Université de Toulouse

THESE

En vue de l'obtention du

DOCTORAT DE L'UNIVERSITÉ DE TOULOUSE

Délivré par *INP Toulouse*
Discipline ou spécialité : *Dynamique des fluides*

Présentée et soutenue par *Marta GARCÍA*
Le *19 Janvier 2009*

Titre : *Développement et validation du formalisme Euler-Lagrange dans un solveur parallèle et non-structuré pour la simulation aux grandes échelles*

JURY

J. RÉVEILLON	*Prof. à l'Université de Rouen*	Rapporteur
K. D. SQUIRES	*Prof. à l'Université d'Arizona*	Rapporteur
L. GIRAUD	*Prof. à l'ENSEEIHT, Toulouse*	Examinateur
J. MÜLLER	*Prof. à l'Université Queen Mary, Londres*	Examinateur
O. SIMONIN	*Prof. à l'INP Toulouse*	Examinateur
V. MOUREAU	*Chargé de recherche au CORIA, Rouen*	Co-directeur de thèse
T. POINSOT	*Directeur de recherche à l'IMFT, Toulouse*	Directeur de thèse

---

Ecole doctorale : *Mécanique, Energétique, Génie civil, Procédés*
Unité de recherche : *CERFACS*
Directeur(s) de Thèse : *Thierry POINSOT*

A mis padres
A Roberto

« Ami, je crois que dans ce monde il ne faut rien faire à moitié et qu'une fois entreprise une honnête carrière, il convient de la poursuivre avec fermeté d'âme en laissant à la Providence l'issue, ou bonne, ou mauvaise, de nos travaux. »

Pascal Paoli à Ferdinando de Leon
(Pastoreccia, 11 juin 1755).

Contents

Partie en français **1**

1 Scientific context **13**

1.1 Energy resources 13

1.2 Energy production by combustion: gas turbines 14

1.3 Numerical simulation of turbulent flows 15

1.4 Two-phase flows numerical simulation 16

1.5 Growth of computer power in the last years 18

1.6 Aim of the work and plan of the thesis 20

2 The AVBP solver **23**

2.1 Governing equations for LES 24

2.1.1 Filtering procedure 24

2.1.2 Filtered Navier-Stokes equations for non-reacting flows 25

2.1.3 Inviscid terms 26

2.1.4 Filtered viscous terms 26

2.1.5 Subgrid-scale turbulent terms 28

2.1.6 Models for the subgrid stress tensor 29

Smagorinsky model 29

Filtered Smagorinsky model 29

Dynamic Smagorinsky model 29

Wall-adapting local eddy-viscosity (WALE) model 30

k-equation model 30

2.2 Numerical method 31
2.2.1 The cell-vertex discretization 31
Weighted Cell Residual Approach 33
Computation of gradients 34
Computation of time step 35
2.2.2 The Lax-Wendroff scheme 35
2.2.3 The TTGC numerical scheme 36
2.3 Boundary conditions 39
2.4 Domain partitioning 40
2.4.1 Static load balancing 41
2.4.2 Partitioning algorithms available in Avbp. 41
2.4.3 Analysis of partitioning algorithm performances in Avbp. 44
2.4.4 Introduction of a new partitioning algorithm 46
2.4.5 Node reordering 50
2.5 Data structure 53
2.5.1 The element-to-node pointer 54
2.5.2 Grouping of cells 54
2.6 Parallel computing 55
2.6.1 Types of parallelism 56
2.6.2 Memory-processor organization 57
2.6.3 Flynn's taxonomy 58
2.6.4 Communication protocol 59
2.7 Rounding errors and reproducibility of LES 60
Effects of node reordering in mesh 62
2.7.1 Finite precision computation 63
Effects of machine precision 65
Effects of initial conditions 67
2.7.2 True nature of turbulence 67
Influence of turbulence 68
2.8 Summary and conclusions 70

3 Numerical implementation of the Lagrangian module 73

3.1 Particle equations of motion 73

3.1.1 Drag force 74

3.1.2 Other forces 76

Added mass 76

Pressure gradient and buoyancy force 76

Basset force 77

3.1.3 Equations retained for this study 77

3.2 Data structure 79

3.2.1 Lagrangian variables 79

3.2.2 Cell-number pointer 81

3.2.3 Number of interface element pointer 82

3.3 Locating particles in elements of arbitrary shape 83

3.4 Search algorithms for particles on unstructured grids 86

3.4.1 Searching particles for the first time 89

3.4.2 Searching injected particles 90

3.4.3 Searching particles during the simulation 91

3.4.4 Searching particles crossing boundaries between processors 92

3.5 Interpolation of gas-phase properties 94

3.5.1 Taylor series 94

3.5.2 Linear Least-Squares 95

3.5.3 Lagrange polynomials 95

3.6 Implementation of two-way coupling 96

3.6.1 Source terms between solvers 96

3.6.2 Validation of two-way coupling implementation 97

3.7 Boundary conditions 98

3.7.1 Particle injection 98

Injection geometry 99

Particle size distribution 99

4 Application to a homogeneous isotropic turbulence 101

4.1 Notions of turbulence . . . 102

4.2 Choice of the configuration: particles in decaying turbulence . . . 105

4.3 Homogeneous Isotropic Turbulence statistics . . . 106

4.3.1 Analysis in the physical space . . . 106

4.3.2 Turbulent kinetic energy temporal decaying . . . 107

4.3.3 Analysis in the spectral space . . . 107

4.3.4 Properties of the Lagrangian field . . . 108

4.4 Single-phase test case . . . 109

4.4.1 Gaseous initial conditions . . . 109

4.5 Dispersed-phase test case . . . 111

4.5.1 Performance analysis of particle locations for the first time . . . 111

4.5.2 General results of the two-phase flow simulation . . . 113

4.5.3 Influence of the interpolation algorithm . . . 115

4.5.4 Illustration of preferential concentration . . . 116

4.6 Summary and conclusions . . . 117

5 Application to a polydisperse two-phase flow of a confined bluff body 119

5.1 Experimental set-up . . . 120

5.1.1 The gaseous phase . . . 121

5.1.2 The dispersed phase . . . 122

5.2 Computational set-up . . . 123

5.2.1 Numerical parameters . . . 125

5.3 Results for single-phase flow case . . . 127

5.4 Results for two-phase flow monodisperse case . . . 131

5.4.1 Gaseous phase . . . 131

5.4.2 Dispersed phase: results for $d_p = 60\ \mu$m . . . 135

5.5 Results for two-phase flow polydisperse case . . . 142

5.5.1 Gaseous phase . . . 142

5.5.2 Dispersed phase: axial velocity profiles . . . 146

5.5.3 Dispersed phase: radial velocity profiles 148
5.5.4 Influence of the number of samples 157
5.5.5 Particle trajectories 164
5.6 Performance 166
5.6.1 Scalability with a one-constraint partitioning algorithm 167
5.6.2 Scalability with a two-constraint partitioning algorithm 169
5.7 Conclusions 172

Conclusions and perspectives **174**

Bibliography **179**

Appendices **197**

A Suggestion of treatment of other Lagrangian boundary conditions **197**

B Proc. of the Summer Program 2006, pp. 197-211 **199**

C 6th Int. Conference on Multiphase Flow, Leipzig, S3_Fri_A_62 (2007) **217**

D AIAA Journal, Vol. 46, N° 7, pp. 1773-1781 (July 2008) **231**

E Journal of Computational Physics, Vol. 228, N° 2, pp. 539-564 (2009) **245**

Partie en français

Chapitre 1 : contexte scientifique

Sources d'énergie

La vie sur terre est contrôlée par l'énergie ; les êtres humains en sont de gros consommateurs et puisent dans les ressources (la consommation de combustibles fossiles a presque doublé tous les 20 ans depuis 1900). Mais l'épuisement des combustibles fossiles, qui fournissent les trois quarts de cette énergie, n'est pas loin, et aucune autre source d'énergie abondante et bon marché n'est suffisante pour prendre leur place. À l'heure actuelle, les principales solutions pour remplacer la combustion sont, par exemple : l'énergie nucléaire, l'énergie solaire, l'énergie éolienne, l'énergie hydroélectrique, l'énergie marémotrice, l'énergie des vagues, l'énergie géothermique et la bioconversion. Toutes avec leurs avantages et leurs inconvénients.

Il est bien connu que la combustion d'hydrocarbures et d'air peut produire une grande quantité de monoxyde de carbone (CO), de dioxyde de carbone (CO2) et d'oxydes d'azote (NOx). Les premiers contribuent largement à l'effet de serre tandis que le dernier est le responsable du smog photochimique, et est, au niveau du sol, l'un des principaux précurseurs de l'ozone, qui est un composé toxique et un problème majeur pour l'environnement. La combustion de carburants fossiles est de plus en plus difficile à éviter. La question clé est donc de les brûler avec une grande efficacité afin de réduire les niveaux de polluants pour (i) compenser la hausse du prix des combustibles fossiles et (ii) limiter la contribution au réchauffement climatique planétaire.

Actuellement, beaucoup de nouvelles centrales électriques utilisent des turbines à gaz et brûlent du gaz naturel. Même si elle est l'une des sources d'énergie les plus efficaces (jusqu'à 57% d'efficacité pour les modèles actuels), des législations de plus en plus strictes ont forcé les fabricants à allonger les phases de conception, développement et recherche afin de créer des systèmes plus propres, moins coûteux et plus efficaces. Des formules empiriques et des études expérimentales ont été utilisées dans le passé pour la phase de conception. Aujourd'hui, la simulation numérique, combinée avec ces outils, est largement utilisée pour la recherche et la conception.

Production d'énergie par combustion : turbine à gaz

Une turbine à gaz est une machine tournante thermodynamique dont le rôle est de produire de l'énergie mécanique (rotation d'un arbre) à partir de l'énergie contenue dans un hydrocarbure (fioul, gaz, etc). Elle est constituée d'un compresseur en amont (flux radial ou axial) couplé mécaniquement en aval à une turbine et d'une chambre de combustion entre les deux. L'énergie est libérée lorsque l'air comprimé est mélangé à du carburant et enflammé dans la chambre de

combustion. Les gaz qui en résultent sont dirigés vers les pales de la turbine, entraînent son arbre en rotation et fournissent de la puissance mécanique au compresseur. Enfin, les gaz passent à travers la tuyère, et génèrent une poussée supplémentaire en accélérant les gaz d'échappement chauds par expansion vers la pression atmosphérique. L'énergie est extraite sous forme de puissance de l'arbre, de l'air comprimé et de la poussée, combinée de différentes façons, et utilisée pour alimenter les avions, trains, bateaux, groupes électrogènes, etc.

Même s'il n'y a pas d'application directe des développements réalisée dans cette thèse pour une chambre de combustion réelle, les cas tests d'application présentés permettent de reproduire certains de leurs phénomènes les plus importants, comme par exemple le mouvement des particules à cause de la turbulence. Compte tenu que cette thèse est orientée vers la simulation numérique de ce type de dispositifs, les principales approches utilisées pour résoudre numériquement les équations utilisées pour décrire ces phénomènes sont présentées ci-après.

Simulation numérique des écoulements turbulents

Il existe trois principales approches pour décrire les processus de combustion turbulente en dynamique des fluides numérique (en anglais, Computational Fluid dynamics, CFD) :

- Simulation numérique directe (SND ou DNS en anglais) : dans ce type de simulations, l'intégralité des équations de Navier-Stokes instantanées sont résolues sans aucun modèle turbulent. Toutes les échelles de la turbulence sont résolues et leurs effets sur la combustion sont capturés. Développée au cours des vingt dernières années grâce au développement de machines à haute performance, la SND a changé l'analyse de la combustion turbulente, mais elle est encore limitée à des cas tests académiques. Dans ce genre de simulations, l'ensemble des phénomènes physiques à représenter est résolu. Cela signifie que la plus petite échelle de la turbulence et toutes les zones réactives de l'écoulement sont résolues.

- Reynolds-averaged Navier-Stokes (RANS) : ces calculs ont été historiquement la première approche, car le calcul du champ instantané d'une flamme turbulente était impossible. Par conséquent, la méthode RANS repose sur une décomposition de Reynolds du champ turbulent fluide instantané suivie d'une moyenne sur l'ensemble des réalisations possibles de ce champ fluide. Les modèles de fermeture qui en résultent doivent ainsi rendre compte de l'ensemble des échelles du spectre turbulent. Ainsi, cette approche perd les informations relatives aux structures turbulentes instationnaires, ce qui constitue un inconvénient majeur pour l'étude des écoulements diphasiques réactifs.

- Simulation aux grandes échelles (SGE, ou LES en anglais) : les grandes échelles de la turbulence sont explicitement résolues, cependant l'effet des plus petites échelles est modélisé avec des modèles de fermeture de sous-maille. Les équations pour la SGE sont obtenues par filtrage des équations instantanées. Cette approche est aujourd'hui largement répandue pour la simulation d'écoulements monophasiques réactifs et montre une capacité manifeste à prédire des phénomènes fortement instationnaires tels que les instabilités de combustion.

La SGE peut être considérée à mi-chemin entre la méthode RANS dans laquelle toutes les échelles de la turbulence sont modélisées et la SND dans laquelle toutes les échelles de la turbulence sont résolues. Dans une SGE, seules les plus grandes échelles - celles qui contiennent le plus d'énergie - sont résolues et l'effet des plus petites échelles est modélisé. Une grande partie du travail

des pionniers sur la SGE (par exemple, Smagorinsky [204], Lilly [117], Deardorff [40]) a été motivée par les applications météorologiques, l'étude des couches limites atmosphériques demeurant un sujet d'intérêt pour la SGE (par exemple, Mason [126]). Le développement et validation des méthodes SGE ont été dirigées principalement sur la turbulence isotrope (par exemple, Kraichnan [104], Chasnov [29]), et le canal turbulent (par exemple, Deardorff [39], Schumann [195], Moin & Kim [134], Piomelli [158]). Un premier objectif dans ce domaine consiste à appliquer la SGE aux écoulements dans des géométries complexes qui sont typiques des applications d'ingénierie (par exemple, Akselvoll & Moin [1], Haworth & Jansen [83]).

Simulation numérique diphasique

L'étude de la combustion turbulente a été développée avec l'utilisation des moteurs à combustion interne et des turbines à gaz. La modélisation et la simulation des écoulements multiphasiques polydisperse est devenue un outil important pour mieux comprendre et contrôler un certain nombre de phénomènes physiques en relation avec la combustion turbulente. La principale raison en est que, dans de nombreux dispositifs industriels, le carburant est stocké sous forme condensée et injecté sous forme liquide (comme des films ou des gouttelettes de taille généralement de 10-200 μm), et il est ensuite mélangé avec de l'air dans la chambre de combustion où il brûle en général par combustion turbulente. Par conséquent, les écoulements réactifs diphasiques combinent les difficultés de la turbulence, de la combustion et des écoulements multiphasiques. La simulation numérique permet d'étudier en détail la plupart de ces phénomènes complexes et d'analyser leurs interactions. La modélisation de la phase dispersée pose la question du choix de la méthode utilisée pour le couplage entre le gaz et le liquide dans une SGE. Lorsque le flux est dilué (ce qui est considéré dans cette thèse), il est possible de simuler les écoulements diphasiques et de supposer que les particules occupent un faible volume par rapport à la phase porteuse de sorte qu'une approche de type point-force (force ponctuelle) peut être considérée. Le calcul numérique des écoulements diphasiques dispersés peut être divisé grosso modo en deux méthodes : la méthode Lagrangienne et la méthode Eulérienne. Lorsque la **méthode Lagrangienne** est utilisée, des forces ponctuelles sont généralement considérées, avec ou sans interaction avec la phase porteuse. Dans ce type de simulations, la phase porteuse est généralement calculée en résolvant les équations de Navier-Stokes. Chaque goutte (ou groupe de gouttes) est calculée individuellement en fonction de sa trajectoire, sa vitesse, sa température et son diamètre. Les forces d'interaction sont calculées à partir du champ local de l'écoulement. Puisque l'emplacement des particules discrètes peut ne pas coïncider nécessairement avec les nœuds de calcul du maillage, les propriétés du fluide à l'emplacement des particules sont obtenues par interpolation. L'autre méthode consiste en l'une des différentes **méthodes Eulériennes** qui s'appuient sur des procédures telles que les moyennes de volume ou la moyenne de l'ensemble pour obtenir des champs continus. Les informations sur les différents parcours et l'état des particules se perdent dans la procédure de moyennage. Dans les applications industrielles, les valeurs moyennes fournissent parfois des informations précieuses et l'état de chaque particule n'a pas d'importance. La phase liquide est résolue de façon homogène pour un ensemble d'équations de conservation de la fraction volumique de liquide, de la vitesse du liquide, de la température et des moments du premier/deuxième ordre de la distribution en taille [147, 180].

Les approches Eulérienne et Lagrangienne sont souvent comparées et ses qualités/défauts mis en évidence. Une liste non-exhaustive de leurs principaux avantages et inconvénients respectifs est présentée ci-dessous (Tableau 1) :

Euler-Euler	Euler-Lagrange
Avantages	
• Simplicité de traitement des zones denses.	• Modélisation simple des mouvements des particules et de leurs interactions.
• Similitude avec les équations gazeuses.	• Méthode robuste et précise si le nombre de particules est suffisant.
• Transport direct des quantités Eulériennes.	• Description simple de la distribution en taille.
• Similitude avec le parallélisme utilisé pour la phase gazeuse.	• implémentation simple de phénomènes physiques (e.g. transfert de chaleur et de masse, interactions particule-paroi).
Inconvénients	
◦ Description de la polydispersion.	◦ Couplage délicat avec la combustion.
◦ Traitement des jets croisés.	◦ Implémentation difficile en parallèle.
◦ Limitation de la méthode en zones très diluées.	◦ Temps passé dans la recherche de particules.

Table 1 - *Avantages et inconvénients des approches Euler-Euler et Euler-Lagrange.*

Il est connu que la SGE est moins dissipative que la méthode RANS. En conséquence, le nombre de gouttes Lagrangiennes à chaque pas de temps et dans chaque cellule doit être suffisant pour assurer un champ continu et précis du combustible gazeux. Ceci est crucial pour les calculs de flammes diphasiques car la distribution de vapeur de carburant, directement produite par les termes sources d'évaporation des gouttes discrètes contrôle la propagation du front de flamme [159, 197]. Une vue d'ensemble des récents progrès dans l'utilisation des SGE sur des géométries complexes à l'aide d'un solveur CFD peut être trouvée dans Moin [132]. Le solveur utilisé (CDP), présente quelques similitudes avec AVBP (décrit au chapitre 2), il est en effet massivement parallèle, traite des maillages non-structurés et dispose d'un module Lagrangien pour la description de la phase dispersée. Dans la géométrie complexe P&W étudiée, environ 3.5 millions de trajectoires de particules sont calculées (ce qui représente environ 15 millions de gouttes physiques) avec un maillage grossier de 2.2 millions de cellules et un maillage plus fin de 4.5 millions de cellules. Les résultats des deux calculs SGE ont été comparés avec les résultats RANS et il en ressort que les solutions avec le maillage plus fin prédisent avec plus de précision le débit en aval de l'injecteur, tandis que les résultats SGE avec le maillage grossier ainsi que les résultats RANS avec le maillage fin ne réussissent pas à capter le jet loin de l'injecteur. Cette simulation (faite il y a presque 4 ans) a montré la faisabilité et les capacités des calculs réactifs SGE en Lagrangien sur les chambres de combustion des turbines à gaz réelles et donne de bonnes perspectives pour la mise en œuvre d'un module Lagrangien dans le solveur utilisé dans cette thèse, AVBP. Bien que l'approche Eulérienne (dite à 'deux fluides' utilisée pour le traitement de la phase liquide et couplée au solveur gazeux) est la solution disponible actuellement dans AVBP ([98, 136, 154, 18, 107, 179, 108]), de nombreuses questions doivent encore être étudiées, telles que le choix des modèles de fermeture et des paramètres du modèle, ainsi que le traitement des gradients de densité locaux qui représente un problème numérique difficile pouvant éventuellement conduire à l'explosion de la simulation, si le schéma numérique et/ou la résolution du maillage ne sont pas adaptés. Pour faire face à ce

genre de problèmes, le CERFACS a décidé en 2005 de développer le formalisme Lagrangien avec l'objectif de disposer d'un module solide et bien parallélisé qui doit être adapté à la complexité du solveur AVBP.

Dans la littérature, d'importants progrès ont été récemment faits pour réduire les différences entre les approches Lagrangienne et Eulérienne. En voici quelques exemples : Apte *et al.* [7] validation du formalisme Lagrangien d'une chambre de combustion avec jet coaxial ; Ham *et al.* [76] validation des écoulements réactifs SGE en Lagrangien sur des chambres de combustion de turbines à gaz réelles ; Réveillon *et al.* [176] analyse de la dispersion des aérosols polydispersée en évaporation dans les écoulements turbulents avec un formalisme Eulérien ; Fede & Simonin [58] étude des effets de turbulence de sous-maille sur les statistiques de la collision des particules lourdes ; Haselbacher *et al.* [82] amélioration de l'efficacité et de la robustesse des algorithmes de localisation des particules sur des maillages non-structurés ; Fréret *et al.* [65] amélioration de la dynamique et de l'évaporation de sprays polydispersée avec un modèle multi-fluide Eulérien ; de Chaisemartin *et al.* [28] faire face au croisement des trajectoires des gouttes avec un modèle multi-fluide Eulérien, etc. Cette thèse suit également cette direction et tente de résoudre les difficultés de mise en œuvre du formalisme Lagrangien en solveurs parallèles et les aspects de la performance sur maillages non-structurés.

Croissance de la puissance de calcul

La figure 1.3 montre la projection des performances du premier (carrés rouges), le 500ème (carrés roses) et la somme des 500 (carrés verts) ordinateurs les plus puissants au monde (source : http://www.top500.org). Cette image est un outil important pour suivre l'évolution historique et pour prédire les tendances futures, par exemple, pour déterminer le moment où le premier système petaflopique sera opérationnel. En ce qui concerne la ligne de tendance de la 500ème machine, on peut observer que les prévisions suivent assez bien le comportement réel. Par contre, les prévisions de l'ordinateur le plus rapide présentent des périodes d'évolution lente (par exemple, 2002-2004), suivies de périodes d'améliorations importantes (par exemple, 2004-2006).

Les ressources informatiques du CERFACS fournissent environ 14 Tflop/s, ce qui n'est pas loin de la puissance fournie par le dernier supercalculateur de la liste TOP500. Toutefois, la puissance du CERFACS est également dans ses ressources extérieures, permettant l'accès à des machines vectorielles (IDRIS et Météo-France) et scalaires (par exemple, le CEA CCRT, CINES, ECMWF, et Marenostrum Argonne) dans le monde entier. Avec ces avantages, et compte tenu de la croissance exponentielle de la puissance de calcul, il était évident pour l'auteur que le potentiel pour le développement d'une formulation Lagrangienne sur un solveur parallèle, non-structuré et hybride comme AVBP repose sur une mise en œuvre efficace sur des machines massivement parallèles et dans l'utilisation des stratégies pour l'équilibrage de charges. L'évolution dans la compréhension des calculs SGE diphasiques devrait permettre une amélioration de la conception des machines (par exemple, les turbines à gaz) pour les rendre plus efficaces. Le principal inconvénient du formalisme Lagrangien, à savoir le fait d'être toujours considéré trop coûteux pour des configurations réelles, sera probablement réduit au cours des prochaines années avec l'augmentation de la performance des machines parallèles et les progrès de l'informatique dans les algorithmes de recherche, l'équilibrage de charges, etc. Par ailleurs, le fait d'avoir un modèle Lagrangien dans un code avec une approche Eulérienne est un élément important qui permet la validation de nouveaux modèles de fermeture Eulériens, ainsi que la comparaison dans les cas où les données expérimentales ne sont pas disponibles. Ni le Lagrangien, ni l'Eulérien n'ont la réponse aux nombreuses questions qui intéressent les scientifiques, mais l'avenir pourra se situer quelque part entre les deux et le fait

de les avoir dans le même code est un avantage clair qui ouvre un certain nombre de nouvelles perspectives pour le CERFACS.

Objectif du travail de thèse

L'objectif principal de cette thèse est d'une part, de montrer la faisabilité des simulations Lagrangiennes efficaces sur machines parallèles, et d'autre part de préparer les bases des calculs futurs pour la prochaine génération de supercalculateurs petaflopiques où des processeurs plus rapides et des algorithmes plus efficaces seront disponibles.

