Development of "all-régime" AMR simulation methods for fluid dynamics, application in astrophysics and two-phase flows

Thomas Padioleau

Thèse de doctorat

NNT: 2020UPASP086

université PARIS-SACLAY

Development of “all-regime” AMR simulation methods for fluid dynamics, application in astrophysics and two-phase flows

Thèse de doctorat de l’université Paris-Saclay

École doctorale n° 127, Astronomie et Astrophysique d’Ile-de-France (AAIF)
Spécialité de doctorat: Astronomie et Astrophysique
Unité de recherche: Université Paris-Saclay, UVSQ, CNRS, CEA, Maison de la Simulation, 91191, Gif-sur-Yvette, France.
Référent: Faculté des sciences d’Orsay

Thèse présentée et soutenue à Gif-sur-Yvette, le 9 décembre 2020, par

Thomas PADIOLEAU

Composition du jury:

Caroline NORE Professeur, Laboratoire d’informatique pour la mécanique et les sciences de l’ingénieur, université Paris-Saclay	Présidente
Ann ALMGREN Directrice de recherche, Lawrence Berkeley National Lab	Rapportrice et Examinatrice
Philippe HELLUY Professeur, université de Strasbourg	Rapporteur et Examinateur
Isabelle BARAFFE Professeur, université d’Exeter	Examinatrice
Carlos PARÉS Professeur, université de Malaga	Examinateur
Edouard AUDIT Cadre scientifique des EPIC, CEA Saclay	Directeur de thèse
Samuel KOKH Cadre scientifique des EPIC, CEA Saclay	Co-encadrant de thèse
Pascal TREMBLIN Cadre scientifique des EPIC, CEA Saclay	Co-encadrant de thèse

Remerciements

Tout d'abord je tiens à remercier mes encadrants Edouard Audit, Pierre Kestener, Samuel Kokh et Pascal Tremblin. En seulement trois années vous m'avez permis d'apprendre énormément de choses sur des domaines très variés. Je remercie tout particulièrement Pascal Tremblin pour sa très grande disponibilité et son état d'esprit positif à toute épreuve. J'ai beaucoup appris grâce à toi et à travers les nombreuses discussions sur les schémas numériques, le calcul intensif et la physique. Je veillerai à appliquer du mieux possible le théorème d'Astérix. Je remercie Samuel Kokh pour son encadrement, notamment sur les modèles diphasiques et pour sa rigueur de relecture. Je remercie également Pierre Kestener pour avoir partagé ses connaissances sur les modèles de programmation parallèle et sur les bonnes pratiques du C++et d'avoir mis à disposition ses nombreuses maquettes "Euler2d".

J'aimerais ensuite remercier mon jury de thèse. Merci à Ann Almgren et Philippe Helluy d'avoir accepté d'être rapporteurs de mon manuscrit, en espérant qu'ils aient apprécié la lecture. Je remercie également Isabelle Baraffe, Caroline Nore et Carlos Parès d'avoir participé au jury en tant qu'examinateurs. Je les remercie pour l'intérêt qu'ils ont porté à mon travail ainsi qu'aux échanges lors de la soutenance.

Je remercie chaleureusement les permanents et non-permanents de mon laboratoire d'accueil, la Maison de la Simulation. Les nombreuses discussions autour des pauses café ou déjeuner ont notablement participé au bon déroulement de la thèse. Je remercie mes anciens collègues thésards Arnaud, Hélène, Ksander, Maxime, Vitaly et tout particulièrement mon collègue de bureau Ziqiang pour toutes les discussions, presque philosophiques, autour des modèles diphasiques. Enfin je souhaite une bonne continuation à Rémi pour cette aventure qu'est la thèse.

Je remercie également mon comité de suivi de thèse Julien Bigot et Patrick Hennebelle pour leurs conseils et pour s'être assurés du bon déroulement de ma thèse.

Je remercie mes amis Nantais de longue date Anthony, Fadel et Tito pour tous ces bons moments et échanges qui ont permis d'évacuer le stress. Merci à Alexandra pour son écoute et pour m'avoir aidé à passer ces longues périodes de

confinement.

Enfin je tiens à remercier toute ma famille, mes parents, mes frères ainsi que mes belles-sœurs pour m'avoir permis et encouragé à poursuivre mes études lorsque la motivation se faisait plus rare.

Contents

List of Figures

List of Tables

Introduction générale **version française**

0.1 Contexte général

La modélisation consiste à donner un cadre et un ensemble d'outils permettant de décrire une partie de la réalité. Ce cadre est formé à partir d'un ensemble d'hypothèses réfutables, au sens de Karl Popper, et se traduisent souvent sous forme d'équations dont la solution fournit une description approchée de phénomènes observés. L'expérimentation permet ensuite de tester ces hypothèses en comparant les résultats de manipulations ou d'observations aux solutions des équations du modèle. Avec un point de vue tourné vers les applications industrielles, la connaissance des solutions d'un modèle, alors supposé pertinent, permet de faciliter le dimensionnement de nombreux systèmes.

Parmi les différents modèles utilisés en astrophysique et dans le domaine des écoulements diphasiques se trouvent des modèles décrivant la dynamique d'un ou plusieurs fluides qui appartiennent à la catégorie des modèles de milieux continus. Ces modèles prennent la forme d'équations aux dérivées partielles et s'appuient souvent sur des principes communs sous-jacent comme l'expression de lois de conservation, par exemple la conservation de la masse, de la quantité de mouvement ou encore de l'énergie auxquelles peuvent s'ajouter d'autres équations. Hormis quelques cas simples (exploitant de nombreuses symétries du problème) les solutions de ces équations ne peuvent pas être obtenues analytiquement.

On a alors recours aux simulations numériques qui consistent à utiliser les capacités de calcul d'un ordinateur pour calculer une approximation de cette solution. Afin de pouvoir réaliser ces simulations numériques, une étape supplémentaire est nécessaire: la discrétisation. La discrétisation consiste à traduire le modèle mathématique en un algorithme composé d'un nombre fini d'opérations élémentaires et réalisables par un ordinateur par exemple: les additions, et les multiplications. L'algorithme obtenu est un nouveau modèle, discret également appelé schéma numérique, qui en un certain sens est une approximation du modèle d'origine qualifié, par opposition, de modèle continu. L'étude du lien entre le modèle continu et un schéma numérique est le domaine de l'analyse numérique au sein des mathématiques appliquées. Plusieurs schémas numériques pouvant approcher un même modèle continu, des critères d'évaluation des schémas numériques sont donc nécessaires. Ces critères peuvent prendre différentes formes par exemple: le temps de restitution de la solution numérique ou encore la précision de la solution.

Dans le cas de l'astrophysique, parmi les différents mécanismes de transfert de chaleur, l'étude de la convection naturelle est un phénomène important, au coeur de la dynamique interne des étoiles (Spruit et al. 1990; Pinsonneault 1997) et des exoplanètes gazeuses (Tremblin et al. 2016), comme illustré Figure 1. Pour des raisons de temps de restitution raisonnable, l'étude par simulation numérique de la convection est souvent réalisée à partir d'approximations supplémentaires connues sous le nom d'approximation de Boussinesq ou anélastique. Ces différentes approximations permettent de filtrer les ondes sonores en supposant un écoulement convectif à faible vitesse devant la vitesse du son (régime bas Mach) et sur de petites échelles de hauteur. Cependant

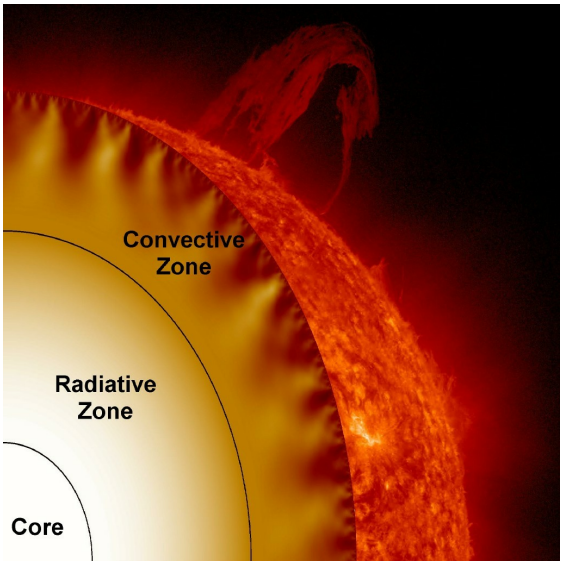


Figure 1: Représentation graphique des différents régimes de transfert de chaleur depuis le coeur d'une étoile vers sa surface. La chaleur émise depuis le coeur de l'étoile par les réactions nucléaires est dans un premier temps diffusé par radiation, sans mouvement du gaz. Le transfert de chaleur est ensuite effectué à partir de mouvements macroscopiques du gaz vers la surface de l'étoile. Source `https://solarscience.msfc.nasa.gov/interior.shtml`

l'étude de régimes intermédiaires où la compressibilité du fluide joue un rôle sort du cadre de ces modèles et ne peut donc pas être capturé numériquement. L'utilisation du modèle hydrodynamique compressible est alors nécessaire mais pose également des difficultés numériques lorsque l'écoulement est proche de l'équilibre hydrostatique.

En ce qui concerne le domaine des écoulements diphasiques, dans de nombreuses situations (souvent d'origines industrielles) le liquide se trouve être en régime bas Mach. Une approximation proche du modèle incompressible est alors utilisée, présentant des similarités avec l'approximation de Boussinesq. Cependant dans des régimes intermédiaires, notamment dans les circuits hydrauliques de réacteurs nucléaires, les variations de pression au sein de gouttes d'eau liquide peuvent être importantes et sortent du cadre de ce modèle. On est alors amené à utiliser des modèles tenant compte des phénomènes compressibles. A terme, ces pics de pression peuvent endommager la paroi du circuit, comme illustré sur la Figure 2. L'étude par simulation numérique des pics de pression au niveau de la paroi fournit une approche complémentaire aux expériences.

La réalisation de ces simulations peut demander de nombreuses heures de calcul (jusqu'à plusieurs dizaines de millions d'heures de calcul). Afin de pouvoir réaliser ces simulations en des temps raisonnables l'utilisation de supercalculateurs, des ordinateurs dédiés à ce type de calcul, est nécessaire. A partir de 2021, les nouveaux supercalculateurs atteindront la puissance de calcul permettant de réaliser jusqu'à 10^{18} opérations à la seconde, ce que l'on appelle couramment l'ère du calcul exaflopique ("exascale" en anglais). Afin d'obtenir cette puissance de calcul, les constructeurs s'appuient sur des architectures matérielles variées et ce qui par conséquent cela requiert l'utilisation de modèles de programmation adaptés à chacune d'entre elles pour un même schéma numérique. Cette duplication de code peut, à terme, poser des problèmes de maintenance. C'est pourquoi différentes initiatives ont donné lieu à des bibliothèques telles que Kokkos, à travers une abstraction des architectures matérielles, permettant de retrouver une

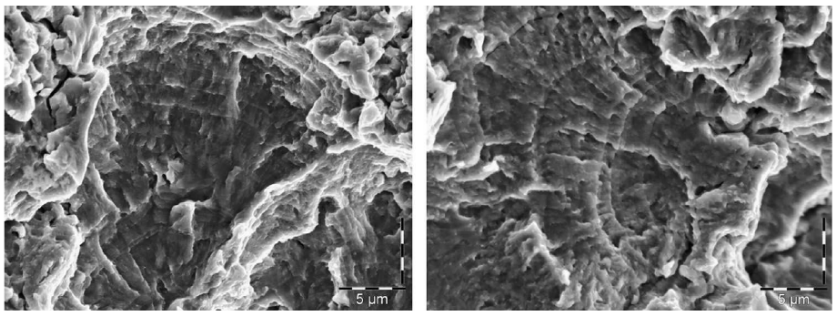


Figure 2: Formation de microstructures après 5 (gauche) et 10 (droite) millions d'impacts de gouttes d'eau à haute vitesse (Luiset et al. 2013).

portabilité de performance tout en écrivant une seule version de l'algorithme.

0.2 Objectifs de la thèse

Cette thèse s'inscrit à l'interface entre les aspects de modélisation astrophysique et diphasique, de développements de schémas numériques ainsi que de leur implémentation parallèle. Elle s'est déroulée au sein du laboratoire Maison de la Simulation, laboratoire multidisciplinaire dont les activités concernent des aspects de modélisation et de calcul haute performance.

Dans cette thèse nous nous intéressons au développement de schémas numériques dits "tout régime", à partir de modèles hydrodynamiques, ainsi qu'à leur implémentation dans un objectif de calcul haute performance. Ces schémas s'appuient sur un découplage des phénomènes acoustiques, a priori rapides, et des phénomènes de transport, a priori lents. Ce découplage permet entre autre de développer des schémas numériques avec diverses propriétés en facilitant leur analyse.

En ce qui concerne l'application astrophysique, nous nous intéressons à des instabilités de convection. Ces instabilités ont lieu autour de l'équilibre hydrostatique et généralement à bas nombre de Mach. Nous cherchons alors à obtenir un schéma numérique capable de simuler des écoulements convectifs à tout nombre de Mach. Nous cherchons donc à maintenir exactement, ou à précision machine près, l'équivalent discret des équilibres hydrostatiques. Les caractères bas nombre de Mach et proche équilibre hydrostatique des écoulements convectifs nécessitent un traitement particulier de la discrétisation des gradients de pression liés aux phénomènes acoustiques. Le caractère haut Mach est garanti par l'utilisation d'un modèle compressible et d'un schéma conservatif.

L'application diphasique de la thèse concerne l'impact d'une goutte d'eau sur une paroi au sein du circuit hydraulique d'un réacteur nucléaire. Les liquides et les gaz ont des propriétés thermodynamiques très différentes. La localisation précise de ces deux fluides au sein du domaine de simulation est donc importante afin de pouvoir calculer correctement les quantités thermodynamiques. De part la raideur de l'équation d'état du liquide, de fortes tensions peuvent conduire le liquide à caviter. Nous nous intéressons alors au Modèle Homogène à l'Equilibre (HEM pour Homogeneous Equilibrium Model) qui suppose, à chaque instant, l'équilibre thermodynamique entre les phases.

En lien avec ces applications nous nous intéressons également à la problématique de performance des schémas numériques et de leur implémentation. L'accélération du calcul peut être envisagée à la fois de manière algorithmique et de manière informatique. Du point de vue algo-

rithmique nous nous intéressons à un schéma d'ordre élevé afin de tirer parti d'une représentation plus compacte et donc a priori moins coûteuse de la solution numérique. Du point de vue informatique, nous abordons l'aspect portabilité de performance à travers l'utilisation des bibliothèques Kokkos et Trilinos. Ces deux bibliothèques fournissent des algorithmes élémentaires, parallèles et portables, qu'il s'agit d'exploiter dans l'implémentation de nos schémas.

0.3 Description des travaux

0.3.1 Chapitre 1

Dans ce chapitre nous nous inscrivons dans la continuité des travaux de Chalons et al. 2016 et de Chalons et al. 2017 pour le développement d'un schéma "tout-régime" et équilibre des équations d'Euler avec terme source de gravité statique. Nous cherchons à capturer les états d'équilibre hydrostatique de ces équations. La séparation des phénomènes acoustiques et de transport en deux sous-systèmes d'équations nous a conduit à placer le terme source de gravité dans le système acoustique. Le système acoustique est discrétisé à partir d'un schéma de type Godunov (Godunov 1959). Le solveur de Riemann approché associé à ce schéma est obtenu à partir d'une méthode de relaxation (Suliciu 1998; Chalons and Coulombel 2008; Chalons and Coquel 2014) des équations permettant une linéarisation des termes associés à l'équation d'état. Le système de transport est quant à lui discrétisé à partir d'un schéma décentré amont tout en garantissant une mise à jour globale conservative. L'observation qu'en gravité statique, par opposition à l'auto-gravité, l'énergie tenant compte du potentiel gravitationnel est également une quantité conservée, nous proposons également un schéma numérique permettant la conservation de cette quantité. De manière similaire au travail de Chalons et al. 2016 et faisant suite des analyses bas Mach des schémas colocalisés (Guillard and Viozat 1999; Dellacherie 2010), une correction de flux est apportée au gradient de pression afin d'améliorer la précision du schéma dans un régime bas Mach.

Ces schémas numériques ont ensuite été utilisés pour des applications d'instabilité de convection dans différents régimes d'écoulement. Il a été montré l'importance de la correction de flux dans des régimes bas Mach. Ils ont notamment été utilisés afin de proposer une théorie visant à unifier différents types de phénomènes convectifs.

0.3.2 Chapitre 2

Ce chapitre porte sur l'application diphasique de la thèse. Nous nous intéressons à la simulation d'impacts d'une goutte d'eau contre une paroi. Lorsque la goutte d'eau impacte la paroi, un important saut de pression se développe connu sous le nom de coup de bélier (Ghidaoui et al. 2005) égal au produit de l'impédance acoustique de l'eau liquide par le saut de vitesse. Afin de capturer ces sauts de pression nous utilisons un modèle compressible: un Modèle Homogène à l'Equilibre (Helluy and Seguin 2006) qui suppose qu'à chaque instant l'équilibre thermodynamique est atteint. Ce modèle est approché par une relaxation instantanée et à partir d'un modèle dit "modèle à 5 équations" (Allaire et al. 2002). Ce dernier est alors discrétisé en séparant également les phénomènes acoustiques et les phénomènes de transport. Le système acoustique est discrétisé à partir d'une méthode de relaxation (Suliciu 1998; Chalons and Coulombel 2008; Chalons and Coquel 2014) adaptée au vide (Bouchut 2004). Le système de transport utilise un schéma anti-diffusif (Kokh and Lagoutière 2010; Lagoutière 2000) afin de préserver du mieux possible les interfaces présentes dans la situation initiale. Des simulations d'impact de goutte ont été menées sur différents types de paroi avec une étude de convergence. Une simulation Grand

Challenge a été menée sur la moitié du supercalculateur Joliot-Curie au Très Grand Centre de Calcul (TGCC), Saclay.

0.3.3 Chapitre 3

Dans ce chapitre nous explorons la possibilité de diminuer le temps de restitution des simulations numériques en adaptant la représentation numérique et le schéma numérique aux propriétés locales de la solution (Schaal et al. 2015; Sonntag and Munz 2017). Dans une région où la solution est discontinue nous souhaitons utiliser les méthodes traditionnelles Volumes Finis par exemple à partir de schémas de type Godunov (Godunov 1959). A l'inverse, dans une région où la solution est régulière nous souhaitons compresser sa représentation numérique à l'aide de polynômes et de schémas spectraux. Nous avons donc développer une méthode hybride à partir d'un schéma MUSCL-Hancock (Toro 2009) et d'un schéma Volume Spectral (Wang 2002; Wang and Liu 2002; Wang and Liu 2004; Wang et al. 2004; Liu et al. 2006). Ce dernier utilise une formulation intégrale des équations afin de fournir une évolution temporelle des coefficients du polynôme. Cette mise à jour est effectuée par l'intermédiaire des valeurs moyennes dans des volumes de contrôle. Cela permet une écriture très proche des schémas standards, non spectraux, et par conséquent facilite le passage d'une représentation à l'autre. Enfin des tests d'ordre de convergence et de performance ont été menés dans le cas des équations d'Euler et une extension au terme source de gravité est proposée.

0.3.4 Chapitre 4

Dans ce chapitre nous abordons différents outils numériques utilisés au cours de la thèse permettant de réaliser des simulations haute performance. Dans un premier temps nous présentons la problématique de portabilité de performance qui se pose avec l'arrivée des calculateurs exaflopiques. Nous présentons ensuite brièvement la bibliothèque Kokkos (Carter Edwards et al. 2014) ainsi que les stratégies de parallélisation des codes s'appuyant sur ARK implémentant les schémas numériques des Chapitres 1 et 2. Nous proposons une adaptation du schéma convectif "tout-régime" aux maillages adaptatifs dans un code 2D. Enfin nous présentons les efforts de portage GPU d'un schéma implicite résolvant une équation de diffusion modélisant le transfert radiatif (Commerçon et al. 2014) dans le code RAMSES (Teyssier 2002), code utilisé parmi la communauté astrophysique.

0.4 Publications et communications

Les travaux effectués durant cette thèse ont fait l'objet de publications et ont été présentés à plusieurs conférences internationales, à savoir

- mini-symposium, CANUM, Cap d'Agde, France, Juin 2018, *A well-balanced scheme for compressible flows with gravity at all Mach number*,
- Thomas Padioleau et al. 2019. "A High-performance and Portable All-Mach Regime Flow Solver Code with Well-balanced Gravity. Application to Compressible Convection." *The Astrophysical Journal* 875, no. 2 (April): 128
- mini-symposium, ICIAM, Valencia, Espagne, Juillet 2019, *Compressible two-phase flow simulations of liquid droplet impacts*,
- webinar invité, EUROfusion, Juillet 2020, *Introduction to Kokkos*.

De plus, les travaux du Chapitre 2 et du Chapitre 3 sont l'objet d'articles en cours de préparation.

General introduction **english version**

Modelling consists in giving a framework and a set of tools allowing to describe part of the reality. This framework is made from a set of falsifiable assumptions, in the sense of Karl Popper, and are often represented as equations providing an approximation of observed phenomena. Then experimentation allows to test these assumptions by comparing the results of manipulations or observations to the solutions of the equations of the model. From the point of view of industrial applications, the knowledge of the solutions of a model, then assumed to be relevant, makes it possible to dimension many systems.

Among the different models used in astrophysics and in the field of two-phase flows, there are models describing the dynamics of one or more fluids that belong to the category of continuous media models. These models take the form of partial differential equations and are often based on common underlying principles such as the expression of conservation laws, for instance conservation of mass, conservation of momentum or energy to which can be added other equations. Apart from a few simple cases (that use multiple symmetries of the considered problem) the solutions of these equations cannot be obtained analytically.

Therefore we resort to use numerical simulations that consist in using the computation capabilities of a computer to calculate an approximation of this solution. In order to run these numerical simulations, a additional step is required: the discretization. This step consists in translating the equations of the model into an algorithm made of a finite number of elementary operations that a computer can realize such as additions and multiplications. The obtained algorithm is a new, discrete model also called numerical scheme, that in a sense to be defined is an approximation the first model. The study of the link between the continuous model and a numerical scheme is the field of numerical analysis within applied mathematics. Multiple numerical schemes can approximate the same continuous model, therefore criteria of evaluation are necessary. These criteria can take different forms for example: the time to numerical solution or the precision of the solution.

In the case of astrophysics, among the different heat transfer mecanisms, natural convection is an important phenomenon, at the basis of the dynamics of stars (Spruit et al. 1990; Pinsonneault 1997) and gaseous giant exoplanets (Tremblin et al. 2016), see Figure 3. In order to have a reasonable time to solution, the numerical study of convection is often carried out from additional approximations known as the Boussinesq approximation or the anelastic approximation. These different approximations filter the sound waves assuming that the convective flow is slow compared to the speed of sound (low Mach regime) and on small scale heights. However the study of intermediate regimes where the compressibility of the fluid plays a role cannot be performed using this type of model and thus cannot be captured numerically. The use of the compressible hydrodynamics model is then required although numerical difficulties arise near the hydrostatic balance.

Regarding the field of two-phase flows, in many situations (often of industrial origin) the liquid is found to be in a low Mach regime. An approximation close to the incompressible model

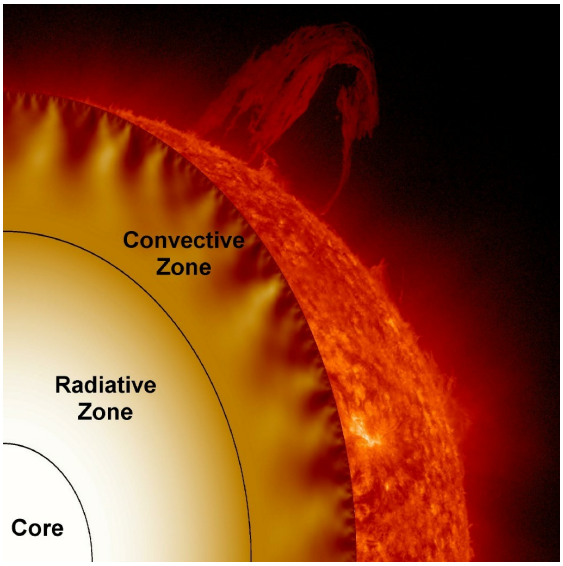


Figure 3: Graphical representation of different heat transfer mecanisms from the core of a star to its surface. At first, the heat generated by nuclear reactions from the core is diffused by radiation. Then the heat transfer is achieved by the advection of the gas up to the surface of the star. Source `https://solarscience.msfc.nasa.gov/interior.shtml`

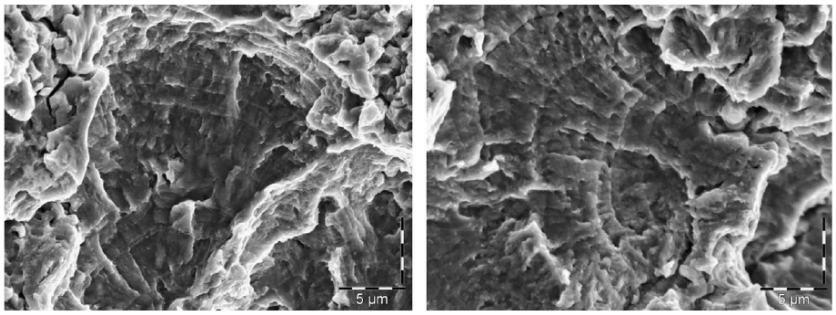


Figure 4: Formation of microstructures after 5 (left) and 10 (right) millions of liquid droplets impacts at high velocity (Luiset et al. 2013).

is then used, showing similarities with the Boussinesq approximation. However in intermediate regimes, particularly in the hydraulic circuits of nuclear reactors, pressure variations within water liquid droplets can be significant and fall out of the scope of this model. We are then brought to use models taking into account compressible phenomena. Ultimately, these pressure peaks can damage the circuit wall, see Figure 4. The study by numerical simulations of pressure peaks at the wall provides a complementary approach to real experiments.

Performing these simulations can take many hours of calculation (up to tens of millions of hours of calculation). In order to be able to carry out these simulations in a reasonable amount of time the use of supercomputers, computers dedicated to this type of calculation, is necessary. Starting from 2021, the new supercomputers will reach a computing power allowing to perform up to 10^{18} operations per second, what is commonly called the exascale computing. In order to obtain this computing power, manufacturers rely on various hardware architectures and thus it requires the use of programming models adapted to each of them for the same numerical scheme. This duplication of code may ultimately become problematic in terms of maintenance. This is why different initiatives have led to the development of libraries such as Kokkos, through an abstraction of material architectures, allowing to achieve the performance portability while writing a single version of the algorithm.

0.5 Aim of the thesis

This thesis is at the interface between the aspects of modeling astrophysics and two-phase flow, development of numerical schemes as well as their parallel implementation. It took place at the Maison de la Simulation laboratory, a multidisciplinary laboratory whose activities concern aspects of modeling and high performance computing.

In this thesis we are interested in the development of numerical schemes so-called “all regime”, from hydrodynamic models, as well as their implementation for the purpose of high computing performance. These numerical schemes are based on a decoupling of the acoustic phenomena, a priori fast, and transport phenomena, a priori slow. This decoupling allows, among other things, to develop numerical schemes with various properties facilitating their analysis.

Regarding the astrophysics application, we are interested in convection instabilities. These instabilities take place near the hydrostatic equilibrium and generally at low Mach number. We then seek to obtain a numerical scheme able of simulating convective flows at any Mach number. We therefore seek to maintain exactly, or up to the machine precision, the discrete equivalent of hydrostatic equilibrium. Both the low Mach and near hydrostatic equilibrium regimes require a particular treatment of the discretization of pressure gradients associated with acoustic phenomena. The high Mach regime is guaranteed by the use of a compressible model and of a conservative scheme.

The two-phase flow application of the thesis concerns the impact of a water droplet on a wall within the hydraulic circuit of a nuclear reactor. Liquids and gases have very different thermodynamic properties. The precise location of these two fluids within the simulation domain is therefore important in order to be able to calculate correctly thermodynamic quantities. Due to the stiffness of the liquid equation of state, strong tensions can lead to cavitation within the liquid phase. We are then interested in the Homogeneous Equilibrium Model (HEM) which assumes, at each instant, the thermodynamic equilibrium between the phases.

Along with these applications we are also interested in the performance issue of numerical schemes and their implementation. The acceleration of computation can be considered in two manners. From the algorithmic point of view we are interested in a high-order scheme in order to take advantage of a more compact representation and therefore a priori less expensive numerical

solution. From a computer science point of view, we address the aspect of performance portability by using the Kokkos and Trilinos libraries. These two libraries provide elementary algorithms, parallel and portable, that shall be used in the implementation of our numerical schemes.

0.6 Description of the work

0.6.1 Chapter 1

In this chapter we are in the continuity of the works initiated by Chalons et al. 2016 and Chalons et al. 2017 for the development of an "all-regime" and well-balanced scheme of Euler equations with the static gravity source term. We seek to capture the hydrostatic equilibrium states of these equations. The splitting between acoustic and transport phenomena in two subsystems of equations led us to put the gravity source term in the acoustic system. The acoustic system is discretized from a Godunov-type scheme (Godunov 1959). The approximate Riemann solver associated with this numerical scheme is obtained from a relaxation method (Suliciu 1998; Chalons and Coulombel 2008; Chalons and Coquel 2014) of the equations allowing the linearization of terms associated with the equation of state. The transport system is discretized using an upwind numerical scheme while ensuring an overall conservative update. Noticing that in the static gravity case, by opposition to the self-gravity, an energy taking into account the gravitational potential is satisfying a conservation law, we also propose a numerical scheme allowing its conservation. Similar to the work of Chalons et al. 2016 and following low Mach analyses of co-located schemes (Guillard and Viozat 1999; Dellacherie 2010), a flux correction is made to the pressure gradient in order to improve the precision of the numerical scheme in a low Mach regime.

These numerical schemes were then used for convection instability applications in different regimes of the flow. It has been shown the importance of the flux correction in low Mach regimes. They were also used to propose a theory aiming at unifying different types of convective phenomena.

0.6.2 Chapter 2

This chapter deals with the two-phase application of the thesis. We are interested in the simulation of a liquid water drop on a wall. When the water droplet hits the wall, a large jump pressure develops known as the water hammer (Ghidaoui et al. 2005) that is equal to the product of the acoustic impedance of the liquid water by the velocity jump. In order to capture these pressure jumps we use a compressible model: a Homogeneous Equilibrium Model (Helluy and Seguin 2006) that assumes at each instant an thermodynamic equilibrium. This model is approached by an instantaneous relaxation from a five-equation model (Allaire et al. 2002). The latter is then also discretized by separating the acoustic phenomena and transport phenomena. The acoustic system is discretized from a relaxation method (Suliciu 1998; Chalons and Coulombel 2008; Chalons and Coquel 2014) adapted to the vacuum (Bouchut 2004). The transport system uses an anti-diffusive scheme (Kokh and Lagoutière 2010; Lagoutière 2000) in order to preserve as much as possible the interfaces present in the initial situation. Simulations of droplet impact were carried out on different types of wall along with a convergence study. A Grand Challenge simulation was carried out on half of the Joliot-Curie supercomputer at the Très Grand Centre de Calcul (TGCC), Saclay.

0.6.3 Chapter 3

In this chapter we explore the possibility of reducing the time to solution of numerical simulations by adapting the numerical representation and the numerical scheme to local properties of the solution (Schaal et al. 2015; Sonntag and Munz 2017). In a region where the solution is discontinuous we want to use standard Finite Volume methods such as Godunov-type methods (Godunov 1959). Conversely, in a region where the solution is regular we want to compress its numerical representation using polynomials and spectral schemes. We have therefore developed a hybrid method using a MUSCL-Hancock scheme (Toro 2009) and a Spectral Volume Method (Wang 2002; Wang and Liu 2002; Wang and Liu 2004; Wang et al. 2004; Liu et al. 2006). The latter uses an integral formulation of the equations in order to provide a time evolution of the polynomial coefficients. This update is carried out through the mean value associated with Control Volumes. This allows a formulation close to standard, non-spectral diagrams, and therefore facilitates the change of one representation to the other. Finally tests of order of convergence and of performance were carried out in the case of the Euler equations and an extension to the source term of gravity is proposed.

0.6.4 Chapter 4

In this chapter we discuss different numerical tools used during the thesis allowing to carry out high performance simulations. First, we present the problem of performance portability that arises with the arrival of exaflop supercomputers. We then briefly present the Kokkos library (Carter Edwards et al. 2014) as well as the parallelization strategies of codes based on ARK that implement the numerical schemes presented in chapters 1 and 2. We propose an extension of the convective "all-regime" scheme to the Adaptive Mesh Refinement (AMR) technique in a 2D code. Finally we present the GPU porting efforts of a implicit numerical scheme solving a diffusion equation that models the radiative transfer (Commerçon et al. 2014) in the code RAMSES (Teyssier 2002), code used among the astrophysical community.

0.7 Publications et communications

The work carried out during the thesis made the object of publications and have been presented in international conferences, namely

- mini-symposium, CANUM, Cap d'Agde, France, June 2018, *A well-balanced scheme for compressible flows with gravity at all Mach number*,
- Thomas Padioleau et al. 2019. "A High-performance and Portable All-Mach Regime Flow Solver Code with Well-balanced Gravity. Application to Compressible Convection." *The Astrophysical Journal* 875, no. 2 (April): 128
- mini-symposium, ICIAM, Valencia, Spain, July 2019, *Compressible two-phase flow simulations of liquid droplet impacts*,
- invited webinar, EUROfusion, July 2020, *Introduction to Kokkos*.

In addition, the work of Chapter 2 and of Chapter 3 are the subject of articles in preparation.

Chapter 1

An all-regime hydrodynamic solver for stratified flows

This chapter is the adaptation of an article published in APJ, see Padioleau et al. 2019. The Appendix 1.C has been added to this article. In this appendix we derive the Boussinesq system which is used to study convection instabilities with general heat and composition source terms.

Introduction

The study of convection is an active topic of research in the astrophysics community because of its major role in different mechanisms such as heat transport in solar and stellar interiors (Spruit et al. 1990), mixing of elements (Pinsonneault 1997) and dynamo (Charbonneau 2014). As these mechanisms play a role in the estimation of the lifetime of these objects it is of great importance for stellar evolution theory.

Different approximations have been developed to ease the study of convection. The Boussinesq and the anelastic approximations simplify the Navier-Stokes system by getting rid of acoustic waves and keeping buoyancy effects. In practice these approximations are derived by looking at the equations satisfied by small perturbations near a reference state (Spiegel and Veronis 1960). The Boussinesq approximation is quite restrictive as it is valid for a small layer of the reference state, such that the flow can be considered incompressible. On the other hand the anelastic approach allows to have a larger scale height by keeping the density stratification of the reference state (Gilman and Glatzmaier 1981). Another way to understand these approximations is to consider the flow regime in terms of the Mach number Ma. As it is shown in Mentrelli 2018, these approximations can be recovered by considering low-Mach asymptotic limits of the Navier-Stokes system. The Froude number, defined as the non-dimensional ratio of kinetic energy to gravitational energy, characterizes the influence of gravity in the flow. By taking into account different Froude regimes, they recover the incompressible, the Boussinesq and the anelastic models. From a numerical point of view the removal of the acoustics waves in these models is quite attractive because it allows to have larger time steps. The anelastic model has been successfully implemented in different codes like Rayleigh (Featherstone and Hindman 2016) or Magic (Gastine and Wicht 2012) and it is widely used in the community (see Glatzmaier 2017). We can also mention the MAESTRO code (see Nonaka et al. 2010) which uses an extended version of the anelastic model. The velocity constraint takes into account the time variation of pressure. However these approaches present some drawbacks. The addition of new physics and source terms

to the model is difficult, one has to derive another asymptotic model to take the new physics into account in the anelastic regime (see Mentrelli 2018). Furthermore one has to be careful that the simulation stays in the regime of validity of the model (especially in the Boussinesq regime). Finally a numerical difficulty is the parallelization of those codes. They usually use pseudo-spectral methods for which it is more difficult to achieve a good scalability (e.g. need to use pencil-type domain decomposition Featherstone and Hindman 2016).

We chose to take a more flexible approach by solving the full compressible Navier-Stokes system, as in the MUSIC code (Viallet et al. 2011; Goffrey et al. 2017) but with a collocated finite volume solver instead of using a staggered grid. Different discretization techniques of the Euler system are used in the astrophysics community. We can classify them in various ways. One way is to separate SPH techniques from grid-based techniques. Furthermore grid-based approaches can be divided in different families, finite difference, finite element and finite volume. The finite volume method is of particular interest because of its natural property of being conservative and to capture shocks and discontinuities. Designing a finite volume scheme essentially resides in the definition of a numerical flux, numerical counterpart of the physical flux. A widely used family of fluxes is the Godunov (see Godunov 1959) flux which is the flux of the — usually approximate — Riemann problem between two neighbour cells.

However we have to face multiple numerical difficulties with this approach. Compressible solvers and mainly Godunov-type solvers are known to have an excessive amount of numerical diffusion in the low-Mach regime which make them unusable in this regime (see Guillard and Viozat 1999; Dellacherie 2010; Miczek et al. 2015; Chalons et al. 2016; Barsukow et al. 2017). In this regime, in which flows are smoother, considering Riemann problems at interfaces is not adapted. Indeed in the work of Miczek et al. 2015 they show that part of the kinetic energy is dissipated into internal energy whereas it should be conserved. To tackle this issue they propose a preconditionned Roe scheme to remove the numerical diffusion. Secondly, hydrodynamics and gravity are usually discretized independently from each other. In the case of highly stratified medium, the numerical scheme does not maintain the hydrostatic equilibrium and produces spurious flows that pollutes the simulation. Different approaches have been investigated to solve this issue both for the Euler and the shallow water equations. In Leroux and Cargo 1994, they rewrite the Euler system as a fully conservative system by defining an hydrostatic pressure satisfying a conservation law. In Chandrashekar and Klingenberg 2015 they use a variable reconstruction by taking advantage of the equilibrium profile. In Chalons et al. 2010; Vides et al. 2014; Chalons et al. 2017 they incorporate the source term in the Riemann problem itself allowing to compensate pressure gradients at the interface. As in Leroux and Cargo 1994, authors from Chertock et al. 2018 also propose to discretize the Euler system with gravity as a fully conservative system but using global fluxes and a reconstruction on equilibrium variables. Finally the last numerical difficulty is the time step in the low-Mach regime. Because of the stability condition involving the fast acoustic waves, the time step becomes very small compared to the material transport timescale. It can either be resolved using a full implicit approach as in the MUSIC code Viallet et al. 2011; Goffrey et al. 2017, or by using an implicit-explicit (IMEX) approach in which only the system with fast acoustic waves is solved implicitly (Chalons et al. 2016; Chalons et al. 2017).

Following the original work of Chalons et al. 2016 and Chalons et al. 2017 we use an acoustic-transport splitting. In Chalons et al. 2016 they derive a finite volume scheme of the Euler system on unstructured mesh. This scheme uses an acoustic splitting to separate acoustic waves from material ones. In the low Mach regime, this translates to a splitting between fast waves and slow waves. In the low Mach regime, the fast waves can be treated with an implicit solver to get rid of the restrictive stability condition. Then in the work of Chalons et al. 2017, the scheme has been adapted to shallow water equations with a source term which is the topography. This

source term is added in the equivalent acoustic subsystem to obtain a well-balanced scheme. In this paper we adapt their approach for the Euler system by taking care of the discretization of the energy equation.

The paper is organized as follows. In Section 1.1 we briefly recall the compressible model we use to study convection, i.e. the Navier-Stokes equations with gravity. In Section 1.2 we present the derivation of the well-balanced and all-regime numerical scheme using a splitting approach between an acoustic step and a transport step both solved explicitly in this work. In Section 1.3 we present some implementation features about the "ARK" [1] code in particular the Kokkos library used for the shared memory parallelization. We also give some performance results. Finally in Section 1.4 we present different numerical test cases illustrating the importance of the low-Mach correction and the well-balanced discretization of gravity.

1.1 Navier-Stokes equations

We want to solve Navier-Stokes equations expressing conservation of mass, balance of momentum and balance of energy, respectively written as follows

$$\begin{aligned} &\partial_t \rho && + \operatorname{div}(\rho \boldsymbol{u}) && = 0, \\ &\partial_t (\rho \boldsymbol{u}) && + \operatorname{div}(\rho \boldsymbol{u} \otimes \boldsymbol{u} + p\mathbb{I} - \boldsymbol{\tau}_{\text{visc}}) && = \rho \boldsymbol{g}, \\ &\partial_t (\rho E) && + \operatorname{div}((\rho E + p)\boldsymbol{u} - \boldsymbol{\tau}_{\text{visc}}\boldsymbol{u} + \boldsymbol{q}_{\text{heat}}) && = \rho \boldsymbol{g} \cdot \boldsymbol{u}, \end{aligned} \tag{1.1}$$

where ρ is the density, $\boldsymbol{u}$ the material velocity, p the pressure, $\boldsymbol{g}$ the external gravitational field, $\rho E = \rho e + \frac{1}{2}\rho \boldsymbol{u}^2$ the density of total energy with e the specific internal energy, $\boldsymbol{q}_{\text{heat}}$ the heat flux and $\boldsymbol{\tau}_{\text{visc}}$ the viscous tensor satisfying

$$\boldsymbol{\tau}_{\text{visc}} = \mu \left(\boldsymbol{\nabla u} + \boldsymbol{\nabla u}^T \right) + \eta \left(\operatorname{div} \boldsymbol{u} \right) \mathbb{I}, \tag{1.2}$$

where μ is the dynamic viscosity and η the bulk viscosity. We use $\cdot$ as a scalar product and thus div represents the divergence operator. In order to close Navier-Stokes system (1.1) we add constitutive equations namely a pressure law p^{EOS} (1.3a), the Fourier's law (1.3b) and the Stokes hypothesis (1.3c)

$$p = p^{\text{EOS}}(\rho, e), \tag{1.3a}$$

$$\boldsymbol{q}_{\text{heat}} = -\kappa \boldsymbol{\nabla} T, \tag{1.3b}$$

$$\eta = -\frac{2}{3}\mu \tag{1.3c}$$

We recall that the gravitational field is derived from a gravitational potential Φ for which $\boldsymbol{g} = -\boldsymbol{\nabla}\Phi$. Dealing with a constant in time external gravity field, $\partial_t \Phi = 0$ and using the conservation of mass we get (1.4)

$$\partial_t (\rho \Phi) + \operatorname{div}(\rho \Phi \boldsymbol{u}) = \rho \boldsymbol{u} \cdot \boldsymbol{\nabla}\Phi. \tag{1.4}$$

Let us emphasize that in this equation, the gravitational energy $\rho(\boldsymbol{x}, t)\Phi(\boldsymbol{x})$ is time dependent only through the density $\rho(\boldsymbol{x}, t)$. Hence the energy equation (1.4) can be rewritten in the following conservative form

$$\partial_t (\rho \mathcal{E}) + \operatorname{div}(\rho \mathcal{E} \boldsymbol{u} - \boldsymbol{\sigma}_{\text{stress}} \boldsymbol{u} + \boldsymbol{q}_{\text{heat}}) = 0 \tag{1.5}$$

where we define $\rho \mathcal{E} = \rho e + \frac{1}{2}\rho \boldsymbol{u}^2 + \rho \Phi$. Equation (1.5) expresses the local conversion between three different energy reservoirs, as depicted in figure 1.1: internal, kinetic and gravitational. There can be a direct transfer between gravitational energy and kinetic energy through the

1. `https://gitlab.erc-atmo.eu/erc-atmo/ark`, version v1.0.0

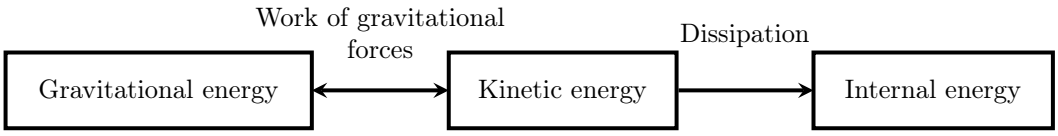


Figure 1.1: Diagram representing energy transfers between energy reservoirs.

work of gravitational forces, from kinetic energy to internal energy because of the second law of thermodynamics but no direct transfer between gravitational energy and internal energy, see also Section 5 of Springel 2010 and Section 2.2 of Marcello and Tohline 2012 for a discussion on energy conservation for both external and self-gravity cases.

Because of this conservation of energy including gravitational energy we will use the formulation (1.5) of the energy equation and we will use the gravitational potential instead of the usual gravitational field $\boldsymbol{g}$. To our knowledge this approach is quite rare, see Graham 1975 or Chertock et al. 2018 where they use global fluxes to have a well-balanced and conservative scheme.

An important steady state solution of this system for stratified objects is the hydrostatic balance. The flow is static and the gravitational force is balanced by the pressure forces, i.e. following equation (1.6)

$$\boldsymbol{\nabla} p = -\rho \boldsymbol{\nabla} \Phi, \qquad \boldsymbol{u} = \boldsymbol{0}, \tag{1.6}$$

As we mentioned in the introduction, convective flows can be considered as a perturbation flow of the hydrostatic equilibrium. Thus this steady state is particularly important in order to study convection problems in stratified flows.

1.2 Numerical scheme

1.2.1 Euler system — Hyperbolic system

Before going into the derivation of the scheme we introduce the notations. We define by Δx (resp. Δy and Δz) the step along the x-direction (resp. the y and z-direction). We note by Δt the time interval between current time t^n and t^{n+1}. We use the notation q_i^n (resp. $q_{i,j,k}^n$) to represent the averaged quantity associated to the field q at time t^n and in the cell i (resp. i, j, k) in the one-dimensional case (resp. the three-dimensional case). We use the notation $q_{i+1/2}^n$ (resp. $q_{i+1/2,j,k}^n$) to represent the quantity associated to the field q at time t^n and at the interface between cells i and $i+1$ (resp. i, j, k and $i+1, j, k$) in the one-dimensional case (resp. the three-dimensional case). Finally we define the notation $[q]_i = q_{i+1/2} - q_{i-1/2}$ in the one-dimensional case.

Acoustic-Transport splitting approach

Following Chalons et al. 2017 we use a splitting strategy that separates acoustic terms and transport terms and we choose to add the gravitational source terms to the acoustic part. This way, pressure gradient can be balanced by the gravity source term.

However we have another equation compared to the shallow water system that is the energy equation. As in Chalons et al. 2017, we want an isentropic acoustic step for smooth solutions. Thereby we choose to solve the equation on the gravitational energy,

$$\begin{aligned}
&\partial_t \rho &&+ \operatorname{div}(\rho \boldsymbol{u}) &&= 0,\\
&\partial_t (\rho \boldsymbol{u}) &&+ \operatorname{div}(\rho \boldsymbol{u} \otimes \boldsymbol{u} + p\mathbb{I}) &&= -\rho \boldsymbol{\nabla} \Phi,\\
&\partial_t (\rho \mathcal{E}) &&+ \operatorname{div}((\rho \mathcal{E} + p)\boldsymbol{u}) &&= 0,\\
&\partial_t (\rho \Phi) &&+ \operatorname{div}(\rho \Phi \boldsymbol{u}) &&= \rho \boldsymbol{u} \cdot \boldsymbol{\nabla} \Phi,
\end{aligned} \tag{1.7}$$

$$\rho\mathcal{E} = \rho e + \frac{1}{2}\rho \boldsymbol{u}^2 + \rho\Phi.$$

However, this leads to a non constant gravitational potential in the acoustic step whose time variations are compensated in the transport step in order to have a constant potential in the full step. The potential is constant in the full step at the continuous level, but discretization errors with the splitting can lead to a non-constant discretized potential. Thus we choose to introduce an approximation of the gravitational called $\Psi \approx \Phi$ and a relaxation parameter λ

$$\begin{aligned}
&\partial_t \rho &&+ \operatorname{div}(\rho \boldsymbol{u}) &&= 0,\\
&\partial_t (\rho \boldsymbol{u}) &&+ \operatorname{div}(\rho \boldsymbol{u} \otimes \boldsymbol{u} + p\mathbb{I}) &&= -\rho \boldsymbol{\nabla}\Phi,\\
&\partial_t (\rho\mathcal{E}) &&+ \operatorname{div}((\rho\mathcal{E} + p)\,\boldsymbol{u}) &&= 0,\\
&\partial_t (\rho\Psi) &&+ \operatorname{div}(\rho\Psi \boldsymbol{u}) &&= \rho \boldsymbol{u}\cdot\boldsymbol{\nabla}\Phi + \frac{\rho}{\lambda}(\Phi - \Psi),
\end{aligned} \tag{1.8}$$

$$\rho\mathcal{E} = \rho e + \frac{1}{2}\rho \boldsymbol{u}^2 + \rho\Psi.$$

We consider the relaxation system (1.8) to be an approximation of the original system (1.7) that we formally recover in the limit $\lambda \to 0$. System (1.8) is solved by first solving the system in the limit $\lambda \to \infty$ and then in the limit $\lambda \to 0$ in which Ψ is projected onto Φ, the initial condition. This way, the evolution of the gravitational potential Ψ, consistent with zero, is forced to be constant. The relaxation technic used here for the gravitational potential is similar to what is done for pressure relaxation in many approximate Riemann solvers and we emphasize that Ψ is just an intermediate used to design the scheme and can be removed when writing the final scheme (see 1.2.1).

We now turn to the discretization of the system (1.8) in the limit $\lambda \to \infty$. Transport phenomena of the form $\boldsymbol{u}\cdot\boldsymbol{\nabla}$ are separated from the other terms to give two subsystems, first the acoustic subsystem (1.9)

$$\begin{aligned}
&\partial_t \rho &&+ \rho \operatorname{div}\boldsymbol{u} &&= 0,\\
&\partial_t (\rho \boldsymbol{u}) &&+ \rho \boldsymbol{u} \operatorname{div}\boldsymbol{u} + \boldsymbol{\nabla} p &&= -\rho\boldsymbol{\nabla}\Phi,\\
&\partial_t (\rho\mathcal{E}) &&+ \rho\mathcal{E} \operatorname{div}\boldsymbol{u} + \operatorname{div}(p\boldsymbol{u}) &&= 0,\\
&\partial_t (\rho\Psi) &&+ \rho\Psi \operatorname{div}\boldsymbol{u} &&= \rho \boldsymbol{u}\cdot\boldsymbol{\nabla}\Phi,
\end{aligned} \tag{1.9}$$

then the transport subsystem (1.10)

$$\begin{aligned}
&\partial_t \rho &&+ \boldsymbol{u}\cdot\boldsymbol{\nabla}\rho &&= 0,\\
&\partial_t (\rho \boldsymbol{u}) &&+ \boldsymbol{u}\cdot\boldsymbol{\nabla}(\rho \boldsymbol{u}) &&= \boldsymbol{0},\\
&\partial_t (\rho\mathcal{E}) &&+ \boldsymbol{u}\cdot\boldsymbol{\nabla}(\rho\mathcal{E}) &&= 0,\\
&\partial_t (\rho\Psi) &&+ \boldsymbol{u}\cdot\boldsymbol{\nabla}(\rho\Psi) &&= 0.
\end{aligned} \tag{1.10}$$

We now briefly study the eigenstructure of systems (1.9)-(1.10). Let $\boldsymbol{n}$ be any unit normal vector, the acoustic system (1.9) involves seven eigenvalues: $-c, 0, c$. The fields associated with 0 (resp. $\pm c$) are linearly degenerate (resp. genuinely nonlinear), see Appendix 1.A for more details. The eigenvalues for transport system (1.10) are given by $\boldsymbol{u}\cdot\boldsymbol{n}$. Both systems (1.9)-(1.10) are hyperbolic. We emphasize here that the choice of using a relaxation procedure for the gravitational potential by introducing the equation on the gravitational potential energy $\rho\Psi$ has been made to obtain this simple wave pattern for the splitted Euler system with gravity. (i.e. the same pattern as without gravity). Other choices for the relaxation procedure (e.g. $\partial_t\Psi = 0$

in both steps) would either lead to the introduction of $\boldsymbol{u}\cdot\boldsymbol{n}$ in the eigenvalues of the acoustic subsystem or would significantly complexify the relaxation procedure for the pressure.

To summarize our numerical procedure, we propose to define a flux interface by approximating system (1.7) with a three-step procedure that involves solving the acoustic system (1.9) (acoustic step), the transport system (1.10) (acoustic step) and finally project Ψ onto Φ (relaxation step). We detail each step in the next sections using the one-dimensional equations.

