = \mathsf{J}^t_{pc} \mathsf{K}_c \mathsf{J}_{pc}$. Since this equation holds for all points in ω_c, we apply it to $\boldsymbol{x}^-_{pc}$ and $\boldsymbol{x}^+_{pc}$ given by

$$\boldsymbol{x}^-_{pc} = \frac{2\boldsymbol{x}_p + \boldsymbol{x}_{p^-}}{3}, \quad \boldsymbol{x}^+_{pc} = \frac{2\boldsymbol{x}_p + \boldsymbol{x}_{p^+}}{3}. \tag{1.82}$$

This results in

$$T_h(\boldsymbol{x}^\pm_{pc}) - T_c = -\mathsf{K}_{pc}^{-1} \begin{pmatrix} q^-_{pc} \\ q^+_{pc} \end{pmatrix} \cdot \begin{bmatrix} (\boldsymbol{x}^\pm_{pc} - \boldsymbol{x}_c)^-_{pc} \\ (\boldsymbol{x}^\pm_{pc} - \boldsymbol{x}_c)^+_{pc} \end{bmatrix}.$$

Knowing that

$$\boldsymbol{x}_{pc}^{-} - \boldsymbol{x}_c = \frac{1}{3}(\boldsymbol{x}_p - \boldsymbol{x}_{p^+}) = -\frac{2}{3} l_{pc}^{+} \boldsymbol{t}_{pc}^{+},$$
$$\boldsymbol{x}_{pc}^{+} - \boldsymbol{x}_c = \frac{1}{3}(\boldsymbol{x}_p - \boldsymbol{x}_{p^-}) = \frac{2}{3} l_{pc}^{-} \boldsymbol{t}_{pc}^{-},$$

where $\boldsymbol{t}_{pc}^{\pm}$ are the half-edge unit tangent vectors such that $\boldsymbol{n}_{pc}^{\pm} \times \boldsymbol{t}_{pc}^{\pm} = \boldsymbol{e}_z$, refer to Figure 1.8, using $\boldsymbol{t}_{pc}^{-} \cdot \boldsymbol{n}_{pc}^{+} = \sin\theta_{pc}$ and $\boldsymbol{t}_{pc}^{+} \cdot \boldsymbol{n}_{pc}^{-} = -\sin\theta_{pc}$ leads to

$$T_h(\boldsymbol{x}_{pc}^{-}) - T_c = -\frac{2}{3} l_{pc}^{+} \sin\theta_{pc} \mathsf{K}_{pc}^{-1} \begin{pmatrix} q_{pc}^{-} \\ q_{pc}^{+} \end{pmatrix} \cdot \begin{pmatrix} 1 \\ 0 \end{pmatrix},$$
$$T_h(\boldsymbol{x}_{pc}^{+}) - T_c = -\frac{2}{3} l_{pc}^{-} \sin\theta_{pc} \mathsf{K}_{pc}^{-1} \begin{pmatrix} q_{pc}^{-} \\ q_{pc}^{+} \end{pmatrix} \cdot \begin{pmatrix} 0 \\ 1 \end{pmatrix}.$$

Rearranging the above equations allows to express the half-edge normal components of the flux as

$$\begin{pmatrix} q_{pc}^{-} \\ q_{pc}^{+} \end{pmatrix} = -\frac{3}{2 l_{pc}^{-} l_{pc}^{+} \sin\theta_{pc}} \mathsf{K}_{pc} \begin{bmatrix} l_{pc}^{-}(T_h(\boldsymbol{x}_{pc}^{-}) - T_c) \\ l_{pc}^{+}(T_h(\boldsymbol{x}_{pc}^{+}) - T_c) \end{bmatrix}. \tag{1.83}$$

We have obtained an expression of the half-edge fluxes which is exact for a linear approximation of the temperature field over a triangular cell. The comparison between this formula and the general formula obtained previously shows that the volume weight is given by $w_{pc} = \frac{2}{3} l_{pc}^{-} l_{pc}^{+} \sin\theta_{pc}$, which is nothing but one third of the cell volume. In addition, this comparison reveals that the piecewise constant half-edge approximations of the temperature have a clear geometrical interpretation since $T_{pc}^{\pm} = T_h(\boldsymbol{x}_{pc}^{\pm})$, refer to Figure 1.8.

Having defined the volume weight for triangular cells, we conclude this paragraph by giving some indications about the volume weight definition for other types of cells. For quadrangular cells, according to [72], a reasonable choice is to set

$$w_{pc}^{\text{quad}} = l_{pc}^{-} l_{pc}^{+} \sin\theta_{pc}. \tag{1.84}$$

This results in a corner volume equal to the half of the area of the triangle formed by points p^-, p and p^+, refer to Fig 1.8. Unfortunately this choice does not allow to preserve linear solution on quadrangular grids, except on grids made of parallelograms, refer to [90]. However, the numerical results obtained on quadrangular grids with this choice appeared to be quite satisfactory as we shall see in the section devoted to the numerical results. For general polygonal cells, two possible choices are obtained setting

$$w_{pc}^{\text{poly1}} = \frac{1}{\mid \mathcal{P}(c) \mid} \mid \omega_c \mid, \quad w_{pc}^{\text{poly2}} = \mid \omega_{pc} \mid, \tag{1.85}$$

where $\mid \mathcal{P}(c) \mid$ is the total number of sub-cells in cell c. Since the behavior of the numerical method will not be assessed on general polygonal grids, we do not pursue investigations about an optimal choice of the volume weight for polygonal cell.

Three-dimensional geometry

We show that the sub-face normal fluxes approximation given by (1.49) preserves linear temperature fields over tetrahedral cells provided that the corner volume w_{pc} is defined by $w_{pc} = \frac{1}{4} \mid \omega_c \mid$. To demonstrate this result, let us consider a generic tetrahedron, ω_c, over which the temperature field, $T = T(\boldsymbol{x})$, is linear with respect to the space variable $\boldsymbol{x}$. The vertices of this tetrahedron

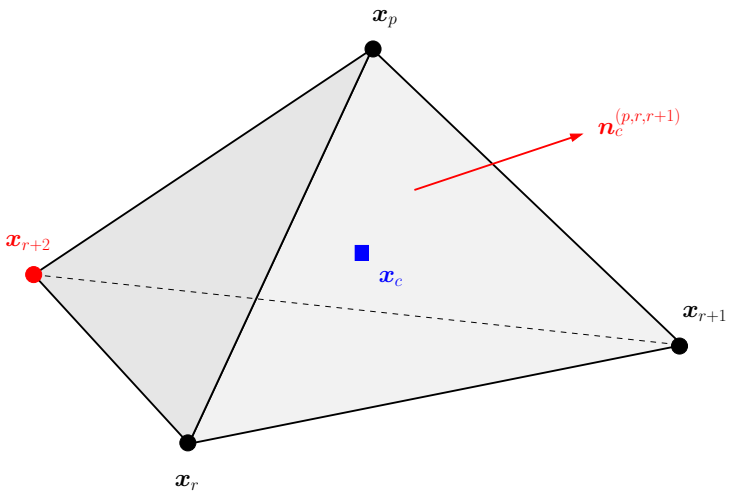


Figure 1.9: Generic tetrahedron with vertices $(\boldsymbol{x}_p, \boldsymbol{x}_r, \boldsymbol{x}_{r+1}, \boldsymbol{x}_{r+4})$ and centroid $\boldsymbol{x}_c = \frac{1}{4}(\boldsymbol{x}_p + \boldsymbol{x}_r + \boldsymbol{x}_{r+1} + \boldsymbol{x}_{r+2})$.

are denoted respectively by p, r, $r+1$ and $r+2$, refer to Figure 1.9. The temperatures at these vertices are T_p, T_r, T_{r+1} and T_{r+2}. They coincide with the point-wise values of the linear temperature field. The constant value of the conductivity tensor over ω_c is $\mathbb{K}_c$. The heat flux is the constant vector $\boldsymbol{q}_c = -\mathbb{K}_c \nabla T$, which satisfies the identity

$$\boldsymbol{q}_c = -\frac{1}{|\,\omega_c\,|} \int_{\omega_c} \mathbb{K}_c \nabla T \, \mathrm{d}v.$$

Utilizing the divergence formula in the above equation turns it into

$$\boldsymbol{q}_c = -\frac{1}{|\,\omega_c\,|} \int_{\partial\omega_c} \mathbb{K}_c T \boldsymbol{n} \, \mathrm{d}s.$$

Now, expanding the surface integral over the triangular faces of the tetrahedral cell yields

$$\boldsymbol{q}_c = -\frac{1}{|\,\omega_c\,|} \sum_{f \in \mathcal{F}(c)} \mathbb{K}_c A_c^f \boldsymbol{n}_c^f \widetilde{T}_c^f,$$

where A_c^f is the area of face f, $\boldsymbol{n}_c^f$ is the unit outward normal to face f and $\widetilde{T}_c^f$ is the face-averaged value of the temperature. This face-averaged temperature is computed by means of

$$\widetilde{T}_c^f = \frac{1}{3} \sum_{s \in \mathcal{P}(c,f)} T_s, \tag{1.86}$$

where $\mathcal{P}(c,f)$ is the set of points of cell c belonging to face f. Before proceeding any further, we explicit our notations to highlight the role played by point p. Each triangular face is characterized by the set of its three vertices. The three faces impinging at point p are $(p, r+k, r+k+1)$ for $k = 1 \ldots 3$ and assuming a cyclic indexing. Their area, unit outward normal and face-averaged temperature are denoted respectively by A_c^k, $\boldsymbol{n}_c^k$ and $\widetilde{T}_c^k$. The remaining face, which is opposite to point p, is $(r, r+1, r+2)$. Its area, unit outward normal and face-averaged temperature are denoted respectively by A_c^r, $\boldsymbol{n}_c^r$ and $\widetilde{T}_c^r$. With the above notations, the heat flux expression becomes

$$\boldsymbol{q}_c = -\frac{1}{|\,\omega_c\,|} \left(\sum_{k=1}^{3} \mathbb{K}_c A_c^k \boldsymbol{n}_c^k \widetilde{T}_c^k + \mathbb{K}_c A_c^r \boldsymbol{n}_c^r \widetilde{T}_c^r \right).$$

Knowing that $A_c^r \boldsymbol{n}_c^r = -\sum_{k=1}^{3} A_c^k \boldsymbol{n}_c^k$ leads to rewrite the above flux expression as

$$\boldsymbol{q}_c = -\frac{1}{\mid \omega_c \mid} \sum_{k=1}^{3} \mathbb{K}_c A_c^k \boldsymbol{n}_c^k \left(\widetilde{T}_c^k - \widetilde{T}_c^r \right).$$

Substituting the expression of the face-averaged temperatures (1.86) in terms of the point temperatures yields

$$\boldsymbol{q}_c = -\frac{1}{3 \mid \omega_c \mid} \sum_{k=1}^{3} \mathbb{K}_c A_c^k \boldsymbol{n}_c^k \left(T_p - T_{r+k+2} \right).$$

Finally, to eliminate the point temperatures in the above expression, we introduce the cell-averaged temperature

$$T_c = \frac{1}{4}(T_p + T_{r+k} + T_{r+k+1} + T_{r+k+2}).$$

Due to the cyclic numbering, this expression is valid for $k = 1 \dots 3$. Expressing T_{r+k+2} in terms of the cell-averaged temperature and the remaining point temperatures leads to write

$$T_p - T_{r+k+2} = 4 \left(\bar{T}_c^{(p,r+k,r+k+1)} - T_c \right),$$

where $\bar{T}_c^{(p,r+k,r+k+1)}$ is the sub-face temperature given by

$$\bar{T}_c^{(p,r+k,r+k+1)} = \frac{1}{4}(2T_p + T_{r+k} + T_{r+k+1}).$$

Since the temperature field is linear with respect to the space variable, we point out that the above expression is the exact value of the temperature field taken at the point $\bar{\boldsymbol{x}}_c^{(p,r+k,r+k+1)}$ located on the triangular face $(p, r+k, r+k+1)$, refer to Figure 1.10, and defined by

$$\bar{\boldsymbol{x}}_c^{(p,r+k,r+k+1)} = \frac{1}{4}(2\boldsymbol{x}_p + \boldsymbol{x}_{r+k} + \boldsymbol{x}_{r+k+1}).$$

Observing the triangular face displayed in Figure 1.10, we note that this point is the midpoint of the median segment coming from vertex p.
Gathering the above results allows to rewrite the expression of the heat flux as

$$\boldsymbol{q}_c = -\frac{4}{3 \mid \omega_c \mid} \sum_{k=1}^{3} \mathbb{K}_c A_c^k \boldsymbol{n}_c^k \left(\bar{T}_c^{(p,r+k,r+k+1)} - T_c \right).$$

It remains to simplify the above expression of the heat flux by employing notations related to the sub-face associated to point p and face k, displayed in blue color in Figure 1.10. It is clear that the area of the sub-face, A_{pc}^k, is equal to one-third of the face area, A_c^k, and thus $A_c^k = 3A_{pc}^k$. In addition, the unit outward normal to the sub-face, $\boldsymbol{n}_{pc}^k$, coincides with the unit outward normal to the face, $\boldsymbol{n}_c^k$. Finally, defining the sub-face temperature $T_{pc}^k \equiv \bar{T}_c^{(p,r+k,r+k+1)}$ leads to write the heat flux

$$\boldsymbol{q}_c = -\frac{4}{\mid \omega_c \mid} \sum_{k=1}^{3} \mathbb{K}_c A_{pc}^k \boldsymbol{n}_{pc}^k \left(T_{pc}^k - T_c \right).$$

We dot-multiply the heat flux by the unit normal $\boldsymbol{n}_{pc}^l$ to obtain the normal component of the heat flux related to the sub-face l

$$q_{pc}^l = -\frac{4}{\mid \omega_c \mid} \sum_{k=1}^{3} \left(\mathbb{K}_c \boldsymbol{n}_{pc}^k \right) \cdot \boldsymbol{n}_{pc}^l A_{pc}^k \left(T_{pc}^k - T_c \right).$$

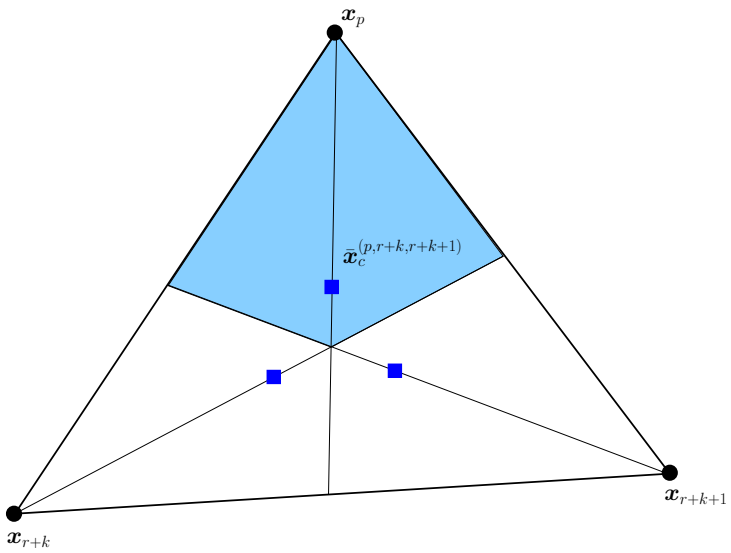


Figure 1.10: Triangular face $(p, r+k, r+k+1)$ related to the tetrahedron displayed in Figure 1.9. The sub-face related to point p has been colored in blue. The three degrees of freedom related to the sub-face temperatures are plotted by means of blue squares.

This formula coincides with the one derived from the variational formulation, refer to Eq. (1.50), provided that the volume weight satisfies $w_{pc} = \frac{1}{4} \mid \omega_c \mid$, which ends the proof.

This shows that the flux approximation (1.50) is exact for linear temperature fields with respect to the space variable. In addition, the sub-face temperatures coincide with the point-wise values taken by the linear temperature field at the midpoint of the median segment coming from each vertex of a triangular face. It is worth pointing out that this results has been already obtained in [86] using a more theoretical framework.

Finally, for general polyhedral cells, the corner volume weight related to sub-cell ω_{pc} is defined by

$$w_{pc} = \frac{1}{\mathsf{P}_c} \mid \omega_c \mid, \tag{1.87}$$

where $\mathsf{P}_c = \mid \mathcal{P}(c) \mid$ is the number of vertices of cell ω_c.

1.5 Time discretization

In this section, we briefly describe the time discretization of the system (1.77). We restrict the presentation to the case of a linear heat equation knowing that in the non linear case the interested reader might refer to [90]. First, let us prescribe the initial condition $\boldsymbol{\mathcal{T}}(0) = \boldsymbol{\mathcal{T}}^0$, where $\boldsymbol{\mathcal{T}}^0$ is the vector of the cell-averaged initial condition. We solve the system over the time interval $[0, \mathfrak{T}]$ using the subdivision

$$0 = t^0 < t^1 < \cdots < t^n < t^{n+1} < \cdots < t^N = \mathfrak{T}.$$

The time step is denoted by $\Delta t^n = t^{n+1} - t^n$. The time approximation of a quantity at time t^n is denoted using the superscript n, for instance $\boldsymbol{\mathcal{T}}^n = \boldsymbol{\mathcal{T}}(t^n)$. Knowing that an explicit time discretization of the diffusion operator necessitates a stability constraint on the time step

which is quadratic with respect to the smallest cell size, we prefer to use an implicit time discretization. Further, we assume that the heat capacity and the conductivity tensor do not depend on temperature. Integrating (1.77) over $[t^n, t^{n+1}]$ yields the first-order in time implicit discrete scheme

$$\mathbb{M}\mathbb{C}_v \frac{\boldsymbol{\mathcal{T}}^{\boldsymbol{n+1}} - \boldsymbol{\mathcal{T}}^{\boldsymbol{n}}}{\Delta t^n} + \mathbb{D}\boldsymbol{\mathcal{T}}^{n+1} = \mathbb{M}\boldsymbol{\mathcal{R}}^n + \boldsymbol{\Sigma}^n. \tag{1.88}$$

We recall that $\boldsymbol{\mathcal{R}}^n$ is the source term vector and $\boldsymbol{\Sigma}^n$ is the boundary vector at time t^n. $\mathbb{M}$ and $\mathbb{C}_v$ are the diagonal matrices whose entries are respectively the cell mass m_c and the cell heat capacity C_{vc}. $\mathbb{D}$ is the global diffusion matrix. The updated cell-centered temperatures are obtained by solving the following linear system

$$\left(\frac{\mathbb{M}\mathbb{C}_v}{\Delta t^n} + \mathbb{D}\right)\boldsymbol{\mathcal{T}}^{n+1} = \frac{\mathbb{M}\mathbb{C}_v}{\Delta t^n}\boldsymbol{\mathcal{T}}^n + \mathbb{M}\boldsymbol{\mathcal{R}}^n + \boldsymbol{\Sigma}^n. \tag{1.89}$$

Let us recall that $\mathbb{D}$ is positive semi-definite. Knowing that $\mathbb{M}\mathbb{C}_v$ is a positive diagonal matrix, we deduce that the matrix $\frac{\mathbb{M}\mathbb{C}_v}{\Delta t^n} + \mathbb{D}$ is positive definite. Thus, the linear system (1.89) always admits a unique solution. Finally, in the absence of source term and assuming periodic or homogeneous boundary conditions, we observe that the above implicit time discretization is stable with respect to the discrete weighted L^2 norm defined by

$$\|\boldsymbol{\mathcal{T}}\|_{w2}^2 = (\mathbb{M}\mathbb{C}_v\boldsymbol{\mathcal{T}} \cdot \boldsymbol{\mathcal{T}}),$$

where $\boldsymbol{\mathcal{T}}$ is a vector of size $\mathsf{C}_{\mathcal{D}}$. To prove this result, we dot-multiply (1.89) by $\boldsymbol{\mathcal{T}}^{n+1}$ and obtain

$$(\mathbb{M}\mathbb{C}_v\boldsymbol{\mathcal{T}}^{n+1} \cdot \boldsymbol{\mathcal{T}}^{n+1}) - (\mathbb{M}\mathbb{C}_v\boldsymbol{\mathcal{T}}^{n} \cdot \boldsymbol{\mathcal{T}}^{n+1}) = -\Delta t^n(\mathbb{D}\boldsymbol{\mathcal{T}}^{n+1} \cdot \boldsymbol{\mathcal{T}}^{n+1}).$$

Due to the positive definiteness of matrix $\mathbb{D}$ the right-hand side of the above equation is negative, hence

$$(\mathbb{M}\mathbb{C}_v\boldsymbol{\mathcal{T}}^{n+1} \cdot \boldsymbol{\mathcal{T}}^{n+1}) \leq (\mathbb{M}\mathbb{C}_v\boldsymbol{\mathcal{T}}^{n} \cdot \boldsymbol{\mathcal{T}}^{n+1}).$$

Employing Cauchy-Schwarz inequality in the right-hand side of the above inequality yields

$$(\mathbb{M}\mathbb{C}_v\boldsymbol{\mathcal{T}}^{n} \cdot \boldsymbol{\mathcal{T}}^{n+1}) \leq \|\boldsymbol{\mathcal{T}}^{n}\|_{w2}\|\boldsymbol{\mathcal{T}}^{n+1}\|_{w2}.$$

Gathering the above results leads to

$$\|\boldsymbol{\mathcal{T}}^{n+1}\|_{w2} \leq \|\boldsymbol{\mathcal{T}}^{n}\|_{w2},$$

which ends the proof.

Comment 11: *The computation of the numerical solution requires to solve the sparse linear system* (1.89)*. This is achieved by employing the localized ILU(0) Preconditioned BiCGStab algorithm, refer to [127, 95]. The parallel implementation of this algorithm and its efficiency are discussed in Section 1.6. Knowing that the matrices encountered in this work are all symmetric, we could have employed a classical conjugate gradient method to solve the corresponding linear system. However, our numerical scheme being able to cope with non-symmetric diffusion equations, refer to [90], we have chosen to implement a more general solver to handle these problems.*

Comment 12: *Let us give more details about the Cauchy-Schwarz inequality we used in this section.*

Let $\mathcal{A}$ and $\mathcal{B}$ be two vectors of size $\mathsf{C}_{\mathcal{D}}$, and $t \in \mathfrak{R}$. We compute $\|\mathcal{A}+t\mathcal{B}\|_{w2}^2$.

$$\begin{aligned}\|\mathcal{A}+t\mathcal{B}\|_{w2}^2 &= (\mathbb{MC}_v\mathcal{A}\cdot\mathcal{A}) + t(\mathbb{MC}_v\mathcal{A}\cdot\mathcal{B}) + t(\mathbb{MC}_v\mathcal{B}\cdot\mathcal{A}) + t^2(\mathbb{MC}_v\mathcal{B}\cdot\mathcal{B}) \\ &= \|\mathcal{A}\|_{w2}^2 + t(\mathbb{MC}_v\mathcal{A}\cdot\mathcal{B}) + t(\mathbb{MC}_v\mathcal{B}\cdot\mathcal{A}) + t^2\|\mathcal{B}\|_{w2}^2 \\ &= \|\mathcal{A}\|_{w2}^2 + 2t\,(\mathbb{MC}_v\mathcal{A}\cdot\mathcal{B}) + t^2\|\mathcal{B}\|_{w2}^2, \text{ because } \mathbb{MC}_v \text{ is symmetric.}\end{aligned}$$

The right-hand side of this equation is a non-negative quadratic polynomial with respect to t, thus its discriminant is non-negative

$$(\mathbb{MC}_v\mathcal{A}\cdot\mathcal{B})^2 - \|\mathcal{A}\|_{w2}^2\|\mathcal{B}\|_{w2}^2 \leq 0.$$

This leads to the final result

$$|\ (\mathbb{MC}_v\mathcal{A}\cdot\mathcal{B})\ | \leq \|\mathcal{A}\|_{w2}\|\mathcal{B}\|_{w2},$$

which is the property we have used in this section.

1.6 Parallelization

When dealing with three-dimensional grids, the computational power needed to solve the problems grows quickly. In fact two problems occur, the memory consumption becomes higher and the computational time becomes longer. These two problems can be overcome with the parallelization of the scheme. The goal is to split the global problem into smaller problems that will run concurrently on different processors. In the distributed memory case, the more processors we add, the more memory we get. On the other hand communications are then needed between the processors to solve the global problem.

First, we have a look at the implementation of the sequential algorithm and identify the parts we need to parallelize in priority. Then, we describe the partitioning step and the communication process. Finally, we present an experimental study to assess the efficiency of our parallelization scheme.

1.6.1 Analysis of the problem

The sequential algorithm can be divided in two steps: assembling the matrix and solving the system.

To build the global matrix we have to solve a local linear system at each vertex of the mesh. This is a Vertex-Centered approach. The solving step is performed through the use of iterative Krylov methods such as BiCGStab [127]. These methods need to perform matrix-vector multiplications and dot products. Here, the matrix involved associates a cell with its neighboring cells; hence the solving step is a Cell-Centered approach.

In a parallel computation we would like to split the problem into equally balanced sub-problems, this is called partitioning. The problem we have to face with our algorithm is that the optimal partition for the Vertex-Centered approach is different from the optimal partition for the Cell-Centered one. We thus have to make a choice, optimizing one step while sacrificing the other.

If we have a look at the sequential timings we can see that the construction process takes approximately 10% of the overall time and the solving step takes 90% of the time. The Amdahl's law [14] tells us that we have more to gain by optimizing the more time consuming step, in our case the solving step, so we will focus on a Cell-Centered partitioning.

1.6.2 Partitioning and communications

The main kernel of iterative Krylov methods is a matrix-vector multiplication. This is why we have to efficiently parallelize the matrix-vector product $\boldsymbol{Y} = \mathbb{A}\boldsymbol{X}$. We assume that we have a partitioning of our problem, it means that every processor owns a specific subset of the global problem. If I is a processor it will only know the subset $\boldsymbol{X}_I$ of the vector $\boldsymbol{X}$ and the subset $\boldsymbol{Y}_I$ of the vector $\boldsymbol{Y}$. This results in the following decomposition for the vectors $\boldsymbol{X}$ and $\boldsymbol{Y}$:

$$\boldsymbol{Y} = \begin{bmatrix} \boldsymbol{Y}_1 \\ \vdots \\ \boldsymbol{Y}_I \\ \vdots \\ \boldsymbol{Y}_N \end{bmatrix}, \quad \boldsymbol{X} = \begin{bmatrix} \boldsymbol{X}_1 \\ \vdots \\ \boldsymbol{X}_I \\ \vdots \\ \boldsymbol{X}_N \end{bmatrix}.$$

Similarly, we decompose the matrix $\mathbb{A}$ and express it in terms of a block matrix with $\mathbb{A}_{IJ}$ elements:

$$\mathbb{A} = \begin{bmatrix} \mathbb{A}_{11} & & \cdots & & \mathbb{A}_{1N} \\ & \ddots & & & \\ \vdots & & \mathbb{A}_{IJ} & & \vdots \\ & & & \ddots & \\ \mathbb{A}_{N1} & & \cdots & & \mathbb{A}_{NN} \end{bmatrix}.$$

With these notations the matrix-vector multiplication $\boldsymbol{Y} = \mathbb{A}\boldsymbol{X}$ may be expressed as:

$$\boldsymbol{Y}_I = \sum_{J=1..N} \mathbb{A}_{IJ}\boldsymbol{X}_J, \quad \forall I \in \{1, \ldots, N\}.$$

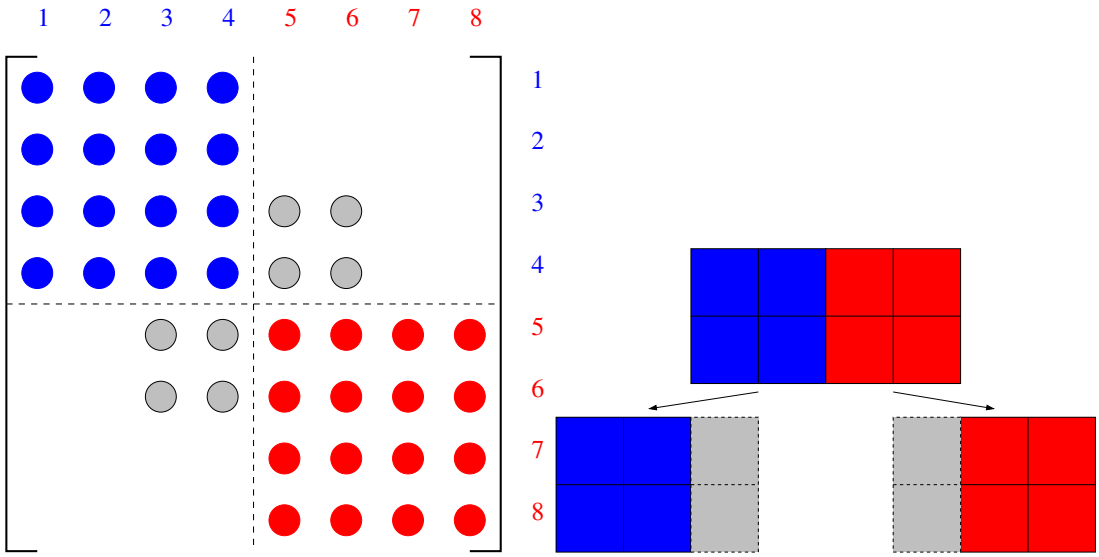
To compute sub-vector $\boldsymbol{Y}_I$ processor I needs to access the bloc matrices $\mathbb{A}_{IJ}$ where $J = 1..N$. More precisely, processor I needs to access all the rows associated to its partition. We say that the matrix is partitioned row-wise. Processor I also potentially needs to know the vectors $\boldsymbol{X}_J$ where $J = 1..N$, which is the whole $\boldsymbol{X}$ vector. As we mentioned before, it only owns $\boldsymbol{X}_I$ vector. To perform the global operation, we thus need to receive the sub-vectors $\boldsymbol{X}_J$ from the processors J ($J \neq I$). As matrix $\mathbb{A}$ is sparse, a block $\boldsymbol{X}_J$ is effectively needed if and only if $\mathbb{A}_{IJ}$ has non zero entries. Furthermore, $\mathbb{A}_{IJ}$ may also be sparse. Thus, only part of the elements of sub-vector $\boldsymbol{X}_J$ may be needed on processor I. We note $\hat{\boldsymbol{X}}_J^I$ the corresponding pruned sub-vector (so-called "overlap") and $\boldsymbol{Y}_I \leftarrow \mathbb{A}_{IJ}\boldsymbol{X}_J$ may be compacted into $\boldsymbol{Y}_I \leftarrow \hat{\mathbb{A}}_{IJ}\hat{\boldsymbol{X}}_J^I$.

The algorithm for the parallel matrix-vector on processor I can then be written as follows:

- For each processor J, send $\hat{\boldsymbol{X}}_I^J$ to processor J,
- Compute $\boldsymbol{Y}_I \leftarrow \mathbb{A}_{II}\boldsymbol{X}_I$,
- For each processor J, receive $\hat{\boldsymbol{X}}_J^I$ from processor J and compute $\boldsymbol{Y}_I \leftarrow \boldsymbol{Y}_I + \hat{\mathbb{A}}_{IJ}\hat{\boldsymbol{X}}_J^I$.

In order to hide the communication process, we consider non-blocking communications, which occur while we compute the local matrix-vector multiplication $\boldsymbol{Y}_I \leftarrow \mathbb{A}_{II}\boldsymbol{X}_I$. If the amount of computation for this operation is big enough the distant sub-vector $\hat{\boldsymbol{X}}_J^I$ may be transferred without impact on the elapsed time.

A simple example of this decomposition is displayed in Figure 1.11(a). In this example, the first



(a) Matrix decomposition. (b) Mesh decomposition.

Figure 1.11: Example of matrix and mesh decomposition on two processors with overlaps construction.

processor in blue owns the first four rows of the matrix and the first six elements of the vector, while the second processor in red owns the last four rows and the last six elements of the vector. On the first processor the overlaps are the element 5 and 6 of the vector, while the overlaps of the second processor consist of the elements 3 and 4. These elements are not computed on the local processor but are received from the other processor. The associated mesh and the corresponding sub-meshes obtained after the decomposition are shown in Figure 1.11(b), the grey cells represent the overlaps. The other parallel operation to perform in our iterative solver is the inner product $p = \boldsymbol{X} \cdot \boldsymbol{Y} = \sum_k \boldsymbol{X}_k \boldsymbol{Y}_k$. With our decomposition it writes $p = \sum_{J=1..N} p_J$ where $p_J = \sum_k \boldsymbol{X}_J^k \boldsymbol{Y}_J^k$ is the local inner product corresponding to the sub-problem J. This is a global operation, every processor has to compute its local inner product and exchange it with all the other processors. This communication cannot be overlapped by computation, this could be a bottleneck in our algorithms.

How should we distribute the rows of the matrix? As we said before we want the load to be balanced between the processors, in our case it is the number of operations in the matrix-vector operation or the number of non-zero elements in the matrix. We also want to overlap the communications with computations, and the communication time depends on the amount of data to exchange. This amounts to reduce the volume of communication between processors. This problem is really complicated, that is why to achieve these goals we use a graph partitioner called Scotch [100]. We process the graph associated to the global matrix with this library which retrieves for each row the partition it belongs to. With these information we can set up the communication scheme explained earlier.

What are the changes needed by the scheme in parallel? When we add an element in the overlap vector we add the corresponding cell in the local sub-mesh. So in every sub-mesh an internal cell is surrounded by all its neighbors, we have all the information to build the row corresponding to this cell in the matrix, so nothing has to be changed in the scheme. The parallelism is only seen in the solving step.

We have implemented these methods in our development code but we can also use the PetSC library [19, 18, 20]. This library implements scalable algorithms to solve scientific applications modeled by partial differential equations. In this library the solving step and the communication process are hidden to the user. The problem we faced is that with the libraries available on our experimental platform we could not use some preconditioners in parallel like ILU(0). Due to that, the iterative method does not converge very well in parallel. This explains why the experiments we ran with PetSC were not very conclusive.

1.6.3 Experiments

In order to quantify the quality of the parallelization we define two metrics: the speedup and the efficiency. If T_p denotes the time needed to solve the problem on p processors the speedup is defined by

$$S(p) = \frac{T_1}{T_p}.$$

This quantity represents how much faster the algorithm is on p processors than in sequential. Ideally on p processors we would like to be p times faster than in sequential, thus the ideal speedup is defined by

$$S_{ideal}(p) = p.$$

An other interesting metric is the efficiency. It is given by

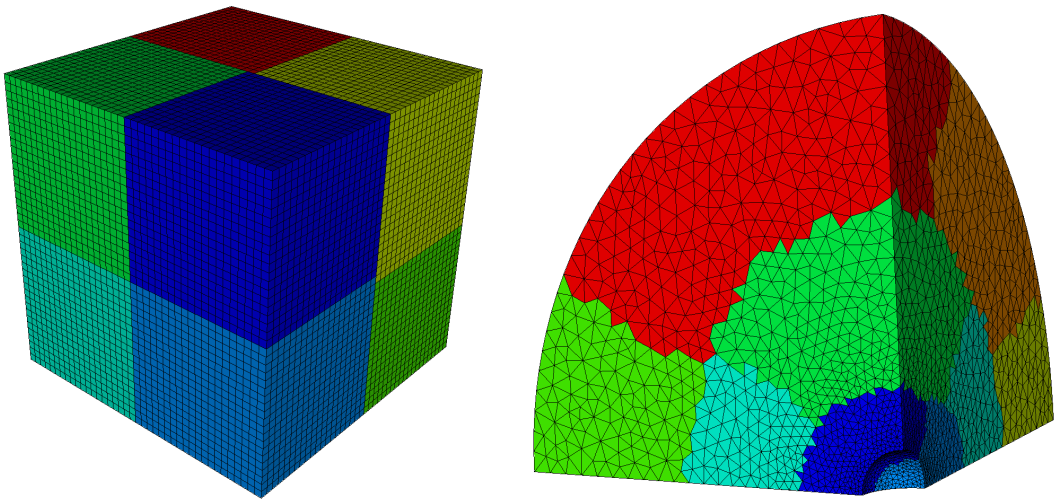
$$E(p) = \frac{T_1}{pT_p}.$$

It assesses how efficiently the processors are used with respect to the ideal case ($E(p) = 1$).

We ran the experiments on the PLAFRIM (IMB/LABRI/INRIA) [2] platform. On each node of this machine we have 2 Quad-core Nehalem Intel Xeon X5550 (8 CPU cores total per node) running at 2,66 GHz. The nodes have 24Gb of RAM (DDR3 1333MHz) and are connected with Infiniband QDR at 40Gb/s. To test the scalability of our method we ran the tests on 1 to 64 CPU cores (using 1 to 8 nodes). We run the speedup tests on two kinds of grids:

- A Cartesian hexahedral grid made of 512 000 cells, 531 441 nodes and with 13 481 272 non-zeros entries in the associated matrix,
- A unstructured tetrahedral grid made of 396 601 cells, 98 218 nodes and with 28 946 047 non-zeros entries in the associated matrix.

We specifically chose these meshes to illustrate the load balancing problem occurring between the matrix construction and the solving step. On the first kind of mesh we have a perfect load balancing while on the second one the load balancing of the construction step can be bad, due to the unstructured feature of the grid. The partitioning of the coarsest versions of these grids are displayed in Figure 1.12(a) for the structured grid and in Figure 1.12(b) for the unstructured grid. On the speedup curve displayed in Figure 1.13 we can see that the more processors we add, the further away from the ideal speedup we get. This highlights two different phenomena. First, in the conjugate gradient method we need to compute some scalar products and vector norms which need collective communications. This kind of communications does not scale very well so the more processors we add the worse it gets. The other phenomenon is that by adding more processors, the local matrices get smaller and we need to communicate more at the same time. So at some point the computation can not overlap the communications anymore and the



(a) Structured hexahedral grid.

(b) Unstructured tetrahedral grid.

Figure 1.12: Partitioning computed with the Scotch library (INRIA) [100] for 8 processors. One distinct color is attributed to each processor.

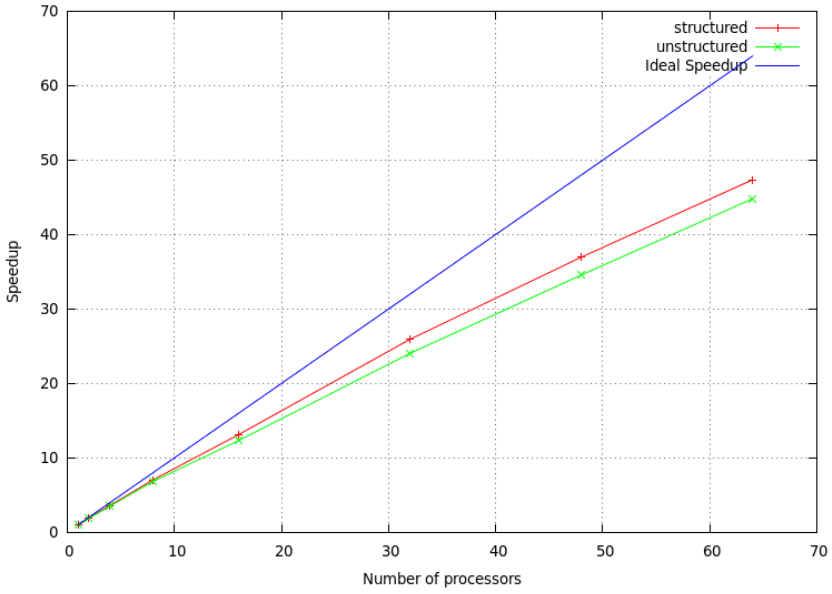


Figure 1.13: Speedup curve for 1 to 64 processors on structured and unstructured 3D meshes.

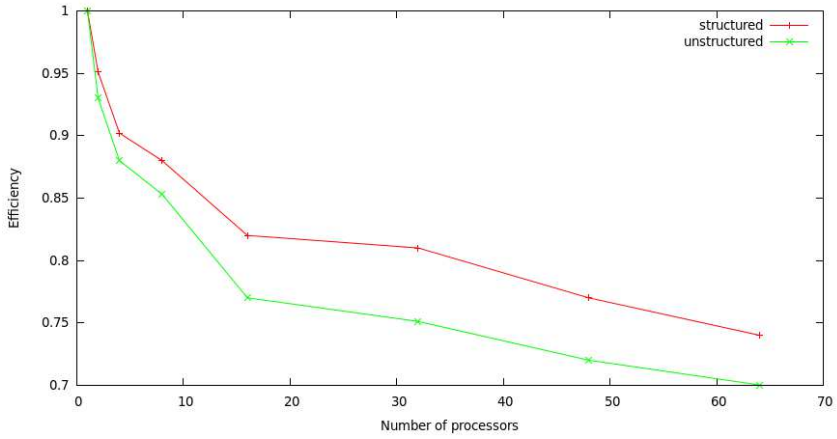


Figure 1.14: Efficiency curve for 1 to 64 processors on structured and unstructured three-dimensional meshes.

speedup gets worse.

On the efficiency curve displayed in Figure 1.14 we can see that from 1 to 8 CPU cores the efficiency quickly drops from 1 to 0.85, then between 8 to 64 CPU cores it decreases more slowly. This reflect the topology of the platform we used for the tests. From 1 to 8 CPU cores we are only using one node. On a single node the communication cost is negligible, so we would expect the efficiency to stay close to 1. The quick drop shows the existence of a bottleneck in the memory usage. This may come from the usage of unstructured methods which uses a lots of memory indirection, some optimization around the matrix numbering should reduce this effect. The decrease in efficiency observed with more than 8 CPU cores is due to the communications between the nodes.

Finally, we can comment the difference between the efficiency obtained on structured and unstructured meshes. We observe the same phenomenon on the two kinds of meshes. We observe that the efficiency is a bit better for the structured meshes. The difference is due to the imbalance in the construction step. This imbalance could be lowered by adding information about the cost of the matrix construction into the graph sent to the partitioner.

To conclude this paragraph we can claim that we developed a parallel implementation of the classical BiCGStab method. This method has some blocking points, the inner products that can not be overlapped by computations, so it is not fully scalable. In [137] the authors present the IBiCGStab method, a modified BiCGStab algorithm with an equivalent numerical stability, in which these blocking points are cured. Only one global synchronization point is needed per iteration instead of four in the original algorithm. This is a more scalable method. In a future work we plan to investigate this modification to improve the efficiency of our implementation.

1.7 Numerical results in two-dimensional geometries

The aim of this section is to assess the robustness and the accuracy of CCLAD scheme against analytical test cases using various types of triangular and quadrangular grids.

1.7.1 Convergence analysis methodology

Let us recall that we are solving the generic diffusion equation

$$\rho C_v \frac{\partial T}{\partial t} - \nabla \cdot (\mathbb{K} \nabla T) = \rho r, \qquad (\boldsymbol{x}, t) \in \mathcal{D} \times [0, \mathcal{T}], \tag{1.90a}$$

$$T(\boldsymbol{x}, 0) = T^0(\boldsymbol{x}), \qquad \boldsymbol{x} \in \mathcal{D}, \tag{1.90b}$$

where $r = r(\boldsymbol{x})$ is a source term. The density and the specific heat capacity are specified such that $\rho = 1$ and $C_v = 1$. The boundary conditions, the source term and the heat conductivity tensor $\mathbb{K}$ will be specified for each test case.
The analytical solutions of all the tests are stationary. Thus, we are going to compute them starting with the initial condition $T^0(\boldsymbol{x}) = 0$ and we run the simulation until the steady state is reached. For these tests, the numerical solutions are obtained solving linear systems by means of the localized ILU(0) Preconditioned BiCGStab algorithm [127, 95]. The relative error tolerance to achieve the convergence is equal to 10^{-16}.

We describe the procedure employed to perform the convergence analysis. First, we define the mesh resolution

$$h = \left(\frac{|\mathcal{D}|}{\mathsf{C}_\mathcal{D}} \right)^{\frac{1}{d}},$$

where $\mathsf{C}_\mathcal{D}$ denotes the number of cells that paved the computational domain and $d = 3$ is the dimension of the space. Let $T = \hat{T}(\boldsymbol{x})$ be the steady analytical solution of the diffusion equation (1.90a). Given a computational grid characterized by h, we denote by $\hat{T}_c^h$ the value of the analytical solution evaluated at the centroid of the cell ω_c, *i.e.*, $\hat{T}_c^h = \hat{T}(\boldsymbol{x}_c)$, where $\boldsymbol{x}_c$ is the cell centroid. If T_c^h denotes the cell averaged temperature computed by the numerical scheme, we define the asymptotic numerical errors based on the discrete L^2 and L^∞ norms

$$E_2^h = \sqrt{\sum_{c=1}^{\mathsf{C}_\mathcal{D}} (T_c^h - \hat{T}_c^h)^2 \mid \omega_c \mid},$$

$$E_\infty^h = \max_{c=1 \ldots \mathsf{C}_\mathcal{D}} \mid T_c^h - \hat{T}_c^h \mid .$$

The asymptotic error for both norms is estimated by

$$E_\alpha^h = C_\alpha h^{q_\alpha} + O(h^{q_\alpha + 1}) \ \text{ for } \ \alpha = 2, \ \infty. \tag{1.91}$$

Here, q_α denotes the order of the truncation error and C_α is the convergence rate-constant which is independent of h. Having computed the asymptotic errors corresponding to two different grids characterized by mesh resolutions h_1 and $h_2 < h_1$, we deduce an estimation of the order of truncation error as

$$q_\alpha = \frac{\log E_\alpha^{h_2} - \log E_\alpha^{h_1}}{\log h_2 - \log h_1}. \tag{1.92}$$

1.7.2 Meshes description

We performed the computations on different kind of meshes. For every mesh category we used five levels of refinement of the meshes from coarse grid to finer grids. Here is a short description of the different meshes used for the numerical tests.

- Meshes made of triangles as shown in Fig 1.15(a). These meshes are composed of 264, 1032, 4178, 16986 and 67548 triangles. There is an interface at $x = \frac{1}{2}$.

- Cartesian meshes as pictured in Fig 1.15(b). The different levels of refinement are made of 10×10, 20×20, 40×40, 80×80 and 160×160 quadrangles.

- Smoothly deformed meshes as shown in Fig 1.15(c). The different levels of refinement are made of 10×10, 20×20, 40×40, 80×80 and 160×160 quadrangles. The deformation is defined by the mapping of the unit square $[0,1]^2$ onto itself:

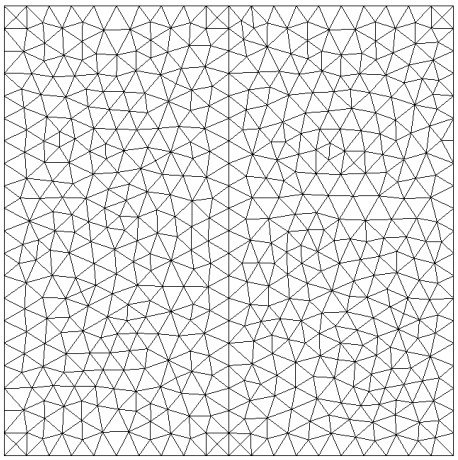
$$\begin{cases} x(\xi,\eta) = \xi + 0.1\sin(2\pi\xi)\sin(2\pi\eta), \\ y(\xi,\eta) = \eta + 0.1\sin(2\pi\xi)\sin(2\pi\eta). \end{cases}$$

- Kershaw type meshes pictured in Fig 1.15(e). The different levels of refinement are made of 12×12, 24×24, 48×48, 96×96 and 192×192 quadrangles.

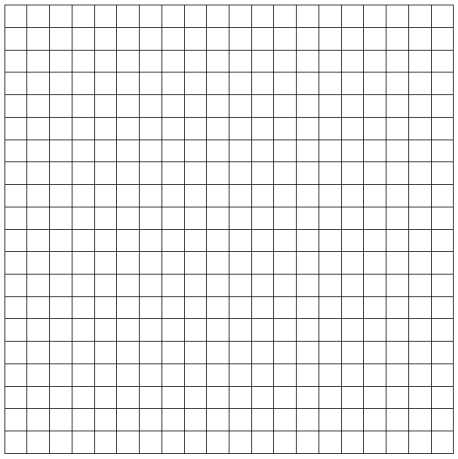
- Randomly perturbed meshes as shown in Fig 1.15(d). The different meshes are made of 10×10, 20×20, 40×40, 80×80 and 160×160 quadrangles. The perturbation is described by the mapping defined on the unit square $[0,1]^2$ by:

$$\begin{cases} x(\xi,\eta) = \xi + 0.2hr_1, \\ y(\xi,\eta) = \eta + 0.2hr_2, \end{cases}$$

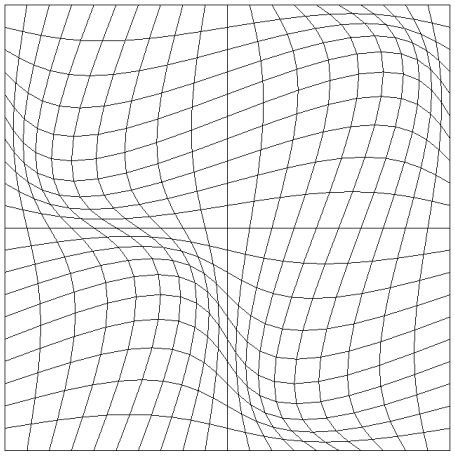
  where r_i are random numbers between -1 and 1, h is the characteristic mesh size. The mapping is not applied on the interface $x = \frac{1}{2}$ in order to preserve it.



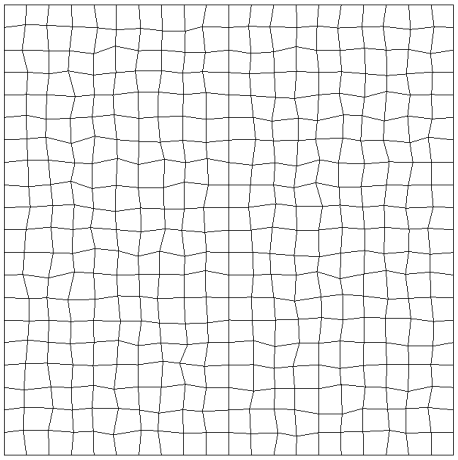
(a) Mesh made of 1032 triangles, the interface at $x = \frac{1}{2}$ is preserved.



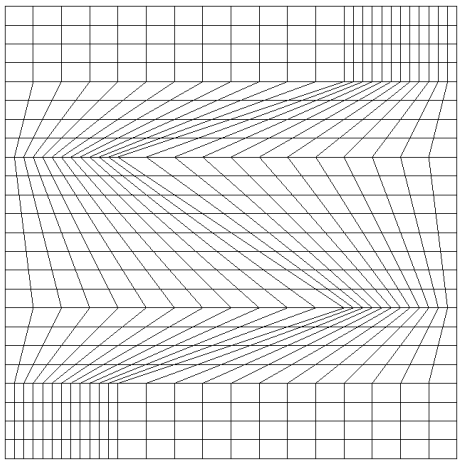
(b) Cartesian mesh made of 20×20 quadrangles.



(c) Smoothly deformed mesh made of 20×20 quadrangles.



(d) Randomly perturbed mesh made of 20×20 quadrangles, the interface at $x = \frac{1}{2}$ is preserved.



(e) Kershaw type mesh made of 24×24 quadrangles.

Figure 1.15: Example of the meshes used for the tests cases. These meshes are the second coarsest meshes used for the scheme order computations.

1.7.3 Piecewise linear problem with discontinuous isotropic conductivity tensor

This problem consists in finding the steady solution of (1.90) with $r = 0$ and an isotropic discontinuous conductivity tensor given by

$$\mathbb{K}(x,y) = \begin{cases} \kappa_l \mathbb{I} & \text{if } 0 \leq x \leq \frac{1}{2}, \\ \kappa_r \mathbb{I} & \text{if } \frac{1}{2} \leq x \leq 1. \end{cases}$$

We want to obtain a piecewise linear analytical solution under the form

$$\hat{T}(x,y) = \begin{cases} a + bx + cy & \text{if } 0 \leq x \leq \frac{1}{2}, \\ a + \frac{1}{2} b \frac{\kappa_r - \kappa_l}{\kappa_r} + b \frac{\kappa_l}{\kappa_r} x + cy & \text{if } \frac{1}{2} \leq x \leq 1, \end{cases}$$

with a, b and c some real constants. In this test we have chosen $\kappa_l = 4$ and $\kappa_r = 1$ and $a = b = c = 1$.

To obtain this solution we apply the analytical solution $\hat{T}(x,y)$ using Dirichlet boundary condition on all the boundaries. The solution obtained on the finest Cartesian mesh is displayed on Figure 1.16.

When running this test on triangular and Cartesian meshes we obtain asymptotic errors equal to zero up to machine precision. This is what we expected since we showed earlier that our scheme preserves linear fields over triangular meshes and parallelograms.

The asymptotic errors and truncation errors obtained on the other quadrangular grids are presented in table 1.2. We did not use the Kershaw meshes for this test since these meshes do not have an interface at $x = \frac{1}{2}$. Table 1.2(a) shows an erratic behaviour in the order of convergence on random meshes as already observed in [32, 90]. In Table 1.2(b) we show that the scheme achieve second-order of accuracy for the two norms on smoothly deformed meshes. It is important to note that the random kind of meshes are not usually encountered in real-life applications, we are more likely to encounter smoothly deformed kind of meshes.

1.7.4 Linear problem with discontinuous anisotropic conductivity tensor

This problem consists in finding the steady solution of (1.90) with $r = 0$ and an anisotropic discontinuous conductivity tensor given by

$$\mathbb{K}(x,y) = \begin{cases} \begin{pmatrix} \kappa_l^{xx} & \kappa_l^{xy} \\ \kappa_l^{yx} & \kappa_l^{yy} \end{pmatrix} & \text{if } 0 \leq x \leq \frac{1}{2}, \\ \\ \begin{pmatrix} \kappa_r^{xx} & \kappa_r^{xy} \\ \kappa_r^{yx} & \kappa_r^{yy} \end{pmatrix} & \text{if } \frac{1}{2} \leq x \leq 1. \end{cases}$$

The one-dimensional solution, *i.e.*, $\hat{T} = \hat{T}(x)$ which corresponds to Dirichlet boundary conditions: $\hat{T}(0) = 0$ and $\hat{T}(1) = 1$, writes as

$$\hat{T}(x) = \begin{cases} \dfrac{2\kappa_r^{xx}}{\kappa_l^{xx} + \kappa_r^{xx}} x, & \text{if } 0 \leq x \leq \dfrac{1}{2}, \\ \\ \dfrac{\kappa_r^{xx} - \kappa_l^{xx}}{\kappa_r^{xx} + \kappa_l^{xx}} + \dfrac{2\kappa_l^{xx}}{\kappa_l^{xx} + \kappa_r^{xx}} x, & \text{if } \dfrac{1}{2} \leq x \leq 1. \end{cases}$$

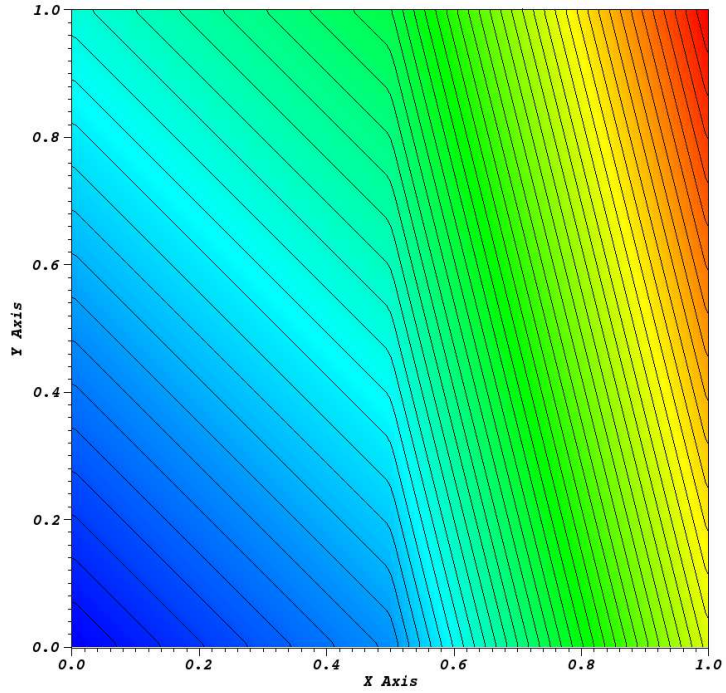


Figure 1.16: Piecewise linear solution on a 160×160 Cartesian mesh. 50 isovalues of the temperature are represented, the color map represents values ranging from 1 (blue) to 4.5 (red).

Table 1.2: Piecewise linear problem with discontinuous isotropic conductivity tensor: asymptotic errors in both L^2 and L^∞ norms and corresponding truncation error orders for quadrangular grids.

(a) Random grids.

h	E_{L2}	q_{L2}	E_∞	q_∞
1.00E-01	1.24E-03	**0.87**	5.05E-03	**0.88**
5.00E-02	6.80E-04	**0.84**	2.75E-03	**0.46**
2.50E-02	3.80E-04	**0.94**	1.99E-03	**1.08**
1.25E-02	1.98E-04	**1.00**	9.43E-04	**0.44**
6.25E-03	9.93E-05	-	6.95E-04	-

(b) Smooth grids.

h	E_{L2}	q_{L2}	E_∞	q_∞
1.00E-01	3.63E-03	**1.72**	1.65E-02	**1.66**
5.00E-02	1.10E-03	**1.90**	5.22E-03	**1.91**
2.50E-02	2.95E-04	**1.97**	1.39E-03	**1.98**
1.25E-02	7.52E-05	**1.99**	3.52E-04	**1.99**
6.25E-03	1.89E-05	-	8.88E-05	-

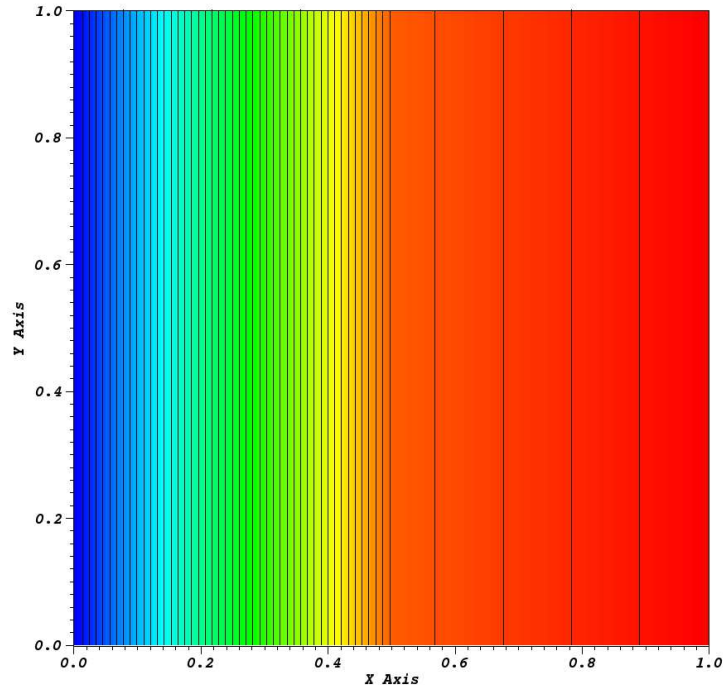


Figure 1.17: Linear solution with discontinuous anisotropic conductivity tensor on a 160×160 Cartesian mesh. 50 isovalues of the temperature are represented, the color map represents values ranging from 0 (blue) to 1 (red).

This is a linear continuous solution for which the heat flux $\hat{\boldsymbol{q}} = -\mathbb{K}\boldsymbol{\nabla} T$ writes as

$$\hat{\boldsymbol{q}} = -\begin{cases} \begin{pmatrix} 2\dfrac{\kappa_l^{xx}\kappa_r^{xx}}{\kappa_l^{xx}+\kappa_r^{xx}} \\ 2\dfrac{\kappa_l^{yx}\kappa_r^{xx}}{\kappa_l^{xx}+\kappa_r^{xx}} \end{pmatrix} & \text{if } 0 \le x \le \frac{1}{2}, \\ \begin{pmatrix} 2\dfrac{\kappa_l^{xx}\kappa_r^{xx}}{\kappa_l^{xx}+\kappa_r^{xx}} \\ 2\dfrac{\kappa_l^{xx}\kappa_r^{yx}}{\kappa_l^{xx}+\kappa_r^{xx}} \end{pmatrix} & \text{if } \frac{1}{2} \le x \le 1. \end{cases}$$

The normal component of the heat flux is continuous at the interface $x = \frac{1}{2}$ whereas its tangential component undergone a jump discontinuity since in general $\kappa_l^{yx}\kappa_r^{xx} \neq \kappa_l^{xx}\kappa_r^{yx}$.

The boundary conditions applied on the top and the bottom boundaries of the computational domain are Dirichlet boundary conditions deduced from the analytical solution. For the numerical applications we have defined the entries of the conductivity tensor as $\kappa_l^{xx} = 1$, $\kappa_l^{xy} = \kappa_l^{yx} = -1$, $\kappa_l^{yy} = 4$ and $\kappa_r^{xx} = 10$, $\kappa_r^{xy} = \kappa_r^{yx} = -3$, $\kappa_r^{yy} = 2$.

As in the previous section, our finite volume scheme preserves linear solutions on triangular grids and on cells which are parallelograms.

The convergence analysis for smooth and random grids is performed computing the asymptotic errors and the corresponding orders of truncation error. The results displayed in Table 1.3(a)

Table 1.3: Anisotropic linear problem with discontinuous conductivity tensor: asymptotic errors in both L^2 and L^∞ norms and corresponding truncation error orders for quadrangular grids.

(a) Random grids.

h	E_{L2}	q_{L2}	E_∞	q_∞
1.00E-01	1.27E-03	**0.76**	3.27E-03	**0.23**
5.00E-02	7.51E-04	**0.87**	2.78E-03	**0.64**
2.50E-02	4.12E-04	**0.63**	1.79E-03	**0.60**
1.25E-02	2.66E-04	**0.28**	1.18E-03	**0.37**
6.25E-03	2.20E-04	-	9.12E-04	-

(b) Smooth grids.

h	E_{L2}	q_{L2}	E_∞	q_∞
1.00E-01	3.04E-03	**1.69**	8.99E-03	**1.29**
5.00E-02	9.45E-04	**1.84**	3.68E-03	**1.48**
2.50E-02	2.65E-04	**1.93**	1.32E-03	**1.73**
1.25E-02	6.92E-05	**1.98**	3.98E-04	**1.87**
6.25E-03	1.75E-05	-	1.09E-04	-

for the randomly perturbed grids shows an even more erratic behaviour as we have seen in the previous section. For the smooth grids, the results displayed in Table 1.3(b) show that the convergence rate is almost second-order in the L^2 norm and a little bit less in the L^∞ norm.

1.7.5 Anisotropic linear problem with a non-uniform symmetric positive definite conductivity tensor

This test problem has been proposed in [101]. Once more, it consists in finding the steady solution of (1.90). However, it is characterized by an anisotropic non-uniform conductivity tensor which writes for all $(x, y) \in [0, 1]^2$

$$\mathbb{K}(x, y) = \begin{pmatrix} y^2 + \eta x^2 & -(1-\eta)xy \\ -(1-\eta)xy & x^2 + \eta y^2 \end{pmatrix},$$

where η is a positive parameter characterizing the level of anisotropy. This tensor is symmetric positive definite. Its eigenvalues are $\lambda^+ = x^2 + y^2$ and $\lambda^- = \eta(x^2 + y^2)$. Thus, its condition number is equal to $\frac{1}{\eta}$. The source term, r, is computed such that the analytical solution (1.90) is given by

$$\hat{T}(x, y) = \sin^2(\pi x) \sin^2(\pi y).$$

We apply a homogeneous Dirichlet boundary condition on the boundaries of the computational domain, *i.e.*, $T(\boldsymbol{x}, t) = 0$, $\forall \boldsymbol{x} \in \partial\mathcal{D}$. For numerical applications, we choose $\eta = 10^{-2}$. We assess the accuracy of our finite volume scheme by running this test problem on sequence of triangular and distorted quadrangular grids.

The convergence analysis results corresponding to the numerical simulations using the five triangular grids are displayed in Table 1.4. They show that our finite volume scheme has a second-order convergence rate in L^2 norm on triangular grids. The convergence rate for the L^∞ is a bit more erratic but seems to lay around second-order convergence rate too.

Concerning the quadrangular grids we perform the convergence analysis on four types of

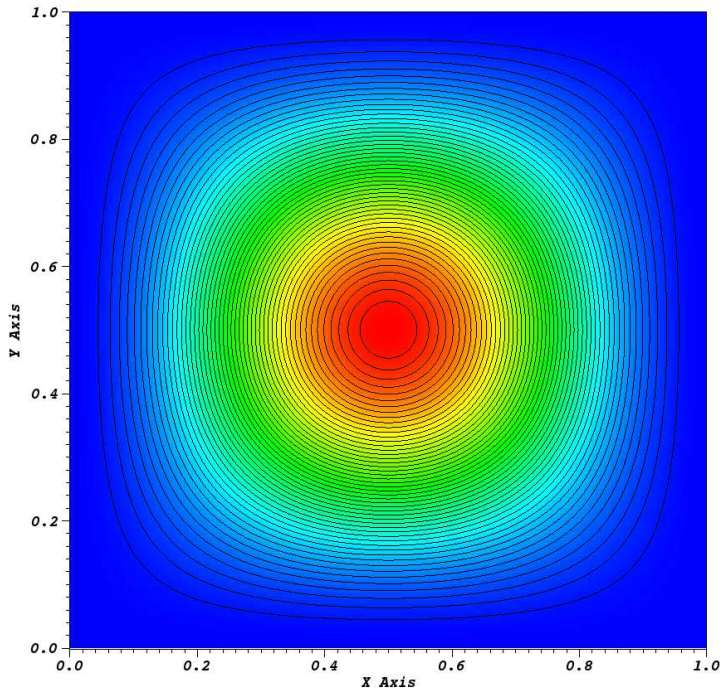


Figure 1.18: Anisotropic linear solution with a non-uniform symmetric positive definite conductivity tensor on a 160×160 Cartesian mesh. 50 isovalues of the temperature are represented, the color map represents values ranging from 0 (blue) to 1 (red).

h	E_{L2}	q_{L2}	E_∞	q_∞
6.15E-02	2.69E-02	**2.71**	9.22E-02	**2.57**
3.11E-02	4.23E-03	**2.16**	1.60E-02	**2.00**
1.55E-02	9.32E-04	**2.14**	3.95E-03	**2.17**
7.67E-03	2.08E-04	**2.03**	8.61E-04	**1.80**
3.85E-03	5.11E-05	-	2.49E-04	-

Table 1.4: Anisotropic linear problem with a non-uniform symmetric positive definite conductivity tensor: asymptotic errors in both L^2 and L^∞ norms and corresponding truncation error orders for triangular grids.

Table 1.5: Anisotropic linear problem with a non-uniform symmetric positive definite conductivity tensor: asymptotic errors in both L^2 and L^∞ norms and corresponding truncation error orders for quadrangular grids.

(a) Rectangular grids.

h	E_{L2}	q_{L2}	E_∞	q_∞
1.00E-01	1.69E-02	**2.07**	3.97E-02	**2.08**
5.00E-02	4.03E-03	**2.02**	9.41E-03	**2.02**
2.50E-02	9.95E-04	**2.00**	2.32E-03	**2.01**
1.25E-02	2.48E-04	**2.00**	5.78E-04	**2.00**
6.25E-03	6.20E-05	-	1.44E-04	-

(b) Smooth grids.

h	E_{L2}	q_{L2}	E_∞	q_∞
1.00E-01	3.07E-02	**2.08**	1.79E-01	**1.95**
5.00E-02	7.25E-03	**2.01**	4.64E-02	**1.94**
2.50E-02	1.80E-03	**2.00**	1.21E-02	**1.97**
1.25E-02	4.48E-04	**2.00**	3.08E-03	**2.00**
6.25E-03	1.12E-04	-	7.71E-04	-

(c) Kershaw grids.

h	E_{L2}	q_{L2}	E_∞	q_∞
8.33E-02	8.16E-02	**2.00**	3.39E-01	**2.24**
4.17E-02	2.04E-02	**2.03**	7.17E-02	**2.04**
2.08E-02	5.01E-03	**2.01**	1.74E-02	**1.99**
1.04E-02	1.24E-03	**2.01**	4.39E-03	**1.91**
5.21E-03	3.09E-04	-	1.17E-03	-

(d) Random grids.

h	E_{L2}	q_{L2}	E_∞	q_∞
1.00E-01	2.17E-02	**1.85**	6.80E-02	**1.49**
5.00E-02	6.01E-03	**1.76**	2.42E-02	**1.52**
2.50E-02	1.77E-03	**1.21**	8.42E-03	**0.88**
1.25E-02	7.69E-04	**0.73**	4.58E-03	**0.78**
6.25E-03	4.63E-04	-	2.68E-03	-

grids: rectangular, smooth, Kershaw and random. We start by giving in Table 1.5(a) the convergence analysis data for a sequence of five rectangular grids. These data demonstrate that our finite volume scheme exhibits a second-order rate of convergence on rectangular grids.

Next, we pursue our investigations using the sequence of the five smooth distorted grids. The convergence analysis results obtained with these five grids are presented in Table 1.5(b). Once more, we observe a second-order convergence rate in L^2 and L^∞ norm. We can draw similar conclusions with the sequence of Kershaw type meshes as presented in Table 1.5(c). In this case we observe a second-order convergence rate in L^2 norm, and the convergence rate in L^∞ norm is almost second-order.

The results of the convergence analysis corresponding to the sequence of random grids are given in Table 1.5(d). In comparison to the previous results, these ones are representative of an

erratic behavior which clearly does not correspond to second-order.

1.8 Numerical results in three-dimensional geometries

The aim of this section is to assess the robustness and the accuracy of our finite volume scheme against analytical test cases using various types of unstructured three-dimensional grids. The tests cases have been chosen to highlight the different features of the scheme. The methodology used is exactly the same as the one used for the two-dimensional geometries. First, we present the three-dimensional structured and unstructured grids employed. Then, we describe the set up of each test case, display the numerical results obtained and discuss the quality of the corresponding convergence analysis.

