Exaflopic simulation of spray combustion

Antoine Stock

Normandie Université

INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES ROUEN NORMANDIE

THÈSE

Pour obtenir le diplôme de doctorat

Spécialité **PHYSIQUE**

Préparée au sein de l'**INSA Rouen Normandie**

Simulation exaflopique de la combustion de sprays

Présentée et soutenue par

ANTOINE STOCK

Thèse soutenue le 15/10/2024

devant le jury composé de :

M. VINCENT MOUREAU	CHARGÉ DE RECHERCHE HDR - CORIA, Saint-Etienne-du-Rouvray	Directeur de thèse
M. BRUNO RENOU	PROFESSEUR DES UNIVERSITÉS - INSA de Rouen Normandie	Président du jury
M. GHISLAIN LARTIGUE	DOCTEUR - INSA de Rouen Normandie	Co-encadrant de thèse
M. CÉDRIC MEHL	DOCTEUR - IFPEN	Membre
M. RENAUD MERCIER	DOCTEUR - Safran Tech	Membre
M. ALESSANDRO PARENTE	PROFESSEUR - Université Libre de Bruxelles	Membre
M. BENOIT FIORINA	PROFESSEUR DES UNIVERSITÉS - Centralesupelec	Rapporteur
M. DANIEL MIRA	PROFESSEUR - Barcelona Supercomputing Center	Rapporteur

Thèse dirigée par **VINCENT MOUREAU** (COMPLEXE DE RECHERCHE INTERPROFESSIONNEL EN AEROTHERMOCHIMIE)

ED 591 PSIME

CORIA UMR 6614

Remerciments

Quatre années de science passées aux côtés de personnes formidables, sans qui cette thèse n'aurait jamais vu le jour. N'hésitez jamais à me faire signe, ma porte vous sera toujours grande ouverte.

Un merci tout particulier à toi, Vincent, pour ta présence dans les moments de joie comme dans les périodes plus difficiles. Ma reconnaissance va également à toute l'équipe : Ghislain, je compte sur toi pour de futures courses à pied ; Kévin, tu as mis des paillettes dans ma vie ; Pierre, tout le bonheur à toi et à Alix ; Léa, Julien, Yacine, François, Renaud, Mélody, Julien, merci pour nos échanges précieux. Enfin, je tiens à remercier l'ensemble du labo pour son soutien tout au long de ce parcours.

Je garde de nombreux souvenirs mémorables des moments passés entre doctorants, que ce soit à l'ECFD, en Italie, en Belgique, au Japon ou tout simplement à Rouen. Des moments de science, mais aussi des instants plus légers, nécessaires pour vider la tête. La thèse, ce n'est pas qu'une affaire de science, il faut aussi les bonnes personnes avec qui la traverser. Comme cette forêt, Ibtissam, Adrien, Pierre et Serge. J'ai peut-être raté Yann, mais vous n'étiez pas mal non plus, Felix, Gab, Clément et Téodor. Merci à vous tous, et à tous les autres ! Je croise les doigts pour ceux dont la soutenance approche : Margot, Ulysse, Andrei, Téodor, Ibtissam... mais je sais que vous allez gérer !

Un grand merci à tous mes amis en dehors du labo : Les mousquetaires, je vous préviens, je vais me remettre sérieusement à la course à pied et au vélo, et je ne vais plus là pour épiler des kiwi ! Les Rouennais, même si je quitte probablement Rouen bientôt, je reviendrai vous voir, promis, pour que nous puissions créer encore plus de souvenirs ensemble. Vous avez tous contribué à rendre cette aventure qu'est la thèse plus belle en me gardant motivé.

Une petite pensée pour Natsu, qui ronronne dans mes bras pendant que j'écris ces lignes, et pour Camille, pour toutes nos escapades en falaise.

Enfin, un immense merci à mes parents, qui ont toujours été présents et qui m'ont permis de devenir la personne que je suis aujourd'hui, ainsi que celle que je souhaite devenir.

Abstract

Keywords: Large-Eddy Simulation (LES), Two-phase Reactive Flow, Adaptive Mesh Refinement (AMR), Load-balancing, Dynamic Cell Clustering (DCC), Euler-Lagrange.

Large Eddy Simulation (LES) has emerged as a powerful computational tool for the design and analysis of spray burners, offering the ability to capture the complex interactions between turbulent flow, combustion, and spray dynamics with high fidelity. However, the high accuracy of LES comes at a significant computational cost. The simulation of these systems often requires the use of large meshes in order to resolve fine-scale turbulence and sharp flame front as well as the handling of numerous species and chemical reactions. These demands pose substantial challenges in terms of the required computational resources and the time required for the simulation process, which can hinder the practical application of LES in industrial design processes. This thesis addresses the computational challenges associated with LES of spray burners by exploring and developing numerical approaches aimed at reducing the computational burden without compromising the accuracy of the simulations. Three primary strategies are investigated: Euler-Lagrange load balancing, Adaptive Mesh Refinement (AMR), and Dynamic Cell Clustering.

First, the thesis examines the Euler-Lagrange approach, which is commonly used in LES to model the dispersed phase in spray combustion. One of the critical challenges in this approach is load balancing, where the computational workload must be evenly distributed across processors to ensure efficient parallel performance. A new load-balancing method is introduced to deal simultaneously with the Lagrangian and the Eulerian phase without compromising on one or another.

Second, the thesis explores Adaptive Mesh Refinement (AMR) as a strategy to manage the large meshes required in LES. AMR allows for the dynamic adjustment of the mesh resolution based on the local flow characteristics, refining the grid in regions of interest. Selection of an adequate criterion to accurately identify the regions of interest is of prime importance. A new criterion is introduced for reactive flows and compared to other commonly used criteria.

Finally, the thesis investigates Dynamic Cell Clustering (DCC), a technique that groups similar computational cells into clusters based on their thermochemical states. By reducing the number of unique states that need to be computed, this method can significantly cut down the overhead associated with solving the chemical kinetics, which is a major bottleneck in LES of reactive flows. Accuracy of the mapping method for DCC is increased from order 0 to 1 with negligible overhead.

Contents

CHAPTER 1

Introduction

This chapter introduces the context and objectives of the thesis.

Contents

1.1 Context

The aviation sector holds a pivotal position in modern society, serving as a vital artery of connectivity and driving economic growth. Yet, this privilege of operation comes with a profound responsibility, to mitigate the adverse impacts, such as noise pollution, air quality degradation and foremost climate change. Recent studies [1] suggests that aviation has contributed to 4% of global warning until now. It is projected to cause a total of about 0.1°C of warming by 2050. As highlighted by the Intergovernmental Panel on Climate Change (IPCC), exceeding a global warming threshold of 2.0°C poses significant risks, while striving to limit it to 1.5°C can substantially mitigate adverse effects [2]. With the current temperature rise almost over 1.2°C [3, 4], the imperative to act becomes increasingly urgent in all sectors including aviation.

Different possible scenarios for aviation in 2050 are described in [5] and recapitulated in Fig 1.1. The estimations of global warming in the scenarios take into account all sources of radiative forcing and the subsequent transient climate response [6]. The first scenario entails a societal shift towards energetic sobriety through reduced mobility of both people and goods. An annual decline of 2.5% of aviation would lead to no additional aviation-induced warming. However, the current observed growth rate of 3% per year is expected to persist in the foreseeable future as shown on Fig 1.2. Following scenarios assume the current growth rate.

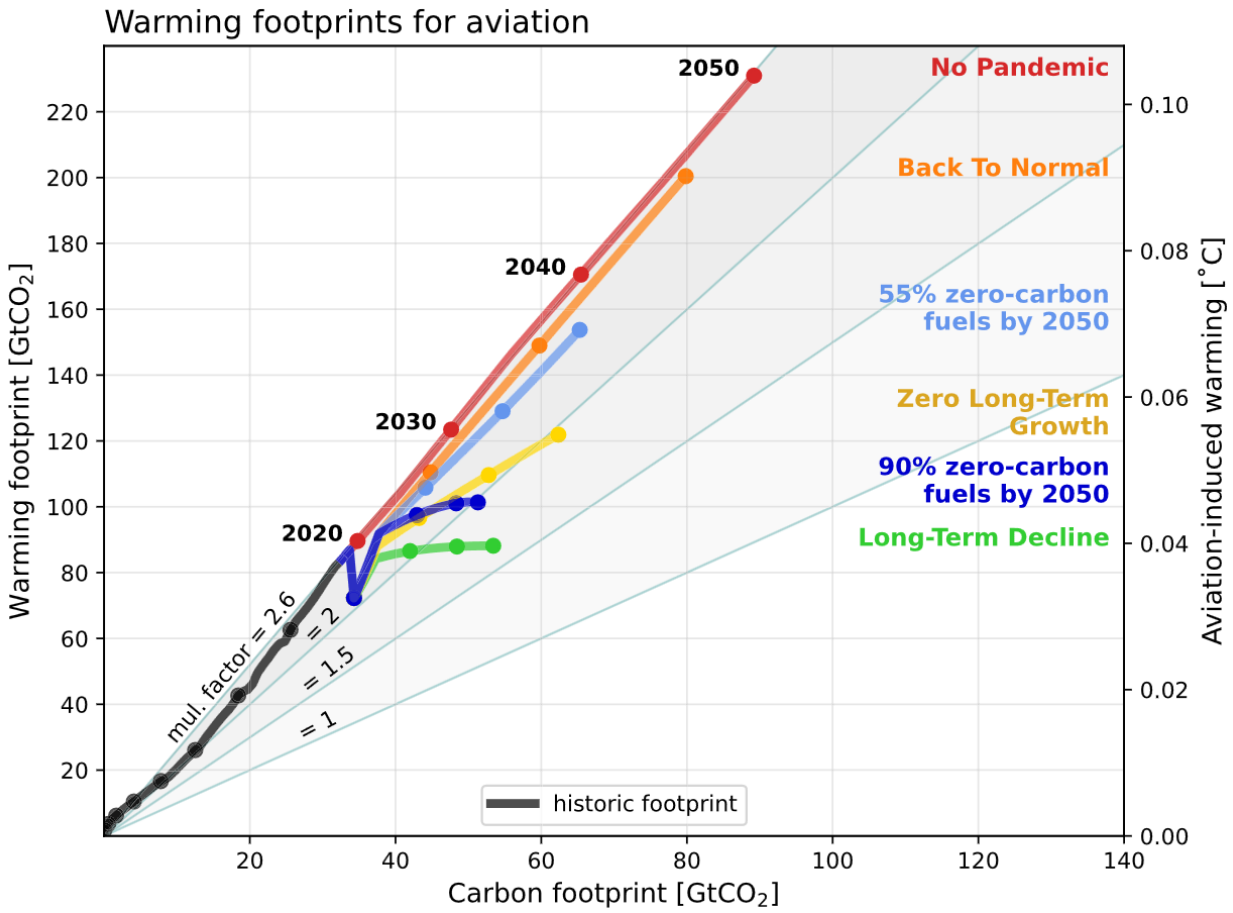


Figure 1.1: The warming footprint of aviation is a scenario and time-dependent multiplicative of its cumulative carbon footprint, about 2–2.6x larger in recent decades. Diagonal lines represent a constant multiplication factor often used in carbon footprint analyses to simplify the non-CO2 effects of aviation. Dots represent decades for all scenarios and historic emissions. Warming footprints are the cumulative CO2 warming-equivalent emissions, including both CO2 and non-CO2 effect. Reproduction from [5].

Other scenarios involve technological advancements to mitigate the warming footprint. Aircraft engines are significant emitters of CO_2 and also nitrogen oxides (NO_x). NO_x further react with the atmosphere to create additional greenhouse gases like methane (CH_4) and ozone (O_3). Furthermore, the water vapor present in exhaust gases condenses upon mixing with the atmosphere, forming contrails that may evolve into cirrus clouds. This results into another climate forcing mechanism, altering radiation absorption and reflection. Increase of engine efficiency has been achieved over the years reducing fuel consumption and thus environmental impact. This is shown in Fig. 1.3.

Progressive use of carbon-free or carbon-neutral fuels allows to limit aviation induced warming to their current state. Alternative fuels are categorized into net carbon-neutral drop-in Sustainable Aviation Fuels (SAF), and non-drop-in fuels such as hydrogen. Drop-in refers to the ability of a fuel to operate properly on current jet fuel engine architecture.

Hydrogen-based engines present promising solutions for emission reduction, yet their technological

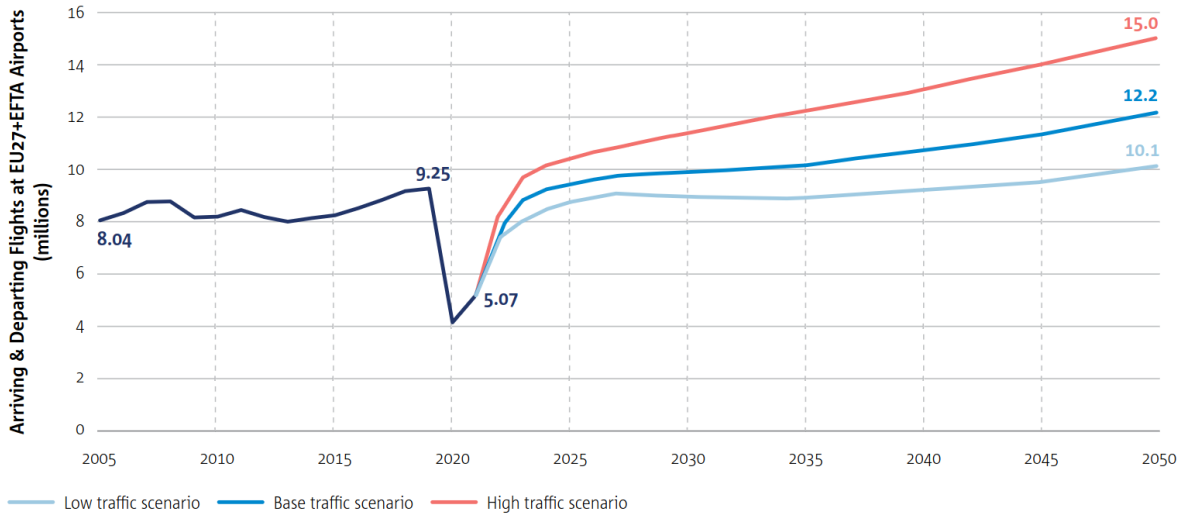


Figure 1.2: Expected increase of demand of flights. Reproduction from [7].

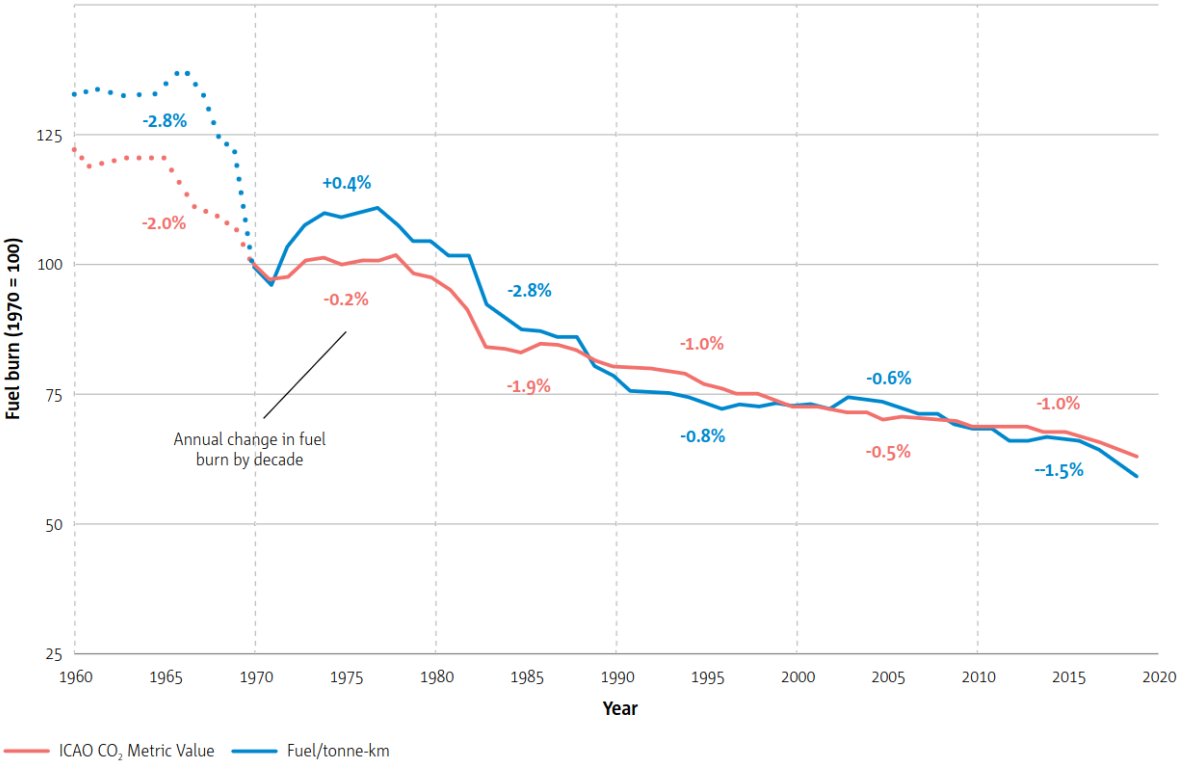


Figure 1.3: Decrease of aircraft fuel consumption over the years of new commercial jet aircraft. Reproduction from [7].

readiness for the 2050 objectives is questionable. The first large commercial planes are announced for the mid 30s but the impact on the global emission is still expected to be very small by 2040 [8]. One of the main challenges in hydrogen-powered aircraft architecture is the hydrogen storage [9]. In addition to mass requirements, volume has to be taken into account as hydrogen volumetric energetic density is several factors lower than jet fuel. High pressure tanks, cryogenic tanks and

chemical storage methods are studied to provide improved volumetric energetic densities. These methods introduce extra complexity and weight with hydrogen currently representing at best 30% of the gravimeter storage density.

$$\eta_{gravi} = \frac{m_{fuel}}{m_{fuel} + m_{tank}} . \tag{1.1}$$

Another consideration is the tank shape, which ideally is spherical for better structural integrity and reduced heat exchanges. This makes it harder to be fit within the wing and alternative solutions are explored. As hydrogen is a non-drop-in fuel, the whole engine design has also to be adapted and validated to fit to the hydrogen fuel properties. This includes accounting for gaseous phase injection, increased maximum adiabatic temperatures and increased flame speeds. Hydrogen also provides a mean to electrical engines through the use of hydrogen fuel cells.

Sustainable Aviation Fuels (SAF) emerge as a more immediate solution to carbon emissions. SAF are produced from renewable or sustainable resources, in contrast to conventional aviation fuels which are derived from fossil fuels such as crude oil. The life cycle of drop-in SAF includes a step of carbon capture from the air, through growth of biomass or industrial processes. The amount of captured carbon is roughly equivalent to the amount of carbon dioxide produced when the fuel is burned in a combustion engine. This allows for carbon neutrality, excluding transport and processing steps to obtain the SAF. Numerous SAF production pathways exist and are presented in Fig 1.4.

Production pathway	Feedstocks[30]	Certification name (blending limit)	TRL
Biomass Gasification + Fischer-Tropsch (Gas+FT)	Energy crops, lignocellulosic biomass, solid waste	FT-SPK[31] (up to 50%)	7-8
Hydroprocessed Esters and Fatty Acids (HEFA)	Vegetable and animal fat	HEFA-SPK (up to 50%)	8-9
Direct Sugars to Hydrocarbons (DSHC)	Conventional sugars, lignocellulosic sugars	HFS-SIP[32] (up to 10%)	7-8 or 5[33]
Biomass Gasification + FT with Aromatics	Energy crops, lignocellulosic biomass, solid waste	FT-SPK/A[34] (up to 50%)	6-7
Alcohols to Jet (AtJ)	Sugar, starch crops, lignocellulosic biomass	ATJ-SPK (up to 50%)	7-8
Catalytic Hydrothermolysis Jet (CHJ)	Vegetable and animal fat	CHJ or CH-SK[35] (up to 50%)	6
HEFA from algae	Microalgae oils	HC-HEFA-SPK[36] (up to 10%)	5
FOG Co-processing	Fats, oils, and greases	FOG (up to 5 %)	-
FT Co-processing	Fischer-Tropsch (FT) biocrude	FT (up to 5 %)	-

Figure 1.4: Drop-in SAF from feedstocks approved production pathways, reproduction from [7].

Spray burners are devices used to atomize liquid fuel, such as SAF, into fine droplets and mix them with air to produce a combustible mixture for efficient combustion. Creation of small droplets drastically increases the surface to volume ratio of the fuel leading to a better evaporation of the fuel. Once evaporated the fuel vapor mixes with the air resulting in an ignitable mixture which is combusted. The combustion process is continuous over time, unlike internal combustion engines revolving around different cycles. Spray flames are commonly used in various applications, including industrial furnaces, boilers, and aircraft engines. Fig. 1.5 provides a schematic of jet engine to show how the spray flame is integrated within the engine.

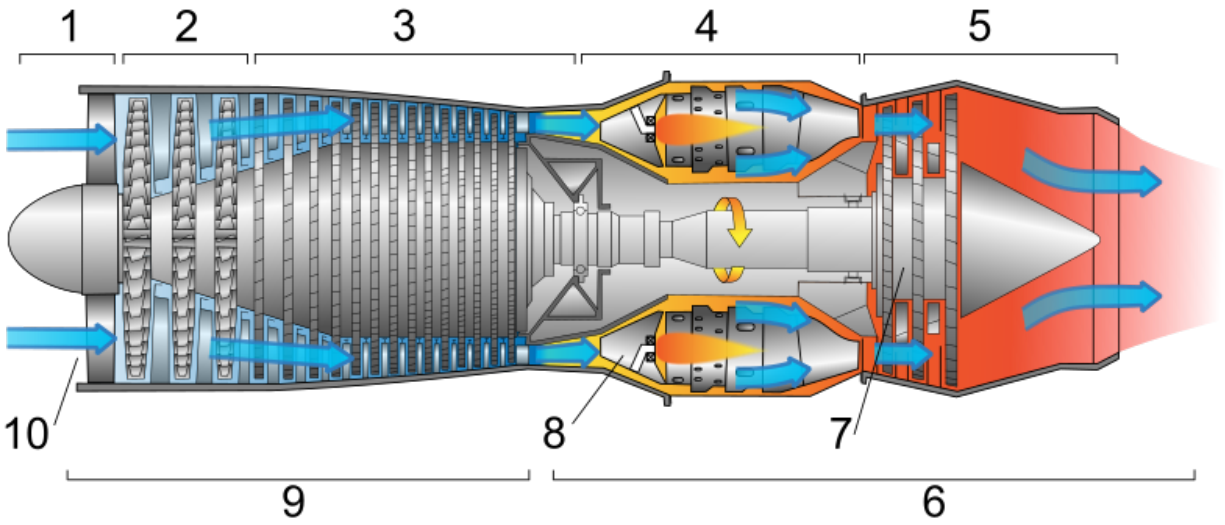


Figure 1.5: Diagram of a typical gas turbine jet engine. Air is compressed by the compressor blades as it enters the engine, and it is mixed and burned with fuel in the combustion section. The hot exhaust gases provide forward thrust and turn the turbines which drive the compressor blades. 1. Intake 2. Low pressure compression 3. High pressure compression 4. Combustion 5. Exhaust 6. Hot section 7. Turbines Low and High pressure 8. Combustion chamber with spray flame 9. Cold section 10. Air inlet.

The combustion chamber is equipped with multiple spray burners arranged in an annular configuration. Simulating the entire chamber provides insights into the interactions between these burners, particularly during ignition and flame propagation. However, this study focuses on a single spray burner to explore the wide array of phenomena associated with its operation. Fig. 1.6 depicts a spray burner and the associated phenomena. When liquid fuel is injected through the nozzle, it undergoes primary atomization: the high-pressure fuel stream disintegrates into droplets due to hydrodynamic instabilities and surface tension. This atomization process is influenced by the fuel's viscosity, density, velocity, and the nozzle's design. Following primary atomization, secondary atomization occurs as larger droplets break down further due to aerodynamic forces from the surrounding air. This results in a diverse range of droplet sizes. Concurrently, the atomized fuel mixes with the air supply, a process driven by turbulent flow dynamics that enhance the fuel-air mixture's homogeneity and mixing efficiency. Combustion begins when the fuel droplets evaporate in the high-temperature environment, converting into a gaseous state where chemical reactions can take place. Key factors such as local temperature, pressure, and equivalence ratio significantly impact the reaction rates, flame speed, and stability. Turbulent eddies can stretch and fold the flame, which enhances mixing and reaction rates but may also create local extinctions or introduce poten-

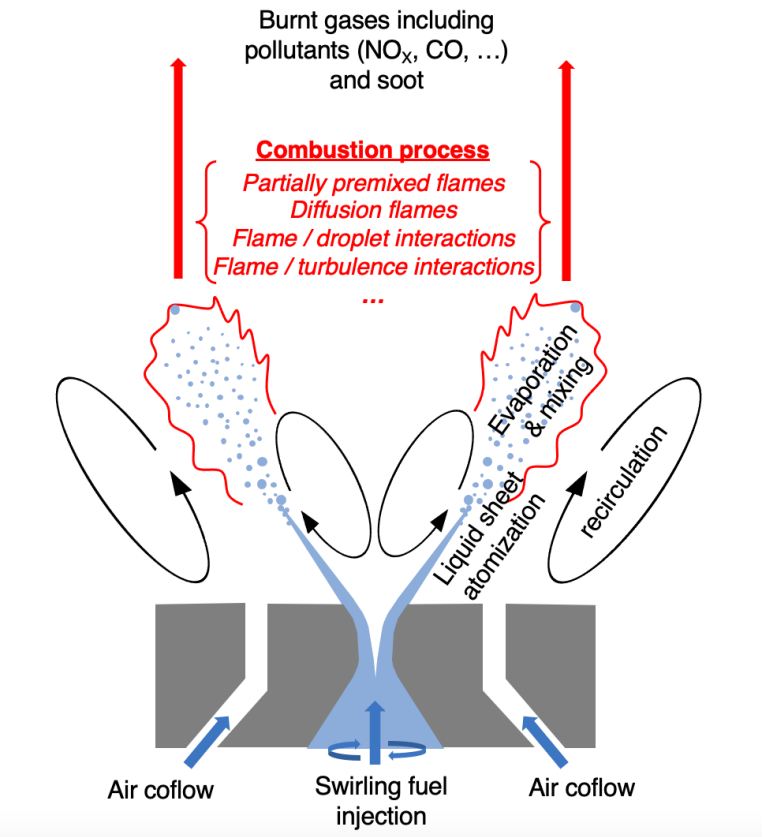


Figure 1.6: Spray burner scheme.

tial instabilities. Additionally, other phenomena such as thermoacoustic instabilities and radiative heat transfer can play a significant roles in the operation of spray burners.

Computational Fluid Dynamics (CFD) provides a powerful framework for simulating the intricate fluid dynamics and combustion phenomena within spray burners. Among CFD techniques, Large-Eddy Simulation (LES) offers a more detailed approach by resolving large-scale turbulent structures while modeling the smaller scales. This makes LES particularly effective for capturing the unsteady, turbulent flows characteristic of spray burners, providing precise insights into how turbulence interacts with combustion. However, as the level of detail in LES increases, so does the computational cost, potentially reaching millions of CPU hours. Therefore, achieving a balance between accuracy and available computational resources is crucial for making these simulations viable in an industrial setting.

1.2 Objectives of the thesis

This thesis concentrates on numerical optimizations of spray burners to reduce LES simulation costs. It is structured into the following chapters:

Chapter 2: Two-phase reactive flows for spray combustion

This chapter introduces the fundamentals of reactive flows for spray combustion. Thermodynamic properties of mixtures are detailed and flame regimes are explained to the reader. Finally, the canonical reference flame, the Coria Rouen Spray Burner (CRSB) is given and its underlying physics are analyzed.

Chapter 3: Large-Eddy Simulation of two-phase Euler-Lagrange reactive flows

Chapter 3 introduces the theoretical background of CFD simulation strategies for spray burners. The continuous phase is solved with an Eulerian approach. The dispersed phase is solved with a coupled Lagrangian solver. The numerical set-up of the CRSB is detailed and the associated CPU costs are given. This allows to raise the cost issues that are addressed in the following chapters.

Chapter 4: Load-balancing of Euler-Lagrange spray burner simulations

The challenges posed by massively parallel simulations are discussed in this chapter. An introduction to parallel computing is provided and details are given of how CFD simulations can be run on a parallel system. A major concern is the distribution of the workload across the computing resources. This aspect is first discussed for Eulerian simulations, which is a well covered topic. When considering Euler-Lagrange simulations, traditional methods fail to provide good load balancing by either prioritising the Eulerian mesh or the Lagrangian particles and neglecting the other. A novel load balancing method for Euler-Lagrange simulations is introduced and validated. The method allows to significantly improve the Lagrangian load balance while maintaining a near ideal Eulerian load balance. This method is then applied to the CRSB configuration.

Chapter 5: Adaptive mesh refinement for reactive flows

The cost of CFD simulations scales almost linearly with the number of control volumes to be resolved, referred to as mesh complexity. In order to increase the fidelity of the simulations, ever smaller control volumes are required, resulting in prohibitive CPU costs. When designing the mesh, small control volumes should only be placed in relevant areas where small scale phenomena occur, thus reducing the cost. This requires significant user experience or well-defined criteria capable of automatically and reliably detecting regions of interest. The use of criteria can be coupled with Adaptive Mesh Refinement (AMR), which allows the computational mesh to be modified on the fly during a simulation. The dynamic generation of a mesh allows for a significant reduction in mesh size, as only the instantaneous regions of interest need to be resolved, as opposed to static meshes that need to account for regions of interest over the entire runtime. Criteria for reactive flows are presented and compared in this chapter. A new criterion based on scale similarity of filtered reaction rates is presented. This criterion is validated on a hydrogen-air triple flame and outperforms other tested criteria. Finally, AMR is performed on the CRSB using the new criterion and allows the mesh size to be reduced by half.

Chapter 6: Reduction of dimensionality and complexity of chemistry

The integration of chemical source terms on a control volume using finite-rate chemistry is a costly operation due to the high dimensionality of the phase space in which the species and temperature evolve. Moreover, the wide range of chemical timescales necessitates the use of a stiff integrator

which brings more dimensionality to the integrated source terms. The optimization of source term computation has the potential to reduce the cost of reactive flows to a significant extent. Several approaches exist, which can be distinguished into reductions of dimensionality and complexity. The reduction of dimensionality consists in reducing the number of species and reactions. A reduction in complexity can be achieved by reducing the total number of stiff integrations performed. This is studied in detail through the use of dynamic cell clustering, which identifies control volumes that can share a single stiff integration operation due to their similarity in composition and thermodynamic state. The implementation of an efficient dynamic cell clustering algorithm is discussed. The state of the art is improved by the introduction of a correction term based on the small dissimilarities of control volumes that have been clustered together. This improvement is validated on canonical flames and finally on the CRSB.

Chapter 7: General conclusion

This section sums up the optimizations that where performed on the CRSB. Perspectives of future optimization are discussed.

1.3 Work dissemination

1.3.1 Peer-reviewed publications

- A. Stock, G. Lartigue and V. Moureau. *Diffusive orthogonal load balancing for Euler–Lagrange simulations.* International Journal for Numerical Methods in Fluids, 2023.
- A. Stock and V. Moureau. *Feature-based adaptive mesh refinement for multi-regime reactive flows.* Proceedings of the Combustion Institute, 2024.
- A. Stock, V. Moureau, J. Leparoux and R. Mercier. *Low-cost Jacobian-free mapping for dynamic cell clustering in multi-regime reactive flows.* Proceedings of the Combustion Institute, 2024.

1.3.2 Conferences as speaker

- A. Stock, G. Lartigue and V. Moureau. *Diffusion based load-balancing method for massively parallel Euler-Lagrange simulations.* 33rd Parallel CFD International Conference (PARCFD), 2022.
- A. Stock, G. Lartigue, V. Moureau and R. Mercier. *Feature-based adaptive mesh refinement of reactive flows using PCA.* 11th European Combustion Meeting (ECM), 2023.
- A. Stock, B. Renou, A. Both, T. Zhang, et al. *Benchmark of CFD codes on the CORIA Rouen Spray Burner: Spray combustion, flame stability and pollutant emissions.* 8th International Workshop on Turbulent Combustion of Sprays (TCS), 2023.
- A. Stock, B. Renou, A. Both, T. Zhang, et al. *Benchmark of CFD codes on the CORIA Rouen Spray Burner: Spray combustion, flame stability and pollutant emissions.* Journée de la Combustion Turbulente (JCT), 2023.
- A. Stock, V. Moureau, J. Leparoux and R. Mercier. *Accelerating spray combustion simulations via adaptive mesh refinement and clustering*, Journée de la Combustion Turbulente (JCT), 2024.
- A. Stock, V. Moureau, J. Leparoux and R. Mercier. *Dynamic cell clustering with PCA for massively parallel multi-regime reactive flows,* 19th International Conference on Numerical Combustion (ICNC), 2024.
- A. Stock and V. Moureau. *Feature based adaptive mesh refinement for multi-regime reactive flows.* CI's 40th International Symposium, 2024.
- A. Stock, V. Moureau, J. Leparoux and R. Mercier. *Low-cost Jacobian-free mapping for dynamic cell clustering in multi-regime reactive flows.* CI's 40th International Symposium, 2024.

CHAPTER 2

Two-phase reactive flows for spray combustion

This chapter provides the theoretical background of reactive flows with focus on spray burners. A canonical reference flame, the CORIA Rouen Spray Burner (CRSB) is introduced and serves as a demonstrator in the following chapters.

Contents

2.1 Thermodynamic properties of gaseous mixtures

2.1.1 Mixture properties

A chemical mixture refers to a combination of two or more substances that are physically combined but not chemically bonded together. In a mixture, each individual substance retains its own chemical properties and characteristics. Mixtures can be composed of elements, compounds, or a

combination of both. The composition of a mixture is expressed using the mass fractions Y_k or molar fraction X_k with k ranging from 1 to the number of species N_{sp}.

$$Y_k = \frac{m_k}{m}\,, \tag{2.1}$$

$$X_k = \frac{n_k}{n}\,, \tag{2.2}$$

with m the mass, n the number of mole. The sum of these fractions is always equal to 1.

$$\sum_{k=1}^{N_{sp}} Y_k = 1\,, \tag{2.3}$$

$$\sum_{k=1}^{N_{sp}} X_k = 1\,. \tag{2.4}$$

Molar concentration $[X_k]$ can be used as well to represent a mixture:

$$[X_k] = \rho\frac{n_k}{m} = \rho\frac{X_k}{W} = \rho\frac{Y_k}{W_k}\,. \tag{2.5}$$

However, it doesn't verify a handy property like Eq. 2.3 and Eq. 2.4 but it is used to express reaction rates when chemistry occurs.

The density ρ of a control volume V holding the mass m is defined as:

$$\rho = \frac{m}{V}\,, \tag{2.6}$$

for a multispecies gas density can be expressed as the sum of the partial densities of all species:

$$\rho = \sum_{k=1}^{N_{sp}} \rho_k = \sum_{k=1}^{N_{sp}} \frac{m_k}{V}\,. \tag{2.7}$$

2.1.2 Enthalpy and internal energy of a mixture

Mass enthalpy h is a thermodynamic property of a system that accounts for its internal energy content and the pressure and volume of the system. It can be decomposed into two contributions.

$$h_k = h_{s,k} + \Delta h^0_{f,k}\,, \tag{2.8}$$

with $\Delta h^0_{f,k}$ the standard enthalpy of formation of a species k at reference temperature T_0, usually taken as $T_0 = 298K$. $\Delta h^0_{f,k}$ represents the energy required to for a species to form from its constituent elements in their standard states. This value cannot be computed but several tabulations from experimental data exist. $h_{s,k}$ is the sensible enthalpy, also called sensible heat, of species k and refers to the portion of the total enthalpy change in a species that results from a change in its

temperature while it remains in the same phase, it is expressed as:

$$h_{s,k} = \int_{T_0}^{T} c_{p,k}\left(T'\right) dT' \,, \tag{2.9}$$

with c_p the mass heat capacity at constant pressure. Enthalpy is used for system with constant pressure, for systems at constant volume an analogous formula based on mass internal energy exists:

$$e_k = e_{s,k} + \Delta h^0_{f,k} = \int_{T_0}^{T} c_{v,k}\left(T'\right) dT' - \frac{RT_0}{W_k} + \Delta h^0_{f,k} \,, \tag{2.10}$$

with c_v the heat capacity at constant volume. When considering spray flames, enthalpy is used as the pressure is assumed to be constant. Therefore only enthalpy will be used in the present work. Values of enthalpy and mass heat capacity are commonly obtained from the NASA polynomials [10]:

$$\begin{cases} c_{\mathrm{p},k} = R\left(a_1 + a_2 T + a_3 T^2 + a_4 T^3 + a_5 T^4\right) \\ h_k = R\left(a_1 T + \frac{a_2}{2} T^2 + \frac{a_3}{3} T^3 + \frac{a_4}{4} T^4 + \frac{a_5}{5} T^5 + a_6\right) \,. \end{cases} \tag{2.11}$$

The enthalpy of the mixture is obtained as the mass weighted sum of the enthalpies of species:

$$h = \sum_{k=1}^{N_{sp}} Y_k h_k \,, \tag{2.12}$$

$$h_s = \sum_{k=1}^{N_{sp}} Y_k h_{s,k} \,. \tag{2.13}$$

2.1.3 Ideal Gas law

The ideal gas law hypothesis is recalled as:

$$P = \rho r T \,, \tag{2.14}$$

with:

$$r = \frac{R}{W} \,. \tag{2.15}$$

2.2 Transport and diffusive properties

Spatial propagation of species and energy can be characterized based on the following properties: diffusion D, conduction λ, and viscosity μ. These properties arise from the movements of fluid particles that make up the studied mixture. Diffusion is a mass transport process driven by concentration gradients, while viscosity can be considered as a momentum transport resulting from velocity gradients, and heat conduction is a heat transport caused by temperature gradients.

2.2.1 Complex transport

These properties are computed based on kinetic theory of gases. Viscosity arises from the collision frequency of the gas particles. Under the assumption of a pure ideal gas of species k, spherical non-deformable particles and a Newtonian fluid it can be shown that the dynamic viscosity reads:

$$\mu_k\left(T\right) = \frac{5}{16}\frac{\left(\pi m_k k_b T\right)^{\frac{1}{2}}}{\pi\sigma_k^2\Omega_{kk}}\,, \tag{2.16}$$

where k_b is the Boltzmann constant, m_k is the particle mass, σ_k the length scale of a particle and Ω_{kk} the collision integral of species k in a pure gas. More details can be found in [11, 12]. For a mixture, the Wilke formula [13] is used:

$$\mu = \sum_{i=1}^{N_{sp}} \frac{X_i\mu_i}{\sum\limits_{j=1}^{N_{sp}} X_j\Phi_{ij}}\,, \tag{2.17}$$

with Φ_{ij}:

$$\Phi_{ij} = \frac{1}{\sqrt{8}}\frac{1}{\sqrt{1+W_i/W_j}}\left(1+\sqrt{\frac{\mu_i}{\mu_j}}\left(\frac{W_i}{W_j}\right)^{1/4}\right)\,. \tag{2.18}$$

Thermal conductivity of a species k can be derived as:

$$\lambda_k = \frac{25}{32}\frac{\sqrt{\pi m_k k_b T}}{\pi\sigma_k^2}\cdot C_{v,k}\,. \tag{2.19}$$

The thermal conductivity of a mixture can be computed using the empirical law of Brokaw [14]:

$$\lambda = \frac{1}{2}\left(\sum_{k=1}^{N_{sp}} X_k\lambda_k + \left(\sum_{k=1}^{N_{sp}}\frac{X_k}{\lambda_k}\right)^{-1}\right)\,. \tag{2.20}$$

Binary diffusion coefficients are expressed as:

$$\mathcal{D}_{ij}(T) = \frac{3}{16}\frac{\left(2\pi k_B^3 T^3/\mathfrak{m}_{ij}\right)^{\frac{1}{2}}}{\rho\pi\sigma_{ij}^2\Omega_{ij}}\,, \tag{2.21}$$

with

$$\sigma_{ij} = \frac{\sigma_i+\sigma_j}{2}\,. \tag{2.22}$$

Species diffusion velocity, V_k can be obtained from the system of Williams [15]:

$$\nabla X_k = \sum_{i=1}^{N}\frac{X_k X_i}{\mathcal{D}_{ki}}\left(V_i - V_k\right) + \left(Y_i - X_i\right)\frac{\nabla P}{P} + \frac{\rho}{p}\sum_{i=1}^{N} Y_k Y_i\left(f_k - f_i\right) \quad \text{for} \quad k = 1, N_{sp}\,. \tag{2.23}$$

The Soret effect (the diffusion of mass due to temperature gradients) is neglected.

2.2.2 Simplified transport

Complex transport properties are expensive to compute thus simplified models have been introduced. Dynamic viscosity can be computed from a reference value using Sutherland's law [16]

$$\mu = \mu_{ref} \frac{T_{ref} + C}{T + C} \left(\frac{T}{T_{ref}} \right)^{3/2} , \tag{2.24}$$

or a simple power law:

$$\mu = \mu_{\text{ref}} \left(\frac{T}{T_{\text{ref}}} \right)^{b} . \tag{2.25}$$

Thermal conductivity is obtained by setting a Prandtl number:

$$\text{Pr} = \frac{\mu C_p}{\lambda} . \tag{2.26}$$

Solving Eq. 2.23 for species diffusion velocities is expensive as all species are coupled one to another which implies a matrix inversion of size $N_{sp} \times N_{sp}$. A simplified approach has been suggested by Curtiss and Hirschfelder [17]:

$$V_{k,i} X_k = -D_k \frac{\partial X_k}{\partial x_i} , \tag{2.27}$$

with D_k the diffusion coefficient of a species into the rest of the mixture.

$$D_k = \frac{1 - Y_k}{\sum_{j \neq k} X_j / D_{jk}} . \tag{2.28}$$

Diffusion coefficients can also be obtained by setting a Schmidt number:

$$D_k = \frac{\mu}{\rho S_{c_k}} . \tag{2.29}$$

2.3 Chemical kinetics

The composition of a mixture changes over time according to chemical kinetics. Consider a chemical system with N_{sp} species reacting via N_r elementary reactions.

$$\sum_{k=1}^{N_{sp}} \nu'_{kj} \mathcal{M}_k \rightleftharpoons \sum_{k=1}^{N_{sp}} \nu''_{kj} \mathcal{M}_k \quad \text{for} \quad j = 1, N_r , \tag{2.30}$$

where $\mathcal{M}_k$ is a symbol for species k, ν'_{kj} and ν''_{kj} are the molar stoichiometric coefficients of species k in reaction j. Mass conservation enforces:

$$\sum_{k=1}^{N_{sp}} \nu'_{kj} W_k = \sum_{k=1}^{N_{sp}} \nu''_{kj} W_k \quad \text{for} \quad j = 1, N_r . \tag{2.31}$$

Mass reaction rates, $\dot{\omega}_k$, are used to express the rate of change of the mixture species. Mass conservation is verified as:

$$\sum_{k=1}^{N_{sp}} \dot{\omega}_k = 0\,. \tag{2.32}$$

Mass reaction rates of a species is the sum of the rates through each individual reaction:

$$\dot{\omega}_k = \sum_{j=1}^{N_r} \dot{\omega}_{kj} = W_k \sum_{j=1}^{N_r} \nu_{kj} \mathcal{Q}_j\,, \tag{2.33}$$

with $\nu_{kj} = \nu''_{kj} - \nu'_{kj}$ and $\mathcal{Q}_j$ the rate of progress of reaction j:

$$\mathcal{Q}_j = K_{fj} \prod_{k=1}^{N_{sp}} [X_k]^{\nu'_{kj}} - K_{rj} \prod_{k=1}^{N_{sp}} [X_k]^{\nu''_{kj}}\,, \tag{2.34}$$

$\mathcal{Q}_j$ is dependent on the molar concentration of the species involved and K_{fj} and K_{rj}, the forward and reverse rate constants of reaction j. The forward rate constant is commonly expressed using Arrhenius' law:

$$K_{fj} = A_{fj} T^{\beta_j} \exp\left(-\frac{E_j}{RT}\right)\,, \tag{2.35}$$

where constants A_{fj}, β_j and the activation energy E_j are empirical. The backward rate constant K_{rj} is obtained with the equilibrium constants:

$$K_{rj} = \frac{K_{fj}}{\left(\frac{P_0}{RT}\right)^{\sum_{k=1}^{N_{sp}} \nu_{kj}} \exp\left(\frac{\Delta S_j^0}{R} - \frac{\Delta H_j^0}{RT}\right)}\,, \tag{2.36}$$

with $P_0 = 1$ bar, ΔH_j^0 and ΔS_j^0 the standard enthalpy and standard entropy change due to reaction j:

$$\Delta H_j^0 = \sum_{k=1}^{N_{sp}} \nu_{kj} H_k^0\,, \tag{2.37}$$

$$\Delta S_j^0 = \sum_{k=1}^{N_{sp}} \nu_{kj} S_k^0\,. \tag{2.38}$$

In addition to the above elementary reactions, more complex reactions can take place. The three-body reaction involves an additional species M that acts as a collision partner and transfers energy to absorb excess energy leading to the stabilization of a product or providing energy to break up a reactant. Species M is preserved through the reaction:

$$\mathcal{Q}_j = [M] \left(K_{fj} \prod_{k=1}^{N_{sp}} [X_k]^{\nu'_{kj}} - K_{rj} \prod_{k=1}^{N_{sp}} [X_k]^{\nu''_{kj}} \right)\,, \tag{2.39}$$

M is composed of inert species within the reaction. Some species are better collision partner that

other introducing the empirical efficiency ϵ_k:

$$[M] = \sum_{k=1}^{N_{sp}} \epsilon_k [X_k] \,. \tag{2.40}$$

Some reactions are dependent on pressure, two pressure limits can be defined. In the low pressure limit, a third-body collision is required to provide the energy necessary for the reaction to proceed. In the high pressure limit, the same reaction occurs as an elementary reaction. When the reaction is between the limits, in the fall-off region, a fall-off reaction takes place. The most basic formulation is Lindemann's approach [18]. k_0 is the low pressure limit reaction rate:

$$k_0 = A_0 T^{\beta_0} exp\left(\frac{-E_0}{RT}\right) , \tag{2.41}$$

k_∞ is the high pressure limit reaction rate:

$$k_\infty = A_\infty T^{\beta_\infty} exp\left(\frac{-E_\infty}{RT}\right) . \tag{2.42}$$

The fall-off reaction rate is defined as:

$$K_f = \frac{k_0[M]}{1 + \frac{k_0[M]}{k_\infty}} \,. \tag{2.43}$$

The reduced pressure is defined as:

$$P_r = \frac{k_0[M]}{k_\infty} \,, \tag{2.44}$$

which leads to:

$$K_f = k_\infty \left(\frac{P_r}{1 + P_r}\right) . \tag{2.45}$$

More accurate formulations introduce an additional function $F\left(T, P_r\right)$:

$$K_f = k_\infty \left(\frac{P_r}{1 + P_r}\right) F\left(T, P_r\right) \,. \tag{2.46}$$

The Troe fall-off function [19] is widely used:

$$\log_{10} F\left(T, P_r\right) = \frac{\log_{10} F_{\text{cent}}\left(T\right)}{1 + f_1^2} \,, \tag{2.47}$$

with

$$\begin{aligned} F_{\text{cent}}\left(T\right) =& (1 - A) \exp\left(-T/T_3\right) + A \exp\left(-T/T_1\right) + \exp\left(-T_2/T\right) \\ f_1 =& \left(\log_{10} P_r + C\right) / \left(N - 0.14 \left(\log_{10} P_r + C\right)\right) \\ C =& -0.4 - 0.67 \log_{10} F_{\text{cent}} \\ N =& 0.75 - 1.27 \log_{10} F_{\text{cent}} \,. \end{aligned} \tag{2.48}$$

Other notable functions are Tsang's approximation to F_{cent} [20] and the SRI falloff function [21, 22].

2.4 Combustion regimes

Combustion regimes describe the various flame types that emerge from the intricate interplay between reaction rates and the transport of species and energy. Two main combustion regimes are to be distinguished:

- **Partially premixed flames**, shown in Fig 2.1, occur when fuel and oxidizer are mixed at molecular level before entering the reaction zone. When oxidizer is present in excess, the flame is considered lean and when fuel is present in excess, the flame is considered rich. At stoichiometry, all the oxidizer and fuel are burnt. In this case, the flame is simply referred to as premixed flame. The flame propagates from burnt gases (noted 2) towards fresh gases (noted 1). Laminar flame speed, s_L depends on the fuel type, increases with initial temperature, decreases with pressure and increases towards stoichiometry. It can be expressed as the ratio between fuel density ρY_F^1 and fuel consumption over the flame brush:

$$s_L = -\frac{1}{\rho Y_F^1} \int_{-\infty}^{+\infty} \dot{\omega}_F \, \mathrm{d}x \,. \tag{2.49}$$

  Laminar flame thickness δ_L [23] is the characteristic width of a laminar flame front, marking the transition from unburned to burned gases. It can be defined based on the thermodynamic properties as [24]:

$$\delta_{L,D} = \frac{1}{s_L} \left(\frac{\lambda}{\rho C_p} \right)_1 . \tag{2.50}$$

  It can also be obtained from the sharpest temperature gradient, initial temperature T_1 and maximum temperature T_2:

$$\delta_L = \frac{T_2 - T_1}{|\frac{\partial T}{\partial x}|_{max}} . \tag{2.51}$$

  These two thicknesses are different but related.

- **Non-premixed flames** also referred as **Diffusion flame**, shown in Fig 2.2, occur when oxidizer and fuel are separated before burning. At the interface, oxidizer and fuel are mixed by molecular diffusion and advection processes. Upon mixing, a burnable mixture is obtained and combusted. The flame does not propagate but is controlled by the mixing rate, therefore it doesn't have a characteristic flame speed. Flame thickness varies strongly depending on local transport properties and mixing.

Some more complex combustion regimes that share similarities with both diffusion and partially premixed regimes can occur:

- **Stratified flames** result when an heterogeneous premixed mixture is burnt. The partially premixed mixture is consumed and diffusion takes place along the mixture gradients. Diffusion takes place in the tangential direction to the flame front.

- **Backward Diffusion Supported Partially Premixed Flames** occur when a premixed flame is supported by fuel diffusion from the burnt gases. Diffusion takes place in the normal direction to the flame front.

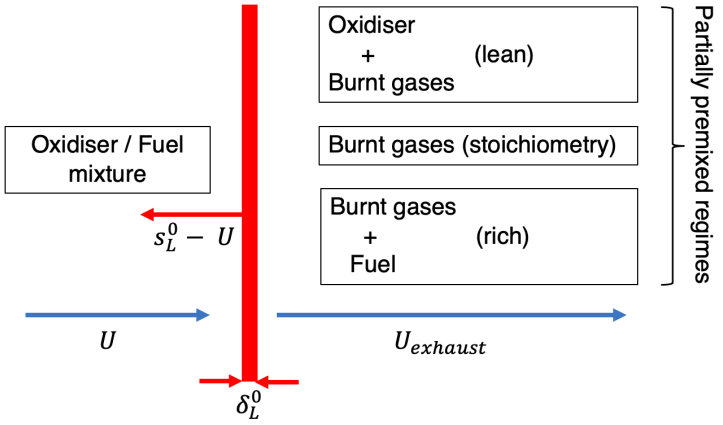


Figure 2.1: Partially premixed flames.

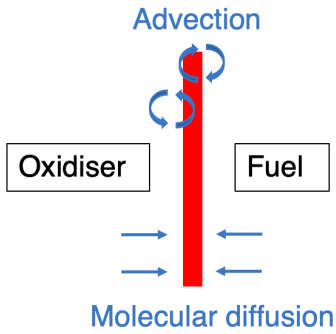


Figure 2.2: Non-premixed / diffusion flame.

2.5 Reactive flow variables

While the phase space made of species and temperature is very large, the evolution in this phase space is often limited and simplified scalar representations can be used to describe a flame instead of species and temperature. These scalars are directly related to main physical processes in the reactive flow.

Progress variable

The progress variable is a scalar used to track the advancement of the chemical reactions and combustion processes. It is usually equal to 0 in unburnt gasses and 1 in burnt gases. The choice of progress variable depends on the specific reaction mechanism being modeled. It can be expressed as a linear combination of species mass fractions. For instance, for a methane-air flame, a progress variable can be:

$$Y_C = Y_{CO2} + Y_{CO} + Y_{H2O} + Y_{H2}\,. \tag{2.52}$$

Temperature itself can serve as a progress variable, especially in cases where the reaction rate is highly temperature-dependent:

$$c = \frac{T - T_u}{T_b - T_u}\,. \tag{2.53}$$

Fuel-air equivalence ratio, mixture fraction

For complex reacting flows additional scalars can be necessary to describe the kinetics well. Fuel-air equivalence ratio is commonly used:

$$\phi = \frac{m_F/m_O}{(m_F/m_O)_{st}}\,, \tag{2.54}$$

or alternatively the mixture fraction:

$$Z = \frac{sY_F - Y_O + Y_{O,0}}{sY_{F,0} + Y_{O,0}}, \tag{2.55}$$

with

$$s = \frac{W_O \nu_O}{W_F \nu_F}. \tag{2.56}$$

where ν_F and ν_O are the fuel and oxygen stoichiometric coefficients.

2.6 Two-phase combustion

Two-phase combustion involves the burning of liquid fuel in the form of droplets dispersed in a gas, typically air. This complex process can be studied in stages, starting from the simplest scenario of a single burning droplet, progressing through mist combustion where multiple droplets interact, and culminating in spray combustion used in practical applications like engines and burners.

2.6.1 Single droplet combustion

The droplet is subjected to external heating, resulting in the evaporation of the liquid and the formation of a vapor phase around the droplet. This phase transition from liquid to vapor generates a Stefan's flow, as the vapor occupies a greater volume than the liquid. Consequently, the vapor disperses from the droplet, while the surrounding oxidizer diffuses towards it. Ignition occurs when the vapor-cloud mixture achieves a flammable concentration and reaches a sufficient temperature. The resulting diffusion flame envelops the droplet, and the heat produced by the flame sustains the ongoing evaporation process.

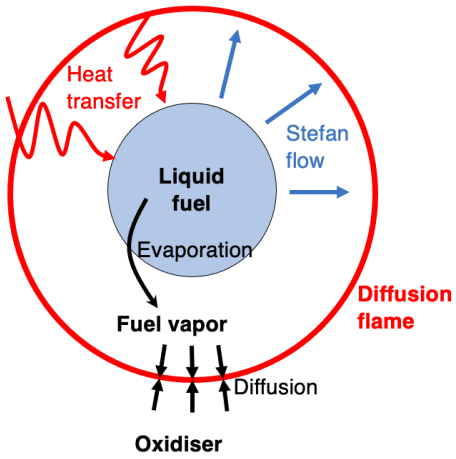


Figure 2.3: Isolated droplet evaporation.

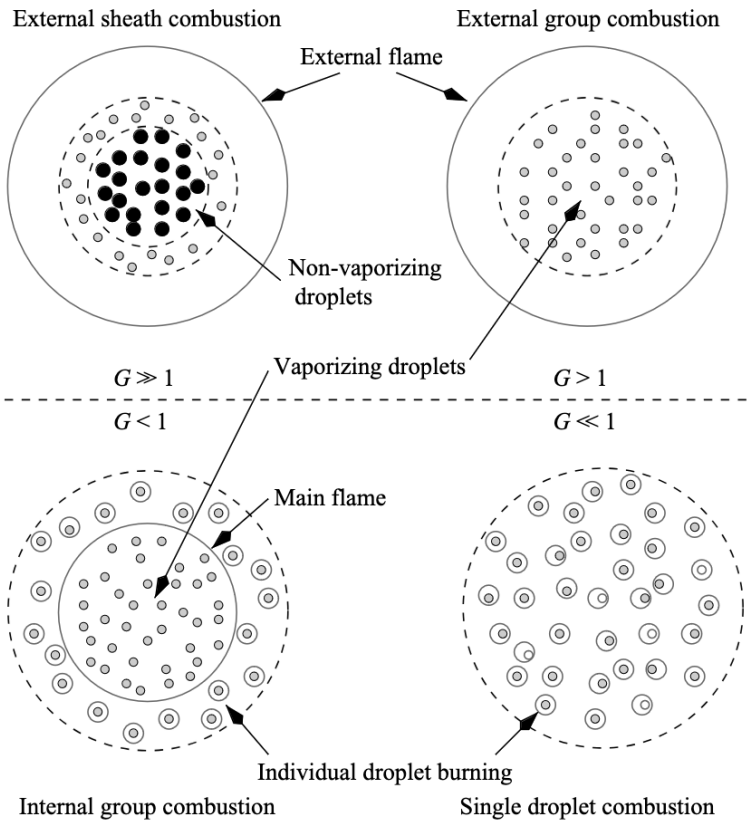


Figure 2.4: Combustion modes of a droplet cloud, reproduction from [25, 26].