Acoustic step

We follow the idea of Chalons et al. 2016 to discretize the acoustic subsystem. They introduce a pressure relaxation $\Pi \approx p$, an acoustic impedance $a \approx \rho c$ and a relaxation parameter ν to get a fully linearly degenerated system. It is then written using Lagrangian variables $(\tau, u, v, \mathcal{E}, \Psi)$ where u represents the normal velocity component at an interface and v a transverse component. We also use a mass variable $\mathrm{d}m = \rho(t^n, x)\mathrm{d}x$ where time is frozen at instant t^n

$$
\begin{aligned}
\partial_t \tau \; - \partial_m u \qquad &= 0,\\
\partial_t u \; + \partial_m \Pi \qquad &= -\frac{1}{\tau}\partial_m \Phi,\\
\partial_t v \qquad\qquad\quad &= 0,\\
\partial_t \mathcal{E} \; + \partial_m \left(\Pi u\right) &= 0,\\
\partial_t \Pi + a^2 \partial_m u \quad &= \frac{1}{\nu}\left(\Pi - p\right),\\
\partial_t \Psi \qquad\qquad\quad &= \frac{u}{\tau}\partial_m \Phi,
\end{aligned}
$$

where

$$
\mathcal{E} = e + \frac{1}{2}(u^2 + v^2) + \Psi.
$$

The discretization of this system is realized with an approximate Riemann solver that accounts for the source term by means of integral consistency and composed by three waves $-a, 0, a$, see Gallice 2002; Chalons et al. 2013; Chalons et al. 2017. After the relaxation, in which $\nu \to 0$, it gives

$$
\begin{aligned}
\widetilde{\tau}_i &= \tau_i^n + \frac{\Delta t}{\Delta m_i}[u^*]_i,\\
\widetilde{u}_i &= u_i^n - \frac{\Delta t}{\Delta m_i}[\Pi^*]_i + \frac{\Delta t}{\Delta m_i} S_i^n,\\
\widetilde{v}_i &= v_i^n,\\
\widetilde{\mathcal{E}}_i &= \mathcal{E}_i^n - \frac{\Delta t}{\Delta m_i}[\Pi^* u^*]_i,\\
\widetilde{\Pi}_i &= p^{\mathrm{EOS}}\left(\frac{1}{\widetilde{\tau}_i}, \widetilde{e}_i\right),\\
\widetilde{\Psi}_i &= \Psi_i^n - \frac{\Delta t}{\Delta m_i}(uS)_i^n,
\end{aligned}
$$

where

$$
\begin{aligned}
u^*_{i+1/2} &= \frac{1}{2}(u^n_{i+1} + u^n_i) - \frac{1}{2a}\left(\Pi^n_{i+1} - \Pi^n_i - S^n_{i+1/2}\right),\\
\Pi^*_{i+1/2} &= \frac{1}{2}\left(\Pi^n_{i+1} + \Pi^n_i\right) - \frac{a^n_{i+1/2}}{2}\left(u^n_{i+1} - u^n_i\right),\\
a^n_{i+1/2} &\geq \max\left(\rho^n_i c^n_i, \rho^n_{i+1} c^n_{i+1}\right),\\
S^n_i &= \frac{1}{2}\left(S^n_{i+1/2} + S^n_{i-1/2}\right),\\
(uS)^n_i &= \frac{1}{2}(u^*_{i+1/2} S^n_{i+1/2} + u^*_{i-1/2} S^n_{i-1/2}),\\
S^n_{i+1/2} &= -\frac{1}{2}\left(\frac{1}{\tau^n_i} + \frac{1}{\tau^n_{i+1}}\right)\left(\Phi^n_{i+1} - \Phi^n_i\right).
\end{aligned}
$$

and $a^n_{i+1/2} \geq \max\left(\rho^n_i c^n_i, \rho^n_{i+1} c^n_{i+1}\right)$ which is the so-called sub-characteristic condition (see Chalons et al. 2013).

The update of the conservative variables is then

$$
\begin{aligned}
\widetilde{L}_i \widetilde{\rho}_i &= \rho^n_i,\\
\widetilde{L}_i \widetilde{(\rho u)}_i &= (\rho u)^n_i - \frac{\Delta t}{\Delta x}[\Pi^*]_i + \frac{\Delta t}{\Delta x} S^n_i,\\
\widetilde{L}_i \widetilde{(\rho v)}_i &= (\rho v)^n_i,\\
\widetilde{L}_i \widetilde{(\rho \mathcal{E})}_i &= (\rho \mathcal{E})^n_i - \frac{\Delta t}{\Delta x}[\Pi^* u^*]_i,\\
\widetilde{L}_i \widetilde{(\rho \Psi)}_i &= (\rho \Psi)^n_i - \frac{\Delta t}{\Delta x}(uS)^n_i
\end{aligned}
$$

where $\widetilde{L}_i = 1 + \frac{\Delta t}{\Delta x}[u^*]_i$.

Transport step

The transport subsystem can be written in the following form, for $b \in \{\rho, \rho u, \rho v, \rho \mathcal{E}, \rho \Psi\}$

$$
\partial_t b + \partial_x (bu) - b \partial_x u = 0,
$$

that is discretized as follows

$$
b^{n+1}_i = \widetilde{b}_i - \frac{\Delta t}{\Delta x}\left[\widetilde{b} u^*\right]_i + \widetilde{b}_i \frac{\Delta t}{\Delta x}[u^*]_i.
$$

The interface term $\widetilde{b}_{i+1/2}$ is defined by the upwind choice with respect to the velocity $u^*_{i+1/2}$

$$
\widetilde{b}_{i+1/2} = \begin{cases} \widetilde{b}_i & \text{if} \quad u^*_{i+1/2} \geq 0 \\ \widetilde{b}_{i+1} & \text{if} \quad u^*_{i+1/2} \leq 0 \end{cases}
$$

Relaxation step

At this stage, the relaxed gravitational potential Ψ still evolves in time. So we perform the relaxation $\lambda \to 0$ that boils down to set $\Psi^{n+1}_i = \Phi_i$.

Overall algorithm

Gathering the previous steps and intermediate variables, the overall scheme reads

$$\begin{aligned}
\rho_i^{n+1} &= \rho_i^n - \frac{\Delta t}{\Delta x}[\widetilde{\rho} u^*]_i, \\
(\rho u)_i^{n+1} &= (\rho u)_i^n - \frac{\Delta t}{\Delta x}\left[\widetilde{(\rho u)} u^* + \Pi^*\right]_i + \frac{\Delta t}{\Delta x} S_i^n, \\
(\rho v)_i^{n+1} &= (\rho v)_i^n - \frac{\Delta t}{\Delta x}\left[\widetilde{(\rho v)} u^*\right]_i, \\
(\rho \mathcal{E})_i^{n+1} &= (\rho \mathcal{E})_i^n - \frac{\Delta t}{\Delta x}\left[\left(\widetilde{(\rho \mathcal{E})} + \Pi^*\right) u^*\right]_i
\end{aligned} \tag{1.11}$$

It may also be expressed as a first-order classic finite-volume scheme involving flux terms for the conservative part for energy $\rho E = \rho e + \frac{1}{2}\rho u^2$ and source terms for gravity

$$\begin{aligned}
\rho_i^{n+1} &= \rho_i^n - \frac{\Delta t}{\Delta x}[\widetilde{\rho} u^*]_i, \\
(\rho u)_i^{n+1} &= (\rho u)_i^n - \frac{\Delta t}{\Delta x}\left[\widetilde{(\rho u)} u^* + \Pi^*\right]_i - \Delta t\{\rho \partial_x \Phi\}_i, \\
(\rho v)_i^{n+1} &= (\rho v)_i^n - \frac{\Delta t}{\Delta x}\left[\widetilde{(\rho v)} u^*\right]_i, \\
(\rho E)_i^{n+1} &= (\rho E)_i^n - \frac{\Delta t}{\Delta x}\left[\left(\widetilde{(\rho E)}^{NG} + \Pi^*\right) u^*\right]_i - \Delta t\{\rho u \partial_x \Phi\}_i,
\end{aligned} \tag{1.12}$$

where

$$\begin{aligned}
\Delta x\{\rho u \partial_x \Phi\}_i &= [\widetilde{\rho} u^* \Phi]_i - [\widetilde{\rho} u^*]_i \Phi_i, \\
\Delta x\{\rho \partial_x \Phi\}_i &= -S_i^n, \\
\widetilde{(\rho E)}_i^{NG} &= (\rho E)_i^n - \frac{\Delta t}{\Delta x}[\Pi^* u^*]_i.
\end{aligned}$$

We emphasize that both formulations are equivalent and conservative for the energy $\rho\mathcal{E}$. A non-conservative energy approach is also detailed in Appendix 1.B.

We can notice that in the case of a constant gravitational potential, we recover the original scheme derived in Chalons et al. 2016.

On the low-Mach correction

As for the scheme of Chalons et al. 2016 and as explained in Dellacherie 2010, the numerical scheme defined by (1.11) poorly performs in the low Mach regime due to truncature error of magnitude $\frac{\Delta x}{\mathrm{Ma}}$ that comes from the term $\Pi^*_{i+1/2}$. To tackle this issue, following Chalons et al. 2016 we modify the upwinding part of $\Pi^*_{i+1/2}$ thanks to an extra parameter $\theta_{i+1/2}$ by setting

$$\begin{aligned}
\Pi^*_{i+1/2} &= \frac{1}{2}\left(\Pi^n_{i+1} + \Pi^n_i\right) - \frac{a^n_{i+1/2}\theta_{i+1/2}}{2}\left(u^n_{i+1} - u^n_i\right), \\
\theta_{i+1/2} &= \min\left(\mathrm{Ma}_{i+1/2}, 1\right), \\
\mathrm{Ma}_{i+1/2} &= \frac{|u^*_{i+1/2}|}{\max\left(c^n_i, c^n_{i+1}\right)}.
\end{aligned} \tag{1.13}$$

Using a truncation analysis in dimensionless form it can be shown that this correction acts like a rescaling of the numerical diffusion induced by the pressure discretization (see Chalons et al. 2016).

As we can see, the low-Mach correction does not directly come from the derivation of the numerical scheme 1.11. Some ongoing works are trying to derive directly all-Mach schemes using more sophisticated relaxation schemes (see Bouchut et al. 2017).

On the well-balanced property

A numerical scheme is said to be well-balanced for equilibrium states satisfying equation (1.6), if it exists a discrete counterpart of equation (1.6) in which solutions are preserved by the numerical scheme.

The discrete counterpart of equation (1.6) for scheme (1.11) is given by

$$\begin{aligned} u_i^n = 0, \quad v_i^n = 0, \\ \Pi_{i+1}^n - \Pi_i^n = -\frac{1}{2}\left(\rho_i^n + \rho_{i+1}^n\right)\left(\Phi_{i+1} - \Phi_i\right), \end{aligned} \tag{1.14}$$

Let us now verify that we have obtained a well-balanced scheme. If at time t^n, for some density profile the initial state reads as in (1.14) then fluxes from the acoustic step reduce to

$$\begin{aligned} u_{i-1/2}^* = u_{i+1/2}^* = 0 \\ \left[\Pi^*\right]_i = \frac{1}{2}\left(\Pi_{i+1}^n - \Pi_i^n\right) + \frac{1}{2}\left(\Pi_i^n - \Pi_{j-1}^n\right) + S_i^n. \end{aligned}$$

Then we have for the acoustic step

$$\begin{aligned} \widetilde{u}_i = u_i^n, \quad \widetilde{v}_i = v_i^n, \\ \widetilde{\rho}_i = \rho_i^n, \quad \widetilde{\mathcal{E}}_i = \mathcal{E}_i^n. \end{aligned}$$

Finally, because $u_{i+1/2}^*$ vanishes, transport step is trivial and the initial state remains unchanged. Once we have made the appropriate choice for the discretization of the gravitational source term in the acoustic step, the well-balanced property is automatically verified without the need to introduce an other algorithmic correction.

1.2.2 Dissipative fluxes — Parabolic system

We now turn to the discretization of dissipative fluxes (1.2)-(1.3b). They are discretized using second order discrete fluxes

$$\begin{aligned} \left[\mathrm{div}\boldsymbol{f}^{dissipative}\right]_{i,j,k} = & \frac{\left(f_{x,i+1/2,j,k} - f_{x,i-1/2,j,k}\right)}{\Delta x} \\ & + \frac{\left(f_{y,i,j+1/2,k} - f_{y,i,j-1/2,k}\right)}{\Delta y} \\ & + \frac{\left(f_{z,i,j,k+1/2} - f_{z,i,j,k-1/2}\right)}{\Delta z} \end{aligned}$$

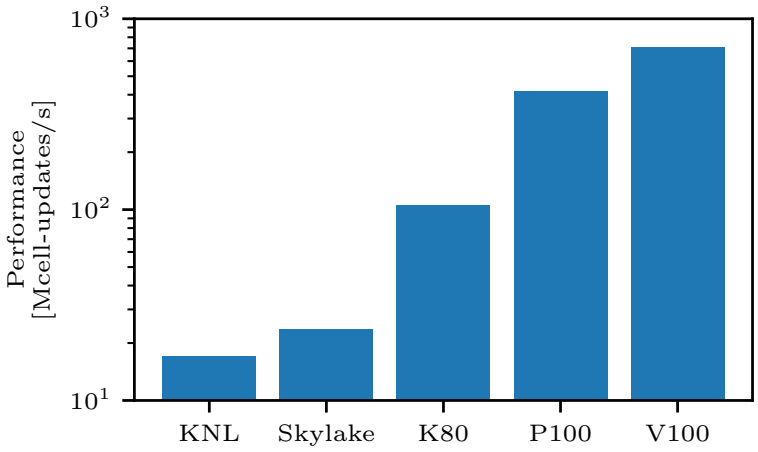


Figure 1.2: Comparison of performance on different architectures: Intel KNL, Intel Skylake (one socket), NVIDIA K80, NVIDIA P100 and NVIDIA V100. Measures on Intel KNL and Intel Skylake were performed on Joliot-Curie's supercomputer at TGCC using the same code. In our case we obtain better results with the Intel Skylake than the Intel KNL due to a lack of vectorization. Going to a GP-GPU we have a speed-up around five with a NVIDIA K80 compared to multi-core architecture and seven between NVIDIA K80 and V100.

where $\boldsymbol{f}^{dissipative}$ is either the heat flux $\boldsymbol{q}_{\text{heat}}$ or the viscous flux $\boldsymbol{\tau}_{\text{visc}}$. In the case of the heat flux we have

$$q_{x,i+1/2,j,k} = -\kappa \frac{(T_{i+1,j,k} - T_{i,j,k})}{\Delta x}$$
$$q_{y,i,j+1/2,k} = -\kappa \frac{(T_{i,j+1,k} - T_{i,j,k})}{\Delta y}$$
$$q_{z,i,j,k+1/2} = -\kappa \frac{(T_{i,j,k+1} - T_{i,j,k})}{\Delta z}$$

With the addition of the viscous terms and the heat flux, this all-regime well-balanced scheme is now well-suited for the study of convection problems in highly stratified flows in both low Mach and high Mach regimes. Before showing validating numerical tests, we present some specificities about the numerical implementation and parallelization used in this work.

1.3 Implementation and parallelization

In this section we describe the implementation of the scheme using Kokkos library. We begin by giving a brief overview of the Kokkos library.

1.3.1 Exascale computing

To reach the exascale, the distributed memory model is not sufficient to take advantage of all the computing power of new architectures. There are mainly two reasons for this. First, nodes of supercomputers tend to grow more and more and hence are more suited to a shared memory

model (Sunderland et al. 2016). Secondly, nodes tend be more and more heterogeneous by using multi-core, many-core and/or accelerators like GP-GPUs. So it means that even if shared memory is exposed, it needs to be handled differently from one architecture to another. For example we can think of OpenMP or C++11 threads for multi-core and many-core processors, and CUDA or OpenACC for GP-GPUs.

Moreover this architecture heterogeneity raises a performance portability issue. Currently, many HPC codes are optimized for some specific architectures to get the maximum computing power. However this optimization process couples the numerical scheme to its implementation details like the memory management, the loop ordering, cache blocking and so on. Hence running a code on a different architecture results in bad performance.

We propose to use the recent C++ library Kokkos (see Carter Edwards et al. 2014) that implements a new shared memory model. Using abstract concepts such as execution spaces (where a function is executed), data spaces (where data resides) and execution policies (how the function is executed) the library is able to efficiently take advantage of multi-core many-core processors and GP-GPUs. This way the portability relies on the library and no more on the numerical code.

1.3.2 Implementation

Following the work of Kestener 2017, the code is then organized with computation kernels:

- Acoustic and transport kernels,
- Viscous and heat diffusion operator kernels,
- Conservative variables to primitive variables kernel,
- Time step kernel.

Each kernel is a C++ functor. They are given to Kokkos through the function Kokkos::parallel_for. Internally, depending on the device chosen at compile-time, it hides a parallelized one-dimensional loop where the current index is given as an argument to the functor. This index is then interpreted as a cell index in the domain.

Kokkos only deals with shared memory systems. We use the Message Passing Interface (MPI) programming model with a regular domain decomposition to take advantage of distributed memory machines across multiples computing nodes. Kokkos is then used as a shared memory programming model inside each node. These domains are endowed with ghost zones which are used to both implement physical boundary conditions and to contain values from neighbour domains. Communications are handled through the ghost cell pattern (see Kjolstad and Snir 2010). Thus for a given direction X, Y (or Z) and a given side, left or right, one MPI process sends data from its domain to its neighbour's ghost zone and receives data into its own ghost zone.

1.3.3 Performance results

Thanks to Kokkos, we were able to use **the same code** on different architectures like Intel Skylake, Intel Knights Landing (KNL) and NVIDIA GP-GPUs (K80, P100, V100). We measured performance on the Intel Skylake and the Intel KNL partition of the Joliot Curie machine at TGCC. Figure 1.2 shows the results. We see that the Kokkos library is able to provide good performance on the different tested architectures. Nevertheless, even if the peak performance of the Intel KNL architecture is higher than the Intel Skylake one we have better performance on

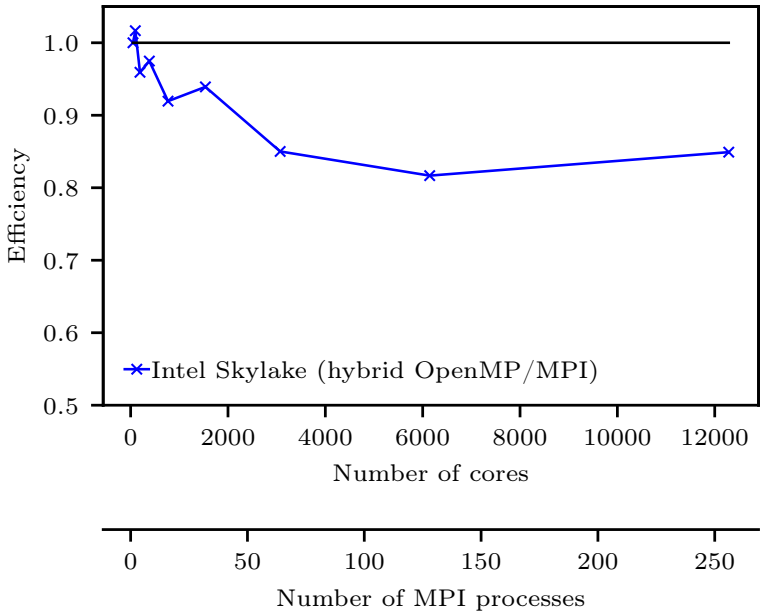


Figure 1.3: Weak scaling results obtained on Joliot Curie's Intel Skylake partition at TGCC. We use a hybrid MPI-OpenMP configuration in which one MPI task is bound to a socket. Simulations run for 1000 time steps and each MPI process treats 128^3 cells. We see that the efficiency reaches a plateau of 85%.

the Intel Skylake architecture. We also notice the important speed-up (around five) between the Intel Skylake architecture and the NVIDIA V100 GP-GPU.

Figure 1.3 shows a weak scaling test performed with a hybrid configuration OpenMP/MPI. We went up to 512 MPI processes, one MPI process per Intel Skylake socket to avoid NUMA effects. It results in a total of 12288 cores at 512 MPI processes. Each MPI process is getting a piece of the whole domain of 128^3, so a domain of 44^3 per core. We can see that we obtain a plateau of 85% of maximum performance from 128 MPI processes.

The performances obtained with the use of the Kokkos library are encouraging for the study of convection problems with the ARK code on massively parallel present and future architectures. In the next section, we use several numerical tests to show that the numerical scheme used in the ARK code is indeed very well suited for the study of convection.

1.4 Numerical results

In this section we specialize the equation of state 1.3a. We will use an ideal gas satisfying

$$p^{\mathrm{EOS}}(\rho, e) = (\gamma - 1)\,\rho e$$

where γ is the adiabatic index of the gas. The speed of sound satifies the following simple relation

$$c^2 = \gamma \frac{p}{\rho}$$

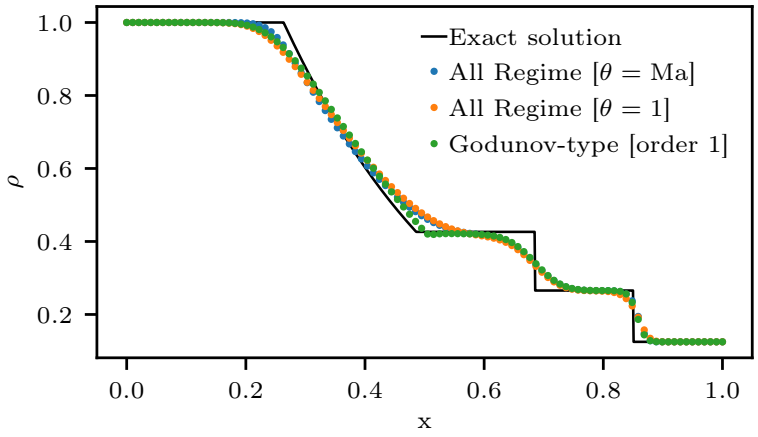


Figure 1.4: Sod's test case simulations. Figure shows a snapshot of the density profile ρ for the All-Regime scheme, with and without the low-Mach correction, a first order Godunov-type scheme (HLLC) and the exact solution. Spatial resolution is nx = 100. We see that the All-Regime scheme gives results close to the Godunov-type scheme around discontinuities but is more diffusive in the rarefaction wave.

We emphasize that it is possible to use a different equation of state with the all-regime well-balanced numerical scheme. Moreover we consider two versions of the all-regime scheme depending on the low-Mach correction. We will refer to the disabled low-Mach correction scheme when $\theta = 1$ and to the enabled one when θ follows equation 1.13.

We will test different properties of the scheme with different test cases: wave speeds with the Sod test (no gravity), low-Mach accuracy with the Gresho vortex test (no gravity), hydrostatic balance with the test of an atmosphere at rest and out of equilibrium behavior with the Rayleigh-Taylor test. We then use the ARK code for the study of Rayleigh-Bénard convection.

1.4.1 Shock tube test

The Sod shock tube (Sod 1978) is a classical test for compressible solvers. It tests the ability of the solver to have correct wave speeds and its numerical diffusion near discontinuities.

The computational domain is the interval [0,1], the initial condition is defined by

$$(\rho, p, u) = \begin{cases} (1, 1, 0) & \text{if} \quad x < 0.5, \\ (0.125, 0.1, 0) & \text{if} \quad x \geq 0.5. \end{cases}$$

Results are shown in figure 1.4 for simulations with nx = 100. First we can observe that the solver is as good as a first order Godunov-type scheme with a HLLC approximate Riemann solver around the contact discontinuity and the shock. However the rarefaction wave is a bit more diffused. We also notice that the low-Mach correction does not influence the behavior of the scheme for this test case. However we want to stress out some instability near discontinuities, as shown in Chalons et al. 2016. This can also be seen in a double shock waves test case.

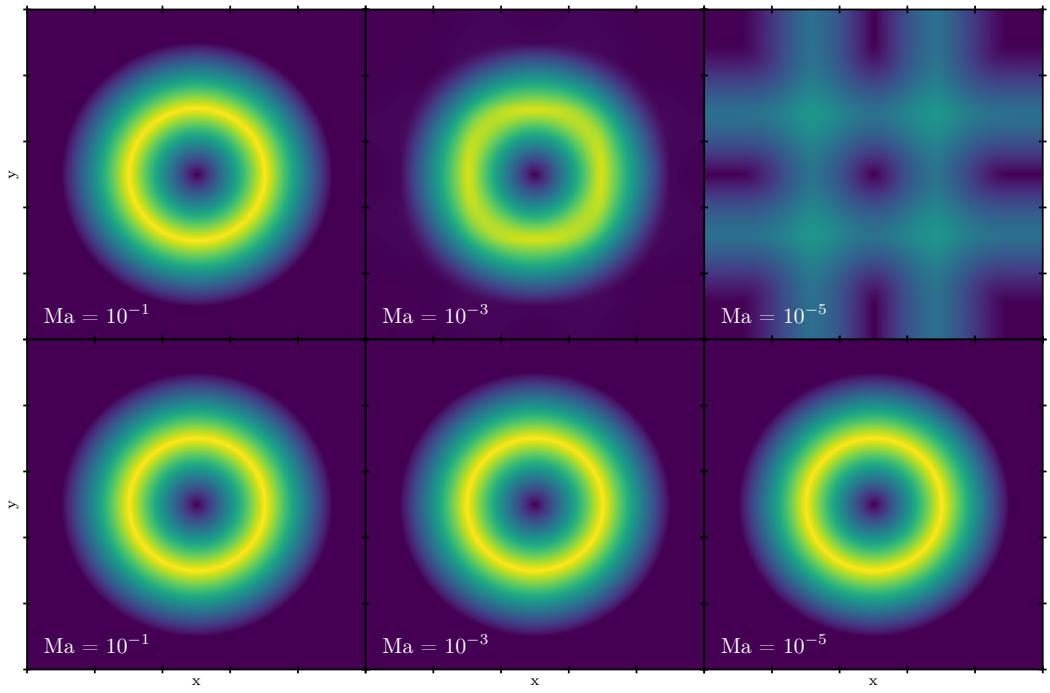


Figure 1.5: Gresho vortex simulations. Snapshots of the magnitude of the velocity field at time $t_f = 10^{-3}$, for a resolution of 512^2 and for different Mach numbers. First line shows results where the low-Mach correction is disabled and second line where it is enabled. We see that without the low-Mach correction the scheme fails at simulating low-Mach flows.

1.4.2 Gresho vortex test case

The Gresho vortex (Gresho and Chan 1990; Miczek et al. 2015) is a test case that has already been used to test numerical schemes in the low Mach regime. It is a two dimensional stationary test case that can be parameterized by the maximum value of the Mach number. It is thus well-suited to study the behavior of the scheme in the low Mach regime. We recall that the test case is defined using polar coordinates (r, θ) defined with respect to the center of the vortex as follows

$$\rho = \rho_0,$$

$$(u_r, u_\theta) = \begin{cases} (0, 5r) & 0 \leq r < 0.2, \\ (0, 2 - 5r)\,, & 0.2 \leq r < 0.4, \\ (0, 0)\,, & 0.4 \leq r \end{cases}$$

$$p = \begin{cases} p_0 + 12.5r^2, & 0 \leq r < 0.2, \\ p_0 + 12.5r^2 + 4 - 20r + 4\ln(5r), & 0.2 \leq r < 0.4, \\ p_0 - 2 + 4\ln 2, & 0.4 \leq r. \end{cases}$$

where p_0 satisfies $p_0 = \frac{1}{\gamma \mathrm{Ma}^2}$. In this case Ma is a parameter and γ is the adiabatic index of

the ideal gas. The velocity is normalized so a particle placed at the peak of velocity ($u = 1.0$ at location $r = 0.2$) make a full rotation in $\Delta t = \frac{2}{5}\pi \approx 1.26$.

We ran a serie of simulations with different solvers where we explored parameter space nx and Ma from 32 to 2048 and from 1.0 to 1.0×10^{-5} respectively. Final time is set to $t_f = 1.0 \times 10^{-3}$, which has been chosen sufficiently small such that the error doesn't saturate.

Figure 1.5 shows snapshots of the the magnitude of the velocity field at the final time and at resolution 512^2. We see that when the Mach number decreases the velocity field becomes more and more degraded when the low-Mach correction is disabled. At $\mathrm{Ma} = 10^{-5}$, the vortex has completely disappeared. Figures 1.6 and 1.7 show more quantitative results where we show absolute L^1 error on velocity in function of the Mach number Ma and the spatial resolution dx respectively. Figure 1.6 shows that L^1 error on velocity depends on the Mach number. More precisely we measure a slope of -1 on schemes or order 1 and a slope of -0.5 on scheme of order 2. On the other hand the low Mach correction of the all-regime scheme gives a uniform error with respect to the Mach number. Figure 1.7 shows convergence curves at $\mathrm{Ma} = 1.0 \times 10^{-3}$. We

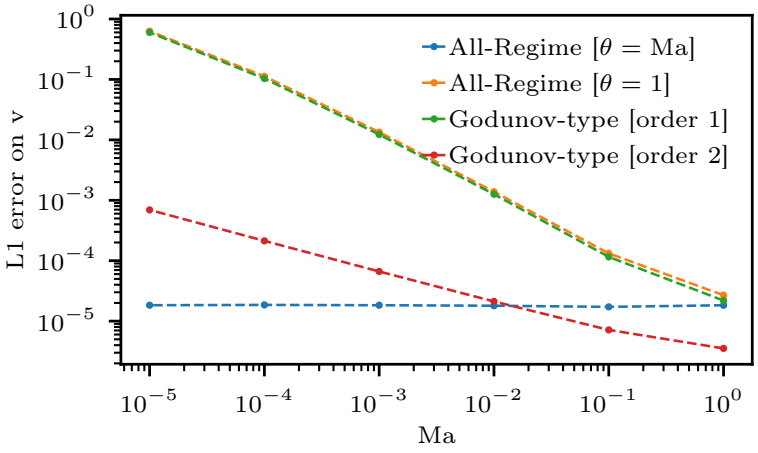


Figure 1.6: Gresho vortex simulations. L^1 error on the velocity in function of the Mach number at a fixed number of points of nx = 2048

see that both Godunov-type and all-regime without the low Mach correction converge at order 1 as expected. Nevertheless Godunov-type with Muscl-Hancock reconstruction converges only at order 1.5. It may be due to the lack of regularity of the velocity field as it can be observed in the case of a contact discontinuity (see Springel 2010). All-Regime scheme shows two different behaviors, at first it converges at order 1.5 then around nx = 1024 the slope changes and it converges at order 1.2. We assume that at higher resolution we would recover order 1. We see that at low Mach number the precision, independently of the order, is better than the one of a Godunov-type scheme.

1.4.3 Well-balanced test case

The well-balanced test case is a simple isothermal column of atmosphere at equilibrium. This column of atmosphere is in a stable equilibrium state. The test allows us to measure the ability

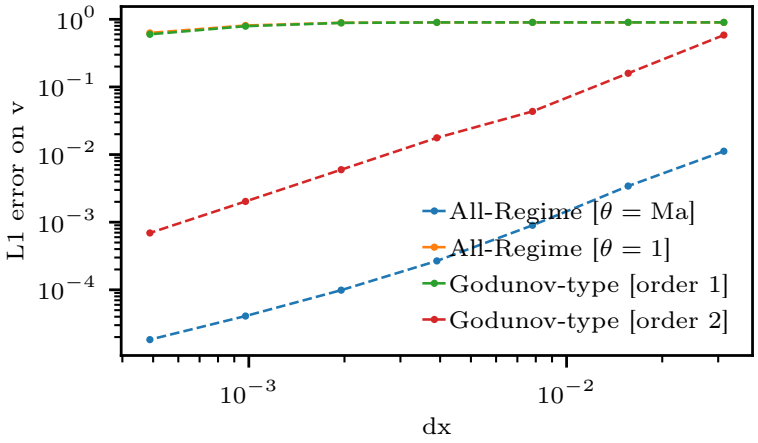


Figure 1.7: Gresho vortex simulations. L^1 error on the velocity in function of the spatial resolution, at a fixed Mach number of $\mathrm{Ma} = 10^{-5}$.

of the scheme to preserve this equilibrium. After normalization, it is given by

$$p(z) = \rho(z) = e^{-z}$$
$$T = 1$$

which is the solution of the following system

$$\frac{\mathrm{d}p}{\mathrm{d}z} = -\rho\frac{\mathrm{d}\Phi}{\mathrm{d}z}$$
$$T = 1$$
$$p = \rho T$$

We take advantage of the formula (1.14) and we initialize the test case with the following formula

$$\frac{p_{i+1} - p_i}{\Delta z} = -\frac{\rho_i + \rho_{i+1}}{2}\frac{\Phi_{i+1} - \Phi_i}{\Delta z}$$
$$T_i = 1$$
$$p_i = \rho_i T_i$$

The computational domain used is the interval $[0, 3]$. Results are displayed in table 1.1 at time $t = 10$, more than three times the sound crossing time in the box. We see that we stay near machine precision at the end of the simulation. We see a shift of two orders of magnitude in the error when using the low-Mach correction. The reason of this shift is not entirely clear and is difficult to interpret as it involves truncature errors. Looking at the spatial pattern of the error in the simulation, it does seem to come from the boundary conditions (extrapolation of the hydrostatic balance for pressure and density and reflexive conditions for the velocity) with the use of the low-Mach correction. A more appropriate boundary condition might remove this shift in the error (which is in any case sufficiently small and stable to allow the use of controlled seeded perturbations).

Table 1.1: Isothermal atmospheres at rest. Table shows for different spatial resolutions the maximum velocity in the domain. We see that the velocity is maintained around zero up to the machine precision, thus illustrating the well-balanced property.

Number of cells	velocity ($\theta = 1$)	velocity ($\theta = Ma$)
128	2.910^{-15}	1.410^{-13}
256	8.110^{-15}	5.710^{-13}
512	1.510^{-14}	1.110^{-12}
1024	2.210^{-14}	2.210^{-12}
2048	4.710^{-14}	1.610^{-12}
4096	1.110^{-13}	4.010^{-12}

1.4.4 Rayleigh-Taylor instability test case

The Rayleigh-Taylor test case is a two dimensional test case where two fluids of different densities are superposed and are at equilibrium. The denser one is on top. A small perturbation is introduced to break equilibrium.

The full setup is as follow, for a domain $[-0.25, 0.25] \times [-0.75, 0.75]$:

$$\begin{aligned}
\rho(x,y) &= \begin{cases} 1 & \text{for } y < 0 \\ 2 & \text{for } y >= 0 \end{cases} \\
p(x,y) &= \rho g y \\
u(x,y) &= 0 \\
v(x,y) &= \frac{C}{4}\left(1+\cos\left(\frac{2\pi x}{L_x}\right)\right)\left(1+\cos\left(\frac{2\pi y}{L_y}\right)\right)
\end{aligned}$$

Where $C = 0.01$ is the magnitude of the velocity perturbation, $L_x = 0.5$ and $L_y = 1.5$ are the size of the domain in each direction. We do not need to use the well-balanced formula (1.14), the equilibrium is preserved in the case $A = 0$.

Figure 1.8 shows two simulations of the Rayleigh-Taylor test case, one with the low Mach correction and the other without it ($\theta^n_{i+1/2} = 1$). Both simulations are at the same time $t = 12.4$ and the same resolution 200×600. The yellow part is at density 2 and the purple is at density 1. We see that we recover the classical linear growing mode. Moreover the simulation with the low Mach correction is able to capture secondary instabilities in the non linear regime. They are closer to the second order Godunov-type simulation than the order one. However the low Mach correction does not help on the interface diffusion between the two mediums. It also shows a peak that is not present without the correction at the same resolution. This spurious behavior is therefore caused by the low Mach correction that removes some numerical diffusion in the scheme. By looking at higher resolutions, we identify that this peak is a grid-seeded secondary RT unstable mode that appears at the top of the large scale seeded mode. This type of secondary modes are not unexpected and can be seen for example in Fig. 9 of Almgren et al. 2010. This peak disappears with the addition of some physical viscosity in the simulation.

1.4.5 Rayleigh-Bénard instability test case

This last test case is about compressible convection simulations both in 2D and 3D. In this test case there are different important parameters. First, from stability analysis we know that the Rayleigh number Ra is an important non-dimensional number. Beyond a threshold, called

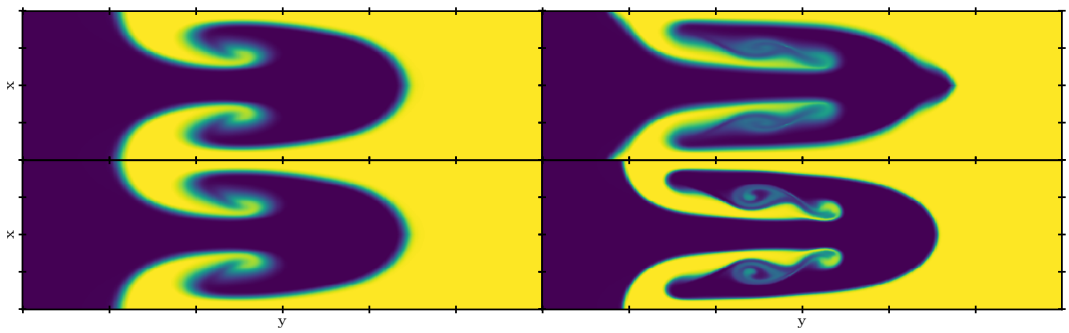


Figure 1.8: Rayleigh-Taylor simulations. Figure shows snapshots of density, one in purple and two in yellow at time $t = 12.4$ and for a resolution of 200×600. First line show results with the the all-regime scheme, where on the left the low-Mach correction is disabled and is enabled on the right. Second line shows results with a Godunov-type scheme, on the left it is first order, on the right it is second order using a Muscl-Hancock scheme. We see that with the low-Mach correction we recover features only present at second order for a standard Godunov-type scheme.

the critical Rayleigh Ra_c, the convection process starts and efficiently transports the heat (see Figures 1 and 3 in Hurlburt et al. 1984). Below this threshold, diffusion processes are sufficient to transport heat and no material displacement is necessary. Then another important parameter is the density stratification χ which the ratio between the density at the bottom of the domain and the density at the top. In the highly stratified case, study of convection becomes more difficult as there is not a unique Rayleigh number but more a whole range of values extending on the scale height. Notice that when $\chi \to 1$ we recover the Boussinesq-like situation. Finally the last parameter is the polytropic index m which is a measure of how close is the initial temperature gradient from the adiabatic gradient. One can show that the Schwarzschild criterion writes

$$m + 1 < \frac{\gamma}{\gamma - 1} = 2.5, \quad \gamma = \frac{5}{3}$$

The initial setup is inspired from Hurlburt et al. 1984; Toomre et al. 1990. Following their notation, the initial state is given by a polytropic profile of polytropic index m

$$T = z, \quad \rho = z^m, \quad p = z^{m+1}$$

where z is the vertical variable. It is initialized using to the recursive formula (1.14). So we begin with a hydrostatic equilibrium that we destabilize whether with a velocity mode perturbation or with a temperature random perturbation.

2D case

We begin with 2D simulations in a weak stratification setup where $\chi = 1.1$ and $m = 1.3$ in order to be close to the adiabatic gradient. The initial perturbation is close to the fundamental velocity mode. The spatial resolution is set to 128^2, and we impose the temperature flux on the bottom boundary. We then obtain stationary symmetric convective rolls. We study the effect of the low-Mach correction on the onset of the Rayleigh-Bénard instability by varying the initial Rayleigh number. Figure 1.9 shows the evolution of the mean absolute velocity. The linear phase, in logarithmic scale, corresponds to the exponential growth of modes. We can see that

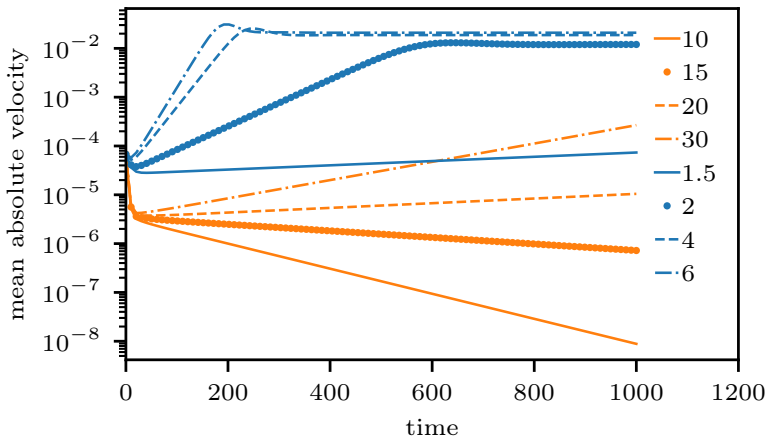


Figure 1.9: Rayleigh-Bénard instability simulations in 2D. Figure shows the time evolution of the mean absolute velocity for different ratios of Rayleigh number over critical Rayleigh number (see legend). Blue points show the case where the low-Mach correction is enabled and orange ones where it is disabled. We observe that when the low-Mach correction is enabled the onset of convection is closer to the expected critical Rayleigh number.

without the low-Mach correction we have an effective critical Rayleigh number between 10 and 15. Whereas with the low-Mach correction we recover an effective critical Rayleigh number close to the theoretical critical one.

If we now turn to a stronger stratification, the convective rolls pattern change. We increase the density ratio to $\chi = 21$. Figure 1.10 shows a snapshot of the local Mach number field with the velocity field, low-Mach correction enabled. As observed in Hurlburt et al. 1984 we see a downward shift of the center of mass of convective rolls compared to the weak stratification case. By conservation of mass, the upper part of the convective roll has to be larger. The strong stratification case also exhibits higher Mach flows, around Ma ≈ 0.5 at the top of the box due to the low density. The all-regime well balanced scheme is indeed able to capture properly convection in highly stratified and high Mach flows.

3D case

We now turn to 3D simulations in a weak stratification situation. In this setup we want to look at the effect of the low-Mach correction on the kinetic energy spectrum in a more turbulent situation. So we change the polytropic index to $m = 0.1$ and increase the initial Rayleigh number to Ra ≈ 650000. We also change the boundary condition to a fixed temperature for both boundaries in order to continuously force a large Rayleigh number in the simulation. By using different upscaling, from 128^3 to 512^3 we reach a stationary state [2]. Figure 1.12 shows a snapshot of the velocity in the box. We see large and structured vertical flows whereas in horizontal plans the flow is more turbulent. In order to study the different scales and the energy in this turbulent state we compute power spectrum of the kinetic energy of the horizontal middle plane. Figure 1.11 shows the results, the orange curve corresponds to the simulation performed with the low-Mach

2. The simulation outputs are available at `http://opendata.erc-atmo.eu`

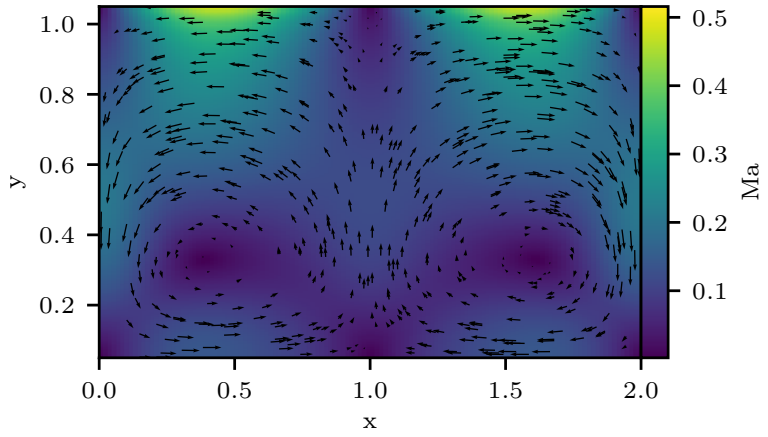


Figure 1.10: Rayleigh-Bénard instability simulations. Snapshot of the local Mach number field and the velocity field. We see that in the strong stratification case, there is a large range of Mach, near zero at the center of rolls up to half at the upper boundary.

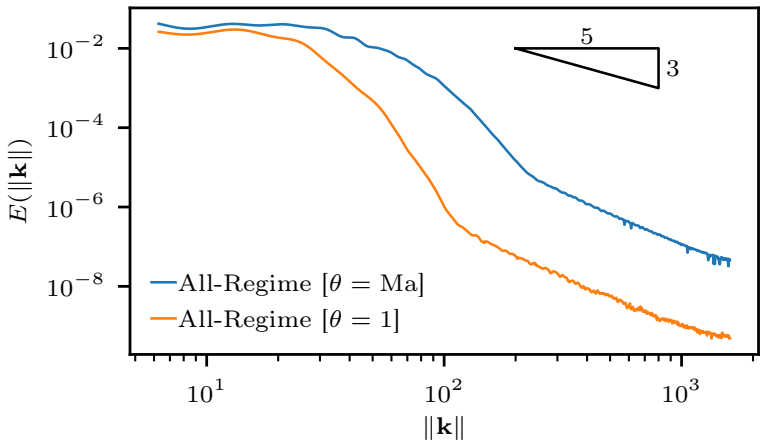


Figure 1.11: Rayleigh-Bénard instability simulations in 3D. Figure shows the kinetic energy spectrum of the horizontal middle plane. The blue line corresponds to the scheme with low-Mach correction and the orange one without the low-Mach correction. We see more kinetic energy at all scales in the case of the low-Mach correction.

correction and the blue one without it. We see a net difference in the overall kinetic energy due to a lower dissipation into the internal energy. We notice that we recover higher kinetic energies at all scales showing that the low Mach correction is important to properly capture the power spectrum of turbulent convection.

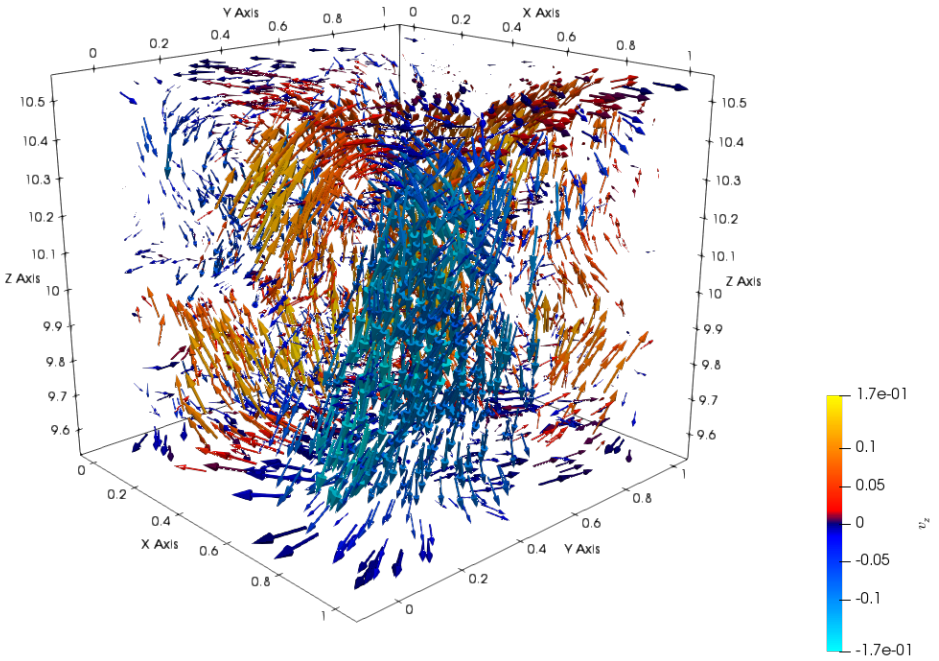


Figure 1.12: Rayleigh-Bénard instability simulations in 3D. Figure shows the velocity field in the box. The length of an arrow is scaled using the magnitude of the local velocity. The colorbar represents the vertical component of the velocity showing the direction of the flow.

Conclusion

We have presented a new numerical code that is able to perform simulations of convection without any approximation of Boussinesq nor anelastic type. To do so we have adapted an all-Mach number scheme into a well-balanced scheme for gravity. We have been able to show that it preserves arbitrary discrete equilibrium states up to the machine precision. Moreover the low-Mach correction in the numerical flux allows to be more precised in the low-Mach regime. This new scheme is well suited to properly study highly stratified and high Mach convective flows. The low Mach correction is important to properly capture convection modes in the laminar low Mach regime and the kinetic energy power spectrum in the turbulent regime. This code has been parallelized using a hybrid approach MPI+Kokkos in order to be well prepared for running on forthcoming exascale machines.

Further work will consist in using the implicit-explicit approach to reach very low Mach number simulations, see Chalons et al. 2016, and still keeping the well-balanced property for the gravity source term. Indeed by solving the acoustic part implicitly we avoid the restrictive CFL condition due to the fast acoustic waves. With both the explicit-explicit and implicit-explicit approach, this numerical scheme will be able to efficiently study convection problems in all regimes, low Mach and high Mach on the largest next generation massively parallel architectures.

Appendix

1.A Eigenstructure of the acoustic system

For the sake of simplicity, the eigenstructure analysis of the acoustic system (1.9) is made in the one-dimensional case. We use the following change of variables, valid for smooth flows

$$(\rho, \rho u, \rho\mathcal{E}, \rho\Psi, \Phi) \to (\rho, u, s, \Psi, \Phi),$$

where s is the specific entropy. By using equation of mass, one obtains

$$\begin{aligned}
\partial_t \rho + \rho \partial_x u &= 0, \\
\partial_t u + \frac{1}{\rho}\partial_x p^{\mathrm{EOS}} + \partial_x \Phi &= 0, \\
\partial_t e - \frac{p}{\rho}\partial_x u &= 0, \\
\partial_t \Psi - u\partial_x \Phi &= 0, \\
\partial_t \Phi &= 0.
\end{aligned}$$

By using the second law of Thermodynamics and the equation on the specific internal energy, one can show that $\partial_t s = 0$ (see Godlewski and Raviart 1996). Thus the acoustic system (1.9) writes equivalently

$$\begin{aligned}
\partial_t \rho + \rho \partial_x u &= 0, \\
\partial_t u + \frac{1}{\rho}\partial_x p^{\mathrm{EOS}} + \partial_x \Phi &= 0, \\
\partial_t s &= 0, \\
\partial_t \Psi - u\partial_x \Phi &= 0, \\
\partial_t \Phi &= 0.
\end{aligned} \tag{1.15}$$

The Jacobian matrix associated to the quasi-linear system 1.15 involves five eigenvalues: $-c < 0 < c$ where 0 is degenerated three times and c satifies $c^2 = \partial_\rho p^{\mathrm{EOS}}(\rho, s)$. It is then hyperbolic. The four eigenvectors are given by

$$\boldsymbol{r}_0^0 = \left(\partial_s p, 0, -c^2, 0, 0\right)^{\mathrm{T}}, \quad \boldsymbol{r}_0^1 = (0, 0, 0, 1, 0)^{\mathrm{T}}, \quad \boldsymbol{r}_{\pm c} = (\rho, \pm c, 0, 0, 0)^{\mathrm{T}}.$$

Clearly the field associated to the stationary wave is linearly degenerated. The fields associated to $\pm c$ are genuinely non-linear under the condition that the following quantity does not vanish

$$\pm \nabla c\,(\rho, s) \cdot \boldsymbol{r}_{\pm c} = \pm \rho \partial_\rho c = \pm \frac{\rho}{2c}\partial^2_{\rho\rho} p^{\mathrm{EOS}}.$$

1.B Non-conservative energy scheme

To obtain the non-conservative scheme, we do not need anymore the relaxation on the gravitational potential. This scheme is then obtained through the following splitting, for the acoustic subsystem

$$\begin{aligned}
\partial_t \rho \quad &+ \quad \rho \mathrm{div} \boldsymbol{u} &&= 0, \\
\partial_t (\rho \boldsymbol{u}) &+ \rho \boldsymbol{u} \mathrm{div} \boldsymbol{u} + \boldsymbol{\nabla} p &&= -\rho \boldsymbol{\nabla} \Phi, \\
\partial_t (\rho E) &+ \rho E \mathrm{div} \boldsymbol{u} + \mathrm{div}\,(p\boldsymbol{u}) &&= -\rho \boldsymbol{u} \cdot \boldsymbol{\nabla} \Phi,
\end{aligned}$$

followed by the transport subsystem

$$\begin{aligned}
\partial_t \rho \quad &+ \boldsymbol{u} \cdot \boldsymbol{\nabla} \rho \quad &&= 0, \\
\partial_t (\rho \boldsymbol{u}) &+ \boldsymbol{u} \cdot \boldsymbol{\nabla} (\rho \boldsymbol{u}) &&= \mathbf{0}, \\
\partial_t (\rho E) &+ \boldsymbol{u} \cdot \boldsymbol{\nabla} (\rho E) &&= 0,
\end{aligned}$$

then we use the same techniques for the acoustic system as in 1.2.1, in other words the use of the mass variable and the Lagrangian variables. The acoustic system in these variables writes

$$\begin{aligned}
\partial_t \tau \; &- \partial_m u \quad &&= 0, \\
\partial_t u \; &+ \partial_m p \quad &&= -\frac{1}{\tau} \partial_m \Phi, \\
\partial_t v \quad & \quad &&= 0, \\
\partial_t E &+ \partial_m (pu) &&= -\frac{u}{\tau} \partial_m \Phi,
\end{aligned}$$

$$E = e + \frac{1}{2}(u^2 + v^2).$$

Using a pressure relaxation, an approximate Riemann solver with source term, see Gallice 2002, and the same upwind scheme for the transport system as in 1.2.1 we obtain the following non-conservative counterpart scheme

$$\begin{aligned}
\rho_i^{n+1} &= \rho_i^n - \frac{\Delta t}{\Delta x} [\widetilde{\rho} u^*]_i, \\
(\rho u)_i^{n+1} &= (\rho u)_i^n - \frac{\Delta t}{\Delta x} \left[\widetilde{(\rho u)} u^* + \Pi^* \right]_i + \frac{\Delta t}{\Delta x} S_i^n, \\
(\rho v)_i^{n+1} &= (\rho v)_i^n - \frac{\Delta t}{\Delta x} \left[\widetilde{(\rho v)} u^* \right]_i, \\
(\rho E)_i^{n+1} &= (\rho E)_i^n - \frac{\Delta t}{\Delta x} \left[\left(\widetilde{(\rho E)} + \Pi^* \right) u^* \right]_i + \frac{\Delta t}{\Delta x} (uS)_i^n,
\end{aligned}$$

where

$$\begin{aligned}
u_{i+1/2}^* &= \frac{1}{2}(u_{i+1}^n + u_i^n) - \frac{1}{2a} \left(\Pi_{i+1}^n - \Pi_i^n - S_{i+1/2}^n \right), \\
\Pi_{i+1/2}^* &= \frac{1}{2} \left(\Pi_{i+1}^n + \Pi_i^n \right) - \frac{a}{2} \left(u_{i+1}^n - u_i^n \right), \\
a_{i+1/2}^n &\geq \max \left(\rho_i^n c_i^n, \rho_{i+1}^n c_{i+1}^n \right), \\
S_i^n &= \frac{1}{2} \left(S_{i+1/2}^n + S_{i-1/2}^n \right), \\
(uS)_i^n &= \frac{1}{2} (u_{i+1/2}^* S_{i+1/2}^n + u_{i-1/2}^* S_{i-1/2}^n), \\
S_{i+1/2}^n &= -\frac{1}{2} \left(\frac{1}{\tau_i^n} + \frac{1}{\tau_{i+1}} \right) (\Phi_{i+1} - \Phi_i).
\end{aligned}$$

This scheme is not conservative for the whole energy but is closer to the scheme proposed for the shallow water equations in Chalons et al. 2017, for which the authors have obtained a discrete entropy inequality. It seems therefore possible to obtain a similar inequality for this non-conservative scheme, but this demonstration is beyond the scope of this paper.