We recall that the numerical solutions are obtained solving linear systems by means of the localized ILU(0) Preconditioned BiCGStab algorithm [127, 95]. The relative error tolerance to achieve the convergence is equal to 10^{-16}.

1.8.1 Computational grids

Here, we present the three-dimensional computational grids employed to run the test cases. There are various types of grids: tetrahedral grids, hexahedral grids and hybrid grids which are composed of tetrahedra, hexahedra and pyramids. The detailed description of these grids is summarized in the list below:

- Tetrahedral grids, displayed in Figure 1.19(a) and Figure 1.19(f), have been constructed using Gmsh, which is a three-dimensional finite element mesh generator [50];
- Hexahedral Cartesian grid displayed in Figure 1.19(b);
- Kershaw-type grid displayed in Figure 1.19(c);
- Smoothly deformed hexahedral grid resulting from the mapping defined on the unit cube $[0,1]^3$ by

$$\begin{aligned} x(\xi,\eta,\theta) =& \xi + a_0 \sin(2\pi\xi)\sin(2\pi\eta)\sin(2\pi\theta), \\ y(\xi,\eta,\theta) =& \eta + a_0 \sin(2\pi\xi)\sin(2\pi\eta)\sin(2\pi\theta), \\ z(\xi,\eta,\theta) =& \theta + a_0 \sin(2\pi\xi)\sin(2\pi\eta)\sin(2\pi\theta), \end{aligned}$$

  where the amplitude of the deformation is $a_0 = 0.1$. Observing that the deformation cancels on the boundary surfaces of the unit cube, this grid has not been displayed since it looks like the Cartesian grid;

- Randomly deformed hexahedral grid, displayed in Figure 1.19(d), resulting from the mapping defined on the unit cube $[0,1]^3$ by:

$$\begin{aligned} x(\xi,\eta,\theta) &= \xi + a_0 h r_1, \\ y(\xi,\eta,\theta) &= \eta + a_0 h r_2, \\ z(\xi,\eta,\theta) &= \theta + a_0 h r_3, \end{aligned}$$

  where $\{r_i\}_{i=1\ldots3}$ are random numbers in $[-1,1]$, h is the characteristic mesh size and $a_0 = 0.2$ the amplitude of the deformation;

- Hybrid grid, displayed in Figure 1.19(e), made of hexahedral cells, pyramidal cells and tetrahedra.

Comment 13: *We have introduced the hybrid grid because of its usefulness regarding real-world applications. Let us point out that it is a convenient way to mesh a domain using both hexahedral and tetrahedra cells with the constraint of keeping a conformal grid. In this case, a layer of pyramids ensures the transition between hexahedra and tetrahedra. This kind of grid can be used in the context of the computation of a viscous flow in the presence of a solid wall. Indeed, the pyramid cells allows to match the boundary layer in the vicinity of the wall, paved by means of hexahedra, with the rest of the domain paved using tetrahedra.*

Comment 14: *The tetrahedral grid displayed in Figure 1.19(f) corresponds to a truncated sphere with an internal radius R_i and an external radius R_e. This grid is also characterized by an interface located at R_m, which allows to separate two distinct materials.*

1.8.2 Isotropic diffusion problem

This problem consists in finding the steady solution of (1.90) with $r = 0$ and an isotropic conductivity tensor defined by $\mathbb{K} = \kappa\mathbb{I}$, where $\mathbb{I}$ is the unit tensor of $\mathfrak{R}^3$ and the scalar conductivity is given by $\kappa = 1$. The computation domain is $\mathcal{D} = [0, 1]^3$ and we apply the following boundary conditions on the boundaries of $\mathcal{D}$

- Dirichlet boundary condition

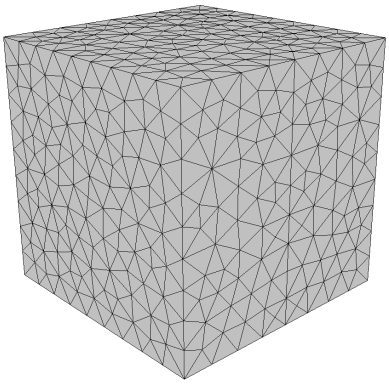
$$\begin{aligned} T(\boldsymbol{x}) &= 0, \quad \text{for } \; x = 0, \\ T(\boldsymbol{x}) &= 1, \quad \text{for } \; x = 1. \end{aligned}$$

- Neumann boundary condition

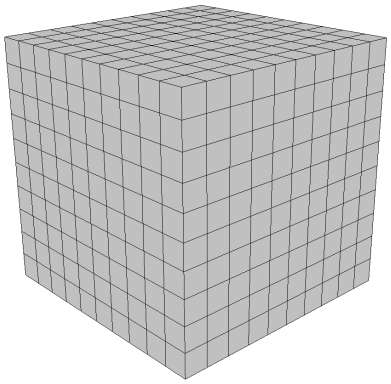
$$\boldsymbol{q}(x) \cdot \boldsymbol{n} = 0, \quad \text{for } \; y = 0, \; y = 1, \; z = 0 \; \text{ and } \; z = 1.$$

The steady analytical solution is $\hat{T}(\boldsymbol{x}) = x$. The aim of this simple test case is to assess the ability of our scheme to preserve linear fields.

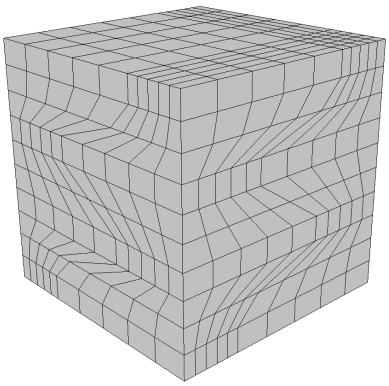
First, we compute the steady numerical solution using a tetrahedral grid made of 8222 cells, refer to Figure 1.19(a). The corresponding asymptotic errors are equal to zero up to machine precision. As expected, our finite volume scheme preserves linear solutions on tetrahedral grids. We observe a similar behavior when computing the numerical solution on the Cartesian hexahedral grid displayed in Figure 1.19(b). Let us point that this result confirms the conclusion already drawn for this type of numerical methods, in the context of two-dimensional geometry, refer to [32, 90]. The convergence analysis for smooth grids, Kershaw grids (refer to Figure 1.19(c)) and random grids (refer to Figure 1.19(d)) are performed computing the asymptotic errors and the corresponding orders of truncation error using formulas (1.91) and (1.92). The results displayed in Table 1.6(a) show that the convergence rate is almost of second-order in the L^2 norm and a little bit less in the L^∞ norm for the smooth grids. In Table 1.6(b), we observe a similar behavior for the convergence analysis corresponding to the Kershaw grids. Proceeding with the convergence analysis for random grids as before, we have displayed the corresponding results in Table 1.6(c). The convergence rate is of first-order for the L^2 norm and almost of first-order for the L^∞ norm.



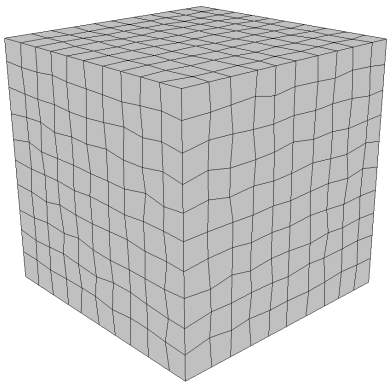
(a) Tetrahedral grid made of 8222 cells.



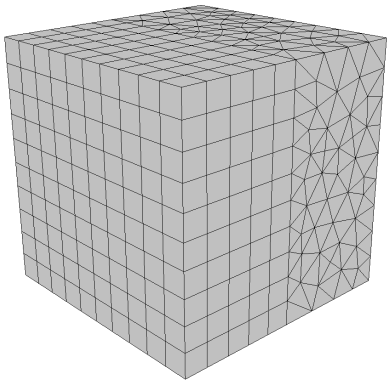
(b) Cartesian hexahedral grid made of 1000 cells.



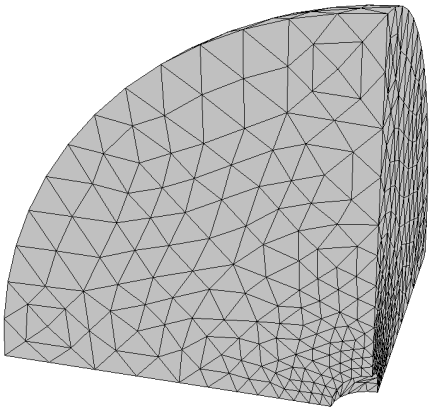
(c) Kershaw-type grid made of 1000 cells.



(d) Randomly perturbed grid made of 1000 cells.



(e) Hybrid grid made of 3432 tetrahedra, 100 pyramids and 500 hexahedra.



(f) Tetrahedral grid of a truncated sphere made of 6623 cells.

Figure 1.19: Three-dimensional grids used for the test cases.

Table 1.6: Isotropic diffusion problem, asymptotic errors in both L^∞ and L^2 norms and corresponding truncation errors for hexahedral grids.

(a) Smooth grids.

h	E_∞^h	q_∞^h	E_2^h	q_2^2	Iterations
1.00D-01	5.65D-03	**1.60**	2.18D-03	**1.69**	11
5.00D-02	1.87D-03	**1.80**	6.75D-04	**1.90**	22
2.50D-02	5.35D-04	**1.92**	1.81D-04	**1.97**	44
1.25D-02	1.41D-04	-	4.63D-05	-	101

(b) Kershaw grids.

h	E_∞^h	q_∞^h	E_2^h	q_2^2	Iterations
1.00D-01	3.22D-02	**1.94**	8.23D-03	**2.04**	14
5.00D-02	8.39D-03	**1.39**	2.00D-03	**1.69**	37
2.50D-02	3.20D-03	**2.09**	6.20D-04	**2.06**	80
1.25D-02	7.53D-04	-	1.49D-04	-	116

(c) Random grids.

h	E_∞^h	q_∞^h	E_2^h	q_2^2	Iterations
1.00D-01	2.09D-03	**0.75**	6.62D-04	**1.00**	11
5.00D-02	1.24D-03	**0.81**	3.31D-04	**1.00**	21
2.50D-02	7.07D-04	**0.93**	1.66D-04	**1.00**	42
1.25D-02	3.72D-04	-	8.29D-05	-	78

h	E_∞^h	q_∞^h	E_2^h	q_2^2	Iterations
6.28D-02	2.06D-03	**0.99**	2.66D-04	**1.49**	14
3.12D-02	1.04D-03	**1.00**	9.39D-05	**1.50**	27
1.56D-02	5.19D-04	**1.74**	3.32D-05	**2.59**	60
1.04D-02	2.58D-04	-	1.17D-05	-	95

Table 1.7: Isotropic diffusion problem, asymptotic errors in both L^∞ and L^2 norms and corresponding truncation errors for hybrid grids.

Comment 15: *Let us point out that the last columns in the above tables represent the number of iterations required by our ILU(0) Preconditioned BiCGStab algorithm to reach the relative error tolerance required to achieve the convergence in solving the linear systems. This error tolerance has been set equal to* 10^{-16} *which is a very small tolerance. This probably explains the relatively high number of iterations of our solver. It is worth mentioning that for real life applications we shall set the error tolerance equal to* 10^{-8}.

Finally, the convergence analysis for the hybrid grids, refer to Figure 1.19(e), is displayed in Table 1.7. The corresponding data demonstrate that our numerical scheme exhibits a rate of convergence located between first and second-order. Let us point out that the maximal error is always located in the layer of pyramids which allows to link the tetrahedral and the hexahedral regions of the grid. This clearly shows that the loss of accuracy is the consequence of the particular treatment applied to pyramids to derive the flux approximation, refer to [74] for more details.

h	E_∞^h	q_∞^h	E_2^h	q_2^2	Iterations
1.20D-01	1.23D-01	**3.19**	2.16D-02	**2.49**	7
5.80D-02	1.22D-02	**1.65**	3.57D-03	**2.25**	18
4.54D-02	8.14D-03	**1.90**	2.05D-03	**2.08**	39
3.17D-02	4.12D-03	-	9.73D-04	-	59

Table 1.8: Isotropic diffusion problem with a discontinuous conductivity, asymptotic errors in both L^∞ and L^2 norms and corresponding truncation errors for tetrahedral grids.

1.8.3 Isotropic diffusion problem with a discontinuous conductivity

Here, the computational domain, $\mathcal{D}$, is the truncated sphere, centered at the origin and characterized by the inner radius $R_i = 0.1$ and the outer radius $R_e = 1$. An interface, located at the radius $R_m = 0.5$, splits the computational domain into two regions filled with two distinct materials. The conductivity tensor is isotropic and piecewise constant, *i.e*, $\mathbb{K} = \kappa\mathbb{I}$ where $\kappa = \kappa(r)$ with $r = \sqrt{x^2 + y^2 + z^2}$. The scalar conductivity is given by

$$\kappa(r) = \begin{cases} \kappa_1 & \text{if } r \in [R_i, R_m[, \\ \kappa_2 & \text{if } r \in]R_m, R_e]. \end{cases}$$

For numerical applications, we choose $\kappa_1 = 10$ and $\kappa_2 = 1$. Dirichlet boundary conditions are prescribed at the inner and the outer boundary of the computational domain, *i.e.*, $T(R_i) = T_i = 0$ and $T(R_e) = T_e = 1$. Due the radial symmetry of the problem, we consider a computational domain restricted to $\frac{1}{8}$ of the truncated sphere. The corresponding coarsest tetrahedral grid is displayed in Figure 1.19(f). Homogeneous Neumann boundary conditions are prescribed at the remaining boundaries of the computational domain to handle the symmetry of the problem.

The steady analytical temperature, $\hat{T}(r)$, field is obtained by solving the following problem

$$\begin{aligned} &\frac{1}{r^2}\frac{\mathrm{d}}{\mathrm{d}r}(r^2\frac{\mathrm{d}T}{\mathrm{d}r}) = 0, \quad r \in]R_i, R_e[\\ &T(R_i) = T_i, \\ &T(R_m^-) = T(R_m^+), \quad \kappa_1\frac{\mathrm{d}T}{\mathrm{d}r}(R_m^-) = -\kappa_2\frac{\mathrm{d}T}{\mathrm{d}r}(R_m^+), \\ &T(R_e) = T_e. \end{aligned}$$

Let us remark that the second equation in the above system expresses the continuity conditions of the temperature and the heat flux across the interface located at R_m. Employing the previous numerical values, the analytical solution reads

$$\hat{T}(r) = \begin{cases} -\frac{1}{18r} + \frac{5}{9} & \text{if } r \in [R_i, R_m], \\ -\frac{5}{9r} + \frac{14}{9} & \text{if } r \in [R_m, R_e]. \end{cases}$$

The steady analytical and numerical solutions are displayed in Figure 1.20. We have plotted the averaged temperature of all the cells versus the cell centroid radius. We observe that the numerical solution is almost superimposed to the analytical solution. **This clearly shows the ability of the scheme to preserve the radial symmetry on a highly anisotropic unstructured grid, which is not aligned with the symmetry of the problem**. We investigate the convergence analysis for this problem using a sequence of four tetrahedral grids made of 902, 6623, 13549 and 37648 cells. The resulting asymptotic errors and rate of convergence in both L^∞ and L^2 norms are presented in Table 1.8. We observe that a second-order

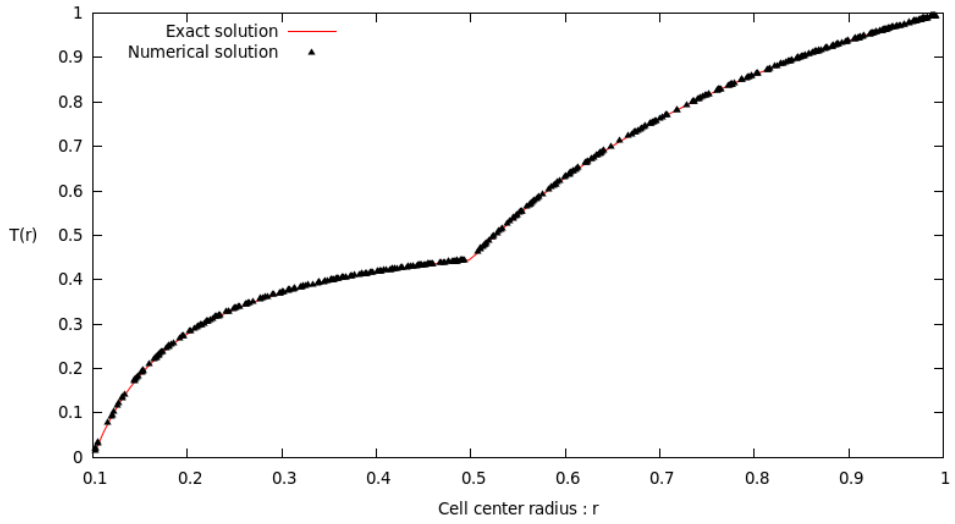


Figure 1.20: Isotropic diffusion problem with a discontinuous conductivity: Temperatures in all the cells with respect to the radii of the cell centroid for a tetrahedral grid composed of 13549 tetrahedra; comparison with the analytical solution.

rate of convergence is asymptotically reached in L^2 norm. It is interesting to note the large gap in the maximum errors between the coarsest grid and the second grid. This might be due to the discretization of the spherical boundaries. On the coarsest grid, the mesh resolution is too poor to properly capture the curvilinear inner and outer boundaries. When the grid is refined, the curvilinear feature of the boundaries is better captured due to the increased mesh resolution.

1.8.4 Anisotropic diffusion with a highly heterogeneous conductivity tensor

This paragraph consists in assessing our finite volume scheme against a test case which is representative of anisotropic diffusion characterized by a highly heterogeneous conductivity tensor. This test case and its manufactured analytical solution are taken from [64]. Here, we solve the problem (1.90) over the computational domain $\mathcal{D} = [0,1]^3$. The conductivity tensor is defined by

$$\mathbb{K} = \mathbb{Q} \begin{pmatrix} 1 & 0 & 0 \\ 0 & \varepsilon & 0 \\ 0 & 0 & \eta(1+x+y+z) \end{pmatrix} \mathbb{Q}^t.$$

where, $\mathbb{Q} = \mathbb{Q}(x)$ is the rotation given by

$$\mathbb{Q} = \begin{pmatrix} \cos(\pi x) & -\sin(\pi x) & 0 \\ \sin(\pi x) & \cos(\pi x) & 0 \\ 0 & 0 & 1 \end{pmatrix}.$$

Here, ε and η are parameters which measure the degree of anisotropy of the conductivity tensor. Indeed, the eigenvalues of the conductivity tensor are: 1, ε and $\eta(1+x+y+z)$. For numerical applications, we shall take $\varepsilon = 0.1$ and $\eta = 10$. The source term, $r = r(\boldsymbol{x})$, is computed such that the analytical steady solution of (1.90) is given by

$$\hat{T}(x,y,z) = \sin(\pi x)\sin(\pi y)\sin(\pi z).$$

We apply a homogeneous Dirichlet boundary condition on the boundaries of the computational domain, *i.e.*, $T(\boldsymbol{x},t) = 0, \quad \forall\ \boldsymbol{x} \in \partial\mathcal{D}$. First, we compute the numerical solution using a se-

h	E_∞^h	q_∞^h	E_2^h	q_2^2	Iterations
1.12D-01	3.01D-01	**2.43**	7.98D-02	**2.55**	10
4.95D-02	4.18D-02	**1.94**	1.01D-02	**2.14**	14
2.47D-02	1.08D-02	**1.64**	2.26D-03	**1.99**	32
1.23D-02	3.45D-03	-	5.70D-04	-	61

Table 1.9: Anisotropic diffusion problem, asymptotic errors in both L^∞ and L^2 norms and corresponding truncation errors for tetrahedral grids.

quence of four tetrahedral grids. The coarsest grid has been displayed in Figure 1.19(a). The asymptotic errors in both L^∞ and L^2 norms and the corresponding truncation errors are summarized in Table 1.9. They show that the convergence rate in L^2 norm is of second-order. Regarding the convergence analysis on hexahedral grids, we have also used a sequence of four grids for the following types of grids: Cartesian, Kershaw, smooth and random. These grids are showed respectively in Figure 1.19(b), Figure 1.19(c) and Figure 1.19(d). Let us recall that the smooth grid has not been displayed since the deformation cancels on the boundaries of the computational domain. We start by giving in Table 1.10(a) the convergence analysis data for a sequence of four Cartesian grids. These data demonstrate that our scheme exhibits an almost second-order rate of convergence on Cartesian grids. The same conclusion holds for the convergence analysis performed on smooth grids, refer to Table 1.10(c). We observe that the rate of convergence in L^2 norm are better than those obtained for the rectangular grids, however the asymptotic errors on smooth grids are approximately three times bigger than the ones corresponding to the Cartesian grids. Next, we pursue our investigation using a sequence of four Kershaw grids. The related convergence analysis is summarized in Table 1.10(b). This time, the rate of convergence in both L^∞ and L^2 norms is lying between first and second-order. Finally, we compute the numerical solution on a sequence of four random grids. The results of the convergence analysis corresponding to this sequence of grids are given in Table 1.10(d). In comparison to the above results, these ones are representative of an erratic behavior, which clearly does not correspond to a second-order rate of convergence.

We achieve the convergence analysis of the present problem by studying the numerical solutions obtained employing a sequence of four hybrid grids, refer to Figure 1.19(e). The asymptotic errors and the convergence rates in both L^∞ and L^2 norms are displayed in Table 1.11. The results demonstrate that the scheme is characterized by a rate of convergence located between first and second-order. Once more, the maximal error is located in the layer of pyramids which allows to link the tetrahedral and the hexahedral regions of the grid. Let us repeat that this loss of accuracy is the consequence of the particular treatment applied to pyramids to derive the flux approximation, refer to [74].

1.9 Conclusion

In this chapter, we have described a cell-centered finite volume scheme, which aims at solving anisotropic diffusion problems on two and three-dimensional unstructured grids. This scheme is characterized by cell-centered unknowns, a local stencil and a symmetric positive definite matrix. The partition of grid cells (resp. faces) into sub-cells (resp. -faces) allows to construct a sub-face fluxes approximation by means of a sub-cell-based variational formulation. The sub-face fluxes are locally expressed at each node in terms of the surrounding cell-centered temperatures invoking continuity conditions of temperature and normal heat flux at each cell interface. Regarding its accuracy, the scheme preserves linear fields with respect to the space

Table 1.10: Anisotropic diffusion problem, asymptotic errors in both L^∞ and L^2 norms and corresponding truncation errors for hexahedral grids.

(a) Cartesian grids.

h	E_∞^h	q_∞^h	E_2^h	q_2^2	Iterations
1.00e-01	1.32D-02	**1.68**	4.86D-03	**1.91**	10
5.00D-02	4.13D-03	**1.62**	1.30D-03	**1.95**	21
2.50D-02	1.34D-03	**1.69**	3.35D-04	**1.98**	43
1.25D-02	4.15D-04	-	8.50D-05	-	112

(b) Kershaw grids.

h	E_∞^h	q_∞^h	E_2^h	q_2^2	Iterations
1.00D-01	3.01D-02	**1.15**	7.80D-03	**1.72**	13
5.00D-02	1.36D-02	**1.25**	2.38D-03	**1.55**	28
2.50D-02	5.72D-03	**1.65**	8.13D-04	**1.99**	53
1.25D-02	1.82D-03	-	2.05D-04	-	109

(c) Smooth grids.

h	E_∞^h	q_∞^h	E_2^h	q_2^2	Iterations
1.00D-01	6.03D-02	**2.09**	1.60D-02	**2.12**	16
5.00D-02	1.41D-02	**1.68**	3.69D-03	**2.03**	37
2.50D-02	4.41D-03	**1.89**	9.03D-04	**2.00**	87
1.25D-02	1.19D-03	-	2.26D-04	-	123

(d) Random grids.

h	E_∞^h	q_∞^h	E_2^h	q_2^2	Iterations
1.00D-01	2.44D-02	**1.41**	7.35D-03	**1.86**	11
5.00D-02	9.17D-03	**1.09**	2.02D-03	**1.56**	21
2.50D-02	4.31D-03	**0.86**	6.85D-04	**0.72**	46
1.25D-02	2.37D-03	-	4.16D-04	-	94

h	E_∞^h	q_∞^h	E_2^h	q_2^2	Iterations
6.28D-02	4.67D-02	**1.63**	9.58D-03	**2.01**	18
3.12D-02	1.49D-02	**1.47**	2.34D-03	**1.93**	26
1.56D-02	5.38D-03	**1.31**	6.14D-04	**1.83**	65
1.04D-02	3.18D-03	-	2.95D-04	-	128

Table 1.11: Anisotropic diffusion problem, asymptotic errors in both L^∞ and L^2 norms and corresponding truncation errors for hybrid grids.

variable over triangular and tetrahedral grids and exhibits an almost second-order rate of convergence on smooth distorted quadrangular and hexahedral grids. The parallel implementation of the scheme is discussed and its evaluation shows a satisfying efficiency. This last point is crucial for dealing with real life three-dimensional applications.

Chapter 2

A Finite Volume scheme for solving tensorial diffusion on unstructured grids

Here, we are interesting in developing an unstructured Finite Volume discretization of the viscous flux contained in the compressible Navier-Stokes equations, refer to Chapter 3. This viscous flux is characterized by a constitutive law which expresses the deviatoric part of the Cauchy stress tensor in terms of the deviatoric part of the strain rate tensor, *i.e.*, the symmetric part of the velocity gradient tensor. In what follows, we present a cell-centered Finite Volume discretization of a generic tensorial diffusion equation which is nothing but the momentum equation of the compressible Navier-Stokes equations wherein the pressure gradient term has been suppressed. This space discretization is the non-trivial extension of the CCLAD scheme to a generic tensorial diffusion equation. This Finite Volume scheme is described in a two-dimensional framework knowing that the three-dimensional extension can be derived straightforwardly. In this approach, the degrees of freedom are located at the cell centers and the divergence of the viscous stress tensor is discretized by means of half-edge viscous fluxes which are the projections of the viscous stress tensor onto the unit outward normal to the cell interfaces. As in CCLAD scheme, the half-edge viscous fluxes approximation relies on a variational formulation of the constitutive law. However, in the present case, this methodology leads to a space approximation of the constitutive law which is only positive semi-definite. This lack of definiteness renders the straightforward CCLAD extension to tensorial diffusion inoperative. It is a direct consequence of the fact that the constitutive law is not invertible over the space of generic second-order tensors. Indeed, the constitutive law is only invertible over the space of traceless symmetric second-order tensors. This problem has been already encountered by Arnold [16] in the context of linear elasticity. To solve it, he has proposed a new mixed variational formulation for the equations of linear elasticity which does not require symmetric tensors and consequently is easy to discretize by adapting mixed finite elements developed for scalar second-order elliptic problems.

Here, we adapt Arnold's approach by adding a divergence free term to the constitutive law such that it renders it invertible over the space of generic second-order tensors while keeping the generic tensorial diffusion equation satisfied by the velocity field unchanged. This methodology allows us to derive a robust approximation of the half-edge viscous fluxes in terms of the cell-centered velocity and the half-edge velocities, which are supplementary degrees of freedom. These auxiliary unknowns are then eliminated by solving node-based invertible linear systems which are obtained by writing the continuity of the half-edge viscous fluxes across cell interfaces impinging at a given node. The remainder of this chapter is organized as follows: we start by

presenting the generic tensorial diffusion equation and its connection to the compressible Navier-Stokes equation. Then, we address the mathematical properties of the constitutive law, which lead us to the construction of an invertible constitutive law by adapting the pioneering work of Arnold. Finally, we apply the CCLAD's methodology developed in Chapter 1 to construct a cell-centered Finite Volume scheme for the tensorial diffusion equation. We finish this chapter by conducting extensive numerical tests cases to assess the robustness and the accuracy of our numerical method.

2.1 The tensorial diffusion equation

Let us study the Finite Volume discretization of the generic tensorial diffusion equation obtained by removing the convective terms from the momentum equation of the classical compressible Navier-Stokes equations, refer to Chapter 3. Let $\mathcal{D}$ be an open set of the d-dimensional space $\mathfrak{R}^d$ and $\boldsymbol{x}$ is the position vector of an arbitrary point inside the domain $\mathcal{D}$ and $t > 0$ is the time. We aim at constructing a numerical scheme to solve the following initial-boundary-value problem for the velocity vector $\boldsymbol{V} = \boldsymbol{V}(\boldsymbol{x}, t)$

$$\rho \frac{\partial \boldsymbol{V}}{\partial t} - \nabla \cdot \mathbb{S} = \boldsymbol{0}, \qquad \forall (\boldsymbol{x}, t) \in \mathcal{D} \times [0, \mathfrak{T}], \tag{2.1a}$$

$$\boldsymbol{V}(\boldsymbol{x}, 0) = \boldsymbol{V}^0(\boldsymbol{x}), \qquad \forall \boldsymbol{x} \in \mathcal{D}. \tag{2.1b}$$

Here $\mathfrak{T} > 0$ is the final time, ρ is a positive real valued function, which stands for the mass density and $\boldsymbol{V}^0$ stands for the initial velocity. The second-order tensor, $\mathbb{S}$, represents the deviatoric part of the Cauchy stress tensor which is defined by the constitutive law

$$\mathbb{S} = 2\mu \left[\mathbb{D} - \frac{1}{3} (\operatorname{tr} \mathbb{D}) \mathbb{I} \right],$$

where $\mu = \mu(\boldsymbol{x}) > 0$ is the kinematic viscosity and $\mathbb{D}$ is the strain rate tensor, which is nothing but the symmetric part of the velocity gradient tensor, *i.e.*

$$\mathbb{D} = \frac{1}{2} \left[\nabla \boldsymbol{V} + (\nabla \boldsymbol{V})^t \right].$$

Being given a generic second-order tensor, $\mathbb{T}$, its deviatoric part, $\mathbb{T}_0$, is given by

$$\mathbb{T}_0 = \mathbb{T} - \frac{1}{3} (\operatorname{tr} \mathbb{T}) \mathbb{I}.$$

Finally, employing the above notation, the constitutive law turns into the more compact form

$$\mathbb{S} = 2\mu \mathbb{D}_0. \tag{2.2}$$

It remains to prescribe the boundary conditions. Although a lot of different boundary conditions can be defined for the above tensorial diffusion equation (2.1a), we restrict our study to the ones that will be used in fluid dynamics computations. We shall develop three kinds of boundary conditions employing the following partition of the domain boundary $\partial\mathcal{D} = \partial\mathcal{D}_k \cup \partial\mathcal{D}_n \cup \partial\mathcal{D}_s$

Kinematic boundary condition on $\partial\mathcal{D}_k$

Here, the velocity is prescribed as

$$\boldsymbol{V}(\boldsymbol{x}, t) = \boldsymbol{V}^\star(\boldsymbol{x}, t), \ \forall \boldsymbol{x} \in \partial\mathcal{D}_k,$$

where $\boldsymbol{V}^\star$ is the boundary velocity. This type of boundary condition corresponds to a wall boundary condition.

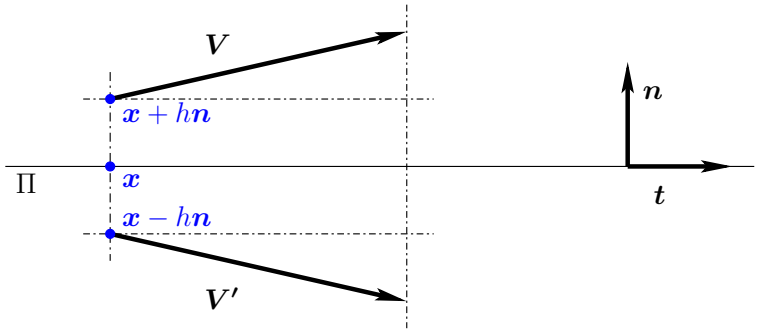


Figure 2.1: Notation related to symmetry boundary condition, for a plane of symmetry Π.

Free boundary condition on $\partial\mathcal{D}_n$

Here, the projection of the velocity gradient tensor onto the direction of the unit outward normal is prescribed as

$$(\nabla \boldsymbol{V})\boldsymbol{n} = \boldsymbol{G}^{\star}(\boldsymbol{x}, t),\ \forall \boldsymbol{x} \in \partial\mathcal{D}_n,$$

where $\boldsymbol{n}$ is the unit outward normal to the boundary and $\boldsymbol{G}^{\star}$ is a real valued vector defining the normal velocity gradient. This boundary condition is applied to outlets, where the gradient of the velocity is supposed to be equal zero in the normal direction to the boundary.

Symmetry boundary condition on $\partial\mathcal{D}_s$

Here, we assume that the flow admits a symmetry plane which is denoted Π. This plane is defined by its unit normal vector $\boldsymbol{n}$, refer to Figure 2.1. Let $\boldsymbol{V}_t$ be the tangential projection of $\boldsymbol{V}$ onto Π. We have the following decomposition of the velocity

$$\boldsymbol{V} = (\boldsymbol{V} \cdot \boldsymbol{n})\boldsymbol{n} + \boldsymbol{V}_t.$$

The symmetric of $\boldsymbol{V}$ with respect to the plane Π writes

$$\boldsymbol{V}' = -(\boldsymbol{V} \cdot \boldsymbol{n})\boldsymbol{n} + \boldsymbol{V}_t.$$

Combining the two above expressions leads to

$$\boldsymbol{V}' = \boldsymbol{V} - 2(\boldsymbol{V} \cdot \boldsymbol{n})\boldsymbol{n}.$$

Let $\boldsymbol{x}$ be a point located on the symmetry plane Π, and let $\boldsymbol{x}+h\boldsymbol{n}$ and $\boldsymbol{x}-h\boldsymbol{n}$ with $h > 0$ be two symmetrical points with respect to plane Π, refer to Figure 2.1. If the fluid flow is symmetrical with respect to the plane Π then

$$\boldsymbol{V}(\boldsymbol{x} + h\boldsymbol{n}) = \boldsymbol{V'}(\boldsymbol{x} - h\boldsymbol{n}).$$

Substituting the expression of $\boldsymbol{V}'$ in the above equation leads to

$$\boldsymbol{V}(\boldsymbol{x} + h\boldsymbol{n}) = \boldsymbol{V}(\boldsymbol{x} - h\boldsymbol{n}) - 2[\boldsymbol{V}(\boldsymbol{x} - h\boldsymbol{n}) \cdot \boldsymbol{n}]\boldsymbol{n}. \tag{2.3}$$

Using a first-order Taylor expansion yields

$$\boldsymbol{V}(\boldsymbol{x} + h\boldsymbol{n}) = \boldsymbol{V}(\boldsymbol{x}) + h\nabla\boldsymbol{V}(\boldsymbol{x})\boldsymbol{n} + O(h^2), \tag{2.4}$$

$$\boldsymbol{V}(\boldsymbol{x} - h\boldsymbol{n}) = \boldsymbol{V}(\boldsymbol{x}) - h\nabla\boldsymbol{V}(\boldsymbol{x})\boldsymbol{n} + O(h^2). \tag{2.5}$$

Substituting (2.4) and (2.5) into (2.3) leads to

$$\boldsymbol{V}(\boldsymbol{x}) + h\nabla\boldsymbol{V}(\boldsymbol{x})\boldsymbol{n} = \boldsymbol{V}(\boldsymbol{x}) - h\nabla\boldsymbol{V}(\boldsymbol{x})\boldsymbol{n} - 2[(\boldsymbol{V}(\boldsymbol{x}) - h\nabla\boldsymbol{V}(\boldsymbol{x})\boldsymbol{n})\cdot\boldsymbol{n}]\boldsymbol{n} + O(h^2).$$

The above equation has to be satisfied up to first-order, then for all $h > 0$ there holds

$$2(\boldsymbol{V}(\boldsymbol{x})\cdot\boldsymbol{n})\boldsymbol{n} + 2h\nabla\boldsymbol{V}(\boldsymbol{x})\boldsymbol{n} - 2h(\nabla\boldsymbol{V}(\boldsymbol{x})\boldsymbol{n}\cdot\boldsymbol{n})\boldsymbol{n} = 0.$$

This is equivalent to the following conditions

$$\begin{aligned}(\boldsymbol{V}(\boldsymbol{x})\cdot\boldsymbol{n})\boldsymbol{n} &= \boldsymbol{0},\\ \nabla\boldsymbol{V}(\boldsymbol{x})\boldsymbol{n} - (\nabla\boldsymbol{V}(\boldsymbol{x})\boldsymbol{n}\cdot\boldsymbol{n})\boldsymbol{n} &= \boldsymbol{0},\end{aligned}$$

Finally, we claim that the boundary conditions to apply on a symmetry boundary condition are respectively $\boldsymbol{V}\cdot\boldsymbol{n} = 0$ and $(\nabla\boldsymbol{V}\boldsymbol{n})\cdot\boldsymbol{t} = 0$ for all $\boldsymbol{x}\in\partial\mathcal{D}_s$. Here, $\boldsymbol{t}$ is the unit tangent vector which defines Π in the case of a two-dimensional geometry. In the case of a three-dimensional geometry, we should write $(\nabla\boldsymbol{V}\boldsymbol{n})\cdot\boldsymbol{t}_1 = 0$ and $(\nabla\boldsymbol{V}\boldsymbol{n})\cdot\boldsymbol{t}_2 = 0$, where $(\boldsymbol{t}_1,\boldsymbol{t}_2)$ is an orthonormal basis of Π.

2.2 Mathematical properties of the constitutive law

In this paragraph we are going to study the invertibility of the constitutive law which defines the viscous flux of the Navier-Stokes equations. Let us begin with some definitions. Let $\mathcal{T}$ be the space of general second-order tensors with components in $\mathfrak{R}^d$ with $d\in\{2,3\}$. Let us note $\mathcal{T}_s$ the space of symmetric second-order tensors defined by

$$\mathcal{T}_s = \left\{\mathbb{T}\in\mathcal{T};\mathbb{T}^t = \mathbb{T}\right\}. \tag{2.6}$$

We note $\mathcal{T}_s^0$ the space of traceless symmetric second-order tensors which writes

$$\mathcal{T}_s^0 = \left\{\mathbb{T}\in\mathcal{T}_s;\operatorname{tr}\mathbb{T} = 0\right\}. \tag{2.7}$$

We define the fourth-order tensor $\bar{\bar{\mathbb{C}}}$ as:

$$\begin{array}{rccl}\bar{\bar{\mathbb{C}}}: & \mathcal{T} & \longrightarrow & \mathcal{T}\\ & \mathbb{T} & \longmapsto & \bar{\bar{\mathbb{C}}}\mathbb{T} = 2\mu\left[\frac{1}{2}\left(\mathbb{T}+\mathbb{T}^t\right) - \frac{1}{3}(\operatorname{tr}\mathbb{T})\mathbb{I}\right].\end{array} \tag{2.8}$$

The constitutive law (2.2) is expressed in terms of $\bar{\bar{\mathbb{C}}}$ as follows:

$$\mathbb{S} = \bar{\bar{\mathbb{C}}}\nabla\boldsymbol{V}. \tag{2.9}$$

Determination of the kernel of $\bar{\bar{\mathbb{C}}}$

Let us study the invertibility of the above constitutive law, by determining the kernel of $\bar{\bar{\mathbb{C}}}$. To this end, we compute the solution of $\bar{\bar{\mathbb{C}}}\mathbb{T} = \mathbb{0}$ by developing the two terms $\frac{1}{2}\left(\mathbb{T}+\mathbb{T}^t\right)$ and $\frac{1}{3}(\operatorname{tr}\mathbb{T})\mathbb{I}$ onto a Cartesian basis as follows

$$\frac{1}{2}\left(\mathbb{T}+\mathbb{T}^t\right) = \frac{1}{2}\begin{pmatrix}2T_{11} & T_{12}+T_{21} & T_{13}+T_{31}\\ T_{12}+T_{21} & 2T_{22} & T_{23}+T_{32}\\ T_{23}+T_{32} & T_{13}+T_{31} & 2T_{33}\end{pmatrix},$$

and

$$\frac{1}{3}(\operatorname{tr}\mathbb{T})\mathbb{I} = \frac{1}{3}\begin{pmatrix}T_{11}+T_{22}+T_{33} & 0 & 0\\ 0 & T_{11}+T_{22}+T_{33} & 0\\ 0 & 0 & T_{11}+T_{22}+T_{33}\end{pmatrix}.$$

Finding the kernel of $\bar{\bar{\mathbb{C}}}$ amounts to solve the linear system

$$\left\{\begin{array}{rcl} 2T_{11}-T_{22}-T_{33} &=& 0\\ T_{12}+T_{21} &=& 0\\ T_{13}+T_{31} &=& 0\\ 2T_{22}-T_{11}-T_{33} &=& 0\\ T_{23}+T_{32} &=& 0\\ 2T_{33}-T_{11}-T_{22} &=& 0 \end{array}\right.$$

which collapses to

$$\left\{\begin{array}{rcl} T_{11} &=& T_{22}\\ T_{22} &=& T_{33}\\ T_{12}+T_{21} &=& 0\\ T_{13}+T_{31} &=& 0\\ T_{23}+T_{32} &=& 0 \end{array}\right.$$

Finally, the elements of the kernel of $\bar{\bar{\mathbb{C}}}$ are second-order tensors of the form

$$\mathbb{T} = \alpha\mathbb{I} + \mathbb{W}, \tag{2.10}$$

where $\alpha \in \mathfrak{R}$ and $\mathbb{W}$ is a generic skew symmetric second-order tensor, *i.e.*, $\mathbb{W}^t = -\mathbb{W}$. Let us point out that the elements of $\mathrm{Ker}\bar{\bar{\mathbb{C}}}$ correspond respectively to pure dilatational deformation, *i.e.*, $\alpha\mathbb{I}$ and to rigid body rotation, *i.e.*, $\mathbb{W}$.

Dimension of the kernel of $\bar{\bar{\mathbb{C}}}$

We pursue by determining the dimension of this kernel, using the characterization of the kernel defined by (2.10). First, the dimension of the space of symmetric tensors is $\dim \mathcal{T}_s = \frac{1}{2}d(d+1)$ and the dimension of the space of skew symmetric tensors is $\frac{1}{2}d(d-1)$. Thus, the dimension of the kernel of $\bar{\bar{\mathbb{C}}}$ is

$$\dim \mathrm{Ker}\,\bar{\bar{\mathbb{C}}} = 1 + \frac{1}{2}d(d-1). \tag{2.11}$$

The dimension of $\mathcal{T}$ being d^2, it is clear that $\bar{\bar{\mathbb{C}}}$ is not invertible over $\mathcal{T}$.

Comment 16: *We recall that* $\dim \mathcal{T} = d^2$, $\dim \mathcal{T}_s = \frac{d}{2}(d+1)$ *and* $\dim \mathcal{T}_s^0 = \dim \mathcal{T}_s - 1 = \frac{d}{2}(d+1) - 1$. *Knowing that* $\dim \mathrm{Ker}\,\bar{\bar{\mathbb{C}}} = 1 + \frac{1}{2}d(d-1)$, *the rank theorem leads to* $\dim \mathrm{Im}\,\bar{\bar{\mathbb{C}}} = \frac{1}{2}d(d+1) - 1 = \dim \mathcal{T}_s^0$.

Invertibility of $\bar{\bar{\mathbb{C}}}$ over $\mathcal{T}_s^0$

By virtue of comment 16, if we restrict the definition of $\bar{\bar{\mathbb{C}}}$ to the space $\mathcal{T}_s^0$ then $\bar{\bar{\mathbb{C}}}$ becomes an invertible operator. We will first verify that the definition (2.10) of the kernel space restricted to $\mathcal{T}_s^0$ is reduced to $\mathbb{0}$. Then we will prove that, over the space of symmetric traceless tensors $\mathcal{T}_s^0$, the fourth-order tensor $\bar{\bar{\mathbb{C}}}$ is invertible.

It is trivial to prove that the kernel of $\bar{\bar{\mathbb{C}}}$ defined on the space of traceless symmetric second-order tensors $\mathcal{T}_s^0$ is reduced to $\mathbb{0}$. We can check it directly.
For all $\mathbb{T} \in \mathcal{T}_s^0$ we have $\mathrm{tr}\,\mathbb{T} = 0$ and $\mathbb{T} = \mathbb{T}^t$. If we take $\mathbb{T} \in \mathrm{Ker}\,\bar{\bar{\mathbb{C}}}$ we also have $\mathbb{T} = \alpha\mathbb{I} + \mathbb{W}$ with $\mathbb{W}^t = -\mathbb{W}$.
We compute the transpose of $\mathbb{T}$ to obtain $\mathbb{T}^t = \alpha\mathbb{I} - \mathbb{W}$. Because $\mathbb{T} = \mathbb{T}^t$ we conclude that $\mathbb{W} = \mathbb{0}$. Thus $\mathbb{T}$ writes under the form $\mathbb{T} = \alpha\mathbb{I}$.
Let us compute the trace of $\mathbb{T}$, $\mathrm{tr}\,\mathbb{T} = \alpha d$ and $\mathrm{tr}\,\mathbb{T} = 0$ which means that $\alpha = 0$. It remains

$\mathbb{T} = \mathbb{0}$, which ends the proof.

Let us recall the definition of the inner product of two tensors $\mathbb{T}$ and $\mathbb{Q} \in \mathcal{T}$

$$\mathbb{T} : \mathbb{Q} = \mathrm{tr}\left(\mathbb{Q}^t \mathbb{T}\right), \tag{2.12}$$

which express in terms of the tensors components as

$$\mathbb{T} : \mathbb{Q} = \sum_{i,j} \mathbb{T}_{ij} \mathbb{Q}_{ij}. \tag{2.13}$$

For all $\mathbb{T} \in \mathcal{T}_s^0$ the constitutive law (2.2) simplifies to $\bar{\bar{\mathbb{C}}}\mathbb{T} = 2\mu\mathbb{T}_0$. Let us compute the inner product with $\mathbb{T}$:

$$\begin{aligned}
\bar{\bar{\mathbb{C}}}\mathbb{T} : \mathbb{T} &= 2\mu\mathbb{T}_0 : \left(\mathbb{T}_0 + \frac{1}{3}(\mathrm{tr}\,\mathbb{T})\mathbb{I}\right) \\
&= 2\mu\mathbb{T}_0 : \mathbb{T}_0 \\
&= 2\mu|\mathbb{T}_0|^2.
\end{aligned}$$

Where $|.|$ is the norm associated with the inner product. Namely $|\mathbb{T}| = \sqrt{\mathbb{T} : \mathbb{T}} = \sqrt{\sum_{i,j} \mathbb{T}_{ij}^2}$.

Thus, for all $\mathbb{T} \in \mathcal{T}_s^0$ we have the following bounds

$$2\underline{\mu}|\mathbb{T}_0|^2 \leq \bar{\bar{\mathbb{C}}}\mathbb{T} : \mathbb{T} \leq 2\overline{\mu}|\mathbb{T}_0|^2, \tag{2.14}$$

where $\underline{\mu} = \min_{\boldsymbol{x} \in \Re^d} \mu(\boldsymbol{x})$ and $\overline{\mu} = \max_{\boldsymbol{x} \in \Re^d} \mu(\boldsymbol{x})$.

Invertibility of $\bar{\bar{\mathbb{C}}}$ over $\mathcal{T}$

What can we say about the invertibility of the constitutive law over the whole space of second-order tensors?

For all $\mathbb{T} \in \mathcal{T}$ we have:

$$\begin{aligned}
\bar{\bar{\mathbb{C}}}\mathbb{T} : \mathbb{T} &= \bar{\bar{\mathbb{C}}}\frac{\mathbb{T} + \mathbb{T}^t}{2} : \frac{\mathbb{T} + \mathbb{T}^t}{2}, \\
&= 2\mu \left[\left|\frac{\mathbb{T} + \mathbb{T}^t}{2}\right|^2 - \frac{1}{3}(\mathrm{tr}\,\mathbb{T})^2\right], \\
&\geq 2\underline{\mu} \left[\frac{1}{4}\left(|\mathbb{T}|^2 + 2\mathbb{T} : \mathbb{T}^t + |\mathbb{T}^t|^2\right) - \frac{1}{3}(\mathrm{tr}\,\mathbb{T})^2\right] \text{ thanks to (2.14)}, \\
&\geq \underline{\mu} \left[|\mathbb{T}|^2 + \mathbb{T} : \mathbb{T}^t - \frac{2}{3}(\mathrm{tr}\,\mathbb{T})^2\right].
\end{aligned} \tag{2.15}$$

We cannot say anything about the sign of the right hand side. This shows that for a generic tensor we cannot conclude that $\bar{\bar{\mathbb{C}}}$ is invertible. This is why in the next section we introduce a modification of the constitutive law to obtain an equivalent invertible constitutive law on $\mathcal{T}$.

2.3 Construction of an invertible constitutive law

In this section, we will focus on the construction of an equivalent invertible constitutive law. It is a very important issue regarding the construction of our numerical scheme. As for the CCLAD scheme presented in Chapter 1, the numerical approximation of the viscous fluxes should be based on a sub-cell variational formulation of the constitutive law, and the elimination of the edge-based auxiliary unknowns should require the inversion of nodal linear systems. Knowing that the invertibility of the nodal linear systems is directly linked to the invertibility of the constitutive law. Thus, it is crucial to ensure its invertibility. Arnold [16], in the domain of linear elasticity, has faced the same difficulties. The Hellinger-Reissner variational formulation he had to solve was defined on the space of symmetric $d \times d$ tensors, and the construction of suitable Finite Element methods for such problems was extremely difficult. In [16], Arnold has introduced a modification of the constitutive law to be able to write an equivalent formulation of the Hellinger-Reissner variational principle defined on the space of generic $d \times d$ tensors. He was then able to apply classical Finite Element method to this problem. In this section, we adapt his methodology to our problem.

Let us define the fourth-order tensor $\bar{\bar{\mathbb{D}}}$ as

$$\bar{\bar{\mathbb{D}}}\mathbb{T} = (\mathrm{tr}\,\mathbb{T})\mathbb{I} - \mathbb{T}^t, \forall \mathbb{T} \in \mathcal{T}. \tag{2.16}$$

Let us compute the inner product with $\mathbb{T}$ to obtain

$$\bar{\bar{\mathbb{D}}}\mathbb{T} : \mathbb{T} = (\mathrm{tr}\,\mathbb{T})^2 - \mathbb{T}^t : \mathbb{T}. \tag{2.17}$$

Then we define $\bar{\bar{\mathbb{R}}}\mathbb{T} = \bar{\bar{\mathbb{C}}}\mathbb{T} + \beta\bar{\bar{\mathbb{D}}}\mathbb{T}$, where $\beta \in \mathfrak{R}$, and compute its inner product with $\mathbb{T}$

$$\begin{aligned}\bar{\bar{\mathbb{R}}}\mathbb{T} : \mathbb{T} &= \bar{\bar{\mathbb{C}}}\mathbb{T} : \mathbb{T} + \beta\bar{\bar{\mathbb{D}}}\mathbb{T} : \mathbb{T}, \\ &\geq \underline{\mu}\left[|\mathbb{T}|^2 + \mathbb{T} : \mathbb{T}^t - \frac{2}{3}(\mathrm{tr}\,\mathbb{T})^2\right] + \beta(\mathrm{tr}\,\mathbb{T})^2 - \beta\mathbb{T} : \mathbb{T}^t, \text{ thanks to (2.15).}\end{aligned}$$

Choosing $\beta = \underline{\mu}$ leads to

$$\bar{\bar{\mathbb{R}}}\mathbb{T} : \mathbb{T} \geq \underline{\mu}\left[|\mathbb{T}|^2 + \frac{1}{3}(\mathrm{tr}\,\mathbb{T})^2\right] \geq 0. \tag{2.18}$$

Comment 17: *Thanks to this manipulation the fourth-order tensor $\bar{\bar{\mathbb{R}}}$ is positive definite on the space of second-order tensors $\mathcal{T}$ and is thus invertible.*

We have defined a positive definite operator, let us show how to use it in our initial problem. Let us first introduce $\mathbb{\Sigma} = \bar{\bar{\mathbb{R}}}\nabla\boldsymbol{V} = (\bar{\bar{\mathbb{C}}} + \beta\bar{\bar{\mathbb{D}}})\nabla\boldsymbol{V}$. We will show that $\mathbb{\Sigma}$ and $\mathbb{S}$ satisfy the same divergence equation.

Recalling the tensorial identity $\nabla \cdot (\nabla\boldsymbol{V})^t = \nabla(\nabla \cdot \boldsymbol{V})$ for all $\boldsymbol{V} \in \mathfrak{R}^d$ we easily show that

$$\begin{aligned}\nabla \cdot (\bar{\bar{\mathbb{D}}}\nabla\boldsymbol{V}) &= \nabla \cdot [\mathrm{tr}(\nabla\boldsymbol{V})\mathbb{I}] - \nabla \cdot (\nabla\boldsymbol{V})^t, \\ &= \nabla(\nabla \cdot \boldsymbol{V}) - \nabla(\nabla \cdot \boldsymbol{V}), \\ &= \boldsymbol{0}.\end{aligned}$$

This implies that $\mathbb{\Sigma} = (\bar{\bar{\mathbb{C}}} + \beta\bar{\bar{\mathbb{D}}})\nabla\boldsymbol{V}$ satisfies

$$\begin{aligned}\nabla \cdot \mathbb{\Sigma} &= \nabla \cdot (\bar{\bar{\mathbb{C}}}\nabla\boldsymbol{V}), \\ &= \nabla \cdot \mathbb{S}.\end{aligned}$$

which ends the proof.

We claim that the fourth-order tensor $\bar{\bar{\mathbb{R}}} = \bar{\bar{\mathbb{C}}} + \beta\bar{\bar{\mathbb{D}}}$, where $\beta = \underline{\mu}$, is positive definite and satisfies $\bar{\bar{\mathbb{R}}}\mathbb{T} : \mathbb{T} \geq \underline{\mu}\left[|\mathbb{T}|^2 + \frac{1}{3}\left(\operatorname{tr}\mathbb{T}\right)^2\right]$ for all $\mathbb{T} \in \mathcal{T}$. This allows us to define the invertible constitutive law $\mathbb{\Sigma} = \bar{\bar{\mathbb{R}}}\nabla\boldsymbol{V}$ such that $\nabla \cdot \mathbb{\Sigma} = \nabla \cdot \mathbb{S}$.

In what follows, we study the discretization of the following initial boundary value problem:

$$\rho\frac{\partial \boldsymbol{V}}{\partial t} - \nabla \cdot \mathbb{\Sigma} = \boldsymbol{0}, \qquad \forall (\boldsymbol{x}, t) \in \mathcal{D} \times [0, \mathcal{T}], \tag{2.19a}$$

$$\boldsymbol{V}(\boldsymbol{x}, 0) = \boldsymbol{V}^0(\boldsymbol{x}), \qquad \forall \boldsymbol{x} \in \mathcal{D}, \tag{2.19b}$$

with the constitutive law given by:

$$\mathbb{\Sigma} = \bar{\bar{\mathbb{R}}}\nabla\boldsymbol{V}. \tag{2.20}$$

The boundary conditions defined earlier remains identical. They were defined using the velocity vector or the velocity gradient tensor, thus the modification of the constitutive law has no impact on it.

2.4 Equivalence of the two formulations

In this section we aim at showing that the new invertible constitutive law $\mathbb{\Sigma}$ is equivalent to the constitutive law $\mathbb{S}$. We start by finding an expression of $\mathbb{S}$ in terms of $\mathbb{\Sigma}$, then we will list the properties of the two formulations in order to prove their equivalence.

2.4.1 Expression of $\mathbb{S}$ in terms of $\mathbb{\Sigma}$

Let us recall the definition of $\mathbb{S} = \bar{\bar{\mathbb{C}}}\nabla\boldsymbol{V}$ and $\mathbb{\Sigma} = \bar{\bar{\mathbb{R}}}\nabla\boldsymbol{V}$ where $\bar{\bar{\mathbb{R}}}\mathbb{T} = \bar{\bar{\mathbb{C}}}\mathbb{T} + \beta\bar{\bar{\mathbb{D}}}\mathbb{T}$, $\forall \mathbb{T} \in \mathcal{T}$.
We have $\bar{\bar{\mathbb{D}}}\mathbb{T} = (\operatorname{tr}\mathbb{T})\mathbb{I} - \mathbb{T}^t$ and $\bar{\bar{\mathbb{C}}}\mathbb{T} = 2\mu\left[\frac{1}{2}\left(\mathbb{T} + \mathbb{T}^t\right) - \frac{1}{3}(\operatorname{tr}\mathbb{T})\mathbb{I}\right]$.
We observe that $\bar{\bar{\mathbb{R}}}^{-1}\mathbb{\Sigma} = \nabla\boldsymbol{V}$, so $\mathbb{\Sigma}$ rewrites:

$$\begin{aligned}
\mathbb{\Sigma} &= \bar{\bar{\mathbb{R}}}\nabla\boldsymbol{V}, \\
&= \left(\bar{\bar{\mathbb{C}}} + \beta\bar{\bar{\mathbb{D}}}\right)\nabla\boldsymbol{V}, \\
&= \mathbb{S} + \beta\bar{\bar{\mathbb{D}}}\bar{\bar{\mathbb{R}}}^{-1}\mathbb{\Sigma}.
\end{aligned}$$

Finally the expression of $\mathbb{S}$ in terms of $\mathbb{\Sigma}$ writes

$$\mathbb{S} = \mathbb{\Sigma} - \beta\bar{\bar{\mathbb{D}}}\bar{\bar{\mathbb{R}}}^{-1}\mathbb{\Sigma}. \tag{2.21}$$

To compute the right-hand side of (2.21) we need to determine $\bar{\bar{\mathbb{R}}}^{-1}$. Let us start by developing the term $\bar{\bar{\mathbb{R}}}\mathbb{T}$. We have

$$\begin{aligned}
\bar{\bar{\mathbb{R}}}\mathbb{T} &= \mu\left(\mathbb{T} + \mathbb{T}^t\right) - \frac{2\mu}{3}(\operatorname{tr}\mathbb{T})\mathbb{I} + \beta(\operatorname{tr}\mathbb{T})\mathbb{I} - \beta\mathbb{T}^t, \\
&= \mu\mathbb{T} + (\mu - \beta)\mathbb{T}^t + \frac{1}{3}(3\beta - 2\mu)(\operatorname{tr}\mathbb{T})\mathbb{I}.
\end{aligned}$$

We use the decomposition of $\mathbb{T}$ in terms of its deviatoric part $\mathbb{T}_0$ to write

$$\begin{aligned}
\bar{\bar{\mathbb{R}}}\mathbb{T} &= \mu\mathbb{T}_0 + (\mu - \beta)\mathbb{T}_0^t + \frac{1}{3}(3\beta - 2\mu + \mu + \mu - \beta)(\operatorname{tr}\mathbb{T})\mathbb{I}, \\
&= \mu\mathbb{T}_0 + (\mu - \beta)\mathbb{T}_0^t + \frac{2}{3}\beta(\operatorname{tr}\mathbb{T})\mathbb{I}.
\end{aligned}$$

Let us note $\mathbb{Q} \in \mathcal{T}$ the right-hand side of the previous equation, namely $\bar{\bar{\mathbb{R}}}\mathbb{T} = \mathbb{Q}$. The decomposition $\mathbb{Q} = \mathbb{Q}_0 + \frac{1}{3}(\operatorname{tr}\mathbb{Q})\mathbb{I}$ leads to

$$\begin{aligned} \mu\mathbb{T}_0 + (\mu-\beta)\mathbb{T}_0^t &= \mathbb{Q}_0, \\ 2\beta \operatorname{tr}\mathbb{T} &= \operatorname{tr}\mathbb{Q}. \end{aligned}$$

The two second-order tensors $\mathbb{T}_0$ and $\mathbb{T}_0^t$ satisfy the following linear system

$$\begin{aligned} \mu\mathbb{T}_0 + (\mu-\beta)\mathbb{T}_0^t &= \mathbb{Q}_0, \\ (\mu-\beta)\mathbb{T}_0 + \mu\mathbb{T}_0^t &= \mathbb{Q}_0^t. \end{aligned}$$

Using linear combinations of the previous equations we obtain

$$\begin{aligned} \mu^2\mathbb{T}_0 + \mu(\mu-\beta)\mathbb{T}_0^t &= \mu\mathbb{Q}_0, \\ -(\mu-\beta)^2\mathbb{T}_0 - \mu(\mu-\beta)\mathbb{T}_0^t &= -(\mu-\beta)\mathbb{Q}_0^t. \end{aligned}$$

Summing these equations allow us to write

$$(2\mu-\beta)\beta\mathbb{T}_0 = \mu\mathbb{Q}_0 - (\mu-\beta)\mathbb{Q}_0^t,$$

which also rewrites under the more convenient form

$$\mathbb{T}_0 = \frac{1}{(2\mu-\beta)\beta}\left[\mu\mathbb{Q}_0 + (\beta-\mu)\mathbb{Q}_0^t\right], \tag{2.22}$$

provided that $2\mu - \beta \neq 0$, which is true since $\beta = \underline{\mu} = \min_{\Re}\mu$.

We recall that $\mathbb{Q}$ also have to verify the following equation

$$\operatorname{tr}\mathbb{T} = \frac{1}{2\beta}\operatorname{tr}\mathbb{Q}. \tag{2.23}$$

Since $\mathbb{T} = \mathbb{T}_0 + \frac{1}{3}(\operatorname{tr}\mathbb{T})\mathbb{I}$ we use the results obtained in (2.22) and (2.23) to write

$$\begin{aligned} \mathbb{T} &= \frac{1}{(2\mu-\beta)\beta}\left[\mu\mathbb{Q}_0 + (\beta-\mu)\mathbb{Q}_0^t\right] + \frac{1}{3}\frac{1}{2\beta}(\operatorname{tr}\mathbb{Q})\mathbb{I}, \\ &= \frac{1}{(2\mu-\beta)\beta}\left[\mu\mathbb{Q} + (\beta-\mu)\mathbb{Q}^t - \frac{1}{3}\beta(\operatorname{tr}\mathbb{Q})\mathbb{I}\right] + \frac{1}{3}\frac{1}{2\beta}(\operatorname{tr}\mathbb{Q})\mathbb{I}. \end{aligned}$$

Finally $\mathbb{T}$ is expressed in terms of $\mathbb{Q}$ under the form

$$\mathbb{T} = \frac{1}{(2\mu-\beta)\beta}\left[\mu\mathbb{Q} + (\beta-\mu)\mathbb{Q}^t\right] + \frac{1}{3}\frac{2\mu-3\beta}{2\beta(2\mu-\beta)}(\operatorname{tr}\mathbb{Q})\mathbb{I}.$$

We recall that $\mathbb{Q}$ was introduced as $\bar{\bar{\mathbb{R}}}\mathbb{T} = \mathbb{Q}$. Then, $\bar{\bar{\mathbb{R}}}^{-1}$ is then defined by $\bar{\bar{\mathbb{R}}}^{-1}\mathbb{Q} = \mathbb{T}$ and writes

$$\bar{\bar{\mathbb{R}}}^{-1}\mathbb{Q} = \frac{1}{(2\mu-\beta)\beta}\left[\mu\mathbb{Q} + (\beta-\mu)\mathbb{Q}^t\right] + \frac{1}{3}\frac{2\mu-3\beta}{2\beta(2\mu-\beta)}(\operatorname{tr}\mathbb{Q})\mathbb{I}, \ \forall\mathbb{Q} \in \mathcal{T}. \tag{2.24}$$

Now that we have the expression of $\bar{\bar{\mathbb{R}}}^{-1}$ we are able to compute $\bar{\bar{\mathbb{S}}}$ in terms of $\mathbb{\Sigma}$. Let us start by computing $\bar{\bar{\mathbb{D}}}\bar{\bar{\mathbb{R}}}^{-1}\mathbb{\Sigma}$ recalling that $\operatorname{tr}\left(\bar{\bar{\mathbb{R}}}^{-1}\mathbb{Q}\right) = \frac{1}{2\beta}\operatorname{tr}\mathbb{Q}$

$$\begin{aligned} \bar{\bar{\mathbb{D}}}\bar{\bar{\mathbb{R}}}^{-1}\mathbb{\Sigma} &= \operatorname{tr}\left(\bar{\bar{\mathbb{R}}}^{-1}\mathbb{\Sigma}\right)\mathbb{I} - \left(\bar{\bar{\mathbb{R}}}^{-1}\mathbb{\Sigma}\right)^t, \\ &= \frac{1}{3}\frac{3(2\mu-\beta)}{2\beta(2\mu-\beta)}(\operatorname{tr}\mathbb{\Sigma})\mathbb{I} - \frac{1}{(2\mu-\beta)\beta}\left[\mu\mathbb{\Sigma}^t + (\beta-\mu)\mathbb{\Sigma}\right] \\ &- \frac{1}{3}\frac{(2\mu-3\beta)}{2\beta(2\mu-\beta)}(\operatorname{tr}\mathbb{\Sigma})\mathbb{I}, \\ &= \frac{1}{3}\frac{2\mu}{\beta(2\mu-\beta)}(\operatorname{tr}\mathbb{\Sigma})\mathbb{I} - \frac{1}{(2\mu-\beta)\beta}\left[\mu\mathbb{\Sigma}^t + (\beta-\mu)\mathbb{\Sigma}\right]. \end{aligned}$$

It remains to compute $\mathbb{S}$ as follow

$$\begin{aligned}\mathbb{S} &= \mathbb{\Sigma} - \beta\bar{\bar{\mathbb{D}}}\bar{\bar{\mathbb{R}}}^{-1}\mathbb{\Sigma},\\ &= \mathbb{\Sigma} - \frac{1}{3}\frac{2\mu\beta}{(2\mu-\beta)\beta}(\operatorname{tr}\mathbb{\Sigma})\mathbb{I} + \frac{\beta}{(2\mu-\beta)\beta}\left[\mu\mathbb{\Sigma}^t + (\beta-\mu)\mathbb{\Sigma}\right],\\ &= \frac{\mu}{(2\mu-\beta)}\left[\mathbb{\Sigma}+\mathbb{\Sigma}^t\right] - \frac{1}{3}\frac{2\mu}{(2\mu-\beta)}(\operatorname{tr}\mathbb{\Sigma})\mathbb{I}.\end{aligned}$$

Finally, $\mathbb{S}$ is expressed in terms of $\mathbb{\Sigma}$ as

$$\mathbb{S} = \frac{2\mu}{(2\mu-\beta)}\left[\frac{1}{2}\left(\mathbb{\Sigma}+\mathbb{\Sigma}^t\right) - \frac{1}{3}\left(\operatorname{tr}\mathbb{\Sigma}\right)\mathbb{I}\right].$$

$\mathbb{S}$ is proportional to the deviatoric part of $\frac{1}{2}\left(\mathbb{\Sigma}+\mathbb{\Sigma}^t\right)$. Note that $\mathbb{S}$ is well defined because $2\mu-\beta>0$ due to the fact that $2\mu-\beta \geq \underline{\mu} > 0$. In addition if $\beta = 0$, then $\mathbb{S}$ collapses to the deviatoric part of $\frac{1}{2}\left(\mathbb{\Sigma}+\mathbb{\Sigma}^t\right)$.

2.4.2 Properties of the two formulations

In this section we list the properties of the two formulations, this will allow us to rule on the equivalence of the formulations.

- The constitutive law of (2.1) is characterized by the fourth-order tensor $\bar{\bar{\mathbb{C}}}$ which is not invertible over the space of second-order tensors.
- The constitutive law of (2.19) is characterized by the fourth-order tensor $\bar{\bar{\mathbb{R}}}$ which is a positive definite fourth-order tensor $\bar{\bar{\mathbb{R}}}\mathbb{T} : \mathbb{T} \geq 0,\ \forall \mathbb{T} \in \mathcal{T}$.
- The viscous stress $\mathbb{S} = \bar{\bar{\mathbb{C}}}\nabla\boldsymbol{V}$ and the pseudo viscous stress $\mathbb{\Sigma} = \bar{\bar{\mathbb{R}}}\nabla\boldsymbol{V}$ are related by $\mathbb{\Sigma} = \mathbb{S} + \beta\bar{\bar{\mathbb{D}}}\nabla\boldsymbol{V}$ and $\mathbb{S}$ is expressed linearly in terms of $\mathbb{\Sigma}$ as follows

$$\mathbb{S} = \frac{2\mu}{(2\mu-\beta)}\left[\frac{1}{2}\left(\mathbb{\Sigma}+\mathbb{\Sigma}^t\right) - \frac{1}{3}\left(\operatorname{tr}\mathbb{\Sigma}\right)\mathbb{I}\right].$$

- By virtue of the identity $\nabla\cdot(\nabla\boldsymbol{V})^t = \nabla(\nabla\cdot\boldsymbol{V})$ one can show that $\nabla\cdot(\bar{\bar{\mathbb{D}}}\nabla\boldsymbol{V}) = \boldsymbol{0}$ and thus

$$\nabla\cdot\mathbb{\Sigma} = \nabla\cdot\mathbb{S}$$

- Finally, the boundary conditions of (2.1) and (2.19) do not depend on the constitutive laws, they are identical in the two formulations.

These properties allow us to conclude that (2.1) and (2.19) share the same solutions (provided that a solution exists). The equation (2.19) is a perturbation of (2.1) parametrized by β which is characterized by a positive definite constitutive law. This positive definiteness is of great importance to derive a robust cell-centered finite volume discretization.

Before proceeding to the construction of the numerical scheme we conclude this section by stating some useful remarks about the properties of the modified formulation (2.19).

- While $\mathbb{S}$ was symmetric by construction, $\mathbb{\Sigma}$ is not. This information will be of great importance when choosing the numerical methods to solve the linear systems in our numerical scheme.