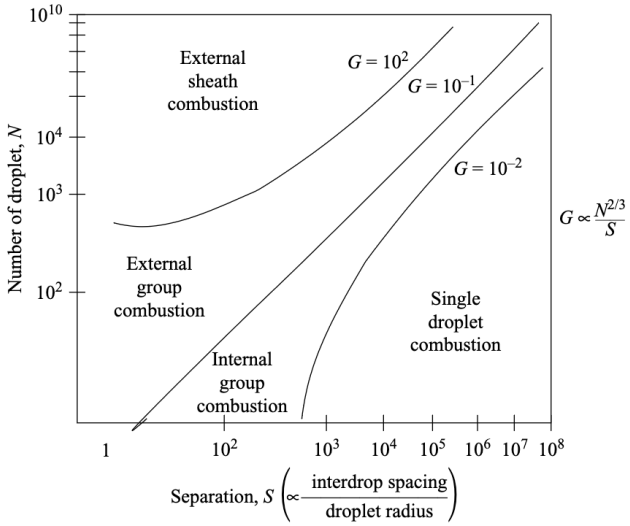


Figure 2.5: Group combustion diagram, reproduction from [27, 26].

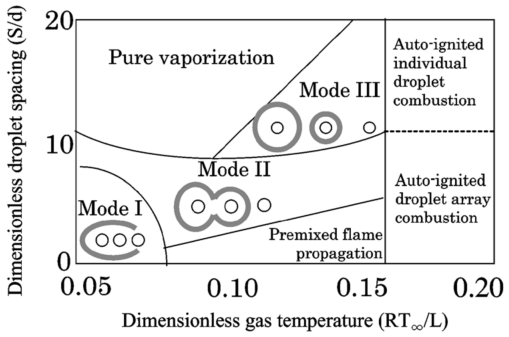


Figure 2.6: Interdroplet flame propagation mode map, reproduction from [28].

2.6.2 Droplet mist combustion

Building on the principles of single droplet combustion, mist combustion involves the burning of a collection of fine liquid droplets suspended in a gas. Inter-droplet flame propagation within the mist is now to be considered. The flame regime is primarily determined by the density of the mist. Fig. 2.4 and Fig. 2.6 illustrate the transition between two distinct combustion modes: internal group combustion, where the flame propagates through the mist, and external group combustion, where the flame surrounds the droplets but does not penetrate the mist. A more subtle distinction of the droplet combustion regimes is introduced in the work of [28] who proposed a map of possible flame regimes depending droplet spacing and surrounding gas temperature, shown in Fig. 2.6. Modes 1 to 3 represent different regimes of diffusion flame behavior. In Mode 3, droplets combust sequentially, each surrounded by its own individual flame. Flame propagation in this mode is only sustained at initially high gas temperatures. In Mode 2, individual diffusion flames are present initially but eventually merge into a single continuous flame. In Mode 1, droplets are closely packed, leading to the formation of a continuous flame that envelops multiple droplets, a phenomenon referred to as group or cluster combustion. At high temperatures for packed droplets, a premixed flame may be observed, as a continuous vaporized fuel supply allows for sustained flame propagation, irrespective of the presence of the unvaporized droplets.

2.6.3 Spray combustion

In spray combustion, liquid fuel is continuously atomized into fine droplets using a nozzle. These injected droplets possess velocity and momentum components that influence flame behavior. A simple spray combustion scenario is illustrated in Fig. 2.7. The droplets are injected at a velocity exceeding the flame propagation speed within the droplet mist. Consequently, the flame cannot exist near the injector. Instead, it stabilizes in regions of lower velocity where it avoids blowoff. Two such regions are possible: far from the injector, where the droplets and entrained air flow have decelerated to match the flame propagation speed, or in the outer region of the jet.

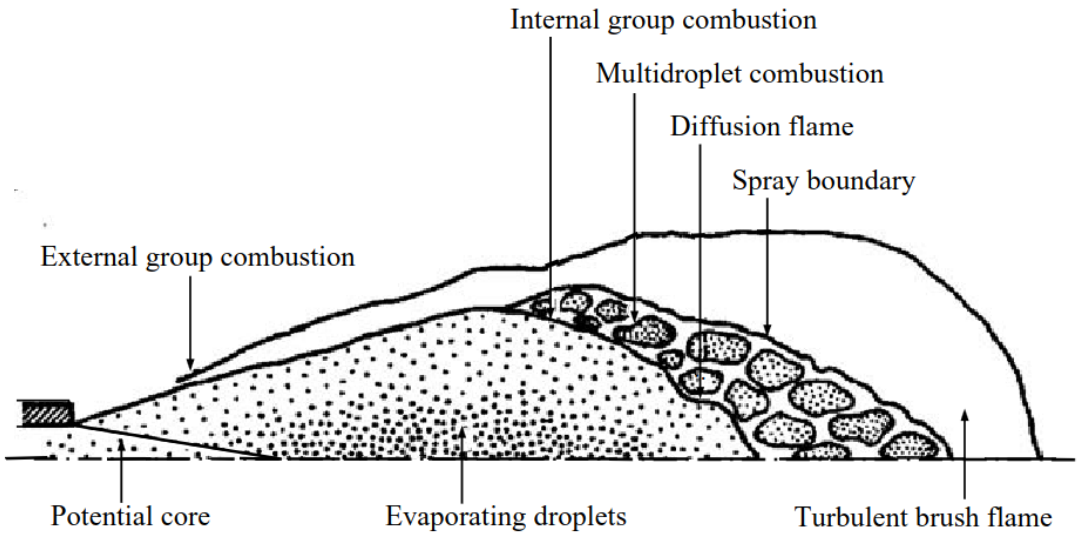


Figure 2.7: Schematic showing the group combustion concept of a spray jet, reproduction from [29, 30].

In this configuration, the oxidizer is solely entrained by the spray. However, in most modern spray burners, an air co-flow is injected around the spray to enhance combustion efficiency and stability. A wide variety of flame behaviors and topologies can be achieved depending on the design of the spray burner.

2.7 Reference set-up: Coria Rouen Spray Burner

The CORIA Rouen Spray Burner (CRSB) is used as the reference case throughout this thesis. This burner has been chosen due to the availability of a large amount of experimental data [31, 32, 33, 34] and many past Large-Eddy Simulations [35, 36, 37] of this configuration. This burner has also been used as a reference case in the Workshop on Turbulent Combustion on Sprays (TCS).

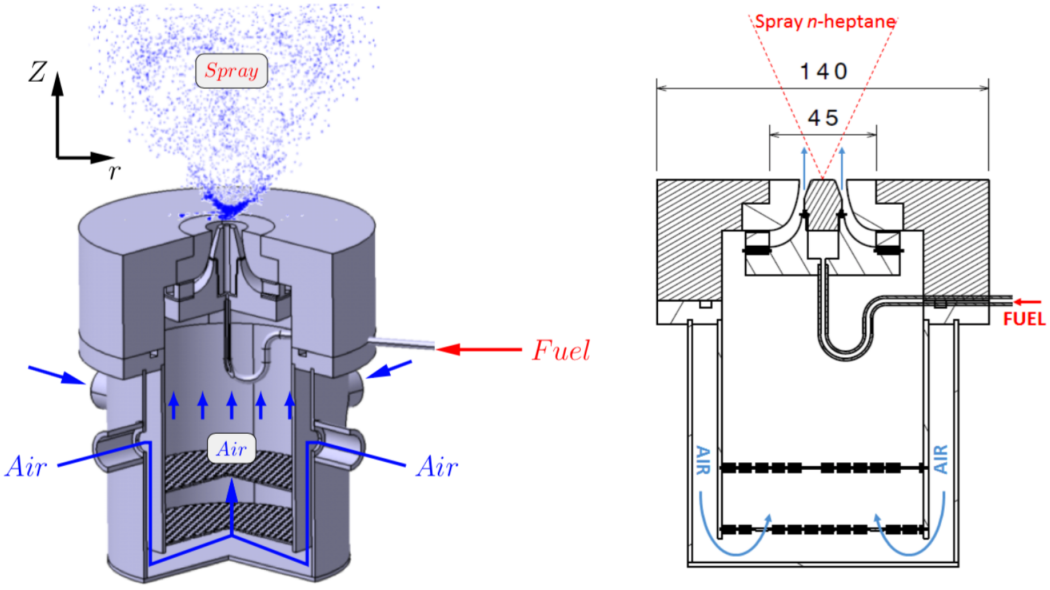


Figure 2.8: CRSB experimental set-up from [31].

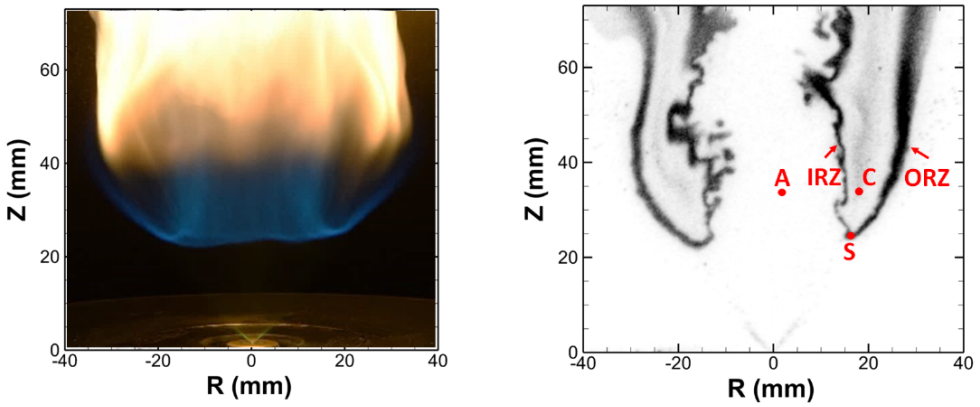


Figure 2.9: (a) Photo of the spray jet flame. (b) OH-PLIF and different zones. Reproduction from [31].

The CRSB is an open atmospheric spray burner. The fuel injection system is composed of a DANFOSS fuel injector. The liquid dispersed phase is composed of n-heptane (C7H16) and injected within an inner half-angle of 40° and a mass flow rate of 0.28 g.s^{-1}. An external annular, non-swirling air co-flow, with inner and outer diameters of 10 and 20 mm is injected with a mass flow rate of 6 g.s^{-1}. Both injections are done at the ambient temperature of 298±2 K. The associated experimental setup is shown in Fig. 2.8, the flame can be seen in Fig. 2.9.

Experimental diagnostics on the dispersed phase include Phase Doppler Anemometry (PDA) providing correlated velocity and size of droplets as well as Global Rainbow Refractometry Technique (GRT) to measure droplet temperature. The reactive flow diagnostics include simultaneous high-speed PIV and high-speed OH-PLIF, and NO-PLIF.

Shortly after injection, the droplets interact with the air stream. Small droplets ($< 20\mu$m) are completely entrained by the flow and act as tracers. They therefore do not maintain their initial injection angle. Large droplets ($> 60\mu$m) are fully inertial and maintain their initial injection angle. Medium droplets are deflected by the co-flow but do not follow it. This results in a radial separation of the droplets according to their size. This is shown in Fig 2.10.

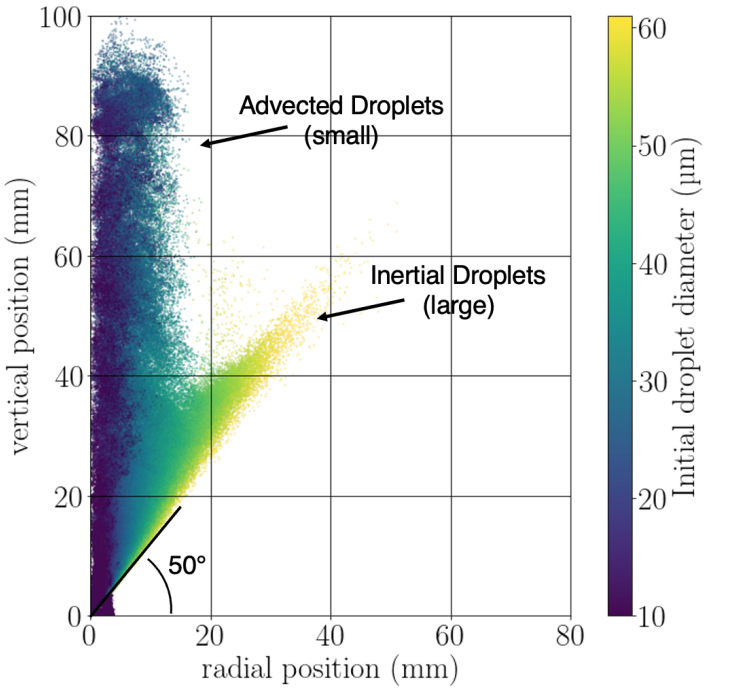


Figure 2.10: CRSB droplet separation by diameter.

The various flame regions are depicted in Fig. 2.11. The spray flame is established 25mm downstream of the injection point, which is referred to as the flame base. Two flame branches emerge from this point, namely the Inner Reaction Zone (IRZ) and the Outer Reaction Zone (ORZ). These flame branches can also be observed in Fig. 2.12 based on the heat release rate. A hot, non-reactive zone (HNRZ) is present between the two flame fronts. Due to its circular symmetry, the IRZ encloses a cold, non-reactive zone (CNRZ).

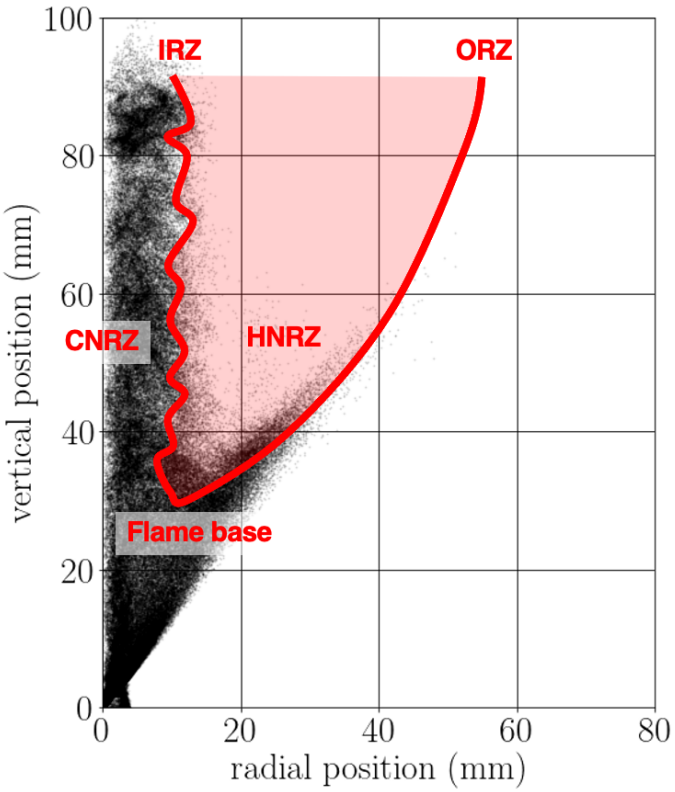


Figure 2.11: CRSB main flame regimes.

The heat of the HNRZ is sustained by the two flame fronts. The spray droplets enter this zone and rapidly evaporate. The loss of heat from the gaseous phase can be observed in the temperature field depicted in Fig. 2.13. This region is fully depleted of the oxidizer O2 and thus non reactive. N-heptane accumulates within this region and diffuses towards the surrounding IRZ and ORZ. Due to the high temperature, N-heptane progressively breaks down into smaller hydrocarbons. Fig. 2.14 shows the fuel-air equivalence ratio of the mixture. This ratio is computed based on the presence of oxygen and carbon atoms. Therefore, all hydrocarbons are accounted for, but burnt gases are also included.

The IRZ flame front is fuelled by three contributions. Small droplets transported by the air coflow slowly evaporate and create a lean premixed gas. This mixture remains under the flammability limit, measured as a fuel-air equivalence ratio of $\phi = 0.256$ for cold atmospheric premixed N-heptane/air flames in [38]. The fuel-air equivalence ratio of the cold gases in the CNRZ is given by Fig. 2.15 and rarely exceeds $\phi = 0.15$. The flame front is also sustained by small droplets directly evaporating within it. Finally, backward diffusion with turbulent mixing occurs from the HNRZ. The topology of this flame front is highly dependent on the turbulent air jet co-flow with the flame existing on its border, as shown in Fig. 2.16. The flame cannot move into the jet as its velocity exceeds flame speed by several orders of magnitude. Furthermore, it cannot move apart from the jet as it is dependent on turbulent mixing. The shear of the jet results in highly strained flames, which can lead to local flame extinctions.

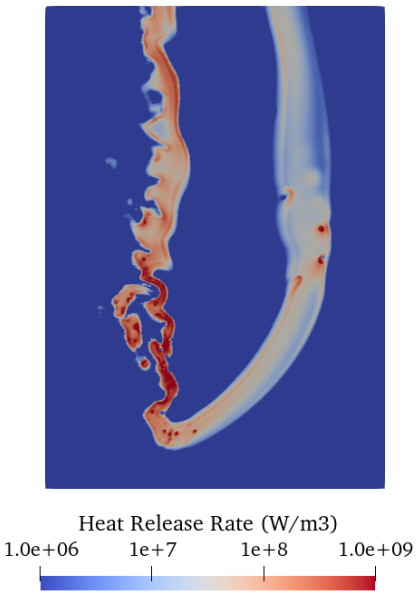


Figure 2.12: Heat Release Rate field from simulation.

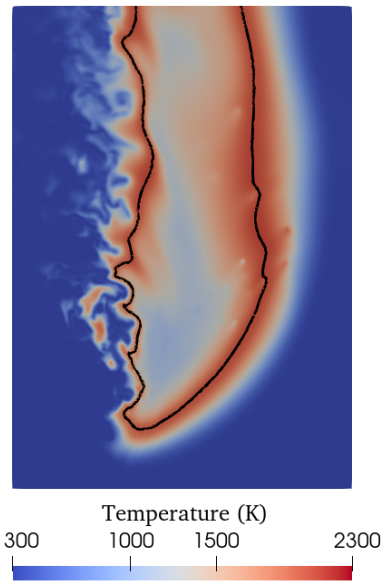


Figure 2.13: Temperature field from simulation. Black isoline correspond to 99% of O_2 depletion.

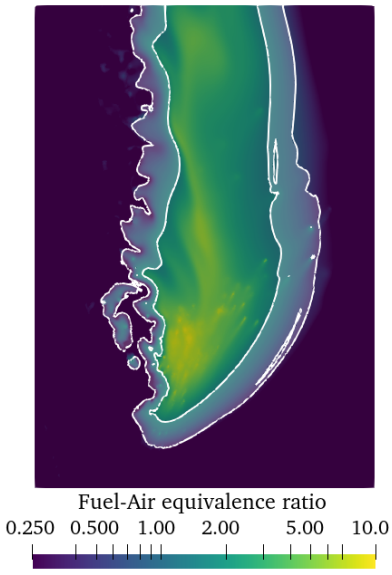


Figure 2.14: Fuel air equivalence ratio in hot gases bases on C and O atomic count (logscale). Reactive region is highlighted.

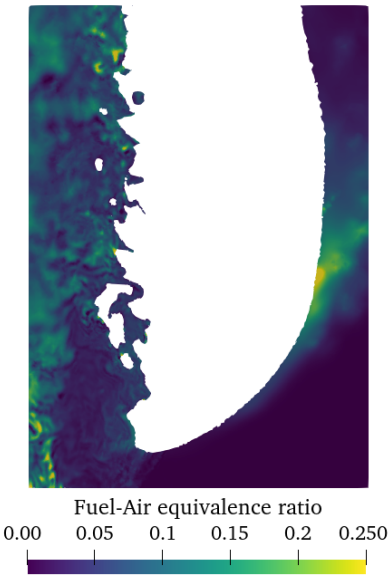


Figure 2.15: Fuel air equivalence ratio in cold gases bases on C_7H_{16} and O_2.

The ORZ is a quasi-steady diffusion flame situated between the HNRZ and the atmosphere. The ORZ emerges from the flame base with an injection angle of 40° that is maintained by the largest droplet. Some slow fluctuations of this front are observed due to the outer atmospheric turbulence and the heterogeneity of the rich mixture in the HNRZ.

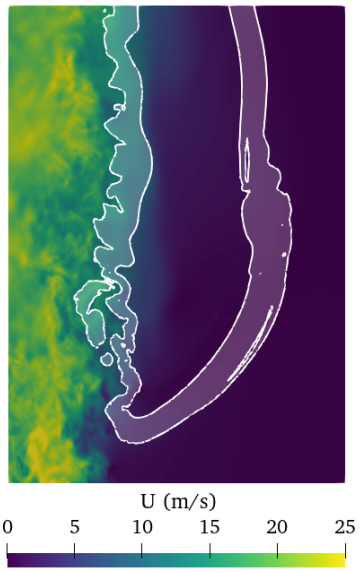


Figure 2.16: Establishment of the interior reaction zone at the border of the air co-flow jet.

CHAPTER 3

Large-Eddy Simulation of two-phase Euler-Lagrange reactive flows

The equations for modeling two-phase reactive flows are introduced, focusing on Large Eddy Simulation (LES) with an Euler-Lagrange approach. The simulation framework YALES2 is discussed and applied to the CRSB to assess its current performance and identify areas for optimization.

Contents

3.1 Continuous gaseous phase

3.1.1 Conservation equations

- **Mass conservation**

$$\frac{\partial \rho}{\partial t} + \frac{\partial \rho u_i}{\partial x_i} = 0\,. \tag{3.1}$$

- **Momentum conservation**, when neglecting volumic forces:

$$\frac{\partial \rho u_j}{\partial t} + \frac{\partial \rho u_i u_j}{\partial x_i} = -\frac{\partial P}{\partial x_j} + \frac{\partial \tau_{ij}}{\partial x_i}\,, \tag{3.2}$$

where P is the static pressure and τ_{ij} is the viscous stress tensor:

$$\tau_{ij} = \mu\left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i}\right) - \frac{2}{3}\mu\frac{\partial u_k}{\partial x_k}\delta_{i,j}\,, \tag{3.3}$$

with μ the dynamic viscosity and $\delta_{i,j}$ the Kronecker tensor.

- **Energy conservation**, written based on sensible enthalpy h_s:

$$\frac{\partial \rho h_s}{\partial t} + \frac{\partial \rho u_i h_s}{\partial x_i} + \frac{\partial q_i}{\partial x_i} = \dot{\omega}_T + \dot{Q}\,, \tag{3.4}$$

with q_i the sensible enthalpy flux due to the Fourrier law and species diffusion:

$$q_i = \lambda\frac{\partial T}{\partial x_i} - \frac{\partial}{\partial x_i}\left(\rho\sum_{k=1}^{N_{sp}} h_{s,k}Y_k V_{k,i}\right)\,, \tag{3.5}$$

with $\dot{\omega}_T$ the kinetic heat release rate:

$$\dot{\omega}_T = -\sum_{k=1}^{N_{sp}} \Delta h^0_{f,k}\dot{\omega}_k\,, \tag{3.6}$$

where $\dot{\omega}_k$ is the mass source term of a species. This term will be further developed in Section 2.3. $\dot{Q}$ represents any additional heat source or sink. Cases considered in this report are purely adiabatic, therefore $\dot{Q}$ is always zero.

- **Species conservation**:

$$\frac{\partial \rho Y_k}{\partial t} + \frac{\partial \rho\left(u_i + V_{k,i}\right)Y_k}{\partial x_i} = \dot{\omega}_k \quad \text{for k=1,}N_{sp}\ , \tag{3.7}$$

where V_k is the diffusion velocity of species k and $\dot{\omega}_k$ is the reaction rate of species k. Due to mass conservation:

$$\sum_{k=1}^{N_{sp}} V_{k,i}Y_k = 0\,. \tag{3.8}$$

Based on the Curtiss and Hirschfelder simplification introduced in Eq. 2.27, species conser-

vation can be rewritten as:

$$\frac{\partial \rho Y_k}{\partial t} + \frac{\partial \rho u_i Y_k}{\partial x_i} = \frac{\partial}{\partial x_i}\left(\rho D_k \frac{W_k}{W}\frac{\partial X_k}{\partial x_i}\right) + \dot{\omega}_k \,. \tag{3.9}$$

The major drawback of this approach is that mass conservation specified by Eq. 3.8 is no longer verified. Therefore a correction velocity V^c is introduced:

$$\frac{\partial \rho Y_k}{\partial t} + \frac{\partial \rho \left(u_i + V_i^c\right) Y_k}{\partial x_i} = \frac{\partial}{\partial x_i}\left(\rho D_k \frac{W_k}{W}\frac{\partial X_k}{\partial x_i}\right) + \dot{\omega}_k \,, \tag{3.10}$$

so that:

$$V_i^c = \sum_{k=1}^{N_{sp}} D_k \frac{W_k}{W}\frac{\partial X_k}{\partial x_i} \,. \tag{3.11}$$

3.1.2 Large-Eddy Simulation

Modeling and scale-resolving approaches (RANS, LES, DNS)

Current numerical methods allow the study of both laminar and turbulent regimes. These methods are referred to as Computational Fluid Dynamics or CFD. Their main advantage is that they allow access to quantities in the flow, while experimental measurements are often limited by the employed techniques. There are three main numerical approaches to simulating turbulent flows, namely: DNS, RANS and LES.

- **Direct Numerical Simulation (DNS):** It consists in solving the conservation equations for all spatial and temporal scales, from large-scale vortical structures down to the Kolmogorov scale. In this approach, no modeling of the smallest structures is used; hence, the solving requires a very fine grid. Thus, this approach is limited to the study of flows ranging from low to moderate Reynolds, as the CPU cost for higher Reynolds numbers is prohibitive today. Physical models have been developed to overcome this limitation and consider the effect of the smallest eddies.

- **Reynolds-Averaged Navier-Stokes (RANS):** It consists of the solving of the steady-state conservation equations. This approach requires the use of a lighter grid as all scales are modeled. For this reason, RANS has been extensively used in industry for the past decades. However, the accuracy of the modeling of velocity fluctuations is limited, making this a drawback of the method.

- **Large-Eddy Simulation (LES):** It consists of the explicit resolution of large structures and the modeling of the smallest ones. These models are based on quantities that are explicitly solved on the grid. The scale separation is carried out by defining a cut-off frequency dependent on a spatial filter. Note that closure models assume that this latter is located in the inertial range, where no dissipation occurs, so that dissipation comes only from the modeled as smaller scales are expected to behave in a universal way. Finally, LES enables the unsteady study of the overall dynamics. It can be seen as a trade-off between DNS and RANS, allowing the study of turbulent flows with a moderate grid at an affordable cost.

The LES filtered equations

As stated in the previous section, LES is based on the explicit resolution of the large structures and the modeling of the smallest ones. This separation is achieved by filtering the conservation equations with a low-pass filter. In the physical space, a convolution product is used. Let $\phi(x,t)$ be a scalar. The characteristic filtering operator reads:

$$\overline{\phi}(x,t) = \int_D \phi\left(x',t\right) G\left(x,x',\overline{\Delta}\right) \mathbf{dx}' \,, \tag{3.12}$$

where $\overline{\phi}(x,t)$ is the filtered scalar and G is the filtering kernel associated to the filter size $\overline{\Delta}$. The filtering operator must be normalized so that:

$$\int_D G\left(x,\overline{\Delta}\right) \mathbf{dx} = 1 \,. \tag{3.13}$$

Moreover, the operator has to ensure both spatial and temporal commutativity, which is verified if $\overline{\Delta}$ is constant in both space and time, namely:

$$\frac{\partial\overline{\phi}}{\partial t} = \overline{\frac{\partial\phi}{\partial t}} \quad \text{and} \quad \frac{\partial\overline{\phi}}{\partial x_i} = \overline{\frac{\partial\phi}{\partial x_i}} \,. \tag{3.14}$$

Consequently, this decomposition allows to split any scalar field $\phi(x,t)$ into two parts:

$$\phi\left(x,t\right) = \overline{\phi}\left(x,t\right) + \phi'\left(x,t\right) \,, \tag{3.15}$$

here $\overline{\phi}\left(x,t\right)$ is referred to as an explicitly resolved quantity and $\phi'\left(x,t\right)$ is referred to as a sub-grid scale (SGS) quantity. It is important to highlight that the filtering operator does not respect the distributivity or idempotence rules, namely:

$$\overline{\phi_1\phi_2} \neq \overline{\phi_1}\,\overline{\phi_2} \quad \text{and} \quad \overline{\overline{\phi}} \neq \overline{\phi} \,. \tag{3.16}$$

In cases where important density variations are expected, a mass-weighted average, also known as the Favre averaging, is performed, leading to:

$$\widetilde{\phi} = \left(\frac{\overline{\rho\phi}}{\overline{\rho}}\right) \,. \tag{3.17}$$

Applying this operator to the conservation equations leads to the filtered balance equations solved in LES.

- **Filtered mass conservation**:

$$\frac{\partial\overline{\rho}}{\partial t} + \frac{\partial\overline{\rho}\widetilde{u}_i}{\partial x_i} = 0 \,. \tag{3.18}$$

- **Filtered momentum conservation**:

$$\frac{\partial\overline{\rho}\widetilde{u}_j}{\partial t} + \frac{\partial\overline{\rho}\widetilde{u_i u_j}}{\partial x_i} = -\frac{\partial\overline{P}}{\partial x_j} + \frac{\partial\overline{\tau_{ij}}}{\partial x_i} \,. \tag{3.19}$$

In practice only $\widetilde{u}_i\widetilde{u}_j$ can be evaluated and not $\widetilde{u_iu_j}$ thus equation is rewritten as:

$$\frac{\partial\overline{\rho}\widetilde{u}_j}{\partial t}+\frac{\partial\overline{\rho}\widetilde{u}_i\widetilde{u}_j}{\partial x_i}=-\frac{\partial\overline{P}}{\partial x_j}+\frac{\partial\overline{\tau_{ij}}}{\partial x_i}-\frac{\partial}{\partial x_i}\left[\overline{\rho}\left(\widetilde{u_iu_j}-\widetilde{u}_i\widetilde{u}_j\right)\right]. \tag{3.20}$$

This form will be given directly for the following equations.

- **Filtered energy conservation:**

$$\frac{\partial\overline{\rho}h_s}{\partial t}+\frac{\partial\overline{\rho}\widetilde{u}_i\widetilde{h}_s}{\partial x_i}+\frac{\partial\overline{q}_i}{\partial x_i}=\overline{\dot{\omega}}_T-\frac{\partial}{\partial x_i}\left[\overline{\rho}\left(\widetilde{u}_i\widetilde{h}_s-\widetilde{u_ih_s}\right)\right]. \tag{3.21}$$

- **Filtered species conservation:**

$$\frac{\partial\overline{\rho}\widetilde{Y_k}}{\partial t}+\frac{\partial\overline{\rho}\widetilde{u}_i\widetilde{Y}_k}{\partial x_i}=\frac{\partial\overline{V_{k,i}Y_k}}{\partial x_i}+\overline{\dot{\omega}}_k-\frac{\partial}{\partial x_i}\left[\overline{\rho}\left(\widetilde{u}_i\widetilde{Y}_k-\widetilde{u_iY_k}\right)\right]. \tag{3.22}$$

In these equations the following terms are unclosed and need to be modeled:

- Unresolved Reynolds stresses $(\widetilde{u_iu_j}-\widetilde{u}_i\widetilde{u}_j)$.
- Unresolved enthalpy advection fluxes $\left(\widetilde{u_ih_s}-\widetilde{u}_i\widetilde{h}_s\right)$.
- Unresolved species advection fluxes $\left(\widetilde{u_iY_k}-\widetilde{u}_i\widetilde{Y}_k\right)$.
- Filtered enthalpy laminar diffusion fluxes $\overline{\lambda\frac{\partial T}{\partial x_i}}$.
- Filtered species laminar diffusion fluxes $\overline{V_{k,i}Y_k}$.
- Filtered chemical reaction rate $\overline{\dot{\omega}}_k$.

3.1.3 Sub-grid scale models

Unclosed terms are modeled based on sub-grid scale models (SGS). Two main modeling approaches can be distinguished, structural and functional models [39]. Structural SGS models aim at reconstructing the effects of the unresolved scales based on the resolved scales assuming a certain turbulence structure. These models use information from the larger, resolved scales to infer the behavior of the smaller, unresolved scales. The underlying assumption is that the smaller scales can be represented by scaling down the structures of the larger scales. While structural model allow for an accurate reconstruction of the SGS terms by relying on the LES equations, they suffer from small error accumulation over time, which can lead to erroneous integral scale behavior. Functional SGS models focus on the overall effects of the unresolved scales on the resolved scales rather than attempting to reconstruct their detailed structures. These models are more phenomenological, emphasizing the impact of the sub-grid scales.

3.1.4 Turbulence modeling

Turbulence models aim at closing the unresolved Reynolds stresses. This term can be decomposed as:

$$\begin{aligned}\tau_{ij}^R =& \widetilde{u_i u_j} - \widetilde{u}_i \widetilde{u}_j \\ =& \underbrace{\left(\widetilde{\widetilde{u}_i \widetilde{u}_j} - \widetilde{u}_i \widetilde{u}_j\right)}_{L_{ij}} + \underbrace{\left(\widetilde{\widetilde{u}_i u'_j} + \widetilde{u'_i \widetilde{u}_j}\right)}_{C_{ij}} + \underbrace{\left(\widetilde{u'_i u'_j}\right)}_{R_{ij}} .\end{aligned} \tag{3.23}$$

The first term L_{ij} , also known as Leonard's tensor, comes from the non-idempotency of the filtering operator presented in Eq. 3.16. In most cases it can be neglected. The second term C_{ij} takes into account the coupling between the largest and smallest eddies. According to Kolmogorov hypothesis, this operator can be neglected if the filtering takes place in the inertial range. Eventually, the last term R_{ij} , accounts for the sub-grid scale stresses to be modeled. It can be further decomposed into a deviatoric and isotropic part:

$$R_{ij} = R_{ij}^D + R_{ij}^I = \tau_{ij}^{SGS} + R_{ij}^I = \tau_{ij}^{SGS} + \frac{1}{3}\tau_{kk}\delta_{ij} . \tag{3.24}$$

This isotropic part is commonly absorbed into the pressure term and consequently requires no modeling. The deviatoric part can be expressed as a turbulent viscosity ν_t based on Boussinesq hypothesis, for incompressible flows:

$$\tau_{ij}^{SGS} = -\nu_t \left(\frac{\partial \widetilde{u}_i}{\partial x_j} + \frac{\partial \widetilde{u}_j}{\partial x_i}\right) = -2\nu_t \widetilde{S_{ij}} , \tag{3.25}$$

for variable density flows:

$$\tau_{ij}^{SGS} = -\nu_t \left(\frac{\partial \widetilde{u}_i}{\partial x_j} + \frac{\partial \widetilde{u}_j}{\partial x_i} - \frac{2}{3}\delta_{ij}\frac{\partial \widetilde{u}_k}{\partial x_k}\right) = -2\nu_t \left(\widetilde{S_{ij}} - \frac{\delta_{ij}}{3}\widetilde{S}_{kk}\right) . \tag{3.26}$$

The closure now consists in evaluating ν_t.

- **The Smagorinsky model** [40] expresses ν_t based on the large scales:

$$\nu_t = (C_S\Delta)^2 \left|\widetilde{S}\right| = (C_S\Delta)^2 \sqrt{2\widetilde{S}_{ij}\widetilde{S}_{ij}} , \tag{3.27}$$

  where Δ is the local filter length and C_S the Smagorinsky constant. It ranges from 0.1 to 0.2 and depends on the flow configuration. This model is known for its excessive damping, as well as its failure to predict laminar to turbulent transition or its inaccurate handling of near-wall regions[41].

- **The Dynamic Smagorinsky model** [42, 43] propose a dynamic and local procedure to evaluate the Smagorinsky constant. A second filtering operator $\widehat{(\cdot)}$ is introduced and the associated unresolved Reynolds stress, T_{ij} is introduced:

$$L_{ij} = T_{ij} - \widehat{\tau}_{ij} = \widehat{\widetilde{u}_i}\widehat{\widetilde{u}_j} - \widehat{\widetilde{u_i}\widetilde{u}_j} , \tag{3.28}$$

where:

$$\begin{aligned} \tau_{ij} &= \widetilde{u_i u_j} - \widetilde{u}_i \widetilde{u}_j \\ T_{ij} &= \widehat{\widetilde{u_i u_j}} - \widehat{\widetilde{u}}_i \widehat{\widetilde{u}}_j \,. \end{aligned} \tag{3.29}$$

Under the assumption of scale invariance, T_{ij} and τ_{ij} can be expressed based on the same Smagorinsky constant:

$$\begin{aligned} \tau_{ij} - \frac{1}{3}\tau_{kk}\delta_{ij} &= C_d \beta_{ij} \\ T_{ij} - \frac{1}{3}T_{kk}\delta_{ij} &= C_d \alpha_{ij} \,, \end{aligned} \tag{3.30}$$

where α_{ij} and β_{ij} represent the deviatoric part of the sub-grid stress tensor, that may be expressed as for the Smagorinsky model. Inserting Eq. 3.30 into Eq. 3.28 gives the relation:

$$L_{ij} - \frac{1}{3}L_{kk}\delta ij \equiv C_d \alpha_{ij} - C_d \widehat{\beta_{ij}} \,, \tag{3.31}$$

in which the following approximation has been performed:

$$C_d \widehat{\beta_{ij}} = \widehat{C_d \beta_{ij}} \,. \tag{3.32}$$

Solving Eq. 3.31 consist in finding C_d that minimize the residual:

$$E_{ij} = L_{ij} - \frac{1}{3}L_{kk}\delta_{ij} - C_d \alpha_{ij} + C_d \widehat{\beta_{ij}} \,. \tag{3.33}$$

The evaluated C_d can be negative inducing back-scattering [44] and leading to numerical instabilities. Those can be avoided by additional numerical treatments. The dynamic Smagorinsky provides more accurate results than the Smagorinsky model for a wide range of flow but is also more computationally expensive.

- Other more elaborated model have been developed. The **Wall-Adapting Local Eddy-viscosity (WALE)** [45] and **SIGMA** [46] model provide improved near-wall behavior of turbulence. These model are not further described as they exceed the scope of this thesis.

3.1.5 Turbulent combustion modeling

Direct closure

The filtered reaction rate $\overline{\dot{\omega}_k}\,(\Phi)$ is unclosed as it relies on the unfiltered species and energy.
The simplest approach, also referred to as no model approach, is to assume perfect mixing at the sub-grid scale, thus reaction rates can be expressed as:

$$\overline{\dot{\omega}_k\,(\Phi)} = \dot{\omega}_k\left(\widetilde{\Phi}\right) \,. \tag{3.34}$$

This assumes that the turbulent timescale is smaller than the chemical timescales, which is rarely the case in combustion applications. Flames that are thin compared to the mesh size should be avoided as the extra mixing from the implicit LES filter can strongly affect the reacting state and

deteriorate flame speed. This formulation may be used on fine meshes close to flame DNS, ensuring a sufficient number of points to resolve the flame and turbulence.

More advanced closures relie on scale similarity assumptions, acting as soft deconvolution. The Scale Similarity Resolved Reaction Rate Model (SSRRRM) is a structural model introduced by [47] and further validated by [48, 49]. The source term is separated into a resolved and sub-grid scale part:

$$\text{SSRRRM:} \quad \overline{\dot{\omega}_k\left(\Phi\right)} = \underbrace{\dot{\omega}_k\left(\widetilde{\Phi}\right)}_{\dot{\omega}_{\text{k,Resolved}}} + \underbrace{\overline{\dot{\omega}_k\left(\Phi\right)} - \dot{\omega}_k\left(\widetilde{\Phi}\right)}_{\dot{\omega}_{\text{k,SGS1}}} . \tag{3.35}$$

$$\text{SSFRRM:} \quad \overline{\dot{\omega}_k\left(\Phi\right)} = \underbrace{\overline{\dot{\omega}_k\left(\widetilde{\Phi}\right)}}_{\dot{\omega}_{\text{k,Filtered}}} + \underbrace{\overline{\dot{\omega}_k\left(\Phi\right)} - \overline{\dot{\omega}_k\left(\widetilde{\Phi}\right)}}_{\dot{\omega}_{\text{k,SGS2}}} . \tag{3.36}$$

To estimate the sub-grid scale term, a test filter is introduced $\widehat{(\cdot)}$. In the original approach the second filtering step has the same filter size as the implicit filtering. However, the use of a different test filter width is a viable option, sometimes referred to as model C. Overall the test filter should remain as close as possible to the LES filter for optimal results. The test filter is noted $\widehat{\cdot}$ and leads to:

$$\widehat{\overline{\dot{\omega}_k\left(\Phi\right)}} = \dot{\omega}_k\left(\widehat{\widetilde{\Phi}}\right) + \underbrace{\widehat{\dot{\omega}_k\left(\widetilde{\Phi}\right)} - \dot{\omega}_k\left(\widehat{\widetilde{\Phi}}\right)}_{\mathcal{L}_{\text{SGS1}}} + \widehat{\dot{\omega}_{\text{k,SGS1}}} , \tag{3.37}$$

$$\widehat{\overline{\dot{\omega}_k\left(\Phi\right)}} = \overline{\dot{\omega}_k\left(\widehat{\widetilde{\Phi}}\right)} + \underbrace{\overline{\widehat{\dot{\omega}_k\left(\widetilde{\Phi}\right)}} - \overline{\dot{\omega}_k\left(\widehat{\widetilde{\Phi}}\right)}}_{\mathcal{L}_{\text{SGS2}}} + \widehat{\dot{\omega}_{\text{k,SGS2}}} , \tag{3.38}$$

where $\mathcal{L}_{\text{SGS}}$ can be evaluated. Based on the scale-similarity hypothesis: $\dot{\omega}_{\text{SGS}} = K\mathcal{L}_{\text{SGS}}$ with K a constant, that can be evaluated based on the ratio between the filter width of the implicit and test filters. Testing against DNS data reveals that K also depends on the mesh size and the Damköhler. A dynamic approach to evaluate K has been introduced in [49] based on the Germano-Lilly as for the Dynamic Smagorinsky model, introducing an additional filter level. However, the dynamic models do not provide improved results compared to the static models.

Scale similarity is a structural model and therefore rarely used in LES simulations. This model is introduced here as Chapter 5, which is dedicated to adaptive mesh refinement, relies on this model.

Flame regime dependent turbulent combustion model

Functional combustion sub-grid scale (SGS) models depend on the flame regime they aim to reproduce. The most widely used model is the Thickened Flame Model (TFM, also known as TFLES or ATF) [50], which is applicable only to premixed flames. Premixed flame thickness usually ranges between 0.1 mm and 1 mm, and accurately resolving the flame requires several grid points within the flame front, presenting a significant challenge. The TFM approach addresses this by artificially thickening the flame, increasing its width so it can be captured by a coarser LES grid. This thickening is done in a manner to preserve the integral behavior of the flame. Flame speed and flame

thickness can be expressed based on thermal diffusivity and reaction rates:

$$s_L^0 \propto \sqrt{D_{th}\dot{\omega}} \quad \text{and} \quad \delta_L^0 \propto \frac{D_{th}}{s_L^0} \propto \sqrt{\frac{D_{th}}{\dot{\omega}}}\,, \tag{3.39}$$

a thickening factor $\mathcal{F}$ is applied:

$$D_{th}^{\mathcal{F}} = \mathcal{F} D_{th} \quad \text{and} \quad \dot{\omega}^{\mathcal{F}} = \frac{\dot{\omega}}{\mathcal{F}}\,, \tag{3.40}$$

as a result, flame width increases proportional to the flame factor but flame speed is preserved:

$$\begin{aligned} s_L^0 \propto \sqrt{D_{th}\dot{\omega}} \rightarrow s_L^{\mathcal{F}} \propto \sqrt{D_{th}\dot{\omega}}, \\ \delta_L^0 \propto \sqrt{\frac{D_{th}}{\dot{\omega}}} \rightarrow \delta_L^{\mathcal{F}} \propto \mathcal{F}\sqrt{\frac{D_{th}}{\dot{\omega}}}\,. \end{aligned} \tag{3.41}$$

Value of $\mathcal{F}$ is chosen based on the desired number of point within the front N_f, local mesh control volume size Δh and the theoretical flame width for the local state $\delta_{L,th}^0(\phi)$:

$$\mathcal{F} = N\frac{\Delta h}{\delta_{L,th}^0(\phi)}\,. \tag{3.42}$$

An adverse effect of the flame thickening, is the increased resistance of the flame to quenching effects [51, 52] leading to a loss of flame surface and thus turbulent flame speed. This is corrected by the use of an efficiency function $\mathcal{E}$:

$$D_{th}^{\mathcal{E}\mathcal{F}} = \mathcal{E}\mathcal{F} D_{th} \quad \text{and} \quad \dot{\omega}^{\mathcal{E}\mathcal{F}} = \frac{\mathcal{E}\dot{\omega}}{\mathcal{F}}\,, \tag{3.43}$$

Introduction of the efficiency function only shows on the flame speed:

$$\begin{aligned} s_L^0 \propto \sqrt{D_{th}\dot{\omega}} \rightarrow s_L^{\mathcal{E}\mathcal{F}} \propto \mathcal{E}\sqrt{D_{th}\dot{\omega}}, \\ \delta_L^0 \propto \sqrt{\frac{D_{th}}{\dot{\omega}}} \rightarrow \delta_L^{\mathcal{E}\mathcal{F}} \propto \mathcal{F}\sqrt{\frac{D_{th}}{\dot{\omega}}}\,. \end{aligned} \tag{3.44}$$

$\mathcal{E}$ can evaluated based on different approaches following the work of [53, 54].

Various geometrical approaches can represent the propagation of the flame front. The G-equation [55, 56] models the flame front as an infinitely thin surface, tracking its evolution by solving a level-set equation, which is particularly effective for capturing large-scale flame structures. The Coherent Flame Model (CFM) [57, 58] represents the flame front as a continuous, coherent sheet and accounts for the effects of turbulence on flame wrinkling and stretching, providing a more comprehensive approach to turbulent premixed combustion. Meanwhile, the Flame-Surface Density (FSD) model [59] focuses on quantifying the flame surface area per unit volume, directly linking it to the reaction rate, which makes it particularly useful for capturing the detailed interactions between turbulence and flame surface.

Probability density functions (PDFs) can be used to account for turbulence effects on the chemical

production rates. Two primary PDF approaches exist: the transported PDF (TPDF) method [60] and the assumed shape PDF methods [61, 62]. In the TPDF approach, the PDF evolves freely based on the underlying physics of the flow, whereas in assumed PDF methods, the PDF is predefined by a mathematical function and is fully characterized by a limited number of statistical moments. Although the transported PDF method generally offers greater accuracy, the assumed PDF approach is often preferred due to its significantly lower computational cost.

3.2 Dispersed liquid phase

3.2.1 Eulerian versus Lagrangian approach

The Eulerian approach represents the dispersed phase on the computational grid. Solving accurately each droplet with its interface dynamics can be done using front tracking or front capturing methods like Volume of Fluid (VOF) [63, 64] or Level-sets [65, 66]. These methods require a mesh size that is finer than the droplet to be represented. Such simulations are conducted to study atomization but are too expensive for reactive flows, due to the strong differences of space and time scales especially when the droplets evaporate.

Two-fluid methods allow a statistical representation of sprays based on space- and time-average properties, no longer requiring a fine mesh to capture interface dynamics [67, 68, 69]. These methods often assume a single droplet size with a single value of droplet velocity for each control volume and time step. However, such method becomes very tedious when the poly-dispersity or when multiple crossing velocities have to be taken into account especially in spray combustors that feature droplet sizes from 5 to 100 microns. Multi-fluid methods [70] allow the representation of different classes of diameter as individual fluids, but results in increased simulation cost. The averaging process which leads to the multi-fluid governing equations also renders some unsteady phenomena, such as a droplet traversing a flame front, more difficult to model.

The Lagrangian approach [71] consists in representing each droplet by a point particle assuming that all the scales associated to the droplet are finer that all the other scales of the flow. The particles interact with the Eulerian grid to exchange sources terms of momentum, mass and heat. This approach, also referred to as the point-force approach, requires the knowledge of the particle size distribution as no atomisation takes place or only with secondary break-up models. Depending on the exchanges of momentum, Lagrangian methods can be classified as:

- One-way coupling: The flow impacts the particles. It is suited for flows with a few small particles, the impact of the flow on the particles is significant, but the transfer of momentum from the particles to the flow is negligible.

- Two-way coupling: The flow impacts the particle and the other way around. Particles carry a significant amount of momentum while remaining within the range of a dispersed particle flow or evaporate.

- Four-way coupling: Similar to two-way coupling augmented by particles-particle collisions. It is required in dense flows like fluidized beds.

3.2.2 Particle kinematics

The motion of a particle is described by:

$$\frac{d\mathbf{x}_p}{dt} = \mathbf{u}_p \,, \tag{3.45}$$

$$\frac{dm_p\mathbf{u}_p}{dt} = \mathbf{F}_p \,, \tag{3.46}$$

with x_p the position of the particle, u_p its velocity, m_p its mass and F_p the forces acting on it. The following forces can be applied to a particle:

- Gravitational forces, this force is neglected in spray burners.
- Drag when moving throughout a fluid.
- Lift due to rotation of the particle.
- Collision forces.
- Electrostatic forces.

For spray burners, only the drag force and gravity is considered.

$$\mathbf{F}_p = \mathbf{F}_p^G + \mathbf{F}_p^D \,, \tag{3.47}$$

with Archimedes' principle:

$$\mathbf{F}_p^G = (\rho_p - \rho)\,\frac{\pi}{6} d_p^3 \mathbf{g} \,, \tag{3.48}$$

where ρ_p is the density of the dispersed phase, ρ the density of the continuous phase and d_p the diameter of the particle. The drag force of an isolated particle reads:

$$\mathbf{F}_p^D = m_p \frac{1}{\tau_p} \left(\mathbf{u}_p - \mathbf{u}_\infty\right) \,, \tag{3.49}$$

u_∞ is the undisturbed velocity of the continuous phase. τ_p is the drag relaxation time:

$$\tau_p = \frac{4}{3 C_D Re_p} \frac{\rho_p}{\rho} \frac{d_p^2}{\nu} \,, \tag{3.50}$$

Re_p is the particle Reynolds number:

$$Re_p = \frac{d_p |\mathbf{u}_p - \mathbf{u}_\infty|}{\nu} \,, \tag{3.51}$$

with ν the kinematic viscosity. C_D is the drag coefficient. It depends on the flow regime and several correlations have been proposed [72]. For flows with $1 < Re_p < 1000$ the Schiller and Naumann correlation is used [73], for high Reynolds number flows, $1000 < Re_p$, C_D isn't impacted by the

wake and thus constant [74].

$$\begin{cases} C_D = \frac{24}{Re_p}, & Re_p < 1 \\ C_D = \frac{24}{Re_p} + \frac{3.6}{Re_p^{0.313}}, & 1 < Re_p < 1000 \\ C_D = 0.44, & 1000 < Re_p < 200000 \end{cases} \tag{3.52}$$

3.2.3 Particle evaporation

In spray burners, the droplets heat up and evaporate quickly when reaching the flame front. Accurate evaporation speed is crucial for the flame dynamics. A reference thermodynamic state for the gaseous phase near the droplet is introduced as a weighting between the state at the droplet surface and the gaseous undisturbed state [75]:

$$T_{\text{ref}} = \frac{2}{3} T_{\text{surf}} + \frac{1}{3} T_{\infty} \,, \tag{3.53}$$

$$Y_{\text{ref},k} = \frac{2}{3} Y_{\text{surf},k} + \frac{1}{3} Y_{\infty,k} \,. \tag{3.54}$$

Temperature within the droplet is assumed constant thus $T_p = T_{surf}$. This is not true for the mass fractions as some species evaporate faster depending on their boiling temperature $T_{boil,k}$ and their latent heat of vaporisation $L_{v,k}$. The surface of the droplet is assumed in thermodynamic equilibrium with the gas thus Clausius-Clapeyron law applies:

$$P_{\text{surf},k} = P_{\text{ref}} \exp\left(\frac{W_k L_{v,k}}{R} \left(\frac{1}{T_{\text{boil},k}} - \frac{1}{T_p} \right) \right) , \tag{3.55}$$

$$X_{surf,k} = \frac{P_{surf,k}}{P} \,, \tag{3.56}$$

$$Y_{surf,k} = X_{surf,k} \frac{W_k}{W_{surf}} \,. \tag{3.57}$$

Based on the reference state, the following dimensionless numbers are defined:

- The **Prandtl number** is the ratio between the momentum diffusivity and the thermal diffusivity:

$$Pr = \frac{\nu_{ref}}{\alpha_{ref}} = \frac{\nu_{ref} \rho_{ref} C_{p,ref}}{\lambda_{ref}} \,, \tag{3.58}$$

  with α_{ref} the thermal diffusivity:

$$\alpha_{ref} = \frac{\lambda_{ref}}{\rho_{ref} C_{p,ref}} \,, \tag{3.59}$$

- The **Schmidt number** is the ratio between the momentum diffusivity and the mass diffusivity:

$$Sc_k = \frac{\nu_{ref}}{D_{k,ref}} \,. \tag{3.60}$$

- The **Sherwood / Nusselt Numbers** characterize the mass and heat transfer from a fluid interface. It is the ratio of the convective transport over the diffusive transport. For a single

sphere with Froessling's correlation [76]:

$$\begin{cases} Sh_k = 2 + 0.552 Re_p^{1/2} Sc_k^{1/3} , \\ Nu = 2 + 0.552 Re_p^{1/2} \, \mathrm{Pr}^{1/3} , \end{cases} \tag{3.61}$$

with the Ranz-Marshall correlation [77]:

$$\begin{cases} Sh_k = 2 + 0.6 Re_p^{1/2} Sc_k^{1/3} , \\ Nu = 2 + 0.6 Re_p^{1/2} \, \mathrm{Pr}^{1/3} . \end{cases} \tag{3.62}$$

Mass evaporation rate for an isolated particle is given by:

$$\dot{m} = \sum_{k=1}^{N} \dot{m}_k = \begin{cases} \pi d_p \rho_g D_{k,g} Sh_k^* \ln\left(1 + B_{m,k}\right) , \\ \pi d_p \frac{\lambda_g}{Cp_v} Nu^* \ln\left(1 + B_T\right) , \end{cases} \tag{3.63}$$

with B_T and $B_{m,i}$ the Spalding heat and mass transfer numbers [78]:

$$B_{m,k} = \frac{Y_{g,k}^s - Y_{g,k}^\infty}{\xi_k - Y_{g,k}^s} , \tag{3.64}$$

$$B_T = (1 + B_{m,k})^{\phi_k} - 1 , \tag{3.65}$$

with ξ_k the ratio of mass flux of species k over total mass flux:

$$\xi_k = \frac{\dot{m}_k}{\dot{m}} , \tag{3.66}$$

and:

$$\phi_k = \frac{Cp_v}{Cp_g} \frac{Sh_k^* P_r}{Nu^* Sc_k} , \tag{3.67}$$

where Cp_v and Cp_g are the heat capacity of the fuel vapor and of the gaseous mixture surrounding the droplet at reference state, respectively. Mass evaporation rate depends on the corrected Sherwood and Nusselt numbers. This correction accounts for the Stefan flow due to evaporation and has been introduced by Abramzon-Sirignano [79]:

$$Sh_k^* = 2 + \frac{Sh_k - 2}{F(B_{m,k})} , \tag{3.68}$$

$$Nu^* = 2 + \frac{Nu - 2}{F(B_T)} , \tag{3.69}$$

with:

$$F(B) = (1 + B)^{0.7} \frac{\ln\left(1 + B\right)}{B} . \tag{3.70}$$

It should be noted that this model doesn't account for condensation, neglects radiation heat transfer and neglects kinetic reactions in the liquid and surrounding gas. Due to the correction of the Nusselt and Sherwood numbers, numerical solution has to be obtained iteratively as shown in Fig. 3.1.