1.C Generalized convection, diabatic convection

Convective instabilities such as standard adiabatic convection, moist convection or even double-diffusive convection are usually studied independently. In this section we present a unified derivation of the instability criteria by introducing a general source term on the energy equation of Navier-Stokes equations and a fraction that parameterizes the equation of state. As it has been presented in Tremblin et al. 2019, this allows to consider other convective instabilities such as the radiative convective instability in the CO/CH_4 transition in the atmosphere of brown dwarfs and exoplanets. Even though all these convective instabilities involve different mechanisms they have in common to either influence local density or temperature which are the key ingredients to convection movements. By opposition to adiabatic convection, we qualify the convection involving such mechanisms as diabatic convection. Omitting the momentum viscosity, the Navier-Stokes system writes

$$\begin{aligned}
&\partial_t \rho + \operatorname{div}(\rho \boldsymbol{u}) = 0,\\
&\partial_t (\rho \boldsymbol{u}) + \operatorname{div}(\rho \boldsymbol{u} \otimes \boldsymbol{u} + p\mathbb{I}) = \rho \boldsymbol{g},\\
&\partial_t (\rho \mathcal{E}) + \operatorname{div}((\rho \mathcal{E} + p)\boldsymbol{u}) = \rho c_p H(C, p, T),\\
&\partial_t (\rho C) + \operatorname{div}(\rho C \boldsymbol{u}) = \rho R(C, p, T),
\end{aligned} \tag{1.16}$$

$$p = p^{\text{eos}}(\rho, e, C). \tag{1.17}$$

in which H is a heat source and R is a composition source. Both source terms and the equation of state depend on a fraction C.

In Sections 1.C.1 and 1.C.2, we derive the Boussinesq system from a rescaled Navier-Stokes system, omitting the source terms H and R. This system consists in studying Navier-Stokes equations on a small length scale in which pressure variations are considered to be small thus removing pressure dependence on density variations. Then in Section 1.C.3 we derive the instability criteria which are obtained by studying stability of eigen modes after linearization of the Boussinesq model.

1.C.1 Non-dimensional form

Before deriving the Boussinesq system, we write the Euler equations in a non-dimensional form. We choose l_0, u_0, ρ_0, p_0, g_0 to be some reference quantities. For the reference speed of sound we choose $c_0^2 = \frac{p_0}{\rho_0}$. We also choose the internal energy as total energy reference $\rho_0 E_0 = \rho_0 e_0 = p_0$. By rescaling variables as $f = f_0 \bar{f}$, the Euler system writes

$$\begin{aligned}
\frac{\rho_0}{t_0}\partial_{\bar{t}}\bar{\rho} + \frac{\rho_0 u_0}{l_0}\operatorname{div}(\bar{\rho}\bar{\boldsymbol{u}}) &= 0\\
\frac{\rho_0 u_0}{t_0}\partial_{\bar{t}}(\bar{\rho}\bar{\boldsymbol{u}}) + \frac{1}{l_0}\operatorname{div}\left(\rho_0 u_0^2 \bar{\rho}\bar{\boldsymbol{u}} \otimes \bar{\boldsymbol{u}} + p_0 \bar{p}\mathbb{I}\right) &= \rho_0 g_0 \bar{\rho}\bar{\boldsymbol{g}}\\
\frac{\rho_0 E_0}{t_0}\partial_{\bar{t}}\left(\bar{\rho}\bar{E}\right) + \frac{1}{l_0}\operatorname{div}\left(\left(\rho_0 E_0 \bar{\rho}\bar{E} + p_0\bar{p}\right)u_0\bar{\boldsymbol{u}}\right) &= \rho_0 g_0 u_0 \bar{\rho}\bar{\boldsymbol{g}}\cdot\bar{\boldsymbol{u}}
\end{aligned}$$

This system can be rewritten as follows:

$$\begin{aligned}
\mathrm{Sr}\partial_{\bar{t}}\bar{\rho} + \mathrm{div}\left(\bar{\rho}\bar{\boldsymbol{u}}\right) &= 0 \\
\mathrm{Sr}\partial_{\bar{t}}\left(\bar{\rho}\bar{\boldsymbol{u}}\right) + \mathrm{div}\left(\bar{\rho}\bar{\boldsymbol{u}}\otimes\bar{\boldsymbol{u}} + \frac{1}{\mathrm{Ma}^2}\bar{p}\mathbb{I}\right) &= \frac{1}{\mathrm{Fr}^2}\bar{\rho}\bar{\boldsymbol{g}} \\
\mathrm{Sr}\partial_{\bar{t}}\left(\bar{\rho}\bar{E}\right) + \mathrm{div}\left(\left(\bar{\rho}\bar{E}+\bar{p}\right)\bar{\boldsymbol{u}}\right) &= \frac{\mathrm{Ma}^2}{\mathrm{Fr}^2}\bar{\rho}\bar{\boldsymbol{u}}\cdot\bar{\boldsymbol{g}} \\
\bar{\rho}\bar{E} &= \bar{\rho}\bar{e} + \mathrm{Ma}^2\frac{\bar{\rho}}{2}\|\bar{\boldsymbol{u}}\|^2
\end{aligned}$$

where we introduced the Strouhal number, the Mach number and the Froude number:

$$\begin{aligned}
\mathrm{Sr} &= \frac{l_0}{u_0 t_0} \\
\mathrm{Fr}^2 &= \frac{u_0^2}{g_0 l_0} = \frac{\rho_0 u_0^2}{\rho_0 g_0 l_0} = 2\frac{E_{\text{kinetic}}}{E_{\text{gravitational}}} \\
\mathrm{Ma}^2 &= \frac{u_0^2}{c_0^2} = \frac{u_0^2}{p_0/\rho_0} = \frac{\rho_0 u_0^2}{\rho_0 e_0} = 2\frac{E_{\text{kinetic}}}{E_{\text{internal}}}
\end{aligned}$$

that are respectively the ratio of kinetic energy over gravitational energy and the ratio of kinetic energy over internal energy. In the following we consider that $\mathrm{Sr} = 1$ and we omit the bar to simplify notations.

1.C.2 Boussinesq regime

In this section we derive the Boussinesq system from Euler system with gravity. We follow the approach in Mentrelli 2018; Feireisl and Novotný 2017 using an Hilbert expansion in terms of a non-dimensional small parameter ϵ. However we choose a different set of variables for the Hilbert expansion. Moreover we set $H = 0$ and $R = 0$.

We start the analysis from the rescaled Euler system

$$\begin{aligned}
\partial_t\rho + \mathrm{div}\left(\rho\boldsymbol{u}\right) &= 0 \\
\partial_t\left(\rho\boldsymbol{u}\right) + \mathrm{div}\left(\rho\boldsymbol{u}\otimes\boldsymbol{u} + \frac{1}{\mathrm{Ma}^2}p\mathbb{I}\right) &= \frac{1}{\mathrm{Fr}^2}\rho\boldsymbol{g} \\
\partial_t\left(\rho E\right) + \mathrm{div}\left(\boldsymbol{u}\left(\rho E + p\right)\right) &= \frac{\mathrm{Ma}^2}{\mathrm{Fr}^2}\rho\boldsymbol{g}\cdot\boldsymbol{u} \\
p &= p^{\text{eos}}\left(\rho, e\right) \\
\rho E = \rho e + \mathrm{Ma}^2\frac{\rho}{2}\|\boldsymbol{u}\|^2, &\quad \rho\mathcal{E} = \rho E + \frac{\mathrm{Ma}^2}{\mathrm{Fr}^2}\rho\phi
\end{aligned} \tag{1.18}$$

We are interested in the Boussinesq regime, which we define to be the limit of the latter system when $\mathrm{Ma} = \epsilon$, $\mathrm{Fr} = \sqrt{\epsilon}$ and $\mathrm{Bo} = \frac{\mathrm{Ma}}{\mathrm{Fr}} = \sqrt{\epsilon}$ for $\epsilon \to 0$. We refer the reader to Feireisl and Novotný 2017 for the scaling leading to the anelastic system. In order to derive the limit we use an Hilbert expansion, variables are then written as $f = \sum_n \epsilon^n f_n$. Because we are interested in smooth solutions, System 1.18 can be written in non-conservative variables $\boldsymbol{u}$, p and T

$$\begin{aligned}
\rho\left(\partial_t\boldsymbol{u} + \boldsymbol{u}\cdot\boldsymbol{\nabla}\boldsymbol{u}\right) + \frac{1}{\epsilon^2}\boldsymbol{\nabla}p &= \frac{1}{\epsilon}\rho\boldsymbol{g} \\
\partial_t p + \boldsymbol{u}\cdot\boldsymbol{\nabla}p + \rho c^2\mathrm{div}\boldsymbol{u} &= 0 \\
\rho c_p\left(\partial_t T + \boldsymbol{u}\cdot\boldsymbol{\nabla}T\right) &= \alpha T\left(\partial_t p + \boldsymbol{u}\cdot\boldsymbol{\nabla}p\right) \\
\rho &= \rho^{\text{eos}}\left(p, T\right)
\end{aligned} \tag{1.19}$$

in which $\alpha(p,T) = -\frac{1}{\rho}\partial_T \rho^{\text{eos}}(p,T)$ is the isobaric thermal expansion coefficient. We also introduce the isothermal compressibility coefficient $\chi(p,T) = \frac{1}{\rho}\partial_p \rho^{\text{eos}}(p,T)$. Temperature equation is obtained from entropy equation and using Maxwell relations. Hilbert expansion for orders $O\left(\epsilon^{-2}\right)$, $O\left(\epsilon^{-1}\right)$ and $O\left(\epsilon^{0}\right)$ reads

$$\begin{aligned}
\boldsymbol{\nabla} p_0 &= \mathbf{0} \\
\boldsymbol{\nabla} p_1 &= \rho_0 \boldsymbol{g} \\
\partial_t p_0 + \boldsymbol{u}_0 \cdot \boldsymbol{\nabla} p_0 + \rho_0 c_0^2 \text{div} \boldsymbol{u}_0 &= 0 \\
\rho_0 \left(\partial_t \boldsymbol{u}_0 + \boldsymbol{u}_0 \cdot \boldsymbol{\nabla} \boldsymbol{u}_0\right) + \boldsymbol{\nabla} p_2 &= \rho_1 \boldsymbol{g} \\
\rho_0 c_{p_0} \left(\partial_t T_0 + \boldsymbol{u}_0 \cdot \boldsymbol{\nabla} T_0\right) &= \alpha_0 T_0 \left(\partial_t p_0 + \boldsymbol{u}_0 \cdot \boldsymbol{\nabla} p_0\right) \\
\rho_0 &= \rho^{\text{eos}}(p_0, T_0)
\end{aligned} \tag{1.20}$$

First equation imposes uniform pressure $p_0(t,\boldsymbol{x}) = p_0(t)$, using compatible boundary conditions we also have a constant pressure $p_0(t) = p_0(0)$. From pressure equation, this imposes a divergence-free velocity field $\text{div}\boldsymbol{u}_0 = 0$. Hence, temperature equation reduces to a transport equation. Assuming uniform initial temperature profile and compatible boundary conditions we deduce that temperature is uniform in space and constant in time, $T_0(t,\boldsymbol{x}) = T_0(t) = T_0(0)$. Using equation of state, ρ^{eos} we also deduce that density is uniform and constant. Density ρ_1 is unknown, we need to go one order further, it writes

$$\begin{aligned}
\rho_0 c_{p_0} \left(\partial_t T_1 + \boldsymbol{u}_0 \cdot \boldsymbol{\nabla} T_1\right) &= \alpha_0 T_0 \left(\partial_t p_1 + \boldsymbol{u}_0 \cdot \boldsymbol{\nabla} p_1\right) \\
\rho_1 &= \partial_p \rho(p_0, T_0) p_1 + \partial_T \rho(p_0, T_0) T_1
\end{aligned} \tag{1.21}$$

Akin to p_0, assuming that p_1 is constant in time we obtain

$$\begin{aligned}
\rho_0 c_{p_0} \left(\partial_t T_1 + \boldsymbol{u}_0 \cdot \boldsymbol{\nabla} T_1\right) &= \alpha_0 T_0 \boldsymbol{u}_0 \cdot \boldsymbol{\nabla} p_1 \\
\rho_1 &= -\rho_0 \alpha_0 T_1 + \rho_0 \chi_0 p_1
\end{aligned} \tag{1.22}$$

We introduce the so-called adiabatic temperature gradient, in hydrostatic equilibrium

$$\boldsymbol{\nabla} T_{\text{ad}} = \frac{\alpha_0 T_0}{\rho_0 c_{p_0}} \boldsymbol{\nabla} p_1 = \frac{\alpha_0 T_0}{c_{p_0}} \boldsymbol{g} \tag{1.23}$$

Boussinesq system then writes

$$\begin{aligned}
\text{div} \boldsymbol{u}_0 &= 0 \\
\partial_t \boldsymbol{u}_0 + \boldsymbol{u}_0 \cdot \boldsymbol{\nabla} \boldsymbol{u}_0 + \frac{1}{\rho_0} \boldsymbol{\nabla} p_2 &= \frac{\rho_1}{\rho_0} \boldsymbol{g} \\
\partial_t T_1 + \left(\boldsymbol{\nabla} T_1 - \boldsymbol{\nabla} T_{\text{ad}}\right) \cdot \boldsymbol{u}_0 &= 0 \\
\rho_1 &= -\rho_0 \alpha_0 T_1 + \rho_0 \chi_0 p_1
\end{aligned} \tag{1.24}$$

It is standard to write the system in variables relative to the hydrostatic equilibrium, we emphasize that this is no more an approximation. Hydrostatic solution follows

$$\begin{aligned}
\boldsymbol{u}_0^h &= \mathbf{0} \\
\boldsymbol{\nabla} p_2^h &= \rho_1^h \boldsymbol{g} \\
\rho_1^h &= -\rho_0 \alpha_0 T_1^h + \rho_0 \chi_0 p_1
\end{aligned} \tag{1.25}$$

Noting $\tilde{f} = f - f^h$ variables relative to the hydrostatic equilibrium, Boussinesq system 1.24 also write

$$\begin{aligned} \mathrm{div}\boldsymbol{u}_0 &= 0 \\ \partial_t \boldsymbol{u}_0 + \boldsymbol{u}_0 \cdot \boldsymbol{\nabla} \boldsymbol{u}_0 + \frac{1}{\rho_0} \boldsymbol{\nabla} \tilde{p}_2 &= \tilde{\rho}_1 \boldsymbol{g} \\ \partial_t \tilde{T}_1 + \boldsymbol{u}_0 \cdot \boldsymbol{\nabla} \tilde{T}_1 + \left(\boldsymbol{\nabla} T_1^h - \boldsymbol{\nabla} T_{\mathrm{ad}} \right) \cdot \boldsymbol{u}_0 &= 0 \\ \tilde{\rho}_1 &= -\rho_0 \alpha_0 \tilde{T}_1 \end{aligned} \tag{1.26}$$

To summarize the expansion of variables we have

$$\begin{aligned} T = T_0 + \epsilon \left(T_1^h + \tilde{T}_1 \right) + \epsilon^2 T_2 + \ldots, \quad T_1 = T_1^h + \tilde{T}_1 \\ \rho = \rho_0 + \epsilon \left(\rho_1^h + \tilde{\rho}_1 \right) + \epsilon^2 \rho_2 + \ldots, \quad \rho_1 = \rho_1^h + \tilde{\rho}_1 \\ p = p_0 + \epsilon \left(p_1^h + 0 \right) + \epsilon^2 \left(p_2^h + \tilde{p}_2 \right) + \ldots, \quad p_1^h = p_1, \quad p_2 = p_2^h + \tilde{p}_2 \end{aligned} \tag{1.27}$$

1.C.3 Diabatic convection instability analysis

To fix ideas the equation of state is chosen to be an ideal gas for which mean molecular weight m depends on a fraction / concentration C

$$p = \rho \frac{k_B}{m\left(C\right)} T \tag{1.28}$$

As an example of source term in this section we consider thermal conduction given by Fourier's law $H\left(C,p,T\right) = H\left(T\right) = \sigma_T \Delta T$ and molecular diffusion given by Fick's law $R\left(C,p,T\right) = R\left(C\right) = \sigma_C \Delta C$. Both source terms have the same structure with σ_T being the thermal diffusion coefficient and σ_C the molecular diffusion coefficient. These source terms correspond to the double diffusive convection. Other source terms are developed in Tremblin et al. 2019.

Following notations introduced in Section 1.C.2, the Boussinesq system writes

$$\begin{aligned} \mathrm{div}\boldsymbol{u}_0 &= 0 \\ \partial_t \boldsymbol{u}_0 + \boldsymbol{u}_0 \cdot \boldsymbol{\nabla} \boldsymbol{u}_0 + \frac{1}{\rho_0} \boldsymbol{\nabla} \tilde{p}_2 &= \tilde{\rho}_1 \boldsymbol{g} \\ \partial_t \tilde{T}_1 + \boldsymbol{u}_0 \cdot \boldsymbol{\nabla} \tilde{T}_1 + \left(\boldsymbol{\nabla} T_1^h - \boldsymbol{\nabla} T_{\mathrm{ad}} \right) \cdot \boldsymbol{u}_0 &= H\left(C_1^h + \tilde{C}_1, p_0, T_1^h + \tilde{T}_1 \right) - H\left(C_1^h, p_0, T_1^h \right) \\ \partial_t \tilde{C}_1 + \boldsymbol{u}_0 \cdot \boldsymbol{\nabla} \tilde{C}_1 + \boldsymbol{\nabla} C_1^h \cdot \boldsymbol{u}_0 &= R\left(C_1^h + \tilde{C}_1, p_0, T_1^h + \tilde{T}_1 \right) - R\left(C_1^h, p_0, T_1^h \right) \\ \tilde{\rho}_1 &= -\rho_0 \alpha_0 \tilde{T}_1 + \rho_0 \beta_0 \tilde{C}_1 \end{aligned} \tag{1.29}$$

with $\beta_0 = \frac{1}{\rho_0} \partial_C \rho^{\mathrm{eos}}\left(C_0, p_0, T_0\right)$ and in which we neglect pressure variations of source terms H and R. By linearizing the system near equilibrium we have

$$\begin{aligned} \mathrm{div}\boldsymbol{u}_0 &= 0, \\ \partial_t \boldsymbol{u}_0 + \frac{1}{\rho_0} \boldsymbol{\nabla} \tilde{p}_2 &= \tilde{\rho}_1 \boldsymbol{g}, \\ \partial_t \tilde{T}_1 - w_0 \left(\nabla T_1^h - \nabla T_{\mathrm{ad}} \right) &= H_C \tilde{C}_1 + H_T \tilde{T}_1, \\ \partial_t \tilde{C}_1 - w_0 \nabla C_1^h &= R_C \tilde{C}_1 + R_T \tilde{T}_1, \\ \tilde{\rho}_1 &= -\rho_0 \alpha_0 \tilde{T}_1 + \rho_0 \beta_0 \tilde{C}_1. \end{aligned} \tag{1.30}$$

in which $\nabla T_1^h = -\boldsymbol{\nabla} T_1^h \cdot \boldsymbol{e_z}$, $\nabla T_{\mathrm{ad}}^h = -\boldsymbol{\nabla} T_{\mathrm{ad}}^h \cdot \boldsymbol{e_z}$ and $\nabla C_1^h = -\boldsymbol{\nabla} C_1^h \cdot \boldsymbol{e_z}$.

In order to study stability of perturbations, a standard method is to first derive dispersion relation by seeking for non-vanishing modes of the form $f(t,\boldsymbol{x}) = \hat{f}e^{\omega t + i\boldsymbol{k}\cdot\boldsymbol{x}}$ in a periodic box. Solutions of this form are said to be unstable if the real part of the dispersion relation is positive, i.e. $\Re(\omega) = \Re(\omega(\boldsymbol{k})) > 0$, the case $\Re(\omega) = 0$ defines the marginal stability (which defines the critical Rayleigh number in the adiabatic situation). It can be noticed that other boundary conditions in the z-direction generally lead to a differential system in variable z resulting in a shift of the marginal stability. Even though this analysis needs to be adapted to each situation, it gives qualitative results regarding physical mechanisms. It results in a linear system given by

$$\begin{aligned}
\boldsymbol{k}\cdot\hat{\boldsymbol{u}}_0 &= 0,\\
\rho_0\omega\hat{\boldsymbol{u}}_0 + i\boldsymbol{k}\hat{p}_2 &= \hat{\rho}_1\boldsymbol{g},\\
\omega\hat{T}_1 - \hat{w}_0\left(\nabla T_1^h - \nabla T_{\mathrm{ad}}^h\right) - \hat{H}_C\hat{C}_1 - \hat{H}_T\hat{T}_1 &= 0,\\
\omega\hat{C}_1 - \hat{w}_0\nabla C_1^h - \hat{R}_C\hat{C}_1 - \hat{R}_T\hat{T}_1 &= 0,\\
\tilde{\rho}_1 &= -\rho_0\alpha_0\tilde{T}_1 + \rho_0\beta_0\tilde{C}_1.
\end{aligned} \tag{1.31}$$

In the particular case of thermal conduction we have $\hat{H}_T\hat{T}_1 = -\sigma_T\boldsymbol{k}^2\hat{T}_1$ and for molecular diffusion $\hat{R}_C\hat{C}_1 = -\sigma_C\boldsymbol{k}^2\hat{C}_1$.

For a given wave-number $\boldsymbol{k}$, we look for non-zero solutions of the linear system 1.31 which are then characterized by a non-zero determinant of the associated matrix. The problem now reduces to find values for ω which both cancel out the determinant and have negative real part. It can be shown that the problem reduces to solving a third degree polynomial. Finally unstable solutions follow one of the two inequalities

$$\left(\nabla T_1^h - \nabla T_{\mathrm{ad}}^h\right)\alpha_0 - \nabla C_1^h\beta_0 > 0, \tag{1.32}$$

$$\left(\nabla T_1^h - \nabla T_{\mathrm{ad}}^h\right)\left(\alpha_0\hat{R}_C + \beta_0\hat{R}_T\right) - \nabla C_1^h\left(\beta_0\hat{H}_T + \alpha_0\hat{H}_C\right) < 0. \tag{1.33}$$

The first inequality 1.32 corresponds to the known Ledoux criterion for the convective region of stars. The second criterion involves the coupling of source terms and is referred as diabatic criterion. Again in the particular case of thermal conduction and molecular diffusion, inequality 1.33 reduces to the standard double-diffusive criterion 1.34

$$\left(\nabla T_1^h - \nabla T_{\mathrm{ad}}^h\right)\frac{\alpha_0}{\sigma_T} - \nabla C_1^h\frac{\beta_0}{\sigma_C} > 0. \tag{1.34}$$

In addition to double diffusion convection Equation 1.33 can also describe the moist convective instability in earth atmospheres when source terms are replaced by condensation/evaporation of liquid water. Note that this general theory has been derived thanks to the simulations that were performed with code ARK. Other applications such as the CO/CO_2 radiative convection in the atmosphere of hot rocky exoplanets are under development (Daley-yates et al in prep., see Figure 1.13). We also point out that the diabatic theory with condensation/evaporation of steam/liquid water could also be of interest for two-phase flow applications in nuclear engineering, the subject of the next chapter.

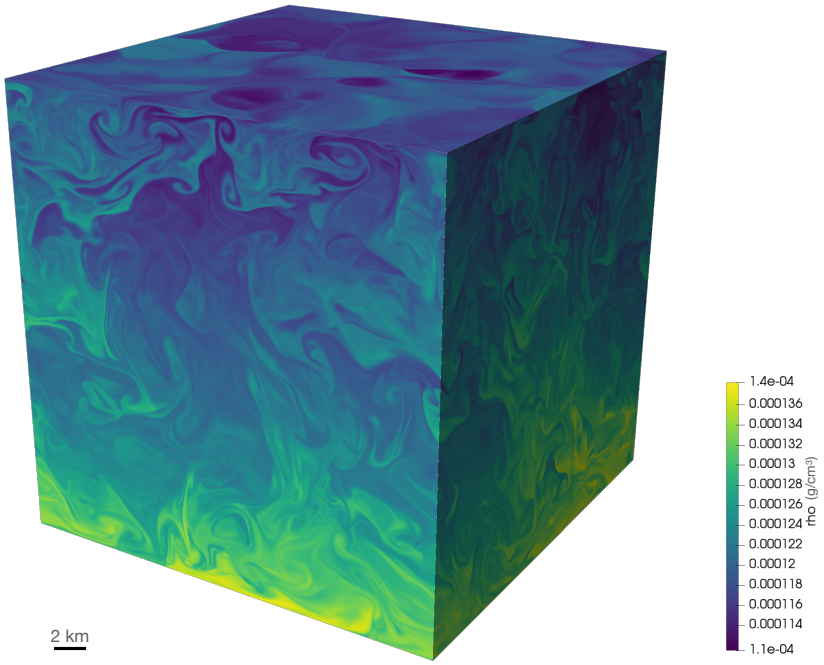


Figure 1.13: Simulation box of an atmosphere, composed of CO/CO_2 molecules, which is unstable to convection. The heating source term depends both on radiative energy and molecular composition; driving the convective instability. The colorbar represents the density.

Chapter 2

Droplet impact

Introduction

In the context of a nuclear reactor, the increase of the power leads to the decrease of the pressure in the secondary circuit and to a decrease of the vapor mass fraction down to 97 %. Liquid droplets can form from the steam matrix. The velocity of the droplets is estimated between $30\,\mathrm{m\,s^{-1}}$ and $100\,\mathrm{m\,s^{-1}}$. When impacting the pipes, for example during a change of direction, liquid droplets can damage them. An estimation of the pressure developed at the moment of the impact can help in the design of the circuitry to better cope with potential high local pressure values.

A few liquid droplet impact experiments have been realized, as for example in the work of Field et al. 1989. They could record snapshots of the density gradient during the impact of a high velocity two-dimensional droplet, see snapshots below 2.2. From the numerical point of view, some simulations investigated the liquid impacts. In the work of Wu et al. 2018, authors explore cylindrical droplets impacts at very high velocities, up to $200\,\mathrm{m\,s^{-1}}$. Three phases are involved, liquid water, steam water and air. They use a pressure threshold to trigger cavitation along with an instantaneous relaxation. They find good agreement in comparing Schlieren images. In Kyriazis et al. 2018, authors model the phases as a mixture at mechanical and thermal equilibrium. They have found a large cavitation area at the back of the droplet.

In this work we investigate droplet impacts by means of a Homogeneous Equilibrium Model. We begin by introducing this model and how we relate it to the five-equation model using a relaxation. The second section presents numerical discretization of these models. The third section presents the numerical validation of the model and numerical simulations of droplet impacts at high resolution.

2.1 Model

Before discussing models and discretization we first introduce necessary notations used in the following. Vectors are noted $\boldsymbol{v}$ and matrices $\mathbb{T}$, in particular identity matrix is noted $\mathbb{I}$.

We study a particular case of two-phase flows that involves water in two phases: liquid and gas. In the following we use as synonyms the terms gas, vapor and steam. We use the index α to generically refer to the phases: $\alpha = 0$ (resp. $\alpha = 1$) will refer to the gas (resp. liquid). Phasic quantities are indexed by α, for example ρ_α. We call pure state a state which is only composed of either gas or liquid and a mixture state otherwise.

Following notations from Chapter 1, each phase is characterized by a density $\rho_\alpha = \frac{1}{\tau_\alpha}$, a specific internal energy e_α and a velocity u_α. Moreover we note s_α and T_α, respectively, the specific phasic entropy and the phasic temperature. We assume that both phases, gas and liquid, are described with given equations of state $(\rho_\alpha, s_\alpha) \mapsto p_\alpha^{\text{eos}}(\rho_\alpha, s_\alpha)$. These equations of state are such that we can define a real valued speed of sound

$$ {c_\alpha}^2 = \frac{\partial p_\alpha^{\text{eos}}}{\partial \rho_\alpha}(\rho_\alpha, s_\alpha) > 0. \tag{2.1} $$

We also recall the definition of the phasic specific chemical potential

$$ g_\alpha = e_\alpha + \frac{p_\alpha}{\rho_\alpha} - T_\alpha s_\alpha. \tag{2.2} $$

2.1.1 Description of the mixture

As in the mono-fluid case, the mixture is characterized by a density ρ, a specific internal energy e, a velocity $\boldsymbol{u}$ and fractions that link the phasic quantities to the mixture quantities.

The volume fraction z_α is the volume occupied by one phase in a given volume. It is used to link the mixture densities and the specific energies to their phasic equivalents

$$ \begin{aligned} \rho &= \sum_\alpha \rho_\alpha z_\alpha, \\ \rho e &= \sum_\alpha \rho_\alpha z_\alpha e_\alpha. \end{aligned} \tag{2.3} $$

These two equations come from the extensivity properties of mass and energy. Furthermore we consider that the two phases are immiscible. This directly translates to

$$ \sum_\alpha z_\alpha = 1. \tag{2.4} $$

This allows us to note $z = z_1 = 1 - z_0$.

The mass fraction, y_α is the fraction of mass of one phase in a given volume, i.e.

$$ y_\alpha = \frac{\rho_\alpha}{\rho} z_\alpha. \tag{2.5} $$

By definition it satisfies $\sum_\alpha y_\alpha = 1$, thus we also note $y = y_1 = 1 - y_0$.

We choose a mixture equation of state of the form $p = p^{\text{eos}}(\rho, e, z, y)$ which depends on the two fractions z and y. In the following we consider the thermodynamics equilibrium equation of state and the isobaric equation of state.

Different definitions can be chosen for p that will characterize the mixture. There exists other closure models related to mixtures such as isothermal and isochoric, see Lagoutière 2000; Allaire et al. 2002; Barberon and Helluy 2005; Helluy and Seguin 2006.

2.1.2 Isobaric mixture, $p_{\text{isob}}^{\text{eos}}$

We consider an isobaric closure that simply writes

$$ p_{\text{isob}}^{\text{eos}}(\rho, e, z, y) = p_1^{\text{eos}}(\rho_1, e_1) = p_0^{\text{eos}}(\rho_0, e_0). \tag{2.6} $$

For a given tuple (ρ, e, z, y) and together with the specific internal energy relation $e = \sum_\alpha e_\alpha$, the pressure equality allows to give a value to phasic specific internal energies e_α, i.e. to the phasic temperatures T_α.

Remark 1 (Domain of validity) *It follows that the liquid pressure must remain positive because of the gas phase. Then this doesn't allow to take into account meta-stable states of liquid like stretching that can lead to negative pressures. However, depending on phasic equations of state, this formula may allow to derive an effective equation of state with a larger domain of validity as it is the case with stiffened gases.*

2.1.3 Thermodynamics equilibrium mixture, $p_{\text{eq}}^{\text{eos}}$

Following the lines from Helluy and Seguin 2006; Barberon and Helluy 2005; Faccanoni et al. 2008, we distinguish two cases

- the equilibrium that corresponds to a pure state,
- the equilibrium that corresponds to a mixture state.

Pure states

In the first case, the equilibrium equation of state reduces to the phasic equation of state, that is to say

$$p_{\text{isob}}^{\text{eos}}(\rho, e, z, y) = \begin{cases} {p^{\text{eos}}}_1(\rho, e), & \text{if } z_1 = y_1 = 1, \\ {p^{\text{eos}}}_0(\rho, e), & \text{if } z_0 = y_0 = 1. \end{cases} \tag{2.7}$$

Mixture states

In the second case, we consider that the two phases have the same pressure, temperature and chemical potential that is to say

$$\begin{aligned} \tau &= (1-y)\,\tau_0 + y\tau_1, \\ e &= (1-y)\,e_0 + ye_1, \\ p_0^{\text{eos}}(\rho_0, e_0) &= p_1^{\text{eos}}(\rho_1, e_1), \\ T_0^{\text{eos}}(\rho_0, e_0) &= T_1^{\text{eos}}(\rho_1, e_1), \\ g_0^{\text{eos}}(p_0, T_0) &= g_1^{\text{eos}}(p_1, T_1). \end{aligned} \tag{2.8}$$

The equality of chemical potentials allows to define a saturation curve $T_{\text{sat}}(p)$. By inverting the system of Equations 2.8 both in τ and e, it writes

$$y = \frac{\tau - \tau_0}{\tau_1 - \tau_0} = \frac{e - e_0}{e_1 - e_0}. \tag{2.9}$$

Along with the saturation curve, the pressure satisfies the non-linear equation

$$\begin{gathered} \frac{\tau - \tau_0^{\text{eos}}(p,T)}{\tau_1^{\text{eos}}(p,T) - \tau_0^{\text{eos}}(p,T)} - \frac{e - e_0^{\text{eos}}(p,T)}{e_1^{\text{eos}}(p,T) - e_0^{\text{eos}}(p,T)} = 0, \\ T = T_{sat}(p). \end{gathered} \tag{2.10}$$

Knowing the equilibrium pressure p we can then recover all necessary thermodynamics quantities

at equilibrium

$$\begin{aligned} T_{\text{eq}} &= T_{sat}\left(p\right), \\ \left(\tau_\alpha\right)_{\text{eq}} &= \tau_\alpha^{\text{eos}}\left(p,T\right), \\ \left(e_\alpha\right)_{\text{eq}} &= \tau_\alpha^{\text{eos}}\left(p,T\right), \\ y_{\text{eq}} &= \frac{\tau-\tau_0}{\tau_1-\tau_0}, \\ z_{\text{eq}} &= \frac{\tau_1}{\tau} y. \end{aligned} \tag{2.11}$$

Finally, we notice that the equation of state at thermodynamics equilibrium is only determined by the mixture density ρ and the specific mixture energy e. Fractions $y = y_{\text{eq}}(\rho, e)$ and $z = z_{\text{eq}}(\rho, e)$ can be deduced from the equilibrium. This is similar to having only one equation of state used for both phases, as in IAPWS (Wagner and Pruß 2002) equation of state, see De Lorenzo et al. 2017 for this equation of state formulated in the (ρ, e) plane.

Remark 2 *IAPWS-95, which stands for International Association for the Properties of Water and Steam, is a standard that formulates properties of water using a serie expansion of the Helmholtz free energy in terms of variables (ρ, T). This serie involves up to 64 terms which are obtained from experimental measures. Other thermodynamics quantities can be recovered from the Helmholtz free energy.*

2.1.4 The Homogeneous Equations Model

The velocity of the water liquid droplet in the pipe is estimated between 30 and 100 m/s. Because of this high velocity, we have to take into account the compressibility of liquid water. We choose an Homogeneous Equations Model, also shortly referred as HEM, that assumes that both phases are at mechanical and thermodynamics equilibrium at each time t. This translates to a velocity equilibrium and to the usage of the thermodynamics equilibrium equation of state

$$\boldsymbol{u} = \boldsymbol{u}_1 = \boldsymbol{u}_0, \quad p = p_{\text{eq}}^{\text{eos}}(\rho, e). \tag{2.12}$$

This model simply takes the form of a single fluid Euler equations

$$\begin{aligned} \partial_t \rho + \operatorname{div}\left(\rho \boldsymbol{u}\right) &= 0, \\ \partial_t\left(\rho \boldsymbol{u}\right) + \operatorname{div}\left(\rho \boldsymbol{u}\boldsymbol{u} + p_{\text{eq}}^{\text{eos}} \mathbb{I}\right) &= \boldsymbol{0}, \\ \partial_t\left(\rho E\right) + \operatorname{div}\left(\rho E + p_{\text{eq}}^{\text{eos}}\right)\boldsymbol{u} &= 0, \\ \rho E = \rho e + \frac{1}{2}\rho \boldsymbol{u}^2, y = y_{\text{eq}}\left(\rho, e\right), z &= z_{\text{eq}}\left(\rho, e\right), \end{aligned} \tag{2.13}$$

that expresses the conservation of mixture quantities: the mixture mass ρ, the mixture momentum $\rho\boldsymbol{u}$ and the mixture total energy ρE. The pressure $p_{\text{eq}}^{\text{eos}} = p_{\text{eq}}^{\text{eos}}(\rho, e)$ is the thermodynamics equilibrium pressure between liquid and gas. We can distinguish two cases in this model, pure phase regions (for which mass fraction is exclusively 0 or 1) and mixture regions.

First in the pure phase case, then Equations 2.13 locally degenerate to the Euler system with $p = p_\alpha^{\text{eos}}(\rho, e)$, in which mixture variables degenerate to phasic variables. The speed of sound also reduces to the phasic speed of sound c_α. On the other hand, in the mixture case, mass transfer between the two phases occurs as mass fraction instantaneously adapts to local variations in density and internal energy. As already mentioned time evolution of the mass fraction solely depends on the time evolution of mass and internal energy equations, formally

$$\partial_t y_{\text{eq}}\left(\rho, e\right) = \partial_\rho y_{\text{eq}}\left(\rho, e\right)\partial_t \rho + \partial_e y_{\text{eq}}\left(\rho, e\right)\partial_t e. \tag{2.14}$$

This allows to potentially create new phase inside region of pure phase, in case of strong pressure drop for instance.

Remark 3 *The kinematic properties of a mixture lead to different physics. For example the Baer and Nunziato model (for example see Baer and Nunziato 1986) allows to model a sub-scale velocity drift between the two phases.*

Remark 4 *In terms of mass transfer we can distinguish models in which it occurs at the interface between two fluids or using a local relaxation, possibly instantaneous, to thermodynamics equilibrium.*

In the first case, mass transfer takes the form of a source term at the interface. Velocity and pressure at interface are not anymore continuous, see Fechter et al. 2017. The interface is moving at its own velocity that is determined by jump relations. The difficulty related to such models is the generation of new interfaces.

In the second case, mass transfer takes the form of a local source term. One assumes that out of equilibrium, the evolution is relaxing towards thermodynamics equilibrium at some rate λ, which may be instantaneous. The difficulty with such model is the choice of the relaxation parameter. The particular choice of $\lambda = \infty$ falls back to using models known as Homogeneous Equations Models, HEM.

2.1.5 The relaxed five-equation model

In order to solve the system of Equations 2.13 we notice that we can formally rewrite the system of Equations 2.13, in the limit of $\lambda \to \infty$, with the following system

$$\begin{aligned}
\partial_t z + \boldsymbol{u} \cdot \boldsymbol{\nabla} z &= \lambda \left(z_{\text{eq}} - z \right), \\
\partial_t \left(\rho y \right) + \operatorname{div} \left(\rho y \boldsymbol{u} \right) &= \lambda \rho \left(y_{\text{eq}} - y \right), \\
\partial_t \rho + \operatorname{div} \left(\rho \boldsymbol{u} \right) &= 0, \\
\partial_t \left(\rho \boldsymbol{u} \right) + \operatorname{div} \left(\rho \boldsymbol{u}\boldsymbol{u} + p^{\text{eos}}_{\text{isob}} \mathbb{I} \right) &= \boldsymbol{0}, \\
\partial_t \left(\rho E \right) + \operatorname{div} \left(\rho E + p^{\text{eos}}_{\text{isob}} \right) \boldsymbol{u} &= 0, \\
\rho E = \rho e + \frac{1}{2} \rho \boldsymbol{u}^2 &,
\end{aligned} \tag{2.15}$$

in which we introduce two new equations on the volume fraction z and the mass fraction y and the isobaric mixture equation of state $p^{\text{eos}}_{\text{isob}}$.

On the one hand, when $\lambda \to \infty$, we formally recover the former HEM system with $p^{\text{eos}}_{\text{isob}} \to p^{\text{eos}}_{\text{eq}}$ along with the following compatibility relation

$$p^{\text{eos}}_{\text{isob}}(\rho, e, z_{\text{eq}}(\rho, e), y_{\text{eq}}(\rho, e)) = p^{\text{eos}}_{\text{eq}}(\rho, e). \tag{2.16}$$

On the other hand, when $\lambda = 0$, the model reduces to the following five-equation model

$$\begin{aligned}
\partial_t z + \boldsymbol{u} \cdot \boldsymbol{\nabla} z &= 0, \\
\partial_t \left(\rho y \right) + \operatorname{div} \left(\rho y \boldsymbol{u} \right) &= 0, \\
\partial_t \rho + \operatorname{div} \left(\rho \boldsymbol{u} \right) &= 0, \\
\partial_t \left(\rho \boldsymbol{u} \right) + \operatorname{div} \left(\rho \boldsymbol{u}\boldsymbol{u} + p^{\text{eos}}_{\text{isob}} \mathbb{I} \right) &= \boldsymbol{0}, \\
\partial_t \left(\rho E \right) + \operatorname{div} \left(\rho E + p^{\text{eos}}_{\text{isob}} \right) \boldsymbol{u} &= 0, \\
\rho E = \rho e + \frac{1}{2} \rho \boldsymbol{u}^2 &.
\end{aligned} \tag{2.17}$$

It is an hyperbolic system with five wavespeeds given by, in the direction $\boldsymbol{n}$,

$$\{\boldsymbol{u}\cdot\boldsymbol{n} - c^{\text{eos}}_{\text{isob}}, c^{\text{eos}}_{\text{isob}}, c^{\text{eos}}_{\text{isob}}, c^{\text{eos}}_{\text{isob}}, \boldsymbol{u}\cdot\boldsymbol{n} + c^{\text{eos}}_{\text{isob}}\}\,, \tag{2.18}$$

with $c^{\text{eos}}_{\text{isob}}$ satisfying

$$\xi_\alpha = \rho_\alpha(\partial_{p_\alpha} e^{\text{eos}}_\alpha)_{\rho_\alpha}, \quad \xi = \sum_\alpha z_\alpha \xi_\alpha, \quad \rho\xi(c^{\text{eos}}_{\text{isob}})^2 = \sum_\alpha \rho_\alpha z_\alpha \xi_\alpha (c^{\text{eos}}_\alpha)^2. \tag{2.19}$$

The relaxation 2.15 is motivated by numerical reasons. Indeed as it is presented in Kokh and Lagoutière 2010, along with an anti-diffusive numerical scheme and an isobaric equation of state, the five-equation model is able to transport interfaces with low numerical diffusion without generating spurious pressure oscillations. This property is important to avoid smearing of the initial interface between the liquid droplet and the steam matrix, see also Section 2.A for a connection between interface models and mixture models.

Other relaxations can be considered as in the work of Barberon and Helluy 2005; Helluy and Seguin 2006 in which they derive relaxation models based on entropy considerations. These models allow to consider non-instantaneous relaxation.

2.2 Discretization

We start this section by introducing some notations required for the discretization. Let $\Omega \subset \mathbb{R}^3$ be a Cartesian domain

$$\Omega = [x_{\min}, x_{\max}] \times [y_{\min}, y_{\max}] \times [z_{\min}, z_{\max}].$$

It is discretized using a Cartesian regular mesh. We define by Δx (resp. Δy and Δz) the step along the X-direction (resp. the Y and Z-direction). In the X-direction (resp. Y, Z-direction) interfaces are located at $x_{i+1/2} = (i+1)\Delta x$ (resp. $y_{j+1/2} = (j+1)\Delta y$, $z_{k+1/2} = (k+1)\Delta z$). Time is discretized into ordered instants t^n, separated by intervals Δt and we identify t^n and n.

Let $\phi\colon (\boldsymbol{x}, t) \mapsto \phi(\boldsymbol{x}, t)$ be some scalar field, we note $\phi^n_{i,j,k}$ the Finite Volume approximation of ϕ in a given cell (i,j,k) at instant t^n. Finally we note $\phi^n_{i+1/2,j,k}$ (resp. $\phi^n_{i,j+1/2,k}$, $\phi^n_{i,j,k+1/2}$) the quantity associated to the field ϕ at the time t^n and at the interface between cells (i,j,k) and $(i+1,j,k)$ (resp. (i,j,k) and $(i,j+1,k)$, (i,j,k) and $(i,j,k+1)$).

As presented above we use a two-step procedure to discretize 2.15

- Step 1 discretizes hydrodynamic evolution. The fluid parameter ϕ will update from $\phi^n_{i,j,k}$ to $\bar{\phi}_{i,j,k}$. Anticipating further notations we introduce sub-steps.
  - Sub-step a, an acoustic sub-step that will update quantities from $\phi^n_{i,j,k}$ to $\tilde{\phi}_{i,j,k}$.
  - Sub-step b, a transport sub-step that will update quantities from $\tilde{\phi}_{i,j,k}$ to $\bar{\phi}_{i,j,k}$.
- Step 2 deals with phase change using instantaneous relaxation. The fluid parameter ϕ will evolve from $\bar{\phi}_{i,j,k}$ to $\phi^{n+1}_{i,j,k}$.

2.2.1 Step 1, hydrodynamic evolution

We begin by recalling the system that is discretized

$$\begin{aligned}
\partial_t z + \boldsymbol{u}\cdot\boldsymbol{\nabla} z = 0,\\
\partial_t(\rho y) + \mathrm{div}(\rho y \boldsymbol{u}) = 0,\\
\partial_t \rho + \mathrm{div}(\rho \boldsymbol{u}) = 0,\\
\partial_t(\rho \boldsymbol{u}) + \mathrm{div}(\rho \boldsymbol{u}\boldsymbol{u} + p^{\mathrm{eos}}_{\mathrm{isob}}\mathbb{I}) = \boldsymbol{0},\\
\partial_t(\rho E) + \mathrm{div}((\rho E + p^{\mathrm{eos}}_{\mathrm{isob}})\,\boldsymbol{u}) = 0.
\end{aligned} \tag{2.20}$$

The discretization of this system is also handled using a two-step procedure by splitting acoustic and transport phenomena leading to two systems.

Remark 5 *This operator splitting procedure is similar to the discretization used in Chapter 1. It involves separation of acoustic and transport waves. In Chapter 1 special treatment in the acoustic step was needed to have a well-balanced numerical for hydrostatic equilibrium. Here we need to pay attention to the transport equation which can lead to numerical, undesired, mixture of the two phases.*

We first consider an acoustic system that reads

$$\begin{aligned}
\partial_t z = 0,\\
\partial_t y = 0,\\
\partial_t \rho + \rho \mathrm{div}\boldsymbol{u} = 0,\\
\partial_t(\rho \boldsymbol{u}) + \rho\boldsymbol{u}\mathrm{div}\boldsymbol{u} + \boldsymbol{\nabla} p^{\mathrm{eos}}_{\mathrm{isob}} = \boldsymbol{0},\\
\partial_t(\rho E) + \rho E \mathrm{div}\boldsymbol{u} + \mathrm{div}(p^{\mathrm{eos}}_{\mathrm{isob}}\boldsymbol{u}) = 0,
\end{aligned} \tag{2.21}$$

and the transport system writes

$$\begin{aligned}
\partial_t z + \boldsymbol{u}\cdot\boldsymbol{\nabla} z = 0,\\
\partial_t(\rho y) + \boldsymbol{u}\cdot\boldsymbol{\nabla}(\rho y) = 0,\\
\partial_t \rho + \boldsymbol{u}\cdot\boldsymbol{\nabla}\rho = 0,\\
\partial_t(\rho \boldsymbol{u}) + \boldsymbol{u}\cdot\boldsymbol{\nabla}(\rho \boldsymbol{u}) = \boldsymbol{0},\\
\partial_t(\rho E) + \boldsymbol{u}\cdot\boldsymbol{\nabla}(\rho E) = 0.
\end{aligned} \tag{2.22}$$

In the transport system we can see that each conservative variable is advected at material velocity $\boldsymbol{u}$. This operator splitting procedure allows to treat independently each system.

Sub-step a, acoustic

In order to discretize the system of Equations 2.21 we propose to use a relaxation for the acoustic system that is reminiscent of the Lagrangian Gas dynamics equations (see Després 2010). To do so, we first rewrite the system using the variables $\tau = \frac{1}{\rho}$, $\boldsymbol{u}$ and E. Then we proceed to a linearization of the equations and then discretize them.