- The generic tensorial diffusion equation (2.19) satisfies a principle of dissipation of kinetic energy.
  Dot-multiplying (2.19a) by $\boldsymbol{V}$ leads to
  $$\rho\frac{\partial}{\partial t}(\frac{\boldsymbol{V}^2}{2}) - (\nabla\cdot\mathbb{\Sigma})\cdot\boldsymbol{V} = 0,$$
  using the tensorial identity $\nabla\cdot(\mathbb{\Sigma}^t\boldsymbol{V}) = \boldsymbol{V}\cdot(\nabla\cdot\mathbb{\Sigma}) + \mathbb{\Sigma}:\nabla\boldsymbol{V}$ allows to rewrite the above equation as
  $$\rho\frac{\partial}{\partial t}(\frac{\boldsymbol{V}^2}{2}) - \nabla\cdot(\mathbb{\Sigma}^t\boldsymbol{V}) + \mathbb{\Sigma}:\nabla\boldsymbol{V} = 0. \tag{2.25}$$
  Integrating the above equation over the region ω yields
  $$\frac{d}{dt}\int_\omega \rho\frac{\boldsymbol{V}^2}{2}dv + \int_{\partial\omega}\boldsymbol{V}\cdot\mathbb{\Sigma}\boldsymbol{n}ds = -\int_\omega \bar{\bar{\mathbb{R}}}\nabla\boldsymbol{V}:\nabla\boldsymbol{V}.$$
  Now, assuming boundary conditions such that either $\mathbb{\Sigma}\boldsymbol{n} = \boldsymbol{0}$ or $\boldsymbol{V} = \boldsymbol{0}$ on $\partial\omega$ and observing that $\bar{\bar{\mathbb{R}}}$ is positive definite, we finally get the following inequality
  $$\frac{d}{dt}\int_\omega \rho\frac{\boldsymbol{V}^2}{2}dv \leq 0.$$
  This shows that the kinetic energy dissipation is ensured by the generic tensorial diffusion equation (2.19).

2.5 Construction of a Finite Volume scheme for tensorial diffusion

Now that we showed the equivalence of the formulations (2.1) and (2.19), we will address the construction of the numerical scheme. From now on we will use the modified formulation (2.19) that we recall here

$$\begin{aligned} \rho\frac{\partial\boldsymbol{V}}{\partial t} - \nabla\cdot\mathbb{\Sigma} &= \boldsymbol{0}, && \forall(\boldsymbol{x},t)\in\mathcal{D}\times[0,\mathcal{T}], \\ \boldsymbol{V}(\boldsymbol{x},0) &= \boldsymbol{V}^0(\boldsymbol{x}), && \forall\boldsymbol{x}\in\mathcal{D}, \end{aligned}$$

with the constitutive law given by:

$$\mathbb{\Sigma} = \bar{\bar{\mathbb{R}}}\nabla\boldsymbol{V}.$$

The boundary conditions can be kinematic, symmetrical or of Neumann kind. For simplicity reasons we will describe the construction of the numerical scheme in two dimensions of space. **The three-dimensional version of the scheme will be straightforward to implement using the notations employed in the previous chapter for the three-dimensional version of the CCLAD scheme.**

We will start this section by introducing useful notations to develop the numerical scheme, we will continue by explaining the expression of a second-order tensor in terms of its projections on a basis. This is the tensorial counterpart of the methodology used for vectors in Chapter 1. We will then build a sub-cell variational formulation of the constitutive law, which will leads us to an expression of the numerical fluxes at the half-edges using auxiliary unknowns. The half-edges auxiliary unknowns can then be eliminated using the continuity conditions around each node, which leads to the construction of a global linear system. We will then discuss the mathematical properties of the numerical scheme and a section will be devoted to the verification of the equivalence of the two-formulations (2.1) and (2.19) at the discrete level. Finally, the robustness and the accuracy of the scheme are assessed using various representative test cases.

2.5.1 Notations

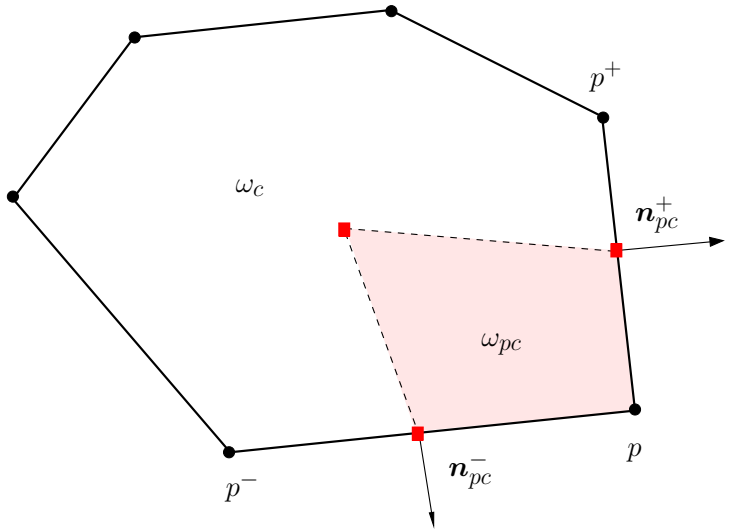


Figure 2.2: Notations related to a generic cell ω_c and one of its sub-cell ω_{pc}.

We consider a partition $\cup_c \omega_c$ of the computational domain $\mathcal{D}$ into polyhedral cells ω_c. In the following a cell is indifferently denoted ω_c or by its index c. The list of points p belonging to a cell c is denoted by $\mathcal{P}(c)$. We note $\mathcal{C}(p)$ the set which contains the cells surrounding a point p. In the counterclockwise ordered list of points of cell c, p^- denotes the point before p and p^+ the point following p as pictured in Figure 2.2.

We define the sub-cell ω_{pc} as the quadrangle obtained by connecting the point p, the mid point of $[p, p^+]$, the centroid of cell c and the mid-point of $[p, p^-]$. It is trivial to realize that $\omega_c = \bigcup_{p\in\mathcal{P}(c)} \omega_{pc}$.

We note $\partial\omega_{pc}^{+/-}$ the half-edge obtained by connecting point p to the mid point of $[p, p^{+/-}]$ and $\boldsymbol{n}_{pc}^{+/-}$ its unit outward pointing normal. Finally we note $l_{pc}^{+/-}$ the length of the half-edge $\partial\omega_{pc}^{+/-}$.

Using these notations, we integrate (2.19a) over the cell ω_c, then we employ the divergence theorem to obtain

$$\frac{d}{dt}(m_c \boldsymbol{V}_c) - \int_{\partial\omega_c} \mathbb{\Sigma}\boldsymbol{n}\,\mathrm{d}s = \boldsymbol{0}, \tag{2.27}$$

where $m_c = \int_{\omega_c} \rho\,\mathrm{d}v$ is the mass of cell ω_c and $\boldsymbol{V}_c = \frac{1}{|\omega_c|}\int_{\omega_c} \boldsymbol{V}\,\mathrm{d}v$ the averaged velocity over the cell and $\boldsymbol{n}$ is the unit outward pointing normal to $\partial\omega_c$ the boundary of ω_c.

The partition of the cell boundary $\partial\omega_c$, in terms of the sub-cells outer boundary writes

$$\partial\omega_c = \bigcup_{p\in\mathcal{P}(c)} (\partial\omega_{pc}^- \cup \partial\omega_{pc}^+). \tag{2.28}$$

Employing (2.28) in the second term of (2.27) leads to

$$\int_{\partial\omega_c} \mathbb{\Sigma}\boldsymbol{n}\,\mathrm{d}s = \sum_{p\in\mathcal{P}(c)} \int_{\partial\omega_{pc}^-} \mathbb{\Sigma}\boldsymbol{n}\,\mathrm{d}s + \int_{\partial\omega_{pc}^+} \mathbb{\Sigma}\boldsymbol{n}\,\mathrm{d}s. \tag{2.29}$$

We introduce the half-edge fluxes $\boldsymbol{\Sigma}_{pc}^{+/-}$ which are defined as

$$\boldsymbol{\Sigma}_{pc}^{+/-} = \frac{1}{l_{pc}^{+/-}} \int_{\partial\omega_{pc}^{+/-}} \mathbb{\Sigma}\boldsymbol{n}\,\mathrm{d}s. \tag{2.30}$$

Substituting (2.30) and (2.29) in (2.27) yields

$$\frac{d}{dt}(m_c\boldsymbol{V}_c) - \sum_{p\in\mathcal{P}(c)} l_{pc}^{-}\boldsymbol{\Sigma}_{pc}^{-} + l_{pc}^{+}\boldsymbol{\Sigma}_{pc}^{+} = \boldsymbol{0}. \tag{2.31}$$

To achieve the space discretization of (2.19) it remains to construct an approximation of the half-edge fluxes. This approximation should take the form

$$\boldsymbol{\Sigma}_{pc}^{+/-} = \boldsymbol{H}_{pc}^{+/-}(\boldsymbol{V}_{pc}^{-} - \boldsymbol{V}_c, \boldsymbol{V}_{pc}^{+} - \boldsymbol{V}_c), \tag{2.32}$$

where $\boldsymbol{V}_{pc}^{+/-}$ are the half-edge velocities defined by

$$\boldsymbol{V}_{pc}^{+/-} = \frac{1}{l_{pc}^{+/-}} \int_{\partial\omega_{pc}^{+/-}} \boldsymbol{V}\,\mathrm{d}s. \tag{2.33}$$

These half-edge velocities, pictured in Figure 2.4, are auxiliary unknowns which will be eliminated using the continuity conditions across the cell interfaces, namely the continuity of the velocity field $\boldsymbol{V}$ and the continuity of the half-edge fluxes $\mathbb{\Sigma}\boldsymbol{n}$.

To exhibit these continuity conditions, let us consider two neighboring cells, denoted by subscripts c and d, which share a given edge, refer to Figure 2.3. This edge corresponds to the segment $[p, p^+]$, where p and p^+ are two consecutive points in the counterclockwise numbering attached to cell c. It also corresponds to the segment $[r^-, r]$, where r^- and r are two consecutive points in the counterclockwise numbering attached to cell d. Obviously, these four labels define the same edge and thus their corresponding points coincide, *i.e.*, $p \equiv r$, $p^+ \equiv r^-$. The sub-cell of cell c attached to point $p \equiv r$ is denoted ω_{pc}, whereas the sub-cell of cell d attached to point $r \equiv p$ is denoted ω_{rd}. This double notation, in spite of its heaviness, allows to define precisely the half-edge fluxes and velocities at the half-edge corresponding to the intersection of the two previous sub-cells. Namely, viewed from sub-cell ω_{pc} (resp. ω_{rd}), the half-edge flux and velocity are denoted $\boldsymbol{\Sigma}_{pc}^{+}$ and $\boldsymbol{V}_{pc}^{+}$ (resp. $\boldsymbol{\Sigma}_{rd}^{-}$ and $\boldsymbol{V}_{rd}^{-}$). Bearing this notation in mind, continuity conditions at the half-edge $(\omega_{pc} \cup \omega_{rd})$ for the half-edge fluxes and velocities write explicitly as

$$\boldsymbol{\Sigma}_{pc}^{+} + \boldsymbol{\Sigma}_{rd}^{-} = \boldsymbol{0}, \tag{2.34a}$$

$$\boldsymbol{V}_{pc}^{+} = \boldsymbol{V}_{rd}^{-}. \tag{2.34b}$$

The continuity condition for the flux follows from the definition of the unit outward normals related to $(\omega_{pc} \cup \omega_{rd})$, *i.e.*, $\boldsymbol{n}_{pc}^{+} = -\boldsymbol{n}_{rd}^{-}$.

We are seeking an approximation of the half-edge fluxes $\boldsymbol{\Sigma}_{pc}^{+/-}$ in terms of the half-edge velocities $\boldsymbol{V}_{pc}^{+/-}$ and the cell-centered velocity $\boldsymbol{V}_c$. To this end we will develop a sub-cell based variational formulation. Before proceeding any further we show how to express a second-order tensor in terms of its projection onto a given basis.

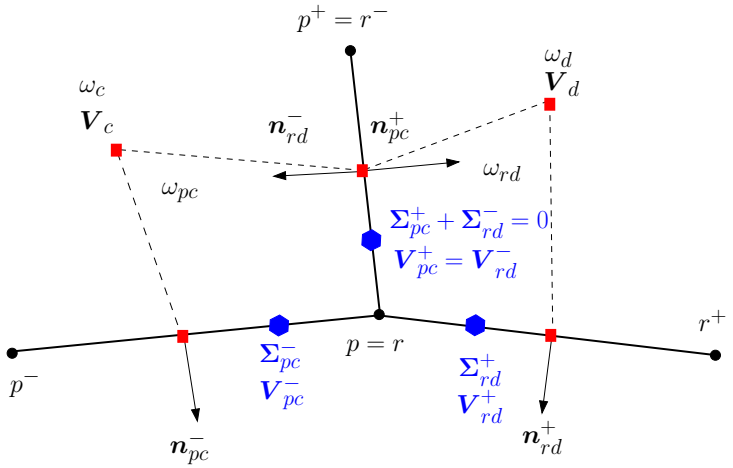


Figure 2.3: Continuity conditions for the half-edges fluxes $\boldsymbol{\Sigma}^{+/-}_{pc}$ and velocities at a half-edge shared by two sub-cells attached to the same point. Labels c and d denote the indices of two neighboring cells. Labels p and r denote the indices of the same point relatively to the local numbering of points in cell c and d. The neighboring sub-cells are denoted by ω_{pc} and ω_{rd}. The half-edge fluxes, $\boldsymbol{\Sigma}^{+/-}_{pc}$, $\boldsymbol{\Sigma}^{+/-}_{rd}$ and velocities, $\boldsymbol{V}^{+/-}_{pc}$, $\boldsymbol{V}^{+/-}_{rd}$ are displayed using blue color.

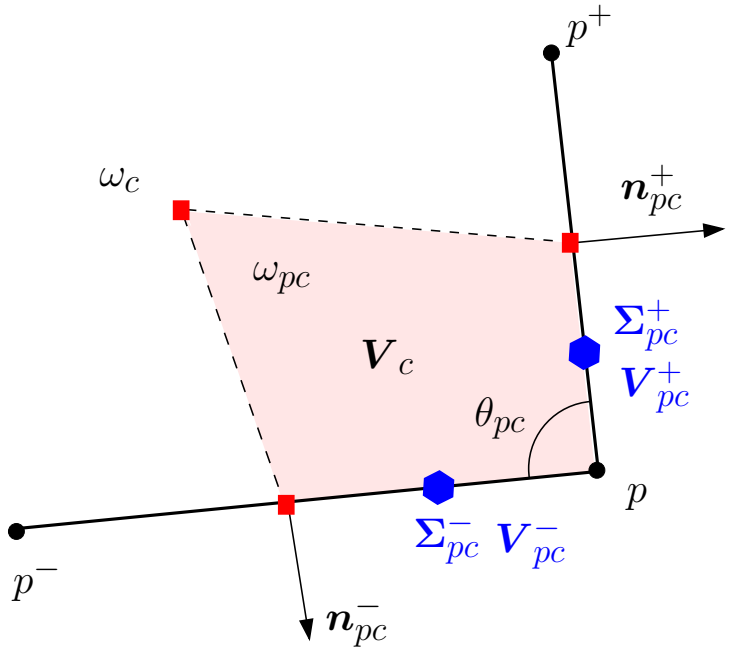


Figure 2.4: Notations for a generic sub-cell ω_{pc}.

2.5.2 Expression of a second-order tensor in terms of its projections on a basis.

Let $\mathbb{T}$ be a second-order tensor and $\{\boldsymbol{n}_1, \boldsymbol{n}_2\}$ a basis of $\mathfrak{R}^2$ where $\boldsymbol{n}_1$ and $\boldsymbol{n}_2$ are unit non-collinear vectors.

$\boldsymbol{T}_i = \mathbb{T}\boldsymbol{n}_i$ for $i = 1, 2$ are the projections of $\mathbb{T}$ on the basis vectors. Since $\{\boldsymbol{n}_1, \boldsymbol{n}_2\}$ is a basis, the knowledge of $\boldsymbol{T}_1$ and $\boldsymbol{T}_2$ is sufficient to fully determine tensor $\mathbb{T}$. The representation of $\mathbb{T}$ by means of its projections is denoted by $[\boldsymbol{T}_1, \boldsymbol{T}_2] = [\mathbb{T}\boldsymbol{n}_1, \mathbb{T}\boldsymbol{n}_2]$.

Introducing the matrix $\mathbb{J}_{12} = [\boldsymbol{n}_1, \boldsymbol{n}_2]$ leads to

$$[\boldsymbol{T}_1, \boldsymbol{T}_2] = \mathbb{T}\mathbb{J}_{12} \Leftrightarrow \mathbb{T} = [\boldsymbol{T}_1, \boldsymbol{T}_2]\mathbb{J}_{12}^{-1}, \tag{2.35}$$

$\mathbb{J}_{12}$ is invertible since $\{\boldsymbol{n}_1, \boldsymbol{n}_2\}$ is a basis.

Let $\mathbb{U}$ be a second-order tensor $\mathbb{U} = [\boldsymbol{U}_1, \boldsymbol{U}_2]\mathbb{J}_{12}^{-1}$. We express the inner product between $\mathbb{U}$ and $\mathbb{T}$ in terms of the projections and the matrix $\mathbb{J}_{12}$.

$$\mathbb{T} : \mathbb{U} = [\boldsymbol{T}_1, \boldsymbol{T}_2](\mathbb{J}_{12}^t\mathbb{J}_{12})^{-1} : [\boldsymbol{U}_1, \boldsymbol{U}_2]. \tag{2.36}$$

Here, $\mathbb{J}_{12}^t\mathbb{J}_{12} = \begin{pmatrix} \boldsymbol{n}_1 \cdot \boldsymbol{n}_1 & \boldsymbol{n}_1 \cdot \boldsymbol{n}_2 \\ \boldsymbol{n}_2 \cdot \boldsymbol{n}_1 & \boldsymbol{n}_2 \cdot \boldsymbol{n}_2 \end{pmatrix}$ is the metric tensor related to the basis $\{\boldsymbol{n}_1, \boldsymbol{n}_2\}$. It is a symmetric positive semi definite matrix.

Indeed, we have $\det(\mathbb{J}_{12}^t\mathbb{J}_{12}) = (\boldsymbol{n}_1 \cdot \boldsymbol{n}_1)(\boldsymbol{n}_2 \cdot \boldsymbol{n}_2) - (\boldsymbol{n}_1 \cdot \boldsymbol{n}_2)^2 \geq 0$ using Cauchy Schwarz. We note that the determinant is equal to zero if and only if $\boldsymbol{n}_1$ and $\boldsymbol{n}_2$ are collinear.

We conclude this paragraph by giving a result which will be useful for the next sections. Let $\boldsymbol{W}_1$ and $\boldsymbol{W}_2$ be two arbitrary vectors, then the matrix product $[\boldsymbol{W}_1, \boldsymbol{W}_2]^t[\boldsymbol{T}_1, \boldsymbol{T}_2]$ writes

$$[\boldsymbol{W}_1, \boldsymbol{W}_2]^t[\boldsymbol{T}_1, \boldsymbol{T}_2] = \begin{pmatrix} \boldsymbol{W}_1 \cdot \boldsymbol{T}_1 & \boldsymbol{W}_1 \cdot \boldsymbol{T}_2 \\ \boldsymbol{W}_2 \cdot \boldsymbol{T}_1 & \boldsymbol{W}_2 \cdot \boldsymbol{T}_2 \end{pmatrix}. \tag{2.37}$$

Recalling that $\mathbb{S} : \mathbb{T} = \mathrm{tr}(\mathbb{S}^t\mathbb{T})$ leads to

$$[\boldsymbol{W}_1, \boldsymbol{W}_2]^t[\boldsymbol{T}_1, \boldsymbol{T}_2] = \boldsymbol{W}_1 \cdot \boldsymbol{T}_1 + \boldsymbol{W}_2 \cdot \boldsymbol{T}_2. \tag{2.38}$$

2.5.3 Sub-cell based variational formulation

In this section we construct an approximation of the half-edges fluxes using a local variational formulation written over the sub-cell ω_{pc}. This variational formulation leads to a local explicit expression of the half-edges fluxes in terms of the half-edges velocities and the mean cell velocity.

The starting point to derive the sub-cell based variational formulation consist in writing the modified constitutive law (2.20) as follow

$$\bar{\bar{\mathbb{R}}}^{-1}\mathbb{\Sigma} - \nabla\boldsymbol{V} = \mathbb{0}. \tag{2.39}$$

Let us take $\mathbb{T} \in \mathcal{T}$ an arbitrary second-order tensor, we apply the inner-product of $\mathbb{T}$ to the previous equation to obtain

$$\bar{\bar{\mathbb{R}}}^{-1}\mathbb{\Sigma} : \mathbb{T} - \mathbb{T} : \nabla\boldsymbol{V} = 0, \, \forall \mathbb{T} \in \mathcal{T}. \tag{2.40}$$

Integrating the above equation over ω_{pc} and using the tensorial identity $\nabla\cdot(\mathbb{T}^t\boldsymbol{V}) = (\nabla\cdot\mathbb{T})\cdot\boldsymbol{V}+\mathbb{T} : \nabla\boldsymbol{V}$ yields

$$\begin{aligned}\int_{\omega_{pc}} \bar{\bar{\mathbb{R}}}^{-1}\mathbb{\Sigma} : \mathbb{T}\,\mathrm{d}v &= \int_{\partial\omega_{pc}} \mathbb{T}^t\boldsymbol{V}\cdot\boldsymbol{n}\,\mathrm{d}s - \int_{\omega_{pc}} (\nabla\cdot\mathbb{T})\cdot\boldsymbol{V}\,\mathrm{d}v,\\ &= \int_{\partial\omega_{pc}} \mathbb{T}\boldsymbol{n}\cdot\boldsymbol{V}\,\mathrm{d}s - \int_{\omega_{pc}} (\nabla\cdot\mathbb{T})\cdot\boldsymbol{V}\,\mathrm{d}v,\end{aligned}$$

Introducing the approximation $\int_{\omega_{pc}}(\nabla\cdot\mathbb{T})\cdot\boldsymbol{V}\,\mathrm{d}v = \boldsymbol{V}_c\cdot\int_{\omega_{pc}}(\nabla\cdot\mathbb{T})\,\mathrm{d}v$ allow us to rewrite the above equation as

$$\int_{\omega_{pc}} \bar{\bar{\mathbb{R}}}^{-1}\mathbb{\Sigma} : \mathbb{T}\,\mathrm{d}v = \int_{\partial\omega_{pc}} \mathbb{T}\boldsymbol{n}\cdot\boldsymbol{V}\,\mathrm{d}s - \boldsymbol{V}_c\cdot\int_{\partial\omega_{pc}} \mathbb{T}\boldsymbol{n}\,\mathrm{d}s. \tag{2.41}$$

The boundary of the sub-cell $\partial\omega_{pc}$ is decomposed into an inner part $\overline{\partial\omega_{pc}}$ and an outer part $\underline{\partial\omega_{pc}}$ as $\partial\omega_{pc} = \overline{\partial\omega_{pc}}\cup\underline{\partial\omega_{pc}}$, where $\underline{\partial\omega_{pc}} = \partial\omega_{pc}^-\cup\partial\omega_{pc}^+$.

We use the following approximation $\int_{\overline{\partial\omega_{pc}}}\mathbb{T}\boldsymbol{n}\cdot\boldsymbol{V}\,\mathrm{d}s = \boldsymbol{V}_c\cdot\int_{\overline{\partial\omega_{pc}}}\mathbb{T}\boldsymbol{n}\,\mathrm{d}s$ and introduce it into (2.41) to obtain

$$\int_{\omega_{pc}} \bar{\bar{\mathbb{R}}}^{-1}\mathbb{\Sigma} : \mathbb{T}\,\mathrm{d}v = \int_{\underline{\partial\omega_{pc}}} (\boldsymbol{V}-\boldsymbol{V}_c)\cdot\mathbb{T}\boldsymbol{n}\,\mathrm{d}s. \tag{2.42}$$

Employing the definitions (2.32) and (2.33) of the half-edge approximations of the tensor $\mathbb{T}$ and the velocity $\boldsymbol{V}$ leads to

$$\int_{\omega_{pc}} \bar{\bar{\mathbb{R}}}^{-1}\mathbb{\Sigma} : \mathbb{T}\,\mathrm{d}v = l_{pc}^-(\boldsymbol{V}_{pc}^- - \boldsymbol{V}_c)\cdot\boldsymbol{T}_{pc}^- + l_{pc}^+(\boldsymbol{V}_{pc}^+ - \boldsymbol{V}_c)\cdot\boldsymbol{T}_{pc}^+. \tag{2.43}$$

Finally, we use (2.38) to rewrite the above equation under the compact form

$$\int_{\omega_{pc}} \bar{\bar{\mathbb{R}}}^{-1}\mathbb{\Sigma} : \mathbb{T}\,\mathrm{d}v = \left[l_{pc}^-(\boldsymbol{V}_{pc}^- - \boldsymbol{V}_c), l_{pc}^+(\boldsymbol{V}_{pc}^+ - \boldsymbol{V}_c)\right] : \left[\boldsymbol{T}_{pc}^-, \boldsymbol{T}_{pc}^+\right]. \tag{2.44}$$

We can now approximate the left-hand side of the above equation by means of the quadrature formula

$$\int_{\omega_{pc}} \bar{\bar{\mathbb{R}}}^{-1}\mathbb{\Sigma} : \mathbb{T}\,\mathrm{d}v = w_{pc}(\bar{\bar{\mathbb{R}}}^{-1}\mathbb{\Sigma})_{pc} : \mathbb{T}_{pc}, \tag{2.45}$$

where $(\bar{\bar{\mathbb{R}}}^{-1}\mathbb{\Sigma})_{pc}$ and $\mathbb{T}_{pc}$ are piecewise constant approximations of $\bar{\bar{\mathbb{R}}}^{-1}\mathbb{\Sigma}$ and $\mathbb{T}$ over the sub-cell ω_{pc}. Note that w_{pc} is a volume weight which satisfies $\sum_{p\in\mathcal{P}(c)} w_{pc} = |\omega_c|$ and $w_{pc} > 0$. Section 2.7.2 will be devoted to the definition of the volume weight for different kinds of cells. As we will see it is a key element to ensure some mandatory properties of the scheme.

Employing the expression (2.36) of the inner product of tensors $(\bar{\bar{\mathbb{R}}}^{-1}\mathbb{\Sigma})_{pc}$ and $\mathbb{T}_{pc}$ in terms of their projections on the basis $\{\boldsymbol{n}_{pc}^-, \boldsymbol{n}_{pc}^+\}$ leads to

$$(\bar{\bar{\mathbb{R}}}^{-1}\mathbb{\Sigma})_{pc} : \mathbb{T}_{pc} = [(\bar{\bar{\mathbb{R}}}^{\mathbf{-1}}\mathbb{\Sigma})_{pc}^-, (\bar{\bar{\mathbb{R}}}^{\mathbf{-1}}\mathbb{\Sigma})_{pc}^+](\mathbb{J}_{pc}^t\mathbb{J}_{pc})^{-1} : [\boldsymbol{T}_{pc}^-, \boldsymbol{T}_{pc}^+]. \tag{2.46}$$

Gathering the left-hand side of (2.44) and the right-hand side of (2.46) of the variational formulation yields

$$w_{pc}[(\bar{\bar{\mathbb{R}}}^{\mathbf{-1}}\mathbb{\Sigma})_{pc}^-, (\bar{\bar{\mathbb{R}}}^{\mathbf{-1}}\mathbb{\Sigma})_{pc}^+](\mathbb{J}_{pc}^t\mathbb{J}_{pc})^{-1} : [\boldsymbol{T}_{pc}^-, \boldsymbol{T}_{pc}^+] = \left[l_{pc}^-(\boldsymbol{V}_{pc}^- - \boldsymbol{V}_c), l_{pc}^+(\boldsymbol{V}_{pc}^+ - \boldsymbol{V}_c)\right] : \left[\mathbb{T}_{pc}^-, \mathbb{T}_{pc}^+\right]. \tag{2.47}$$

Knowing that it must hold for all $\mathbb{T} \in \mathcal{T}$, this implies

$$w_{pc}\left[(\bar{\bar{\mathbb{R}}}^{-\mathbf{1}}\mathbb{\Sigma})^-_{pc}, (\bar{\bar{\mathbb{R}}}^{-\mathbf{1}}\mathbb{\Sigma})^+_{pc}\right](\mathbb{J}^t_{pc}\mathbb{J}_{pc})^{-1} = \left[l^-_{pc}(\boldsymbol{V}^-_{pc} - \boldsymbol{V}_c), l^+_{pc}(\boldsymbol{V}^+_{pc} - \boldsymbol{V}_c)\right], \tag{2.48}$$

then

$$\left[(\bar{\bar{\mathbb{R}}}^{-\mathbf{1}}\mathbb{\Sigma})^-_{pc}, (\bar{\bar{\mathbb{R}}}^{-\mathbf{1}}\mathbb{\Sigma})^+_{pc}\right] = \frac{1}{w_{pc}}\left[l^-_{pc}(\boldsymbol{V}^-_{pc} - \boldsymbol{V}_c), l^+_{pc}(\boldsymbol{V}^+_{pc} - \boldsymbol{V}_c)\right]\mathbb{J}^t_{pc}\mathbb{J}_{pc}. \tag{2.49}$$

Recalling that $(\bar{\bar{\mathbb{R}}}^{-1}\mathbb{\Sigma})_{pc} = \left[(\bar{\bar{\mathbb{R}}}^{-\mathbf{1}}\mathbb{\Sigma})^-_{pc}, (\bar{\bar{\mathbb{R}}}^{-\mathbf{1}}\mathbb{\Sigma})^+_{pc}\right]\mathbb{J}^{-1}_{pc}$ leads to

$$(\bar{\bar{\mathbb{R}}}^{-1}\mathbb{\Sigma})_{pc} = \frac{1}{w_{pc}}\left[l^-_{pc}(\boldsymbol{V}^-_{pc} - \boldsymbol{V}_c), l^+_{pc}(\boldsymbol{V}^+_{pc} - \boldsymbol{V}_c)\right]\mathbb{J}^t_{pc}. \tag{2.50}$$

Observing that $(\bar{\bar{\mathbb{R}}}^{-1}\mathbb{\Sigma})_{pc} = \bar{\bar{\mathbb{R}}}^{-1}_c\mathbb{\Sigma}_{pc}$ where $\bar{\bar{\mathbb{R}}}^{-1}_c$ is the piecewise constant approximation of $\bar{\bar{\mathbb{R}}}^{-1}$ over the cell ω_c we obtain

$$\mathbb{\Sigma}_{pc} = \frac{1}{w_{pc}}\bar{\bar{\mathbb{R}}}_c\left\{\left[l^-_{pc}(\boldsymbol{V}^-_{pc} - \boldsymbol{V}_c), l^+_{pc}(\boldsymbol{V}^+_{pc} - \boldsymbol{V}_c)\right]\mathbb{J}^t_{pc}\right\}. \tag{2.51}$$

Finally, $\left[\boldsymbol{\Sigma}^-_{pc}, \boldsymbol{\Sigma}^+_{pc}\right] = \mathbb{\Sigma}_{pc}\mathbb{J}_{pc}$ and

$$\left[\boldsymbol{\Sigma}^-_{pc}, \boldsymbol{\Sigma}^+_{pc}\right] = \frac{1}{w_{pc}}\bar{\bar{\mathbb{R}}}_c\left\{\left[l^-_{pc}(\boldsymbol{V}^-_{pc} - \boldsymbol{V}_c), l^+_{pc}(\boldsymbol{V}^+_{pc} - \boldsymbol{V}_c)\right]\mathbb{J}^t_{pc}\right\}\mathbb{J}_{pc}. \tag{2.52}$$

We conclude this section by recalling that $\bar{\bar{\mathbb{R}}}$ is the fourth-order tensor defined by

$$\bar{\bar{\mathbb{R}}}\mathbb{T} = \mu\mathbb{T} + (\mu - \beta)\mathbb{T}^t + \frac{1}{3}(3\beta - 2\mu)\,\mathrm{tr}(\mathbb{T})\mathbb{I}, \tag{2.53}$$

μ has a piecewise constant approximation and β is a constant parameter over $\mathcal{D}$, since $\beta = \min_{\mathfrak{R}}\mu$.

2.5.4 Expression of the velocity gradient tensor

Let $\mathbb{G} = \nabla\boldsymbol{V}$ be the velocity gradient tensor. Using the above variational formulation with $\bar{\bar{\mathbb{R}}} = \bar{\bar{\mathbb{I}}}$ we get

$$\left[\mathbb{G}^-_{pc}, \mathbb{G}^+_{pc}\right] = \frac{1}{w_{pc}}\left[l^-_{pc}(\boldsymbol{V}^-_{pc} - \boldsymbol{V}_c), l^+_{pc}(\boldsymbol{V}^+_{pc} - \boldsymbol{V}_c)\right]\mathbb{J}^t_{pc}\mathbb{J}_{pc}. \tag{2.54}$$

We recall that $\mathbb{J}^t_{pc}\mathbb{J}_{pc} = \begin{pmatrix}\boldsymbol{n}^-_{pc}\cdot\boldsymbol{n}^-_{pc} & \boldsymbol{n}^-_{pc}\cdot\boldsymbol{n}^+_{pc}\\ \boldsymbol{n}^+_{pc}\cdot\boldsymbol{n}^-_{pc} & \boldsymbol{n}^+_{pc}\cdot\boldsymbol{n}^+_{pc}\end{pmatrix}$ and we use the following identities $(\boldsymbol{a}\otimes\boldsymbol{b})\boldsymbol{c} = \boldsymbol{a}(\boldsymbol{b}\cdot\boldsymbol{c})$ and $(\boldsymbol{a}\otimes\boldsymbol{c})\boldsymbol{b} = \boldsymbol{a}(\boldsymbol{c}\cdot\boldsymbol{b}) = (\boldsymbol{a}\otimes\boldsymbol{b})\boldsymbol{c}$ in order to express $\mathbb{G}^-_{pc}$ as

$$\begin{aligned}\mathbb{G}^-_{pc} &= \frac{1}{w_{pc}}\left[l^-_{pc}(\boldsymbol{V}^-_{pc} - \boldsymbol{V}_c)(\boldsymbol{n}^-_{pc}\cdot\boldsymbol{n}^-_{pc}) + l^+_{pc}(\boldsymbol{V}^+_{pc} - \boldsymbol{V}_c)(\boldsymbol{n}^+_{pc}\cdot\boldsymbol{n}^-_{pc})\right],\\ &= \frac{1}{w_{pc}}\left\{l^-_{pc}\left[(\boldsymbol{V}^-_{pc} - \boldsymbol{V}_c)\otimes\boldsymbol{n}^-_{pc}\right]\boldsymbol{n}^-_{pc} + l^+_{pc}\left[(\boldsymbol{V}^+_{pc} - \boldsymbol{V}_c)\otimes\boldsymbol{n}^+_{pc}\right]\boldsymbol{n}^-_{pc}\right\},\\ &= \frac{1}{w_{pc}}\left[l^-_{pc}(\boldsymbol{V}^-_{pc} - \boldsymbol{V}_c)\otimes\boldsymbol{n}^-_{pc} + l^+_{pc}(\boldsymbol{V}^+_{pc} - \boldsymbol{V}_c)\otimes\boldsymbol{n}^+_{pc}\right]\boldsymbol{n}^-_{pc}.\end{aligned}$$

This implies that the piecewise approximation of the velocity gradient tensor over the sub-cell ω_{pc} is given by

$$\mathbb{G}_{pc} = \frac{1}{w_{pc}}\left[l^-_{pc}(\boldsymbol{V}^-_{pc} - \boldsymbol{V}_c)\otimes\boldsymbol{n}^-_{pc} + l^+_{pc}(\boldsymbol{V}^+_{pc} - \boldsymbol{V}_c)\otimes\boldsymbol{n}^+_{pc}\right]. \tag{2.55}$$

It is trivial to show that $\mathbb{G}_{pc}^{+} = \mathbb{G}_{pc}\boldsymbol{n}_{pc}^{+}$.

Substituting $\mathbb{G}_{pc} = \left[\mathbb{G}_{pc}^{-}, \mathbb{G}_{pc}^{+}\right] \mathbb{J}_{pc}^{-1}$ into (2.54) we obtain

$$\mathbb{G}_{pc} = \frac{1}{w_{pc}} \left[l_{pc}^{-}(\boldsymbol{V}_{pc}^{-} - \boldsymbol{V}_c), l_{pc}^{+}(\boldsymbol{V}_{pc}^{+} - \boldsymbol{V}_c)\right] \mathbb{J}_{pc}^{t}. \tag{2.56}$$

The comparison to the expression of $\mathbb{\Sigma}_{pc}$ obtained in (2.51) leads to $\mathbb{\Sigma}_{pc} = \bar{\bar{\mathbb{R}}}_c \mathbb{G}_{pc}$ where $\mathbb{G}_{pc}$ is defined by (2.55).

Comment 18: *Let us point out that the expression of the velocity gradient tensor given by* (2.55) *could have been obtained by simply applying the Green-Gauss formula as follows*

$$\int_{\omega_{pc}} \nabla \boldsymbol{V} \, \mathrm{d}v = \int_{\partial\omega_{pc}} \boldsymbol{V} \otimes \boldsymbol{n} \, \mathrm{d}s.$$

2.5.5 Expression of $\boldsymbol{\Sigma}_{pc}^{+/-}$ in terms of $\boldsymbol{V}_{pc}^{+/-}$ and $\boldsymbol{V}_c$

In this section we seek an expression of the half-edges viscous fluxes $\boldsymbol{\Sigma}_{pc}^{+/-}$ in terms of the auxiliary unknowns $\boldsymbol{V}_{pc}^{+/-}$ and the mean cell velocities $\boldsymbol{V}_c$. Let us start by recalling that the approximation of the half-edge viscous fluxes is given by $\boldsymbol{\Sigma}_{pc}^{+/-} = \mathbb{\Sigma}_{pc}\boldsymbol{n}_{pc}^{+/-}$. The tensor $\mathbb{\Sigma}_{pc}$ is determined by $\mathbb{\Sigma}_{pc} = \bar{\bar{\mathbb{R}}}_c \mathbb{G}_{pc}$, where $\bar{\bar{\mathbb{R}}}_c$ is the local approximation of the fourth-order tensor $\bar{\bar{\mathbb{R}}}$ given by

$$\bar{\bar{\mathbb{R}}}_c \mathbb{T} = \mu_c \mathbb{T} + (\mu_c - \beta)\mathbb{T}^t + \frac{1}{3}(3\beta - 2\mu_c)\,\mathrm{tr}(\mathbb{T})\mathbb{I}, \; \forall \mathbb{T} \in \mathcal{T}. \tag{2.57}$$

Here, μ_c is the piecewise constant approximation of μ over the cell c. These definitions lead us to write $\mathbb{\Sigma}_{pc}$ as

$$\mathbb{\Sigma}_{pc} = \mu_c \mathbb{G}_{pc} + (\mu_c - \beta)\mathbb{G}_{pc}^{t} + \frac{1}{3}(3\beta - 2\mu_c)\,\mathrm{tr}(\mathbb{G}_{pc})\mathbb{I}.$$

Let us compute the different terms of the right hand-side of this equation. First $\mathbb{G}_{pc}$ is defined by

$$\mathbb{G}_{pc} = \frac{1}{w_{pc}} \left[l_{pc}^{-}(\boldsymbol{V}_{pc}^{-} - \boldsymbol{V}_c) \otimes \boldsymbol{n}_{pc}^{-} + l_{pc}^{+}(\boldsymbol{V}_{pc}^{+} - \boldsymbol{V}_c) \otimes \boldsymbol{n}_{pc}^{+}\right],$$

we also need its transpose matrix, which writes

$$\mathbb{G}_{pc}^{t} = \frac{1}{w_{pc}} \left[l_{pc}^{-}\boldsymbol{n}_{pc}^{-} \otimes (\boldsymbol{V}_{pc}^{-} - \boldsymbol{V}_c) + l_{pc}^{+}\boldsymbol{n}_{pc}^{+} \otimes (\boldsymbol{V}_{pc}^{+} - \boldsymbol{V}_c)\right],$$

and the trace operator applied to $\mathbb{G}_{pc}$ yields

$$\mathrm{tr}(\mathbb{G}_{pc}) = \frac{1}{w_{pc}} \left[l_{pc}^{-}(\boldsymbol{V}_{pc}^{-} - \boldsymbol{V}_c) \cdot \boldsymbol{n}_{pc}^{-} + l_{pc}^{+}(\boldsymbol{V}_{pc}^{+} - \boldsymbol{V}_c) \cdot \boldsymbol{n}_{pc}^{+}\right].$$

We recall that $\boldsymbol{\Sigma}_{pc}^{+/-}$ are defined as $\boldsymbol{\Sigma}_{pc}^{+/-} = \mathbb{\Sigma}_{pc}\boldsymbol{n}_{pc}^{+/-}$. Let us start by computing $\boldsymbol{\Sigma}_{pc}^{-}$. The first term $\mathbb{G}_{pc}^{-}$ writes

$$\begin{aligned}
\mathbb{G}_{pc}^{-} &= \mathbb{G}_{pc}\boldsymbol{n}_{pc}^{-}, \\
&= \frac{1}{w_{pc}} \left\{ \left[l_{pc}^{-}(\boldsymbol{V}_{pc}^{-} - \boldsymbol{V}_c) \otimes \boldsymbol{n}_{pc}^{-}\right] \boldsymbol{n}_{pc}^{-} + \left[l_{pc}^{+}(\boldsymbol{V}_{pc}^{+} - \boldsymbol{V}_c) \otimes \boldsymbol{n}_{pc}^{+}\right] \boldsymbol{n}_{pc}^{-} \right\}, \\
&= \frac{1}{w_{pc}} \left[l_{pc}^{-}\mathbb{I}(\boldsymbol{V}_{pc}^{-} - \boldsymbol{V}_c) - l_{pc}^{+}\cos\theta_{pc}\mathbb{I}(\boldsymbol{V}_{pc}^{+} - \boldsymbol{V}_c)\right].
\end{aligned}$$

The second term $(\mathbb{G}_{pc}^{-})^t$ writes

$$\begin{aligned}(\mathbb{G}_{pc}^{-})^t &= \mathbb{G}_{pc}^t \boldsymbol{n}_{pc}^{-},\\ &= \frac{1}{w_{pc}}\left[l_{pc}^{-}\boldsymbol{n}_{pc}^{-}\otimes(\boldsymbol{V}_{pc}^{-}-\boldsymbol{V}_c)+l_{pc}^{+}\boldsymbol{n}_{pc}^{+}\otimes(\boldsymbol{V}_{pc}^{+}-\boldsymbol{V}_c)\right]\boldsymbol{n}_{pc}^{-},\\ &= \frac{1}{w_{pc}}\left[l_{pc}^{-}(\boldsymbol{n}_{pc}^{-}\otimes\boldsymbol{n}_{pc}^{-})(\boldsymbol{V}_{pc}^{-}-\boldsymbol{V}_c)+l_{pc}^{+}(\boldsymbol{n}_{pc}^{+}\otimes\boldsymbol{n}_{pc}^{-})(\boldsymbol{V}_{pc}^{+}-\boldsymbol{V}_c)\right].\end{aligned}$$

Finally, taking the trace of the trace $\mathbb{G}_{pc}$ we get

$$\begin{aligned}(\mathrm{tr}(\mathbb{G}_{pc})\mathbb{I})\boldsymbol{n}_{pc}^{-} &= \frac{1}{w_{pc}}\left[l_{pc}^{-}(\boldsymbol{V}_{pc}^{-}-\boldsymbol{V}_c)\cdot\boldsymbol{n}_{pc}^{-}\cdot\boldsymbol{n}_{pc}^{-}+l_{pc}^{+}(\boldsymbol{V}_{pc}^{+}-\boldsymbol{V}_c)\cdot\boldsymbol{n}_{pc}^{+}\cdot\boldsymbol{n}_{pc}^{-}\right],\\ &= \frac{1}{w_{pc}}\left[l_{pc}^{-}(\boldsymbol{n}_{pc}^{-}\otimes\boldsymbol{n}_{pc}^{-})(\boldsymbol{V}_{pc}^{-}-\boldsymbol{V}_c)+l_{pc}^{+}(\boldsymbol{n}_{pc}^{-}\otimes\boldsymbol{n}_{pc}^{+})(\boldsymbol{V}_{pc}^{+}-\boldsymbol{V}_c)\right],\end{aligned}$$

Summing all these terms together allow us to write

$$\begin{aligned}\boldsymbol{\Sigma}_{pc}^{-} = \frac{1}{w_{pc}}\Bigg\{&l_{pc}^{-}\mu_c\left[\mathbb{I}+\frac{1}{3}(\boldsymbol{n}_{pc}^{-}\otimes\boldsymbol{n}_{pc}^{-})\right](\boldsymbol{V}_{pc}^{-}-\boldsymbol{V}_c)\\ &+l_{pc}^{+}\left[-\mu_c\cos\theta_{pc}\mathbb{I}+(\mu_c-\beta)(\boldsymbol{n}_{pc}^{+}\otimes\boldsymbol{n}_{pc}^{-})+\frac{1}{3}(3\beta-2\mu_c)(\boldsymbol{n}_{pc}^{-}\otimes\boldsymbol{n}_{pc}^{+})\right](\boldsymbol{V}_{pc}^{+}-\boldsymbol{V}_c)\Bigg\}.\end{aligned}\tag{2.58}$$

Proceeding similarly, the term $\boldsymbol{\Sigma}_{pc}^{+}$ writes

$$\begin{aligned}\boldsymbol{\Sigma}_{pc}^{+} = \frac{1}{w_{pc}}\Bigg\{&l_{pc}^{-}\left[-\mu_c\cos\theta_{pc}\mathbb{I}+(\mu_c-\beta)(\boldsymbol{n}_{pc}^{-}\otimes\boldsymbol{n}_{pc}^{+})+\frac{1}{3}(3\beta-2\mu_c)(\boldsymbol{n}_{pc}^{+}\otimes\boldsymbol{n}_{pc}^{-})\right](\boldsymbol{V}_{pc}^{-}-\boldsymbol{V}_c)\\ &+l_{pc}^{+}\mu_c\left[\mathbb{I}+\frac{1}{3}(\boldsymbol{n}_{pc}^{+}\otimes\boldsymbol{n}_{pc}^{+})\right](\boldsymbol{V}_{pc}^{+}-\boldsymbol{V}_c)\Bigg\}.\end{aligned}\tag{2.59}$$

Let us introduce the three following 2×2 matrices

$$\mathbb{M}_{pc}^{-} = \mu_c\left[\mathbb{I}+\frac{1}{3}(\boldsymbol{n}_{pc}^{-}\otimes\boldsymbol{n}_{pc}^{-})\right],\tag{2.60}$$

$$\mathbb{M}_{pc}^{+} = \mu_c\left[\mathbb{I}+\frac{1}{3}(\boldsymbol{n}_{pc}^{+}\otimes\boldsymbol{n}_{pc}^{+})\right],\tag{2.61}$$

$$\mathbb{E}_{pc} = -\mu_c\cos\theta_{pc}\mathbb{I}+(\mu_c-\beta)(\boldsymbol{n}_{pc}^{+}\otimes\boldsymbol{n}_{pc}^{-})+\frac{1}{3}(3\beta-2\mu_c)(\boldsymbol{n}_{pc}^{-}\otimes\boldsymbol{n}_{pc}^{+}),\tag{2.62}$$

where θ_{pc} is the corner angle of the sub-cell ω_{pc}. Using the above 2×2 matrices, formulas (2.58) and (2.59) are expressed in the much more compact form

$$\boldsymbol{\Sigma}_{pc}^{-} = \frac{1}{w_{pc}}\left[\mathbb{M}_{pc}^{-}l_{pc}^{-}(\boldsymbol{V}_{pc}^{-}-\boldsymbol{V}_c)+\mathbb{E}_{pc}l_{pc}^{+}(\boldsymbol{V}_{pc}^{+}-\boldsymbol{V}_c)\right],\tag{2.63}$$

$$\boldsymbol{\Sigma}_{pc}^{+} = \frac{1}{w_{pc}}\left[\mathbb{E}_{pc}^{t}l_{pc}^{-}(\boldsymbol{V}_{pc}^{-}-\boldsymbol{V}_c)+\mathbb{M}_{pc}^{+}l_{pc}^{+}(\boldsymbol{V}_{pc}^{+}-\boldsymbol{V}_c)\right].\tag{2.64}$$

Thus, the half-edge pseudo-stresses are expressed in terms of the half-edge velocities and the cell-centered velocities as follows

$$\begin{pmatrix}\boldsymbol{\Sigma}_{pc}^{-}\\ \boldsymbol{\Sigma}_{pc}^{+}\end{pmatrix} = \frac{1}{w_{pc}}\begin{pmatrix}\mathbb{M}_{pc}^{-} & \mathbb{E}_{pc}\\ \mathbb{E}_{pc}^{t} & \mathbb{M}_{pc}^{+}\end{pmatrix}\begin{pmatrix}l_{pc}^{-}(\boldsymbol{V}_{pc}^{-}-\boldsymbol{V}_c)\\ l_{pc}^{+}(\boldsymbol{V}_{pc}^{+}-\boldsymbol{V}_c)\end{pmatrix}.\tag{2.65}$$

Let us prove that the block matrix $\begin{pmatrix} \mathbb{M}_{pc}^- & \mathbb{E}_{pc} \\ \mathbb{E}_{pc}^t & \mathbb{M}_{pc}^+ \end{pmatrix}$ is invertible. This will be a useful property in order to build the global system associated with our numerical scheme. We start by introducing the scalar K_{pc} defined by $K_{pc} = \boldsymbol{\Sigma}_{pc}^- \cdot l_{pc}^-(\boldsymbol{V}_{pc}^- - \boldsymbol{V}_c) + \boldsymbol{\Sigma}_{pc}^+ \cdot l_{pc}^+(\boldsymbol{V}_{pc}^+ - \boldsymbol{V}_c)$. Recalling that $\boldsymbol{\Sigma}_{pc}^{+/-} = \mathbb{\Sigma}_{pc}\boldsymbol{n}_{pc}^{+/-}$ leads to

$$\begin{aligned} K_{pc} &= \mathbb{\Sigma}_{pc} l_{pc}^- \boldsymbol{n}_{pc}^- \cdot (\boldsymbol{V}_{pc}^- - \boldsymbol{V}_c) + \mathbb{\Sigma}_{pc} l_{pc}^+ \boldsymbol{n}_{pc}^+ \cdot (\boldsymbol{V}_{pc}^+ - \boldsymbol{V}_c) \\ &= l_{pc}^- \boldsymbol{n}_{pc}^- \cdot \mathbb{\Sigma}_{pc}^t (\boldsymbol{V}_{pc}^- - \boldsymbol{V}_c) + l_{pc}^+ \boldsymbol{n}_{pc}^+ \cdot \mathbb{\Sigma}_{pc}^t (\boldsymbol{V}_{pc}^+ - \boldsymbol{V}_c). \end{aligned}$$

Employing the identity $\boldsymbol{a} \cdot \boldsymbol{b} = \mathrm{tr}\,(\boldsymbol{a} \otimes \boldsymbol{b})$ yields

$$\begin{aligned} K_{pc} &= \mathrm{tr}\left\{ l_{pc}^- \boldsymbol{n}_{pc}^- \otimes \mathbb{\Sigma}_{pc}^t (\boldsymbol{V}_{pc}^- - \boldsymbol{V}_c) + l_{pc}^+ \boldsymbol{n}_{pc}^+ \otimes \mathbb{\Sigma}_{pc}^t (\boldsymbol{V}_{pc}^+ - \boldsymbol{V}_c) \right\} \\ &= \mathrm{tr}\left\{ \left[l_{pc}^- \boldsymbol{n}_{pc}^- \otimes (\boldsymbol{V}_{pc}^- - \boldsymbol{V}_c) \right] \mathbb{\Sigma}_{pc} + \left[l_{pc}^+ \boldsymbol{n}_{pc}^+ \otimes (\boldsymbol{V}_{pc}^+ - \boldsymbol{V}_c) \right] \mathbb{\Sigma}_{pc} \right\}. \end{aligned}$$

Recalling the definition of the inner product $\mathbb{T} : \mathbb{Q} = \mathrm{tr}\left(\mathbb{Q}^t \mathbb{T}\right)$ yields

$$\begin{aligned} K_{pc} &= \left[l_{pc}^-(\boldsymbol{V}_{pc}^- - \boldsymbol{V}_c) \otimes \boldsymbol{n}_{pc}^- + l_{pc}^+(\boldsymbol{V}_{pc}^+ - \boldsymbol{V}_c) \otimes \boldsymbol{n}_{pc}^+ \right] : \mathbb{\Sigma}_{pc} \\ &= w_{pc} \mathbb{G}_{pc} : \mathbb{\Sigma}_{pc} \text{ thanks to (2.55)} . \end{aligned}$$

Now, using $\mathbb{\Sigma}_{pc} = \bar{\bar{\mathbb{R}}}_c \mathbb{G}_{pc}$ yields $K_{pc} = w_{pc}(\bar{\bar{\mathbb{R}}}_c \mathbb{G}_{pc}) : \mathbb{G}_{pc}$. Since $w_{pc} > 0$ and $\bar{\bar{\mathbb{R}}}_c$ is positive definite, we get $K_{pc} \geq 0$.

It is trivial to see that K_{pc} can also be written as

$$K_{pc} = \frac{1}{w_{pc}} \begin{pmatrix} \mathbb{M}_{pc}^- & \mathbb{E}_{pc} \\ \mathbb{E}_{pc}^t & \mathbb{M}_{pc}^+ \end{pmatrix} \begin{pmatrix} l_{pc}^-(\boldsymbol{V}_{pc}^- - \boldsymbol{V}_c) \\ l_{pc}^+(\boldsymbol{V}_{pc}^+ - \boldsymbol{V}_c) \end{pmatrix} \cdot \begin{pmatrix} l_{pc}^-(\boldsymbol{V}_{pc}^- - \boldsymbol{V}_c) \\ l_{pc}^+(\boldsymbol{V}_{pc}^+ - \boldsymbol{V}_c) \end{pmatrix}.$$

With this notation, we observe that K_{pc} is a quadratic form with respect to $l_{pc}^-(\boldsymbol{V}_{pc}^- - \boldsymbol{V}_c)$ and $l_{pc}^+(\boldsymbol{V}_{pc}^+ - \boldsymbol{V}_c)$, we just showed K_{pc} is non-negative, so we can conclude that the 2×2 block matrix $\begin{pmatrix} \mathbb{M}_{pc}^- & \mathbb{E}_{pc} \\ \mathbb{E}_{pc}^t & \mathbb{M}_{pc}^+ \end{pmatrix}$ is positive definite and thus invertible.

2.5.6 Elimination of the auxiliary unknowns

From (2.65), it appears that the numerical approximation of the half-edge fluxes at a corner depends on the difference between the cell-centered velocity and the half-edges velocities. The cell-centered velocities are the primary unknown whereas the half-edge velocities are auxiliary unknowns, which can be eliminated by means of continuity argument (2.34a). Namely, we use the fact that the half-edge normal fluxes are continuous across each half-edges impinging at a given point. This local elimination procedure, which will be described below, yields a linear system satisfied by the half-edge velocities. We will show that this system admits always a unique solution which allows to express the half-edge velocities in terms of the cell-centered velocities of the cells surrounding the point under consideration. Therefore, this local elimination procedure results in a Finite Volume discrete scheme with one unknown per cell.

Local notation around a point

To derive the local elimination procedure, we shall introduce some convenient notation. Let p denotes a generic point which is not located on the boundary $\partial\mathcal{D}$. The treatment of boundary points is postponed to Section 2.5.8. Let $\mathcal{C}(p)$ be the set of cells that surround point p. The edges impinging at point p are labelled using the subscript c ranging from 1 to $\mathfrak{C}_p$, where $\mathfrak{C}_p$

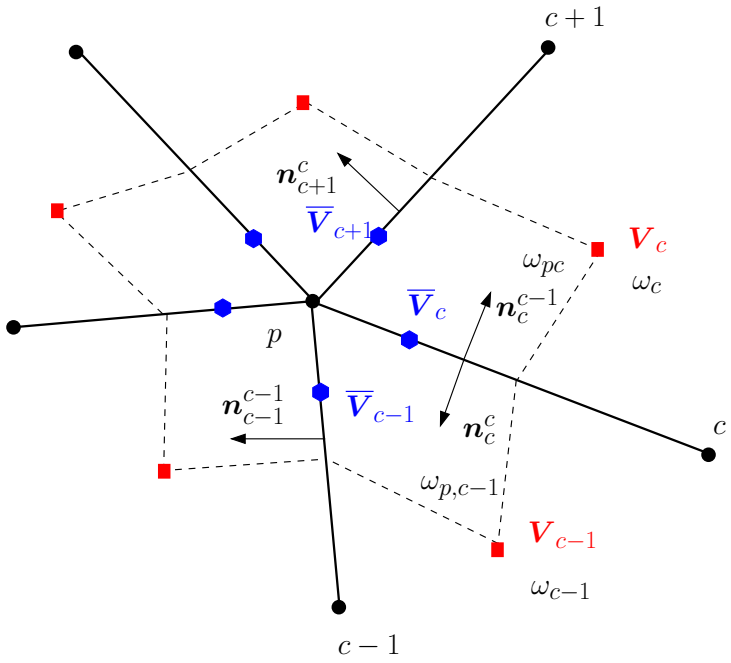


Figure 2.5: Notations employed around a point.

denotes the total number of cells surrounding point p. The cell (sub-cell) numbering follows the edge numbering, that is, cell ω_c (sub-cell ω_{pc}) is located between edges c and $c+1$, refer to Figure 2.5. The unit outward normal to cell ω_c at edge c is denoted by $\boldsymbol{n}^c_c$ whereas the unit outward normal to cell ω_c at edge $c+1$ is denoted by $\boldsymbol{n}^{c+1}_c$. Assuming the continuity of the half-edge velocities leads to denote by $\overline{\boldsymbol{V}}_c$ the unique half-edge velocity of the half-edge c impinging at point p. Finally we note $\boldsymbol{\Sigma}^c_c$ the half-edge flux defined on cell c for the half-edge c and $\boldsymbol{\Sigma}^c_{c+1}$ the half-edge flux defined on cell c for the half-edge $c+1$. Note that we have omitted the dependency on point p in the indexing each time this is possible to avoid too heavy notation.

Let us rewrite with this new notation the expression of the half-edge fluxes (2.63) and (2.64).

$$\boldsymbol{\Sigma}^c_{c+1} = \alpha_c \left[l_{c+1} \mathbb{M}^c_{c+1} (\overline{\boldsymbol{V}}_{c+1} - \boldsymbol{V}_c) + l_c \mathbb{E}_c (\overline{\boldsymbol{V}}_c - \boldsymbol{V}_c) \right], \tag{2.66}$$

$$\boldsymbol{\Sigma}^c_c = \alpha_c \left[l_{c+1} \mathbb{E}^t_c (\overline{\boldsymbol{V}}_{c+1} - \boldsymbol{V}_c) + l_c \mathbb{M}^c_c (\overline{\boldsymbol{V}}_c - \boldsymbol{V}_c) \right], \tag{2.67}$$

where we have introduced α_c defined by $\alpha_c = \frac{1}{w_c} > 0$. Here we recall the definition of the different matrices used in these expressions

$$\mathbb{M}^c_{c+1} = \mu_c \left[\mathbb{I} + \frac{1}{3} (\boldsymbol{n}^c_{c+1} \otimes \boldsymbol{n}^c_{c+1}) \right], \tag{2.68}$$

$$\mathbb{M}^c_c = \mu_c \left[\mathbb{I} + \frac{1}{3} (\boldsymbol{n}^c_c \otimes \boldsymbol{n}^c_c) \right], \tag{2.69}$$

$$\mathbb{E}_c = -\mu_c \cos\theta_c \mathbb{I} + (\mu_c - \beta)(\boldsymbol{n}^c_c \otimes \boldsymbol{n}^c_{c+1}) + \frac{1}{3}(3\beta - 2\mu_c)(\boldsymbol{n}^c_{c+1} \otimes \boldsymbol{n}^c_c). \tag{2.70}$$

Shifting the index c to $c-1$ in (2.66) allows us to obtain the definition of the half-edge flux defined on cell $c-1$ for the half-edge c. It yields

$$\boldsymbol{\Sigma}^{c-1}_c = \alpha_{c-1} \left[l_c \mathbb{M}^{c-1}_c (\overline{\boldsymbol{V}}_c - \boldsymbol{V}_{c-1}) + l_{c-1} \mathbb{E}_{c-1} (\overline{\boldsymbol{V}}_{c-1} - \boldsymbol{V}_{c-1}) \right]. \tag{2.71}$$

Linear system satisfied by the half-edge velocities

Using the previous definitions we are now in position to express the half-edge velocities in terms of the cell centered velocities. To do so we need to write the continuity conditions at the half-edges. The continuity condition of the half-edge flux across the half-edge c writes

$$l_c \boldsymbol{\Sigma}_c^{c-1} + l_c \boldsymbol{\Sigma}_c^{c} = \mathbf{0} \tag{2.72}$$

Substituting (2.71) and (2.67) into (2.72) leads to

$$\begin{aligned}\alpha_{c-1} l_{c-1} l_c \mathbb{E}_{c-1}(\overline{\boldsymbol{V}}_{c-1} - \boldsymbol{V}_{c-1}) + \alpha_{c-1} l_c^2 \mathbb{M}_c^{c-1}(\overline{\boldsymbol{V}}_c - \boldsymbol{V}_{c-1}) \\ + \alpha_c l_c^2 \mathbb{M}_c^{c}(\overline{\boldsymbol{V}}_c - \boldsymbol{V}_{c}) + \alpha_c l_c l_{c+1} \mathbb{E}_c^{t}(\overline{\boldsymbol{V}}_{c+1} - \boldsymbol{V}_{c}) = \mathbf{0},\end{aligned}$$

then rearranging the terms yields

$$\begin{aligned}&\alpha_{c-1} l_{c-1} l_c \mathbb{E}_{c-1} \overline{\boldsymbol{V}}_{c-1} + l_c^2(\alpha_{c-1}\mathbb{M}_c^{c-1} + \alpha_c \mathbb{M}_c^{c})\overline{\boldsymbol{V}}_c + \alpha_c l_c l_{c+1}\mathbb{E}_c^t \overline{\boldsymbol{V}}_{c+1} = \\ &\alpha_{c-1} l_c (l_{c-1}\mathbb{E}_{c-1} + l_c \mathbb{M}_c^{c-1})\boldsymbol{V}_{c-1} + \alpha_c l_c (l_c \mathbb{M}_c^{c} + l_{c+1}\mathbb{E}_c^t)\boldsymbol{V}_c.\end{aligned}$$

We remark that this continuity conditions provides $\mathfrak{C}_p$ vectorial equations. Here $\mathfrak{C}_p$ denotes the number of edges surrounding a node, which, in two dimensions, is equal to the number of cells surrounding a node (if not on a boundary). We also remark that we have $\mathfrak{C}_p$ auxiliary unknowns $\overline{\boldsymbol{V}}_c$.
We introduce the following block vector $\boldsymbol{V} = (\boldsymbol{V}_1, \dots, \boldsymbol{V}_{\mathfrak{C}_p})^t$ as the block vector of the cell-centered velocities around point p and $\overline{\boldsymbol{V}} = (\overline{\boldsymbol{V}}_1, \dots, \overline{\boldsymbol{V}}_{\mathfrak{C}_p})^t$ as the block vector of the half-edge velocities around point p. Using these block vectors the continuity conditions around point p writes

$$\mathbb{N}\overline{\boldsymbol{V}} = \mathbb{S}\boldsymbol{V}. \tag{2.73}$$

Let us specify the expression of the block matrices $\mathbb{N}$ and $\mathbb{S}$ we just introduced. The non-zero terms corresponding to the cth row of block matrix $\mathbb{N}$ write as

$$\begin{cases} \mathbb{N}_{c,c-1} = \alpha_{c-1} l_{c-1} l_c \mathbb{E}_{c-1}, \\ \mathbb{N}_{c,c} = l_c^2(\alpha_{c-1}\mathbb{M}_c^{c-1} + \alpha_c \mathbb{M}_c^{c}), \\ \mathbb{N}_{c,c+1} = \alpha_c l_c l_{c+1}\mathbb{E}_c^t. \end{cases} \tag{2.74}$$

We can see that the half-edge velocities satisfies a linear system which has a tridiagonal block structure. We can also note it is a cyclic system due to the cyclic numbering of the half-edges and cells. From the first equation it follows that

$$\mathbb{N}_{c+1,c} = \alpha_c l_c l_{c+1}\mathbb{E}_c.$$

We observe that $\mathbb{N}_{c+1,c}$ is exactly the transpose of $\mathbb{N}_{c,c+1}$. Finally, the bidiagonal cyclic matrix $\mathbb{S}$ is characterized by writing the non-zero entries corresponding to the cth row

$$\begin{cases} \mathbb{S}_{c,c-1} = \alpha_{c-1} l_c (l_{c-1}\mathbb{E}_{c-1} + l_c \mathbb{M}_c^{c-1}), \\ \mathbb{S}_{c,c} = \alpha_c l_c (l_c \mathbb{M}_c^{c} + l_{c+1}\mathbb{E}_c^t). \end{cases} \tag{2.75}$$

To achieve the elimination of the half-edge velocities, it remains to show that the system (2.73) is invertible.

2.5.7 Properties of the matrices $\mathbb{N}$ and $\mathbb{S}$.

Here, we present some interesting properties of the matrices $\mathbb{N}$ and $\mathbb{S}$.

Invertibility of matrix $\mathbb{N}$

Let us first show that matrix $\mathbb{N}$ is invertible. To this end, we define the matrix $\mathbb{P}^c$, which is a block matrix of size $2 \times \mathfrak{C}_p$ characterized by the two following non-zero blocks

$$\begin{cases} \mathbb{P}^c_{1,c} = l_c \mathbb{I}, \\ \mathbb{P}^c_{2,c+1} = l_{c+1} \mathbb{I}. \end{cases} \tag{2.76}$$

Employing this matrix allows to express matrix $\mathbb{N}$ as

$$\mathbb{N} = \sum_{c=1}^{\mathfrak{C}_p} \alpha_c (\mathbb{P}^c)^t \mathbb{H}^c \mathbb{P}^c,$$

where $\mathbb{H}^c$ is the block matrix $\begin{pmatrix} \mathbb{M}^c_c & \mathbb{E}^t_c \\ \mathbb{E}_c & \mathbb{M}^c_{c+1} \end{pmatrix}$. In Section 2.5.5 we have shown that this block matrix is positive definite and thus invertible. Bearing in mind this decomposition we claim that $\mathbb{N}$ is a positive definite matrix, namely

$$\mathbb{N}\overline{\boldsymbol{V}} \cdot \overline{\boldsymbol{V}} > 0, \, \forall \overline{\boldsymbol{V}} \in \mathfrak{R}^{\mathfrak{C}_p}.$$

We use the decomposition of $\mathbb{N}$ to write

$$\mathbb{N}\overline{\boldsymbol{V}} \cdot \overline{\boldsymbol{V}} = \left(\sum_{c=1}^{\mathfrak{C}_p} \alpha_c (\mathbb{P}^c)^t \mathbb{H}^c \mathbb{P}^c \right) \overline{\boldsymbol{V}} \cdot \overline{\boldsymbol{V}},$$

rearranging the terms yields

$$\mathbb{N}\overline{\boldsymbol{V}} \cdot \overline{\boldsymbol{V}} = \sum_{c=1}^{\mathfrak{C}_p} \alpha_c \mathbb{H}^c (\mathbb{P}^c \overline{\boldsymbol{V}}) \cdot (\mathbb{P}^c \overline{\boldsymbol{V}}),$$

and since $\mathbb{H}^c$ is positive definite, the right-hand side of this equation is always positive, which ends the proof.

Preservation of uniform velocity fields

An interesting property of the scheme is that it preserves uniform velocity fields. This property is obtained thanks to the following relation

$$\left(\mathbb{N}^{-1}\mathbb{S}\right) \mathbf{1}_{\mathfrak{C}_p} = \mathbf{1}_{\mathfrak{C}_p}, \tag{2.77}$$

where $\mathbf{1}_n$, with n an integer, is the vector of size $2n$, whose entries are equal to 1. The size $2n$ is due to the block-definition of the matrices which are composed of blocks of size 2×2. To demonstrate the above relation, let us show that $\mathbb{S}\mathbf{1}_{\mathfrak{C}_p} = \mathbb{N}\mathbf{1}_{\mathfrak{C}_p}$ by developing respectively the left and the right-hand side of this equality. Substituting the non-zero entries of matrix $\mathbb{S}$ leads to write the left-hand side

$$\begin{aligned} \left(\mathbb{S}\mathbf{1}_{\mathfrak{C}_p}\right)_c &= \mathbb{S}_{c,c-1} + \mathbb{S}_{c,c} \\ &= \alpha_{c-1} l_c (l_{c-1}\mathbb{E}_{c-1} + l_c \mathbb{M}^{c-1}_c) + \alpha_c l_c (l_c \mathbb{M}^c_c + l_{c+1}\mathbb{E}^t_c) \\ &= \alpha_{c-1} l_{c-1} l_c \mathbb{E}_{c-1} + l_c^2 (\alpha_{c-1}\mathbb{M}^{c-1}_c + \alpha_c \mathbb{M}^c_c) + \alpha_c l_c l_{c+1} \mathbb{E}^t_c. \end{aligned}$$

Replacing the non-zero entries of matrix $\mathbb{N}$ allows to express the right-hand side as

$$\begin{aligned} \left(\mathbb{N}\mathbf{1}_{\mathfrak{C}_p}\right)_c &= \mathbb{N}_{c,c-1} + \mathbb{N}_{c,c} + \mathbb{N}_{c,c+1} \\ &= \alpha_{c-1} l_{c-1} l_c \mathbb{E}_{c-1} + l_c^2 (\alpha_{c-1}\mathbb{M}^{c-1}_c + \alpha_c \mathbb{M}^c_c) + \alpha_c l_c l_{c+1} \mathbb{E}^t_c. \end{aligned}$$

Comparing the two above expressions shows that the equality (2.77) is satisfied.