Le travail réalisé tout au long de cette thèse et présenté ici est une nouveauté dans le sens où il représente une implémentation efficace du formalisme Lagrangien dans le solveur parallèle, non-structuré et hybride AVBP. Deux des principales difficultés ont été : (i) le fait de travailler avec un solveur non-structuré qui nécessite une profonde compréhension de la structure interne du code, et (ii) une implémentation efficace en parallèle pour maintenir de bonnes performances spécialement pour des futures simulations sur les machines massivement parallèles. Par rapport à ce dernier point, l'utilisation de l'équilibrage de charges est mis en évidence comme un concept prometteur et une solution efficace quand le déséquilibrage entre la phase gazeuse et la phase liquide est trop important. Une attention particulière a également été portée aux aspects liés à la portabilité du code sur différentes architectures puisque cela représente une caractéristique importante d'AVBP. Le module Lagrangien a été conçu pour être assez modulaire et faciliter les modifications des algorithmes et des modèles disponibles.

Bien que les cas tests étudiés ici ne soient pas des calculs diphasiques sur géométries complexes, ce travail représente une étude de faisabilité des calculs Lagrangiens avec AVBP. Beaucoup d'aspects importants de la description Lagrangienne des écoulements diphasiques, tels que les collisions entre particules, la coalescence et les interactions particule-paroi, n'ont pas été abordés, mais il est certain qu'ils représentent des sujets de recherche de grand intérêt pour les années à venir.

Le contenu de cette thèse est organisé comme suit :

Chapitre 2 : le solveur AVBP

Le projet AVBP est né avec l'idée de construire un outil moderne pour la dynamique des fluides numérique, de grande flexibilité, efficacité et modularité. Il a débuté au CERFACS en Janvier 1993 sous l'initiative de Michael Rudgyard et Thilo Schönfeld. L'objectif était de créer un solveur non-structuré capable de gérer des maillages constitués d'éléments de type quelconque. L'utilisation de ces maillages hybrides permet notamment une grande efficacité à la génération du maillage et une bonne précision des résultats de calcul. La structure d'AVBP est fortement inspirée des bibliothèques logicielles pour répondre au mieux aux exigences de modularité.

AVBP est un code de calcul CFD qui permet de résoudre les équations laminaires et turbulentes compressibles de Navier-Stokes en deux et trois dimensions. Pour les écoulements turbulents stationnaires, deux modèles de turbulence de type RANS sont disponibles (cette option n'est plus disponible depuis 2002 (V4.8)), alors que pour la prédiction de la turbulence instationnaire, différents modèles de sous-maille ont été implémentés. AVBP a été conçu initialement pour le calcul des écoulements stationnaires externes pour les applications d'aérodynamique. Depuis le milieu des années quatre-vingt-dix, l'accent a été mis sur la modélisation des écoulements turbulents insta-

tionnaires (avec et sans réactions chimiques) principalement pour les configurations d'écoulements internes. Ces activités sont en partie liées à la croissance de la compréhension de la structure des écoulements et les mécanismes en relation avec la turbulence. La prédiction de ces écoulements turbulents instationnaires est fondée sur l'approche LES qui a émergé comme une technique prometteuse pour l'étude des problèmes associés aux phénomènes dépendants du temps et à la cohérence des structures des tourbillons.

Le traitement des maillages non-structurés ou hybrides est un élément clé d'Avbp. Avec l'utilisation de ces maillages hybrides, où une combinaison de plusieurs éléments de différents types est utilisée sur le même maillage, les avantages des maillages structurés et non-structurés sont combinés en termes de flexibilité et de précision de la solution. Afin de traiter des maillages hybrides arbitraires, la structure des données d'Avbp utilise une approximation cell-vertex volume fini. Le deux schémas numériques les plus utilisés sont basés sur une discrétisation de type Lax-Wendroff [110, 111] ou éléments finis Taylor-Galerkin [43, 44, 171, 33] en combinaison avec un modèle de viscosité artificielle.

Avbp est construit sur une bibliothèque logicielle modulaire qui inclut le découpage de domaine de calcul et des outils de réorganisation de données, le contrôle des échanges de messages, gère les routines d'allocation dynamique de mémoire, les routines parallèles I/O et les méthodes itératives. Avbp est écrit en Fortran 77 et C, mais il est progressivement traduit en Fortran 90. Une de ses principales caractéristiques est sa portabilité et son bon speedup sur différentes architectures et machines parallèles.

Avbp est actuellement développé par plus de 30 doctorants et Post-Docs en collaboration avec des chercheurs et des ingénieurs. Aujourd'hui, la propriété d'Avbp est partagée entre le CERFACS et l'Institut Français du Pétrole (IFP), situé dans la région parisienne. Le développement du code est orienté principalement vers les applications turbines à gaz et moteurs à piston. Avbp est utilisé dans le cadre de nombreuses collaborations industrielles bilatérales et pour des programmes de recherche. Au niveau européen, il est utilisé dans plusieurs projets des programmes FP5, FP6 et FP7 de la Communauté Européenne (CE) et plusieurs chercheurs l'utilisent dans le cadre des projets Marie Curie. Des liens importants ont également été établis avec l'industrie, par exemple : Groupe Safran (Snecma, Turbomeca), Air Liquide, Gaz de France ainsi qu'Alstom et Siemens Power Generation.

Ce chapitre donne un aperçu général des principales caractéristiques du solveur Avbp. La section 2.1 décrit les équations de SGE compressibles de Navier-Stokes et les modèles disponibles pour les modèles de sous-maille. La formulation cell-vertex utilisée pour la discrétisation des équations est présentée dans la section 2.2. Puis, la section 2.3 présente une brève description des conditions aux limites. Le découpage de maillage est décrit en détail dans la section 2.4 ainsi que les principales raisons de l'introduction d'un nouvel algorithme de partitionnement. Les sections 2.5 et 2.6 donnent quelques notions de base sur la structure de données et le parallélisme utilisés dans Avbp. Enfin, la section 2.7 est consacrée à l'étude de la croissance des erreurs d'arrondi et la reproductibilité des simulations aux grandes échelles. Les principales conclusions sont données à la fin du chapitre (section 2.8).

Chapitre 3 : implementation numérique du module Lagrangien

Ce chapitre présente les principales caractéristiques de l'implémentation du module Lagrangien dans le solveur AVBP pour le traitement de la phase dispersée. La section 3.1 présente les équations qui décrivent le mouvement des particules. Les forces agissant sur les particules sont d'abord rappelées pour en suite présenter les équations retenues pour cette étude. L'implémentation efficace des équations dans le code dépend des structures de données considérées. La section 3.2 présente brièvement les structures de données utilisées dans le module Lagrangien. Les variables Lagrangiennes sont stockées séparément dans deux vecteurs, un de type réel et un autre de type entier. Les pointeurs utilisés pour localiser les particules qui traversent les interfaces entre les processeurs sont également décrits. Les particules sont localisées à chaque pas de temps à l'intérieur du maillage avant de les avancer, en sachant que les éléments du maillage non-structuré peuvent être de forme arbitraire. La section 3.3 présente le critère de localisation des particules utilisé avec les éléments disponibles dans AVBP. Les différents algorithmes de recherche utilisés sont résumés dans la section 3.4. Les différentes situations dans lesquelles les particules doivent être localisées, sont également examinées dans cette section. Une fois que les particules se trouvent à l'intérieur d'une cellule, elles récupèrent les informations de la phase gazeuse avant de changer de position. Les algorithmes d'interpolation utilisés pour calculer les propriétés du fluide à l'endroit où la particule se trouve sont présentés dans la section 3.5. Dans certains cas (par exemple, le couplage inverse), les particules sont également censées d'échanger des informations avec la phase gazeuse. La méthode de couplage inverse entre la phase gazeuse et la phase dispersée est présentée et validée dans la section 3.6. Pour conclure, les modèles d'injection de particules dans un point ou dans un disque sont décrits dans la section 3.7.

Chapitre 4 : calcul d'une turbulence homogène isotrope

Le mélange et la dispersion de particules sont des sujets d'intérêt certain de la mécanique des fluides, non seulement d'un point de vue académique, mais aussi dans un contexte industriel. La dynamique des particules sous l'effet de la turbulence est étudiée pour un grand nombre d'applications, du transport atmosphérique des polluants, au transport des particules de charbon dans les centrales électriques ; de la dispersion dans les turbines à gaz aux lits fluidisés de l'industrie chimique.

Deux des processus fondamentaux qui caractérisent les interactions particule/turbulence sont la dispersion de particules par la turbulence et la modification de la turbulence par les particules. Le premier est généralement étudié avec l'hypothèse que les propriétés du champ turbulent ne sont pas modifiées par la présence des particules (couplage direct). Ce dernier a été étudié expérimentalement (Snyder & Lumley [205], Wells & Stock [226]) et récemment par des simulations numériques qui facilitent l'obtention des statistiques très détaillées à partir de la trajectoire de chaque particule (Squires & Eaton [207], Elghobashi & Truesdell [51], Deutsch [42], Elghobashi & Truesdell [52], Février *et al.* [61], Moreau [135]). L'analyse théorique de la dispersion de particules a été lancée par Tchen [216] entre autres (Reeks [173, 174]), conduisant à une définition du temps et des échelles qui caractérisent le comportement des particules dans la turbulence. Les méthodes analytiques mises au point par Maxey [128] ont permis la prédiction du phénomène bien connu de la concentration préférentielle, c'est-à-dire, l'accumulation de particules inertielles dans les régions de basse vorticité. Ces phénomènes ont été précédemment étudiés avec l'approche Lagrangienne (Deutsch [42], Elghobashi & Truesdell [52], Février *et al.* [61], Moreau [135]).

Dans ce chapitre, les résultats d'une SND d'une turbulence homogène isotrope (THI) en utilisant le module Lagrangien développé pendant cette thèse sont validées par comparaison avec les résultats d'un autre solveur Lagrangien, NTMIX3D. Ce code, entièrement parallèle, utilise la méthode des différences finies. La partie relative à la phase gazeuse a été principalement développée au CERFACS par Baum & Poinsot [11], Poinsot *et al.* [162], Stoessel *et al.* [210], Cuenot & Poinsot [36], Boughanem & Trouvé [24] alors que la partie diphasique a été réalisée par Albrecht *et al.* [2], Vermorel *et al.* [220], Paoli *et al.* [151], Moreau [135], Paoli & Shariff [153], entre autres. Pour la phase gazeuse, les équations compressibles de Navier-Stokes sont résolues avec une formulation adimensionnelle. La discrétisation spatiale est faite avec un schéma compact du sixième ordre (Lele [114]), tandis que la discrétisation temporelle est effectuée par un schéma à trois étapes de type Runge-Kutta. Pour la phase dispersée, l'équation d'avancement des trajectoires des particules est faite avec ce schéma. Une interpolation basée sur les polynômes de Lagrange du quatrième ordre est utilisée pour calculer la vitesse du fluide le long des trajectoires des particules. Cette vitesse est déterminante pour le calcul de la force de traînée, des termes sources de condensation/évaporation, etc.

Le code NTMIX3D est un code structuré développé initialement pour la SND. La comparaison des résultats d'Avbp, conçu pour les SGE, avec ce code est un exercice intéressant. Les résultats montrent que la solution numérique du cas test considéré, obtenue avec le schéma spatial du troisième ordre d'Avbp (TTGC) est en très bon accord avec celle obtenue avec NTMIX3D (sixième ordre). Les résultats mettent également en évidence les effets et l'importance des algorithmes d'interpolation pour la phase dispersée.

Le chapitre est organisé comme suit : la section 4.1 passe en revue quelques notions de base de la turbulence. La section 4.2 présente les principales catégories dans lesquelles les écoulements THI sont classés. La section 4.3 résume les fondements théoriques de la THI et présente les variables qui sont utilisées pour valider la dispersion de particules avec le modèle Lagrangien. La section 4.4 présente les principaux paramètres utilisés pour le calcul monophasique et le spectre d'énergie obtenu avant d'introduire la phase dispersée. Les résultats du calcul diphasique sont présentés dans la section 4.5 et les conclusions dans la section 4.6.

Chapitre 5 : calcul d'un jet recirculant confiné (ou "bluff body") chargé en particules

Avant de réaliser de calculs diphasiques sur de géométries complexes, il est nécessaire de valider l'effet de la turbulence sur la dispersion de particules. Le cas test choisi pour valider les développements du module Lagrangien est un jet chargé en particules de type "bluff body"(configuration étudiée par Borée *et al.* [22]) où des particules de verre sont injectées dans un écoulement turbulent comportant une zone de recirculation. Cette configuration est typique d'une application industrielle où l'objectif est de contrôler le mélange du combustible avec l'air. Ces tests sont effectués sans combustion ni évaporation. Le choix de cette configuration est motivé par plusieurs raisons : la présence d'une zone de recirculation qui permet d'étudier la dispersion de particules dans une configuration où des particules de petite taille sont capturées à l'intérieur de la zone de recirculation, tandis que les plus grandes la traversent et arrivent a y échapper vers l'aval de la chambre ; la relative simplicité de la géométrie par rapport aux configurations plus réalistes, et, plus important encore, la grande quantité de données disponibles pour le calcul monophasique et le calcul diphasique (l'ensemble complet des mesures expérimentales, y compris les conditions aux limites, a été choisi comme cas test du

'9ième workshop pour la prédiction des écoulements diphasiques' Ishima *et al.* [92] et est disponible en ligne à l'adresse suivante : http://www-mvt.iw.uni-halle.de/english/index.php?bluff_body_flow).

Les résultats des simulations obtenues avec le modèle Lagrangien d'AVBP sont comparés avec les résultats d'un autre solveur Lagrangien développé à l'Université de Stanford (CDP) et aussi avec les résultats expérimentaux. Les résultats du solveur CDP pour la phase gazeuse et la phase dispersée ont été fournis par Vincent Moureau. Une étude plus complète de l'influence de différents paramètres (type de maillage, schéma numérique et conditions aux limites d'entrée) pour le cas monophasique peut être consultés dans l'article publié dans le *Journal of Computational Physics* (Vol. 228, N° 2, pp. 539-564, 2009), inclu dans l'annexe E.

Ce chapitre est organisé comme suit : la section 5.1 décrit la configuration de Borée *et al.* [22]. Les paramètres numériques sont présentés dans la section 5.2. Les sections 5.3 à 5.5 comparent les résultats obtenus avec les deux codes et les données expérimentales pour le calcul monophasique, les calculs diphasiques monodisperse et polydisperse, respectivement. La section 5.6 présente une analyse des performances du code et plus précisément de son efficacité sur des machines parallèles.

Conclusions et perspectives

Conclusions

Comprendre les phénomènes de combustion est un élément clé pour améliorer l'approvisionnement en énergie, pour obtenir de meilleures performances et pour réduire la consommation de la plupart des équipements industriels, mais aussi parce que la combustion contrôle directement les conditions de formation des polluants. La combustion est hautement non-linéaire et est un processus complexe dans lequel la chimie, la mécanique des fluides, la thermodynamique, le rayonnement et le changement de phases sont fortement couplés. Les premiers pas dans la compréhension des phénomènes de combustion ont été obtenus de façon expérimentale, mais le potentiel de la simulation numérique comme un outil pour l'étude de ces phénomènes, a augmenté considérablement au cours des dernières années avec l'application de la simulation numérique directe (SND) et la simulation aux grandes échelles (SGE) dans le cadre des écoulements diphasiques avec combustion. Dans de nombreux dispositifs industriels, le carburant est stocké sous forme condensée et injecté sous forme liquide dans la chambre de combustion où il est mélangé au comburant et brûlé en général par le biais d'un processus de combustion turbulente. Afin de comprendre la physique des écoulements diphasiques réactifs, le formalisme Lagrangien est proposé pour le traitement de la phase dispersée. Une des motivations de cette étude est l'augmentation rapide de la puissance de calcul qui ouvre une nouvelle voie pour des simulations considérées prohibitives, il y a une décennie.

L'objectif de cette thèse est le développement et la validation d'un formalisme Lagrangien ainsi que son implémentation dans le solveur parallèle et non-structuré AVBP, pour les calculs SGE réactifs. Ce solveur permet de résoudre les équations laminaires et turbulentes compressibles de Navier-Stokes en deux et trois dimensions. Le traitement des maillages non-structurés ou hybrides est un élément clé d'AVBP et représente un défi majeur pour l'implémentation des structures de données du module Lagrangien. Un autre élément analysé dans cette thèse est l'étude de nouveaux algorithmes de découpage du maillage afin d'améliorer les performances des calculs dans les machines massivement parallèles en réduisant la taille des sous-domaines et le temps CPU lié au découpage lui-même. Une étude de la performance des algorithmes de partitionnement est faite et le besoin d'un nouvel algorithme de partitionnement est mis en évidence. L'algorithme retenu fait

partie du package METIS qui contient des algorithmes de partitionnement multi-poids en versions séquentielle et parallèle. Cette fonctionnalité a été utilisée pour permettre l'équilibrage de charges dans la version Lagrangien développée au cours de cette thèse. Une étude comparative entre le nouvel algorithme de partitionnement et ceux qui sont déjà disponibles montre une réduction significative du temps CPU utilisé pour découper le maillage et une réduction du nombre de nœuds dupliqués après le partitionnement.

L'étude de la sensibilité des systèmes chaotiques aux conditions initiales est liée au fait d'utiliser un solveur SGE parallèle. Tout écoulement SGE turbulent montre une certaine sensibilité aux petites perturbations, ce qui conduit à des solutions instantanées qui peuvent être totalement différentes. Au contraire, les écoulements laminaires sont presque insensibles à ces perturbations, même avec des conditions périodiques. Une raison de la divergence des solutions est la propagation des erreurs d'arrondi dans un écoulement naturellement instable (turbulent) à cause des techniques de découpage du domaine ou de l'ordre des opérations. L'effet des différents paramètres a été étudié et les résultats ont fait l'objet d'une publication au *Journal AIAA* (Vol. 46, N° 7, pp. 1773-1781, 2008) (voir l'annexe D).

Le module diphasique Lagrangien a été validé pour une turbulence homogène isotrope (THI), qui permet une analyse simple des différents aspects liés à la performance du code et du comportement des particules. Tout d'abord, une analyse de la performance de l'algorithme de recherche de particules a été réalisée. L'algorithme de type arbre (octree) utilisé dans cette thèse a été comparé à un simple algorithme de type force brute pour un nombre différent de particules par cellule, et pour un nombre différent de processeurs. Comme prévu, l'algorithme octree est sensiblement plus rapide que l'algorithme de type force brute. Ensuite, les résultats de l'énergie cinétique du fluide et des particules ont été analysés et comparés aux résultats d'un autre solveur Lagrangien, NTMIX3D, d'ordre plus élevé. Les comparaisons montrent que les résultats obtenus avec le schéma spatial de troisième ordre (TTGC) sont proches de ceux obtenus avec NTMIX3D (sixième ordre), ce qui permet une première validation du formalisme Lagrangien.

Le deuxième cas test a été choisi pour valider le module Lagrangien dans des configurations similaires à celles rencontrées dans des chambres de combustion réelles, en particulier les propriétés de dispersion de particules dans la turbulence. Le cas test considéré est étudié dans Borée *et al.* [22], où des particules de verre sont injectées dans un écoulement turbulent comportant une zone de recirculation. Dans cette configuration où les particules sont inertielles, mais très dépendantes du gaz, la prédiction de mouvement des particules dépend fortement de la phase gazeuse. Les résultats obtenus avec le formalisme Lagrangien sont en bon accord avec les expériences, et avec les résultats fournis par le code de calcul CDP développé à l'Université de Stanford. D'une manière générale, l'évolution axiale et radiale des profils moyens et RMS des vitesses des particules dans le cas monodisperse et polydisperse est en bon accord avec les résultats de CDP. Cela permet une validation des développements faits dans le module Lagrangien. En ce qui concerne les résultats gazeux, le cas monodisperse et polydisperse sont très similaires. La principale différence est le niveau de la vitesse moyenne du gaz dans la zone de recirculation qui est plus faible que prévu. Cela implique une réduction de la taille de la zone de recirculation pour le cas polydisperse. Comme il n'y a pas de différence entre le temps de calcul du cas monodisperse et celui polydisperse, simuler l'écoulement chargé en particules avec une distribution polydisperse est plus proche des expériences et de la réalité. L'effet du nombre de particules sur les profils de vitesse radiale à trois instants différents a été étudié pour le cas polydisperse. Les profils de vitesse moyenne et RMS des particules montrent une amélioration importante avec le cas où le temps utilisé pour faire les moyennes est le plus grand.

Une étude de la scalabilité de la version Lagrangienne d'AVBP a été effectuée sur un supercalculateur Cray XD1 du CERFACS qui compte jusqu'à 64 processeurs. Aucun problème particulier lié au équilibrage de charges n'a été observé avec le cas hexaédrique qui présente un speedup très bon. Les résultats avec le maillage tétraédrique conduisent à un mauvais speedup dans le cas où la présence des particules n'est pas prise en compte au moment du découpage, ce qui génère un déséquilibrage de charges. Néanmoins, l'équilibrage des particules avec des algorithmes multi-poids permet une amélioration considérable du speedup. En conséquence, pour préserver une bonne efficacité sur des machines massivement parallèles dans les cas où le déséquilibrage est important, il est indispensable d'utiliser des stratégies d'équilibrage de charges. Les résultats obtenus dans cette thèse confirment la position concurrentielle du formalisme Lagrangien par rapport au formalisme Eulérien une fois que ce type de stratégie est utilisé. Tous ces résultats ont fait l'objet d'une publication dans *Journal of Computational Physics* (voir l'annexe E).

Perspectives

Malgré tout le travail accompli pendant cette thèse, un certain nombre de développements importants doivent encore se mettre en place. On peut citer entre autres : l'introduction d'un modèle d'évaporation pour les calculs diphasiques réactifs, le traitement des interactions particule-paroi, l'introduction des modèles de collision et de coalescence, l'amélioration de l'injection des particules (avec nouvelles géométries, nouvelles distributions de taille des particules et des options d'injection multi-point), l'introduction de la composante de sous-maille de la vitesse du gaz dans le calcul de la vitesse des particules, l'amélioration des algorithmes de recherche, etc.

Pendant cette thèse, plus précisément au début de l'année 2007, deux nouvelles thèses associées au projet européen ECCOMET (*Efficient and clean combustion experts training*, FP6) ont commencées sur la version Lagrangienne pour résoudre un nombre important de ces problèmes :

- F. Jaegle a mis en place un modèle d'évaporation qui a été validé pour de cas tests académiques et il travaille actuellement sur les interactions particule-paroi ;
- J.-M. Senoner travaille sur l'amélioration des options d'injection de particules.

Dans les deux cas, certains de leurs développements font partie de la version officielle Lagrangienne et les autres seront inclus dans un futur proche.

Chapter 1

Scientific context

1.1 Energy resources

Life on Earth is driven by energy and human beings are high consumer of energy and energy resources (consumption of fossil fuels has nearly doubled every 20 years since 1900). But the exhaustion of fossil fuels, which supply three quarters of this energy, is not far off, and no other energy source is abundant and cheap enough to take their place. At the current time, the main alternatives to combustion are:

- **Nuclear power** produces around 11% of the world's energy needs, and produces huge amounts of energy from small amounts of fuel, without the pollution that you'd get from burning fossil fuels. However, although not much waste is produced, it is very dangerous and despite of high security measures nuclear accident can be a major disaster.
- **Solar power** is a good alternative since every minute, enough energy arrives at the Earth to meet our demands for a whole year. Solar energy is free, it needs no fuel and produces no waste or pollution. The main disadvantage is that is quite expensive, and it doesn't work at night but current technologies are making important improvements.
- **Wind power** is an energy source used for a long time. Wind is free and produces no waste or greenhouse gases but it is not always predictable.
- **Hydroelectric power**, nowadays there are many hydro-electric power stations, providing around 20% of the world's electricity. Although there are many suitable sites around the world, hydro-electric dams are very expensive to build. However, once the station is built, the water comes free of charge, and there is no waste or pollution.
- **Tidal power**, the tide moves a huge amount of water twice each day. Although the energy supply is reliable and plentiful, converting it into useful electrical power is not easy. A major drawback of tidal power stations is that they can only generate when the tide is flowing in or out - in other words, only for 10 hours each day. Fortunately, tides are totally predictable.
- **Waves power**, waves are a powerful source of energy. Ocean waves are caused by the wind as it blows across the sea. The problem is that it's not easy to harness this energy and convert it into electricity in large amounts. Thus, wave power stations are rare.

- **Geothermal power** does not produce any pollution, and does not contribute to the greenhouse effect but the big problem is the lack of sites that are suitable to build a geothermal power station.
- **Bioconversion** uses plant and animal wastes to produce biofuels such as methanol, natural gas, and oil. It makes sense to use waste materials but collecting or growing the fuel in high quantities can be difficult and it also generates greenhouse gases just like fossil fuels do.