It then writes

$$\begin{aligned}
\partial_t \boldsymbol{V} + \tau \mathrm{div}(\boldsymbol{F}(\boldsymbol{V})) = 0,\\
\boldsymbol{V} = (z, y, \tau, \boldsymbol{u}, E)^T, \quad \boldsymbol{F}(\boldsymbol{V}) = \begin{bmatrix} \boldsymbol{0}^T \\ \boldsymbol{0}^T \\ -\boldsymbol{u}^T \\ p^{\mathrm{eos}}_{\mathrm{isob}}\mathbb{I} \\ p^{\mathrm{eos}}_{\mathrm{isob}}\boldsymbol{u}^T \end{bmatrix}.
\end{aligned} \tag{2.23}$$

We use a relaxation based on ideas introduced by Bouchut 2004 for the Euler system that extends the Suliciu-based relaxation (see Chalons and Coulombel 2008; Chalons and Coquel 2014; Suliciu 1998). This relaxation allows to remove some non-linearities due to the equation of state. The relaxation introduces two variables Π and a which are approximations of the pressure $p_{\text{isob}}^{\text{eos}}$ and the Lagrangian speed of sound $\rho c_{\text{isob}}^{\text{eos}}$, respectively. The augmented acoustic system writes

$$\begin{gathered}\partial_t \boldsymbol{W} + \tau \operatorname{div}\left(\boldsymbol{\mathcal{F}}\left(\boldsymbol{W}\right)\right) = 0,\\ \boldsymbol{W} = \left(z, y, \tau, \boldsymbol{u}, E, a, \frac{\Pi}{a^2}\right)^T, \quad \boldsymbol{\mathcal{F}}\left(\boldsymbol{W}\right) = \begin{bmatrix} \boldsymbol{0}^T \\ \boldsymbol{0}^T \\ -\boldsymbol{u}^T \\ \Pi\mathbb{I} \\ \Pi\boldsymbol{u}^T \\ \boldsymbol{0}^T \\ \boldsymbol{u}^T \end{bmatrix}.\end{gathered} \tag{2.24}$$

The connection with respect to the equation of state is restored at each time step by initializing $\Pi = p_{\text{isob}}^{\text{eos}}\left(\rho, e, z, y\right)$ and $a = \rho c_{\text{isob}}^{\text{eos}}\left(\rho, e, z, y\right)$. We recover the initial variable V by a projection $\mathcal{P}$ from the relaxation variables

$$\boldsymbol{\mathcal{P}} : \boldsymbol{W} \to \boldsymbol{V} = \left(z, y, \tau, \boldsymbol{u}, E\right)^T. \tag{2.25}$$

This system is close to have a conservative form, up to the variable τ in front of the divergence. Then we freeze this variable at discrete time t^n, which can also be interpreted as a relaxation procedure, it then becomes

$$\partial_t \boldsymbol{W} + \tau^n\left(\boldsymbol{x}\right) \operatorname{div}\left(\boldsymbol{\mathcal{F}}\left(\boldsymbol{W}\right)\right) = 0. \tag{2.26}$$

This system is then discretized using a Finite Volume method

$$\begin{aligned}\tilde{\boldsymbol{W}}_{i,j,k} = \boldsymbol{W}_{i,j,k}^n &- \tau_{i,j,k}^n \frac{\Delta t}{\Delta x}\left(\boldsymbol{\mathcal{F}}_{i+1/2,j,k} - \boldsymbol{\mathcal{F}}_{i-1/2,j,k}\right)\\ &- \tau_{i,j,k}^n \frac{\Delta t}{\Delta y}\left(\boldsymbol{\mathcal{F}}_{i,j+1/2,k} - \boldsymbol{\mathcal{F}}_{i,j-1/2,k}\right)\\ &- \tau_{i,j,k}^n \frac{\Delta t}{\Delta z}\left(\boldsymbol{\mathcal{F}}_{i,j,k+1/2} - \boldsymbol{\mathcal{F}}_{i,j,k-1/2}\right),\end{aligned} \tag{2.27}$$

in which we have $\boldsymbol{\mathcal{F}}_{i+1/2,j,k} = \boldsymbol{\mathcal{F}}_{\Delta}\left(\boldsymbol{W}_{i,j,k}^n, \boldsymbol{W}_{i+1,j,k}^n; \boldsymbol{e}_x\right)$ and for $\boldsymbol{e} \in \{\boldsymbol{e}_x, \boldsymbol{e}_y, \boldsymbol{e}_z\}$,

$$\begin{aligned}\boldsymbol{\mathcal{F}}_{\Delta}\left(\boldsymbol{W}_L, \boldsymbol{W}_R; \boldsymbol{e}\right) &= \left(0, 0, -u^*, \Pi^*\boldsymbol{e}, \Pi^* u^*, 0, u^*\right)^T,\\ u^* &= \frac{a_L \boldsymbol{u}_L \cdot \boldsymbol{e} + a_R \boldsymbol{u}_R \cdot \boldsymbol{e}}{a_L + a_R} - \frac{\Pi_R - \Pi_L}{a_L + a_R},\\ \Pi^* &= \frac{a_R \Pi_L + a_L \Pi_R}{a_L + a_R} - \frac{a_L a_R\left(\boldsymbol{u}_R \cdot \boldsymbol{e} - \boldsymbol{u}_L \cdot \boldsymbol{e}\right)}{a_L + a_R},\end{aligned} \tag{2.28}$$

where the velocity waves a_L, a_R are given by

$$\left.\begin{aligned}a_L &= \rho_L\left(c_L + \tfrac{1}{2}\left(1 + \max\left(\gamma_0, \gamma_1\right)\right)\left(\tfrac{\Pi_R - \Pi_L}{\rho_R c_R} + \boldsymbol{u}_L \cdot \boldsymbol{e} - \boldsymbol{u}_R \cdot \boldsymbol{e}\right)^+\right)\\ a_R &= \rho_R\left(c_R + \tfrac{1}{2}\left(1 + \max\left(\gamma_0, \gamma_1\right)\right)\left(\tfrac{\Pi_L - \Pi_R}{a_L} + \boldsymbol{u}_L \cdot \boldsymbol{e} - \boldsymbol{u}_R \cdot \boldsymbol{e}\right)^+\right)\end{aligned}\right\} \text{ if } \Pi_R > \Pi_L, \tag{2.29}$$

and

$$\left.\begin{aligned} a_R &= \rho_R \left(c_R + \tfrac{1}{2}\left(1+\max\left(\gamma_0,\gamma_1\right)\right)\left(\tfrac{\Pi_L-\Pi_R}{\rho_L c_L} + \boldsymbol{u}_L\cdot\boldsymbol{e} - \boldsymbol{u}_R\cdot\boldsymbol{e}\right)^+ \right) \\ a_L &= \rho_L \left(c_L + \tfrac{1}{2}\left(1+\max\left(\gamma_0,\gamma_1\right)\right)\left(\tfrac{\Pi_L-\Pi_R}{a_R} + \boldsymbol{u}_L\cdot\boldsymbol{e} - \boldsymbol{u}_R\cdot\boldsymbol{e}\right)^+ \right) \end{aligned}\right\} \text{ if } \Pi_R \leq \Pi_L. \quad (2.30)$$

Remark 6 *This scheme falls back to a regular Suliciu-type relaxation when $a_L = a_R = a$, indeed we get*

$$\begin{aligned} u^* &= \frac{1}{2}\left(\boldsymbol{u}_L\cdot\boldsymbol{e} + \boldsymbol{u}_R\cdot\boldsymbol{e}\right) - \frac{1}{2a}\left(\Pi_R - \Pi_L\right), \\ \Pi^* &= \frac{1}{2}\left(\Pi_L + \Pi_R\right) - \frac{a}{2}\left(\boldsymbol{u}_R\cdot\boldsymbol{e} - \boldsymbol{u}_L\cdot\boldsymbol{e}\right). \end{aligned} \quad (2.31)$$

Thus a low-Mach correction can also be used to reduce numerical diffusion in the pressure Π^*, see Chapter 1. We have noticed that using both a low-Mach correction and a directional splitting leads to an unstable numerical scheme.

After relaxation towards equilibrium, it simply results in

$$\tilde{\boldsymbol{V}}_{i,j,k} = \boldsymbol{\mathcal{P}}\left(\tilde{\boldsymbol{W}}_{i,j,k}\right). \quad (2.32)$$

It can be noticed that the numerical scheme also writes

$$\begin{aligned} L_{i,j,k}\tilde{\rho}_{i,j,k}\tilde{\boldsymbol{V}}_{i,j,k} = \rho^n_{i,j,k}\boldsymbol{V}^n_{i,j,k} &- \frac{\Delta t}{\Delta x}\left(\boldsymbol{\mathcal{P}}\left(\boldsymbol{\mathcal{F}}_{i+1/2,j,k}\right) - \boldsymbol{\mathcal{P}}\left(\boldsymbol{\mathcal{F}}_{i-1/2,j,k}\right)\right) \\ &- \frac{\Delta t}{\Delta y}\left(\boldsymbol{\mathcal{P}}\left(\boldsymbol{\mathcal{F}}_{i,j+1/2,k}\right) - \boldsymbol{\mathcal{P}}\left(\boldsymbol{\mathcal{F}}_{i,j-1/2,k}\right)\right) \\ &- \frac{\Delta t}{\Delta z}\left(\boldsymbol{\mathcal{P}}\left(\boldsymbol{\mathcal{F}}_{i,j,k+1/2}\right) - \boldsymbol{\mathcal{P}}\left(\boldsymbol{\mathcal{F}}_{i,j,k-1/2}\right)\right), \end{aligned} \quad (2.33)$$

with

$$\begin{aligned} L_{i,j,k} = 1 &+ \frac{\Delta t}{\Delta x}\left(u^*_{i+1/2,j,k} - u^*_{i-1/2,j,k}\right) \\ &+ \frac{\Delta t}{\Delta y}\left(u^*_{i,j+1/2,k} - u^*_{i,j-1/2,k}\right) \\ &+ \frac{\Delta t}{\Delta z}\left(u^*_{i,j,k+1/2} - u^*_{i,j,k-1/2}\right). \end{aligned} \quad (2.34)$$

Up to the compression factor $L_{i,j,k}$, the variable $\tilde{\rho}_{i,j,k}\tilde{\boldsymbol{V}}_{i,j,k}$ admits a quasi-conservative update that will be used in the next section.

Sub-step b, transport

We first recall the equation we want to discretize, for $\phi \in \{z, \rho y, \rho, \rho\boldsymbol{u}, \rho E\}$

$$\partial_t\phi + \boldsymbol{u}\cdot\boldsymbol{\nabla}\phi = 0. \quad (2.35)$$

To take advantage of the first acoustic step, this equation is simply rewritten as

$$\partial_t\phi + \mathrm{div}\left(\phi\boldsymbol{u}\right) - \phi\mathrm{div}\boldsymbol{u} = 0, \quad (2.36)$$

and is partially discretized as follows

$$\begin{aligned}\bar{\phi}_{i,j,k} = \tilde{\phi}_{i,j,k} &- \frac{\Delta t}{\Delta x}\left(\phi^*_{i+1/2,j,k}u^*_{i+1/2,j,k} - \phi^*_{i-1/2,j,k}u^*_{i-1/2,j,k}\right)\\ &- \frac{\Delta t}{\Delta y}\left(\phi^*_{i,j+1/2,k}u^*_{i,j+1/2,k} - \phi^*_{i,j-1/2,k}u^*_{i,j-1/2,k}\right)\\ &- \frac{\Delta t}{\Delta z}\left(\phi^*_{i,j,k+1/2}u^*_{i,j,k+1/2} - \phi^*_{i,j,k-1/2}u^*_{i,j,k-1/2}\right)\\ &+ \frac{\Delta t}{\Delta x}\tilde{\phi}_{i,j,k}\left(u^*_{i+1/2,j,k} - u^*_{i-1/2,j,k}\right)\\ &+ \frac{\Delta t}{\Delta y}\tilde{\phi}_{i,j,k}\left(u^*_{i,j+1/2,k} - u^*_{i,j-1/2,k}\right)\\ &+ \frac{\Delta t}{\Delta z}\tilde{\phi}_{i,j,k}\left(u^*_{i,j,k+1/2} - u^*_{i,j,k-1/2}\right),\end{aligned} \tag{2.37}$$

for which u^* has been defined in Equations 2.24. Using the factor $L_{i,j,k}$, it equivalently writes

$$\begin{aligned}\bar{\phi}_{i,j,k} = L_{i,j,k}\tilde{\phi}_{i,j,k} &- \frac{\Delta t}{\Delta x}\left(\phi^*_{i+1/2,j,k}u^*_{i+1/2,j,k} - \phi^*_{i-1/2,j,k}u^*_{i-1/2,j,k}\right)\\ &- \frac{\Delta t}{\Delta y}\left(\phi^*_{i,j+1/2,k}u^*_{i,j+1/2,k} - \phi^*_{i,j-1/2,k}u^*_{i,j-1/2,k}\right)\\ &- \frac{\Delta t}{\Delta z}\left(\phi^*_{i,j,k+1/2}u^*_{i,j,k+1/2} - \phi^*_{i,j,k-1/2}u^*_{i,j,k-1/2}\right)\end{aligned} \tag{2.38}$$

In this way, the discretization is conservative with respect to variables $\rho y, \rho, \rho\boldsymbol{u}, \rho E$. It remains to define the flux ϕ^* to have a well-defined numerical scheme.

We want to minimize numerical diffusion in the case of an interface. Different numerical techniques can be used to define the flux ϕ^*. A first approach would be to use a standard upwind scheme. It is able to transport accurately interfaces when time step is adapted to the transport velocity and with a CFL condition exactly to 1. This constraint is too strong as we wish to treat shocks, it is thus not possible to use such a scheme. Hence we choose to use an anti-diffusive approach, originally developed in Després and Lagoutière 2001 and adapted to the five-equation model in Kokh and Lagoutière 2010. The main idea of the latter scheme is to be the most downwind possible but while ensuring stability. We start by defining fluxes relative to the volume fraction z^* using standard upwind scheme relative to u^* from the acoustic step

$$\begin{aligned}u^*_{i+1/2,j,k} &= \begin{cases}\tilde{u}_{i,j,k} \text{ if } u^*_{i+1/2,j,k} > 0,\\ \tilde{u}_{i+1,j,k} \text{ otherwise.}\end{cases}\\ (\rho_\alpha)^*_{i+1/2,j,k} &= \begin{cases}(\tilde{\rho}_\alpha)_{i,j,k} \text{ if } u^*_{i+1/2,j,k} > 0,\\ (\tilde{\rho}_\alpha)_{i+1,j,k} \text{ otherwise.}\end{cases}\\ (e_\alpha)^*_{i+1/2,j,k} &= \begin{cases}(\tilde{e}_\alpha)_{i,j,k} \text{ if } u^*_{i+1/2,j,k} > 0,\\ (\tilde{e}_\alpha)_{i+1,j,k} \text{ otherwise.}\end{cases}\\ (\rho y)^*_{i+1/2,j,k} &= (\rho_1)^*_{i+1/2,j,k}z^*_{i+1/2,j,k},\\ \rho^*_{i+1/2,j,k} &= z^*_{i+1/2,j,k}(\rho_1)^*_{i+1/2,j,k} + (1 - z^*_{i+1/2,j,k})(\rho_0)^*_{i+1/2,j,k}.\end{aligned} \tag{2.39}$$

Thus, these fluxes are consistent regarded that the flux $z^*_{i+1/2,j,k}$ is consistent. Then it suffices to give $z^*_{i+1/2,j,k}$ a definition. As it is detailed in Appendix 2.B and in Kokh and Lagoutière 2010, we express both consistency and stability in one direction to define this latter flux. To treat the

multi-dimensional case we can either apply a directional splitting or use a mean decomposition as in Peluchon 2017. Multi-dimensional update can be rewritten as

$$\bar{z}_{i,j,k} = \frac{1}{3}\left(\bar{z}^x_{i,j,k} + \bar{z}^y_{i,j,k} + \bar{z}^z_{i,j,k}\right), \tag{2.40}$$

in which $\bar{z}^x_{i,j,k}$, $\bar{z}^y_{i,j,k}$, $\bar{z}^z_{i,j,k}$ are one dimensional updates along the X, Y and Z directions, respectively. As an example, developing $\bar{z}^x_{i,j,k}$ gives

$$\begin{aligned}\bar{z}^x_{i,j,k} = \tilde{z}^x_{i,j,k} &- 3\frac{\Delta t}{\Delta x}\left(z^x_{i+1/2,j,k}u^*_{i+1/2,j,k} - z^x_{i-1/2,j,k}u^*_{i-1/2,j,k}\right)\\ &+ 3\frac{\Delta t}{\Delta x}\tilde{z}^x_{i,j,k}\left(u^*_{i+1/2,j,k} - u^*_{i-1/2,j,k}\right).\end{aligned} \tag{2.41}$$

Hence a sufficient condition for stability and consistency for update 2.40 is that each individual update is stable and consistent. It results that, using similar stability and consistency developments as in Appendix 2.B, we adapt the trust intervals such that, along the X-direction it gives

$$\begin{aligned}
&z^*_{i+1/2,j,k} = \mathrm{P}_{\tilde{I}_{i,j,k}}\left(\tilde{z}_{i_\rightleftarrows,j,k}\right)\\
&\tilde{I}_{i,j,k} = \left[\tilde{m}_{i+1/2,j,k}, \tilde{M}_{i+1/2,j,k}\right] \cap \left[\tilde{a}_{i+1/2,j,k}, \tilde{A}_{i+1/2,j,k}\right],\\
&\tilde{a}_{i+1/2,j,k} = \tilde{z}_{i_\rightleftarrows,j,k} + \left(\tilde{M}_{(i+1/2)_\rightleftarrows,j,k} - \tilde{z}_{i_\rightleftarrows,j,k}\right)\left(\frac{u^*_{(i+1/2)_\rightleftarrows,j,k}}{u^*_{i+1/2,j,k}} - \frac{\Delta x}{3\Delta t}\frac{1}{u^*_{i+1/2,j,k}}\right),\\
&\tilde{A}_{i+1/2,j,k} = \tilde{z}_{i_\rightleftarrows,j,k} + \left(\tilde{m}_{(i+1/2)_\rightleftarrows,j,k} - \tilde{z}_{i_\rightleftarrows,j,k}\right)\left(\frac{u^*_{(i+1/2)_\rightleftarrows,j,k}}{u^*_{i+1/2,j,k}} - \frac{\Delta x}{3\Delta t}\frac{1}{u^*_{i+1/2,j,k}}\right).
\end{aligned} \tag{2.42}$$

This final volume fraction flux is used in fluxes for other variables.

2.2.2 Step 2, thermodynamics relaxation

Finally the last step concerns relaxation to thermodynamics equilibrium. We recall that the equations formally write, in the limit $\lambda \to +\infty$,

$$\begin{aligned}
\partial_t z &= \lambda\left(z_{eq} - z\right),\\
\partial_t y &= \lambda\left(y_{eq} - y\right),\\
\partial_t \rho &= 0,\\
\partial_t\left(\rho \boldsymbol{u}\right) &= 0,\\
\partial_t\left(\rho E\right) &= 0.
\end{aligned} \tag{2.43}$$

Numerical discretization is straightforward and only consists in setting

$$\begin{aligned}
z^{n+1}_{i,j,k} &= \left(z_{eq}\right)_{i,j,k},\\
y^{n+1}_{i,j,k} &= \left(y_{eq}\right)_{i,j,k},\\
\rho^{n+1}_{i,j,k} &= \widetilde{\rho}_{i,j,k},\\
\left(\rho\boldsymbol{u}\right)^{n+1}_{i,j,k} &= \left(\widetilde{\rho\boldsymbol{u}}\right)_{i,j,k},\\
\left(\rho E\right)^{n+1}_{i,j,k} &= \left(\widetilde{\rho E}\right)_{i,j,k}.
\end{aligned} \tag{2.44}$$

We need to find $(z_{eq})_{i,j,k}$ and $(y_{eq})_{i,j,k}$ knowing $\bar{\rho}_{i,j,k}$ and $\bar{e}_{i,j,k}$. As the process is local to each cell (i,j,k), in the following we omit indices to simplify notations. We solve the non-linear Equations 2.10 in pressure. Knowing p_{eq} we can then recover all necessary thermodynamics quantities at equilibrium

$$\begin{aligned} T_{eq} &= T_{sat}\left(p_{eq}\right), \\ \left(\tau_\alpha\right)_{eq} &= \tau_\alpha\left(p_{eq}, T_{eq}\right), \\ \left(e_\alpha\right)_{eq} &= \tau_\alpha\left(p_{eq}, T_{eq}\right), \\ y_{eq} &= \frac{\bar{\tau} - \left(\tau_0\right)_{eq}}{\left(\tau_1\right)_{eq} - \left(\tau_0\right)_{eq}}, \\ z_{eq} &= \frac{\left(\tau_1\right)_{eq} y_{eq}}{\bar{\tau}}. \end{aligned} \tag{2.45}$$

For a given couple $(\bar{\tau}, \bar{e})$, if Equations 2.10 have a solution then it is a mixture state, otherwise it is a pure state. As there is no analytical expression for the function T_{sat}, it needs to be discretized.

Saturation curve $T_{\text{sat}}\left(p\right)$

Both phases have their own equation of state. At thermodynamics equilibrium they obey to

$$g_0\left(p_{eq}, T_{eq}\right) = g_1\left(p_{eq}, T_{eq}\right), \tag{2.46}$$

By enforcing equality between the two fluids, one is able to invert the relation

$$\forall p \in \left[p_{\min}, p_{\max}\right], \text{find } T_{\text{sat}} \text{ such that} \quad g_0\left(p, T_{\text{sat}}\right) = g_1\left(p, T_{\text{sat}}\right), \tag{2.47}$$

We use this non-linear equation to build a polynomial interpolation curve

$$T_{\text{sat}}\left(p\right) = \sum_i^n a_i (\log\left(p\right))^i. \tag{2.48}$$

Coefficients are computed using the numpy.polyfit function.

2.2.3 Overall scheme

We summarize the numerical scheme in this section

$$\begin{aligned} z_{i,j,k}^{n+1} &= \left(z_{eq}\right)_{i,j,k}, \\ y_{i,j,k}^{n+1} &= \left(y_{eq}\right)_{i,j,k}, \end{aligned}$$

$$\begin{aligned} \rho_{i,j,k}^{n+1} = \rho_{i,j,k}^{n} &- \frac{\Delta t}{\Delta x}\left(\rho_{i+1/2,j,k}^{*} u_{i+1/2,j,k}^{*} - \rho_{i-1/2,j,k}^{*} u_{i-1/2,j,k}^{*}\right) \\ &- \frac{\Delta t}{\Delta y}\left(\rho_{i,j+1/2,k}^{*} u_{i,j+1/2,k}^{*} - \rho_{i,j-1/2,k}^{*} u_{i,j-1/2,k}^{*}\right) \\ &- \frac{\Delta t}{\Delta z}\left(\rho_{i,j,k+1/2}^{*} u_{i,j,k+1/2}^{*} - \rho_{i,j,k-1/2}^{*} u_{i,j,k-1/2}^{*}\right), \end{aligned}$$

$$
\begin{aligned}
(\rho \boldsymbol{u})_{i,j,k}^{n+1} = (\rho \boldsymbol{u})_{i,j,k}^{n} &- \frac{\Delta t}{\Delta x}\left(\rho^*_{i+1/2,j,k}\boldsymbol{u}^*_{i+1/2,j,k}u^*_{i+1/2,j,k} - \rho^*_{i-1/2,j,k}\boldsymbol{u}^*_{i-1/2,j,k}u^*_{i-1/2,j,k}\right) \\
&- \frac{\Delta t}{\Delta y}\left(\rho^*_{i,j+1/2,k}\boldsymbol{u}^*_{i,j+1/2,k}u^*_{i,j+1/2,k} - \rho^*_{i,j-1/2,k}\boldsymbol{u}^*_{i,j-1/2,k}u^*_{i,j-1/2,k}\right) \\
&- \frac{\Delta t}{\Delta z}\left(\rho^*_{i,j,k+1/2}\boldsymbol{u}^*_{i,j,k+1/2}u^*_{i,j,k+1/2} - \rho^*_{i,j,k-1/2}\boldsymbol{u}^*_{i,j,k-1/2}u^*_{i,j,k-1/2}\right) \\
&- \frac{\Delta t}{\Delta x}\left(\Pi^*_{i+1/2,j,k} - \Pi^*_{i-1/2,j,k}\right) \\
&- \frac{\Delta t}{\Delta y}\left(\Pi^*_{i,j+1/2,k} - \Pi^*_{i,j-1/2,k}\right) \\
&- \frac{\Delta t}{\Delta z}\left(\Pi^*_{i,j,k+1/2} - \Pi^*_{i,j,k-1/2}\right), \\
(\rho E)_{i,j,k}^{n+1} = (\rho E)_{i,j,k}^{n} &- \frac{\Delta t}{\Delta x}\left(\rho^*_{i+1/2,j,k}E^*_{i+1/2,j,k}u^*_{i+1/2,j,k} - \rho^*_{i-1/2,j,k}E^*_{i-1/2,j,k}u^*_{i-1/2,j,k}\right) \\
&- \frac{\Delta t}{\Delta y}\left(\rho^*_{i,j+1/2,k}E^*_{i,j+1/2,k}u^*_{i,j+1/2,k} - \rho^*_{i,j-1/2,k}E^*_{i,j-1/2,k}u^*_{i,j-1/2,k}\right) \\
&- \frac{\Delta t}{\Delta z}\left(\rho^*_{i,j,k+1/2}E^*_{i,j,k+1/2}u^*_{i,j,k+1/2} - \rho^*_{i,j,k-1/2}E^*_{i,j,k-1/2}u^*_{i,j,k-1/2}\right) \\
&- \frac{\Delta t}{\Delta x}\left(\Pi^*_{i+1/2,j,k}u^*_{i+1/2,j,k} - \Pi^*_{i-1/2,j,k}u^*_{i-1/2,j,k}\right) \\
&- \frac{\Delta t}{\Delta y}\left(\Pi^*_{i,j+1/2,k}u^*_{i,j+1/2,k} - \Pi^*_{i,j-1/2,k}u^*_{i,j-1/2,k}\right) \\
&- \frac{\Delta t}{\Delta z}\left(\Pi^*_{i,j,k+1/2}u^*_{i,j,k+1/2} - \Pi^*_{i,j,k-1/2}u^*_{i,j,k-1/2}\right),
\end{aligned}
$$

in which star quantities are defined in Sections 2.2.1 and 2.2.1. We notice that, as expected, the final form of the scheme is conservative for mixture density, momentum and total energy. Volume and mass fractions are at equilibrium even though they temporary evolved out of equilibrium during acoustic and transport steps.

In the case where the thermodynamics relaxation is omitted, the numerical scheme simply falls back to the discretization of the five-equation model using anti-diffusive scheme.

2.3 Numerical experiments

2.3.1 Global setup

We use stiffened gases as equations of state for both phases. These equations of state write

$$
\begin{aligned}
p_\alpha^{\mathrm{eos}}\left(\rho_\alpha, e_\alpha\right) &= \left(\gamma_\alpha - 1\right)\rho_\alpha\left(e_\alpha - e_{0\alpha}\right) - \gamma_\alpha \Pi_\alpha, \\
T_\alpha^{\mathrm{eos}}\left(\rho_\alpha, p_\alpha\right) &= \frac{p_\alpha + \Pi_\alpha}{\rho_\alpha c_{v\alpha}\left(\gamma_\alpha - 1\right)}, \\
s_\alpha^{\mathrm{eos}}\left(\rho_\alpha, e_\alpha\right) &= c_{v\alpha}\log\left(e_\alpha - e_{0\alpha} - \frac{\Pi_\alpha}{\rho_\alpha}\right) - c_{v\alpha}\left(\gamma_\alpha - 1\right)\log\left(\rho_\alpha\right) + s_{0\alpha}, \\
c_\alpha^2\left(\rho_\alpha, p_\alpha\right) &= \gamma_\alpha \frac{p_\alpha + \Pi_\alpha}{\rho_\alpha}.
\end{aligned}
\tag{2.49}
$$

The pressure shift Π_α of the stiffened gas can be interpreted as the macroscopic manifestation of attractive interactions between water molecules (see Le Métayer and Saurel 2016). This term is responsible for negative pressures, even though the speed of sound remains positive.

Coefficient	Value
a_0	$-3.504\,131\,855\,669\,312\,8 \times 10^{1}$
a_1	$3.148\,542\,674\,925\,358\,8 \times 10^{2}$
a_2	$-1.441\,955\,094\,577\,394\,3 \times 10^{2}$
a_3	$3.537\,935\,461\,268\,914\,8 \times 10^{1}$
a_4	$-4.215\,560\,359\,123\,924\,3$
a_5	$2.064\,843\,331\,029\,337\,4 \times 10^{-1}$

Table 2.1: Parameters used for droplet simulations. These are interpolation coefficients for the saturation curve.

Phase	Gas ($\alpha = 0$)	Liquid ($\alpha = 1$)
γ_α	1.135020250284312	1.549523809523810
$c_{v\alpha}\,(\mathrm{J\,K^{-1}\,kg^{-1}})$	2.803×10^{3}	3.15×10^{3}
$p_{0_\alpha}\,(\mathrm{Pa})$	0	7.18×10^{8}
$e_{0_\alpha}\,(\mathrm{J\,kg^{-1}})$	$9.991\,579\,270\,858\,164 \times 10^{5}$	-1.4665×10^{6}
$s_{0_\alpha}\,(\mathrm{J\,K^{-1}\,kg^{-1}})$	$-2.104\,807\,279\,208\,952 \times 10^{3}$	0

Table 2.2: Parameters used for droplet simulations, taken from Hurisse 2017.

This setup uses the values in Table 2.2 for the thermodynamics, which are taken from Hurisse 2017 (Table B.3), along with values from Table 2.1 for the interpolation of the saturation curve.

In order to avoid spurious phase transition in the bulk of the liquid droplet after the first time step we need to pay attention to initialize it under the saturation curve. The liquid droplet size is 1 mm, the density is initialized at $828.34\,\mathrm{kg/m^3}$ and the pressure at 30 bar. Steam is initialized at $15.76\,\mathrm{kg/m^3}$. Both phases are initialized at $50\,\mathrm{m\,s^{-1}}$.

2.4 Droplet test case, 2D

We start with 2D simulations to provide qualitative comparison with similar experiments from Field et al. 1989. In this experiment, the droplet size is 10 mm, the initial velocity is at $110\,\mathrm{m\,s^{-1}}$ and at ambient pressure, see Figure 2.2 for the snapshots from this experiment. We realized a two-dimensional simulation, see 2.1.

Even though the experiment is led in different conditions from the nuclear reactor conditions, we recover the main stages. A shock wave propagates from the point of impact towards the back of the droplet. It reflects on the interface, inside the droplet, and gets focused, we refer the reader to Wu et al. 2018 for a detailed analysis on the wave propagation inside the droplet.

In our simulation the liquid jets at the wall are less developed because the droplet velocity is lower. Starting from time $t = 3 \times 10^{-6}\,\mathrm{s}$ we notice the development of cavitation regions ($z \neq 0, 1$), which do not seem to be present on the experiment 2.2, at least on such long time scale. The cavitation region is in agreement with results from Kyriazis et al. 2018.

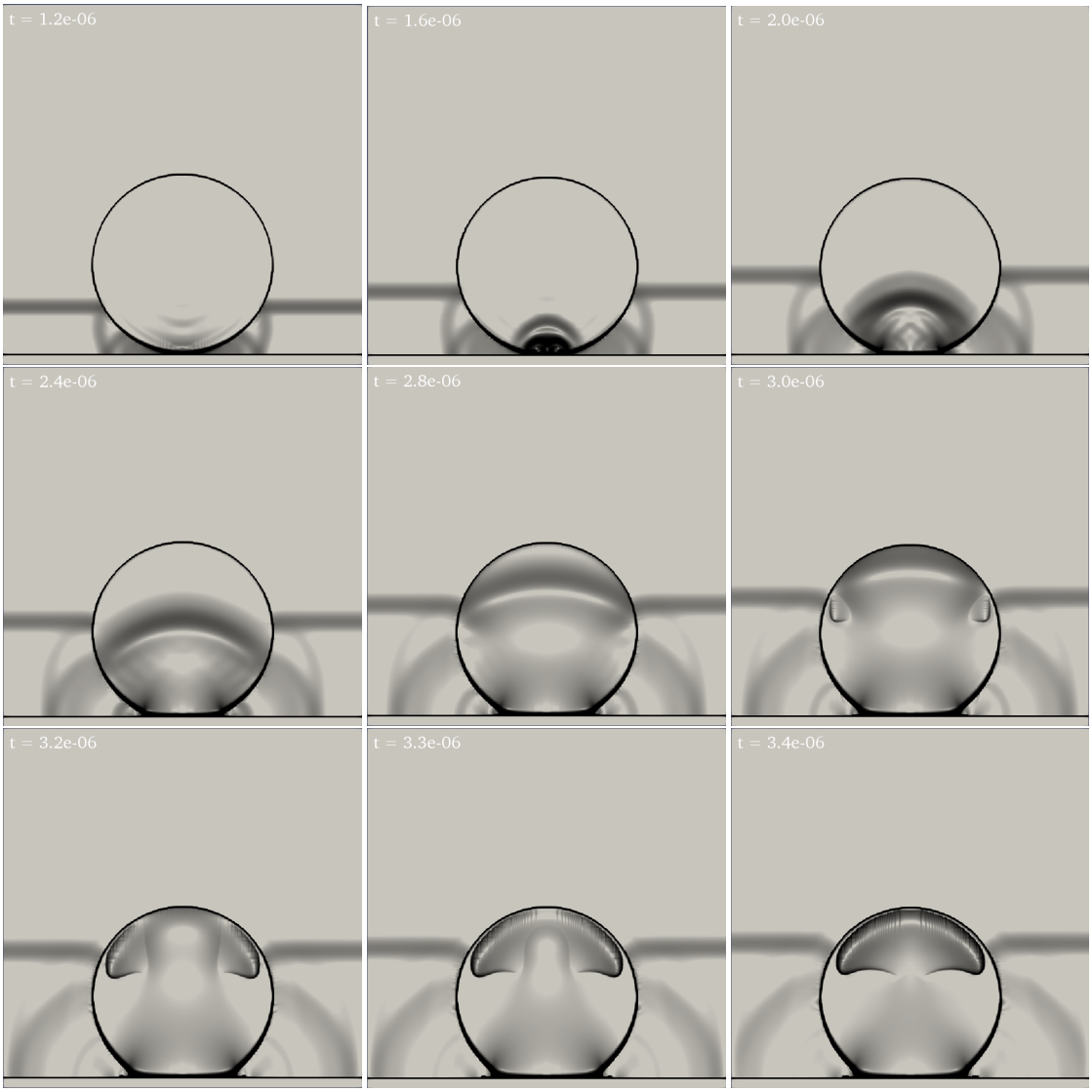


Figure 2.1: Time sequence of numerical Schlieren images from a 256^2 simulation droplet impact onto a plane wall. Time is in s.

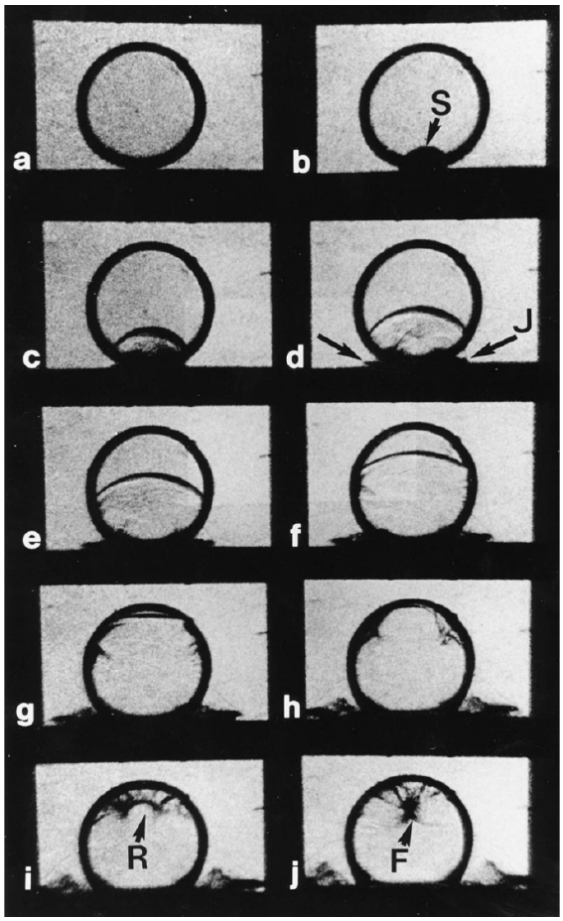


Figure 2.2: Time sequence of density contrast from Field et al. 1989. The liquid droplet size is 10 mm at ambient pressure. At point F, cavitation occurs.

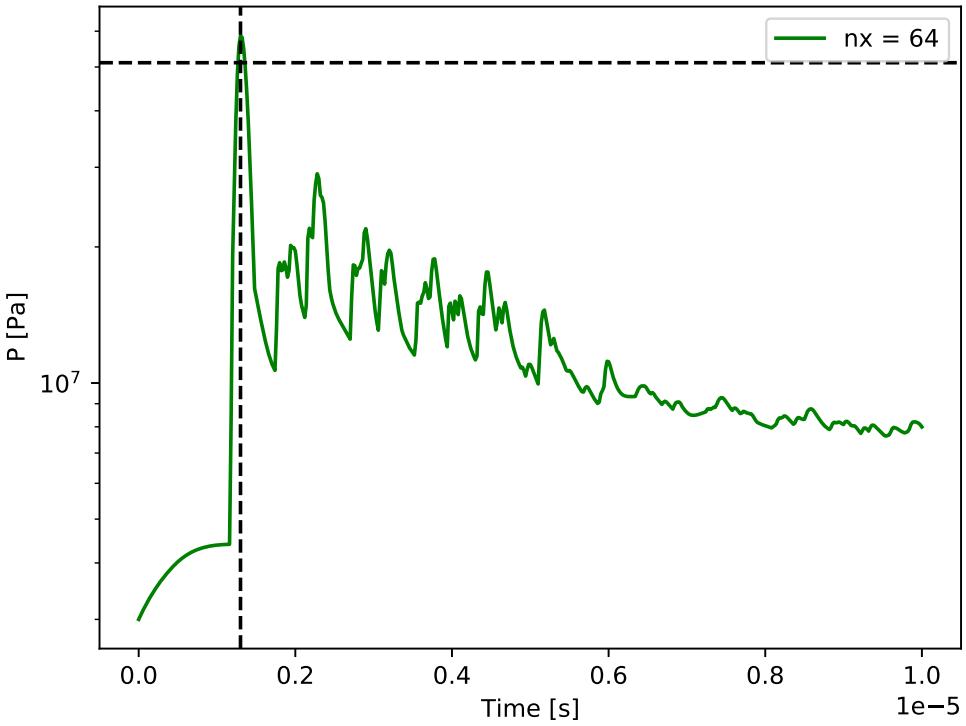


Figure 2.3: The figure represents the time evolution of the maximum pressure at the wall for a simulation at coarse resolution 64^3. The black vertical line indicates the impact of the droplet whereas the horizontal line indicates the water hammer pressure.

2.5 Droplet test case, 3D

2.5.1 Ideal wall

We begin with a slippery plane wall, we consider it ideal. Figure 2.3 represents the time evolution of the maximum pressure at the position of the wall. We observe a peak pressure that develops when the droplet impacts the wall. The maximum pressure is $p_{\max} \approx 586\,\text{bar}$. This pressure can be compared to the analytical water hammer pressure (see Ghidaoui et al. 2005, for a review)

$$\Delta p = \rho_1 c_1 \Delta u. \tag{2.50}$$

In our configuration we have $\rho_1 = 828.34\,\text{kg/m}^3$, $c_1 \approx 1161\,\text{m s}^{-1}$ and $\Delta u = u_1 = 50\,\text{m s}^{-1}$. It results in a water hammer pressure $p_{\text{hammer}} \approx 30 + 481 = 511\,\text{bar}$. Thus the numerical measured value presents a pressure excess of 15% compared to the water hammer pressure. After the impact, the pressure globally decreases with time. However we notice oscillations right after the initial peak, they seem to be related to cells undergoing phase transition at the interface between liquid and vapor and in contact with the wall. Thus, these local peaks may not be physically relevant as such, and a mean profile should be considered.

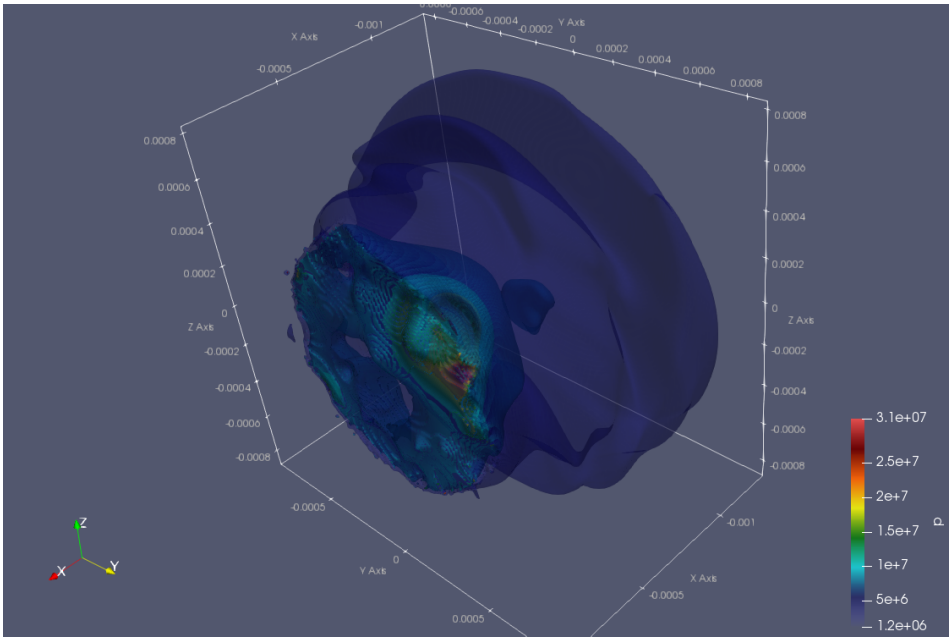


Figure 2.4: Droplet impact at resolution 2048^3. It is represented contour surfaces of the pressure field inside the liquid droplet. We see a peak pressure localized on a Gaussian perturbation.

2.5.2 Perturbation of the wall

In this section we explore the impact of a droplet on a perturbed wall. This non-ideal wall represents a damaged wall because of, for instance, previous droplet impacts that have formed cavities. We choose to perturbe the wall using 2D Gaussian obstacles, the wall position then writes

$$x_{\text{wall}}(y,z) = x_0 + \sum_{i=1}^{i=N} a_i e^{-\frac{1}{2\sigma_i^2}\left((y-y_i)^2+(z-z_i)^2\right)}. \tag{2.51}$$

Coefficients a_i, σ_i, y_i and z_i are randomly chosen and represent respectively the amplitude, the width and the position each Gaussian perturbation. Figure 2.5 represents the wall used for the following simulations.

2.5.3 Numerical results

As part of a Grand Challenge on the AMD partition of the Joliot-Curie supercomputer at TGCC, Saclay, we have realized two simulations at resolutions 1024^3 and 2048^3, for a total of 13 million CPU hours. Figure 2.4 shows isopressure surfaces after the droplet has impacted the wall. We can see pressure waves in the liquid droplet propagating from the wall towards the droplet's end. Moreover we see a peak pressure which is developed inside a local cavity.

Interested in the pressure developed near the wall, we have extracted the pressure field on the first cell layer after the wall. Thus the maximum pressure evolution at the wall is shown in Figure 2.6 for the two simulations. Both time series exhibit high frequencies in pressure. They can be related to a threshold effect due to the phase transition and the first order discretization of the wall. Indeed, there is no hysteresis in the phase transition, thus if a fluid particle is in a state close to saturation it can oscillate between the two phases and produce pressure oscillations.

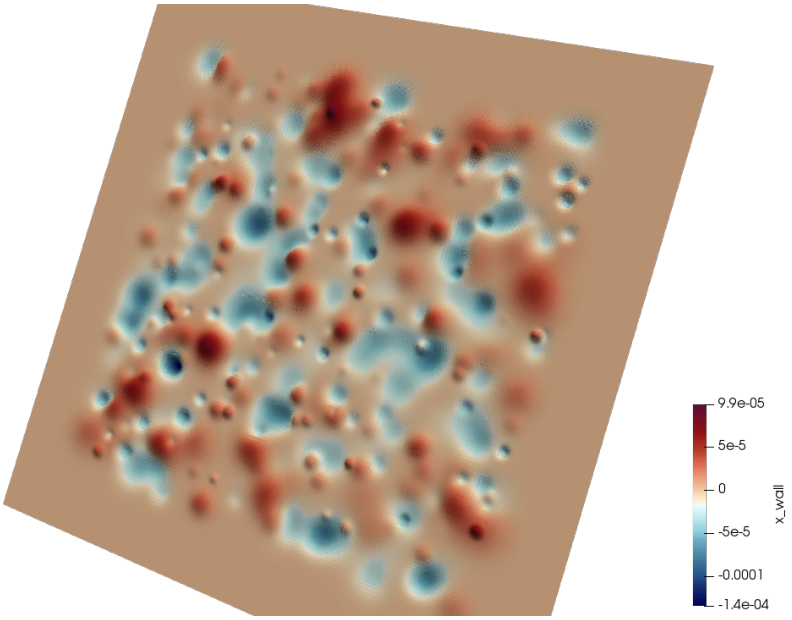


Figure 2.5: This represents the perturbed wall. Color shows the height of Gaussian perturbations. Perturbations are removed near borders of the box.

Thus we remove high frequencies by computing the moving average over time steps of previous pressure evolution, results are shown on Figure 2.7.

As we can see on Figure 2.6, the simulation at resolution 2048^3 does not have higher pressures than simulation at lower resolution. In term of peak pressure we can assume that convergence is achieved.

On both figures, we distinguish two stages. In a first stage we see an increase in pressure up until the droplet impacts the wall, represented by the black vertical line. After the impact we see a global decrease of the maximum pressure to which is superposed rapid peaks. These peaks are due to the inhomogeneities in the wall. Early peaks even surpass the initial impact pressure. This may be due to small but rapid jets that impact the wall on the perturbations.

2.5.4 Cavitation

Figure 2.8 shows the liquid droplet after the impact onto the wall. We can see in the white area inside the droplet, vapor formation, void fraction near 0.01, due to cavitation, that is to say a pressure drop which leads to formation of vapor. This cavitation zone appears after the reflection of the shock on the back of the droplet.

Cavitation refers to the creation of vapor bubbles inside liquids. This phase transition occurs due to a pressure drop in the liquid, a pressure lower than the saturation pressure $p_{\text{sat}}(T)$. The standard picture follows with a vapor bubble which collapses usually in an asymmetric way resulting in high velocity and localized liquid jets. As a result strong pressure can develop.

Following the work of Hantke and Thein 2019, the previous description of cavitation can be referred as strong cavitation because it produces pure vapor phase. It is opposed to weak cavitation which produces wet steam. As it is explained in Hantke and Thein 2019, an adiabatic expansion in equilibrium models can only produce wet steam. For that reason equilibrium models

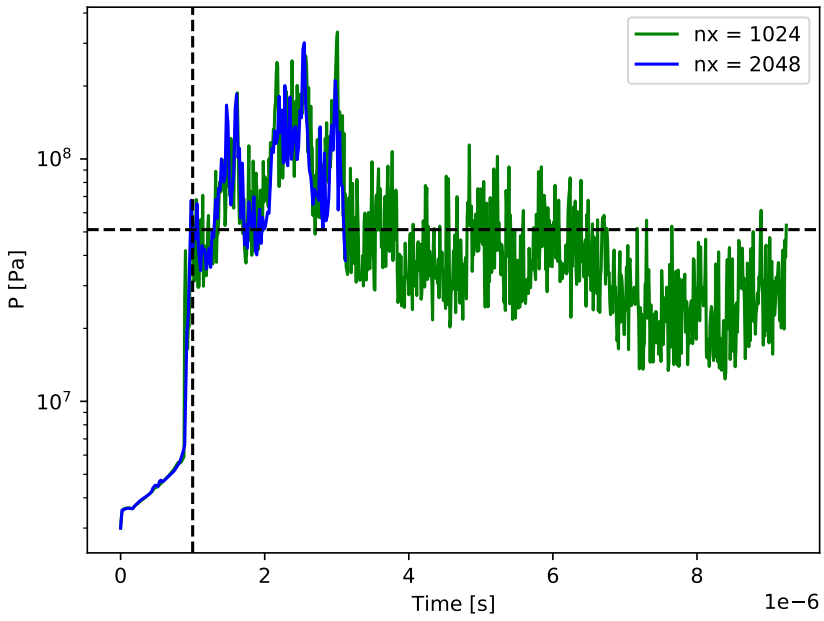


Figure 2.6: The figure represents the time evolution of the maximum pressure at the wall for simulations at high resolution 1024^3 and 2048^3. The black vertical line indicates the impact of the droplet whereas the horizontal line indicates the water hammer pressure.

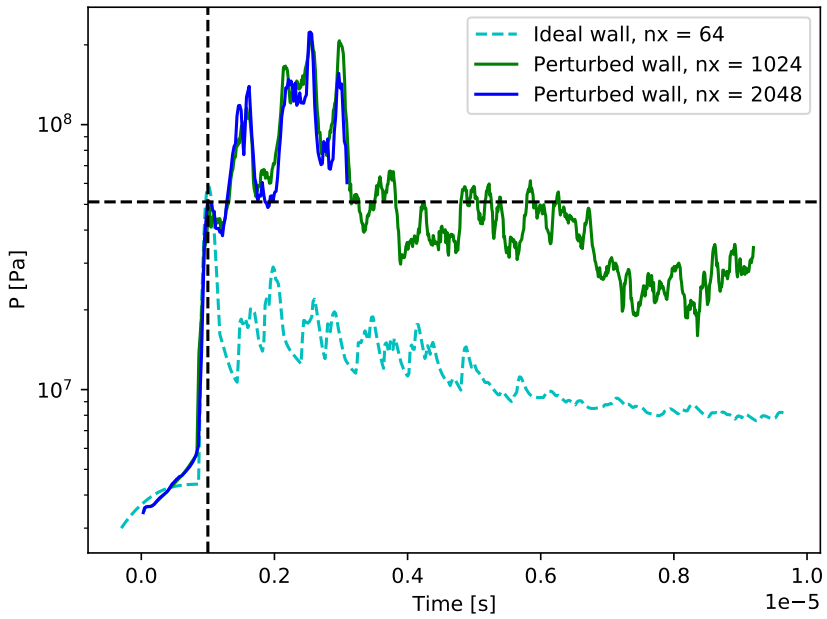


Figure 2.7: The figure represents the time evolution of the moving average maximum pressure at the wall for simulations at high resolution 1024^3 and 2048^3. We add the ideal wall as a reference. The black vertical line indicates the impact of the droplet whereas the horizontal line indicates the water hammer pressure.

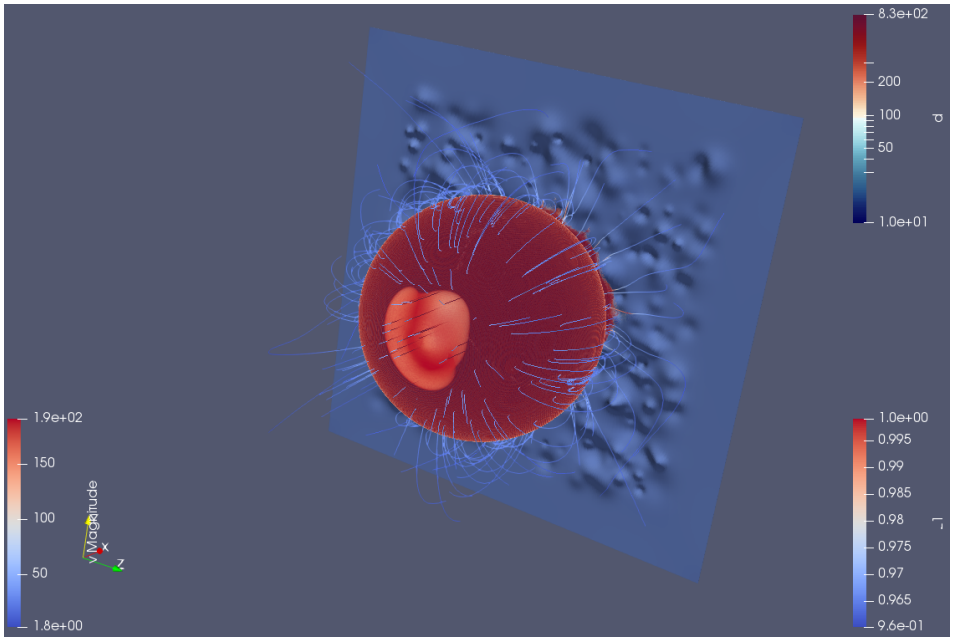


Figure 2.8: Droplet impact at resolution 1024^3. The liquid density is represented in red. The white zone at the back of the droplet represents the vapor volume fraction. Velocity field lines are displayed near the droplet.

cannot capture correctly cavitation effects from nucleation to the bubble collapse. Only the final phase of collapsing can be captured.

2.6 Discussion and interpretation of models

Even though the driving physical test formulation is simple, a liquid droplet impacting a wall, we have experienced multiple modeling and interpretation difficulties. The two-phase flow literature is rich of many models. Many of them consider different types of fractions, either volume, mass, energy. However their interpretation is rather difficult as they seem to introduce cut-off length scale or time scale. This avoids to capture phenomena such as nucleation.

We also chose to represent vapor-liquid as two phases with two different equations of state. In the initial situation this makes sense as both phases are under the critical point. However when the liquid droplet impacts the wall, pressure rises sufficiently high to reach the critical region, i.e. above the critical pressure. In this region the fluid is neither steam neither liquid and is usually referred as critical fluid. This leads to an interpretation difficulty using a color function as it either represents a liquid or vapor.

As the droplet impact leads to high pressure, mainly following the water hammer pressure, the liquid phase cannot be considered anymore as incompressible. In the compressible model, pressure is then evaluated using the couple density and internal energy. This results in a stiff formulation of thermodynamics compared to the formulation couple pressure-temperature.

The modeling of phase change, in particular driven by pressure changes, is still a challenging question. Contrary to temperature driven phase transition which can be treated in a sharp interface method as a source term at the interface, cavitation may appear in the bulk of the

liquid leading to out of equilibrium liquid which may translate to temporary negative pressures.

Finally negative pressures, also called tension, is the ability of liquids to support stretching. Contrary to gases, in the liquid phase inter-molecular interactions are important. From statistical mechanics it can be shown that liquid pressure is composed of two terms: a thermal pressure due to shock interactions, as in gases, and a term representing short range attractive interactions which is negative. Experiments, see Caupin and Herbert 2006 for a review, and molecular dynamics simulations, for example see Tan and Woodcock 2007, show that in well-prepared conditions a liquid may resist to stretching instead of undergoing phase transition. In a sufficiently strong stretching, water liquid "breaks" and steam is formed: this is called homogeneous cavitation. In the case of stiffened gases they are known to lead to negative pressures while keeping a positive speed of sound. The interpretation of negative pressures appearing in such a dynamical context is not clear as it is rather different from experiments. Further modeling along with numerical simulations is thus needed to properly establish if negative pressures arise for physical or numerical reasons.

2.7 Conclusion

As a conclusion we have used an Homogeneous Equilibrium Model to explore liquid droplet impacts onto a wall. We have used successive operator splittings to discretize the equations. A first splitting decouples the flow dynamic from phase transition. As a result, the flow dynamic can be temporarily out of thermodynamics equilibrium. Then a second splitting allows to decouple acoustic waves from transport waves. Each splitting stage allows to use specific numerical algorithms. In particular, the acoustic step is solved using a solver allowing large density ratios to avoid strong stability constraints on the time step. The transport system is solved using an anti-diffusive solver in order to keep a sharp interface.