2.5.8 Boundary conditions

We describe a generic methodology to implement the boundary conditions. It is worth mentioning that the boundary terms discretization is derived in a consistent manner with the scheme construction. To take into account the boundary terms, let us write the linear system linking the half-edge velocities with the mean cell velocities under the form

$$\mathbb{N}\bar{\boldsymbol{V}} = \mathbb{S}\boldsymbol{V} + \boldsymbol{B}, \tag{2.78}$$

where the extra term $\boldsymbol{B}$ is the block vector containing the boundary conditions contribution, which shall be defined in the next paragraphs. We have to note that on the boundaries a point p is surrounded by $\mathfrak{C}_p$ cells and $\mathfrak{C}_p+1$ edges. The size of the matrix $\mathbb{N}$ is then $(\mathfrak{C}_p+1)\times(\mathfrak{C}_p+1)$ on the boundaries, the size of the matrix $\mathbb{S}$ is $(\mathfrak{C}_p+1)\times\mathfrak{C}_p$. The size of the block-vector $\bar{\boldsymbol{V}}$ is $(\mathfrak{C}_p+1)$, $\boldsymbol{V}$ is of size $\mathfrak{C}_p$ and the size of $\boldsymbol{B}$ is $(\mathfrak{C}_p+1)$.

Let us consider a half-edge c located on the boundary of the domain, in the next paragraphs, we describe the modifications to bring to the matrices and boundary vector, depending on the boundary conditions types under consideration. If not specified the value of $\boldsymbol{B}_c$ is equal to zero.

Free boundary condition

On this boundary we want to impose the value of the half-edge velocity to be equal to $\boldsymbol{V}^\star$. It means we want to write

$$\bar{\boldsymbol{V}}_c = \boldsymbol{V}^\star.$$

To do so it remains only one non-zero terms on the c-th row of block matrix $\mathbb{N}$. It writes as

$$\mathbb{N}_{c,c} = \mathbb{I}_2.$$

The c-th row of block matrix $\mathbb{S}$ is set equal to zero. To impose the boundary value it remains to set

$$\boldsymbol{B}_c = \boldsymbol{V}^\star.$$

The Neumann boundary condition

On this boundary we impose the normal gradient to the boundary, namely

$$(\nabla\boldsymbol{V})\boldsymbol{n} = \boldsymbol{G}^\star.$$

Let us consider that we are dealing with the edge $\mathfrak{C}_p+1$, in that case the discrete expression of $\nabla\boldsymbol{V}\boldsymbol{n}$ is defined by $\mathbb{G}^c_{c+1}$. We recall its definition

$$\mathbb{G}^c_{c+1} = \alpha_c\left[l_{c+1}\mathbb{I}(\overline{\boldsymbol{V}}_{c+1} - \boldsymbol{V}_c) - l_c\cos\theta_c\mathbb{I}(\overline{\boldsymbol{V}}_c - \boldsymbol{V}_c)\right].$$

We can rewrite this expression under a matricial form. The non zero terms of the $(\mathfrak{C}_p+1)$-th row of matrix $\mathbb{N}$ write

$$\begin{cases} \mathbb{N}_{\mathfrak{C}_p+1,\mathfrak{C}_p} = -\alpha_{\mathfrak{C}_p} l_{\mathfrak{C}_p}\cos\theta_{\mathfrak{C}_p}\mathbb{I}, \\ \mathbb{N}_{\mathfrak{C}_p+1,\mathfrak{C}_p+1} = \alpha_{\mathfrak{C}_p} l_{\mathfrak{C}_p+1}\mathbb{I}. \end{cases}$$

There is only one non-zero term on the $(\mathfrak{C}_p+1)$-th row of matrix $\mathbb{S}$. It writes

$$\mathbb{S}_{\mathfrak{C}_p+1,\mathfrak{C}_p} = \alpha_{\mathfrak{C}_p} l_{\mathfrak{C}_p+1}\mathbb{I} - \alpha_{\mathfrak{C}_p} l_{\mathfrak{C}_p}\cos\theta_{\mathfrak{C}_p}\mathbb{I}.$$

To impose the boundary value it remains to set

$$\boldsymbol{B}_{\mathfrak{C}_p+1} = \boldsymbol{G}^\star.$$

The case corresponding to edge number 1 is obtained in a similar manner utilizing the definition of $\mathbb{G}^c_c$.

The symmetrical boundary condition

On this boundary we need to apply the two scalar expressions $\boldsymbol{V}\cdot\boldsymbol{n}=0$ and $(\nabla\boldsymbol{V}\boldsymbol{n})\cdot\boldsymbol{t}=0$. These two scalar expressions are the projections on a basis of the two boundary conditions we just described above. The first relation is a kinematic boundary condition projected on vector $\boldsymbol{n}$ and the second one, a Neumann boundary condition projected on $\boldsymbol{t}$. Once again we consider we are dealing with the edge $\mathfrak{C}_p+1$ so that we can express $\nabla\boldsymbol{V}\boldsymbol{n}$ as $\mathbb{G}^c_{c+1}$. Note that we could have presented the discretization with edge number 1 and the definition of $\mathbb{G}^c_c$. We note $\boldsymbol{n}=\begin{pmatrix}n_1\\n_2\end{pmatrix}$ and $\boldsymbol{t}=\begin{pmatrix}t_1\\t_2\end{pmatrix}$. Let us write the two scalar expressions under the matricial form. The non-zero terms of the $(\mathfrak{C}_p+1)$-th row of matrix $\mathbb{N}$ write

$$\begin{cases}\mathbb{N}_{\mathfrak{C}_p+1,\mathfrak{C}_p}=-\alpha_{\mathfrak{C}_p}l_{\mathfrak{C}_p}\cos\theta_{\mathfrak{C}_p}\begin{pmatrix}0&0\\t_1&t_2\end{pmatrix},\\ \mathbb{N}_{\mathfrak{C}_p+1,\mathfrak{C}_p+1}=\begin{pmatrix}n_1&n_2\\\alpha_{\mathfrak{C}_p}l_{\mathfrak{C}_p+1}t_1&\alpha_{\mathfrak{C}_p}l_{\mathfrak{C}_p+1}t_2\end{pmatrix}.\end{cases}$$

There is only one non-zero term on the $(\mathfrak{C}_p+1)$-th row of matrix $\mathbb{S}$. It writes

$$\mathbb{S}_{\mathfrak{C}_p+1,\mathfrak{C}_p}=\alpha_{\mathfrak{C}_p}(l_{\mathfrak{C}_p+1}-l_{\mathfrak{C}_p}\cos\theta_{\mathfrak{C}_p})\begin{pmatrix}0&0\\t_1&t_2\end{pmatrix}.$$

The block vector $\boldsymbol{B}$ is set to

$$\boldsymbol{B}_{\mathfrak{C}_p+1}=\boldsymbol{0}.$$

2.5.9 Construction of the global linear system

In this paragraph, we achieve the space discretization of the diffusion equation gathering the results obtained in the previous sections. We recall that the semi discrete scheme (2.31) writes

$$m_c\frac{d}{dt}\boldsymbol{V}_c-\sum_{p\in\mathcal{P}(c)}l^-_{pc}\boldsymbol{\Sigma}^-_{pc}+l^+_{pc}\boldsymbol{\Sigma}^+_{pc}=\boldsymbol{0}. \tag{2.79}$$

We define the contribution of the sub-cell ω_{pc} to the diffusion flux as

$$\boldsymbol{Q}_{pc}=l^-_{pc}\boldsymbol{\Sigma}^-_{pc}+l^+_{pc}\boldsymbol{\Sigma}^+_{pc}$$

Using the local numbering of the half-edges surrounding point p yields to rewrite the above expression as

$$\boldsymbol{Q}_{pc}=l_c\boldsymbol{\Sigma}^c_c+l_{c+1}\boldsymbol{\Sigma}^c_{c+1}.$$

Now, we replace the normal fluxes by their corresponding expressions (2.66) and (2.67) to get

$$\boldsymbol{Q}_{pc}=\alpha_c l_c\left[l_{c+1}\mathbb{E}^t_c(\overline{\boldsymbol{V}}_{c+1}-\boldsymbol{V}_c)+l_c\mathbb{M}^c_c(\overline{\boldsymbol{V}}_c-\boldsymbol{V}_c)\right]+\alpha_c l_{c+1}\left[l_{c+1}\mathbb{M}^c_{c+1}(\overline{\boldsymbol{V}}_{c+1}-\boldsymbol{V}_c)+l_c\mathbb{E}_c(\overline{\boldsymbol{V}}_c-\boldsymbol{V}_c)\right].$$

Rearranging the order of the terms in the right-hand side yields

$$\boldsymbol{Q}_{pc}=\alpha_c l_{c+1}\left[l_c\mathbb{E}^t_c+l_{c+1}\mathbb{M}^c_{c+1}\right](\overline{\boldsymbol{V}}_{c+1}-\boldsymbol{V}_c)+\alpha_c l_c\left[l_c\mathbb{M}^c_c+l_{c+1}\mathbb{E}_c\right](\overline{\boldsymbol{V}}_c-\boldsymbol{V}_c).$$

To obtain a more compact form of $\boldsymbol{Q}_{pc}$, we define the block-matrix $\widetilde{\mathbb{S}}$ whose entries write

$$\begin{cases}\widetilde{\mathbb{S}}_{c,c}=\alpha_c l_c(l_c\mathbb{M}^c_c+l_{c+1}\mathbb{E}_c),\\ \widetilde{\mathbb{S}}_{c+1,c}=\alpha_c l_{c+1}(l_c\mathbb{E}^t_c+l_{c+1}\mathbb{M}^c_{c+1}).\end{cases} \tag{2.80}$$

Employing this notation, the sub-cell contribution to the diffusion flux reads

$$\boldsymbol{Q}_{pc} = \widetilde{\mathbb{S}}_{c+1,c}(\overline{\boldsymbol{V}}_{c+1} - \boldsymbol{V}_c) + \widetilde{\mathbb{S}}_{c,c}(\overline{\boldsymbol{V}}_c - \boldsymbol{V}_c).$$

Which rewrites

$$\boldsymbol{Q}_{pc} = \sum_{d \in \mathcal{C}(p)} \widetilde{\mathbb{S}}^t_{cd}(\overline{\boldsymbol{V}}_d - \boldsymbol{V}_c).$$

Eliminating the half-edge velocities by means of (2.73) and using the property (2.77) leads to

$$\boldsymbol{Q}_{pc} = \sum_{d \in \mathcal{C}(p)} \mathbb{G}^p_{cd}(\boldsymbol{V}_d - \boldsymbol{V}_c) + \boldsymbol{B}^p_d, \tag{2.81}$$

where $\mathbb{G}^p$ is a $\mathfrak{C}_p \times \mathfrak{C}_p$ matrix defined at point p by

$$\mathbb{G}^p = \widetilde{\mathbb{S}}^t \mathbb{N}^{-1} \mathbb{S}. \tag{2.82}$$

$\boldsymbol{B}^p$ is the vector containing the boundary conditions information. It is defined by

$$\boldsymbol{B}^p = \widetilde{\mathbb{S}}^t \mathbb{N}^{-1} \boldsymbol{B}. \tag{2.83}$$

Taking into account the previous results, the semi-discrete scheme over cell c reads

$$m_c \frac{\mathsf{d}}{\mathsf{d}t} \boldsymbol{V}_c - \sum_{p \in \mathcal{P}(c)} \sum_{d \in \mathcal{C}(p)} \mathbb{G}^p_{cd}(\boldsymbol{V}_d - \boldsymbol{V}_c) = \sum_{p \in \mathcal{P}(c)} \sum_{d \in \mathcal{C}(p)} \boldsymbol{B}^p_d, \tag{2.84}$$

where $\mathcal{P}(c)$ is the set of points of cell c and $\mathcal{C}(p)$ is the set of cells surrounding the point p. This equation allows to construct the generic entries of the global tensorial diffusion matrix, $\mathbb{T}$, as follows

$$\mathbb{T}_{cc} = - \sum_{p \in \mathcal{P}(c)} \sum_{d \in \mathcal{C}(p)} \mathbb{G}^p_{cd}, \tag{2.85a}$$

$$\mathbb{T}_{cd} = \sum_{p \in \mathcal{P}(c)} \mathbb{G}^p_{cd}, c \neq d. \tag{2.85b}$$

Let $\mathfrak{C}_\mathcal{D}$ be the total number of cells composing the computational grid, we denote by $\boldsymbol{\mathcal{V}} \in \mathfrak{R}^{2\mathfrak{C}_\mathcal{D}}$ the vector of cell-centered velocities which has $2\mathfrak{C}_\mathcal{D}$ components in the Cartesian frame $(0, x, y)$. Gathering the previous results, this vector is the solution of the global linear system

$$\mathbb{M} \frac{\mathsf{d}}{\mathsf{d}t} \boldsymbol{\mathcal{V}} + \mathbb{T} \boldsymbol{\mathcal{V}} = \boldsymbol{\mathcal{B}}. \tag{2.86}$$

Here, $\mathbb{M}$ is a $2\mathfrak{C}_\mathcal{D} \times 2\mathfrak{C}_\mathcal{D}$ block diagonal matrix whose entries are the cell mass m_c, and $\boldsymbol{\mathcal{B}}$ is the vector collecting the boundary conditions contributions. Finally, matrix $\mathbb{T}$ is a $2\mathfrak{C}_\mathcal{D} \times 2\mathfrak{C}_\mathcal{D}$ block matrix whose entries are given by (2.85).
To achieve the construction of the numerical method it remains to describe the time discretization. This is the topic of the next section.

2.6 Time discretization

In this section, we briefly describe the time discretization of the system (2.86). First, let us prescribe the initial condition $\boldsymbol{\mathcal{V}}(0) = \boldsymbol{\mathcal{V}}^0$, where $\boldsymbol{\mathcal{V}}^0$ is the vector of the cell-averaged initial condition. We solve the system over the time interval $[0, \mathfrak{T}]$ using the subdivision

$$0 = t^0 < t^1 < \cdots < t^n < t^{n+1} < \cdots < t^N = \mathfrak{T}.$$

The time step is denoted by $\Delta t^n = t^{n+1} - t^n$. The time approximation of a quantity at time t^n is denoted using the superscript n, for instance $\boldsymbol{\mathcal{V}}^n = \boldsymbol{\mathcal{V}}(t^n)$. Knowing that an explicit time discretization of the tensorial diffusion operator necessitates a stability constraint on the time step which is quadratic with respect to the smallest cell size, we prefer to use an implicit time discretization. Integrating (2.86) over $[t^n, t^{n+1}]$ yields the first-order in time implicit discrete scheme

$$\mathbb{M}\frac{\boldsymbol{\mathcal{V}}^{\boldsymbol{n+1}} - \boldsymbol{\mathcal{V}}^{\boldsymbol{n}}}{\Delta t^n} + \mathbb{T}\boldsymbol{\mathcal{V}}^{n+1} = \boldsymbol{\mathcal{B}}^n. \tag{2.87}$$

The updated cell-centered velocities are obtained by solving the following linear system

$$\left(\frac{\mathbb{M}}{\Delta t^n} + \mathbb{T}\right)\boldsymbol{\mathcal{V}}^{n+1} = \frac{\mathbb{M}}{\Delta t^n}\boldsymbol{\mathcal{V}}^n + \boldsymbol{\mathcal{B}}^n. \tag{2.88}$$

Let us recall that $\mathbb{T}$ is a positive semi-definite block matrix. Knowing that $\mathbb{M}$ is a positive block diagonal matrix, we deduce that the matrix $\frac{\mathbb{M}}{\Delta t^n} + \mathbb{T}$ is positive definite. Thus, the linear system (2.88) always admits a unique solution. Finally, in the absence of source term and assuming periodic or homogeneous boundary conditions, we observe that the above implicit time discretization is stable with respect to the discrete weighted L^2 norm defined by

$$\|\boldsymbol{\mathcal{V}}\|_{w2}^2 = (\mathbb{M}\boldsymbol{\mathcal{V}} \cdot \boldsymbol{\mathcal{V}}),$$

where $\boldsymbol{\mathcal{V}}$ is a vector of size $2\mathfrak{C}_{\mathcal{D}}$. To prove this result, we dot-multiply (2.88) by $\boldsymbol{\mathcal{V}}^{n+1}$ and obtain

$$(\mathbb{M}\boldsymbol{\mathcal{V}}^{n+1} \cdot \boldsymbol{\mathcal{V}}^{n+1}) - (\mathbb{M}\boldsymbol{\mathcal{V}}^n \cdot \boldsymbol{\mathcal{V}}^{n+1}) = -\Delta t^n(\mathbb{T}\boldsymbol{\mathcal{V}}^{n+1} \cdot \boldsymbol{\mathcal{V}}^{n+1}).$$

Due to the positive definiteness of matrix $\mathbb{T}$ the right-hand side of the above equation is negative, hence

$$(\mathbb{M}\boldsymbol{\mathcal{V}}^{n+1} \cdot \boldsymbol{\mathcal{V}}^{n+1}) \leq (\mathbb{M}\boldsymbol{\mathcal{V}}^n \cdot \boldsymbol{\mathcal{V}}^{n+1}).$$

Employing Cauchy-Schwarz inequality in the right-hand side of the above inequality yields

$$(\mathbb{M}\boldsymbol{\mathcal{V}}^n \cdot \boldsymbol{\mathcal{V}}^{n+1}) \leq \|\boldsymbol{\mathcal{V}}^n\|_{w2}\|\boldsymbol{\mathcal{V}}^{n+1}\|_{w2}.$$

Gathering the above results leads to

$$\|\boldsymbol{\mathcal{V}}^{n+1}\|_{w2} \leq \|\boldsymbol{\mathcal{V}}^n\|_{w2},$$

which ends the proof.

Comment 19: *The computation of the numerical solution requires to solve the sparse linear system* (2.88)*. Once again, this is achieved by employing the localized ILU(0) Preconditioned BiCGStab algorithm, refer to [127, 95]. The matrices encountered in this chapter are not symmetric anymore, so using a classical conjugate gradient method was not sufficient. Therefore, the implementation of a more general solver is this time justified.*

Comment 20: *We are now going to mention the parallelization of the scheme. The construction of the tensorial diffusion scheme we just presented follows the same methodology used by the CCLAD scheme. This means that it can also benefits from the developments made for the parallelization of the CCLAD scheme. The only notable difference when constructing the global linear system is, that the notion of matrix and vector needs to be replaced by the notion of block-matrix and block-vector. This small modification was straightforward to implement in our code. With these developments, our code is able to solve a distributed linear system represented as a block-matrix and block-vector using a parallel localized block-ILU(0) Preconditioned [61] BiCGStab algorithm. The only difference in the algorithm being in the storage of the matrix and vector, the parallel efficiency observed are the same as the one presented in section 1.6.*

2.7 Mathematical properties of the scheme

The aim of this section is to present the mathematical properties of the numerical scheme. We start by a discussion on the L^2 stability of the semi discrete scheme. Then, we prove that it is possible to compute the volume weights in such a way that the fundamental identity $\nabla \cdot (\bar{\bar{\mathbb{D}}} \nabla \boldsymbol{V}) = \boldsymbol{0}$, where $\bar{\bar{\mathbb{D}}}$ is the fourth-order tensor defined by Eq. (2.16), is satisfied at the discrete level by our Finite Volume space discretization. This compatibility condition is of great importance since it ensures that the viscous tress, $\mathbb{S}$, and the pseudo-viscous stress, $\mathbb{\Sigma}$, have the same divergence, *i.e.*, $\nabla \cdot \mathbb{S} = \nabla \cdot \mathbb{\Sigma}$. In this sense, we have constructed a compatible Finite Volume method which mimics at the discrete level the identity satisfies by the mathematical operators at the continuum level.

2.7.1 L^2 stability of the semi discrete scheme

Let us recall that the semi-discrete form of our Finite Volume scheme writes (refer to Eq. (2.31))

$$m_c \frac{d}{dt} \boldsymbol{V}_c - \sum_{p \in \mathcal{P}(c)} l_{pc}^- \boldsymbol{\Sigma}_{pc}^- + l_{pc}^+ \boldsymbol{\Sigma}_{pc}^+ = \boldsymbol{0},$$

where $\boldsymbol{\Sigma}_{pc}^{\pm}$ is the half-edge flux related to the pseudo-viscous stress, refer to Section 2.3. To obtain the time rate of change of kinetic energy we dot-multiply the above equation by the cell-centered velocity $\boldsymbol{V}_c$ as follows

$$m_c \frac{d}{dt} \left(\frac{\boldsymbol{V}_c^2}{2} \right) = \sum_{p \in \mathcal{P}(c)} \left(l_{pc}^- \boldsymbol{\Sigma}_{pc}^- + l_{pc}^+ \boldsymbol{\Sigma}_{pc}^+ \right) \cdot \boldsymbol{V}_c. \tag{2.89}$$

We assume that the solution is periodic or characterized by a compact support so that we do not care about the boundary conditions. Summing (2.89) over all the cells yields

$$\begin{aligned} \frac{d}{dt} \left(\sum_c \frac{1}{2} m_c \boldsymbol{V}_c^2 \right) &= \sum_c \sum_{p \in \mathcal{P}(c)} \left(l_{pc}^- \boldsymbol{\Sigma}_{pc}^- + l_{pc}^+ \boldsymbol{\Sigma}_{pc}^+ \right) \cdot \boldsymbol{V}_c, \\ &= \sum_p \sum_{c \in \mathcal{C}(p)} \left(l_{pc}^- \boldsymbol{\Sigma}_{pc}^- + l_{pc}^+ \boldsymbol{\Sigma}_{pc}^+ \right) \cdot \boldsymbol{V}_c, \end{aligned}$$

where $\mathcal{C}(p)$ is the set of cells sharing the vertex p.
To investigate the sign of the right-hand side of the above equation, we set

$$I_p = \sum_{c \in \mathcal{C}(p)} \left(l_{pc}^- \boldsymbol{\Sigma}_{pc}^- + l_{pc}^+ \boldsymbol{\Sigma}_{pc}^+ \right) \cdot \boldsymbol{V}_c.$$

The continuity of the pseudo-viscous stress across cell interfaces implies that

$$\sum_{c \in \mathcal{C}(p)} l_{pc}^- \boldsymbol{\Sigma}_{pc}^- \cdot \boldsymbol{V}_{pc}^- + l_{pc}^+ \boldsymbol{\Sigma}_{pc}^+ \cdot \boldsymbol{V}_{pc}^+ = \boldsymbol{0}.$$

Thus, I_p can be rewritten as $I_p = -\sum_{c \in \mathcal{C}(p)} l_{pc}^- \boldsymbol{\Sigma}_{pc}^- \cdot (\boldsymbol{V}_{pc}^- - \boldsymbol{V}_c) + l_{pc}^+ \boldsymbol{\Sigma}_{pc}^+ \cdot (\boldsymbol{V}_{pc}^+ - \boldsymbol{V}_c)$.

Recalling that $\boldsymbol{\Sigma}_{pc}^{+/-} = \mathbb{\Sigma}_{pc} \boldsymbol{n}_{pc}^{+/-}$ leads to

$$I_p = -\sum_{c \in \mathcal{C}(p)} l_{pc}^- \mathbb{\Sigma}_{pc} \boldsymbol{n}_{pc}^- \cdot (\boldsymbol{V}_{pc}^- - \boldsymbol{V}_c) + l_{pc}^+ \mathbb{\Sigma}_{pc} \boldsymbol{n}_{pc}^+ \cdot (\boldsymbol{V}_{pc}^+ - \boldsymbol{V}_c).$$

Noticing that for all vectors $\boldsymbol{a}, \boldsymbol{b}$ and tensor $\mathbb{T}$, there holds $\boldsymbol{a} \cdot \mathbb{T}^t \boldsymbol{b} = (\boldsymbol{b} \otimes \boldsymbol{a}) : \mathbb{T}$, we rewrite I_p as

$$\begin{aligned} I_p &= -\sum_{c \in \mathcal{C}(p)} \left[l_{pc}^-(\boldsymbol{V}_{pc}^- - \boldsymbol{V}_c) \otimes \boldsymbol{n}_{pc}^- + l_{pc}^+(\boldsymbol{V}_{pc}^+ - \boldsymbol{V}_c) \otimes \boldsymbol{n}_{pc}^+ \right] : \mathbb{\Sigma}_{pc}, \\ &= -\sum_{c \in \mathcal{C}(p)} w_{pc} \mathbb{G}_{pc} : \bar{\bar{\mathbb{R}}}_c(\mathbb{G}_{pc}). \end{aligned}$$

Here, $\bar{\bar{\mathbb{R}}}$ is the fourth-order tensor defined by Eq. (2.53). Since $\bar{\bar{\mathbb{R}}}$ is positive definite, the inner product $\bar{\bar{\mathbb{R}}}_c(\mathbb{G}_{pc}) : \mathbb{G}_{pc}$ is non negative and thus $I_p < 0$. Therefore, our Finite Volume scheme satisfies

$$\frac{d}{dt}\left(\sum_c \frac{1}{2} m_c \boldsymbol{V}_c^2 \right) < 0.$$

This shows its L^2 stability at the semi-discrete level.

2.7.2 Definition of the volume weight

During the construction of our Finite Volume method we have introduced the volume weight w_{pc}, which needs to be defined for over each sub-cell ω_{pc}. So far, we have just stated that w_{pc} should be non negative and that $\sum_{p \in \mathcal{P}(c)} w_{pc} = |\omega_c|$. Here, we shall show how to compute the volume weight w_{pc} to ensure that the identity $\nabla \cdot \mathbb{S} = \nabla \cdot \mathbb{\Sigma}$ is satisfied at the discrete level. More precisely, this amounts to prove that the following tensorial identity $\nabla \cdot (\bar{\bar{\mathbb{D}}} \nabla \boldsymbol{V}) = \boldsymbol{0}$ holds at the discrete level.

Formulas for triangular and quadrangular cells

In Section 2.5, employing a sub-cell-based variational formulation, we have derived the following approximation of the velocity gradient tensor over the sub-cell ω_{pc}

$$\mathbb{G}_{pc} = \frac{1}{w_{pc}} \left[l_{pc}^-(\boldsymbol{V}_{pc}^- - \boldsymbol{V}_c) \otimes \boldsymbol{n}_{pc}^- + l_{pc}^+(\boldsymbol{V}_{pc}^+ - \boldsymbol{V}_c) \otimes \boldsymbol{n}_{pc}^+ \right],$$

where, w_{pc} denotes the volume weight related to ω_{pc}. It remains to specify how to compute this volume weight to ensure a sufficient accuracy for the above formula. To this end, let us consider a polygonal cell ω_c. We set $\mathbb{G} = \nabla \boldsymbol{V}$ and we define the cell-averaged value of the velocity gradient as $\mathbb{G}_c = \frac{1}{|\omega_c|} \int_{\omega_c} \nabla \boldsymbol{V} dv$. Applying the divergence formula leads to $\mathbb{G}_c = \frac{1}{|\omega_c|} \int_{\partial \omega_c} \boldsymbol{V} \otimes \boldsymbol{n} \, \mathrm{d}s$ and

$$\mathbb{G}_c = \frac{1}{|\omega_c|} \sum_{f \in \mathcal{F}(c)} l_f \boldsymbol{V}_f \otimes \boldsymbol{n}_f,$$

where $\mathcal{F}(c)$ is the set of faces of cell c and l_f, $\boldsymbol{V}_f$ and $\boldsymbol{n}_f$ denotes the length, the velocity and the unit outward pointing normal to the face f. Let us focus on the triangular cell ω_c with vertices p,q and r. We assume that the velocity field is linear with respect to the space variable. The following results hold:

- If $f = [p, q]$ then $\boldsymbol{V}_f = \frac{1}{2}(\boldsymbol{V}_p + \boldsymbol{V}_q)$.
- We have the geometric identity $\sum_{f \in \mathcal{F}(c)} l_f \boldsymbol{n}_f = \boldsymbol{0}$.

One can note that the above properties are true on any polygonal cell. Using the geometric identity we get $l_{qr}\boldsymbol{n}_{qr} = -\left(l_{pq}\boldsymbol{n}_{pq} + l_{rp}\boldsymbol{n}_{rp}\right)$, thus the velocity gradient over ω_c writes

$$\mathbb{G}_c = \frac{1}{|\omega_c|}\left[l_{pq}(\boldsymbol{V}_{pq} - \boldsymbol{V}_{qr}) \otimes \boldsymbol{n}_{pq} + l_{rp}(\boldsymbol{V}_{rp} - \boldsymbol{V}_{qr}) \otimes \boldsymbol{n}_{rp}\right], \tag{2.90}$$

with

$$\boldsymbol{V}_{pq} - \boldsymbol{V}_{qr} = \frac{1}{2}\left(\boldsymbol{V}_p + \boldsymbol{V}_q - \boldsymbol{V}_q - \boldsymbol{V}_r\right) = \frac{1}{2}\left(\boldsymbol{V}_p - \boldsymbol{V}_r\right), \tag{2.91}$$

$$\boldsymbol{V}_{rp} - \boldsymbol{V}_{qr} = \frac{1}{2}\left(\boldsymbol{V}_r + \boldsymbol{V}_p - \boldsymbol{V}_q - \boldsymbol{V}_r\right) = \frac{1}{2}\left(\boldsymbol{V}_p - \boldsymbol{V}_q\right). \tag{2.92}$$

Using theses notations equation (2.90) rewrites

$$\mathbb{G}_c = \frac{1}{2|\omega_c|}\left[l_{pq}(\boldsymbol{V}_p - \boldsymbol{V}_r) \otimes \boldsymbol{n}_{pq} + l_{rp}(\boldsymbol{V}_p - \boldsymbol{V}_q) \otimes \boldsymbol{n}_{rp}\right]. \tag{2.93}$$

We introduce the cell-centered velocity $\boldsymbol{V}_c = \frac{1}{3}\left(\boldsymbol{V}_p + \boldsymbol{V}_q + \boldsymbol{V}_r\right)$. It can be used under the following forms $\boldsymbol{V}_r = 3\boldsymbol{V}_c - \boldsymbol{V}_p - \boldsymbol{V}_q$ and $\boldsymbol{V}_q = 3\boldsymbol{V}_c - \boldsymbol{V}_p - \boldsymbol{V}_r$ to obtain

$$\begin{aligned}\boldsymbol{V}_p - \boldsymbol{V}_r &= \boldsymbol{V}_p - 3\boldsymbol{V}_c + \boldsymbol{V}_p + \boldsymbol{V}_q,\\ &= 3\left(\frac{2\boldsymbol{V}_p + \boldsymbol{V}_q}{3} - \boldsymbol{V}_c\right),\end{aligned}$$

and

$$\begin{aligned}\boldsymbol{V}_p - \boldsymbol{V}_q &= \boldsymbol{V}_p - 3\boldsymbol{V}_c + \boldsymbol{V}_p + \boldsymbol{V}_r,\\ &= 3\left(\frac{2\boldsymbol{V}_p + \boldsymbol{V}_r}{3} - \boldsymbol{V}_c\right).\end{aligned}$$

Finally, the cell-averaged velocity gradient writes

$$\mathbb{G}_c = \frac{1}{\frac{2|\omega_c|}{3}}\left[l_{pq}(\boldsymbol{V}^\star_{pq} - \boldsymbol{V}_c) \otimes \boldsymbol{n}_{pq} + l_{rp}(\boldsymbol{V}^\star_{pr} - \boldsymbol{V}_c) \otimes \boldsymbol{n}_{rp}\right], \tag{2.94}$$

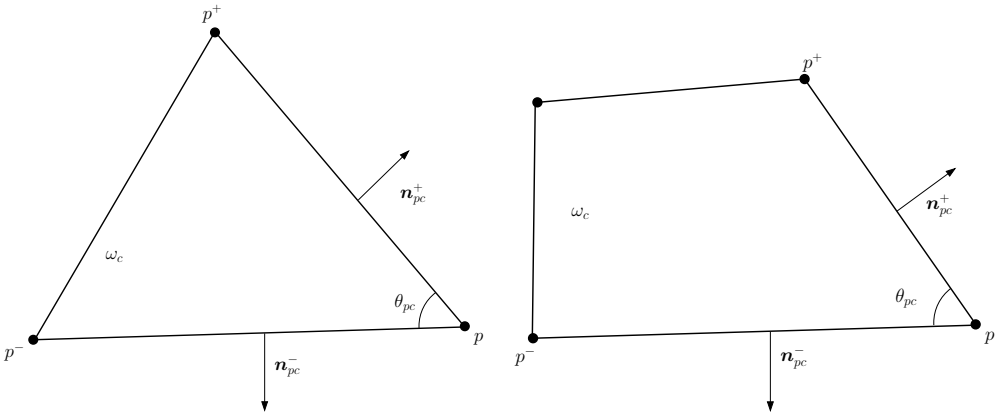
where $\boldsymbol{V}^\star_{pq} = \frac{2\boldsymbol{V}_p + \boldsymbol{V}_q}{3}$ and $\boldsymbol{V}^\star_{pr} = \frac{2\boldsymbol{V}_p + \boldsymbol{V}_r}{3}$.

With a slight change of notation, refer to Figure 2.6(a), we recover the formula obtained by means of the variational formulation. It is sufficient to set $\frac{1}{2}l_{pq} = l^+_{pc}$, $\frac{1}{2}l_{rp} = l^-_{pc}$, $\boldsymbol{n}_{pq} = \boldsymbol{n}^+_{pc}$, $\boldsymbol{n}_{rp} = \boldsymbol{n}^-_{pc}$, $\boldsymbol{V}^\star_{pq} = \boldsymbol{V}^+_{pc}$ and $\boldsymbol{V}^\star_{rp} = \boldsymbol{V}^-_{pc}$ in order to obtain

$$\mathbb{G}_c = \frac{1}{\frac{|\omega_c|}{3}}\left[l^-_{pc}(\boldsymbol{V}^-_{pc} - \boldsymbol{V}_c) \otimes \boldsymbol{n}^-_{pc} + l^+_{pc}(\boldsymbol{V}^+_{pc} - \boldsymbol{V}_c) \otimes \boldsymbol{n}^+_{pc}\right]. \tag{2.95}$$

The volume weight is given by $w_{pc} = \frac{1}{3}|\omega_c|$ moreover the half-edge velocities correspond to point wise values of the velocity field. We also have

$$\begin{aligned}|\omega_c| &= \frac{1}{2}\left(\boldsymbol{p}^-\boldsymbol{p} \times \boldsymbol{p}\boldsymbol{p}^+\right) \cdot \boldsymbol{e_z},\\ &= \frac{1}{2} 2l^-_{pc} 2l^+_{pc} \sin\theta_{pc},\\ &= 2l^-_{pc} l^+_{pc} \sin\theta_{pc}.\end{aligned}$$



(a) Triangular cell. (b) Quadrangular cell.

Figure 2.6: Local notation related to the point p and the cell ω_c.

so we can define w_{pc} as

$$w_{pc} = \frac{2}{3} l_{pc}^- l_{pc}^+ \sin\theta_{pc}. \tag{2.96}$$

We claim that this definition of the volume weight provides a space approximation of the velocity gradient tensor which is exact for linear velocity fields.

For a quadrangular cell, refer to Figure 2.6(b), we can define the volume weight as follows. We consider the triangle $p^- p p^+$, related to point p, its volume is given by $V_{pc} = \frac{1}{2}\left(\boldsymbol{p}^-\boldsymbol{p} \times \boldsymbol{p}\boldsymbol{p}^+\right)\cdot\boldsymbol{e_z} = 2 l_{pc}^- l_{pc}^+ \sin\theta_{pc}$. The summation of this volume over the set of nodes of the quadrangle ω_c is equal to twice the volume of ω_c. Thus, we have $|\omega_c| = \sum_{p\in\mathcal{P}(c)} l_{pc}^- l_{pc}^+ \sin\theta_{pc}$ and it is natural to introduce in this case

$$w_{pc} = l_{pc}^- l_{pc}^+ \sin\theta_{pc}. \tag{2.97}$$

Discretization of $\nabla\cdot(\bar{\bar{\mathbb{D}}}\nabla\boldsymbol{V})$

Having defined the volume weights for triangular and quadrangular cells in the previous paragraph, we are in position to prove that their definitions allows the formula $\nabla\cdot\mathbb{S} = \nabla\cdot\mathbb{\Sigma}$ to be satisfied at the discrete level. Let us point out that this formula is crucial since it ensures the equivalence between the tensorial diffusion equation expressed in terms of the viscous stress, $\mathbb{S}$ and the one expressed in terms of the pseudo-viscous stress, $\mathbb{\Sigma}$. Since this formula is a consequence of the tensorial identity $\nabla\cdot(\bar{\bar{\mathbb{D}}}\nabla\boldsymbol{V}) = \boldsymbol{0}$, where $\bar{\bar{\mathbb{D}}}$ is the fourth-order tensor defined by $\bar{\bar{\mathbb{D}}}\mathbb{T} = \mathrm{tr}\,\mathbb{T}\mathbb{I} - \mathbb{T}^t$ for all second-order tensor $\mathbb{T}$, we start by investigating the space discretization of the above tensorial identity. First, applying the divergence theorem over the cell ω_c leads to rewrite this tensorial identity as

$$\int_{\partial\omega_c} \bar{\bar{\mathbb{D}}}\nabla\boldsymbol{V}\boldsymbol{n}\,\mathrm{d}s = \boldsymbol{0},$$

where $\boldsymbol{n}$ is the unit outward normal to the cell boundary $\partial\omega_c$. Using the notation defined in Section 2.5, leads to write the space approximation of right-hand side of the above equation as follows

$$\int_{\omega_c} \bar{\bar{\mathbb{D}}}\nabla\boldsymbol{V}\boldsymbol{n}\,\mathrm{d}s = \sum_{p\in\mathcal{P}(c)} l_{pc}^-(\bar{\bar{\mathbb{D}}}\nabla\boldsymbol{V})_{pc}^- + l_{pc}^+(\bar{\bar{\mathbb{D}}}\nabla\boldsymbol{V})_{pc}^+. \tag{2.98}$$

Utilizing the definition of the fourth-order tensor, $\bar{\bar{\mathbb{D}}}$, the half-edge flux $(\bar{\bar{\mathbb{D}}}\nabla \boldsymbol{V})_{pc}^{-}$ is given by

$$\begin{aligned}
(\bar{\bar{\mathbb{D}}}\nabla \boldsymbol{V})_{pc}^{-} &= \operatorname{tr} \mathbb{G}_{pc}\boldsymbol{n}_{pc}^{-} - \mathbb{G}_{pc}^{t-}, \\
&= \frac{1}{w_{pc}}\left[l_{pc}^{-}(\boldsymbol{n}_{pc}^{-}\otimes\boldsymbol{n}_{pc}^{-})(\boldsymbol{V}_{pc}^{-}-\boldsymbol{V}_c) + l_{pc}^{+}(\boldsymbol{n}_{pc}^{-}\otimes\boldsymbol{n}_{pc}^{+})(\boldsymbol{V}_{pc}^{+}-\boldsymbol{V}_c)\right] \\
&- \frac{1}{w_{pc}}\left[l_{pc}^{-}(\boldsymbol{n}_{pc}^{-}\otimes\boldsymbol{n}_{pc}^{-})(\boldsymbol{V}_{pc}^{-}-\boldsymbol{V}_c) + l_{pc}^{+}(\boldsymbol{n}_{pc}^{+}\otimes\boldsymbol{n}_{pc}^{-})(\boldsymbol{V}_{pc}^{+}-\boldsymbol{V}_c)\right], \\
&= \frac{1}{w_{pc}} l_{pc}^{+}\left[(\boldsymbol{n}_{pc}^{-}\otimes\boldsymbol{n}_{pc}^{+}) - (\boldsymbol{n}_{pc}^{+}\otimes\boldsymbol{n}_{pc}^{-})\right](\boldsymbol{V}_{pc}^{+}-\boldsymbol{V}_c).
\end{aligned}$$

Similarly, the half-edge flux $(\bar{\bar{\mathbb{D}}}\nabla \boldsymbol{V})_{pc}^{+}$ writes

$$\begin{aligned}
(\bar{\bar{\mathbb{D}}}\nabla \boldsymbol{V})_{pc}^{+} &= \operatorname{tr} \mathbb{G}_{pc}\boldsymbol{n}_{pc}^{+} - \mathbb{G}_{pc}^{t+}, \\
&= \frac{1}{w_{pc}}\left[l_{pc}^{-}(\boldsymbol{n}_{pc}^{+}\otimes\boldsymbol{n}_{pc}^{-})(\boldsymbol{V}_{pc}^{-}-\boldsymbol{V}_c) + l_{pc}^{+}(\boldsymbol{n}_{pc}^{+}\otimes\boldsymbol{n}_{pc}^{+})(\boldsymbol{V}_{pc}^{+}-\boldsymbol{V}_c)\right] \\
&- \frac{1}{w_{pc}}\left[l_{pc}^{-}(\boldsymbol{n}_{pc}^{-}\otimes\boldsymbol{n}_{pc}^{+})(\boldsymbol{V}_{pc}^{-}-\boldsymbol{V}_c) + l_{pc}^{+}(\boldsymbol{n}_{pc}^{+}\otimes\boldsymbol{n}_{pc}^{+})(\boldsymbol{V}_{pc}^{+}-\boldsymbol{V}_c)\right], \\
&= \frac{1}{w_{pc}} l_{pc}^{-}\left[(\boldsymbol{n}_{pc}^{+}\otimes\boldsymbol{n}_{pc}^{-}) - (\boldsymbol{n}_{pc}^{-}\otimes\boldsymbol{n}_{pc}^{+})\right](\boldsymbol{V}_{pc}^{-}-\boldsymbol{V}_c).
\end{aligned}$$

Combining these two terms and summing over all $p \in \mathcal{P}(c)$ yields

$$\begin{aligned}
\int_{\omega_c} \bar{\bar{\mathbb{D}}}\nabla \boldsymbol{V}\boldsymbol{n}\,\mathrm{d}s &= \sum_{p\in\mathcal{P}(c)} \frac{l_{pc}^{-}l_{pc}^{+}}{w_{pc}}\left[(\boldsymbol{n}_{pc}^{-}\otimes\boldsymbol{n}_{pc}^{+}) - (\boldsymbol{n}_{pc}^{+}\otimes\boldsymbol{n}_{pc}^{-})\right](\boldsymbol{V}_{pc}^{+}-\boldsymbol{V}_c) \\
&\quad + \frac{l_{pc}^{-}l_{pc}^{+}}{w_{pc}}\left[(\boldsymbol{n}_{pc}^{+}\otimes\boldsymbol{n}_{pc}^{-}) - (\boldsymbol{n}_{pc}^{-}\otimes\boldsymbol{n}_{pc}^{+})\right](\boldsymbol{V}_{pc}^{-}-\boldsymbol{V}_c), \\
&= \sum_{p\in\mathcal{P}(c)} \frac{l_{pc}^{-}l_{pc}^{+}}{w_{pc}}\left[(\boldsymbol{n}_{pc}^{-}\otimes\boldsymbol{n}_{pc}^{+}) - (\boldsymbol{n}_{pc}^{+}\otimes\boldsymbol{n}_{pc}^{-})\right](\boldsymbol{V}_{pc}^{+}-\boldsymbol{V}_{pc}^{-}).
\end{aligned}$$

Observing that $(\boldsymbol{b}\otimes\boldsymbol{a}-\boldsymbol{a}\otimes\boldsymbol{b})\boldsymbol{x} = (\boldsymbol{a}\times\boldsymbol{b})\times\boldsymbol{x}$ for all vectors $\boldsymbol{a}$, $\boldsymbol{b}$, $\boldsymbol{c}$ and $\boldsymbol{x}$ leads to transform the above equation into

$$\int_{\omega_c} \bar{\bar{\mathbb{D}}}\nabla \boldsymbol{V}\boldsymbol{n}ds = \sum_{p\in\mathcal{P}(c)} \frac{l_{pc}^{-}l_{pc}^{+}}{w_{pc}}(\boldsymbol{n}_{pc}^{+}\times\boldsymbol{n}_{pc}^{-})\times(\boldsymbol{V}_{pc}^{+}-\boldsymbol{V}_{pc}^{-}) \tag{2.99}$$

To be compatible with the continuous identity $\nabla\cdot(\bar{\bar{\mathbb{D}}}\nabla \boldsymbol{V}) = \boldsymbol{0}$, the right-hand side of the above equation should be equal to zero. We shall show that this is true provided that the volume weight is properly defined. To this end, we start by computing the right-hand side of (2.99). First, we observe that $\boldsymbol{n}_{pc}^{\pm}\times\boldsymbol{t}_{pc}^{\pm} = \boldsymbol{e}_z$ and $\boldsymbol{n}_{pc}^{\pm} = \boldsymbol{t}_{pc}\times\boldsymbol{e}_z$, where $\boldsymbol{t}_{pc}^{\pm}$ denotes the unit tangential vector related to $[p^{-},p]$, refer to Figure 2.7. Then, we develop the cross product of the unit outward normals as follows

$$\begin{aligned}
\boldsymbol{n}_{pc}^{+}\times\boldsymbol{n}_{pc}^{-} &= \boldsymbol{n}_{pc}^{+}\times(\boldsymbol{t}_{pc}^{-}\times\boldsymbol{e}_z), \\
&= (\boldsymbol{n}_{pc}^{+}\cdot\boldsymbol{e}_z)\boldsymbol{t}_{pc}^{-} - (\boldsymbol{n}_{pc}^{+}\cdot\boldsymbol{t}_{pc}^{-})\boldsymbol{e}_z, \\
&= -\left[(\boldsymbol{t}_{pc}^{+}\times\boldsymbol{e}_z)\cdot\boldsymbol{t}_{pc}^{-}\right]\boldsymbol{e}_z, \\
&= -\left[(\boldsymbol{t}_{pc}^{-}\times\boldsymbol{t}_{pc}^{+})\cdot\boldsymbol{e}_z\right]\boldsymbol{e}_z, \\
&= -\sin\theta_{pc}\boldsymbol{e}_z,
\end{aligned}$$

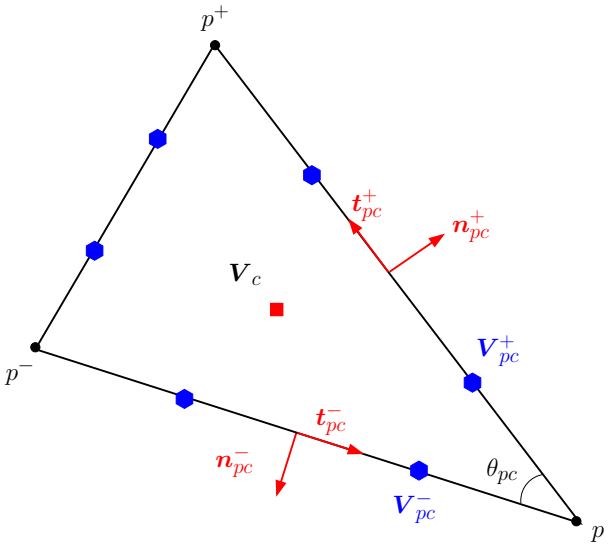


Figure 2.7: Notations in a triangular cell.

where θ_{pc} is the measure of the corner angle related to point p and cell c, refer to Figure 2.7. Substituting the above geometrical result into (2.99) yields

$$\begin{aligned}\int_{\omega_c} \bar{\bar{\mathbb{D}}}\nabla \boldsymbol{V}\boldsymbol{n}\mathrm{d}s &= \sum_{p\in\mathcal{P}(c)} \frac{l_{pc}^- l_{pc}^+}{w_{pc}} (\boldsymbol{n}_{pc}^+ \times \boldsymbol{n}_{pc}^-) \times (\boldsymbol{V}_{pc}^+ - \boldsymbol{V}_{pc}^-), \\ &= \sum_{p\in\mathcal{P}(c)} \frac{l_{pc}^- l_{pc}^+ \sin\theta_{pc}}{w_{pc}} (\boldsymbol{V}_{pc}^+ - \boldsymbol{V}_{pc}^-) \times \boldsymbol{e}_z.\end{aligned}$$

Due to the cyclic numbering of the half-edges we get $\sum_{p\in\mathcal{P}(c)} (\boldsymbol{V}_{pc}^+ - \boldsymbol{V}_{pc}^-) \times \boldsymbol{e}_z = \boldsymbol{0}$. **Thus, a sufficient condition to ensure the vanishing of the right-hand side of** (2.99) **consists in setting**

$$w_{pc} = \nu_c l_{pc}^- l_{pc}^+ \sin\theta_{pc}, \tag{2.100}$$

where ν_c is a non-negative constant, which uniquely depends on cell c. Moreover, this constant should be such that $\sum_{p\mathcal{P}(c)} w_{pc} =\mid \omega_c \mid$. We claim that the definitions of the volume weights (2.96) and (2.97) that we have proposed in the previous paragraph satisfy the sufficient condition (2.100). Indeed, for triangular and quadrangular cells we respectively get

- For triangular cells, the cell volume reads $|\omega_c| = 2 l_{pc}^- l_{pc}^+ \sin\theta_{pc}$, whereas the sub-cell volume is given by $w_{pc} = \frac{2}{3} l_{pc}^- l_{pc}^+ \sin\theta_{pc}$, thus the coefficient ν_c writes $\nu_c = \frac{2}{3}$ and is indeed a constant number over cell c.
- For quadrangular cells, the cell volume reads $|\omega_c| = \sum_{p\in\mathcal{P}(c)} l_{pc}^- l_{pc}^+ \sin\theta_{pc}$, whereas the sub-cell volume is given by $w_{pc} = l_{pc}^- l_{pc}^+ \sin\theta_{pc}$, thus the coefficient ν_c writes $\nu_c = 1$ and is also a constant number over cell c.

We conclude that the definitions (2.96) and (2.97) of the volume weights ensures that our Finite Volume discretization is compatible with the integral form of the tensorial identity

$$\int_{\partial\omega_c} \bar{\bar{\mathbb{D}}}\nabla \boldsymbol{V}\, \mathrm{d}s = \boldsymbol{0}.$$

This important compatibility property implies that the divergence of the viscous stress and the divergence of the pseudo-viscous stress coincide at the discrete level, *i.e.*, $\nabla \cdot \mathbb{S} = \nabla \cdot \mathbb{\Sigma}$.

2.8 Numerical results

This section aims at assessing the robustness and the accuracy of our numerical method against numerous representative test cases. Firstly, we describe the methodology employed to performed the convergence analysis. Then, we present the numerical tests and the grids used. We conclude by commenting the features of the numerical results obtained.

2.8.1 Methodology used for convergence analysis

In what follows, we solve the following generic tensorial diffusion equation

$$\rho \frac{\partial \boldsymbol{V}}{\partial t} - \nabla \cdot \mathbb{S} = \rho \boldsymbol{r}, \qquad \forall (\boldsymbol{x}, t) \in \mathcal{D} \times [0, \mathcal{T}], \tag{2.101a}$$

$$\boldsymbol{V}(\boldsymbol{x}, 0) = \boldsymbol{V}^0(\boldsymbol{x}), \qquad \forall \boldsymbol{x} \in \mathcal{D}, \tag{2.101b}$$

where $\boldsymbol{r} = \boldsymbol{r}(\boldsymbol{x})$ is a given source term. We are interested in steady analytical solutions, and thus we shall compute them starting with the initial condition $\boldsymbol{V}^0(\boldsymbol{x}) = \boldsymbol{0}$ and running the simulation until the steady state is reached. The density and the dynamic viscosity are defined by $\rho = 1$ and $\mu = 1$. The boundary conditions and the source term $\boldsymbol{r}$ are specified for each test case.
To assess the accuracy of the numerical method we introduce the L^2 and L^∞ discrete norms as follows. If $\boldsymbol{V}$ is a generic vector function its discrete L^2 norm writes

$$\|\boldsymbol{V}\|_2 = \sqrt{\sum_{c=1}^{\mathfrak{C}_\mathcal{D}} |\omega_c| \boldsymbol{V}_c{}^2},$$

whereas its discrete L^∞ norm is given by

$$\|\boldsymbol{V}\|_\infty = \max_{c=1 \ldots \mathfrak{C}_\mathcal{D}} |\boldsymbol{V}_c|.$$

For a given mesh composed of $\mathfrak{C}_\mathcal{D}$ cells we define the characteristic length h as

$$h = \left(\frac{|\ \mathcal{D}\ |}{\mathfrak{C}_\mathcal{D}} \right)^{\frac{1}{d}},$$

where $d = 1, 2, 3$ denotes the space dimension and $|\ \mathcal{D}\ |$ is the measure of the computational domain volume. To perform a convergence analysis of the numerical method we introduce the absolute errors between the analytical solution, $\boldsymbol{V}_e$, of (2.101a) and the numerical solution $\boldsymbol{V}_h$: the L^2 absolute error writes

$$E_2^h = \|\boldsymbol{V}_h - \boldsymbol{V}_e\|_2 ,$$

whereas its L^∞ counterpart writes

$$E_\infty^h = \|\boldsymbol{V}_h - \boldsymbol{V}_e\|_\infty .$$

Let us note that the computation of both errors are performed using the analytical solution evaluated at the cells centroid. The asymptotic error for both norms is estimated by

$$e_\alpha^h = C_\alpha h^{q_\alpha} + O(h^{q_\alpha + 1}), \text{ for } \alpha = 2, \infty,$$

where q_α denotes rate of convergence and C_α the convergence rate-constant which is independent of h. Having computed the absolute errors corresponding to two different grids characterized by mesh resolutions h_1 and $h_2 < h_1$, we deduce the estimation of the convergence rate as follows

$$q_\alpha = \frac{\log E_\alpha^{h_2} - \log E_\alpha^{h_1}}{\log h_2 - \log h_1}.$$

2.8.2 Meshes description

We perform the computations on different kind of meshes. For every mesh category, we employ 5 levels of refinement of the meshes from coarse grid to finer grids. A short description of the different grids used for the numerical tests is provided hereafter.

- A triangular grid is displayed in Figure 2.8(a). There are 5 grids, which are respectively composed of 242, 1054, 4262, 16818 and 67872 triangles.
- A Cartesian grid is displayed in Figure 2.8(b). The 5 grids are made of 10×10,20×20, 40×40, 80×80 and 160×160 squares.
- A smoothly deformed grid is displayed in Figure 2.8(c). The 5 five grids are composed of 10×10, 20×20, 40×40, 80×80 and 160×160 quadrangles. The deformation is characterized by the mapping of the unit square $[0,1]^2$ onto itself

$$\begin{cases} x(\xi,\eta) = \xi + 0.1\sin(2\pi\xi)\sin(2\pi\eta), \\ y(\xi,\eta) = \eta + 0.1\sin(2\pi\xi)\sin(2\pi\eta). \end{cases}$$

- A Kershaw type grid is displayed in Figure 2.8(e). The 5 grids are made of 12×12, 24×24, 48×48, 96×96 and 192×192 quadrangles.
- A randomly perturbed grid is displayed in Figure 2.8(d). The 5 grid are made of 10×10, 20×20, 40×40, 80×80 and 160×160 quadrangles. The random perturbation is characterized by the mapping defined on the unit square $[0,1]^2$ by:

$$\begin{cases} x(\xi,\eta) = \xi + 0.2hr_1, \\ y(\xi,\eta) = \eta + 0.2hr_2, \end{cases}$$

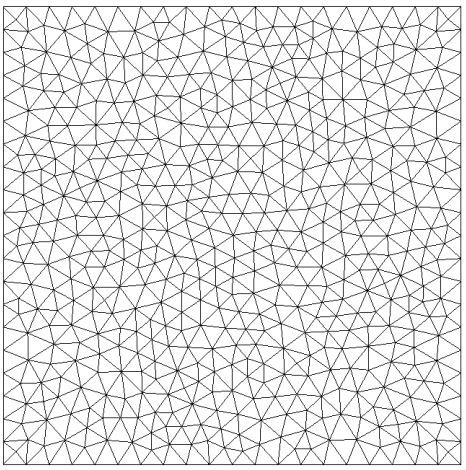
  where r_i for $i = 1,2$ are random numbers chosen in $]-1,1[$ and h is the characteristic mesh size.
- A hybrid grid made of triangles and quadrangles is displayed in Figure 2.8(f). The sequence of 5 hybrid grids contains respectively 132, 508, 2064, 8516 and 33796 triangles and 50, 200, 800, 3200 and 12800 quadrangles.

2.8.3 Convergence analysis for solutions characterized by a linear behavior with respect to the space variable

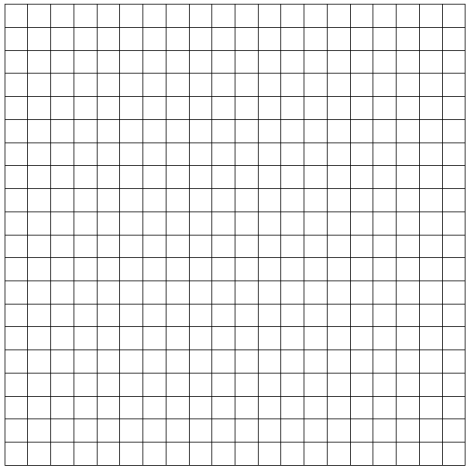
Here, we investigate the ability of our Finite Volume scheme to preserve a linear solution with respect to the space variable $\boldsymbol{x}$. The analytical steady solutions under consideration are defined by

$$\boldsymbol{V}_e(x,y) = \mathbb{A}\begin{pmatrix} x \\ y \end{pmatrix} + \boldsymbol{B}, \tag{2.102}$$

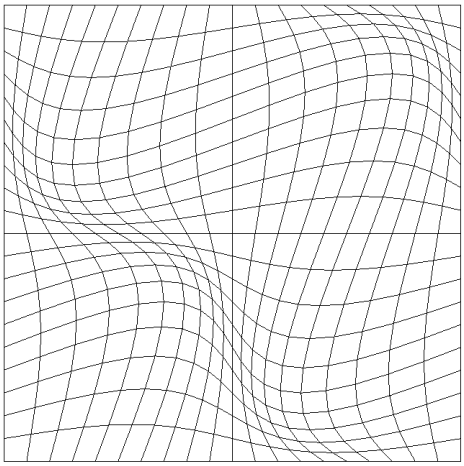
where $\boldsymbol{V}_e$ is the analytical solution, x and y are the Cartesian coordinates of the position vector $\boldsymbol{x}$, $\mathbb{A}$ is a 2×2 constant matrix and $\boldsymbol{B}$ is a constant vector of $\mathfrak{R}^2$. The boundary conditions is specified for each test case, namely it can be of kinematic or symmetric type depending on the definition of the matrix $\mathbb{A}$.



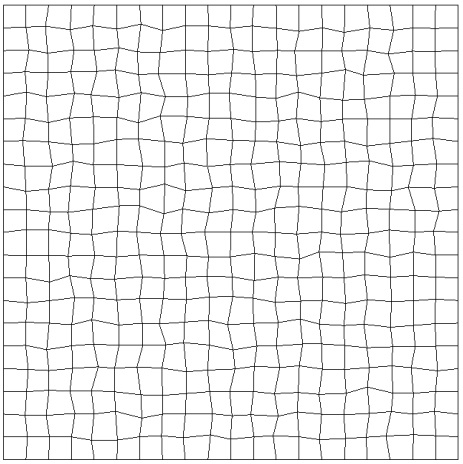
(a) Mesh made of 1054 triangles.



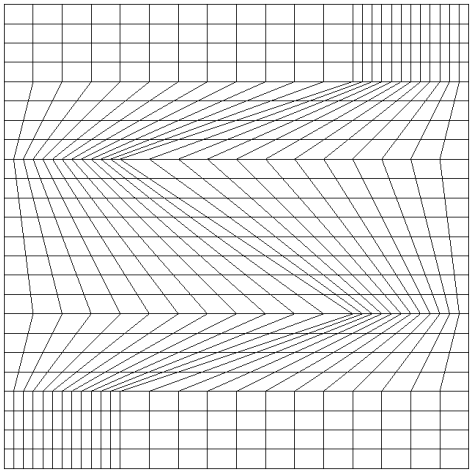
(b) Cartesian mesh made of 20×20 quadrangles.



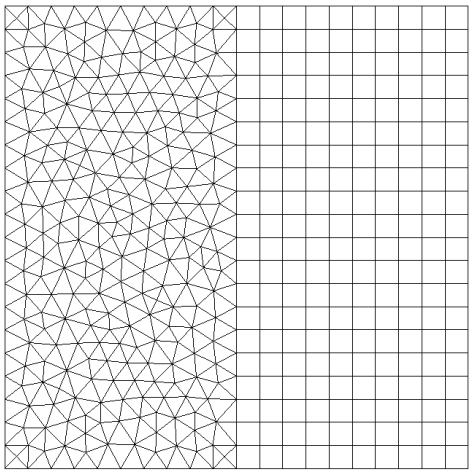
(c) Smoothly deformed mesh made of 20×20 quadrangles.



(d) Randomly perturbed mesh made of 20×20 quadrangles.



(e) Kershaw type mesh made of 24×24 quadrangles.



(f) Hybrid mesh made of 508 triangles and 10×20 quadrangles.

Figure 2.8: Example of the grids employed for the tests cases. For each category, we have displayed the grids related to the second level of refinement among the 5 levels available.

Linear velocity field with respect to x coordinate

Here, we set $\mathbb{A} = \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix}$ and $\boldsymbol{B} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$. Thus, the analytical solution $\boldsymbol{V}_e$ is a linear function with respect with the x coordinate, namely $\boldsymbol{V}(x, y) = x\, \boldsymbol{e}_x$. The computational domain is the unit square. Regarding the boundary conditions, they are of symmetric type at $y = 0$ and at $y = 1$ and of kinematic type elsewhere, that is $\boldsymbol{V}(0, y) = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$ and $\boldsymbol{V}(1, y) = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$. We have displayed the numerical solution in Figure 2.9. The convergence analysis is obtained by

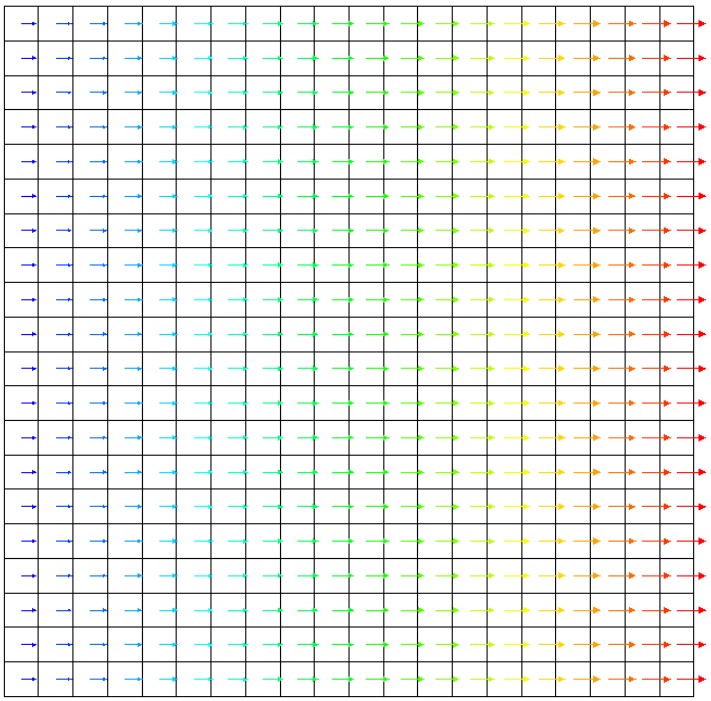


Figure 2.9: Linear velocity field with respect to x coordinate. Representation of the numerical solution on the 20×20 Cartesian grid.

looking at the numerical results displayed in Table 2.1. We observe that the numerical scheme is characterized by a second-order rate of convergence on smooth and Kershaw grids, whereas it exhibits an erratic behavior on randomly perturbed grids. It is worth mentioning that the numerical method captures exactly the analytical solution on the unstructured triangular grids.

Rigid rotation velocity field

For this test case we prescribe $\mathbb{A} = \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix}$ and $\boldsymbol{B} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$. The analytical solution, $\boldsymbol{V}_e$ corresponds a rigid rotation velocity field. The computational domain is still the unit square. The kinematic boundary condition $\boldsymbol{V}(x, y) = \begin{pmatrix} -y \\ x \end{pmatrix}$ is specified on all the boundaries of the computational domain. The numerical velocity field has been plotted in Figure 2.10. We observe that the velocity contours are circles centered at the origin of the Cartesian frame $(0, x, y)$. The numerical results dedicated to the convergence analysis of the numerical scheme on quadrangular grids are summarized in Table 2.2. They show that the numerical method has an almost second-order rate of convergence on smoothly distorted grids. On the other hand, on randomly distorted grids the convergence rate is almost of first-order. We have also assessed

Table 2.1: Linear velocity field with respect to x coordinate. Asymptotic errors in both L^2 and L^∞ norms and corresponding truncation error orders for quadrangular grids.

(a) Smooth grids.

h	E_{L2}	q_{L2}	E_∞	q_∞
1.00e-01	2.20e-03	**1.95**	4.54e-03	**1.78**
5.00e-02	5.69e-04	**1.99**	1.32e-03	**1.94**
2.50e-02	1.44e-04	**2.00**	3.44e-04	**1.98**
1.25e-02	3.60e-05	**2.00**	8.72e-05	**2.00**
6.25e-03	9.02e-06	-	2.19e-05	-

(b) Kershaw grids.

h	E_{L2}	q_{L2}	E_∞	q_∞
8.33e-02	3.99e-03	**1.86**	1.49e-02	**1.63**
4.17e-02	1.10e-03	**1.95**	4.80e-03	**1.84**
2.08e-02	2.85e-04	**1.98**	1.34e-03	**1.87**
1.04e-02	7.21e-05	**2.00**	3.65e-04	**1.88**
5.21e-03	1.81e-05	-	9.92e-05	-

(c) Random grids.

h	E_{L2}	q_{L2}	E_∞	q_∞
1.00e-01	1.21e-03	**0.79**	3.11e-03	**0.11**
5.00e-02	7.00e-04	**0.99**	2.87e-03	**1.17**
2.50e-02	3.52e-04	**0.97**	1.28e-03	**0.59**
1.25e-02	1.79e-04	**0.99**	8.53e-04	**1.27**
6.25e-03	9.03e-05	-	3.54e-04	-

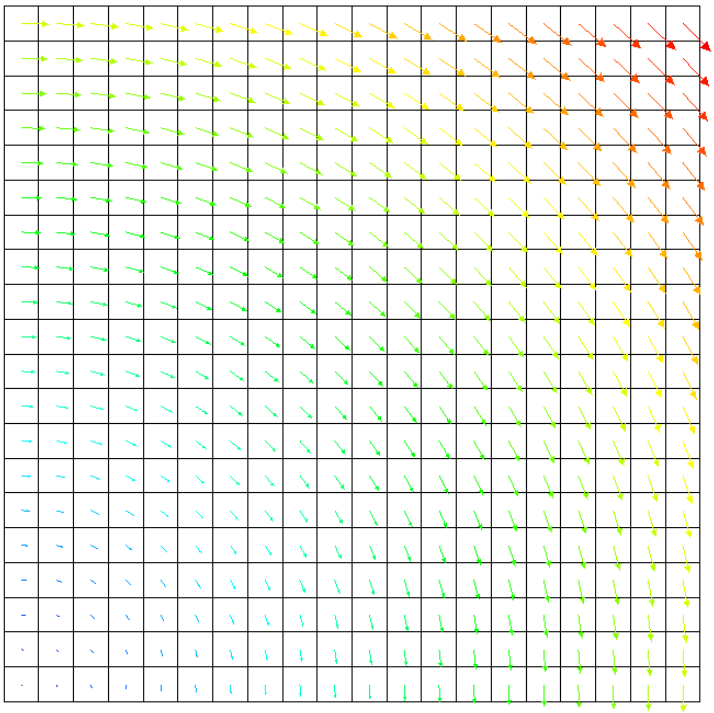


Figure 2.10: Rigid rotation velocity field. Representation of the solution on the 20×20 Cartesian grid.

Table 2.2: Rigid rotation velocity field. Asymptotic errors in both L^2 and L^∞ norms and corresponding truncation error orders for the different categories of quadrangular grids.

(a) Smooth grids.

h	E_{L2}	q_{L2}	E_∞	q_∞
1.00e-01	2.56e-03	**1.93**	3.91e-03	**1.68**
5.00e-02	6.73e-04	**1.97**	1.22e-03	**1.87**
2.50e-02	1.71e-04	**1.99**	3.35e-04	**1.96**
1.25e-02	4.31e-05	**2.00**	8.61e-05	**1.99**
6.25e-03	1.08e-05	-	2.17e-05	-

(b) Kershaw grids.

h	E_{L2}	q_{L2}	E_∞	q_∞
8.33e-02	4.56e-03	**1.85**	1.38e-02	**1.62**
4.17e-02	1.26e-03	**1.95**	4.48e-03	**1.79**
2.08e-02	3.25e-04	**1.98**	1.30e-03	**1.79**
1.04e-02	8.22e-05	**2.00**	3.74e-04	**1.91**
5.21e-03	2.06e-05	-	9.98e-05	-

(c) Random grids.

h	E_{L2}	q_{L2}	E_∞	q_∞
1.00e-01	1.54e-03	**0.86**	3.44e-03	**0.44**
5.00e-02	8.49e-04	**0.95**	2.53e-03	**1.10**
2.50e-02	4.39e-04	**0.97**	1.18e-03	**0.64**
1.25e-02	2.24e-04	**0.98**	7.59e-04	**1.06**
6.25e-03	1.14e-04	-	3.64e-04	-

Table 2.3: Rigid rotation velocity field. Asymptotic errors in both L^2 and L^∞ norms and corresponding truncation error orders for hybrid grids.

h	E_{L2}	q_{L2}	E_∞	q_∞
7.41e-02	3.05e-04	**1.50**	1.39e-03	**0.26**
3.76e-02	1.10e-04	**1.64**	1.16e-03	**0.94**
1.87e-02	3.51e-05	**1.81**	6.01e-04	**1.54**
9.24e-03	9.79e-06	**1.42**	2.02e-04	**0.34**
4.63e-03	3.68e-06	-	1.60e-04	-

the accuracy of the numerical methods on a sequence of 5 hybrid grids composed of triangular and quadrangular cells. In this particular case, the rate of convergence is located between first and second-order as it is shown in Table 2.3

Stretching velocity field

Here, we set $\mathbb{A} = \begin{pmatrix} 3 & 0 \\ 0 & 2 \end{pmatrix}$ and $\boldsymbol{B} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$. The analytical solution, $\boldsymbol{V}_e$, corresponds a to a stretching velocity field. The computational domain is once more the unit square. Since x and y axis are symmetry axis for the solution, we apply symmetry boundary conditions at $x = 0$ and at $y = 0$. We specify the kinematic boundary condition, $\boldsymbol{V}(x, y) = \begin{pmatrix} 3x \\ 2y \end{pmatrix}$, at $x = 1$ and $y = 1$. The analytical solution of this problem has been plotted in Figure 2.11. We observe that the deformation rate tensor related to this velocity field is equal to $\mathbb{A}$. The eigenvalues are 3 and 2 respectively associated to the eigenvectors $\boldsymbol{e}_x$ and $\boldsymbol{e}_{\boldsymbol{y}}$. Thus, the coordinate axis are the principal directions of the tensor of deformation rate. The results of the convergence analysis

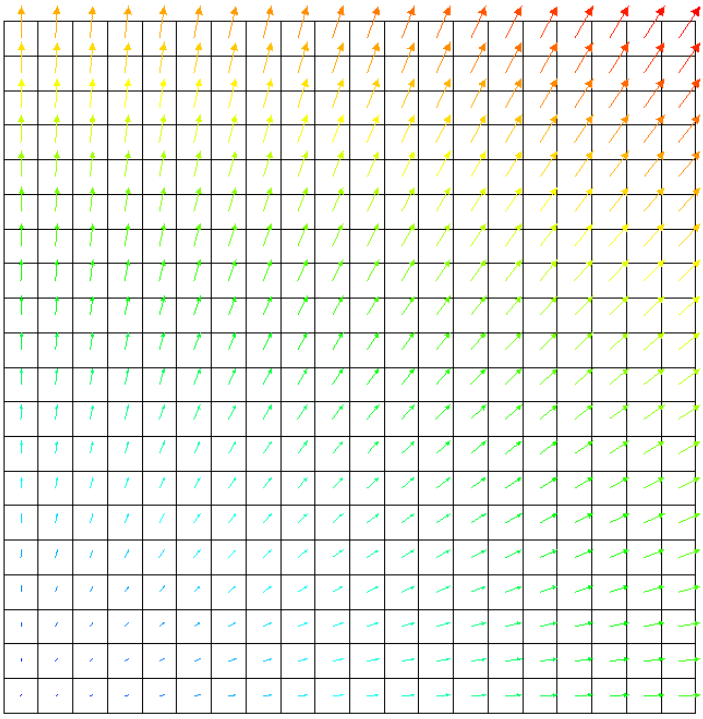


Figure 2.11: Stretching velocity field. Representation of the solution on the 20×20 Cartesian grid.

performed on quadrangular grids are displayed in Table 2.4. Once more, the numerical scheme is characterized by a second-order rate of convergence on smooth grids, whereas the order of convergence is almost one on randomly perturbed grids. We also note that the convergence rate in L^∞ norm for the random grids, refer to Table 2.4(c), is rather erratic. Regarding the hybrid grids, refer to Table 2.5, we observe a convergence rate which is between first and second-order.