It is well known that combustion of hydrocarbon fuel and air can produce a large amount of carbon mono-(CO) and di-oxide(CO_2) and oxides of nitrogen (NO_x). The former contribute largely to the greenhouse effect while the latter creates photochemical smog and, at ground level, is a main precursor for ozone, which is a toxic compound and a major problem in today's urban environment. Burning fossil fuels is increasingly difficult to avoid so the key question is to burn them with high efficiencies and reduced pollutant levels, so as (i) to counteract the increase of the price of fossil fuels and (ii) to limit the contribution to global climate warming.

Presently many new large electric power-plants use gas turbines and burn natural gas. Even though it is one of the most efficient power sources (up to 57% efficiency for present designs), increasingly stringent legislations have forced manufacturers to increase the turn-over of the research and development phases in order to have cleaner, cheaper and more efficient designs. Empirical formulas and experimental studies were used in the past for the design phase. Today, numerical simulation, in combination with these tools, is widely used for research and designs purposes.

1.2 Energy production by combustion: gas turbines

A gas turbine (Fig. 1.1) is a rotary machine designed to produce energy by combustion and it consists of three main components - a compressor (in blue), a combustion chamber (in yellow) and a turbine (in red). The air after being compressed into the compressor is heated either by directly burning fuel in it or by burning fuel externally in a heat exchanger. The heated air with or without products of combustion is expanded in a turbine resulting in work output, a substantial part (about two-thirds) of which is used to drive the compressor. The rest is available as useful work output.

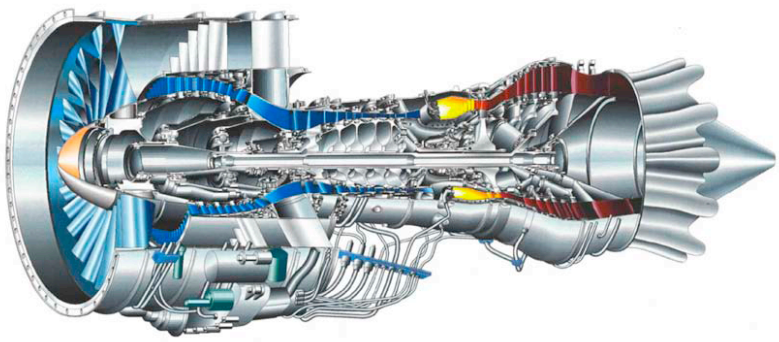


Figure 1.1 - *Gas turbine. The three main components: a compressor, a combustion chamber and a turbine are colored in blue, yellow and red, respectively.*

A gas turbine extracts energy from a flow of hot gas produced by combustion of gas or fuel oil in a stream of compressed air. It has an upstream air compressor (radial or axial flow) mechanically

coupled to a downstream turbine and a combustion chamber in between. Energy is released when compressed air is mixed with fuel and ignited in the combustor. The resulting gases are directed over the turbine's blades, spinning the turbine, and mechanically powering the compressor. Finally, the gases are passed through a nozzle, generating additional thrust by accelerating the hot exhaust gases by expansion back to atmospheric pressure. Energy is extracted in the form of shaft power, compressed air and thrust, in any combination, and used to power aircraft, trains, ships, electrical generators, and even tanks.

Even if there is no direct application to a realistic combustion chamber of the developments done in this thesis, the application test cases presented reproduce some of their most important phenomena: particle motion on turbulence. As the context of this PhD is focused on the numerical simulation of this kind of devices, the main approaches used to resolve numerically the equations used to describe these phenomena are presented hereafter.

1.3 Numerical simulation of turbulent flows

There are three main approaches to describe turbulent combustion processes using Computational Fluid Dynamics (CFD):

- **Direct numerical simulation (DNS):** in this kind of simulations, the full instantaneous Navier-Stokes equations are solved without any model for turbulent motions. All turbulence scales are explicitly determined and their effects on combustion are captured. DNS would predict all time variations of temperature exactly like a high-resolution sensor would measure them in an experiment. Developed in the last twenty years thanks to the development of high performance computers, DNS have changed the analysis of turbulent combustion but are still limited to simple academic flows (e.g., combustion in a small cubic box). In this kind of simulations, the whole spectrum of physical phenomena to represent is solved. This means that the smallest turbulence scale and all the reacting zones of the flow are solved.

- **Reynolds-averaged Navier-Stokes (RANS):** computations have historically been the first approach because the computation of the instantaneous flow field in a turbulent flame was impossible. Therefore, RANS techniques were developed to solve for the mean values of all quantities. The balance equations for Reynolds of Favre (i.e., mass-weighted) averaged quantities are obtained by averaging the instantaneous balance equations. The average equations require closure models: a turbulence model to deal with the flow dynamics in combination with a turbulent combustion model to describe chemical species conversion and heat release. Solving these equations provide averaged quantities corresponding to averages over different realizations (or cycles) for periodic flows like those found in piston engines, i.e., phase averaging.

- **Large-eddy simulations (LES):** the turbulent large scales are explicitly calculated whereas the effects of smaller ones are modeled using subgrid closure models. The balance equations for LES are obtained by filtering the instantaneous balance equations. LES determine the instantaneous position of a "large scale" resolved flame front but a subgrid model is still required to take into account the effects of small turbulent scales on combustion. LES would capture the low-frequency variations of temperature.

RANS, LES and DNS properties are summarised in terms of energy spectrum in Fig. 1.2, where the turbulence kinetic energy spectrum E_d is plotted as a function of the inverse length scale (which is proportional to the associated frequency, normally expressed by the wave number $k = 2\pi/d$). It can be seen that the large turbulent structures (l_t) carry the main part of the turbulence kinetic energy which flows through the Kolmogorov cascade and is dissipated at the smallest scales (η_k).

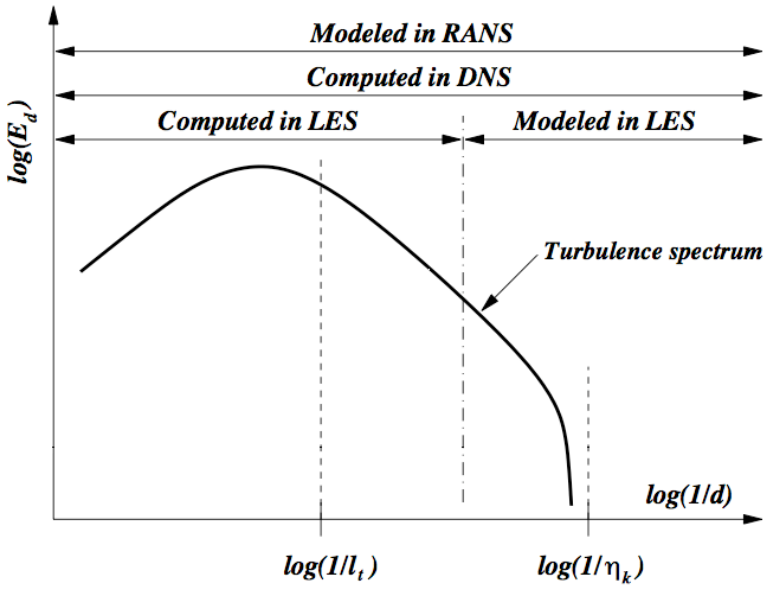


Figure 1.2 - *Turbulence kinetic energy spectrum plotted as a function of the inverse length scale (proportional to the wavenumber). RANS, LES and DNS are summarised in terms of spatial frequency range.*

The LES can be considered as a midpoint between the RANS (Reynolds-averaged Navier-Stokes) approach in which all the turbulent scales are modeled and the DNS in which all the turbulent scales are computed. In a LES simulation, only the largest scales - the scales that contain most of the energy - are computed; the effect of the smallest scales are modeled. The smallest scales have a more predictable behaviour and should be easier to model. Much of the pioneering work on LES (e.g., Smagorinsky [204], Lilly [117], Deardorff [40]) was motivated by meteorological applications, and atmospheric boundary layers remain a focus of LES activities (e.g., Mason [126]). The development and testing of LES methodologies have focused primarily on isotropic turbulence (e.g., Kraichnan [104], Chasnov [29]), and on fully-developed turbulent channel flow (e.g., Deardorff [39], Schumann [195], Moin & Kim [134], Piomelli [158]). A primary goal of work in this area is to apply LES to flows in complex geometries that occur in engineering applications (e.g., Akselvoll & Moin [1], Haworth & Jansen [83]).

1.4 Two-phase flows numerical simulation

The study of turbulent combustion has developed with the use of internal combustion engines and gas turbines. The modeling and simulation of polydisperse multiphase flows has become an important tool to understand and control a number of physical phenomena that involve turbulent combustion. The main reason is that in many industrial devices, fuel is stored in condensed form and injected as a dispersed liquid phase (as films or small droplets typically of size of 10-200 μm),

and it is then mixed with the air in the combustion chamber where it burns usually via turbulent combustion. Therefore, reactive two-phase flows combine the difficulties of turbulence, combustion and multi-phase flow, and numerical simulation allows one to study in detail many of these complex phenomena and to analyse their interactions. Modeling of the dispersed phase raises the question of the choice of the method used to couple the liquid and the gas phases in a LES. When the flow is dilute (as is considered in this thesis), one possibility to simulate two-phase flows is to assume that particles occupy a small volume compared to the carrier phase so that a point force approximation can be made. Numerical computations of dispersed two-phase flow can be divided roughly in two methods: Lagrangian particle tracking and Eulerian methods. When using **Lagrangian particle tracking**, point forces on an inclusion with or without interaction with the carrier phase are usually assumed. In this kind of simulations the carrier phase is typically computed by solving the Navier-Stokes equations. Each droplet (or group of droplets) is computed individually in terms of its trajectory, velocity, temperature and diameter. The interaction forces are computed from the local flow field. Since the location of the discrete particle does not necessarily coincide with the numerical grid of the carrier phase, the flow field at the particle location is obtained by interpolation. The other method consists of one of the different **Eulerian methods** that rely on averaging procedures such as volume or ensemble averaging to obtain continuous fields. The information on the individual particle path and status is lost in the averaging procedure. In industrial applications mean values sometimes already provide valuable information and the status of the individual particle is unimportant. The liquid phase is homogenized and solved for using a set of conservation equations for the liquid volume fraction, the liquid phase velocity and temperature and the first/second order moments of the size distribution [147, 180].

Eulerian and Lagrangian approaches are usually compared and their qualities (or shortcomings) are highlighted. A non-exhaustive list of the main advantages and drawbacks of both formulations is presented in the following table:

Euler-Euler	**Euler-Lagrange**
Advantages	
• Numerically straightforward treatment of dense zones.	• Numerically straightforward modeling of particle movements and interactions.
• Similarity with gaseous equations.	• Robust and accurate if enough particles are used.
• Direct transport of Eulerian quantities.	• Size distributions easy to describe.
• Similarity with gaseous parallelism.	• Numerically straightforward to implement physical phenomena (e.g., heat and mass transfer, wall-particle interaction).
Drawbacks	
◦ Difficult description of polydispersion.	◦ Delicate coupling with combustion.
◦ Difficulty of droplet crossing treatment.	◦ Difficult parallel implementation.
◦ Limitation of the method in very dilute zones.	◦ CPU time spent in locating particles on unstructured grids.

Table 1.1 - *Advantages and drawbacks of Euler-Euler and Euler-Lagrange approaches.*

It is known that the LES technique is less dissipative than RANS methods. As a consequence, the number of Lagrangian droplets at each time step in each cell must be sufficient to provide a smooth and accurate continuous field of gaseous fuel. This is crucial for two-phase flame computations because the fuel vapor distribution, directly produced by the discrete droplet evaporation source terms, controls the propagation of the flame front [159, 197]. An overview of the recent advances in performing LES in complex combustor geometries using a CFD solver can be found in Moin [132]. The solver used (CDP) has some similarities with AVBP (described in Chapter 2): massively parallel, unstructured grid solver and disposes of a Lagrangian particle tracking module for the dispersed phase. In the complex P&W geometry studied, around 3.5 million particle trajectories are computed (which represent approximately 15 million physical droplets) within a coarser grid of 2.2 million cells and a finer grid of 4.5 million cells. Results of both LES simulations were compared with RANS results and it was found that the fine grid LES solution accurately predicts the flow downstream of the injector, while the coarse grid LES as well as the fine grid RANS fail to capture the jet spreading away from the injector. This simulation (performed almost 4 years ago) showed the feasibility and capacities of LES Lagrangian reacting simulations on realistic gas turbine engine combustors and gives good expectations for the result of the implementation of a Lagrangian module within the solver used in this thesis, AVBP. Although the 'two-fluid' approach (Eulerian formulation used for the liquid phase and coupled to the LES solver) is the current solution available in AVBP ([98, 136, 154, 18, 107, 179, 108]), there are still many open questions that have to be studied such as the choice of closure models and model parameters, as well as the treatment of local high-density gradients which represents a tough numerical issue that can eventually lead to the blow-up of the simulation if the numerical scheme and/or the grid resolution are not adapted. To face this kind of problems, CERFACS decided in 2005 to develop the Lagrangian formulation with the objective of having a robust and well parallelised module that has to be adapted to the complexity of the AVBP solver.

In the literature, important advancements have been recently done to reduce differences between Eulerian and Lagrangian approaches, some examples are: Apte *et al.* [7] to validate Lagrangian formulation on a coaxial-jet combustor; Ham *et al.* [76] to validate Lagrangian LES reacting flows in realistic gas turbine combustors; Réveillon *et al.* [176] to analyse dispersion of evaporating polydisperse sprays in turbulent flows with an Eulerian framework; Fede & Simonin [58] to study subgrid turbulence effects on the statistics of heavy colliding particles; Haselbacher *et al.* [82] to improve efficiency and robustness of particle-localisation algorithms on unstructured grids; Fréret *et al.* [65] to better capture dynamics and evaporation of polydisperse sprays with an Eulerian multi-fluid model; de Chaisemartin *et al.* [28] to cope with crossing droplet trajectories with an Eulerian multi-fluid model, etc. This PhD follows also this direction trying to address the difficulties of Lagrangian implementation in parallel solvers and performance aspects on unstructured grids.

1.5 Growth of computer power in the last years

Figure 1.3 shows the projected performance development of the first (red squares), the 500th (pink squares) and the sum of all the 500 (green squares) most powerful computers in the world (source: http://www.top500.org)[1]. This graph provides an important tool to track historical development and to predict future trends, for example, to identify when the first petaflop system will be op-

[1] The 499th powerful computer is the new IRLIN 504 - Supermicro QC Opteron 2.3 GHz - 16.78 TFlops, located in the Institut Français du Pétrole (IFP) who shares with CERFACS the ownership of the solver AVBP.

erational. Concerning the 500th trend line, it can be observed that previsions follows quite well the real behaviour. On the contrary, previsions of the fastest computer presents periods of slow developments (e.g., 2002-2004), follow of periods of important improvements (e.g., 2004-2006).

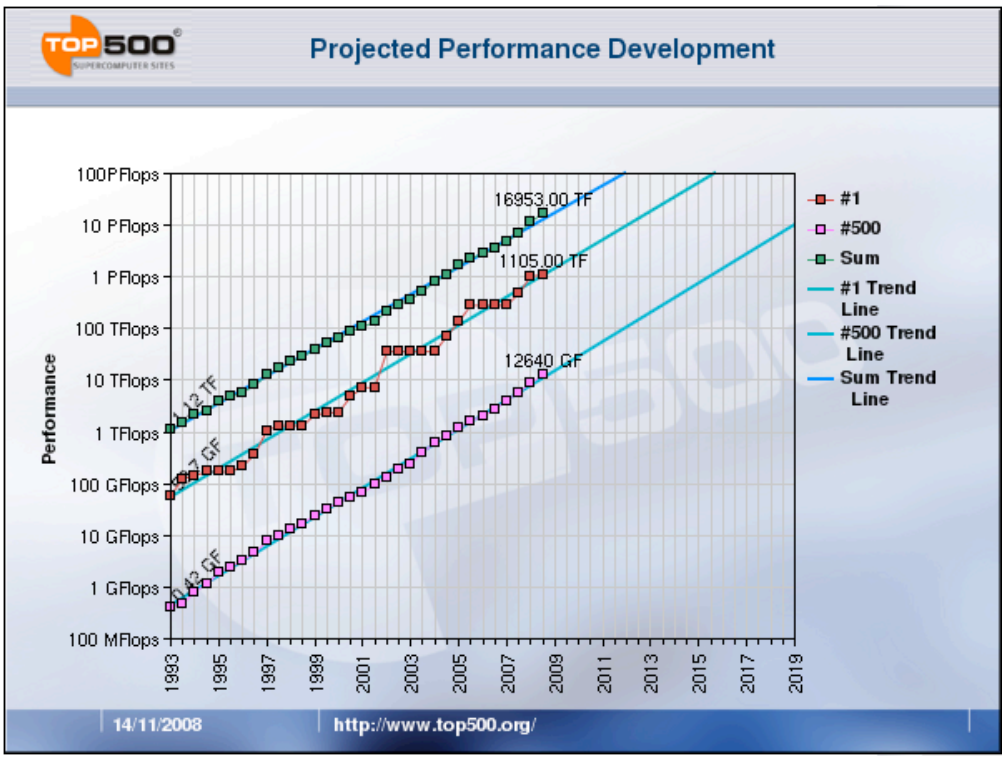


Figure 1.3 - *Projected performance graph. The first (red squares), 500th (pink squares) and the sum of all the 500 (green squares) most powerful computers in the word. From http://www.top500.org.*

CERFACS' computing servers deliver about 14 Tflop/s, which is not far from the power supplied by the last supercomputer of the Top500 list. However, CERFACS power is also in its additional outdoors resources, allowing access to vector (IDRIS and Météo-France) and scalar (e.g., CEA CCRT, CINES, ECMWF, Marenostrum and Argonne) computing all over the world. With these advantages and considering the exponential growth of computer power, it was evident for the author that the potential for developing a Lagrangian formulation on a parallel, unstructured and hybrid solver like AVBP relies on an efficient implementation on massively parallel machines and in the use of load-balancing strategies. The progress in LES two-phase flow understanding will allow an improvement in design of more efficient machines (e.g., gas turbines). The (sometimes claimed) drawback of Lagrangian formulations being too expensive for practical applications, will probably be reduced over the next few years with the increasing performance of current parallel machines and the progress of computer science in search algorithms, load-balancing capabilities, ordering techniques, etc. In addition, having a Lagrangian formulation within a code with an Eulerian approach for the dispersed phase is an important element of validation of new Eulerian two-phase flow closure models as well as in specific applications where experimental data are not available. Neither Lagrangian, nor Eulerian two-phase flow simulations have the answer to the numerous questions that challenge scientists but the future may be somewhere in the middle and having them together is a clear advantage that opens a number of new directions to CERFACS.

1.6 Aim of the work and plan of the thesis

The main objective of all this thesis was, on one hand to show the feasibility of performing efficient Lagrangian simulations on parallel machines and on the other hand to prepare the computational environment for next-generation of petaflop supercomputers where faster computers and more efficient algorithms will be available.

The work developed throughout this thesis and presented here is novel in the sense that it represents the first efficient implementation of a Lagrangian approach within the parallel, unstructured and hybrid solver AVBP. Two major difficulties have been: (i) the capability to deal with a fully unstructured solver which requires a deep understanding of the internal structures of the code, and (ii) an efficient parallel implementation to maintain good performance specially for future simulations on massively parallel machines. Concerning this point, the use of load-balancing capabilities is highlighted as a promising and an efficient solution in two-phase flow simulations when strong imbalance of the disperse phase is present. Special attention has also been taken to keep the portability of the code to different machine architectures which is a key feature of AVBP. The Lagrangian module has been conceived to be modular enough to make future modifications of the current algorithms and models easy to implement.

The contents of this thesis are organised as follows:

- **Chapter 2**: provides a general overview of the main features of the AVBP solver. The chapter begins with the description of the governing equations of LES and the cell-vertex method used for the discretization of these governing equations. Then, a brief description of boundary conditions is also included. For the first time, a detailed description of the partitioning algorithms currently available in AVBP is presented. In addition, basic notions of the data structure and parallel strategies are also included. To conclude, a complete section is dedicated to the study of the growth of rounding errors and the reproducibility of LES.

- **Chapter 3**: is dedicated to the description of the Lagrangian module implementation. First, the equations of particle motion are presented. Then, the principal data structures developed in the Lagrangian module are described. The main characteristics of the particle-tracking scheme developed for unstructured grids have to deal with: localisation of particles in elements of arbitrary shape, particles searching in different situations (initially, after injection, during the simulation or while crossing boundaries between processors), description of the interpolation algorithms used to obtain gas-phase properties at particle position, two-way coupling and injection boundary conditions.

- **Chapter 4**: summarises the main results of the application of a Lagrangian two-phase flow simulation to a homogeneous isotropic turbulence (HIT) test case. It begins with the revision of some basic notions of turbulence and the main categories in which the HIT flows are classified. Then, the theoretical basis of these flows are presented. Results of the single and two-phase HIT simulations are included in the last sections.

- **Chapter 5**: before computing reactive two-phase flows in realistic combustion chambers, a validation of the turbulent dispersion of the particles in similar flows is needed. A more realistic configuration has been chosen to validate the numerical developments of the Lagrangian module. It consists of a vertical axisymmetric particle-laden confined bluff-body flow. The data were obtained in a flow loop of EDF-R&D, called Hercule and results are analysed in

Borée *et al.* [22]. This configuration is typical of an industrial application where the objective is to control the mixing of fuel with air. The chapter begins with the description of the configuration. Then, the computational set-up is also presented. Next sections compare results obtained with the Lagrangian module and an external Lagrangian solver for the single-phase simulation, the two-phase monodisperse and the polydisperse simulations. An analysis of the code performance is included at the end of the chapter.

- **Conclusions and perspectives**: is the last section of the dissertation before the bibliography and Appendices. It contains a summary of the main conclusions obtained during this thesis and a list of future developments and perspectives.
- **Bibliography**: a list of the references mentioned during this work in alphabetical order.
- **Appendices**: include the four publications related to this thesis: a proceedings of the Summer Program 2006, result of the month of work at the Stanford University; a paper presented at the 6th International Conference on Multiphase Flow, ICMF 2007, at Leipzig; an article published in the AIAA Journal in July 2008 and a last paper accepted for publication in the Journal of Computational Physics to appear (Vol. 228, N° 2, pp. 539-564) in 2009.

Although the test cases studied here are not complex two-phase-flow simulations in realistic geometries, they provide a fundamental study of the feasibility of Lagrangian simulations with AVBP. Many important aspects of the Lagrangian description of two-phase flows such as particle collisions, coalescence and particle-wall interactions, have not been addressed, but they will certainly insure interesting research topics in the years to come.

Chapter 2

The AVBP solver

« A Very Big Project »

Thilo Schönfeld and Michael Rudgyard
(a long time ago)

The AVBP project was historically motivated by the idea of building a modern software tool for Computational Fluid Dynamics (CFD) of high flexibility, efficiency and modularity. It was started at CERFACS in January 1993 as an initiative of Michael Rudgyard and Thilo Schönfeld. The aim was to create an unstructured solver capable of handling grids of any cell type. The use of these so-called hybrid grids is motivated by the efficiency of unstructured grid generation, the accuracy of the computational results (using regular structured elements) and the ease of mesh adaptation. The philosophy of building AVBP upon software libraries was adopted to best meet the modularity requirement.

AVBP is a parallel CFD code that solves the laminar and turbulent compressible Navier-Stokes equations in two and three space dimensions. Steady state or unsteady flows may be simulated. For stationary turbulent flows two different RANS type turbulence models were available (no longer supported after 2002 (V4.8)), while for the prediction of unsteady turbulence, various Large-Eddy Simulation (LES) subgrid scale models have been implemented. AVBP was initially conceived for primarily stationary external flows for aerodynamics applications. Since the mid-nineties the emphasis of applications is on the modeling of unsteady turbulent flows (with and without chemical reactions) for mainly internal flow configurations. These activities are partially related to the rising importance of the understanding of the flow structure and mechanisms leading to turbulence. The prediction of these unsteady turbulent flows is based on the LES approach which has emerged as a prospective technique for problems associated with time dependent phenomena and coherent eddy structures. An Arrhenius law reduced chemistry model allows investigating combustion for complex configurations.

The handling of unstructured or hybrid grids is one key feature of AVBP. With the use of these hybrid grids, where a combination of several elements of different type is used in the framework of the same mesh, the advantages of structured and unstructured grid methodologies are combined in terms of gridding flexibility and solution accuracy. In order to handle arbitrary hybrid grids, the

data structure of AVBP employs a cell-vertex finite-volume approximation. The basic numerical methods are based on a Lax-Wendroff [110, 111] or a Finite-Element type low-dissipation Taylor-Galerkin [43, 44, 171, 33] discretization in combination with a linear-preserving artificial viscosity model.