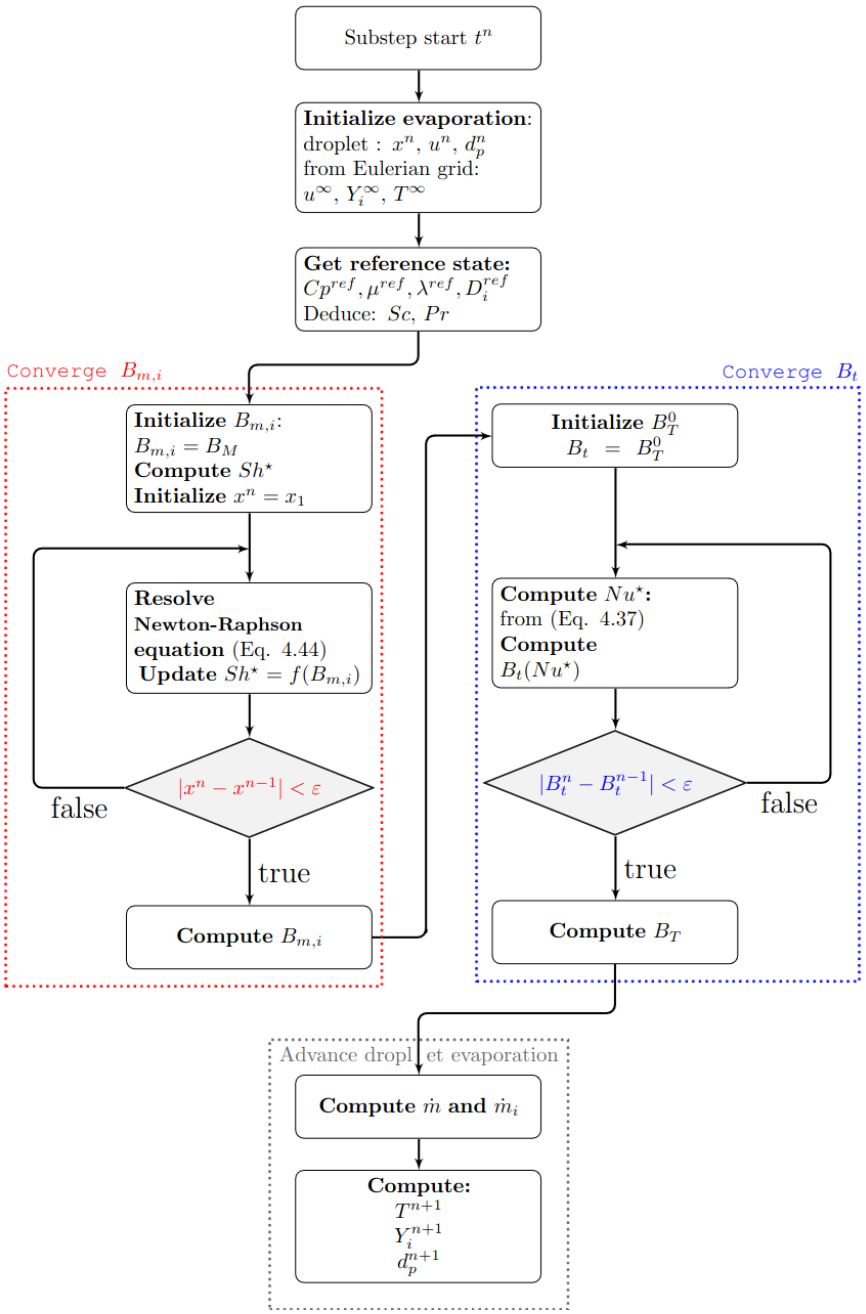


Figure 3.1: Description of the evaporation model. Reproduction from H. Larabi [80].

3.2.4 Lagrangian spray modeling

Spray burners generate a poly-dispersed spray, which means that the droplets produced have a wide range of diameters. To characterize such spray, various mean diameters can be defined using the following generic formula:

$$D_{pq} = \left(\frac{\sum_{i=1}^{N_p} d_i^p}{\sum_{i=1}^{N_p} d_i^q} \right)^{\frac{1}{p-q}} . \tag{3.71}$$

The Sauter Mean Diameter (SMD) is commonly used to express the ratio of droplet volume to surface area, which is relevant for heat and mass transfer in combustion applications:

$$SMD = D_{32} = \frac{\sum_{i=1}^{N_p} d_i^3}{\sum_{i=1}^{N_p} d_i^2} . \tag{3.72}$$

A probability density function of the particle distribution can be given to represent the variety of droplet size. The Rosin-Rammler distribution [81] reads:

$$F(D) = 1 - exp\left(- \left(\frac{D}{m} \right)^q \right) , \tag{3.73}$$

which can be reformulated by differentiation as:

$$f(D) = \frac{q}{m} \left(\frac{D}{m} \right)^{q-1} exp\left(- \left(\frac{D}{m} \right)^q \right) , \tag{3.74}$$

where q and m are the parameters. m is related to the peak location of the distribution and q to the spread. m can be related to the Sauter Mean Diameter by using the Gamma function:

$$D_{32} = \frac{m^3 \Gamma\left(\frac{3}{q} + 1 \right)}{m^2 \Gamma\left(\frac{2}{q} + 1 \right)} . \tag{3.75}$$

Other correlation like Nukiyama-Tanasawa [82] exists as well:

$$f(D) = K_D D^p exp\left(- \left(\frac{D}{m} \right)^n \right) . \tag{3.76}$$

However, it is more difficult to parametrize than the Rosin-Rammler because it relies on four parameters: K_D, m, n and p. Number of parameters can be reduced to only two following the work of [83].

3.3 Integration of governing equations in YALES2

3.3.1 YALES2 platform

YALES2 [84] is a massively parallel low-Mach number code for the DNS and LES of reacting two-phase flows in complex unstructured geometries. The code includes several solvers allowing for simulations in a wide range of scientific fields. This includes among others, combustion, wind

turbines and biomedical modeling. YALES2 is also capable of dynamic adaptive mesh refinement at runtime. The structure of YALES2 relies on a double-domain decomposition [84]. This property is used throughout the code for optimization purposes allowing better CPU cache usage. It is also used to design highly efficient algorithms. Lagrangian methods, implemented in particle-in-cell formalism, also benefit from the double-domain decomposition and particles are always bound to a given group of cells. If many particles are located on a same group of cells, several particle groups are used to optimize once again the CPU cache usage, adding a third level of decomposition. This section highlights the implementation choices made in YALES2.

3.3.2 Finite-volume method

The Finite-Volume Method (FVM) is a numerical technique used for solving partial differential equations. FVM discretizes the computational domain into a finite number of control volumes, and the governing equations are integrated over these volumes.

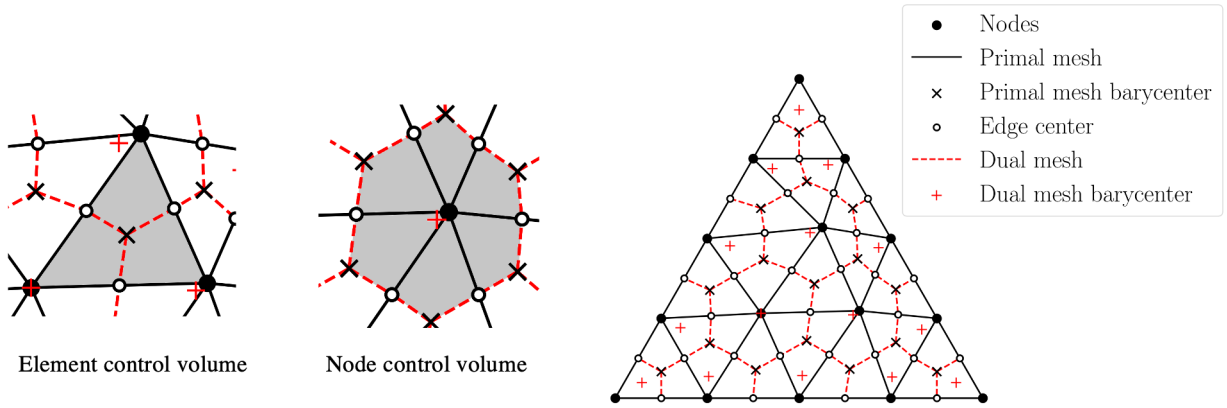


Figure 3.2: 2D Primal and Dual mesh. Primal mesh is obtained by joining the mesh nodes. Dual mesh is created by joining the center of the edges and the barycenters of the primal mesh. Control volumes are colored in grey.

For a variable ϕ, its volume average $\bar{\phi}$ is integrated as:

$$\bar{\phi} = \frac{1}{\Omega_i} \int_{\Omega_i} \phi \mathrm{d}\Omega \,, \tag{3.77}$$

with Ω_i is the given control volume. This control volume can be taken at the elements using a primal mesh or at the nodes using a dual mesh. Performing both time and spatial integration of the conservation equations leads to the following:

$$\frac{\overline{\phi^{n+1}} - \overline{\phi^n}}{\Delta t} + \frac{1}{\Delta t} \int_n^{n+1} \left[\frac{1}{\Omega_i} \int_{\Omega_i} \nabla \cdot \mathcal{F} \mathrm{d}\Omega \right] \mathrm{d}t = 0 \,. \tag{3.78}$$

Here, $\nabla \cdot \mathcal{F}$ represents the sum of the so-called Eulerian (or inviscid) and viscous fluxes. The application of Green-Ostrogradsky's theorem and the splitting of the control volume surface into the union of smaller surfaces S_i allows to rewrite the second term in its discrete form:

$$\frac{\phi^{n+1}-\phi^{n}}{\Delta t}+\frac{1}{\Delta t}\int_{n}^{n+1}\frac{1}{\Omega_i}\sum_{S_i}\vec{F}\cdot\overrightarrow{dS}\;\mathrm{d}t=0\,. \tag{3.79}$$

3.3.3 Multi-level domain decomposition

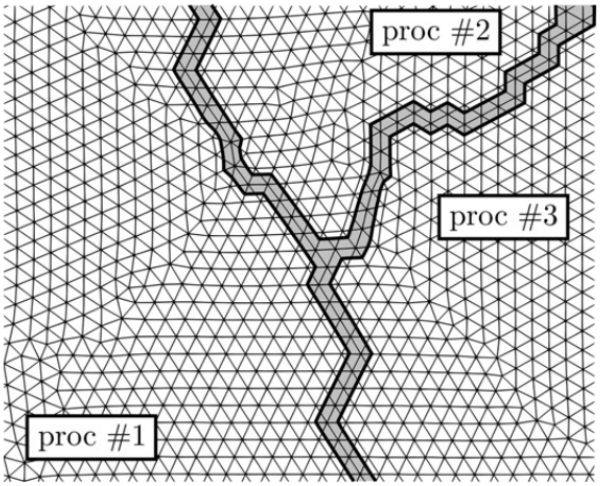


Figure 3.3: Single Domain Decomposition.

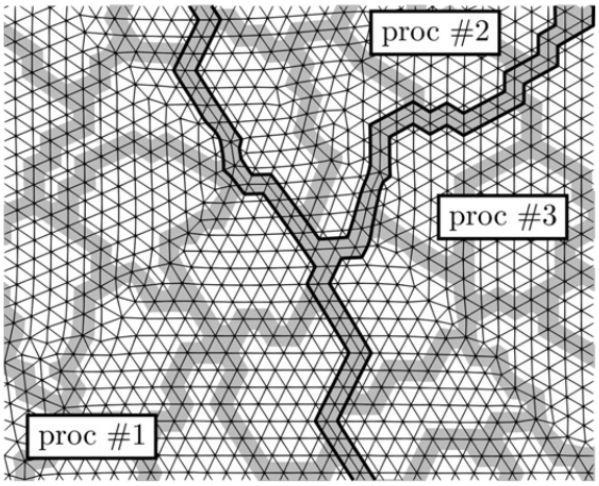


Figure 3.4: Double Domain Decomposition.

In order to be able to perform massively parallel computations, many CFD solvers adopt a parallelization strategy called Single Domain Decomposition, or SDD. This means that, when multiple processes are used to tackle a big problem, the mesh is split among those processes, and each one computes only its portion of the entire domain. However, the sub-problems that each process treats are not independent and data must be exchanged at the frontier between the processes. Fig. 3.3 represents this domain decomposition, with the region highlighted in dark grey being the interface between processes.

SDD is inefficient for memory accesses. By default data is stored in the RAM. When used for a read or write operation data is loaded into a temporary memory referred to as cache memory. Accessing data in the RAM has lot of latency but accessing cache memory is way more efficient. However, cache memory can only hold small amounts of data. Typical latencies and storage space are shown in Tab. 3.1. When performing operations on arrays, it is of interest to fit the whole working arrays into the cache. Thus, arrays should not exceed a few thousand elements, which is by far exceeded with SDD. This represents a huge constraint for CFD solvers as the ratio of operations per accessed data, also called the arithmetic intensity, is low.

	Latency (cycles)	Size
L1	~ 4	32KB
L2	~ 13	512KB
L3	~ 34	256MB
DRAM	~ 400	Several GB

Table 3.1: Memory size and access latency for AMD EPYC 7742 DDR4-3200 processor.

On each process, the sub-domain is split into smaller elements groups. This is referred as Double-domain decomposition (DDD) and shown in Fig. 3.4. Element groups range from 1000 to 5000 elements for optimal performance in YALES2. This decomposition acts as a cache blocking technique as the element groups are small enough to fit entirely in the cache memory. Some algorithms also benefit from the domain decomposition, like relocalization algorithms based on bounding boxes or load-balancing.
The downside of this double decomposition is the increased complexity of the connectivity between the nodes. Nodes that belong to multiple element groups are duplicated, increasing the total node count. Whenever a differential operator, like a gradient operator, is performed on those nodes, contributions of the duplicated nodes need to be reduced so that each duplicated node holds the same value.

3.3.4 Pressure-based low-Mach number formalism

3.3.5 Incompressible solver (ICS)

The density is assumed constant in space and time:

$$\frac{\partial \rho}{\partial t} = 0 \quad \frac{\partial \rho}{\partial x_i} = 0\,, \tag{3.80}$$

thus conservation of mass (Eq. 3.1) becomes:

$$\frac{\partial u_i}{\partial x_i} = 0\,, \tag{3.81}$$

stating that the velocity is a divergence free field.
Conservation of momentum (Eq. 3.2) reads:

$$\frac{\partial u_j}{\partial t} + \frac{\partial u_i u_j}{\partial x_i} = -\frac{1}{\rho}\frac{\partial P}{\partial x_j} + \frac{1}{\rho}\frac{\partial \tau_{ij}}{\partial x_i}\,. \tag{3.82}$$

These equations are solved based on Chorin's approach [85] that has later been modified by Kim and Moin [86]. In this approach, velocity is staggered with respect to pressure. The velocity field is decomposed into an irrotationnal and solenoidal part using the Helmholtz-Hodge decomposition:

$$u = u^i + u^s\,. \tag{3.83}$$

The solving of the momentum equation in YALES2 relies on this decomposition and is done in two steps, known as the prediction and correction steps.

- Prediction step: In the prediction step, a first guess $\mathbf{u}^*$ of the velocity at a time $n+1$ is computed without the irrotational contribution of $P^{n+1/2}$:

$$\frac{\mathbf{u}^* - \mathbf{u}^n}{\Delta t} = -\boldsymbol{\nabla}\cdot(\mathbf{u}^*\mathbf{u}^n) - \frac{1}{\rho}\boldsymbol{\nabla}P^{n-1/2} + \frac{1}{\rho}\boldsymbol{\nabla}\cdot\tau^n\,. \tag{3.84}$$

  Note that even if the pressure gradient only contributes to the irrotational part of the velocity field, the contribution of the gradient of $P^{n-1/2}$ is considered following the work of Kim and

Moin [86] to mitigate splitting errors. This term is subtracted in the correction step.

- Correction step: The velocity at time $\mathbf{u}^{n+1}$ is then corrected from the velocity $\mathbf{u}^*$ as:

$$\frac{\mathbf{u}^{n+1}-\mathbf{u}^*}{\Delta t} = -\frac{1}{\rho}\boldsymbol{\nabla} P^{n+1/2} + \frac{1}{\rho}\boldsymbol{\nabla} P^{n-1/2}\,. \tag{3.85}$$

This later step requires the pressure at time $n+1/2$. Such a field is obtained by taking the divergence of the prediction Eq. 3.84 to ensure a divergence free u^{n+1}:

$$\nabla^2\left(P^{n+1/2}-P^{n-1/2}\right) = \frac{\rho}{\Delta t}\nabla\cdot\mathbf{u}^\star\,. \tag{3.86}$$

3.3.6 Variable-density solver (VDS)

Density can change due to mixing of different fluids or chemical kinetics. Thus, mass and momentum equations can no longer be simplified as for the incompressible solver. The density is staggered relative to velocity. Density is predicted and corrected:

- Density prediction step: In the prediction step, a first guess ρ^* of the density at a time $n+3/2$ is computed based on mass conservation:

$$\frac{\rho^*-\rho^{n+1/2}}{\Delta t} = -\nabla\cdot\left(\rho^n\mathbf{u}^n\right). \tag{3.87}$$

- Scalar transport step: Scalars related to density are transported. For reactive flows the species and enthalpy are predicted. For a scalar ϕ:

$$\frac{\rho^*\phi^*-\rho^{n+1/2}\phi^{n+1/2}}{\Delta t} = -\boldsymbol{\nabla}\cdot\left((\rho\mathbf{u})^n\phi^{n+1/2}\right) + \mathcal{D}\left(\phi^{n+1/2}\right) + \mathcal{R}\left(\phi^{n+1/2}\right), \tag{3.88}$$

where $\mathcal{D}$ and $\mathcal{R}$ are the diffusive and reactive contributions.

- Density correction step: Density at $n+3/2$ is recomputed from the equation of state. For a multispecies gas using ideal gas law:

$$\rho^{EOS} = \frac{P_0}{RT^*}\left(\sum_{k=1}^{N_{sp}}\frac{Y_k^*}{W_k}\right)^{-1}, \tag{3.89}$$

where T^* is obtained by solving iteratively:

$$h_s^* = \sum_{n=1}^{N_{sp}} Y_k^*\int_{T_0}^{T^*} c_{p,k}\left(T'\right)dT'\,. \tag{3.90}$$

Density at $n+3/2$ is obtained as a weighting between the predicted transported density and the thermodynamic state density:

$$\rho^{n+3/2} = \alpha\rho^{EOS} + (1-\alpha)\rho^*\,. \tag{3.91}$$

This weighting is required to ensure numerical stability. $\alpha = 0.7$ is commonly used. For turbulent flames with fast changes of density, the value may be lowered to $\alpha = 0.3$ for increased stability. Density at $n+1$ is estimated linearly:

$$\rho^{n+1} = \frac{1}{2}\left(\rho^{n+1/2} + \rho^{n+3/2}\right) . \tag{3.92}$$

- Velocity prediction step: A first guess u^* of the velocity at a time $n+1$ is computed as:

$$\frac{(\rho\mathbf{u})^* - (\rho\mathbf{u})^n}{\Delta t} = -\boldsymbol{\nabla}\cdot((\rho\mathbf{u})^n\mathbf{u}^n) - \boldsymbol{\nabla}P^{n-1/2} + \boldsymbol{\nabla}\cdot\boldsymbol{\tau}^n . \tag{3.93}$$

- Velocity correction step: Velocity at u^{n+1} is corrected with the pressure:

$$\frac{(\rho\mathbf{u})^{n+1} - (\rho\mathbf{u})^*}{\Delta t} = -\nabla\left(P^{n+1/2} - P^{n-1/2}\right) . \tag{3.94}$$

Pressure is obtained by ensuring mass conservation at $n+1$:

$$\boldsymbol{\nabla}\cdot\left(\boldsymbol{\nabla}P^{n+1/2} - \boldsymbol{\nabla}P^{n-1/2}\right) = \frac{\rho^{n+3/2} - \rho^{n+1/2}}{\Delta t^2} + \frac{\boldsymbol{\nabla}(\rho\mathbf{u})^*}{\Delta t} . \tag{3.95}$$

- Scalar correction step: The previously predicted scalar values are corrected based on corrected density and velocity:

$$\frac{(\rho\phi)^{n+3/2} - (\rho\phi)^{n+1/2}}{\Delta t} = -\nabla\cdot\left((\rho\mathbf{u})^{n+1}\phi^{n+1/2}\right) + \mathcal{D}\left(\phi^{n+1/2}\right) + \mathcal{R}\left(\phi^{n+1/2}\right) . \tag{3.96}$$

This low-Mach number approach for variable density flows is fully mass, momentum and species conservative but it relaxes the equation-of-state which is not strictly satisfied at $n+1$.

3.3.7 Poisson equation

Both variable and constant density solvers (VDS - ICS) rely on the solving of a Poisson equation that can be generalized as:

$$\nabla\cdot(\nabla P) = RHS \Longleftrightarrow AX = B . \tag{3.97}$$

Such a linear system must be solved to compute the pressure for each node of the grid, which can be represented by a vector of unknowns X. Current linear solvers are based on iterative numerical methods, and a high number of iterations is often required to achieve the desired convergence tolerance. On top of that, the characteristics of the discrete Laplacian operator can influence the total number of required iterations. Note that if no special effort is made to optimize the solution of this equation, more than 90% of the time between each time step can be spent solving it [87]. Hence, YALES2 relies on the extensive use of performant linear solvers. Several algorithms are available, such as the Preconditioned Conjugate Gradient (PCG) [88], the Deflated PCG (DPCG) [89] or the BICGSTAB(2) scheme. The DPCG algorithm is used in this work as it relies on a coarse-grid preconditioning which drastically reduces the cost on large meshes.

3.3.8 Stiff integration of chemical source terms

The calculation of chemical source terms is a real challenge for the numerical simulation of reactive flows, due to the multitude of characteristic times that are not correlated with the other characteristic time scales of the fluid. The minimum chemical time step is of the order of 10^{-9} s, or even 10^{-12} s for hydrocarbon chemistry, under normal temperature and pressure conditions. This is several orders of magnitude less than the convective time step, which is usually 10^{-5} to 10^{-7} s.

The conservation equation of the species and enthalpy can be divided as follows:

$$\frac{\partial \Phi}{\partial t} = S\left(\Phi\left(t\right)\right) + M\left(\Phi\left(t\right)\right) , \tag{3.98}$$

where the source term S is the rate of change due to chemical reactions, and the mixing term M is the rate of change due to transport and molecular diffusion. This ODE may be decomposed using a splitting method. First, source terms are computed:

$$\frac{\Phi_S^{n+1} - \Phi^n}{\Delta t} = S\left(\Phi^n\right) . \tag{3.99}$$

From this intermediate state mixing is solved:

$$\frac{\Phi^{n+1} - \Phi_S^{n+1}}{\Delta t} = M\left(\Phi_S^{n+1}\right) . \tag{3.100}$$

This method has been shown to be first order accurate in time in [90]. The decoupling of mixing and kinetics allow to apply adequate time integration methods to each. Time step of the solver is chosen with respect to mixing and kinetics are sub-stepped. $S\left(\Phi^n\right)$ only depends on time without any spatial effects, therefore it is solved under the assumption of a 0D kinetic reactor. The reactor is assumed homogeneous and with constant pressure and enthalpy.

$$\begin{cases} \frac{\mathrm{d}\rho Y_k}{\mathrm{d}t} = \dot{\omega}_k \\ \\ \frac{\mathrm{d}h}{\mathrm{d}t} = 0 \\ \\ \frac{\mathrm{d}P}{\mathrm{d}t} = 0 . \end{cases} \tag{3.101}$$

The stiff integrator CVODE [91] is used to solve this system. It relies on the Backward Differentiation Formula (BDF) [92]. It is a linear multistep method that, for a given function and time, approximates the derivative of a function using information from already computed time points, thereby increasing the accuracy of the approximation. Stability analysis of the BDF allows to adapt locally the time step to the stiffness of the system. For reactive mixtures, time step is adapted to the characteristic chemical time and non-reactive mixtures are solved within a single step.

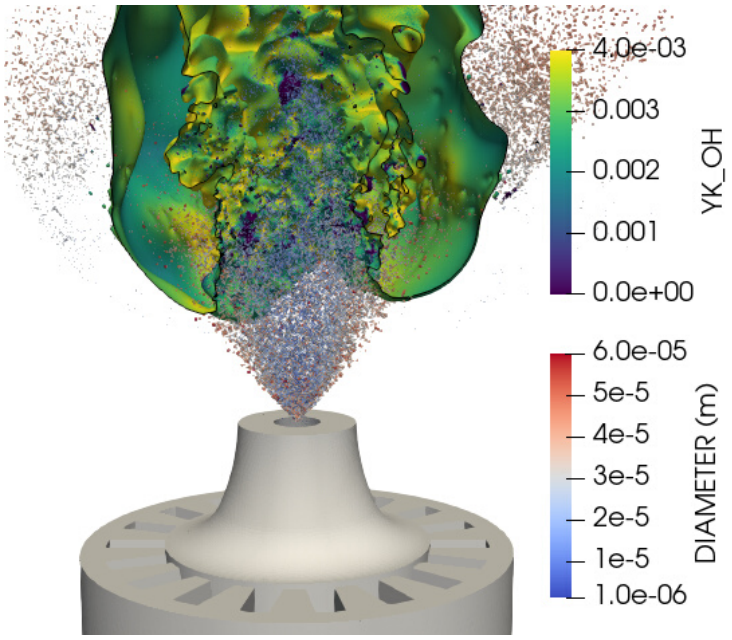


Figure 3.5: Stoichiometric mixture fraction iso-surface colored by Y_{OH}. Scaled spray droplets colored by diameter.

3.4 CRSB numerical set-up

This section outlines the various modeling choices made for the CRSB simulation. The finite-volume numerical methods employed are 4th-order accurate in both time and space. The low-Mach variable density approach of Section 3.3.6 is utilized, wherein the Poisson equation is solved using the DPCG method. The time step is constrained by the Courant–Friedrichs–Lewy (CFL) condition:

$$CFL = \frac{u\Delta t}{\Delta x} = 0.4\,, \tag{3.102}$$

where Δt is the time step, Δx the local mesh size and u the local velocity. CFL value is set relatively low to ensure good numerical stability of the reactive flows, resulting in a time step on the order of 1 μs. Diffusion is handled implicitly and is governed by the Fourier Number:

$$Fo = \alpha\frac{\Delta t}{\Delta x^2} = 0.15\,. \tag{3.103}$$

The Fourier number is computed for velocity, where α represents the viscosity, and for the species, where α denotes the related diffusion coefficient. Viscosity is determined using a power law (Eq. 2.25), and no artificial viscosity is introduced.

Finite-rate chemistry is integrated using CVODE [91] with the Analytically Reduced Chemistry (ARC) n-heptane scheme [93]. This scheme involves 24 species, listed in Table 3.2, enabling the progressive breakdown of n-heptane and its combination with the oxidizer through 217 reactions. Additionally, 32 Quasi-Steady-State species are incorporated in the process. NO_X chemistry is not accounted for, and N_2 is treated as a perfectly inert species.

Initial	Intermediate		Final	Inert
O_2	CH_3	O	CO_2	N_2
C_7H_{16}	CH_4	H	H_2O	
	C_2H_2	H_2		
	C_2H_4	OH		
	C_2H_6	HO_2		
	C_3H_4	H_2O_2		
	C_3H_6	CO		
	C_4H_6	CH_2O		
	C_4H_{10}	CH_3OH		
		C_2H_3CHO		

Table 3.2: Species involved in the ARC n-heptane mechanism.

Sub-grid scale transport is modeled using the dynamic Smagorinsky [40]. The Thickened Flame Model (TFLES) has been attempted with the proposed methodology to solve the inner reaction zone. However, thickening strongly increases the thermal inertia of the flame and decreases its sensibility to shear stress and to droplets traversing the front. This results in a lack of flame dynamics and less local extinctions. Additionally, the inner flame front is only partially premixed as it is supported by diffusion on its back-end [36]. Due to these limitations, no turbulent combustion model is used here. From the analysis of the simulations, flame fronts in the CRSB burn in two main regimes: very lean premixed and very rich diffusion flames. As a result, flame fronts are thicker than the equivalent flames at stoichiometry and the use of a minimal target element size of 200 μm ensures at least 5 points within reactive zones.

The liquid dispersed phase is modeled using a Lagrangian point-particle approach. Drag is obtained using the Schiller-Naumann correlation [73] and evaporation is represented with the Abramzon-Sirignano model [79]. The spray is depicted in Fig 3.5. The solver time step is determined based on the Eulerian phase only. Within the solver time step the particles can take smaller sub-steps if required:

- CFL condition: Particle motion should not be large compared to the mesh elements. Particle dynamics are computed based on locally interpolated values from the eulerian phase. These values need to be updated regularly during the particle displacement, thus sub-stepping. CFL=2 is used.

- Drag condition: The relaxation time for the drag is defined as:

$$t_{drag} = \frac{m_p |u_p - u_\infty|}{F^D}, \tag{3.104}$$

  when the drag time is less than 10% of the solver time, the particle is considered to be a tracer and the condition is dropped.

- Evaporation condition: Evaporated mass is directly related to the particle diameter and decreases as the particle shrinks. Therefore, large time steps would overestimate evaporation. Sub-step is constrained so that a particle evaporates less than 10% of its current mass. When

the particle diameter is smaller than 0.5μs it is fully evaporated to avoid increasingly tiny sub-steps.

- Heating condition: Particle temperature should not vary by more than 1K per sub-step.

For each Lagrangian droplet, the following parameters have to be set at the injection: temperature, diameter, velocity and injection angle. Temperature is set for all to the ambient temperature of 298K. Velocity is derived from the LISA (Liquid Injection for Swirled Atomizers) model [94] resulting in 33m.s^{-1}. Similar estimates can also be obtained using the FIM-UR (Fuel Injection Method by Upstream Reconstruction) model [95]. Mean injection angle is set to 40°, some droplets are emitted with a slightly smaller or larger angle. This can be explained experimentally due to the oscillation of the liquid sheet during atomization. Minimum and maximum angles as well as the droplet angle distribution in-between are uncertain. For large inertial droplet this could be impactful and control in which flame region they vaporize, ultimately affecting the flame shape. Droplet diameter distribution is set using a fitted probability density function (PDF). The PDF is derived from experimental data using a Rosin-Rammler correlation, which reads:

$$f\left(D\right) = \frac{qD^{q-1}}{X^q} exp\left(-\left(\frac{D}{X}\right)^q\right), \tag{3.105}$$

where q and X are the fitting parameters. X is related to the peak location of the distribution and q to the spread. X can be related to the Sauter Mean Diameter by:

$$D_{32} = \frac{X^3\Gamma\left(\frac{3}{q}+1\right)}{X^2\Gamma\left(\frac{2}{q}+1\right)}. \tag{3.106}$$

There is currently no consensus in the community on which values of D_{32} and q should be used for a good representation of the spray. Droplet diameter is measured experimentally using PDA, this method allows to perform the measure on a very small control volume at certain radial positions, but not on the whole injection plane at once, which would be required to properly set D_{32} and q. The values $D_{32} = 32\mu$m and q=2.3 are used as in prior work [96], but could likely be improved in the future.

Tab. 3.3 summarizes all the modeling choices.

3.4.1 CPU cost of simulation

Simulation cost of 1ms physical time of the CRSB with a mesh of 500 million cells, reaching 200 microns in the flame, is evaluated at 27k CPU hours. In order to obtain meaningful statistics, a convergence time of approximately 100ms is necessary, resulting in a total cost of 2.7M CPU hours. The CPU cost can be expressed as:

$$CPU_{cost} = \frac{\text{Total Physical Time}}{dt} N_{nodes}(RCT_{Euler} + RCT_{Lagrange}) \underbrace{\left(\times \frac{1}{3600.10^6}\right)}_{\mu s \text{ to hours}}, \tag{3.107}$$

CFD code	YALES2
Solver	Low mach - Variable density
Temporal scheme	4^{th} order (TFV4A)
Spatial scheme	4^{th} order
CFL	0.4
Fourier	0.15
Poisson solver	DPCG
Turbulence model	Dynamic Smagorinsky
Turbulence chemistry model	None
Viscosity	Powerlaw
Artificial viscosity	None
Chemistry	Finite-rate chemistry
Chemistry stiff integrator	CVODE
Chemistry kinetic scheme	n-heptane ARC (24S 217R)
Species transport	Mixture averaged
Lagrangian drag	Schiller-Naumann
Lagrangian evaporation	Abramzon-Sirignano

Table 3.3: CRSB numerical set-up summary.

where dt is the time step, the RCT is the Reduced Computational Time:

$$RCT = \frac{\text{Wall clock time}(\mu s).N_{\text{process}}}{N_{\text{nodes}}.N_{\text{iterations}}}. \tag{3.108}$$

The RCT is expressed per node and not per element because the control volumes in YALES2 are the nodes (use of the dual mesh). Based on this equation of the CPU expense, optimizations can be performed on several terms to mitigate the cost.

The CPU cost scales linearly with the number of control volumes (nodes) as described by Eq. 3.107. In practice, the scaling can be even worse due to parallel overheads. Consequently, reducing the mesh size is of significant interest to minimize computational costs. This reduction can be achieved by adapting the mesh to the physical phenomena of interest. Regions of importance should have a fine resolution, while less critical regions can be represented with a coarser resolution, thereby reducing the total number of control volumes and saving on computational resources. Fig. 3.6 shows an illustration of such a mesh optimization for a 2D bunsen flame. Chapter 5 delves into this topic in detail, exploring various strategies for mesh adaptation to optimise performance while maintaining the accuracy of the simulation.

The RCT is a key performance indicator of the numerical methods employed to solve the physics equations within each control volume. The RCT for the unoptimized CRSB simulation is depicted in Fig. 3.7. This figure breaks down the RCT into the main physical phenomena addressed during each solver iteration, providing insight into the computational demands of each component of the simulation. Cost of initiating the simulation and saving the solutions are not accounted for. The main costs can be explained as follows:

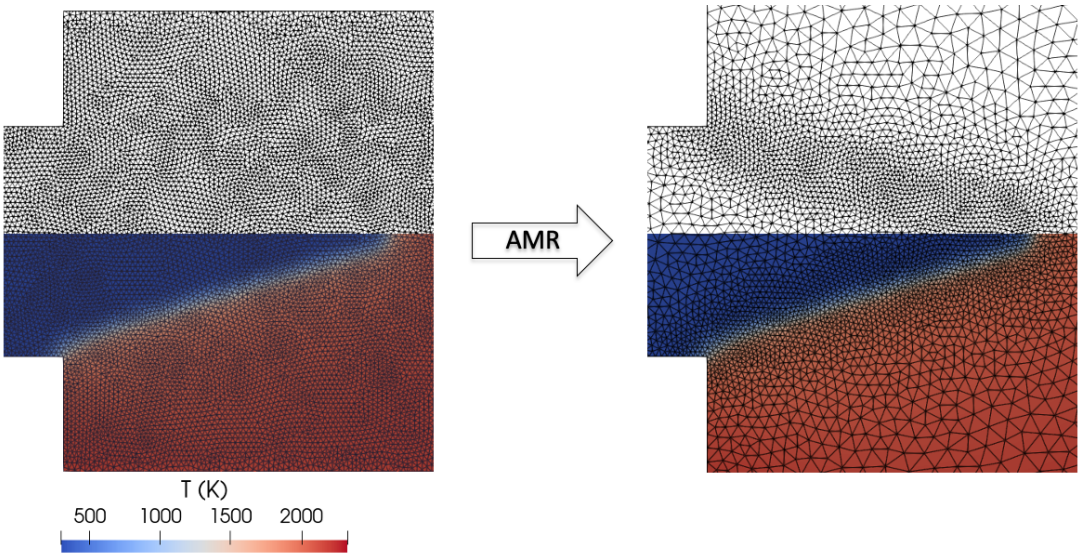


Figure 3.6: Homogeneous mesh (left) and optimised mesh (right) using Adaptive Mesh Refinement (AMR) for a 2D bunsen flame

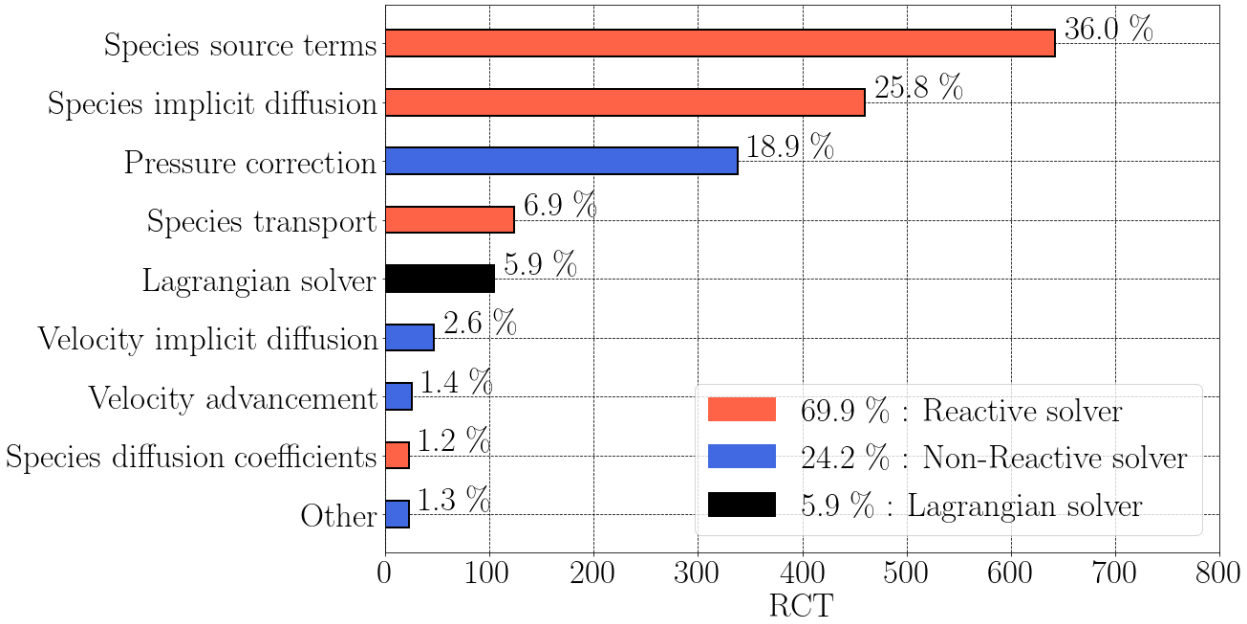


Figure 3.7: Detailed RCT of the unoptimized CRSB simulation.

- **36% Species source terms:** Source terms are obtained by integrating the kinetic mechanism. This process incurs high computational costs due to the high dimensionality of species and reactions, as well as the stiffness of the problem, which requires a large number of substeps in reactive regions, even with the use of a stiff ODE integrator. This computational challenge is addressed in this thesis, with Chapter 6 dedicated to exploring strategies for optimizing the integration of source terms and mitigating these costs.

- **25.8% Species implicit diffusion:** The high computational cost is attributed to the high

dimensionality of the species and the significant number of diffusion sub-steps required. The diffusion of species is performed implicitly with a Fourier number of 0.15. For the CRSB on a 200μm grid resolution, an average of 20 sub-steps are executed. The number of sub-steps depends on the grid resolution, as the CFL time step scales with Δx^{-1} and the diffusive time step scales with Δx^{-2}. Consequently, on more refined grids, this cost will become increasingly problematic. While this issue is not addressed in this thesis, it is identified as an area for future studies to explore and optimize.

- **18.9% Pressure correction:** The underlying numerical methods have already been thoroughly optimized, as explained in Sec. 3.3.7, leaving little room for further improvement. Solving the Poisson equation, which is a global parallel operation, tends to scale poorly as the mesh size increases. Therefore, limiting the mesh size through adaptive mesh refinement will have a beneficial impact on this computational cost.

- **6.9% Species transport**: This operation is expensive due to the high number of species involved. Future optimizations could be achieved through dimensionality reduction techniques.

- **5.9% Lagrangian solver**: The interpretation of the Lagrangian RCT differs significantly from that of the Eulerian approach. This is because the Lagrangian RCT scales with the number of particles rather than the number of control volumes. Currently, the number of particles per process is not regulated, leading to considerable fluctuations in the RCT. This can result in unpredictable and potentially high RCT peaks, which are challenging for the user to anticipate. The issue of load balancing for Lagrangian particles is thoroughly addressed in Chapter 4.

The cost analysis of the CRSB has revealed significant potential for numerical optimizations. These optimizations are now thoroughly addressed in the forthcoming chapters.

CHAPTER 4

Load-balancing of Euler-Lagrange spray burner simulations

This chapter focuses on the parallel load-balancing of the Euler-Lagrange simulation. First, it introduces the common load-balancing strategy used in Large-Eddy Simulations. Then, the limitations of this strategy are shown for Euler-Lagrange simulations. A novel load-balancing strategy is introduced to deal with both the Lagrangian and Eulerian phases.

Published in International Journal of Numerical Methods in Fluids (2023) [97].

Contents

4.1 Introduction to parallel computing

Large-Eddy simulation capabilities rely strongly on the availability of computational resources and the parallelization of the code. Central to the rapid advancements in computing power is Moore's law, an observation made by Gordon Moore in 1965 [98]. Moore's Law predicts that the number of transistors on a microchip doubles approximately every two years, leading to exponential increases in computational power and decreases in relative cost for the users. This trend can easily be observed based on the Top500, ranking of the most powerful computing system in the world [99], shown in Fig. 4.1. Most performant machines are nowadays able to cross the exascale barrier, of 10^{18} floating point operations per second.

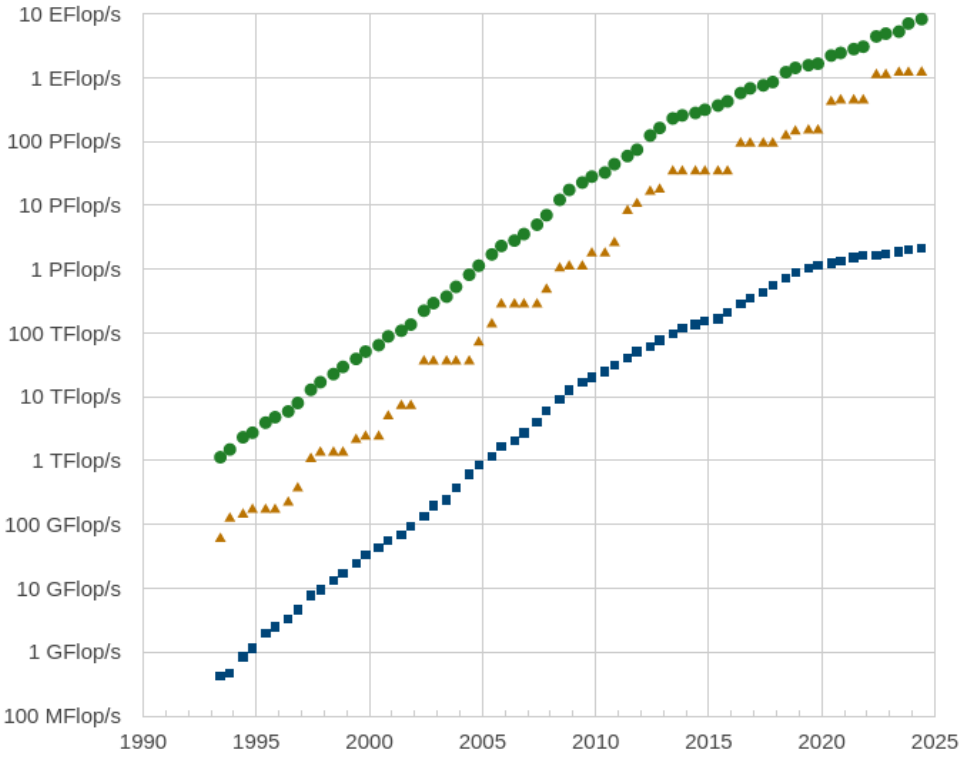


Figure 4.1: Development of supercomputer performance over the years. Green circles correspond to the summed performance of Top 500 machines, yellow triangles to the performance of the Top 1 machine and blue squares to the top 500th machine performance. Reproduction from [99].

CFD codes need to keep up with the advances in computing power and propose strategies to efficiently utilize a high number of processors. The speedup S_{max} of a parallel code can be measured using Amdahl's law [100]:

$$S_{max} = \frac{\text{Sequential run time}}{\text{Parallelised run time}} = \frac{1}{(1-p) + p/s}\,, \tag{4.1}$$

where p is the temporal fraction of the code run in parallel and s the speedup of this parallel part with:

$$s \leq n_{proc}\,. \tag{4.2}$$

In order to achieve the best speedup all part of the code should be parallelized so that $p \to 1$. Parallel speedup s mainly depends on two factors. Parallel overheads and workload distribution.

In order to understand parallel overheads hardware needs to be considered. Parallel computing involves executing multiple processes—instances of a program simultaneously, with each process running on a separate hardware core. Cores are organized and linked to memory within Central Processing Units (CPUs). There are two primary memory architectures: shared memory and distributed memory, illustrated in Fig. 4.3 and Fig. 4.2. In shared memory systems, all processors have access to a common memory space, allowing for easy communication and data sharing. This architecture is typically simpler to program but can suffer from memory contention as the number of processors increases. Distributed memory systems, on the other hand, consist of multiple processors, each with its own local memory. Processors communicate by passing messages, which can scale more efficiently for large numbers of processors but is more complex to program. Recent developments have increasingly favoured the shared memory approach. For a detailed explanation of CPU architecture, refer to [101] and manufacturer specifications, such as those from AMD in [102]. Modern shared memory CPUs can contain up to 64 cores. In High-Performance Computing (HPC) clusters, CPUs are further grouped into nodes as shown in Fig. 4.4, with the total performance of the CPU cluster dependent on the number of nodes. As a result, HPC cluster act as a shared-distributed memory hybrid, with shared memory at the CPU level and distributed memory at the node and cluster level.

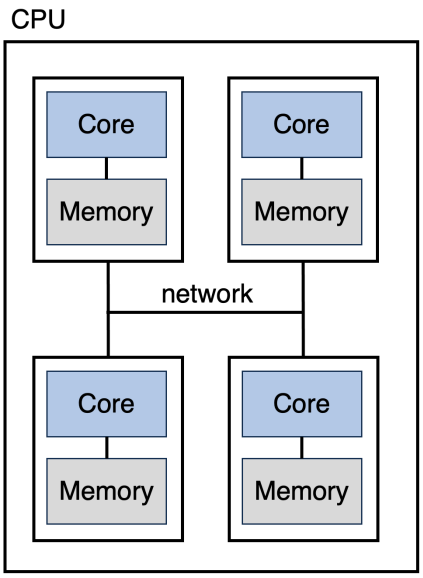


Figure 4.2: Distributed memory.

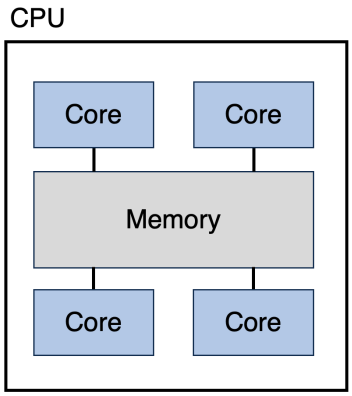


Figure 4.3: Shared memory.

Data exchanges need to be performed between the processes and are referred as communications. These communications allow to synchronise work performed by the multitude of processor and perform global operations. The two most common approaches are MPI [103] and OpenMP [104].

- OpenMP is an application programming interface for platform-independent shared-memory parallel programming. It simplifies parallel programming by using compiler directives to

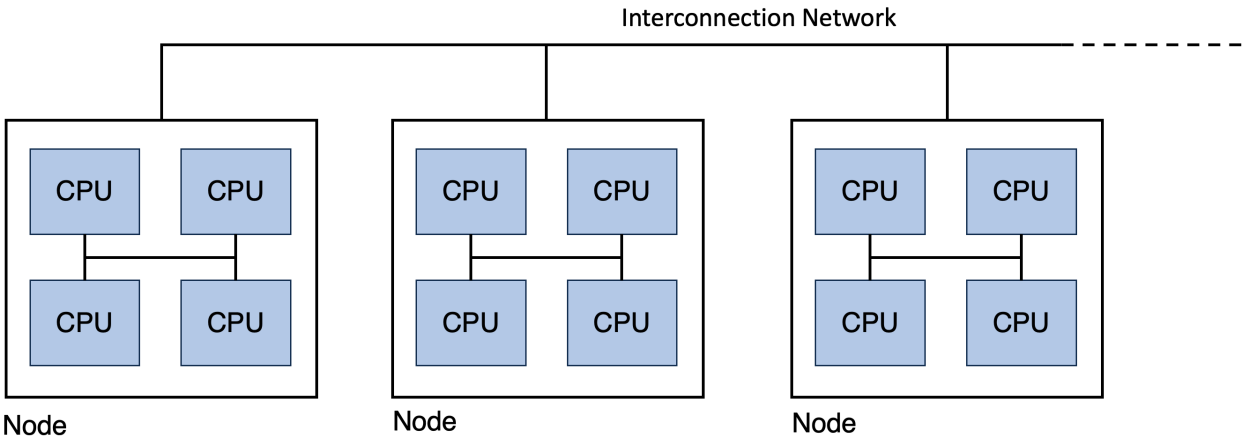


Figure 4.4: HPC cluster.

parallelize code on multi-core processors. OpenMP is easy to implement for parallelizing loops and sections of code, making it suitable for applications running on shared memory systems where tasks can be efficiently divided among cores. As downside, it can create concurrent memory accesses acting as a bottleneck and is unable to function on distributed memory systems.

- Message Passing Interface (MPI) is a standard for distributed memory systems. It enables communication between processes running on different nodes in a cluster by passing messages. It is also able to run on shared memory system but at a lower efficiency than OpenMP. MPI is highly scalable and can be used on both small and large-scale systems. It is particularly useful for applications where tasks can be divided into independent processes that need to communicate occasionally.

MPI has to be used as the HPC cluster acts as a distributed memory system on the node level. Hybrid MPI—OpenMP approaches often allow even greater parallel scaling. Overall, it is of interest to keep communications minimal as they introduce a cost overhead, especially on massively parallel jobs. For MPI communication, overhead can be expressed as:

$$t_{comm} = \alpha + \frac{n}{\beta}\,, \tag{4.3}$$

with α the base cost to initiate the communication (or latency) and n/β the cost based on the amount of transferred data n through a network of bandwidth β. Thus, cutting communication cost can be done by limiting the frequency of communication and the amount of data to exchange. Overhead also scales with the complexity of the communication, i.e. the number of processes that are involved. More details about parallel communications in CFD, with focus on YALES2, can be found in [105].

The second factor limiting parallel speedup is the load-balancing, which is the main subject of this chapter. Load-balancing refers to the distribution of computational tasks or workloads across multiple resources, such as processors. The amount of workload on each processor should be the

same to avoid idle time on any process, i.e. loss of resources and thus deteriorated speedup. Two main load-balancing methods exist: static and dynamic load balancing. Static load balancing involves making an a priori estimate of the workload and distributing it among the processes accordingly. However, during execution, imbalances may arise due to inaccurate initial workload estimations, leading to inefficiencies and underutilization of resources. Dynamic load balancing, on the other hand, adjusts the distribution of the workload in real-time based on the current state of the system. In this method, a master process continually monitors the workload and reallocates tasks to maintain an even distribution, thus adapting to changes and ensuring more efficient utilization of resources. Dynamic load balancing involves several strategies, such as work-stealing, where idle processors take over tasks from busier processors, and work-sharing, where tasks are proactively distributed from overloaded processors to those with spare capacity. The choice of the load-balancing approach depends strongly on the underlying problem. The following sections detail the load balancing strategies used in CFD simulation, with focus on Euler-Lagrange approaches.

4.2 Eulerian load-balancing

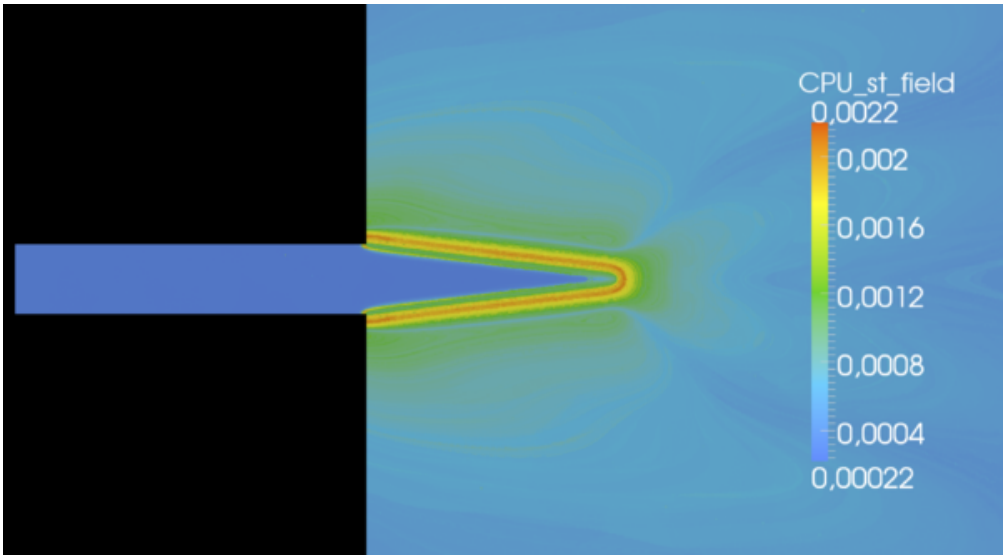


Figure 4.5: Field of chemical source terms CPU cost in seconds per node on the 2D Bunsen flame.

In Eulerian approaches, load-balancing methods often consist in giving each core the same amount of contiguous mesh elements. It is referred to as domain decomposition and achieved by using single-constraint partitioning methods [106, 107]. This form of load balancing is highly efficient for pure aerodynamic simulations [108]. This is due to the fact that aerodynamic phenomena are solved using the same set of equations at each mesh element making the cost proportional to the number of elements. However, these methods struggle with multi-physics problems that may introduce space or time irregular workload. For spray burners, two phenomena are to be considered: i) dispersed two-phase flows solved with an Euler-Lagrange approach, which will be discussed in detail in section 4.3. ii) Chemical source terms. When taking a finite-rate chemistry approach to

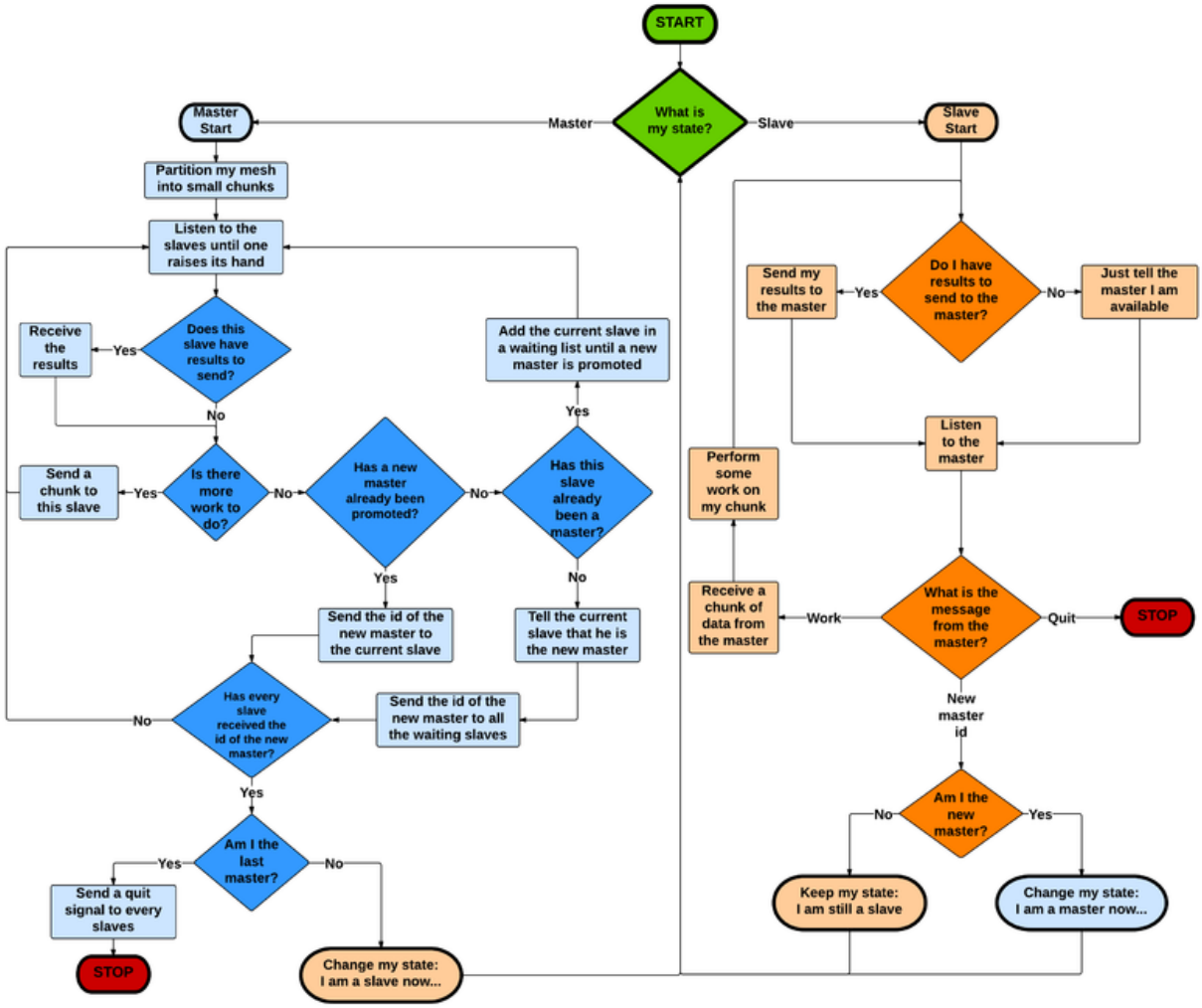


Figure 4.6: Dynamic scheduler algorithm flowchart in YALES2.