General comments for the linear tests

For all of the linear tests conducted on the triangular and on the Cartesian grids, we point out that we get round-off error. This fact is consistent with the theoretical result that the

Table 2.4: Stretching velocity field. Asymptotic errors in both L^2 and L^∞ norms and corresponding truncation error orders for quadrangular grids.

(a) Smooth grids.

h	E_{L2}	q_{L2}	E_∞	q_∞
1.00e-01	7.16e-03	**1.96**	1.17e-02	**1.80**
5.00e-02	1.84e-03	**1.99**	3.36e-03	**1.96**
2.50e-02	4.62e-04	**2.00**	8.64e-04	**1.98**
1.25e-02	1.16e-04	**2.00**	2.18e-04	**2.00**
6.25e-03	2.89e-05	-	5.47e-05	-

(b) Kershaw grids.

h	E_{L2}	q_{L2}	E_∞	q_∞
8.33e-02	1.01e-02	**1.90**	2.97e-02	**1.63**
4.17e-02	2.71e-03	**1.96**	9.59e-03	**1.84**
2.08e-02	6.94e-04	**1.99**	2.68e-03	**1.87**
1.04e-02	1.75e-04	**2.00**	7.30e-04	**1.88**
5.21e-03	4.38e-05	-	1.98e-04	-

(c) Random grids.

h	E_{L2}	q_{L2}	E_∞	q_∞
1.00e-01	3.94e-03	**0.88**	8.04e-03	**0.33**
5.00e-02	2.14e-03	**0.95**	6.40e-03	**1.08**
2.50e-02	1.11e-03	**0.98**	3.03e-03	**0.68**
1.25e-02	5.63e-04	**0.98**	1.90e-03	**1.07**
6.25e-03	2.85e-04	-	9.03e-04	-

Table 2.5: Stretching velocity field. Asymptotic errors in both L^2 and L^∞ norms and corresponding truncation error orders for hybrid grids.

h	E_{L2}	q_{L2}	E_∞	q_∞
7.41e-02	2.46e-04	**1.15**	2.01e-03	**0.26**
3.76e-02	1.13e-04	**1.55**	1.69e-03	**0.94**
1.87e-02	3.83e-05	**2.03**	8.74e-04	**1.60**
9.24e-03	9.14e-06	**1.20**	2.83e-04	**0.29**
4.63e-03	3.99e-06	-	2.32e-04	-

Table 2.6: Vortex-like velocity field. Asymptotic errors in both L^2 and L^∞ norms and corresponding truncation error orders for triangular grids.

h	E_{L2}	q_{L2}	E_∞	q_∞
6.43e-02	3.60e-03	**1.94**	4.95e-03	**1.89**
3.08e-02	8.62e-04	**1.99**	1.23e-03	**1.71**
1.53e-02	2.15e-04	**2.00**	3.72e-04	**2.07**
7.71e-03	5.42e-05	**2.01**	9.01e-05	**1.91**
3.84e-03	1.34e-05	-	2.38e-05	-

numerical method is able to preserve linear solutions provided that the volume weights are properly chosen, refer to Section 2.7.2.
All these linear tests are characterized by quadratic convergence rate on both smoothly deformed and Kershaw meshes. For the randomly deformed meshes and the hybrid meshes it is harder to draw conclusions. Let us point out that the randomly perturbed grids are scarcely encountered in real life applications. Regarding the hybrid grids, we get a round-off error for the linear velocity field with respect to x coordinate, and observe convergence rates between first and second-order for the other tests. While for all the other types of grids the 5 levels of refinement were obtained in a continuous manner by multiplying the number of cells in x and y directions, it is not the case anymore for the random and the hybrid grids. Indeed, the refined versions of the random grids are made of smaller cells but due to the random process used to generate them there is no continuous relations linking two different levels of refinement together. For the hybrid grids, the Cartesian part undergoes a continuous refinement, whereas it is not the case for the triangular part. This is due to the fact that the triangles are generated using a Delaunay algorithm. This non-uniformity in the grid generation may explain the poor rate of convergence obtained on hybrid grids.

2.8.4 Convergence analysis for solutions characterized by a non-linear behavior with respect to the space variable

Vortex-like velocity field

In this test problem we consider the analytical solution generated by the velocity field

$$\boldsymbol{V}_e = \begin{pmatrix} \left(y - \frac{1}{2}\right) \sin(\pi x) \sin(\pi y) \\ \left(\frac{1}{2} - x\right) \sin(\pi x) \sin(\pi y) \end{pmatrix}$$

This analytical solution corresponds to a vortex, which is obtained by specifying the source term, $\boldsymbol{r}$, as

$$\boldsymbol{r} = \begin{pmatrix} -\frac{7}{3}\left(y - \frac{1}{2}\right)\pi^2 \sin(\pi x) \sin(\pi y) + \frac{5}{3}\pi \sin(\pi x) \cos(\pi y) + \frac{1}{3}\left(\frac{1}{2} - x\right)\pi^2 \cos(\pi x) \cos(\pi y) \\ -\frac{7}{3}\left(\frac{1}{2} - x\right)\pi^2 \sin(\pi x) \sin(\pi y) - \frac{5}{3}\pi \cos(\pi x) \sin(\pi y) + \frac{1}{3}\left(y - \frac{1}{2}\right)\pi^2 \cos(\pi x) \cos(\pi y) \end{pmatrix}$$

We prescribe the kinematic boundary condition $\boldsymbol{V}(x, y) = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$ at all the boundaries of the computational domain, which is the unit square. The corresponding numerical solution has been plotted on a 20×20 Cartesian grid in Figure 2.12. We observe that the velocity contours are circles centered at $(\frac{1}{2}, \frac{1}{2})$. We run this test problem on a sequence of 5 triangular grids. The corresponding numerical errors are displayed in Table 2.6. We observe that the numerical method exhibits a second-order rate of convergence on this type of grids. The numerical results obtained on the different categories of quadrangular grids are displayed in Table 2.7. Once more

Table 2.7: Vortex-like velocity field. Asymptotic errors in both L^2 and L^∞ norms and corresponding truncation error orders for quadrangular grids.

(a) Cartesian grids.

h	E_{L2}	q_{L2}	E_∞	q_∞
1.00e-01	5.59e-03	**2.00**	7.20e-03	**1.93**
5.00e-02	1.40e-03	**2.00**	1.88e-03	**1.97**
2.50e-02	3.50e-04	**2.00**	4.81e-04	**1.99**
1.25e-02	8.75e-05	**2.00**	1.22e-04	**1.99**
6.25e-03	2.19e-05	-	3.05e-05	-

(b) Smooth grids.

h	E_{L2}	q_{L2}	E_∞	q_∞
1.00e-01	5.20e-03	**1.92**	1.10e-02	**1.84**
5.00e-02	1.37e-03	**1.97**	3.08e-03	**1.80**
2.50e-02	3.51e-04	**1.99**	8.84e-04	**1.83**
1.25e-02	8.84e-05	**2.00**	2.48e-04	**1.92**
6.25e-03	2.21e-05	-	6.55e-05	-

(c) Kershaw grids.

h	E_{L2}	q_{L2}	E_∞	q_∞
8.33e-02	5.88e-03	**1.76**	1.39e-02	**1.72**
4.17e-02	1.74e-03	**1.91**	4.24e-03	**1.75**
2.08e-02	4.61e-04	**1.97**	1.26e-03	**1.81**
1.04e-02	1.18e-04	**1.99**	3.61e-04	**1.90**
5.21e-03	2.98e-05	-	9.69e-05	-

(d) Random grids.

h	E_{L2}	q_{L2}	E_∞	q_∞
1.00e-01	5.51e-03	**2.20**	9.33e-03	**1.63**
5.00e-02	1.20e-03	**1.63**	3.02e-03	**0.84**
2.50e-02	3.87e-04	**-0.09**	1.69e-03	**1.04**
1.25e-02	4.12e-04	**-0.16**	8.23e-04	**0.14**
6.25e-03	4.61e-04	-	7.47e-04	-

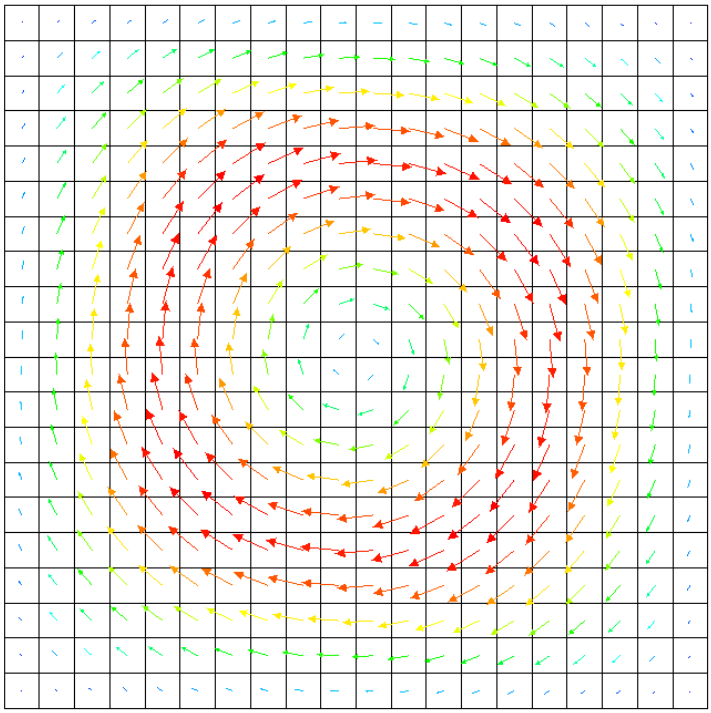


Figure 2.12: Vortex-like velocity field. Representation of the solution on a 20×20 Cartesian grid.

Table 2.8: Vortex-like velocity field. Asymptotic errors in both L^2 and L^∞ norms and corresponding truncation error orders for hybrid grids.

h	E_{L2}	q_{L2}	E_∞	q_∞
7.41e-02	4.22e-03	**2.12**	7.15e-03	**1.97**
3.76e-02	9.99e-04	**2.05**	1.87e-03	**1.15**
1.87e-02	2.39e-04	**1.76**	8.40e-04	**1.52**
9.24e-03	6.93e-05	**1.12**	2.88e-04	**0.29**
4.63e-03	3.20e-05	-	2.36e-04	-

the Cartesian, smooth and Kershaw grids are characterized by an almost second-order rate of convergence whereas the randomly perturbed grids exhibit an erratic rate of convergence. Finally, on hybrid grids, refer to Table 2.8, we observe a rate of convergence which is located between first and second-order.

Boundary layer-like velocity field

Here, we consider the test problem characterized by the analytical velocity field

$$\boldsymbol{V}_e = \begin{pmatrix} 1 - \frac{\exp(\lambda(1-y))}{\exp(\lambda)} \\ 0 \end{pmatrix},$$

where λ is a non-negative real valued parameter. This analytical solution is obtained by specifying the source term, $\boldsymbol{r}$, as follows

$$\boldsymbol{r} = \begin{pmatrix} \lambda^2 \frac{\exp(\lambda(1-y))}{\exp(\lambda)} \\ 0 \end{pmatrix}.$$

Table 2.9: Boundary layer-like field. Asymptotic errors in both L^2 and L^∞ norms and corresponding truncation error orders for boundary layer grids.

h	E_{L2}	q_{L2}	E_∞	q_∞
5.00e-02	1.04e-03	**1.99**	3.40e-03	**1.91**
2.50e-02	2.61e-04	**2.00**	9.00e-04	**1.96**
1.25e-02	6.54e-05	**2.00**	2.32e-04	**1.98**
6.25e-03	1.64e-05	-	5.87e-05	-

We prescribe the kinematic boundary condition $\boldsymbol{V}(x,y) = \boldsymbol{V}_e(x,y)$ at all the boundaries of the computational domain, which is still the unit square. The parameter λ is set $\lambda = 10$. Let us note that the x-component of the velocity field is maximum and equal to one at the top boundary ($y = 1$), whereas it decreases exponentially to zero at the bottom boundary ($y = 0$). The λ parameter is a stretching parameter, which characterizes the exponential decaying of the x-component of the velocity field when approaching the bottom boundary. This test problem allows to mimic a boundary layer problem, which is representative of viscous flow in presence of a solid wall. The corresponding numerical solution has been plotted in Figure 2.13 employing a 20×20 stretched Cartesian grid. This Cartesian stretched grid has been refined near the axis

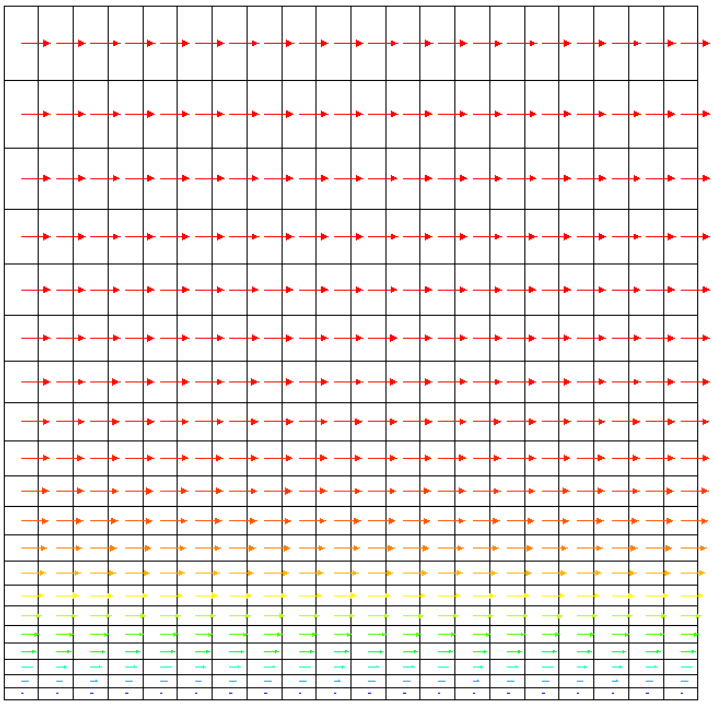


Figure 2.13: Boundary layer-like velocity field. Representation of the solution on a 20×20 stretched Cartesian grid.

$y = 0$ where the solution varies the most. This kind of mesh is classically used when dealing with viscous flow problems in the presence of a solid boundary. We have displayed in Table 2.9 the numerical results obtained using a sequence of stretched Cartesian grids in y direction. They show a second-order rate of convergence for our numerical method. We also perform the convergence analysis on a sequence of 5 unstructured triangular grids and we observe a second-order convergence rate in Table 2.10. The results obtained for the convergence analysis made for the different categories of quadrangular grids are displayed in Table 2.11. Once more, we

Table 2.10: Boundary layer-like field. Asymptotic errors in both L^2 and L^∞ norms and corresponding truncation error orders for triangular grids.

h	E_{L2}	q_{L2}	E_∞	q_∞
6.43e-02	1.47e-02	**1.80**	5.47e-02	**1.61**
3.08e-02	3.91e-03	**2.13**	1.68e-02	**1.96**
1.53e-02	8.86e-04	**1.93**	4.28e-03	**1.88**
7.71e-03	2.36e-04	**1.99**	1.18e-03	**1.89**
3.84e-03	5.90e-05	-	3.15e-04	-

Table 2.11: Boundary layer-like field. Asymptotic errors in both L^2 and L^∞ norms and corresponding truncation error orders for quadrangular grids.

(a) Cartesian grids.

h	E_{L2}	q_{L2}	E_∞	q_∞
1.00e-01	2.55e-02	**1.85**	8.07e-02	**1.66**
5.00e-02	7.08e-03	**1.96**	2.55e-02	**1.85**
2.50e-02	1.82e-03	**1.99**	7.08e-03	**1.93**
1.25e-02	4.59e-04	**2.00**	1.86e-03	**1.96**
6.25e-03	1.15e-04	-	4.77e-04	-

(b) Smooth grids.

h	E_{L2}	q_{L2}	E_∞	q_∞
1.00e-01	2.72e-02	**1.58**	1.11e-01	**1.22**
5.00e-02	9.07e-03	**1.87**	4.75e-02	**1.57**
2.50e-02	2.48e-03	**1.97**	1.60e-02	**1.80**
1.25e-02	6.34e-04	**1.99**	4.58e-03	**1.91**
6.25e-03	1.59e-04	-	1.22e-03	-

(c) Kershaw grids.

h	E_{L2}	q_{L2}	E_∞	q_∞
8.33e-02	1.87e-02	**1.89**	6.10e-02	**1.73**
4.17e-02	5.03e-03	**1.97**	1.84e-02	**1.88**
2.08e-02	1.29e-03	**1.99**	5.01e-03	**1.94**
1.04e-02	3.23e-04	**2.00**	1.30e-03	**1.97**
5.21e-03	8.09e-05	-	3.32e-04	-

(d) Random grids.

h	E_{L2}	q_{L2}	E_∞	q_∞
1.00e-01	2.26e-02	**1.76**	8.00e-02	**1.37**
5.00e-02	6.69e-03	**2.05**	3.09e-02	**1.47**
2.50e-02	1.61e-03	**1.27**	1.11e-02	**0.99**
1.25e-02	6.68e-04	**-0.16**	5.62e-03	**1.09**
6.25e-03	7.45e-04	-	2.63e-03	-

Table 2.12: Boundary layer-like field. Asymptotic errors in both L^2 and L^∞ norms and corresponding truncation error orders for hybrid grids.

h	E_{L2}	q_{L2}	E_∞	q_∞
7.41e-02	1.83e-02	**1.86**	7.37e-02	**1.61**
3.76e-02	5.20e-03	**2.00**	2.47e-02	**1.81**
1.87e-02	1.29e-03	**2.06**	6.97e-03	**1.69**
9.24e-03	3.01e-04	**1.79**	2.12e-03	**1.34**
4.63e-03	8.77e-05	-	8.42e-04	-

draw the same conclusions than before: the numerical scheme is characterized by an almost second-order rate of convergence on Cartesian, smooth and Kershaw grids, whereas it shows an erratic behavior on randomly distorted grids, refer to Table 2.11(d). Finally, we conclude this section, by running this test problem on a sequence of 5 hybrid grids composed of triangular and quadrangular cells. The convergence analysis results are summarized in Table 2.12. They show a convergence rate located between first and second-order.

2.9 Conclusion

In this chapter we have constructed an original compatible cell-centered Finite Volume method for solving generic tensorial diffusion equations, knowing that this generic equation is nothing but the momentum equation of Navier-Stokes equations without the pressure gradient term, refer to Chapter 3. Adapting the seminal idea of Arnold [16], we have modified the constitutive law, which defines the deviatoric part of the Cauchy stress tensor in terms of the deviatoric part of the strain rate tensor, by adding a supplementary term, which renders it invertible. This amounts to define a pseudo-viscous stress tensor, which is characterized by an invertible constitutive law over the space of second-order tensors. The divergence free feature of the aforementioned extra term ensures the equivalence of the formulations of the tensorial diffusion equation either based on the viscous stress or based on the pseudo viscous stress. A sub-cell-based variational formulation of the invertible constitutive law allows us to construct a numerical approximation of the half-edge pseudo-viscous fluxes in terms of the half-edge velocities and the cell-centered velocity at a cell corner. The half-edge velocities are extra degrees of freedom which are eliminated by invoking the half-edge viscous fluxes continuity at cell interfaces. This elimination procedure consists in solving invertible small linear systems located at each node of the computational grid. The construction of the global diffusion matrix provides a global linear system which is characterized by a positive definite matrix. The corresponding numerical scheme is characterized by a L^2 stability property, it preserves linear solutions on triangular grids and achieves second-order convergence rate on smoothly deformed meshes. We have also showed numerically that it is well suited to capture boundary layer-like solutions that we will need to handle when dealing with Navier-Stokes equations. Although we have given a two-dimensional presentation of this scheme, we claim that its three-dimensional extension may be easily developed using the notation changes presented for the CCLAD scheme in Chapter 1.

Chapter 3

A Finite Volume scheme for solving Fluid Dynamics on unstructured grids

We start this chapter by a brief overview of the main numerical methods used in the resolution of fluid dynamics problems. The interested reader should refer to [135] for a more detailed review of these methods.

Finite Volume Method

The Finite Volume schemes [82, 45] are certainly the most mature and the most documented numerical methods for fluid dynamics. To this day they are present at the core of most of the industrial codes devoted to Computational Fluid Dynamics (CFD). These methods can be developed on structured and unstructured meshes, and we can distinguish two kinds of approaches, the vertex-centered and the cell-centered approach. These methods differ in the definition of the control volume and its interpretation in regards to the mesh. In the cell-centered approach, the control volumes are defined as the cells of the mesh. In the vertex-centered approach the unknowns of the problem are located at the nodes of the mesh, the control volumes are defined as a volume in the vicinity of the nodes and correspond to the cells of the dual mesh. For the discretization of the inviscid fluxes the two approaches are equivalent. The idea is to solve a Riemann problem at each interface of the mesh [53]. The main difference is obtained for the development of the viscous fluxes. The classical methodology used with the vertex-centered approach is to use an hybrid Finite Volume-Finite Element scheme [46]. The gradients of the unknowns are obtained with a Finite Element method defined on each cell, and are then used in the Finite Volume scheme through the use of Green formulas. For the cell-centered approach a wide range of methods are used for the discretization of the viscous terms [41]. We can cite for example the Coirier's diamond path reconstruction [132, 126] or the weighted least squares methods. High-order can be achieved with Finite Volume Methods using the ENO [58] or WENO [87, 75] methodology on structured grids, or the MUSCL [128] methodology on unstructured grids for instance.

Finite Element Method

The Finite Element method (FE) is widely used in the domain of structural analysis, it can be also applied to CFD problems. In the Finite Element method the discrete solution is expressed as a linear combination of basis functions which are continuous piecewise polynomials. High-order versions of the FE method are easily obtained by increasing the degree of the polynomial basis

functions. The basic FE formulation brings to the well known Galerkin scheme, which is however not stable for wave equations and a stabilization mechanism must be added. Many types of stabilizing techniques have been developed to cure the stability issue. We can cite for instance the streamline upwind Petrov-Galerkin (SUPG) method [70, 136, 30], the Galerkin/least squares approach [71] or the Taylor Galerkin method [88]. Moreover, when discontinuities are present in the solution an additional shock capturing term must be added in order to guarantee the monotonicity of the solution. Unfortunately, Venkatakrishnan et al. [131] noted that no shock-capturing terms can guarantee stability for SUPG in the general case.

Discontinuous Galerkin Method

The Discontinuous Galerkin (DG) method combines features of the FV and FE methods. As in the FE method, the DG method is based on the Galerkin formulation of the governing equations. Here, the solution is assumed to be discontinuous between two adjacent elements [116]. The discontinuities of the solution at the faces of the elements are taking into account by numerical fluxes, usually under the form of Riemann solvers, as what is done in the FV approach. For a review of the DG method, see [39, 31].
Higher-order solution is obtained by using high-order polynomials within elements. Due to the local definition of the polynomials the DG schemes are extremely compact and flexible. These features makes the DG method ideally suited for parallel computations. Impressive results have been already obtained for the Euler equation [24] and the compressible Navier-Stokes equations [23]. One of the major drawbacks of these approaches is the computational cost induced by the huge amount of degrees of freedom obtained when dealing with high-order polynomials. This computational cost grows a lot faster than the cost of classical FE methods when using the same polynomials. Moreover, in the presence of discontinuities, the high order polynomial approximation produces spurious oscillations in the numerical solution and thus reduce the benefit of using a high-order method. Once again some stabilization technique is required to prevent these oscillations. This can be done by the introduction of artificial viscosity terms [60] or with the use of slope limiters [34].

Residual Distribution Scheme

The Residual Distribution Scheme (RDS) is a class of method that uses a continuous representation of the variables, similarly to the standard Finite Element methods. It has been first studied by Roe in [109] under the name of Fluctuation Splitting method. The method defines the residual, which is an integral quantity over each element, representing the balance of information entering the element. Following well defined rules [8], this residual is then distributed to the variables defining this element. An impressive bibliography on these methods is available in Ricchiuto's PhD and HDR thesis [105, 106]. Larat in his PhD thesis [79] developed high-order RDS schemes in two and three dimensions of space. He showed that this method is suitable to solve Navier-Stokes equations and is able to solve industrial problems in three dimensional geometry with the help of an efficient parallelization [9, 10]. Later De Santis [111, 12, 11] improved the discretization of the viscous terms and the boundary conditions treatment. He also extended the formulation to the Reynolds Averaged Navier-Stokes (RANS) equations. This impressive work makes this kind of schemes a serious candidate to become the cornerstone in the construction of the future generation of industrial high-order CFD codes. It is worth mentioning that the methodology has been also successfully applied to other equations such as those of Magneto-Hydro-Dynamics (MHD) [69].

In the following, after describing the construction and properties of the equations of fluid dynamics, we present a cell-centered Finite Volume scheme on unstructured meshes for solving the Navier-Stokes equations. We start by describing the discretization of the inviscid fluxes by building a scheme for the resolution of the Euler equations. It requires the use of approximate Riemann solvers. We then discuss the high-order extension of this scheme with the description of the classical MUSCL method. We continue the description of our scheme by the presentation of an explicit version of the time discretization. Since we need to solve the equations to steady state we observe that, the time step limitation of the explicit method is a barrier to obtain a sufficient convergence rate. This time step limitation is a consequence of the very small cell sizes needed to capture the thin boundary layers present in the supersonic regime. That leads us to develop an implicit version of the scheme. We then use this scheme as a starting point to the construction of a numerical scheme for solving the Navier-Stokes equation. The novelty in our approach is that we use the numerical schemes developed in Chapter 1 and Chapter 2 in order to discretize the diffusive terms of the equations. Finally, we present various numerical tests cases in order to assess the robustness and the accuracy of our numerical method.

3.1 Governing Equations of Fluid Dynamics

The aim of this section is to recall briefly the Fluid Dynamics equations. These equations express simply the classical principles of conservation of mass, momentum and total energy which relies on a continuum description of the fluid flow. This means that the fluid is viewed as continuum medium and the length scale at which it is studied is very large with respect to the mean free path of the particles it contains. Let us recall that the mean free path is the average distance traveled by a moving particle between successive collisions, which modify its direction or energy or other particle properties.

The physical derivation of the conservation laws of Fluid Dynamics is beyond the scope of the present work. It is the main subject of many classical textbooks [55, 117, 49, 118, 110, 112, 77, 138], which the interested reader may refer to. We want also to mention the book of Chorin and Marsden [38]. Let us note that an interesting presentation of these equations, with a focus on numerical methods, is given in the book of Toro [120]. The interested reader may also consult [120] and [67, 66] to discover the wide spectrum of numerical methods devoted to Computational Fluid Dynamics. The remainder of this section is organized as follows. Firstly, we recall the main properties and the various forms of the Fluid Dynamics equations. Secondly, we recall the constitutive laws and the thermodynamic closure which lead to the well known compressible Navier-Stokes. Then, we simplify this model and recall the compressible Euler equations dedicated to the modeling of inviscid non heat conducting fluids. Let us also mention the interesting work of Caltagirone about a new formulation of the conservation laws of continuum mechanics [36].

3.1.1 Conservation laws of Fluid Dynamics

The conservation laws of Fluid Dynamics consist of a set of partial differential equations which describe respectively the conservation of mass, momentum and total energy of a fluid particle. Before writing these equations, let us introduce the primarily physical variables which are the mass density ρ, the velocity field $\boldsymbol{V}$ and the specific total energy E. All these variables depend on the vector position, $\boldsymbol{x}$, of the fluid particle and also on time t. These physical quantities are

governed by the following set of equations

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \boldsymbol{V}) = 0, \tag{3.1a}$$

$$\frac{\partial}{\partial t}(\rho \boldsymbol{V}) + \nabla \cdot (\rho \boldsymbol{V} \otimes \boldsymbol{V}) = \nabla \cdot \mathbb{T}, \tag{3.1b}$$

$$\frac{\partial}{\partial t}(\rho E) + \nabla \cdot (\rho E \boldsymbol{V}) = \nabla \cdot (\mathbb{T}\boldsymbol{V}) - \nabla \cdot \boldsymbol{q}. \tag{3.1c}$$

Here, $\mathbb{T}$ stands for the Cauchy stress tensor and $\boldsymbol{q}$ is the heat flux vector. It is worth mentioning that the Cauchy stress tensor is symmetric, *i.e.*, $\mathbb{T}^t = \mathbb{T}$, to ensure the conservation of angular momentum, refer to [55]. Let us note that in the above system we do not take into account body forces such as gravity neither heat source term since they are useless for the type of applications under consideration (aerodynamics). Decomposing the specific total energy into specific internal energy, e, and specific kinetic energy, $\frac{1}{2} \mid \boldsymbol{V} \mid^2$, leads to $E = e + \frac{1}{2} \mid \boldsymbol{V} \mid^2$.
We remark that the above equations are derived consistently with fundamental laws of mechanics and thermodynamics. Namely, (3.1a) states that the mass of fluid enclosed in a region moving with the fluid velocity is conserved. The momentum equation (3.1b) is nothing but the second Newton law applied to a co-moving fluid region, which states that the time rate of change of momentum is balanced by the sum of the external forces exerted on the fluid domain. Here, since we have ignored the body forces it only remains the surface forces expressed by means of the Cauchy stress tensor. More precisely, if we consider an element of fluid surface, $\mathrm{d}s$ characterized by its unit normal, $\boldsymbol{n}$, then the element surface force, $\mathrm{d}\boldsymbol{f}$, exerted onto this surface writes $\mathrm{d}\boldsymbol{f} = \mathbb{T}\boldsymbol{n}\,\mathrm{d}s$ according to Cauchy theorem [55]. Finally, the total energy equation (3.1c) is a direct consequence of the First Law of Thermodynamics, which states that the total energy of a co-moving fluid region is the sum of the power of the external forces exerted supplemented by the heat rate supplied to the fluid region.
System (3.1) represents the conservative local form of the Fluid Dynamics equations. We also introduce its local non-conservative form as

$$\rho \frac{d}{dt}(\frac{1}{\rho}) - \nabla \cdot \boldsymbol{V} = 0, \tag{3.2a}$$

$$\rho \frac{d}{dt}\boldsymbol{V} - \nabla \cdot \mathbb{T} = \boldsymbol{0}, \tag{3.2b}$$

$$\rho \frac{d}{dt}E - \nabla \cdot (\mathbb{T}\boldsymbol{V}) + \nabla \cdot \boldsymbol{q} = 0. \tag{3.2c}$$

Here, $\frac{d}{dt}() = \frac{\partial}{\partial t}() + \boldsymbol{V} \cdot \nabla()$ denotes the material derivative, that is the time derivative following the fluid particles paths. We point out that the two above formulations, (3.1) and (3.2) are strictly equivalent provided that the smoothness of the fluid variables is granted, refer to [55]. Using the above non-conservative form, it is useful to construct the time rate of change of the specific internal energy. To this end, we first derive the time rate of change of kinetic energy dot-multiplying the momentum equation by $\boldsymbol{V}$

$$\rho \frac{d}{dt}(\frac{1}{2}\boldsymbol{V}^2) + \boldsymbol{V} \cdot \nabla \cdot \mathbb{T} = 0. \tag{3.3}$$

Now, subtracting the above equation to the total energy equation (3.2c) and employing the tensorial identity

$$\nabla \cdot (\mathbb{T}\boldsymbol{V}) = \boldsymbol{V} \cdot \nabla \cdot \mathbb{T} + \mathbb{T} : \mathbb{D},$$

where $\mathbb{D} = \frac{1}{2}[\nabla \boldsymbol{V} + (\nabla \boldsymbol{V})^t]$ is the symmetric part of the velocity gradient tensor, $\nabla \boldsymbol{V}$, leads to the internal energy equation

$$\rho \frac{de}{dt} - \mathbb{T} : \mathbb{D} + \nabla \cdot \boldsymbol{q} = 0. \tag{3.4}$$

Let us recall that the symbol : denotes the inner product between second-order tensors, namely $\mathbb{T} : \mathbb{D} = \mathrm{tr}(\mathbb{T}^t\mathbb{D})$, where tr is the trace operator.
We observe that the system (3.1) is not closed since there are much more unknowns than equations. To achieve the closure of this system, one needs to specify the constitutive laws which define the Cauchy stress tensor and the heat flux vector in terms of the physical variables. Moreover, one also needs to provide a thermodynamic closure which allows to relate the thermodynamic variables. The constitutive laws specification relies on two fundamental principles which are:

- The requirement that the constitutive laws should obey the Principle of Material Frame Indifference [55].
- The constitutive laws must be determined in a such a way that they ensure the consistency with the Second Law of Thermodynamics.

The Principle of Material Frame Indifference simply states that the constitutive laws should not depend on whatever external frame of reference is used to describe them. Regarding the Second Law of Thermodynamics, it is an evolution principle which states that all real thermodynamic transformation is accompanied by a transfer of energy due to the fluctuations of atoms and/or molecules at the microscopic level [55]. The disorder in the system induced by these fluctuations is measured by means of the entropy. The Second Law of Thermodynamics simply states that the entropy production is non-negative. The entropy imbalance is governed by the famous Clausius-Duhem inequality, which in our case writes

$$\frac{\partial}{\partial t}(\rho\eta) + \nabla \cdot (\rho\eta \boldsymbol{V}) + \nabla \cdot (\frac{\boldsymbol{q}}{T}) \geq 0. \tag{3.5}$$

Here, η denotes the specific entropy and $T > 0$ is the absolute temperature. Introducing the material derivative and employing the mass conservation equation leads to rewrite the above inequality under the non-conservative form

$$\rho T \frac{d\eta}{dt} + T\nabla \cdot (\frac{\boldsymbol{q}}{T}) \geq 0. \tag{3.6}$$

Introducing $\mathcal{P}_{irr}$ as the net entropy production allows to transform the above inequality into the equation

$$\rho T \frac{d\eta}{dt} + T\nabla \cdot (\frac{\boldsymbol{q}}{T}) = \mathcal{P}_{irr}, \tag{3.7}$$

where the net entropy production term is always non-negative, *i.e.* $\mathcal{P}_{irr} \geq 0$. It remains to determine the expressions of $\mathbb{T}$ and $\boldsymbol{q}$ to ensure the compatibility of the constitutive laws with the Principle of Material Frame Indifference and the Second Law of Thermodynamics. This is the topic of the next paragraph.

3.1.2 Constitutive laws

Here, we recall the construction of the constitutive laws for an isotropic, compressible, heat conducting, viscous fluid. As a consequence of frame-indifference [55], the constitutive relations for the Cauchy stress tensor and the heat flux vector are of the form

$$\mathbb{T} \equiv \mathbb{T}(\rho, T, \mathbb{D}), \quad \boldsymbol{q} \equiv \boldsymbol{q}(\rho, T, \nabla T).$$

In addition, the constitutive relation for the Cauchy stress tensor admits the decomposition

$$\mathbb{T} = -p(\rho, T)\,\mathbb{I} + \mathbb{S}(\rho, T, \mathbb{D}). \tag{3.8}$$

Here, the scalar valued function, p, represents the stress in the fluid in the absence of flow, it is called the thermodynamic pressure. The tensor function $\mathbb{S}$, which is termed the viscous stress tensor, represents the part of the stress due to the flow, it vanishes in the absence of fluid motion. It is an isotropic function, that is $\mathbb{Q}\mathbb{S}(\rho, T, \mathbb{D})\mathbb{Q}^t = \mathbb{S}(\rho, T, \mathbb{Q}\mathbb{D}\mathbb{Q}^t)$, where $\mathbb{Q}$ is an arbitrary rotation, *i.e.*, $\mathbb{Q}^t\mathbb{Q} = \mathbb{I}$ and $\det \mathbb{Q} = 1$. Usually, one also defines the mechanical pressure as $p^m = -\frac{1}{3}\operatorname{tr}\mathbb{T}$. This quantity is related to the thermodynamic pressure by means of

$$p^m = p(\rho, T) - \frac{1}{3}\operatorname{tr}\mathbb{S}. \tag{3.9}$$

We observe that the mechanical pressure in a compressible viscous fluid includes both a thermodynamic contribution and a dynamical contribution generated by the viscosity of the fluid. Regarding the thermodynamic pressure, p, it is related to the other thermodynamic variables (density, temperature, internal energy and entropy) by means of the fundamental Gibbs relation

$$T\mathrm{d}\eta = \mathrm{d}e + p\mathrm{d}(\frac{1}{\rho}). \tag{3.10}$$

This relation allows to define properly the thermodynamic pressure and the temperature by means of the fundamental equation of state written under the form $e \equiv e(\rho, \eta)$. Taking the total differential of this expression and employing the Gibbs relation (3.10) yields

$$p = \rho^2 \left(\frac{\partial e}{\partial \rho}\right)_\eta, \quad T = \left(\frac{\partial e}{\partial \eta}\right)_\rho.$$

Let us point out that the equation of state is an intrinsic property of the material under consideration. It can be specified under various forms depending on the number of state variables which are employed. For practical applications involving thermal processes, it is convenient to define the equation of state by expressing the specific internal energy and the thermodynamic pressure in terms of the density and the temperature, *i.e.*, $e = e(\rho, T)$ and $p = p(\rho, T)$.
Gathering the previous results, we are now in position to exhibit the thermodynamic restriction on the constitutive laws to ensure their compatibility with the Second Law. To this end, we shall compute the net entropy production term defined by (3.7). Employing the fundamental Gibbs relation (3.10) leads to write the time rate of change of entropy as

$$\rho T\frac{d\eta}{dt} = \rho\frac{de}{dt} + p\rho\frac{d}{dt}(\frac{1}{\rho}).$$

Then, substituting the time rates of change of specific internal energy (3.4) and specific volume (3.2a) into the above equation yields

$$\rho T\frac{d\eta}{dt} = (\mathbb{T} + p\,\mathbb{I}) : \mathbb{D} - \nabla\cdot\boldsymbol{q}.$$

Finally, comparing the above equation to the Clausius-Duhem equation (3.7) leads to the following expression of the net entropy production

$$\mathcal{P}_{irr} = \mathbb{S} : \mathbb{D} - \frac{1}{T}\boldsymbol{q}\cdot\nabla T. \tag{3.11}$$

Here, we have made use of the decomposition (3.8) of the Cauchy stress tensor. The compatibility with the Second Law requires to have $\mathcal{P}_{irr} \geq 0$. Recalling that the temperature is non-negative, this amounts to require that the constitutive laws satisfy the two following conditions

$$\mathbb{S} : \mathbb{D} \geq 0, \tag{3.12a}$$

$$-\boldsymbol{q}\cdot\nabla T \leq 0. \tag{3.12b}$$

These are the constraints imposed by Thermodynamics on the modeling of the constitutive laws that characterize the fluid. The first constraint simply expresses that viscous deformation converts mechanical energy into heat while the second states that heat flux direction is opposite to temperature gradient.
We assume that the fluid is linearly viscous in the sense that the viscous stress is linear in $\mathbb{D}$. In this case, since the viscous stress tensor, $\mathbb{S}$, is an isotropic function, the representation theorem for isotropic linear tensor functions [55] states that $\mathbb{S}$ is expressed in terms of $\mathbb{D}$ as follows

$$\mathbb{S} = 2\mu\mathbb{D} + \lambda \operatorname{tr}(\mathbb{D})\,\mathbb{I}. \tag{3.13}$$

Here, the real valued coefficients, $\lambda = \lambda(\rho, T)$ and $\mu = \mu(\rho, T)$, are called the viscosity coefficients. A linearly viscous fluid is also named a Newtonian viscous fluid. Introducing the classical decomposition of the strain rate tensor, $\mathbb{D}$, into deviatoric and dilatational part yields

$$\mathbb{S} = 2\mu\mathbb{D}_0 + (\lambda + \frac{2}{3}\mu)(\operatorname{tr}\mathbb{D})\,\mathbb{I}, \tag{3.14}$$

where the deviatoric part of the strain rate tensor writes $\mathbb{D}_0 = \mathbb{D} - \frac{1}{3}(\operatorname{tr}\mathbb{D})\,\mathbb{I}$. The coefficient $\lambda + \frac{2}{3}\mu$ is known as the bulk or dilatational viscosity whereas μ is also named the dynamic or shear viscosity. In order to investigate the consequence of the thermodynamic restriction (3.12a) on the viscosity coefficients, we compute the inner product $\mathbb{S} : \mathbb{D}$. Utilizing definition (3.14) yields

$$\mathbb{S} : \mathbb{D} = 2\mu\mathbb{D}_0 : \mathbb{D}_0 + (\lambda + \frac{2}{3}\mu)(\operatorname{tr}\mathbb{D})^2.$$

This shows that the thermodynamic constraint (3.12a) is satisfied provided that the viscosity coefficients are such that

$$\mu \geq 0, \quad \text{and} \quad \lambda + \frac{2}{3}\mu \geq 0. \tag{3.15}$$

We also assume that the heat flux is linear in the temperature gradient, thus the heat flux writes under the form

$$\boldsymbol{q} = -\mathbb{K}\nabla T, \tag{3.16}$$

where $\mathbb{K} = \mathbb{K}(\rho, T)$ is the conductivity tensor of the fluid. This constitutive law is called the Fourier law. To satisfy the second thermodynamic constraint (3.12b), the conductivity tensor must be definite positive

$$\mathbb{K}\boldsymbol{A} \cdot \boldsymbol{A} \geq 0, \quad \forall \boldsymbol{A} \in \mathfrak{R}^d. \tag{3.17}$$

For classical fluids such as air, the conductivity tensor is diagonal and collapses to $\mathbb{K} = \kappa\,\mathbb{I}$, where κ denotes the thermal conductivity of the fluid, which has to be non-negative to satisfy the thermodynamic requirement. Combining the above results leads to write the net entropy production as

$$\mathcal{P}_{irr} = 2\mu\mathbb{D}_0 : \mathbb{D}_0 + (\lambda + \frac{2}{3}\mu)(\operatorname{tr}\mathbb{D})^2 + \frac{1}{T}\mathbb{K}\nabla T \cdot \nabla T.$$

Finally, the time rate of change of entropy for a linearly viscous fluid characterized by a linear heat flux writes

$$\rho T\frac{d\eta}{dt} + T\nabla \cdot (\frac{\boldsymbol{q}}{T}) = 2\mu\mathbb{D}_0 : \mathbb{D}_0 + (\lambda + \frac{2}{3}\mu)(\operatorname{tr}\mathbb{D})^2 + \frac{1}{T}\mathbb{K}\nabla T \cdot \nabla T \geq 0. \tag{3.18}$$

This shows that the Newtonian fluid satisfies the Clausius-Duhem inequality and is thus thermodynamically consistent.
Recalling the relationship (3.9) between the mechanical and the thermodynamic pressure and utilizing the expression of the constitutive law yields

$$p^m = p(\rho, T) - (\lambda + \frac{2}{3}\mu)\operatorname{tr}\mathbb{D}.$$

We observe that these two pressures coincide either if $\operatorname{tr}\mathbb{D} = 0$ or if $\lambda + \frac{2}{3}\mu = 0$. The former case corresponds to an incompressible fluid flow and is excluded for the present study, whereas the latter corresponds to the Stokes relation

$$\lambda + \frac{2}{3}\mu = 0. \tag{3.19}$$

This assumption is valid for monoatomic gases and for gases that are rarefied enough, refer to [55]. In what follows, we shall assume that the Newtonian fluid under consideration satisfied the Stokes relation (3.19). Thus, the constitutive law of a Newtonian fluid satisfying the Stokes assumption writes

$$\mathbb{S} = 2\mu\mathbb{D}_0. \tag{3.20}$$

For practical applications it remains to give explicit expressions of the transport coefficients, μ and κ, in terms of the mass density and temperature. These expressions depend on the flow regime. Let us point out that it is possible to obtain accurate approximation of the transport coefficients utilizing the kinetic theory of rarefied gases. In this framework, an asymptotic expansion of the Boltzmann equation provides formulas for the transport coefficients which depend on the nature of the intermolecular forces. The interested reader may refer to [68]. Here, we shall employ simple approximations of these transport coefficients knowing that their applicability is physically restricted. First, the dynamic viscosity coefficient is expressed in terms of temperature by means of the Sutherland formula

$$\mu(T) = \mu_0 \left(\frac{T}{T_0}\right)^{\frac{3}{2}} \frac{T_0 + C}{T + C}, \tag{3.21}$$

where μ_0 is the reference viscosity at the reference temperature T_0 and C is the Sutherland constant for the fluid under consideration. For the air, we shall use the following values, $\mu_0 = 1.789 \cdot 10^{-5}\,\mathrm{Pa\ s}$, $T_0 = 288\,\mathrm{K}$ and $S = 110\,\mathrm{K}$. Regarding the heat conductivity, κ, it shall be defined later from the viscosity coefficient.

3.1.3 Equation of state

We have already seen in the above paragraph that the thermodynamic closure of the Fluid Dynamics equations (3.1) is ensured by means of an equation of state written under the generic form

$$p = p(\rho, T), \text{ and } e = e(\rho, T). \tag{3.22}$$

Here, we have chosen to work with the temperature. It is also possible to consider the equation of state written under the form $p = p(\rho, \eta)$ in which we are working with entropy. Taking the differential of this equation of state yields

$$\mathrm{d}p = (\frac{\partial p}{\partial \rho})_\eta \mathrm{d}\rho + (\frac{\partial p}{\partial \eta})_\rho \mathrm{d}\eta.$$

If the flow undergoes a thermodynamic transformation for which entropy is conserved, then $\mathrm{d}\eta = 0$ and thus the above formula collapses to $\mathrm{d}p = (\frac{\partial p}{\partial \rho})_\eta \mathrm{d}\rho$. This kind of transformation is called isentropic and characterized by the fact that the pressure fluctuation is proportional to the density fluctuation. Assuming that the equation of state is such that this coefficient satisfies $(\frac{\partial p}{\partial \rho})_\eta > 0$ allows to introduce the isentropic sound speed as

$$c = \sqrt{(\frac{\partial p}{\partial \rho})_\eta}. \tag{3.23}$$

For our numerical applications, we shall consider the simple situation which corresponds to a calorically perfect gas knowing that the applicability of such an assumption is limited to moderate levels of temperatures, refer to [15]. For instance, in the case of real life applications such as those encountered in the domain of atmospheric reentry flows, the level of reached temperatures is so high that one needs to employ more sophisticated equation of states to properly describe the complex thermodynamic processes at play in such flows, refer to [15]. A calorically perfect gas is characterized by the fact that the specific internal energy and the specific enthalpy are linear functions with respect to temperature, knowing that the specific enthalpy is defined by $h = e + \frac{p}{\rho}$. Namely, e and h write

$$e = C_v T \text{ and } h = C_p T, \tag{3.24}$$

where C_v and C_p are the specific heat respectively at constant volume and constant pressure. For a calorically perfect gas the specific heats are constant and in turn, their ratio, $\gamma = \frac{C_p}{C_v}$, which is also constant and such that $\gamma > 1$, is called the polytropic index. Finally, the specific heats are defined by means of the Mayer relationship

$$C_p - C_v = \frac{R}{\mathcal{M}}, \tag{3.25}$$

where $R = 8.314\,\mathrm{J\ K^{-1}mol^{-1}}$ is the ideal gas constant and $\mathcal{M}$ is the molar mass of the gas. For air in standard conditions, the molar mass is about $\mathcal{M} = 29\,10^{-3}\,\mathrm{kg\ mol^{-1}}$. Employing the Mayer relationship (3.25) and the polytropic index, γ, leads to

$$C_p = \frac{\gamma R}{(\gamma - 1)\mathcal{M}}, \quad C_v = \frac{R}{(\gamma - 1)\mathcal{M}}.$$

For air, which is a diatomic gas, the value of the polytropic index is set to $\gamma = \frac{7}{5}$.
Now, using the definition of the specific enthalpy and (3.24) allows to write the equation of state under the form

$$p = \rho \frac{R}{\mathcal{M}} T, \quad e = C_v T. \tag{3.26}$$

We remark that employing the expression of C_v in terms of the molar mass, it also possible to express the pressure under the form $p = (\gamma - 1)\rho e$. For a calorically perfect gas, the isentropic sound speed is given by

$$c = \sqrt{\gamma \frac{p}{\rho}} = \sqrt{\gamma \frac{R}{\mathcal{M}} T}.$$

3.1.4 Summary: the Navier-Stokes equations

Gathering the above results, we are now in position to write the partial differential equations which govern the fluid under consideration. This set of equations consists of the famous compressible Navier-Stokes equations

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \boldsymbol{V}) = 0, \tag{3.27a}$$

$$\frac{\partial}{\partial t}(\rho \boldsymbol{V}) + \nabla \cdot (\rho \boldsymbol{V} \otimes \boldsymbol{V}) + \nabla p = \nabla \cdot \mathbb{S}, \tag{3.27b}$$

$$\frac{\partial}{\partial t}(\rho E) + \nabla \cdot (\rho E \boldsymbol{V}) + \nabla \cdot (p \boldsymbol{V}) = \nabla \cdot (\mathbb{S} \boldsymbol{V}) - \nabla \cdot \boldsymbol{q}. \tag{3.27c}$$

These equations are supplemented by the constitutive laws which define respectively the viscous stress tensor, $\mathbb{S}$, and the heat flux, $\boldsymbol{q}$, as

$$\begin{aligned}\mathbb{S} &= 2\mu\mathbb{D}_0,\\ \boldsymbol{q} &= -\kappa\nabla T.\end{aligned}$$

Here, $\mu > 0$ is the dynamic viscosity and $\kappa > 0$ the heat conductivity, which are intrinsic properties of the fluid under consideration. We recall that the deviatoric part of the strain rate, $\mathbb{D}_0$, writes

$$\mathbb{D}_0 = \frac{1}{2}[\nabla\boldsymbol{V} + (\nabla\boldsymbol{V})^t] - \frac{1}{3}(\nabla\cdot\boldsymbol{V})\,\mathbb{I}.$$

The pressure, p, and the specific internal energy, $e = E - \frac{1}{2}\boldsymbol{V}^2$, are expressed in terms of the density and the temperature by means of the equation of state. In $\mathfrak{R}^d$, the Navier-Stokes system consists of $d+2$ scalar equations, whereas the number of unknowns (for instance ρ, $\boldsymbol{V}$ and T) is also equal to $d+2$ thanks to the constitutive laws and the equation of state, thus the system of governing equations is closed. The specific energy equation related to the Navier-Stokes system is obtained by combining (3.4) and the mass conservation equation

$$\frac{\partial}{\partial t}(\rho e) + \nabla\cdot(\rho e\boldsymbol{V}) + p\nabla\cdot\boldsymbol{V} = \mathbb{S}:\mathbb{D} - \nabla\cdot\boldsymbol{q}.$$

Substituting the specific enthalpy, $h = e + \frac{p}{\rho}$, into the above equation yields

$$\frac{\partial}{\partial t}(\rho h) + \nabla\cdot(\rho h\boldsymbol{V}) - \frac{dp}{dt} = \mathbb{S}:\mathbb{D} - \nabla\cdot\boldsymbol{q}, \tag{3.28}$$

where $\frac{dp}{dt}$ denotes the material derivative of the pressure. Utilizing the Fourier law and the calorically perfect gas equation of state, we get the following convection-diffusion equation satisfied by the temperature field

$$\frac{\partial}{\partial t}(\rho C_p T) + \nabla\cdot(\rho C_p T\boldsymbol{V}) - \frac{dp}{dt} - \nabla\cdot(\kappa\nabla T) = \mathbb{S}:\mathbb{D}. \tag{3.29}$$

Here, the right-hand side represents the viscous heating which is nothing but the work rate of the viscous stress.

We point out that system (3.27) represents the local conservative form of the Navier-Stokes equations. For the subsequent Finite Volume discretization of the Navier-Stokes, it is worth taking into consideration the integral form of this system. It is simply obtained by integrating equations (3.27a), (3.27b) and (3.27c) over the fixed region $\omega_f \subset \mathcal{D}$ and employing the divergence theorem to get

$$\frac{\partial}{\partial t}\int_{\omega_f}\rho\,\mathrm{d}v + \int_{\partial\omega_f}\rho\boldsymbol{V}\cdot\boldsymbol{n}\,\mathrm{d}s = 0, \tag{3.30a}$$

$$\frac{\partial}{\partial t}\int_{\omega_f}\rho\boldsymbol{V}\,\mathrm{d}v + \int_{\partial\omega_f}\boldsymbol{V}(\rho\boldsymbol{V})\cdot\boldsymbol{n}\,\mathrm{d}s + \int_{\partial\omega_f}p\boldsymbol{n}\,\mathrm{d}s = \int_{\partial\omega_f}\mathbb{S}\boldsymbol{n}\,\mathrm{d}s, \tag{3.30b}$$

$$\frac{\partial}{\partial t}\int_{\omega_f}\rho E\,\mathrm{d}v + \int_{\partial\omega_f}E(\rho\boldsymbol{V})\cdot\boldsymbol{n}\,\mathrm{d}s + \int_{\partial\omega_f}p\boldsymbol{V}\cdot\boldsymbol{n}\,\mathrm{d}s = \int_{\partial\omega_f}(\mathbb{S}\boldsymbol{V})\cdot\boldsymbol{n}\,\mathrm{d}s - \int_{\partial\omega_f}\boldsymbol{q}\cdot\boldsymbol{n}\,\mathrm{d}s. \tag{3.30c}$$

Here, $\partial\omega_f$ denotes the boundary of the fixed region, ω_f, and $\boldsymbol{n}$ its unit outward normal.

We conclude this paragraph by recalling the time rate of change of entropy. Combining (3.7), (3.11) and the expression of the constitutive laws leads to the entropy equation

$$\frac{\partial}{\partial t}(\rho\eta) + \nabla\cdot(\rho\eta\boldsymbol{V}) + \nabla\cdot(\frac{\boldsymbol{q}}{T}) = \frac{2\mu}{T}\mathbb{D}_0:\mathbb{D}_0 + \frac{\kappa}{T^2}(\nabla T)^2 \geq 0, \tag{3.31}$$

where $\boldsymbol{q} = -\kappa\nabla T$ is the heat flux vector. We note in passing that the right-hand side of the above equation is always positive, which ensures the fulfillment of the Clausius-Duhem inequality and thus the thermodynamic consistency of this model. Finally, integrating the above equation over the fixed domain ω_f and applying the divergence theorem yields the integral form of the entropy equation

$$\frac{d}{dt}\int_{\omega_f}\rho\eta\,\mathrm{d}v + \int_{\partial\omega_f}\eta(\rho\boldsymbol{V})\cdot\boldsymbol{n}\,\mathrm{d}s + \int_{\partial\omega_f}\frac{\boldsymbol{q}}{T}\cdot\boldsymbol{n}\,\mathrm{d}s = \int_{\omega_f}\frac{1}{T}\left[2\mu\mathbb{D}_0:\mathbb{D}_0 + \frac{\kappa}{T}(\nabla T)^2\right]\mathrm{d}v. \quad (3.32)$$

The integrand at the right-hand side is the dissipation function, it is a source term characterizing the intrinsic irreversibilities related to the viscosity and conductivity of the fluids.

3.1.5 Non dimensional form of the compressible Navier-Stokes equations

The aim of this paragraph is to recall briefly the non dimensional form of the Navier-Stokes equations. The main interest of this non dimensional form lies in the fact that it allows to measure the order of magnitude of the different terms present in the Navier-Stokes equations. In addition, this procedure provides a natural introduction to the dimensionless numbers of compressible fluid mechanics such as the Mach, Reynolds and Prandtl numbers.
Being given the physical variable, $\varphi = \varphi(\boldsymbol{x}, t)$, we define its corresponding order of magnitude, $\varphi^\star$, and also the corresponding dimensionless variable $\bar{\varphi} = \frac{\varphi}{\varphi^\star}$. The order of magnitude is chosen such that $\bar{\varphi} = O(1)$. We remark that the length scale is denoted by L, where L is a characteristic length attached to the flow, this might be for instance the length of the chord line when one is dealing with the aerodynamic flow around a wing. The order of magnitude of the thermodynamic pressure is taken to be equal to the density scale times the square of the sound speed scale, *i.e.* $p^\star = \rho^\star(c^\star)^2$. Regarding the specific internal energy and the specific enthalpy, they share the same scale which is the square of the sound speed scale, *i.e.*, $e^\star = h^\star = (c^\star)^2$. These choices are justified for high-speed compressible flows to the extent that for such flows the internal energy is strongly related to the kinetic energy induced by the thermal agitation of the molecules composing the fluid. We assume that the order of magnitude of time corresponds to convective time scale, that is $t^\star = \frac{L}{V^\star}$, where $V^\star$ is the order of magnitude of velocity. Introducing the decomposition $\varphi = \varphi^\star\,\bar{\varphi}$ into the Navier-Stokes equations yields

$$\bar{\rho}\frac{d}{d\bar{t}}(\frac{1}{\bar{\rho}}) - \bar{\nabla}\cdot\bar{\boldsymbol{V}} = 0, \quad (3.33a)$$

$$\bar{\rho}\frac{d}{d\bar{t}}\bar{\boldsymbol{V}} + \frac{1}{M_a^2}\bar{\nabla}p - \frac{1}{Re}\bar{\nabla}\cdot\bar{\mathbb{S}} = \boldsymbol{0}, \quad (3.33b)$$

$$\bar{\rho}\frac{d\bar{h}}{d\bar{t}} - \frac{d\bar{p}}{d\bar{t}} - \frac{M_a^2}{Re}(\bar{\mathbb{S}}:\bar{\mathbb{D}}) + \frac{1}{RePr}\bar{\nabla}\cdot\bar{\boldsymbol{q}} = 0. \quad (3.33c)$$

The above equations are the dimensionless form of the Navier-Stokes equations. They contain the dimensionless numbers M_a, R_e and P_r which respectively stands for the Mach, Reynolds and Prandtl numbers. These dimensionless numbers are defined by

$$M_a = \frac{V^\star}{c^\star},$$
$$R_e = \frac{\rho^\star V^\star L}{\mu^\star},$$
$$P_r = \frac{\mu^\star C_p^\star}{\kappa^\star}.$$

Here, $\mu^\star$, $\kappa^\star$ and $C_p^\star$ are respectively the orders of magnitude of the dynamic viscosity, heat conductivity and specific heat at constant pressure. The Mach number represents the ratio

of the flow speed to the sound speed. For $M_a < 1$ the flow is called subsonic whereas it is termed supersonic for $M_a > 1$. The Reynolds number represents the ratio of the inertial forces to the viscous forces. Introducing the convective and the viscous time scales as $t_{conv} = \frac{L}{V^\star}$ and $t_{visc} = \frac{\rho^\star L^2}{\mu^\star}$ leads to rewrite the Reynolds number as the ratio of the aforementioned time scales, *i.e.*, $R_e = \frac{t_{visc}}{t_{conv}}$. Finally, introducing the conductive time scale $t_{cond} = \frac{\rho^\star C_p^\star L^2}{\kappa^\star}$ yields to present the Prandtl number as the ratio of the conductive time scale to the viscous time scale, *i.e.*, $P_r = \frac{t_{cond}}{t_{visc}}$. It can be also viewed as the ratio of the energy dissipated by viscosity to the energy transported by thermal conduction. Let us note that for low speed flows, *i.e.* $M_a \ll 1$, the viscous heating term plays no role in the energy equation (3.33c). On the other hand, in the case of high speed flows, for which $M_a \gg 1$, the viscous heating term has a significant importance in the energy balance. It also implies a strong coupling between the momentum and the energy equations.
We point out that the Prandtl number uniquely depends on the nature of the fluid. For air at moderate level of temperatures, it is almost constant and set to the constant value $P_r = 0.71$, refer to [15]. In our numerical applications, we shall use these fixed value of the Prandtl number to define the heat conductivity, knowing the dynamic viscosity.

3.1.6 Initial and boundary conditions

We consider that the fluid is flowing inside a domain $\mathcal{D}$ which is a sub-region of the d-dimensional space $\mathfrak{R}^d$. If the flow is unsteady, the determination of the solution of the Navier-Stokes equations at time $t > 0$ (provided that it exists) requires the specification of the initial values of the following flow variables

$$\rho(\boldsymbol{x}, 0) = \rho^0(\boldsymbol{x}), \quad \boldsymbol{V}(\boldsymbol{x}, 0) = \boldsymbol{V}^0(\boldsymbol{x}) \text{ and } T(\boldsymbol{x}, 0) = T^0(\boldsymbol{x}), \text{ for all } \boldsymbol{x} \in \mathcal{D}. \tag{3.34}$$

Here, ρ^0, $\boldsymbol{V}^0$ and T^0 are respectively the initial density, velocity field and temperature, which are given.
Concerning the boundary conditions specification for the Navier-Stokes system, one usually makes the distinction between two situations which correspond respectively either to the case of a solid boundary or to the case of an inlet or outlet boundary. The former is encountered in the presence of a solid wall in the domain where the fluid is flowing. The latter is a consequence of the fact that the spatial domain occupied by the fluid is unbounded whereas the computational domain is bounded. Then, arises the subtle problem of specifying boundary data on this inlet or outlet boundary. We discuss briefly these two cases.

Wall boundary conditions

Let Γ be a solid surface contained into the computational domain $\mathcal{D}$. The velocity of fluid particle which lies on the solid wall must be equal to the velocity of the solid wall, that is

$$\boldsymbol{V}(\boldsymbol{x}, t) = \boldsymbol{V}_\Gamma(\boldsymbol{x}, t), \text{ for all } \boldsymbol{x} \in \Gamma \text{ and } t > 0. \tag{3.35}$$

Here, $\boldsymbol{V}_\Gamma$ is the given solid wall velocity. In the case of a fixed solid wall, this boundary condition is termed a no-slip boundary condition and the fluid velocity vanishes at the solid wall.
Regarding the thermal behavior of the fluid, one has the three following exclusive choices: either the normal heat flux at the wall is prescribed (Neumann boundary condition) or the wall temperature is imposed (Dirichlet boundary condition) or one can specify a linear combination of the normal heat flux and the temperature (Robin boundary condition). We summarize these conditions as follows

- Dirichlet boundary condition

$$T(\boldsymbol{x},t) = T_\Gamma(\boldsymbol{x},t), \quad \text{for all } \boldsymbol{x} \in \Gamma \text{ and } t > 0, \tag{3.36}$$

where T_Γ is the solid wall temperature.

- Neumann boundary condition

$$\boldsymbol{q}(\boldsymbol{x},t) \cdot \boldsymbol{n} = q_\Gamma(\boldsymbol{x},t), \quad \text{for all } \boldsymbol{x} \in \Gamma \text{ and } t > 0, \tag{3.37}$$

where $\boldsymbol{n}$ is the unit outward boundary to the boundary surface and q_Γ is a given wall heat flux.

- Robin boundary condition

$$\alpha T(\boldsymbol{x},t) + \beta \boldsymbol{q}(\boldsymbol{x},t) \cdot \boldsymbol{n} = \tilde{q}_\Gamma(\boldsymbol{x},t), \quad \text{for all } \boldsymbol{x} \in \Gamma \text{ and } t > 0, \tag{3.38}$$

where α, β are given parameter and $\tilde{q}_\Gamma$ is a given wall heat flux.

Inlet and outlet boundary conditions

These particular boundary conditions are encountered when the fluid enters or leaves the bounded computational domain. According to [31] the number of boundary conditions to apply depends on whether the boundary is an inlet or an outlet. For the Navier-Stokes equations, a full set of boundary conditions has to be specified at both supersonic and subsonic inflows and one less at the outflow. An heuristic argument to justify this relies on the fact that the mass conservation is a first-order advection equation which requires only one boundary condition at the inlet whereas the momentum and the total energy equations are second-order advection diffusion equations which require boundary conditions at both inlet and outlet.

3.2 The Euler equations

In this section, we describe a simpler model than the Navier-Stokes equations, which is deduced from them by assuming a non viscous and non heat conducting fluid. This simplification of the Navier-Stokes equations corresponds to the famous Euler equations. Then, we shall recall some useful mathematical properties of this model.

3.2.1 Governing equations for an inviscid non heat conducting fluid

The Euler equations have been first described by Euler [44] and at that time they only consisted of the continuity equation and the momentum equation. The energy equation has been added sixty years later by Laplace [78]. Nowadays, the whole set of equations is called the Euler equations. These equations are simply obtained from the Navier-Stokes equations by assuming that the fluid under consideration is non viscous and does not conduct heat. Namely, the constitutive laws (3.20) and (3.16) collapses to $\mathbb{S} = \mathbb{0}$ and $\boldsymbol{q} = \boldsymbol{0}$ since the dynamic viscosity, μ, and the heat conductivity, κ, are equal to zero. Substituting these assumptions into (3.27) leads to

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \boldsymbol{V}) = 0, \tag{3.39a}$$

$$\frac{\partial}{\partial t}(\rho \boldsymbol{V}) + \nabla \cdot (\rho \boldsymbol{V} \otimes \boldsymbol{V}) + \nabla p = \boldsymbol{0}, \tag{3.39b}$$

$$\frac{\partial}{\partial t}(\rho E) + \nabla \cdot (\rho E \boldsymbol{V}) + \nabla \cdot (p \boldsymbol{V}) = 0. \tag{3.39c}$$

The thermodynamic closure of the above system is ensured by an equation of state written under the form $p = p(\rho, e)$, where $e = E - \frac{1}{2}\boldsymbol{V}^2$ is the specific internal energy. The above system of equations consists of the local conservative form of the Euler equations. Introducing the material derivative and employing the mass conservation equation leads to the local non conservative form of the Euler equations

$$\rho \frac{d}{dt}(\frac{1}{\rho}) - \nabla \cdot \boldsymbol{V} = 0, \tag{3.40a}$$

$$\rho \frac{d}{dt}\boldsymbol{V} + \nabla p = \boldsymbol{0}, \tag{3.40b}$$

$$\rho \frac{d}{dt}E + \nabla \cdot (p\boldsymbol{V}) = 0. \tag{3.40c}$$

This system is the counterpart of (3.2) in which we set $\mathbb{T} = -p\,\mathbb{I}$ and $\boldsymbol{q} = \boldsymbol{0}$. Using this non conservative form and the fundamental Gibbs relation (3.10), we obtain the time rate of change of entropy

$$\rho T \frac{d\eta}{dt} = 0. \tag{3.41}$$

This equation shows that the entropy is conserved along the fluid particles paths. We point out that this result only holds for smooth flows since the equivalence between the non conservative and the conservative form of the Euler equations is only valid under a sufficient smoothness assumption of the flow variables. It is well known that the Euler equations can admit discontinuous solutions such as shock waves or contact discontinuities, refer to [52]. In this case, the use of the Euler equations written under the non conservative form is forbidden. For this reason, the most convenient form of the Euler equations is the integral form which is deduced from (3.30)

$$\frac{\partial}{\partial t}\int_{\omega_f} \rho \,\mathrm{d}v + \int_{\partial\omega_f} \rho\boldsymbol{V}\cdot\boldsymbol{n}\,\mathrm{d}s = 0, \tag{3.42a}$$

$$\frac{\partial}{\partial t}\int_{\omega_f} \rho\boldsymbol{V}\,\mathrm{d}v + \int_{\partial\omega_f} \boldsymbol{V}(\rho\boldsymbol{V})\cdot\boldsymbol{n}\,\mathrm{d}s + \int_{\partial\omega_f} p\boldsymbol{n}\,\mathrm{d}s = \boldsymbol{0}, \tag{3.42b}$$

$$\frac{\partial}{\partial t}\int_{\omega_f} \rho E\,\mathrm{d}v + \int_{\partial\omega_f} E(\rho\boldsymbol{V})\cdot\boldsymbol{n}\,\mathrm{d}s + \int_{\partial\omega_f} p\boldsymbol{V}\cdot\boldsymbol{n}\,\mathrm{d}s = 0, \tag{3.42c}$$

where ω_f is a fixed sub region of the domain, $\mathcal{D}$ occupied by the fluid. If ω_f contains a discontinuity surface, through which the flow variables undergo a jump, then the above system has to be supplemented by jump relations, which express the conservation of mass, momentum and total energy through the discontinuity surface. These jump relations, which allow to define weak solutions of the Euler equations in the presence of a discontinuity, are called the Rankine-Hugoniot relations, refer to [55, 52].
The integral form of the entropy equation is deduced from its Navier-Stokes counterpart (3.32) passing to the limit $\mu \to 0$ and $\kappa \to 0$ and we get the entropy imbalance

$$\frac{d}{dt}\int_{\omega_f} \rho\eta\,\mathrm{d}v + \int_{\partial\omega_f} \eta(\rho\boldsymbol{V})\cdot\boldsymbol{n}\,\mathrm{d}s \geq 0. \tag{3.43}$$

This entropy inequality ensures the thermodynamic consistency of the Euler equations. In the presence of flow discontinuities, it guarantees that the kinetic energy is properly dissipated into internal energy through irreversible processes such as shock waves, according to the Second Law of Thermodynamics. In other words, from a mathematical point of view, it provides a criterion to select the physically relevant weak solution when dealing with discontinuous solutions of the

Euler equations. For a rigorous mathematical treatment of this subtle topic, the interested reader might refer to [52].
We conclude this paragraph by writing the Euler equations under the following compact form, which shall be useful for the subsequent space discretization

$$\frac{\partial}{\partial t}\boldsymbol{U} + \nabla \cdot \boldsymbol{F}^e(\boldsymbol{U}) = \boldsymbol{0}, \tag{3.44}$$

where $\boldsymbol{U} = (\rho, \rho\boldsymbol{V}, \rho E)^t$ is the vector of the conservative variables and $\boldsymbol{F}^e$ is the corresponding fluxes vector.