AVBP is built upon a modular software library that includes integrated parallel domain partition and data reordering tools, handles message passing and includes supporting routines for dynamic memory allocation, routines for parallel I/O and iterative methods. AVBP is written in standard Fortran 77 and C but it is being upgraded to Fortran 90 in a gradual fashion. One of its main features is its portability to different machine architectures and it has proven to be efficient on most parallel architectures.

AVBP is currently developed by more than 30 PhD students and Post-Docs together with research scientists and engineers. Today, the ownership of AVBP is shared between CERFACS and Institut Français du Pétrole (IFP), located in the Paris area, following an agreement of joint code development oriented towards gas turbines and piston engine applications. It is used in the framework of many bilateral industrial collaborations and national research programs. At a European level it is used in several projects of the 5th, 6th and 7th Framework Programs of the European Community (EC) and several research fellows use it in the frame of the Marie Curie actions. Important links to industry have also been established with Safran Group (Snecma, Turbomeca), Air Liquide, Gaz de France as well as with Alstom and Siemens Power Generation.

This chapter gives a general overview of the main features of the AVBP solver. Section 2.1 describes the governing equations used in LES to resolve the set of compressible Navier-Stokes equations and the models available for the subgrid stress tensor. The cell-vertex method used for the discretization of the governing equations is presented in Section 2.2. Then, Section 2.3 includes a brief description of boundary conditions. The partitioning strategy is described in detail in Section 2.4 together with the main reasons for the introduction of a new partitioning algorithm. Sections 2.5 and 2.6 contain some basic notions of data structure and parallel strategies used in AVBP. Finally, Section 2.7 is dedicated to the study of the growth of rounding errors and the reproducibility of large-eddy simulations. Some conclusions are listed at the end of the chapter (Section 2.8).

2.1 Governing equations for LES

The principle of LES has been presented in Chapter 1. The application cases presented in this thesis will not be reactive and therefore this section describes only the filtered equations solved by AVBP for a turbulent non-reacting flow. With this intention, the filtering procedure is presented in Subsection 2.1.1. Subsection 2.1.2 describes the equations solved for LES of non-reacting flows. Then, the different terms of the flux tensor are presented in Subsections 2.1.3–2.1.5. Finally, different models of the subgrid stress tensor available in AVBP are described in Subsection 2.1.6.

2.1.1 Filtering procedure

To separate large and small scales, a low-pass (in wavenumber) filter, G_Δ, is applied to the equations of motion [115]. Mathematically, it consists of a convolution of any quantity, f, with the filter

function G_Δ:

$$\overline{f}(x) = \int f(x')\ G_\Delta(x - x')\ dx'. \tag{2.1}$$

The resulting filtered quantity, $\overline{f}$, represents the large-scale structures of the flow whereas all the structures of size smaller than the filter length, Δ, are contained in the residual field, f':

$$f' = f - \overline{f}. \tag{2.2}$$

To apply this filtering procedure to the instantaneous balance equations [165], the filter G_Δ (typical a box or a Gaussian filter [186]) must satisfy some properties which are: conservation of constants, linearity and commutation with temporal and spatial derivatives. The latter is satisfied only for homogeneous filters (i.e., grid meshes). For the sake of simplicity, this property is assumed hereafter.

For variable density ρ, a density-weighted filter $\widetilde{f}$ (Favre [56] averaging) is used, in order to avoid modeling of additional terms introduced by density fluctuations:

$$\widetilde{f} = \frac{\overline{\rho f}}{\overline{\rho}}. \tag{2.3}$$

2.1.2 Filtered Navier-Stokes equations for non-reacting flows

The balance equations (mass, momentum, energy and species) for large-eddy simulations are obtained by filtering the instantaneous balance equations [165]:

$$\frac{\partial \overline{\rho}}{\partial t} + \frac{\partial}{\partial x_i}(\overline{\rho}\widetilde{u}_i) = 0 \tag{2.4}$$

$$\frac{\partial \overline{\rho}\widetilde{u}_i}{\partial t} + \frac{\partial}{\partial x_j}(\overline{\rho}\widetilde{u}_i\widetilde{u}_j) = -\frac{\partial}{\partial x_j}[\overline{P}\delta_{ij} - \overline{\tau}_{ij} - \overline{\tau}^t_{ij}] \tag{2.5}$$

$$\frac{\partial \overline{\rho}\widetilde{E}}{\partial t} + \frac{\partial}{\partial x_j}(\overline{\rho}\widetilde{E}\widetilde{u}_j) = -\frac{\partial}{\partial x_j}[\overline{u_i(P\delta_{ij} - \tau_{ij})} + \overline{q}_j + \overline{q}^t_j] + \overline{\dot{\omega}}_T + \overline{Q}_r \tag{2.6}$$

$$\frac{\partial \overline{\rho}\widetilde{Y}_k}{\partial t} + \frac{\partial}{\partial x_j}(\overline{\rho}\widetilde{Y}_k\widetilde{u}_j) = -\frac{\partial}{\partial x_j}[\overline{J}_{j,k} + \overline{J}^t_{j,k}] + \overline{\dot{\omega}}_k, \tag{2.7}$$

where $\widetilde{u}_i$, $\widetilde{E}$ and $\widetilde{Y}_k$ denote the filtered velocity vector, total energy per unit mass and species mass fractions, respectively. A repeated index implies summation over this index (Einstein's rule of summation). Note also that the index k is reserved for referring to the kth species and does not follow the summation rule (unless specifically mentioned).

Writing the vector of the filtered conservative variables as follows: $\overline{\mathbf{w}} = (\bar{\rho}\tilde{u}, \bar{\rho}\tilde{v}, \bar{\rho}\tilde{w}, \bar{\rho}\tilde{E}, \bar{\rho}\tilde{Y}_k)$, Eqs. (2.5)-(2.7), can be expressed as:

$$\frac{\partial \overline{\mathbf{w}}}{\partial t} + \nabla \cdot \overline{\mathbf{F}} = \overline{\mathbf{s}}, \tag{2.8}$$

where $\overline{\mathbf{s}}$ is the filtered source term and $\overline{\mathbf{F}}$ is the flux tensor which can be divided in three parts:

$$\overline{\mathbf{F}} = \overline{\mathbf{F}}^I + \overline{\mathbf{F}}^V + \overline{\mathbf{F}}^t, \tag{2.9}$$

with

$$\text{Inviscid terms:} \quad \overline{\mathbf{F}}^I = \left(\overline{\mathbf{f}}^I, \overline{\mathbf{g}}^I, \overline{\mathbf{h}}^I\right)^T \tag{2.10}$$

$$\text{Viscous terms:} \quad \overline{\mathbf{F}}^V = \left(\overline{\mathbf{f}}^V, \overline{\mathbf{g}}^V, \overline{\mathbf{h}}^V\right)^T \tag{2.11}$$

$$\text{Turbulent subgrid-scale terms:} \quad \overline{\mathbf{F}}^t = \left(\overline{\mathbf{f}}^t, \overline{\mathbf{g}}^t, \overline{\mathbf{h}}^t\right)^T. \tag{2.12}$$

The cut-off scale corresponds to the mesh size (implicit filtering). As usually done, we assume that the filter operator and the partial derivative commute.

2.1.3 Inviscid terms

The three spatial components of the inviscid flux tensor based on the filtered quantities are:

$$\overline{\mathbf{f}}^I = \begin{pmatrix} \overline{\rho}\widetilde{u}^2 + \overline{P} \\ \overline{\rho}\widetilde{u}\widetilde{v} \\ \overline{\rho}\widetilde{u}\widetilde{w} \\ \overline{\rho}\widetilde{E}\widetilde{u} + \overline{P\ u} \\ \overline{\rho}_k\widetilde{u} \end{pmatrix}, \ \overline{\mathbf{g}}^I = \begin{pmatrix} \overline{\rho}\widetilde{u}\widetilde{v} \\ \overline{\rho}\widetilde{v}^2 + \overline{P} \\ \overline{\rho}\widetilde{v}\widetilde{w} \\ \overline{\rho}\widetilde{E}\widetilde{v} + \overline{P\ v} \\ \overline{\rho}_k\widetilde{v} \end{pmatrix}, \ \overline{\mathbf{h}}^I = \begin{pmatrix} \overline{\rho}\widetilde{u}\widetilde{w} \\ \overline{\rho}\widetilde{v}\widetilde{w} \\ \overline{\rho}\widetilde{w}^2 + \overline{P} \\ \overline{\rho}\widetilde{E}\widetilde{w} + \overline{P\ w} \\ \overline{\rho}_k\widetilde{w} \end{pmatrix}. \tag{2.13}$$

2.1.4 Filtered viscous terms

The components of the viscous flux tensor take the form:

$$\overline{\mathbf{F}}^V = \begin{pmatrix} -\overline{\tau_{xx}} \\ -\overline{\tau_{xy}} \\ -\overline{\tau_{xz}} \\ -(\overline{u\ \tau_{xx}} + \overline{v\ \tau_{xy}} + \overline{w\ \tau_{xz}}) + \overline{q_x} \\ \overline{J_{x,k}} \end{pmatrix}, \tag{2.14}$$

$$\overline{\mathbf{G}}^V = \begin{pmatrix} -\overline{\tau_{xy}} \\ -\overline{\tau_{yy}} \\ -\overline{\tau_{yz}} \\ -(\overline{u\ \tau_{xy}} + \overline{v\ \tau_{yy}} + \overline{w\ \tau_{yz}}) + \overline{q_y} \\ \overline{J_{y,k}} \end{pmatrix}, \tag{2.15}$$

$$\overline{\mathbf{H}}^V = \begin{pmatrix} -\overline{\tau_{xz}} \\ -\overline{\tau_{yz}} \\ -\overline{\tau_{zz}} \\ -(\overline{u\ \tau_{xz}} + \overline{v\ \tau_{yz}} + \overline{w\ \tau_{zz}}) + \overline{q_z} \\ \overline{J_{z,k}} \end{pmatrix}. \tag{2.16}$$

Filtering the balance equations leads to unclosed quantities which need to be modeled. The filtered diffusion terms are (see Poinsot & Veynante [165], Chapter 4):

- Laminar filtered stress tensor $\widetilde{\tau_{ij}}$

$$\overline{\tau_{ij}} = \overline{2\mu(S_{ij} - \frac{1}{3}\delta_{ij}S_{ll})} \tag{2.17}$$

$$\text{approximation:} \quad \overline{\tau_{ij}} \approx 2\overline{\mu}(\widetilde{S}_{ij} - \frac{1}{3}\delta_{ij}\widetilde{S}_{ll}) \tag{2.18}$$

$$\text{with:} \quad \widetilde{S}_{ij} = \frac{1}{2}\left(\frac{\partial \widetilde{u}_j}{\partial x_i} + \frac{\partial \widetilde{u}_i}{\partial x_j}\right) \tag{2.19}$$

$$\overline{\mu} \approx \mu(\widetilde{T}). \tag{2.20}$$

Eq. (2.18) may also be written as:

$$\begin{aligned} \overline{\tau_{xx}} &\approx \tfrac{2\overline{\mu}}{3}(2\tfrac{\partial \widetilde{u}}{\partial x} - \tfrac{\partial \widetilde{v}}{\partial y} - \tfrac{\partial \widetilde{w}}{\partial z}), \quad \overline{\tau_{xy}} \approx \overline{\mu}(\tfrac{\partial \widetilde{u}}{\partial y} + \tfrac{\partial \widetilde{v}}{\partial x}) \\ \overline{\tau_{yy}} &\approx \tfrac{2\overline{\mu}}{3}(2\tfrac{\partial \widetilde{v}}{\partial y} - \tfrac{\partial \widetilde{u}}{\partial x} - \tfrac{\partial \widetilde{w}}{\partial z}), \quad \overline{\tau_{xz}} \approx \overline{\mu}(\tfrac{\partial \widetilde{u}}{\partial z} + \tfrac{\partial \widetilde{w}}{\partial x}) \\ \overline{\tau_{zz}} &\approx \tfrac{2\overline{\mu}}{3}(2\tfrac{\partial \widetilde{w}}{\partial z} - \tfrac{\partial \widetilde{u}}{\partial x} - \tfrac{\partial \widetilde{v}}{\partial y}), \quad \overline{\tau_{yz}} \approx \overline{\mu}(\tfrac{\partial \widetilde{v}}{\partial z} + \tfrac{\partial \widetilde{w}}{\partial y}). \end{aligned} \tag{2.21}$$

- Diffusive species flux vector $\overline{J_{i,k}}$

For non-reacting flows:

$$\overline{J_{i,k}} = -\overline{\rho\left(D_k \frac{W_k}{\overline{W}} \frac{\partial X_k}{\partial x_i} - Y_k V_i^c\right)} \tag{2.22}$$

$$\text{approximation:} \quad \overline{J_{i,k}} \approx -\overline{\rho}\left(\overline{D}_k \frac{W_k}{\overline{W}} \frac{\partial \widetilde{X}_k}{\partial x_i} - \widetilde{Y_k}\widetilde{V}_i^c\right) \tag{2.23}$$

$$\text{with:} \quad \widetilde{V}_i^c = \sum_{k=1}^{N} \overline{D}_k \frac{W_k}{\overline{W}} \frac{\partial \widetilde{X}_k}{\partial x_i} \tag{2.24}$$

$$\overline{D}_k \approx \frac{\overline{\mu}}{\overline{\rho} Sc_k}. \tag{2.25}$$

- Filtered heat flux $\overline{q_i}$

$$\overline{q_i} = -\lambda \overline{\frac{\partial T}{\partial x_i}} + \sum_{k=1}^{N} \overline{J_{i,k} h_{s,k}} \tag{2.26}$$

$$\text{approximation:} \quad \overline{q_i} \approx -\overline{\lambda} \frac{\partial \widetilde{T}}{\partial x_i} + \sum_{k=1}^{N} \overline{J_{i,k}}\, \widetilde{h}_{s,k} \tag{2.27}$$

$$\text{with:} \quad \overline{\lambda} \approx \frac{\overline{\mu}\, \overline{C}_p(\widetilde{T})}{Pr}. \tag{2.28}$$

These forms assume that the spatial variations of molecular diffusion fluxes are negligible and can be modeled through simple gradient assumptions.

2.1.5 Subgrid-scale turbulent terms

The three components of the turbulent subgrid-scale flux take the form:

$$\overline{\mathbf{F}}^t = \begin{pmatrix} -\overline{\tau_{xx}}^t \\ -\overline{\tau_{xy}}^t \\ -\overline{\tau_{xz}}^t \\ \overline{q_x}^t \\ \overline{J_{x,k}}^t \end{pmatrix}, \ \overline{\mathbf{G}}^t = \begin{pmatrix} -\overline{\tau_{xy}}^t \\ -\overline{\tau_{yy}}^t \\ -\overline{\tau_{yz}}^t \\ \overline{q_y}^t \\ \overline{J_{y,k}}^t \end{pmatrix}, \ \overline{\mathbf{H}}^t = \begin{pmatrix} -\overline{\tau_{xz}}^t \\ -\overline{\tau_{yz}}^t \\ -\overline{\tau_{zz}}^t \\ \overline{q_z}^t \\ \overline{J_{z,k}}^t \end{pmatrix}. \tag{2.29}$$

As highlighted above, filtering the transport equations leads to a closure problem evidenced by the so called "subgrid-scale" (SGS) turbulent fluxes. For the system to be solved numerically, closures need to be supplied. Details on the closures are:

- The Reynolds tensor $\overline{\tau_{ij}}^t$

$$\overline{\tau_{ij}}^t = -\overline{\rho}(\widetilde{u_i u_j} - \widetilde{u}_i \widetilde{u}_j) \tag{2.30}$$

$$\text{modeled as:} \quad \overline{\tau_{ij}}^t = 2\,\overline{\rho}\,\nu_t(\widetilde{S}_{ij} - \frac{1}{3}\delta_{ij}\widetilde{S}_{ll}) \tag{2.31}$$

$$\text{with:} \quad \widetilde{S}_{ij} = \frac{1}{2}\left(\frac{\partial \widetilde{u}_i}{\partial x_j} + \frac{\partial \widetilde{u}_j}{\partial x_i}\right) - \frac{1}{3}\frac{\partial \widetilde{u}_k}{\partial x_k}\,\delta_{ij}. \tag{2.32}$$

In Eq. (2.31), $\overline{\tau_{ij}}^t$ is the SGS tensor, ν_t is the SGS turbulent viscosity, and $\widetilde{S}_{ij}$ is the resolved strain rate tensor. The modeling of ν_t is explained in section 2.1.6.

- The subgrid scale diffusive species flux vector $\overline{J_{i,k}}^t$

$$\overline{J_{i,k}}^t = \overline{\rho}\,(\widetilde{u_i Y_k} - \widetilde{u}_i \widetilde{Y}_k) \tag{2.33}$$

$$\text{modeled as:} \quad \overline{J_{i,k}}^t = -\overline{\rho}\left(D_k^t \frac{W_k}{\overline{W}}\frac{\partial \widetilde{X}_k}{\partial x_i} - \widetilde{Y_k}\widetilde{V}_i^{c,t}\right) \tag{2.34}$$

$$\text{with:} \quad \widetilde{V}_i^{c,t} = \sum_{k=1}^{N} D_k^t \frac{W_k}{\overline{W}}\frac{\partial \widetilde{X}_k}{\partial x_i} \tag{2.35}$$

$$D_k^t = \frac{\nu_t}{Sc_k^t}. \tag{2.36}$$

The turbulent Schmidt number $Sc_k^t = 1$ is the same for all species and is fixed in the source code (like Pr^t). Note also that having one turbulent Schmidt number for all the species does not imply, $\widetilde{V}_i^{c,t} = 0$ because of the $W_k/\overline{W}$ term in Eq. (2.34).

- The subgrid scale heat flux vector $\overline{q_i}^t$

$$\overline{q_i}^t = \overline{\rho}(\widetilde{u_i E} - \widetilde{u}_i \widetilde{E}) \tag{2.37}$$

$$\text{modeled as:} \quad \overline{q_i}^t = -\lambda_t \frac{\partial \widetilde{T}}{\partial x_i} + \sum_{k=1}^{N} \overline{J_{i,k}}^t\, \widetilde{h}_{s,k} \tag{2.38}$$

$$\text{with:} \quad \lambda_t = \frac{\mu_t \overline{C_p}}{Pr^t}. \tag{2.39}$$

2.1.6 Models for the subgrid stress tensor

LES models for the subgrid stress tensor (see Eq. (2.31)) are derived on the theoretical ground that the LES filter is spatially and temporally invariant. Variations in the filter size due to non-uniform meshes or moving meshes are not directly accounted for in the LES models. Change of cell topology is only accounted for through the use of the local cell volume, that is $\triangle = V_{cell}^{1/3}$.

The filtered compressible Navier-Stokes equations exhibit SGS tensors and vectors describing the interaction between the non-resolved and resolved motions. The influence of the SGS on the resolved motion is taken into account in AVBP by a SGS model based on the introduction of a turbulent viscosity, ν_t (Boussinesq [25] model). Such an approach assumes the effect of the SGS field on the resolved field to be purely dissipative. This hypothesis is essentially valid within the cascade theory of turbulence introduced by Kolmogorov [103].

The several LES models for the subgrid stress tensor which are currently available in AVBP only differ through the estimation of ν_t, whose expressions are given below:

Smagorinsky model

$$\nu_t = (C_S \triangle)^2 \sqrt{2\, \widetilde{S}_{ij}\, \widetilde{S}_{ij}}, \tag{2.40}$$

where $\triangle$ denotes the filter characteristic length (cube-root of the cell volume), C_S is the model constant set to 0.18 but can vary between 0.1 and 0.18 depending on the flow configuration. The Smagorinsky model [204] was developed in the sixties and heavily tested for multiple flow configurations. This closure has the particularity of supplying the right amount of dissipation of kinetic energy in homogeneous isotropic turbulent flows. Locality is however lost and only global quantities are maintained. It is known to be "too dissipative" and transitioning flows are not suited for its use [187].

Filtered Smagorinsky model

$$\nu_t = (C_{S_F} \triangle)^2 \sqrt{2\, HP(\widetilde{S}_{ij})\, HP(\widetilde{S}_{ij})}, \tag{2.41}$$

where $\triangle$ denotes the filter characteristic length (cube-root of the cell volume), $C_{S_F} = 0.37$ is the model constant and $HP(\widetilde{S}_{ij})$ denotes the resolved strain rate tensor obtained from a high-pass filtered velocity field. This model was developed in order to allow a better representation of local phenomena typical of complex turbulent flows Ducros *et al.* [46]. With the Filtered Smagorinsky model transition is better predicted and locality is in general better preserved.

Dynamic Smagorinsky model

$$\nu_t = (C_{S_D} \triangle)^2 \sqrt{2\, \widetilde{S}_{ij}\, \widetilde{S}_{ij}}, \tag{2.42}$$

where $\triangle$ denotes the filter characteristic length (cube-root of the cell volume). The difference with the standard expression obtained for the conventional Smagorinsky model (Eq. (2.40)) comes from

the evaluation of the closure coefficient C_{S_D}. In the Dynamic Smagorinsky approach proposed by Germano *et al.* [72], the coefficient is obtained within the simulation and is no more a user defined variable. The expression from which C_{S_D} is obtained stems from the Germano inequality and follows Lilly's procedure [118]:

$$C_{S_D}{}^2 = \frac{1}{2}\frac{M_{ij}M_{ij}}{L_{ij}L_{ij}}. \tag{2.43}$$

In the previous expression, the following tensors are defined by,

$$M_{ij} = \hat{\Delta}^2\sqrt{2 < \widetilde{S}_{ij} > < \widetilde{S}_{ij} >} < \widetilde{S}_{ij} > \qquad L_{ij} =< \widetilde{u}_i >< \widetilde{u}_j > - < \widetilde{u}_i\widetilde{u}_j > \tag{2.44}$$

and introduce the notion of "test" filter of characteristic length $\hat{\Delta}$ equal to the cubic root of the volume defined by all the cells surrounding the cell of interest. Note that clipping and smoothing ensures none negative values for C_{S_D}.

Wall-adapting local eddy-viscosity (WALE) model

$$\nu_t = (C_w\triangle)^2\frac{(s^d_{ij}s^d_{ij})^{3/2}}{(\widetilde{S}_{ij}\widetilde{S}_{ij})^{5/2}+(s^d_{ij}s^d_{ij})^{5/4}} \tag{2.45}$$

$$\text{with:} \qquad s^d_{ij} = \frac{1}{2}\,(\widetilde{g}^2_{ij} + \widetilde{g}^2_{ji}) - \frac{1}{3}\,\widetilde{g}^2_{kk}\,\delta_{ij}, \tag{2.46}$$

where $\triangle$ denotes the filter characteristic length (cube-root of the cell volume), $C_w = 0.4929$ is the model constant and $\widetilde{g}_{ij}$ denotes the resolved velocity gradient. The WALE model Ducros *et al.* [47] was developed for wall bounded flows in an attempt to recover the scaling laws of the wall. Similarly to the Smagorinsky model locality is lost and only global quantities are to be trusted.

***k*-equation model**

$$\nu_t = C_k\triangle k^{1/2}_{sgs}. \tag{2.47}$$

where C_k is a model constant, $\triangle$ denotes the filter characteristic length (cube-root of the cell volume) and k_{sgs} is the subgrid-scale turbulent kinetic energy defined as:

$$k_{sgs} = \frac{1}{2}(\widetilde{u_i u_i} - \widetilde{u}_i\widetilde{u}_i). \tag{2.48}$$

The filtered transport equation for k_{sgs} writes:

$$\frac{\partial\overline{\rho}k_{sgs}}{\partial t} + \frac{\partial\overline{\rho}k_{sgs}\widetilde{u}_j}{\partial x_j} = \frac{\partial}{\partial x_j}\left(\mu_t\frac{\partial k_{sgs}}{\partial x_j}\right) + \left(2\mu_t\widetilde{S}_{ij} - \frac{2}{3}\overline{\rho}k_{sgs}\delta_{ij}\right)\frac{\partial\widetilde{u}_i}{\partial x_j} - \overline{\rho}C_\epsilon\frac{k^{3/2}_{sgs}}{\Delta}, \tag{2.49}$$

where the terms on the left-hand side represent the turbulent diffusion of k_{sgs}, its production and its dissipation.