As a result we have performed high resolution three-dimensional simulations as part of a Grand Challenge. We have been able to reproduce in a full 3D simulation the peak pressure at the impact onto the wall, compatible with the water hammer pressure. Adding roughness to the wall exhibits secondary peak pressures even higher than the initial peak.

Considering this model, a parametric study, for different initial speed, droplet radius would be easier using AMR. Indeed AMR would be of great value to help in reducing computation in the gas phase and in sharpening pressure gradients in the liquid droplet. However as it has been emphasized, AMR would not help in capturing cavitation zones in the strong sense as nucleation does not seem to be part of the model, even though they may also lead to strong pressure peaks. It can be argued that even though the cavitation mechanism is not well captured, the region where it appears is far from the wall. However we have assumed homogeneous liquid droplet at initial time, if the droplet presents initial steam cavities, this could also lead to high pressures when they collapse.

Appendix

2.A Interface models and mixture models

In this section we make a connection between interface models and mixture models.

We define $\Omega \subset \mathbb{R}^3$ a domain which contains two fluids that are separated by an interface. We note $\Omega_0(t) \subset \mathbb{R}^3$ the domain occupied by the gas and $\Omega_1(t) \subset \mathbb{R}^3$ the domain occupied by the liquid. As the two phases are separated by an interface we have $\Omega_0(t) \cap \Omega_1(t) = \emptyset$. We note $\Gamma(t) = \overline{\Omega}_0(t) \cap \overline{\Omega}_1(t)$ the location of the interface at a time t and for $\boldsymbol{x} \in \Gamma(t)$ we note $\boldsymbol{n}(t, \boldsymbol{x})$ the normal to the interface, directed towards the liquid phase. We also assume that they fill the whole domain i.e. $\overline{\Omega_0(t) \cup \Omega_1(t)} = \Omega$. In the bulk of the phases, i.e. in $\Omega_0(t) \cup \Omega_1(t)$, the interface model writes

$$\begin{aligned} \partial_t \rho + \operatorname{div}(\rho \boldsymbol{u}) &= 0, \\ \partial_t (\rho \boldsymbol{u}) + \operatorname{div}(\rho \boldsymbol{u}\boldsymbol{u} + p^{\text{eos}} \mathbb{I}) &= \boldsymbol{0}, \\ \partial_t (\rho E) + \operatorname{div}((\rho E + p^{\text{eos}}) \boldsymbol{u}) &= 0, \\ \rho E = \rho e + \frac{1}{2} \rho \boldsymbol{u}^2, & \end{aligned} \tag{2.52}$$

in which

$$p^{\text{eos}}(\rho, e) = \begin{cases} p_1^{\text{eos}}(\rho, e) & \text{if } \boldsymbol{x} \in \Omega_1(t), \\ p_0^{\text{eos}}(\rho, e) & \text{if } \boldsymbol{x} \in \Omega_0(t). \end{cases} \tag{2.53}$$

Neglecting the surface tension and any other source term at the location of the interface, the two phases are connected with jump relations of Rankine-Hugoniot type at the interface velocity $\sigma \boldsymbol{n}$

$$\begin{aligned} \sigma [\rho] &= [\rho \boldsymbol{u} \cdot \boldsymbol{n}], \\ \sigma [\rho \boldsymbol{u} \cdot \boldsymbol{n}] &= \left[\rho (\boldsymbol{u} \cdot \boldsymbol{n})^2 \right] + (p_1^{\text{eos}} - p_0^{\text{eos}}), \\ \sigma [\rho E] &= [(\rho E + p^{\text{eos}}) \boldsymbol{u} \cdot \boldsymbol{n}]. \end{aligned} \tag{2.54}$$

The pressure jump involves the two equations of state. Neglecting the mass transfer, noted $j = \rho_1 (\sigma - \boldsymbol{u}_1 \cdot \boldsymbol{n}) = \rho_0 (\sigma - \boldsymbol{u}_0 \cdot \boldsymbol{n})$, between the two phases we have $j = 0$. Thus both the normal component of the velocity and the pressure fields are continuous across the interface. The interface velocity σ reduces to the normal component of the material velocity $\boldsymbol{u} \cdot \boldsymbol{n}$. The interface behaves as a contact wave in a single phase Euler system, there is no constraint on the density jump and the tangential component of the velocity field.

Let us introduce a color function z defined by

$$z(t, \boldsymbol{x}) = \begin{cases} 0 \text{ if } \boldsymbol{x} \in \Omega_0(t), \\ 1 \text{ if } \boldsymbol{x} \in \Omega_1(t). \end{cases} \tag{2.55}$$

Because the interface is only advected with the flow velocity $\boldsymbol{u}$, the interface model 2.52 equivalently reads in Ω

$$\begin{aligned} \partial_t z + \boldsymbol{u} \cdot \boldsymbol{\nabla} z &= 0, \\ \partial_t \rho + \operatorname{div}(\rho \boldsymbol{u}) &= 0, \\ \partial_t (\rho \boldsymbol{u}) + \operatorname{div}(\rho \boldsymbol{u}\boldsymbol{u} + p^{\text{eos}} \mathbb{I}) &= \mathbf{0}, \\ \partial_t (\rho E) + \operatorname{div}((\rho E + p^{\text{eos}})\, \boldsymbol{u}) &= 0, \\ \rho E = \rho e + \frac{1}{2} \rho \boldsymbol{u}^2, & \end{aligned} \tag{2.56}$$

in which the pressure takes the form

$$p^{\text{eos}}(\rho, e, z) = (1 - z) p_0^{\text{eos}}(\rho, e) + z p_1^{\text{eos}}(\rho, e). \tag{2.57}$$

Let us emphasize that there is no new assumption in Equation 2.57, only one equation of state is used at a time, either p_0^{eos} or p_1^{eos}. Indeed if the initialization of the color function z lies in the ensemble $\{0, 1\}$, then we have $z(t, \boldsymbol{x}) \in \{0, 1\}$ due to the properties of the advection equation. From a numerical point of view, it is standard and straightforward to use a grid-based method to discretize the interface model 2.56. As a drawback, such a discretization does not respect the condition $z \in \{0, 1\}$ near the interface due to the numerical diffusion. Even though this numerical diffusion can be made small using adapted techniques such as an anti-diffusive scheme it cannot be entirely removed. As a consequence a region in which $z \notin \{0, 1\}$ needs a proper definition for the quantities required by the numerical scheme such as the pressure and the speed of sound. We then choose to model these regions using a mixture.

Let us now outline some reasonable properties that a candidate mixture model should fulfill for approximating a sharp interface flow. Let ω be a small volume which contains a portion of the interface, then both phases are present in this volume. For a sufficiently small volume ω and based on the advection properties of the interface, the mixture should satisfy the following properties

- the components of the mixture are immiscible in the sense of the Section 2.1.1,
- the components of the mixture have the same pressure.

These properties motivate the usage of the five-equation model along with the isobaric equation of state in the modelling of such two-phase flows, see Section 2.1.5. Indeed for a sufficiently thin isobaric mixture zone it can be interpreted as an interface, with the volume fraction tending to the color function. The isobaric closure allows any temperature jump as in the interface model. However the five-equation model imposes that the mixture components are in mechanical equilibrium. The thin mixture zone then locally enforces adherence of the two phases.

2.B Anti-diffusive flux

In this appendix we recall derivation of the anti-diffusive flux in the 1D context, see Després and Lagoutière 2001; Kokh and Lagoutière 2010 for more details. We follow notations previously introduced. The equation to be solved writes

$$\partial_t z + u \partial_x z = 0, \tag{2.58}$$

in which $u = u(t, x)$ is a given function. We assume to be known a set of discrete values $u^*_{i+1/2}$ consistent with u associated with the position $x_{i+1/2}$ as defined in the Section 2.2. We further

introduce min and max values of z at interfaces

$$\begin{aligned} m_{i+1/2}^n &= \min\left(z_i^n, z_{i+1}^n\right), \\ M_{i+1/2}^n &= \max\left(z_i^n, z_{i+1}^n\right). \end{aligned} \tag{2.59}$$

Discretization is carried out in the equivalent semi-conservative following form

$$\partial_t z + \partial_x (zu) - z\partial_x u = 0. \tag{2.60}$$

We choose a first order discretization of the form

$$z_i^{n+1} = z_i^n - \frac{\Delta t}{\Delta x}\left(z_{i+1/2}^* u_{i+1/2}^* - z_{i-1/2}^* u_{i-1/2}^*\right) + z_i^n \frac{\Delta t}{\Delta x}\left(u_{i+1/2}^* - u_{i-1/2}^*\right). \tag{2.61}$$

This discretization is consistent as long as $z_{i+1/2}^*$ is a consistent flux. The main idea of the derivation is to have the most downwind flux with a stability constraint. This is achieved in two steps using stability and consistency criteria.

2.B.1 Flux consistency

A sufficient condition for the flux consistency is given by

$$z_{i+1/2}^* \in \left[m_{i+1/2}^n, M_{i+1/2}^n\right]. \tag{2.62}$$

Indeed if $(z_i^n, u_i^n) = \left(z_{i+1}^n, u_{i+1}^n\right) = (z, u)$ then $z_{i+1/2}^* = z$ and the flux satisfies $z_{i+1/2}^* u_{i+1/2}^* = zu$ which is the definition of the flux consistency.

2.B.2 Stability

Stability is analyzed depending on local velocities. We distinguish four cases to analyze the stability depending on the sign of $u_{k+1/2}^*$ for $k = i-1, i, i+1$

1. $u_{i+1/2}^* > 0$ and $u_{i-1/2}^* > 0$
2. $u_{i+1/2}^* > 0$ and $u_{i-1/2}^* < 0$
3. $u_{i+1/2}^* < 0$ and $u_{i+3/2}^* > 0$
4. $u_{i+1/2}^* < 0$ and $u_{i+3/2}^* < 0$

In the following we develop only the two first cases.

Case $u_{i+1/2}^* > 0$, $u_{i-1/2}^* > 0$

In the case of $u_{i+1/2}^* > 0$ and $u_{i-1/2}^* > 0$, a sufficient condition for obtaining local stability within cell i is given by, see Després and Lagoutière 2001,

$$z_i^{n+1} \in \left[m_{i-1/2}, M_{i-1/2}\right]. \tag{2.63}$$

Thus developing update z_i^{n+1} from Equation 2.61, Equation 2.63 writes

$$m_{i-1/2} \leq z_i^n - \frac{\Delta t}{\Delta x}\left(z_{i+1/2}^* u_{i+1/2}^* - z_{i-1/2}^* u_{i-1/2}^*\right) + z_i^n \frac{\Delta t}{\Delta x}\left(u_{i+1/2}^* - u_{i-1/2}^*\right) \leq M_{i-1/2}. \tag{2.64}$$

It equivalently writes

$$\begin{aligned} -z_i^n u_{i+1/2}^* + z_i^n \left(u_{i-1/2}^* - \frac{\Delta x}{\Delta t} \right) + \frac{\Delta x}{\Delta t} m_{i-1/2} - z_{i-1/2}^* u_{i-1/2}^* \leq -z_{i+1/2}^* u_{i+1/2}^* \\ -z_{i+1/2}^* u_{i+1/2}^* \leq -z_i^n u_{i+1/2}^* + z_i^n \left(u_{i-1/2}^* - \frac{\Delta x}{\Delta t} \right) + \frac{\Delta x}{\Delta t} M_{i-1/2} - z_{i-1/2}^* u_{i-1/2}^* \end{aligned} \tag{2.65}$$

By consistency of fluxes, see 2.62 and hypothesis $u_{i-1/2}^* > 0$, we have a sufficient condition for stability

$$\begin{aligned} -z_i^n + \left(z_i^n - m_{i-1/2} \right) \left(\frac{u_{i-1/2}^*}{u_{i+1/2}^*} - \frac{\Delta x}{\Delta t} \frac{1}{u_{i+1/2}^*} \right) \leq -z_{i+1/2}^* \\ -z_{i+1/2}^* \leq -z_i^n + \left(z_i^n - M_{i-1/2} \right) \left(\frac{u_{i-1/2}^*}{u_{i+1/2}^*} - \frac{\Delta x}{\Delta t} \frac{1}{u_{i+1/2}^*} \right) \end{aligned} \tag{2.66}$$

Using hypothesis $u_{i+1/2}^* > 0$, we obtain the following result

$$\begin{cases} z_{i+1/2}^* \in \left[a_{i+1/2}^n, A_{i+1/2}^n \right] \\ a_{i+1/2}^n = z_i^n + \left(M_{i-1/2} - z_i^n \right) \left(\frac{u_{i-1/2}^*}{u_{i+1/2}^*} - \frac{\Delta x}{\Delta t} \frac{1}{u_{i+1/2}^*} \right) \\ A_{i+1/2}^n = z_i^n + \left(m_{i-1/2} - z_i^n \right) \left(\frac{u_{i-1/2}^*}{u_{i+1/2}^*} - \frac{\Delta x}{\Delta t} \frac{1}{u_{i+1/2}^*} \right) \end{cases} \tag{2.67}$$

Case $u_{i+1/2}^* > 0$, $u_{i-1/2}^* < 0$

In the case of $u_{i+1/2}^* > 0$ and $u_{i-1/2}^* < 0$ we use a standard upwind flux given by

$$z_{i+1/2}^* = z_i^n \tag{2.68}$$

Case $u_{i+1/2}^* < 0$, $u_{i+3/2}^* < 0$

In the case of $u_{i+1/2}^* < 0$ and $u_{i-1/2}^* < 0$ we have a similar result as for the case $u_{i+1/2}^* > 0$, $u_{i-1/2}^* > 0$

$$\begin{cases} z_{i+1/2}^* \in \left[a_{i+1/2}^n, A_{i+1/2}^n \right] \\ a_{i+1/2}^n = z_{i+1}^n + \left(M_{i+3/2} - z_{i+1}^n \right) \left(\frac{u_{i+3/2}^*}{u_{i+1/2}^*} - \frac{\Delta x}{\Delta t} \frac{1}{u_{i+1/2}^*} \right) \\ A_{i+1/2}^n = z_{i+1}^n + \left(m_{i+3/2} - z_{i+1}^n \right) \left(\frac{u_{i+3/2}^*}{u_{i+1/2}^*} - \frac{\Delta x}{\Delta t} \frac{1}{u_{i+1/2}^*} \right) \end{cases} \tag{2.69}$$

Case $u_{i+1/2}^* < 0$, $u_{i+3/2}^* > 0$

In the case of $u_{i+1/2}^* > 0$ and $u_{i+3/2}^* < 0$ we use a standard upwind flux given by

$$z_{i+1/2}^* = z_{i+1}^n \tag{2.70}$$

2.B.3 Overall algorithm

Thus in order to have both consistency and stability, we can choose the flux in the interval $I^n_{i+1/2}$ defined by

$$I^n_{i+1/2} = \left[\omega^n_{i+1/2}, \Omega^n_{i+1/2}\right] = \left[a^n_{i+1/2}, A^n_{i+1/2}\right] \cap \left[m^n_{i+1/2}, M^n_{i+1/2}\right]. \tag{2.71}$$

One can show that $I^n_{i+1/2} \neq \emptyset$ as $z^n_i \in I^n_{i+1/2}$ (resp. $z^n_{i+1} \in I^n_{i+1/2}$) when $u_{i+1/2} > 0$ (resp. $u_{i-1/2}$). The flux value is then chosen to be the most downwind possible value that lies within $I^n_{i+1/2}$. This can be formulated as the projection of the downwind value onto the trust interval $I^n_{i+1/2}$, noted $\mathrm{P}_{I^n_{i+1/2}}$, defined by

$$\mathrm{P}_{I^n_{i+1/2}}(x) = \begin{cases} \omega^n_{i+1/2} & \text{if } x < \omega^n_{i+1/2}, \\ x & \text{if } \omega^n_{i+1/2} \leq x \leq \Omega^n_{i+1/2}, \\ \Omega^n_{i+1/2} & \text{if } \Omega^n_{i+1/2} < x. \end{cases} \tag{2.72}$$

We define the downwind and upwind indices relative to the velocity on the right face $u^*_{i+1/2}$. We define the downwind indices as

$$i_{\rightrightarrows} = \begin{cases} i+1 & \text{if } u^*_{i+1/2} > 0 \\ i & \text{else} \end{cases} \qquad (i+1/2)_{\rightrightarrows} = \begin{cases} i+3/2 & \text{if } u^*_{i+1/2} > 0 \\ i-1/2 & \text{else} \end{cases} \tag{2.73}$$

and the upwind index as

$$i_{\leftrightarrows} = \begin{cases} i & \text{if } u^*_{i+1/2} > 0 \\ i+1 & \text{else} \end{cases} \qquad (i+1/2)_{\leftrightarrows} = \begin{cases} i-1/2 & \text{if } u^*_{i+1/2} > 0 \\ i+3/2 & \text{else} \end{cases} \tag{2.74}$$

Thus, when $u^*_{i+1/2} u^*_{(i+1/2)_{\leftrightarrows}} > 0$ the anti-diffusive flux writes

$$\begin{cases} z^*_{i+1/2} = \mathrm{P}_{I^n_{i+1/2}}\left(z^n_{i_{\rightrightarrows}}\right), \\ m^n_{i+1/2} = \min\left(z^n_i, z^n_{i+1}\right), \\ M^n_{i+1/2} = \max\left(z^n_i, z^n_{i+1}\right), \\ a^n_{i+1/2} = z^n_{i_{\leftrightarrows}} + \left(M^n_{(i+1/2)_{\leftrightarrows}} - z^n_{i_{\leftrightarrows}}\right)\left(\dfrac{u^*_{(i+1/2)_{\leftrightarrows}}}{u^*_{i+1/2}} - \dfrac{\Delta x}{\Delta t}\dfrac{1}{u^*_{i+1/2}}\right), \\ A^n_{i+1/2} = z^n_{i_{\leftrightarrows}} + \left(m^n_{(i+1/2)_{\leftrightarrows}} - z^n_{i_{\leftrightarrows}}\right)\left(\dfrac{u^*_{(i+1/2)_{\leftrightarrows}}}{u^*_{i+1/2}} - \dfrac{\Delta x}{\Delta t}\dfrac{1}{u^*_{i+1/2}}\right), \end{cases} \tag{2.75}$$

otherwise we set

$$z^*_{i+1/2} = z^n_{i_{\leftrightarrows}}. \tag{2.76}$$

Chapter 3

Spectral Volume method

Introduction

In the field of hyperbolic conservation laws, a lot of work has been dedicated to improve precision of simulations since the developments of the first order Godunov method (Godunov 1959) and subsequent Godunov-like methods. This method has provided a general framework in which to derive upwind numerical schemes i.e. taking into account the direction of the flow. However this stability can be interpreted into larger numerical diffusion not only near discontinuities but also in smooth regions. This can lead to precision difficulties when dealing with smooth solutions. When the solution is smooth enough, the truncation error of a numerical scheme can formally be estimated at

$$Ch^p$$

These three letters offer three options to improve precision

- decreasing grid spacing with parameter h also referred as h-convergence
- increasing order of convergence of the numerical method with parameter p also referred as p-convergence
- lowering the value of the coefficient C: this can happen by adapting the numerical scheme to better fit specific simulation cases.

When a numerical scheme has a convergence order $p \geq 2$, it is qualified as high-order. A first category of robust and high-order methods is based on the reconstruction of the solution using neighbours of Finite Volume Cells. The general idea is to follow Godunov's method but using reconstructed states at faces of cells instead of cell centered values. Such a well-known method is called MUSCL-Hancock scheme which stands for "Monotonic Upstream-centered Scheme for Conservation Laws" along with a predictor-corrector for time integration, see Toro 2009.

A second category is called Spectral Methods, inspired by Finite Elements, in which one adds degrees of freedom inside cells in order to reconstruct a polynomial, such cells are then called Spectral Cells. Because of the polynomial reconstruction, the solution is smooth inside a Spectral Cell and this can be exploited to derive a numerical scheme. Again at interfaces between Spectral Cells, the solution is discontinuous and a Riemann Solver is used as in the original Godunov's method. There are essentially three families of Spectral Methods: Discontinuous Galerkin method, Spectral Difference method and Spectral Volume method. We can distinguish them based on the equation formulation used to derive the numerical scheme; strong, integral

and weak formulation. In the case of the Discontinuous Galerkin method is based on a weak formulation of the conservation laws and leads to different numerical schemes depending on the definition of the test function space, for example see Cockburn et al. 2000. It has recently gained popularity because of its ease to adapt to different types of equations and meshes. Spectral Difference method relies on the reconstruction of both smooth solutions and flux functions. Evolution of degrees of freedom is then based on a strong formulation of the equations and finite differences (Liang et al. 2009). Its strength is an easy formulation. Finally Spectral Volume method has been introduced in a serie of papers (Wang 2002; Wang and Liu 2002; Wang and Liu 2004; Wang et al. 2004; Liu et al. 2006) and is close to the original Finite Volume method. Indeed it is based on the reconstruction of smooth solutions but degrees of freedom are evolved using the integral formulation of the equations. In the latter, the conservation property is not only expressed at the Spectral Cell level but also for degrees of freedom. All of them are here presented as spatial reconstructions and can be used along with a Runge-Kutta time integrator.

Because of the presence of discontinuities in conservation laws, a limitation process has to be used on the reconstructed solution. They play an important role in high-order schemes in order to avoid spurious oscillations near discontinuities by removing artificial extrema generated by the reconstruction step. Limiters force the numerical scheme to locally fall back to first order in accordance with the local regularity of the solution. As a consequence a too sensitive limiter forces the numerical scheme to behave like a first order scheme. Examples of well-known limiters are minmod, gminmod, superbee, TVD, ... This limiting process can be referred as a priori in comparison to an a posteriori method like MOOD which stands for "Multi-dimensional Optimal Order Detection", see Clain et al. 2011. In such approach, different numerical schemes, which may be of different order of convergence are tried for a given cell update and the best solution is selected based on predefined criteria.

Another aspect is the computational cost of a numerical scheme. High-order methods usually become more expensive for a given mesh because they involve more computation per cell. However an other point of view consists in evaluating efficiency by comparing computational costs at a given error. In that sense it has been successfully shown that high-order methods are more efficient than a second order MUSCL-Hancock method on smooth examples (see Schaal et al. 2015). Different arguments have been invoked

- more efficient p-convergence over h-convergence on smooth test cases leading to coarser grid hence less time iterations,
- less expensive non-linear computation such as Riemann problems and limiters,
- easier data locality coming from Spectral Cells making them good candidates to target new supercomputers in which cache plays an important role.

In the case of discontinuous solutions, p-convergence is however not suitable. Indeed discontinuous solutions benefit from h-convergence rather than p-convergence. Moreover as it has been said, computation cost per cell is more expensive when using a high-order scheme. This usually results in using an expensive high-order numerical scheme in discontinuous regions. Hence we propose to solve this problem by using cheaper standard Finite Volume scheme near discontinuities and accelerate computation in smooth regions by taking advantage of p-convergence. Acceleration can be achieved by building Spectral Cells from Finite Volume Cells resulting in less cells to update in smooth regions. The Spectral Volume method is used as it provides an easy transition between Finite Volume Cells and its degrees of freedom.

The chapter is organized as follows, we start by defining some notation in Section 3.1. Section 3.2 recalls the main idea of the Spectral Volume Method using a polynomial flux reconstruction for Cartesian meshes to improve efficiency. Then in Section 3.3 we present our numerical

method. In Section 3.4 we give a matrix formulation of the Spectral Volume Method. In Section 3.5 we present numerical experiments for common test cases found in the literature.

3.1 Notations

We begin by defining some notations. Let

$$\Omega = [x_{\min}, x_{\max}] \times [y_{\min}, y_{\max}] \times [z_{\min}, z_{\max}]$$

be a Cartesian subdomain of $\mathbf{R}^3$. We want to solve an Initial-Boundary Value Problem (IBVP) in the case of a hyperbolic scalar conservation law (Godlewski and Raviart 1991; Bouchut 2004; Dafermos 2010), that takes the form

$$\begin{gathered} u(0,x) = u_0(x), \\ \forall t>0, \forall \boldsymbol{x} \in \Omega \quad \frac{\partial}{\partial t} u(t,\boldsymbol{x}) + \operatorname{div}\left(\boldsymbol{f}(u(t,\boldsymbol{x}))\right) = 0, \\ \text{B.C. on } \partial\Omega, \end{gathered} \tag{3.1}$$

in which $\operatorname{div}(.)$ is the 3D spatial divergence operator and $\boldsymbol{f} = (f^x, f^y, f^z)$ is called the flux function.

Equation 3.1 also admits an integral formulation. Let u be a solution of Equation 3.1. By integrating the Equation 3.1 over a given volume $\omega \subset \Omega$ and by applying the flux-divergence theorem the solution u satisfies

$$\begin{aligned} \bar{u}(t) &= \frac{1}{|\omega|} \iiint_{\omega} u(t,\boldsymbol{x}) \mathrm{d}\boldsymbol{x}, \\ \frac{\mathrm{d}}{\mathrm{d}t}\bar{u}(t) + \frac{1}{|\omega|} \iint_{\partial\omega} \boldsymbol{f}(u(t,\boldsymbol{x})) \cdot \boldsymbol{n}(\boldsymbol{x}) \, \mathrm{d}\sigma(\boldsymbol{x}) &= 0. \end{aligned} \tag{3.2}$$

This integral formulation will be used to derive a conservative numerical scheme. Conservative numerical schemes are known to be able to capture weak solutions, see Lax and Wendroff 1960.

As it will be developed in following sections, we will define two meshes $\mathcal{M}$ and $\mathcal{M}^S$ over Ω in order to use two different representations of the numerical solution, see Figure 3.1.

First we define a Cartesian mesh, referred to as Finite Volume mesh noted $\mathcal{M}$: it is composed of N_x, N_y and N_z cells in each dimension and Δx, Δy, Δz are the corresponding mesh space steps. In the X-direction (resp. Y and Z-direction) interfaces are located at $x_{i+1/2} = (i+1)\Delta x$, (resp. $y_{j+1/2} = (j+1)\Delta y$, $z_{k+1/2} = (k+1)\Delta z$) with $(x_{\min}, x_{\max}) = (x_{-1/2}, x_{N_x+1/2})$ (resp. $(y_{\min}, y_{\max}) = (y_{-1/2}, y_{N_y+1/2})$ and $(z_{\min}, z_{\max}) = (z_{-1/2}, z_{N_z+1/2})$). Hence the domain Ω is partitioned into non-overlapping regular cells noted $\omega_{i,j,k}$ with

$$\begin{aligned} \omega_{i,j,k} &= \left[x_{i-1/2}, x_{i+1/2}\right] \times \left[y_{j-1/2}, y_{j+1/2}\right] \times \left[z_{k-1/2}, z_{k+1/2}\right], \\ \Omega &= \bigcup_{i,j,k} \omega_{i,j,k}, \end{aligned} \tag{3.3}$$

and $|\omega_{i,j,k}| = \Delta x \Delta y \Delta z$. We define x_i, y_j and z_k by setting

$$x_i = \frac{1}{2}\left(x_{i-1/2} + x_{i+1/2}\right), \; y_j = \frac{1}{2}\left(y_{j-1/2} + y_{j+1/2}\right), \; z_k = \frac{1}{2}\left(z_{k-1/2} + z_{k+1/2}\right). \tag{3.4}$$

A second Cartesian mesh, referred to as the Spectral Mesh noted $\mathcal{M}^S$, is built by aggregating Finite Volume Cells within a set of Spectral Cells $\omega^S_{i_s,j_s,k_s}$ that also forms a new Cartesian grid over Ω, see Figure 3.1. More precisely: a Spectral Cell $\omega^S_{i_s,j_s,k_s}$ is an aggregate of

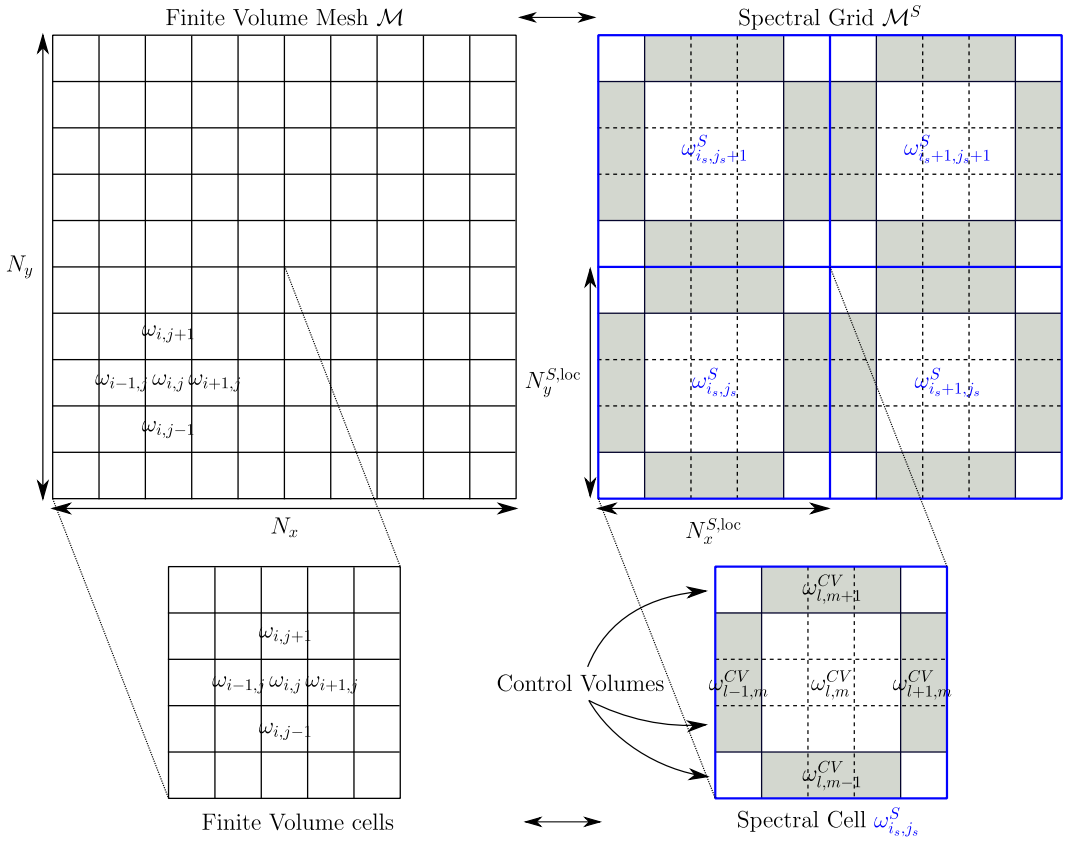


Figure 3.1: Example of two 2D-grids used to represent the numerical solution. The mesh on the left is a standard Finite Volume mesh, each cell approximating the mean value of the exact solution. The mesh on the right is the corresponding spectral mesh with spectral cells highlighted by a blue border. Inside each Spectral Cell are represented Control Volumes, also called degrees of freedom, colored alternatively in grey and white. Both Spectral Cells and Control Volumes exactly match a set of Finite Volume Cells for a given pattern; $(1, 3, 1)$ in each direction in this example. If Finite Volume Cells are interpreted as degrees of freedom, we see that the spectral scheme can be interpreted as a compression of information when a spectral representation is adapted to the solution.

$N_x^{S,\mathrm{loc}} N_y^{S,\mathrm{loc}} N_z^{S,\mathrm{loc}}$ Finite Volume Cells i.e.

$$\begin{aligned}\omega^S_{i_s,j_s,k_s} &= \bigcup_{i=N_x^{S,\mathrm{loc}} i_s}^{N_x^{S,\mathrm{loc}}(i_s+1)} \bigcup_{j=N_y^{S,\mathrm{loc}} j_s}^{N_y^{S,\mathrm{loc}}(j_s+1)} \bigcup_{k=N_z^{S,\mathrm{loc}} k_s}^{N_z^{S,\mathrm{loc}}(k_s+1)} \omega_{i,j,k} \\ &= \left[x^S_{i_s-1/2}, x^S_{i_s+1/2}\right] \times \left[y^S_{j_s-1/2}, y^S_{j_s+1/2}\right] \times \left[z^S_{k_s-1/2}, z^S_{k_s+1/2}\right].\end{aligned} \tag{3.5}$$

Thus, for a Spectral Cell $\omega^S_{i_s,j_s,k_s} \in \mathcal{M}^S$ we introduce the following set of indices

$$FV_{i_s,j_s,k_s} = \{(i,j,k) \quad \text{such that} \quad \omega_{i,j,k} \subset \omega^S_{i_s,j_s,k_s}\}.$$

This way a Spectral Cell $\omega^S_{i_s,j_s,k_s} \in \mathcal{M}^S$ can be written as

$$\omega^S_{i_s,j_s,k_s} = \bigcup_{(i,j,k)\in FV_{i_s,j_s,k_s}} \omega_{i,j,k}.$$

The number of Finite Volume Cells inside each Spectral Cell is a free parameter and is discussed in Section 3.3.
Finally each Spectral Cell is composed of degrees of freedom, also called Control Volumes in the case of the Spectral Volume Method, noted $\omega^{CV}_{l,m,n}$. Let us consider three sets of $p \geq 1$ integers n_r^x, n_r^y and n_r^z following

$$\sum_{r=0}^{p-1} n_r = N_x^{S,\mathrm{loc}}, \quad \sum_{r=0}^{p-1} n_r^y = N_y^{S,\mathrm{loc}}, \quad \sum_{r=0}^{p-1} n_r^z = N_z^{S,\mathrm{loc}}. \tag{3.6}$$

We consider the sets of positions defined by

$$\begin{aligned}&\left\{x^S_{i_s-1/2} + n_r^x \Delta x, 0 \leq r \leq p-1\right\}, \\ &\left\{y^S_{j_s-1/2} + n_r^y \Delta x, 0 \leq r \leq p-1\right\}, \\ &\left\{z^S_{k_s-1/2} + n_r^z \Delta x, 0 \leq r \leq p-1\right\},\end{aligned} \tag{3.7}$$

that are respectively spanned by the strictly increasing sequences $\left(x^{CV}_{l+1/2}\right)$, $\left(y^{CV}_{m+1/2}\right)$ and $\left(z^{CV}_{n+1/2}\right)$. The Control Volumes are then defined by

$$\omega^{CV}_{l,m,n} = \left[x^{CV}_{l-1/2}, x^{CV}_{l+1/2}\right] \times \left[y^{CV}_{m-1/2}, y^{CV}_{m+1/2}\right] \times \left[z^{CV}_{n-1/2}, z^{CV}_{n+1/2}\right]. \tag{3.8}$$

We also define x_l^{CV}, y_m^{CV} and z_n^{CV} by setting

$$x_l^{CV} = \frac{1}{2}\left(x^{CV}_{l-1/2} + x^{CV}_{l+1/2}\right), y_m^{CV} = \frac{1}{2}\left(y^{CV}_{m-1/2} + y^{CV}_{m+1/2}\right), z_n^{CV} = \frac{1}{2}\left(z^{CV}_{n-1/2} + z^{CV}_{n+1/2}\right). \tag{3.9}$$

We also introduce two new sets of indices

$$\begin{aligned}CV_{i_s,j_s,k_s} &= \{(l,m,n) \quad \text{such that} \quad \omega^{CV}_{l,m,n} \subset \omega^S_{i_s,j_s,k_s}\}, \\ FV_{l,m,n} &= \{(i,j,k) \quad \text{such that} \quad \omega_{i,j,k} \subset \omega^{CV}_{l,m,n}\}.\end{aligned}$$

In the following we identify the triplet of indices (i,j,k) (resp. (i_s,j_s,k_s), (l,m,n)) to the corresponding cell $\omega_{i,j,k}$ (resp. $\omega^S_{i_s,j_s,k_s}$, $\omega^{CV}_{l,m,n}$). We define $u_{i,j,k}$ (resp. u_{i_s,j_s,k_s}, $u_{l,m,n}$) to be

a value associated with a Finite Volume Cell (i, j, k) (resp. Spectral Cell (i_s, j_s, k_s), Control Volume (l, m, n)).

Let us note the difference of construction of the two meshes compared to classical high-order numerical methods. In a classical high-order method one defines a unique spectral grid first and then defines degrees of freedom inside each Spectral Cell. This difference is related to the point of view of accelerating computation using a spectral method rather than increasing precision.

For the sake of clarity we will first present our discretization strategy in the case of a scalar conservation law 3.1. Then we will present the extension to the full Euler system in Section 3.5.1 consisting in applying the same procedure to each equation.

3.2 Spectral Volume method

3.2.1 Semi-discrete scheme

In this section we only deal with the spectral mesh $\mathcal{M}^S$ and forget about the initial Finite Volume mesh $\mathcal{M}$. The main idea of the Spectral Volume Method is to enforce Spectral Cell's polynomial to follow a conservation principle for each Control Volume. As it has already been mentioned, in the Spectral Volume Method, degrees of freedom can be interpreted as a Finite Volume sub-grid within the Spectral Cell. This method only deals with spatial discretization and does not provide any insight into the time integration, we are free to choose one's preferred method. Therefore we shall present the method as a semi-discrete numerical scheme.

Let $u^S_{i_s,j_s,k_s}$ be a polynomial associated with the Spectral Cell $\omega^S_{i_s,j_s,k_s}$ of the form

$$u^S_{i_s,j_s,k_s}(t,x,y,z) = \sum_{0\leq q,r,s\leq p-1} \left(u^S_{i_s,j_s,k_s}(t)\right)_{q,r,s} x^q y^r z^s. \tag{3.10}$$

The number of coefficients of this polynomial matches the number of Control Volumes per Spectral Cell and is of degree $p-1$ per direction. At each time t, $u^S_{i_s,j_s,k_s}$ represents a polynomial approximation of the exact solution u within the Spectral Cell $\omega^S_{i_s,j_s,k_s}$. In order to derive a numerical scheme we take advantage of the following property: the polynomial $u^S_{i_s,j_s,k_s}$ is characterized by its mean values in each Control Volume $\omega^{CV}_{l,m,n}$ of the Spectral Cell $\omega^S_{i_s,j_s,k_s}$. This allows us to seek for a numerical scheme evolving the polynomial $u^S_{i_s,j_s,k_s}$ based on the discretization of the Equation 3.2

$$\begin{aligned} u_{l,m,n}(t) &= \frac{1}{|\omega^{CV}_{l,m,n}|} \iiint_{\omega^{CV}_{l,m,n}} u^S_{i_s,j_s,k_s}(t,\boldsymbol{x})\mathrm{d}\boldsymbol{x} \\ \frac{\mathrm{d}}{\mathrm{d}t}u_{l,m,n}(t) + \frac{1}{|\omega^{CV}_{l,m,n}|} &\iint_{\partial\omega^{CV}_{l,m,n}} \boldsymbol{\mathcal{F}}(t,\boldsymbol{x})\cdot\boldsymbol{n}(\boldsymbol{x})\,\mathrm{d}\sigma(\boldsymbol{x}) = 0 \end{aligned} \tag{3.11}$$

where $\boldsymbol{\mathcal{F}} = (\mathcal{F}^x, \mathcal{F}^y, \mathcal{F}^z)$ is a polynomial numerical flux that will be defined in the next section (for the X-direction).

3.2.2 Polynomial flux reconstruction $\mathcal{F}^x$

We need a high-order estimation of the numerical flux $\mathcal{F}^x$ in the Equation 3.11. To this point there are two types of faces for the Control Volumes: internal faces (black lines in Figure 3.2) and external faces (blue lines in Figure 3.2) relative to the Spectral Cell $\omega^S_{i_s,j_s,k_s}$. Even though for internal faces, the numerical flux

$$f^x\left(u^S_{i_s,j_s,k_s}(t,\boldsymbol{x})\right) \tag{3.12}$$

can be analytically computed for some simple flux functions, in the general case the flux function f^x is highly non-linear so that the computation of the flux integral can be difficult. The standard choice for Spectral Methods is to use quadrature formulas for both internal and external faces. These quadrature rules are chosen with a sufficient order of accuracy preserve the order of accuracy of the Spectral Method. So for both external and internal faces, the evaluation of the flux integral falls back to provide an approximation of the flux function f^x at quadrature points.

In the original work of Wang 2002, they used a quadrature formula per Control Volume. This choice had the advantage to work for any unstructured meshes in which distribution of Control Volumes is more complex. The inconvenient is that it is more expensive than other spectral methods such as Discontinuous Galerkin or Spectral Difference methods.

In our case we aim at taking advantage of the Cartesian structure of the mesh $\mathcal{M}^S$ to reduce the number of quadrature points per Spectral Cell. To do so we consider a layered polynomial flux reconstruction using a Lagrange interpolation inside a Spectral Cell at centers of the Control Volumes, the points of coordinates $\left(\left(y_j^{CV}, z_k^{CV}\right)\right)_{1\leq j\leq p, 1\leq k\leq p}$, it writes

$$\mathcal{F}^x(t,x,y,z) = \sum_{m,n} \mathcal{F}^x_{m,n}(t,x) L_{m,n}(y,z), \tag{3.13}$$

in which $L_{m,n}$ is a normalized Lagrange interpolation polynomial

$$L_{m,n}(y,z) = \prod_{(j,k)\neq(m,n)} \frac{y - y_j^{CV}}{y_m^{CV} - y_j^{CV}} \frac{z - z_k^{CV}}{z_n^{CV} - z_k^{CV}}, \tag{3.14}$$

and interpolating the flux values $\left(\mathcal{F}^x_{m,n}(t,x)\right)_{1\leq j\leq p, 1\leq k\leq p}$ defined as in Equation 3.15 and Equation 3.16. Figure 3.2 shows an example of such a distribution of point interpolations in the case of a third order method.

For internal faces, we take advantage of the polynomial representation $u^S_{i_s,j_s,k_s}$ to directly evaluate the flux function, we set

$$\mathcal{F}^x_{m,n}\left(t, x^{CV}_{l+1/2}\right) = f^x\left(u^S_{i_s,j_s,k_s}(t, x^{CV}_{l+1/2}, y^{CV}_m, z^{CV}_n)\right). \tag{3.15}$$

In the case of external faces, such that $x^{CV}_{l+1/2} = x^S_{i_s+1/2}$, the polynomials of the two adjacent Spectral Cells are a priori discontinuous. We then use a Riemann Solver $\mathcal{F}^{RS}$ to have a unique flux at the face

$$\begin{gathered} \mathcal{F}^x_{m,n}(t, x^{CV}_{l+1/2}) = \mathcal{F}^{RS}(u_L, u_R), \\ u_L = u^S_{i_s,j_s,k_s}(t, x^{CV}_{l+1/2}, y^{CV}_m, z^{CV}_n), \quad u_R = u^S_{i_s+1,j_s,k_s}(t, x^{CV}_{l+1/2}, y^{CV}_m, z^{CV}_n). \end{gathered} \tag{3.16}$$

The numerical flux $\mathcal{F}^{RS}$ can be an Exact Riemann Solver (Godunov 1959) or any Approximate Riemann Solver, we refer the reader to the large literature for this topic, local Lax-Friedrichs Lax 1954, HLL Harten et al. 1983, HLLC Toro and Chakraborty 1994, relaxation, Roe Roe 1981 solvers...

From the semi-discrete numerical scheme 3.11, the interpolated flux functions $(\mathcal{F}^x, \mathcal{F}^y, \mathcal{F}^z)$ are then integrated over the faces of Control Volumes. These integrals can be exactly computed and reformulated using a matrix-vector product, see Section 3.4.

If this method is used as such, one needs to add a limitation procedure. In our case, no limiter is used at this stage as it will be coupled to a Finite Volume scheme using the MOOD procedure.

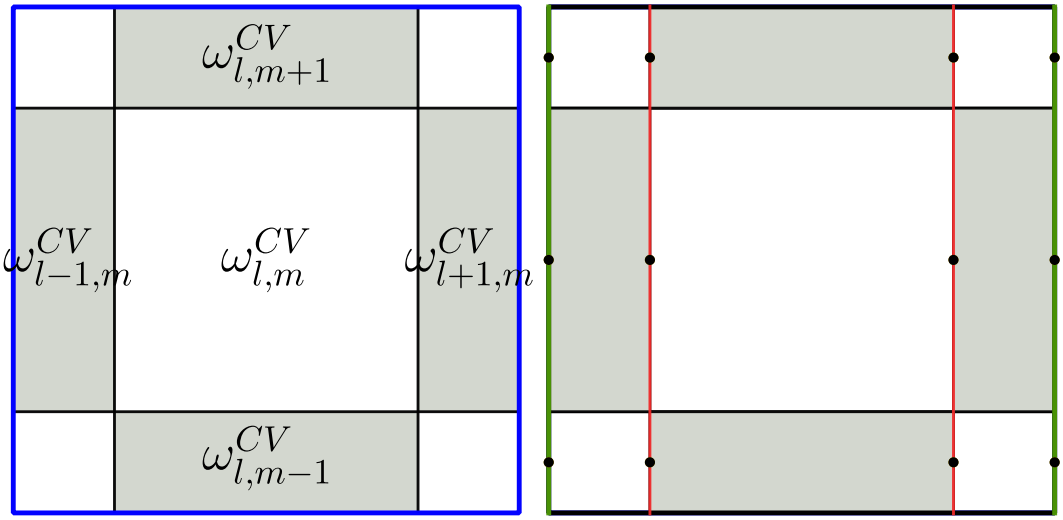


Figure 3.2: Example of a 2D Spectral Cell $\omega^S_{i_s,j_s}$ for a third order Spectral volume method, i.e. it has 3 Control Volumes in each direction. On the right figure the vertical green lines (resp. red lines) represent the external faces (resp. internal faces) in the X-direction where the polynomial flux $\mathcal{F}^x$ is reconstructed. The Lagrange interpolation is realized at the center of Control Volumes represented by the black points.

3.3 Presentation of an hybrid FV-SV method

In this section we present an hybrid FV-SV scheme. We begin by defining two kinds of representation of the solution according to the local regularity of the solution. Then we define numerical schemes adapted to each representation. Finally we define how to select the representation and how to switch from one to the other.

3.3.1 Discussion on the solution representation

In a classical Finite Volume method, being first order or second order like in a MUSCL-Hancock reconstruction, cell values represent an integral approximation of the exact solution u

$$u_{i,j,k}(t) \approx \frac{1}{|\omega_{i,j,k}|} \iiint_{\omega_{i,j,k}} u(t,\boldsymbol{x})\mathrm{d}\boldsymbol{x}. \tag{3.17}$$

Increasing the order of accuracy of the scheme is synonym to increasing the precision for this unique value.
On the other hand, in the case of spectral schemes, cells hold high-order representation of the solution. A cell does not hold anymore only a single value but degrees of freedom akin to the Finite Element method. These degrees of freedom are used to represent solution which becomes then a point-wise approximation of the solution.
As far as we are concern, we keep the grid $\mathcal{M}$ defined earlier as in the Finite Volume method. Each value in the cell is an approximation of the mean value of the exact solution. We then use the second mesh $\mathcal{M}^S$ when solution is considered smooth enough.Thus with these two grids we are able to have two types of representation for the solution. First, we have a piecewise constant

representation as in a standard Finite Volume scheme which formally writes

$$u_{i_s,j_s,k_s}(t,\boldsymbol{x}) = \sum_{i,j,k \in FV_{i_s,j_s,k_s}} \phi^{i,j,k}(t)\mathbb{1}_{\omega_{i,j,k}}(\boldsymbol{x}), \tag{3.18}$$

$\mathbb{1}_{\omega_{i,j,k}}$ being the characteristic function of cell $\omega_{i,j,k}$ and $\phi^{i,j,k}(t)$ being coefficients at time t. This representation is composed of mean values in FVC. It is a priori discontinuous and is well suited when the solution is presenting discontinuities such as contacts or shocks.
When the solution is smooth enough a spectral representation is used with a basis of functions which again formally writes on a smooth function basis $(f_\alpha)_\alpha$ to be specified

$$u_{i_s,j_s,k_s}(t,\boldsymbol{x}) = \sum_{\alpha} \phi^{\alpha}_{i_s,j_s,k_s}(t) f_\alpha(\boldsymbol{x}) \tag{3.19}$$

$\phi^{\alpha}_{i_s,j_s,k_s}$ are again coefficients at time t. To be as close as possible to the Finite Volume representation we choose to use the Spectral Volume Method as a high-order method in a Spectral Cell. As it has been recalled in Section 3.2, degrees of freedom are averaged values in sub-volumes of the Spectral Cell called Control Volumes, and these Control Volumes are cells of different sizes which also have a time evolution made with fluxes. In the case of the Spectral Volume Method reconstruction, the definition of $(f_\alpha)_\alpha$ falls back to the definition of a partition of Control Volumes inside a Spectral Cell. Once again, Control Volumes are chosen to follow the original Cartesian structure of the Finite Volume Grid of Ω. Such an example is given in Figure 3.1 in a 2D mesh. Comparing both Equations 3.18 and 3.19, we see that if there are less Control Volumes than Finite Volume Cells we have a compression of information in regions where the solution is smooth enough. This leads to less computational needs in smooth regions.

3.3.2 Choice of Control Volumes

It remains to define distribution of Control Volumes. The number of Control Volumes is determined by the order of accuracy we wish to achieve. On the one hand, sizes of Control Volumes determines the stability of the method. As it has already been explored in Wang and Liu 2004, some distributions are invalid for a stable numerical scheme. This has to be related to the so-called Runge phenomenon, see Boyd and Ong 2009. It refers to the oscillations of polynomials near the end of the interpolation intervals in the case of an equidistant point distribution. A simple way to avoid these spurious oscillations is to use closer points near end of interpolation interval. On the other hand, it also determines the precision of the numerical scheme. Larger Control Volumes will lead to a coarser spectral mesh thus less computations. We see that a trade off has to be found between the precision and the cost of the computation.
In practice we use a distribution close to Gauss-Lobatto points. Table 3.1 gives 1D distribution of Control Volumes in terms of Finite Volume Cells up to fourth order.

3.3.3 Time integration

In the case of Finite Volume Cells we wish to use the cheapest robust scheme able to deal correctly with discontinuities either shocks or contacts. A first order scheme is cheap and able to deal with shocks. However contact discontinuities are rapidly regularized. We have chosen a second order MUSCL-Hancock which is still able to deal with shocks using a slope limiter and contact discontinuities. Again a minmod slope limiter tend to be too much diffusive and we have chosen the generalized minmod limiter, see Equation 3.20

$$\text{minmod}_\theta(a,b) = \text{minmod}\left(\frac{1}{2}(a+b), \theta a, \theta b\right), \quad \theta \in [1,2] \tag{3.20}$$

order	distribution	$N_x^{S,\mathrm{loc}} = N_y^{S,\mathrm{loc}} = N_z^{S,\mathrm{loc}}$
1	(1)	1
2	(1, 1)	2
3	(1, 6, 1)	8
4	(1, 4, 4, 1)	10

Table 3.1: This table gives the distribution of Control Volumes in terms of Finite Volume Cells.

which reduces to minmod when $\theta = 1$. It has been used with $\theta = 2$ to keep contact discontinuities as sharp as possible.
In the case of Spectral Volume Method, we have a semi-discrete scheme which writes

$$\frac{\mathrm{d}}{\mathrm{d}t} u^S = L_\Delta^{\mathrm{SV}}\left(u^S\right)$$

operator L_Δ^{SV} being our Spectral Volume semi-discrete scheme. We then use a classical Runge-Kutta time integration of the same order of accuracy. A Strong Stability Preserving Runge-Kutta see Gottlieb et al. 2001, referred as SSP-RK, is usually preferred in the literature for conservation laws to avoid non-admissible states. It has been tested but did not show any major difference for our test cases.

3.3.4 Limitation procedure

As presented above we have two types of solution representation but only one of them can be used at a time. We need to define when to use them. Moreover we need a mapping to go from one representation to the other. We choose to use the Multi-dimensional Optimal Order Detection method, referred as MOOD see Clain et al. 2011. The general philosophy of this method could be summarized as a "try-catch" algorithm. It means that a high-order update candidate is realized, if it doesn't satisfy some predefined criteria it is discarded and a fallback scheme is used. Criteria used are observations of NaN, negative pressure or density and a relaxed discrete maximum principle. The result may fail to satisfy these criteria because of trying to construct a polynomial when the solution is locally discontinuous. The polynomial then presents overshoots that may go into negative densities or pressures, or violates the stability condition.
In our case, the fallback scheme is already determined by the numerical scheme on the Finite Volume mesh $\mathcal{M}$. When the Spectral Volume scheme fails to update a Spectral Cell, the solution on the Finite Volume mesh is updated using MUSCL-Hancock scheme.
The idea of using sub-grids inside a Spectral Cell is not new and has already been explored in different ways Dumbser et al. 2014; Sonntag and Munz 2017. To our understanding they both have drawbacks regarding discontinuities. In the work of Dumbser et al. 2014, using a DG scheme at order n, a uniform Finite Volume sub-grid of 2n+1 cells is used along with operators of projection $\mathcal{P}$ and reconstruction $\mathcal{R}$. This choice of sub-grid is made to match the time step in the DG scheme. The major inconvenient is the need for a reconstruction operator to compute the solution on a Spectral Cell. This means that if a Spectral Cell is detected two times in a row, because $\mathcal{P} \circ \mathcal{R} \neq I$ even though they follow $\mathcal{R} \circ \mathcal{P} = I$, there is a loss of information at the sub-grid level. In the paper of Sonntag and Munz 2017 they use a nodal DG scheme at order n. This scheme is closer to what we propose. A non-uniform sub-grid with n + 1 Finite Volume Cells, centered on Gauss points, is used to apply a standard Finite Volume scheme. There is no

more loss of information between two time steps. However non-uniform sub-grid does not make it adapted to shocks. Indeed, at a fixed number of degrees of freedom, when the order increases the size ratio between the smallest and largest cell also increases. In the scheme we propose, we try to solve both problems by using uniform Finite Volume grid as a basis. If the fallback scheme is used on Finite Volume Cells, it is applied to solution at time n from the Finite Volume mesh without modification. This means that up to the quality of the detection criteria, shocks should be as precise as a fallback scheme simulation on this regular grid.