3.2.2 Mathematical properties of the Euler equations

In this section, we aim at briefly recalling some important well known results concerning the mathematical structure of the compressible Euler equations. One of its most important features lies in the fact that the system of Euler equations is hyperbolic. We describe this property by means of a plane-wave analysis.

Hyperbolicity and plane-wave analysis

For sake of completeness, we briefly recall the definition of hyperbolicity for a system of conservation law in several space variables [52]. Let $\mathcal{D}$ be an open subset of $\mathfrak{R}^p$, and let $\boldsymbol{F}_j$, $j = 1 \dots d$, be d smooth functions from $\mathcal{D}$ into $\mathfrak{R}^p$. The general form of a system of conservation laws writes

$$\frac{\partial}{\partial t}\boldsymbol{U} + \sum_{j=1}^{d} \frac{\partial}{\partial x_j}\boldsymbol{F}_j(\boldsymbol{U}) = \boldsymbol{0}. \tag{3.45}$$

Here, $\boldsymbol{U} = \boldsymbol{U}(\boldsymbol{x}, t) \in \mathfrak{R}^p$ is the vector of unknowns, $\boldsymbol{x} \in \mathbb{R}^d$ and $t > 0$ denote the position vector and the time. The function $\boldsymbol{F}_j$ is called the flux function and we denote by $\mathbb{A}_j = \mathbb{A}_j(\boldsymbol{U})$ its Jacobian matrix, defined as follows

$$\mathbb{A}_j = \nabla_{\boldsymbol{U}} \boldsymbol{F}_j(\boldsymbol{U}),$$

where $\nabla_{\boldsymbol{U}}$ denotes the gradient operator with respect to the components of $\boldsymbol{U}$. With this notation, the system (3.45) reads

$$\frac{\partial}{\partial t}\boldsymbol{U} + \sum_{j=1}^{d} \mathbb{A}_j(\boldsymbol{U}) \frac{\partial}{\partial x_j}\boldsymbol{U} = \boldsymbol{0}. \tag{3.46}$$

This equation corresponds to the non conservative form of the system (3.45). We are now in position to recall the definition of hyperbolicity. The system (3.45) is called hyperbolic if, for any $\boldsymbol{U} \in \mathcal{D}$ and any $\boldsymbol{\omega} \in \mathfrak{R}^d$, $\boldsymbol{\omega} \neq \boldsymbol{0}$, the matrix

$$\mathbb{A}(\boldsymbol{U}, \boldsymbol{\omega}) = \sum_{j=1}^{d} \omega_j \mathbb{A}_j(\boldsymbol{U}) \tag{3.47}$$

has p real eigenvalues and p linearly independent corresponding eigenvectors.
Now, let us show the links between the notions of hyperbolicity and plane-wave analysis. The plane-wave analysis consists in looking for a solution of system (3.45) which writes under the form $\boldsymbol{U} = \boldsymbol{U}(\boldsymbol{x} \cdot \boldsymbol{n} - at)$. Here, $\boldsymbol{n}$, is a unit vector of $\mathfrak{R}^d$ which characterizes the wave direction

and $a \in \mathfrak{R}$ is the wave speed. We introduce $\xi = \boldsymbol{x} \cdot \boldsymbol{n} - at$ and denote by $\boldsymbol{U}'$ the derivative of $\boldsymbol{U}$ with respect to ξ. Applying the chain rules leads to

$$\frac{\partial}{\partial t}\boldsymbol{U} = -a\boldsymbol{U}', \quad \frac{\partial}{\partial x_j}\boldsymbol{U} = n_j\boldsymbol{U}',$$

where n_j denotes the jth component of $\boldsymbol{n}$. Substituting the previous formulas into the system written under the non conservative form (3.46) leads to

$$-a\boldsymbol{U}' + \sum_{j=1}^{d} n_j \mathbb{A}_j(\boldsymbol{U})\boldsymbol{U}' = \boldsymbol{0}.$$

Utilizing the definition (3.47), we observe that this system rewrites

$$\left[\mathbb{A}(\boldsymbol{U}, \boldsymbol{n}) - a\mathbb{I}\right]\boldsymbol{U}' = \boldsymbol{0}.$$

This equation is nothing but the eigenvalue problem which corresponds to the study of the hyperbolicity of system (3.45). More precisely, the plane wave characterized by $(\boldsymbol{n}, a)$ exists if and only if a is a real eigenvalue of $\mathbb{A}$. This shows the links between the plane-wave analysis and the hyperbolic feature of a system of conservation laws.

Plane-wave analysis of the Euler equations

Before proceeding any further, assuming a sufficient smoothness of the flow variables, the non conservative form of the Euler equations writes

$$\frac{d}{dt}p + \rho c^2 \nabla \cdot \boldsymbol{V} = 0, \tag{3.48a}$$

$$\frac{d}{dt}\boldsymbol{V} + \frac{1}{\rho}\nabla p = \boldsymbol{0}, \tag{3.48b}$$

$$\frac{d}{dt}\eta = 0. \tag{3.48c}$$

This system has been derived from the non conservative system (3.40) replacing the total energy equation by the entropy equation (3.41) which is valid granted a sufficient smoothness of the flow variables. In addition, considering an equation of state written under the form $p = p(\rho, \eta)$ allows to express the density variation in terms of the pressure variation as follows $\mathrm{d}p = c^2\,\mathrm{d}\rho$, where c is the isentropic sound speed. Substituting this relation into the mass conservation equation (3.40a) yields the first equation of the above system.
Let $\varphi = \varphi(\boldsymbol{x}, t)$ be a generic flow variable, we are looking for plane-wave solutions of the above system, written under the form $\varphi = \varphi(\boldsymbol{x} \cdot \boldsymbol{n} - at)$. Here, $\boldsymbol{n}$ is a unit vector of $\mathfrak{R}^d$ which characterizes the direction of the wave and a is the wave speed. Setting $\xi = \boldsymbol{x} \cdot \boldsymbol{n} - at$, we denote by $\varphi'(\xi)$ the derivative of φ with respect to ξ. Now, using the previous notation and the chain rule derivative leads to the following useful formulas

$$\nabla P = P'\boldsymbol{n}, \tag{3.49a}$$

$$\nabla \cdot \boldsymbol{V} = \boldsymbol{V}' \cdot \boldsymbol{n}. \tag{3.49b}$$

Recalling that the material derivative reads $\frac{d}{dt}\varphi = \frac{\partial \varphi}{\partial t} + \boldsymbol{V} \cdot \nabla\varphi$ yields the practical formulas

$$\frac{d}{dt}P = -(a - \boldsymbol{V} \cdot \boldsymbol{n})P', \tag{3.50a}$$

$$\frac{d}{dt}\boldsymbol{V} = -(a - \boldsymbol{V} \cdot \boldsymbol{n})\boldsymbol{V}'. \tag{3.50b}$$

It is convenient for the subsequent computations to introduce $w = a - \boldsymbol{V}\cdot\boldsymbol{n}$ which is the relative speed of the wave with respect to the flow velocity projected onto the wave direction. Substituting the plane-wave form of the variables $(p, \boldsymbol{V}, \eta)$ into (3.48) yields

$$\begin{aligned} &- wp' + \rho c^2 \boldsymbol{V}'\cdot\boldsymbol{n} = 0,\\ &- w\boldsymbol{V}' + \frac{1}{\rho}p'\boldsymbol{n} = \boldsymbol{0},\\ &- w\eta' = 0. \end{aligned}$$

Let us introduce the vector $\boldsymbol{W} = (p, \boldsymbol{V}, \eta)^t$ of size $d+2$ and the $(d+2)\times(d+2)$ block matrix

$$\mathbb{A} = \begin{pmatrix} 0 & \rho c^2 \boldsymbol{n}^t & 0\\ \frac{1}{\rho}\boldsymbol{n} & \mathbb{0}_d & \boldsymbol{0}_d\\ 0 & 0 & 0 \end{pmatrix}, \tag{3.51}$$

where $\mathbb{0}_d$ denotes the square matrix of size d and $\boldsymbol{0}_d$ the vector of size d whose all entries are equal to zero. Employing this notation allows to rewrite the above system under the compact form

$$(\mathbb{A} - w\mathbb{I}_{d+2})\boldsymbol{U} = \boldsymbol{0}_{d+2},$$

where $\mathbb{I}_{d+2}$ is the unit matrix of size $d+2$. As expected, this shows that the plane-wave analysis amounts to investigate the eigenstructure of matrix $\mathbb{A}$. It remains to study this eigenstructure. To this end, we first observe that $\boldsymbol{W} = (0, \boldsymbol{0}_d, 1)$ satisfies $\mathbb{A}\boldsymbol{W} = \boldsymbol{0}_{d+2}$, thus $w = 0$ is an eigenvalue whose eigenvector writes $\boldsymbol{W} = (0, \boldsymbol{0}_d, 1)$. Secondly, let $\boldsymbol{\tau}$ be a vector of size d, then $\boldsymbol{W} = (0, \boldsymbol{\tau}, 0)$ satisfies $\mathbb{A}\boldsymbol{W} = (\rho c^2 \boldsymbol{n}\cdot\boldsymbol{\tau}, \boldsymbol{0}_d, 0)^t$. Choosing $\boldsymbol{\tau} \in \{\boldsymbol{n}\}^\perp$ yields $\mathbb{A}\boldsymbol{W} = \boldsymbol{0}_{d+2}$. Thus, $w = 0$ is once more an eigenvalue associated to the eigenspace $\{\boldsymbol{n}\}^\perp$ of dimension $d-1$.
Now, setting $\boldsymbol{W} = (1, \frac{1}{\rho c}\boldsymbol{n}, 0)^t$ leads to $\mathbb{A}\boldsymbol{W} = c(1, \frac{1}{\rho c}\boldsymbol{n}, 0)^t$. This means the $w = c$ is the eigenvalue related to the eigenvector $\boldsymbol{W} = (1, \frac{1}{\rho c}\boldsymbol{n}, 0)^t$. Similarly, $\boldsymbol{W} = (1, -\frac{1}{\rho c}\boldsymbol{n}, 0)^t$ is the eigenvector associated to the eigenvalue $w = -c$.
We are now in position to summarize the previous analysis. Matrix $\mathbb{A}$ admits the following eigenstructure

- 0 is an eigenvalue of multiplicity d and its corresponding eigenvectors are respectively $(0, \boldsymbol{0}_d, 1)^t$ (multiplicity 1) and $(0, \boldsymbol{\tau}, 0)^t$ (multiplicity $d-1$), where $\boldsymbol{\tau}$ belongs to $\{\boldsymbol{n}\}^\perp$;
- c is the eigenvalue corresponding to the eigenvector $(1, \frac{1}{\rho c}\boldsymbol{n}, 0)^t$;
- $-c$ is the eigenvalue corresponding to the eigenvector $(1, -\frac{1}{\rho c}\boldsymbol{n}, 0)^t$.

This shows that matrix $\mathbb{A}$ admits $d+2$ real eigenvalues, thus the system of unsteady compressible Euler equations is an hyperbolic system. The corresponding plane waves are respectively characterized by the wave speeds $a = \boldsymbol{V}\cdot\boldsymbol{n}$ for $w = 0$, $a = \boldsymbol{V}\cdot\boldsymbol{n} + c$ for $w = c$ and $a = \boldsymbol{V}\cdot\boldsymbol{n} - c$ for $w = -c$.
Let us point out that the mathematical structure of the steady Euler equations is more complex since the steady system is hyperbolic if and only if the flow is supersonic. More precisely, this system becomes elliptic in the subsonic regions. This change of mathematical structure renders quite difficult the design of numerical methods to solve directly the steady compressible Euler equations for both subsonic and supersonic fluid flows. In this case, it is easier to reach the steady state solution by employing a time marching algorithm. This strategy consists in solving the unsteady compressible Euler equations to reach the steady state, knowing that a lot of robust and accurate numerical methods are available for solving these hyperbolic equations.

Boundary conditions

Contrary to the Navier-Stokes equations, the Euler equations are only first-order in space partial differential equations. This change has an impact on the definition of the boundary conditions for the Euler equations. Let us pursue the discussion initiated in Section 3.1.6 for the Navier-Stokes equations.
Regarding the wall boundary conditions, they are prescribed as follows on the solid surface, Γ, contained in the computational domain

$$\boldsymbol{V}(\boldsymbol{x},t)\cdot\boldsymbol{n}=\boldsymbol{V}_\Gamma(\boldsymbol{x},t)\cdot\boldsymbol{n},\ \text{ for all }\ \boldsymbol{x}\in\Gamma \text{ and } t>0. \tag{3.52}$$

Here, $\boldsymbol{V}_\Gamma$ is the given solid wall velocity and $\boldsymbol{n}$ is the unit outward normal to the solid wall. This boundary condition is termed as a slip boundary condition, since it means that the flow does not cross the boundary but may move tangentially. For the energy equation, no boundary condition is required at the wall.
It remains to discuss the case of the artificial boundary conditions related to the boundaries of the computational domain where the fluid may leave or enter. In this case, a theoretical analysis which relies on the study of the sign of the eigenvalues of the Jacobian matrix of the Euler system, provides the number of boundary conditions to specify. More precisely, this allows to determine the number of incoming waves in the direction normal to the boundary. The interested reader may consult [52]. Hereafter, we summarize this study by claiming that

- For a supersonic inflow $d+2$ boundary conditions have to be prescribed;
- For a subsonic inflow $d+1$ boundary conditions have to be prescribed;
- For a supersonic outflow 0 boundary condition has to be prescribed;
- For a subsonic outflow 1 boundary condition has to be prescribed.

3.3 Construction of a Finite Volume method for the Euler equations

3.3.1 Godunov scheme

The Euler equations are a system of conservation laws. The Finite Volume methods were constructed to deal with conservation laws. Thus, the Finite Volume method is a natural choice when deciding to develop a numerical scheme for these equations. We integrate the equation (3.44) over a cell ω_c

$$\int_{\omega_c}\frac{\partial\boldsymbol{U}}{\partial t}ds+\int_{\omega_c}\nabla\cdot\boldsymbol{F}^e(\boldsymbol{U})ds=\boldsymbol{0}, \tag{3.53}$$

and we use the Green theorem to obtain

$$\frac{d}{dt}\int_{\omega_c}\boldsymbol{U}ds+\int_{\partial\omega_c}\boldsymbol{F}^e(\boldsymbol{U})\cdot\boldsymbol{n}dl=\boldsymbol{0}. \tag{3.54}$$

We introduce the mean cell conservative variables $\boldsymbol{U}_c$ defined by

$$\boldsymbol{U}_c=\frac{1}{w_c}\int_{\omega_c}\boldsymbol{U}ds,$$

with $w_c = \int_{\omega_c} ds$. We introduce $\mathcal{C}_f(c)$ the set of cells sharing the edge f with cell c. We note l_{cd} the length of the edge shared by cell c and cell d. We denote by $\boldsymbol{n}_{cd}$ the unit normal to this edge pointing from cell c to cell d. With these notations equation (3.54) writes

$$\frac{d}{dt} w_c \boldsymbol{U}_c + \sum_{d \in \mathcal{C}_f(c)} l_{cd} \hat{\boldsymbol{F}}^e(\boldsymbol{U}_c, \boldsymbol{U}_d) \cdot \boldsymbol{n}_{cd} = \boldsymbol{0}. \tag{3.55}$$

We have introduced the numerical flux $\hat{\boldsymbol{F}}^e(\boldsymbol{U}_c, \boldsymbol{U}_d)$ between cells c and d. It now remains to give the expression of this flux.

The equation (3.55) with the numerical flux $\hat{\boldsymbol{F}}^e(\boldsymbol{U}_c, \boldsymbol{U}_d)$ defined as the solution to the Riemann problem defined by the two states, $\boldsymbol{U}_c$ and $\boldsymbol{U}_d$, is called the Godunov scheme. It was first presented by Godunov [53] in 1959. In the next section we give the definition of the Riemann problem followed by the presentation of some approximate Riemann solvers.

3.3.2 Riemann problem for the one-dimensional Euler Equations

A Riemann problem is defined by a system of hyperbolic conservation laws subject to the simplest, non-trivial, initial conditions. For the one-dimensional Euler equations it writes

$$\frac{\partial \boldsymbol{U}}{\partial t} + \frac{\partial \boldsymbol{F}^e(\boldsymbol{U})}{\partial x} = \boldsymbol{0}, \tag{3.56}$$

with $\boldsymbol{U} = [\rho, \rho u, \rho E]^t$ and $\boldsymbol{F}^e(\boldsymbol{U}) = [\rho u, \rho u^2 + p, u(\rho E + p)]^t$, with u the velocity component along x axis. This equation is completed with the initial conditions

$$\boldsymbol{U}(x, 0) = \begin{cases} \boldsymbol{U}_L & \text{if } x < 0, \\ \boldsymbol{U}_R & \text{if } x > 0. \end{cases} \tag{3.57}$$

While the definition of this problem seems extremely simple, its solution contains the fundamental physics and mathematical properties of the system of equations. The solution of the equations with more complex initial conditions can be seen as a superposition of the solutions of local Riemann problem, this is how the Riemann problems are used in the Godunov scheme.

There is no exact expression for the solution to the Riemann problem for the Euler equations. However it is possible to construct numerical solutions to the Riemann problem, with the accuracy of your choice, using iterative procedures. Once again Godunov was the first to present a so called exact Riemann solver for the Euler equations [53]. A lot of work has been made to improve this solver, and we can cite for example the work of Toro [121], which is suitable for ideal gas and covolume gases. The key elements to take into account when designing an efficient exact Riemann solver are the variables used, the number of equations used, the iterative procedure technique and a method to avoid nonphysical values.

The development of exact Riemann solvers is important. It is useful to study the physical properties of the equations and to design numerical tests cases in order to assess the validity of numerical methods. However, an exact solver cannot be used within a Godunov scheme to compute the numerical flux for practical computations. The problem is twofold. First, the construction of the solution is expensive. Then, when used in a Godunov method only a small portion of the solution is in fact needed. It means that a huge computational effort is wasted in computing the solution. In practice, approximate Riemann solvers are used instead of exact solvers. This is the topic of the next section.

Comment 21: *The multidimensional evaluation of the numerical flux at cell interfaces is usually performed by solving exactly or approximately a one-dimensional Riemann problem in the direction normal to the interface. The left and right states are the states located on both sides of the interface.*

3.3.3 Approximate Riemann solvers

A lot of research has been done about this topic. When designing an approximate Riemann solver, assumptions have to be made in order to simplify the problem. Depending on the properties of the flow you want to study you can try to satisfy these properties in the development of your approximate Riemann solver. From these different assumptions a huge amount of solvers was created. For an exhaustive review on Riemann solvers the reader should refer to [120]. Other Riemann solvers with interesting properties were developed by Gallice in [47, 48]. In this thesis we restrict the presentation to the construction of four classical Riemann solvers, namely the Rusanov solver, the approximate solver of Roe, the HLL solver and the HLLC solver.

Rusanov solver

The Rusanov solver, also called local Lax-Friedrichs solver, is one of the simplest approximate Riemann solver. It writes

$$\boldsymbol{F}^{Rusanov}(\boldsymbol{U}_L, \boldsymbol{U}_R) = \frac{1}{2}\left(\boldsymbol{F}_L + \boldsymbol{F}_R\right) + \frac{|\lambda_m|}{2}\left(\boldsymbol{U}_L - \boldsymbol{U}_R\right),$$

where $\boldsymbol{F}_L = \boldsymbol{F}^e(\boldsymbol{U}_L)$ and $\boldsymbol{F}_R = \boldsymbol{F}^e(\boldsymbol{U}_R)$. We note $|\lambda_m|$ the absolute value of the largest eigenvalue of the linearised problem, namely,

$$|\lambda_m| = \max(|u_L| + c_L, |u_R| + c_R).$$

This numerical flux is very dissipative but is straightforward to implement. It can be useful in the early stages of the development of a CFD code. For these purposes, it is also worth mentioning the numerical scheme developed in [134], which is based on a hybridization of the Lax-Wendroff and the Lax-Friedrichs fluxes. This approach is more accurate than the simple Rusanov solver, while its implementation remains straightforward.

Roe solver

In [108] Roe described the well known Roe solver. The Roe solver is in fact an exact Riemann solver applied to the linearized Riemann problem. The linearized Riemann problem writes

$$\frac{\partial \boldsymbol{U}}{\partial t} + \hat{\mathbb{A}}(\boldsymbol{U}_L, \boldsymbol{U}_R)\frac{\partial \boldsymbol{U}}{\partial x} = 0, \tag{3.58}$$

where $\hat{\mathbb{A}}$ is the Roe averaged Jacobian matrix. The problem is completed with the initial condition

$$\boldsymbol{U}(x,0) = \begin{cases} \boldsymbol{U}_L & \text{if } x < 0 \\ \boldsymbol{U}_R & \text{if } x > 0. \end{cases} \tag{3.59}$$

The Roe averaged Jacobian matrix is required to satisfy the three following properties:

- $\hat{\mathbb{A}}$ is required to have real eigenvalues, we note them $\hat{\lambda}_i$. We can order them as $\hat{\lambda}_1 \leq \hat{\lambda}_2 \leq \cdots \leq \hat{\lambda}_n$. $\hat{\mathbb{A}}$ is also required to have a complete set of linearly independent left and right eigenvectors denoted $\boldsymbol{L}$ and $\boldsymbol{R}$ respectively. The matrix can be written under the form

  $$\hat{\mathbb{A}} = \mathbb{L}\mathbb{\Lambda}\mathbb{R}$$

  where $\mathbb{\Lambda} = \mathrm{diag}(\lambda_1, \lambda_2, \lambda_3)$.

- The consistency is ensured with the exact Jacobian

$$\hat{\mathbb{A}}(\boldsymbol{U},\boldsymbol{U}) = \mathbb{A}(\boldsymbol{U}).$$

- The conservation is ensured across discontinuities

$$\boldsymbol{F}_R - \boldsymbol{F}_L = \hat{\mathbb{A}}(\boldsymbol{U}_R - \boldsymbol{U}_L).$$

The Roe averaged Jacobian matrix can be constructed by different methods. For instance, one can use the classical Roe method [108] or the Roe-Pike method [107]. These methods leads to the definition of the Roe averaged quantities

$$\hat{\rho} = \sqrt{\rho_L \rho_R}, \tag{3.60}$$

$$\hat{u} = \frac{u_L\sqrt{\rho_L} + u_R\sqrt{\rho_R}}{\sqrt{\rho_L} + \sqrt{\rho_R}}, \tag{3.61}$$

$$\hat{v} = \frac{v_L\sqrt{\rho_L} + v_R\sqrt{\rho_R}}{\sqrt{\rho_L} + \sqrt{\rho_R}}, \tag{3.62}$$

$$\hat{H} = \frac{H_L\sqrt{\rho_L} + H_R\sqrt{\rho_R}}{\sqrt{\rho_L} + \sqrt{\rho_R}}, \tag{3.63}$$

$$\hat{c} = \sqrt{(\gamma-1)\left(\hat{H} - \frac{\hat{u}^2+\hat{v}^2}{2}\right)}. \tag{3.64}$$

Finally the Roe flux writes

$$\boldsymbol{F}^{Roe}(\boldsymbol{U}_L,\boldsymbol{U}_R) = \frac{1}{2}\left(\boldsymbol{F}_L + \boldsymbol{F}_R\right) - \frac{1}{2}\mathbb{R}\hat{\mathbb{\Lambda}}\mathbb{L}\left(\boldsymbol{U}_R - \boldsymbol{U}_L\right),$$

where $\hat{\mathbb{\Lambda}} = \mathrm{diag}(|\lambda_1|,|\lambda_2|,|\lambda_3|) = \mathrm{diag}(|\hat{u}-\hat{c}|,|\hat{u}|,|\hat{u}+\hat{c}|)$. We have $\mathbb{R}$ the right eigenvector of matrix $\hat{\mathbb{A}}$ and $\mathbb{L} = \mathbb{R}^{-1}$.
The linearized Riemann problem solutions obtained consists of discontinuous jumps only. This is a correct approximation for contact discontinuities and shocks. For the rarefaction waves on the other hand the approximation can lead to nonphysical phenomenons called rarefaction shocks. To avoid having this problem we need to add an entropy fix to the Roe approximate solver. This entropy fix can be introduced by modifying the modulus of the eigenvalue of the nonlinear fields [57] as follows

$$|\lambda_k|^* = \begin{cases} |\lambda_k|, & \text{if } |\lambda_k| \geq \delta, \\ \frac{1}{2\delta}(|\lambda_k|^2 + \delta^2), & \text{if } |\lambda_k| < \delta, \end{cases} \tag{3.65}$$

where $\delta = 0.2$ for instance and $k = 1$ or $k = 3$.

HLL solver

Harten, Lax and van Leer described the HLL solver in [59]. The construction of this solver is based on three basic assumptions:

- The possible wave patterns are represented by a single wave pattern,
- The only waves represented are the left and right non-linear waves,
- Estimates of S_L and S_R representing these wave speeds are available.

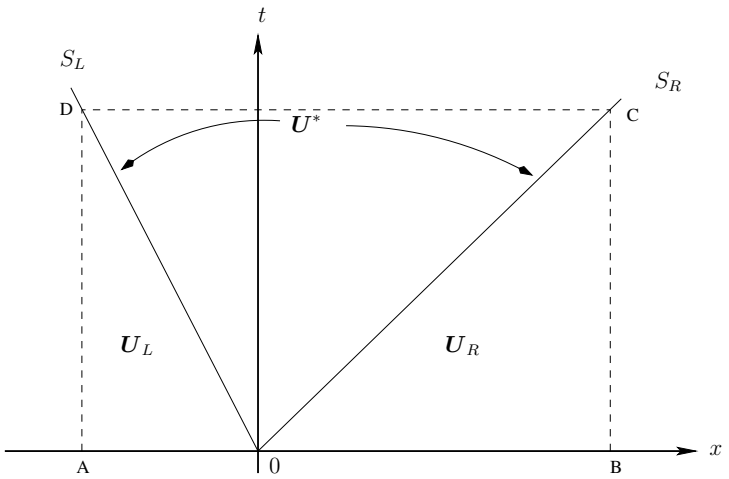


Figure 3.1: Structure of the solution of the Riemann problem in the x-t plane for the HLL approximate Riemann solver.

The structure of the approximate solver obtain is presented in Figure 3.1. We note that the contact discontinuity is not present in this representation of the solution, the whole star region is modelled by a unique constant state $\boldsymbol{U}^*$. The numerical flux is then determined by

$$\boldsymbol{F}(\boldsymbol{U}_L, \boldsymbol{U}_R)^{HLL} = \begin{cases} \boldsymbol{F}(\boldsymbol{U}_L), & 0 \leq S_L \\ \boldsymbol{F}^*, & S_L \leq 0 \leq S_R \\ \boldsymbol{F}(\boldsymbol{U}_R), & S_R \geq 0. \end{cases}$$

To obtain the values of $\boldsymbol{U}^*$ and $\boldsymbol{F}^*$ we see that equation (3.56) recasts to

$$\oint \boldsymbol{U} dx - \boldsymbol{F}^e(\boldsymbol{U}) dt = \boldsymbol{0}. \tag{3.66}$$

Evaluating the integral (3.66) for the rectangle ABCD depicted in Figure 3.1 yields

$$\boldsymbol{U}^* = \frac{S_R \boldsymbol{U}_R - S_L \boldsymbol{U}_L - (\boldsymbol{F}_R - \boldsymbol{F}_L)}{S_R - S_L}, \tag{3.67}$$

with $\boldsymbol{F}_L = \boldsymbol{F}^e(\boldsymbol{U}_L)$ and $\boldsymbol{F}_R = \boldsymbol{F}^e(\boldsymbol{U}_R)$. We can also compute directly the numerical flux $\boldsymbol{F}^*$ as

$$\boldsymbol{F}^* = \frac{S_R \boldsymbol{F}_L - S_L \boldsymbol{F}_R + S_L S_R (\boldsymbol{U}_R - \boldsymbol{U}_L)}{S_R - S_L}. \tag{3.68}$$

To conclude the description of this numerical flux, we need to talk about the wave speed estimates S_L and S_R. Once again an impressive amount of literature deals with this particular topic, refer to [43] and [124] for instance.
A simple choice for the wave speed estimates is to take

$$\begin{cases} S_L = \min\left(u_L - c_L, u_R - c_R\right), \\ S_R = \max\left(u_L + c_L, u_R + c_R\right). \end{cases}$$

An other classical choice is to use the Roe averaged states $\hat{u}$ and $\hat{c}$ defined earlier as follows

$$\begin{cases} S_L = \min\left(u_L - c_L, \hat{u} - \hat{c}\right), \\ S_R = \max\left(\hat{u} + \hat{c}, u_R + c_R\right). \end{cases}$$

HLLC solver

Toro et al. developed the HLLC solver in [124], it is an extension of the HLL solver where the contact discontinuity is also resolved. The structure of the solution, presented in Figure 3.2, is composed of four distinct constant states separated by three different waves. The additional wave speed is denoted by S_*. The numerical flux is determined by

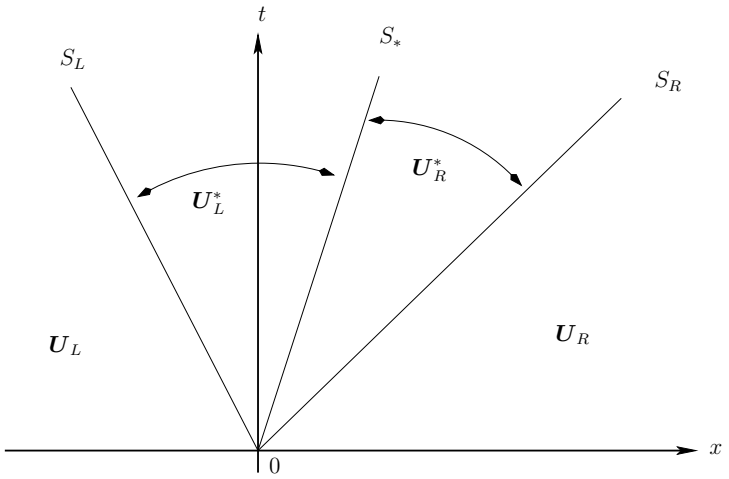


Figure 3.2: Structure of the solution of the Riemann problem in the x-t plane for the HLLC approximate Riemann solver.

$$\boldsymbol{F}(\boldsymbol{U}_L, \boldsymbol{U}_R)^{HLLC} = \begin{cases} \boldsymbol{F}(\boldsymbol{U}_L), & 0 \leq S_L, \\ \boldsymbol{F}_L^*, & S_L \leq 0 \leq S_*, \\ \boldsymbol{F}_R^*, & S_* \leq 0 \leq S_R, \\ \boldsymbol{F}(\boldsymbol{U}_R), & S_R \geq 0, \end{cases}$$

where

$$\boldsymbol{F}_L^* = \boldsymbol{F}_L + S_L(\boldsymbol{U}_L^* - \boldsymbol{U}_L),$$

and

$$\boldsymbol{F}_R^* = \boldsymbol{F}_R + S_R(\boldsymbol{U}_R^* - \boldsymbol{U}_R).$$

Once again the values of the different unknowns are obtained using equation (3.66). Different choices for the wavespeeds S_L and S_R are discussed in [25]. The intermediate speed S_* is computed in terms of S_L and S_R as

$$S_* = \frac{p_R - p_L + \rho_L u_L(S_L - u_L) - \rho_R u_R(S_R - u_R)}{\rho_L(S_L - u_L) - \rho_R(S_R - u_R)}$$

The intermediate states $\boldsymbol{U}_K^*$ with $K = L$ or $K = R$ are given by

$$\boldsymbol{U}_K^* = \rho_K \left(\frac{S_K - u_k}{S_K - S_*}\right) \begin{bmatrix} 1 \\ S_* \\ E_K + (S_* - u_K)\left(S_* + \frac{p_K}{\rho_K(S_K - u_K)}\right) \end{bmatrix}. \tag{3.69}$$

3.3.4 Extension to higher-order

The numerical method we have presented is only first-order accurate, which is clearly not enough for real life applications. Many different methods were developed to build higher-order Finite Volume schemes. We can cite for instance the ENO method [58] which stands for Essentially Non Oscillatory, and the improved WENO method [87, 75] which stands for Weighted ENO. These methods are well suited for structured meshes, but, due to the use of a large stencil, their implementation is not straightforward on unstructured meshes. Furthermore, the parallelization of these methods leads to an increase in the number of communications which decreases the parallel efficiency. We can also mention the ADER methodology [125] which involves the resolution of Generalized Riemann Problems [120]. Finally we can cite the Weighted Average Flux (WAF) method [122, 123] where the intercell flux is represented by an integral average of the physical flux across the full structure of the solution of a local Riemann problem. In this thesis we use the MUSCL method which we describe in the next paragraph.

The MUSCL approach

The MUSCL method which stands for Monotone Upwind Scheme for Conservation Laws, was first introduced by van Leer in 1976 [128] and is well explained by Barth [21] and Leveque [82]. It consists of constructing a linear representation of the unknowns and using the reconstructed states in the Riemann solvers. When constructing the linear states we have to take care of not creating artificial nonphysical extrema. To do so we introduce the notion of slope limiter, which prevents the creation of nonphysical states.

We recall that we use the following constant approximation of the solution

$$\boldsymbol{U}_c = \frac{1}{w_c} \int_{\omega_c} \boldsymbol{U} ds. \tag{3.70}$$

From these constant states we are going to build a linear reconstruction. To do so we define the approximation of the gradient in cell c as $(\nabla \boldsymbol{U})_c$, it is a constant tensor in each cell. Using this discrete gradient we can then build the linear reconstruction with

$$\tilde{\boldsymbol{U}}_c(\boldsymbol{x}) = \boldsymbol{U}_c + (\nabla \boldsymbol{U})_c(\boldsymbol{x} - \boldsymbol{x}_c), \tag{3.71}$$

where $\boldsymbol{x}_c$ is the gravity center of cell ω_c. Using this definition we have

$$\frac{1}{w_c} \int_{\omega_c} \tilde{\boldsymbol{U}}_c(\boldsymbol{x}) ds = \frac{1}{w_c} \int_{\omega_c} \boldsymbol{U}(\boldsymbol{x}) ds = \boldsymbol{U}_c. \tag{3.72}$$

It means that the MUSCL approach conserves the mean values of the unknowns. In order to use this MUSCL approach we need to define a procedure to build the gradient tensors. We do this by using a least squares method.

Least squares method

In order to build the gradient tensor $(\nabla \boldsymbol{U})_c$ in the cell ω_c we use the value of the unknown $\boldsymbol{U}_c$ and the values of the unknowns $\boldsymbol{U}_d$ belonging to the neighbors of cell ω_c. First we need to define the neighborhood of cell c. There is many different ways of defining this, we chose here to match the neighborhood defined by the Godunov scheme. Namely $\mathfrak{N}(c)$ is the set of cells sharing an edge with cell c. Then, we can define the gradient tensor by ensuring that for all cell $d \in \mathfrak{N}(c)$

$$\tilde{\boldsymbol{U}}_c(\boldsymbol{x}_d) = \boldsymbol{U}_d, \tag{3.73}$$

the problem is that this system is overdeterminated. We introduce U_c and U_d as respectively $\boldsymbol{U}_c$ and $\boldsymbol{U}_d$ taken component by component. $(\nabla U)_c$ then represent the gradient of $\boldsymbol{U}$ in cell c componentwise. With these notations, we can satisfy (3.73) in a least squares sense by minimizing the function

$$I_c = \sum_{d\in\mathfrak{N}(c)} \frac{1}{2}(\tilde{U}_c(\boldsymbol{x}_d) - U_d)^2 = \sum_{d\in\mathfrak{N}(c)} \frac{1}{2}\left[U_c - U_d + (\nabla U)_c(\boldsymbol{x}_d - \boldsymbol{x}_c)\right]^2. \tag{3.74}$$

A straightforward computation shows that the solution to this problem writes

$$(\nabla U)_c = \mathbb{M}_c^{-1} \sum_{d\in\mathfrak{N}(c)} (\boldsymbol{x}_d - \boldsymbol{x}_c)(U_c - U_d), \tag{3.75}$$

where $\mathbb{M}_c$ is the 2×2 matrix defined as

$$\mathbb{M}_c = \sum_{d\in\mathfrak{N}(c)} (\boldsymbol{x}_d - \boldsymbol{x}_c)\otimes(\boldsymbol{x}_d - \boldsymbol{x}_c).$$

It is a positive definite and thus invertible matrix, which means that the tensor $(\nabla \boldsymbol{U})_c$ is well defined. The interesting features of the least squares procedure is that it is valid for any type of cells and moreover it preserves the linear fields.

Slope limiters

When constructing the states we can create new non-physical extrema. To avoid this, we need to introduce slope limiters. In each cell we compute the quantity ϕ_c which will limit the slope. The reconstructed value (3.71) becomes

$$\tilde{U}_c(\boldsymbol{x}) = U_c + \phi_c(\nabla U)_c(\boldsymbol{x} - \boldsymbol{x}_c). \tag{3.76}$$

In order to not create extremum we want for all $p\in\mathcal{P}(c)$ to have

$$\begin{cases} \tilde{U}_c(\boldsymbol{x}_p) & < U_c^{\max} = \max\left(\max_{d\in\mathfrak{N}(c)}(U_d), U_c\right), \\ \tilde{U}_c(\boldsymbol{x}_p) & > U_c^{\min} = \min\left(\min_{d\in\mathfrak{N}(c)}(U_d), U_c\right). \end{cases} \tag{3.77}$$

Thanks to this formula, we can define the slope limiter as

$$\phi_c = \min_{p\in\mathcal{P}(c)} \phi_{c,p}, \tag{3.78}$$

knowing that

$$\phi_{c,p} = \begin{cases} \mu\left(\frac{U_c^{\max} - U_c}{\tilde{U}_c(\boldsymbol{x}_p) - U_c}\right) & \text{if } \tilde{U}_c(\boldsymbol{x}_p) - U_c > 0, \\ \mu\left(\frac{U_c^{\min} - U_c}{\tilde{U}_c(\boldsymbol{x}_p) - U_c}\right) & \text{if } \tilde{U}_c(\boldsymbol{x}_p) - U_c < 0, \\ 1 & \text{if } \tilde{U}_c(\boldsymbol{x}_p) - U_c = 0. \end{cases} \tag{3.79}$$

Here, μ denotes a real valued function characterizing the limiter. In the following we define three limiters and give their associated μ function. The representation of these functions is presented in Figure 3.3.

In [27] Berger makes an analysis of the properties of different slope limiters on irregular grids. Barth and Jespersen [22] demonstrated the use of limited-reconstruction on unstructured grids for the solution of the Euler equations. They defined $\mu(x) = \min(1, x)$. In practice their limiter causes a degradation in the convergence performance because of its non-differentiability.

Venkatakrishnan [130] modified the limiter to be differentiable and managed to obtain an efficient convergence to steady state. He defined $\mu(x) = \frac{x^2+2x}{x^2+x+2}$.

Michalak [92] pointed out that the Venkatakrishnan's limiter has the drawback of reducing the gradient in regions with no extrema. Hence he developed a new limiter, monotone and differentiable, that can achieve good convergence properties while maintaining high-order accuracy in smooth regions where no extrema exist. He proposed the following definition of $\mu(x) = \widetilde{\min}(1, x)$ with

$$\widetilde{\min}(1, x) = \begin{cases} P(x) & \text{if } x < x_t, \\ 1 & \text{if } x \geq x_t, \end{cases}$$

where $1 < x_t < 2$ is a threshold and $P(x)$ is the cubic polynomial satisfying

$$\begin{array}{ll} P|_0 = 0 & P|_{x_t} = 1 \\ \frac{dP}{dx}|_0 = 1 & \frac{dP}{dx}|_{x_t} = 0 \end{array}$$

The choice of the threshold x_t is a compromise between maintaining accuracy near extrema, which is obtained with $x_t < 2$, and maintaining good convergence properties, which is easier with larger values of x_t. In [92] the authors used the value $x_t = 1.5$, we follow their choice in our work.

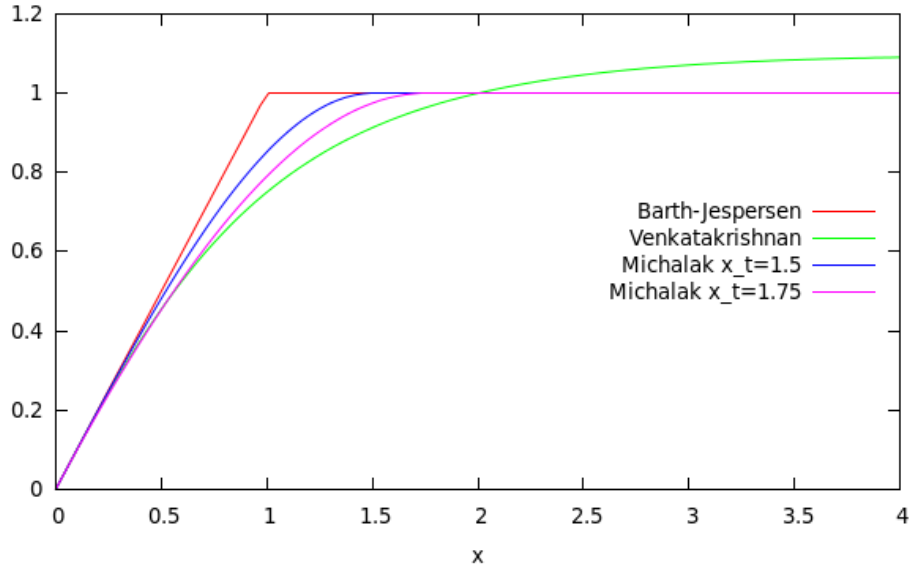


Figure 3.3: Representation of the limiter function μ, for the limiters of Barth-Jespersen, Venkatakrishnan and Michalak.

3.3.5 Time discretization

To complete the description of our scheme it remains to deal with the time discretization. Let us start by giving the latest expression of the semi-discrete scheme (3.55) when using the MUSCL method.

$$\frac{d}{dt} w_c \boldsymbol{U}_c + \sum_{d \in \mathcal{C}_f(c)} l_{cd} \hat{\boldsymbol{F}}^e(\tilde{\boldsymbol{U}}_c^d, \tilde{\boldsymbol{U}}_d^c) \cdot \boldsymbol{n}_{cd} = \boldsymbol{0}, \tag{3.80}$$

where $\tilde{\boldsymbol{U}}_c^d$ stands for the limited reconstructed state of the cell c at the interface between cells c and d. $\tilde{\boldsymbol{U}}_c^d$ stands for the limited reconstructed state of the cell d at the interface between cells c and d.

We can define the residual of cell ω_c

$$\boldsymbol{R}_c(\boldsymbol{\mathcal{U}}) = \sum_{d \in \mathcal{C}_f(c)} l_{cd} \hat{\boldsymbol{F}}^e(\tilde{\boldsymbol{U}}_c^d, \tilde{\boldsymbol{U}}_d^c) \cdot \boldsymbol{n}_{cd}, \tag{3.81}$$

where $\boldsymbol{\mathcal{U}}$ is the block-vector of size $\mathfrak{C}_{\mathcal{D}}$ defined by $\boldsymbol{\mathcal{U}}_c = \boldsymbol{U}_c$. Using these notations equation (3.80) yields

$$\frac{d}{dt} w_c \boldsymbol{U}_c = -\boldsymbol{R}_c(\boldsymbol{\mathcal{U}}). \tag{3.82}$$

Explicit scheme

When dealing with a second order space discretization as the one described in (3.80), it is common practice to develop a second order time discretization. To do so we develop an explicit Runge-Kutta method of order two (RK2), also called predictor-corrector method.

The RK2 method is a two step method, the first step writes

$$\boldsymbol{U}_c^* = \boldsymbol{U}_c^n - \frac{\Delta t}{2 w_c} \boldsymbol{R}_c(\boldsymbol{\mathcal{U}}^n), \tag{3.83}$$

where $\boldsymbol{U}_c^n$ is the variable taken at time t^n, $\Delta t = t^{n+1} - t^n$ is the time step, and $\boldsymbol{U}_c^*$ represent the value of the variables in cell ω_c at the intermediate time step $t^{n+\frac{1}{2}}$. The solution at time t^{n+1} can then be updated by applying the second step

$$\boldsymbol{U}_c^{n+1} = \boldsymbol{U}_c^n - \frac{\Delta t}{w_c} \boldsymbol{R}_c(\boldsymbol{\mathcal{U}}^*). \tag{3.84}$$

We need to append to this scheme a CFL condition in order keep it stable. For the Euler equations this condition depends on the value of the largest eigenvalue of the system $|u| + c$ and the size of the mesh.

$$\Delta t \leq \min_{1 \leq d \leq \mathfrak{C}_{\mathcal{D}}} \left(\frac{r_d}{\|\boldsymbol{V}_d\| + c_d} \right), \tag{3.85}$$

where r_d is the radius of the cell ω_d. $\boldsymbol{V}_d$ and c_d are respectively the velocity of the fluid and the speed of sound in cell ω_d. This formula is in practice used under the form

$$\Delta t = CFL \min_{1 \leq d \leq \mathfrak{C}_{\mathcal{D}}} \left(\frac{r_d}{\|\boldsymbol{V}_d\| + c_d} \right), \tag{3.86}$$

where $0 < CFL \leq 1$ is the CFL number.

Comment 22: *In our applications we need to solve the Euler equations to reach a steady state. We recall that the flows we are studying are hypersonic. If we use an explicit scheme to solve our problem we are limited by a time step bounded by a very restrictive CFL property* (3.85)*. Reaching steady state with this kind of approaches requires solving a lot of time steps. The solution to this problem is to use an implicit scheme, which is the topic of the next paragraph.*

Implicit scheme

We apply the simple backward Euler methodology to equation (3.82), it yields

$$\frac{w_c^{n+1}\boldsymbol{U}_c^{n+1} - w_c^n \boldsymbol{U}_c^n}{\Delta t} = -\boldsymbol{R}_c(\boldsymbol{\mathcal{U}}^{n+1}). \tag{3.87}$$

Since the mesh remains the same during the computation, the volume of the cells does not evolve, we have $w_c^{n+1} = w_c^n = w_c$. $\boldsymbol{R}_c(\boldsymbol{\mathcal{U}}^{n+1})$ is the residual in cell c taken at time t^{n+1}. We introduce $\boldsymbol{\delta U}_c = \boldsymbol{U}_c^{n+1} - \boldsymbol{U}_c^n$, equation (3.87) yields

$$\frac{w_c}{\Delta t^n}\boldsymbol{\delta U}_c = -\boldsymbol{R}_c(\boldsymbol{\mathcal{U}}^{n+1}). \tag{3.88}$$

We have to find an expression of the residual at time t^{n+1}. To do so we remark that $\boldsymbol{U}_c^{n+1} = \boldsymbol{U}_c^n + \boldsymbol{\delta U}_c$. We can write a truncated Taylor expansion of the residual which writes

$$\boldsymbol{R}_c(\boldsymbol{\mathcal{U}}^{n+1}) = \boldsymbol{R}_c(\boldsymbol{\mathcal{U}}^n) + \sum_{1\leq d\leq \mathfrak{C}_\mathcal{D}} \frac{\partial \boldsymbol{R}_c}{\partial \boldsymbol{U}_d}\boldsymbol{\delta U}_d + \mathcal{O}\left(\Delta t^2\right), \tag{3.89}$$

where $\frac{\partial \boldsymbol{R}_c}{\partial \boldsymbol{U}_d}$ are the Jacobians of the numerical flux. Equation (3.88) yields

$$\left(\frac{\mathbb{V}}{\Delta t^n} + \mathbb{E}^n\right)\boldsymbol{\delta \mathcal{U}} = -\boldsymbol{\mathcal{R}}^n, \tag{3.90}$$

where $\boldsymbol{\delta \mathcal{U}}$ is the global block-vector of size $\mathfrak{C}_\mathcal{D}$ defined by $\boldsymbol{\delta \mathcal{U}}_c = \boldsymbol{\delta U}_c$. $\boldsymbol{\mathcal{R}}^n$ is the global block-vector of size $\mathfrak{C}_\mathcal{D}$ containing the residuals. It is defined by $\boldsymbol{\mathcal{R}}_c^n = \boldsymbol{R}_c(\boldsymbol{\mathcal{U}}^n)$. Finally $\mathbb{V}$ is the diagonal block-matrix containing the volumes of the cells, it is defined by $\mathbb{V}_{cc} = w_c\mathbb{I}$, and $\mathbb{E}^n$ is the block-matrix containing the Jacobians of the numerical flux computed at time t^n.

In [26], Batten et al., present a detailed version of an exact Jacobian for the HLLC fluxes. They show that with this method they can achieve an efficient convergence rate. The method they developed is quite complicated and it is not straightforward to extend it to all the numerical fluxes.

In practice we seek to find an approximate version of the Jacobian matrix. We want to solve (3.90) to steady state, which means we want to find $\boldsymbol{\delta \mathcal{U}} \to \mathbf{0}$, which is achieved if $\boldsymbol{\mathcal{R}}(\boldsymbol{\mathcal{U}}^n) \to \mathbf{0}$. The first approximation that is usually made is to compute the Jacobian of the first-order residuals, even if we use an higher-order approximation. In this thesis, we also make the approximation of computing the Jacobian of the residuals considering that the simple Rusanov flux is used, even if we use a more precise solver.

We recall the definition of the Rusanov flux

$$\boldsymbol{F}^{Rusanov} = \frac{1}{2}\left(\boldsymbol{F}_L + \boldsymbol{F}_R\right) + \frac{|\lambda_m^{LR}|}{2}\left(\boldsymbol{U}_L - \boldsymbol{U}_R\right),$$

where $|\lambda_m^{LR}|$ is the absolute value of the largest eigenvalue of the linearised problem at the interface between the states L and R. We then use a truncated Taylor expansion to write

$$\boldsymbol{F}_K^{n+1} = \boldsymbol{F}_K^n + \frac{\partial \boldsymbol{F}^e}{\partial \boldsymbol{U}}|_K \delta \boldsymbol{U}_K + \mathcal{O}\left(\Delta t^2\right). \tag{3.91}$$

$\boldsymbol{F}_K^{n+1}$ represent the flux in cell K at time t^{n+1} and $\boldsymbol{F}_K^n$ represent the flux at time t^n. $\frac{\partial \boldsymbol{F}^e}{\partial \boldsymbol{U}}|_K$ is the exact Jacobian of the Euler equations estimated at $\boldsymbol{U}_K$.

The implicit Rusanov flux writes

$$\boldsymbol{F}^{n+1}(\boldsymbol{U}_L,\boldsymbol{U}_R) = \boldsymbol{F}^{n}(\boldsymbol{U}_L,\boldsymbol{U}_R) + \frac{1}{2}\left(\frac{\partial \boldsymbol{F}^e}{\partial \boldsymbol{U}}|_L \delta\boldsymbol{U}_L + \frac{\partial \boldsymbol{F}^e}{\partial \boldsymbol{U}}|_R \delta\boldsymbol{U}_R\right) + \frac{|\lambda_m^{LR}|}{2}\left(\delta\boldsymbol{U}_L - \delta\boldsymbol{U}_R\right). \quad (3.92)$$

From this equation we can deduce the expression of $\mathbb{E}^n$. The diagonal block at cell c writes

$$\mathbb{E}^n_{cc} = \frac{1}{2}\sum_{d\in\mathcal{C}_f(c)} l_{cd}\left(\frac{\partial \boldsymbol{F}}{\partial \boldsymbol{U}}|_c + |\lambda_m^{cd}|\mathbb{I}\right). \quad (3.93)$$

The extra diagonal blocks for the line c writes for all $d \in \mathcal{C}_f(c)$

$$\mathbb{E}^n_{cd} = \frac{1}{2} l_{cd}\left(\frac{\partial \boldsymbol{F}}{\partial \boldsymbol{U}}|_d - |\lambda_m^{cd}|\mathbb{I}\right). \quad (3.94)$$

Backward Euler method is unconditionally stable, we are not constrained by small time steps bounded with a restrictive CFL condition anymore. However, due to the use of an approximate Jacobian we cannot use arbitrary large time steps until the convergence is "well enough" established. This is even more true at the beginning of a computation, where we usually start from a constant state, some nonphysical phenomenons can occurs if the time steps are too important [26]. This is of course not only due to the approximation of the Jacobian, non-linear unsteady effects takes place during the transition to steady state, and are not very well handled for large time steps.

To converge to steady state, we use a CFL ramping technique. We start the computation by specifying a small CFL number, usually taken around 0.5, and increase it during a certain number of iterations. We define the starting CFL number CFL_s and a target CFL number CFL_t. The CFL number grows to the target in k_t iterations using the formula

$$CFL = \min\left(1, \frac{k}{k_t}\right) CFL_t + \left(1 - \min\left(1, \frac{k}{k_t}\right)\right) CFL_s,$$

where k represent the iteration number.

Comment 23: *This very simple CFL ramping technique may not be optimal in order to reach steady-state. Furthermore the choice of the different parameters is test case and mesh dependent. As an improvement we should consider using a more advanced CFL evolution technique such as the switched evolution relaxation (SER) method [94] for instance. This method increases the CFL inversely to the residual norm reduction and is supposed to increase the convergence speed.*

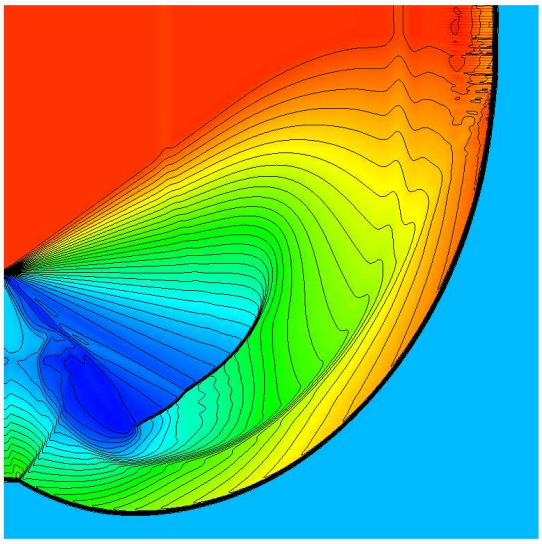
3.3.6 The Carbuncle Phenomenon: Causes and Cure

A non-physical phenomenon

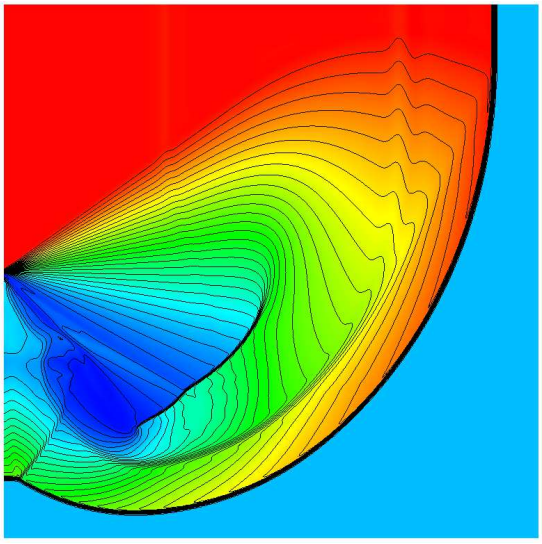
In [103] Quirk exposed the failures of classical approximate Riemann solvers. Theses failures are commonly named carbuncles and occurs mainly when dealing with supersonic computations. Due to the kind of applications we want to model, it is important to talk about this phenomenon in this thesis. The carbuncle phenomenon can appears in different situations.

In Figure 3.4 we consider the diffraction of a shock around a 90° corner, refer to section 3.3.7 for details. The mesh used in this computation is a Cartesian 400×400 mesh and we observe

Figure 3.4: Carbuncle in the shock diffraction test at t=0.18.



(a) Contours of density for the Shock Diffraction test case using HLLC solver. The carbuncle phenomenon appears in the top right.



(b) Contours of density for the Shock Diffraction test case using HLL solver. There is no carbuncle in that case.

that the Carbuncle appears along the shock when the HLLC solver is used. It does not form when using the more dissipative HLL solver.

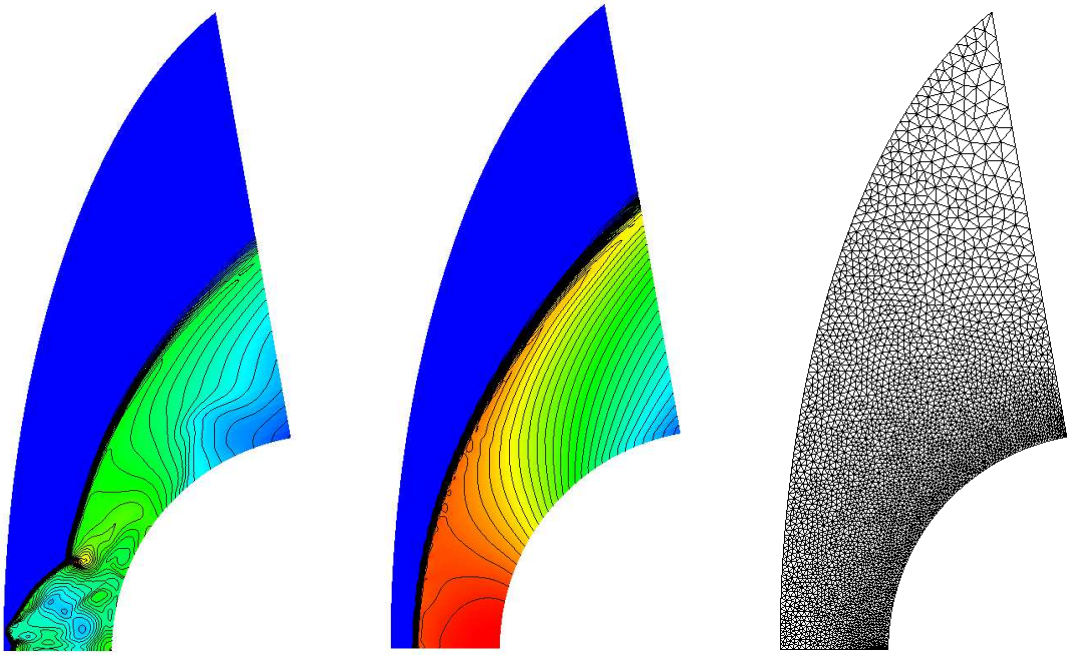
The Carbuncle phenomenon also appears when dealing with a supersonic flow around a cylinder at $M = 8$, as pictured in Figure 3.5. It forms in front of the cylinder and then pollutes the whole computational domain. This phenomenon is highly unstable and the scheme is unable to converge when using a solver that creates a carbuncle. Once again we can remark that the more diffusive HLL solver does not produces a carbuncle while the HLLC solver produces it.

The Carbuncle phenomenon is highly mesh dependent. The problem is that it appears when refining the mesh, refer to [99] for instance. Moreover, we are facing a dilemma here, when we use an accurate approximate Riemann solver capable of capturing the whole set of waves of the problem it may creates some nonphysical phenomenons. If we use a more dissipative solver, the carbuncles are not produced, but the physics we solve is also less precise. In the next section we present a possible cure to this problem, combining the properties of these two kinds of solvers.

A possible cure

In [97] Nishikawa introduces an approximate Riemann solver that have an excellent boundary-layer-resolving capability and does not produces carbuncles. The authors observed that diffusive solvers such as the two-waves HLL solver do not produce carbuncles but are too much dissipative in the rest of the domain. On the other side full-wave solvers such as HLLC or Roe solvers gives an excellent resolution of the flow but can produce carbuncles near shocks. To combine the good properties of these two kinds of solvers they used the technique of rotated Riemann solver, introduced by [40] and [83]. These solvers adaptively select a direction suitable for upwinding and applies a Riemann solver along that direction. Therefore they are able to capture multidimensional flow features very accurately using one-dimensional physic. Originally, this approach was proposed to better resolve shocks and shear layers in [40] and [83], but the gain

Figure 3.5: Carbuncle around a cylinder at Mach 8



(a) Contours of density for the cylinder at Mach 8 using the HLLC solver. The carbuncle phenomenon pollutes the solution in all the domain.

(b) Contours of density for the cylinder at Mach 8 using the HLL solver. There is no carbuncle.

(c) Unstructured mesh used for the computations. It is composed of 7257 triangles and 3757 Nodes.

was very limited when it was used with second-order methods. In his work, Ren [104] applied this approach to build a robust shock-capturing scheme. He applied the Roe solver on two directions, one aligned with the velocity difference vector, which is normal to shocks, and the second orthogonal to that direction. This rotated flux demonstrates a robust shock-capturing capability and was shown to suppress the carbuncle phenomena by an extra dissipation introduced by the rotated flux mechanism. In [97] Nishikawa applies two different Riemann solvers in the two directions. He applies a carbuncle-free flux function along to the velocity difference vector direction. In the other direction, he employs the Roe solver to prevent the resulting flux from being too dissipative. Furthermore when the velocity difference vector is too small the main direction is defined as being aligned with the cell-tangent instead of the cell-normal which is usually chosen. With this modified definition, the Roe flux is activated for smoothly varying flows, instead of the more dissipative solvers, and thus the accuracy is improved. Let us describe how this rotated solver is obtained.

First, we note $\boldsymbol{n}$ the unit outward normal to the edge where we are deriving the flux. We note $\boldsymbol{n}_\perp$ a unit vector perpendicular to $\boldsymbol{n}$. We define $\Delta \boldsymbol{q} = \boldsymbol{V}_R - \boldsymbol{V}_L$ the velocity difference vector. As we said earlier we can define the main direction of our rotated solver using this vector. We note $\boldsymbol{n}_1$ the unit vector defining this direction, it is defined by

$$\boldsymbol{n_1} = \begin{cases} \frac{\Delta \boldsymbol{q}}{\|\Delta \boldsymbol{q}\|} & \text{if } \|\Delta \boldsymbol{q}\| > \epsilon, \\ \boldsymbol{n}_\perp & \text{otherwise.} \end{cases} \tag{3.95}$$

The parameter ϵ is a small number, it can be adjusted to define what is a smoothly varying flow. Then we define the unit vector $\boldsymbol{n}_2$ perpendicular to $\boldsymbol{n}_1$, i.e.,

$$\boldsymbol{n}_1 \cdot \boldsymbol{n}_2 = 0,$$

and $\|\boldsymbol{n}_1\| = \|\boldsymbol{n}_2\| = 1$. We project the cell-edge normal $\boldsymbol{n}$ on these two vectors to obtain

$$\boldsymbol{n} = \alpha_1 \boldsymbol{n}_1 + \alpha_2 \boldsymbol{n}_2,$$

where $\alpha_1 = \boldsymbol{n} \cdot \boldsymbol{n}_1$ and $\alpha_2 = \boldsymbol{n} \cdot \boldsymbol{n}_2$. We also want $\alpha_1 \geq 0$ and $\alpha_2 \geq 0$, so we may change the direction of $\boldsymbol{n}_1$ and $\boldsymbol{n}_2$ to ensure this last property.

Finally the rotated solver writes

$$\boldsymbol{F}^{RHLL}(\boldsymbol{n}) = \begin{cases} \alpha_1 \boldsymbol{F}^{HLL}(\boldsymbol{n_1}) + \alpha_2 \boldsymbol{F}^{Roe}(\boldsymbol{n_2}) & \text{if } \|\Delta \boldsymbol{q}\| > \epsilon, \\ \boldsymbol{F}^{Roe}(\boldsymbol{n}_\perp) & \text{otherwise.} \end{cases} \tag{3.96}$$

It is clear that we can exchange the HLL solver by an other diffusive solver, Nishikawa [97] also used the Rusanov solver for instance to create the Rotated-RR solver. The solver of Roe can also be replaced by any other solver like the HLLC solver for instance. This was done for instance in [76] where the authors presented the HLLC-HLL scheme. They did not used the rotated framework, but defined a sensor to switch between the two solvers, they also manage to produce a carbuncle free solver. The interest of the rotated solvers is that the approximate Riemann solvers are applied along directions characterized by the flow itself and thus does not rely on the mesh description anymore. This last property is even more important when dealing with unstructured meshes, where the normals to the edges of the mesh may vary a lot.

3.3.7 Numerical results

In this section we present numerical results to assess the quality of the schemes we have presented. Most of these tests are taken from [51] where the authors present a collection of fluid mechanics problems with exact solutions.

Supersonic Jet

This test case taken from [97] is a one-dimensional Riemann problem in a two-dimensional domain. The computational domain is the unit square, the left boundary is split in two parts, they are defined as supersonic inlets with the following values

$$\begin{bmatrix} \rho \\ u \\ v \\ p \end{bmatrix}_{Top} = \begin{bmatrix} 0.25 \\ 4 \\ 0 \\ 0.25 \end{bmatrix} \tag{3.97}$$

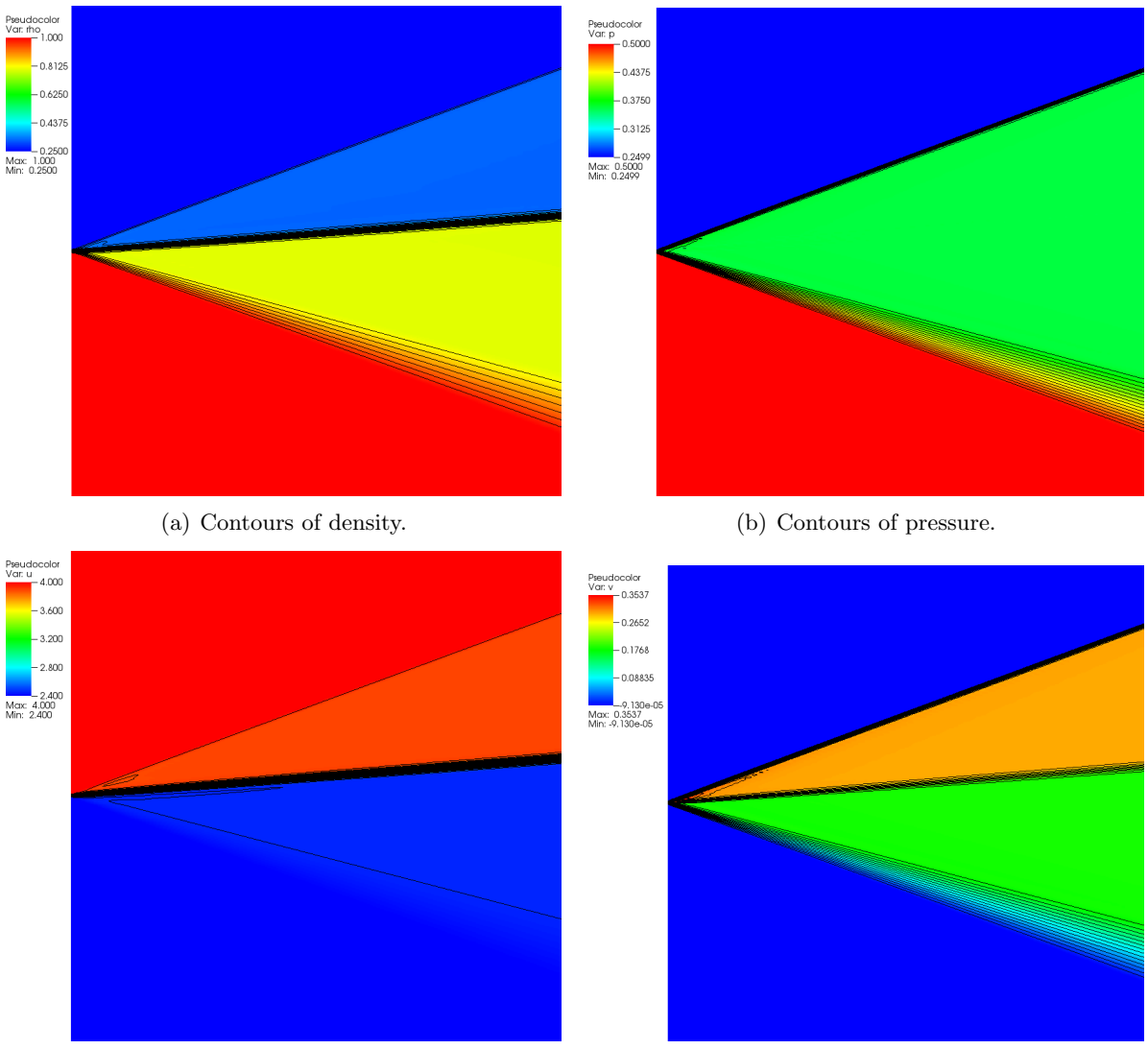
on the top half. This is an inflow at $M = 3.38$. On the bottom half we specify an inflow at $M = 2.87$ with the values

$$\begin{bmatrix} \rho \\ u \\ v \\ p \end{bmatrix}_{Bottom} = \begin{bmatrix} 1 \\ 2.4 \\ 0 \\ 0.5 \end{bmatrix}. \tag{3.98}$$

The remaining boundaries are treated as supersonic outlets. The numerical results computed on a 200×200 Cartesian mesh using HLLC fluxes and the implicit Euler method are displayed

in Figure 3.6. In Figure 3.7 we compare our solution at the outlet, at x = 1, to the analytical solution obtained with an exact Riemann solver. We observe that our numerical results are really close to the exact solution. The shock is captured in two cells, the contact discontinuity lies in three or four cells and the expansion wave is well captured too.