The model constants C_k and C_ϵ (see e.g., Moureau [137] for the derivation) can be shown to depend as follows from the standard Smagorinsky constant C_S (see Eq. (2.40)) and $C_K = 1.4$ is the Kolmogorov constant:

$$C_k = \pi^{\frac{1}{3}} \left(\frac{2}{3\,C_K} \right)^{\frac{1}{2}} (C_S)^{\frac{4}{3}} \quad , \quad C_\epsilon = \pi \left(\frac{2}{3\,C_K} \right)^{\frac{3}{2}} \tag{2.50}$$

yielding the following values for the standard values of $C_S = 0.18$ for a HIT without shear and $C_S = 0.1$ for a pure shear layer:

C_S	C_ϵ	C_k
0.18	1.03	0.10
0.10	1.03	0.05

The advantage of the k-equation model is that it does not directly relate the level of turbulent viscosity to the local instantaneous resolved velocity gradients, as e.g., the Smagorinsky or WALE models. Turbulent viscosity is rather deduced from the transported subgrid-scale turbulent kinetic energy k_{sgs}. Thus, it can account for cases where local levels of subgrid-scale energy are high in regions of low resolved gradients, because the kinetic energy has been convected from regions with high subgrid turbulence.

2.2 Numerical method

2.2.1 The cell-vertex discretization

The flow solver used for the discretization of the governing equations is based on the finite volume (FV) method [89]. There are three common techniques for implementing FV methods: the so-called *cell-centred*, *vertex-centred* and *cell-vertex* approaches. In the first two ones, the discrete values of the solution are stored at the centre of the control volume (grid cells for the *cell-centred* formulation and median dual cells for the *vertex-centred* one, see Fig. 2.1) and neighbouring values are averaged across the control volume boundaries in order to calculate the fluxes.

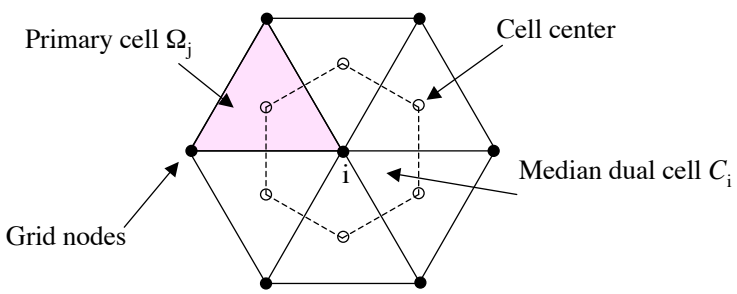


Figure 2.1 - *Cell-vertex cells.*

In the alternative cell-vertex technique, used as underlying numerical discretization method of AVBP (Rudgyard [183, 184]), the discrete values of the conserved variables are stored at the *cell*

vertices (or grid nodes), while conservation relations are applied to the grid (or primary) cells. The advantages of using such a discretization are:

- The native capability of handling unstructured hybrid meshes.
- An easy and efficient parallelization.
- Increased accuracy without an important additional cost, can be obtained by using the same spatial differential operators in a finite element framework (see Section 2.2.3).

In the cell-vertex method employed within AVBP both solution and coordinate vectors are stored at the nodes of the grid. However, most of the operations are done on the elements and often a transfer from the cell vertices (the nodes) to the cell centers is required. This collecting of the nodal information to temporary arrays that contain the information of the vertices for an element is done in a so-called data *gather* operation (Fig. 2.2 (a)). At this stage each cell has locally its information available at the vertices and for example can calculate the cell gradient. The cell quantity is then distributed back to the global nodes through an inverse so-called *scatter* operation (Fig. 2.2 (b)).

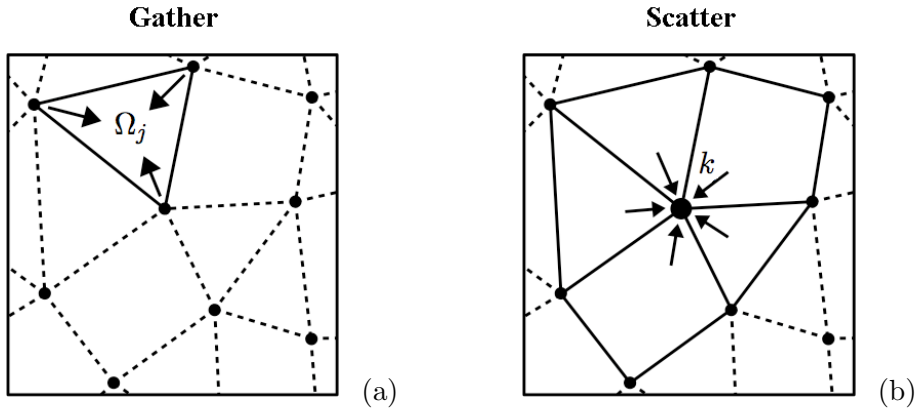


Figure 2.2 - *Cell-vertex principle: (a) gather and (b) scatter operations.*

Nomenclature In the rest of the section the following subscripts are used:

- $i \in [1, N_{node}]$ is the index used for the global node numbering and the nodal values.
- $j \in [1, N_{cell}]$ is used for the cell numbering.
- $k \in [1, n_v(\Omega_j)]$ is the local numbering of the vertices of a cell Ω_j, with $n_v(\Omega_j)$ the number of vertices of the cell Ω_j.
- Ω_j is the index used to design a value at the centre or associated with the j-th cell.
- R_i is the global nodal residual.
- R_{Ω_j} is the global cell residual.
- $R_{i|\Omega_j}$ is the part of the residual of element j to be scattered to node i.

Weighted Cell Residual Approach

For the description of the weighted cell-residual approach the laminar Navier-Stokes equations are considered in their conservative formulation:

$$\frac{\partial \mathbf{w}}{\partial t} + \nabla \cdot \mathbf{F} = \mathbf{0}, \tag{2.51}$$

where $\mathbf{w}$ is the vector of conserved variables and $\mathbf{F}$ is the corresponding flux tensor. For convenience, the latter is divided into an inviscid and a viscous part, $\mathbf{F} = \mathbf{F}^I(\mathbf{w}) + \mathbf{F}^V(\mathbf{w}, \vec{\nabla}\mathbf{w})$. The spatial terms of the equations are then approximated in each control volume Ω_j to give the *residual*

$$\mathbf{R}_{\Omega_j} = \frac{1}{V_{\Omega_j}} \int_{\partial\Omega_j} \mathbf{F} \cdot \vec{n} \; d\mathcal{S}, \tag{2.52}$$

where $\partial\Omega_j$ denotes the boundary of Ω_j with normal $\vec{n}$.

This cell-vertex approximation is readily applicable to arbitrary cell types and is hence straightforward to apply for hybrid grids. The residual (Eq. (2.52)) is first computed for each *element* by making use of a simple integration rule applied to the *faces*. For triangular faces, a straightforward mid-point rule is used, which is equivalent to the assumption that the individual components of the flux vary linearly on these faces. For quadrilateral faces, where the nodes may not be co-planar, in order to ensure that the integration is exact for arbitrary elements if the flux functions do indeed vary linearly, each face is divided into triangles and then integrated over the individual triangles (Fig. 2.3). The flux value is then obtained from the average of four triangles (two divisions along the two diagonals). This so-called 'linear preservation property' plays an important part in the algorithm for ensuring that accuracy is not lost on irregular meshes. Computationally, it is useful

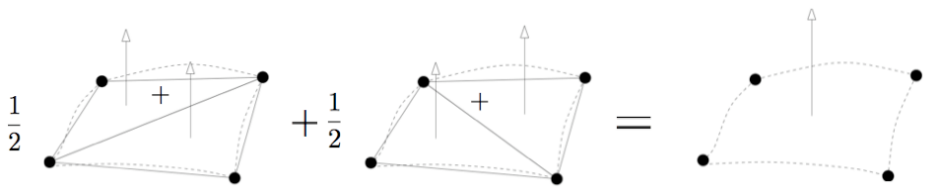


Figure 2.3 - *Calculation of the normal for a quadrilateral face.*

to write the discrete integration of Eq. (2.52) over an arbitrary cell as

$$\mathbf{R}_{\Omega_j} = \frac{1}{N_d V_{\Omega_j}} \sum_{i \in \Omega_j} \mathbf{F}_i \cdot d\vec{S}_i, \tag{2.53}$$

where $\mathbf{F}_i$ is an approximation of $\mathbf{F}$ at the nodes, N_d represents the number of space dimensions and $\{i \in \Omega_j\}$ are the vertices of the cell. In this formulation the geometrical information has been factored into terms $d\vec{S}_i$ that are associated with individual nodes of the cell but not faces; $d\vec{S}_i$ is merely the average of the area-weighted normals for triangulated faces with a common node i, $i \in \Omega_j$ (Fig. 2.4). Note, that for consistency one has $\sum_{i \in \Omega_j} d\vec{S}_i = \vec{0}$. A linear preserving approximation of the divergence operator is obtained if the volume V_{Ω_j} is defined consistently as

$$V_{\Omega_j} = \frac{1}{N_d^2} \sum_{i \in \Omega_j} \vec{x}_i \cdot d\vec{S}_i \tag{2.54}$$

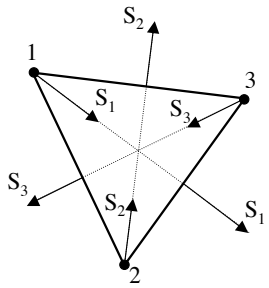


Figure 2.4 - *Node normals for triangular elements. The normal associated with node 1 is given by:* $S_1 = -(S_2 + S_3)$.

since $\nabla \cdot \vec{x} = N_d$. Once the cell residuals are calculated, one may then define the semi-discrete scheme

$$\frac{d\mathbf{w}_k}{dt} = -\frac{1}{V_k} \sum_{j|k\in\Omega_j} D^k_{\Omega_j} V_{\Omega_j} R_{\Omega_j}, \tag{2.55}$$

where $D^k_{\Omega_j}$ is a distribution matrix that weights the cell residual from cell centre Ω_j to node k (scatter operation), and V_k is a control volume associated with each node. Conservation is guaranteed if $\sum_{k\in\Omega_j} D^k_{\Omega_j} = I$. In the present context, Eq. (2.55) is solved to obtain the steady-state solution using explicit Euler or Runge-Kutta time-stepping.

The family of schemes of interest makes use of the following definition of the distribution matrix:

$$D^k_{\Omega_j} = \frac{1}{n_n}(I + C\frac{\delta t_{\Omega_j}}{V_{\Omega_j}} \mathcal{A}_{\Omega_j} \cdot \vec{dS}_k), \tag{2.56}$$

where n_n is the number of nodes of Ω_j and $\mathcal{A}$ is the Jacobian of the flux tensor. The simplest 'central difference' scheme is obtained by choosing $C = 0$ and is neutrally stable when combined with Runge-Kutta time-stepping. A Lax-Wendroff type scheme may also be formulated in which case C is chosen to be a constant that depends on the number of space dimensions and the type of cells used — it may be shown that this takes the simple form $C = n_v^2/2N_d$. If one replaces the cell 'time-step' δt_{Ω_j} by a matrix Φ_{Ω_j} with suitable properties, one may also obtain an SUPG-like scheme (for Streamwise Upwind Petrov-Galerkin) [26] which has slightly better convergence and shock-capturing behaviour, however, at some extra computational cost.

Computation of gradients

In order to recover the nodal values of the gradients $\vec{\nabla}\mathbf{w}$ a cell approximation $\left(\vec{\nabla}\mathbf{w}\right)_{\Omega_j}$ is first calculated and then distributed to the nodes. The cell-based gradient is defined in a manner similar to the divergence (Eq. (2.53)) so as to be transparent to linear solution variations:

$$\left(\frac{\partial \mathbf{w}}{\partial x}\right)_C \approx \frac{1}{V_C} \int\int_{\partial\Omega_C} \mathbf{w} \cdot \vec{n} \partial S, \tag{2.57}$$

which leads to the approximation:

$$\left(\vec{\nabla}\mathbf{w}\right)_{\Omega_j} = \frac{1}{V_{\Omega_j}} \sum_{i\in\Omega_j} \overline{\mathbf{w}}_i \overline{\vec{dS}}_i. \tag{2.58}$$

A nodal approximation of the gradient is then obtained using of a volume-weighted average of the cell-based gradients:

$$\left(\vec{\nabla}\mathbf{w}\right)_k = \frac{1}{V_k} \sum_{j|k\in\Omega_j} V_{\Omega_j} (\vec{\nabla}\mathbf{w})_{\Omega_j}. \tag{2.59}$$

Computation of time step

Temporal discretization is explicit for all numerical schemes in AVBP. The practical implementation of this kind of approach is relatively straightforward and the computational cost per iteration is small. The main drawback of explicit codes is that the time step Δt is limited for stability reasons:

$$\Delta t < CFL \frac{min(\Delta x)}{max|\mathbf{u}| + c}, \tag{2.60}$$

where $\mathbf{u}$ is the propagation speed of a perturbation in the flow, c is the sound speed, Δx is the mesh size and CFL is the Courant-Friedrichs-Lewy number. The CFL value required for stability changes slightly depending on the scheme adopted. In AVBP, the CFL value is fixed to 0.7.

2.2.2 The Lax-Wendroff scheme

The form of the distribution matrix $D_{i|\Omega_j}$ (see Eq. (2.56)) determines the different numerical schemes available in AVBP. In the following $D_{i|\Omega_j}$ is derived for the Lax-Wendroff scheme [110, 111]. This scheme (second-order accurate in space and time) is based on a Taylor expansion in time of the solution $\mathbf{w}$.

$$\mathbf{w}^{n+1} = \mathbf{w}^n + \Delta t \left(\frac{\partial \mathbf{w}}{\partial t}\right)^n + \frac{1}{2}\Delta t^2 \left(\frac{\partial^2 \mathbf{w}}{\partial t^2}\right)^n + \mathcal{O}(\Delta t^3), \tag{2.61}$$

Considering Eq. (2.51), the first temporal derivative can be expressed as:

$$\frac{\partial \mathbf{w}}{\partial t} = -\nabla \cdot \mathbf{F}. \tag{2.62}$$

In a similar manner, the second derivative can be recast as:

$$\frac{\partial^2 \mathbf{w}}{\partial t^2} = \frac{\partial}{\partial t}\left(-\nabla \cdot \mathbf{F}\right) = -\nabla \cdot \frac{\partial \mathbf{F}}{\partial t} = -\nabla \cdot \left[\mathcal{A}\left(\frac{\partial \mathbf{w}}{\partial t}\right)\right] = \nabla \cdot \left[\mathcal{A}\left(\nabla \cdot \mathbf{F}\right)\right] \tag{2.63}$$

assuming that temporal and spatial derivatives can be exchanged and defining $\mathcal{A} = \frac{\partial \mathbf{F}}{\partial \mathbf{w}}$ as the Jacobian matrix. Hence, substituting Eq. (2.62) and (2.63) into Eq. (2.61) the solution at time $n+1$ can be written as:

$$\mathbf{w}^{n+1} = \mathbf{w}^n - \Delta t \left\{\nabla \cdot \mathbf{F} - \frac{1}{2}\Delta t \, \nabla \cdot \left[\mathcal{A}\left(\nabla \cdot \mathbf{F}\right)\right] - \mathcal{O}(\Delta t^2)\right\}. \tag{2.64}$$

In discrete form, remembering the basic principle of the cell-vertex approach, the nodal residual R_i is obtained by summing the contributions of all the surrounding elements. The value is then scaled by the nodal volume V_i:

$$R_i = \frac{1}{V_i} \sum_{j|i\in\Omega_j} R_{i|\Omega_j}. \tag{2.65}$$

The residual contribution to node i of element j can be written as:

$$R_{i|\Omega_j} = R_{\Omega_j} \frac{V_{\Omega_j}}{n_{v(\Omega_j)}} - \mathrm{LW}_{i|\Omega_j}. \tag{2.66}$$

The first term in Eq. (2.66) is the cell residual computed as in Eq. (2.53). It is weighted by the volume of the cell divided by the number of vertices of the element. The $\mathrm{LW}_{i|\Omega_j}$ term is computed on the dual cell $\mathcal{C}_i$ taking advantage of the Green-Gauss theorem:

$$\mathrm{LW}_{i|\Omega_j} = \frac{1}{2}\Delta t \iiint_{\Omega_j \cap \mathcal{C}_i} \nabla \cdot [\mathcal{A}(\nabla \cdot \mathbf{F})]\, \mathrm{d}V = \frac{1}{2}\Delta t \iint_{\partial \mathcal{C}_i} \nabla \cdot [\mathcal{A}(\nabla \cdot \mathbf{F})]\, \mathrm{d}S. \tag{2.67}$$

This term is then discretized to give:

$$\mathrm{LW}_{i|\Omega_j} \simeq \frac{1}{2}\Delta t\, [\mathcal{A}(\nabla \cdot \mathbf{F})]_{\Omega_j} \cdot \frac{S_{i|\Omega_j}}{n_d}, \tag{2.68}$$

where $S_{i|\Omega_j}$ is the normal associated with node i and cell j it is computed according to the scaling by n_d. It should be noticed that no weighting is required for the LW term because it is computed on the dual cell. Substituting Eq. (2.53) and (2.68) into Eq. (2.66) leads to:

$$R_{i|\Omega_j} = \left(I - \frac{\Delta t}{2n_d} \frac{n_{v(\Omega_j)}}{V_{\Omega_j}} \mathcal{A}_{\Omega_j} \cdot S_{i|\Omega_j} \right) R_{\Omega_j} \frac{V_{\Omega_j}}{n_{v(\Omega_j)}}. \tag{2.69}$$

Recalling now Eq. (2.65) the distribution matrix takes the form:

$$D_{i|\Omega_j} = \frac{1}{n_v(\Omega_j)} \left(I - \frac{\Delta t}{2n_d} \frac{n_{v(\Omega_j)}}{V_{\Omega_j}} \mathcal{A}_{\Omega_j} \cdot S_{i|\Omega_j} \right). \tag{2.70}$$

2.2.3 The TTGC numerical scheme

Taylor-Galerkin (TG) schemes were first derived by Donea [43, 44] the key idea of the method being the coupling between a Taylor expansion in time and a Galerkin discretization in space. Quartapelle [171] extended this approach deriving a family of Two-step Taylor Galerkin (TTG) schemes. Finally, Safjan [185] generalized this technique for multi-step schemes. The TTGC [33] scheme is a variation, more suited for LES, of the work of Quartapelle. It is formally third order in time and second-order in space but reaches third to fourth-order accuracy in space on coarse grids. Following the development of Colin [33], it can be written as:

$$\tilde{\mathbf{w}}^n = \mathbf{w}^n + \alpha \Delta t \left(\frac{\partial \mathbf{w}}{\partial t} \right)^n + \beta \Delta t^2 \left(\frac{\partial^2 \mathbf{w}}{\partial t^2} \right)^n \tag{2.71}$$

$$\mathbf{w}^{n+1} = \mathbf{w}^n + \Delta t \left(\frac{\partial \tilde{\mathbf{w}}}{\partial t} \right)^n + \gamma \Delta t^2 \left(\frac{\partial^2 \mathbf{w}}{\partial t^2} \right)^n \tag{2.72}$$

$$\alpha = \frac{1}{2} - \gamma \qquad \text{and} \qquad \beta = \frac{1}{6}. \tag{2.73}$$

First and second time derivatives can be replaced as done for the Lax-Wendroff scheme (see Eq. (2.62) and (2.63)) giving:

$$\tilde{\mathbf{w}}^n = \mathbf{w}^n - \alpha\Delta t\nabla\cdot\mathbf{F}^n + \beta\Delta t^2\nabla\cdot\left[\mathcal{A}\left(\nabla\cdot\mathbf{F}^n\right)\right] \tag{2.74}$$

$$\mathbf{w}^{n+1} = \mathbf{w}^n - \Delta t\nabla\cdot\widetilde{\mathbf{F}^n} + \gamma\Delta t^2\nabla\cdot\left[\mathcal{A}\left(\nabla\cdot\mathbf{F}^n\right)\right]. \tag{2.75}$$

Multiplying these equations by a set of linear test functions ϕ_i ("redskin tent" functions) and integrating them over the computational domain Ω, leads to the following weak formulation:

$$\int_\Omega \tilde{R}^n\phi_i dV = -\alpha L_i(\mathbf{w}^n) - \beta\Delta t LL_i(\mathbf{w}^n) \tag{2.76}$$

$$\int_\Omega R^{n+1}\phi_i dV = -L_i(\tilde{\mathbf{w}}^n) - \gamma\Delta t LL_i(\mathbf{w}^n) \tag{2.77}$$

with

$$\tilde{R}^n = \frac{\tilde{\mathbf{w}}^n - \mathbf{w}^n}{\Delta t}, \qquad R^{n+1} = \frac{\mathbf{w}^{n+1} - \mathbf{w}^n}{\Delta t} \tag{2.78}$$

and

$$L_i(\mathbf{w}^n) = \int_\Omega \nabla\cdot\mathbf{F}(\mathbf{w}^n)\phi_i dV \tag{2.79}$$

$$LL_i(\mathbf{w}^n) = \underbrace{\int_\Omega \mathcal{A}\left(\nabla\cdot\mathbf{F}(\mathbf{w}^n)\right)\nabla\phi_i dV}_{LL_i^0(\mathbf{w}^n)} - \underbrace{\int_{\partial\Omega}\phi_i\mathcal{A}\left(\nabla\cdot\mathbf{F}(\mathbf{w}^n)\right)dS}_{BT_i(\mathbf{w}^n)}. \tag{2.80}$$

The LL_i term can be split by performing an integration by parts assuming the surface normal dS external. The first contribution $LL_i^0(\mathbf{w}^n)$ is integrated over all the computational domain while the second one, $BT_i(\mathbf{w}^n)$, is non zero only at boundaries. It should be noticed that the LL_i term involves second spatial derivatives (like the LW_i term, see for example Eq. (2.68)), that are not expected when dealing with convection problems. The Galerkin method is then applied to the flux divergence and to residuals. Hence, they can be expressed as a sum of linear shape-functions (same functions as the test-functions used to derive the weak formulation), leading to:

$$R^n = \sum_k R_k^n\phi_k \tag{2.81}$$

$$\nabla\cdot\mathbf{F} = \sum_k F_k\nabla\phi_k, \tag{2.82}$$

where F_k is the discrete flux at each point of computational domain. With this choice of shape-functions, the residuals are recast as:

$$\int_\Omega \tilde{R}^n\phi_i dV = \sum_k\left(\int_\Omega \phi_i\phi_k dV\right)\tilde{R}_k^n = \sum_k \mathcal{M}_{ik}\tilde{R}_k^n, \tag{2.83}$$

denoting $\mathcal{M}_{ik}$ as the components of the so-called mass matrix which, in AVBP, is inverted locally by an iterative Jacobi method.

Coming now to the spatial discretization, the contributions of integrals in Eq. (2.79) and (2.80) to node i come only from the adjacent cells.

$$L_i(\mathbf{w}^n) = \sum_{j|i\in\Omega_j} L_i(\mathbf{w}^n)_{\Omega_j} \tag{2.84}$$

$$LL_i(\mathbf{w}^n) = \sum_{j|i\in\Omega_j} LL_i(\mathbf{w}^n)_{\Omega_j}. \tag{2.85}$$

Taking advantage of Eq. (2.82), $L_i(\mathbf{w}^n)_{\Omega_j}$ and $LL_i(\mathbf{w}^n)_{\Omega_j}$ can be written as:

$$L_i(\mathbf{w}^n)_{\Omega_j} = \sum_{k|k\in\Omega_j} F_k^n \int_{\Omega_j} \phi_i \nabla\phi_k dV \tag{2.86}$$

$$LL_i(\mathbf{w}^n)_{\Omega_j} = A_{\Omega_j}^n \sum_{k|k\in\Omega_j} F_k^n \int_{\Omega_j} \nabla\phi_i \cdot \nabla\phi_k dV - A_{\Omega_j}^n \sum_{k|k\in\partial\Omega_j\cap\partial\Omega} F_k^n \int_{\partial\Omega_j\cap\partial\Omega} \phi_i \nabla\phi_k dS. \tag{2.87}$$

For triangular and tetrahedron elements the gradient of the shape function is constant[1] over each element and the integral of ϕ_i takes a simple form [33].

$$\nabla\phi_k = -\frac{\vec{S}_k}{n_d V_{\Omega_j}} \tag{2.88}$$

$$\int_{\Omega_j} \phi_k dV = \frac{V_{\Omega_j}}{n_{v(\Omega_j)}} \qquad \forall k \in \Omega_j. \tag{2.89}$$

Substituting relations (2.88) and (2.89) in Eq. (2.86) yields:

$$L_i(\mathbf{w}^n)_{\Omega_j} = \sum_{k|k\in\Omega_j} F_k^n \nabla\phi_k \int_{\Omega_j} \phi_i dV = (\nabla\cdot\mathbf{F}^n)_{\Omega_j} \int_{\Omega_j} \phi_i dV = R_{\Omega_j}^n \frac{V_{\Omega_j}}{n_{v(\Omega_j)}}. \tag{2.90}$$

Applying the same procedure to the first term of Eq. (2.87) leads to:

$$LL_i^0(\mathbf{w}^n)_{\Omega_j} = A_{\Omega_j}^n \sum_{k|k\in\Omega_j} F_k^n \nabla\phi_k \cdot \int_{\Omega_j} \nabla\phi_i dV = -\frac{1}{n_d}(A_{\Omega_j}^n R_{\Omega_j}^n)\cdot S_{i|\Omega_j}. \tag{2.91}$$

These two operators are therefore equivalent to the ones encountered in the cell-vertex finite volume discretization (see Eq. (2.69)). The scaling for the nodal volume does not appear explicitly in this derivation but it is taken into account in the mass matrix.