3.3.5 Time step computation

We suppose that both Finite Volume and Spectral Volume methods are constrained by a CFL time relation of the form

$$\Delta t = C_{\mathrm{CFL}} h. \tag{3.21}$$

where h is the space step involved with the grid of each discretization. Because both schemes can be used to advance cells between time n and $n+1$, both schemes have to use the same time step. In our case to have compatibility between spectral and Finite Volume representations, we choose to have at least one Control Volume to match exactly a Finite Volume Cell, see Section 3.3.2. This enforces that both schemes have a time step computed with the same, minimal space step. Then the CFL factor C_{CFL} has to be chosen so that both schemes are stable.

3.3.6 Algorithm summary

We finally give a summary of the different identified steps in our algorithm:

1. Initialization on the Finite Volume mesh $\mathcal{M}$.
2. Time loop, while $t < t_{\max}, \quad n < n_{\max}$ do:
   (a) Construction of the Spectral representation on the Spectral mesh $\mathcal{M}^S$.
   (b) Apply the MOOD procedure:
      i. Try a high-order time evolution of the Spectral representation.
      ii. Projection of the Spectral representation onto the Finite Volume mesh $\mathcal{M}$.
      iii. Catch bad evolution of Spectral Cells.
      iv. Apply the MUSCL-Hancock scheme on Finite Volume Cells of detected Spectral Cells and change fluxes between detected and non-detected Spectral Cells.

3.4 Implementation details and matrix formulation

Implementation is realized using two global grids. A grid is dedicated to hold Finite Volume representation and the other grid to hold Control Volume values. We emphasize that latter grid is small compared to the Finite Volume grid. When high-order time evolution is realized, these fluxes are also stored in a global array. This way, fluxes can be further modified if a Spectral Cell is detected. If a Spectral Cell is not detected but one neighbour is, it also needs to be projected on the Finite Volume grid again because of corrected fluxes at edges.

In the following we omit the time dependency in the Spectral representation and for the corresponding discrete values. By a translation, the following calculations can always be done from the Spectral Cell $(i_s, j_s, k_s) = (0, 0, 0)$. Thus we write $CV = CV_{i_s,j_s,k_s}$ and $FV = FV_{i_s,j_s,k_s}$. This allows to interpret triplets of indices (l, m, n) and (i, j, k) as local indices relative to this Spectral Cell.

Let us also note $U^S = \left(u^S_{q,r,s}\right)_{0\leq q,r,s\leq p-1}$ (resp. $U^{CV} = (u_{l,m,n})_{(l,m,n)\in CV}$ and $U^{FV} = (u_{i,j,k})_{(i,j,k)\in FV}$) the vector of coefficients of this polynomial (resp. the vector of values associated to Control Volume and the vector of values associated to the Finite Volume Cells).

Regarding the flux, we only consider a face in the X-direction, located at $x^{CV}_{l+1/2}$, as an example. We note $F^x_{l+1/2}$ the vector of flux integrals and $F^{x,pt}_{l+1/2} = \left(\mathcal{F}^x_{m,n}\left(x^{CV}_{l+1/2}\right)\right)_{0\leq m,n\leq p-1}$ the vector of flux points associated with the Lagrange interpolation 3.13.

3.4.1 Finite Volume Cells to Control Volumes

This step is a simple projection onto a coarser grid. Finite Volume Cells that belong to the same Control Volume simply add their content

$$\forall (l,m,n)\in CV,\quad u_{l,m,n} \overset{\text{def}}{=} \frac{1}{|\omega^{CV}_{l,m,n}|}\sum_{(i,j,k)\in FV_{l,m,n}} |\omega_{i,j,k}|u_{i,j,k}. \tag{3.22}$$

3.4.2 Control Volumes to Finite Volume Cells

This step is the counterpart of Finite Volume Cells to Control Volumes. For a valid Spectral Cell it consists in projecting the polynomial u^S representation onto the Finite Volume Cells.

$$\forall\,(i,j,k)\in FV,\quad u_{i,j,k} \overset{\text{def}}{=} \frac{1}{|\omega_{i,j,k}|}\iiint_{\omega_{i,j,k}} u^S(x,y,z)\mathrm{d}x\mathrm{d}y\mathrm{d}z. \tag{3.23}$$

This integral can be explicitly computed and only consists in a matrix-vector product, more details in the following.

From polynomial coefficients

We want to give a matrix formulation of Equation 3.23, i.e. to compute from the values associated with the Control Volumes to the values associated with the Finite Volume Cells, $U^{FV} = (u_{i,j,k})_{(i,j,k)\in FV}$. We have

$$\iiint_{\omega_{i,j,k}} u^S(x,y,z)\mathrm{d}x\mathrm{d}y\mathrm{d}z = \sum_{q,r,s} u^S_{q,r,s}\iiint_{\omega_{i,j,k}} x^q y^r z^s \mathrm{d}x\mathrm{d}y\mathrm{d}z \tag{3.24}$$

$$= |\omega_{i,j,k}|\sum_{q,r,s} M_{((i,j,k),(q,r,s))}U^S_{(q,r,s)} \tag{3.25}$$

in which we define the matrix M by

$$M_{((i,j,k),(q,r,s))} = \frac{1}{|\omega_{i,j,k}|}\iiint_{\omega_{i,j,k}} x^q y^r z^s \mathrm{d}x\mathrm{d}y\mathrm{d}z \tag{3.26}$$

The integral in the Equation 3.26 is only geometrical and thus only depends on the mesh. It is then straightforward that we have a matrix formulation

$$U^{FV} = MU^S \tag{3.27}$$

From Control Volumes

We need to define how to obtain the polynomial coefficients from the Control Volumes data. We start by computing the vector of coefficients U^S from the vector of Control Volumes U^{CV}

$$\frac{1}{|\omega^{CV}_{l,m,n}|}\iiint_{\omega^{CV}_{l,m,n}} u^S(x,y,z)\mathrm{d}x\mathrm{d}y\mathrm{d}z = u_{l,m,n} \tag{3.28}$$

Developing the polynomial u^S we obtain

$$\iiint_{\omega^{CV}_{l,m,n}} u^S(x,y,z)\mathrm{d}x\mathrm{d}y\mathrm{d}z = \sum_{q,r,s} u^S_{q,r,s}\iiint_{\omega^{CV}_{l,m,n}} x^q y^r z^s \mathrm{d}x\mathrm{d}y\mathrm{d}z \tag{3.29}$$

$$= |\omega^{CV}_{l,m,n}|\sum_{q,r,s} N_{((l,m,n),(q,r,s))}U^S_{(q,r,s)}, \tag{3.30}$$

with

$$N_{((l,m,n),(q,r,s))} = \frac{1}{|\omega^{CV}_{l,m,n}|}\iiint_{\omega^{CV}_{l,m,n}} x^q y^r z^s \mathrm{d}x\mathrm{d}y\mathrm{d}z. \tag{3.31}$$

Because there are as much coefficients as Control Volumes we can invert this system and obtain the polynomial coefficients with

$$U^S = N^{-1}U^{CV} \tag{3.32}$$

If we combine both results we are able to compute the values of the Finite Volume Cells from the values of Control Volumes without having to compute the polynomial coefficients

$$U^{FV} = MN^{-1}U^{CV}. \tag{3.33}$$

This formulation has the main advantage to be independent of the location of Spectral Cell (i_s, j_s, k_s) in the mesh $\mathcal{M}^S$. This implies that matrix MN^{-1} can be computed before the time loop on a Spectral Cell of reference.

3.4.3 Flux integral

In this section we aim at formulating flux integrals as a matrix-vector product using the flux points $F^{x,pt}_{l+1/2}$ computed at the center of Control Volumes. As it has been presented in Section 3.2.2 the integral of the polynomial flux reconstruction 3.13 writes

$$\begin{aligned} F^x_{l+1/2,m,n} &= \frac{1}{|\Gamma_{l+1/2,m,n}|}\iint_{\Gamma_{l+1/2,m,n}} \mathcal{F}^x(x^{CV}_{l+1/2},y,z)\mathrm{d}y\mathrm{d}z, \\ \Gamma_{l+1/2,m,n} &= \overline{\omega^{CV}_{l,m,n}}\cap\overline{\omega^{CV}_{l+1,m,n}}, \\ \mathcal{F}^x(x,y,z) &= \sum_{m,n}\mathcal{F}^x_{m,n}(x)L_{m,n}(y,z), \\ L_{m,n}(y,z) &= \prod_{(j,k)\neq(m,n)}\frac{y-y^{CV}_j}{y^{CV}_m-y^{CV}_j}\frac{z-z^{CV}_k}{z^{CV}_n-z^{CV}_k}. \end{aligned} \tag{3.34}$$

To have a matrix-vector product we first find the polynomial coefficients of $\mathcal{F}^x\left(x^{CV}_{l+1/2},y,z\right) = \sum_{0\leq r,s\leq p-1} c_{r,s}y^r z^s$ by solving the following linear system

$$N^f C = F^{x,pt}_{l+1/2}, \tag{3.35}$$

in which $N^f_{((m,n),(r,s))} = \left(y_m^{CV}\right)^r \left(z_n^{CV}\right)^s$ and $C_{r,s} = c_{r,s}$. As in the previous section it results that

$$\begin{aligned} F^x_{l+1/2} &= M^f {N^f}^{-1} F^{x,pt}_{l+1/2}, \\ M^f_{((m,n)(r,s))} &= \frac{1}{|\Gamma_{l+1/2,m,n}|} \iint_{\Gamma_{l+1/2,m,n}} y^r z^s \mathrm{d}y\mathrm{d}z. \end{aligned} \tag{3.36}$$

3.5 Numerical experiments

In this section we present numerical experiments both in term of precision and efficiency. The tests are done with order ranging from 1 to 4. For a given variable ϕ, the L^1 error on ϕ is measured as

$$\left\|\phi^{num} - \phi^{sol}\right\|_{L^1} = \frac{1}{N_x N_y} \sum_{i,j} |\phi^{num}_{i,j} - \bar{\phi}^{sol}_{i,j}|$$

in which

$$\bar{\phi}^{sol}_{i,j} = \frac{1}{|C_{i,j}|} \int_{C_{i,j}} \phi(x,y)\mathrm{d}x\mathrm{d}y$$

In the following we refer to "CV perf" as the number of Control Volumes updated per second whereas "FV perf" refers to the number of Finite Volume Cells updated per second which are standard metrics in grid based algorithms. Of course one has to be careful when comparing such numbers as they depend on the algorithm, their implementation and the hardware architecture. In our case all performance numbers are presented on a single Intel Sandy-Bridge E5–2670 processor. The first order simulation is implemented with the SVM algorithm which may lower performances due to unnecessary computations.
We chose to test our numerical method on the well-known adiabatic Euler system and selected test cases showing different aspects of numerical schemes such as transport, acoustic, smooth, discontinuous or high Mach flows.

3.5.1 Adiabatic Euler system

The adiabatic Euler system is used to model dynamics of inviscid fluids, see Godlewski and Raviart 1991; Toro 2009 for basic properties on this system. It is a system of conservation laws for the mass density ρ, the momentum density $\boldsymbol{m} = \rho\boldsymbol{u}$ and the energy density ρE

$$\begin{aligned} \frac{\partial}{\partial t}\rho + \operatorname{div}(\rho\boldsymbol{u}) &= 0 \\ \frac{\partial}{\partial t}(\rho\boldsymbol{u}) + \operatorname{div}(\rho\boldsymbol{u}\otimes\boldsymbol{u} + p\mathbb{I}) &= \mathbf{0} \\ \frac{\partial}{\partial t}(\rho E) + \operatorname{div}((\rho E + p)\boldsymbol{u}) &= 0 \\ \rho E = \rho e + \frac{1}{2}\rho\boldsymbol{u}^2, \quad p &= p^{\mathrm{EOS}}(\rho, e) \end{aligned} \tag{3.37}$$

where $\boldsymbol{u}$ is the material velocity, p^{EOS} is a given pressure law and e is the specific internal energy of the fluid. Bold expressions represent vectors in 2 or 3D, product $\boldsymbol{u}\otimes\boldsymbol{u}$ denotes tensor product, i.e. $(\boldsymbol{u}\otimes\boldsymbol{u})_{i,j} = u_i u_j$ and $\mathbb{I}$ is the identity matrix in 3D. The pressure law in the following is chosen to be a perfect gas

$$p^{\mathrm{EOS}}(\rho, e) = (\gamma - 1)\rho e \tag{3.38}$$

spectral	N_x	L^1 error on ρ	effective order	CV perf	FV perf
40	40	2.2213×10^{-1}	—	3.7	3.7
80	80	1.3576×10^{-1}	0.7104	4.2	4.2
160	160	7.5613×10^{-2}	0.8444	4.2	4.2
320	320	3.9979×10^{-2}	0.9194	4.1	4.1
640	640	2.0566×10^{-2}	0.9590	3.9	3.9
1280	1280	1.0432×10^{-2}	0.9793	3.9	3.9

Table 3.2: Summary of L^1 errors using SVM (1) on the density variable and corresponding effective orders obtained on the advection of sinusoide.

where $\gamma > 1$ is the adiabatic index of the gas.
Even though the numerical scheme in Section 3.3 is presented for a scalar conservation law it can be directly applied to a system of conservation by using the same procedure for each equation in the system.

3.5.2 Smooth transport test case

This test consists in the advection of a sinusoide density profile at constant velocity. Initial conditions are given at $t = 0$ by, $\forall x, y \in [0, 1]$

$$\rho(x, y) = 2 + \sin(2\pi x)\sin(2\pi y)$$
$$u(x, y) = 10, \quad v(x, y) = 10$$
$$p(x, y) = 1$$

This test is used for measuring the effective order of the spectral volume method of order p when there is no limitation procedure. Indeed this test case is smooth which makes it well-suited for a convergence study. We report L^1 errors on density variable in tables 3.3, 3.4 and 3.5 for orders between 2 and 4 and for the same number of Finite Volume Cells. Effective order tends to match formal accuracy by lower values when the number of Spectral Cells increases. Figures 3.3a and 3.3b summarize the results from the tables. We notice that the plots of orders 2 and 3 cross at 160 fine cells. This can be understood by the way the numerical scheme is constructed. Assuming that the error decreases as $C\Delta x^p$ for a smooth solution, the number of Control Volumes specifies the order p of convergence but the number of fine cells inside a Control Volumes specifies the constant C. The more fine cells are aggregated to form a Control Volume the bigger C is.
We also show performance results in the tables 3.2, 3.3, 3.4 and 3.5. In a smooth test case, with MOOD disabled, we only measure performance relative to the Spectral Volume method stage. We first notice a decrease in performance when going to higher orders. This is expected as more computation is done per Control Volume. However for a given error we can see that a high-order scheme helps in allowing to use coarser grids hence less time iterations. We also give FV performances as an indicator to the maximal performance that can be achieved when when the fallback scheme is enabled. Of course for first and second order, there is no benefit as Control Volumes exactly map Finite Volume Cells. However we see a boost of performances in the third and fourth orders.

spectral	N_x	L^1 error on ρ	effective order	CV perf	FV perf
20	40	1.3380×10^{-2}	—	2.7	2.7
40	80	3.3680×10^{-3}	1.9902	3	3
80	160	8.4528×10^{-4}	1.9944	3.1	3.1
160	320	2.1179×10^{-4}	1.9968	3	3
320	640	5.3012×10^{-5}	1.9983	2.8	2.8
640	1280	1.3262×10^{-5}	1.9991	2.5	2.5

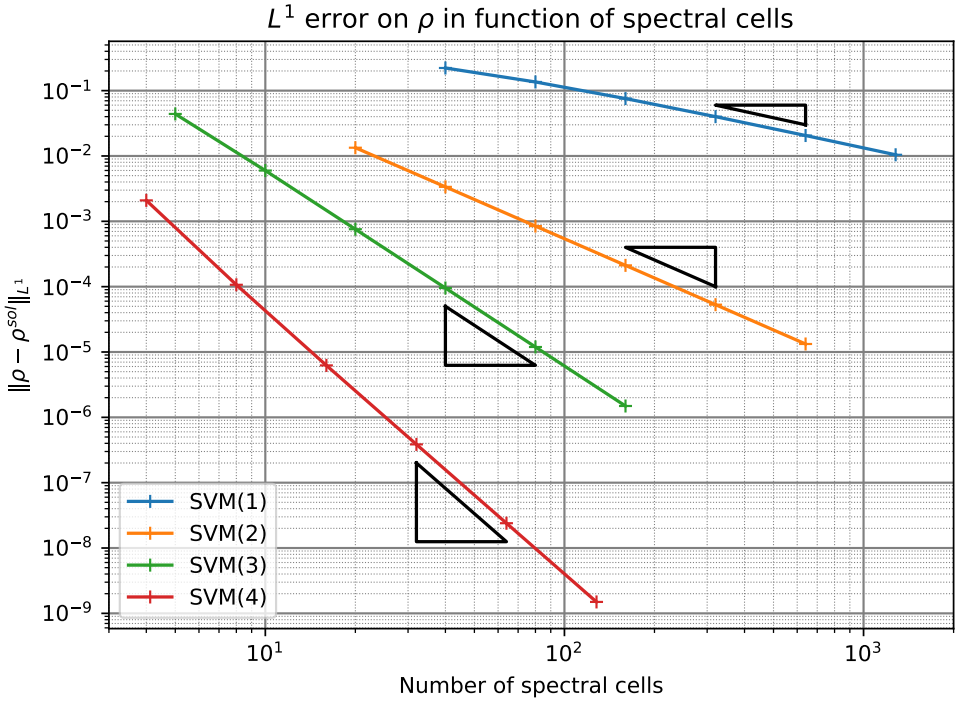
Table 3.3: Summary of L^1 errors using SVM (2) on the density variable and corresponding effective orders obtained on the advection of sinusoide.

spectral	N_x	L^1 error on ρ	effective order	CV perf	FV perf
5	40	4.3987×10^{-2}	—	1.0	7.2
10	80	5.9800×10^{-3}	2.8789	1.3	9.0
20	160	7.5716×10^{-4}	2.9815	1.3	9.1
40	320	9.5360×10^{-5}	2.9891	1.3	9.1
80	640	1.1926×10^{-5}	2.9993	1.2	8.8
160	1280	1.4913×10^{-6}	2.9994	1.2	8.5

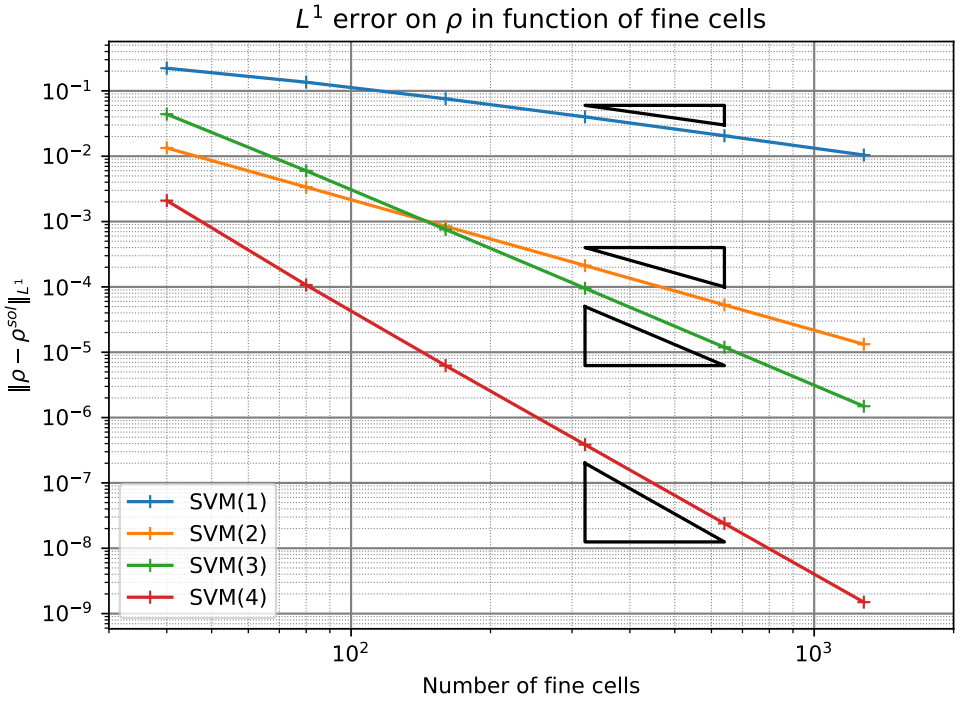
Table 3.4: Summary of L^1 errors using SVM (3) on the density variable and corresponding effective orders obtained on the advection of sinusoide.

spectral	N_x	L^1 error on ρ	effective order	CV perf	FV perf
4	40	2.0912×10^{-3}	—	0.67	4.2
8	80	1.0682×10^{-4}	4.2910	0.85	5.3
16	160	6.2350×10^{-6}	4.0987	0.94	5.9
32	320	3.8533×10^{-7}	4.0162	0.92	5.8
64	640	2.3968×10^{-8}	4.0069	0.9	5.6
128	1280	1.4963×10^{-9}	4.0017	0.87	5.5

Table 3.5: Summary of L^1 errors using SVM (4) on the density variable and corresponding effective orders obtained on the advection of sinusoide.



(a) L^1 errors in function of the number of Spectral Cells.



(b) L^1 errors in function of the number of Spectral Cells.

Figure 3.3: The convergence rates of the Spectral Volume Method for the advection of a sinusoide test case.

3.5.3 Discontinuous transport test case

This test case is a transport test with a contact discontinuity. This test is meant to measure the smearing of the discontinuity in time and to detect spurious oscillations. Even in the case of the Godunov scheme that involves the exact solution of the Riemann problem, the contact discontinuity will be smeared by the averaging steps. We consider the initial conditions

$$\rho(x,y) = \begin{cases} 1 & \text{if } 0.1 < x < 0.3, \quad 0.1 < y < 0.3, \\ 0.1 & \text{otherwise,} \end{cases}$$
$$p(x,y) = 1,$$
$$u(x,y) = 1, \quad v(x,y) = 1.$$

The initialization is performed on the fine grid in order to avoid early smearing of the interface. We use periodic boundary conditions.

Results are shown in Figure 3.4. All four simulations involve the same number of fine cells 320, which implies that there are 320, 160, 80, 40 Spectral Cells for orders 1, 2, 3 and 4 respectively. Results are plotted after one advection tour. Top left quadrant of Figure 3.4a shows the result of the standard finite volume scheme at order 1. The contact discontinuity is highly smeared up to the point of loosing track of the initial shape. Other quadrants of Figures 3.4a show results at orders 2, 3 and 4 respectively. We notice that at order 3, there is a larger smearing of the interface compared to the second order case. This can be related to the convergence study in the sinusoide test case 3.5.2 and the observation of a larger error at order 3 compared to order 2 when the mesh is coarse. The more the discontinuity is smeared in time, the less cells are detected.

Figure 3.4b shows the profile of density along the first diagonal at the final time for all orders. The exact solution is also represented in dash grey line. First we notice no spurious oscillations due to the use of the fallback scheme that also involves a slope limiter. Even though SVM (2) to SVM (4) use the same fallback scheme in the discontinuous region we notice a slightly less smearing of the discontinuity.

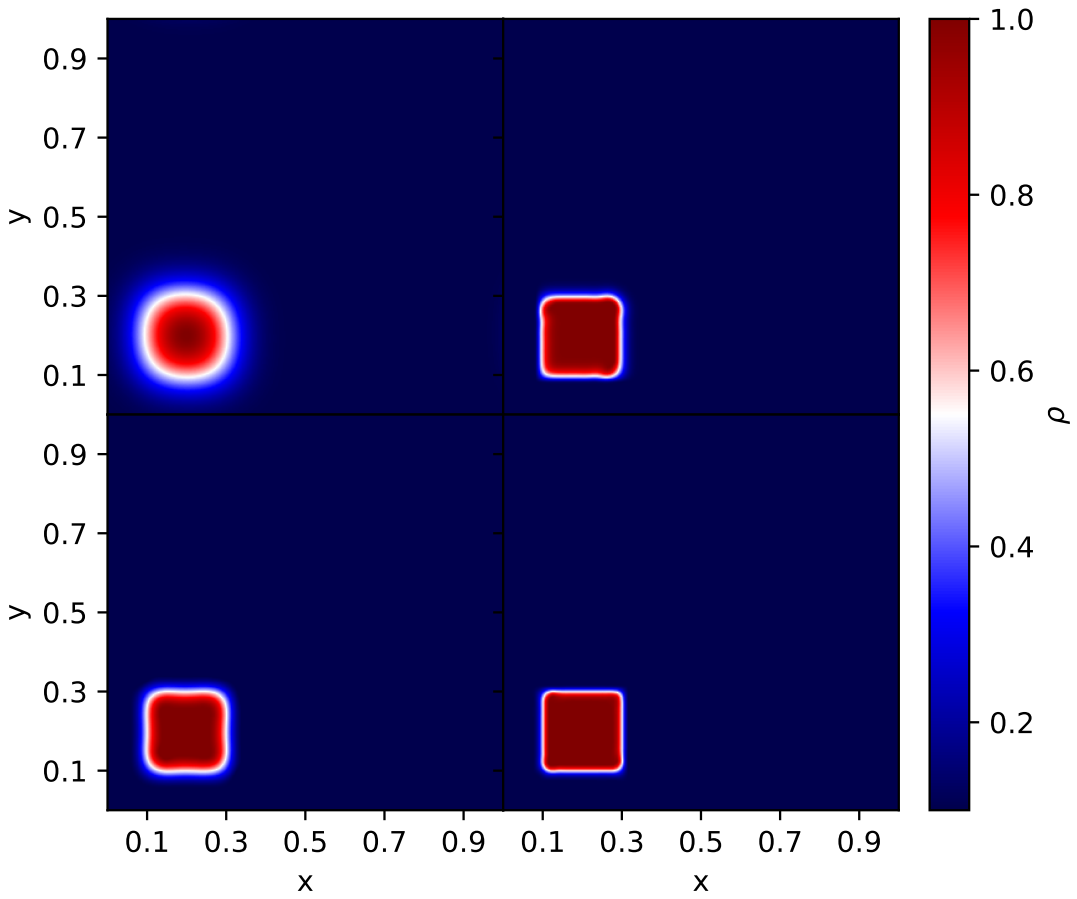
3.5.4 Sod shock tube test case

The Sod test case is a Riemann problem where initial conditions are given by following two states

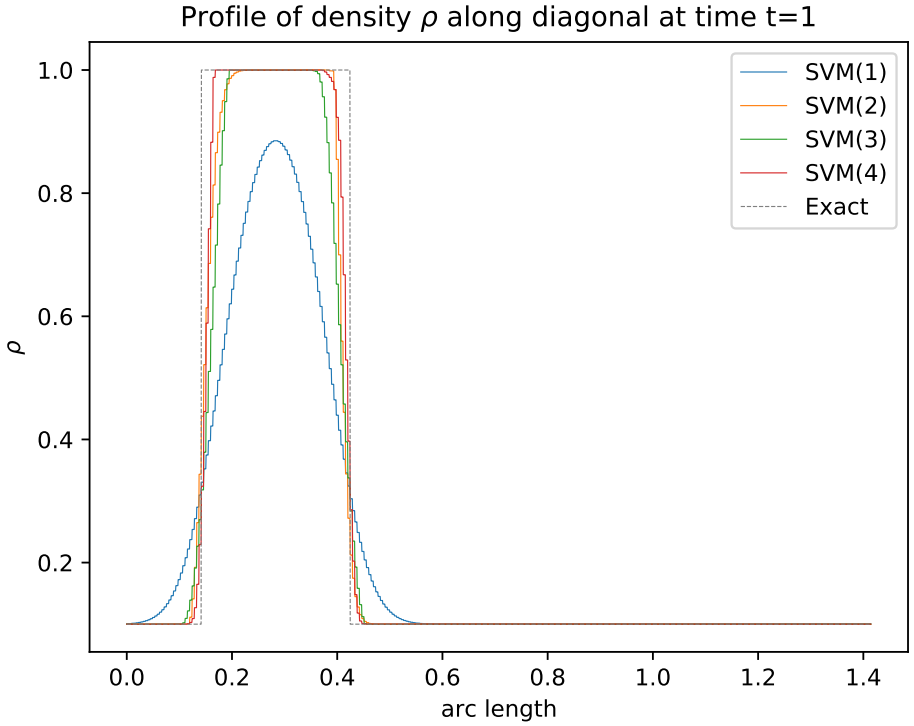
$$\begin{aligned} \rho_L = 1, \quad p_L = 1, \quad u_L = 0 \\ \rho_R = 0.125, \quad p_R = 0.1, \quad u_R = 0 \end{aligned} \tag{3.39}$$

The exact solution is composed of three elementary waves, an expansion wave, a contact wave and a shock wave. As the Spectral Volume scheme is only used in smooth regions, we cannot expect much improvement issued from high order method in this test that involves two discontinuities. Figure 3.5 shows result at time $t = 0.2$ for a fixed Finite Volume mesh of $N_x = 320$ and different orders. On the left panel we display density profiles and on the right panel detected bad cells. We see that detection mainly happens near discontinuities, or at a loss of regularity near the ending of the expansion wave. At order 3 and 4 we notice that a lot of Spectral Cells are detected. It is expected because we are at low resolution.

We do not observe a major improvement with methods of order higher than 2. However for the shock wave, we notice that it can be localized up to the size of a Finite Volume Cell and not to the size of a Spectral Cell. Small oscillations near the contact discontinuity are related to the slope limiter 3.20.



(a) Density color maps for different orders: 1, 2, 3 and 4 correspond respectively to the top left, top right, bottom left and bottom right quadrants. We can see a major improvement between the orders 1 and 2, however from the order 2 the smearing is not reduced. This is due to the common fallback MUSCL-Hancock scheme, that controls the smearing.



(b) Density profiles along the diagonal for different orders.

Figure 3.4: Advection of a square simulations, at constant velocity.

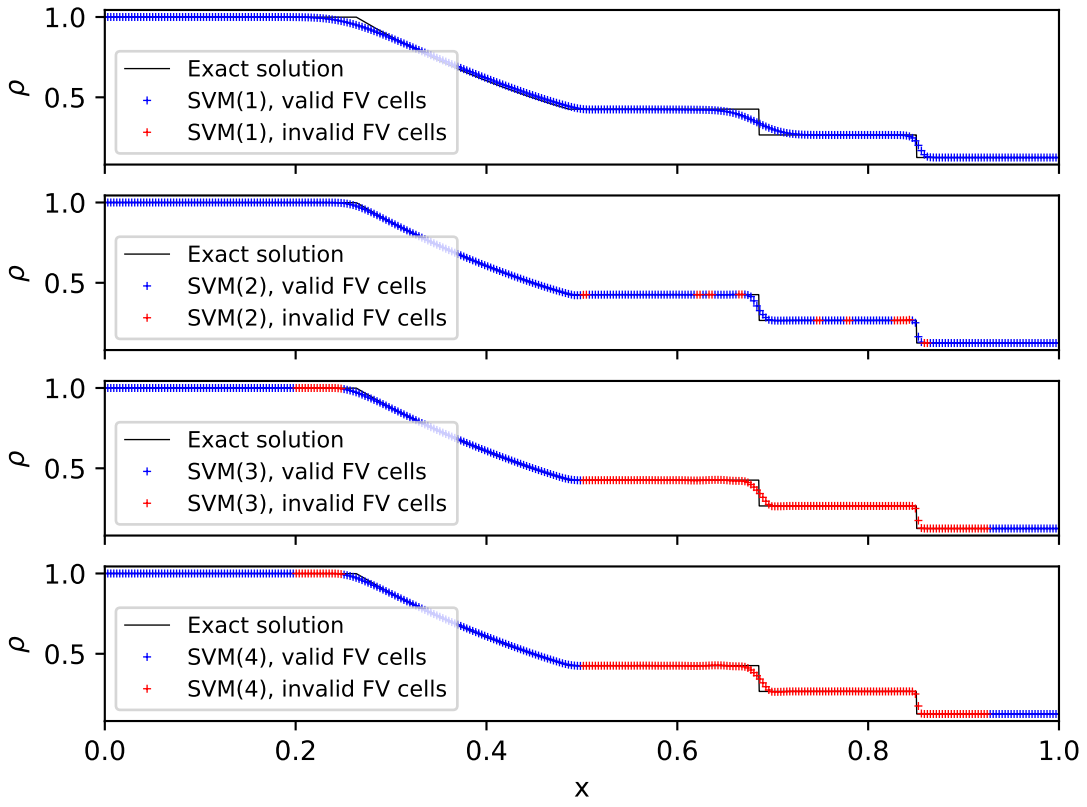


Figure 3.5: Sod shock tube simulations at time $t = 0.2$ and for a fixed Finite Volume grid of $N_x = 320$ cells. We show density profiles for different orders. The detected cells are colored in red whereas the valid, spectral, cells are colored in blue.

spectral	N_x	L^1 error on ρ	effective order
40	40	2.3678×10^{-2}	—
80	80	1.9570×10^{-2}	0.2981
160	160	1.4387×10^{-2}	0.4522
320	320	9.2509×10^{-3}	0.6422
640	640	5.3656×10^{-3}	0.7887
1280	1280	2.9139×10^{-3}	0.8823

Table 3.6: Summary of L^1 errors using SVM (1) on the density variable and corresponding effective orders obtained for the advection of the isentropic vortex.

3.5.5 Isentropic vortex test case

The isentropic vortex test involves the transport of a vortex by the velocity field $(u_0, v_0) = (1, 1)$. The vortex center is located at $(x_c, y_c) = (5, 5)$ in the Cartesian domain $[0, 10]^2$. The solution is given by

$$\begin{aligned} \rho(x,y) &= \left(1 - \frac{(\gamma-1)\beta^2}{8\gamma\pi^2}\exp\left(\frac{1}{2}(1-r^2)\right)\right)^{\frac{1}{\gamma-1}} \\ u(x,y) &= u_0 - \frac{\beta}{2\pi}\exp\left(\frac{1}{2}\left(1-r^2\right)\right)(y-y_c) \\ v(x,y) &= v_0 + \frac{\beta}{2\pi}\exp\left(\frac{1}{2}\left(1-r^2\right)\right)(x-x_c) \\ p(x,y) &= \rho(x,y)^\gamma \end{aligned} \tag{3.40}$$

We use periodic boundary conditions.
The initialization of the Finite Volume Cells is achieved through the initialization of the spectral grid using conservative variables. It has been noticed that the initialization of the Finite Volume grid with high order quadrature can produce errors of the order of 10^{-8}. This has resulted in a failure of the measurement of the order of convergence at time $t = 10$. However the order of convergence can be shown to be recovered on a longer time scale. In consequence, it has been decided to initialize conservative variables on the spectral grid in order to avoid this spurious error.
Tables 3.6, 3.7, 3.8 and 3.9 give results for different meshes. For orders up to 3, we obtain a convergence rate close to the formal order. However in the case of SVM (4) we see a higher convergence rate when increasing number of cells that has also been observed in the case of discontinuous Galerkin method see Schaal et al. 2015.

3.5.6 Kelvin-Helmholtz instability test case

The Kelving-Helmholtz instability test simulates the development of an instability at the interface between two layers of different densities that undergoes a velocity shear. This density contrast can have different origins such as a gradient of temperature, a gradient of mean molecular weight, a phase difference between gas and liquid. Because this flow is unstable, a small perturbation can break the stationary fluid layering state and develop vortices. In the context of the Euler equations, the density ratio is due to a temperature gradient. Moreover because there

spectral	N_x	L^1 error on ρ	effective order
20	40	6.7078×10^{-3}	—
40	80	2.0151×10^{-3}	1.7350
80	160	5.2340×10^{-4}	1.9432
160	320	1.3050×10^{-4}	2.0055
320	640	3.2454×10^{-5}	2.0076
640	1280	8.0906×10^{-6}	2.0041

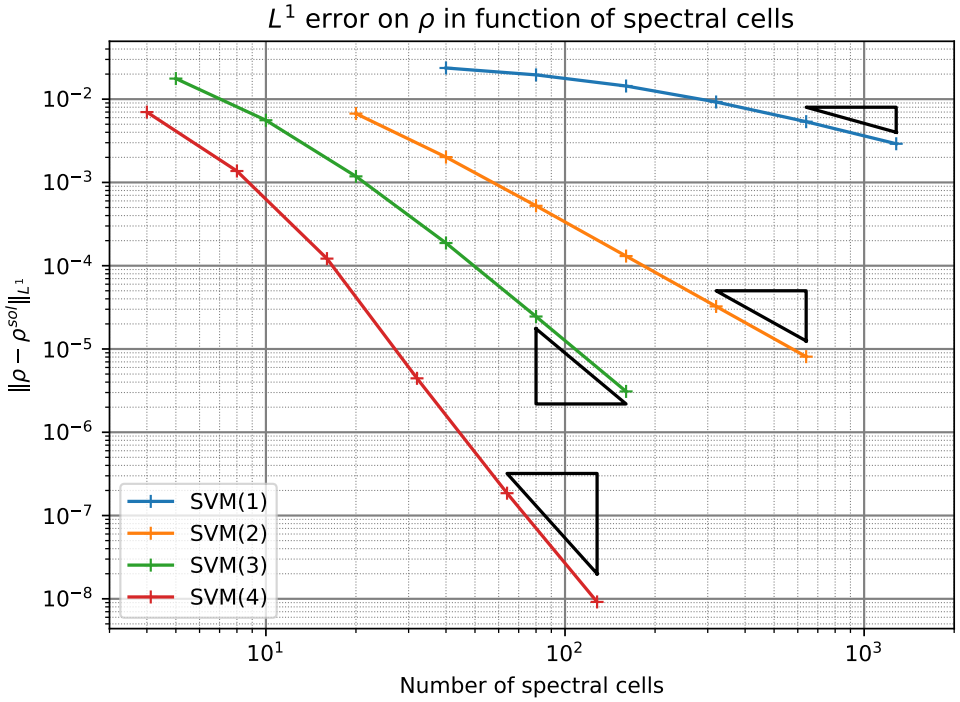
Table 3.7: Summary of L^1 errors using SVM (2) on the density variable and corresponding effective orders obtained for the advection of the isentropic vortex.

spectral	N_x	L^1 error on ρ	effective order
5	40	1.7665×10^{-2}	—
10	80	5.5822×10^{-3}	1.6620
20	160	1.1825×10^{-3}	2.2390
40	320	1.8721×10^{-4}	2.6592
80	640	2.4543×10^{-5}	2.9313
160	1280	3.0931×10^{-6}	2.9882

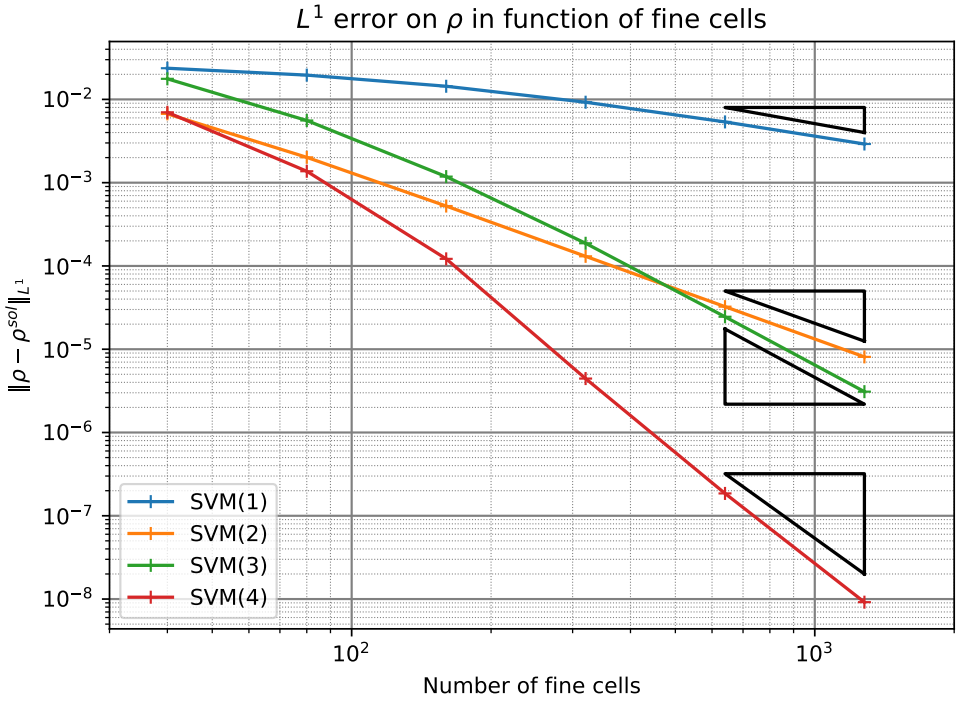
Table 3.8: Summary of L^1 errors using SVM (3) on the density variable and corresponding effective orders obtained for the advection of the isentropic vortex.

spectral	N_x	L^1 error on ρ	effective order
4	40	6.9975×10^{-3}	—
8	80	1.3714×10^{-3}	2.3512
16	160	1.2167×10^{-4}	3.4945
32	320	4.4428×10^{-6}	4.7754
64	640	1.8553×10^{-7}	4.5817
128	1280	9.1754×10^{-9}	4.3377

Table 3.9: Summary of L^1 errors using SVM (4) on the density variable and corresponding effective orders obtained for the advection of the isentropic vortex.



(a) L^1 errors in function of the number of Spectral Cells.



(b) L^1 errors in function of the number of Finite Volume Cells.

Figure 3.6: Convergence rates of the Spectral Volume Method for the isentropic vortex test case.

parameters	values
δ	0.02
(y_1, y_2)	(0.25, 0.75)
(ρ_1, ρ_2)	(2, 1)
(u_x^1, u_x^2)	(-0.5, 0.5)
u_y^0	0.1

Table 3.10: Parameter values for the Kelvin-Helmholtz instability.

is no temperature diffusion in the Euler equations, a temperature jump should be indefinitely maintained. The interface between the two layers keeps growing in time within the computational domain.
The initial situation is given by a small regularization version defined as follows, for $x, y \in [0, 1]$

$$\begin{aligned} \rho(x,y) &= \rho_1 + (\rho_2 - \rho_1)\Theta(y; y_1, y_2) \\ u_x(x,y) &= u_x^1 + (u_x^2 - u_x^1)\Theta(y; y_1, y_2) \\ u_y(x,y) &= u_y^0 \sin(2\pi x) \\ \Theta(y; y_1, y_2) &= \left(1 + \exp\left(2\frac{y - y_1}{\delta}\right)\right)^{-1}\left(1 + \exp\left(2\frac{y_2 - y}{\delta}\right)\right)^{-1} \end{aligned} \tag{3.41}$$

with parameter values displayed in Table 3.10 and periodic boundary conditions.
A simulation at resolution 1280^2 was performed and the result is shown in Figure 3.7 for different times. We can see a well-defined interface in the first two snapshots that develops two spirals. In the last two snapshots, other Kelvin-Helmholtz instability modes have developed at the interface leading to the mixing of the two layers. Figure 3.8 shows the time evolution of valid Spectral Cells. As we can see, from time $t = 1.2$, the number of valid Spectral Cells keeps decreasing. This time corresponds to the development of new Kelvin-Helmholtz instability modes as shown in figure 3.8. It should be emphasized that no diffusion process is at stake, hence there is no limitation in the development of new Kelvin-Helmholtz modes. Such a process would define a length scale which, if resolved, could limit the use of the fallback scheme.

3.5.7 Mach 80 jet test case

We now turn to a more demanding test coming from the astrophysics field Ha et al. 2005; Gardner and Dwyer 2009. It consists in a high mach jet flow entering into a lower density material at rest. High mach flows are demanding for numerical schemes when solving the Euler system because of the total energy conservation. In these flows, the total energy is dominated by kinetic energy which make it difficult to be accurate for internal energy and thus may lead to negative energies. The jet enters with the following characteristics

$$\rho = 5, \quad p = 0.4127, \quad u = 30, \tag{3.42}$$

into a hotter material at rest where

$$\rho = 0.5, \quad p = 0.4127, \quad u = 0. \tag{3.43}$$

Both materials have the same adiabatic index $\gamma = 1.666$. It is straightforward to compute the Mach number 80 for the jet. Domain of study is set to $[0, 2] \times [-0.5, 0.5]$. Jet enters at $(0, 0)$ and

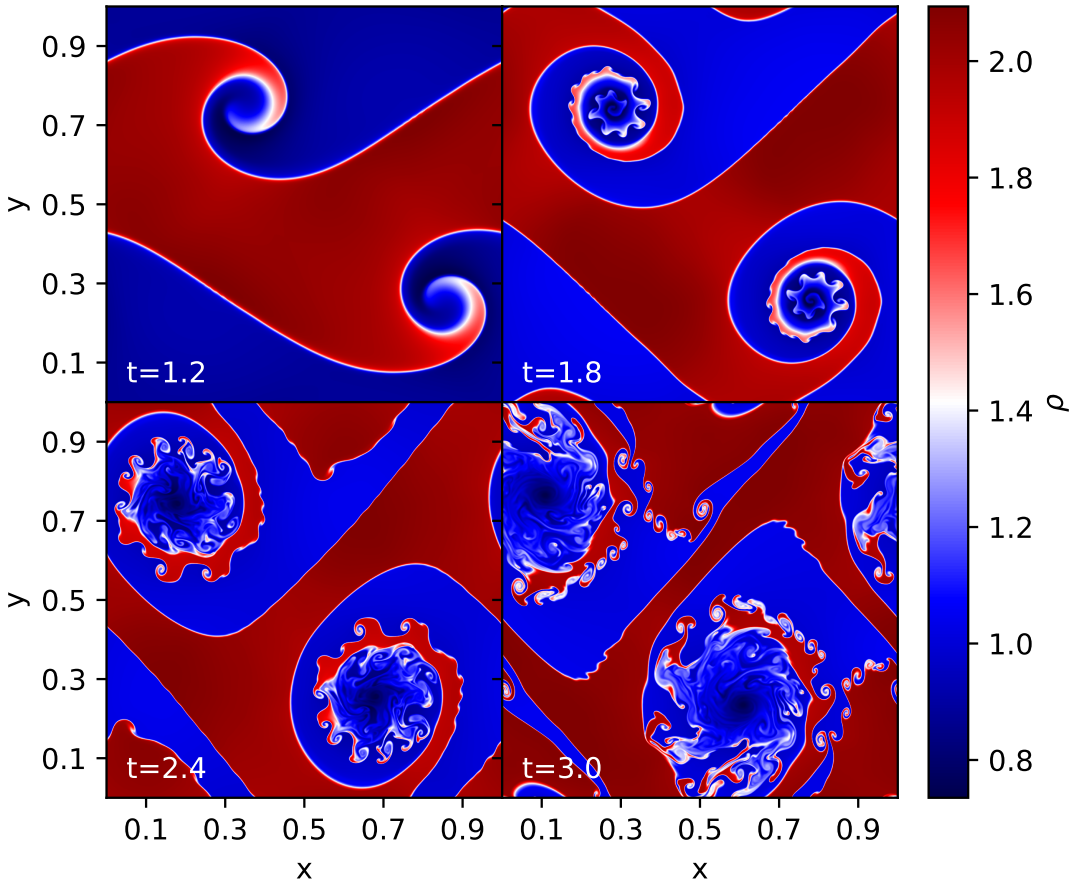


Figure 3.7: Kelvin-Helmholtz instability test case. We show four snapshots at different times $t = 1.2$, $t = 1.8$, $t = 2.4$ and $t_f = 3$.

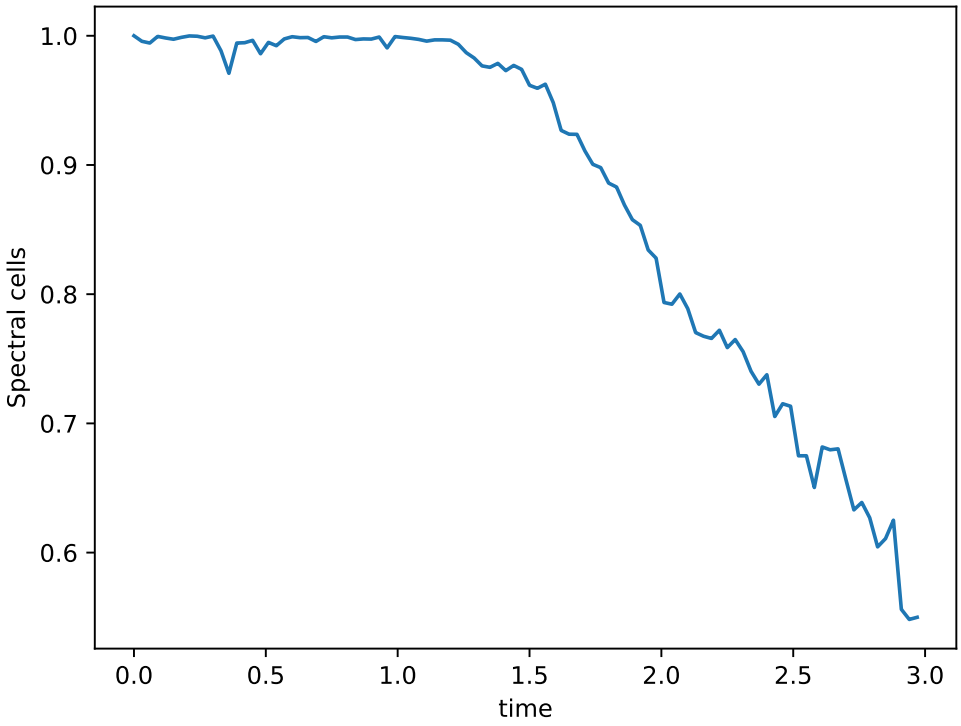


Figure 3.8: Time evolution of the fraction of valid Spectral Cells. We can see that due to interfaces instabilities, the number of valid Spectral Cells keeps decreasing.

has a width of 0.1. In order to avoid to rely on a boundary related problem, we set only the first layer of Finite Volume Cells to jet values and we set Neumann boundary conditions for all faces. We observed that using only negative energy and density as MOOD detection criteria is not enough. Indeed it may lead to cells in a admissible but still non-physical state and thus to a near zero time step. We show results on Figures 3.9 and 3.10 at time $t = 0.7$. Figure 3.9 is a low-resolution simulation on a Finite Volume mesh 500×250 similar to results given in Ha et al. 2005. We notice a numerical shock instability at the head of the jet. It recalls a carbuncle instability, see Quirk 1994 which is usually observed for stationary shocks aligned to the grid. Moreover we can see that because of the low resolution and large Spectral Cells (fourth order), there is a large ratio of cells using the fallback scheme.
We increased the resolution up to 3200×1600, see Figure 3.10 where we can see a Kelvin-Helmholtz instability developing near the head of the jet. This simulation with a grid spacing of $\Delta x = 6.2510^{-4}$ was performed at a mean speed of 3.7M Finite Volume Cells update per second.The performance is no longer constant but depends on the number of detected cells — reaching 12% of total Spectral Cells at the end of the simulation, less than in the low resolution simulation. This number is similar to a sequential second order MUSCL-Hancock scheme on a regular grid.

Conclusions

In this chapter we have presented a new hybrid Spectral Volume and Finite Volume method. It aims at accelerating computation using high-order schemes in smooth regions to benefit from efficient p-convergence and standard Finite Volume schemes in discontinuous regions to benefit from standard h-convergence. We have shown results validating the order of convergence and a good behavior in presence of discontinuities. We have obtained good performance results near the performances of the second order MUSCL-Hancock scheme for third and fourth orders numerical schemes. Future works on the optimization of the scheme might lead to better performances. We also point out that a factor 2 can be easily obtained by simply switching from RK4 to RK2 without much impact on the simulation results for the KH and jet tests.

To fully take advantage of new supercomputers, numerical schemes have to present high degree of parallelism. Spectral schemes are known to ease data locality thanks to Spectral Cells. In the case of regular meshes numerical schemes usually have data parallelism. Thus domain decomposition is simple and there is no load-balancing issue. In our case the use of two schemes may break this load-balancing. Indeed if only one domain presents discontinuities, only this domain needs to use a fallback scheme, thus extra computation is needed only for some domains.