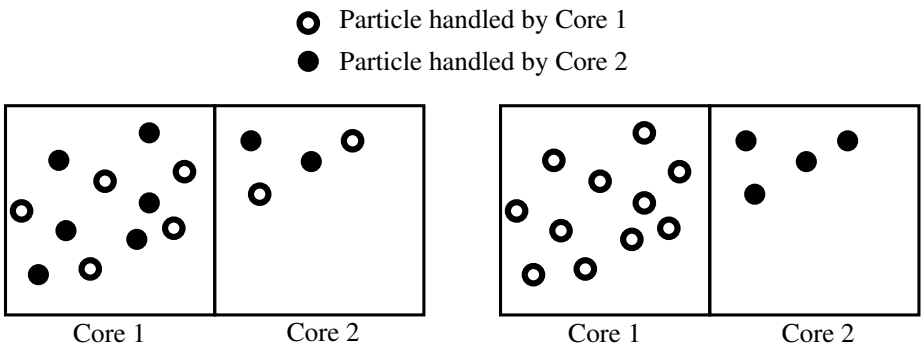
solve the source terms, cost is dependent on the stiffness of the problem. In non-reactive regions, the solution is trivial and requires little to none computational effort. In highly reactive regions, the problem is stiff and needs to be sub-stepped a lot making it very costly. This can be seen in Fig. 4.5, showing the source term CPU cost per node for a premixed 2D bunsen flame. Lowest cost is in the unburnt gases on the left, burnt gases have a low cost as well as only a few slow reaction take place. Highest cost is located in the reactive front where most of the reaction take place. A simple partitioning ensuring the same number of elements for each process fails to load-balance this case as one process could have mainly reactive nodes and another only unburnt gases. This load-balance issue can however be solved without changing the partitioning. Instead, a dynamic scheduling approach is used. First, groups of processes are formed, with about 32 processes per groups. Groups are formed so that source term cost is evened out between the groups based on the cost of the prior flow iterations. Within each group, a master process is defined and distributes its workload dynamically to the other available processes, which send back the resulting source terms. When all the source terms of the master have been computed, another process becomes

the master, this is done until all processes have been master and had their source terms computed. This approach is efficient because the amount of data to be sent and the parallel latency is low compared to the cost of source term computation. A more detailed overview of the process is given in Fig. 4.6.

4.3 Euler-Lagrange load-balancing

4.3.1 Particle sharing

Particle-laden flows can be handled in two separate ways. The first one, consists in distributing the particles evenly among the cores with no regards to their spatial localisation on the mesh. This is known as particle sharing algorithm [109, 110, 111, 112] or as dual grid approach in AMReX [113] and is represented on Fig. 4.7a. It ensures a close to perfect load balance since the Eulerian and Lagrangian constraints are fully decoupled: a particle and its containing mesh cell are not owned by the same core. As a consequence, every Euler-Lagrange interaction will require parallel communications which can result in high overheads and thus bad parallel scalability on large core counts. Mirror domain decomposition has been proposed by [114] to solve this issue. A complete copy of the data, mesh and particles is given to each core. However, this drastically increases memory usage and forces many communications to keep data synchronised, rendering it inefficient on more than 32 cores [114].



(a) Particle sharing **(b)** Spatial particle partitioning

Figure 4.7: Illustration of particle sharing and spatial particle partitioning mechanisms.

4.3.2 Spatial particle partitioning

In large parallel simulations of particle-laden flows, frequent data exchanges between the Eulerian and Lagrangian phases often impose to have a consistent spatial partitioning of the two phases. The particles and mesh elements are distributed among the cores based on the same domain decomposition. This method is known as spatial particle partitioning and is represented in Fig. 4.7b. Communications due to Euler-Lagrange interactions are reduced, but particles can be ill balanced. For rather homogeneous particle distributions, spatial particle partitioning is efficient [114]. However, if the particle distribution is heterogeneous, Lagrangian parallel efficiency is likely to be deteriorated [115]. Heterogeneous particle distributions occur in many common CFD applications,

like spray injectors used in combustion and atomization processes. A typical spray injector particle distribution is characterised by a high particle concentration close to the injection point. Further downstream, the particle concentration decreases sharply due to the particles spreading out or being evaporated. The challenge is to propose a partitioning that provide simultaneously a good load balance for the Eulerian elements and the Lagrangian particles.

A first idea to perform an ideal partitioning for both Euler and Lagrange phases would be to gather the partitioning weights into a single constraint. The weights here correspond to the total amount of work associated either to a Lagrangian particle or to an Eulerian control volume. Then, any classical graph partitioning tool such as METIS [116] or SCOTCH [117] could be used to perform the mesh partitioning. This is relevant when the particle distribution is close to homogeneous but provides a bad individual Eulerian and Lagrangian balance as soon as the particle load becomes very irregular. Limits of this approach will be shown during validation (Fig. 4.17).

The second idea is to use a double-constraint approach where both Lagrangian and Eulerian weights are considered separately by the partitioning algorithm. Double-constraint partitioning is a more complex problem than single-constraint partitioning as both constraints may contradict one another. This is the case for Euler-Lagrange simulations, especially when the particle distribution is very heterogeneous. On the one hand, the Lagrangian constraint attempts to maximize the number of cores in regions with high particle concentration. On the other hand, the Eulerian constraint attempts to provide a very homogeneous partitioning without any local core concentration. Standard partitioning libraries like METIS [116] provide double-constraint load balancing routines, i.e. the load imbalance and the cost function minimization during graph partitioning are evaluated for two different constraints at the same time. However, these methods rarely converge to a solution when contradictory constraints are requested, making them too unreliable to be used. Load refinement algorithms are able to bypass this issue: they take a basic partitioning as an input and attempt to improve its balance. Usually, a single constraint Eulerian partitioning serves as the initial state and the Lagrangian constraint is improved by shifting load away from the most loaded parts, until no more improvement can be made. Methods based on hierarchical domain decomposition [118] and diffusion have been developed [119]. In these approaches the load is refined by shifting the boundaries between the cores and notable improvement on the load balance is made. However, refinement tends to deteriorate one constraint to improve another, resulting into a compromise between both constraints. In these cases, it is likely for the efficiency gain on one constraint to be absorbed by the efficiency loss on the other.

4.3.3 Hybrid approaches

Hybrid approaches combine particle sharing and spatial particle partitioning. Computational cores are grouped into a larger structures called nodes. Inside a node, all cores benefit from shared memory access, meaning that each core can access every particle and mesh part on the node. A particle sharing algorithm can be used inside a node without any data exchange or data duplication. Therefore, the load is well balanced inside each node, without any of the previously mentioned downsides. But since the memory is not shared among the nodes, spatial partitioning must still be performed at the node level, which can finally result in load imbalance among the nodes. This

method can provide great performance improvement and it has been shown to scale well for application with 100,000 particles [120]. Recent developments have been proposed to improve this method by adding an asynchronous Euler-Lagrange framework [121, 122]. The downside of this method is its heavy dependency on the hardware while still requiring a load-balancing algorithm at the node level. Hybrid approaches are not considered in this works but could also benefit from the improvements suggested hereafter for spatial particle partitioning.

4.4 DOB-EL

A new method for double-constraint spatial particle partitioning is introduced. It is referred to as DOB-EL (Diffusion based Orthogonal load Balancing for Euler-Lagrange simulations). Its key feature, orthogonality, implies that the Lagrangian balance can be improved without any alteration to the already balanced Eulerian load. The method has been designed to be generic and work on most simulation set-ups. It is able to handle massively parallel simulations with complex geometries and unstructured 3D meshes. It can be performed either statically or dynamically during a simulation. A few guidelines are provided below to assess if DOB-EL should be used to optimise a given simulation based on the results described in this section. Because DOB-EL starts from a given partitioned mesh and particle sets, it performs best on large heterogeneous particle distributions. It should be avoided to perform DOB-EL on:

- Large particle distributions with low particle density gradient within the particle populated area, this includes among others fluidized beds. This is because the sub-part exchange process bases itself on the particle gradient between the sub-parts, if no gradient is present no exchanges will be done and the final state will be equal to the initial state.

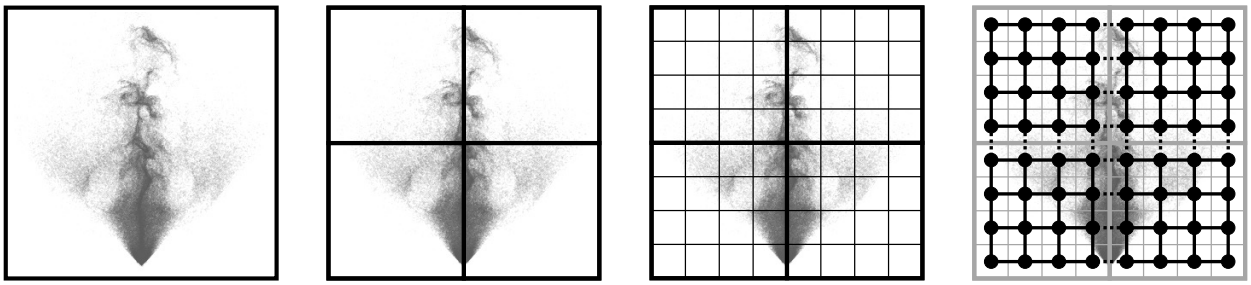
- Small simulations, the method has a some overhead that would deteriorate performance on these small simulations.

- The Lagrangian cost should not completely dominate the Eulerian cost, if it does, not considering the Eulerian constraint and doing single-constraint Lagrangian load balancing is likely to be more efficient.

4.4.1 Double domain decomposition

Dealing with each mesh element individually is not a viable strategy for load balancing as meshes can be composed of billions of elements. The solution is to use a double-domain decomposition to create small contiguous groups of elements that will act as elementary bricks for all mesh manipulations. This double domain-decomposition technique is illustrated in Fig. 4.8 and can be decomposed into two steps.

Step 1: A first domain decomposition is performed to split the mesh into contiguous parts with a similar amount of mesh elements. The number of parts created equals the number of computational cores, thus each core receives a single part. This is commonly done in parallel CFD codes. Fig. 4.8b represents the first domain decomposition.



(a) CFD Domain

(b) Single domain decomposition with 4 cores

(c) Double domain decomposition with 4 cores and 16 subparts per core

(d) Graph created based on the double domain decomposition

Figure 4.8: Double domain decomposition and associated graph. This is a simplified example with a square shape decomposition but sub-parts may have irregular shapes forming no pattern.

Step 2: A second domain decomposition is then performed to split each part into contiguous sub-parts with a similar amount of mesh elements. These will act as the elementary bricks for the load balancing algorithm. The impact of the number of sub-parts on the efficiency of the DOB-EL method will be discussed extensively in section 4.4.3. Fig. 4.8c represents the second domain decomposition.

Both decompositions are achieved by using a single-constraint partitioning algorithm: the multilevel k-way partitioning [123] available in METIS [116] has been used throughout this work.

Finally, a non-oriented graph based on this double domain decomposition can be built and used in the load balancing process. Each node of the graph represents a single mesh sub-part and the edges represent the connectivity between theses sub-parts. The core to graph vertex correspondence is referred to as the graph coloring, each color representing a given core. Fig. 4.8d represents a graph built based on the double domain decomposition.

The following notations are used throughout this chapter:

- n : total number of vertices of the graph / total number of sub-parts in the mesh
- k : number of cores used in the simulation
- V : set of all n vertices, represents the whole mesh
- V_c : set of all vertices on core c, represents a mesh part
- v_i : i^{th} vertex, represents a mesh sub-part

In order to describe the Eulerian and Lagrangian constraints on the graph, the following weight functions are introduced:

- ω_e : Eulerian weight function
- ω_l : Lagrangian weight function

The weights are real values associated to each vertex of the graph which are then used by the load balancing algorithm whose goal is to achieve an equipartition of these weights. They can also be evaluated for vertex sets as the sum of the weight of all vertices. Throughout this work, the most basic weight model is used, i.e. ω_e counts the number of elements per vertex and ω_l counts the number of particles per vertex. This assumes that every element has the same Eulerian cost and that every particle has the same Lagrangian cost. More complex models could be implemented in the same manner.

Balance is attained for a constraint when the associated weight function result is the same for all parts. A more general way to express this is to introduce the imbalance function. The Lagrangian imbalance of a sub-part V_i is:

$$LI(V_i) = k\frac{\omega_l(V_i)}{\omega_l(V)}\,. \tag{4.4}$$

$LI(V_i) < 1$ means that V_i is underloaded, $LI(V_i) > 1$ means that V_i is overloaded and $LI(V_i) = 1$ corresponds to the perfect balance.

Performance is dependent on the maximum load, that is to say the maximum imbalance. Therefore, the aim of the load balancing is to reduce the maximum imbalance. The maximum Lagrangian imbalance will be referred as LI_{MAX}.

4.4.2 Diffusion and dimension exchange

The load balance is improved by moving sub-parts from one core to another, which on the graph consists in changing the color of the associated vertices. While load-balancing the graph, the mesh is left untouched and will be updated later on. The first step is to identify which cores should exchange load and how much load has to be exchanged [124]

This is achieved by using a diffusion based algorithm [125]. Diffusion is a well-known iterative load-balancing method in which tasks are moved from heavy-loaded parts to lightly-loaded neighbor parts. Assuming that no tasks are created during load balancing, the discrete diffusion of the Lagrangian load can be expressed as a Laplace-Beltrami operator [126]:

$$\omega_l^{t+1}(V_i) = \omega_l^t(V_i) + \sum_j \alpha_{ij}\left(\omega_l^t(V_j) - \omega_l^t(V_i)\right)\,, \tag{4.5}$$

where t is the current iterative step and α_{ij} is the fraction of load difference to be exchanged. In the present algorithm, we choose to set $\alpha_{ij} = 0$ if V_i and V_j share no edge. It can be noted that the proposed approach can be seen as a first-order time scheme. Higher-order schemes exist for iterative diffusion but they bring no improvements on systems with coarse grain loads [127]. The convergence of Eq. 4.5 is guaranteed if and only if [126]:

$$\alpha_{ij} \geq 0 \qquad \forall i,j \in [1;k]\,, \tag{4.6}$$

$$1 - \sum_j \alpha_{ij} \geq 0 \qquad \forall i,j. \tag{4.7}$$

Eq. 4.5 assumes that each part interacts with all its neighbors at every step. However, this does not perform well on supercomputers as point to point communications can cause an important network contention. Dimension exchange on hypercubes has been introduced by Cybenko [126] to address this specific issue. The main idea of this algorithm is to traverse all the dimensions of the hypercube sequentially and to perform load exchange between neighboring vertices in this dimension only at a given step. The dimension exchange algorithm is not limited to hypercubes as it has been extended to all graphs [128][129]. A dimension exchange step is expressed as:

$$\omega_l^{t+1}(V_i) = \omega_l^t(V_i) + \alpha_{ij}\left(\omega_l^t(V_j) - \omega_l^t(V_i)\right) . \tag{4.8}$$

The convergence criteria given by Eq. 4.6 and Eq. 4.7 can be simplified as:

$$0 \leq \alpha_{ij} \leq 1 \qquad \forall i, j \in [1; k] . \tag{4.9}$$

The intermediate value $\alpha_{ij} = \frac{1}{2}$ is commonly used as it usually grants the fastest convergence rate.

On top of its simplified formulation, dimension exchange also converges faster than regular diffusion. An analogy can be made for dimension exchange and diffusion with respectively a Gauss-Seidel and Gauss-Jacobi method [126].

4.4.3 Orthogonal load exchange

While dimension exchange specifies the amount of load to be exchanged, it does not state how to select the load to be exchanged. This subsection details the selection rules that are applied to exchange load orthogonally while preserving a good aspect ratio of the parts.

Orthogonality

Orthogonality consists in the ability of changing the balance of one constraint without impacting the other by any mean. In the present case, Eulerian balance is considered to be good and should not be altered when improving the Lagrangian balance.

Let us assume that the partitioning algorithm used for the double domain decomposition is ideal. The parts are built based on the Eulerian constraint: this means that after the first domain partitioning, each part holds exactly the same Eulerian load:

$$\omega_e(V_i) = \frac{\omega_e(V)}{k} \qquad \forall i \in [1; k] . \tag{4.10}$$

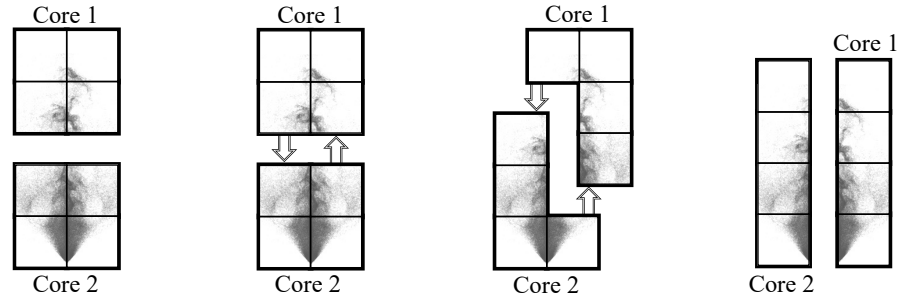
After the second partitioning, if the same number of sub-parts are created from each part, every sub-part holds exactly the same Eulerian load:

$$\omega_e(v_i) = \frac{\omega_e(V)}{n} \qquad \forall i \in [1; n] . \tag{4.11}$$

All load exchanges are performed by swapping sub-parts between processors: each time a sub-part is given away, an other sub-part is received. This swapping mechanism ensures the preservation

of the Eulerian balance as the given and received sub-parts have the same Eulerian load and 7500 CV/proc. However, both sub-parts do not have the same Lagrangian load and the swap will change the Lagrangian balance: swaps can then be selected in such a manner that Lagrangian balance gets strictly improved.

Fig. 4.9 gives a representation of this swapping process. Both cores have 4 sub-parts at any time, that is to say the same Eulerian load. Initially, the core #2 holds the majority of the particles, the Lagrangian balance is evened out after two swaps, as both parts have the same amount of particles in the final state. This is an ideal example, in real cases, the Lagrangian balance improves but doesn't reach a perfectly balanced state as shown later in Sec 4.5.2 and Sec 4.6.2.



(a) Initial state **(b)** First swap **(c)** Second swap **(d)** Final state

Figure 4.9: Load balancing using the orthogonal swapping process. This is a simplified example with a square shape decomposition but sub-parts may have irregular shapes forming no pattern.

Selection of swapped vertices

Let V_H and V_L be two adjacent parts such that:

$$\omega_l(V_H) - \omega_l(V_L) = D > 0\,. \tag{4.12}$$

Let v_H and v_L be the swapped vertices so that:

$$\omega_l(v_H) - \omega_l(v_L) = d\,. \tag{4.13}$$

In order to ensure optimal swaps, the following rules are used:

- (R1) : Convergence towards load homogeneity

  Convergence requirement expressed by Eq. 4.9 can be reformulated as:

  $$0 < d < D \tag{4.14}$$

  In order to speed up convergence, Eq. 4.14 can be narrowed:

  $$c < d < D - c\,, \tag{4.15}$$

  with c a positive constant.

- (R2) : Contiguity

  V_H and V_L have to remain contiguous at all times. This is a vital condition for some CFD applications and increases part quality.

- (R3) : Edge-cut control

  Many pairs of vertices satisfy (R1) and (R2). Using the pair ensuring the faster convergence would result in very entangled parts and large edge-cut. The edge-cut of a part is the number of its external edges, i.e. the number of edges connecting it with another part. It is thus directly linked to the size of the interfaces between parts. It can be expressed globally on the graph or locally for a color. Edge-cut should always be kept as low as possible to ensure a low communication overhead in parallel codes (section 4.4.3 will provide more in-depth explanations on this matter). The variation of the maximum local edge-cut is expressed as:

$$\Delta Ec = max\left[Ec(V_L), Ec(V_H)\right]_{\text{after swap}} - max\left[Ec(V_L), Ec(V_H)\right]_{\text{before swap}} . \tag{4.16}$$

  If there is a node pair inducing $\Delta Ec \leq 0$, it should be swapped. If all the possible swaps imply an increase of the edge-cut, the pair with the highest balance gain over edge-cut loss is swapped:

$$(v_H, v_L) = argmax\left(\frac{\frac{D}{2} - \left|d - \frac{D}{2}\right|}{\Delta Ec}\right) . \tag{4.17}$$

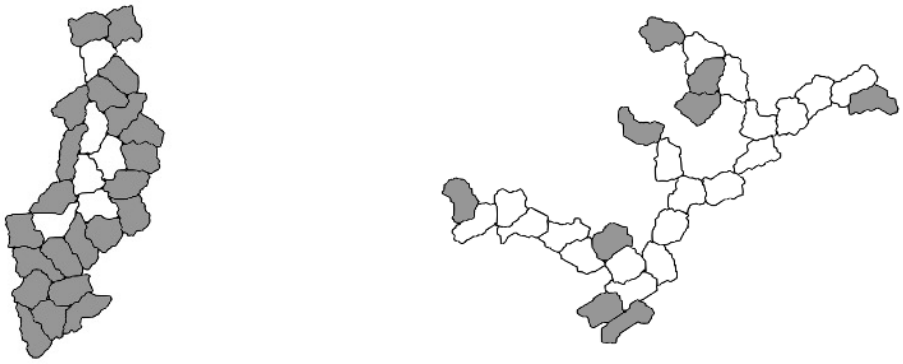
  Section 4.5.2 will demonstrate the benefits of using edge-cut control on a 2D example.

Importance of edge-cut control

The edge-cut of a part can be related to its geometrical shape. Minimizing the edge-cut of a part can actually be interpreted physically as minimizing the capillary force arising form surface tension in liquids and thus driving the part to a spherical shape. Fig. 4.10 represents two parts as an example of good and bad edge-cut.

When edge-cut control is not included in the DOB-EL algorithm, most parts quickly tend to have a tree or filament shape instead of a rather round shape which can create a variety of issues. The most important one in CFD simulation is because of point to point communications that take place at core to core interfaces. Point to point communication cost consists of a constant overhead occurring at the beginning of the communication, referred to as latency, and a variable part that scales with the amount of data to be transferred. An increased edge-cut implies an increased interface size between the cores and therefore more data to be transferred between the cores, and finally an increase in the variable part of the communication cost. Additionally, if a mesh part is very elongated, it is susceptible to interact with an increased number of neighboring cores, increasing also the number of communications to be performed and the latency cost. Impact of edge-cut on performance will not be measured in this work as it is very dependent on the CFD code and the physics that are solved.

During the swapping process, having many swappable sub-parts without breaking the contiguity is a big advantage as it increases greatly the likelihood of finding a good swap. When the parts are elongated, the number of swappable sub-parts decreases greatly due to the presence of filaments. This phenomenon is represented in Fig. 4.10. In these cases, the parts can get stuck in that shape and become inactive in the load balancing process. This happens especially when the load balancing is used dynamically. It should be noted that controlling the edge-cut does not forbid elongated shapes, but helps elongated shapes to recover a rounder shape later on.



(a) Part with low edgecut **(b)** Part with high edgecut

Figure 4.10: Comparison between parts with good and bad edge-cut, sub-parts that could be swapped without breaking contiguity are greyed. For this example edge-cut of part (b) is 2.2 times greater than part (a).

Grain size

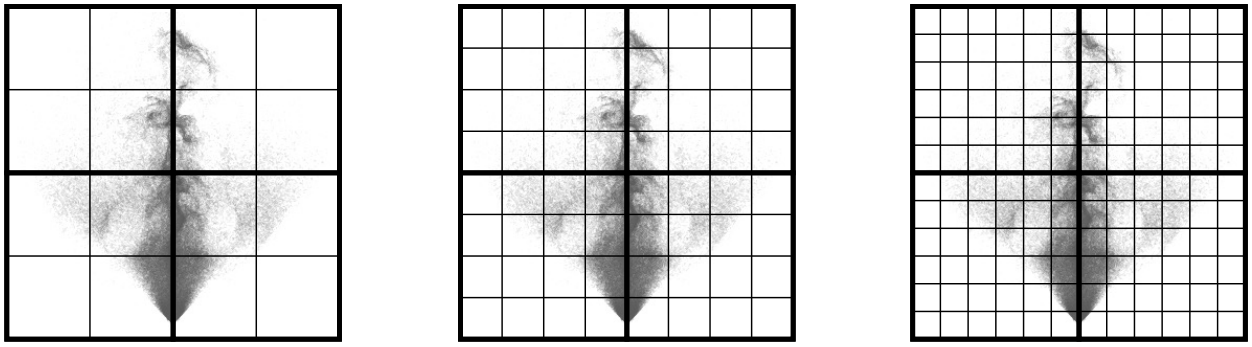
Grain size refers to the smallest unit of load that can be exchanged. In the current context, the grain size corresponds to the size of the sub-parts as shown in Fig. 4.11. The grain size has a major impact on the behavior of the algorithm:

- A high number of sub-parts leads to a better balance but edge-cut and load balancing cost increase.
- Conversely, a low number of sub-parts leads to a lower load balancing cost but has worse and less consistent results.

A detailed analysis of the impact of the number of sub-parts on the performance of DOB-EL is provided in Section 4.5.2 for a 2D test case and in Section 4.6.2 on large-scale cases.

4.5 Implementation and validation

This section gives an overview of the implementation of the load-balancing method. A 2D validation case is used as a pedagogical example to show the behavior of the DOB-EL algorithm.



(a) Large grain size **(b)** Medium grain size **(c)** Small grain size

Figure 4.11: Illustration of double-domain decomposition for different grain sizes. This is a simplified example with a square shape decomposition but sub-parts may have irregular shapes forming no pattern.

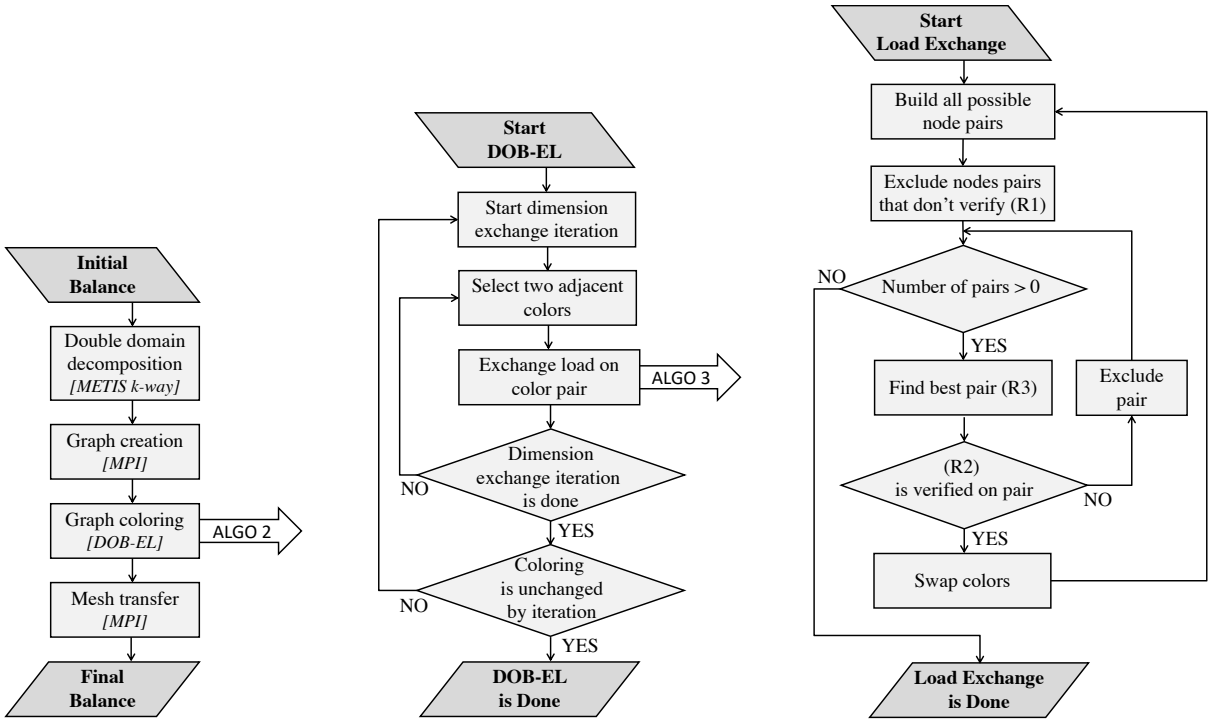
4.5.1 Implementation

The implementation of the load-balancing process is illustrated in Fig. 4.12 and is decomposed into three algorithms going from the largest scope to the smallest.

Algorithm 1 describes the high-level implementation of the method. A double-domain decomposition is performed on each core and the associated local graph is created. This graph is then assembled on a single core, using MPI gathering. This graph is a light weight representation of the sub-part weights and connectivity, as represented in Fig 4.8d. A sequential coloring is performed using DOB-EL, no communication is required during the coloring process. Afterwards, the new coloring is scattered among all cores and mesh parts are exchanged accordingly using non-blocking MPI point-to-point communications. This implementation has the benefit of simplicity and performing the least communications. Its scalability could be questioned because a single core performs the coloring of the full mesh but its cost is low as shown in Sec 4.6.3.

The main stage of the method is depicted in Algorithm 2. The cores are paired two by two sequentially, starting with the most loaded cores and highest load differences, and sub-parts are exchanged on the graph. After all the cores have performed an iteration of dimension exchange, another iteration is to be performed only if a favorable load exchange has happened. As the main objective of the algorithm is to reduce LI_{MAX}, only pairs including a highly loaded core are considered. In the proposed implementation, a core is considered highly loaded if its Lagrangian imbalance exceeds 90% of the current LI_{MAX}.

Algorithm 3 details the swapping process between two parts. All the sub-parts connected to their opposing color by an edge are selected. All the pairs built using these sub-parts are considered. If a pair does not verify (R1) it is rejected. Afterwards, the optimal pair is considered (R3) and contiguity is checked (R2). If contiguity is verified, the swap is performed and Algorithm 3 is repeated. If contiguity is not verified, the pair is rejected and a new optimal pair is searched. When no selectable pairs remain, the load exchange between the two parts ends.



(a) Algorithm 1: Main **(b)** Algorithm 2: DOB-EL **(c)** Algorithm 3: Detailed load exchange

Figure 4.12: Algorithmic implementation.

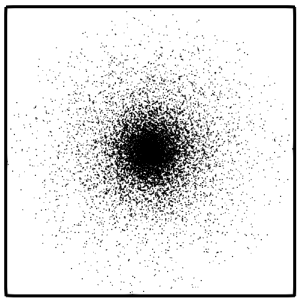
4.5.2 2D validation case

A simple test case is used to give a visual representation of how the algorithm works and to validate the behavior of the Eulerian and Lagrangian balance. The considered domain is a 2D square box in which particles are injected at the center of the domain and spread out in every direction. Fig. 4.13a shows the resulting particle distribution. Particles density decreases as the inverse of the distance to the injection point. The particle distribution is similar to a 3D injector distribution projected on a plane perpendicular to the injection direction.

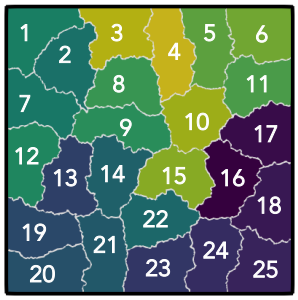
The case is composed of 1,835,664 triangular elements, 13,344 particles and is run on 25 cores.

It must be noticed that the DOB-EL algorithm will produce different results depending on the initial partitioning that is provided. Therefore the following analysis is conducted statistically on a large batch of runs, each run being based on a different initial state. A multitude of distinct initial states is achieved by modifying the seeding of the random number generator in the k-way partitioning algorithm of the METIS library. For the present study, a total of 10,000 runs have been performed.

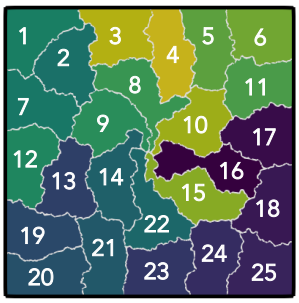
A typical initial partitioning is represented in Fig. 4.13b and the corresponding final partitioning is represented in Fig. 4.13c.



(a) Particle distribution in the 2D square domain (outline)



(b) An initial single constraint Eulerian partitioning



(c) Partitioning after DOB-EL

(d) Link to video of the load balancing process

Figure 4.13: Representation of the 2D case and load balancing, on 25 cores with 40 sub-parts/core.

Eulerian balance

The Eulerian balance is assumed to be ideal in the model by Eq. 4.11. However, in real cases, balance is slightly off due to the error margin of single-constraint partitioning. A maximum imbalance equal to 1.01 is imposed on the k-way partitioning algorithm of the Eulerian mesh. This means that a part can exceed the ideal load by 1% at maximum.

On average, the initial maximum Eulerian imbalance is 1.0092 and it never exceeds the 1.01 threshold as expected. The same threshold is used to create the sub-parts during the second decomposition. As a consequence, the load exchange process is no longer perfectly orthogonal to the Eulerian load as Eq. 4.11 is no longer verified. However, this effect is minimal and will be neglected in the remaining of this study.

In 61% of the 10,000 runs, the maximum Eulerian imbalance remained the same, as the part with the maximum load did not exchange any load. This happens when the part is skipped by the diffusion process because its Lagrangian load is low and not worth diffusing.

In 28% of the cases, the maximum Eulerian imbalance decreased and in 11% of the cases, the maximum Eulerian imbalance increased slightly. The worst recorded increase is 0.23% relative to the initial balance, which remains a negligible increase.

Lagrangian balance

The Lagrangian load is shifted away from the most loaded cores. After DOB-EL is done there is no longer one core with a peak load but instead several cores sharing most of the load, as shown in Fig 4.14. The balance has been improved substantially but is still not ideal because the cores located too far away from the areas with high particle load couldn't receive a part of the load without losing contiguity or deteriorating the edge cut a lot.

Load balancing is performed using four different grain sizes and results are compared to the reference case without load balancing. Fig. 4.15 represents the Probability Density Function (PDF) of the final LI_{MAX} for all cases. The initial LI_{MAX} (reference) is very inconsistent and ranges from 6 to

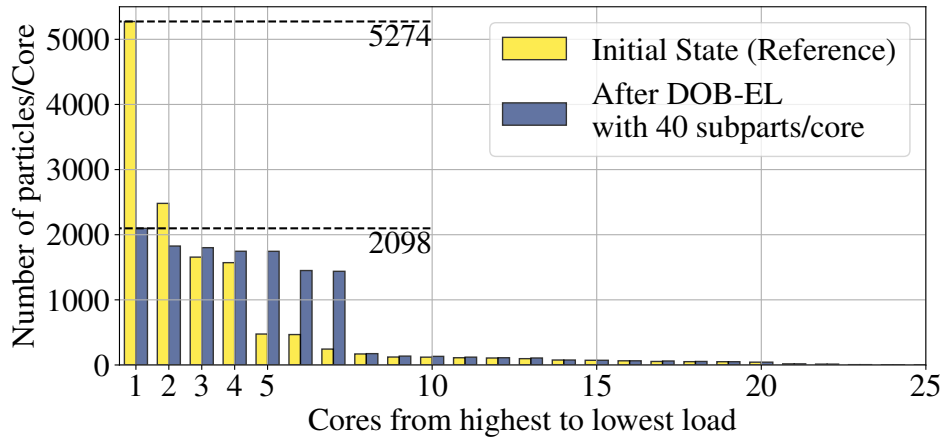


Figure 4.14: Distribution of the Lagrangian load among the cores before and after DOB-EL for a given run.

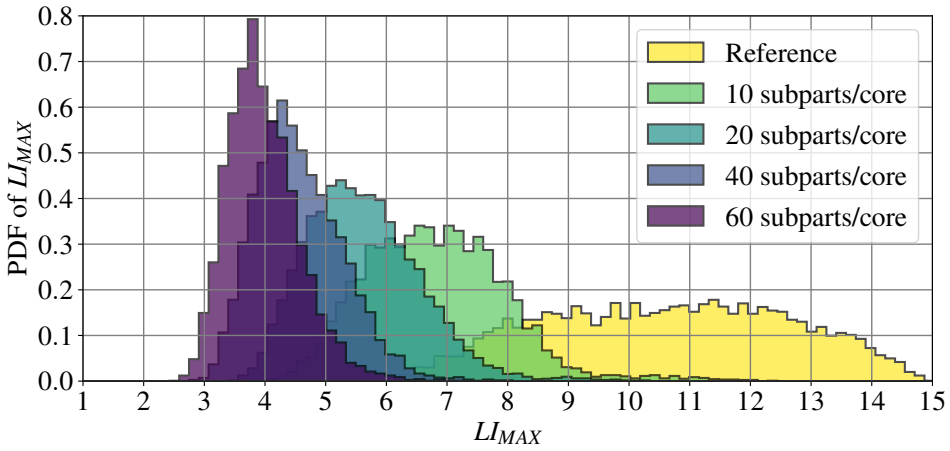


Figure 4.15: PDF of Lagrangian imbalance for different grain sizes.

15. This stresses the need for a load balancing algorithm.

The final LI_{MAX} decreases significantly when load balancing is performed even with large grain size. Reducing grain size leads to smaller imbalance and consistency increases. In very rare cases, the imbalance remains rather high as the algorithm gets stuck early because no more satisfactory swaps can be performed. Using a smaller grain size or reiterating the algorithm with another double-domain decomposition can alleviate this issue.

Edge-cut control

The efficiency of edge-cut control (R3), detailed in section 4.4.3, is demonstrated by comparing the results of the 2D case with edge-cut control and without it. When edge-cut control is enabled the ideal swap is maximising the ratio of load balance improvement over local edge-cut increase.

Without edge-cut control the ideal swap is the one maximising load balance improvement regardless of local edge-cut increase.

Table 4.1 shows by how much the local edge-cut increase with and without (R3) for different numbers of sub-parts/core. It appears that (R3) limits the edge-cut increase by approximately a factor 2. It should also be noted that local edge-cut increases when the number of sub-parts per cores increases.

It is also important to consider the impact of (R3) on the final LI_{MAX}. Results in Table 4.2 indicate that (R3) benefits DOB-EL as a slightly better final load balance is reached with it.

	Number of sub-parts/core			
	20	40	60	80
With edge-cut control	+13.7%	+18.9%	+21.0%	+21.1%
Without edge-cut control	+23.6%	+35.4%	+42.1%	+46.8%

Table 4.1: Influence of edge-cut control on maximum edge-cut increase.

	Number of sub-parts/core			
	20	40	60	80
Final LI_{MAX} with edge-cut control	5.27	4.30	3.95	3.84
Final LI_{MAX} without edge-cut control	5.58	4.93	4.65	4.57

Table 4.2: Influence of edge-cut control on LI_{MAX}.

Comparison to single-constraint Euler-Lagrange approach

When performing Euler-Lagrange load balancing using a single-constraint approach, the simplest approach that can be envisioned is to define a composite weight function as:

$$\omega = (1 - \alpha)\,\omega_e + \alpha\omega_l \qquad \forall\alpha \in [0; 1]\,, \tag{4.18}$$

α is a user-defined variable that describes how much importance is given to the Eulerian weight relative to the Lagrangian weight. An example of partitioning based on this approach is given in Fig 4.16. In regions with high particle density, partition will have an under-average element count, thus covering less space and including fewer particles. Overall, it can be interpreted as a compromise between Eulerian and Lagrangian balance.

A set of 10.000 runs using the single constraint approach have been performed for various $\alpha \in [0; 1]$. The dependence between the Eulerian and Lagrangian imbalance is shown in Fig 4.17. When one imbalance decreases, the other increases. In contrast, for the DOB-EL approach, no dependence exists between the two constraints. Reducing the Lagrangian imbalance does not increase the Eulerian balance, thus orthogonality is achieved.

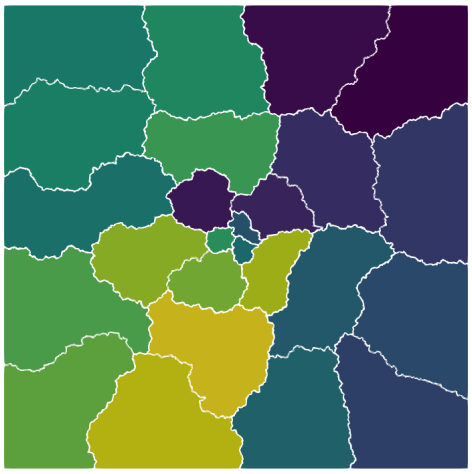


Figure 4.16: Example of single-constraint Euler-Lagrange partitioning.

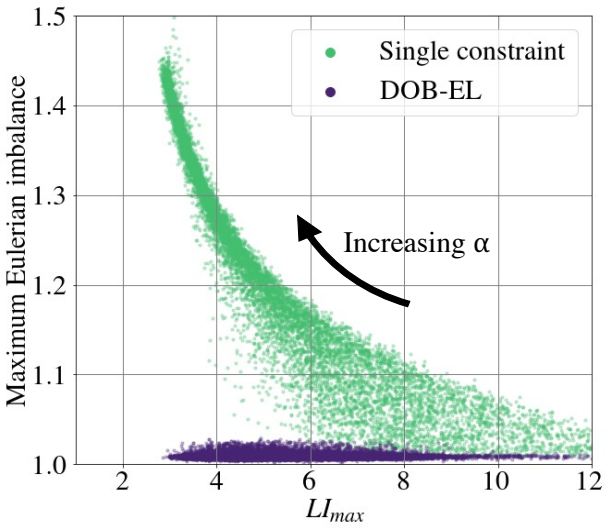


Figure 4.17: Dependency between Eulerian and Lagrangian imbalance for the Euler-Lagrange single-constraint and DOB-EL approach. Each point represents a run.

4.6 Large-scale applications

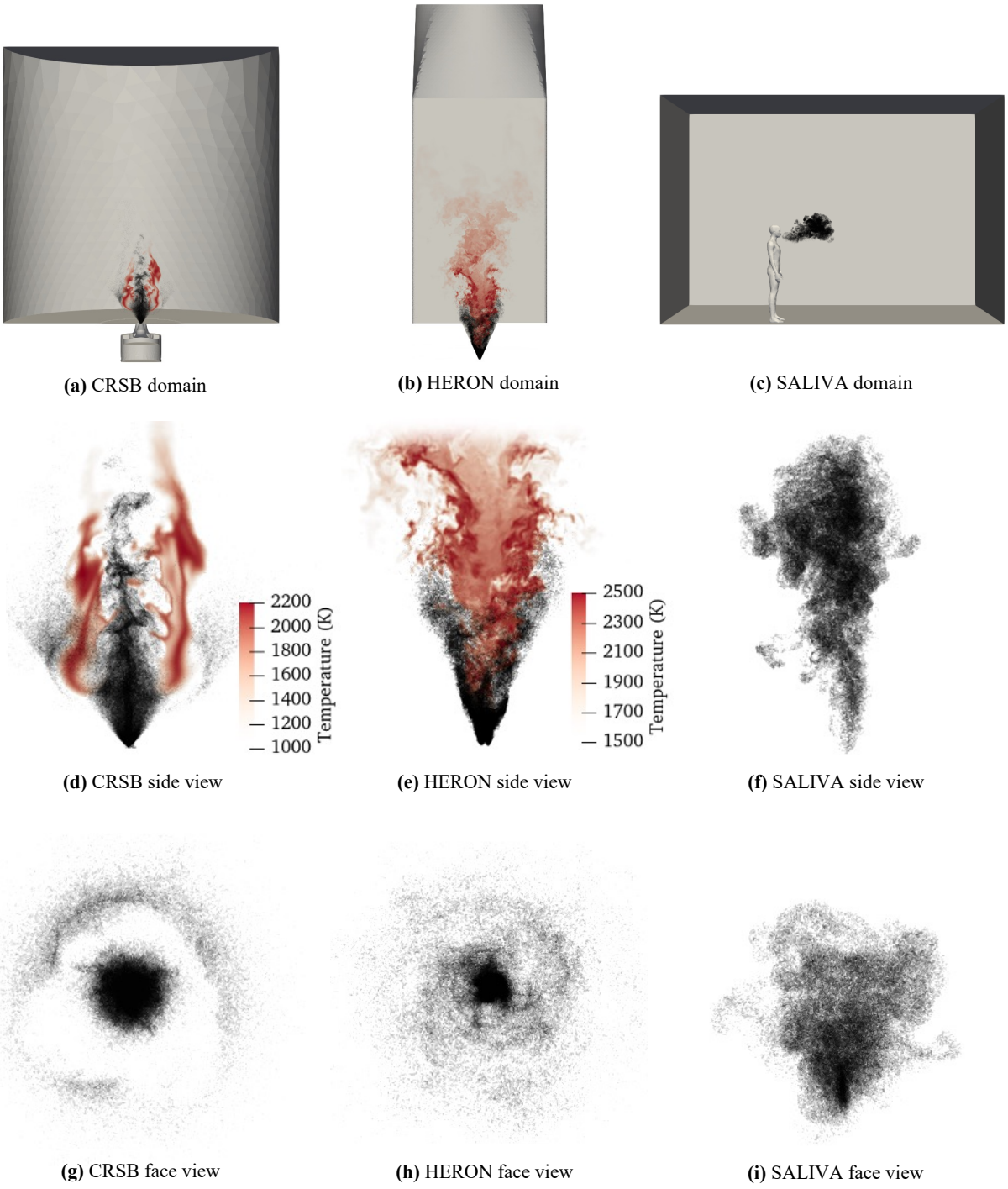
4.6.1 Presentation of the simulation set-ups

Three large scale simulations are considered. These applications share the common aspect of being spray injectors. As spray injector feature very heterogeneous particle distributions, they are well fit to demonstrate the efficiency of the diffusive load-balancing method.

- The CRSB (Coria Rouen Spray Burner) injector is a spray burner. The droplets are injected and are quickly fully evaporated due to the presence of a flame close by. In this set up, droplets are driven by the air co-flow and clump up along the injection center-line. The experimental set-up is detailed in [31] and the associated simulation is available in [96]. Finite-rate chemistry with transported species is used in the Eulerian phase.

- The HERON (High prEssuRe facility for aerO-eNgine combustion) injector is also a spray burner but distinguishes from the CRSB injector due to the droplets inertia being greater and therefore not driven by the co-flow. A cone shaped droplet distribution is observed. The experimental set-up is detailed in [31, 130] and the associated simulation is available in [131]. Tabulated chemistry is used in the Eulerian phase.

- The SALIVA injector replicates saliva droplets in human breath. These droplets are only partially evaporated and stay suspended in the stagnating surrounding air. Their residence time in the domain is greater than for the spray burner droplets. The flow and droplets are gradually slowed down and form a cloud like distribution. The simulation of this set-up is detailed in [132].



(a) CRSB domain **(b)** HERON domain **(c)** SALIVA domain

(d) CRSB side view **(e)** HERON side view **(f)** SALIVA side view

(g) CRSB face view **(h)** HERON face view **(i)** SALIVA face view

Figure 4.18: Spatial particle distributions and flame topology (only for CRSB and HERON cases).

An overview of the spatial particle distributions of these injectors is given in Fig. 4.18 and additional information about the set-ups is given in Tab. 4.3.

	CRSB	HERON	SALIVA
Number of particles/droplets	183k	180k	117k
Number of elements	74M	237M	10M
Number of cores	560	896	384
Number of elements/core	132k	265k	26k
Initial Lagrangian cost / iteration	4.12s	13.59s	1.12s
Eulerian cost / iteration	201.9s	71.3s	3.03s
Lagrangian cost in %	2%	16%	27%
% of cores with particles (of total cores)	84.2%	14.4%	41.3%

Table 4.3: Additional information about the configurations.

4.6.2 Load balancing of an instantaneous distribution

The DOB-EL algorithm is applied in the three configurations described above. Each case is studied for three different load-balancing grain sizes: 10, 20 and 30 sub-parts per core. A reference without any particle load balancing is also considered. A total of 100 runs is performed for each of those sub-cases, using different initial partitionings. All the runs have been performed on the Occigen supercomputer from CINES, France [133].

The LI_{MAX} is measured directly after load balancing and is presented in Tab. 4.4. The LI_{MAX} decreases as the number of sub-parts is increased like for the 2D case. However, most of the decrease is already achieved with 10 sub-parts per core, using 30 sub-parts only improves slightly the LI_{MAX}.

The evolution of LI_{MAX} is a theoretical estimation of the evolution of the Lagrangian performance. The actual Lagrangian performance is assessed based on the Wall Clock Time (WCT) per solver iteration using internal timers of the YALES2 code. The results are displayed in Tab. 4.5. A notable performance gain is achieved, up to -70% for HERON with 30 sub-parts.

The relation between the Lagrangian performance and LI_{MAX} is represented in Fig. 4.19 for all runs. A very clear correlation between performance and LI_{MAX} is observed, meaning that LI_{MAX} is a good performance predictor. This correlation consists of a static overhead and a linearly scaling part. For small values of LI_{MAX}, the overhead is dominant, as it is not improved by the load balancing.

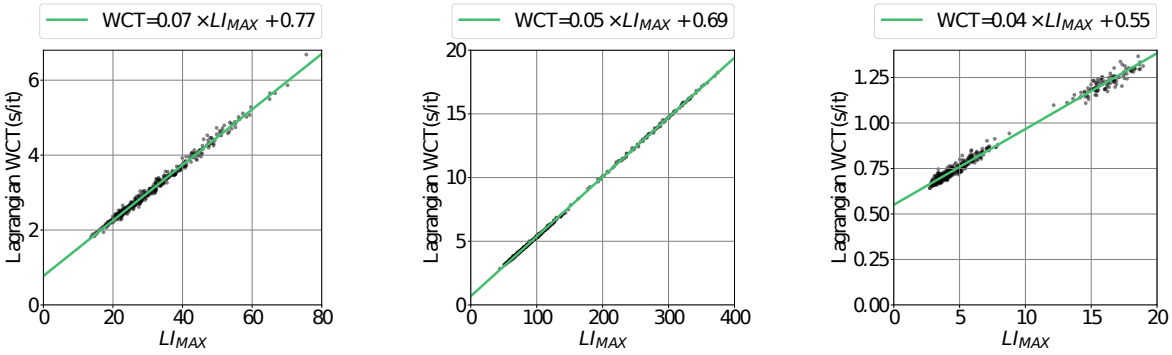
Eulerian balance remains mostly unchanged in all simulations, which is the expected orthogonal behavior. Some slight Eulerian performance variations have been observed. This is due to the fact that the Eulerian constraint assumes that the cost is perfectly proportional to the number of elements, which is not. The edge-cut increase can also increase slightly the Eulerian cost. The maximum edge-cut of a part remained constant for all cases. The global edge-cut increase didn't exceed 1% for the CRSB and HERON and 7% for the SALIVA injector. Eulerian performance variations never exceed 5% of the total Eulerian cost.

	CRSB		HERON		SALIVA	
	$\langle LI_{max} \rangle$	$\sigma\,(LI_{max})$	$\langle LI_{max} \rangle$	$\sigma\,(LI_{max})$	$\langle LI_{max} \rangle$	$\sigma\,(LI_{max})$
Reference	45.1	8.8	275.2	49.0	16.19	1.36
10 sub-parts/core	30.5	8.0	107.7	19.3	5.85	0.87
20 sub-parts/core	27.7	6.9	82.6	15.1	4.17	0.62
30 sub-parts/core	24.7	5.4	73.0	14.3	3.46	0.55

Table 4.4: Mean LI_{max} and associated statistical variance.

	CRSB		HERON		SALIVA	
	$\langle WCT \rangle$	$\sigma\,(WCT)$	$\langle WCT \rangle$	$\sigma\,(WCT)$	$\langle WCT \rangle$	$\sigma\,(WCT)$
Reference	4.12	0.67	13.59	2.28	1.12	0.06
10 sub-parts/core	3.03	0.60	5.70	0.87	0.80	0.05
20 sub-parts/core	2.82	0.50	4.56	0.68	0.72	0.03
30 sub-parts/core	2.60	0.39	4.12	0.64	0.70	0.03

Table 4.5: Lagrangian wall clock time (s) per solver iteration and associated statistical variance.



(a) CRSB **(b)** HERON **(c)** SALIVA

Figure 4.19: Lagrangian performance scaling.

4.6.3 Cost of the load-balancing method

The cost of the load-balance method is crucial as it has to be smaller than the induced performance gain. In order to understand the cost properly, it is split into several parts. The detailed results are given in Tab. 4.6 and a visual aid is given in Fig. 4.20, the cost decomposition is the following:

- The double-domain decomposition cost, using METIS k-way partitioning, increases with the number of elements per core. This is why this cost is low for the SALIVA case and more pronounced for the CRSB and HERON cases. It also increases with the number of created sub-parts. This cost can be fully skipped on codes based on a double-domain decomposition like YALES2, which already provides these sub-parts.
- The graph creation cost scales with the number of cores and number of sub-parts but is negligible for less than a thousand cores.
- The diffusive coloring cost scales with the number of sub-parts because more potential vertex pairs need to be tested to find the optimal swap. Additionally, as smaller parts are exchanged,

more swaps are required in the balancing process. However, it is very dependent on the particle distribution and its cost is likely to increase on rather homogeneous cases like SALIVA. The cost does not appear to be scaling with the number of cores, this is because the algorithm is performed only on highly loaded cores. On very large core counts, the cost could probably be further reduced using a parallel implementation.

- The transfer cost corresponds to the communications performed to exchange the mesh parts. It scales with the number of sub-parts as a better balance is reached and therefore more mesh elements are likely to be transferred.
- The post-processing cost corresponds to the time required to rebuild data structures and connectivity once mesh sub-parts have been transferred. This cost is very code dependent and scales with the number of elements per core. This cost is large but can be mitigated for example if the load balancing is done when data structures are rebuilt anyway, like dynamic adaptive mesh refinement.

Globally, it can be considered that the diffusive load balancing method scales mainly with the number of elements per core and the number of sub-parts. The number of elements per core is supposed to be similar whatever the case is. It also has been shown that the algorithm works well with a small number of sub-parts. Therefore, this algorithm should scale well towards larger simulations.

The cost of the method can be compared to the induced cost reduction of Tab. 4.5. The load balancing is worth it after very few iterations and can therefore be considered as very efficient: it is thus possible to perform the load balancing dynamically during runs if needed.

	CRSB			HERON			SALIVA		
Sub-parts/core	10	20	30	10	20	30	10	20	30
WCT Metis (s)	0.392	0.461	0.534	0.795	0.933	1.074	0.067	0.085	0.104
WCT Graph (s)	0.005	0.006	0.007	0.007	0.008	0.009	0.003	0.003	0.004
WCT coloring (s)	0.003	0.013	0.042	0.006	0.019	0.048	0.027	0.144	0.530
WCT Transfer (s)	0.677	0.817	0.937	1.068	1.097	1.157	0.546	0.876	1.295
WCT Post-processing (s)	1.442	1.437	1.449	3.031	3.008	3.510	0.458	0.467	0.471
WCT TOTAL (s)	2.519	2.734	2.969	4.907	5.065	5.798	1.101	1.578	2.404
Cost of DOB-EL compared to a Lagrangian iteration prior to load-balancing	61.1%	66.3%	72.1%	36.1%	37.2%	42.7%	98.2%	140.9%	214.6%
Number of iterations to compensate DOB-EL cost	2.31	2.10	1.95	0.62	0.56	0.61	3.44	3.95	5.72

Table 4.6: Detailed load balancing cost for large-scale cases and break-even iteration.

4.6.4 Dynamic load balancing

Load balancing is done on an instantaneous particle distribution. This distribution can change over time as particles are moving and balance may deteriorate. When this happens, load balancing needs

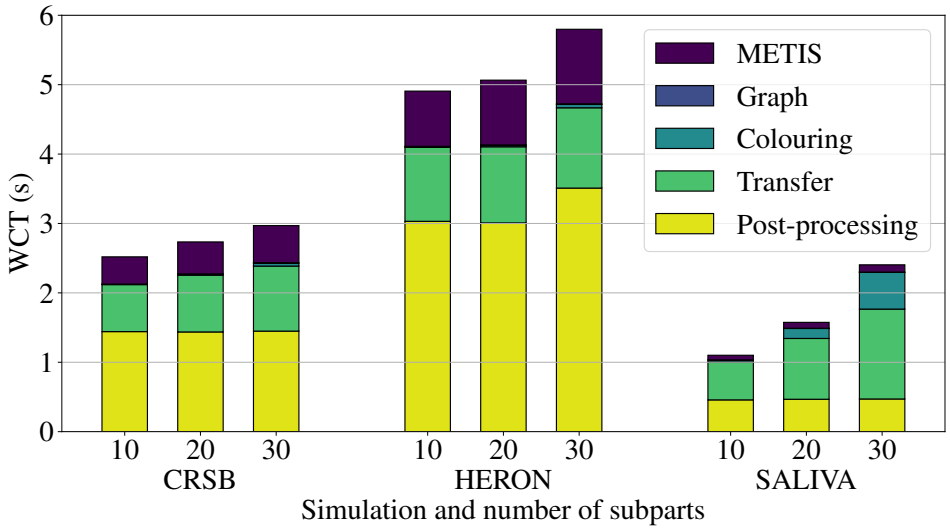
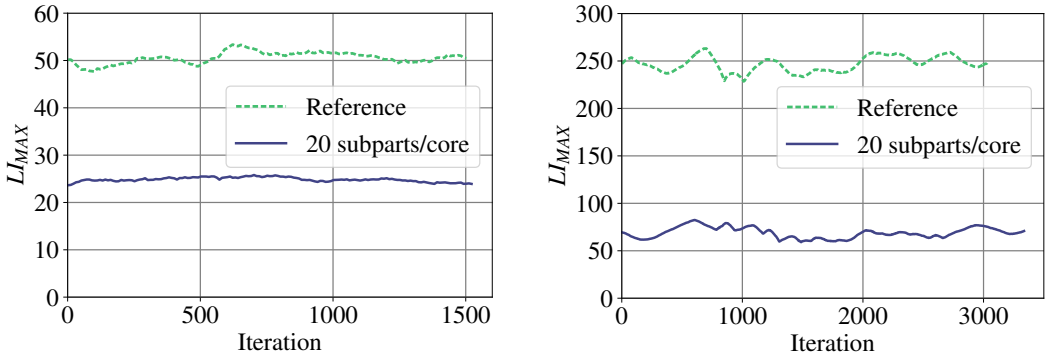


Figure 4.20: Load balancing cost, visual representation of Tab. 4.6.

to be performed again. The three considered cases fall into two categories: steady and unsteady particle distributions.