A more complete study of numerical schemes available in AVBP can be consulted in Chapter 5 of N. Lamarque [107] thesis.

[1] For bilinear and trilinear elements (quads, hexahedra and pyramids for example) the gradient of the shape function over the element is no more constant. This difficulty is overcome by adding a correction to the residual computed as for linear element.

2.3 Boundary conditions

Boundary Conditions (BC) are an essential part in any CFD code, and especially in AVBP because of acoustics present in the governing equations [194, 164]. The correct formulation of the boundary conditions has been the subject of an extensive investigation [143]. Because of the wide range of applications of AVBP, a large set of BC is necessary, notably for internal flow configurations. Characteristic boundary conditions may be applied at inlet and outlet boundaries. At run time the user may select either the widely used full residual or the normal approach. In the normal approach (following [163]) the characteristic boundary conditions are applied to the normal flux derivatives of the residual, which is in general more accurate and implies a decomposition of variations in the conservative variables into a set of ingoing and outgoing waves. Hereafter, a brief overview of the theoretical method used in AVBP for prescribing the different BC available in the code.

The time integration in AVBP is performed with a multi-stage Runge-Kutta (RK) method. For simplicity, we only consider here a single-stage RK. Knowing the solution $\mathbf{w}^n$ at time t, the solution $\mathbf{w}^{n+1}$ at time $t + \Delta t$ is computed for each node i as:

$$\mathbf{w}_i^{n+1} = \mathbf{w}_i^n - \frac{\Delta t}{V_i} \cdot d\mathbf{w}_i^n, \tag{2.92}$$

where $\mathbf{w}_i^n = \mathbf{w}(t, \vec{x}_i)$ and $\mathbf{w}_i^{n+1} = \mathbf{w}(t + \Delta t, \vec{x}_i)$; $\vec{x}_i$ is the coordinate vector, V_i is the nodal volume around node i and $d\mathbf{w}_i^n$ is the nodal residual at node i, as computed by the numerical scheme. The n superscript is here to remind that AVBP is an explicit code, hence this nodal residual only depends on quantities known at the time step n. This formula is applied to each node of the computational domain (Ω). If no physical BC was imposed, the computed solution at each time step would only depend on the initial solution and on the numerical scheme. In order to impose a BC on the border of the domain ($\partial\Omega$), we write:

$$\begin{cases} \mathbf{w}_i^{n+1} = \mathbf{w}_i^n - \frac{\Delta t}{V_i} \cdot (d\mathbf{w}_i^n)_{scheme} & \forall x_i \in \Omega/\partial\Omega \\ \mathbf{w}_i^{n+1} = \mathbf{w}_i^n - \frac{\Delta t}{V_i} \cdot (d\mathbf{w}_i^n)_{BC} & \forall x_i \in \partial\Omega. \end{cases} \tag{2.93}$$

For each node lying on the boundary, we have replaced the *scheme-predicted* residual by a *BC-corrected* residual. This operation is known as imposing the BC in a "hard way". A "weak" method is additionally used in conjunction with this hard method for certain BC. In this case, some gradients and fluxes are modified before applying the numerical scheme. It is mainly used for Von Neuman like conditions, as adiabaticity or impermeability. The AVBP BC can be classified through two categories:

- Non-characteristic BC, that work directly on conserved variables.
- Characteristic BC, that use a wave decomposition to modify residuals.

The latter approach is one of the key features in AVBP, as it enables a general treatment of acoustic BC (see Struijs *et al.* [212] for detail information in wave decomposition and the theory of characteristics). Finally, two types of BC do not fit in this classification and are not treated like the other BC:

- Periodicity conditions.
- Axisymmetric conditions.

2.4 Domain partitioning

Many large-scale computational problems in engineering are solved on unstructured computational domains. Some examples are unstructured grid calculations based on finite volume methods in CFD, structural analysis problems based on finite element approximations, pattern recognition and image processing [199], very-large-scale integration (VLSI) design [49, 3], data mining [231, 14], etc. One of the key problems when implementing such large-scale unstructured problems on a distributed memory machine is to partition the underlying computational domain in an efficient manner. The fundamental problem is that of splitting a large irregular graph into k parts (that usually corresponds to the number of processors of the parallel computation). Graph partitioning software targeted at parallel computing began to appear in the early 90's. A non-exhaustive list of partitioning algorithms is presented in Fig. 2.5, where algorithms are classified as a function of their partitioning quality and their computational requirements. Research in the last fifteen years has provided a number of computationally efficient algorithms [192]. The combination of existing algorithms and parallel implementations of existing ones has also contributed to broaden the area of grid partitioning. Multilevel graph partitioning algorithms (see Fig. 2.5) are currently considered to be the state-of-the-art and are used extensively.

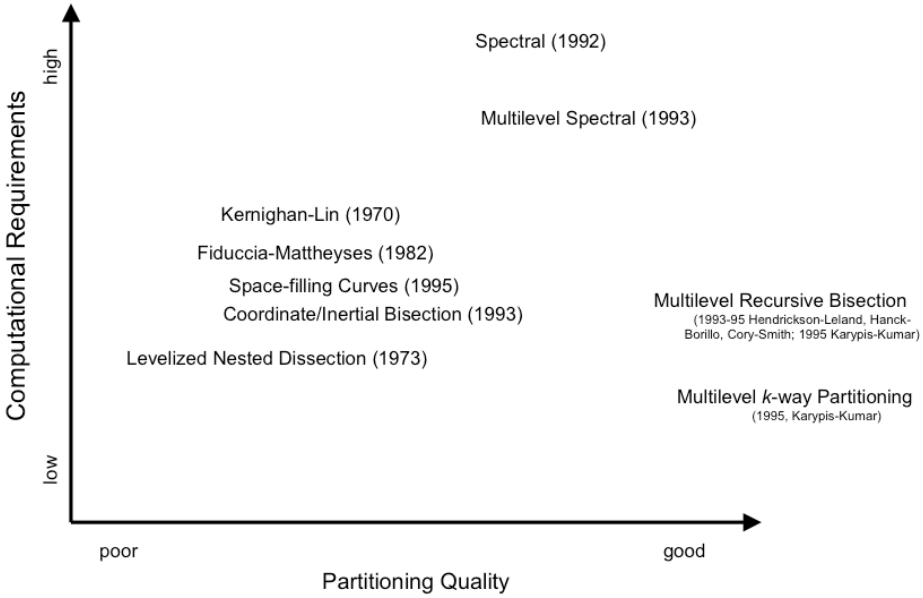


Figure 2.5 - *History of partitioning algorithms. From Kumar [105].*

This section describes the partitioning strategy used in the AVBP code. For standard AVBP runs (which use partitioning at run time and not pre-partitioning), partitioning is a sequential task performed solely by the master processor at the beginning of the simulation, no dynamic load balancing option is currently available. Some basic notions about static[2] load balancing are first given in Subsection 2.4.1. Then, a general description of the three current partitioning algorithms is presented in Subsection 2.4.2. Subsection 2.4.3 analyses performances of the current partitioning algorithms. Subsection 2.4.4 details the introduction of a new option for partitioning grids with the multilevel *k-way* algorithm described in Karypis & Kumar [96]. Finally, Subsection 2.4.5 explains ordering strategies available in AVBP.

[2] A problem is called static when the graph does not change during the computation.

2.4.1 Static load balancing

The numerical solution of partial differential equations [172] usually involves dividing up the physical domain into small elements or volumes that form the computational mesh [15] (Fig. 2.6 (a)). To solve the problem on a distributed memory parallel computer, the mesh should be decomposed into subdomains, with each subdomain assigned to a unique processor. The static load balancing problem consists of decomposing the mesh into subdomains in such a way that each processor has about the same amount of computation and the communication cost between processors is minimized, with the overall aim of minimizing the runtime of the calculation. Static load balancing implies also that neither the graph connectivity nor the number of nodes change during the computation. Figure 2.6 (b) displays the dual graph representation (gray) of the computational mesh (black) to partition, where each vertex represents a finite element in the mesh and each edge represents a connexion between the faces of two elements (and therefore, the communication between adjacent mesh elements). The number of edges of the dual graph that are being cut in the partitioning is called the *edge-cut*. This subdivision results in an increase of the total number of the grid points due to a duplication of nodes at the interface between two subdomains. The communication cost of a subdomain is a function of its edge-cut as well as the number of neighbouring subdomains that share edges with it. In practice, the edge-cut of a partitioning is usually used as an important indicator of the quality of the partitioning. The sparse matrix related to the dual graph (Fig. 2.6 (c)) of this grid is shown in Fig. 2.6 (d). The axes represent the vertices in the graph and the black squares show an edge between vertices (i, j). In scientific computing, the order of the elements in the sparse matrix often affects the performance of numerical algorithms (see Subsection 2.4.5).

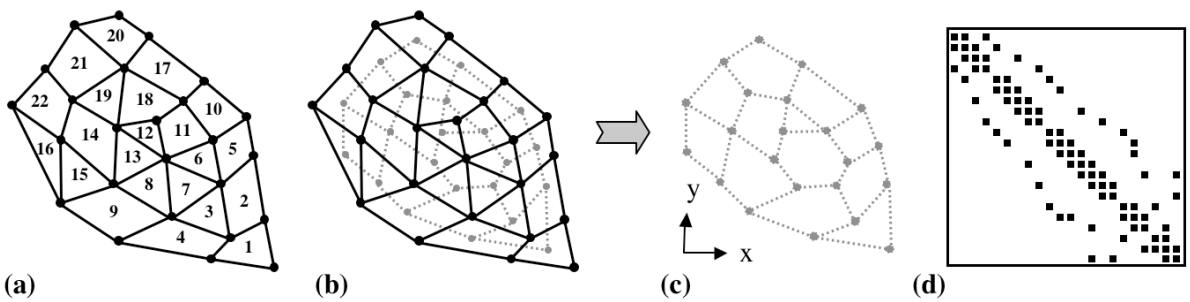


Figure 2.6 - *Example of grid. (a) The nodal graph; (b) nodal (black) and dual (gray) graph; (c) dual graph; (d) sparse matrix associated with the dual graph.*

Different partitions of the mesh may result in quite different CPU times to get a computation to completion. It is therefore necessary to examine the quality of the partitioning based on its effect on the application. This is discussed next. These algorithms belong to a modular software library called COUPL [193] (now called MPL) that aims to free the nonspecialist user from the need to consider aspects of high-performance computing.

2.4.2 Partitioning algorithms available in AVBP.

This section presents the three different partitioning algorithms currently available in AVBP. The first two are within the category of geometric based algorithms since they take advantage of coordinates and distances between nodes (*Euclidean distance*) and the third one is an example of graph theory based algorithms which make usage of the connectivity information of the graph.

- **Recursive Coordinate Bisection (RCB):**

  RCB (Berger & Bokhari [13]) is a well-known algorithm[3] which bisects a domain into two pieces of equal workload recursively until the number of subdomains is equal to the number of processors. Figure 2.7 shows the result of applying RCB to partition a grid (see Fig. 2.6) into four subdomains. The dual graph of a simple unstructured grid (Fig. 2.7 (a)) is first divided into two regions by a cutting plane orthogonal to one of the coordinate axes so that half the workload is associated to the sub-regions (Fig. 2.7 (b)). The splitting direction is determined by computing in which coordinate direction the set of objects is most elongated, based upon the geometric locations of the objects. The sub-regions are then further divided by recursive application of the same splitting algorithm (Fig. 2.7 (c)) until the number of sub-regions equals the number of processors (Fig. 2.7 (d)). Although this algorithm was first devised to cut into a number of sets which is a power of two ($k = 2^p$) (where k is the number of partitions and p the number of steps) the set sizes in a particular cut do not need to be equal. By adjusting the partition sizes appropriately, any number of equally-sized sets can be created. The weakness of this algorithm is that it does not take advantage of the connectivity information given by the graph.

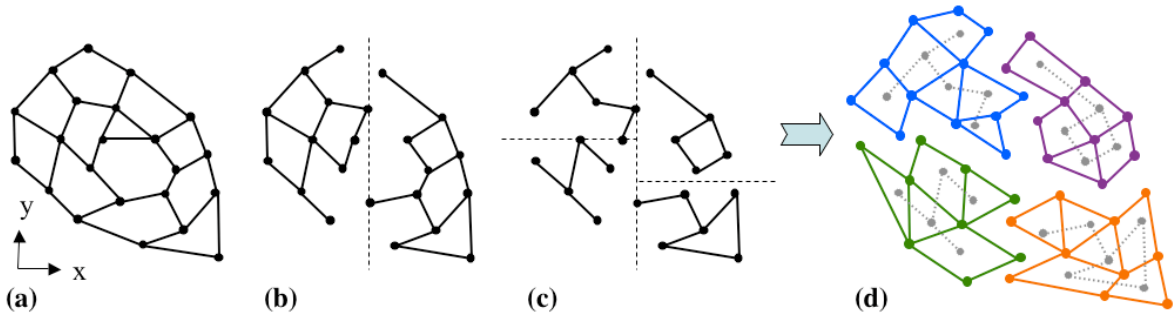


Figure 2.7 - *Example of partitioning a graph into 4 subdomains by RCB algorithm. (a) The initial graph; (b) partitions after the first recursive step; (c) partitions after the second and last recursive step; (d) the four subdomains of the unstructured grid. Edge-cut = 13. N^o of duplicated nodes = 16.*

- **Recursive Inertial Bisection (RIB):**

  RIB was proposed as a load-balancing algorithm by Williams [229] and later studied by Keyser & Roose [99] and Taylor & Nour-Omid [215]. RIB is similar to RCB in that it divides the domain based on the location of the objects being partitioned by use of cutting planes. Partitioning using coordinate bisection is susceptible to the orientation of the mesh, a simple rotation of the mesh would result in different partition. Inertial bisection remedies this by using a procedure that is invariant to rotation. Figure 2.8 shows the result of applying RIB to partition a grid (see Fig. 2.6) into four subdomains. The dual graph of a simple unstructured grid (Fig. 2.8 (a)) is first divided into two regions by a cutting plane orthogonal to the principal axis of the graph[4] (or longest direction of the domain) so that half the workload is in each of the sub-regions (Fig. 2.8 (b)). The sub-regions are then further divided by recursive application of the same splitting algorithm (Fig. 2.8 (c)) until the number of sub-regions equals the number of processors (Fig. 2.8 (d)). As for RCB, the weakness of this algorithm is that it does not take advantage of the connectivity information.

[3] It is closely related to the *multidimensional binary tree* or *k-D* tree data structure proposed by Bentley [12].

[4] If one assumes that each vertex has a unit mass associated with it, then the principal axis is the one that minimizes the angular momentum when the graph rotates around it.

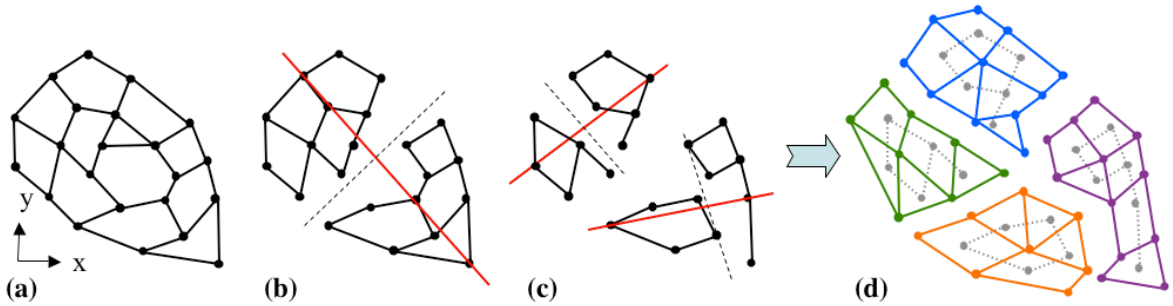


Figure 2.8 - *Example of partitioning a graph into 4 subdomains by RIB algorithm. (a) The initial graph; (b) partitions after the first recursive step; (c) the partitioned graph after RIB. Red lines represent the principle axis; (d) the four subdomains of the unstructured grid. Edge-cut = 11. N° of duplicated nodes = 14.*

- **Recursive Graph Bisection (RGB):**

  Instead of using the *Euclidean distance* between nodes (as done by RCB and RIB), RGB (Simon [200]) rather consider the *graph distance* between two nodes given by the number of edges on the shortest path connecting them. Figure 2.9 shows the result of applying RGB to partition a grid (see Fig. 2.6) into four subdomains. Figure 2.9 (a) represents the dual graph of a simple unstructured grid. This algorithm begins by finding a pseudo-peripheral node in the graph (Fig. 2.9 (b)), i.e., one of a pair of nodes (called the *root*) that are approximately at the greatest graph distance[5] from each other in the graph. Then, starting in the selected node, the graph distance from this node to every other node is determined (Fig. 2.9 (c)) and all other nodes are sorted in order of increasing distance from one of the extremal vertices. Finally, nodes are assigned to two subdomains according to the graph distance. This process is then recursively executed on each of the subdomains. The only difficulty is the determination of the diameter (or at least of a pseudo-diameter) of the graph. However, a heuristic procedure can be employed for this purpose such as Cuthill-McKee (CM) [38] or the reverse Cuthill-McKee (RCM) [71] algorithms, quite well-known algorithms by sparse matrix community [48]. In Avbp, pseudo-peripheral nodes are obtained with the Lewis' implementation [116] of the Gibbs-Poole-Stockmeyer algorithm [73] and nodes reordering can be done by the CM or RCM (default value) algorithms (see also Subsection 2.4.5).

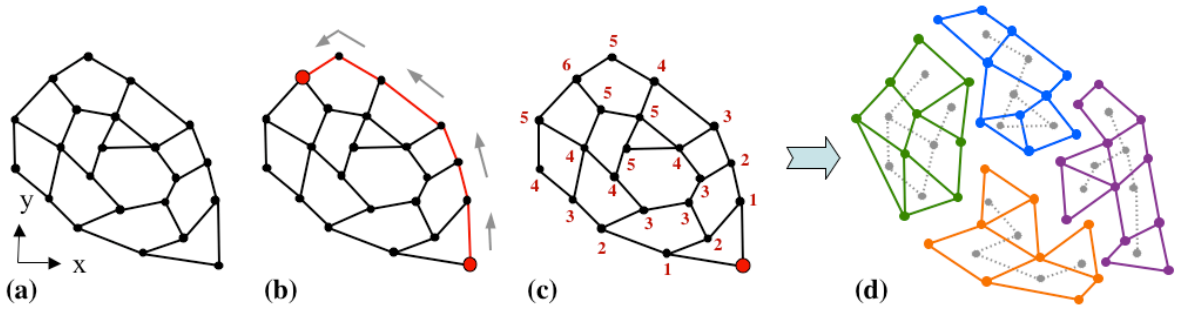


Figure 2.9 - *Example of partitioning a graph into 4 subdomains by RGB algorithm. (a) The initial graph; (b) pseudo-peripheral nodes; (c) nodes distances to the root; (d) the four subdomains of the unstructured grid. Edge-cut = 12. N° of duplicated nodes = 15.*

[5] Their distance is called the *diameter* of the graph.

2.4.3 Analysis of partitioning algorithm performances in AVBP.

Recursive bisection algorithms produce partitions of reasonable quality for static irregular problems, with relatively low overhead when compared with other partitioning algorithms, e.g., recursive spectral bisection (RSB) (Pothen *et al.* [166]). Simon & Teng [201], von Hanxleden & Scott [80] and Williams [229] discuss the qualities of partitions produced by recursive bisection algorithms, and some of them compare their performance with other partitioning methods in several aspects. In the following section, a set of simple tests are performed for grids representing the kind of problems resolved in CFD. Figure 2.10 (a)-(c) shows the snapshot of a grid partitioned in 256 subdomains with the three partitioning algorithms mentioned before: RCB, RIB and RGB, respectively. As it can be observed, the subdomains interfaces resulting from the coordinate bisection try to follow the coordinate axis. Inertial bisection produces a similar result in the axial direction but cutting planes in the x-y view are mainly oriented in the radial direction. As expected from the description of the RGB algorithm, graph bisection produces a completely different partitioning.

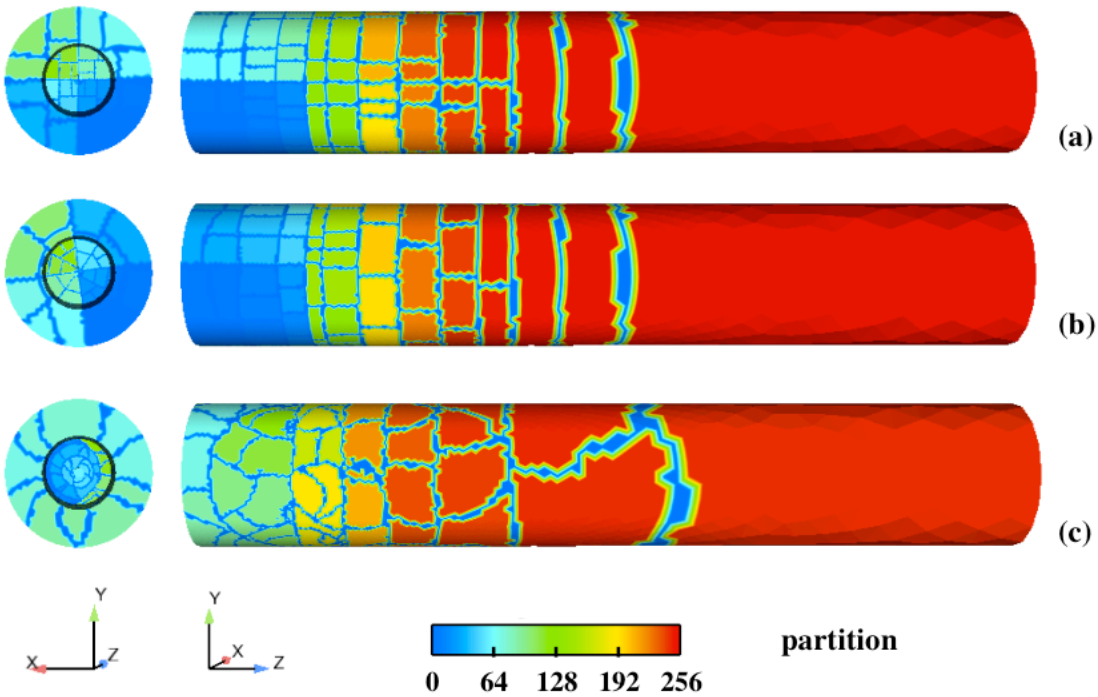


Figure 2.10 - *Snapshot of a grid partitioned with different algorithms: (a) RCB; (b) RIB and (c) RGB.*

Table 2.1 summarises grid parameters of a computation performed on an IBM Blue Gene/L[6] The number of nodes and cells of the grid are: 367,313 and 2,058,883. An important difference in the number of nodes duplicated by the partitioning can be observed. The differences in CPU time to perform 1000 iterations with AVBP comes from the increase in the total number of nodes since the higher the number of nodes duplicated after the partition, the bigger the communication cost and therefore, the higher the amount of CPU time used during the calculation with AVBP. On massively parallel machines, the choice of the partitioning algorithm can have an important effect on the total CPU time of the computation since the communication costs can easily increase if the

[6] This machine has 1024 nodes with 2 processors/node and 512 MB/processor.

number of duplicated nodes is too high. For example, in this particular case, the RCB partitioning algorithm runs on 256 processors 12% faster than the RGB partitioning algorithm and slightly faster than the RIB (1.5%) algorithm.

Partitioning algorithm	**RCB**	**RIB**	**RGB**
N^o of nodes after partitioning	495,232	503,230	530,852
N^o of duplicated nodes	127,919	135,917	163,539
N^o of nodes increment	+35%	+37%	+45%
CPU time for 1000 it. (s)	361.55	366.96	405.64
CPU time increment	—	+1.5%	+12%

Table 2.1 - *Summary of the number of nodes after partitioning a tetrahedron-based grid on 256 subdomains with three partitioning algorithms and CPU time to perform 1000 iterations. Simulations performed on an IBM Blue Gene/L supercomputer.*

Among the factors contributing to the selection of a good partitioning algorithm, the quality of the partition and the time to perform it play an important role. Other desirable properties are discussed in more detail in Hendrickson & Devine [85] but we limit our study to these two criteria since they are clear enough to emphasize the need for new solutions to perform calculations in massively parallel machines with Avbp. In addition, the higher the number of partitioning algorithms available within the same code, the higher the possibilities to choose the right one for each case since often it is not clear which one is better for a given scenario.