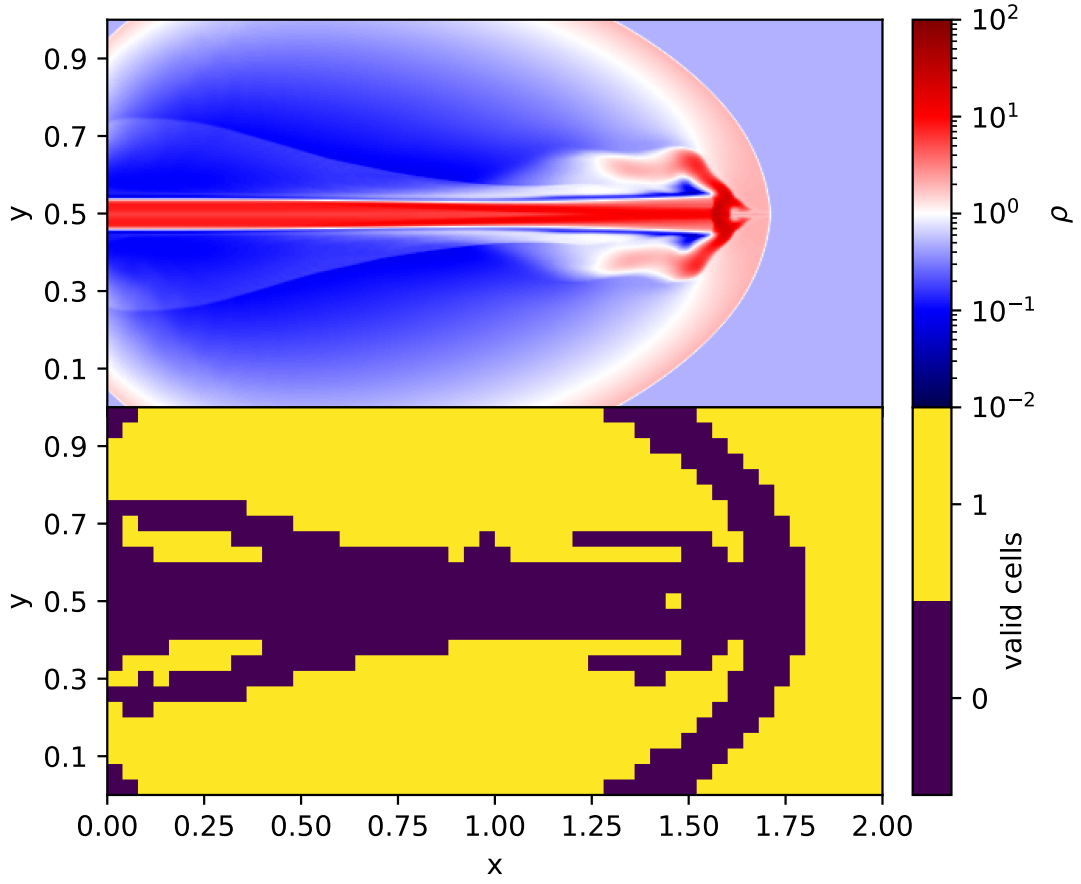


Figure 3.9: Snapshot of a jet simulation at Mach 80 with SVM (4) at a Finite Volume resolution 500×250. The top panel shows the density in a log scale. The bottom panel shows valid cells when the value is 1, i.e. when the update is done with the spectral scheme. We see that the fallback scheme is essentially enabled near discontinuities.

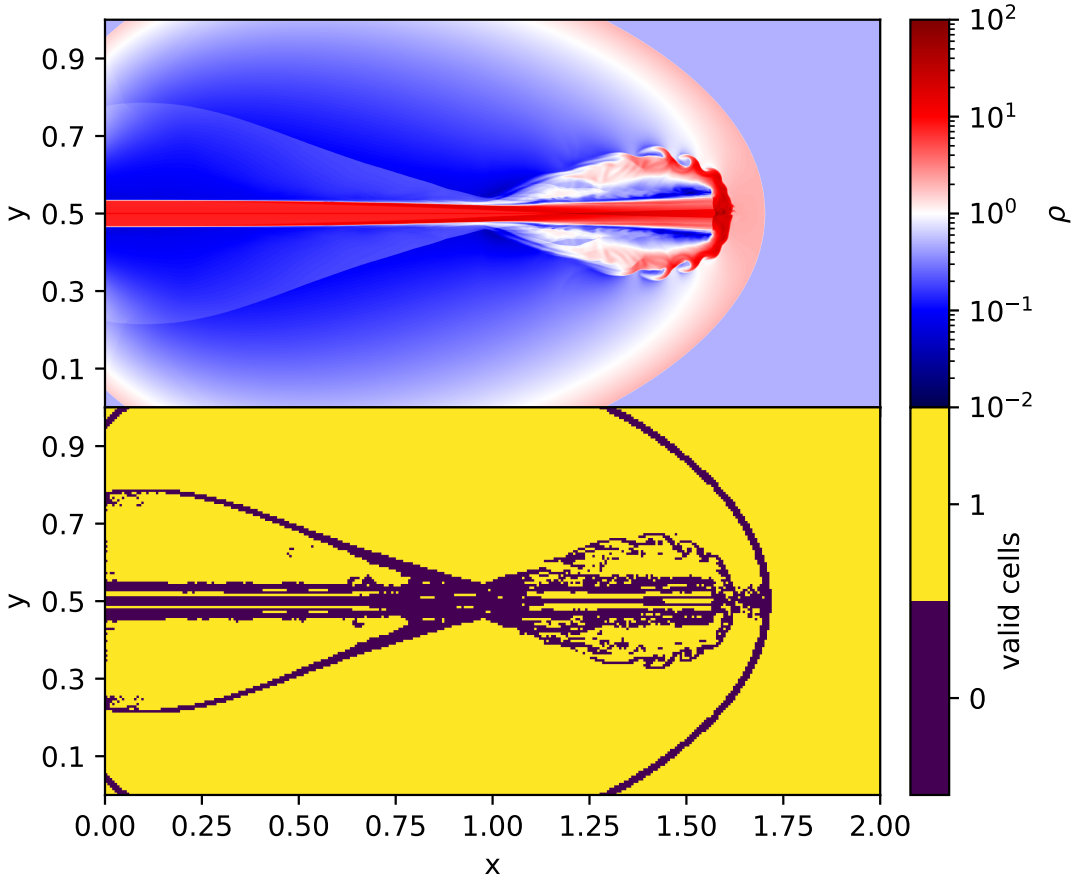


Figure 3.10: Snapshot of a jet simulation at Mach 80 with SVM (4) at a Finite Volume resolution 3200×1600. The top panel shows the density in a log scale. The bottom panel shows valid cells when the value is 1, i.e. when the update is done with the spectral scheme. We see that the fallback scheme is essentially enabled near discontinuities.

Appendix

3.A Extension to source term

One can be interested to study hyperbolic conservation laws involving source terms, also referred to as balance laws, Equation 3.1 now takes the following form

$$\frac{\partial}{\partial t}u + \operatorname{div}\left(\boldsymbol{f}(u)\right) = s(u) \tag{3.44}$$

In integral form over a Spectral Cell, keeping the notations of Section 3.2, we have

$$\begin{aligned} u_{l,m,n}(t) &= \frac{1}{|\omega^{CV}_{l,m,n}|}\iiint_{\omega^{CV}_{l,m,n}} u^{S}_{i_s,j_s,k_s}(t,\boldsymbol{x})\mathrm{d}\boldsymbol{x}, \\ s_{l,m,n}(t) &= \frac{1}{|\omega^{CV}_{l,m,n}|}\iiint_{\omega^{CV}_{l,m,n}} s(u^{S}_{i_s,j_s,k_s}(t,\boldsymbol{x}))\mathrm{d}\boldsymbol{x}, \\ \frac{\mathrm{d}}{\mathrm{d}t}u_{l,m,n}(t) + \frac{1}{|\omega^{CV}_{l,m,n}|}&\iint_{\partial\omega^{CV}_{l,m,n}} \boldsymbol{\mathcal{F}}(t,\boldsymbol{x})\cdot\boldsymbol{n}\left(\boldsymbol{x}\right)\mathrm{d}\sigma(\boldsymbol{x}) = s_{l,m,n}. \end{aligned} \tag{3.45}$$

The source term $s_{l,m,n}$ is simply approximated by

$$s_{l,m,n}(t) \approx s\left(u_{l,m,n}(t)\right) \tag{3.46}$$

We use a Runge-Kutta time integrator as in the conservative case. In the MOOD strategy, if the Spectral Cell is detected, we use the MUSCL-Hancock numerical scheme on the Finite Volume Cells with a source term that is evaluated by the unique value in the cell.

3.A.1 Rayleigh-Taylor instability

We now consider the Euler system, see Section 3.5.1, with gravity

$$\begin{aligned} \frac{\partial}{\partial t}\rho + \operatorname{div}\left(\rho\boldsymbol{u}\right) &= 0 \\ \frac{\partial}{\partial t}\left(\rho\boldsymbol{u}\right) + \operatorname{div}\left(\rho\boldsymbol{u}\otimes\boldsymbol{u} + p\mathbb{I}\right) &= \rho\boldsymbol{g} \\ \frac{\partial}{\partial t}\left(\rho E\right) + \operatorname{div}\left(\left(\rho E + p\right)\boldsymbol{u}\right) &= \rho\boldsymbol{u}\cdot\boldsymbol{g} \\ \rho E = \rho e + \frac{1}{2}\rho\boldsymbol{u}^2, &\quad p = p^{\mathrm{EOS}}\left(\rho, e\right) \end{aligned} \tag{3.47}$$

The gravity source term has the particularity of being linear in conservative variables so that the approximation in the previous section is exact.

A standard test case is the Rayleigh-Taylor interface instability. It is defined as a dense layer over a light layer at equilibrium. The initial state is given in a box $[-0.25, 0.25] \times [-0.75, 0.75]$ by

$$\begin{aligned} \rho(x,y) &= \begin{cases} 2 \text{ if } y > 0, \\ 1 \text{ else,} \end{cases} \\ p(x,y) &= p_0 + \rho g_y y, \\ u_y(x,y) &= \frac{A}{4}\left(1+\cos\left(2\pi\frac{x}{L_x}\right)\right)\left(1+\cos\left(2\pi\frac{y}{L_y}\right)\right), \\ u_x(x,y) &= 0, \end{aligned} \tag{3.48}$$

with values $p_0 = 2.5$, $g_y = -0.1$ and $A = 0.01$. A velocity perturbation is added so that initial equilibrium cannot be sustained.
Results of simulation are given in Figure 3.11 at resolution 500×1500. We can see dense layer going down and light layer going up. As in the test case 3.5.6, a Kelvin-Helmholtz instability is developing at the interface between both layers leading to a numerical mixture at time $t = 12.4$. The behavior of the numerical scheme is really close to the Kelvin-Helmholtz simulation. The dynamics is focused near the interface between two large and smooth layers. The effect of the Spectral scheme is not really important in terms of accuracy gain, however it is significant in terms of computational cost as it allows to decrease the number of Riemann problems to be computed.

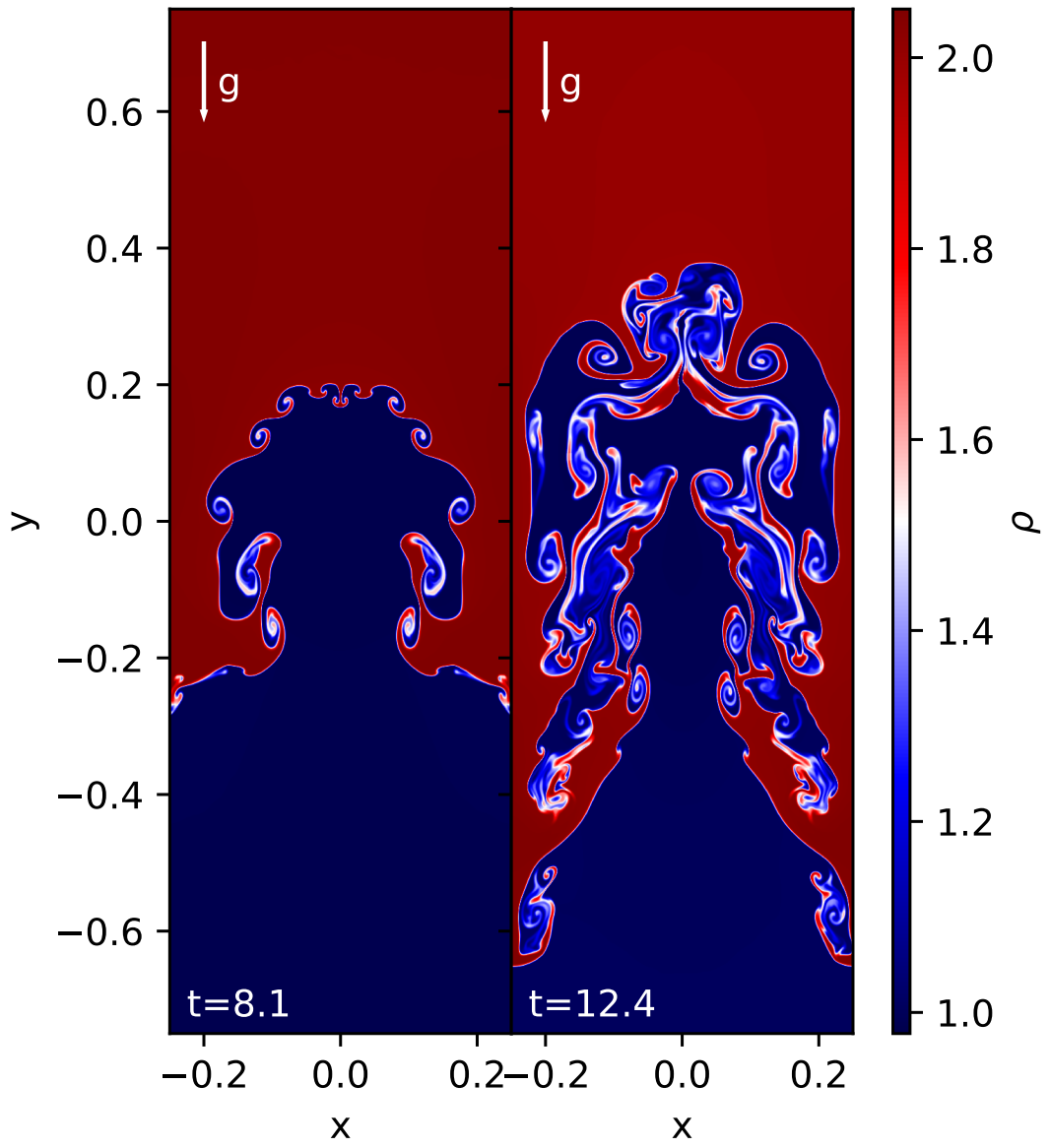


Figure 3.11: Two snapshots of a Rayleigh-Taylor simulation at resolution 500×1500 and times $t = 8.1$ and $t = 12.4$. We observe dense material going down due to instability.

Chapter 4

High Performance Computing tools

In the previous chapter, we have explored acceleration by using high order methods in order to adapt the representation of the solution depending on the local smoothness of the solution. In this chapter we continue by exploring other ways of accelerating the time to solution. We mainly explore two approaches: software and hardware acceleration by exploiting parallelism, implicit algorithms, Adaptive Mesh Refinements and a mix of them.

We begin by presenting the performance portability problematic, the Kokkos and Trilinos libraries. Then we present the different parallelism strategies explored in the codes ARK and ARK 2 used in previous chapters using Kokkos and MPI. Then we explore acceleration by using the acoustic implicit solver in low-Mach regimes and by using the Trilinos library. We explore AMR which allows to have a sparse representation of the numerical solution on the splitted numerical scheme. Finally we explore the porting to GPU of an implicit solver for radiative transfer from RAMSES using Trilinos.

4.1 The performance portability problematic

When going to large simulations one needs to use dedicated computers called supercomputers. By definition a supercomputer is a computer with high-end performance capabilities at a given time. To reach best performances, these computers use the state of the art technologies in terms of processors, networks, storage system, cooling system and software stack. Supercomputers are also called clusters because they are built upon aggregating nodes, connected together by a fast network. An example of such a supercomputer is shown on Figure 4.1 which is the Joliot-Curie machine hold at TGCC, Saclay, France.

In order to run on such clusters, a popular and standardized way is to use the Message Passing Interface, MPI. This programming model, qualified as distributed memory model, allows a process to access data from an other process' memory by explicitly engaging a communication. Hence the programmer has to explicitly split the work and distribute it between processes. MPI provides essential and basic communication patterns such as one-to-one, gather, scatter, all-to-all... Then the main difficulty is to have correct load balancing between processes. Nowadays this model is mainly used to communicate between nodes or inside a node between Non-Uniform Memory Access regions, also called NUMA regions. At the NUMA level, the shared memory model is better suited even though MPI can still be used.

Technologies evolve rapidly and more specifically processors. Moore observed this evolution for the density of transistors which has become a law named after him. To be more precise Moore's law is an economic law predicting the doubling of the density of transistors on a CPU

Figure 4.1: Example of a supercomputer, here Joliot-Curie at TGCC, Bruyères-le-Châtel, France

chip every 18 months. Figure 4.2 shows the Moore's law in orange dots. It also shows other characteristics of processors such as frequency and the number of logical cores. We can see that up until the beginning of the years 2000, computation power was increased by the increase of frequency. The same software could then run faster on the next generation of processors, this is usually referred as "free lunch", see Sutter 2013. After reaching the "heat wall", CPU manufacturers changed strategy and started to develop multi-core CPUs. Such processors are able to run at the same time two or more threads.

With the advent of multi-core processors, the Open Multi-Processing programming model has been used to exploit shared memory parallelism. In this model, work is again splitted among threads which are mapped to physical cores. However all threads share the same memory address space and thus can access any allocated data in the memory at any time without a request to an other thread. It is of the responsibility of the programmer to avoid conflicting accesses between threads. This type of conflict is called a race condition. In a multi-core processor, threads are independent and can execute different instructions on different data. The computation power of cores comes from both the usage of hierarchical memory, called caches, and complex control units able to do pipelining of instructions, executing instructions out-of-order... In addition to the core complexity, vector instructions allow to reduce the number of cycles necessary to realize a computation. For example some architectures are able to realize vector additions on 8 double precision floating point in the same number of cycles as the scalar equivalent operation.

Originally used for graphics computation GPUs, Graphics Processing Units, are architectures dedicated to massive parallelism. Thus, near 2010, with the release of the programming model CUDA from NVIDIA, which stands for Compute Unified Devices Architectures, GP-GPU started to be used as accelerators for scientific applications. Contrary to a processor, a GP-GPU is composed of thousands of simple cores, called CUDA cores, grouped into Streaming Multiprocessors. From the programming point of view threads are logically grouped into CUDA blocks. Each block is assigned to a Streaming Multiprocessor. The Streaming Multiprocessor is then responsible to schedule groups of 32 threads, called warps, onto the CUDA cores. Warps have the particularity to execute the same instruction on different data. If a branch divergence occurs, the warp executes both branches resulting in a loss of performance. As we can see

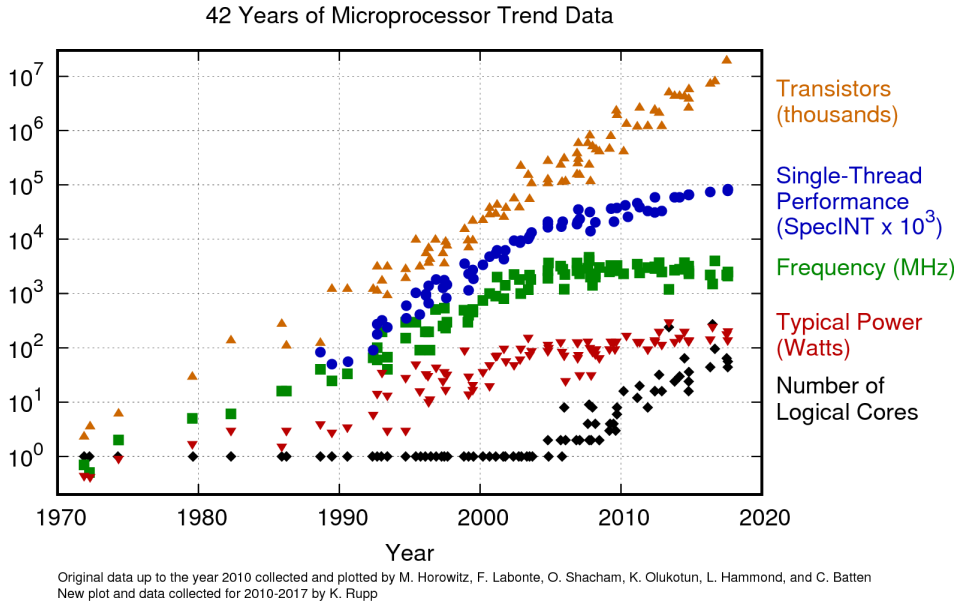


Figure 4.2: The figure represents the evolution over time of different characteristics for chips.

Rank	Supercomputer	Country	Main architecture
1	Fugaku	Japan	ARM
2	Summit	USA	NVIDIA V100
3	Sierra	USA	NVIDIA V100
4	Sunway TaihuLight	China	Sunway SW26010
5	Tienha-2A	China	Intel Xeon E5–2692v2

Table 4.1: TOP500 of June 2020

maximum performance is reached when threads within a warp agree on the flow of instructions.

As a result, the diversity in the main architectures of supercomputers increased as we can see from Table 4.1 which shows the first positions from the TOP500 ranking of the most powerful supercomputers in the world in June 2020. This diversity raises two related difficulties, portability and performance portability. Portability relates to the ability to run on different computers. The difficulty is due to differences in build systems, software dependencies, operating systems... Whereas performance portability is the ability to use efficiently different hardware architectures. The latter implies portability. As presented above, architectures are different and thus need different programming models to achieve good performances. This may lead to implement the same algorithm multiple times. In addition to the diversity of architectures, node size tends to increase. Multiple architectures can be on the same node thus requiring memory management and thus a flexible programming model to use all computation resources.

To address the aforementioned problems, a typical solution is to use OpenACC or OpenMP which avoids to refactor an existing code. The advantage is to add instructions to the compiler to generate parallel loops and kernels for GPUs. However memory management is difficult and can be a bottleneck on some applications as we will see further. We choose an other path which is to use a dedicated C++ library called Kokkos. We emphasize that it exists alternatives to

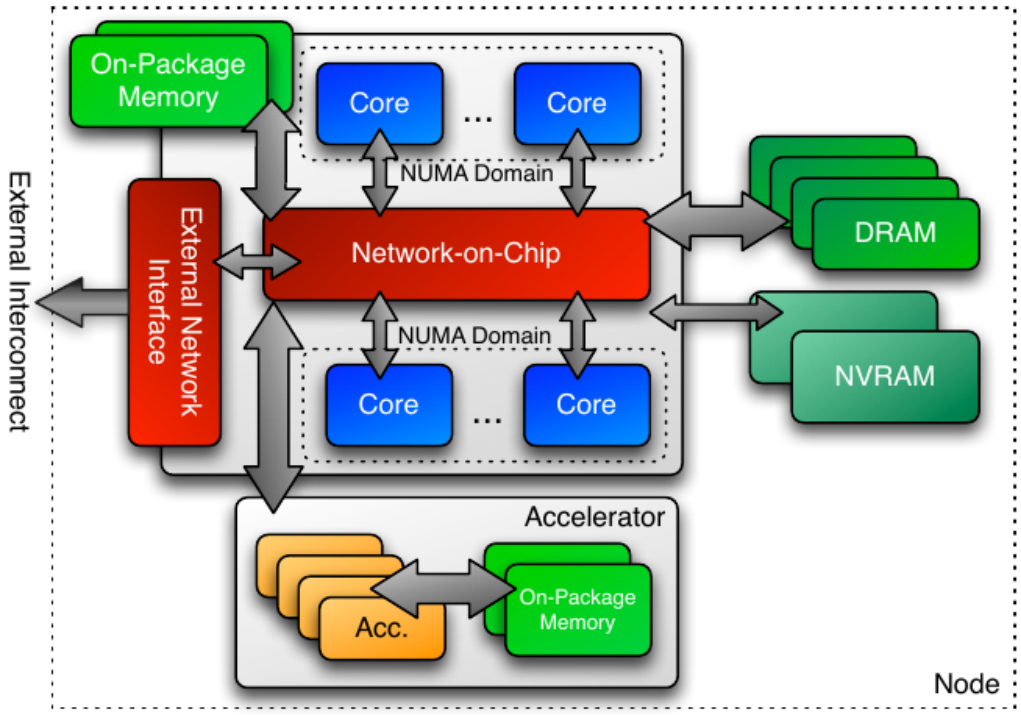


Figure 4.3: Figure represents an asbtract machine model of an exascale node.

Kokkos such as RAJA, Alpaka or the SYCL standard.

4.2 The Kokkos library

4.2.1 The Kokkos abstract machine model

The Kokkos programming model, as of version 2.9, is based on an abstract machine model representing an exascale node, see Figure 4.3. Resources from this machine can be classified into two categories: execution spaces and memory spaces. First an execution space represents resources able to do computation. From Figure 4.3 this corresponds to CPU cores (multi-core and many-core) to which are added accelerators (GPUs). Then a memory space represents resources able to store data. From Figure 4.3 different memories are available such as on-chip memory, central memory and memory on accelerators.

The Kokkos programming model allows the user to manage which execution space to use, when computation takes place and to control data transfer between memory spaces which may be costly. Kokkos is built upon existing programming models such as OpenMP or CUDA. We refer to these internal programming models as backends.

The model is based on different key concepts

- memory space: specifies where data are stored.
- memory layout: specifies the multidimensional data layout.
- execution pattern: basic parallel algorithm to execute.

- execution space: specifies where computation takes place.
- execution policy: specifies how computation is executed.

Similar to the CUDA programming model, it defines host to be the CPU which dispatches work on devices. In the CUDA programming model, the device corresponds to the GPU, in the Kokkos context it corresponds to any backend, which can be the CPU itself.

4.2.2 The Kokkos execution patterns

Execution patterns are the building blocks for more complex parallel algorithms. Kokkos provides three of them

- `Kokkos::parallel_for`,
- `Kokkos::parallel_reduce`,
- `Kokkos::parallel_scan`,

First the `Kokkos::parallel_for` is a basic loop pattern. Each loop iteration calls the functor passed as an argument. The `Kokkos::parallel_reduce` pattern realizes a reduction such as a summation. It allows to do some computation before the reduction. The `Kokkos::parallel_scan` pattern is also called a cumulative reduction such as a cumulative summation.

Similarly to CUDA, the code executed in an execution space is called a Kokkos kernel. The Kokkos kernel is a C++ functor. A functor is an object that can be called like a function thanks to the function call class operator, `operator ()`. This operator takes usually one argument that is the index, see example 4.1. Contrary to a function, the advantage of a functor is that it can store data.

Listing 4.1: Example of a functor

```
struct Functor
{
    KOKKOS_INLINE_FUNCTION
    void operator()( int index ) const
    {
        // unit of work
    }
};
```

An execution pattern is called from the host and executed on the device. As in the CUDA programming model, when such a pattern is called we say that a kernel is launched.

Kokkos does not guarantee any order of execution but only that the amount of work, specified by the execution policy, will be executed.

4.2.3 The Kokkos execution space

The Kokkos execution space directly maps to the machine execution space defined earlier. It specifies where the code will be executed. There are four possibilities:

- `Kokkos::Serial` represents the host and code is executed sequentially,
- `Kokkos::Threads` represents the host and uses POSIX threads,
- `Kokkos::OpenMP` represents the host and uses OpenMP,

- `Kokkos::Cuda` represents the device GPU and uses CUDA library.

At compile time, one matches the execution spaces to the architectures available on the target node. If multiple devices are enabled at compile time, one can choose at run time the execution space to use when launching a kernel. If no execution space is provided to the execution pattern, Kokkos relies on a default ranking of available execution spaces.

For a given execution space, kernels are executed sequentially. Between execution spaces Kokkos does not guarantee any order. To enforce an explicit barrier, `Kokkos::fence` waits until all kernels are finished to continue.

4.2.4 The Kokkos memory space and memory layout

The Kokkos memory space is also close to the memory space defined in the abstract machine model. It defines where the data will be stored. There are two main possibilities:

- `Kokkos::HostSpace` represents the memory on the host
- `Kokkos::CudaSpace`, `Kokkos::CudaHostPinnedSpace`,`Kokkos::CudaUVMSpace` represent different memory spaces on the GPU.

To remedy the lack of native multi-dimensional data container in C++, Kokkos provides its own called `Kokkos::View`. It is a multi-dimensional container for which the memory layout can be specified at compile-time. This feature is fundamental for both vectorization on CPU and coalescent memory access on GPUs.

4.2.5 The Kokkos execution policies

An execution policy specify how to execute an algorithm. Kokkos provides four of them

- `Kokkos::RangePolicy`
- `Kokkos::MDRangePolicy`
- `Kokkos::TeamPolicy`
- `Kokkos::TaskPolicy`

The Range policy is the simplest policy that Kokkos provides. It gives to the user an integer work index. This work index spans a range given by the user.

The MDRange policy is the multi-dimensional equivalent of the Range policy. Kokkos provides a tuple of integers which spans a Cartesian range.

In the Team policy, one exposes different levels of parallelism in an algorithm. In this policy, one does not deal with an integer as work index but with a team of Kokkos threads, which are mapped to backend OpenMP and CUDA threads. Further details are given in the next section.

Finally the Task policy provides a way to express dependencies between unit of works.

4.3 Parallelism in ARK and ARK 2

ARK, which stands for All-Regime Kokkos, is a fork from an existing code originally developed by P. Kestener at Maison de la Simulation. It is a C++ code using both MPI and Kokkos for respectively distributed and shared memory parallelism.

4.3.1 Shared memory parallelism using Kokkos

We begin with the shared memory parallelism and the usage of the Kokkos library. In the first version of the code ARK, we use the flat parallelism from the Range policy. The index range used corresponds to the local mesh size, including boundary layers. The parallel index is then unlinearized to test if it corresponds to an internal cell. Because it is close to a raw CUDA implementation this makes it efficient on GPUs. On CPUs it works well for distribution of work among threads. However the unlinearization along with the test usually prevents desired auto-vectorization from the compiler. As mentioned in the introduction CPU vectorization is the ability of a CPU to use particular instructions which are able to realize operations on multiple floating points in the same number of cycles that the scalar version. When relying on auto-vectorization, the usage of vector instructions is a shared responsibility between the compiler and the programmer. Depending on annotations from the programmer and loop complexity, the compiler determines if vector instructions can be used and if it provides a performance gain.

As Kokkos 2.9 relies on auto-vectorization in all execution policies, we follow general guidelines from Intel to achieve vectorization, see *Programming Guidelines for Vectorization* 2020, namely

- avoid branches,
- functions called should be inline,
- favor unit stride memory accesses.

To avoid branches, we remove the unlinearization at the inner loop level by using a two-level parallelism strategy from the three-level hierarchical parallelism from the Team policy.

Our parallelism strategy, implemented in ARK 2 is then as follows

- work assignment, see also Figure 4.4:
  - a Kokkos team maps to a line of cells along the X direction,
  - a Kokkos thread maps to a chunk of cells,
  - a Kokkos lane maps to a cell update,
- functions called inside the inner loops are either marked `KOKKOS_INLINE_FUNCTION` or `KOKKOS_FORCE_INLINE_FUNCTION`,
- replace small loops by multiple function calls,
- the layout is fixed to the left layout.

An example of such kernel is given by listing 4.3.1.

```
template < int dim >
KOKKOS_INLINE_FUNCTION
void Kernel<dim>::operator()(const Team& team) const
{
    // From given team compute coordinates of cell (0, j, k).
    const IntVector cell_coord0;
    // Get linear index from cell_coord0.
    const int cell0_index;

    Int i_start;
    Int i_end;
```

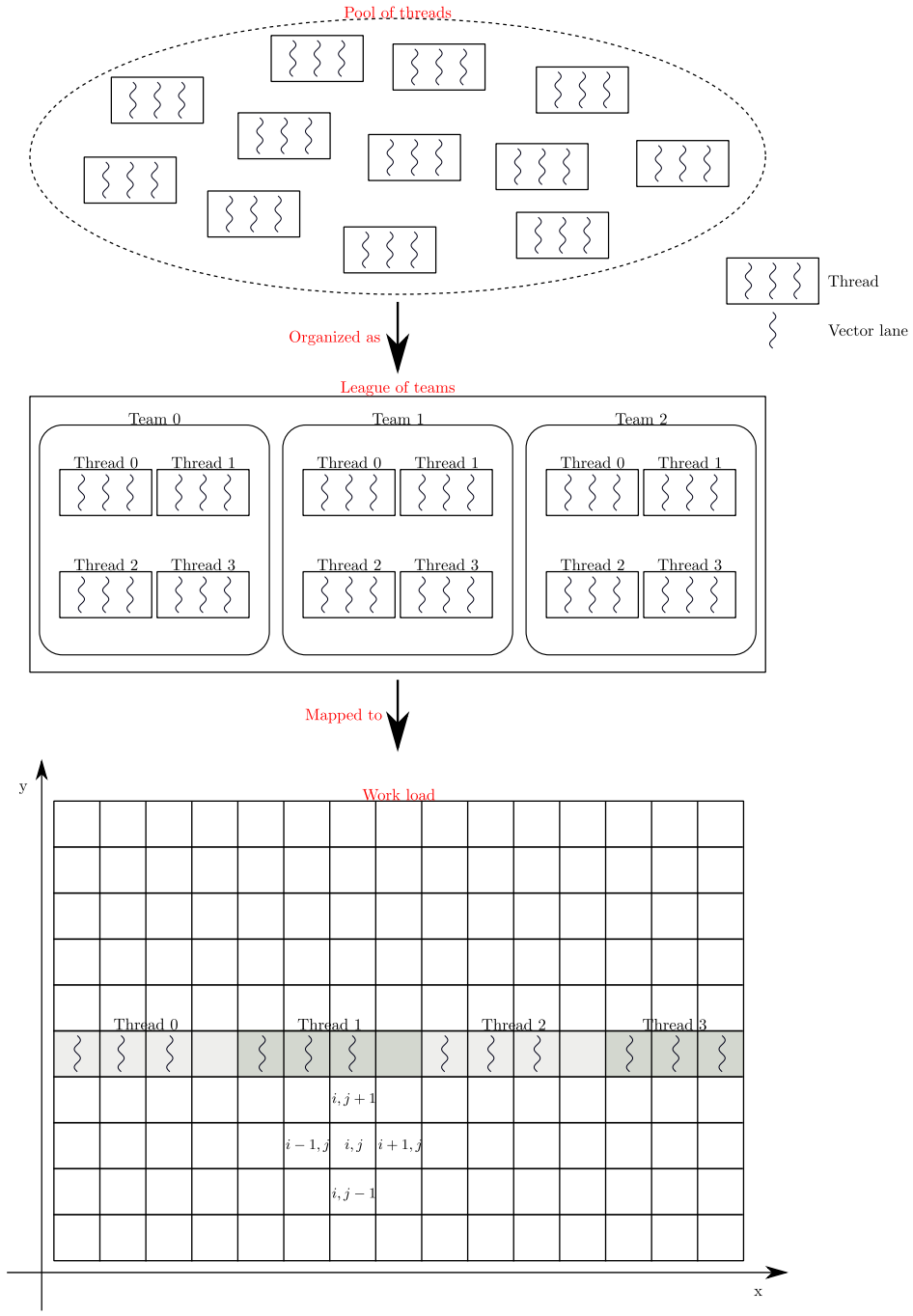


Figure 4.4: Graphical representation of the work assignment using hierarchical parallelism on a 2D mesh. A line of cells is assigned to a team of threads. Within the team, threads 0 to 3 split total work into contiguous chunks. Each lane in a thread is responsible of a cell.

```
    Kokkos::parallel_for(ArkTeamVectorRange(team, i_start, i_end),
      [&] (const Int i)
      {
          const Int cell_index = cell0_index + i;
          auto density = view_in( cell_index, 0 );

          // computation on density

          view_out( cell_index, 0 ) = density;
      });
}
```

```
template < class iType, class Team >
KOKKOS_INLINE_FUNCTION
static auto ArkTeamVectorRange(const Team& team,
                               const iType start,
                               const iType end) noexcept
    -> decltype(Kokkos::ThreadVectorRange(team, start, end))
{
    const iType chunk = (end - start) / team.team_size();
    const iType start_vector = start + team.team_rank() * chunk;
    const iType end_vector = (team.team_rank() == team.team_size() - 1)
                           ? end
                           : start_vector + chunk;

    return Kokkos::ThreadVectorRange(team, start_vector, end_vector);
}
```

This strategy mainly follows the aforementioned guidelines. Although explicit loop unrolling is not advised, we noticed better results in term of desired vectorization. The small loops are mainly coming from the way of writing kernels for multiple dimensions at a time through template parameter. The chosen alternative has been to develop a small template library for basic linear algebra operations and to manually unroll these loops. In order to check status of auto-vectorization we had to relied on vectorization reports from Intel compiler, see 4.A

Results of this strategy is shown on Figure 4.5. We compare the initial strategy using range policy in ARK and the second strategy in ARK 2 for two architectures. The first architecture Intel KNL heavily relies on vectorization. We see that in the range policy case, whether vectorization is enabled or not does not impact performance. Whereas with the team policy we see a gain between 3 and 4. The second architecture is a NVIDIA V100 GPU. We don't notice loss in performance. Indeed a Kokkos thread corresponds to the first dimension of a CUDA block, this also ensures coalescent memory access on GPUs.

4.3.2 Distributed memory parallelism using MPI

In the distributed memory model, we use an explicit domain decomposition also referred as MPI domain decomposition. We recall that we work on a regular domain hence, we use a Cartesian decomposition. We use a one-to-one mapping between MPI processes and domains.

We then use a ghost cell pattern. Each MPI process has extra layer of cells, called ghost cells, which are used to implement physical boundary conditions or exchange data with MPI neighbours. All MPI processes advance in a synchronized way due to the computation of the time step through a reduction and data exchange with neighbours. Then processes are independent and apply the different stages of the numerical scheme.

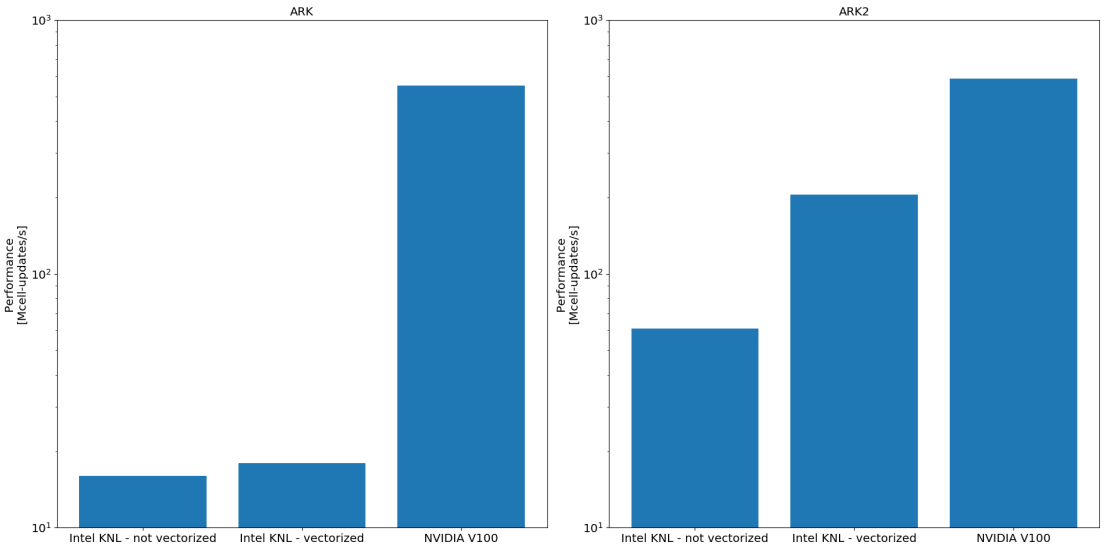


Figure 4.5: This figure shows difference of performance between initial range policy in ARK and team policy in ARK2 for different architectures.

Figures 4.6 and 4.7 show strong and weak scaling tests respectively. The weak scaling test was realized on a GPU partition of the Jean-Zay supercomputer, without the MPI CUDA-aware feature.

4.3.3 Hybrid parallelism using MPI and Kokkos

In the MPI standard, one has to pass raw pointers to communicate data with the corresponding data type description. We use the fact that Kokkos exposes internal raw pointers to the user. However depending on the Kokkos backend used, the raw pointer may not be handled by MPI and extra copy to the host is needed. In order to avoid unnecessary copies between the host and the device we use the decision diagram represented in Figure 4.8. This diagram helps to determine if one can pass the raw pointer from the `Kokkos::View` or if copy to the host is needed. We use Kokkos utilities to determine if the device space is accessible from the host. If it is not we need to copy data to the host. One exception corresponds to the case in which the device is CUDA and the MPI implementation, for example Open MPI, has the non-standard ability to handle CUDA pointers, usually referred as the CUDA-aware feature.

4.4 Parallelism using Trilinos

As it has been presented in Chapter 1, we can use implicit solvers in a low-Mach regime. As a portable implementation of linear system solvers is a work beyond this thesis we use the Trilinos library to solve the acoustic system. The advantage of using Trilinos, over other widely used libraries such as PETSc, is that it uses Kokkos at its core to achieve performance portability. Hence this minimizes changes in terms of data structures in ARK. However a major difference with Kokkos is that Trilinos also handles the distributed memory parallelism using MPI.

As it has been presented in Section 4.3.2 we use a ghost cell pattern to deal with the MPI domain decomposition. Hence one does not need an explicit global numbering of cells because

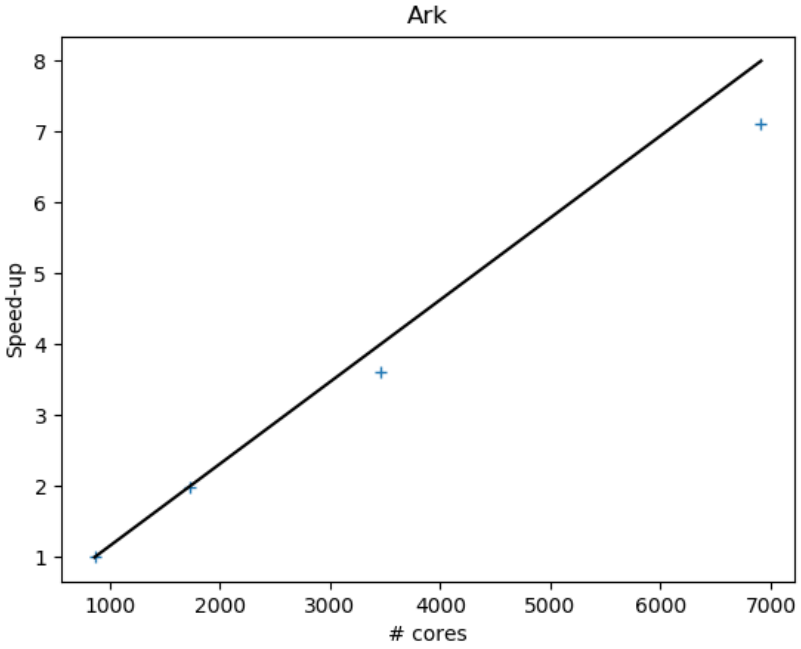


Figure 4.6: Strong scaling test on the Intel Skylake partition of the Joliot-Curie supercomputer, TGCC. The time measured is the global execution time without IO operations.

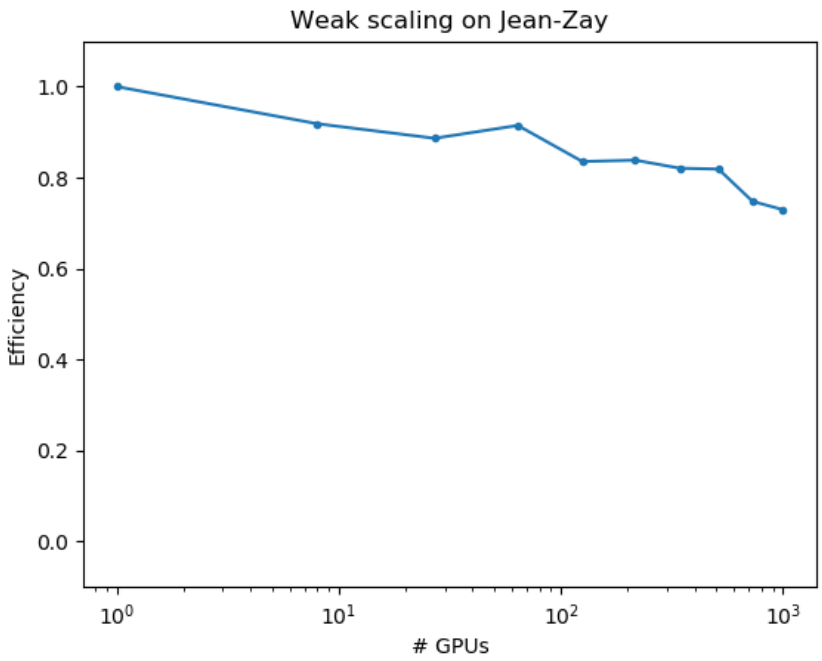


Figure 4.7: Weak scaling test on the NVIDIA V100 GPU partition of the Jean-Zay supercomputer, IDRIS. The time measured is the global execution time without IO operations

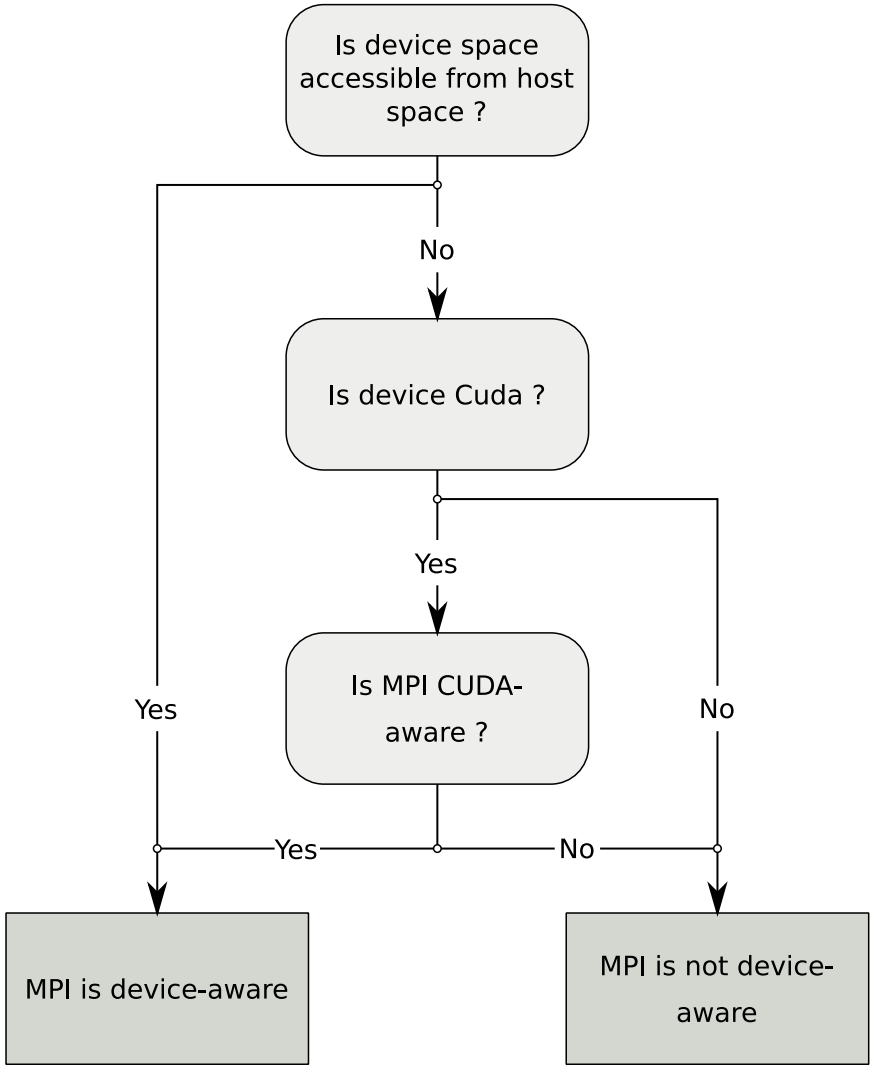


Figure 4.8: Decision diagram to determine whether MPI is device-aware. This decision can be made at compile time.

Mach number	implicit time (s)	iterations	explicit time (s)	acceleration
10^{-3}	19		14	0.74
10^{-4}	17		197	11
10^{-5}	17		1970 (estimated)	115

Table 4.2: MPI processes = 1, Threads=8, $N_x \times N_y = 128^2$

Mach number	implicit no correction		with correction	
	time (s)	iterations	time (s)	iterations
10^{-3}	19	39	200	
10^{-4}	17	74	200	
10^{-5}	17	74	200	

Table 4.3: MPI processes = 1, Threads=8, $N_x \times N_y = 128^2$

stencils from explicit numerical schemes connect only a few cells between two time steps. In the case of an implicit numerical scheme, all cells are in interaction in one time step. Moreover Trilinos vector and matrix data structures are distributed and thus require a global numbering. We take advantage of the Cartesian structure and use the canonical global coordinates of cells within the mesh. These coordinates are then linearized to have a global index.

Even though matrix values evolve at each time step, the matrix keeps a static symmetric profile due to the uniform mesh. The matrix has a maximum of $1 + 5d$ non-zero entries per line, d corresponding to the spatial dimension because of the pressure equation. The assembling of the matrix is distributed however there is no shared memory parallelism: the assembling of the matrix is realized sequentially from the host.

4.4.1 Numerical experiments

We use a GMRES iterative solver along with a Schwarz preconditioner to take into account MPI domain decomposition. Each MPI process also uses an inner preconditioner that we choose to be an incomplete factorization, ILUT. We use the Gresho vortex as a test case to control the flow regime.

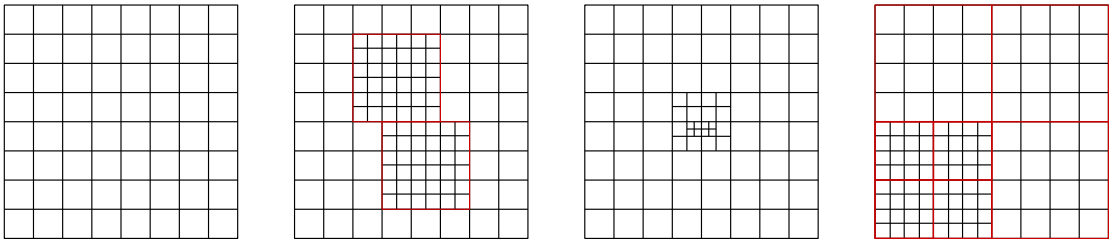
Table 4.2 compares the time to solution between the explicit and implicit solvers for different Mach regimes. We see that from Mach 10^{-3} and lower Mach regimes there is a great benefit in using an implicit solver which is near 10^{-3}Ma^{-1}. We emphasize that, even though the solution is captured faster, the solver presents the same drawbacks as in the explicit case. The solution diffuses and the vortex is lost.

As we can see from table 4.3 the solver shows a different behavior in time to solution when enabling the low-Mach correction. Indeed we notice difficulties in the convergence of the linear system. The solution exhibits checkerboard modes in density which are not damped due to the removal of numerical viscosity. Thus the linear solver error reaches a plateau higher than the threshold and stops on the maximal number of iterations. However, as in the explicit case, the vortex is preserved.

In table 4.4 we vary the size of the problem to be solved and compare it to the explicit case. We see a scaling as in the explicit numerical scheme. Indeed the numerical scheme is IM-EX, hence in a problem in dimension d, if the size is multiplied by 2 the cost is multiplied by 2^{d+1}, which includes the CFL condition.

$N_x \times N_y$	implicit time (s)	explicit time (s)	acceleration
64^2	2.5	28	11
128^2	17	197	11
256^2	124	1500 (estimated)	12

Table 4.4: MPI processes = 1, Threads=8, Mach = 10^{-4}



Uniform mesh Patch AMR mesh Cell AMR mesh Block AMR mesh

Figure 4.9: Different AMR types.

4.5 Towards an Adaptive Mesh Refinement All-Regime

The motivation of using AMR algorithms is to have a dynamic sparse representation of the numerical solution. This sparse representation can be used to focus computational efforts only in some parts of the domain like discontinuities and to capture strong gradients that would be too costly on a uniform mesh.

AMR algorithms usually fall into two opposite categories: cell based and patch based, see Figure 4.9. In the former approach, cells can be refined independently from each other. This is a flexible approach which allows to have very localized and deep refinement. The mesh is represented using a tree, an octree in 3D, in which AMR leaves represent one cell of the mesh. As a drawback of this data structure the looping pattern is complex. Moreover this approach tends to couple the numerical scheme to the AMR algorithm when looping over neighbours.