The particle distributions of CRSB and HERON are steady once the flame is established and don't require repeated load balancing. Fig. 4.21 shows the evolution of LI_{MAX} during a 10 hour run, starting from the distribution shown in Fig. 4.18. Small imbalance changes are caused by turbulent structures clumping particle together in some areas. This makes the load balancing cost negligible as it is only performed once at the beginning of the run.

The particle distribution of SALIVA changes as the cloud move away from the subject, thus load-balance and performance deteriorate over time. Frequent DOB-EL calls are required to restore a good load-balance. Load-balancing is triggered whenever LI_{MAX} exceeds a maximum threshold that is user-set. Additionally, Adaptive Mesh Refinement (AMR) is performed dynamically with Mmg [134, 135] based on an error estimator of the mean flow field and vorticity [136]. The AMR process produces a new unbalanced partitioning that requires a DOB-EL call as follow-up. Fig. 4.22 shows the evolution of LI_{MAX} during a 1 hour run, starting from the distribution shown in Fig. 4.18. Average behavior has been assessed on 10 runs, LI_{MAX} is reduced by 69% and Lagrangian WCT by 33%. The cost of the load balancing represents less than 5% of the observed gain. LI_{MAX} is not as good on average as in the instantaneous case. This is because imbalance increase over time after load balancing, resulting in a higher imbalance on average. The relation between LI_{MAX} and the WCT remains like the linear regression obtained in Fig. 4.19.



(a) CRSB **(b)** HERON

Figure 4.21: Evolution of imbalance over time with load balancing at iteration 0.

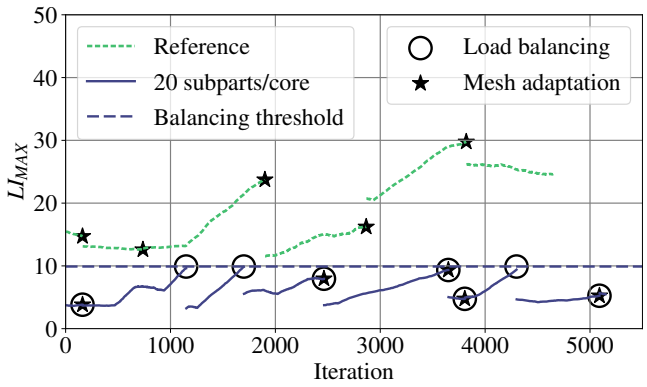


Figure 4.22: Evolution of imbalance over time, on the SALIVA case, with dynamic mesh adaptation and dynamic load balancing.

4.7 Conclusion

This chapter focused on investigating Euler-Lagrange load balancing, specifically addressing the challenge of double-constrained load balancing with the introduction of a new method, DOB-EL. This method effectively optimizes Lagrangian load distribution without compromising the Eulerian load balancing. The effectiveness of DOB-EL was demonstrated on three large-scale spray simulations, where it achieved significant improvements in computational efficiency. Depending on the specific case, DOB-EL reduced the computational cost of the Lagrangian solver by 35% to 70% through improved load distribution.

CHAPTER 5

Adaptive mesh refinement for reactive flows

Adaptive Mesh Refinement (AMR) is introduced, with a discussion on various criteria used to identify regions for refinement. A novel criterion based on scale similarity of resolved reaction rates is presented and validated. A dynamic AMR strategy is then applied to the CRSB to enhance simulation accuracy and efficiency.

Published in Proceedings of the Combustion Institute (2024) [137].

Contents

5.1 Meshes

Mesh resolution plays a crucial role in LES, as it is directly related to the cut-off length between the resolved and modeled scales of the flow. Turbulent combustion simulations are highly challenging due to the thinness and dynamics of reaction zones. These reactive layers result in sharp spatial and temporal temperature, species, and density gradients as shown in [138].

Meshes can be classified into two categories, structured and unstructured meshes represented in Fig. 5.1 and 5.2.

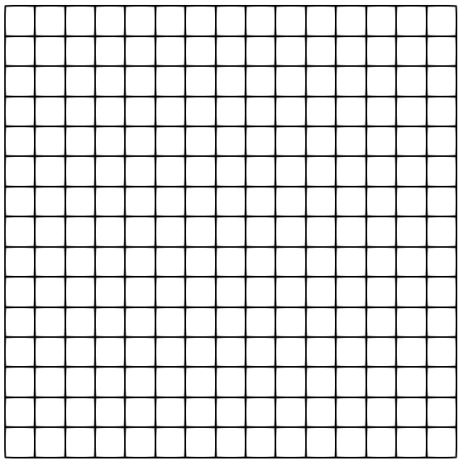


Figure 5.1: 2D structured mesh.

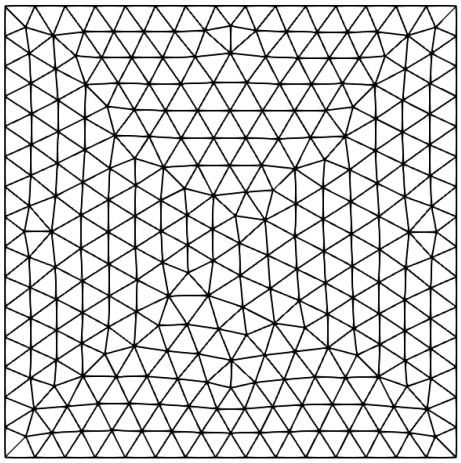
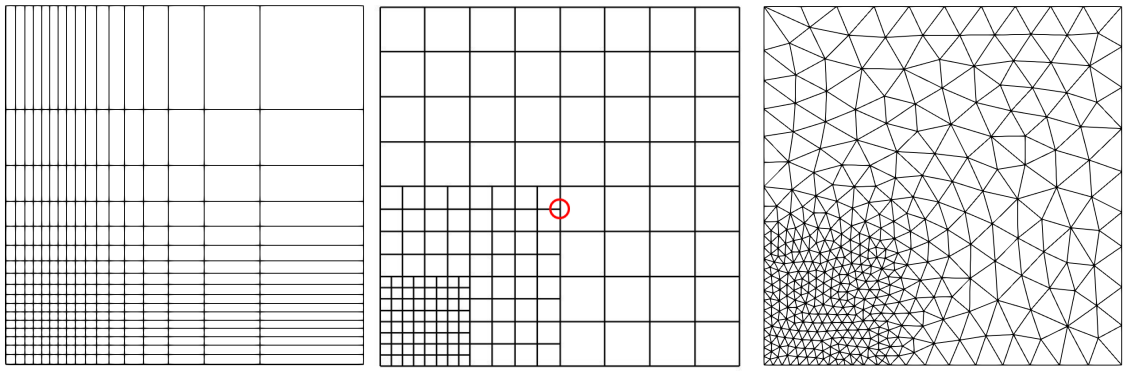


Figure 5.2: 2D unstructured mesh.

Structured grids are composed of quadrilateral in 2D and hexahedra in 3D. They benefit of regular connectivity where each grid point is connected to its neighboring points in a predictable manner. This makes them easy to generate, manage and allows a simpler implementation of numerical schemes. However, they are not suited for complex geometries. Multi-block structured meshes allow to represent moderately complex geometries by associating several structured patches but their generation become increasingly complex. Local refinement is challenging as well for structured meshes due to the regular connectivity, resulting in hanging nodes and high aspect ratio elements. This is shown in Fig. 5.3a where a high resolution is applied to the lower left corner. An alternative method is the use of octree meshes as shown in Fig 5.3b. Octree meshes use a hierarchical tree structure, based on patches or directly on elements, to recursively divide the computational domain into smaller cubic or rectangular cells. This allows to refine without the presence of hanging nodes but introduces hanging nodes that exists on an element face where the refinement levels of adjacent elements differ. Transition between different element length scales is sudden and can lead to modeling issues in the LES framework, where a smoothly varying filter width is assumed.

Unstructured grids use a wide variety of shapes, ranging from polygonal structures of Voronoï meshes to a combination of various elements of different nature, such as prism layers and simplicial elements such as tetrahedra. They allow to mesh complex topologies and provide gradual cell size transitions. Connectivity is no longer regular and needs to be stored. Degenerate elements like flat triangles can be generated and need to be avoided to ensure numeric stability and accuracy.



(a) Structured **(b)** Octree **(c)** Unstructured

Figure 5.3: Refinement on different mesh type, the region of interest is the lower left corner.

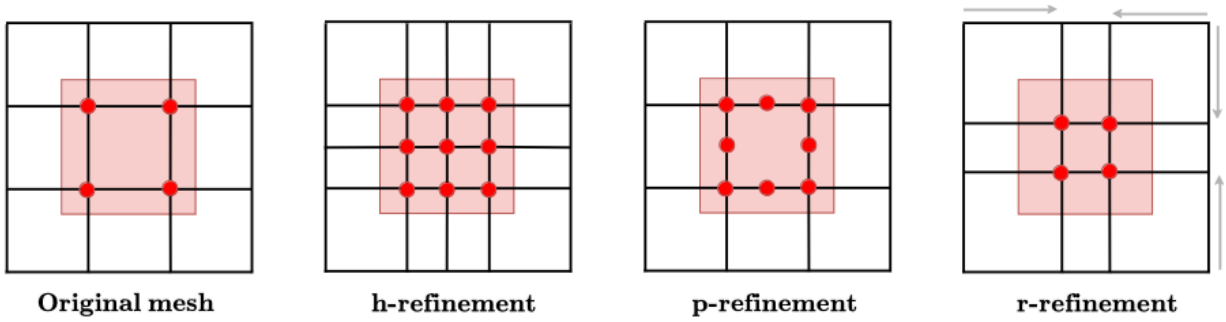


Figure 5.4: Types of mesh adaptation. Reproduction from [139].

To improve the discretization of the grid for the numerical solution of a set of equations, several methods have been developed. Fig. 5.4 gives a visual representation of these approaches:

- The h-refinement approach relies on the prescription of the local cell-size h to improve the discretization of the grid. The overall number of elements is therefore increased. When the refinement is homogeneous, this approach dramatically increases the CPU cost of the simulation. Nowadays, this method aims at refining regions where the solution varies rapidly or where high values of any quantity of interest are found while coarsening areas where the solution is smooth.

- The p-refinement approach relies on an increase of the polynomial order used to represent the solution. Once again the order can be tuned based on the local values of the quantity of interest. Note that it is possible to combine these two latter approaches in the so-called hp-refinement.

- The r-refinement approach relies on a modification of the local discretization by a displacement of the nodes, thus keeping the number of elements constant. In this approach, the connectivity between the cells is never rebuilt. This approach is strongly constrained by the initial mesh.

The h-refinement approach is considered in this thesis. p-refinement is under development in the YALES2 community and r-refinement has been implemented in [140].

5.2 Mesh adaptation

5.2.1 Metric

A metric is a measure of a distance in a so-called metric space. The idea of continuous metric, which transforms the mesh from the real space to a metric space in which the distance between the nodes is the closest to one as possible, is due to [141]. This concept is very strong as it enables to perform mesh adaptation within a solid mathematical framework. It can be extended to an arbitrary number of dimensions as demonstrated for 4D space-time adaptation [142].
Let E be a vector of $\mathbb{R}^n$, the function $d : E \times E$ is called metric over $\mathbb{R}^n$ if:

- $d(P,Q) \geq 0$ for all P,Q of E.
- $d(P,Q) = 0$ only if $P = Q$.
- $d(P,Q) = d(Q,P)$ for all P,Q of E.
- $d(P,Q) \leq d(P,O) + d(O,Q)$ for all P,Q,O of E.

d corresponds to the shortest path between two points. The norm of the scalar product of a vector $\overrightarrow{PQ}$ gives the Eulerian distance. In $\mathbb{R}^3$:

$$d(P,Q) = \langle \overrightarrow{PQ}, \overrightarrow{PQ} \rangle^{\frac{1}{2}} = \|\overrightarrow{PQ}\| = \sqrt{(x_P - x_Q)^2 + (y_P - y_Q)^2 + (z_P - z_Q)^2}\,. \tag{5.1}$$

In $\mathbb{R}^n$, let the metric tensor $\mathcal{M}$ be a symmetric positive definite $n \times n$ matrix. In 3D:

$$\mathcal{M} = \begin{pmatrix} a & b & c \\ b & d & e \\ c & e & f \end{pmatrix}\,. \tag{5.2}$$

Three components are used to define the metric length and three components are required to define metric orientation. The components can be isolated by diagonalising $\mathcal{M}$. Eigenvalues correspond to the metric lengths and eigenvectors to the orientation. A vector can be expressed in the metric space $d_{\mathcal{M}}$, i.e. $\mathcal{M}$ can be projected onto a vector:

$$d_{\mathcal{M}}(P,Q) = \langle \overrightarrow{PQ}, \overrightarrow{PQ} \rangle_{\mathcal{M}}^{\frac{1}{2}} = \sqrt{\overrightarrow{PQ}^T \mathcal{M} \overrightarrow{PQ}}\,. \tag{5.3}$$

The current anisotropic metric tensor of a simplex of a mesh can be obtained by solving:

$$(P_i - P_j)^T \mathcal{M} (P_i - P_j) = 1 \quad \text{for} \quad 1 \leq i < j \leq n_v\,, \tag{5.4}$$

with P_i and P_j the n_v vortices of the simplex. The metric tensor $\mathcal{M}$ allows to describe anisotropic meshes. For isotropic meshes, a single metric length component is necessary and the metric is invariant by rotation requiring no orientation component. Thus, the metric tensor can be reduced to a scalar m for isotropic approaches, which is the case in this work. The isotropic scalar metric is obtained based of the invariance of the determinant of $\mathcal{M}$.

$$h = det(\mathcal{M})^{\frac{1}{n}}\,. \tag{5.5}$$

The determinant being the n dimensional equivalent of the notion of area in 2D or volume in 3D.

5.2.2 Mesh adaptation in YALES2

Once a metric has been defined, a mesh can be generated based on it. The mesh should reproduce the metric as accurately as possible. As a first step, the metric is truncated to make it usable in a CFD context:

- **Metric truncation by bounding:** A minimum and maximum metric value is imposed and the metric field is truncated accordingly. The minimum value truncation avoids generation of very small mesh elements that would impose a prohibitively small time step due to CFL and Fourier conditions. The maximum value truncation avoids large elements with high discretization errors. In the isotropic formalism, this operation is written:

$$h_{truncated} = max\left(h_{min}, min\left(h, h_{max}\right)\right) . \tag{5.6}$$

- **Metric truncation by metric gradient:** Imposing a maximum metric gradient ensures a smooth and slow evolution of the metric field. On a discrete mesh, this allows to control the element growth rate. This is important to reduce numerical error on spatial operators, prevent generation of very skewed elements and avoid high commutation errors. In the isotropic formalism, the gradient can be computed between two points as:

$$h_{grad,PQ} = \frac{|h_P - h_Q|}{d(P,Q)} , \tag{5.7}$$

  if $h_{grad,MAX} < h_{grad,PQ}$, the metric is adjusted. Usually the point with the smallest metric is considered to be the most restrictive and its metric is propagated:

$$h_{Q,truncated} = h_P + d(P,Q)h_{grad,MAX} \quad \text{with} \quad h_P < h_Q . \tag{5.8}$$

  This propagation is performed iteratively until the gradient condition is satisfied the whole metric field. The $h_{grad,MAX}$ used in YALES2 is 0.3 based on various numerical experiments.

Quality of a mesh is measured based on two criteria:

- **Element skewness** is measured as:

$$Sk = 0 \leq \frac{V_{reg} - V_{tet}}{V_{reg}} \leq 1 . \tag{5.9}$$

  When equal to 0, the element is ideal and when equal to 1, its volume is zero. Very skewed elements can lead to high numerical errors and should be avoided.

- **Metric error** compares the imposed targeted metric and the mesh metric. When the mesh is refined:

$$E_{refine} = \frac{h_{mesh} - h_{target}}{h_{target}} . \tag{5.10}$$

When the mesh is coarsened:

$$E_{coarsen} = \frac{h_{target} - h_{mesh}}{h_{mesh}} \,. \tag{5.11}$$

This error should be kept under a certain threshold to ensure good agreement between the mesh and the imposed metric.

The mesh adaptation process can be decomposed into a sequential part and a parallel part. Within a process, the mesh is adapted sequentially by using Mmg [134, 135]. This mesh adaptation process relies on four local operations [134] that can be simplified as:

- Edge split, to enrich an undersampled mesh by adding new vertices.
- Edge collapse, to remove vertices from an oversampled mesh.
- Edge swap, to decrease element skewness.
- Vertex movement, to decrease element skewness and metric error.

The mesh returned by Mmg ensures that for all edges of the grid:

$$\frac{1}{\sqrt{2}} \leq d_{\mathcal{M}}\left(P, Q\right) \leq \sqrt{2} \,. \tag{5.12}$$

Parallel mesh adaptation poses two main issues. First, when strong refinement is performed locally in a processor, the number of mesh elements increases drastically and can lead to out of memory errors. Second, vertices located at the interface of two processes are duplicated. The same mesh adaptation operations need to be applied to these points to keep consistancy between the mesh parts. As a consequence, these nodes are not adapted by Mmg. Both of these issues are solved by mesh re-partitioning. This new mesh partitioning is obtained through load-balancing providing an even distribution of the mesh elements. At the same time, an objective function is minimized to ensure that most interface nodes do not remain at the interface in the new partitioning and can be adapted. Sequential mesh adaptation and load-balancing is performed alternatively until metric error and skewness are small enough. This procedure is further detailed in [135] and Fig 5.5.
Using this strategy, a viable mesh can be obtained in 5 to 10 cycles. Reducing the number of cycles is key to reduce mesh refinement overheads. This is achieved by performing a parallel edge split step before the above described procedure. Whenever an edge verifies:

$$\sqrt{2} < d_{\mathcal{M}}\left(P, Q\right) \,, \tag{5.13}$$

it is split by adding a point at the edge center. This operation is easy to parallelize if edges at the block interfaces are split consistantly. However, this operation on its own is not sufficient as it can generate very skewed elements, high element growth rates and is unable to coarsen the mesh. Afterwards, the Mmg and re-partitioning steps are performed, but with this improved starting mesh, it is able to converge in only 1 to 3 cycles on average effectively cutting adaptation costs.

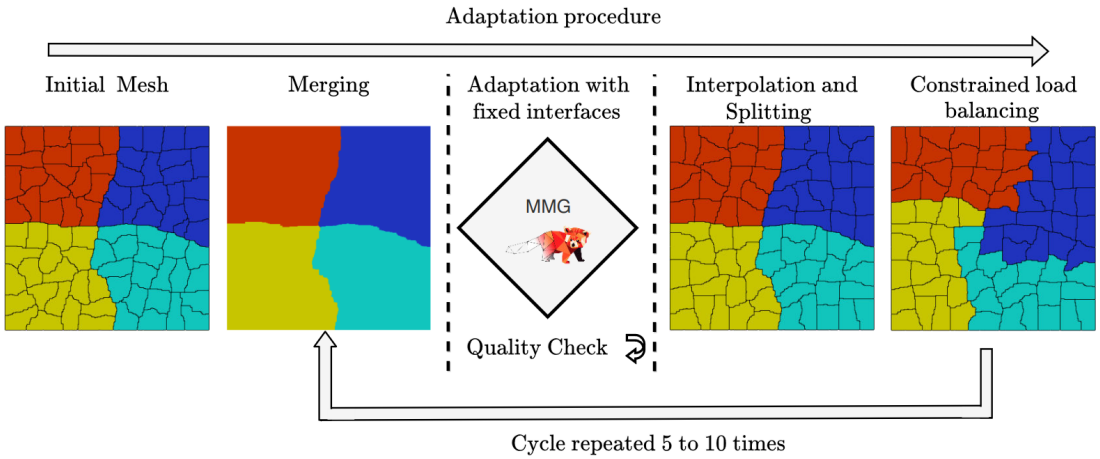


Figure 5.5: Mesh adaptation algorithm. Reproduction from [135].

5.3 Mesh refinement criteria

Criteria that enable to define a continuous metric from a flow solution can be categorized into two types: feature-based and error estimators. Feature-based criteria involve identifying particular flow features and characteristics that one aims at resolving accurately. On the other hand, error estimator-based criteria focus on quantifying the numerical error or uncertainty in the computational solution to obtain an optimal mesh. These criteria frequently use mathematical methods to assess errors in solution variables such as residues of governing equations or interpolation errors.

5.3.1 Gradient based

The gradient is the most commonly used feature-based criterion. High gradients often indicate the presence of significant flow features such as shocks, vortices, or combustion fronts. A metric Δh_{target} is derived to ensure a certain number of computational points N_p are used when transitioning from the minimum to maximum values of the considered scalar. This avoids abrupt changes that can result in high errors. For reactive flows, temperature or any other energy is frequently used. Species mass fractions can be used as well. The criterion can be written for the sensible enthalpy H_s as:

$$\Delta h_{\text{target}} \|\nabla H_s\| = \frac{\max(H_s) - \min(H_s)}{N_p}. \tag{5.14}$$

5.3.2 Hessian based

Hessian-based criteria have been examined in the context of non-reactive flows [136], but with limited application to reactive flows. The Hessian is particular interesting in flames because it identifies well the highly non-linear regions of the considered scalar profiles. Moreover, the Hessian is linked to interpolation errors [143] and can be also be used as an error estimator. Once the Hessian of the considered scalar, the sensible enthalpy for instance, is computed $H = \nabla^2 H_s$, tensor invariants have to be used to derive a consistent metric that is Galilean invariant. Here, eigenvalues are obtained by diagonalizing $H = PDP^{-1}$, where $D = \{\lambda_1, \lambda_2, \lambda_3\}$ is the diagonal matrix of eigenvalues. The largest eigenvalue $\lambda_{\max}$ is the dominant curvature of the scalar field and

is used to derive the metric following the same formalism as for the gradient criterion:

$$\Delta h^2 \lambda_{\max} = \frac{\max\left(H_s\right) - \min\left(H_s\right)}{N_p} \,. \tag{5.15}$$

Other tensor invariants could be considered. For instance, the trace is computationally more efficient as it does not require matrix inversion, $\mathrm{Tr}(H) = \sum_i H_{ii}$, but summing the eigenvalues reduces the detection of high-curvature regions.

5.3.3 Combustion model based

Turbulent combustion models can serve as effective indicators of insufficient grid resolution. For premixed flames, the Thickened Flame model (TFLES) has been used as an error-estimator [144]. To address multi-regime combustion, it is necessary to use a more general combustion model. In this study, the Scale Similarity Resolved Reaction Rate model (SSRRRM) is considered.

Scale Similarity Resolved Reaction Rate Model

The SSRRRM for combustion LES was first introduced by [47] and further validated by [48, 49]. This model is a direct closure approach. When solving the filtered LES transport equations, chemical source terms $\overline{\dot{\omega}\left(\Phi\right)}$ need to be evaluated. However, only the implicitly filtered variables $\widetilde{\Phi}$ are available on the grid due to the LES filtering and the finite-volume integration. Thus, $\overline{\dot{\omega}\left(\Phi\right)}$ can be decomposed into resolved and sub-grid scale contributions:

$$\overline{\dot{\omega}\left(\Phi\right)} = \underbrace{\dot{\omega}\left(\widetilde{\Phi}\right)}_{\dot{\omega}_{\text{Resolved}}} + \underbrace{\overline{\dot{\omega}\left(\Phi\right)} - \dot{\omega}\left(\widetilde{\Phi}\right)}_{\dot{\omega}_{\text{SGS}}} \,. \tag{5.16}$$

To estimate the sub-grid scale term, a test filter is introduced analogously to the dynamic Smagorinsky model [40] or to the dynamic wrinkling factor model [145]. In the original approach of SSRRRM, the second filtering step has the same filter size as the implicit filtering. However, the use of a different test filter width is a viable option, sometimes referred to as model C. The test filter is noted $\widehat{\cdot}$ and leads to:

$$\widehat{\overline{\dot{\omega}\left(\Phi\right)}} = \dot{\omega}\left(\widehat{\widetilde{\Phi}}\right) + \underbrace{\widehat{\dot{\omega}\left(\widetilde{\Phi}\right)} - \dot{\omega}\left(\widehat{\widetilde{\Phi}}\right)}_{\mathcal{L}_{\text{SGS}}} + \widehat{\dot{\omega}_{\text{SGS}}} \,, \tag{5.17}$$

where $\mathcal{L}_{\text{SGS}}$ can be evaluated. Based on the scale-similarity hypothesis: $\dot{\omega}_{\text{SGS}} = K\mathcal{L}_{\text{SGS}}$ with K a constant, that can be evaluated based on the ratio between the filter width of the implicit and test filters.

SSRRRM metric

In order to derive a metric Δh, $\dot{\omega}_{\text{SGS}}$ needs to be formulated as a function of the metric itself. Filtering can be expressed with truncated expansions [146]:

$$\overline{\dot{\omega}\left(\Phi\right)} \simeq \dot{\omega}\left(\Phi\right) + \mathcal{M}_2 \Delta^2 \nabla \cdot \nabla \dot{\omega}\left(\Phi\right) \,, \tag{5.18}$$

where $\mathcal{M}_2$ is the second-order moment of the filter and Δ the filter width.

$$\dot{\omega}\left(\widetilde{\Phi}\right) \simeq \dot{\omega}\left(\Phi + \mathcal{M}_2\Delta^2\nabla\cdot\nabla\Phi\right) , \tag{5.19}$$

which can be linearized as:

$$\dot{\omega}\left(\widetilde{\Phi}\right) \simeq \dot{\omega}\left(\Phi\right) + \mathcal{M}_2\Delta^2\frac{\partial\dot{\omega}}{\partial\Phi}\cdot\nabla\cdot\nabla\Phi\,. \tag{5.20}$$

Inserting Eq. 5.18 and Eq. 5.20 into the expression of $\dot{\omega}_{\mathrm{SGS}}$ in Eq. 5.16, one obtains that $\dot{\omega}_{\mathrm{SGS}}$ scales with Δ^2:

$$\begin{aligned}\dot{\omega}_{\mathrm{SGS}} &\simeq \mathcal{M}_2\Delta^2\left(\nabla\cdot\nabla\dot{\omega} - \frac{\partial\dot{\omega}}{\partial\Phi}\nabla\cdot\nabla\Phi\right)\\ &\simeq \Delta^2 f\left(\Phi\right) .\end{aligned} \tag{5.21}$$

Assuming that the metric Δh is directly proportional to Δ, the target metric is defined to homogenize $\dot{\omega}_{SGS}$ spatially $\dot{\omega}_{\mathrm{SGS,target}} = \Delta h^2_{\mathrm{target}} f\left(\Phi\right)$. Then, following Eq. 5.21, the target metric $\Delta h_{\mathrm{target}}$ can be expressed from the current metric Δh:

$$f\left(\Phi\right) = \frac{\dot{\omega}_{SGS}}{\Delta h^2} = \frac{\dot{\omega}_{SGS,target}}{\Delta h^2_{target}}\,. \tag{5.22}$$

$$\begin{aligned}\Delta h_{\mathrm{target}} &= \Delta h\left(\frac{\dot{\omega}_{\mathrm{SGS,target}}}{\dot{\omega}_{\mathrm{SGS}}}\right)^{\frac{1}{2}} ,\\ &= \Delta h\left(\frac{\dot{\omega}_{\mathrm{SGS,target}}}{K\mathcal{L}_{\mathrm{SGS}}}\right)^{\frac{1}{2}} ,\\ &= \Delta h\left(\frac{\dot{\omega}_{\mathrm{SGS,target}}}{K\left(\widehat{\dot{\omega}\left(\widetilde{\Phi}\right)} - \dot{\omega}\left(\widehat{\widetilde{\Phi}}\right)\right)}\right)^{\frac{1}{2}} .\end{aligned} \tag{5.23}$$

The homogeneous SGS source term $\dot{\omega}_{\mathrm{SGS,target}}$ may be set as a fraction of the maximum resolved reaction rate or adjusted iteratively to impose a prescribed number of grid elements. Eq. 5.23 is valid for any source term. For the composition, this criterion is:

$$\Delta h_{\mathrm{target}} = \Delta h\left(\frac{\sum_{k=1}^{N_{\mathrm{species}}}\dot{\omega}_{\mathrm{k,SGS,target}}}{K\sum_{k=1}^{N_{\mathrm{species}}}\mathcal{L}_{\mathrm{k,SGS}}}\right)^{\frac{1}{2}} . \tag{5.24}$$

In practice, this metric has to be bounded in order to control the minimum and maximum cell size:

$$\Delta h_{\mathrm{min}} < \Delta h_{\mathrm{target}} < \Delta h_{\mathrm{max}}\,. \tag{5.25}$$

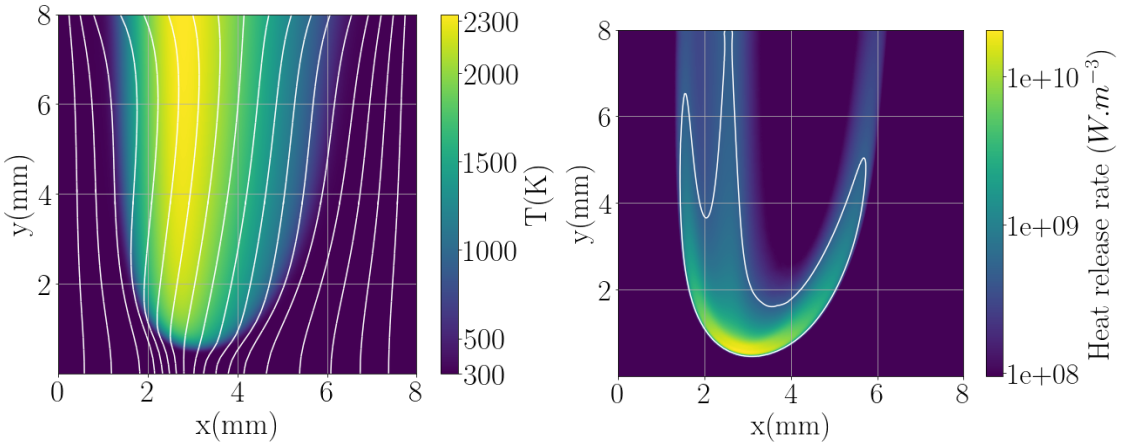
In the most reactive regions, prohibitively small cells could be imposed by the unbounded criterion.

$\Delta h_{\min}$ has be set according to available computational power or desired accuracy. In non-reactive regions, cell size will tend to infinity, $\Delta h_{\max}$ should be set according to the cell size requirements of the non-reactive flow.

5.4 Validation

5.4.1 Simulation set-up

A H_2-Air triple flame [147, 148] in standard conditions is used as reference and demonstration case. It is solved using the San Diego mechanism [149], which counts 21 species and 64 reactions for H_2-air combustion with nitrogen chemistry. Inlet velocity is uniform and equal to $1 m.s^{-1}$ and air-fuel equivalence ratio ranges from 0 to 24 to have non flammable conditions on the sides. This set-up is chosen as it spans a large region in the phase space with lean-premixed, rich-premixed and diffusive flame regimes altogether, which is highly challenging for feature-based adaptation criteria. Dynamic mesh refinement is performed on-the-fly and in parallel with the MMG library [134, 135]. Once the calculation has reached a steady state, the mesh becomes static.



(a) Temperature field in the full simulation domain. Velocity streamlines are represented in white.

(b) Heat release rate (HRR) field in the full simulation domain with a HRR contour line to materialise the flame branches.

Figure 5.6: 2D H_2-Air Triple flame. H2 is injected on the right, air on the left.

5.4.2 Results

Adapted meshes for various cell counts are shown in Fig. 5.7. The only constraint added to the adaptation criteria is the minimum cell size $\Delta h_{\min}$ set to 20 micrometers. Adapted meshes considering the different criteria are very different, illustrating the challenge of deriving feature-based adaptation criteria in multi-regime combustion. Gradient-based criteria only capture the exterior premixed flames branches, while the Hessian-based criteria also capture the internal diffusion flame, but still favour the exterior premixed branches. The SSRRRM criterion leads to a fine resolution of the full diffusion flame and only the base of the premixed flames.
To compare the adapted meshes to a reference mesh, a very fine mesh with an homogeneous 20 micrometer cell size is computed. This resolution enables to resolve all the species source term

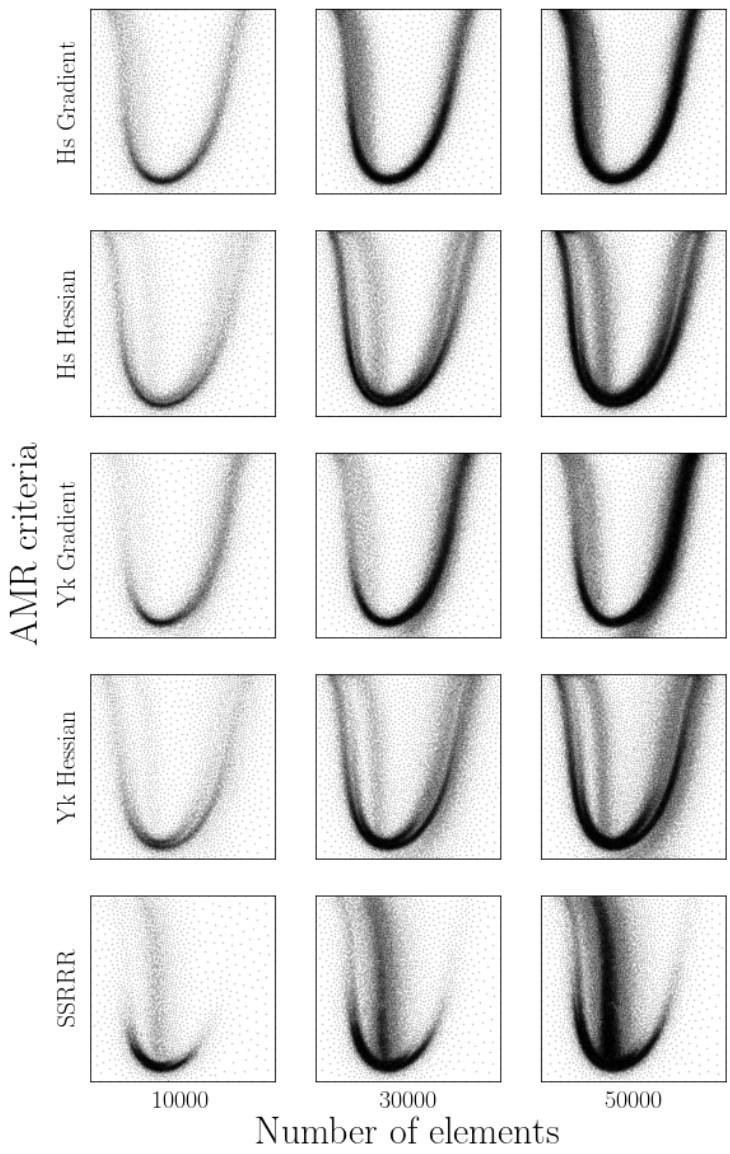


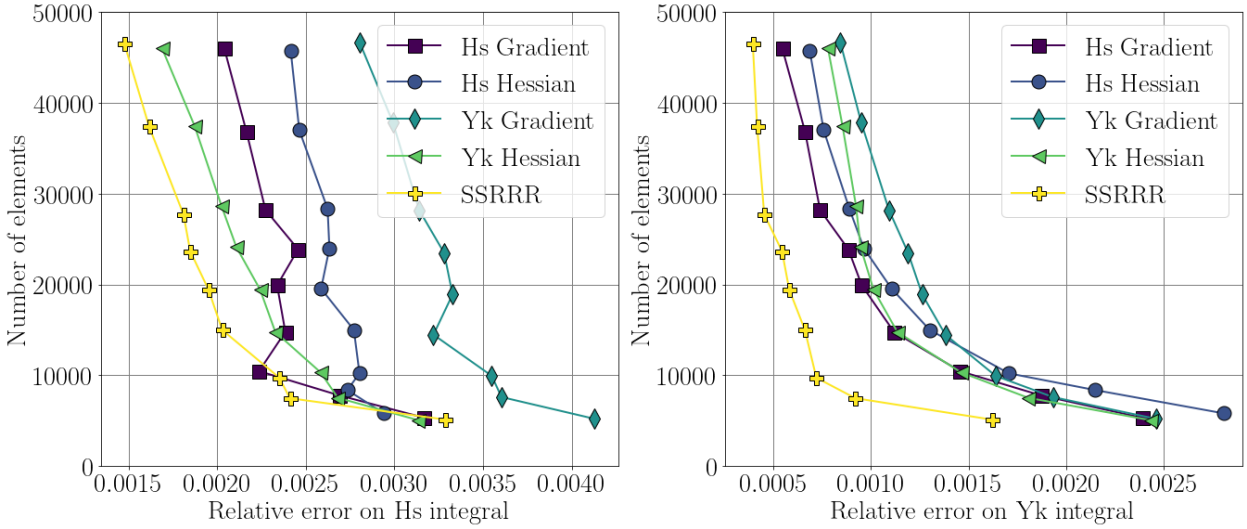
Figure 5.7: Adapted meshes for different criteria and element count. Every node is plotted to show the mesh node density.

properly. In order to avoid any statistical error, 100 distinct independent meshes are generated for each converged target metric. The error can then be computed as the average of the error for quantities that integrated on the full simulation domain. For the sensible enthalpy:

$$E(H_s) = \left| 1 - \frac{\sum_{}^{N_{\text{run}}} \sum_{i=1}^{N_{\text{elem}}} H_{s,\text{AMR},i} V_i}{\sum_{}^{N_{\text{run}}} \sum_{i=1}^{N_{\text{elem}}} H_{s,\text{ref},i} V_i} \right| . \tag{5.26}$$

The same error is also evaluated for the species: the absolute error contributions of all species are summed together before computing the relative error. The evolution of the error for the different

criteria and the number of mesh elements is given in Figs. 5.8a and 5.8b. Noticeably, the best performing criterion is the SSRRRM criterion, which for an equivalent error on Y_k and H_s allows a reduction of 50% of the cell count. A good consistency over the different transported quantities is also obtained by the H_s gradient-based criterion and outperformed by the Hs Hessian-based criterion. The worst criterion is based on the Y_k gradient, which is probably due to the fact that it prioritizes the rich-premixed flame branch over the others.



(a) Relative error on sensible enthalpy for the different criteria.

(b) Relative error on species composition for the different criteria.

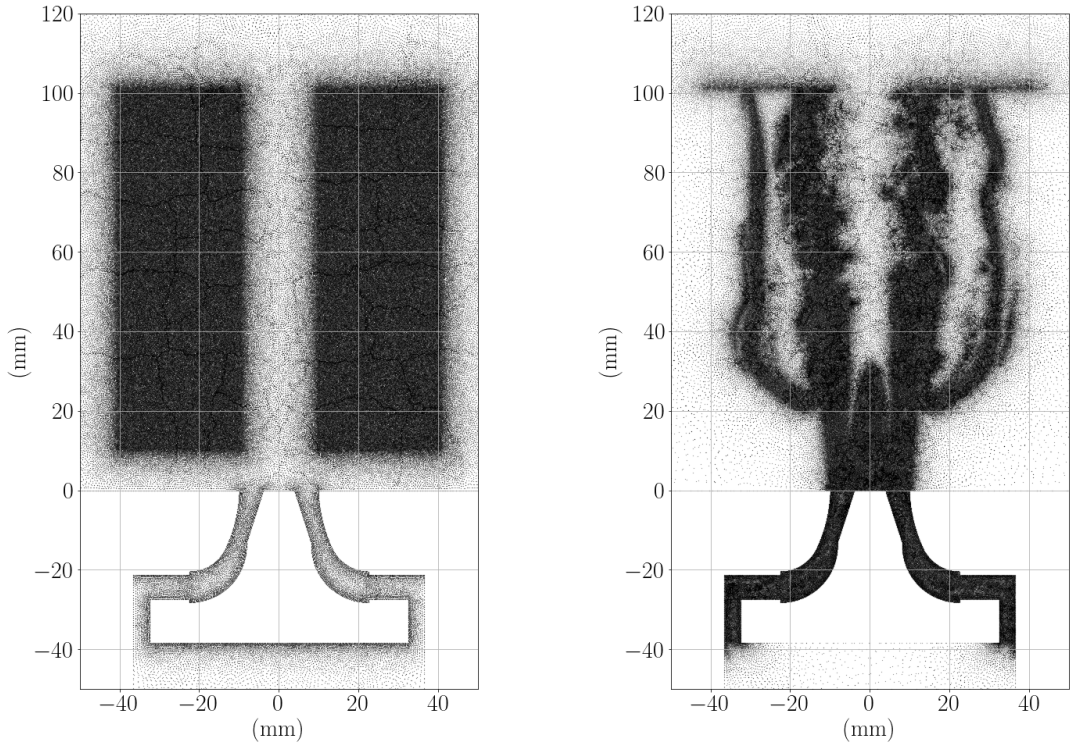
Figure 5.8: Error due to AMR on the triple flame.

5.5 Application of AMR on the CRSB

The adaptive mesh refinement strategy is now applied to the CORIA Rouen Spray Burner (CRSB). In prior simulations, a static mesh of 520M elements was used. The static mesh is represented in Fig. 5.9a. The refined region consisted of a annular block located around the main reactive region and slight refinement within the injector. The adapted mesh is generated dynamically during the run to capture the flame at a given instant. The mesh is regenerated every few hundred solver iterations to properly follow the flame dynamics. Fig. 5.9b shows a snapshot of an instantaneous mesh. To perform a valid comparison between the static and dynamic mesh, the minimum target cell size is kept identical at $200\mu m$. On the static mesh, the flame is properly resolved until a height of 100mm, after which the mesh is coarsened even thought kinetics still take place. The same approach is performed for the dynamic mesh neglecting flame dynamic beyond 100mm.

5.5.1 AMR process on the CRSB

AMR is performed based on two criteria: i) the previously introduced SSRRRM criterion, which was the best performing, and ii) an aerodynamic criterion to accurately resolve the turbulent non-



(a) Static mesh

(b) Adapted mesh

Figure 5.9: Comparison of the static mesh and instantaneous adapted mesh.

reactive flow. Vorticity is computed as:

$$\Omega = \nabla \times \mathbf{u}\,. \tag{5.27}$$

A threshold of the temporal mean norm is used:

$$\begin{cases} \Delta h = 0.2mm & \text{if } \langle \|\Omega\| \rangle_{time} \geq \text{threshold} \\ \Delta h = 20mm & \text{if } \langle \|\Omega\| \rangle_{time} < \text{threshold}\,. \end{cases} \tag{5.28}$$

This ensures a good resolution on average of the turbulence as the air co-flow has a constant mass flow rate and doesn't feature any recurring large scale oscillation. This criterion converges to a static mesh and doesn't require frequent remeshing to remain accurate. The threshold is set so that the refined region meets the refined flame region. Both criteria are represented individually in Fig. 5.10 and Fig. 5.11. The metrics are intersected using a minimum operator:

$$\Delta h = min\left(\Delta h_{SSRRRM}, \Delta h_{\langle \|\Omega\| \rangle_{time}}\right)\,, \tag{5.29}$$

and the metric field of Fig. 5.12 is obtained. This field is truncated to satisfy a maximum metric gradient of 0.1, shown in Fig. 5.13. Finally, the mesh is generated based on the target metric. The actual metric of generated mesh is represented in Fig. 5.14 and is very close to the target metric.

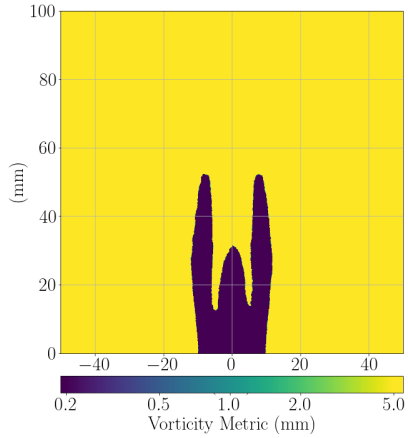


Figure 5.10: Metric from threshold of temporal mean of vorticity norm.

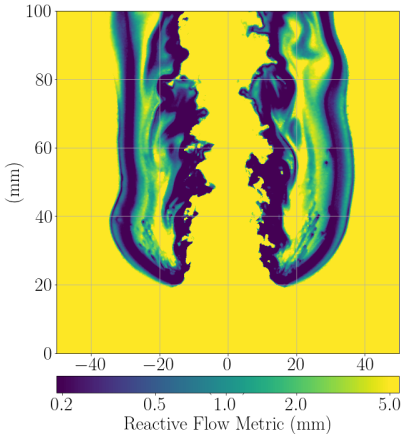


Figure 5.11: Metric from SSRRR.

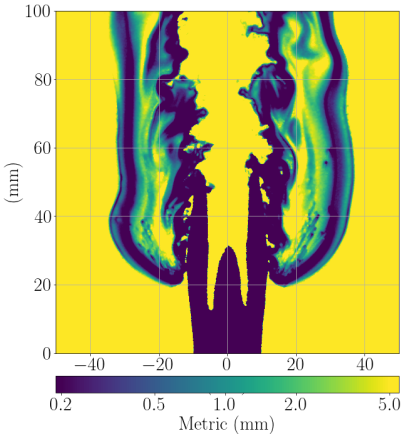


Figure 5.12: Intersected Metric.

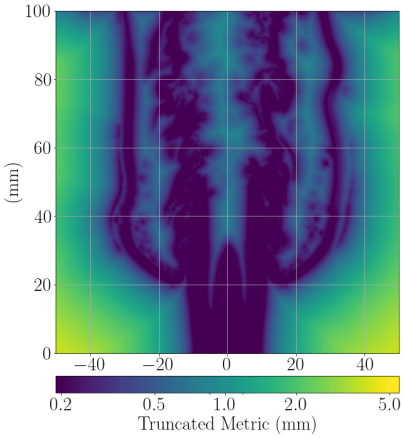


Figure 5.13: Truncated Metric with hgrad = 0.1.

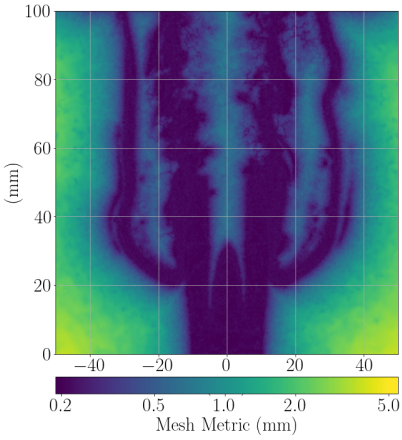


Figure 5.14: Adapted Mesh Metric.

The size of the new mesh is on average 290 million elements and 72 million nodes. Fluctuations of the mesh size over time are low with ±2% and are shown in Fig. 5.15. This behavior is ideal as the element count per parallel process remains steady over time. Thus the number of processes doesn't need to be adjusted for optimal performance. 75% of the elements can be attributed to the reactive flow criterion and 25% to the vorticity criterion.

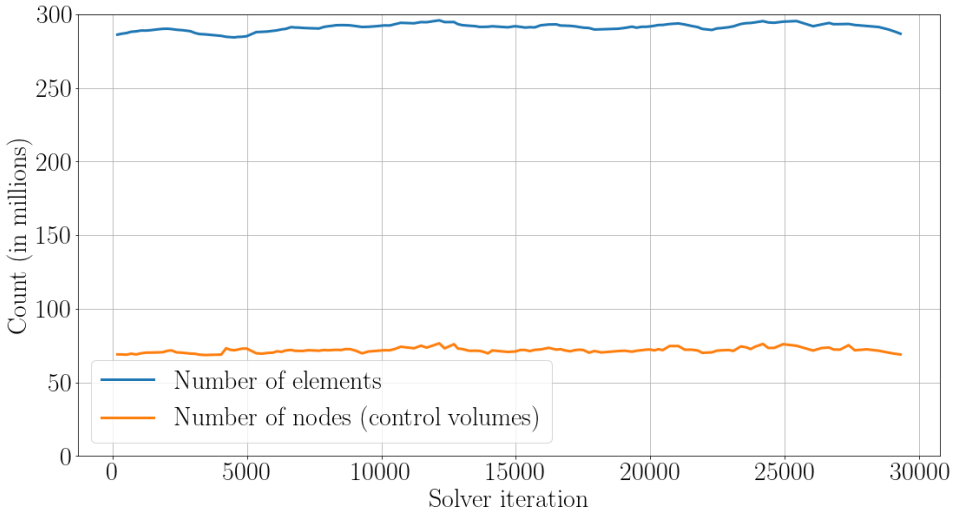


Figure 5.15: Number of mesh elements and nodes over time (50ms).

AMR is an expensive process and can't be performed at every single solver iteration. Instead AMR is triggered based on the metric error, this procedure is shown in Fig. 5.16. Every 10 iterations, a target metric is computed. This frequency is chosen as a trade-off between AMR frequency and systematic computation at every iteration. A lower value would induce notable overhead due to target metric computation and a higher value would in late AMR triggering due to undersampled metric checks. The error between the current and target metric is assessed based on Eq. 5.10. If the error exceeds 200% the AMR process is triggered. Only the error based on refinement needs is used as it signals regions that are too coarse and likely to induce high physical errors. Error based on coarsening needs is not considered as it indicates overdiscretized regions where performing AMR would only slightly reduce the mesh size, which isn't worth it compared to AMR overheads. AMR iterations are performed until the refinement and coarsening errors drops below 100%. This threshold allows fast convergence of the AMR cycles while having reasonable metric error. The triggering at 200% error results in a mean AMR frequency of every 215 solver iterations. The minimum solver iterations between two AMRs is 140 and the maximum is 500. The cumulated PDF of AMR frequency is shown in Fig. 5.17. The triggering threshold is set in a manner that the flame front is well resolved at all time but the AMR cost remains affordable. The evolution of the maximum metric error is represented in Fig. 5.18. After each AMR the error value is slighly below the 100% threshold, then rises gradually until 200%. The gradual increase relates to the flame slowly moving out of the refined region. The absence of sharp jumps of metric error directly after AMR indicates the consistency of the SSRRRM criterion on the pre-AMR and post-AMR meshes and thus of the scale similarity hypothesis.

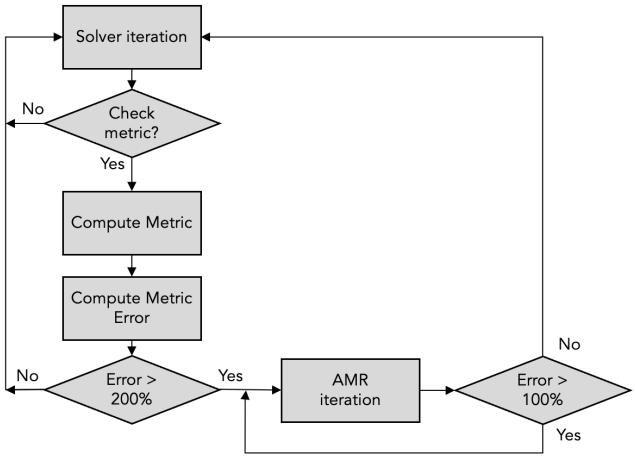


Figure 5.16: AMR procedure on the CRSB.

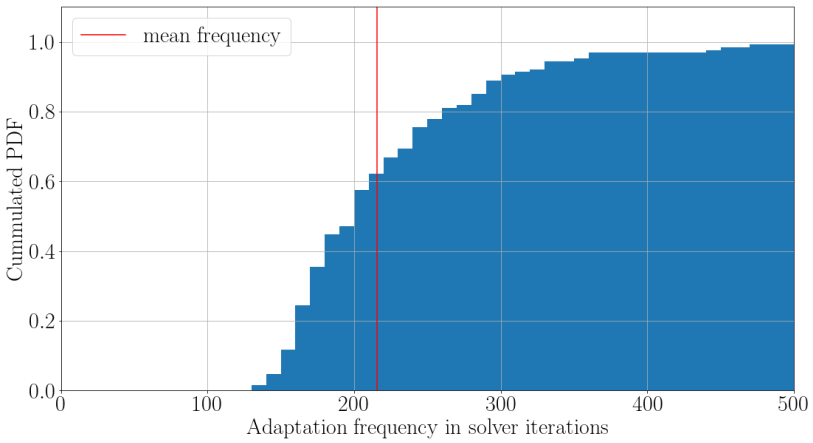


Figure 5.17: Triggering frequency of the AMR, from 50ms simulated time.

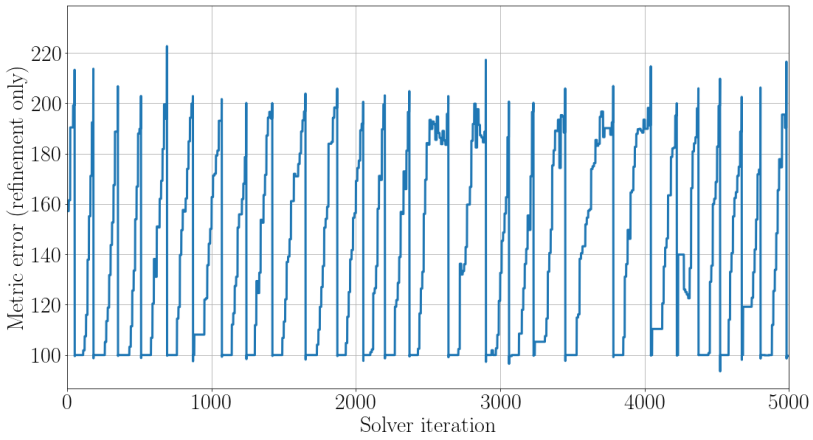
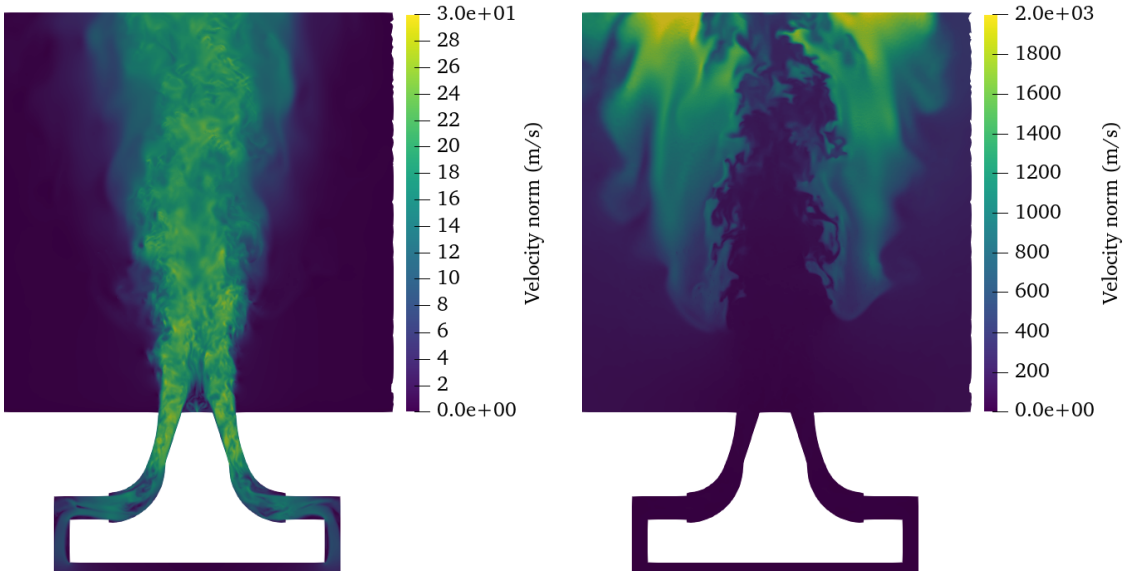


Figure 5.18: Metric error over time.

After a new mesh has been generated, values from the old mesh are interpolated to the new one. Linear interpolation is used as it is cost efficient, however this method is non-conservative. Conservative mesh interpolation relies on mesh intersection and supermeshes [150] which is too expensive to be incorporated in dynamic adaptive mesh refinement. Interpolation errors mainly occur where flow quantities strongly vary but these zones are well refined because of the AMR procedure. AMR itself strongly limits interpolation errors so that the non-conservation of mass and energy is negligible at the larger scale. At the node level, fluxes need to be adjusted to satisfy mass conservation of Eq. 3.1, this is done by correcting velocity based on the Poisson equation. For reactive flows, conservation is slightly more affected due to the change of density through the equation of state and non conservation of species.



(a) Expected velocity. **(b)** Velocity peak due to interpolation.

Figure 5.19: Velocity fields showing the interpolation induced velocity peak.

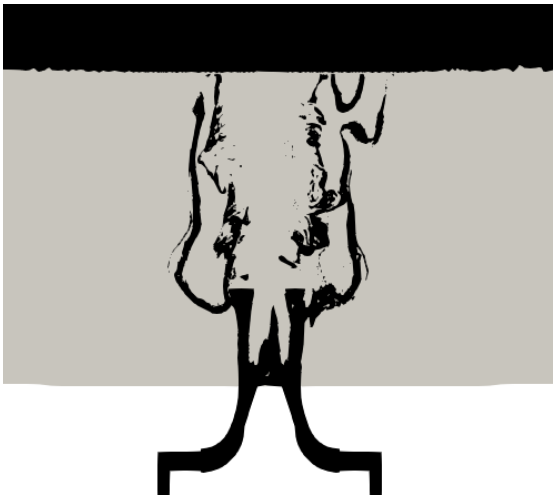


Figure 5.20: Black regions are frozen during adaptation, grey regions are adapted.