Figure 3.6: Supersonic jet problem using HLLC fluxes on a 200×200 Cartesian mesh.



(a) Contours of density.

(b) Contours of pressure.

(c) Contours of velocity along x-axis.

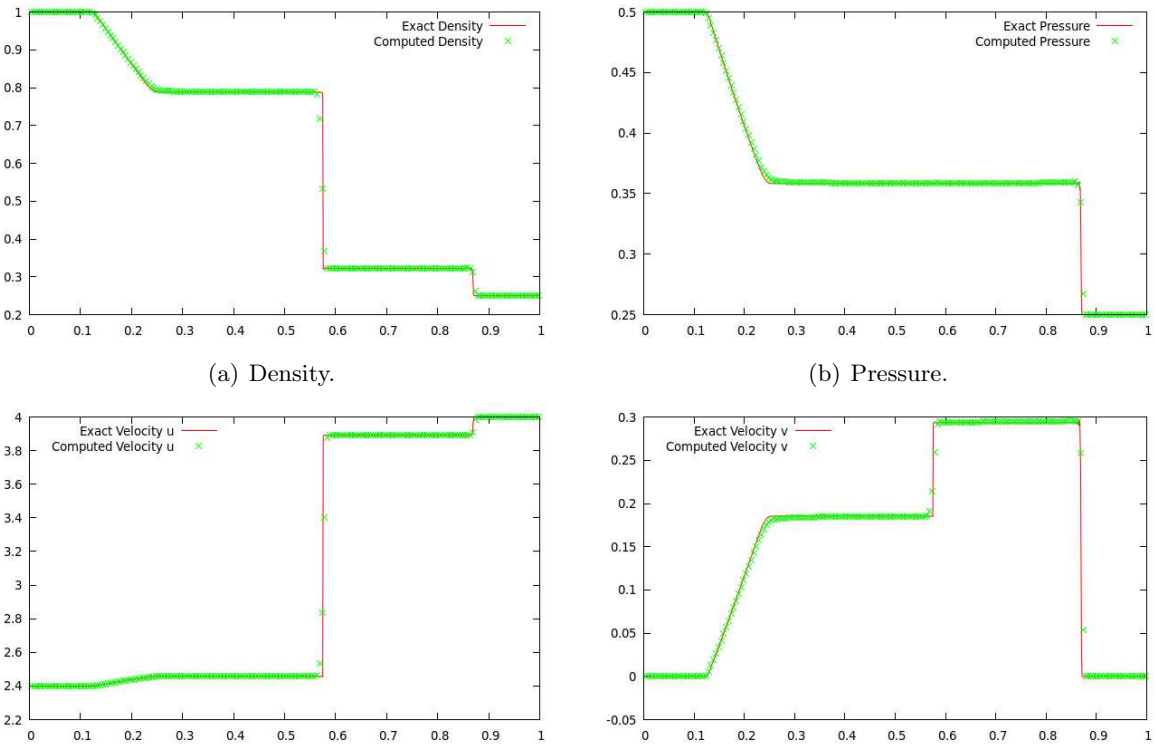
(d) Contours of velocity along y-axis.

Oblique shock problem

This test presents an oblique shock along a wedge. Once again we are able to compute the exact solution of this problem. We use the oblique shock wave theory, well explained in the book of Anderson [15]. For a given Mach number, M_1, and corner angle, θ, the oblique shock angle, β, and the downstream Mach number, M_2, can be calculated. M_2 is always smaller than M_1. The problem is pictured in Figure 3.8 and the relation linking these values is given by

$$\tan\theta = 2\cot\beta \frac{M_1^2 \sin^2\beta - 1}{M_1^2(\gamma + \cos 2\beta) + 2}. \tag{3.99}$$

Figure 3.7: Supersonic jet problem : Comparison of the numerical solution and the exact solution at x=1.



(a) Density.

(b) Pressure.

(c) Velocity along x-axis.

(d) Velocity along y-axis.

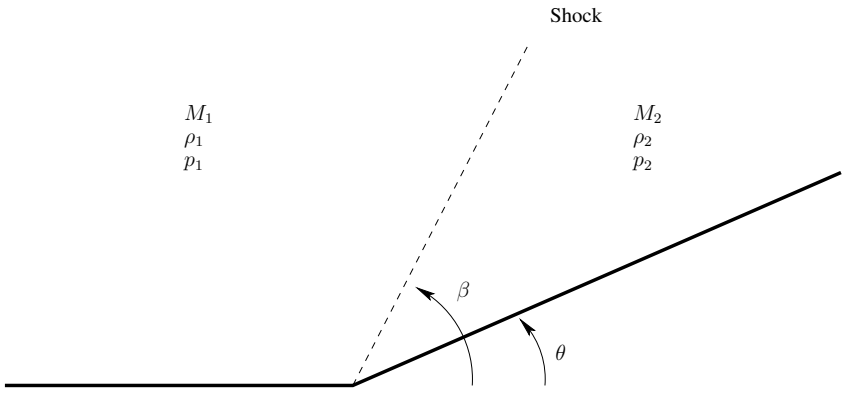


Figure 3.8: Notations used for the oblique shock problem.

This relation is pictured in Figure 3.9. We can also compute the pressure ratio with

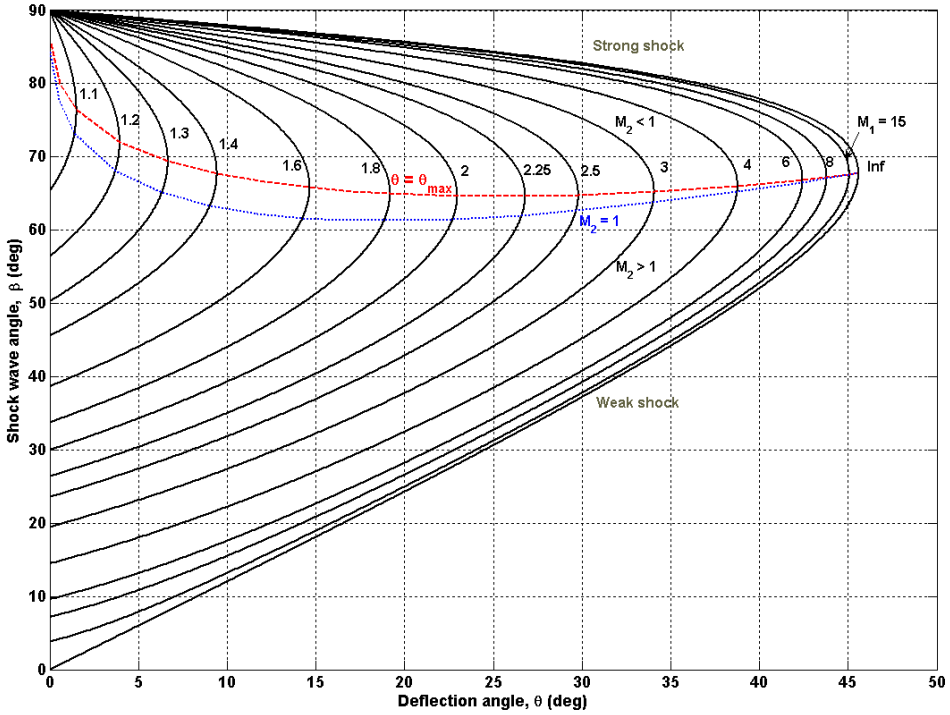


Figure 3.9: Relation between the deflection angle and the shock wave angle for various Mach numbers.

$$\frac{p_2}{p_1} = 1 + \frac{2\gamma}{\gamma+1}(M_1^2 \sin^2\beta - 1). \tag{3.100}$$

The density ratio is given by

$$\frac{\rho_2}{\rho_1} = \frac{(\gamma+1)M_1^2 \sin^2\beta}{(\gamma-1)M_1^2 \sin^2\beta + 2}. \tag{3.101}$$

Finally the Mach after the shock is given by the relation

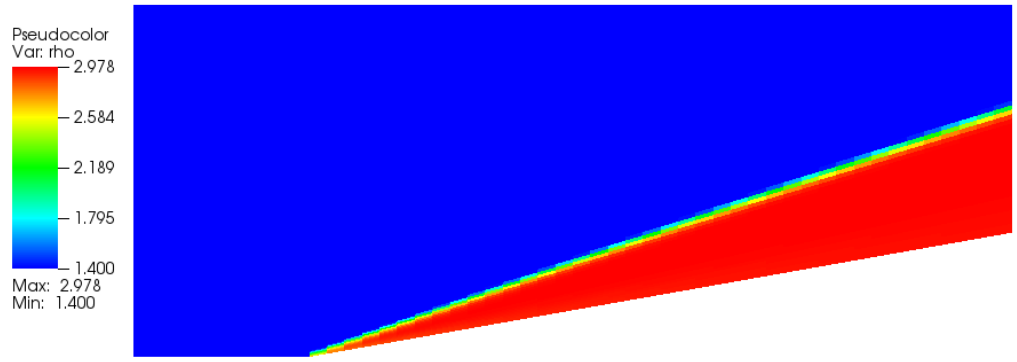
$$M_2 = \frac{1}{\sin(\beta-\theta)}\sqrt{\frac{1+\frac{\gamma-1}{2}M_1^2\sin^2\beta}{\gamma M_1^2 \sin^2\beta - \frac{\gamma-1}{2}}}.$$

We have now all the information needed to generate a test case.

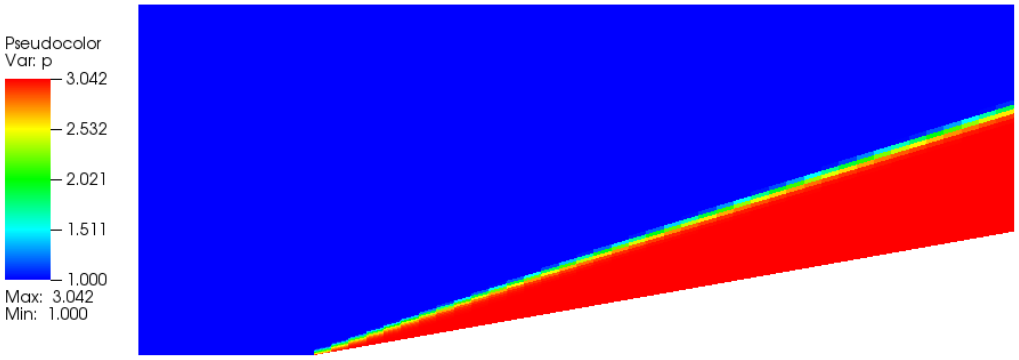
For this purpose we deal with a wedge with an angle $\theta = 10°$ at Mach 5. We take $\gamma = 1.4$. Using Equations (3.99), (3.100) and (3.101) we obtain the admissible weak shock solution $\beta = 19.376°$, $\frac{p_2}{p_1} = 3.044$ and $\frac{\rho_2}{\rho_1} = 2.129$.

We run the test case on a structured 50×50 mesh, refer to Figure 3.10, using the HLLC solver. We use the implicit Euler time discretization until steady state is reached. The solution obtained is presented in Figure 3.10. To compare the results with the theoretical values we plot the solution at the outlet of the mesh at $x = 0.2$. The profiles are displayed in Figure 3.11 and compared to the exact solution. After some post-processing of the results we obtain the numerical values $\beta = 19.676°$, $\frac{p_2}{p_1} = 3.042$ and $\frac{\rho_2}{\rho_1} = 2.127$. These results are really close to the

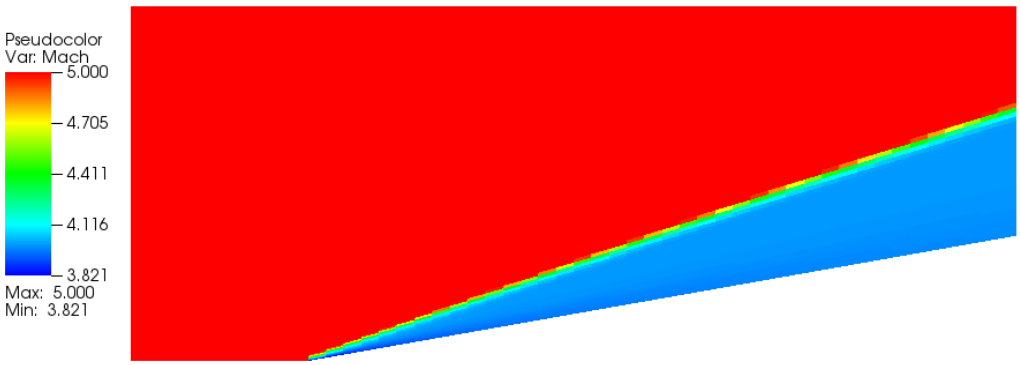
Figure 3.10: Oblique shock test along a wedge at 10 degrees at Mach 5.



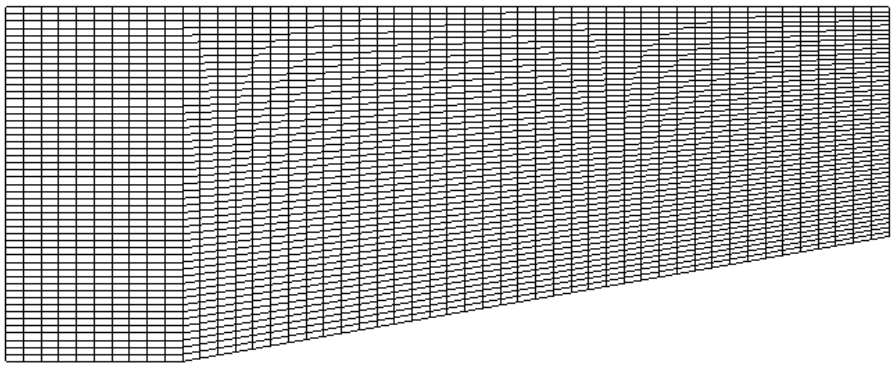
(a) Contours of density.



(b) Contours of pressure.

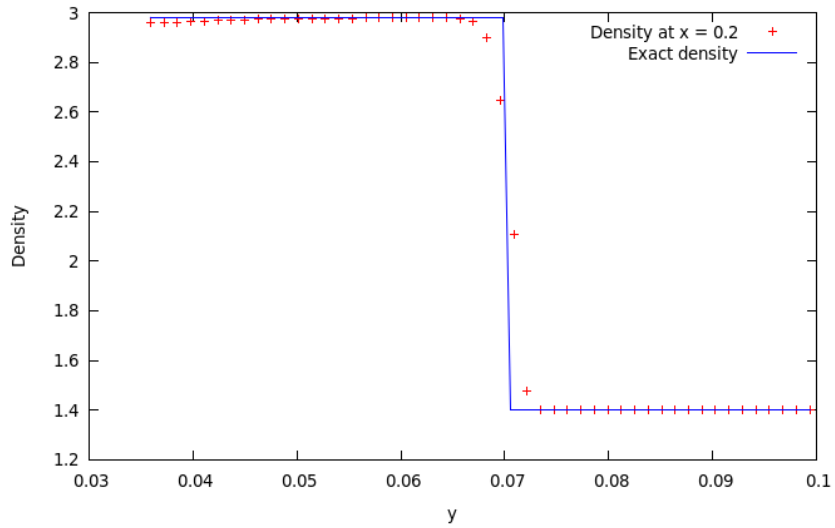


(c) Contours of Mach number.

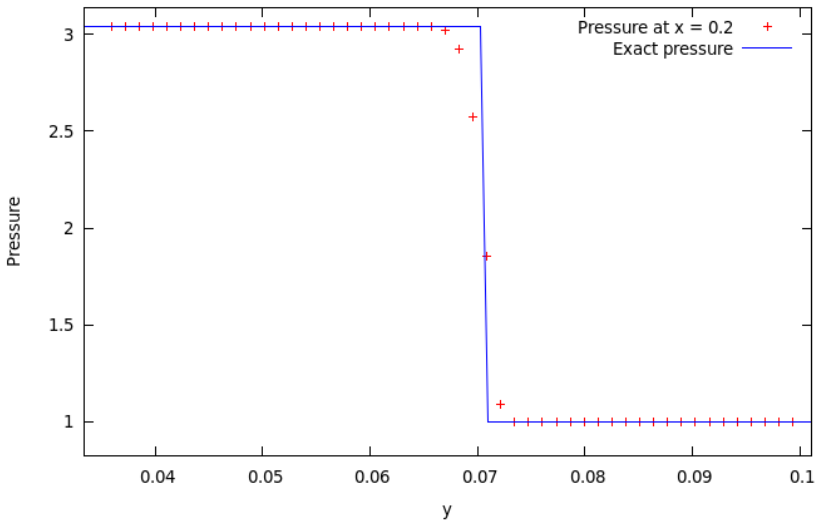


(d) Mesh used in the computations. Composed of 50×50 quadrangles.

Figure 3.11: Oblique shock test along a wedge at 10 degrees at Mach 5.



(a) Comparison of the exact and computed density at x = 0.2.



(b) Comparison of the exact and computed pressure at x = 0.2.

theoretical values. For the pressure we observe that the numerical solution matches perfectly the theoretical values on both sides of the shock, and the shock spans over three cells. For the density, we can see that the freestream density remains unperturbed. The numerical result underestimate a bit the density near the wall. We can note that this error when compared to the theoretical value is below 1%.

Shock diffraction problem

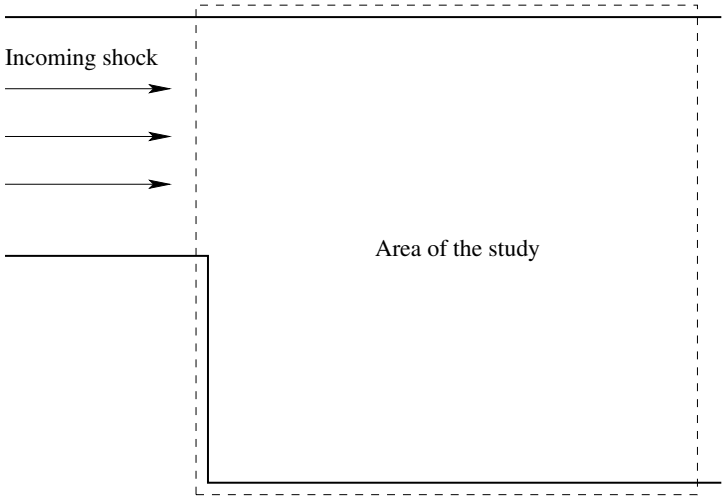


Figure 3.12: Description of the geometry of the shock diffraction problem.

This test taken from [103] models a normal shock diffracting around a 90 degree corner. The geometry of the problem is presented in Figure 3.12. We only mesh a unit square domain, named area of study in Figure 3.12. The corner is located in the middle of the left boundary. The top left boundary is an inlet while the bottom left boundary is a slipping wall. The top and bottom boundary conditions are also modeled with slipping walls while the right boundary condition is defined as a supersonic outlet. The inflow conditions are given using the conditions computed behind a normal shock with the following formulas

$$\rho_\infty = \rho_0 \frac{(\gamma+1)M_{shock}^2}{(\gamma-1)M_{shock}^2+2},$$

where ρ_0 is the density before the shock, ρ_∞ is the density behind the shock and M_{shock} is the Mach number of the shock.

$$p_\infty = p_0 \frac{2\gamma M_{shock}^2 - (\gamma-1)}{(\gamma+1)},$$

where p_0 is the pressure before the shock and p_∞ behind the shock.

$$u_\infty = \left(1 - \frac{\rho_0}{\rho_\infty}\right) u_{shock},$$

where u_0 is the velocity along x-axis before the shock and p_∞ after the shock. Finally we set the velocity along y-axis v_∞ to zero.

In this test case we set $M_{shock} = 5.09$, $\rho_0 = 1$ and $p_0 = \frac{\rho_0}{\gamma}$. The inflow conditions are

$$\begin{bmatrix} \rho_\infty \\ u_\infty \\ v_\infty \\ p_\infty \end{bmatrix}_{Inflow} = \begin{bmatrix} 7.04 \\ 4.08 \\ 0.0 \\ 30.06 \end{bmatrix}. \tag{3.102}$$

We run the computation up to $t = 0.18$, on a 400×400 Cartesian mesh. This test case is unsteady, therefore we need to use a precise time discretization. For this purpose we use the explicit Runge-Kutta method of order two described previously.
As we showed in section 3.3.6, this test produces carbuncles along the normal shock near the top wall, when we use Roe solver or HLLC solver as flux functions. In Figure 3.13 and 3.14 we picture the contours of density and pressure when using the rotated-RHLL fluxes. As expected no carbuncle is formed in this case. For comparison purposes we used the OSSAN-Euler2D CFD code available at [96]. It is an unstructured second-order node-centered Finite Volume CFD code developed by Nishikawa for educational purposes. The results obtained by this code are in good agreement with the ones obtained with our scheme.

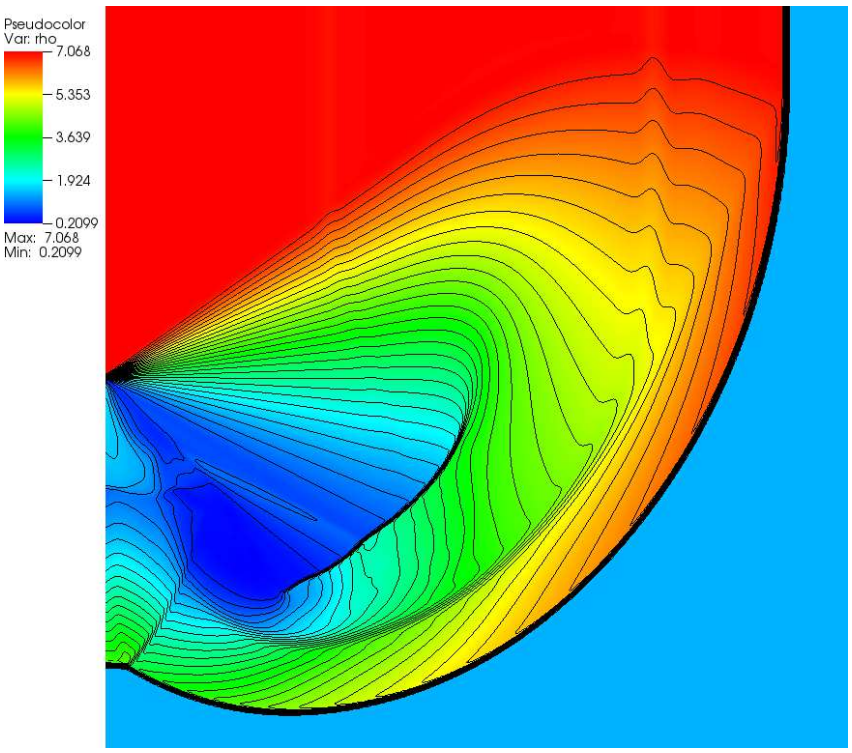


Figure 3.13: Contours of density for the shock diffraction problem using rotated-RHLL fluxes.

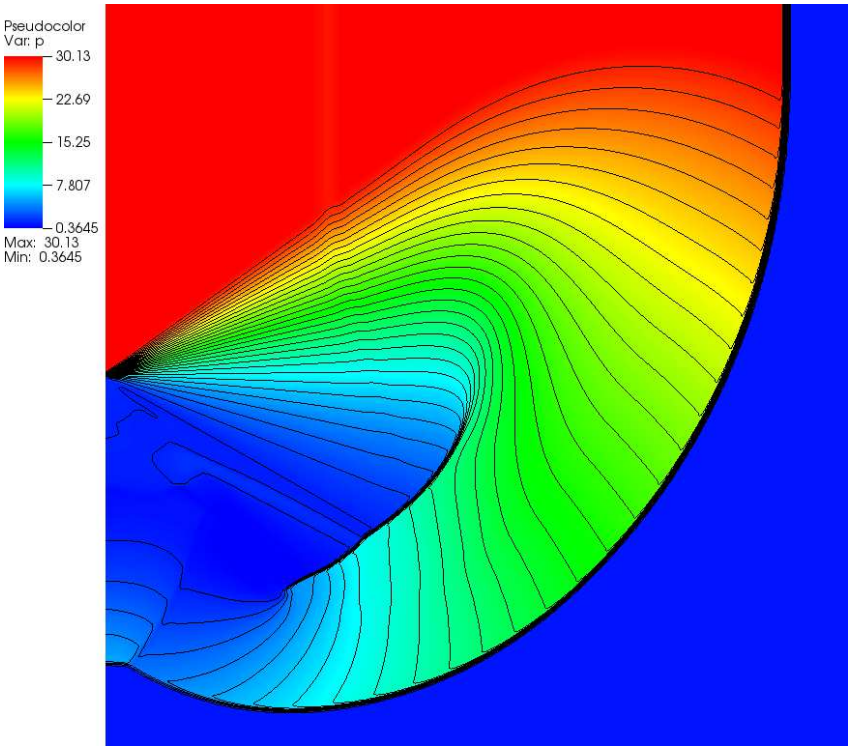


Figure 3.14: Contours of pressure for the shock diffraction problem using rotated-RHLL fluxes.

Cylinder at Mach 17.6

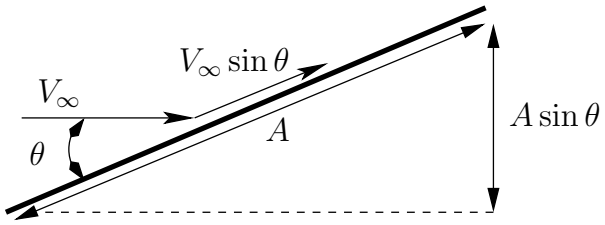


Figure 3.15: Notations used in the Newtonian model.

In this test case we study the flow around a cylinder at Mach 17.6 and compare the results to the Newtonian theory for hypersonic flows [15]. Before proceeding any further, we recall briefly the main features of this theory. Newton modeled a fluid flow as a stream of particles in rectilinear motion, which, when striking a surface lose all their momentum in the direction normal to the surface, but keep their momentum along the surface. In Figure 3.15 we picture a stream with velocity V_∞ impacting on a surface of area A inclined at the angle θ to the freestream. We see that the change in normal velocity is $V_\infty \sin\theta$. The mass flux incident on the surface is equal to $\rho_\infty V_\infty A \sin\theta$. Finally the time rate of change of momentum of this mass flux is $(\rho_\infty V_\infty A \sin\theta)(V_\infty \sin\theta) = \rho_\infty V_\infty^2 A \sin^2\theta$.
The second law of Newton states that the time rate of change of momentum is equal to the force F exerted on the surface

$$F = \rho_\infty V_\infty^2 A \sin^2\theta,$$

which also writes

$$\frac{F}{A} = \rho_\infty V_\infty^2 \sin^2\theta. \tag{3.103}$$

In Equation (3.103), $\frac{F}{A}$ has the dimension of a pressure and must be interpreted as the pressure difference above the freestream static pressure, namely,

$$\frac{F}{A} = p - p_\infty. \tag{3.104}$$

Equations (3.103) and (3.104) yield

$$p - p_\infty = \rho_\infty V_\infty^2 \sin^2\theta,$$

then

$$\frac{p - p_\infty}{\frac{1}{2}\rho_\infty V_\infty^2} = 2\sin^2\theta,$$

the left-hand side is the definition of the pressure coefficient, hence the Newtonian law writes

$$C_p = 2\sin^2\theta. \tag{3.105}$$

This is the classical Newtonian Law. Lees [81] proposed a modification to this theory writing Equation (3.105) as

$$C_p = C_{p_{max}} \sin^2\theta, \tag{3.106}$$

where $C_{p_{max}}$ is the maximum value of the pressure coefficient, evaluated at a stagnation point behind a normal shock wave. We have

$$C_{p_{max}} = \frac{p_s - p_\infty}{\frac{1}{2}\rho_\infty V_\infty^2}, \tag{3.107}$$

where p_s is the total pressure behind a normal shock wave at the freestream Mach number. From normal shock-wave theory we have

$$\frac{p_s}{p_\infty} = \left[\frac{(\gamma+1)^2 M_\infty^2}{4\gamma M_\infty^2 - 2(\gamma-1)}\right]^{\frac{\gamma}{\gamma-1}} \left[\frac{1-\gamma+2\gamma M_\infty^2}{\gamma+1}\right]. \tag{3.108}$$

Noting that $\frac{1}{2}\rho_\infty V_\infty^2 = \frac{\gamma}{2} p_\infty M_\infty^2$, Equation (3.107) yields

$$C_{p_{max}} = \frac{2}{\gamma M_\infty^2}\left[\frac{p_s}{p_\infty} - 1\right]. \tag{3.109}$$

Combining Equations (3.108) and (3.109) we obtain the final expression of $C_{p_{max}}$

$$C_{p_{max}} = \frac{2}{\gamma M_\infty^2}\left\{\left[\frac{(\gamma+1)^2 M_\infty^2}{4\gamma M_\infty^2 - 2(\gamma-1)}\right]^{\frac{\gamma}{\gamma-1}} \left[\frac{1-\gamma+2\gamma M_\infty^2}{\gamma+1}\right] - 1\right\}. \tag{3.110}$$

We can now compute $C_{p_{max}}$ for our test case. We have $\gamma = 1.4$ and $M_\infty = 17.6$, we obtain $C_{p_{max}} = 1.8369$. In Figure 3.16 we compare the pressure coefficient obtained on a structured 30×30 mesh and a structured 60×60 mesh to the Modified Newtonian Law. We observe a good agreement between the two approaches for angles below 40 degrees. For angles above 40 degrees the two approaches do not match anymore. We can also see on this Figure that the coarse mesh give an overestimation of the $C_{p_{max}}$, but the finer mesh gives the expected value. In Figure 3.17 we compare the pressure coefficient obtained by the Modified Newtonian Law against the pressure coefficient obtained on two unstructured meshes made of 1030 and 4006 triangles. Once again we show a that the pressure coefficient matches the Newtonian theory for angles below 40 degrees. We can note that the coarsest mesh manages to give the expected value for the $C_{p_{max}}$.

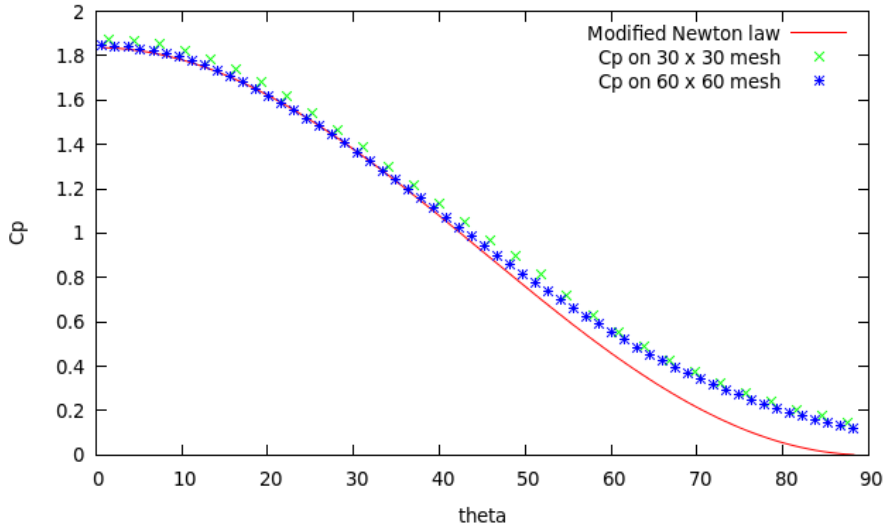


Figure 3.16: Comparison between the pressure coefficient given by the modified Newton law and the numerical results obtained on a 30×30 mesh and a 60×60 mesh.

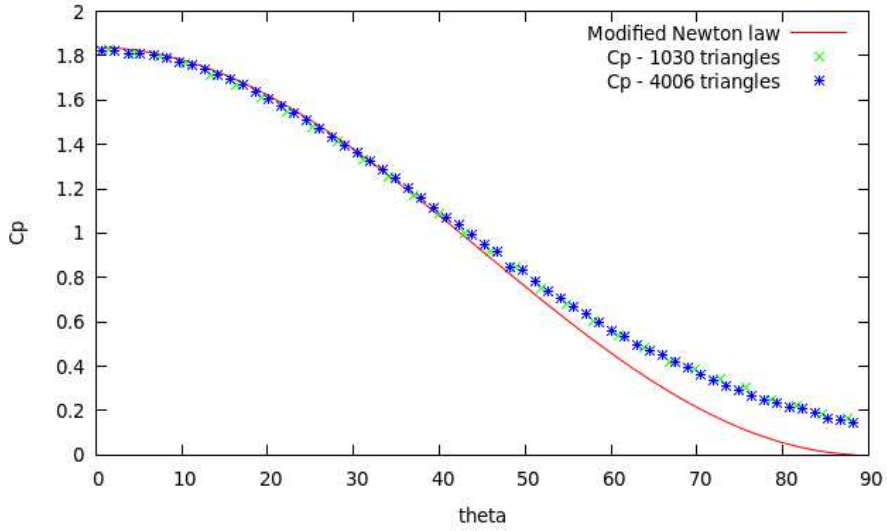


Figure 3.17: Comparison between the pressure coefficient given by the modified Newton law and the results obtained on two unstructured meshes made of 1030 and 4006 triangles.

3.4 Numerical scheme for solving Navier-Stokes equations

Let us go back to the full set of Navier-Stokes equations. We present in this section a numerical scheme for solving these equations using all the work presented in this thesis. We first present the decomposition of the equation in terms of the contributions of Euler equations, heat transfer and tensorial diffusion. Then, we show how the different contributions from the numerical schemes developed for each equation can be introduce in the global scheme. In the last section we present numerical results which assess the robustness and the accuracy of the obtained scheme.

3.4.1 Construction of a Finite Volume scheme for the Navier-Stokes equations

First, we recall the Navier-Stokes equations

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \boldsymbol{V}) = 0 \tag{3.111a}$$

$$\frac{\partial \rho \boldsymbol{V}}{\partial t} + \nabla \cdot (\rho \boldsymbol{V} \otimes \boldsymbol{V} + p\mathbb{I}) = \nabla \cdot \mathbb{S} \tag{3.111b}$$

$$\frac{\partial \rho E}{\partial t} + \nabla \cdot ((\rho E + p)\boldsymbol{V}) = \nabla \cdot (\mathbb{S}\boldsymbol{V}) - \nabla \cdot \boldsymbol{q}. \tag{3.111c}$$

It also writes in a more compact form as

$$\frac{\partial \boldsymbol{U}}{\partial t} + \nabla \cdot \boldsymbol{F}^e(\boldsymbol{U}) = \nabla \cdot \boldsymbol{F}^v(\boldsymbol{U}, \nabla \boldsymbol{U}), \tag{3.112}$$

with $\boldsymbol{U} = [\rho, \rho\boldsymbol{V}, \rho E]^t$, the vector of conservative variables. The left hand side of this equation represents the Euler equations and the right hand-side contains the viscous part of the Navier-Stokes equations.

In the right-hand side of equation (3.111b), we find the term $\nabla \cdot \mathbb{S}$ that we studied in Chapter 2. In the right-hand side of equation (3.111c), we find the term $\nabla \cdot \boldsymbol{q}$ that we studied in Chapter 1. In equation (3.111c) we also find the term $\nabla \cdot (\mathbb{S}\boldsymbol{V})$. This term has not been derived yet but we will discuss how to reuse elements of the scheme developed for the tensorial diffusion to discretize it in section 3.4.3. What we want to point out here is that the Navier-Stokes equations are an assembly of all the equations we described in this thesis. We are going to explain how to assemble the numerical schemes we developed in order to construct a numerical scheme for the Navier-Stokes equations.

Let us write the Finite Volume method for the Navier-Stokes equations. We integrate equation (3.112) over a cell ω_c

$$\int_{\omega_c} \frac{\partial \boldsymbol{U}}{\partial t} ds + \int_{\omega_c} \nabla \cdot \boldsymbol{F}^e(\boldsymbol{U}) ds = \int_{\omega_c} \nabla \cdot \boldsymbol{F}^v(\boldsymbol{U}, \nabla \boldsymbol{U}), \tag{3.113}$$

we use the Green theorem

$$\frac{d}{dt} \int_{\omega_c} \boldsymbol{U} ds + \int_{\partial\omega_c} \boldsymbol{F}^e(\boldsymbol{U}) \cdot \boldsymbol{n} dl = \int_{\partial\omega_c} \boldsymbol{F}^v(\boldsymbol{U}, \nabla \boldsymbol{U}) \cdot \boldsymbol{n} dl. \tag{3.114}$$

We introduce the mean cell conservative variables $\boldsymbol{U}_c$ as defined for the Euler equation. Equation (3.114) rewrites

$$\frac{d}{dt} w_c \boldsymbol{U}_c + \sum_{d \in \mathcal{C}_f(c)} l_{cd} \hat{\boldsymbol{F}}^e(\tilde{\boldsymbol{U}}_c^d, \tilde{\boldsymbol{U}}_d^c) \cdot \boldsymbol{n}_{cd} = \sum_{d \in \mathcal{C}(c)} l_{cd} \hat{\boldsymbol{F}}^v(\boldsymbol{U}_c, \boldsymbol{U}_d) \cdot \boldsymbol{n}_{cd}. \tag{3.115}$$

$\hat{\boldsymbol{F}}^e$ is the Euler numerical flux defined earlier, which is evaluated by an approximate Riemann solver. $\tilde{\boldsymbol{U}}_c^d$ and $\tilde{\boldsymbol{U}}_d^c$ are the second-order reconstructed states obtained by a MUSCL type procedure. $\hat{\boldsymbol{F}}^v$ is the numerical flux gathering the diffusion terms. It is defined in the neighborhood $\mathcal{C}(c)$ of all the cells sharing a node with cell ω_c. We introduce the backward Euler scheme (3.90) obtained in the beginning of this chapter for the Euler equations in (3.115), which yields

$$\left(\frac{\mathbb{V}}{\Delta t^n} + \mathbb{E}^n\right)\boldsymbol{\delta\mathcal{U}} = -\boldsymbol{\mathcal{R}}^n + \boldsymbol{\mathcal{N}}(\boldsymbol{\mathcal{U}}), \tag{3.116}$$

where $\boldsymbol{\mathcal{N}}(\boldsymbol{\mathcal{U}})$ contains the viscous fluxes of the Navier-Stokes equations, that we still have not discretized. In this numerical flux we gather the numerical fluxes defined in chapter 1 and 2. This is the topic of the following two paragraphs.

3.4.2 Gathering the contribution of Heat transfer

Let us start by recalling the conservation of energy in the Navier-Stokes equations. It writes

$$\frac{\partial \rho E}{\partial t} + \nabla\cdot((\rho E + p)\boldsymbol{V}) = \nabla\cdot(\mathbb{S}\boldsymbol{V}) - \nabla\cdot\boldsymbol{q}. \tag{3.117}$$

In this section we are interested by the discretization of the term $\nabla\cdot\boldsymbol{q}$. We recall that Chapter 1 was dealing with the discretization of anisotropic heat equation, namely

$$\rho C_v\frac{\partial T}{\partial t} + \nabla\cdot\boldsymbol{q} = 0. \tag{3.118}$$

We recall here the final version of the numerical scheme we obtained, when neglecting the source terms

$$\left(\frac{\mathbb{MC}_v}{\Delta t^n} + \mathbb{D}\right)\boldsymbol{\mathcal{T}}^{n+1} = \frac{\mathbb{MC}_v}{\Delta t^n}\boldsymbol{\mathcal{T}}^n + \boldsymbol{\Sigma}^n. \tag{3.119}$$

We note $\boldsymbol{\delta\mathcal{T}} = \boldsymbol{\mathcal{T}^{n+1}} - \boldsymbol{\mathcal{T}^n}$, equation (3.119) becomes

$$\left(\frac{\mathbb{MC}_v}{\Delta t^n} + \mathbb{D}\right)\boldsymbol{\delta\mathcal{T}} = -\mathbb{D}\boldsymbol{\mathcal{T}}^n + \boldsymbol{\Sigma}^n. \tag{3.120}$$

The important things to note in this equation are the matrix $\mathbb{D}$ which discretize the term $\nabla\cdot\boldsymbol{q}$, and the right hand side $\boldsymbol{\Sigma}^n$ which contains the boundary conditions. The matrix $\frac{\mathbb{MC}_v}{\Delta t^n}$ corresponds to the discretization of the term $\rho C_v\frac{\partial T}{\partial t}$, which is not needed in equation (3.117), the time discretization being taken care of in the Euler part of the scheme.

In the Navier-Stokes equation we are dealing with $\boldsymbol{\delta\mathcal{U}}$ instead of $\boldsymbol{\delta\mathcal{T}}$. Therefore, we need to express equation (3.120) in terms of $\boldsymbol{\delta\mathcal{U}}$. We have the relation

$$\boldsymbol{\delta\mathcal{T}} = \frac{\partial T}{\partial \boldsymbol{U}}\boldsymbol{\delta\mathcal{U}}. \tag{3.121}$$

$\boldsymbol{U} = (\rho, \rho u, \rho v, \rho E)^t$ and for an ideal gas $e = C_v T$. We recall that $e = E - \frac{1}{2}(u^2+v^2)$. We can then compute $\frac{\partial \boldsymbol{U}}{\partial T}$ as follow

$$\frac{\partial T}{\partial \rho} = \frac{\partial}{\partial \rho}\left(\frac{\rho E - \frac{(\rho u)^2}{2\rho} - \frac{(\rho v)^2}{2\rho}}{\rho C_v}\right) = \frac{u^2+v^2-E}{\rho C_v} \tag{3.122a}$$

$$\frac{\partial T}{\partial \rho u} = \frac{-u}{\rho C_v} \tag{3.122b}$$

$$\frac{\partial T}{\partial \rho v} = \frac{-v}{\rho C_v} \tag{3.122c}$$

$$\frac{\partial T}{\partial \rho E} = \frac{1}{\rho C_v} \tag{3.122d}$$

We introduce the bloc matrix $\bar{\mathbb{D}}$ defined by

$$\bar{\mathbb{D}}_{ij} = \mathbb{D}_{ij}\mathbb{P}_j^T, \tag{3.123}$$

where $\mathbb{P}_c^T$ is the transformation matrix defined for each cell c by

$$\mathbb{P}_c^T = \begin{pmatrix} \frac{\partial T_c}{\partial \rho} & \frac{\partial T_c}{\partial \rho u} & \frac{\partial T_c}{\partial \rho v} & \frac{\partial T_c}{\partial \rho E} \end{pmatrix}. \tag{3.124}$$

This matrix allows us to transform the operator $\nabla \cdot \boldsymbol{q}$ expressed for the unknown T as the same operator defined in terms of the unknown $\boldsymbol{U}$.

Finally, since we are dealing with the full set of Navier-Stokes equations we need to integrate the heat transfer scheme in the global system. The global system is composed of 4×4 blocks in two dimensions, expressing the four equations in terms of the four unknowns. The heat transfer is only present in the last equation, so we define the matrix $\mathbb{Q}^T$ which is defined by

$$\mathbb{Q}^T = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \end{pmatrix}. \tag{3.125}$$

This transformation matrix allows to define matrix $\hat{\mathbb{D}}$ as

$$\hat{\mathbb{D}}_{ij} = \mathbb{Q}^T\mathbb{D}_{ij}\mathbb{P}_j^T = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ \frac{\partial T_c}{\partial \rho} & \frac{\partial T_c}{\partial \rho u} & \frac{\partial T_c}{\partial \rho v} & \frac{\partial T_c}{\partial \rho E} \end{pmatrix} \mathbb{D}_{ij}, \tag{3.126}$$

which is the contribution of the heat transfer in terms of the conservative variables in the equation of energy. We also define the vector $\hat{\boldsymbol{\Sigma}}^n$ by

$$\hat{\boldsymbol{\Sigma}}_c^n = \mathbb{Q}^T\left((-\mathbb{D}\boldsymbol{\mathcal{T}}^n)_c + \boldsymbol{\Sigma}_c^n\right) = \begin{pmatrix} 0 \\ 0 \\ 0 \\ (-\mathbb{D}\boldsymbol{\mathcal{T}}^n)_c + \boldsymbol{\Sigma}_c^n \end{pmatrix}, \tag{3.127}$$

it contains the explicit contribution of the heat transfer and the boundary conditions.

3.4.3 Gathering the contribution of Tensorial Diffusion

Let us now consider the conservation of momentum in the Navier-Stokes equations. It writes

$$\frac{\partial \rho \boldsymbol{V}}{\partial t} + \nabla \cdot (\rho \boldsymbol{V} \otimes \boldsymbol{V} + p\mathbb{I}) = \nabla \cdot \mathbb{S}. \tag{3.128}$$

In this section we are interested by the term $\nabla \cdot \mathbb{S}$. In Chapter 2, we have dealt with tensorial diffusion, namely

$$\rho \frac{\partial \boldsymbol{V}}{\partial t} - \nabla \cdot \mathbb{S} = \boldsymbol{0}. \tag{3.129}$$

We then introduced the modified constitutive law $\mathbb{\Sigma}$ such that $\nabla \cdot \mathbb{\Sigma} = \nabla \cdot \mathbb{S}$. We use this modified constitutive law in this section in order to solve equation (3.128) under the form

$$\frac{\partial \rho \boldsymbol{V}}{\partial t} + \nabla \cdot (\rho \boldsymbol{V} \otimes \boldsymbol{V} + p\mathbb{I}) = \nabla \cdot \mathbb{\Sigma}. \tag{3.130}$$

In Chapter 2, we expressed equation (3.129) under the form

$$\rho \frac{\partial \boldsymbol{V}}{\partial t} - \nabla \cdot \mathbb{\Sigma} = \boldsymbol{0}. \tag{3.131}$$

We developed a numerical scheme to solve this problem. We recall the final version of this scheme in the absence of source term.

$$\left(\frac{\mathbb{M}}{\Delta t^n} + \mathbb{T} \right) \boldsymbol{\mathcal{V}}^{n+1} = \frac{\mathbb{M}}{\Delta t^n} \boldsymbol{\mathcal{V}}^n + \boldsymbol{\mathcal{B}}^n. \tag{3.132}$$

We note $\boldsymbol{\delta \mathcal{V}} = \boldsymbol{\mathcal{V}^{n+1}} - \boldsymbol{\mathcal{V}^n}$, equation (3.132) becomes

$$\left(\frac{\mathbb{M}}{\Delta t^n} + \mathbb{T} \right) \boldsymbol{\delta \mathcal{V}} = -\mathbb{T} \boldsymbol{\mathcal{V}}^n + \boldsymbol{\mathcal{B}}^n. \tag{3.133}$$

Once again, the important matrix here is $\mathbb{T}$, which is the discrete equivalent of $\nabla \cdot \mathbb{\Sigma}$, and, the right-hand side $\boldsymbol{\mathcal{B}}^n$, which contains the boundary conditions.

We express $\boldsymbol{\delta \mathcal{U}}$ in terms of $\boldsymbol{\delta \mathcal{V}}$ using the relation

$$\boldsymbol{\delta \mathcal{V}} = \frac{\partial \boldsymbol{V}}{\partial \boldsymbol{U}} \boldsymbol{\delta \mathcal{U}}. \tag{3.134}$$

We have $\boldsymbol{V} = (u, v)^t$, let us deal with u first

$$\frac{\partial u}{\partial \rho} = \frac{\partial}{\partial \rho} \left(\frac{\rho u}{\rho} \right) = \frac{-u}{\rho}, \tag{3.135a}$$

$$\frac{\partial u}{\partial \rho u} = \frac{1}{\rho}, \tag{3.135b}$$

$$\frac{\partial u}{\partial \rho v} = 0, \tag{3.135c}$$

$$\frac{\partial u}{\partial \rho E} = 0. \tag{3.135d}$$

We do the same for v

$$\frac{\partial v}{\partial \rho} = \frac{\partial}{\partial \rho} \left(\frac{\rho v}{\rho} \right) = \frac{-v}{\rho}, \tag{3.136a}$$

$$\frac{\partial v}{\partial \rho u} = 0, \tag{3.136b}$$

$$\frac{\partial v}{\partial \rho v} = \frac{1}{\rho}, \tag{3.136c}$$

$$\frac{\partial v}{\partial \rho E} = 0. \tag{3.136d}$$

We introduce the bloc matrix $\bar{\mathbb{T}}$ defined by

$$\bar{\mathbb{T}}_{ij} = \mathbb{T}_{ij} \mathbb{P}_j^V, \tag{3.137}$$

where $\mathbb{P}_c^V$ is the transformation matrix defined for each cell c by

$$\mathbb{P}_c^V = \begin{pmatrix} \frac{\partial u_c}{\partial \rho} & \frac{\partial u_c}{\partial \rho u} & \frac{\partial u_c}{\partial \rho v} & \frac{\partial u_c}{\partial \rho E} \\ \frac{\partial v_c}{\partial \rho} & \frac{\partial v_c}{\partial \rho u} & \frac{\partial v_c}{\partial \rho v} & \frac{\partial v_c}{\partial \rho E} \end{pmatrix} = \frac{1}{\rho} \begin{pmatrix} -u & 1 & 0 & 0 \\ -v & 0 & 1 & 0 \end{pmatrix}. \tag{3.138}$$

This matrix allows us to transform the operator $\nabla \cdot \mathbb{\Sigma}$ expressed in terms of the unknown $\boldsymbol{V}$ as the same operator defined in terms of the unknown $\boldsymbol{U}$.

In order to integrate this in the full set of Navier-Stokes equations we introduce the matrix $\mathbb{Q}^V$ which is defined by

$$\mathbb{Q}^V = \begin{pmatrix} 0 & 0 \\ 1 & 0 \\ 0 & 1 \\ 0 & 0 \end{pmatrix}. \tag{3.139}$$

Using this transformation we are able to define the matrix $\hat{\mathbb{T}}$ which express the operator $\nabla \cdot \mathbb{\Sigma}$ in the equation of momentum in terms of the conservative variables. It writes

$$\hat{\mathbb{T}}_{ij} = \mathbb{Q}^V \mathbb{T}_{ij} \mathbb{P}_j^V, \tag{3.140}$$

We also define the vector $\hat{\boldsymbol{\mathcal{B}}}^n$, which contains the explicit terms of the numerical scheme for tensorial diffusion and the boundary terms. It is defined by

$$\hat{\boldsymbol{\mathcal{B}}}^n{}_c = \mathbb{Q}^V \left((-\mathbb{T} \boldsymbol{\mathcal{V}}^n)_c + \boldsymbol{\mathcal{B}}_c^n \right). \tag{3.141}$$

Contribution of the work of the viscous forces

In the energy equation we still have to discretize the term $\nabla \cdot (\mathbb{S}\boldsymbol{V})$.

$$\boldsymbol{W}_c = \int_{\omega_c} \nabla \cdot (\mathbb{S}\boldsymbol{V}) dv = \int_{\partial\omega_c} (\mathbb{S}\boldsymbol{V}) \cdot \boldsymbol{n} ds. \tag{3.142}$$

Using the notations defined in chapter 2 this terms rewrites

$$\boldsymbol{W}_c = \sum_{p \in \mathcal{P}(c)} l_{pc}^- (\mathbb{S}_{pc} \boldsymbol{V}_{pc}^-) \cdot \boldsymbol{n}_{pc}^- + l_{pc}^+ (\mathbb{S}_{pc} \boldsymbol{V}_{pc}^+) \cdot \boldsymbol{n}_{pc}^+. \tag{3.143}$$

All the terms in this equation have been defined in chapter 2 and are easily accessible when building matrix $\mathbb{T}$. It means that we can define the vector $\hat{\boldsymbol{W}}_c^n$

$$\hat{\boldsymbol{W}}_c^n = \mathbb{Q}^T \boldsymbol{W}_c^n. \tag{3.144}$$

This vector contains the contributions of the work of the viscous forces in the equation of conservation of energy expressed in the global system.

Comment 24: *We have to point out that the time discretization of this term is explicit. This is the only term of the global scheme that is not implicited. It may cause convergence issues. The implicitation of this term should be studied in order to obtain a fully implicit scheme. We will show that the scheme can be still used without the implicitation.*

3.4.4 Final expression of the Finite Volume scheme

We are now able to build the Finite volume scheme for the Navier-Stokes equations. We gather the terms defined by (3.90), (3.126),(3.127),(3.140),(3.141),(3.144) to obtain

$$\left(\frac{\mathbb{V}}{\Delta t^n} + \mathbb{E}^n + \hat{\mathbb{D}} + \hat{\mathbb{T}}\right) \boldsymbol{\delta\mathcal{U}} = \boldsymbol{\mathcal{R}}^n + \hat{\boldsymbol{\mathcal{B}}}^n + \hat{\boldsymbol{\Sigma}}^n + \hat{\boldsymbol{W}}^n. \tag{3.145}$$

This is a sparse linear system, the dimension of the matrix is $\mathfrak{C}_{\mathcal{D}} \times \mathfrak{C}_{\mathcal{D}}$ and is composed of block-matrices of size 4×4 in two dimensions. The right-hand side and the vector $\boldsymbol{\delta\mathcal{U}}$ are block-vectors composed of $\mathfrak{C}_{\mathcal{D}}$ sub-vectors of size 4 in two dimensions. In order to solve this sparse linear system we employ the localized ILU(0) Preconditioned BiCGStab algorithm [127, 95] already described in chapter 1 and 2. As mentioned in comment 19 the Conjugate Gradient method would not be sufficient to solve this linear system because it is not symmetric.

Comment 25: *We need to mention the parallelization of this numerical scheme. In comment 20 we presented the small amount of developments needed by the tensorial diffusion scheme to benefit from the parallel implementation of the CCLAD scheme. We have to mention that these developments also benefits to the Navier-Stokes numerical scheme we constructed. The Finite Volume scheme allows us to build a linear system composed of a block-matrix and a block-vector. The difference with the tensorial diffusion scheme only lies in the dimension of the blocks. It means that we benefit from a parallel numerical Navier-Stokes scheme without further developments.*

3.4.5 Numerical results

In this section we present numerical results to assess the quality of the schemes we presented here. The first test case is designed to verify the validity of the discretization of the viscous terms. Then we present the classical flat plate test case. It allows us to understand how the numerical scheme handle the mesh refinement needed to correctly capture the boundary layer. Finally, we compare the results obtained with our scheme on a supersonic viscous flow around a cylinder, with the results obtained with two CFD codes developed by NASA [1].

Thermal Couette flow

This test case adapted from [67] will allow us to verify the discretization of the viscous terms. The test describes a laminar flow between two infinitely long walls. These walls are parallel and separated by a distance L. The top wall is moving at a constant speed u_t while the bottom wall stands still. Finally, the top wall is heated at a temperature T_t and the bottom wall is heated at a temperature T_b such that $T_t > T_b$.

Since the walls are considered infinitely long all the x-derivatives have to vanish. We also point out that the velocity v needs to be equal to zero, which yields

$$\boldsymbol{V} = \begin{pmatrix} u(y) \\ 0 \end{pmatrix}.$$

We also consider that the pressure is constant in the whole domain

$$p = p_\infty.$$

With this assumptions the conservation of momentum simplifies to

$$\frac{\partial}{\partial y}(\mu \frac{\partial u}{\partial y}) = 0. \tag{3.146}$$

We take $\mu = Cte$ so equation (3.146) reduces to

$$\frac{\partial^2 u}{\partial y^2} = 0. \tag{3.147}$$

The boundary conditions are given by $u(0) = 0$ and $u(L) = u_t$, which yields

$$u(y) = \frac{y}{L} u_t. \tag{3.148}$$

The conservation of energy yields

$$\kappa \frac{\partial^2 T}{\partial y^2} + \mu \left(\frac{\partial u}{\partial y} \right)^2 = 0, \tag{3.149}$$

which simplifies to

$$\frac{\partial^2 T}{\partial y^2} = -\frac{\mu u_t^2}{\kappa L^2}. \tag{3.150}$$

We can deduce that the temperature profile is parabolic. $T(y)$ writes under the form

$$T(y) = -\frac{\mu u_t^2}{\kappa L^2} y^2 + ay + b, \tag{3.151}$$

with $T(0) = T_b$ and $T(L) = T_t$. The temperature writes

$$T(y) = -\frac{\mu u_t^2}{\kappa} \left(\frac{y}{L} \right)^2 + (T_t - T_b + \frac{\mu u_t^2}{\kappa}) \frac{y}{L} + T_b. \tag{3.152}$$

It is interesting to introduce the adimensionned length $\tilde{y} = \frac{y}{L}$ and the adimensionned temperature $\tilde{T} = \frac{T - T_b}{\Delta T}$ with $\Delta T = T_t - T_b$. Equation (3.152) rewrites

$$\tilde{T}(\tilde{y}) = (1 + \frac{\mu u_t^2}{\kappa \Delta T}(1 - \tilde{y}))\tilde{y}. \tag{3.153}$$

We introduce the Prandtl number $P_r = \frac{\mu C_p}{\kappa}$ and the Eckert number $E_c = \frac{u_t^2}{C_p \Delta T}$. This adimensionned number is the ratio of the dynamic temperature induced by fluid motion to the characteristic temperature difference in the fluid. Equation (3.153) yields

$$\tilde{T}(\tilde{y}) = (1 + \frac{1}{2} P_r E_c (1 - \tilde{y}))\tilde{y}. \tag{3.154}$$

An interesting variation of temperature is obtained for $P_r E_c = 4$ where the maximum temperature is greater than the maximal temperature at the wall.

For this test case we impose the values $T_b = 300K$, $T_t = 310K$, $u_t = 10$. We use the classical value of the Prandtl number for the air $P_r = 0.72$. We impose $\mu = 1$. It remains to specify C_p such that $P_r E_c = 4$, we obtain $C_p = 1.8$.

Figure 3.18 shows the profile of temperature obtained at the outlet for a series of Cartesian meshes. We observe that the maximal temperature obtained gives a small overestimation of the theoretical maximal value, but, with the help of mesh refinement the approximation tends to become more accurate. In Figure 3.19 we can observe the profile of temperature obtained at the outlet for a series of triangular meshes. The same remarks apply.

Table 3.1: Thermal Couette flow: asymptotic errors in both L^2 and L^∞ norms and corresponding truncation error orders for the velocity on Cartesian grids.

h	E_{L2}	q_{L2}	E_∞	q_∞
2.00e-03	4.20e-04	**1.89**	8.00e-04	**1.92**
1.00e-03	1.13e-04	**1.96**	2.11e-04	**1.94**
5.00e-04	2.29e-05	-	5.50e-05	-

Table 3.2: Thermal Couette flow: asymptotic errors in both L^2 and L^∞ norms and corresponding truncation error orders for the velocity on triangular grids.

h	E_{L2}	q_{L2}	E_∞	q_∞
1.23e-03	3.41e-04	**1.95**	5.63e-04	**1.78**
6.12e-04	8.74e-05	**2.01**	1.62e-04	**1.94**
3.08e-04	2.20e-05	-	4.29e-05	-

In Figure 3.20 we picture the profiles of velocity for a series of Cartesian meshes and for a series of triangular meshes. We observe that we obtain a very good approximation of the linear velocity profile for all of the meshes.

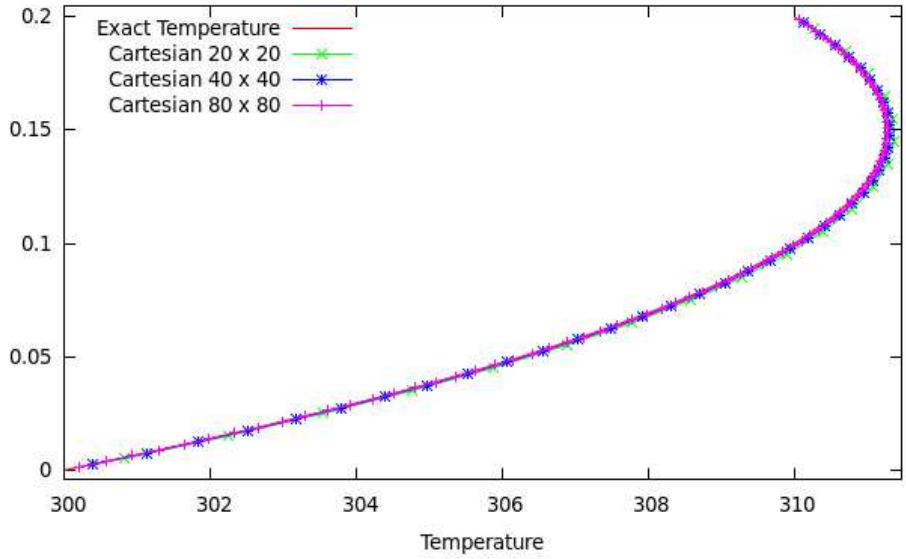
In Tables 3.3 and 3.4 we compute the L^2 and L^∞ errors of the scheme for the temperature field, along with their order of convergence rate. As expected in regards with the results obtained in chapter 1 for the heat transfer we achieve second order accuracy for the temperature field on both the Cartesian and triangular meshes.

In Tables 3.1 and 3.2 we compute the L^2 and L^∞ errors of the scheme for the linear velocity field, along with their respective order of convergence. We observe that we only reach second order accuracy for the two kinds of meshes. In chapter 2 we showed that we achieved round-off error on Cartesian meshes and on triangular meshes for linear fields. As a first explanation we can say that this accuracy is not obtained for this Navier-Stokes test case, because, we are this time dealing with a system of equations in which some of the fields are not linear. The error obtained on the other variables, such as the temperature, has then an influence on the velocity field. Still we obtain second order accuracy which is the expected order of the scheme for general solutions. An other possible explanation is that theoretically the pressure is constant and the vertical velocity v is equal to zero. This is not the case in practice, the variables p and v appears to have small variations. This can be interpreted as the presence of compressible effects which are not taken into account in the theory we presented.

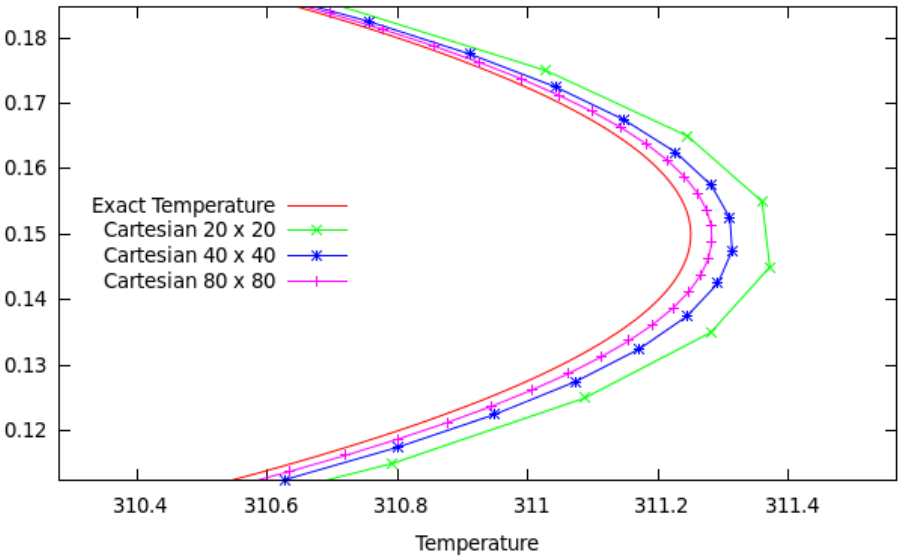
Table 3.3: Thermal Couette flow: asymptotic errors in both L^2 and L^∞ norms and corresponding truncation error orders for the temperature on Cartesian grids.

h	E_{L2}	q_{L2}	E_∞	q_∞
2.00e-03	4.11e-03	**1.91**	5.05e-03	**2.02**
1.00e-03	1.09e-03	**1.97**	1.24e-03	**1.95**
5.00e-04	2.78e-04	-	3.22e-04	-

Figure 3.18: Thermal Couette flow on Cartesian meshes.



(a) Profile of temperatures.

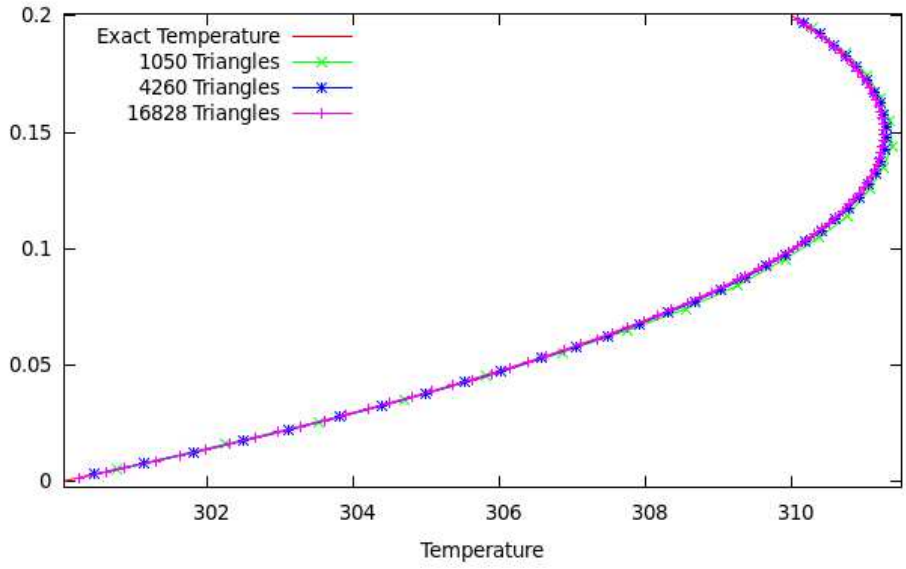


(b) Profile of temperature, zoom around the maximal value.

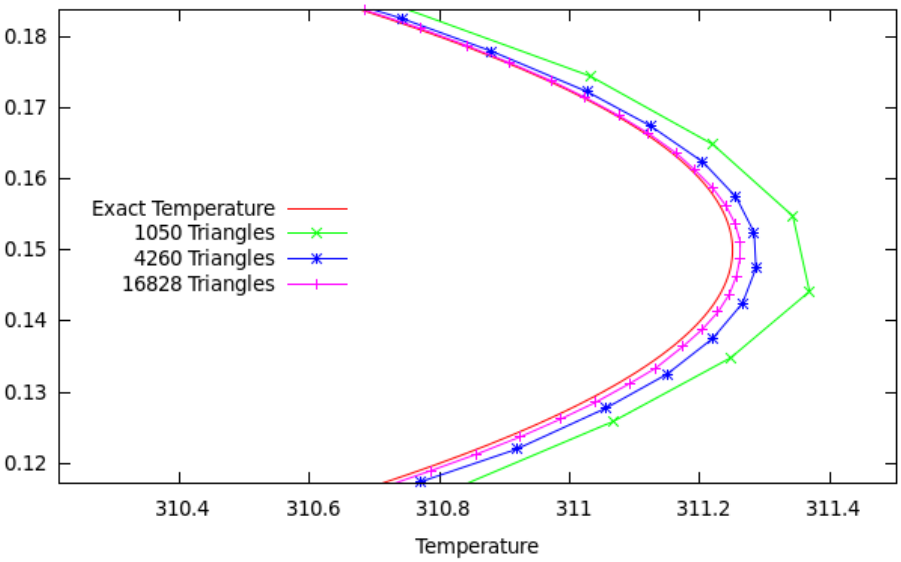
Table 3.4: Thermal Couette flow: asymptotic errors in both L^2 and L^∞ norms and corresponding truncation error orders for the temperature on triangular grids.

h	E_{L2}	q_{L2}	E_∞	q_∞
1.23e-03	4.52e-03	**1.95**	7.19e-03	**1.97**
6.12e-04	1.16e-03	**1.91**	1.82e-03	**1.93**
3.08e-04	3.00e-04	-	4.83e-04	-

Figure 3.19: Thermal Couette flow on triangular meshes.

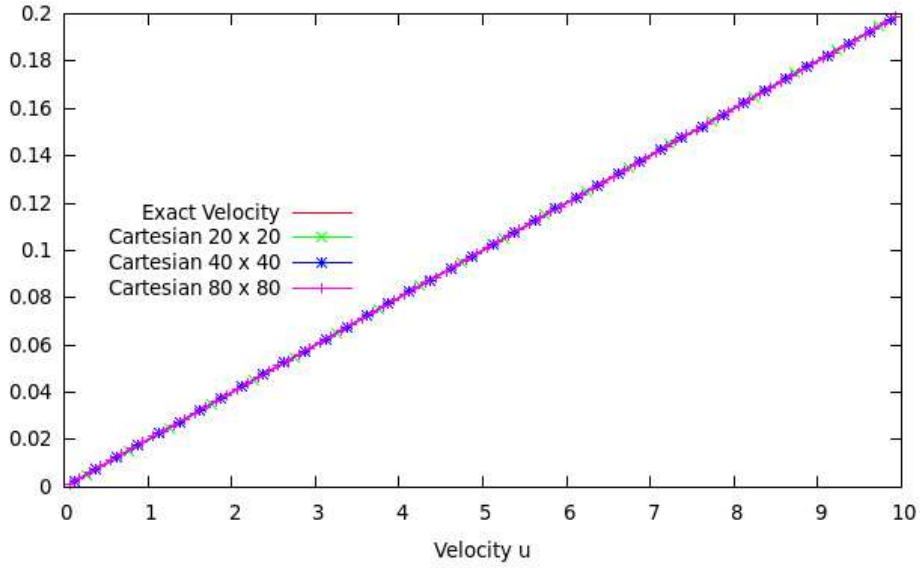


(a) Profile of temperatures.