The next tests evaluate the performance of the current partitioning algorithms on three different grids summarised in Table 2.2. The increasing factor in the number of cells has been explicitly chosen to study its effect. All tests have been performed on an AMD Opteron computer with 8 GB/processor to handle memory required for partitioning such large grids. In the following, we distinguish between the total (or global) CPU time used for partitioning a grid with Avbp (which includes: graph partitioning, nodes reordering and blocks generation) and the CPU time needed only by the partitioning algorithm to cut the graph. The wide range of the number of subdomains cut — from 64 (2^6) to 16,384 (2^{14}) — was chosen to evaluate the scalability of the different partitioning algorithms for a high number of applications (present and future ones). With the arrival of massively parallel computers with thousands of processors[7] one needs to consider the partitioning as one of the elements that affect the overall efficiency of the computation.

Grid name	**N^o of nodes**	**N^o of cells**
MESHSTVM3D7	3,255,085	3,207,960
ARRIUS2_10M	1,875,835	10,620,245
ARRIUS2_44M	7,661,005	43,949,682

Table 2.2 - *Characteristics of three different grids used to evaluate the different partitioning algorithms.*

[7] The 32nd TOP500 list (November 2008) of the most powerful supercomputers in the world can be consulted on http://www.top500.org/lists/2008/11.

Figure 2.11 displays the CPU time used to perform the total partition of *ARRIUS2_44M* (solid lines) with the three algorithms. CPU time spent to cut the graph (dashed lines) is also included to highlight its contribution in the whole partitioning process. One curve captures our attention: the exponential behaviour observed for the RGB algorithm and the CPU time needed was unexpected: more than 4.5 hours to partition a grid of 44 millions cells on 4096 (2^{12}) subdomains! Even if the slope for coordinate and inertial bisection is less rough (30 and 44 minutes, respectively, to do the same task) the time needed to partition such a large grid is not negligible.

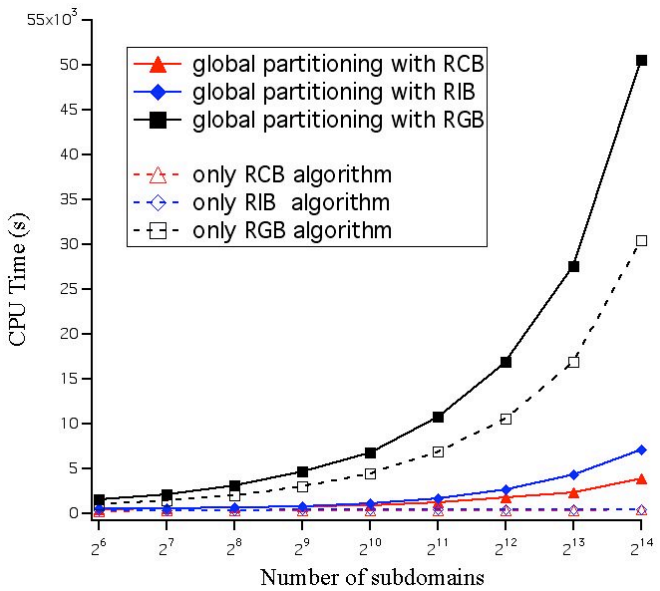


Figure 2.11 - *CPU time spent on the global partitioning task (solid lines) and to cut the graph (dashed lines) with RCB, RIB and RGB for ARRIUS2_44M.*

Of course, the effect of partitioning is less evident on a simulation with the number of nodes and cells more representative of the grid dimensions currently used with Avbp. Figure 2.12 (a) shows the CPU time used by the three different partitioning algorithms mentioned before to partition *MESHSTVM3D7*. CPU time is quite reasonable for such low size calculations. It can be observed that the increment in the number of processors has a very small effect on the run time of both coordinate algorithms (RCB and RIB) which require moderate amount of time but graph bisection displays the same exponential behaviour mentioned before while increasing the number of subdomains to partition. Another simple test is the calculation of the effect of different grid sizes with the same partitioning algorithm. Figure 2.12 (b) shows the CPU time spent to cut the graph for the three grids with the RIB partitioning algorithm (default one in Avbp). It can be observed that the higher the size of the grid, the higher the CPU time needed to partition it. The slope for grids with low and moderate sizes is quite low, on the contrary, the tendency for high size problems is higher.

2.4.4 Introduction of a new partitioning algorithm

With Lagrangian calculations in perspective and for calculations in massively parallel machines, the choice of a new partitioning algorithm becomes necessary. Firstly, one important problem related to Lagrangian calculations is load balancing (Ham *et al.* [76], García *et al.* [69]). Depending on

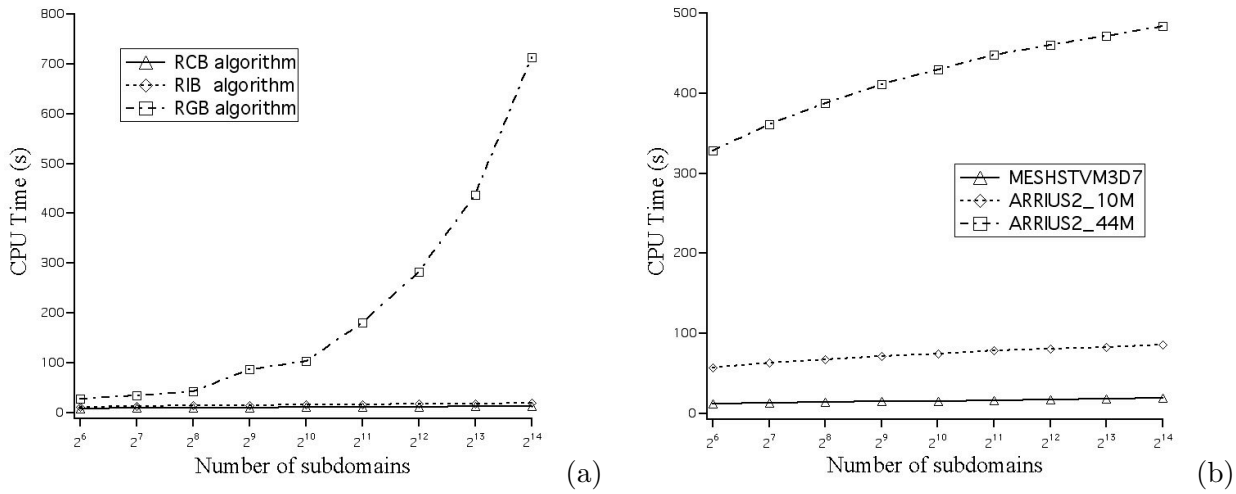


Figure 2.12 - *(a) CPU time to cut the dual graph of MESHSTVM3D7 with different algorithms: RCB, RIB and RGB; (b) CPU time to cut three different grids with the same algorithm: RIB.*

the kind of calculation, particles may need to be repartitioned repeatedly to reduce imbalance. Load balancing is therefore essential for applications such as particle-in-cell codes where workload distribution fluctuates as particles move around. Current algorithms in AVBP are not able to take into account particles information while partitioning the grid.

The need of a multi-constraint partitioning algorithm can be crucial in some cases to balance both the number of mesh elements and the number of particles across the subdomains to reduce time of the two-phase flow computation. Secondly, time spent during partitioning is time lost to the application, and can only be justified by a resulting improvement in application performance. Available algorithms in AVBP seem not to be completely adapted for high size problems. Obtaining a good efficiency to partition a grid on expensive parallel machines can be important since the cost of one hour of calculation by processor is around 0.2 euros. Note that running a parallel computation with a sequential partitioning on this kind of machines can easily increase its cost: e.g., to partition *ARRIUS2_44M* with RCB, 0.5 [hours] * 4096 [processors] * 0.2 [euros/processor/hour]= 409.6 euros!

One of the first tasks of this thesis is the introduction of a new partitioning algorithm more adapted to Lagrangian simulations in massively parallel machines. It should be faster and produce better partitions that the actual ones. Among the currently available alternatives in serial graph partitioners are Chaco [87], METIS [95], Jostle [222], PARTY [167] and SCOTCH [156]. A general comparison of these different packages shows that only two of them include a formulation capable of performing multiconstraint graph partitioning: Jostle and METIS. In addition, both partitioners offer a parallel version that could be interesting for future applications (PJostle and ParMETIS). METIS has already been used in two-phase flow applications to solve the dispersed phase with a Lagrangian approach (Ham *et al.* [76]) and is also used in commercial codes like FLUENT or CFX. Therefore, it has been retained as a good candidate for testing. The results of some preliminary tests are discussed after a brief description of this algorithm package.

- **METIS:**

  METIS is a software package for partitioning large irregular graphs, partitioning large meshes, and computing fill-reducing orderings of sparse matrices. The METIS algorithms are based on multilevel graph partitioning: multilevel recursive bisectioning described in Karypis & Kumar [95] and multilevel *k-way* partitioning described in Karypis & Kumar [96]. Multilevel partitioning algorithms first reduce the size of the graph by coarsening the graphs details (Fig. 2.13). This takes form as collapsing adjacent vertices and edges. The smaller graph is then partitioned and refined into the original graph. As the partitioning algorithms operate with the reduced-size graph, they are extremely fast compared to traditional partitioning algorithms that compute a partition directly on the original graph. Extensive testing has also shown that the partitions provided by METIS are 10% to 60% better than those produced by spectral partitioning algorithms [166, 9], and 5% to 30% than those produced by other multilevel algorithms [86].

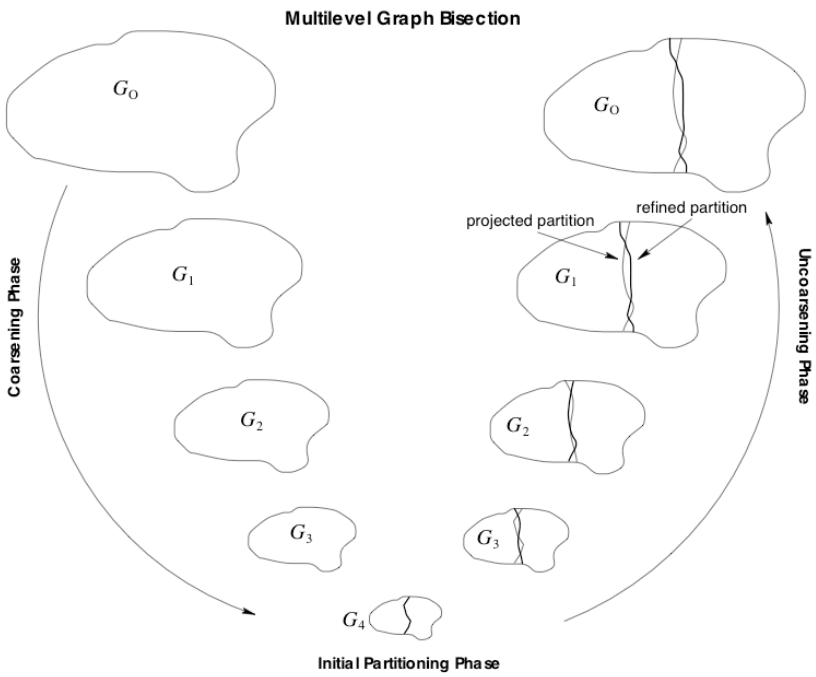


Figure 2.13 - *The various phases of the multilevel graph bisection. During the coarsening phase, the size of the graph is successively decreased; during the initial partitioning phase, a bisection of the smaller graph is computed; and during the uncoarsening phase,the bisection is successively refined as it is projected to the larger graphs. During the uncoarsening phase the light lines indicate projected partitions, and dark lines indicate partitions that were produced after refinement. From Karypis & Kumar [95].*

During this thesis, the version 4.0.1 of METIS has been integrated into the code AVBP and the multilevel *k-way* partitioning algorithm is now available. Several tests with different grid sizes presented in Table 2.2 have confirmed an improvement in the CPU time spent to perform the partitioning and a reduction in the communication cost. Partitioning memory requirements are higher since METIS subroutines need a new array with the information of the adjacency structure of the graph and the weights of the vertices and edges (if any) in compressed storage row (CSR) format. To this purpose, the package contains a subroutine to determine if the amount of memory in the system is sufficient for METIS.

Figure 2.14 (a) displays the CPU time needed to partition the three different grids with this new algorithm. Comparing this figure with Fig. 2.12 (b) it can be observed that METIS is always faster, except for small grid sizes and for large number of processors ($> 2^{13}$). Figure 2.14 (b) displays the total CPU time to perform the total partitioning. It can be observed that the new partitioning algorithm is clearly faster than the fastest algorithm currently available with high grid sizes. This fact is reinforced by plotting the number of nodes after partitioning. Figure 2.15 shows this information for the (a) moderate and (b) large grid sizes with RCB and *k-way*. The reduction in the number of nodes duplicated with the new algorithm is evident. The gap is higher as the number of subdomains increases and implies an important gain in the efficiency of the calculation due to the reduction of communication costs.

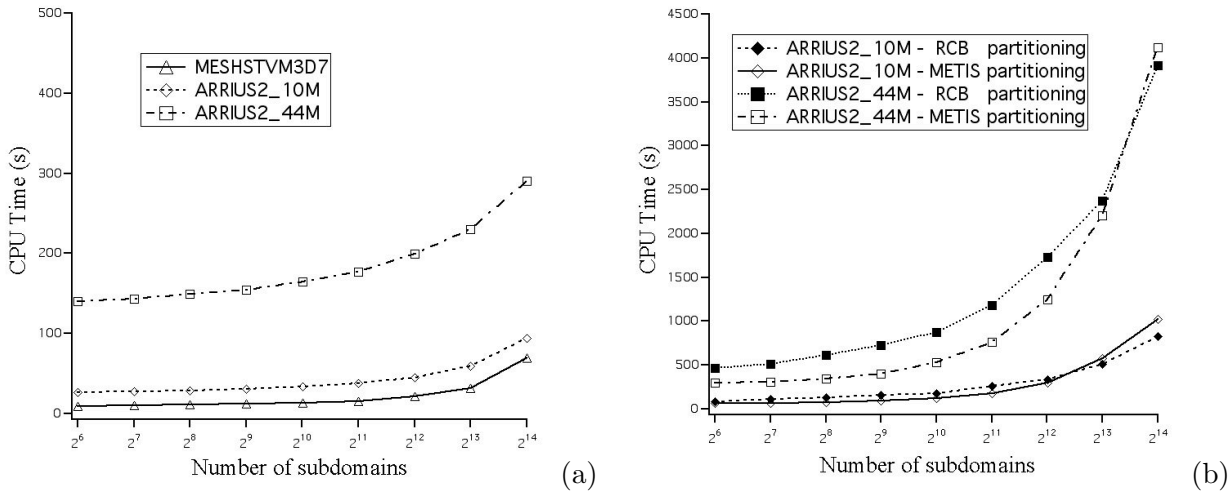


Figure 2.14 - *(a) CPU time with the multilevel k-way algorithm for three different grids. (b) CPU time to perform the whole partition for the moderate and high grids with RCB and the multilevel k-way algorithms.*

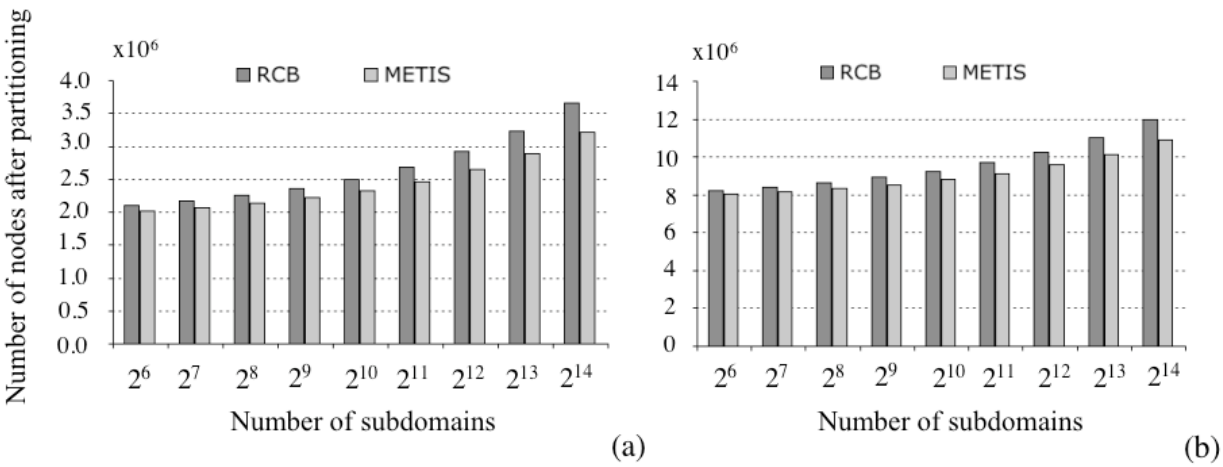


Figure 2.15 - *Total number of nodes after partitioning the grid with RCB and METIS (k-way) for (a) ARRIUS2_10M and (b) ARRIUS2_44M.*

2.4.5 Node reordering

This section addresses another issue of partitioning that is data reordering techniques. The idea is to minimize the bandwidth[8] of the sparse matrix (which represents the problem to solve) by renumbering the nodes of the computational mesh in such a way that they are as close as possible to their neighbours. This guaranties a best use of memory in computational resources and is mandatory in unstructured CFD solvers. We will use the simple two-dimensional (2D) mesh shown in Fig. 2.16 to demonstrate the use of these techniques.

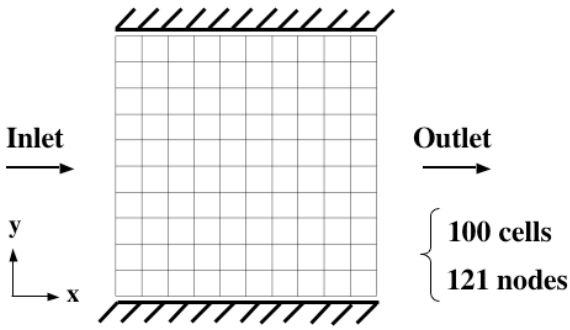


Figure 2.16 - *Example of a two-dimensional (2D) mesh with rectangular cells.*

An example of reordering applied to this simple grid is presented in Fig. 2.17. The nodes of the grid are initially numbered starting from left to right and from bottom to top (Fig. 2.17 (a)). This initial distribution presents lowest node numbers on the bottom of the grid (across the lower wall) and highest node numbers at the top (on the opposite wall). As previously mentioned, partitioning is done with the dual graph of the mesh and therefore reordering is firstly applied to the elements of the grid. In addition to the natural ordering, two algorithms are available in AVBP to reorder elements after partitioning: the Cuthill-McKee (CM) [38] and its reverse (RCM) [71]. Once elements reordering is finished, nodes are also renumbered respecting local numbering of the cell vertices (see Fig. 2.19). Results of node reordering with both algorithms are presented in Fig. 2.17 (b-c).

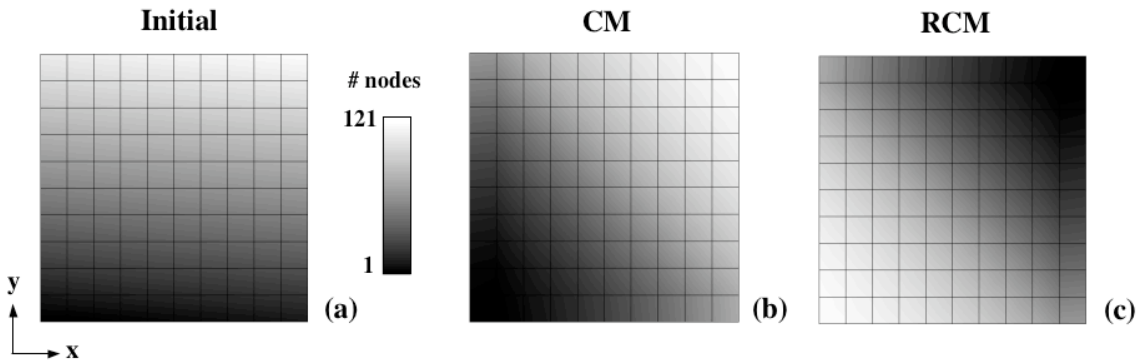


Figure 2.17 - *Example of reordering techniques: (a) initial order; (b) Cuthill-McKee (CM) and (c) Reverse Cuthill-McKee (RCM).*

After the representation of the grid as a graph, the procedure is as follows in both cases:

[8] We say that a symmetrically structured matrix $\mathbf{A}$ has *bandwidth* $2m+1$ if m is the smallest integer such that $a_{ij} = 0$ whenever $|i-j| > m$.

1. Choose a suitable starting node (pseudo-peripheral). This choice may be important and in AVBP, pseudo-peripheral nodes are obtained with the Lewis implementation [116] of the Gibbs-Poole-Stockmeyer algorithm [73].

2. For each node reachable from the current node:

   - Output the node.
   - Find all unvisited neighbours.
   - Order them based on increasing/decreasing degree.
   - Visit them in that order.

Looking at Fig. 2.17 (b) and (c) we observe that in the CM (RCM) case the starting point is the element of the bottom-left (top-right) corner of the grid. In both cases, all the elements connected to the starting point are visited in order. The result of reordering in this simple case produces a reorganization of nodes in the diagonal direction. This leads to a higher computational efficiency since node numbering is closer and cache memory is better used.

There exists another extended way to visualize relational data, i.e., as a reordered matrix. The basic idea has been around since the early days of computer science [16, 17, 81, 30]. In the matrix representation of a graph, each row and column in the matrix corresponds to a node in the graph. Each edge between two nodes (a, b) is rendered as a nonzero at (i, j), where i is the row for a and j is the column for b[9]. Figures 2.18 (a) and (b) display the pattern of the sparse matrices (or nonzero maps) of the initial and the reordered distribution. In this case, the bandwidth is equal to 21 for the initial and the reordered matrices. Note that, even if there is no difference between CM and RCM for this particular case, Liu & Sherman [119] show that reversing the Cuthill-McKee order is a better solution in general in terms of amount of matrix storage. The reduction in the number of zeros within the variable-band form is clearly visible for the reordered matrices (Fig. 2.18 (b)) due to the reduction of spaces mainly at the extremes of the diagonal. The effect of this reduction on a simple simulation is: the improvement of cache utilization and data locality, a reduced indirect addressing, a minimization of data movement in memory hierarchy and a gain of processor performance.

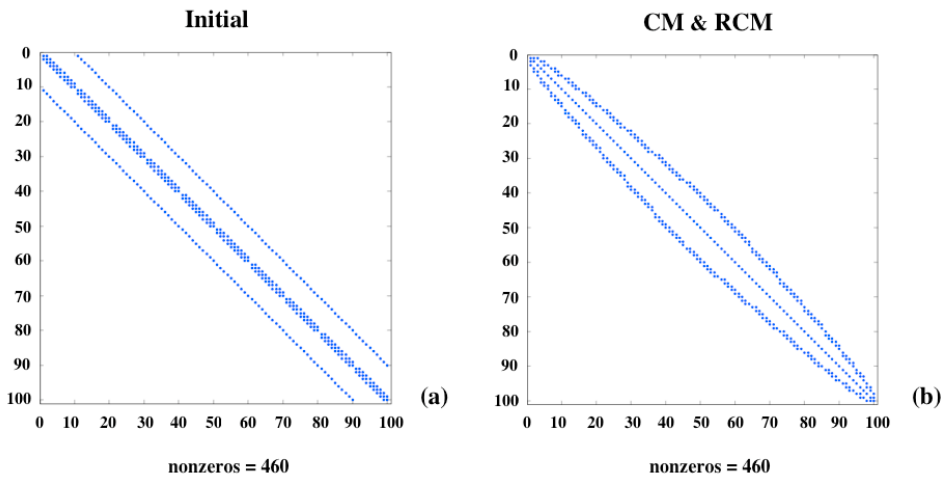


Figure 2.18 - *Sparse matrix: (a) initial order and (b) CM and RCM ordering. Bandwidth is* 21 *in all cases.*

[9] The solid diagonal shows the identity relationship for each node.