Another effect from the interpolation is the introduction of extra numerical diffusion due to the finite-volume method. Impact on non-reactive flow is negligible but for reacting flows, this acts as extra mixing of the species. Within a premixed flame, a slight increase of source term production is induced as the flame is slightly thickened due to the diffusion. The impact on diffusion flames is more drastic as the combustion process is controled by the diffusion process. The interpolation-induced diffusion can exceed the physical diffusion by several orders especially in coarser regions, as a result, source term production is greatly enhanced at all nodes. This leads to an abrupt thermal expansion of the diffusion flame thus high velocities through the Poisson equation correction. Observed velocity peaks can be several order of magnitude higher than the normal maximum velocity of the computation and lead to crashes or formation of strong nonphysical vortices. Fig. 5.19 represents instantaneous velocity fields before and after interpolation. To prevent this behavior, adapting directly in the flame front is strongly discouraged. Instead the adaptation is performed upstream and downstream of the flame, the flame remains in a region at the minimum cell size and never adapted itself. Fig 5.20 shows the so-called "frozen" regions during the adaptation. The injector is frozen at all times, same goes for regions above a certain height where the mesh is no longer refined. Inbetween, where the vorticity and reactive criteria are active, the mesh is frozen if the current and new metrics are both saturated at the minimum cell size.

5.5.2 AMR performance

The impact of AMR on solver CPU cost is evaluated. In order to perform a meaningful comparison the following characteristics are the same for both the static and dynamic mesh:

1. Smallest target element: $\Delta h_{min} = 200\mu m$.
2. Mean time step: $dt = 1.6\mu s$.
3. Number of elements per process: 50'000.
4. Hardware: AMD cores.

Performance is measured using RCT recalled as:

$$RCT = \frac{\text{Wall clock time}(\mu s).N_{\text{process}}}{N_{\text{nodes}}.N_{\text{iterations}}}\,. \tag{5.30}$$

which expresses the solver cost of one iteration for one control volume. The RCT of the main parts of the code are given in Fig. 5.21. Overall the RCT is increased by 12% with AMR. This increase comes from the AMR overhead and computation involving species source term and diffusion increases as well. This species cost increase is due to a higher fraction of nodes being located in highly reactive region on the adapted mesh. The cost of pressure correction decreases as it is a global parallel operation and number of processes is reduced on the adapted mesh leading to better parallel performance. CPU relative cost is obtained as:

$$\frac{N_{nodes,AMR}}{N_{nodes,static}}\frac{RCT_{AMR}}{RCT_{static}} = 0.626\,, \tag{5.31}$$

which mean that CPU cost has been cut by 37.4 %.

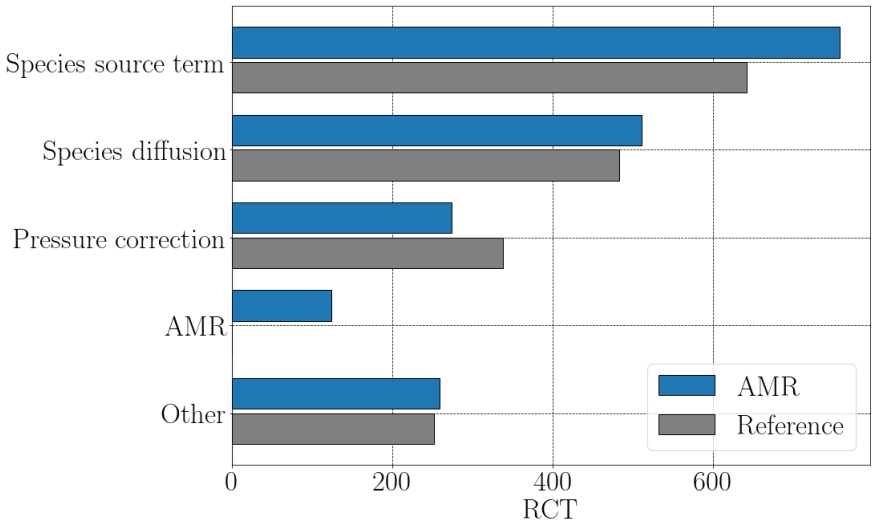
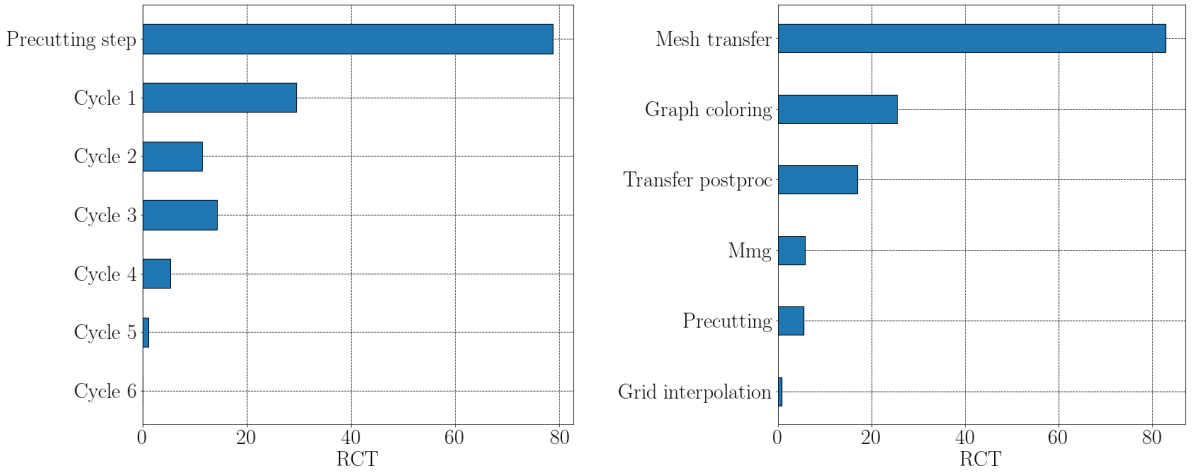


Figure 5.21: Adaptive mesh refinement performance.



(a) AMR step RCT.

(b) AMR algorithm RCT.

Figure 5.22: Detailed RCT of AMR overhead.

AMR cost is further decomposed in Fig. 5.22a. Most cost is located in the precutting step, as most of the elements are generated during that step. Mmg based AMR cycles mainly ensure good element quality and metric gradient. Cost of subsequent cycles reduces as less work needs to be done or are not always performed. Fig. 5.23 shows how many cycles are performed on average. Most of the time 3 cycles are sufficient to complete the adaptation. This also explains the slight overcost of cycle 3 in Fig. 5.22a as the last cycle involves extra post processing to rebuild all the connectivity and objects.

A more interesting cost decomposition is obtained by looking at individual algorithms. Each AMR step involves mesh generation and load-balancing. This is shown in Fig. 5.22b. Mesh generation with Mmg and precutting only represent 9% of adaptation cost.

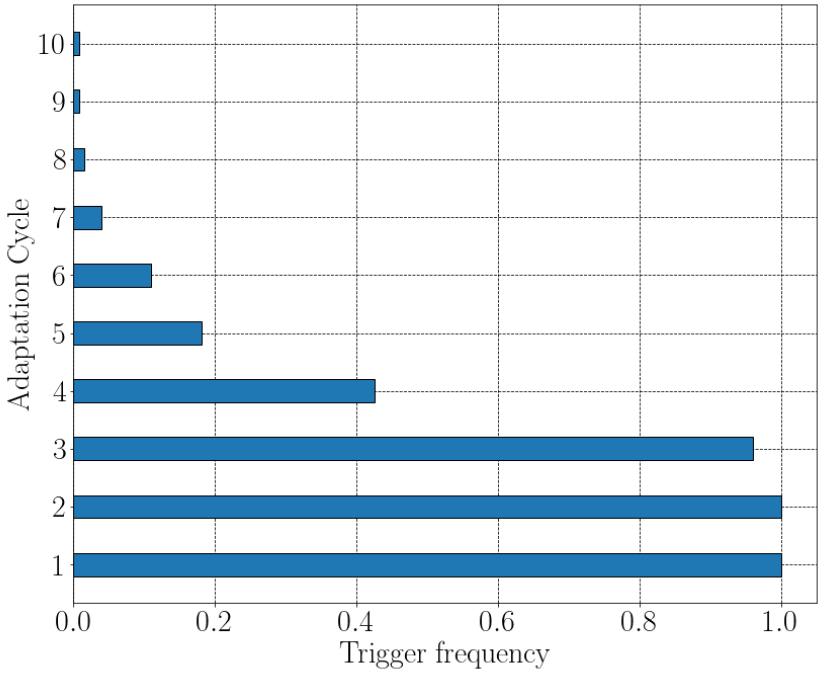
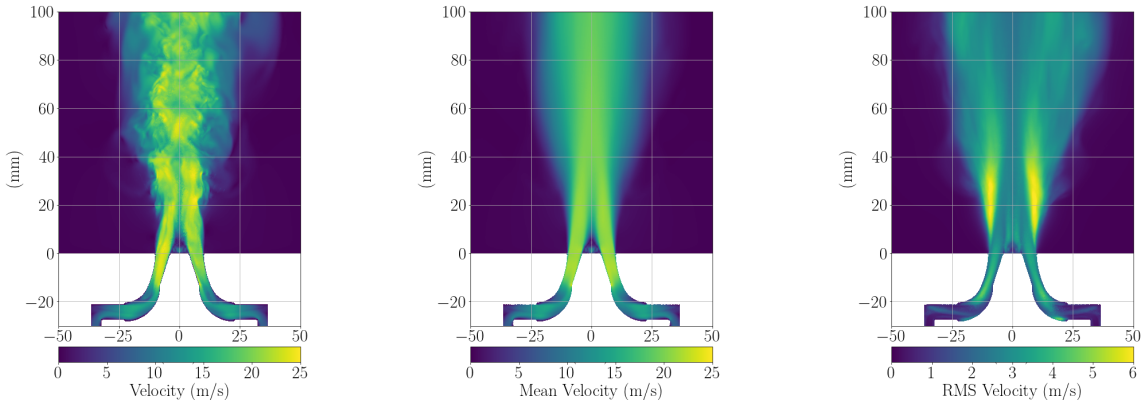


Figure 5.23: Frequency of adaptation cycles.

5.5.3 Comparison to experimental data and static mesh

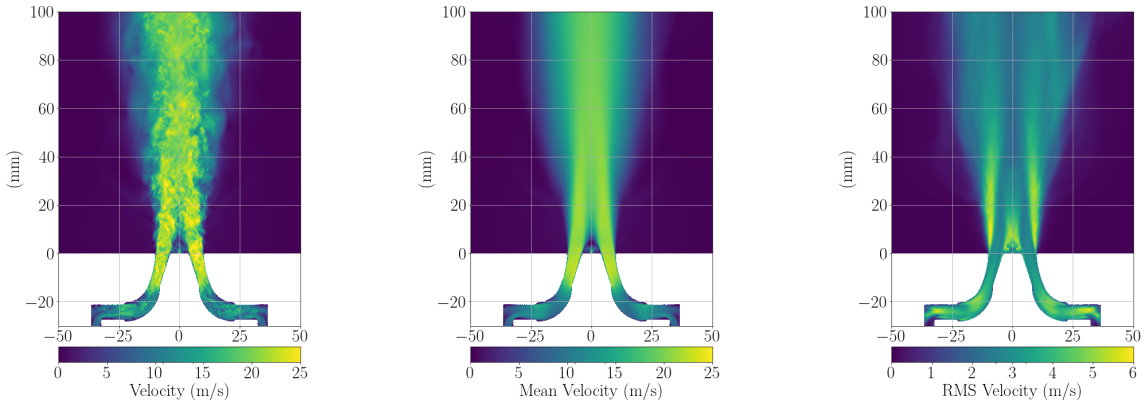
The main difference between the dynamic and static mesh is the presence of a criterion to properly resolve turbulence of the air coflow, that was neglected before. Velocity fields are given in Fig. 5.24 and Fig. 5.25. While the mean velocity field remains the same, the length scale of the instantaneous fluctuations greatly decreases. This turbulence is crucial to capture the dynamics of the Inner Reaction Zone (IRZ), where flame wrinkling is dependent on the turbulent flow. Fig.5.26 shows the mean and instantaneous fields of OH mass fraction for both simulations and the experiment. The experiment and the AMR simulation are in good agreement for the instantaneous inner front dynamics. Small structures are present as well as the periodical appearance of small wrinkles due to turbulence and some more diffuse areas. The static mesh exhibits large structures, due to a lack of upstream resolution between the air inlet and the refined region. The position of the mean inner flame is identical for the simulations and the experiments.

Turbulent flow also causes local flame extinction in the IRZ due to high strain rates. On the AMR simulation, a decrease of OH mass fraction towards the base of the flame is notable. This decrease can be directly linked to the flame extinctions. Fig. 5.27 shows a very good agreement between the experimental measures of flame extinction due to strain and the simulated OH mass fraction field weakening. This extinction dynamics is further supported by a sequence of instantaneous velocity fields and Y_{OH} isolines in Fig. 5.28. The impact of the turbulent structure on the flame base can be seen leading to a small extinction.



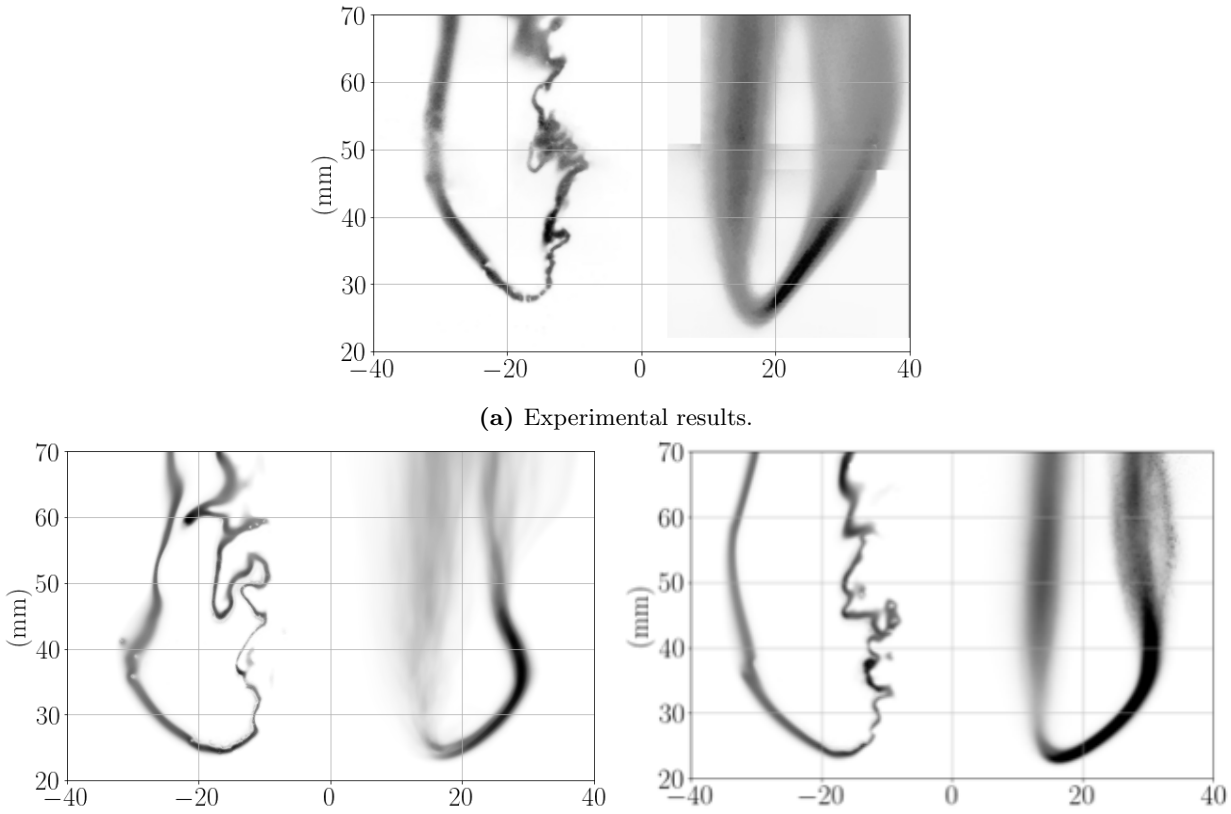
(a) Instantaneous velocity field. **(b)** Mean velocity field. **(c)** RMS velocity field.

Figure 5.24: Velocity fields on static mesh.



(a) Instantaneous velocity field. **(b)** Mean velocity field. **(c)** RMS velocity field.

Figure 5.25: Velocity fields on adapted mesh.



(a) Experimental results.

(b) Simulation on static mesh.

(c) Simulation on adapted mesh.

Figure 5.26: OH instantaneous(left) and mean(right) fields from experiment and simulations.

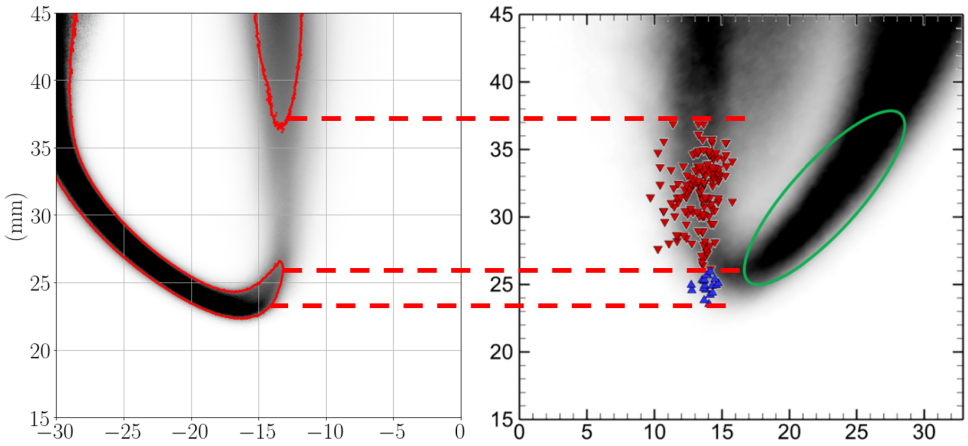
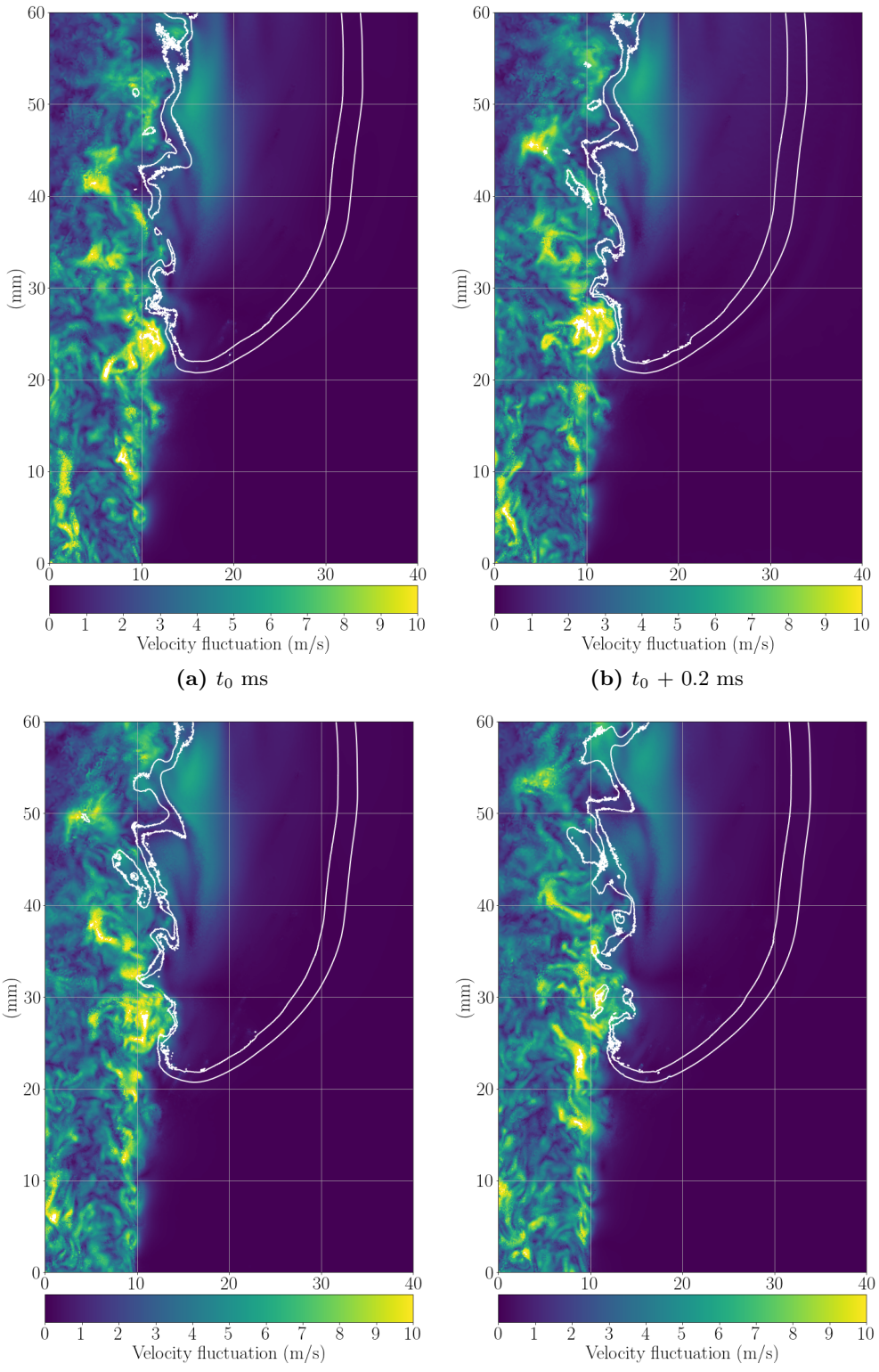


Figure 5.27: Flame extinctions on numerical (left) and experimental (right) flame, based on the OH mean field. Red triangles represent recorded flame extinctions due to high strain rates and blue triangle extinctions due to ballistic droplets crossing the flame front. Red outline is the isovalue $Y_{OH} = 0.001$. Experimental image is reproduced from [31].



(a) t_0 ms **(b)** $t_0 + 0.2$ ms

(c) $t_0 + 0.4$ ms **(d)** $t_0 + 0.6$ ms

Figure 5.28: Velocity fluctuations $\|U - U_{mean}\|$ and flame extinction.

5.6 Conclusion

This chapter proposes an evaluation of several criteria for feature-based mesh adaptation of multi-regime turbulent flames. The best-performing criterion is based on the Scale-Similarity Resolved Reaction-Rate Model (SSRRRM) which enables to evaluate the local errors in the chemical source terms. The adaptation strategies are evaluated on a 2D H2/air triple flame. This criterion is then applied to the CRSB in conjunction with a vorticity criterion to improve the resolution of the turbulent flow. Improved flame dynamics are obtained while reducing the simulation cost by 37.4%. This opens the ways to even more refined simulations.

CHAPTER 6

Reduction of dimensionality and complexity of chemistry

Dynamic Cell Clustering (DCC) is explored as a method to reduce the computational cost of source term integration in simulations. To minimize the errors introduced by DCC, a new Jacobian-Free mapping method is proposed. The effectiveness of this DCC strategy is demonstrated on the CRSB, showing its potential for improving simulation efficiency while maintaining accuracy.

Published in Proceedings of the Combustion Institute (2024) [151].

Contents

6.1 Intro

Finite-rate chemistry (FRC) is the most direct approach in combustion simulations. It involves solving a system of transport equations for each participating chemical species and calculating the rates of chemical reactions based on kinetic mechanisms. In reaction regions, chemical timescales can be several order of magnitude smaller than the flow timescales, thus requiring splitting approaches where chemistry is integrated separately from the flow. As a result, FRC offers precise representation of chemical processes but can become prohibitively expensive, particularly when increasing the size of kinetic schemes.

6.2 Dimensionality reduction

The aim of detailed chemistry is to reproduce, with the highest possible precision, the chemical pathways involved in combustion. A detailed scheme is an exhaustive list of all possible elementary reactions between all species involved in the conversion process of a given fuel by an oxidizer. Most general schemes are designed to be valid on the entire range of thermodynamic states and mixture compositions. This leads to a high number of elementary reactions and intermediate species. Grimech 3.0 [152] is one of the most well known schemes for CH4 and includes 53 species and 325 reaction. More complete schemes like the CRECK-POLIMI [153] can reach hundreds of species for thousands of reactions. These schemes can be solved for 0D reactors or 1D flames but have a prohibitive cost on large multidimensional reactive flows. This section references method that allow to decrease the dimensionality of the chemistry integration.

6.2.1 Kinetic scheme reduction

Chemical scheme reduction methods allow to decrease the number of species and reactions by introducing minimal error. The reduction can be performed by targeting only specific combustion regimes, which are required by the user in a given reactive flow configuration. This allows for a greater reduction potential but leads to large errors if combustion regimes are reached that were ignored in the reduction. The first reduction step consists in generating a skeletal mechanism. This is done by reducing the number of species and reactions:

- **Direct Relation Graph with Error Propagation (DRGEP)** [154, 155]: Identifies the species and reactions that are not relevant to the target combustion regimes. According to the type of DRGEP reduction, each species or reaction is assigned a coefficient that quantifies how strongly it is related to the target quantity of interest. The reactions and species with the lowest coefficients are successively removed until the induced error tolerance is exceeded.

- **Lumping** [156, 157, 158]: Groups multiple species or reactions into a single "lumped" or pseudo-species. Candidates for lumping are found based on their molecular composition, the thermodynamic data and kinetic parameters. Lumping is only done on intermediate species.

Systematically/Analytical reduced schemes [159, 93] allow to reduce the scheme complexity by using the Quasi Steady-State Assumption (QSSA). For some species, consumption rate constants are large compared to production rates constants. QSSA supposes that these species are fully consumed as soon as they are produced. This removes the high stiffness and low characteristic timescales associated to such species. These species also no longer need to be represented and transported on the CFD mesh.

6.2.2 Virtual chemistry

An alternative reduction method is virtual chemistry [160, 161]. As for the previous reductions specific combustion regimes are targeted. The reduced scheme is not obtained by reducing the initial scheme but is instead built from scratch. Main species of interest are kept as a base for the scheme creation. Virtual chemistry approach then gradually increases the number of species and reactions to properly capture the targeted properties. Introduced virtual species and virtual reactions do not represent real chemical entities or kinetic paths but are optimised to induce realistic behaviors for the main species. The optimization is done through stochastic methods and evolutionary algorithms.

6.2.3 Adaptive chemistry

Adaptive Chemistry [162] further builds on the idea of kinetic mechanism reduction by using several reduced mechanism within a single simulation. Each scheme is constrained to a very specific combustion regime and applied locally in the simulation. Reductions are done mainly based on DRGEP and allow to exclude inter species or negligible reaction paths. Dynamic Adaptive Chemistry (DAC) [163, 164] suggests on-the-fly kinetic mechanism reductions to best fit to the underlying regimes. Delimitation of zones for given reductions is first done on user set scalars like the equivalence ratio but is later automated through dimensionality reduction and data clustering [165, 166, 167].

6.2.4 Principal Component Analysis

Mixture composition can have a huge number of degrees of freedom depending on the kinetic scheme. However, evolution of species is heavily correlated due to the occurring reactions. If a species is consumed, another one is always produced in the process. Thus, dimensionality of such system can be reduced by considering scalar variables like the progress variable and the mixture fraction, relating to flame features. The expression of these variables is heavily dependent on the the fuel and the flame archetype. This requires user knowledge and is likely to result in a sub-optimal reduction. Alternatively, automated methods like Principal Component Analysis (PCA) [168] allow to extract main flame features and measure the induced error. PCA has been extensively used in conjunction with optimization techniques like ISAT [169] or adaptive chemistry [170, 165]. PCA automatically creates an optimal low-dimensional representation of a mixture [171, 172, 173] by identifying the

principal components, which are linear combinations of the state variables. Mathematically, PCA involves finding the eigenvectors and eigenvalues of the covariance matrix of the input data. Let X be the matrix of species mass fraction with shape $[N_{sp}; n]$, where N_{sp} is the number of species and n the number of grid nodes. The covariance matrix is expressed as:

$$C = \frac{X^T X}{n-1}\,. \tag{6.1}$$

Diagonalization of C allows to extract the principal directions along which species variance is maximized:

$$C = V\frac{D}{n-1}V^{-1}\,, \tag{6.2}$$

with the diagonal matrix $D = \{\lambda_1, ..., \lambda_{N_{sp}}\}$ and V the eigenvectors containing the principal directions. Most of the variance is contained in the first principal directions while the last ones hold negligible variance. method to perform PCA is Singular Value Decomposition (SVD). SVD provides the following decomposition:

$$X = USV^T\,, \tag{6.3}$$

with U and unitary matrix, $S = \{\sigma_1, ..., \sigma_{N_{sp}}\}$ a diagonal matrix and V the right singular vectors. The relation to PCA can be shown be taking the covariance matrix:

$$\begin{aligned} C &= \frac{\left(USV^T\right)^T\left(USV^T\right)}{n-1} \\ &= \frac{VSU^TUSV^T}{n-1}\,, \end{aligned} \tag{6.4}$$

for an unitary matrix:

$$U^TU = I\,, \tag{6.5}$$

and V is an orthonormal matrix implying:

$$V^T = V^{-1}\,. \tag{6.6}$$

Using these properties in Eq. 6.4:

$$C = V\frac{S^2}{n-1}V^{-1}\,. \tag{6.7}$$

Showing the relation between standard PCA and SVD:

$$\begin{aligned} D &= S^2 \\ \{\lambda_1, ..., \lambda_{N_{sp}}\} &= \{s_1^2, ..., s_{N_{sp}^2}\}\,. \end{aligned} \tag{6.8}$$

Dimensionality can be reduced by considering only the k first PC. The low-dimensional representation of ϕ is referred to as M. Reconstruction error can be expressed as:

$$\epsilon_M = \|X - XV_MV_M^T\|^2\,, \tag{6.9}$$

$$\epsilon_M = \sum_{i=M+1}^{N_{sp}} \lambda_i \,. \tag{6.10}$$

Instead of solely reducing the mixture, the full state could be considered as temperature is correlated to the species. The main difficulty is that PCA has to be performed on data that may not have the same unit. Data needs to be rescaled as PCA is very sensitive to the amplitude because it is based on the Euclidean norm. Several scalings have been suggested to balance the importance of all state-space variables. Some existing scalings are listed in Tab. 6.1. There is no clear consensus on which scaling is optimal and it is case dependent.

Method	Scaling Factor
Auto	σ_j
Pareto	$\sigma_j^{0.5}$
Range	$max\,(X_j) - min\,(X_j)$
VAST	$\sigma_j^2/\overline{X}_j$

Table 6.1: Common data scaling techniques, σ_j is the standard deviation of X_j.

6.3 Complexity reduction of source term integration

Complexity reduction can be achieved through various methods. Geometric modeling of the flame front, such as the G-equation or Flame-Surface Density approach, eliminates the need for detailed kinetic schemes by focusing solely on the flame geometry. Similarly, the assumption of infinitely fast chemistry [174], in diffusion flames, posits that chemical species reach equilibrium almost immediately after mixing, thereby bypassing the need to account for finite-rate chemical reactions. Additionally, neural networks have been explored as a means to replicate chemical behavior at a lower computational cost [175, 176]. Finally, source term precomputation, storage, and retrieval methods enable the preservation of high-fidelity finite-rate chemistry while significantly reducing computational costs. These latter methods are discussed in this section.

6.3.1 Tabulated chemistry

Tabulated chemistry consists in precomputing a thermo-chemical database by using a given detailed, skeletal or reduced mechanism. During the CFD simulation, source terms are interpolated on the fly from the look-up table instead of being solved. This allows to strongly mitigate chemical cost. The table is constructed based on a reduced number of dimensions than the original kinetic scheme to produce a table of reasonable size. The choice of the dimensions is crucial to generate an appropriate chemical database, introducing minimal error. Intrinsic Low-Dimensional manifold (ILDM) [177] method identifies a low-dimensional attractor sub-space which describes the slow reaction dynamics. Fastest timescales are cut-off and assumed at local equilibrium. This analysis can be performed based on the Jacobian of the kinetic system. Overall, highly-reduced ILDM manifolds do not correctly reproduce the low-temperature regions of the flame which are sensitive to molecular diffusion [178]. The flame generated manifold (FGM) [179] is an alternative approach that assumes that a turbulent flame can be decomposed into 1D flame elements. Flamelets are generated

using a reduced number of dimensions referred as control variables. Selection of these variables is done based to include specific phenomena like mixing effect, kinetics and heat exchanges. Flame Prolongation of ILDM (FPI) [180] combines both of the approaches. The coupling of tabulated chemistry with turbulent combustion can be done using the FPI-PCM (Presumed Conditional Moment) [181] relying on Probability Density Functions (PDF) to take into account sub-grid scale wrinkling and mixing. This formulation does not guarantee a proper prediction of regimes where the sub-grid scale flame wrinkling vanishes but where most of the reaction zone remains at the sub-grid scale. Filtered TAbulated Chemistry model for LES (F-TACLES) [182] proposes a correction based on 1D filtered flames. In all mentioned methods, a downside of tabulation is its memory complexity. The size of the tabulated database can become important especially when multi-regime combustion is considered. In parallel simulations, the full table needs to be loaded on each process which can cause memory issues.

6.3.2 Dynamic tabulated chemistry

Dynamically tabulated chemistry involves creating a chemical database during the simulation using a storage and retrieval mechanism. When a required source term is not already in the database, it is computed on-the-fly and added for future use; if it is already present, the source term is efficiently retrieved from the table. This approach results in a database that only contains the combustion regimes actually encountered in the simulated reactive flow, leading to a significantly smaller database compared to a precomputed table. Moreover, it ensures the coverage of all encountered combustion regimes, unlike precomputed tables which are generated for specific regimes and are not able to exceed them at runtime. In-Situ Adaptive Tabulation (ISAT) [183, 169, 184] is the most famous approach. Three main steps are performed when a source term is required:

- **Retrieve attempt:** The first operation consists in checking if an appropriate source term is already present in the database. If found, a source term is retrieved and adjusted with a linear Jacobian-based approximation to account for the slight difference between the stored and requested composition. If not found, the source term is computed, then the table is updated via a growth or addition operation.
- **Growth attempt:** Database source terms have an ellipsoid of accuracy which is the region in phase space for which a given entry is valid. If an existing entry is not accurate enough but close, ISAT checks if the ellipsoid of accuracy can be extended. This is done by comparing the computed source term and the database predicted source term. If the error remain under an user-defined threshold, the ellipsoid of accuracy of the entry is extended. If not, the addition is performed.
- **Addition:** If neither retrieval nor growth can be done, the computed source term is added as a new entry to the table.

6.3.3 Dynamic cell clustering

Dynamic Cell Clustering (DCC) is also referred as Cell Agglomeration or Multi-zone algorithm. Similarly to ISAT, DCC aims at reducing the number of source term integrations but doesn't rely on building a database. It consists of three steps: i) grouping of elements with similar composition into

clusters, ii) computation of a single element per cluster and iii) mapping of the computed elements to the remaining elements of the cluster. This method is hereafter described in greater details as it is core to the presented work. This method has been explored in various works [185, 186, 187, 188, 189].

Cluster creation

Clustering methods aim at grouping similar computational cells together. Each cell acts as a chemical reactor, therefore clustering has to consider each parameter defining the reactor's behavior. A common state vector is $\phi = \{P, T, Y_1, ..., Y_{Nsp}\}$, comprising the pressure, temperature and mixture composition.

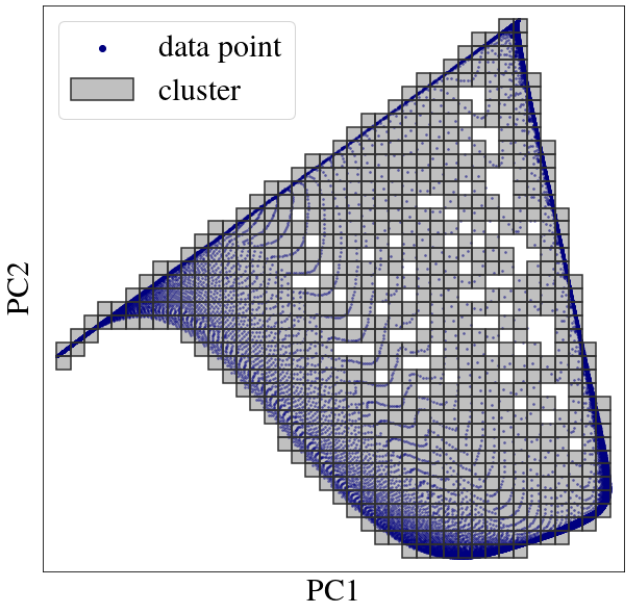


Figure 6.1: Coarse clustering of data points along two PCA dimensions of a H2-air triple flame. Those dimensions are a rotation of a mixture fraction and a progress variable.

Two main clustering algorithms are used for DCC: K-means and grid clustering. K-Means [190] is one of the most popular and simplest clustering algorithms. It aims at partitioning data into K clusters, where each data point belongs to the cluster with the nearest mean value. It iteratively minimizes the sum of squared distances between data points and their assigned cluster's centroid by moving the latter until an optimum is found. It has been used for DCC in [186]. An alternative approach is grid-based clustering [191]. This method divides the data space into a grid of cells. Each data point is then assigned to the grid cell that corresponds to its state coordinates. The goal is to group data points that fall within the same grid cell, effectively simplifying the clustering process and potentially making it more efficient for large datasets or high cluster counts. A representation is given in Fig. 6.1. This method has been used for DCC in [187]. This latter method is used in this work as it is a requirement for the new mapping method introduced in Sec. 6.4.

In order to reduce the clustering overhead, PCA can be used to reduce the dimensionality of the species and thus the dimensionality of the clusters.

Source term computation

For each cluster, a single source term has to be computed. The Cell Agglomeration algorithm name initially refers to the grouping of all reactors of a cluster into a single large averaged reactor with state:

$$\overline{\phi} = \frac{1}{N} \sum_{i=1}^{N} \phi_i \,. \tag{6.11}$$

Other methods consist in selecting one the the reactors present within the cluster. This can be done by selecting the most centered reactor relative to the cluster:

$$\arg\min f(\phi_i) = \|\phi_{cluster} - \phi_i\|_2 \,, \tag{6.12}$$

with $\phi_{cluster}$ the center of the cluster, or relative to the other reactors:

$$\arg\min f(\phi_i) = \sum_{j=1}^{N} \|\phi_j - \phi_i\|_2 \,. \tag{6.13}$$

The latter is used in this work. No significant accuracy changes have been found between Eq.6.12 and Eq. 6.13, comparison of these methods is shown during the validation.

Source term mapping

Once a single reactor has been integrated, information needs to be propagated to the other reactors. Several approaches exist:

- Average mapping consists in assuming the same state for all reactors following Eq. 6.11. When a reactor other than the average reactor is computed, this method is no longer viable as it disrespects atomic conservation and mass conservation.

- Backward mapping no longer assumes a common state for all reactor but rather a common source term. When negative source term is weighted to avoid negative mass for any species. This mapping is introduced in [188] and expressed as:

$$\begin{cases} \rho_{k,i}^{t+\Delta t} = \rho_{k,i}^{t} + \Delta m_{k,cluster} \dfrac{\rho_i^t}{\sum_{j=1}^{n} \left(\rho_j^t V_j\right)} & \text{if} \quad \Delta m_{k,cluster} \geq 0 \\ \rho_{k,i}^{t+\Delta t} = \rho_{k,i}^{t} + \Delta m_{k,cluster} \dfrac{\rho_{k,i}^t}{\sum_{j=1}^{n} \left(\rho_{k,j}^t V_j\right)} & \text{if} \quad \Delta m_{k,cluster} < 0 \,. \end{cases} \tag{6.14}$$

  Terms with subscript i refer to the node being mapped. $\Delta m_{k,cluster}$ is the mass produced by the agglomerated reactor. This formulation can be rewritten as:

$$\begin{cases} Y_{k,i}^{t+\Delta t} = Y_{k,i}^{t} + \omega_{k,cluster} & \text{if} \quad \omega_{k,cluster} \geq 0 \\ Y_{k,i}^{t+\Delta t} = Y_{k,i}^{t} + \omega_{k,cluster} \frac{Y_{k,i}^{t}}{Y_{k}^{t}} & \text{if} \quad \omega_{k,cluster} < 0 \,. \end{cases} \tag{6.15}$$

- Kinetics can be used to provide a linear approximation of the source term. The Jacobian describes the instantaneous evolution of the source terms relative to the composition:

$$J_{ij} = \frac{\partial \dot{\omega}_i}{\partial \phi_j} \,, \tag{6.16}$$

as the source terms are integrated over time, the integrated Jacobian is required:

$$A_{ij} = \frac{\partial R_i}{\partial \phi_j} \,. \tag{6.17}$$

A linear approximation of the source terms is expressed as:

$$\phi^{t_0+\Delta t} = \phi^{t_0} + \omega_{cluster} + \left(\phi^{t_0} - \phi^{t_0}_{cluster}\right) A \,. \tag{6.18}$$

This method provides great accuracy but evaluation of the integrated Jacobian, A, is very expensive as $N_{sp} + 1$ additional reactors need to be solved. In methods involving storage like ISAT, this cost is smoothed out over time, but is far too great to be evaluated at every iteration and defeats the purpose of DCC.

6.4 Jacobian-free mapping

This thesis presents a new mapping method, based on a Jacobian-free estimation. This estimation is designed to be low-cost while providing similar accuracy to an explicit Jacobian. Connectivity is created between the clusters, which is then used to approximate the Jacobian. Mapping is done based on those, while limiting degenerate cases.

6.4.1 Cluster connectivity

A connectivity map of adjacent clusters is created based on the cluster grid coordinates. Two clusters M_{C_1} and M_{C_2} are considered adjacent when they connect orthogonally:

$$\begin{aligned} |M_{C_1,i} - M_{C_2,i}| &= 1 \,, \\ M_{C_1,j} - M_{C_2,j} &= 0 \quad \text{for all } j \neq i \,. \end{aligned} \tag{6.19}$$

An example of this connectivity, which can be computed efficiently using sparse matrices and sorting, is represented in Fig. 6.2. In this figure, the cluster center is the computed reactor, which is chosen as the one that minimizes composition difference $f(M_{\phi_i})$ to the other reactors:

$$f(M_{\phi_i}) = \sum_{j \neq i} \left\| M_{\phi_i} - M_{\phi_j} \right\|_2^2 \,. \tag{6.20}$$

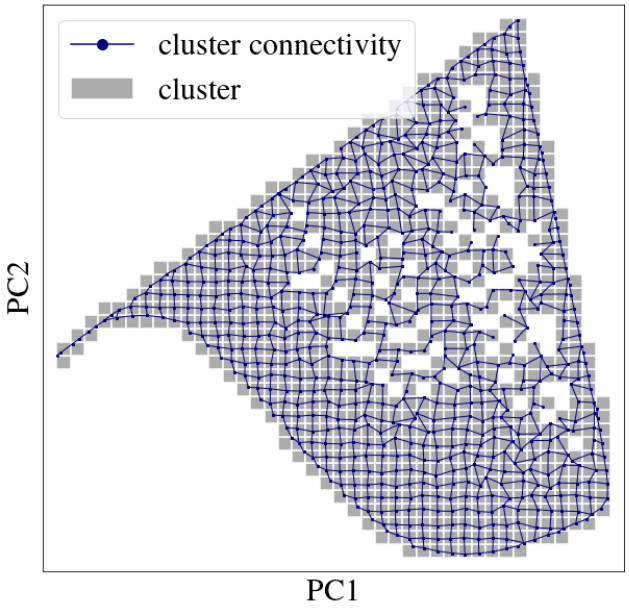


Figure 6.2: Cluster connectivity based on computed reactor position based on Eq. 6.20.

Each edge of the connectivity graph allows to compute the Jacobian vector of the reaction rates within the reduced space M integrated along the edge. Let ϕ_{Ci} and ϕ_{Cj}, be the time-integrated reactors of two neighbor clusters C_i and C_j:

$$\phi_{Ci}^{t_0+\Delta t} = \phi_{Ci}^{t_0} + \omega_i \,, \tag{6.21}$$

with $\dot{\omega}$ the source term. The source term difference between two adjacent clusters i and j is expressed as:

$$\Delta\omega_{j\to i} = \omega_i - \omega_j \,. \tag{6.22}$$

Similarly, composition difference between clusters i and j is expressed as:

$$\Delta M_{j\to i} = M_{\phi_{Ci}^{t_0}} - M_{\phi_{Cj}^{t_0}} \,. \tag{6.23}$$

The ratio of those differences is first-order approximation of the Jacobian J_M projected onto the unit vector $dM_{j\to i}$.

$$J_M dM_{j\to i} \approx \frac{\Delta\omega_{j\to i}}{\Delta M_{j\to i}} \,. \tag{6.24}$$

6.4.2 Jacobian-free estimation

Let ϕ_e be a reactor to be estimated within C_i, thus $\Delta\omega_{i\to e}$ needs to be estimated from the know displacement $\Delta M_{i\to e}$. Based on the knowledge of the Jacobian projected onto several known directions, the full Jacobian may be reconstructed and used to compute the change in the source

terms:

$$\Delta\omega_{i\to e} = J_M \Delta M_{i\to e}\,. \tag{6.25}$$

However, degenerate cases can happen due to an insufficient number of projection directions or highly co-linear directions (see Sec. 6.6). Rather than computing the Jacobian at the cluster level, the source term variation can be expressed as a weighted sum of variations in known directions:

$$\Delta\omega_{i\to e} = \sum_{j=1}^{n} \alpha_j \Delta\omega_{i\to j}\,, \tag{6.26}$$

with n the number of connected clusters and α_j the interpolation coefficients to be determined from:

$$\Delta M_{i\to e} = \sum_{j=1}^{n} \alpha_j \Delta M_{i\to j} = \mathcal{M}\alpha\,. \tag{6.27}$$

Eq. 6.27 can directly be inverted only if $n = d$, with d the number of dimensions of subspace M:

$$\alpha = \mathcal{M}^{-1}\Delta M_{i\to e}\,. \tag{6.28}$$

The case $n = 0$ is very unlikely as chemistry is continuous or is characteristic of an over-resolved clustering. This case is shared with other mapping methods. Solving for $1 \leq n < d$ may be obtained by a least square algorithm:

$$\alpha = \left(\mathcal{M}\mathcal{M}^t\right)^{-1}\mathcal{M}\Delta M_{i\to e}\,. \tag{6.29}$$

The least square method returns α which minimizes f:

$$f(\alpha) = \|\Delta M_{i\to e} - \mathcal{M}\alpha\|_2^2\,. \tag{6.30}$$

The obtained solution is the best possible projection of $\Delta M_{i\to e}$ given the insufficient amount of vectors in $\mathcal{M}$. This case is illustrated in Fig. 6.3a.

For $d < n$, Eq. 6.29 has an infinite number of solutions as there is an infinite number of vector combinations from $\mathcal{M}$ that are equal to $\Delta M_{i\to e}$. An additional constraint is set to obtain α with the smallest norm, which is expected to introduce minimal error. This is achieved with a least-square algorithm with ridge regression [192]:

$$\alpha = \left(\mathcal{M}\mathcal{M}^t + \lambda I\right)^{-1}\mathcal{M}\Delta M_{i\to e}\,. \tag{6.31}$$

The ridge regression returns α which minimizes f:

$$f(\alpha) = \|\Delta M_{i\to e} - \mathcal{M}\alpha\|_2^2 + \lambda\,\|\alpha\|_2^2\,. \tag{6.32}$$

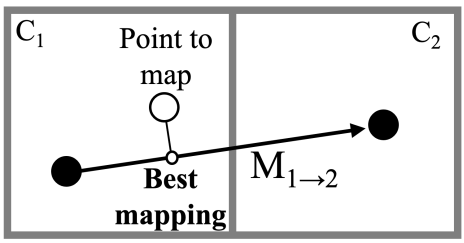
λ is chosen to be negligible compared to eigenvalues of $\mathcal{M}$ to not deteriorate the solution but way larger than machine accuracy to break the super-colinearity of the system. The constraint minimizing the norm of α is optimal as it reduces extrapolation and thus error magnitude. Once α is found, degenerate cases have to be handled. It can be shown that Eq. 6.26 is an interpolation and not an extrapolation if and only if:

$$\begin{cases} \sum_{j=1}^{n} \alpha_j \leq 1\,, & (6.33) \\ \alpha_j \geq 0 \quad \text{for all } j \in \{1, ..., n\}\,. & (6.34) \end{cases}$$

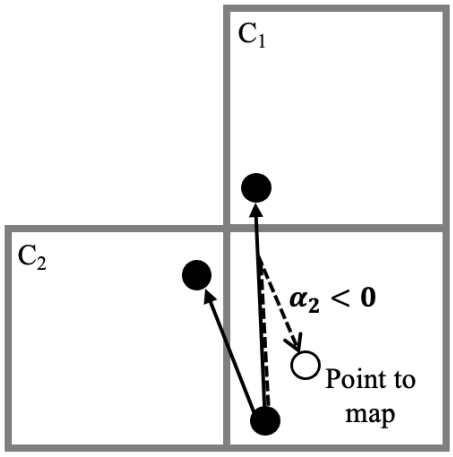
Illustrations of these special cases are given in Figs. 6.3b and 6.3c. It will be shown that extrapolation is beneficial to some extent in Sec 6.6, however avoiding excessive extrapolations remains crucial. Rescaling of α may be introduced with a user-defined limit α_{lim}:

$$\alpha_{\text{rescaled}} = \frac{\alpha}{\max\left(1; \frac{\sum_{j=1}^{n} |\alpha_j|}{\alpha_{\text{lim}}}\right)}\,, \tag{6.35}$$

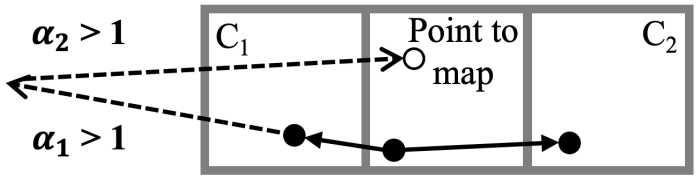
with typical values of α_{lim} ranging from 1 to 10.



(a) Projection due to $n < d$.



(b) Extrapolation due to violation of Eq. 6.34.



(c) Extrapolation due to violation of Eq. 6.33.

Figure 6.3: Special cases for Jacobian-free mapping.

6.5 Implementation

When implementing an optimization algorithm, it is essential to ensure that the algorithm's computational overhead is negligible compared to the expected gain. This section details the most important implementation choices.

6.5.1 Memory complexity

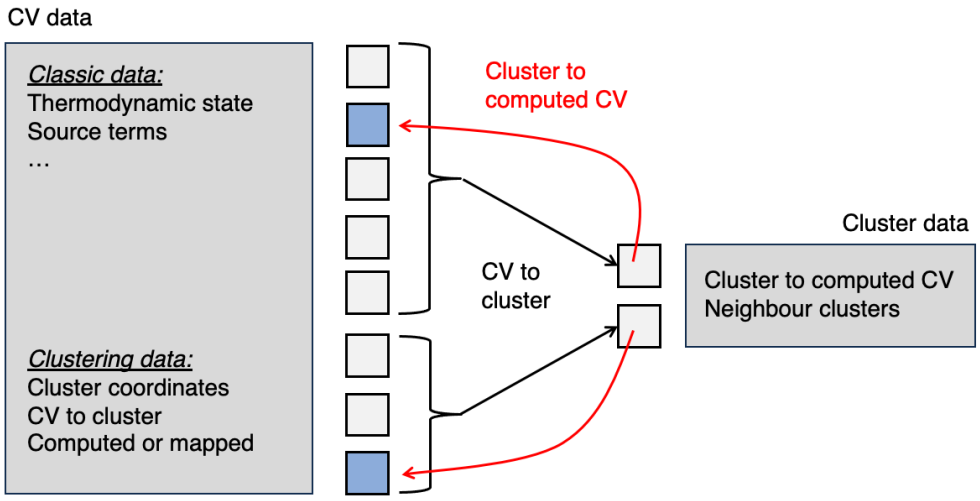


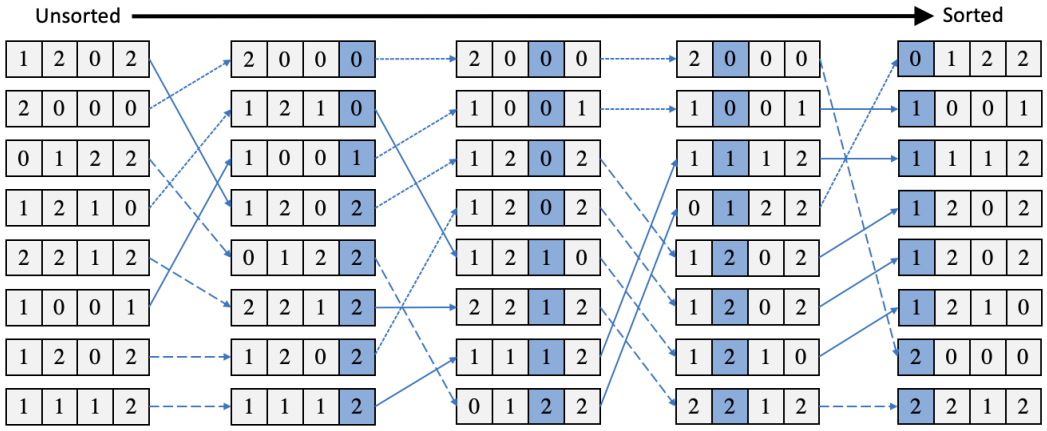
Figure 6.4: Basic clustering data structure.

Dynamic Cell Clustering (DCC) is designed to be a low-memory approach, where source terms are always stored directly within the control volumes. Clusters serve as a simple reference system, pointing to the control volume that holds the computed source term, as illustrated in Fig. 6.4. Instead of using Cartesian coordinates within the clustering dimensions, clusters are identified by a unique color. This method avoids the need to create a multi-dimensional array based on coordinates, which would result in an extremely sparse and memory intensive data structure due to most clusters being empty. However, using a 1D array to hold the clusters brings extra complexity which needs to be adressed to maintain a good time efficiency.

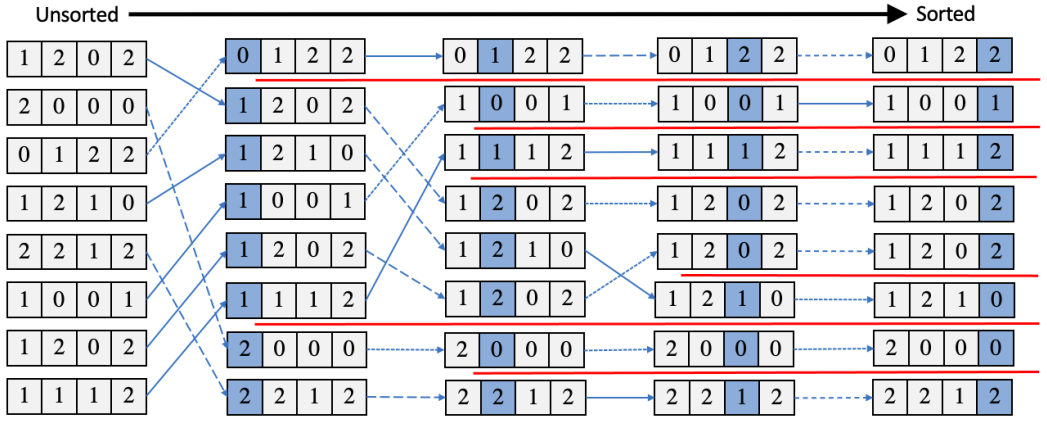
6.5.2 Time complexity

The grid-based clustering process is detailed to explain how time efficiency is achieved: Coordinates of every control volume are computed in the cluster dimensions, so that unit length corresponds to the desired cluster width. These coordinates are then floored to integer values, ensuring that control volumes sharing the same integer coordinates are grouped into the same cluster. Due to the sparse memory implementation, integer coordinates cannot be directly mapped to specific clusters. Instead, a duplication detection process is employed to identify control volumes belonging to the same cluster. This is achieved by sorting the control volumes based on their integer coordinates, thereby aligning control volumes with identical coordinates sequentially in the sorted list. The efficiency of the sorting algorithm used in this process is crucial for overall performance and radix sort method has been selected.

Radix sort



(a) Least Significant Digit (LSD) radix sort on 8 numbers with radix=3.



(b) Most Significant Digit (MSD) radix sort on 8 numbers with radix=3. Red lines represent the need for recursivity.

Figure 6.5: Radix sort.