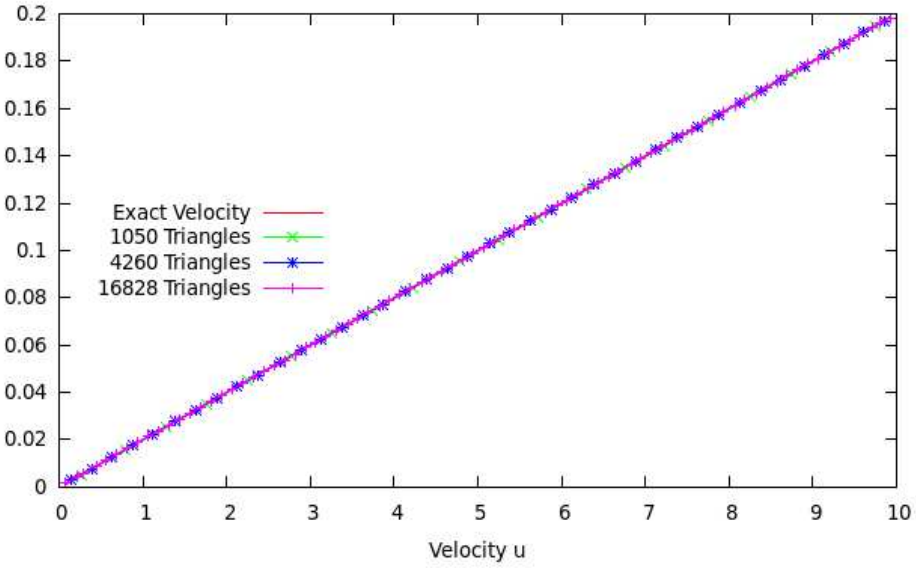


(b) Profile of temperature, zoom around the maximal value.

Figure 3.20: Thermal Couette flow comparison of the velocity.



(a) Profile of velocity on a series of Cartesian meshes.



(b) Profile of velocity on a series of triangular meshes.

Supersonic flow over an adiabatic flat plate

We are going to study an adiabatic flat plate in a supersonic flow [102] at the Mach number 1.7605. The freestream conditions are as follows:

$$\begin{aligned} V_\infty &= 500 m.s^{-1}, \\ \rho_\infty &= 10^{-3} kg.m^{-3}, \\ T_\infty &= 300 K, \\ M_a &= 1.7605 \end{aligned}$$

The flat plate has a length of one meter. We use the Sutherland formula (3.21) to compute the dynamic viscosity. We can compute the Reynolds Number using

$$R_e = \frac{\rho_\infty V_\infty L}{\mu_\infty}.$$

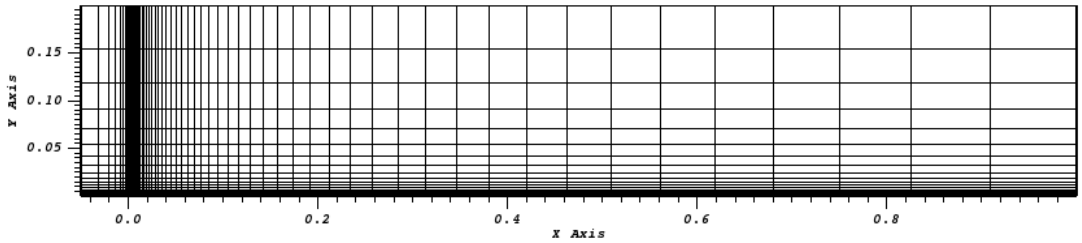
We have $L = 1$ and $\mu_\infty = \mu(T_\infty) \approx 1.7366 \cdot 10^{-5}$. Which yields

$$R_e = \frac{10^{-3} \cdot 500 \cdot 1}{1.7366 \cdot 10^{-5}} \approx 28800.$$

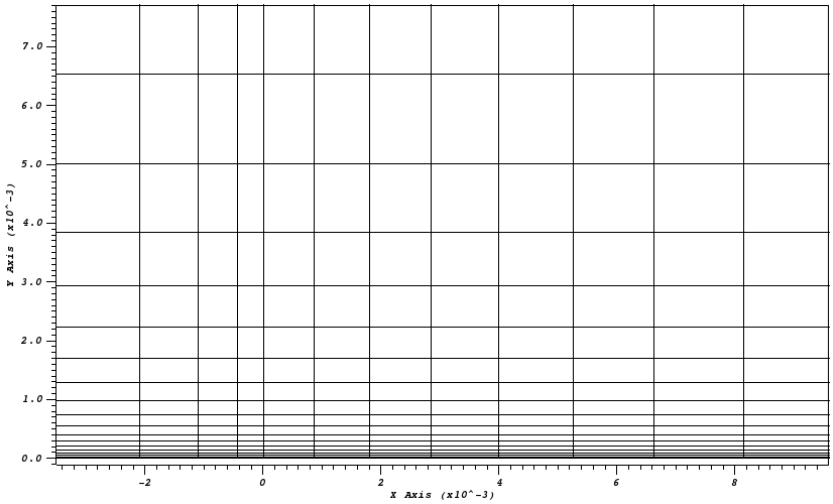
For the values of the specific heat ratio and Prandtl number, we use the classical values $\gamma = 1.4$ and $P_r = 0.71$. The flat plate is adiabatic so we have $\boldsymbol{q}_{wall} \cdot \boldsymbol{n} = 0$. We should note that the mesh used for the computation starts a bit before the plate itself (0.1m). This is to avoid the use of a complicated inflow condition. The boundary condition applied in the lower part of the mesh before the actual plate is a slipping wall.

The mesh is presented in Figure 3.21. We can see the refinement in the boundary layer. In Figure 3.22 we show the contours of temperature and velocity. In Figure 3.23 we have plotted the profiles of the velocity at different positions along the plate. We have also displayed the profiles of temperatures at different positions. The zoom in Figure 3.23 allows us to see that the adiabatic condition $\boldsymbol{q}_{wall} \cdot \boldsymbol{n} = 0$ is well respected on the plate.

Figure 3.21: Supersonic viscous flow over an adiabatic flat plate: Mesh.

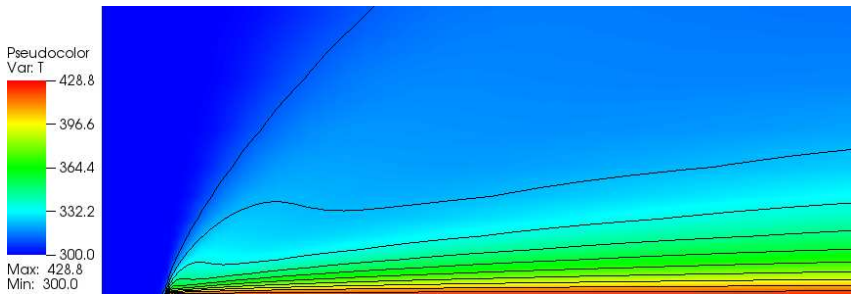


(a) Presentation of the mesh. 50×30 structured mesh. 50 cells along the plate and 30 cells perpendicular to the plate.

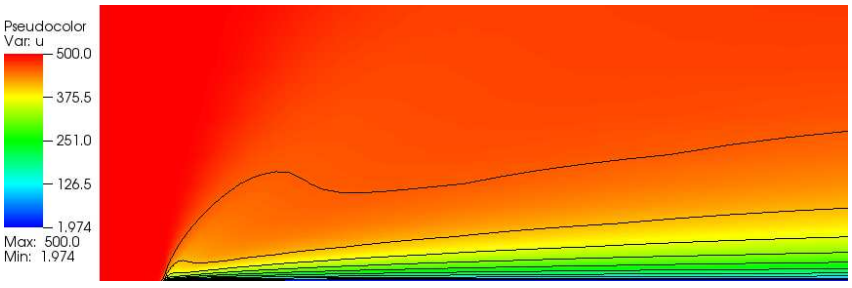


(b) Zoom of the mesh in the vicinity of the leading edge of the flat plate.

Figure 3.22: Supersonic viscous flow over an adiabatic flat plate : Contours.

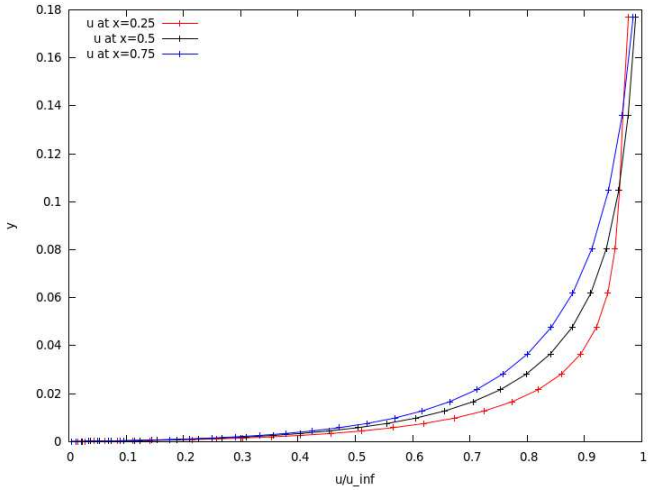


(a) Contours of temperature.

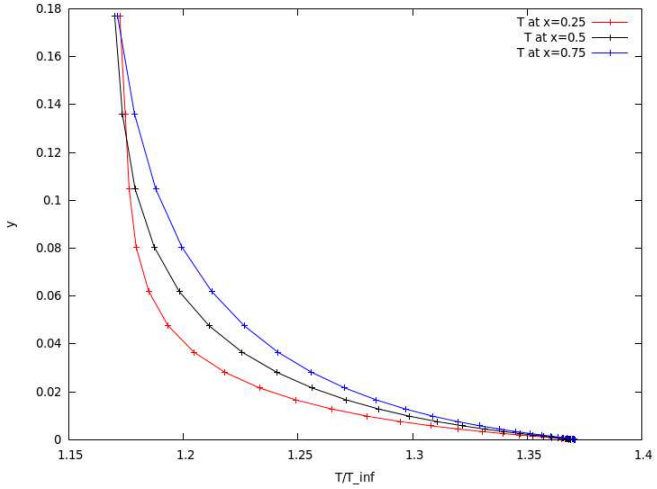


(b) Contours of velocity.

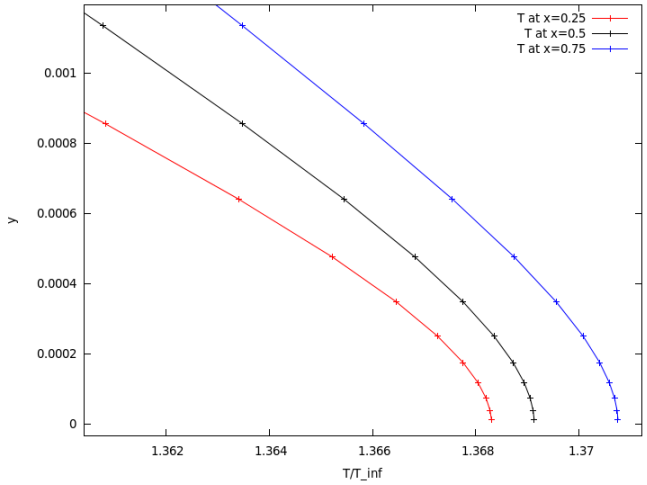
Figure 3.23: Supersonic viscous flow over an adiabatic flat plate : Profiles.



(a) Profile of velocity at different sections of the mesh.



(b) Profile of temperature at different sections of the mesh.



(c) Profile of temperature at different sections of the mesh. Zoom near the plate.

Supersonic flow around a cylinder at constant temperature

We are going to study a cylinder at constant temperature in a supersonic flow at the Mach number 17.605. This test case is used for validation purposes by the CFD codes FUN2D and LAURA from NASA and is described in [1]. The freestream conditions are as follows:

$$\begin{aligned} V_\infty &= 5000 m.s^{-1}, \\ \rho_\infty &= 10^{-3} kg.m^{-3}, \\ T_\infty &= 200K, \\ T_{wall} &= 500K, \\ M_a &= 17.605, \\ R_e &= 376930. \end{aligned}$$

The cylinder has a radius of one meter. We use the Sutherland formula (3.21) to compute the dynamic viscosity. We can verify that with this choice we obtain the expected Reynolds Number.

We recall that the Reynolds Number is obtained by

$$R_e = \frac{\rho_\infty V_\infty L}{\mu_\infty}.$$

We have $L = 1$ and $\mu_\infty = \mu(T_\infty) \approx 1.329 \cdot 10^{-5}$. Which yields

$$R_e = \frac{10^{-3} \cdot 5000 \cdot 1}{1.329 \cdot 10^{-5}} \approx 376930.$$

In [1] no information is given for the choice of the specific heat ratio or for the Prandtl number. We chose to use the classical values $\gamma = 1.4$ and $P_r = 0.71$ in our work. This can be the reason of the differences obtained between their results and ours.

In Figure 3.24 we picture the numerical results we obtained. We display the contour of temperature and contour of the scaled pressure $\bar{p} = \frac{p}{\rho_\infty \boldsymbol{V}_\infty^2}$. We display the same isovalues as the ones used for the representation of the results obtained by FUN2D and LAURA in Figure 3.25. In Figure 3.24 we also show the mesh we used for the computations and a zoom in the vicinity of the boundary layer.

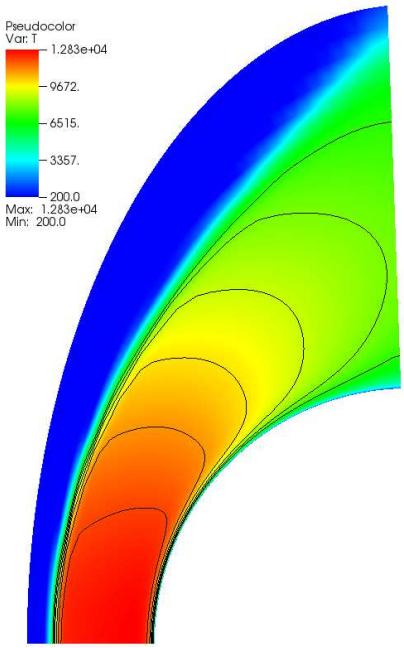
After this qualitative verification against the NASA codes we also make a more quantitative verification. In Figure 3.26 we present the pressure coefficient obtained around the cylinder. These results are compared with the ones obtained with FUN2D and LAURA. The pressure coefficients are compared to the modified Newtonian law presented in equation (3.106).
In Figure 3.27 we present the heating rate coefficient obtained around the cylinder. The results obtained with our scheme are compared with the ones obtained with FUN2D and LAURA. The definition of the heating rate coefficient is given by

$$C_H = \frac{\boldsymbol{q} \cdot \boldsymbol{n}}{\frac{1}{2}\rho_\infty V_\infty^3}. \tag{3.155}$$

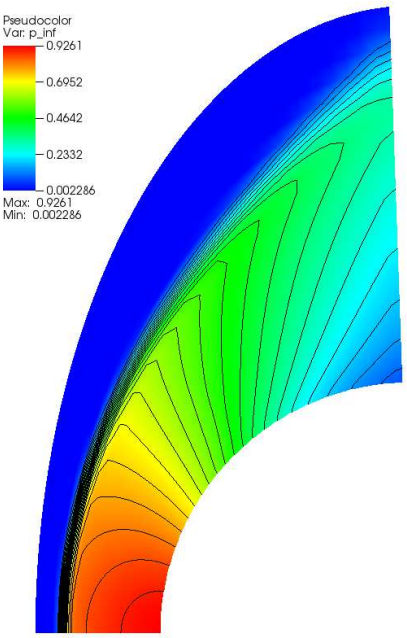
The results we obtain for the pressure coefficient are in good agreement with the ones obtained by FUN2D and LAURA. It is also interesting to note that the very simple Newtonian model also matches perfectly the results for angles below 50 degrees.
Concerning the heating rate coefficient we first need to comment the huge difference in the results

obtained with FUN2D and LAURA. At the leading edge FUN2D gives an overestimation of the heating fluxes which is 50 percent higher than the fluxes obtained with LAURA, the NASA reference code. This is why we are proud to present the results obtained with our code. First, using the same mesh used by FUN2D we obtain a good approximation of the heating rate near the leading edge. The difference with the results obtained on the structured and unstructured mesh remains very small and is mainly due to the non-symmetry of the unstructured mesh. However some differences can be spotted between our results and the results obtained with LAURA. These differences can come from different factors. For example, the test case lacks of some information. We have made some assumptions that can differ from the ones used in the two NASA codes. There is also different ways of computing the heating rate coefficient, which leads to apply the Sutherland formula with different temperatures. Due to the high variation of the temperature in the vicinity of the boundary layer, this can explain the differences between the heating rate coefficient obtained with LAURA and with our code.

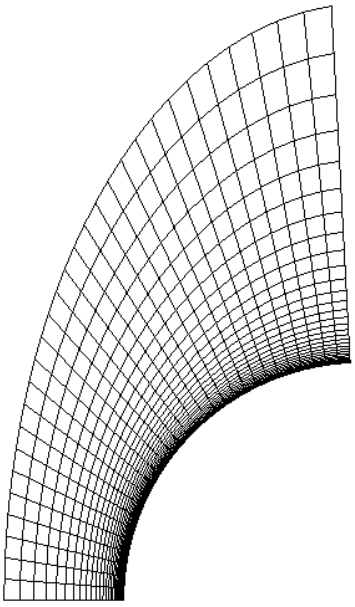
Figure 3.24: Supersonic viscous flow around a cylinder.



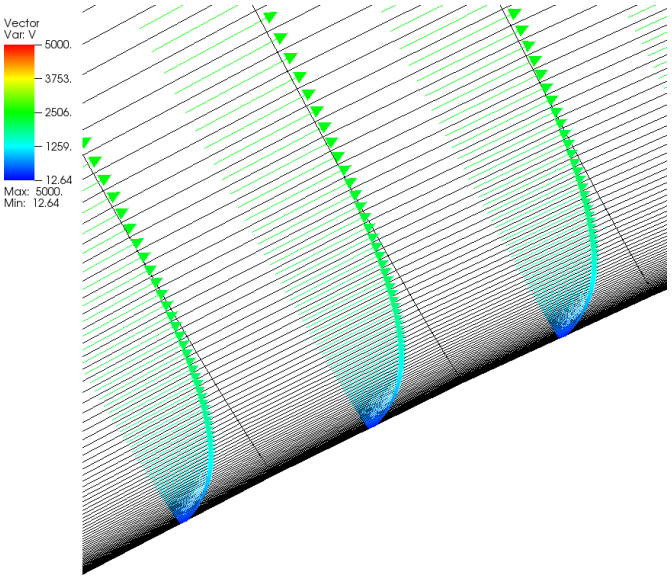
(a) Contours of temperature. Isolines ranging from 7000K to 12000K every 1000K.



(b) Contours of the scaled pressure $\bar{p} = \frac{p}{\rho_\infty \boldsymbol{V}_\infty^2}$. 20 isolines ranging from 0.1 to 0.9.

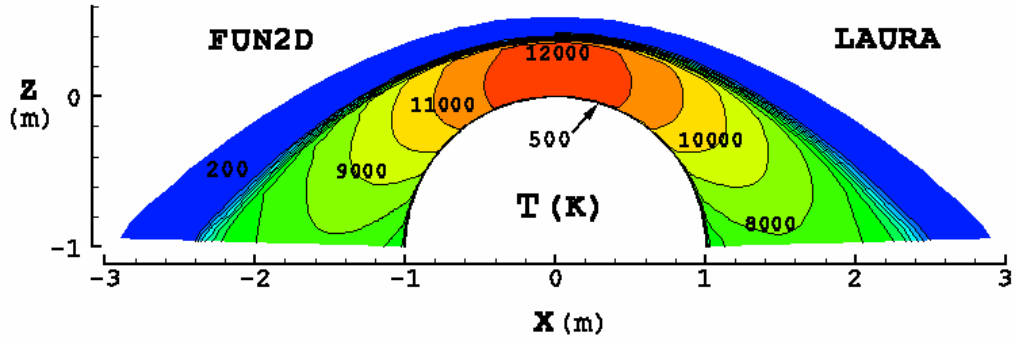


(c) Coarse mesh used for the computations. 32 cells along the cylinder, 64 cells in the direction normal to the cylinder.

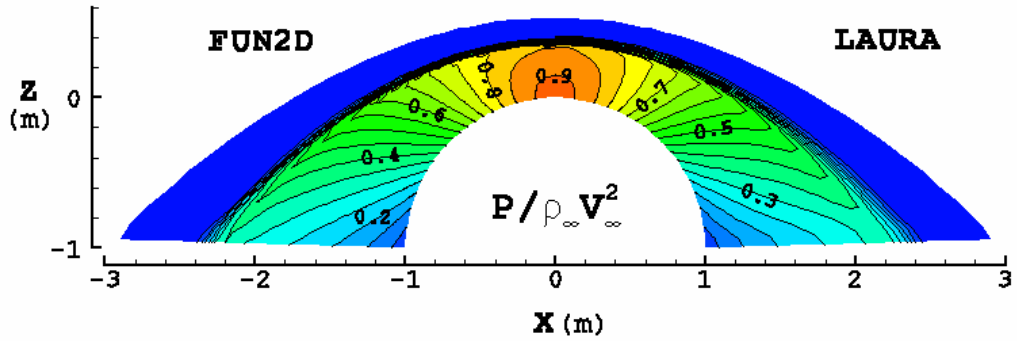


(d) Zoom in the vicinity of the boundary layer. Representation of the velocity vectors on the finest grid (64×128).

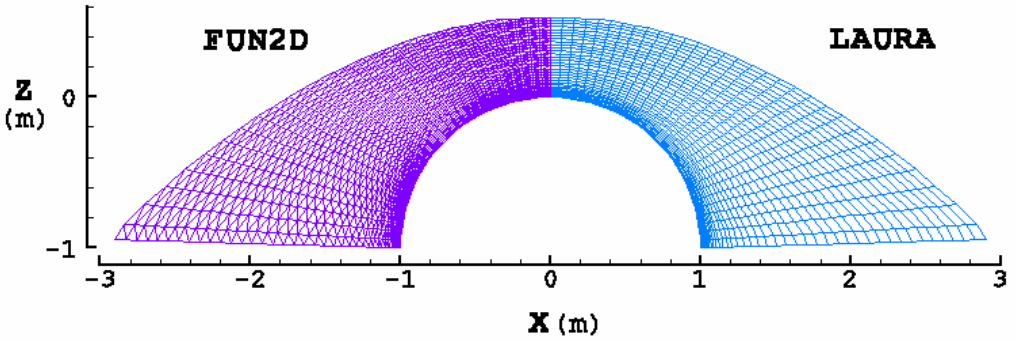
Figure 3.25: Supersonic viscous flow around a cylinder. Results from the NASA codes FUN2D and LAURA [1].



(a) Contours of temperature.



(b) Contours of the scaled pressure $\bar{p} = \frac{p}{\rho_\infty V_\infty^2}$.



(c) Presentation of the meshes used for the computations.

Figure 3.26: Supersonic viscous flow around a cylinder: Comparison of the pressure coefficient obtained with FUN2D and LAURA from NASA and with our code on the same meshes.

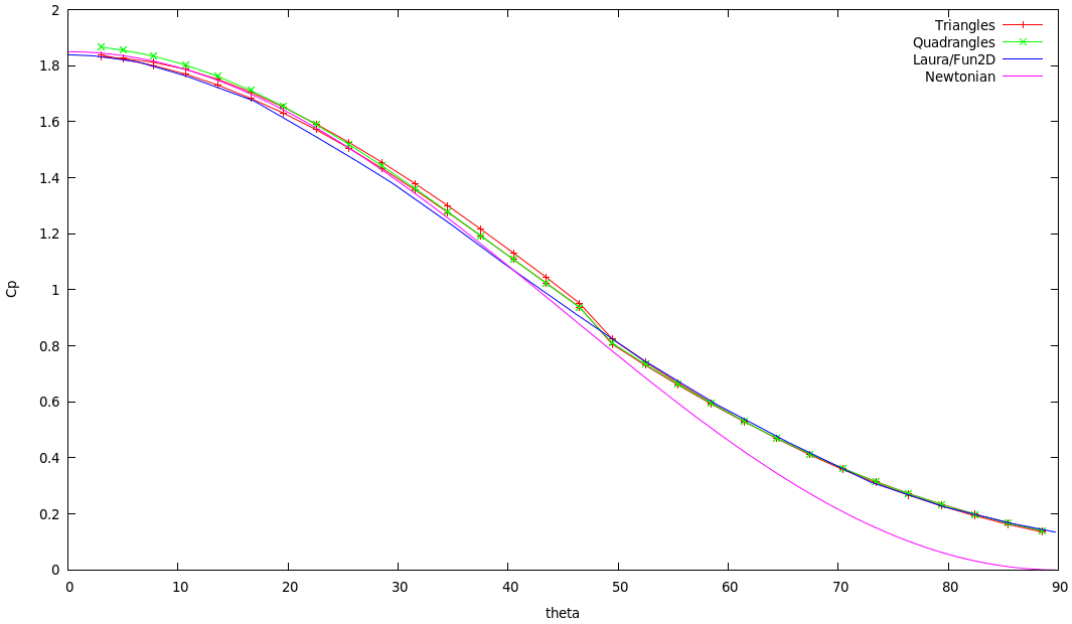
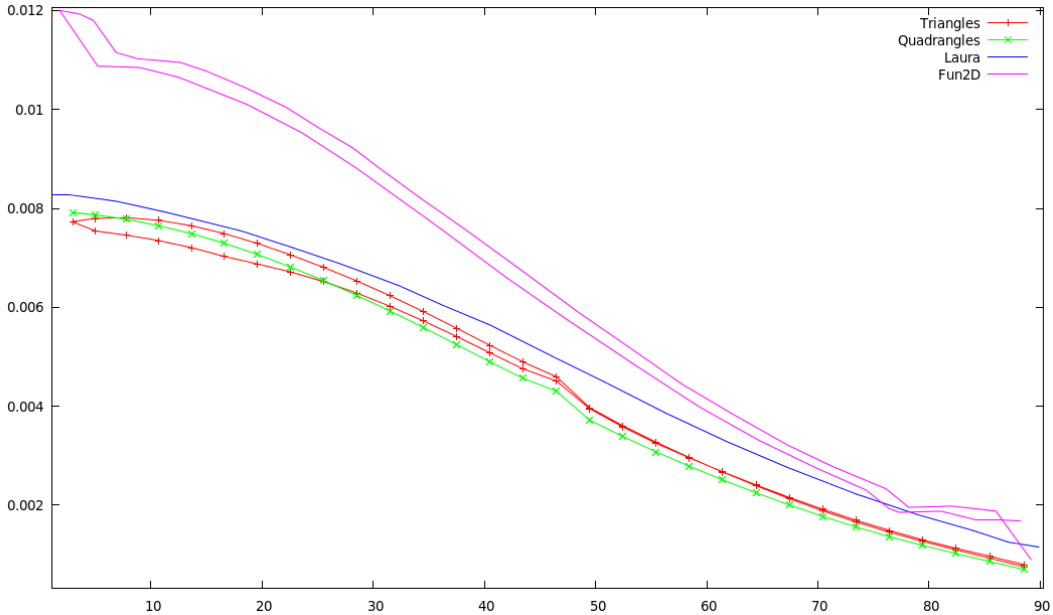


Figure 3.27: Supersonic viscous flow around a cylinder: Comparison of the heating rate coefficient obtained with FUN2D and LAURA from NASA and with our code on the same meshes.



3.5 Conclusion

In the beginning of this chapter we have constructed a classical cell-centered Finite Volume method for solving the Euler equations. This numerical scheme relies on the resolution of mono-dimensional Riemann problems along the directions defined by the normals of the cell interfaces. These Riemann problems are solved with the help of approximate Riemann solvers. We showed that some of these approximate solvers could introduce nonphysical phenomenons usually called carbuncles in the presence of strong shocks aligned with the directions of the mesh. We have presented a possible cure to this problem through the use of rotated Riemann solvers able to switch from an accurate Riemann solver in smooth regions to a more diffusive solver in the presence of strong shocks. We also mentioned the higher-order space discretization that we have developed. It is based on the classical MUSCL reconstruction method and can be used with slope limiters developed by Barth-Jespersen, Venkatakrishnan and Michalak. We then discussed the time discretization of the numerical scheme. We have developed a second order explicit Runge-Kutta method and a backward Euler implicit method. Then we were able to asses the accuracy of the our scheme by running a series of validation cases. We compared our results with exact solutions or to the results obtained with other computational codes.
We then presented a numerical cell-centered Finite Volume method for solving the Navier-Stokes equations. Due to the very small cell sizes needed to capture the thin boundary layers that appears in viscous supersonic flows, the time limitation induced by the CFL condition of an explicit scheme forced us to develop an implicit version of the scheme in order to reach steady-state. The Euler scheme developed earlier was used as the starting point for the construction of this numerical scheme. We then used the numerical schemes developed in chapter 1 and 2 to discretize the viscous fluxes. We explained how to append the contributions of these schemes to the matrix and to the right-hand side of the global system describing the implicit numerical scheme. This had lead us to construct a new kind of cell-centered Finite Volume Navier-Stokes numerical scheme, characterized by an accurate discretization of the viscous fluxes. We then presented three tests cases that allowed us to assess the accuracy and robustness of the discretization of the viscous fluxes. We finished the presentation by successfully showing some comparative results with two CFD codes developed by NASA on a complete supersonic viscous test case.

Conclusion and perspectives

General conclusion

We started this dissertation by the presentation of a Cell-Centered Finite Volume scheme which solves anisotropic diffusion on unstructured meshes. We first presented the CCLAD scheme of Maire and Breil [32, 90]. Then, we extended this two-dimensional scheme to three-dimensional geometries on unstructured grids. This extension was not trivial due to some specifically two-dimensional mesh properties used by the original scheme. Then, we have discussed the mathematical properties of this scheme and showed by means of various numerical experiments that it is a nominaly second-order accuracte numerical method. To cope with the evolution of modern supercomputers, we presented the parallel version of this scheme. Due to a compact stencil the efficiency of this parallel implementation is rather good, which is an important factor to take into account when designing numerical methods.

Then, we developed a Cell-Centered Finite Volume scheme which solves tensorial diffusion on unstructured meshes. This equation corresponds to the viscous fluxes present in the momentum equation of the Navier-Stokes equations. In order to apply the CCLAD methodology we needed to introduce a penalization of the original constitutive law. Indeed, after studying the mathematical properties of this constitutive law, we were able to conclude on its non invertibility on the space of generic second-order tensors. Using the methodology introduced by Arnold [16], we added a divergence free term in the constitutive law, which rendered it invertible while having no influence on the original equation. The CCLAD methodology was then successfully applied to this modified constitutive law, which lead us to the construction of an innovative numerical scheme. Once again, we observed a nominally second-order accuracy using numerous numerical tests cases. The parallel implementation was easily obtained from the parallel CCLAD implementation, with the introduction of block-matrices and block-vectors.

Finally, we discussed of the development of a Cell-Centered Finite Volume scheme which solves the Navier-Stokes equations on unstructured meshes. We started the Chapter by describing the construction of a second-order scheme to solve the Euler equations. The construction of this scheme follows the classical Cell-Centered MUSCL approach with the use of approximate Riemann solvers. An explicit Runge-Kutta method of order two is presented along with the implicit backward-Euler time discretization. The novelty of the methodology lies in the construction of the Navier-Stokes scheme. We used the numerical schemes developed in Chapter 1 and 2 to discretize the viscous terms of the equations. The parallelization of this method was obtain without further effort by reusing the parallel implementation of the tensorial diffusion scheme. This Chapter ended with the presentation of various tests cases that assessed the accuracy and robustness of this numerical scheme. The scheme was also successfully compared to two state-of-the-art CFD solvers from NASA [1], on a characteristic test case.

To put it in a nutshell, we used the CCLAD methodology and applied it to various equations. It allowed us to build an accurate numerical scheme to solve anisotropic diffusion equation, which allows us to model the heat transfer inside a reentry vehicle. Due to the use of unstructured meshes, we can easily handle the complex geometries associated with the multi-materials construction of the Thermal Protection System of this kind of objects. Using the CCLAD methodology also allowed us to construct an original Cell-Centered Finite Volume discretization for solving the Navier-Stokes equations. This numerical method is robust and accurate and allows us to model the fluid flow around the reentry vehicle. The use of unstructured grids is also very well suited to the numerical modeling of fluid flows around complex geometries.

Perspectives

Putting an end to this thesis gives me mixed feelings. On one hand it is rewarding to settle down and take a look on all the work that has been done during these three and a half years. Of course this manuscript represents only a subset of the work we really produced. Not all of our developments had room in this dissertation, and, many of the paths we explored where finally abandoned. Not knowing which road to take is one of the many things that keep research interesting. On the other hand, it is frustrating to have to put an end to this work. We just reached a point where very interesting topics can be studied, and, were small developments could allow us to make a huge step forward in the modeling of the whole problem. In fact, this may also be one of the interests of a PhD thesis, raising more questions than it answers. In the following we give a quick overview of some of these perspectives.

In our work we used the Ideal Gas model as the Equation Of State (EOS) for the fluids. However this model does not apply for the very high temperatures reached in hypersonic viscous flows. In Anderson [15] the author presents different EOS that are better suited for the high-temperature gas dynamics. These models produce more accurate results, the temperatures reached in the fluid around the hypersonic cylinder test case with these kinds of methods are much lower than the ones predicted with the Ideal Gas model. These models are called Real Gas models and usually consider that the gas state is defined as an equilibrium of the reactions between multiple species. These models should be straightforward to implement in our code, where the EOS are taken into account in a generic way. A set of functions needs to be defined in order to compute the appropriate values or derivatives to be used in the numerical scheme.

An other interesting improvement of this work would be the extension to three-dimensional geometries of the tensorial diffusion. As we explained in Chapter 2, the methodology used for the development of the tensorial diffusion scheme follows the construction of the CCLAD scheme, the extension to three dimensional geometries should then be straightforward. With these developments, we should also be able to extend easily the Navier-Stokes numerical scheme to three dimensional geometries. Finally, we point out one more time the fact that the parallel version of all these methods comes free of charge.

When we introduced the numerical scheme for anisotropic diffusion we explained that the purpose was to compute the heat transfer that occurs inside a reentry vehicle. These heat transfer occurs because of the heating induced by the fluid. One of the next step toward the resolution of the global problem is the study of the coupling between the heat transfer in the solid and the equations of Navier-Stokes. This can be done in different ways. During the construction of the Navier-Stokes equation we build the matrix associated to the heat transfer in the fluid. Instead of building this matrix only in the fluid we could build it in the whole domain composed of

the fluid and the solid. The matrix subset corresponding to the fluid could then be used in the Navier-Stokes scheme, and the matrix subset corresponding to the solid portion could be used "as is". This would allow us to build a strong coupling of the heat transfer at the interface between the fluid and the solid. An other way to perform the coupling is to do it in a weak form. The numerical systems in the fluid and in the solid are constructed independently. Then an iterative procedure at the interface tries to determine at the appropriate temperatures which leads to an equilibrium of the thermal fluxes.

All these perspectives leads to the resolution of the global problem we described in the introduction. Namely, the coupling of Navier-Stokes equations and heat transfer through an ablation model. We already started to work toward this direction by studying a simplified ablation-like problem. As a model problem for ablation we started to study the problem of Stefan [91] which deals with ice melting issues. This work was done with the help of the internship of Pierre Cantin [37]. He studied this problem in one-dimensional geometries, and managed to develop an iterative procedure to obtain the energy balance at the moving interface between ice and water. Following this methodology, we then have extended his work to multi-dimensions. It lead us to the development an Arbitrary Lagrangian Eulerian version to the CCLAD scheme that preserve the Discrete Geometrical Conservation Law on moving meshes. The mesh motion is only given by the model at the melting interface. The motion of the rest of the mesh is performed by a numerical scheme that solve an elastodynamic problem at the nodes of the mesh. These developments were presented in the YIC2013 conference. They still requires some improvements but will be a good starting point for the full Navier-Stokes, ablation and heat transfer coupling.

Appendix A

Using pyramid cells in the three-dimensional anisotropic diffusion scheme

Pyramid cells are required to construct a conformal partition of a computational domain made of tetrahedral and hexahedral cells. Indeed, the pyramid cells allow to make the transition between the tetrahedral zones and the hexahedral zones. In this case, we have to slightly modify our finite volume scheme to take into account the fact that pyramids are cells for which the number of faces incident to one vertex is strictly greater than 3. We describe the needed modifications by considering a generic pyramid ω_c and we denote by p the vertex characterized by $\mathsf{F}_{pc} = 4$, where F_{pc} denotes the number of faces of cell c impinging at point p, refer to Figure A.1. Knowing that $\mathsf{F}_{pc} = 4$ faces are incident to the vertex p, the decomposition of a vector in terms of its normal components within sub-cell ω_{pc}, refer to Paragraph 1.3.2, is not possible. Indeed, the number of equations, *i.e.*, $\mathsf{F}_{pc} = 4$, being greater than the number of unknowns, *i.e.*, the 3 Cartesian components of the vector under consideration, we end up with an overdetermined system.

To overcome this difficulty, we subdivide the sub-cell ω_{pc} into the $\mathsf{F}_{pc} = 4$ fictive sub-cells ω_{pcf} defined by

$$\omega_{pcf} = \bigcup_{e \in \mathcal{E}(p,f)} \mathcal{I}^{pfe}, \quad \text{for } f \in \mathcal{F}(p,c).$$

Here, $\mathcal{E}(p,f)$ is the set of edges of face f impinging at point p. Namely, being given a face f incident to the vertex p, the sub-cell ω_{pcf} is constructed by gathering the two iota tetrahedra attached to the two edges of face f incident to point p. We observe that there is one fictive sub-cell, ω_{pcf}, per face impinging at vertex p. Each fictive sub-cell ω_{pcf} has 3 faces impinging at node p: the outer sub-face $\partial\omega_{pc}^{f}$ and two inner sub-faces which result from the subdivision. Bearing this in mind, we can employ (1.51) to write the flux approximation within each fictive sub-cell ω_{pcf}. Having added the supplementary fictive sub-cells, the number of sub-cells surrounding point p, which was equal to C_p, becomes equal to $\mathsf{C}_p^{\triangle} = \mathsf{C}_p + \mathsf{F}_{pc} - 1$. Here, without loss of generality, we suppose that there is only one pyramid in the set of cells surrounding vertex p. Regarding the number of faces incident to vertex p, it was equal to F_p and becomes equal to $\mathsf{F}_p^{\triangle} = \mathsf{F}_p + \mathsf{F}_{pc}$. Therefore, at the vertex p, the vector of sub-face temperatures, $\bar{\boldsymbol{T}}^{\triangle}$, is of size $\mathsf{F}^{\triangle}$ and the vector of cell-centered temperatures, $\boldsymbol{T}^{\triangle}$, is of size $\mathsf{C}_p^{\triangle}$. Utilizing the flux approximation (1.51) and enforcing the normal flux continuity across the cell interfaces surrounding vertex p in the same manner than in Paragraph 1.3.4 leads to the linear system satisfied by the sub-face

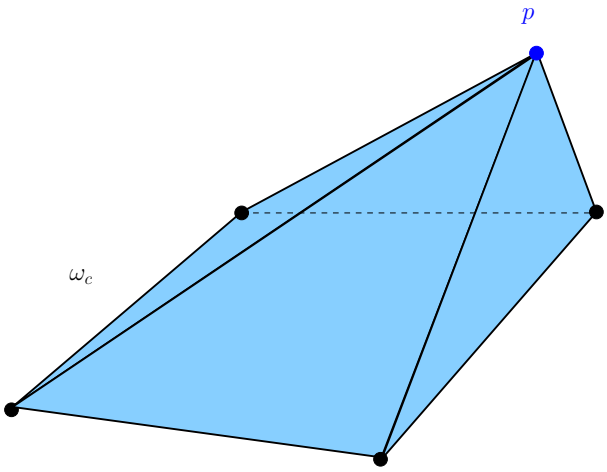


Figure A.1: Sketch of a pyramid cell.

temperatures

$$\mathbb{N}^{\Delta}\bar{\boldsymbol{T}}^{\Delta} = \mathbb{S}^{\Delta}\boldsymbol{T}^{\Delta}.$$

Here, $\mathbb{N}^{\Delta}$ and $\mathbb{S}^{\Delta}$ are respectively matrices of size $\mathsf{F}_p^{\Delta} \times \mathsf{F}_p^{\Delta}$ and $\mathsf{F}_p^{\Delta} \times \mathsf{C}_p^{\Delta}$ which are constructed in the same way than in Paragraph 1.3.4. The matrix $\mathbb{N}^{\Delta}$ is invertible, refer to Paragraph 1.4.1, and the solution of the above linear system writes

$$\bar{\boldsymbol{T}}^{\Delta} = \left(\mathbb{N}^{\Delta}\right)^{-1} \mathbb{S}^{\Delta}\boldsymbol{T}^{\Delta}.$$

This formula allows to express the sub-face temperatures p in terms of the cell-centered temperatures surrounding vertex p. Finally, using the same procedure than in Paragraph 1.4.2, the contribution of cell c to the diffusion flux at vertex p writes

$$Q_{pc} = -\sum_{d \in \mathcal{C}^{\Delta}(p)} \mathbb{G}_{cd}^{p,\Delta}(T_d^{\Delta} - T_c^{\Delta}), \tag{A.1}$$

where $\mathcal{C}^{\Delta}(p)$ is the set of cells surrounding vertex p including the fictive sub-cells. The $\mathsf{C}_p^{\Delta} \times \mathsf{C}_p^{\Delta}$ matrix $\mathbb{G}^{p,\Delta}$ is given by $\mathbb{G}^{p,\Delta} = \left(\widetilde{\mathbb{S}}^{\Delta}\right)^t \left(\mathbb{N}^{\Delta}\right)^{-1} \mathbb{S}^{\Delta}$, refer to Paragraph 1.4.2 for the definition of $\widetilde{\mathbb{S}}$. We point out that the cell index, d, employed in (A.1), can refer to a fictive sub-cell. More precisely, Q_{pc} contains contributions coming from temperatures attached to the fictive sub-cells. These supplementary degrees of freedom are eliminated equating them to the cell temperature T_c. This amounts to express the vector of the cell-centered temperatures including the temperatures of the fictive sub-cells, $\boldsymbol{T}^{\Delta} \in \mathfrak{R}^{\mathsf{C}_p^{\Delta}}$, in terms of the initial vector of the cell-centered temperatures $\boldsymbol{T} \in \mathfrak{R}^{\mathsf{C}_p}$ as follows

$$\boldsymbol{T}^{\Delta} = \mathbb{P}\boldsymbol{T}. \tag{A.2}$$

Here, $\mathbb{P}$ is a rectangular matrix of size $\mathsf{C}_p^{\Delta} \times \mathsf{C}_p$. Let i (resp. j) be the generic index of a cell in the local numbering of the cells belonging to $\mathcal{C}^{\Delta}(p)$ (resp. $\mathcal{C}(p)$), then according to (A.2), temperature T_i^{Δ} writes

$$T_i^{\Delta} = \sum_{j=1}^{\mathsf{C}_p} \mathbb{P}_{ij} T_j.$$

For $i = 1 \dots \mathsf{C}_p^{\triangle}$ and $j = 1 \dots \mathsf{C}_p$, the generic entry of $\mathbb{P}$ writes

$$\mathbb{P}_{ij} = \begin{cases} 1 & \text{if } i \text{ corresponds to a fictive sub-cell of } c \text{ and } j \text{ corresponds to cell } c, \\ 1 & \text{if } i \text{ corresponds to cell } c \text{ and } j \text{ corresponds to cell } c, \\ 0 & \text{elsewhere.} \end{cases}$$

Finally, substituting (A.2) into (A.1) leads to the expression of Q_{pc} in terms of cell-centered temperatures

$$Q_{pc} = - \sum_{d \in \mathcal{C}(p)} \mathbb{G}_{cd}^{p} (T_d - T_c), \tag{A.3}$$

where $\mathbb{G}_{cd}^{p} = \mathbb{P}^t \mathbb{G}^{p,\triangle} \mathbb{P}$. It is worth pointing out that the definition of the global diffusion matrix remains unchanged.

We have described the above modification in the particular case of a pyramid but there is nothing to prevent us from applying it to general polyhedral cells.

Bibliography

[1] FUN3D Web page: Hypersonic benchmarks. Available at `http://fun3d.larc.nasa.gov/chapter-9.html#hypersonic_benchmarks`.

[2] PlaFrim Web page. Available at `https://plafrim.bordeaux.inria.fr/doku.php?id=start`.

[3] I. Aavatsmark, T. Barkve, O. Boe, and T. Mannseth. Discretization on unstructured grids for inhomogeneous, anistropic media. Part I: derivation of the methods. *SIAM J. Sci. Comput.*, 19:1700–1716, 1998.

[4] I. Aavatsmark, T. Barkve, O. Boe, and T. Mannseth. Discretization on unstructured grids for inhomogeneous, anistropic media. Part II: discussion and numerical results. *SIAM J. Sci. Comput.*, 19:1717–1736, 1998.

[5] I. Aavatsmark, T. Barkve, and T. Mannseth. Control volume discretization methods for 3D quadrilateral grids in inhomogeneous, anisotropic reservoirs. *SPE J.*, 3:146–154, 1998.

[6] I. Aavatsmark, G.T. Eigestad, B.-O. Heimsund, B.T. Mallison, J.M. Nordbotten, and E. Øian. A new finite-volume approach to efficient discretization on challenging grids. *SPE J.*, 15:658–669, 2010.

[7] I. Aavvatsmark, G.T. Eigestad, R.A. Klausen, M.F. Wheeler, and I. Yotov. Convergence of a symmetric MPFA method on quadrilateral grids. Technical Report TR-MATH 05-14, University of Pittsburgh, 2005.

[8] R. Abgrall. Toward the ultimate conservative scheme: following the quest. *J. Comput. Phys.*, 167:277–315, 2001.

[9] R. Abgrall, G. Baurin, P. Jacq, and M. Ricchiuto. Some examples of high order simulations parallel of inviscid flows on unstructured and hybrid meshes by residual distribution schemes. *Computers and Fluids*, 61(0):6 – 13, 2012.

[10] R. Abgrall, R. Butel, P. Jacq, C. Lachat, X. Lacoste, A. Larat, and M. Ricchiuto. Implicit strategy and parallelization of a high order residual distribution scheme. In *ADIGMA - A European Initiative on the Development of Adaptive Higher-Order Variational Methods for Aerospace Applications*, Notes on Numerical Fluid Mechanics and Multidisciplinary Design, pages 209–224. Springer (Kluwer Academic Publishers), 2010.

[11] R. Abgrall and D. de Santis. Linear and non-linear high order accurate residual distribution schemes for the discretization of the steady compressible Navier-Stokes equations. *J. Comput. Phys.* in revision.

[12] R. Abgrall and D. de Santis. High-order preserving residual distribution schemes for advection-diffusion scalar problems on arbitrary grids. *SIAM J. Sci. Comput.*, 2014. in press.

[13] L. Agelas and R. Masson. Convergence of the finite volume MPFA O-scheme for heterogeneous anisotropic diffusion problems on general meshes. *Comptes Rendus Mathématique*, (346):1007–1012, 2008.

[14] G. Amdahl. Validity of the single processor approach to achieving large-scale computing capabilities. *AFIPS Conference Proceedings*, (30):483–485, 1967.

[15] J. D. Anderson. *Hypersonic and high-temperature gas dynamics*. AIAA education series. American Institute of Aeronautics and Astronautics, 2006.

[16] D. N. Arnold and R. S. Falk. A new mixed formulation for elasticity. *Numer. Math*, 53:1–2, 1988.

[17] S. Atzeni and J. Meyer-Ter-Vehn. *The physics of inertial fusion*. Oxford Science publications, Oxford University Press, 2004.

[18] S. Balay, J. Brown, K. Buschelman, V. Eijkhout, W. D. Gropp, D. Kaushik, M. G. Knepley, L. C. McInnes, B. F. Smith, and H. Zhang. PETSc users manual. Technical Report ANL-95/11 - Revision 3.3, Argonne National Laboratory, 2012.

[19] S. Balay, J. Brown, K. Buschelman, W.D. Gropp, D. Kaushik, M.G. Knepley, L.C. McInnes, B.F. Smith, and H. Zhang. PETSc Web page, 2012. Available at `http://www.mcs.anl.gov/petsc`.

[20] S. Balay, W. D. Gropp, L. C. McInnes, and B. F. Smith. Efficient management of parallelism in object oriented numerical software libraries. In E. Arge, A. M. Bruaset, and H. P. Langtangen, editors, *Modern Software Tools in Scientific Computing*, pages 163–202. Birkhäuser Press, 1997.

[21] T. J. Barth. Numerical methods for conservation laws on structured and unstructured meshes. Technical report, VKI Lecture Series, 2003.

[22] T. J. Barth and D. C. Jespersen. The design and application of upwind schemes on unstructured meshes. In *AIAA paper 89-0366*, 27th Aerospace Sciences Meeting, Reno, Nevada, 1989.

[23] F. Bassi and S. Rebay. A high-order accurate discontinuous finite element method for the numerical solution of the compressible Navier-Stokes equations. *J. Comput. Phys.*, 131(2):267–279, 1997.

[24] F. Bassi and S. Rebay. High-order accurate discontinuous finite element solution of the 2D Euler equations. *J. Comput. Phys.*, 138(2):251–285, 1997.

[25] P. Batten, N. Clarke, C. Lambert, and D. M. Causon. On the choice of wavespeeds for the HLLC Riemann solver. *J. Comput. Phys.*, 18:1553–1570, 1997.

[26] P. Batten, M.A. Leschziner, and U.C. Goldberg. Average-state Jacobians and implicit methods for compressible viscous and turbulent flows. *J. Comput. Phys.*, 137(1):38 – 78, 1997.

[27] M. Berger, S. M. Murman, and M. J. Aftosmis. Analysis of slope limiters on irregular grids. *43rd AIAA Aerospace Sciences Meeting*, 2005.

[28] D. Bianchi. *Modeling of ablation phenomena in space applications*. PhD thesis, Universitá degli Studi di Roma, 2007.

[29] D. Bianchi, F. Nasuti, and E. Martelli. Navier-Stokes simulations of hypersonic flows with coupled graphite ablation. *Journal of Spacecraft and Rockets*, 47(4), 2010.

[30] M. Billaud, G. Gallice, and B. Nkonga. A simple stabilized finite element method for solving two phase compressible-incompressible interface flows. *Comput. Methods Appl. Mech. Engrg.*, 200(9-12):1272 – 1290, 2011.

[31] P. Birken. *Numerical methods for the unsteady compressible Navier-Stokes equations*. Hdr thesis, Universität Kassel, 2012.

[32] J. Breil and P.-H. Maire. A cell-centered diffusion scheme on two-dimensional unstructured meshes. *J. Comput. Phys.*, 224(2):785–823, 2007.

[33] F. Brezzi, K. Lipnikov, M. Shashkov, and V. Simoncini. A new discretization methodology for diffusion problems on generalized polyhedral meshes. *Comput. Methods Appl. Mech. Engrg.*, 196:3682–3692, 2007.

[34] A. Burbeau, P. Sagaut, and C. H. Bruneau. A problem-independent limiter for high-order Runge-Kutta discontinuous Galerkin methods. *J. Comput. Phys.*, 169:111–150, 2011.

[35] D.E. Burton. Multidimensional discretization of conservation laws for unstructured polyhedral grids. Technical Report UCRL-JC-118306, Lawrence Livermore National Laboratory, 1994.

[36] J.-P. Caltagirone. Conservation laws in discrete mechanics. Technical report, Université de Bordeaux, Institut de Mécanique et d'Ingénierie, 2014. `http://hal.archives-ouvertes.fr/hal-00927279`.

[37] P. Cantin. *Méthode des Volumes Finis sur maillage mobile pour la résolution de problèmes issus de la rentrée atmosphérique*. Projet de Fin d'Études, Ensta-Supaero, 2013.

[38] A. J. Chorin and J. E. Marsden. *A Mathematical Introduction to Fluid Mechanics*. Springer, third edition, 2010.

[39] B. Cockburn, G.E. Karniadakis, and C.W. Shu. Discontinuous Galerkin methods: theory, computation and application. *Lecture notes in computational science and engineering*, 2000. Springer, Berlin.

[40] S. F. Davis. A rotationally biased upwind difference scheme for the Euler equations. *J. Comput. Phys.*, 56(1):65 – 92, 1984.

[41] B. Diskin, J. L. Thomas, E. J. Nielsen, H. Nishikawa, and J. A. White. Comparison of node-centered and cell-centered unstructured finite-volume discretizations. part I: Viscous fluxes. *American Institute of Aeronautics and Astronautics*, 48:1326–1338, 2010.

[42] K. Domelovo and P. Omnes. A finite volume for the Laplace equation on almost arbitrary two-dimensional grids. *Mathematical Modelling and Numerical Analysis*, 39(6):1203–1249, 2005.

[43] B. Einfeldt. On Godunov-type methods for gas dynamics. *SIAM J. Numer. Anal.*, 25:294–318, 1988.

[44] L. Euler. Principes généraux du mouvement des fluides. *Mémoires de l'Academie des Sciences de Berlin*, 11:274–315, 1757.

[45] R. Eymard, T. Gallouët, and R. Herbin. *Finite Volume methods. Handbook of Numerical Analysis*. Elsevier Sciences, 2000.

[46] L. Flandrin. *Cell-centered methods for Euler and Navier-Stokes computations on unstructured meshes*. PhD thesis, Université Bordeaux I, 1995.

[47] G. Gallice. Schémas de type godunov entropiques et positifs préservant les discontinuités de contact. *Comptes Rendus de l'Académie des Sciences - Series I - Mathematics*, 331(2):149 – 152, 2000.

[48] G. Gallice. Solveurs simples positifs et entropiques pour les systèmes hyperboliques avec terme source. *Comptes Rendus Mathématique*, 334(8):713 – 716, 2002.

[49] P. Germain. *Mécanique*, volume I. Ellipses, 1986.

[50] C. Geuzaine and J.-F. Remacle. Gmsh: a three-dimensional finite element mesh generator with built-in pre- and post-processing facilities. *International Journal for Numerical Methods in Engineering*, 79(11):1309–1331, 2009.

[51] U. Ghia, S. Bayyuk, S. Habchi, C. Roy, T. Shih, T. Conlisk, C. Hirsch, and J. M. Powers. The AIAA code verification project - test cases for CFD code verification. *Aerospace Sciences Meetings*, 2010.

[52] E. Godlewski and P.-A. Raviart. *Hyperbolic Systems of Conservation Laws*. Springer Verlag, 2000.

[53] S. K. Godunov. A difference scheme for numerical solution of discontinuous solution of hydrodynamic equations. *Math. Sbornik*, 47:271–306, 1959.

[54] D. Gunasekera, P. Childs, J. Herring, and J. Cox. A multi-point flux discretization scheme for general polyhedral grids. In *Proceedings of the SPE International Oil and Gas Conference and Exhibition in China*, number SPE 48855, Beijing, China, 1998.

[55] M.E. Gurtin, E. Fried, and L. Anand. *The Mechanics and Thermodynamics of Continua*. Cambridge University Press, 2009.

[56] T. Harribey. *Simulation numérique directe de la turbulence en présence d'une paroi ablatable*. PhD thesis, ISAE Toulouse, 2011.

[57] A. Harten. High resolution schemes for hyperbolic conservation laws. *J. Comput. Phys.*, 49(3):357 – 393, 1983.

[58] A. Harten, B. Engquist, S. Osher, and S.R. Chakravarthy. Uniformly high order accurate essentially non-oscillatory schemes, III. *J. Comput. Phys.*, 71(2):231 – 303, 1987.

[59] A. Harten, P.D. Lax, and B. van Leer. On upstream differencing and Godunov-type schemes for hyperbolic conservation laws. *SIAM Review*, 25(1):35–61, 1983.

[60] R. Hartmann. Adaptive discontinuous Galerkin methods with shock-capturing for the compressible Navier-Stokes equations. *Int. J. Numer. Meth. Fluids*, 51:1131–1156, 2006.

[61] P. Hénon, P. Ramet, and J. Roman. On finding approximate supernodes for an efficient block-ILU(k) factorization. *Parallel Computing*, 34(6 - 8):345 – 362, 2008. Parallel Matrix Algorithms and Applications.

[62] F. Hermeline. A finite volume method for the approximation of diffusion operators on distorded meshes. *J. Comput. Phys.*, 160:481–499, 2000.

[63] F. Hermeline. Approximation of 2-D and 3-D diffusion operators with variable full tensor coefficients on arbitrary meshes. *Comput. Methods Appl. Mech. Engrg.*, 196:2497–2526, 2007.

[64] F. Hermeline. A finite volume method for approximating 3D diffusion operators on general meshes. *J. Comput. Phys.*, 228:5763–5786, 2009.

[65] F. Hermeline. Un point sur les méthodes DDFV, 2010. Advanced methods for the diffusion equation on general meshes; Université Pierre et Marie Curie, Paris France, July 2010; available at `http://www.ann.jussieu.fr/~despres/WEB/Talks/hermeline.pdf`.

[66] C. Hirsch. *Numerical computation of internal and external flows. Vol. 2. , Computational methods for inviscid and viscous flows*. Wiley series in numerical methods in engineering. J. Wiley, 1990.

[67] C. Hirsch. *Numerical computation of internal and external flows. Vol. 1. , The Fundamentals of Computational Fluid Dynamics*. Butterworth-Heinemann, second edition, 2007.

[68] J.O. Hirschfelder, C.F. Curtiss, and R.B. Bird. *Molecular theory of gases and liquids*. Wiley, 1954.

[69] R. Huart. *Simulation numérique d'écoulements magnétohydrodynamiques par des schémas distribuant le résidu*. PhD thesis, Université Bordeaux I, 2012.

[70] T. J.R. Hughes. Recent progress in the development and understanding of SUPG methods wi th special reference to the compressible Euler and Navier-Stokes equations. *Int. J. Numer. Meth. Fluids*, 7(11):1261–1275, 1987.

[71] T. J.R. Hughes, L. P. Franca, and G. M. Hulbert. A new finite element formulation for computational fluid dynamics: Viii. the galerkin/least-squares method for advective-diffusive equations. *Comput. Methods Appl. Mech. Engrg.*, 73(2):173 – 189, 1989.

[72] J. Hyman, J.E. Morel, M. Shashkov, and S. Steinberg. Mimetic finite difference methods for diffusion equations. *Computational Geosciences*, 6:333–352, 2002.

[73] J. Hyman, M. Shashkov, and S. Steinberg. The numerical solution of diffusion problems in strongly heterogeneous non-isotropic materials. *J. Comput. Phys.*, 132:130–148, 1997.

[74] P. Jacq, P.-H. Maire, and R. Abgrall. High-order cell-centered finite volume scheme for simulating three-dimensional anisotropic diffusion equations on unstructured grids. *Com. in Comp. Phys.*, 16(4):841–891, 2014.

[75] G.-S. Jiang and C.-W. Shu. Efficient implementation of weighted ENO schemes. *J. Comput. Phys.*, 126(1):202 – 228, 1996.

[76] S. D. Kim, B. J. Lee, H. J. Lee, and I.-S. Jeung. Robust HLLC Riemann solver with weighted average flux scheme for strong shock. *J. Comput. Phys.*, 228(20):7634 – 7642, 2009.

[77] L. Landau and E. Lifchitz. *Mécanique des Fluides*. Mir, 1989.

[78] P. S. Laplace. Sur la vitesse du son dans l'air et dans l'eau. *Ann. de Chim. et de Phys. iii*, 238, 1816.

[79] A. Larat. *Conception and analysis of very high order distribution schemes: Application to fluid mechanics*. PhD thesis, Université Bordeaux I, 2009.

[80] M. Latige. *Simulation numérique de l'ablation liquide*. PhD thesis, Université Bordeaux I, 2013.

[81] L. Lees. Hypersonic flow. *Fifth International Aeronautical Conference, Los Angeles*, pages 241–276, 1955.

[82] R. J. Leveque. *Finite Volume Methods for Hyperbolic Problems.* Cambridge University Press, 2002.

[83] D. W. Levy, K. G. Powell, and B. van Leer. Use of a rotated Riemann solver for the two-dimensional Euler equations. *J. Comput. Phys.*, 106(2):201 – 214, 1993.

[84] K. Lipnikov, J.E. Morel, and M. Shashkov. Mimetic finite difference methods for diffusion equations on non-othogonal non-conformal meshes. *J. Comput. Phys.*, 199:589–597, 2004.

[85] K. Lipnikov, M. Shashkov, and D. Svyatskiy. The mimetic finite difference discretization of diffusion problem on unstructured polyhedral meshes. *J. Comput. Phys.*, 211:473–491, 2006.

[86] K. Lipnikov, M. Shashkov, and I. Yotov. Local flux mimetic finite difference methods. *Numer. Math*, 112(1):115–152, 2009.

[87] X.-D. Liu, S. Osher, and T. Chan. Weighted essentially non-oscillatory schemes. *J. Comput. Phys.*, 115(1):200 – 212, 1994.

[88] R. Löhner, K. Morgan, and O.C. Zienkiewicz. An adaptive finite element procedure for compressible high speed flows. *Comput. Methods Appl. Mech. Engrg.*, 51(1-3):441–465, 1985.

[89] P.-H. Maire. *Contribution to the numerical modeling of Inertial Confinement Fusion.* Habilitation à diriger des recherches, Université Bordeaux I, 2011.

[90] P.-H. Maire and J. Breil. A high-order finite volume cell-centered scheme for anisotropic diffusion on two-dimensional unstructured grids. *J. Comput. Phys.*, 224(2):785–823, 2011.

[91] A.M. Meirnanov. *The Stefan Problem.* Walter de Gruyter Edition, 1992.

[92] K. Michalak and C. Ollivier-Gooch. Limiters for unstructured higher-order accurate solutions of the Euler equations. *46th AIAA Aerospace Sciences Meeting*, 2008.

[93] J.E. Morel, R. M. Roberts, and M. Shashkov. A local support-operators diffusion discretization scheme for quadrilateral r-z meshes. *J. Comput. Phys.*, 144:17–51, 1998.

[94] W.A. Mulder and B. van Leer. Experiments with implicit upwind methods for the Euler equations. *J. Comput. Phys.*, 59:232–246, 1985.

[95] K. Nakajima, H. Nakamura, and T. Tanahashi. Parallel iterative solvers with localized ilu preconditioning. In Bob Hertzberger and Peter Sloot, editors, *High-Performance Computing and Networking*, volume 1225 of *Lecture Notes in Computer Science*, pages 342–350. Springer Berlin Heidelberg, 1997.

[96] H. Nishikawa. I do like CFD. Free CFD Codes Web page. Available at `http://www.cfdbooks.com/cfdcodes.html`.

[97] H. Nishikawa and K. Kitamura. Very simple, carbuncle-free, boundary-layer-resolving, rotated-hybrid Riemann solvers. *J. Comput. Phys.*, 227(4):2560 – 2581, 2008.

[98] M. Pal and M.G. Edwards. Quasi monotonic continuous Darcy-flux approximation for general 3-D gids on any element type. In *Proceedings of the SPE International Reservoir Simulation Symposium*, number SPE 106486, Houston, USA, 2007.

[99] M. Pandolfi and D. D'Ambrosio. Numerical instabilities in upwind methods: Analysis and cures for the carbuncle phenomenon. *J. Comput. Phys.*, 166(2):271 – 301, 2001.

[100] F. Pellegrini. Scotch Web page, 2012. Available at `https://gforge.inria.fr/projects/scotch/`.

[101] C. Le Potier. Schéma volumes finis monotone pour des opérateurs de diffusion fortement anisotropes sur des maillages de triangles non structurés. *Comptes Rendus Mathématique*, 341:787–792, 2005.

[102] C.B. Allen P.R. Ess. Parallel computation of two-dimensional laminar inert and chemically reactive multi-species gas flows. *International Journal of Numerical Methods for Heat and Fluid Flow*, 15:228–256, 2005.

[103] J.J. Quirk. A contribution to the great Riemann solver debate. *International Journal for Numerical Methods in Fluids*, (18):555–574, 1994.

[104] Y.-X. Ren. A robust shock-capturing scheme based on rotated Riemann solvers. *Computers and Fluids*, 32(10):1379 – 1403, 2003.

[105] M. Ricchiuto. *Construction and Analysis of Compact Residual Discretizations for Conservation Laws on Unstructured Meshes*. PhD thesis, Von Karman Institute for Fluid Dynamics, 2005.

[106] M. Ricchiuto. *Contributions to the development of residual discretizations for hyperbolic conservation laws with application to shallow water flows*. Hdr thesis, Université Bordeaux I, 2011.

[107] P. L. Roe and J. Pike. Efficient construction and utilisation of approximate Riemann solutions. In *Proc. Of the Sixth Int'L. Symposium on Computing Methods in Applied Sciences and Engineering, VI*, pages 499–518. North-Holland Publishing Co., 1985.

[108] P.L. Roe. Approximate Riemann solvers, parameter vectors, and difference schemes. *J. Comput. Phys.*, 43(2):357 – 372, 1981.

[109] P.L. Roe. Fluctuations and signals – a framework for numerical evolution problems. *Numerical Methods for Fluid Dynamics, K.W.Morton, M.J.Baines (Eds.), Academic Press*, pages 219–257, 1982.

[110] J. Salençon. *Mécanique des milieux continus*, volume I, Concepts généraux. Editions de l'Ecole Polytechnique, 2005.

[111] D. De Santis. *Development of a high-order residual distribution method for Navier-Stokes and RANS equation*. PhD thesis, Université Bordeaux I, 2013.

[112] J. Serrin. Mathematical Principles of Classical Fluid Mechanics. In *Handbuch der Physik*, volume VIII, pages 125–263. Springer Verlag, 1959.

[113] M. Shashkov. *Conservative Finite-Difference Methods on General Grids*. CRC Press, 1996.

[114] M. Shashkov and S. Steinberg. Support-operator finite-difference algorithms for general elliptic problems. *J. Comput. Phys.*, 118:131–151, 1995.

[115] M. Shashkov and S. Steinberg. Solving diffusion equations with rough coefficients in rough grids. *J. Comput. Phys.*, 129:383–405, 1996.

[116] C.-W. Shu and B. Cockburn. The Runge-Kutta discontinuous Galerkin method for conservation laws V: Multidimensional systems. *J. Comput. Phys.*, 141(2):199–224, 1998.

[117] A.J.M. Spencer. *Continuum Mechanics*. Dover, 2004.

[118] P. Le Tallec. *Modélisation et calcul des milieux continus*. Editions de l'Ecole Polytechnique, 2009.

[119] J.-M. Thomas and D. Trujillo. Mixed finite volume methods. *International Journal for Numerical Methods in Engineering*, 46:1351–1366, 1999.

[120] E. F. Toro. *Riemann Solvers and Numerical Methods for Fluid Dynamics : A Practical Introduction*. Springer, third edition, 2009.

[121] E.F. Toro. A fast Riemann solver with constant covolume applied to the random choice method. *Int. J. Numer. Meth. Fluids*, 9:1145–1164, 1989.

[122] E.F. Toro. A Weighted Average Flux method for hyperbolic conservation laws. *Proc. R. Soc. Lond. A*, 423(1865):401–418, 1989.

[123] E.F. Toro. The weighted average flux method applied to the Euler equations. *Philos. Trans. R. Soc. Lond. A*, 341(1662):499–530, 1992.

[124] E.F. Toro, M. Spruce, and W. Speares. Restoration of the contact surface in the HLL-Riemann solver. *Shock Waves*, 4(1):25–34, 1994.

[125] E.F. Toro and V.A. Titarev. ADER schemes for scalar non-linear hyperbolic conservation laws with source terms in three-space dimensions. *J. Comput. Phys.*, 202(1):196 – 215, 2005.

[126] J.-M. Vaassen, D. Vigneron, and J.-A. Essers. An implicit high order finite volume scheme for the solution of 3d Navier- Stokes equations with new discretization of diffusive terms. *Journal of Computational and Applied Mathematics*, 215(2):595 – 601, 2008.

[127] H.A. van der Vorst. Bi-CGStab: A fast and smoothly converging variant of Bi-CG for the solution of nonsymmetric linear systems. *SIAM J. Sci. and Stat. Comput.*, 13(2):631–644, 1992.

[128] B. van Leer. Towards the ultimate conservative difference scheme. IV-a new approach to numerical convection. *Lecture Notes in Physics*, 23:276–299, 1977.

[129] A. Velghe. *Modélisation de l'interaction entre un écoulement turbulent et une paroi ablatable*. PhD thesis, INP Toulouse, 2007.

[130] V. Venkatakrishnan. Convergence to steady state solutions of the Euler equations on unstructured grids with limiters. *J. Comput. Phys.*, 118:120–130, 1995.

[131] V. Venkatakrishnan, S. Allmaras, D. Kamenetskii, and F. Johnson. Higher order schemes for the compressible Navier-Stokes equations. *American Institute of Aeronautics and Astronautics*, 2003. 16th AIAA Computational Fluid Dynamics Conference.

[132] D. Vigneron, J.-M. Vaassen, and J.-A. Essers. An implicit high order cell-centered finite volume scheme for the solution of three-dimensional Navier-Stokes equations on unstructured grids. *Computational Fluid and Solid Mechanics*, pages 211–230, 2005. Third MIT Conference on Computational Fluid ans Solid Mechanics.

[133] M. Vohralik. Equivalence between lowest-order mixed finite element and multi-point finite volume methods on simplicial meshes. *Mathematical Modelling and Numerical Analysis*, 40(2):367–391, 2006.

[134] F. De Vuyst. Stable and accurate hybrid finite volume methods based on pure convexity arguments for hyperbolic systems of conservation law. *J. Comput. Phys.*, 193(2):426 – 468, 2004.

[135] Z.J. Wang. High-order methods for the Euler and Navier-Stokes equations on unstructured grids. *Progress in Aerospace Sciences*, 43(1-3):1 – 41, 2007.

[136] C. Wervaecke. *Simulation d'écoulements turbulents compressibles par une méthode d'éléments finis stabilisés*. PhD thesis, Université Bordeaux I, 2010.

[137] L.T. Yang and R.P. Brent. The improved BiCGStab method for large and sparse unsymmetric linear systems on parallel distributed memory architectures. In *Algorithms and Architectures for Parallel Processing, 2002. Proceedings. Fifth International Conference on*, pages 324 –328, oct. 2002.

[138] Ya. B. Zel'dovich and Yu. P. Raizer. *Physics of Shock Waves and High-Temperature Hydrodynamic Phenomena*, volume I. Academic Press, 1967.