In the latter approach refinement occurs by patch of cells, potentially including cells that do not need refinement. This allows to have regular patches thus simplifying the looping pattern inside a patch. Patches are then nested to allow deeper refinement.

In our case we are interested in an intermediate approach, called the block based AMR, which allows to keep the flexibility of the cell based AMR and the regularity from the patch based AMR. In this method an octree is again used to represent the mesh. However, the major change is that leaves are not mesh cells anymore but regular block of cells. This work has been done in collaboration with O. Abramkina[1].

4.5.1 PABLO and Kokkos

As it has been presented in the sections above, the regularity of the computation is necessary to efficiently exploit new architectures through data parallelism. The introduction of block of cells inside leaves in the block based AMR reduces the problem of parallelization to the uniform mesh case.

Université Paris-Saclay, UVSQ, CNRS, CEA, Maison de la Simulation, 91191, Gif-sur-Yvette, France.
1. Université Paris-Saclay, CNRS, Institut du développement et des ressources en informatique scientifique, 91403, Orsay, France.

So far three aspects have been mentioned

- the AMR algorithm,
- the numerical scheme,
- the performance portability.

In order to decouple them, the strategy adopted is to use a dedicated library PABLO to handle AMR and the Kokkos library for performance portability. In order to avoid to manage the combination of neighbours when implementing a numerical solver, a ghost layer is added to each AMR leaf. The costly combination of possibilities is then factorized and executed once per time step. Numerical solvers can then be implemented as in the uniform mesh case. However one needs to be careful about fluxes at jumps of levels. Because of the non-linearity of fluxes, two neighbours do not compute the same flux. This problem is addressed in the next section.

Because ghost layers are added to each AMR leaf, this increases memory footprint. Thus to reduce it, it has been chosen to have two types of data structures. A first data structure contains all leaves without ghost layers and a second data structure (called group) contains only a part of leaves along with ghost layer.

4.5.2 All-Regime numerical scheme adaptation

As presented in Chapter 1, the numerical scheme uses an operator splitting. This extends the stencil compared to a regular first order numerical scheme. There are two alternatives of the algorithm adaptation. Either we apply the numerical scheme for one group of leaves at once, thus requiring two ghost layers, either we apply the acoustic step for all groups and then apply the transport step, thus requiring only one ghost layer. Constraints with the PABLO library lead us to choose the second option.

In order to deal with the conservation issue, we use averaging. If l and $l+1$ denote the levels between the two leaves, we average cells from leaves from fine level $l+1$, see Figure 4.10. This enforces ghost cells from level l to match values of cells from level $l+1$. The averaging is done before and after the acoustic step. The drawback is to increase numerical dissipation but as a consequence it provides smooth transition between levels. Figure 4.11 shows the result of a Blast wave simulation generated by a high energy deposit at the center of the box, see Sedov 1959. We compare an AMR simulation using the previous strategy (left hand panel) with a MOOD simulation as in Chapter 3 (right panel). On the bottom panel we show AMR levels and valid cells for the AMR and the SVM simulations respectively. We can see that the regions where cells are detected as invalid in the MOOD procedure corresponds to the regions where the mesh is refined.

4.6 Adaptive Mesh Refinement and implicit

4.6.1 Presentation of the RAMSES code

RAMSES is a widely used simulation code in the astrophysics community. It has been originally developed by R. Teyssier in the context of cosmological simulations at CEA/Irfu. For this purpose they use particles representing matter. Since then new physics modules have been added, a non-exhaustive list

- hydrodynamics or magneto-hydrodynamics equations,

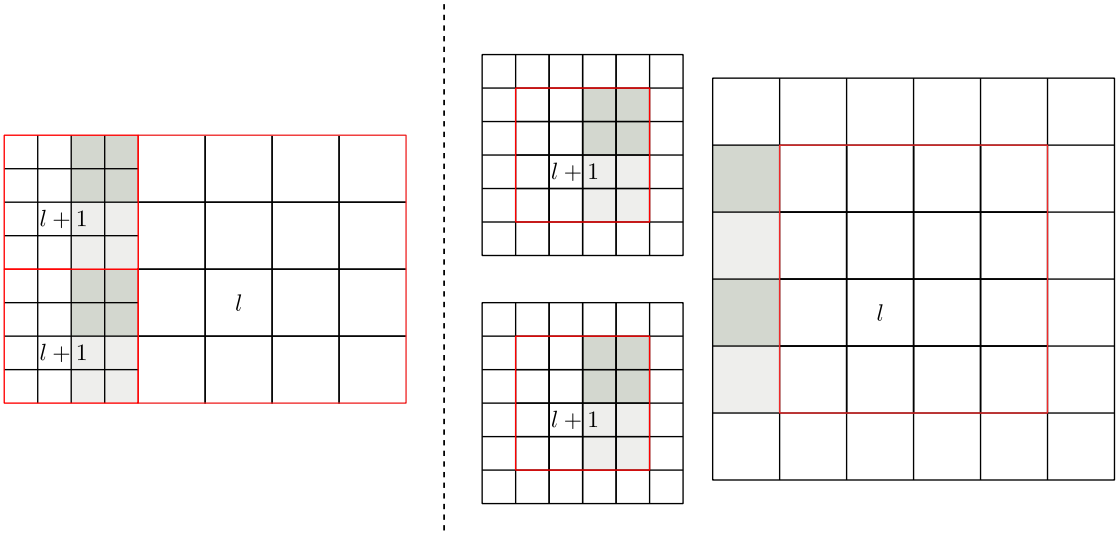


Figure 4.10: Figure shows the averaging procedure to recover conservation between two levels l and $l + 1$. Both panels represent the same leaves, on the right panel ghost layer is added. In grey backgrounds, cells are averaged.

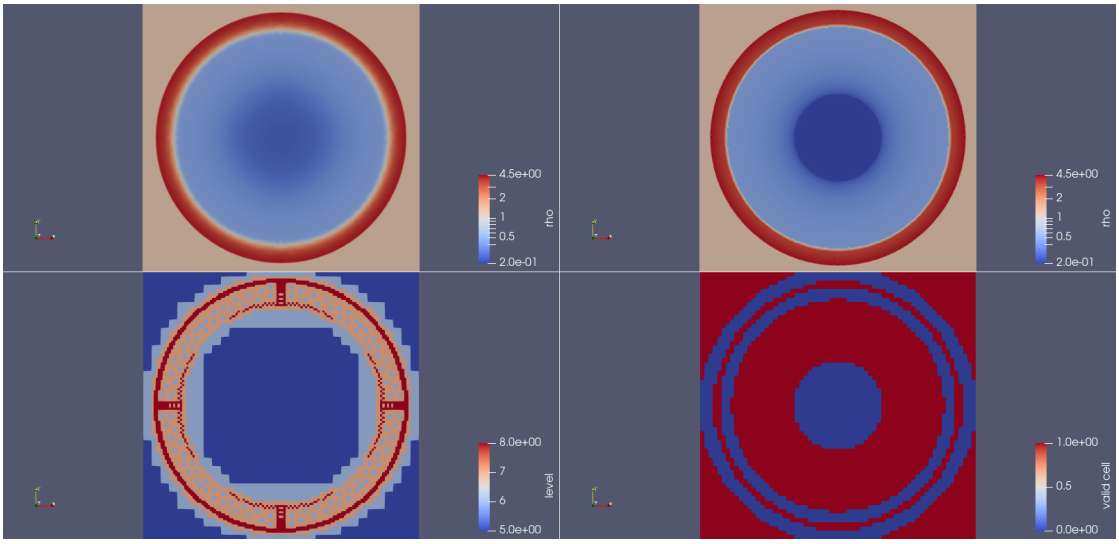


Figure 4.11: Blast wave simulation results. The left hand side shows an AMR simulation, with AMR levels between 5 and 8 along with 4^2 patches inside an AMR leaf. The right hand side shows a Spectral Volume simulation at order 4 with 64^2 spectral cells. The limitation is realized with MOOD using a second order MUSCL-Hancock scheme.

- radiative transfer equations,
- Poisson equation for self-gravity,
- particles of matter, sinks.

RAMSES uses Adaptive Mesh Refinement. AMR implementation is based on a tree also called octree to emphasize the number of potential sub-nodes within a node. As memory is organized as a one-dimensional array, a mapping is necessary between the three-dimensional mesh and the one-dimensional array. The Peano-Hilbert space filling curve is then used to realize this mapping.

In terms of parallelization, RAMSES uses MPI. The parallelization consists in chunking the space filling curve and assigning each chunk to a MPI process. Each MPI process is then responsible to apply the different numerical schemes. As the mesh refinement occurs at runtime, the initial work distribution may become unbalanced between MPI processes. Hence a load balancing procedure is applied consisting in a new chunking of the space filling curve.

4.6.2 Radiative transfer module

The radiative transfer module is used to model the propagation of radiative energy and its interaction with matter. It exists different moment models for the radiative transfer depending on different approximations. In this work we are interested in the so-called grey Flux Limited Diffusion, also known as FLD model. As the name suggests, the equation on the radiative energy E_r is similar to a diffusion equation with source terms, it writes

$$\partial_t E_r - \operatorname{div}\left(\frac{c\lambda}{\kappa_R \rho}\boldsymbol{\nabla} E_r\right) = \kappa_P \rho c \left(aT^4 - E_r\right) \tag{4.1}$$

in which a is the Stefan-Boltzmann constant, c is the light speed, κ_R and κ_P are given opacity functions that solely depend on matter temperature T, ρ the density of matter and finally λ a flux limiter. The model is qualified as grey because the energy E_r takes into account all frequencies at once.

An implicit time discretization is used in order to avoid severe stability condition restriction when coupled to hydrodynamics due to the parabolic nature of Equation 4.1. The numerical scheme has the particularity to solve the radiative transfer equations per level of the AMR tree: one level uses other levels as boundary conditions, see Commerçon et al. 2014. Hence, at a given AMR level, the numerical discretization of Equation 4.1 writes as in the uniform mesh case

$$\begin{aligned}
{E_r}^{n+1}_{i,j,k} = {E_r}^{n}_{i,j,k} &+ \frac{\Delta t}{\Delta x}\left(\sigma^n_{i+1/2,j,k}(\boldsymbol{\nabla} E_r)^{n+1}_{i+1/2,j,k} - \sigma^n_{i-1/2,j,k}(\boldsymbol{\nabla} E_r)^{n+1}_{i-1/2,j,k}\right) \\
&+ \frac{\Delta t}{\Delta y}\left(\sigma^n_{i,j+1/2,k}(\boldsymbol{\nabla} E_r)^{n+1}_{i,j+1/2,k} - \sigma^n_{i,j-1/2,k}(\boldsymbol{\nabla} E_r)^{n+1}_{i,j-1/2,k}\right) \\
&+ \frac{\Delta t}{\Delta z}\left(\sigma^n_{i,j,k+1/2}(\boldsymbol{\nabla} E_r)^{n+1}_{i,j,k+1/2} - \sigma^n_{i,j,k-1/2}(\boldsymbol{\nabla} E_r)^{n+1}_{i,j,k-1/2}\right) \\
&+ {\kappa_P}^n_{i,j,k}{\rho}^n_{i,j,k} c\left(a\left({T}^n_{i,j,k}\right)^4 - {E_r}^{n+1}_{i,j,k}\right)
\end{aligned}$$

$$\sigma^n_{i+1/2,j,k} = \frac{c\lambda^n_{i+1/2,j,k}}{{\kappa_R}^n_{i+1/2,j,}{\rho}^n_{i+1/2,j,k}}$$

$$(\boldsymbol{\nabla} E_r)^{n+1}_{i+1/2,j,k} = \frac{1}{\Delta x}\left({E_r}^{n+1}_{i+1,j,k} - {E_r}^{n+1}_{i,j,k}\right)$$

along with boundary conditions, either physical or virtual between two AMR levels. Dirichlet virtual boundary conditions are used at interface between two levels. Moreover the radiative energy gradient is modified to account for the level difference, see Commerçon et al. 2014.

We note that this numerical scheme can have a different interpretation, it can be seen as the first iteration in a Schwarz domain decomposition. We recall that the non-overlapping Schwarz method consists in solving a global boundary value problem by solving smaller boundary value problems iteratively using appropriate virtual boundary conditions. If we consider AMR levels as the subdomains, we recover the above numerical scheme at first iteration.

RAMSES uses its own implementation of a BiCGSTAB solver along with a diagonal Jacobi preconditioner. A standard approach in such a code is to use a matrix-free implementation of the BiCGSTAB. However in the case of the radiative transfer module, the matrix is stored to avoid to compute too many times the costly coefficients.

4.6.3 Porting to GPU

Given the current state of the RAMSES code, it is not possible to run on GPUs. There exists different approaches to port existing Fortran codes to GPUs such OpenMP or OpenACC. However these approaches would require modifications of the code and possible loop reordering. As it has been presented in Section 4.4, Trilinos implements linear solvers and preconditioners which can run on different architectures. Hence the objective of this work is to port the radiative transfer solver module to GPUs using Trilinos capabilities. This work has been done in collaboration with H. Bloch[2] and S. Donfack[3].

We try to minimize modifications on the Fortran code by taking advantage of the assembling of the matrix. Matrix elements are computed per matrix line on the Fortran side and sent to the Trilinos data structure on the C++ side. To ease the assembling we use our own global numbering of AMR leaves.

Once the matrix is assembled for one AMR level, the Trilinos linear problem is prepared and solved. The solution is then copied back to RAMSES data structures.

In order to manage the interoperability between the Trilinos solver, which is C++, and the RAMSES code, which is Fortran code, we call C functions from the Fortran code, mainly from the diffusion_cg routine.

4.6.4 Results and discussion

Validation is realized using the "Dirac test case" from the RAMSES test suite. This test case consists in having an energy impulse at the center of the box and let it diffuse.

Figure 4.12 shows strong scaling results in a three-dimensional Dirac simulation with AMR fixed at level 8. Simulations have run on the Jean-Zay cluster at IDRIS, France. Nodes are made of Intel Skylake with 40 cores accelerated with 4 NVIDIA V100 GPUs. We compare the GPU version, using Trilinos, and the original version, using only MPI. We see that 8 MPI processes are equivalent to 1 GPU. This result is due to the time spent in assembling the matrix which encompasses memory transfer between host and device. For instance in the case of using 4 GPUs, the total time spent to solve the linear systems is only of 4s which is only 12% of the total time.

2. Université Paris-Saclay, UVSQ, CNRS, CEA, Maison de la Simulation, 91191, Gif-sur-Yvette, France.
Université Paris-Saclay, UVSQ, CNRS, CEA, Maison de la Simulation, 91191, Gif-sur-Yvette, France.
3. Université Paris-Saclay, CNRS, Institut du développement et des ressources en informatique scientifique, 91403, Orsay, France.

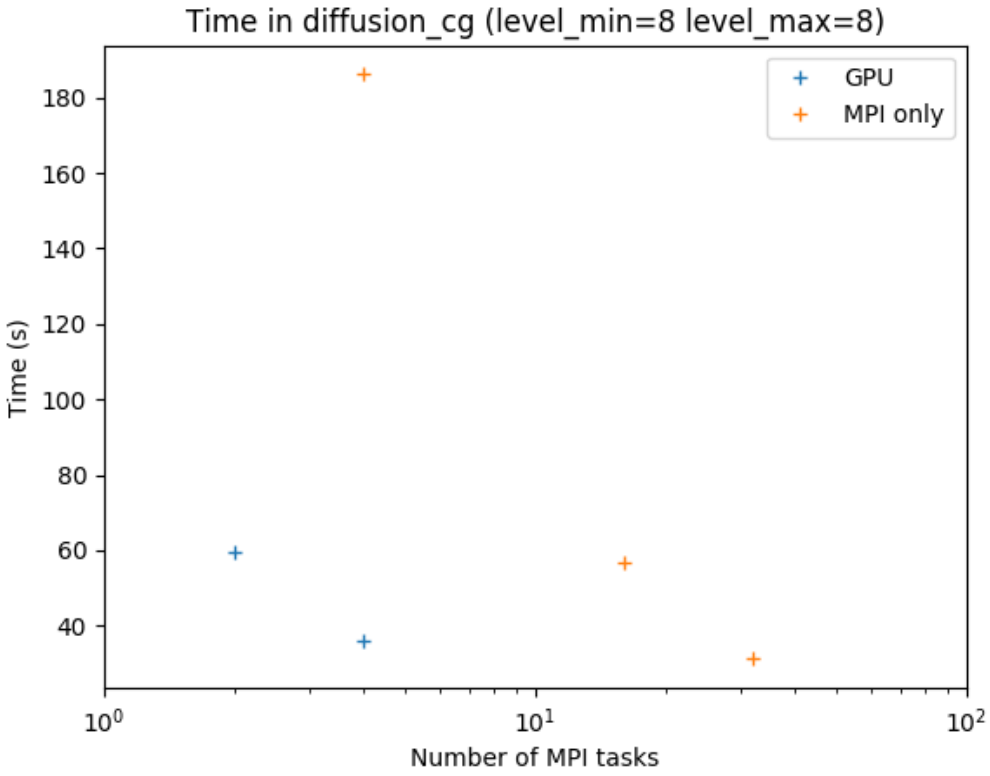


Figure 4.12: Strong scaling test. The time is measured inside the FORTRAN diffusion_cg routine.

4.7 Conclusion

Even though vectorization on Intel CPUs has been achieved using Kokkos hierarchical parallelism, auto-vectorization is not satisfactory as it relies both on the compiler and the respect of some vectorization rules. This approach makes it difficult to anticipate successful vectorization as it can only be checked after compilation. For instance in our case the GNU 8 compiler is not able to vectorize ARK 2.

This is a known problem and the alternative to auto-vectorization is explicit vectorization by using architecture intrinsics. By definition intrinsics are language-bindings to architecture specific assembler instructions. By using them we enforce vectorization but we loose portability. As a remedy to this problem C++ wrappers have been developed relying on C++ features. The wrapper consists in defining a class which stores an intrinsic object. Operations on these objects are then vectorized thanks to internal calls to intrinsics operations. Portability between CPUs is recovered by choosing at compile time the correct architecture to use. This solution has been adopted in different libraries such as Boost.SIMD, xsimd and will be part of C++23 with std::simd. A future version of Kokkos (>3.1) should include such a solution. As of today, the Kokkos wrapper of intrinsics is also able to deal with GPUs, contrary to other known wrappers. Indeed, based on the hierarchical parallelism model the wrapper maps work to the first dimension of a CUDA block. Hence, by choosing at compile-time a vector length to be the size of warp, that is 32, this enforces a warp to execute the same instruction and to have coalescent memory access. As a drawback this implies some refactoring of the code. Scalar functions that are called in vectorized loops need to be redefined using the wrapper type.

Algorithmic acceleration using an implicit acoustic solver is promising. It revealed convergence difficulties when modifying the fluxes with the low Mach correction. The convergence is slow and shows that further work on the preconditioner is needed to be used in production.

AMR is a powerful tool to avoid a waste of computation resources in non-essential regions. We explored a simple extension to the uniform mesh formulation of the all-regime numerical scheme. There is still work to provide a well-balanced formulation on an AMR mesh. The difficulty may reside in the dynamic refinement. The initial one-dimensional profile can still be used to initialize the simulation. However the behavior of refinement on the profile may not be able to keep both equilibrium and conservation property.

Finally we explored the usage of Trilinos, an external library, to provide linear solvers in the AMR code RAMSES. Even though it successfully allowed the usage of GPU using the portability of Trilinos, at this stage the benefit is not clear. A lot of time is spent in the assembling phase mixing both computation of matrix coefficients and data transfer to the GPU.

Appendix

4.A Compilation time considerations

This section deals with different strategies of compilation when using Kokkos, however it may also apply to other header oriented libraries.

The C++ language is a strongly typed language meaning that the type of a variable is fixed at compile time. The language also offers generic programming, also referred as template meta-programming, meaning that one can write at once multiple functions, for different types. The snippet 4.2 gives an example of a generic function returning the addition of objects of the same type `class T`. Of course this function can only be called if the body makes sense for the given type.

Listing 4.2: Template example

```
template < class T >
T plus( T&& t1, T&& t2 )
{
    return t1 + t2;
}
```

The template parameter types of such generic function are then fixed when the function is called, either explicitly or implicitly by following deduction rules. Template code is automatically considered inline to avoid a violation of the one definition rule, keyword `inline` can be omitted.

From the compilation point of view this implies that binary code is generated only when these generic functions are called in a translation unit. This also means that if a generic function is called in multiple translation units, the compiler has to generate multiple times binary code related to this function. Because a generic function is inline, the compiler may choose to replace the function call by its body where it is called.

Listing 4.3: Kokkos example

```
// main.cpp
#include "KernelA.hpp"
#include "KernelB.hpp"

int main( int argc, char** argv )
{
    // ...
    Kokkos::parallel_for( n, KernelA( view1, view2 ) );
    Kokkos::parallel_for( n, KernelB( view1, view2 ) );
    Kokkos::deep_copy( view1, view2 );
    // ...
```

```
}
```

We consider a first example with the snippet 4.3. In this example two kernels are executed in a `parallel_for` evolving `view2` from `view1` and then copy `view2` back to `view1`. Three computation kernels are called, two from the user's code and one from KOKKOS. To avoid a loss of performance kernel's function call operator `void operator () (/*...*/) const` is inline and thus defined in a header file. From this example, we notice some drawbacks of generic code about compilation

- any modification to `KernelA` body has the side effect to trigger recompilation of unchanged `KernelB` body,
- if the `KernelA` is called in an other source file, with the same call to `parallel_for` its body is recompiled,
- increases size of a translation thus exposing less parallelism,
- chaining kernels may reach rapidly inline limits of compilers, for example INTEL compiler,
- analysis of inlining and vectorization of functors `KernelA`, `KernelB` and `deep_copy` are all placed in the same report. Hence it results in long and obscure optimization reports when they are generated on a per translation unit basis.

Hence we have tried to reduce these side effects in ARK. We consider then two types of inline code

- critical inline functions for optimization from the compiler such as auto-vectorization,
- kernel inline functions that should not be optimized in the context where they are called such as KOKKOS kernels.

Thus we consider that optimization from the compiler should be dedicated to a given kernel. This prevents for example the compiler to do optimizations such as loop fusion between kernels that may benefit from cache reuse on CPU.

4.A.1 Solution 1

Listing 4.4: Solution 1

```
// Kernel.hpp
#pragma once

class Kernel
{
public:
    Kernel( Kokkos::View& view1, Kokkos::View& view2 );

    static void apply( int n, Kokkos::View& view1, Kokkos::View& view2 );

    KOKKOS_FUNCTION
    void operator()( int i ) const;

    Kokkos::View m_view1;
    Kokkos::View m_view2;
};
```

```
// Kernel.cpp
#include "Kernel.hpp"

Kernel::Kernel( Kokkos::View& view1, Kokkos::View& view2 )
    : m_view1 ( view1 ), m_view2 ( view2 )
{
}

void Kernel::apply( int n, Kokkos::View& view1, Kokkos::View& view2 )
{
    Kokkos::parallel_for( n, Kernel( view1, view2 ) );
}

KOKKOS_FUNCTION
void Kernel::operator()( int i ) const
{
    //...
}

// main.cpp
#include "Kernel.hpp"

int main( int argc, char** argv )
{
    Kernel::apply( 10 );
    return 0;
}
```

In snippet 4.4 we introduce a static method `apply` which is responsible to create the functor and launch the parallel execution. This answers all aforementioned issues. In the case of a template functor, one also uses explicit template instantiation for a set of template parameters commonly used.

Listing 4.5: Solution 1

```
// main.cpp
#include "Kernel.hpp"

int main( int argc, char** argv )
{
    // ...
    Kokkos::parallel_for( n, Kernel( view1, view2 ) );
    // ...
    return 0;
}
```

However this introduces a minor issue. As we can see in snippet 4.5, it is a valid code but compiler is not able to inline function call as the definition is not visible in this translation unit. Hence inter-procedural optimization (IPO) can be used to trigger inlining. A related issue happens if the declaration of the function call operator is marked to be forced to be inlined. Hence snippet 4.5 will not even compile.

4.A.2 Solution 2

A more flexible approach would involve a third file.

Listing 4.6: Solution 2

```
// Kernel.hpp
#pragma once

class Kernel
{
public:
    Kernel( Kokkos::View& view1, Kokkos::View& view2 );
        : m_view1 ( view1 ), m_view2 ( view2 )
    {
    }

    KOKKOS_INLINE_FUNCTION
    void operator()( int i ) const;
    {
        //...
    }

    Kokkos::View m_view1;
    Kokkos::View m_view2;
};

// KernelApply.hpp
#pragma once

void apply_kernel( int n, Kokkos::View& view1, Kokkos::View& view2 );

// KernelApply.cpp
#include "KernelApply.hpp"
#include "Kernel.hpp"

void apply_kernel( int n, Kokkos::View& view1, Kokkos::View& view2 )
{
    Kokkos::parallel_for( n, Kernel( view1, view2 ) );
}

// main.cpp
#include "KernelApply.hpp"

int main( int argc, char** argv )
{
    // ...
    apply_kernel( 10, view1, view2 );
    // ...
    return 0;
}
```

Conclusion and perspectives

The aim of the thesis is to provide "all-regime" numerical schemes for convection simulations and liquid droplet impacts. We used schemes based on a splitting of operator between acoustic and transport phenomena.

In the Chapter 1, we have been able to adapt this splitting of operator with a gravity source term. Accouting for the source term in the acoustic system this allowed to derive two well-balanced numerical schemes. A first version of the scheme is a sequel to the work of Chalons et al. 2017. A second version (more complex) allows to comply with the conservation of a new energy that includes the gravitational energy. Moreover it has been shown that the flux correction based on the Mach number, referred as low-Mach correction, allows to remove unnecessary diffusion in the low Mach regime. In order to test a generalized theory of convection a Grand Challenge simulation on the Jean-Zay supercomputer has been performed using the code ARK that implements this numerical scheme.

In the Chapter 2 we have applied this splitting of operator to a Homogeneous Equilibrium Model for simulating liquid droplet impacts. In order to keep a sharp interface between the liquid droplet and the gas, we have used an instantaneous relaxation from a five-equation model. The splitting of operator was then applied to the five-equation model allowing to use an anti-diffusive scheme for the transport step. We have performed different simulations of liquid droplet impacts including a Grand Challenge simulation on the Joliot-Curie supercomputer using a fork of ARK.

In the Chapter 3 we have explored the possibility of reducing the time to simulation using high-order numerical schemes on regular grids. By adapting the solution representation to the regularity of the solution we sought to reduce the number of expensive, non-linear computations. It has shown promising results using a sequential implementation.

Finally in the Chapter 4 we explored different strategies of accelerating the time to solution using different parallel programming models. In particular we have been able to obtain a single source code able to run efficiently on both GPUs and CPUs (with vectorization) by using the Kokkos library.

Different perspectives arise from the present work. Even though the numerical scheme for convective flows enables the use of time implicit numerical methods, performance and scalability also require an appropriate preconditioner for solving the linear systems involved in the simulation in order to obtain an efficient discretization that can be used in production, see Chapter 4.

We have worked on the building blocks of an AMR implementation of this numerical scheme without the gravity source term. It is not clear that the refinement (and derefinement) process is compatible with the well-balanced property. Future works could consist in studying if the well-balanced feature of the numerical scheme is impacted by the AMR context.

The hybrid high-order numerical scheme has shown promising results and suggests different further developments. A first strategy could be to use this numerical scheme for each sub-system of the splitting of operator. A second strategy could be to replace the MUSCL-Hancock fallback scheme by a numerical scheme focusing on contact discontinuities such as an anti-diffusive

solver (Després and Lagoutière 2001) or the THINC method (a similar strategy is developed by M. Kucharik, R. Loubère). A last strategy would be to couple the hybrid method with AMR. The result would benefit from both the p-convergence in smooth regions and h-convergence from AMR in discontinuous regions.

Finally, a Kokkos SIMD wrapper is in development as an alternative to the auto-vectorization strategy used inside Kokkos. The usage of such a wrapper would enforce to think in terms of a chunk of cells. The vectorization would no longer rely on the ability of the compiler to find vectorizable loops, without loosing performance with GPUs.

Bibliography

Allaire, Grégoire, Sébastien Clerc, and Samuel Kokh. 2002. "A Five-Equation Model for the Simulation of Interfaces between Compressible Fluids" [in en]. *Journal of Computational Physics* 181, no. 2 (September): 577–616.

Almgren, A. S., V. E. Beckner, J. B. Bell, et al. 2010. "CASTRO: A NEW COMPRESSIBLE ASTROPHYSICAL SOLVER. I. HYDRODYNAMICS AND SELF-GRAVITY." *The Astrophysical Journal* 715, no. 2 (June): 1221–1238.

Baer, M.R., and J.W. Nunziato. 1986. "A two-phase mixture theory for the deflagration-to-detonation transition (ddt) in reactive granular materials" [in en]. *International Journal of Multiphase Flow* 12, no. 6 (November): 861–889.

Barberon, Thomas, and Philippe Helluy. 2005. "Finite volume simulation of cavitating flows" [in en]. *Computers & Fluids* 34, no. 7 (August): 832–858.

Barsukow, Wasilij, Philipp V. F. Edelmann, Christian Klingenberg, Fabian Miczek, and Friedrich K. Röpke. 2017. "A Numerical Scheme for the Compressible Low-Mach Number Regime of Ideal Fluid Dynamics" [in en]. *Journal of Scientific Computing* 72, no. 2 (August): 623–646.

Bouchut, François. 2004. *Nonlinear Stability of Finite Volume Methods for Hyperbolic Conservation Laws.* Frontiers in Mathematics. Basel: Birkhäuser Basel.

Bouchut, François, Christophe Chalons, and Sébastien Guisset. 2017. "An entropy satisfying two-speed relaxation system for the barotropic Euler equations. Application to the numerical approximation of low Mach number flows." December.

Boyd, John P, and Jun Rong Ong. 2009. "Exponentially-Convergent Strategies for Defeating the Runge Phenomenon for the Approximation of Non-Periodic Functions, Part I: Single-Interval Schemes" [in en]. *Commun. Comput. Phys.* 14.

Carter Edwards, H., Christian R. Trott, and Daniel Sunderland. 2014. "Kokkos: Enabling many-core performance portability through polymorphic memory access patterns" [in en]. *Journal of Parallel and Distributed Computing* 74, no. 12 (December): 3202–3216.

Caupin, Frédéric, and Eric Herbert. 2006. "Cavitation in water: a review" [in en]. *Comptes Rendus Physique* 7, nos. 9-10 (November): 1000–1017.

Chalons, Christophe, and Frederic Coquel. 2014. "Modified Suliciu relaxation system and exact resolution of isolated shock waves" [in en]. *Mathematical Models and Methods in Applied Sciences* 24, no. 05 (May): 937–971.

Chalons, Christophe, Frédéric Coquel, Edwige Godlewski, Pierre-Arnaud Raviart, and Nicolas Seguin. 2010. "Godunov-type schemes for hyperbolic systems with parameter dependent source. The case of Euler system with friction" [in en]. *Mathematical Models and Methods in Applied Sciences* 20, no. 11 (November): 2109–2166.

Chalons, Christophe, and Jean-François Coulombel. 2008. "Relaxation approximation of the Euler equations" [in en]. *Journal of Mathematical Analysis and Applications* 348, no. 2 (December): 872–893.

Chalons, Christophe, Mathieu Girardin, and Samuel Kokh. 2013. "Large Time Step and Asymptotic Preserving Numerical Schemes for the Gas Dynamics Equations with Source Terms" [in en]. *SIAM Journal on Scientific Computing* 35, no. 6 (January): A2874–A2902.

———. 2016. "An All-Regime Lagrange-Projection Like Scheme for the Gas Dynamics Equations on Unstructured Meshes" [in en]. *Communications in Computational Physics* 20, no. 01 (July): 188–233.

Chalons, Christophe, Pierre Kestener, Samuel Kokh, and Maxime Stauffert. 2017. "A large time-step and well-balanced Lagrange-projection type scheme for the shallow water equations" [in en]. *Communications in Mathematical Sciences* 15 (3): 765–788.

Chandrashekar, Praveen, and Christian Klingenberg. 2015. "A Second Order Well-Balanced Finite Volume Scheme for Euler Equations with Gravity" [in en]. *SIAM Journal on Scientific Computing* 37, no. 3 (January): B382–B402.

Charbonneau, Paul. 2014. "Solar Dynamo Theory" [in en]. *Annual Review of Astronomy and Astrophysics* 52, no. 1 (August): 251–290.

Chertock, Alina, Shumo Cui, Alexander Kurganov, Şeyma Nur Özcan, and Eitan Tadmor. 2018. "Well-balanced schemes for the Euler equations with gravitation: Conservative formulation using global fluxes" [in en]. *Journal of Computational Physics* 358 (April): 36–52.

Clain, S., S. Diot, and R. Loubère. 2011. "A high-order finite volume method for systems of conservation laws—Multi-dimensional Optimal Order Detection (MOOD)" [in en]. *Journal of Computational Physics* 230, no. 10 (May): 4028–4050.

Cockburn, Bernardo, George E. Karniadakis, Chi-Wang Shu, et al., eds. 2000. *Discontinuous Galerkin Methods: Theory, Computation and Applications* [in en]. Vol. 11. Lecture Notes in Computational Science and Engineering. Berlin, Heidelberg: Springer Berlin Heidelberg.

Commerçon, B., V. Debout, and R. Teyssier. 2014. "A fast, robust, and simple implicit method for adaptive time-stepping on adaptive mesh-refinement grids." *Astronomy & Astrophysics* 563 (March): A11.

Dafermos, Constantine M. 2010. *Hyperbolic Conservation Laws in Continuum Physics.* Vol. 325. Grundlehren der mathematischen Wissenschaften. Berlin, Heidelberg: Springer Berlin Heidelberg.

De Lorenzo, M., Ph. Lafon, M. Di Matteo, et al. 2017. "Homogeneous two-phase flow models and accurate steam-water table look-up method for fast transient simulations" [in en]. *International Journal of Multiphase Flow* 95 (October): 199–219.

Dellacherie, Stéphane. 2010. "Analysis of Godunov type schemes applied to the compressible Euler system at low Mach number" [in en]. *Journal of Computational Physics* 229, no. 4 (February): 978–1016.

Després, Bruno. 2010. *Lois de Conservations Eulériennes, Lagrangiennes et Méthodes Numériques.* Vol. 68. Mathématiques et Applications. Berlin, Heidelberg: Springer Berlin Heidelberg.

Després, Bruno, and Frédéric Lagoutière. 2001. "Contact Discontinuity Capturing Schemes for Linear Advection and Compressible Gas Dynamics." *Journal of Scientific Computing* 16 (4): 479–524.

Dumbser, Michael, Olindo Zanotti, Raphaël Loubère, and Steven Diot. 2014. "A posteriori subcell limiting of the discontinuous Galerkin finite element method for hyperbolic conservation laws" [in en]. *Journal of Computational Physics* 278 (December): 47–75.

Faccanoni, Gloria, Samuel Kokh, and Grégoire Allaire. 2008. "Numerical Simulation with Finite Volume of Dynamic Liquid-Vapor Phase Transition" [in en] (June).

Featherstone, Nicholas A., and Bradley W. Hindman. 2016. "THE SPECTRAL AMPLITUDE OF STELLAR CONVECTION AND ITS SCALING IN THE HIGH-RAYLEIGH-NUMBER REGIME." *The Astrophysical Journal* 818, no. 1 (February): 32.

Fechter, Stefan, Claus-Dieter Munz, Christian Rohde, and Christoph Zeiler. 2017. "A sharp interface method for compressible liquid–vapor flow with phase transition and surface tension" [in en]. *Journal of Computational Physics* 336 (May): 347–374.

Feireisl, Eduard, and Antonín Novotný. 2017. *Singular Limits in Thermodynamics of Viscous Fluids.* Advances in Mathematical Fluid Mechanics. Cham: Springer International Publishing.

Field, J. E., J. P. Dear, and J. E. Ogren. 1989. "The effects of target compliance on liquid drop impact" [in en]. *Journal of Applied Physics* 65, no. 2 (January): 533–540.

Gallice, Gérard. 2002. "Solveurs simples positifs et entropiques pour les systèmes hyperboliques avec terme source" [in fr]. *Comptes Rendus Mathematique* 334, no. 8 (January): 713–716.

Gardner, Carl L., and Steven J. Dwyer. 2009. "Numerical simulation of the XZ Tauri supersonic astrophysical jet" [in en]. *Acta Mathematica Scientia* 29, no. 6 (November): 1677–1683.

Gastine, T., and J. Wicht. 2012. "Effects of compressibility on driving zonal flow in gas giants" [in en]. *Icarus* 219, no. 1 (May): 428–442.

Ghidaoui, Mohamed S., Ming Zhao, Duncan A. McInnis, and David H. Axworthy. 2005. "A Review of Water Hammer Theory and Practice" [in en]. *Applied Mechanics Reviews* 58, no. 1 (January): 49–76.

Gilman, P. A., and G. A. Glatzmaier. 1981. "Compressible convection in a rotating spherical shell. I - Anelastic equations. II - A linear anelastic model. III - Analytic model for compressible vorticity waves" [in en]. *The Astrophysical Journal Supplement Series* 45 (February): 335.

Glatzmaier, Gary A. 2017. *Introduction to Modeling Convection in Planets and Stars* [in en]. Vol. 1. Princeton University Press, October.

Godlewski, Edwige, and Pierre-Arnaud Raviart. 1991. *Hyperbolic systems of conservation laws* [in en]. Google-Books-ID: hzvvAAAAMAAJ. Ellipses.

———. 1996. *Numerical Approximation of Hyperbolic Systems of Conservation Laws.* Edited by J. E. Marsden, L. Sirovich, and F. John. Vol. 118. Applied Mathematical Sciences. New York, NY: Springer New York.

Godunov, Sergei K. 1959. "Finite difference method for numerical computation of discontinuous solutions of the equations of fluid dynamics." *Matematicheskii Sbornik* 47(89) (3): 271–306.

Goffrey, T., J. Pratt, M. Viallet, et al. 2017. "Benchmarking the Multidimensional Stellar Implicit Code MUSIC." *Astronomy & Astrophysics* 600 (April): A7.

Gottlieb, Sigal, Chi-Wang Shu, and Eitan Tadmor. 2001. "Strong Stability-Preserving High-Order Time Discretization Methods" [in en]. *SIAM Review* 43, no. 1 (January): 89–112.

Graham, Eric. 1975. "Numerical simulation of two-dimensional compressible convection" [in en]. *Journal of Fluid Mechanics* 70, no. 04 (August): 689.

Gresho, Philip M., and Stevens T. Chan. 1990. "On the theory of semi-implicit projection methods for viscous incompressible flow and its implementation via a finite element method that also introduces a nearly consistent mass matrix. Part 2: Implementation" [in en]. *International Journal for Numerical Methods in Fluids* 11, no. 5 (October): 621–659.

Guillard, Hervé, and Cécile Viozat. 1999. "On the behaviour of upwind schemes in the low Mach number limit" [in en]. *Computers & Fluids* 28, no. 1 (January): 63–86.

Ha, Youngsoo, Carl L. Gardner, Anne Gelb, and Chi-Wang Shu. 2005. "Numerical Simulation of High Mach Number Astrophysical Jets with Radiative Cooling" [in en]. *Journal of Scientific Computing* 24, no. 1 (July): 29–44.

Hantke, Maren, and Ferdinand Thein. 2019. "On the Impossibility of First-Order Phase Transitions in Systems Modeled by the Full Euler Equations" [in en]. *Entropy* 21, no. 11 (October): 1039.

Harten, Amiram, Peter D. Lax, and Bram van Leer. 1983. "On Upstream Differencing and Godunov-Type Schemes for Hyperbolic Conservation Laws" [in en]. *SIAM Review* 25, no. 1 (January): 35–61.

Helluy, Philippe, and Nicolas Seguin. 2006. "Relaxation models of phase transition flows." *ESAIM: Mathematical Modelling and Numerical Analysis* 40, no. 2 (March): 331–352.

Hurisse, Olivier. 2017. "Numerical simulations of steady and unsteady two-phase flows using a homogeneous model." *Computers & Fluids* 152 (July): 88–103.

Hurlburt, N. E., J. Toomre, and J. M. Massaguer. 1984. "Two-dimensional compressible convection extending over multiple scale heights" [in en]. *The Astrophysical Journal* 282 (July): 557.

Programming Guidelines for Vectorization [in en]. 2020.

Kestener, Pierre. 2017. *Implementing High-Resolution Fluid Dynamics Solver in a Performance Portable Way with Kokkos,* May.

Kjolstad, Fredrik Berg, and Marc Snir. 2010. "Ghost Cell Pattern" [in en]. In *Proceedings of the 2010 Workshop on Parallel Programming Patterns - ParaPLoP '10,* 1–9. Carefree, Arizona: ACM Press.

Kokh, Samuel, and Frédéric Lagoutière. 2010. "An anti-diffusive numerical scheme for the simulation of interfaces between compressible fluids by means of a five-equation model" [in en]. *Journal of Computational Physics* 229, no. 8 (April): 2773–2809.

Kyriazis, Nikolaos, Phoevos Koukouvinis, and Manolis Gavaises. 2018. "Modelling cavitation during drop impact on solid surfaces" [in en]. *Advances in Colloid and Interface Science* 260 (October): 46–64.

Lagoutière, Frédéric. 2000. "Modelisation mathematique et resolution numerique de problemes de fluides compressibles a plusieurs constituants." These de doctorat, Paris 6.

Lax, Peter D. 1954. "Weak solutions of nonlinear hyperbolic equations and their numerical computation" [in en]. *Communications on Pure and Applied Mathematics* 7, no. 1 (February): 159–193.

Lax, Peter, and Burton Wendroff. 1960. "Systems of conservation laws" [in en]. *Communications on Pure and Applied Mathematics* 13, no. 2 (May): 217–237.

Le Métayer, Olivier, and Richard Saurel. 2016. "The Noble-Abel Stiffened-Gas equation of state" [in en]. *Physics of Fluids* 28, no. 4 (April): 046102.

Leroux, A. Y., and P. Cargo. 1994. "Un schéma équilibre adapté au modèle d'atmosphère avec termes de gravité." *Comptes rendus de l'Académie des sciences. Série 1, Mathématique* 318 (1): 73–76.

Liang, Chunlei, Antony Jameson, and Z.J. Wang. 2009. "Spectral difference method for compressible flow on unstructured grids with mixed elements" [in en]. *Journal of Computational Physics* 228, no. 8 (May): 2847–2858.

Liu, Yen, Marcel Vinokur, and Z.J. Wang. 2006. "Spectral (finite) volume method for conservation laws on unstructured grids V: Extension to three-dimensional systems" [in en]. *Journal of Computational Physics* 212, no. 2 (March): 454–472.

Luiset, B., F. Sanchette, A. Billard, and D. Schuster. 2013. "Mechanisms of stainless steels erosion by water droplets" [in en]. *Wear* 303, nos. 1-2 (June): 459–464.

Marcello, Dominic C., and Joel E. Tohline. 2012. "A NUMERICAL METHOD FOR STUDYING SUPER-EDDINGTON MASS TRANSFER IN DOUBLE WHITE DWARF BINARIES." *The Astrophysical Journal Supplement Series* 199, no. 2 (April): 35.

Mentrelli, Andrea. 2018. "Modelling of the convective plasma dynamics in the Sun: anelastic and Boussinesq MHD systems" [in en]. *Ricerche di Matematica* (July).

Miczek, F., F. K. Röpke, and P. V. F. Edelmann. 2015. "New numerical solver for flows at various Mach numbers." *Astronomy & Astrophysics* 576 (April): A50.

Nonaka, A., A. S. Almgren, J. B. Bell, et al. 2010. "MAESTRO: AN ADAPTIVE LOW MACH NUMBER HYDRODYNAMICS ALGORITHM FOR STELLAR FLOWS." *The Astrophysical Journal Supplement Series* 188, no. 2 (June): 358–383.

Padioleau, Thomas, Pascal Tremblin, Edouard Audit, Pierre Kestener, and Samuel Kokh. 2019. "A High-performance and Portable All-Mach Regime Flow Solver Code with Well-balanced Gravity. Application to Compressible Convection." *The Astrophysical Journal* 875, no. 2 (April): 128.

Peluchon, Simon. 2017. "Approximation numérique et modélisation de l'ablation liquide." These de doctorat, Bordeaux.

Pinsonneault, M. 1997. "MIXING IN STARS" [in en]. *Annual Review of Astronomy and Astrophysics* 35, no. 1 (September): 557–605.

Quirk, James J. 1994. "A contribution to the great Riemann solver debate" [in en]. *International Journal for Numerical Methods in Fluids* 18, no. 6 (March): 555–574.

Roe, P.L. 1981. "Approximate Riemann solvers, parameter vectors, and difference schemes" [in en]. *Journal of Computational Physics* 43, no. 2 (October): 357–372.

Schaal, Kevin, Andreas Bauer, Praveen Chandrashekar, et al. 2015. "Astrophysical hydrodynamics with a high-order discontinuous Galerkin scheme and adaptive mesh refinement" [in en]. *Monthly Notices of the Royal Astronomical Society* 453, no. 4 (November): 4279–4301.

Sedov, L. I. 1959. "Similarity and Dimensional Methods in Mechanics." *Similarity and Dimensional Methods in Mechanics, New York: Academic Press, 1959.*

Sod, Gary A. 1978. "A survey of several finite difference methods for systems of nonlinear hyperbolic conservation laws" [in en]. *Journal of Computational Physics* 27, no. 1 (April): 1–31.

Sonntag, Matthias, and Claus-Dieter Munz. 2017. "Efficient Parallelization of a Shock Capturing for Discontinuous Galerkin Methods using Finite Volume Sub-cells" [in en]. *Journal of Scientific Computing* 70, no. 3 (March): 1262–1289.

Spiegel, E. A., and G. Veronis. 1960. "On the Boussinesq Approximation for a Compressible Fluid." [in en]. *The Astrophysical Journal* 131 (March): 442.

Springel, Volker. 2010. "E pur si muove: Galilean-invariant cosmological hydrodynamical simulations on a moving mesh" [in en]. *Monthly Notices of the Royal Astronomical Society* 401, no. 2 (January): 791–851.

Spruit, H. C., A. Nordlund, and A. M. Title. 1990. "Solar Convection" [in en]. *Annual Review of Astronomy and Astrophysics* 28, no. 1 (September): 263–303.

Suliciu, I. 1998. "On the thermodynamics of rate-type fluids and phase transitions. I. Rate-type fluids" [in en]. *International Journal of Engineering Science* 36, no. 9 (July): 921–947.

Sunderland, D., B. Peterson, J. Schmidt, et al. 2016. "An Overview of Performance Portability in the Uintah Runtime System through the Use of Kokkos." In *2016 Second International Workshop on Extreme Scale Programming Models and Middlewar (ESPM2),* 44–47. November.

Sutter, Herb. 2013. *The Free Lunch Is Over A Fundamental Turn Toward Concurrency in Software* [in en].

Tan, Huey Ling, and Leslie V. Woodcock. 2007. "Molecular dynamics study of a simple liquid at negative pressures" [in en]. *Journal of Molecular Liquids* 136, no. 3 (December): 281–287.

Teyssier, R. 2002. "Cosmological hydrodynamics with adaptive mesh refinement: A new high resolution code called RAMSES." *Astronomy & Astrophysics* 385, no. 1 (April): 337–364.

Toomre, Juri, Nicholas Brummell, Fausto Cattaneo, and Neal E. Hurlburt. 1990. "Three-dimensional compressible convection at low prandtl numbers" [in en]. *Computer Physics Communications* 59, no. 1 (May): 105–117.

Toro, E. F., and A. Chakraborty. 1994. "The development of a Riemann solver for the steady supersonic Euler equations" [in en]. *The Aeronautical Journal* 98, no. 979 (November): 325–339.

Toro, Eleuterio F. 2009. *Riemann Solvers and Numerical Methods for Fluid Dynamics* [in en]. Berlin, Heidelberg: Springer Berlin Heidelberg.

Tremblin, P., D. S. Amundsen, G. Chabrier, et al. 2016. "CLOUDLESS ATMOSPHERES FOR L/T DWARFS AND EXTRASOLAR GIANT PLANETS." *The Astrophysical Journal* 817, no. 2 (January): L19.

Tremblin, P., T. Padioleau, M. W. Phillips, et al. 2019. "Thermo-compositional Diabatic Convection in the Atmospheres of Brown Dwarfs and in Earth's Atmosphere and Oceans." *The Astrophysical Journal* 876, no. 2 (May): 144.

Viallet, M., I. Baraffe, and R. Walder. 2011. "Towards a new generation of multi-dimensional stellar evolution models: development of an implicit hydrodynamic code." *Astronomy & Astrophysics* 531 (July): A86.

Vides, J., B. Braconnier, E. Audit, C. Berthon, and B. Nkonga. 2014. "A Godunov-Type Solver for the Numerical Approximation of Gravitational Flows" [in en]. *Communications in Computational Physics* 15, no. 01 (January): 46–75.

Wagner, W., and A. Pruß. 2002. "The IAPWS Formulation 1995 for the Thermodynamic Properties of Ordinary Water Substance for General and Scientific Use" [in en]. *Journal of Physical and Chemical Reference Data* 31, no. 2 (June): 387–535.

Wang, Z. J., and Y. Liu. 2004. "Spectral (Finite) Volume Method for Conservation Laws on Unstructured Grids III: One Dimensional Systems and Partition Optimization" [in en]. *Journal of Scientific Computing* 20, no. 1 (February): 137–157.

Wang, Z. J., and Yen Liu. 2002. "Spectral (Finite) Volume Method for Conservation Laws on Unstructured Grids: II. Extension to Two-Dimensional Scalar Equation" [in en]. *Journal of Computational Physics* 179, no. 2 (July): 665–697.

Wang, Z.J. 2002. "Spectral (Finite) Volume Method for Conservation Laws on Unstructured Grids. Basic Formulation" [in en]. *Journal of Computational Physics* 178, no. 1 (May): 210–251.

Wang, Z.J, Laiping Zhang, and Yen Liu. 2004. "Spectral (finite) volume method for conservation laws on unstructured grids IV: extension to two-dimensional systems" [in en]. *Journal of Computational Physics* 194, no. 2 (March): 716–741.

Wu, Wangxia, Gaoming Xiang, and Bing Wang. 2018. "On high-speed impingement of cylindrical droplets upon solid wall considering cavitation effects" [in en]. *Journal of Fluid Mechanics* 857 (December): 851–877.

université PARIS-SACLAY

ÉCOLE DOCTORALE
Astronomie
et Astrophysique
d'Île-de-France (AAIF)

Titre: Développement de méthodes de simulation AMR "tout régime" pour la dynamique des fluides, applications en astrophysique et aux écoulements diphasiques

Mots clés: hydrodynamique compressible, diphasique compressible, calcul haute performance, nombre de Mach, schéma ordre élevé, convection

Résumé: Bien que performantes pour la capture des chocs, la plupart des méthodes de simulation standards ne sont pas adaptées à des régimes de Mach variés. Des méthodes numériques innovantes, utilisant des schémas de type Volumes Finis, robustes et précises uniformément selon le nombre de Mach (dites "tout régime") ont été récemment élaborées au CEA. Ces méthodes permettent de résoudre les équations de la mécanique des fluides compressibles pour capturer des chocs, mais aussi pour simuler des écoulements à très faible vitesse. Fort de ces résultats prometteurs, nous proposons dans cette thèse de mettre à l'épreuve ces nouvelles méthodes dans deux domaines applicatifs différents: les écoulements diphasiques à petit échelle et les écoulements compressibles en astrophysique. Pour ces deux domaines la simulation multi-régime est un point difficile. En effet, ces deux contextes d'applications ont pour cœur une modélisation d'écoulement compressible mais mettent en jeu des phénomènes de convection et de compressibilité à des régimes de Mach très variés. L'approche "tout régime" permettra de capturer des phénomènes très compressibles tout en gardant la précision sur les écoulements basse vitesse.

Title: Development of "all-regime" AMR simulation methods for fluid dynamics, application in astrophysics and two-phase flows

Keywords: compressible hydrodynamics, compressible two-phase flow, high performance computing, Mach number, high order scheme, convection

Abstract: Although classic simulation methods for compressible flow are efficient for shock capturing, they are not adapted to variable Mach regimes. Innovative methods using Finite Volume numerical schemes, robust and uniformly accurate with respect to the Mach number (so-called "all-regime"), were recently developed at CEA. These methods allow to solve the equations of compressible flows for both shocks capturing and flows involving very low material speed. Using the ground of these promising results, we propose within this thesis to challenge these new methods in two different application areas: small scale two-phase flows and compressible flows in astrophysics. For both contexts the multi-regime simulation is a key issue: they both rely on a compressible flow modeling but involve convection and compressibility in highly-variable Mach regimes. The "all-regime" approach is a good candidate for capturing highly compressible phenomena while preserving the accuracy in the low speed flows.

Université Paris-Saclay
Espace Technologique / Immeuble Discovery
Route de l'Orme aux Merisiers RD 128 / 91190 Saint-Aubin, France