Radix Sort [193] is a non-comparative, integer-based sorting algorithm that sorts data by processing individual digits of numbers. It is particularly effective for sorting datasets composed of fixed-size integers with a dense distribution [194, 195, 196], often outperforming comparison-based algorithms like quicksort or mergesort. This efficiency arises from the linear time complexity of radix sort, $\mathcal{O}(n.k)$, where n is the number of integers to sort and k is the number of digits in the largest integer. In contrast, comparison-based sorting algorithms have a time complexity of $\mathcal{O}(n.log(n))$. Fig. 6.5 illustrates how radix sort works. The algorithm decomposes the numbers to be sorted into a sequence of digits, using a specific base, also known as the radix. The sorting process occurs digit by digit, beginning either from the Most Significant Digit (MSD) or the Least Significant Digit (LSD). The LSD approach is more straightforward to implement, whereas the MSD approach often requires recursion. Each digit set is sorted using a stable sorting algorithm, typically Counting Sort, ensuring that the initial relative order of records with equal keys is preserved, which is an essential property for the LSD method.

Radix sort is particularly well adapted to sort the control volumes coordinates as the coordinates already act as a radix decomposition. This allows to bypass the main computational cost of traditional radix sort, which involves expensive division operations for decomposition. The largest coordinate per dimension ($\sim$ 100–1000) is a very suitable radix as recommendations suggest the choice of a large radix [196]. Most of the time radix is chosen as a power of 2 to decrease the decomposition cost based using the binary representation but this is of no concern here. The upper limit of the radix is primarily constrained by processor hardware, as the intermediate arrays created during sorting need to fit within the L1 cache. Additionally, variations in radix values between different cluster dimensions do not present any issues.

Cluster connectivity

Due to the sparse representation of clusters, determining connectivity directly can be challenging. However, radix sort can be utilized for an efficient reconstruction of this connectivity. Consider an array of coordinates for the existing clusters, applying radix sort to this array, sorting from dimension 1 to dimension d, enables the determination of connectivity along the last sorted dimension dd, as defined in Eq. 6.19. To assess connectivity along another dimension, a complete re-sorting is not required. Instead, sorting is performed only on the dimension of interest, while the other dimensions remain sorted due to the stability property of counting sort. This approach allows for the efficient recalculation of connectivity. Initially, a full radix sort is performed, followed by counting sort on each dimension from 1 to d-1. This results in a total computational cost of slightly less than two full radix sorts. It is important to note that the initial radix sort is redundant with the sort conducted for cluster creation and can thus be avoided.

6.5.3 Parallel considerations

Load-balancing

Clustering leads to an uneven, unpredictable and fluctuating number of control volumes to be computed on each process at every time step. This inevitably creates load imbalance which needs to be addressed. Processors within the most reactive regions will have many clusters and need to integrate a high number of source terms. Processors within non-reactive regions will have few clusters and need to integrate a low number of source terms. The dynamic scheduling strategy introduced earlier in Sec. 4.2 allows to fully solve this concern.

Parallel cluster creation

Clusters are normally created within each process. A parallel clustering approach has been investigated, but the expected performance gain is low, parallel overheads are high and code complexity is greatly increased. The potential improvement is minimal because parallel clustering optimizes only the residual cost after process-local clustering. Furthermore, processes are distributed across different spatial regions, each involving potentially distinct chemical processes, which naturally reduces cluster overlap and diminishes the potential benefits of parallelization.
Parallel clustering also introduces new constraints:

- Clustering dimensions need to be fully consistent between all processes. Therefore PCA needs to be performed globally, which drastically increases its overhead.
- Finding parallel overlapping clusters requires a parallel distributed memory sorting algorithm. Implementation of such an algorithm has been performed to demonstrate its scaling capabilities. Radix sort is used with the Most Significant Digit (MSD) approach. MSD is advantageous as it involves less data transfer compared to the Least Significant Digit (LSD) approach and allows for quick subdivision into smaller recursive problems. This facilitates the construction of MPI sub-communicators and reduces the range of MPI communications. The algorithm's scaling performance is illustrated in Fig. 6.6. In order to be consistent with the local clustering overheads, RCT should not exceed a value of 1–10. RCT remains satisfactory only up to a few hundred processes, whereas Large-Eddy Simulation (LES) simulations can easily scale to a few thousand processes.

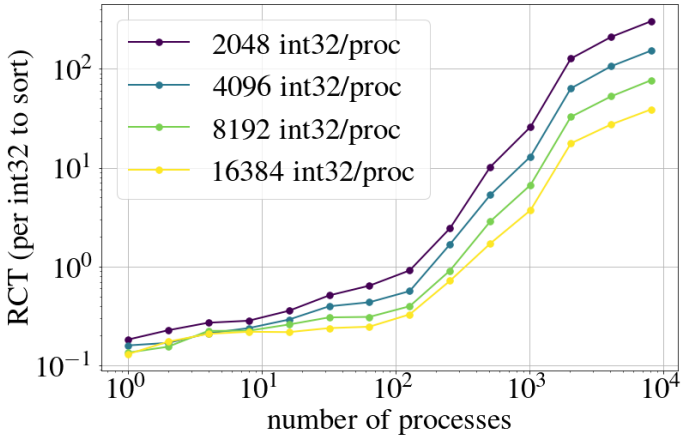


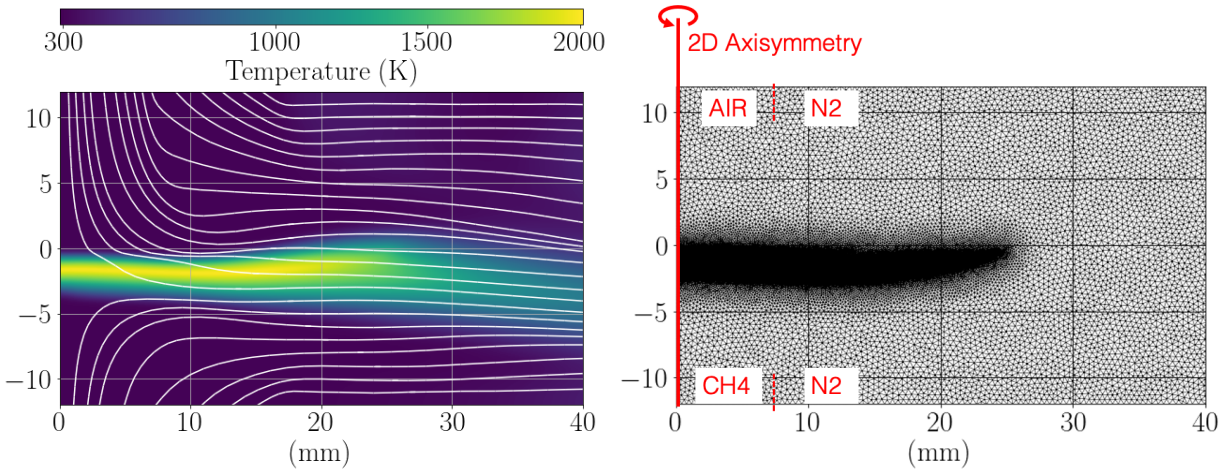
Figure 6.6: Weak scaling of parallel MSD radix sort implementation

- Once the parallel clusters are established, additional load-balancing is required to allocate the computation of each cluster to the appropriate process.
- Retrieving the source term must be performed in parallel, leading to an all-to-all communication pattern, which can be costly.

In conclusion, fully parallel clustering is likely to incur overheads that outweigh the potential benefits. A multi-level approach, such as clustering within smaller sub-groups of processes, similar to the dynamic scheduling, could be considered. However, the benefits do not justify the increased implementation complexity.

6.6 Validation

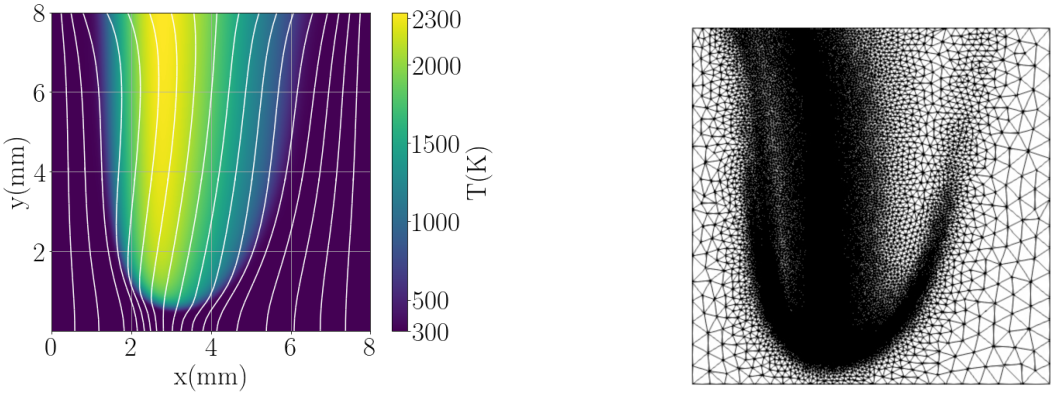
6.6.1 Validation cases



(a) Temperature field in the full simulation domain. Velocity streamlines are represented in white.

(b) Mesh made of 50'000 elements with smallest elements size of 60 microns.

Figure 6.7: CH4/air Counter-flow diffusion flame



(a) Temperature field in the full simulation domain. Velocity streamlines are represented in white.

(b) Triple flame mesh, made of 50'000 elements with smallest elements size of 20 microns.

Figure 6.8: H2/Air Triple flame

Two reference cases are used to show the performance improvement attained by using clustering. A H2/air triple flame is used with the same set-up as in Sec. 5, Fig. 6.8 shows the triple flame and associated mesh. Additionally, a counter-flow CH4/air diffusion flame is used to show that results are consistent independently of the underlying chemistry. This configuration is illustrated in Fig. 6.7. A 2D asymmetric variable density solver is used. Two counter-flow jets, of radius 7.5mm, separated by 24mm are considered. Air and CH4 are injected respectively with an initial velocity of 1m.s^{-1}. Both jets are surrounded by a N2 co-flow with an initial velocity of 0.24m.s^{-1}, to avoid accumulation of burnt gases. Chemistry is solved using a reduced scheme of 24 species and 132 reaction generated with ARCANE [93]. Meshes of both flames are created based of the SSRRR

criterion introduced in the previous chapter. Key information of both configurations is summed up in Tab. 6.2.

	Counter-flow flame	Triple flame
Fuel/Ox	CH4/air	H2/air
Scheme	In-house reduction	San Diego [149]
Number of species	24	21
Number of reaction	132	64
Number of elements	50'000	50'000
Smallest element	60 μm	20 μm
Largest element	400 μm	400 μm

Table 6.2: Chemistry and mesh comparison for the counter-flow and triple flame

6.6.2 Results

Error and performance measurement

Error is measured on instantaneous quantities produced by the reactors: the Heat Release Rate (HRR) and the species source terms $\dot{\omega}_k$ related by:

$$HRR = \sum_{k=1}^{N_{species}} \dot{\omega}_k H^0_{f,k} \,, \tag{6.36}$$

with $H^0_{f,k}$ the standard enthalpy of formation of species k. The analysis is focused on HRR here but the same conclusions are obtained for individual species. Within a solver iteration, reactors are solved twice, with and without clustering. Relative error is obtained based on the integrated difference:

$$E = \frac{\sum_{i=1}^{N_{\text{elem}}} |HRR_{\text{cluster},i} - HRR_{\text{ref},i}| \, V_i}{\sum_{i=1}^{N_{\text{elem}}} |HRR_{\text{ref},i}| \, V_i} \,. \tag{6.37}$$

This error tends to zero when the size of the clusters is reduced.
Performance is assessed based on the Reduced Computational Time (RCT):

$$RCT = \frac{\text{Wall clock time}(\mu s).N_{\text{cores}}}{N_{\text{elements}}.N_{\text{iterations}}} \,. \tag{6.38}$$

Only the cost related to source term computation is considered. Results are presented as the relation of performance to induced error. An example is given in Fig 6.9 to ease the interpretation of the results. For each clustering parametrization, a curve is given, showing the dependency of the results on the cluster resolution, i.e. the tolerance to consider nodes similar. When cluster resolution is increased, more reactors are solved, which leads to lower error but increased cost. This cost which converges to the reference cost without clustering plus clustering overhead. The optimal cluster resolution is subjective, depending on the tolerated induced error by the user. A good practice is to keep the error low or of the same order than other sources of error in finite-rate chemistry. The other sources of error are mainly due to scheme reduction, stiff integration,

splitting and finite-volume numerics for transport and diffusion. A tolerance of 1% to 10% error is a reasonable choice.

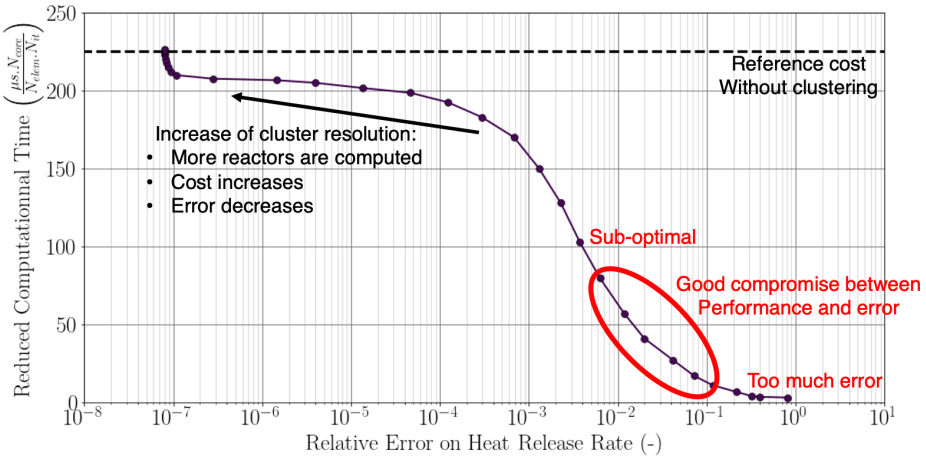


Figure 6.9: Example of performance to error curve with clustering. Figure is annotated to ease the interpretation of future figures.

Principal Component Analysis

Principal component analysis is used to minimize the number of required cluster dimensions. PCA is performed with PETSc [197] using *dgesvd()*. The four first Principal Components are given for the 2D reference flames. A physical interpretation of the features stressed by the PCs is done:

- **Counter-flow diffusive CH_4/air flame** (Fig 6.10): The first PC identifies the fuel rich areas of the flow. The second PC is a progress variable. The third PC is related to the mixing of the N_2 co-flow used to stabilize the flame. The fourth PC corresponds to the reactive flame front based on the presence of radicals.
- **H_2/air triple flame** (Fig 6.11): The first two PCs are a rotation of the progress variable and the fuel-air equivalence ratio. The first PC prioritizes the rich zones and the second PC the lean zones. The third and fourth PC identify the inner flame branch and insist on the lean and rich sides, respectively.

Variance of the Principle Components is summarised in Tab. 6.3.

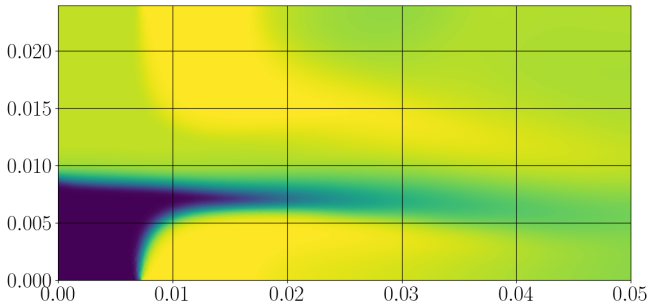
	Variance	Cumulated Variance
PC1	0.8049	0.8049
PC2	0.1233	0.9282
PC3	0.0701	0.9983
PC4	0.0014	0.9997

(a) CH_4/air counter-flow diffusion flame.

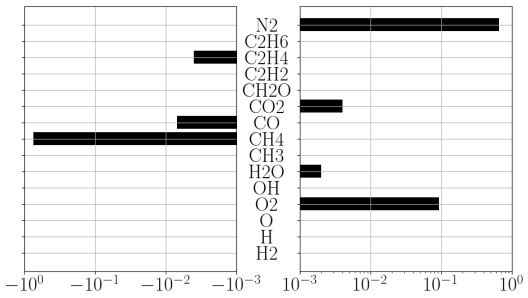
	Variance	Cumulated Variance
PC1	0.7526	0.7526
PC2	0.2454	0.9980
PC3	0.0015	0.9995
PC4	0.0004	0.9999

(b) H_2/air triple flame.

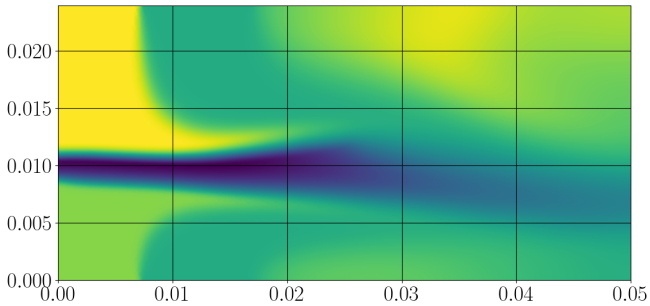
Table 6.3: Cumulated variance of the four first PCs of the reference flames.



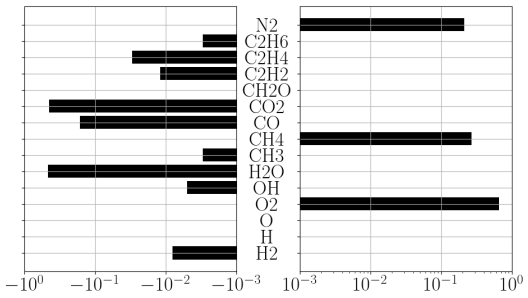
(a) PC1 field.



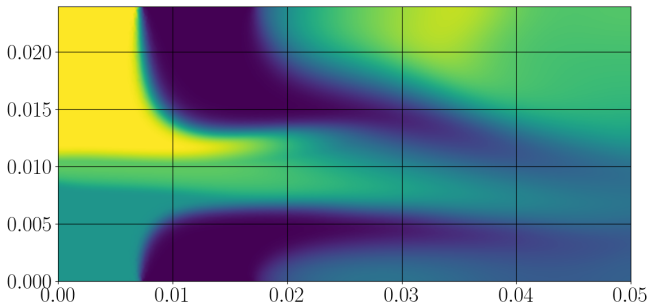
(b) PC1 coefficients.



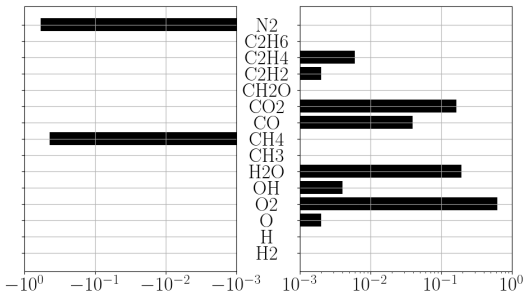
(c) PC2 field.



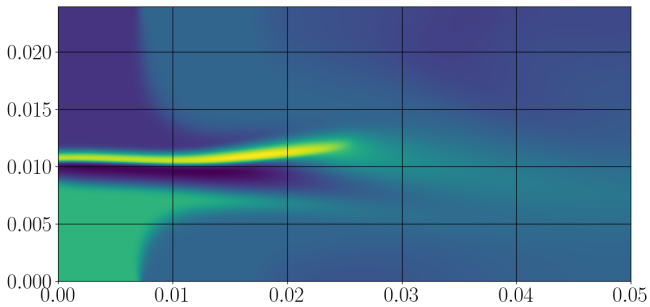
(d) PC2 coefficients.



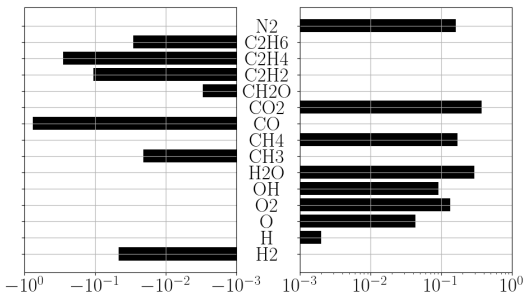
(e) PC3 field.



(f) PC3 coefficients.

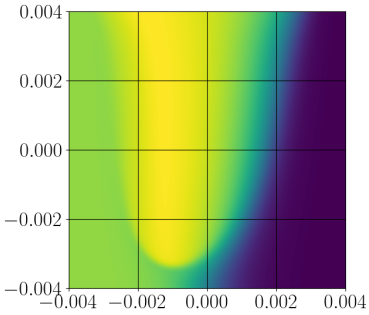


(g) PC4 field.

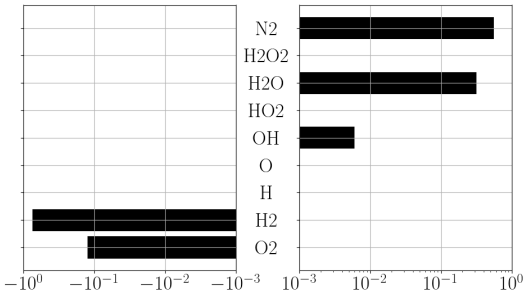


(h) PC4 coefficients.

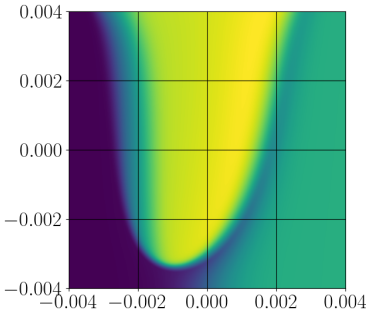
Figure 6.10: Principal Component fields of the counter-flow diffusion flame.



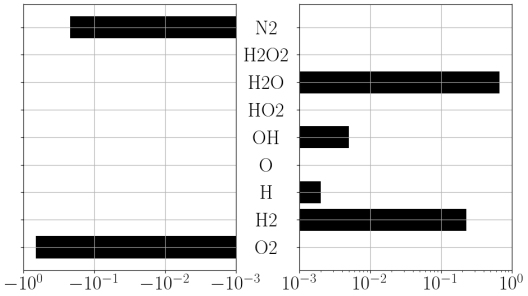
(a) PC1 field.



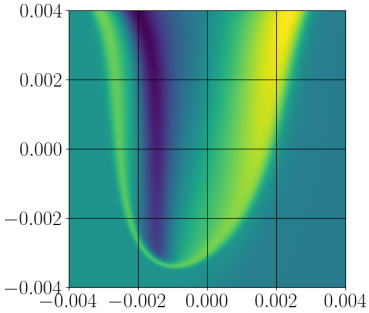
(b) PC1 coefficients.



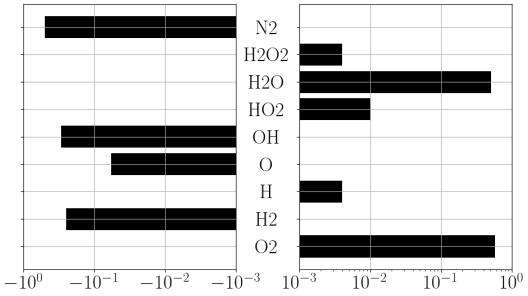
(c) PC2 field.



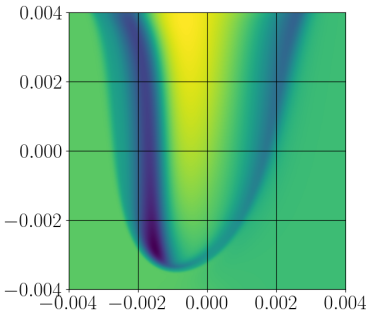
(d) PC2 coefficients.



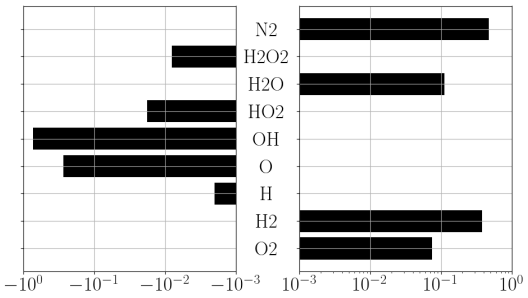
(e) PC3 field.



(f) PC3 coefficients.



(g) PC4 field.

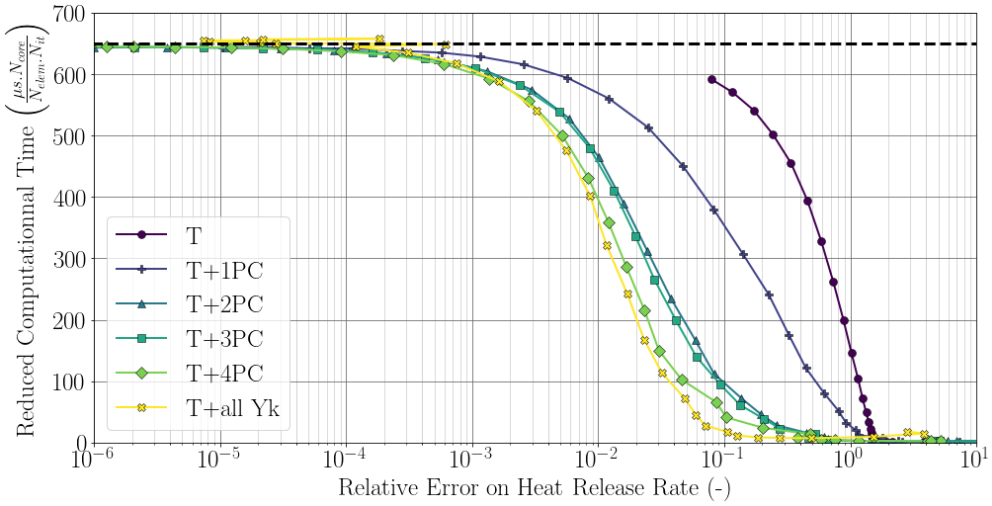


(h) PC4 coefficients.

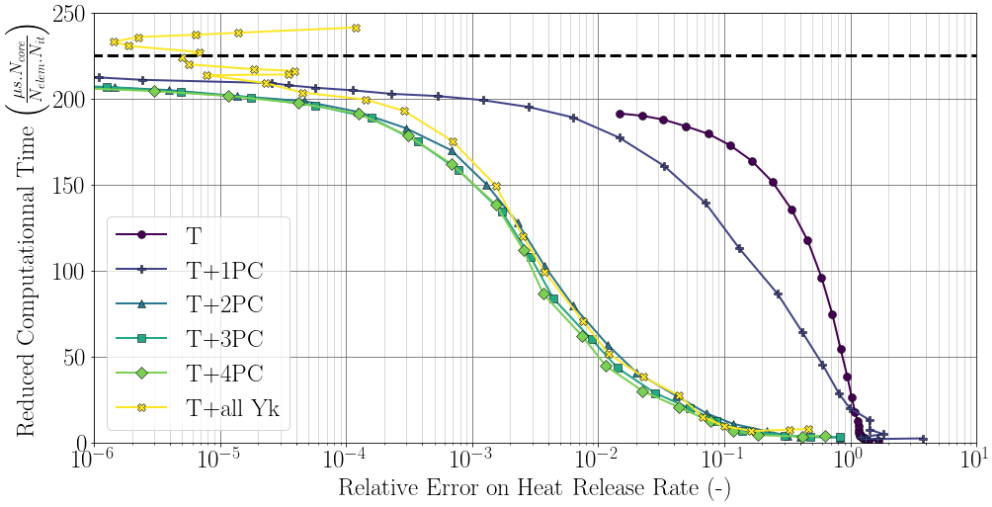
Figure 6.11: Principal Component fields of the triple flame.

Clustering with PCA

The effectiveness of using PCA for clustering dimensions is evaluated in Fig.6.12.



(a) Counter-flow diffusion flame.



(b) Triple flame.

Figure 6.12: RCT of source term computation depending on the number of Principal Components. Dashed line represents the reference cost without DCC.

When considering either temperature alone (T) or temperature combined with the first Principal Component (T+1PC), the resulting representation of the reactors is inadequate, leading to significant errors for both cases. In contrast, the use of two or more Principal Components (PCs) yields accurate results for the triple flame, while at least four PCs are necessary for the counter-flow flame to achieve good accuracy. This observation aligns well with the cumulative variance of the PCs presented in Tab.6.3 , which shows that the variance for the triple flame converges to 1 more rapidly. Utilizing all species in the clustering process yields results similar to those obtained with a well-chosen set of PCs. This is because PCA does not enhance the quality of the clustering itself; rather, it is intended to reduce its computational overhead. On the triple flame with 4 PC, the clus-

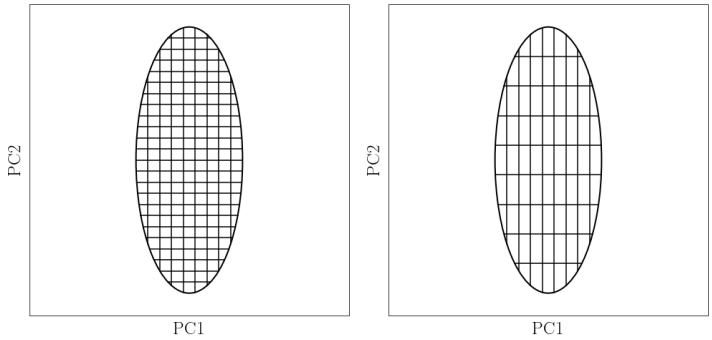
tering algorithm overhead is 1.5 RCT against 12 RCT for all species. However, as the complexity of the kinetic scheme increases and more species are involved, PCA may become more impactful. Additionally, the ability to perform clustering using PCA paves the way for integration with other methods, such as Principal Component Transport [171, 198].

Relative cluster dimension resolution

When several Principal Components are retained, each one requires a clustering resolution. The relative resolution between the PCs has a strong impact on efficiency. Two main approaches are considered:

- **Uniform spacing:** PCs are a simple rotation of composition in the phase space and still share a common unit when considering mass fractions. Thus, a uniform spacing of mass fraction can be done. This is represented in Fig. 6.13a. Note that PCA allows negative weights, therefore data may range from [-1,1] instead of [0,1] for standard mass fractions.
- **Uniform number of clusters:** Each PC can be considered as an independent feature of the reactive flow. Each of these features should be resolved with the same number of clusters. This is represented in Fig. 6.13b.

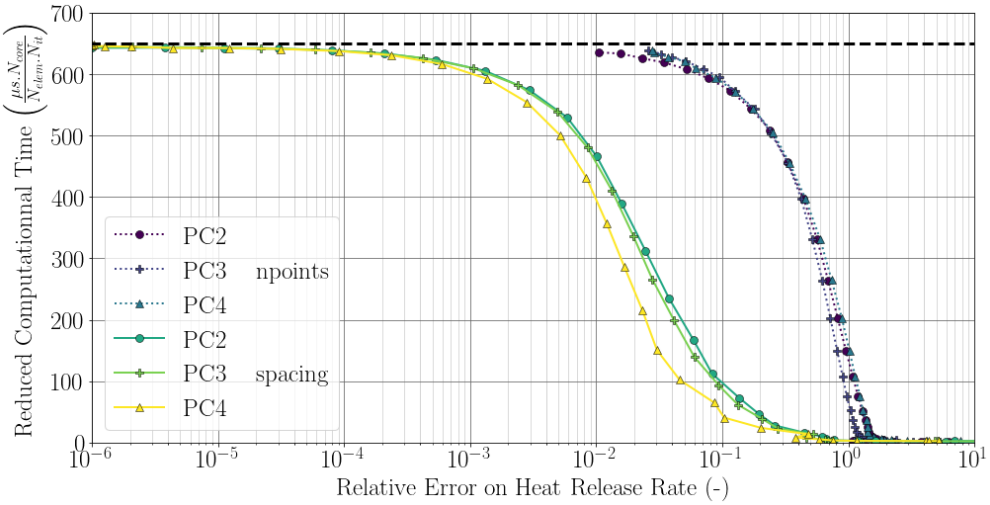
Results on reference flames are given in Fig. 6.14. In all cases, uniform spacing in phase space yields significantly better performance. This can be attributed to the fact that maintaining the same number of clusters along a minor PCA dimension imposes a high resolution on minor species. This increased resolution requires more source term integrations to accurately resolve the minor species. However, accurately resolving the minor species does not substantially improve the accuracy of the global chemistry integration. Note that is analogous to the debate over the relative scaling of species for PCA, where all the scaling methods proposed in Tab. 6.1 tend to place greater emphasis on minor species.



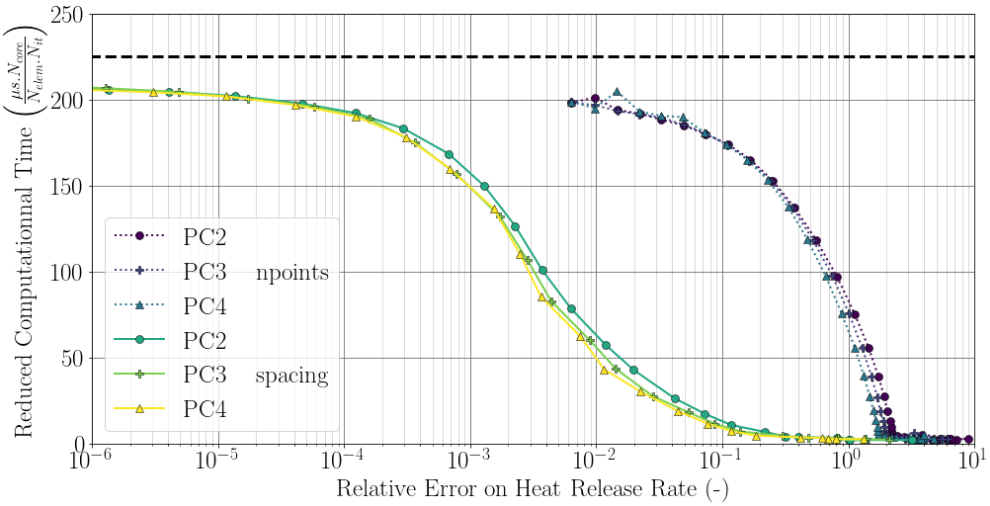
(a) Uniform spacing

(b) Uniform number of clusters

Figure 6.13: Relative resolutions of cluster dimensions.



(a) Counter-flow diffusion flame.

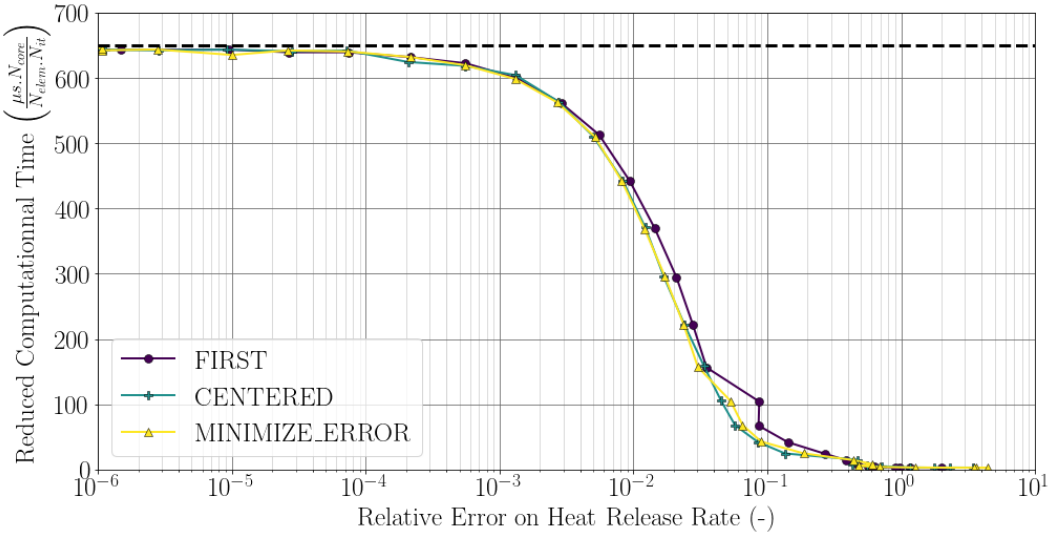


(b) Triple flame.

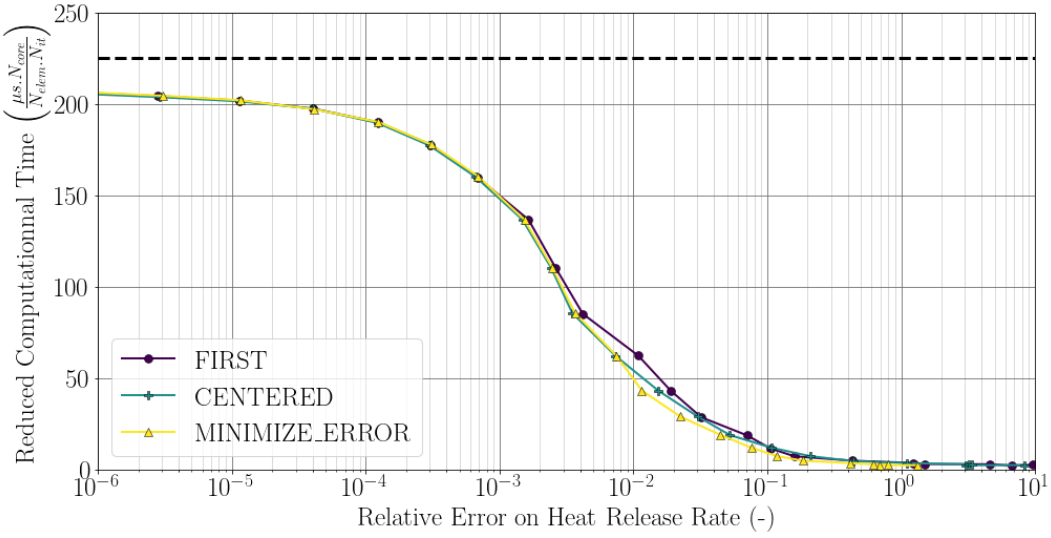
Figure 6.14: Impact of the relative cluster dimension resolution on performance.

Selection of computed node

Once all control volumes within a cluster have been identified, the next step is to determine which of these control volumes should be selected for computation. Sec. 6.3.3 proposes various approaches for this selection. One method is to choose the most central control volume, which is likely to best represent the cluster. Another approach is to select the control volume that most accurately represents the overall population of the cluster. Additionally, the first node encountered can be chosen, which essentially amounts to a random selection. Impact of this choice is shown in Fig. 6.15. Simply selecting the first encountered control volume has worse performance than the other two methods which are equivalent.



(a) Counter-flow diffusion flame.

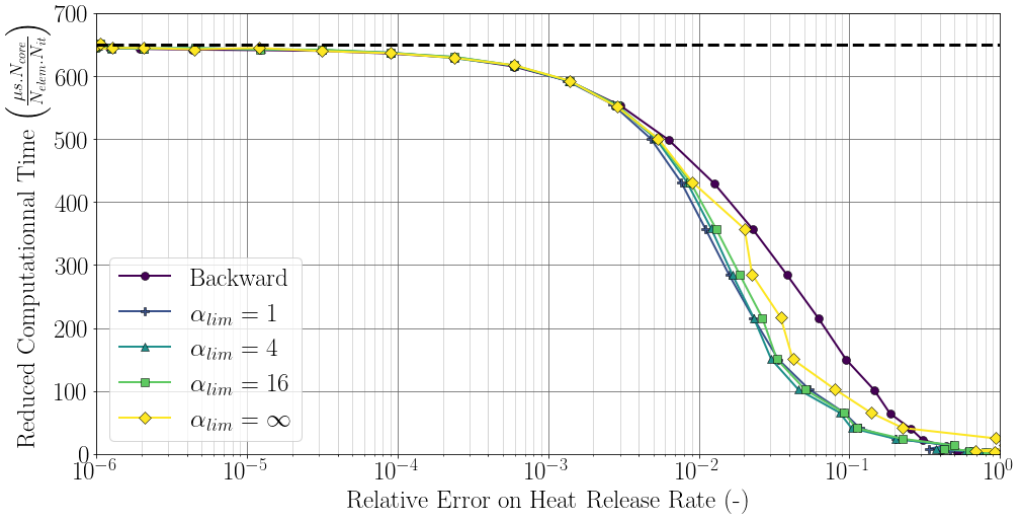


(b) Triple flame.

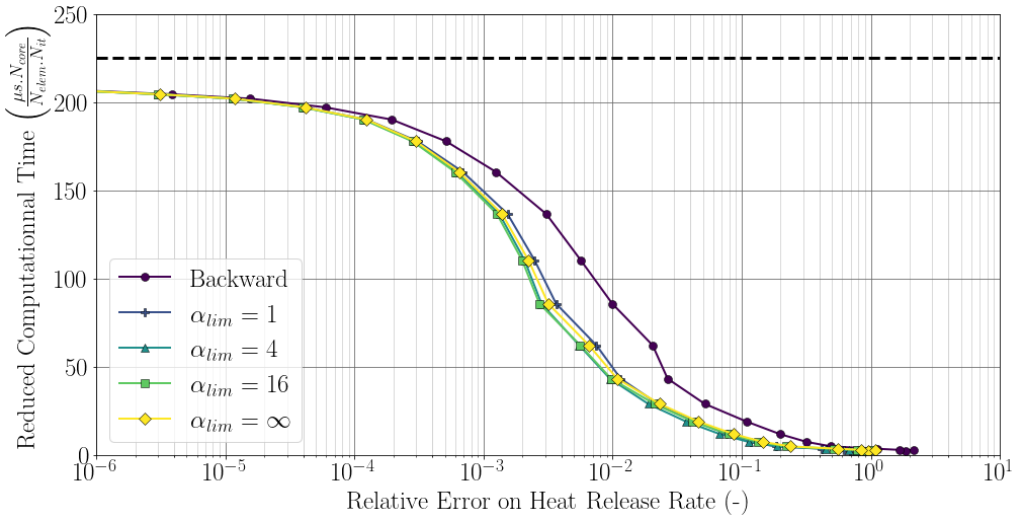
Figure 6.15: Impact of the selection process of the control volume to be integrated.

Jacobian-free mapping

Performance of the Jacobian-free mapping is given in Fig. 6.16. At all times, Jacobian-free mapping performs better than the reference backward mapping with error decreases up to 75% at equivalent RCT for both cases. α_{lim} refers to the extrapolation limitation introduced in Eq. 6.35. Without any limitation ($\alpha_{lim} = \infty$) error is reintroduced compared to pure interpolation ($\alpha_{lim} = 1$). However, some extrapolation seems to be beneficial in the triple flame.



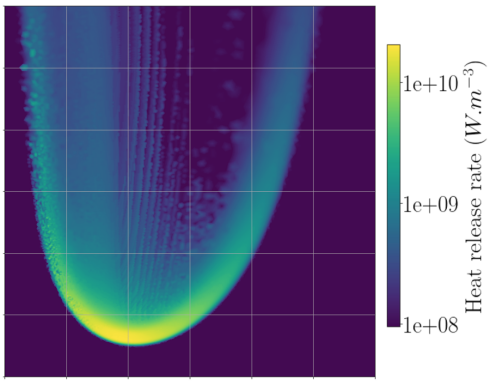
(a) Counter-flow diffusion flame.



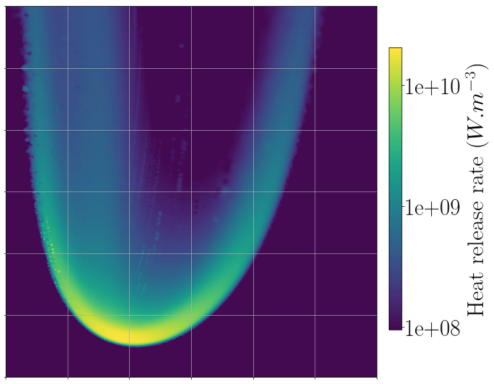
(b) Triple flame.

Figure 6.16: Impact of the mapping method on error.

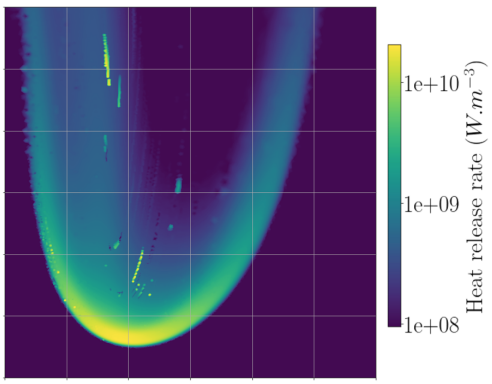
The impact of the mapping method can be seen on the heat release rate fields of the triple flame shown in Fig. 6.17. While $\alpha_{\text{lim}} = 4$ provides a smoother and more realistic solution than backward mapping, using extrapolation without a limiter can create extremely sharp local errors. Fig. 6.18 also stresses the need for a limiter by showing that a few elements can reach prohibitive values of α, reaching a magnitude of over 1000.



(a) Backward



(b) $\alpha_{\text{lim}} = 4$



(c) $\alpha_{\text{lim}} = \infty$

Figure 6.17: triple flame HRR field at α at RCT=70.

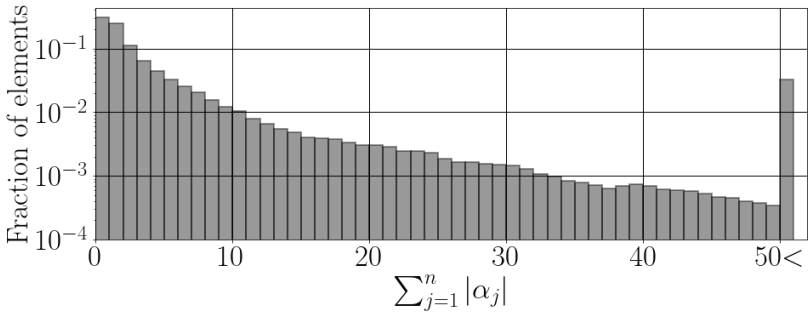


Figure 6.18: Values of α per mapped element on the triple flame at RCT=70 (logscale).

Overhead induced by the DCC remains less than 1% of the initial source term cost. Largest cost is associated with PCA. It should be noted that this is the cost if PCA was to be performed at every single iteration, which is not needed in most combustion cases. The cluster creation cost is negligible as it mainly relies on radix sort, which is very efficient on integers. Mapping cost consists mainly in solving the ridge regression while creation of connectivity is negligible.

6.7 Application of DCC on the CRSB

Clustering in now performed on the CRSB. Prior cases were computed on a single process whereas the CRSB requires a few thousand processes. The AMR procedure from Chapter 5 is kept active during the clustering process. It should be noted that this increases the percentage of control volumes located in highly reactive regions and thus decreases the efficiency of clustering as less chemically inert control volumes can be grouped together. As illustrated in Fig. 6.19, the effectiveness of clustering is influenced by the number of control volumes assigned to each processor, thus the total number of processes used. Increasing the number of control volumes per core is advantageous as more are clustered together.

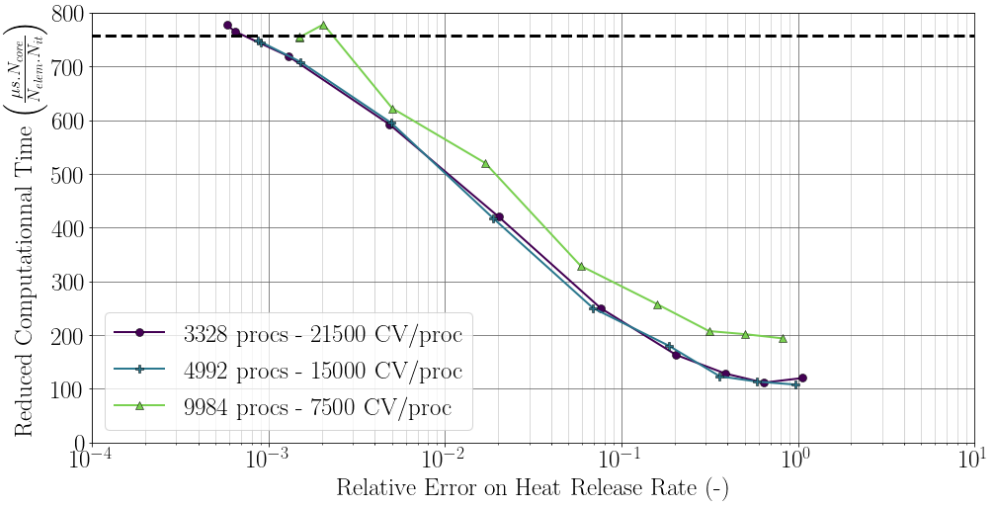
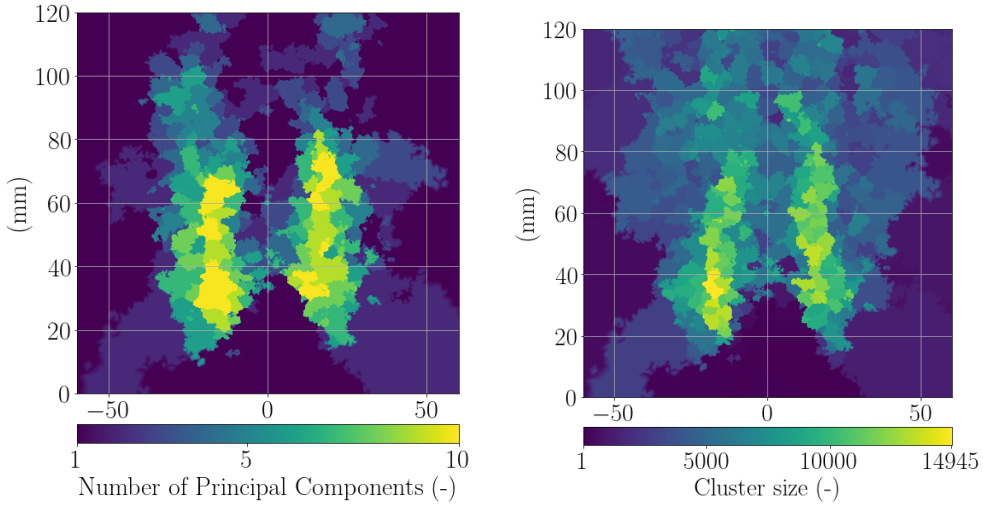


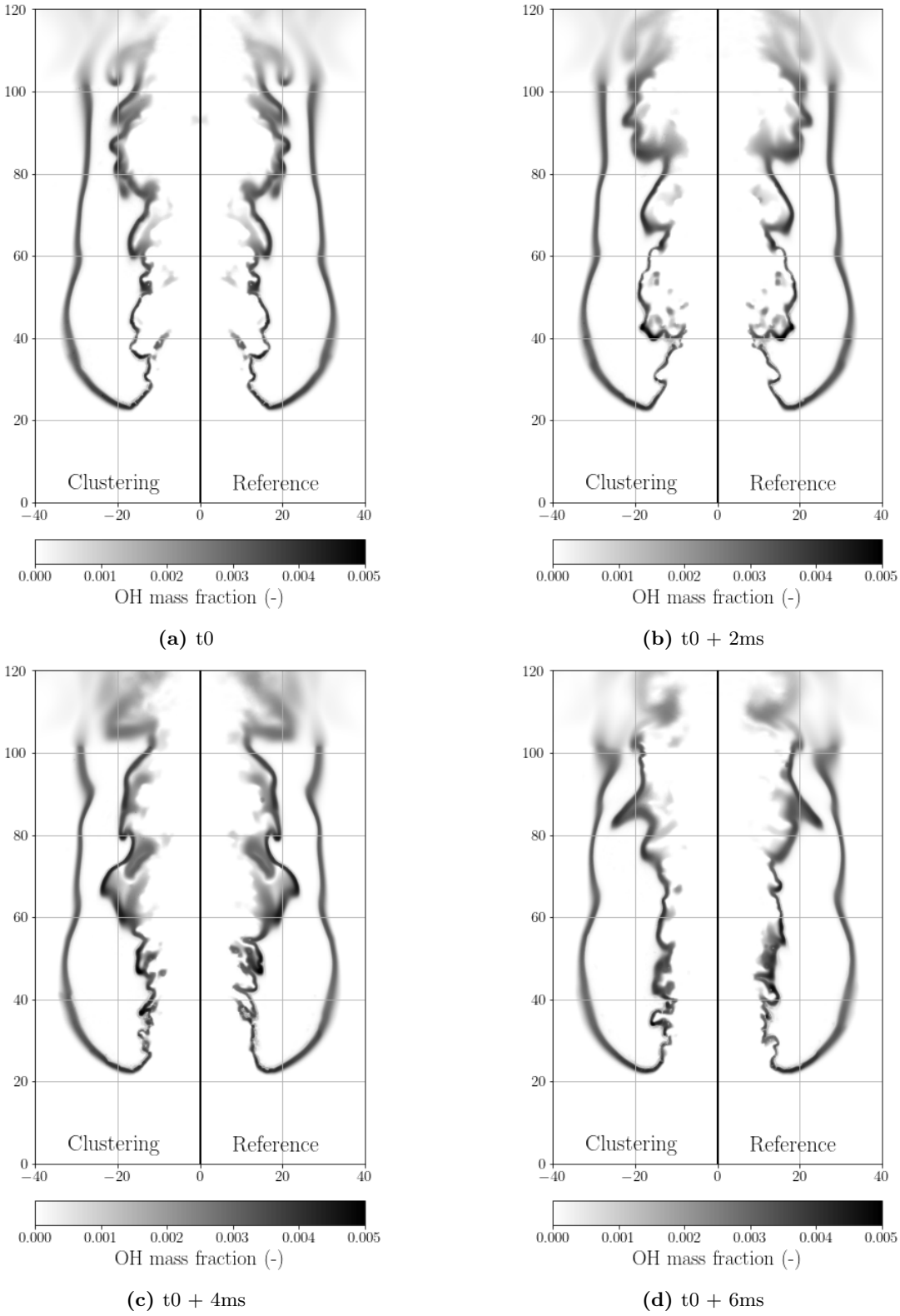
Figure 6.19: Impact of the number of processes on performance.



(a) Number clustering dimensions based on PCA on each process

(b) Number of computed control volumes on each process

Figure 6.20: Disparity of clustering behavior between processes, with 4992 processes at RCT = 250.



(a) t0

(b) t0 + 2ms

(c) t0 + 4ms

(d) t0 + 6ms

Figure 6.21: Comparison between a clustering and a reference run, starting from the same initial state over one re-circulation period at RCT = 250 on 4992 procs.

The number of PCA clustering dimensions is adjusted for each process to ensure that the cumulative variance of the principal components exceeds 0.9999. As a result shown in Fig. 6.20, a higher number of dimensions is required within the inner reaction zone (IRZ), where the reactivity is most intense and a single one is sufficient in unburnt gases. Within the IRZ about 90% of the control volumes are still integrated when source term cost is reduced by two thirds. The impact on the flame shape is shown in Fig. 6.21. Introduction of the clustering process doesn't introduce any disturbance that directly disrupts the flame shape and properties. The case with clustering and the reference case slowly diverge over time due to the chaotic nature of the turbulent flame. This divergence is enhanced by the remeshing process which is very sensible to slight changes in requested metric, which here depends on the reactive flow. While already saving two thirds of the cost, even more could likely be saved without notable degradation of the solution, however it is hard to assess whether the introduced error will cause significant changes of the flame behavior.

6.8 Conclusion

In this chapter, the implementation of dynamic cell clustering (DCC) for chemical source terms in YALES2 was investigated, with a particular focus on introducing a novel Jacobian-Free mapping method. This approach was evaluated on both a 2D H2/air triple flame and a 2D CH4/air counter-flow diffusion flame. Under realistic conditions, the Jacobian-Free mapping method demonstrated a significant reduction in error, achieving an improvement of 75%. The DCC technique was further applied to the CRSB, resulting in a two-thirds reduction in computational cost for source term calculations, without any noticeable impact on flame dynamics. This underscores the potential of the method to enhance computational efficiency while maintaining accuracy in complex combustion simulations.

CHAPTER 7

Conclusions and perspectives

This chapter summarizes the work conducted throughout the thesis and highlights the key optimization milestones achieved. Additionally, it discusses potential future optimizations that could further enhance LES spray burner simulations.

Contents

7.1 Optimization of two-phase reactive flows

This thesis is a contribution to reducing the cost of Large-Eddy Simulation (LES) in two-phase reactive flow simulations by implementing three key strategies: Load Balancing, Adaptive Mesh Refinement (AMR), and Dynamic Cell Clustering (DCC). Novel methodologies have been derived, implemented and assessed for each of those methods in YALES2. Validation has been carried out on small 2D cases then the methodology has been demonstrated on the CRSB.

7.1.1 Euler-Lagrange load balancing

Load balancing in Euler-Lagrange applications has been thoroughly evaluated. Traditional Eulerian load balancing proves inadequate, as Lagrangian particles tend to concentrate on specific processes, leading to parallel inefficiencies. To address this issue, a new load-balancing method tailored for Euler-Lagrange applications has been introduced. This approach effectively balances both the Eulerian mesh elements and the Lagrangian particles simultaneously. The process begins by balancing the Eulerian load as usual. Subsequently, the Lagrangian load is refined while maintaining an even distribution of the Eulerian load. This dual-stage balancing strategy ensures that the

partitioning remains close to ideal for the Eulerian mesh elements while optimizing the distribution of Lagrangian particles as much as possible. In large-scale spray simulations, this method has achieved cost reductions ranging on average from 35% to 70%, depending on the configuration.

7.1.2 Adaptive Mesh Refinement

Adaptive Mesh Refinement has been studied in the context of reactive flows. Building on the existing AMR framework, a new feature-based adaptation criterion specifically designed for multi-regime reactive flows has been implemented. This criterion is based on the similarity of scales of resolved reaction rates, identifying regions where reaction rates require an important sub-grid-scale contribution. These identified regions are then refined. A comparative study was conducted using a H2/air triple flame, comparing the new criterion with more traditional approaches, such as gradient and Hessian approaches on enthalpy and species. The results demonstrate that the new criterion significantly outperforms the others, achieving comparable accuracy while requiring only half the mesh size in the best cases.

7.1.3 Dynamic Cell Clustering

Dynamic Cell Clustering (DCC) has been implemented from scratch in YALES2. In this approach, control volumes are clustered based on their chemical states. A single source term is then integrated for each cluster, and finally, this source term is mapped back to all control volumes within the cluster. However, this mapping procedure introduces some errors, as the integrated source term does not account for the subtle variations in the states of individual control volumes within the cluster. A new mapping method was developed to reduce the error introduced by DCC, without incurring significant additional computational cost. This Jacobian-free method leverages the connectivity between adjacent clusters to provide a linear estimate of the source term's evolution in phase space, similar to how a Jacobian would operate. Validation of this method on both a triple flame and a counter-flow diffusion flame demonstrated a reduction in error of approximately 75% under realistic DCC operation conditions.

7.1.4 Resulting performance of the CRSB

Initially, the simulation cost for 1 ms of physical time for the CRSB spray burner was estimated at 27,000 CPU hours. To achieve meaningful statistical convergence, a simulation time of approximately 100 ms is required, resulting in a total computational cost of 2.7 million CPU hours. The implementation of Adaptive Mesh Refinement (AMR) has significantly reduced the mesh size from 520 million elements to 290 million, while simultaneously improving the accuracy of the captured physics. This reduction in mesh size led to a 37.4% decrease in the overall computational cost. Dynamic Cell Clustering, reduced the cost of source term integration by two-thirds. This improvement contributed an additional 15.7% reduction in the total cost. Furthermore, the application of Euler-Lagrange load balancing provided a further reduction in computational cost, which varied depending on the initial load balance, but generally contributed a few more percentage points of savings. Overall, these combined strategies have resulted in a reduction of the total simulation cost for the CRSB by more than 50%. This allows to save 1.35 millions of CPU hour per convergence

of the CRSB, making it more affordable. These advances will strongly benefit to future studies on the combustion of liquid sustainable aviation fuels.

7.1.5 Optimization perspectives

Several optimization approaches have yet to be studied and incorporated into YALES2 if deemed viable. The key approaches that should be considered are the following:

- Dynamic Adaptive Chemistry (DAC) dynamically reduces the kinetic mechanism on the fly, tailoring it to represent only the combustion regime present in a specific location. This approach is particularly effective in multi-regime reactive flows, where the underlying mechanism must account for all possible regimes, necessitating a large set of species and reactions. However, in any given local region, the flow can be reduced to a specific regime that requires significantly fewer species and reactions. By adapting the chemistry to these local conditions, DAC significantly enhances computational efficiency. Due to the multi-regime nature, spray burners are likely to benefit from this approach and it could enable the use of larger kinetic scheme instead of analytically reduced schemes.

- Source term integration is currently performed on CPU only. GPU approaches are likely viable especially given that the stiff integrator third party library, CVODE, offers increases GPU compatibility over the recent years. This is further supported by the dynamic scheduler that has shown that it is worth it to outsource source term computation to other CPUs as the ratio of data to transfer over the number of operation to solve the system is low. Thus, data transfer to GPU would not be prohibitive.

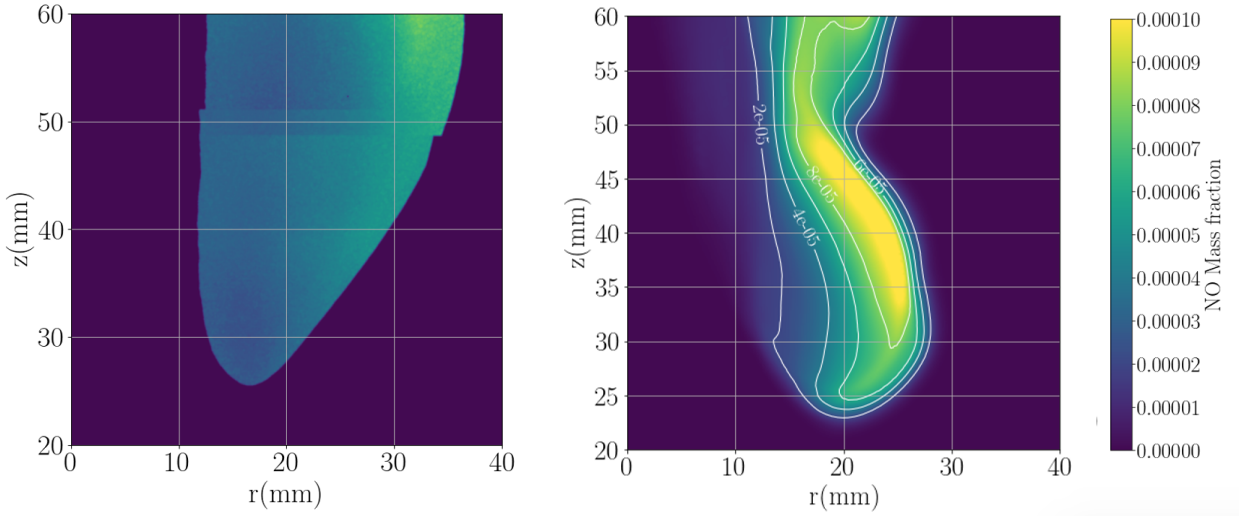
- Implicit diffusion of species is a significant computational expense in large-scale reactive flow simulations, particularly as element sizes decrease due to quadratic scaling of the diffusion timestep. The Exponential Propagation Method (EPM) [199, 200] offers a more efficient alternative by leveraging matrix exponentials to integrate the linear diffusion terms directly, allowing for larger time steps and improved stability. EPM's ability to decouple stiff linear terms from non-linear dynamics facilitates adaptive time-stepping, reducing the overall computational cost, especially in high-resolution simulations where diffusion dominates. This makes EPM a compelling choice over traditional implicit methods in handling diffusion efficiently.

- Adaptive Mesh Refinement encounters large parallel overhead on simulation that exceed 10'000 processes. At this scale, overheads become too great for Dynamic AMR to be viable. This is partially due to the sequential operation of METIS where the master process splits the mesh into number of parts equivalent to the number of processes. One could envisage a multi-level METIS approach, splitting the mesh into two part repeated involving more processes at each split until reaching the desired amount of mesh partitions. Second, relying on a third party library for remeshing, MMG, limits parallel implementation of the AMR process. Designing a fully parallel remesher with more parallel remeshing kernels may be of interest.

7.2 Perspectives of the CRSB simulation

In addition to optimization efforts, the CRSB simulation could be enhanced to better represent physical phenomena. Some observations made during the thesis regarding the modeling of the CRSB are described here:

- The eighth Turbulent Spray Combustion (TCS8) workshop focused on simulating NO pollutants in the CRSB, which proved challenging for all participants. A significant difficulty was identifying an appropriate n-heptane scheme that adequately incorporates NOx chemistry. The Polimi-Creck [153] scheme was used as a foundation for tabulation and reduction but resulted in excessive NO production. The experimental and simulated fields using YALES2 are given in Fig. 7.1 These findings are also corroborated by recent work [201].



(a) Experiment

(b) LES Simulation, performed before the use of AMR improving flame shape

Figure 7.1: NO fields of the CRSB

- Granulometry significantly influences flame topology, particularly with large inertial droplets dominating the Outer Reaction Zone. Accurately injecting a Lagrangian spray to match Phase Doppler Anemometry (PDA) measurements is a complex task. These experimental measurements are taken at various radial positions downstream of the injector, where the air co-flow has already altered the droplet dynamics. Deriving the appropriate poly-dispersed spray distributions for Lagrangian injection to align with these downstream measurements is not straightforward. Typically, a simple diameter Probability Density Function (PDF) is used, assuming uniform distribution across the spray angle and velocity. However, this approach overlooks potential correlations between droplet diameter, injection angle, and velocity. Furthermore, most validations rely on mean diameters (D_{10}, D_{32}) and velocities at the radial positions, neglecting the mass flow rate at these positions. Good radial averages and RMS mainly reflect the separation of inertial droplets from flow-carried droplets by the coflow, an effect that occurs regardless of the initial diameter distribution. An alternative to

the Lagrangian injection is to model the atomisation process to recover the full droplet dynamics. Purely Eulerian approaches involve resolving the atomization process and discretizing the resulting droplets directly on the computational mesh. This method has the advantage of not requiring any prior assumptions about the characteristics of the polydispersed spray, offering a more detailed and accurate representation of the atomization process. Currently Eulerian atomization simulation are studied in the CRSB [202, 203, 204]. However, despite these advantages, the computational cost of such an approach is currently prohibitive on a full spray burner simulation given existing computational resources. To mitigate this limitation, hybrid methods have been developed that transform Eulerian droplets into Lagrangian particles based on a set of criteria [205, 206]. These transformation techniques enable the coupling of Euler atomization processes with Euler-Lagrange reactive flow simulations. This hybrid approach allows for a detailed simulation of the atomization process while taking advantage of the computational efficiency of Euler-Lagrange methods for tracking droplets and their interactions within the flow. By balancing accuracy with computational feasibility, this coupling represents a significant advancement in the simulation of reactive flows involving complex spray dynamics.

List of Tables

List of Figures

Bibliography

[1] D. S. Lee, D. W. Fahey, A. Skowron, M. R. Allen, U. Burkhardt, Q. Chen, S. J. Doherty, S. Freeman, P. M. Forster, J. Fuglestvedt, A. Gettelman, R. R. De León, L. L. Lim, M. T. Lund, R. J. Millar, B. Owen, J. E. Penner, G. Pitari, M. J. Prather, R. Sausen, and L. J. Wilcox, "The contribution of global aviation to anthropogenic climate forcing for 2000 to 2018," *Atmospheric Environment*, vol. 244, 1 2021. 9

[2] IPCC, *Global Warming of 1.5°C*. Cambridge University Press, 6 2022. 9

[3] K. Haustein, M. R. Allen, P. M. Forster, F. E. Otto, D. M. Mitchell, H. D. Matthews, and D. J. Frame, "A real-time Global Warming Index," *Scientific Reports*, vol. 7, 12 2017. 9

[4] C. P. Morice, J. J. Kennedy, N. A. Rayner, J. P. Winn, E. Hogan, R. E. Killick, T. J. Osborn, P. D. Jones, I. R. Simpson, and C. Morice, "An updated assessment of near-surface temperature change from 1850: the HadCRUT5 dataset," *JGR Atmosheres*, 2020. 9

[5] M Klöwer, M R Allen, D S Lee, S R Proud, L Gallagher, and A Skowron, "Quantifying aviation's contribution to global warming," *Environmental research letters*, vol. 16, 2021. 9, 10, 155

[6] M. Cain, J. Lynch, M. R. Allen, J. S. Fuglestvedt, D. J. Frame, and A. H. Macey, "Improved calculation of warming-equivalent emissions for short-lived climate pollutants," *npj Climate and Atmospheric Science*, vol. 2, 12 2019. 9

[7] European Union Aviation Safety Agency (EASA), "European Aviation Environmental Report," tech. rep., European Union Aviation Safety Agency (EASA), 2022. 11, 12, 155

[8] International Air Transport Association (IATA), "Aircraft Technology Net Zero Roadmap 2 Aircraft Technology Net Zero Roadmap," tech. rep., International Air Transport Association (IATA), 2022. 11

[9] M. Prewitz, J. Schwärzer, and A. Bardenhagen, "Potential analysis of hydrogen storage systems in aircraft design," *International Journal of Hydrogen Energy*, vol. 48, pp. 25538–25548, 7 2023. 11

[10] D. O. Allison, "Polynomial approximations of thermodynamic properties of arbi- trary gas mixtures over wide pressure and density ranges," tech. rep., National Aeronautics and Space Administration, Washington D.C, 1972. 21

[11] N. Peters and J. Warnatz, *Numerical methods in laminar flame propagation.* Vieweg, 1982. 22

[12] L. Monchick and E. A. Mason, "Transport properties of polar gases," *The Journal of Chemical Physics*, vol. 35, no. 5, pp. 1676–1697, 1961. 22

[13] C. R. Wilke, "A viscosity equation for gas mixtures," *The Journal of Chemical Physics*, vol. 18, no. 4, pp. 517–519, 1950. 22

[14] R. S. Brokaw, "Approximate formulas for the viscosity and thermal conductivity of gas mixtures," *The Journal of Chemical Physics*, vol. 29, no. 2, pp. 391–397, 1958. 22

[15] F. A. Williams, *Combustion theory.* Benjamin Cummings, 1985. 22

[16] W. Sutherland, "The viscosity of gases and molecular force.," *Philosophical Magazine Series 5*, vol. 36, no. 223, pp. 507–531, 1893. 23

[17] C. F. Curtiss and J. O. Hirschfelder, "Transport properties of multicomponent gas mixtures," *The Journal of Chemical Physics*, vol. 17, no. 6, pp. 550–555, 1949. 23

[18] F. A. Lindemann, Svante Arrhenius, Irving Langmuir, N. R. Dhar, J. Perrin, and W. C. Lewis, "Discussion on "The Radiation Theory of Chemical Action."," *Transaction of the Faraday society*, vol. 17, pp. 598–606, 1922. 25

[19] R. G. Gilbert, K. Luther, and J. Troe, "Theory of thermal unimolecular reactions in the fall-off range - 2. Weak collision rate constants.," *Berichte der Bunsengesellschaft/Physical Chemistry Chemical Physics*, vol. 87, no. 2, pp. 169–177, 1983. 25

[20] W. Tsang and J. T. Herron, "Chemical Kinetic Data Base for Propellant Combustion I. Reactions Involving NO, N02, HNO, HN02, HeN and N20," *Journal of Physical and Chemical Reference Data*, vol. 20, no. 4, 1991. 25

[21] R. J. Kee, F. M. Rupley, and J. A. Miller, "Chemkin-II: A Fortran chemical kinetics package for the analysis of gas-phase chemical kinetics," tech. rep., Sandia National Laboratories, 1989. 25

[22] P. H. Stewart, C. W. Larson, and D. M. Golden, "Pressure and Temperature Dependence of Reactions Proceeding Via A Bound Complex. 2. Application to 2CH3 ˜ C2Hs + H," *Combustion and Flame*, vol. 75, pp. 25–31, 1989. 25

[23] J. Jarosinski, "The Thickness of Laminar Flames," *Combustion and Flame*, vol. 56, pp. 337–342, 1984. 26

[24] Y. B. Zeldovich, *The Theory of Combustion and Detonation.* N. N. Semenov, 1944. 26

[25] H. H. Chiu, H. Y. Kim, and E. J. Croke, "Internal group combustion of liquid droplets," in *19th Symposium on Combustion*, 1982. 29, 155

[26] K. Kuo, *Principles of Combustion*. John Wiley and Sons., 1986. 29, 155

[27] N. Chigier, "Group combustion models and laser diagnostic methods in sprays: a review," *Combustion and Flame*, vol. 51, pp. 127–139, 1983. 29, 155

[28] A. Umemura and S. Takamori, "Percolation theory for flame propagation in non- or less-volatile fuel spray: A conceptual analysis to group combustion excitation mechanism," *Combustion and Flame*, vol. 141, pp. 336–349, 6 2005. 29, 30, 155

[29] H. Chiu, R. Ahluwalia, and E. Croke, "Spray group combustion," in *16th Aerospace Sciences Meeting*, 1978. 30, 155

[30] C. K. Law, "Combustion in two-phase flows," in *Combustion Physics*, Cambridge University Press, 2006. 30, 155

[31] A. Verdier, J. Marrero Santiago, A. Vandel, S. Saengkaew, G. Cabot, G. Grehan, and B. Renou, "Experimental study of local flame structures and fuel droplet properties of a spray jet flame," *Proceedings of the Combustion Institute*, vol. 36, no. 2, pp. 2595–2602, 2017. 31, 85, 114, 155, 157

[32] A. Verdier, J. Marrero Santiago, A. Vandel, G. Godard, G. Cabot, and B. Renou, "Local extinction mechanisms analysis of spray jet flame using high speed diagnostics," *Combustion and Flame*, vol. 193, pp. 440–452, 7 2018. 31

[33] I. A. Mulla, G. Godard, G. Cabot, F. Grisch, and B. Renou, "Quantitative imaging of nitric oxide concentration in a turbulent n-heptane spray flame," *Combustion and Flame*, vol. 203, pp. 217–229, 5 2019. 31

[34] C. S. Vegad, S. Idlahcen, L. Huang, G. Cabot, B. Renou, B. Duret, J. Reveillon, and F.-X. Demoulin, "Planar two-photon fluorescence imaging of dense spray to estimate spray characteristics: application in pressure-swirl atomizers," *Atomization and Sprays*, pp. 1–20, 2023. 31

[35] A. Chatelier, B. Fiorina, V. Moureau, and N. Bertier, "Large Eddy Simulation of a Turbulent Spray Jet Flame Using Filtered Tabulated Chemistry," *Journal of Combustion*, vol. 2020, 2020. 31

[36] F. Shum-Kivan, J. Marrero Santiago, A. Verdier, E. Riber, B. Renou, G. Cabot, and B. Cuenot, "Experimental and numerical analysis of a turbulent spray flame structure," *Proceedings of the Combustion Institute*, vol. 36, no. 2, pp. 2567–2575, 2017. 31, 59

[37] J. Benajes, J. M. García-Oliver, J. M. Pastor, I. Olmeda, A. Both, and D. Mira, "Analysis of local extinction of a n-heptane spray flame using large-eddy simulation with tabulated chemistry," *Combustion and Flame*, vol. 235, 1 2022. 31

[38] W. Liang and C. K. Law, "Extended flammability limits of n-heptane/air mixtures with cool flames," *Combustion and Flame*, vol. 185, pp. 75–81, 2017. 33

[39] P. Sagaut, *Large Eddy Simulation for Incompressible Flows*. Springer, 1998. 41

[40] P. Moin, K. Squires, W. Cabot, and S. Lee, "A dynamic subgrid-scale model for compressible turbulence and scalar transport," *Physics of Fluids*, vol. 3, no. 11, pp. 2746–2757, 1991. 42, 59, 100

[41] U. P. Piomelli, T. A. Zang, C. G. Speziale, and M. Y. Hussaini, "On the Large-Eddy Simulation of transitional wall-bounded flows," tech. rep., ICASE, NASA Contractor Report, 1989. 42

[42] D. K. Lilly, "A proposed modification of the Germano subgrid-scale closure method," *Physics of Fluids A*, vol. 4, no. 3, pp. 633–635, 1992. 42

[43] M. Germano, U. Piomelli, P. Moin, and W. H. Cabot, "A dynamic subgrid-scale eddy viscosity model," *Physics of Fluids A*, vol. 3, no. 7, pp. 1760–1765, 1991. 42

[44] D. Carati, S. Ghosal, and P. Moin, "On the representation of backscatter in dynamic localization models," *Physics of Fluids*, vol. 7, no. 3, pp. 606–616, 1995. 43

[45] F. Nicoud and F. Ducros, "Subgrid-Scale Stress Modelling Based on the Square of the Velocity Gradient Tensor," *Flow, Turbulence and Combustion*, vol. 62, pp. 183–200, 1999. 43

[46] F. Nicoud, H. B. Toda, O. Cabrit, S. Bose, and J. Lee, "Using singular values to build a subgrid-scale model for large eddy simulations," *Physics of Fluids*, vol. 23, no. 8, p. 85106, 2011. 43

[47] P. E. DesJardin and S. H. Frankel, "Large eddy simulation of a nonpremixed reacting jet: Application and assessment of subgrid-scale combustion models," *Physics of Fluids*, vol. 10, no. 9, pp. 2298–2314, 1998. 44, 100

[48] A. Shamooni, A. Cuoci, T. Faravelli, and A. Sadiki, "Prediction of combustion and heat release rates in non-premixed syngas jet flames using finite-rate scale similarity based combustion models," *Energies*, vol. 11, 9 2018. 44, 100

[49] A. Shamooni, A. Cuoci, T. Faravelli, and A. Sadiki, "New Dynamic Scale Similarity Based Finite-Rate Combustion Models for LES and a priori DNS Assessment in Non-premixed Jet Flames with High Level of Local Extinction," *Flow, Turbulence and Combustion*, vol. 104, pp. 233–260, 1 2020. 44, 100

[50] O. Colin, F. Ducros, D. Veynante, and T. Poinsot, "A thickened flame model for large eddy simulations of turbulent premixed combustion," *Physics of Fluids*, vol. 12, no. 7, pp. 1843–1863, 2000. 44

[51] C. Meneveau and T. Poinsot, "Stretching and Quenching of Flamelets in Premixed Turbulent Combustion," *Combustion and Flame*, vol. 86, pp. 311–332, 1991. 45

[52] T. Poinsot, D. Veynante, and S. Candel, "Quenching processes and premixed turbulent combustion diagrams," *J. Fluid Mech*, 1991. 45

[53] F. Charlette, C. Meneveau, and D. Veynante, "A Power-Law Flame Wrinkling Model for LES of Premixed Turbulent Combustion Part II: Dynamic Formulation," *Combustion and Flame*, vol. 131, pp. 181–197, 2002. 45

[54] G. Wang, M. Boileau, and D. Veynante, "Implementation of a dynamic thickened flame model for large eddy simulations of turbulent premixed combustion," *Combustion and Flame*, vol. 158, pp. 2199–2213, 11 2011. 45

[55] A. R. Kerstein, W. T. Ashurst, and F. A. W'illiams, "Field equation for interface propagation in an unsteady homogeneous flow field," *Physical review*, vol. 37, no. 7, 1988. 45

[56] H. Pitsch, "A consistent level set formulation for large-eddy simulation of premixed turbulent combustion," *Combustion and Flame*, vol. 143, pp. 587–598, 12 2005. 45

[57] F. E. Marble and J. E. Broadwell, "The Coherent Flame Model for turbulent chemical reactions.," tech. rep., California Institute of Technology, 1977. 45

[58] S. Candel, D. Veynante, F. Lacas, E. Maistret, N. Darabiha, and T. E. Poinsot, "Coherent Flamelet Model: Application and recent extensions," in *Recent Advances in Combustion Modelling*, pp. 19–64, World Scientific, 1990. 45

[59] M. Boger, D. Veynante, H. Boughanem, and A. Trouvé, "Direct numerical simulation analysis of flame surface density concept for large eddy simulation of turbulent premixed combustion," in *27th Symposium (International) on Combustion*, pp. 917–925, 1998. 45

[60] S. B. Pope, "PDF methods for turbulent reactive flows," *Proo. Energy Combust. Sci*, vol. 11, p. 192, 1985. 46

[61] F. C. Lockwood and A. S. Naguib, "The Prediction of the Fluctuations in the Properties of Free, Round-Jet, Turbulent, Diffusion Flames," *Combustion and Flame*, vol. 24, p. 109, 1975. 46

[62] S. S. Girimaji, "Assumed β-pdf Model for Turbulent Mixing: Validation and Extension to Multiple Scalar Mixing," *Combustion Science and Technology*, vol. 78, pp. 177–196, 8 1991. 46

[63] C. W. Hirt and B. D. Nichols, "Volume of Fluid (VOF) Method for the Dynamics of Free Boundaries," *Journal of Computational Physics*, vol. 39, pp. 201–225, 1981. 46

[64] R. Scardovelli and S. Zaleski, "Direct numerical simulation of free-surface and interfacial flow," *Annu. Rev. Fluid Mech*, vol. 31, pp. 567–603, 1999. 46

[65] Sethian J, "A fast marching level set method for monotonically advancing fronts," *Applied Mathematics Communicated*, vol. 93, pp. 1591–1595, 1996. 46

[66] S. Osher and J. A. Sethian, "Fronts Propagating with Curvature-Dependent Speed: Algorithms Based on Hamilton-Jacobi Formulations," *Journal of Computational Physics*, vol. 79, pp. 12–49, 1988. 46

[67] O. A. Druzhinin and S. Elghobashi, "Direct numerical simulations of bubble-laden turbulent flows using the two-fluid formulation," *Physics of Fluids*, vol. 10, no. 3, pp. 685–697, 1998. 46

[68] J. Ferry and S. Balachandar, "A fast Eulerian method for disperse two-phase flow," *International Journal of Multiphase Flow*, vol. 27, pp. 1199–1226, 2001. 46

[69] M. Boileau, S. Pascaud, E. Riber, B. Cuenot, L. Gicquel, and T. Poinsot, "Investigation of two-fluid methods for Large Eddy Simulation of spray combustion in Gas Turbines," tech. rep., CERFACS, 2007. 46

[70] F. Laurent and M. Massot, "Multi-fluid modelling of laminar polydisperse spray flames: Origin, assumptions and comparison of sectional and sampling methods," *Combustion Theory and Modelling*, vol. 5, no. 4, pp. 537–572, 2001. 46

[71] S. Subramaniam, "Lagrangian-Eulerian methods for multiphase flows," *Progress in Energy and Combustion Science*, vol. 39, pp. 215–245, 4 2013. 46

[72] R. L. C. Flemmer and C. L. Banks, "On the Drag Coefficient of a Sphere," *Powder Technology*, vol. 48, pp. 217–221, 1986. 47

[73] L. Schiller and A. Naumann, "A Drag Coefficient Correlation," *Zeitschrift des Vereins Deutscher Ingenieure*, vol. 77, pp. 318–320, 1935. 47, 59

[74] R. Clift, J. R. Grace, and M. E. Weber, "Bubbles, Drops and Particles," *J. Fluid. Mech.*, vol. 94, no. 4, pp. 794–797, 1978. 48

[75] G. Hubbard, V. Denny, and A. Mills, "Droplet evaporation : effects of transients and variable properties," *International Journal of Heat and Mass Transfer*, vol. 18, no. 9, pp. 1003–1008, 1975. 48

[76] N. Froessling, "The evaporation of falling drops," *Gerlands Beiträge Geophysik*, 1938. 49

[77] W. Ranz and W. Marshall, "Evaporation from drops," *Chemical Engineering Progress*, vol. 48, no. 3, 1952. 49

[78] D. B. Spalding, "The combustion of liquid fuels," *Symposium (International) on Combustion*, vol. 4, no. 1, pp. 844–867, 1953. 49

[79] B. Abramzon and W. A. Sirignano, "Droplet vaporization model for spray combustion calculations," *I. Hear Mass Tranfirr*, vol. 32, no. 9, pp. 1605–1618, 1989. 49, 59

[80] H. Larabi, *Vers la modélisation multi-composants des flammes de spray.* PhD thesis, INSA Rouen Normandie, 2019. 50, 156

[81] P. Rosin and E. Rammler, "The Laws Governing the Fineness of powdered coal," *Journal of the Institute of Fuel*, vol. 7, pp. 29–36, 1933. 51

[82] L. I. Xianguo and R. S. Tankin, "Droplet Size Distribution: A Derivation of a Nukiyama-Tanasawa Type Distribution Function," *Combustion Science and Technology*, vol. 56, pp. 65–76, 9 1987. 51

[83] P. González-Tello, F. Camacho, J. M. Vicaria, and P. A. González, "A modified Nukiyama-Tanasawa distribution function and a Rosin-Rammler model for the particle-size-distribution analysis," *Powder Technology*, vol. 186, pp. 278–281, 9 2008. 51

[84] V. Moureau, P. Domingo, and L. Vervisch, "Design of a massively parallel CFD code for complex geometries," *Comptes Rendus - Mecanique*, vol. 339, pp. 141–148, 2 2011. 51, 52

[85] A. Chorin, "Numerical solution of the navier-stokes equations," *Mathematics of computation*, vol. 22, no. 104, pp. 745–762, 1968. 54

[86] J. Kim and P. Moin, "Application of a Fractional-Step Method to Incompressible Navier-Stokes Equations," *Journal of Computational Physics*, vol. 59, pp. 308–323, 1985. 54, 55

[87] M. Malandain, N. Maheu, and V. Moureau, "Optimization of the deflated Conjugate Gradient algorithm for the solving of elliptic equations on massively parallel machines," *Journal of Computational Physics*, vol. 238, pp. 32–47, 4 2013. 56

[88] E. F. Kaasschieter, "Preconditioned conjugate gradients for solving singular systems," *Journal of Computational and Applied Mathematics*, vol. 24, pp. 265–275, 1988. 56

[89] R. A. Nicolaides and R. A. Nicolaidest, "Deflation of Conjugate Gradients with Applications to Boundary Value Problems," *Society for Industrial and Applied Mathematics*, vol. 24, no. 2, pp. 355–365, 1987. 56

[90] B. Yang and S. B. Pope, "An Investigation of the Accuracy of Manifold Methods and Splitting Schemes in the Computational Implementation of Combustion Chemistry," *Combustion and Falme*, vol. 112, no. 1-2, pp. 16–32, 1998. 57

[91] D. Cohen and A. Hindmarsh, "CVODE, A Stiff/Nonstiff ODE Solver in C," *Computers in Physics*, vol. 10, no. 2, pp. 138–143, 1996. 57, 58

[92] C. F. Curtiss and J. . Hirschfelder, "Integration of stiff equations," *Proceedings of the National Academy of Sciences*, vol. 38, no. 11, pp. 235–243, 1952. 57

[93] Q. Cazères, P. Pepiot, E. Riber, and B. Cuenot, "A fully automatic procedure for the analytical reduction of chemical kinetics mechanisms for Computational Fluid Dynamics applications," *Fuel*, vol. 303, 11 2021. 58, 119, 133

[94] L. Guedot, *Developpement de methodes numeriques pour la caracterisation des grandes structures tourbillonaires dans les bruleurs aeronautiques Application aux systemes d'injection multi-points*. PhD thesis, INSA Rouen Normandie, 2015. 60

[95] M. Sanjosé, J. M. Senoner, F. Jaegle, B. Cuenot, S. Moreau, and T. Poinsot, "Fuel injection model for Euler-Euler and Euler-Lagrange large-eddy simulations of an evaporating spray inside an aeronautical combustor," *International Journal of Multiphase Flow*, vol. 37, pp. 514–529, 6 2011. 60

[96] H. Larabi, G. Lartigue, and V. Moureau, "LES study of an n-heptane/air turbulent spray jet flame," in *AIAA Aerospace Sciences Meeting*, 2018. 60, 85

[97] A. Stock, G. Lartigue, and V. Moureau, "Diffusive orthogonal load balancing for Euler–Lagrange simulations," *International Journal for Numerical Methods in Fluids*, vol. 95, pp. 1220–1239, 8 2023. 65

[98] G. E. Moore, "Cramming more components onto integrated circuits," *Electronics*, vol. 38, 1965. 66

[99] "HPC Top 500, https://top500.org/." 66, 156

[100] G. M. Amdahl, "Validity of the single processor approach to achieving large scale computing capabilities.," in *AFIPS Spring Joint Computer Conference*, pp. 483–485, 1967. 66

[101] B. Schmidt, J. González-Domínguez, C. Hundt, and M. Schlarb, *Parallel Programming, Concepts and Practice*. Morgan Kaufmann, 2018. 67

[102] AMD, "4th gen AMD EPYC™ processor architecture," tech. rep., AMD, 2023. 67

[103] Message Passing Interface Forum, "Message Passing Interface Forum, MPI: A Message-Passing Interface Standard Version 3.1," tech. rep., University of Tennessee, 2015. 67

[104] OpenMP Architecture Review Board, "OpenMP Application Programming Interface," 2015. 67

[105] F. Gava, *Optimisation des performances parallèles d'un solveur cfd pour les plateformes de calcul émergentes*. PhD thesis, INSA Rouen Normandie, 2021. 68

[106] B. L. Chamberlain, "Graph Partitioning Algorithms for Distributing Workloads of Parallel Computations," 1998. 69

[107] M. Predari, *On the Inverse of the Sum of Matrices*. PhD thesis, Université de Bordeaux, 2016. 69

[108] N. Gourdain, L. Gicquel, M. Montagnac, O. Vermorel, M. Gazaix, G. Staffelbach, M. Garcia, J.-F. Boussuge, and T. Poinsot, "High performance parallel computing of flows in complex geometries: I. Methods," *Computational Science & Discovery*, vol. 2, p. 015003, 11 2009. 69

[109] D. K. Kafui, S. Johnson, C. Thornton, and J. P. Seville, "Parallelization of a Lagrangian-Eulerian DEM/CFD code for application to fluidized beds," *Powder Technology*, vol. 207, pp. 270–278, 2 2011. 71

[110] I. F. Sbalzarini, J. H. Walther, M. Bergdorf, S. E. Hieber, E. M. Kotsalis, and P. Koumoutsakos, "PPM - A highly efficient parallel particle-mesh library for the simulation of continuum systems," *Journal of Computational Physics*, vol. 215, pp. 566–588, 7 2006. 71

[111] Y. Shigeto and M. Sakai, "Parallel computing of discrete element method on multi-core processors," *Particuology*, vol. 9, pp. 398–405, 8 2011. 71

[112] H. Sitaraman and R. Grout, "Balancing conflicting requirements for grid and particle decomposition in continuum-Lagrangian solvers," *Parallel Computing*, vol. 52, pp. 1–21, 2 2016. 71

[113] W. Zhang, A. Myers, K. Gott, A. Almgren, and J. Bell, "AMReX: Block-Structured Adaptive Mesh Refinement for Multiphysics Applications," *The International Journal of High Performance Computing Applications*, vol. 35, pp. 508–526, 9 2021. 71

[114] J. Capecelatro and O. Desjardins, "An Euler-Lagrange strategy for simulating particle-laden flows," *Journal of Computational Physics*, vol. 238, pp. 1–31, 4 2013. 71

[115] R. Pankajakshan, B. J. Mitchell, and L. K. Taylor, "Simulation of unsteady two-phase flows using a parallel Eulerian-Lagrangian approach," *Computers and Fluids*, vol. 41, pp. 20–26, 2 2011. 71

[116] G. Karypis and V. Kumar, "METIS: A Software Package for Partitioning Unstructured Graphs, Partitioning Meshes, and Computing Fill-Reducing Orderings of Sparse Matrices," tech. rep., University of Minnesota, 1997. 72, 74

[117] F. Pellegrini, "Scotch and libScotch 5.1 User's Guide," 2008. 72

[118] P. M. Campbell, E. A. Carmona, and D. W. Walker, "Hierarchical Domain Decomposition With Unitary Load Balancing For Electromagnetic Part icle-In-Cell Codes *," *Distributed Memory Computing Conference*, vol. 5, pp. 943–950, 1990. 72

[119] M. Sauget and G. Latu, "Dynamic Load Balancing for PIC code using Eulerian/Lagrangian partitioning," in *Distributed Memory Computing Conference*, 6 2011. 72

[120] S. Yakubov, B. Cankurt, M. Abdel-Maksoud, and T. Rung, "Hybrid MPI/OpenMP parallelization of an euler-lagrange approach to cavitation modelling," *Computers and Fluids*, vol. 80, no. 1, pp. 365–371, 2013. 73

[121] A. Thari, N. C. Treleaven, M. Staufer, and G. J. Page, "Parallel load-balancing for combustion with spray for large-scale simulation," *Journal of Computational Physics*, vol. 434, 6 2021. 73

[122] A. Thari, M. Staufer, and G. J. Page, "Asynchronous task based Eulerian-Lagrangian parallel solver for combustion applications," *Journal of Computational Physics*, vol. 458, 6 2022. 73

[123] G. Karypis and V. Kumar, "Multilevel k-way Partitioning Scheme for Irregular Graphs 1," *Journal of parallel and distributed computing*, vol. 48, pp. 96–129, 1998. 74

[124] M. H. Willebeek-Lemair and A. P. Reeves, "Strategies for Dynamic Load Balancing on Highly Parallel Computers," *IEEE Transaction on Parallel and Distributed Systems*, vol. 4, no. 9, 1993. 75

[125] R. Subramanian and I. D. Scherson, "An analysis of diffusive load-balancing," in *Proceedings of the 6th Annual ACM Symposium on Parallel Algorithms and Architectures, SPAA 1994*, pp. 220–225, Association for Computing Machinery, Inc, 8 1994. 75

[126] G. Cybenko, "Dynamic Load Balancing for Distributed Memory Multiprocessors," *Journal of Parallel and Distributed Computing*, vol. 7, pp. 279–309, 1989. 75, 76

[127] H. Arndt, *Load balancing auf Parallelrechnern mit Hilfe endlicher Dimension-Exchange-Verfahren*. PhD thesis, Bergischen Universit¨at Wuppertal, 2003. 75

[128] S. H. Hosseini, B. Litow, M. Malkawi, J. Mcpherson, and K. Vairavan, "Analysis of a Graph Coloring Based Distributed Load Balancing Algorithm," *Journal of Parallel and Distributed Computing*, vol. 10, pp. 160–166, 1990. 76

[129] C. Z. Xu and F. C. M. Lau, "Analysis of the Generalized Dimension Exchange Method for Dynamic Load Balancing," *Journal of Parallel and Distributed Computing*, vol. 16, pp. 385–393, 1992. 76

[130] E. Salaün, *Etude de la formation de NO dans les chambres de combustion haute pression kérosène/air par fluorescence induite par laser*. PhD thesis, INSA Rouen Normandie, 2017. 85

[131] P. Domingo-Alvarez, P. Bénard, V. Moureau, G. Lartigue, and F. Grisch, "Impact of Spray Droplet Distribution on the Performances of a Kerosene Lean/Premixed Injector," *Flow, Turbulence and Combustion*, vol. 104, pp. 421–450, 3 2020. 85

[132] M. Abkarian, S. Mendez, N. Xue, F. Yang, and H. A. Stone, "Speech can produce jet-like transport relevant to asymptomatic spreading of virus," *PNAS*, vol. 117, 2020. 85

[133] "Occigen supercomputer hardware: https://www.cines.fr/en/supercomputing-2/hardwares/the-supercomputer-occigen/configuration/." 87

[134] C. Dapogny, C. Dobrzynski, P. Frey, C. Dapogny, C. Dobrzynski, and P. Frey, "Three-dimensional adaptive domain remeshing, implicit domain meshing, and applications to free and moving boundary problems," *Journal of Com-putational Physics*, p. 10, 2014. 90, 98, 102

[135] G. Balarac, F. Basile, P. Bénard, F. Bordeu, J.-B. Chapelier, L. Cirrottola, G. Caumon, C. Dapogny, P. Frey, A. Froehly, G. Balarac, F. Basile, F. Bordeu, J.-B. Chapelier, L. Cirrottola, G. Caumon, C. Dapogny, P. Frey, A. Froehly, G. Ghigliotti, R. Laraufie, G. Lartigue, C. Legentil, R. Mercier, V. Moureau, C. Nardoni, S. Pertant, and M. Zakari, "Tetrahedral Remeshing in the Context of Large-Scale Numerical Simulation and High Performance Computing," *MathematicS In Action*, vol. 11, no. 1, pp. 129–164, 2022. 90, 98, 99, 102, 157

[136] P. Benard, G. Balarac, V. Moureau, C. Dobrzynski, G. Lartigue, and Y. D'Angelo, "Mesh adaptation for large-eddy simulations in complex geometries," *International Journal for Numerical Methods in Fluids*, vol. 81, pp. 719–740, 8 2016. 90, 99

[137] A. Stock and V. Moureau, "Feature-based adaptive mesh refinement for multi-regime reactive flows," *Proceedings of the Combustion Institute*, 2024. 93

[138] G. Boudier, L. Y. Gicquel, and T. J. Poinsot, "Effects of mesh resolution on large eddy simulation of reacting flows in complex geometry combustors," *Combustion and Flame*, vol. 155, pp. 196–214, 10 2008. 94

[139] A. Grenouilloux, *Modelisation haute fidelite de l'aerodynamique dans les inverseurs de poussee des turboreacteurs a flux melange*. PhD thesis, INSA Rouen Normandie, 2023. 95, 157

[140] T. Fabbri, *Development of a high fidelity fluid-structure interaction solver : towards flexible foils simulation*. PhD thesis, Université Grenoble Alpes, 2022. 95

[141] P.L. George, H. Borouchaki, F. Alauzet, P. Laug, A. Loseille, and L. Maréchal, *Meshing, Geometric Modeling and Numerical Simulation 2. Metrics, Meshes and Mesh Adaptation, Numerical Methods in Engineering Series - Geometric Modeling and Applications Set*. ISTE Edition, 2018. 96

[142] P. C. Caplan, R. Haimes, D. L. Darmofal, and M. C. Galbraith, "Four-Dimensional Anisotropic Mesh Adaptation," *CAD Computer Aided Design*, vol. 129, 12 2020. 96

[143] W. G. Habashi, J. Dompierre, Y. Bourgault, D. Ait-Ali-Yahia, M. Fortin, M.-G. Vallet, and W. G. Habashi, "Anisotropic mesh adaptation: towards user-independent, mesh-independent and solver-independent CFD. Part I: general principles," *Int. J. Numer. Meth. Fluids*, vol. 32, pp. 725–744, 2000. 99

[144] P. W. Agostinelli, B. Rochette, D. Laera, J. Dombard, B. Cuenot, and L. Gicquel, "Static mesh adaptation for reliable Large Eddy Simulation of turbulent reacting flows," *Physics of Fluids*, vol. 33, no. 3, 2021. 100

[145] T. Schmitt, M. Boileau, and D. Veynante, "Flame wrinkling factor dynamic modeling for large eddy simulations of turbulent premixed combustion," *Flow, Turbulence and Combustion*, vol. 94, pp. 199–217, 1 2015. 100

[146] O. V. Vasilyev, T. S. Lund, and P. Moin, "A General Class of Commutative Filters for LES in Complex Geometries," *Journal of Computational Physics*, vol. 146, pp. 82–104, 1998. 100

[147] J. W. Dold, "Flame Propagation in a Nonuniform Mixture: Analysis of a Slowly Varying Triple Flame," *Combustion and Flame*, vol. 76, pp. 71–88, 1989. 102

[148] D. Veynante, L. Vervisch, t. T. Poinso, A. Linan, and G. Ruetsch, "Triple flame structure and diffusion flame stabilization," *Center for turbulent research, Proceedings of the Summer Program*, 1994. 102

[149] ""Chemical-Kinetic Mechanisms for Combustion Applications", San Diego Mechanism web page, Mechanical and Aerospace Engineering (Combustion Research), University of California at San Diego." 102, 134

[150] P. E. Farrell, M. D. Piggott, C. C. Pain, G. J. Gorman, and C. R. Wilson, "Conservative interpolation between unstructured meshes via supermesh construction," *Computer Methods in Applied Mechanics and Engineering*, vol. 198, pp. 2632–2642, 7 2009. 109

[151] A. Stock, V. Moureau, J. Leparoux, and R. Mercier, "Low-cost Jacobian-free mapping for dynamic cell clustering in multi-regime reactive flows," *Proceedings of the Combustion Institute*, vol. 40, 1 2024. 117

[152] G.P. Smith, D. M. Golden, M. Frenklach, N. W. Moriarty, B. Eiteneer, M. Goldenberg, C. T. Bowman, R. K. Hanson, S. Song, W. C. Gardiner, V. V. Lissianski, and Z. Qin, "GRI-MECH 3.0." 118

[153] "Creck modeling." 118, 150

[154] T. Lu and C. K. Law, "A directed relation graph method for mechanism reduction," *Proceedings of the Combustion Institute*, vol. 30, no. 1, pp. 1333–1341, 2005. 118

[155] P. Pepiot-Desjardins and H. Pitsch, "An efficient error-propagation-based reduction method for large chemical kinetic mechanisms," *Combustion and Flame*, vol. 154, pp. 67–81, 7 2008. 118

[156] G. Li and H. Rabitz, "A general analysis of exact lumping in chemical kinetics," *Chemical Engineering Science*, vol. 44, no. 6, pp. 1413–1430, 1989. 119

[157] G. Li and H. Rabitz, "A general analysis of approximate lumping in chemical kinetics," *Chemical Engineering Science*, vol. 45, no. 4, pp. 977–1002, 1990. 119

[158] H. Huang, M. Fairweather, J. F. Griffiths, A. S. Tomlin, and R. B. Brad, "A systematic lumping approach for the reduction of comprehensive kinetic models," *Proceedings of the Combustion Institute*, vol. 30, no. 1, pp. 1309–1316, 2005. 119

[159] T. Lu and C. K. Law, "Toward accommodating realistic fuel chemistry in large-scale computations," *Progress in Energy and Combustion Science*, vol. 35, pp. 192–215, 4 2009. 119

[160] M. Cailler, N. Darabiha, D. Veynante, and B. Fiorina, "Building-up virtual optimized mechanism for flame modeling," *Proceedings of the Combustion Institute*, vol. 36, no. 1, pp. 1251–1258, 2017. 119

[161] M. Cailler, N. Darabiha, and B. Fiorina, "Development of a virtual optimized chemistry method. Application to hydrocarbon/air combustion," *Combustion and Flame*, vol. 211, pp. 281–302, 1 2020. 119

[162] D. A. Schwer, P. Lu, and W. H. Green, "An adaptive chemistry approach to modeling complex kinetics in reacting flows," *Combustion and Flame*, vol. 133, pp. 451–465, 6 2003. 119

[163] L. Liang, J. G. Stevens, and J. T. Farrell, "A dynamic adaptive chemistry scheme for reactive flow computations," *Proceedings of the Combustion Institute*, vol. 32 I, no. 1, pp. 527–534, 2009. 119

[164] L. Liang, J. G. Stevens, S. Raman, and J. T. Farrell, "The use of dynamic adaptive chemistry in combustion simulation of gasoline surrogate fuels," *Combustion and Flame*, vol. 156, pp. 1493–1502, 7 2009. 119

[165] G. D'Alessio, A. Parente, A. Stagni, and A. Cuoci, "Adaptive chemistry via pre-partitioning of composition space and mechanism reduction," *Combustion and Flame*, vol. 211, pp. 68–82, 1 2020. 119

[166] G. D'Alessio, A. Cuoci, G. Aversano, M. Bracconi, A. Stagni, and A. Parente, "Impact of the partitioning method on multidimensional adaptive-chemistry simulations," *Energies*, vol. 13, 5 2020. 119

[167] P. Pagani, R. Malpica Galassi, R. Amaduzzi, A. Parente, and F. Contino, "An enhanced Sample-Partitioning Adaptive Reduced Chemistry method with a-priori error estimation," *Combustion and Flame*, vol. 260, 2 2024. 119

[168] I. T. Jolliffe, *Principal Component Analysis* . Springer New York, NY, 2002. 119

[169] L. Lu and S. B. Pope, "An improved algorithm for in situ adaptive tabulation," *Journal of Computational Physics*, vol. 228, pp. 361–386, 2 2009. 119, 122

[170] I. Banerjee and M. G. Ierapetritou, "Development of an adaptive chemistry model considering micromixing effects," *Chemical Engineering Science*, vol. 58, no. 20, pp. 4537–4555, 2003. 119

[171] J. C. Sutherland and A. Parente, "Combustion modeling using principal component analysis," *Proceedings of the Combustion Institute*, vol. 32 I, no. 1, pp. 1563–1570, 2009. 119, 139

[172] K. Zdybał, G. D'Alessio, G. Aversano, M. R. Malik, A. Coussement, J. C. Sutherland, and A. Parente, "Advancing Reacting Flow Simulations with Data-Driven Models," in *Data-Driven Fluid Mechanics: Combining First Principles and Machine Learning*, Cambridge University Press, 9 2022. 119

[173] Y. Yang, S. B. Pope, and J. H. Chen, "Empirical low-dimensional manifolds in composition space," *Combustion and Flame*, vol. 160, pp. 1967–1980, 10 2013. 119

[174] S. Akaotsu, R. Ozawa, Y. Matsushita, H. Aoki, and W. Malalasekera, "Effects of infinitely fast chemistry on combustion behavior of coaxial diffusion flame predicted by large eddy simulation," *Fuel Processing Technology*, vol. 199, 3 2020. 121

[175] F. Flemming, A. Sadiki, and J. Janicka, "LES using artificial neural networks for chemistry representation," *Progress in Computational Fluid Dynamics*, vol. 5, no. 7, pp. 375–385, 2005. 121

[176] Z. Nikolaou, L. Vervisch, and P. Domingo, "Criteria to switch from tabulation to neural networks in computational combustion," *Combustion and Flame*, vol. 246, 12 2022. 121

[177] U. Maas and S. B. Pope, "Implementation of simplified chemical kinetics based on intrinsic low-dimensional manifolds," *24th Symposium (International) on Combustion*, 1992. 121

[178] O. Gicquel, D. Thevenin, M. Hilka, and N. Darabiha, "Direct numerical simulation of turbulent premixed flames using intrinsic low-dimensional manifolds," *Combustion Theory and Modelling*, vol. 3, pp. 479–502, 9 1999. 121

[179] J. A. Van Oijen, F. A. Lammers, and L. P. H. De Goey, "Modeling of Complex Premixed Burner Systems by Using Flamelet-Generated Manifolds," tech. rep., Eindhoven University of Technology, 2001. 121

[180] O. Gicquel, N. Darabiha, and D. Thé Venin, "Laminar premixed hydrogen/air counterflow flame simulations using flame prolongation of ILDM with differential diffusion," in *Proceedings of the Combustion Institute*, vol. 28, pp. 1901–1908, 2000. 122

[181] J. Galpin, A. Naudin, L. Vervisch, C. Angelberger, O. Colin, and P. Domingo, "Large-eddy simulation of a fuel-lean premixed turbulent swirl-burner," *Combustion and Flame*, vol. 155, pp. 247–266, 10 2008. 122

[182] B. Fiorina, R. Vicquelin, P. Auzillon, N. Darabiha, O. Gicquel, and D. Veynante, "A filtered tabulated chemistry model for LES of premixed combustion," *Combustion and Flame*, vol. 157, pp. 465–475, 3 2010. 122

[183] S. B. Pope, "Computationally efficient implementation of combustion chemistry using in situ adaptive tabulation," *Combustion Theory and Modelling*, vol. 1, no. 1, pp. 41–63, 1997. 122

[184] B. J. Liu and S. B. Pope, "The performance of in situ adaptive tabulation in computations of turbulent flames," *Combustion Theory and Modelling*, vol. 9, pp. 549–568, 11 2005. 122

[185] A. Cuoci, A. Nobili, A. Parente, T. Grenga, and H. Pitsch, "Tabulation based sample-partitioning adaptive reduced chemistry and cell agglomeration," in *Proceedings of the Combustion Institute*, 2024. 123

[186] F. Perini, "High-dimensional, unsupervised cell clustering for computationally efficient engine simulations with detailed combustion chemistry," *Fuel*, vol. 106, pp. 344–356, 2013. 123

[187] G. M. Goldin, Z. Ren, and S. Zahirovic, "A cell agglomeration algorithm for accelerating detailed chemistry in CFD," *Combustion Theory and Modelling*, vol. 13, pp. 721–739, 8 2009. 123

[188] L. Liang, J. G. Stevens, and J. T. Farrell, "A dynamic multi-zone partitioning scheme for solving detailed chemical kinetics in reactive flow computations," *Combustion Science and Technology*, vol. 181, pp. 1345–1371, 11 2009. 123, 124

[189] A. Babajimopoulos, D. N. Assanis, D. L. Flowers, S. M. Aceves, and R. P. Hessel, "A fully coupled computational fluid dynamics and multi-zone model with detailed chemical kinetics for the simulation of premixed charge compression ignition engines," *International Journal of Engine Research*, vol. 6, pp. 497–512, 10 2005. 123

[190] J. Macqueen, "Some methods for classification and analysis of multivariate observations," in *Berkeley Symp. on Math. Statist. and Prob.*, pp. 281–297, 1967. 123

[191] G. Gan, C. Ma, and J. Wu, "Grid-based Clustering Algorithms," *Data clustering: Theory, Algorithms, and Applications*, vol. 6, no. 12, pp. 209–2017, 2017. 123

[192] A. E. Hoerl and R. W. Kennard, "Ridge Regression: Biased Estimation for Nonorthogonal Problems," *Technometrics*, vol. 12, no. 1, pp. 55–67, 1970. 127

[193] D. Knuth, "The Art of Computer Programming," in *Sorting and Searching*, vol. Volume 3, ch. 5.2.5, Addison-Wesley, 1997. 130

[194] A. Sabah, S. Abu-Naser, H. Emad, A. Fikri, Abu S., A. Helmi, M. Maher, and A. Hamouda, "Comparative Analysis of the Performance of Popular Sorting Algorithms on Datasets of Different Sizes and Characteristics," *International Journal of Academic Engineering Research*, vol. 7, 2023. 130

[195] M. Marcellino, D. W. Pratama, S. S. Suntiarko, and K. Margi, "Comparative of Advanced Sorting Algorithms (Quick Sort, Heap Sort, Merge Sort, Intro Sort, Radix Sort) Based on Time and Memory Usage," in *2021 1st International Conference on Computer Science and Artificial Intelligence (ICCSAI)*, 2021. 130

[196] I. J. Davis, "A Fast Radix Sort," *The computer journal*, vol. 35, no. 6, 1992. 130, 131

[197] S. Balay, S. Abhyankar, and F. Adams, Mark, "PETSc Web page," 2023. 135

[198] A. Kumar, M. Rieth, O. Owoyele, J. H. Chen, and T. Echekki, "Acceleration of turbulent combustion DNS via principal component transport," *Combustion and Flame*, vol. 255, 9 2023. 139

[199] D. A. Pope, "An Exponential Method of Numerical Integration of Ordinary Differential Equations," *Communications of the ACM*, 1963. 149

[200] M. Tokman, "Efficient integration of large stiff systems of ODEs with exponential propagation iterative (EPI) methods," *Journal of Computational Physics*, vol. 213, pp. 748–776, 4 2006. 149

[201] T. Naess, E. Riber, and B. Cuenot, "Numerical prediction of nitric oxide formation in a turbulent n-heptane spray flame using Large Eddy Simulation," in *Proceedings of the 11th combustion meeting*, 2023. 150

[202] J. Carmona, J. Leparoux, I. El Yamani, and V. Moureau, "High-fidelity simulation of a pressure swirl fuel atomizer : In-depth analysis of in-nozzle flow dynamics and liquid sheet disintegration," in *5th international conference on numerical methods in multiphase flow*, (Reykjaviv, Iceland), 6 2024. 151

[203] Y. Sun, C. S. Vegad, Y. Li, L. Dreßler, B. Renou, K. Nishad, F. X. Demoulin, C. Hasse, and A. Sadiki, "Liquid sheet formation and spray characterization of N-heptane spray jet from a swirl atomizer: Numerical analysis and validation," *Physics of Fluids*, vol. 36, 3 2024. 151

[204] D. Ferrando, M. Carreres, M. Belmar-Gil, D. Cervelló-Sanz, B. Duret, J. Reveillon, F. J. Salvador, and F. X. Demoulin, "Modeling internal flow and primary atomization in a simplex pressure-swirl atomizer," *Atomization and Sprays*, pp. 1–28, 2022. 151

[205] M. Herrmann, "Detailed numerical simulations of the primary atomization of a turbulent liquid jet in crossflow," *Journal of Engineering for Gas Turbines and Power*, vol. 132, no. 6, pp. 1–10, 2010. 151

[206] I. El Yamani, V. Moureau, M. Cailler, and L. Voivenel, "A multi-scale Eulerian-Lagrangian Method based on unstructured AMR for the simulation of atomization," *WIP not published at this date*, 2024. 151

Simulation exaflopique de la combustion de sprays

Keywords: Large-Eddy Simulation (LES), Two-phase Reactive Flow, Adaptive Mesh Refinement (AMR), Load-balancing, Dynamic Cell Clustering (DCC), Euler-Lagrange.

Large Eddy Simulation (LES) has emerged as a powerful computational tool for the design and analysis of spray burners, offering the ability to capture the complex interactions between turbulent flow, combustion, and spray dynamics with high fidelity. However, the high accuracy of LES comes at a significant computational cost. The simulation of these systems often requires the use of large meshes in order to resolve fine-scale turbulence and sharp flame front as well as the handling of numerous species and chemical reactions. These demands pose substantial challenges in terms of the required computational resources and the time required for the simulation process, which can hinder the practical application of LES in industrial design processes. This thesis addresses the computational challenges associated with LES of spray burners by exploring and developing numerical approaches aimed at reducing the computational burden without compromising the accuracy of the simulations. Three primary strategies are investigated: Euler-Lagrange load balancing, Adaptive Mesh Refinement, and Dynamic Cell Clustering.

La simulation numérique des grandes échelles (LES, pour Large Eddy Simulation) s'est imposée comme un outil numérique puissant pour la conception et l'analyse des brûleurs de sprays, permettant de capturer avec une grande précision les interactions complexes entre l'écoulement turbulent, la combustion et la dynamique des sprays. Cependant, cette précision accrue de la LES s'accompagne d'un coût CPU élevé. Ces simulations nécessitent souvent l'utilisation de maillages de grande taille afin de résoudre les turbulences à petite échelle et les fronts de flamme nets, ainsi que la gestion de nombreuses espèces chimiques et réactions. Ces exigences posent des défis importants en termes de ressources informatiques et de temps de calcul, ce qui peut freiner l'application pratique de la LES dans les processus de conception industrielle. Cette thèse s'attaque aux défis numériques associés à la LES des brûleurs de sprays en explorant et développant des approches numériques visant à réduire coût de calcul sans compromettre la précision des simulations. Trois stratégies principales sont étudiées : l'équilibrage de charge Euler-Lagrange, l'adaptation dynamique de maillage, et le clustering dynamique des termes source.