An example of the effects of node reordering in the performance of a simulation with AVBP is presented in Table 2.3. The grid used in these tests is *MESHSTVM3D7* (see Table 2.2), the partitioning algorithm is RIB. As it can be observed, even for such a small grid (3.2 million cells) the gain obtained just reordering nodes before the beginning of the computation is of 37% for a 8-processor simulation and of 25% for a 32-processor simulation. These results let imagine the importance of reordering strategies for the performance of unstructured CFD solvers like AVBP which are focused on the resolution of large grid size problems on massively parallel machines. The influence of memory defaults due to bad cell local numbering in these cases could also be important.

Graph ordering	**N^o of procs.**	**Initial**	**CM**	**RCM**
CPU time for 150 it. (s)	8	3689.75	2339.18	2324.99
CPU time reduction	8	—	−36.6%	−37%
CPU time for 150 it. (s)	32	810.73	603.19	607.51
CPU time reduction	32	—	−25.6%	−25%

Table 2.3 - *Summary of the CPU time to perform 150 iterations with MESHSTVM3D7 and the RIB algorithm. Simulations performed on an IBM JS21 supercomputer.*

In LES of turbulent flows on parallel machines, node reordering plays an additional role which was studied in detail during this thesis: it determines the growth of rounding errors and can control the differences between instantaneous LES solutions computed on different numbers of processors. The results of this work are included in an article published in July 2008, in AIAA Journal (see Appendix D: *Effects of node ordering in mesh*), treating the growth of rounding errors and the reproducibility character of large-eddy simulations. Before discussing these results – and the idea that was behind this article and its consequences – the data structure available in AVBP and the parallel strategies are presented in Sections 2.5 and 2.6. In Section 2.7 we discuss the basic notions of finite-precision computations and the concept of "the true nature of turbulence"as stated by Jiménez [93], which are essential for a clear understanding of the problem. A brief summary of the main results and conclusions close this chapter.

2.5 Data structure

The flow solver AVBP is based on the unstructured and hybrid grid technology which is capable to handle grids composed of arbitrary element types. In a so-called hybrid grid, a combination of elements of different cell type is used in the same mesh. Figure 2.19 shows the different kind of elements available in AVBP with the local numbering of cell vertices.

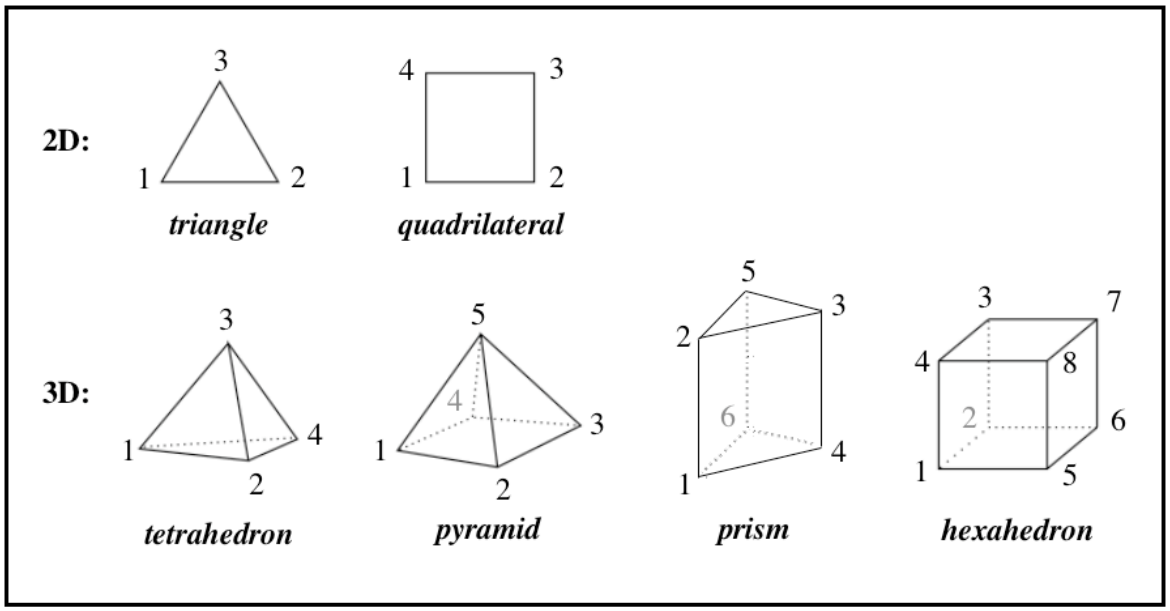


Figure 2.19 - *List of basic canonical elements used in* AVBP *for two-dimensional (2D) and three-dimensional (3D) grids. Local numbering of the cell vertices is also included.*

The hybrid grid approach aims to combine the advantages of the structured and unstructured grid methodologies. Although structured multi-block methods are generally more efficient for a given solution accuracy when compared to unstructured grid methods, they often need an impractical number of grid points due to the topology of hexahedra or the fact that they usually only permit length-scale variations in one direction at a time. Unstructured methods are distinguished by their flexibility for handling complicated geometries and their ability to permit varying length scales in all three directions independently. In addition, they allow local refinements in a certain region, without affecting the grid point distribution outside that region. Their drawbacks include the higher computational costs per element and potentially lower accuracy due to skewed elements. Thus, one seeks a method that uses mostly hexahedra elements but that offers the flexibility of unstructured grids. Note, that by unstructured we only refer to the data connectivity but not the element types, hence quadrilaterals or hexahedra (the structured elements) can well occur in an unstructured grid framework.

Mesh related aspects of AVBP are handled by the multi-function grid-preprocessor hip (a package for manipulating unstructured computational grids), created and developed by J.-D. Müller [141]. Its origin dates back to 1997 when it was created at CERFACS to convert structured multiblock meshes into unstructured ones. This grid manipulation tool allows various operations such as generic solution interpolation between two grids, grid cutting or merging, grid validation, adaptive local grid refinement, grid extrusion or creation of axisymmetric grids.

2.5.1 The element-to-node pointer

From a programming point of view, handling an unstructured mesh has an increased complexity to manage the necessary data structures with respect to structured mesh data structures. In AVBP, a generalized data structure allows one to consider unstructured and hybrid meshes in two and three dimensions. In the context of cell-vertex data structures used in AVBP, both solution variables and coordinate arrays are stored at the nodes of the grid. Unlike in an structured data description, where each node can be directly addressed in an $i-j-k$ index system, the unstructured description of AVBP requires indirect addressing using pointer arrays. Each type of generalized element has a number of vertices, while pointer arrays indicate the relationship between the element and the nodes, that is, element-to-node connectivity. Gather and scatter techniques are then used throughout to pass information between elements and nodes (see subsection 2.2.1). The use of indirect addressing results in a loss of computational efficiency compared to structured grids methods. Because additional pointers are necessary for providing connectivity information, this also leads to increased memory requirements. To a large extent, these disadvantages are compensated by the requirements for parallelizing unstructured methods.

The advantage of using an element-to-node pointer, over alternative pointers such as for example a node-to-element pointer, is that, for a given cell type, the number of vertices per element is known *a priori*. This is not the case with a node-to-element pointer, since each node may be surrounded by a different number of elements. The element-to-node pointer simplifies the Fortran implementation, and allows to concentrate the computational intensive tasks in loops over elements. Indeed, this is a natural programming in the framework of cell-vertex schemes.

2.5.2 Grouping of cells

Because a large part of the work in constructing most numerical discretizations takes place at the level of the elements, AVBP disposes of a grouping strategy. The element arrays are reordered into chunks of small groups that are typically composed of 80-100 elements (depending on the computer). The number of elements is chosen to minimize memory requirements and to optimize efficient cache usage. The solver then operates on these groups of elements after gathering information from nodes and then scattering the results back. For hybrid grids, each group contains a fixed type of elements, and all groups of the same type within each partition are arranged consecutively in memory. The numerical scheme is then completely cell-type independent once the metric for the different elements is calculated. The cell grouping approach drastically reduces the memory requirements of the underlying cell-vertex technique, where quantities are transferred between nodes and cells.

Figure 2.20 shows an example of a 2D grid divided into four subdomains. The interface between subdomains is represented by a solid line and the limit between cells of type 1 and 2 is represented by a dashed line. Each subdomain contains two types of cells[10]:

- **Type 1:** cells located entirely in the interior of a parallel domain (owned cells).
- **Type 2:** cells that possess at least one vertex lying on the interface between two (or more) processors.

[10] A third group containing ghost cells located in an adjacent partition was used to overlap the communication with the computation in order to reduce the global CPU time, however, this option is not longer available.

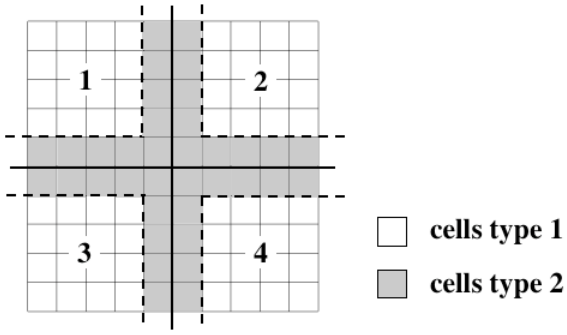


Figure 2.20 - *Example of grouping for a simple 2D grid partitioned into 4 subdomains. White cells are internal cells without interface nodes (type 1). Gray cells have at least one node on the interface between processors (type 2).*

The total number of groups is composed of the sum of this two types of cells. For example, if the number of elements of each chunk is 100, each subdomain in Fig. 2.20 contains one group with 16 elements of type 1 and one group with 9 elements of type 2, i.e., the total number of groups is 2. On the contrary, if the number of elements in the chunk is 10 (lower than the number of elements of the subdomain), each subdomain should contain two groups of type 1: with 10 and 6 elements, respectively, and one group of type 2 with 9 elements, i.e., the total number of groups is 3.

2.6 Parallel computing

Parallel computing is the simultaneous use of multiple compute resources (or processing units) to solve a computational problem. Then, a problem can (theoretically) be divided into n subproblems (where n is typically the number of available processing units), and each part of the problem is solved by one of the processing units (PU) concurrently. Ideally, the overall time of the computation will be T_{run}/n, where T_{run} is the completion time for the problem on a computer containing only one PU. The concept *scalability* is related to parallel computing since it is a measure that specifies whether or not a given problem can be solved faster as more PU are added to the computer. The analysis of a code performance done in this thesis and presented in the application chapters will use two parameters: the simulation speedup and the reference single-phase CPU time ratio (introduced to compare with two-phase flow simulations). The former is defined as the ratio between the CPU time of a simulation with n processors and the CPU time of a simulation with a given number of processors, N_{procs}:

$$Speedup = \frac{T_{run}(n)}{T_{run}(N_{procs})} * n. \tag{2.94}$$

The latter is defined as the ratio between the CPU time of a simulation with a given number of processors and the CPU time of the reference single-phase simulation with n processors:

$$CPU\ time\ ratio = \frac{T_{run}(N_{procs})}{T_{single-phase}(n)}. \tag{2.95}$$

Default value of n is 1 but sometimes it is not possible to run the serial simulation mainly due to high memory requirements, in which case both ratios are reported to a reference parallel simulation.

Two different classification of parallel computing are presented in this section. The objective is to give a general overview of the parallel strategies used in AVBP.

2.6.1 Types of parallelism

There are several different forms of parallel computing that are briefly summarised hereafter:

- **Bit-level parallelism:**
  It is a form of parallel computing based on increasing processor word size. From the advent of very-large-scale integration (VLSI) computer chip fabrication technology in the 1970s until about 1986, advancements in computer architecture were done by increasing bit-level parallelism [37]. Increasing the word size reduces the number of instructions the processor must execute in order to perform an operation on variables whose sizes are greater than the length of the word. For example, consider a case where an 8-bit processor must add two 16-bit integers. The processor must first add the 8 lower-order bits from each integer, then add the 8 higher-order bits, requiring two instructions to complete a single operation. A 16-bit processor would be able to complete the operation with single instruction.

- **Instruction-level parallelism (ILP):**
  It is a measure of how many of the operations in a computer program can be performed simultaneously. Ordinary programs are typically written under a sequential execution model where instructions execute one after the other and in the order specified by the programmer. ILP allows the compiler and the processor to overlap the execution of multiple instructions or even to change the order in which instructions are executed.

- **Task parallelism:**
  It consists of distributing execution processes across different parallel computing nodes. In a multiprocessor system, task parallelism is achieved when each processor executes a different process on the same or different data. The processes may execute the same or different code. In the general case, different execution processes communicate with one another as they work. Communication takes place usually to pass data from one process to the next as part of a workflow. Task parallelism emphasizes the distributed (parallelized) nature of the processing, as opposed to the data (data parallelism). Task parallelism does not usually scale with the size of a problem [37].

- **Data parallelism:**
  It consists of distributing the data across different parallel computing nodes. In a multiprocessor system executing a single set of instructions (SIMD) (see Table 2.4), data parallelism is achieved when each processor performs the same task on different pieces of distributed data. Data parallelism emphasizes the distributed (parallelized) nature of the data, as opposed to the processing (task parallelism). Data parallelism is parallelism inherent in program loops, which focuses on distributing the data across different computing nodes to be processed in parallel. Many scientific and engineering applications exhibit data parallelism and AVBP is an example of them.

Parallel computers can be classified by various aspects of their architecture: for example by the way the processors are connected with the memory or by the number of instruction and the number of data taken into account (Flynn's taxonomy). In the following, both classifications are briefly described.

2.6.2 Memory-processor organization

In terms of memory-processor organization three main groups of architectures can be identified:

- **Shared memory architectures (SMA):**
  It refers to a design where processors does not have its own dedicated memory, and instead shares the main system RAM (for Random Access Memory) with the CPU and other components. The main property of SMA is therefore that all processors in the system have access to the same memory since there is only one global address space (Fig. 2.21 (a)). A major advantage of shared memory computers is that programming a shared memory computer is very convenient due to the fact that all data are accessible by all processors, such that there is no need to copy data. Furthermore, the programmer does not have to care for synchronization, since this is carried out by the system automatically. However, it is very difficult to obtain high levels of parallelism with shared memory machines; most systems do not have more than 64 processors.

- **Distributed memory architectures (DMA):**
  In this case there is no common address space, i.e., the processors can access only their own memories. Communication and synchronisation between the processors is done by exchanging messages over the interconnection network (Fig. 2.21 (b)). Contrary to a SMA, a DMA scales very well, since all processors have their own local memory which means that there are no memory access conflicts. Typical representatives of a pure distributed memory architecture are the clusters of computers. In a cluster each node is a complete computer, and these computers are usually connected through a low-cost commodity network.

- **Distributed shared memory (DSM) architectures:**
  In this kind of organization, each node of a cluster has access to a large shared memory in addition to each node's limited non-shared private memory (Fig. 2.21 (c)). This third kind of architecture tries to combine the advantages of the SMA and DMA. Here, each processor has its own local memory, but, contrary to the DMA, all memory modules form one common address space, i.e., each memory cell has a system-wide unique address. In order to avoid the disadvantage of shared memory computers, namely the low scalability, each processor uses a cache, which keeps the number of memory access conflicts and the network connection low.

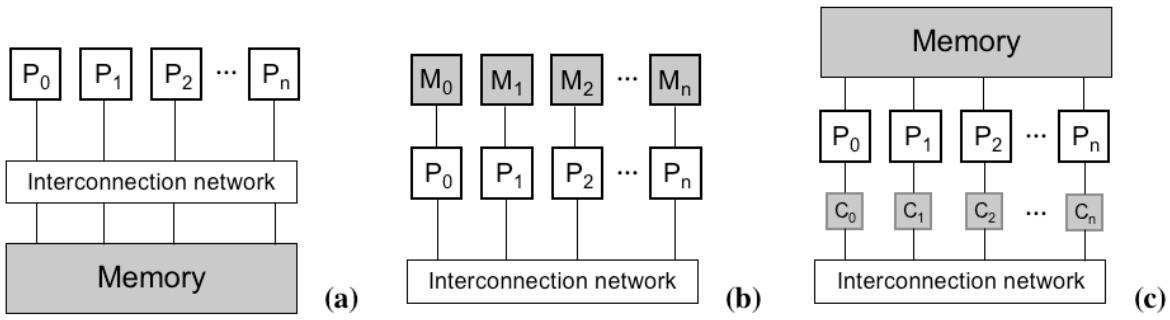


Figure 2.21 - *(a) Shared memory architecture; (b) Distributed memory architecture; (c) Distributed shared memory architecture. P = Processor, M = Memory, C = Cache.*

2.6.3 Flynn's taxonomy

Michael J. Flynn created one of the earliest classification systems for parallel (and sequential) computers and programs, now known as Flynn's taxonomy [64]. Flynn classified programs and computers by whether they were operating using a single set or multiple sets of instructions, whether or not those instructions were using a single or multiple sets of data (Table 2.4).

	Single instruction	**Multiple instruction**
Single data	SISD	SIMD
Multiple data	MISD	MIMD

Table 2.4 - *Flynn taxonomy.*

The four classifications defined by Flynn are based upon the number of concurrent instruction (or control) and data streams available in the architecture:

- **Single Instruction, Single Data (SISD):**
  A sequential computer which exploits no parallelism in either the instruction or data streams (Fig. 2.22 (a)). It is equivalent to an entirely sequential program. Examples of SISD architecture are the traditional monoprocessor machines like PC (Personal Computer).
- **Multiple Instruction, Single Data (MISD):**
  Multiple instructions operate on a single data stream (Fig. 2.22 (b)). Heterogeneous systems operate on the same data stream and must agree on the result. It is a rarely used classification.
- **Single Instruction, Multiple Data (SIMD):**
  A computer which exploits multiple data against a single instruction to perform operations which may be naturally parallelized (Fig. 2.22 (c)). It is analogous to do the same operation repeatedly over a large data set. For example, an array processor or graphics processing unit.
- **Multiple Instruction, Multiple Data (MIMD):**
  Multiple autonomous processors simultaneously executing different instructions on different data (Fig. 2.22 (d)). Distributed systems are generally recognized to be MIMD architectures; either exploiting a single shared memory space or a distributed memory space.

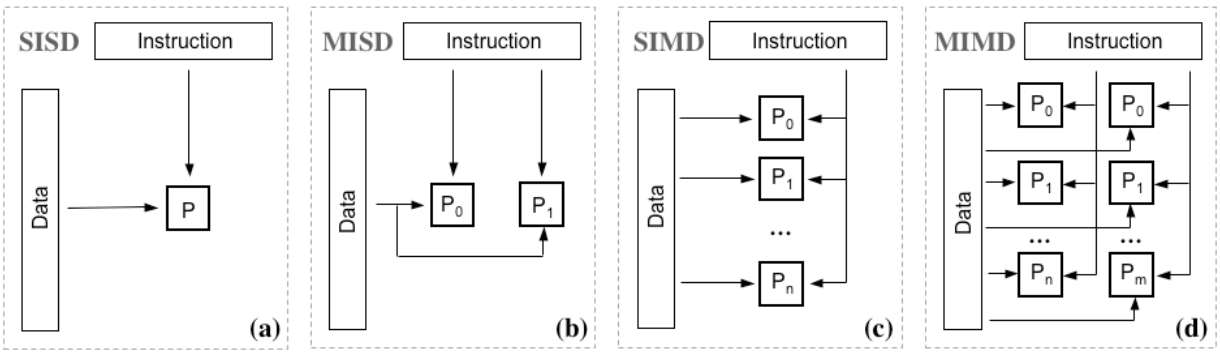


Figure 2.22 - *Flynn taxonomy representation: (a) SISD; (b) MISD; (c) SIMD; (d) MIMD.*

MIMD programs are by far the most common type of parallel programs. Some further divide this category into the following ones:

- **Single Program, Multiple Data (SPMD)**:
  Multiple autonomous processors simultaneously executing the same program (but at independent points, rather than in the lockstep that SIMD imposes) on different data. Also referred to as "Single Process, multiple data". SPMD is the most common style of parallel programming.
- **Multiple Program, Multiple Data (MPMD):**
  Multiple autonomous processors simultaneously operating at least 2 independent programs. Typically such systems pick one node to be the *host* or *master* (the "master-slave"strategy), which runs one program that farms out data to all the other nodes (*slaves*) which all run a second program. Those other nodes then return their results directly to the master for I/O.

At present in Avbp, the parallel library MPL (based on the COUPL library) allows for different parallelization strategies: (i) the master-slave paradigm and (ii) the SPMD approach previously mentioned. In the former implementation, a unique master process is responsible for the I/O and grid partitioning only, and is hidden to the user. The slaves are distributed across the parallel machine in question and perform the computations on a particular subset of the grid. Note that for some collective reductions, a slave processor must take the role of the *master of the slaves* (called *root*) since the master processor does not contain simulation data. This kind of paradigm is more adapted for current massively parallel machine calculations where the master is less load in memory since it does not calculate. In the alternative SPMD approach, which is arguably more appropriate for many parallel machines that only support single tasks per processor, the host process acts as both, the master and slave (and also the root). The only restriction is that processors memory should be enough to allow the master to work also as a slave. Through the standard MPL interface, the swap between the master-slave and SPMD modes is transparent to Avbp (and the user). The tendency with Avbp is the use of master-slave paradigm in massively parallel machines like IBM Blue Gene/L (due to their low memory capacity per processor) and the SPMD strategy for the other kind of computers.

2.6.4 Communication protocol

Message passing for parallel computations is based on the standard library MPI (for Message Passing Interface). Avbp does not use directly the syntax of the MPI library, but rather employs the Integrated Parallel Macros (IPM) developed at CERFACS [74]. There is a historical reason since initially it was also possible to use the Parallel Virtual Machine (PVM) library to perform message passing and the choice of the parallel library (MPI or PVM) was done at compilation time by using the standard Unix preprocessor M4. Then, IPM acted as an intermediate layer to easily switch between both parallel message passing libraries. These macros are still in use but only MPI communication protocol is available.

2.7 Rounding errors and reproducibility of LES

A parallel program is called nondeterministic if it computes different results in subsequent runs even if the same input data are provided. The nondeterministic character of AVBP was stated and studied in 2003 by García [67]. The origin of this "irreproducibility effect" is the arbitrary message arrival of variables to be updated at partition interfaces and the subsequent differences in the addition of the contributions of cell residuals at these boundary nodes. A solution to force a deterministic behaviour would be the use of blocking messages but this will affect the overall simulation time, mainly for simulations performed with a high number of processors and where the communication cost is not negligible. The solution proposed in [67] is focused on the reception of the overall contributions of the interface nodes (by keeping the arbitrary message arrival) and its posterior addition always in the same order. This variant is used in the validation of new versions of the code and for error detection or debugging, but it would be memory consuming in massively parallel machines and therefore it is not permanently used. However, this technique will be used in all the simulations presented in this section to study the effects of number of processors since this problem cannot be studied with a nondeterministic code.

The goal of this section is, therefore, to quantify the effects of the number of processors on large-eddy simulations with AVBP. The differences between two LES solutions produced by any pair of runs are compared in the following. Table 2.5 summarises the main characteristics of all simulations. The effects of multiple parameters (node reordering, machine precision, initial conditions, etc.) are compared to the effect of using a different number of processors with a simulation performed with four (TC1) and eight (TC2) processors with identical initial conditions. The results presented in this section are taken from an article published in AIAA Journal (see Appendix D). In that paper it was shown that instantaneous flowfields produced by LES are partially controlled by rounding errors and depends on multiple parameters. These results confirm that LES reflects the true nature of turbulence insofar as it may exponentially amplify very small perturbations on initial conditions. A laminar Poiseuille pipe flow and a fully-developed turbulent channel flow are the test cases presented next. Results of the application to a complex burner geometry and some details of the geometries used are not reported here but can be consulted in Appendix D.

Run ID	N^o **proc.**	**Initial condition**	**Precision**	**Node reordering**	**CFL**
TC1	4	Fixed	Double	CM	0.7
TC2	8	Fixed	Double	CM	0.7
TC3	1	Fixed	Double	CM	0.7
TC4	1	Modif.	Double	CM	0.7
TC5	1	Fixed	Double	RCM	0.7
TC6	4	Fixed	Double	CM	0.35
TC7	8	Fixed	Double	CM	0.35
TC8	4	Fixed	Simple	CM	0.7
TC9	8	Fixed	Simple	CM	0.7
TC10	28	Fixed	Quadr.	CM	0.7
TC11	32	Fixed	Quadr.	CM	0.7

Table 2.5 - *Summary of LES runs with the fully-developed turbulent channel (TC) flow.*

Figure 2.23 shows instantaneous axial velocity fields in the central plane of the channel at three instants after the run initialization. The simulations were performed with different numbers of processors but with identical initial conditions. At the beginning of the simulation ($t^+ = 7.68$) both flowfields observed are identical (Fig. 2.23 (a)). The differences start to become visible around $t^+ = 18.43$ (Fig. 2.23 (b)) and finally at $t^+ = 26.11$, the instantaneous flowfields are totally different (Fig. 2.23 (c)).

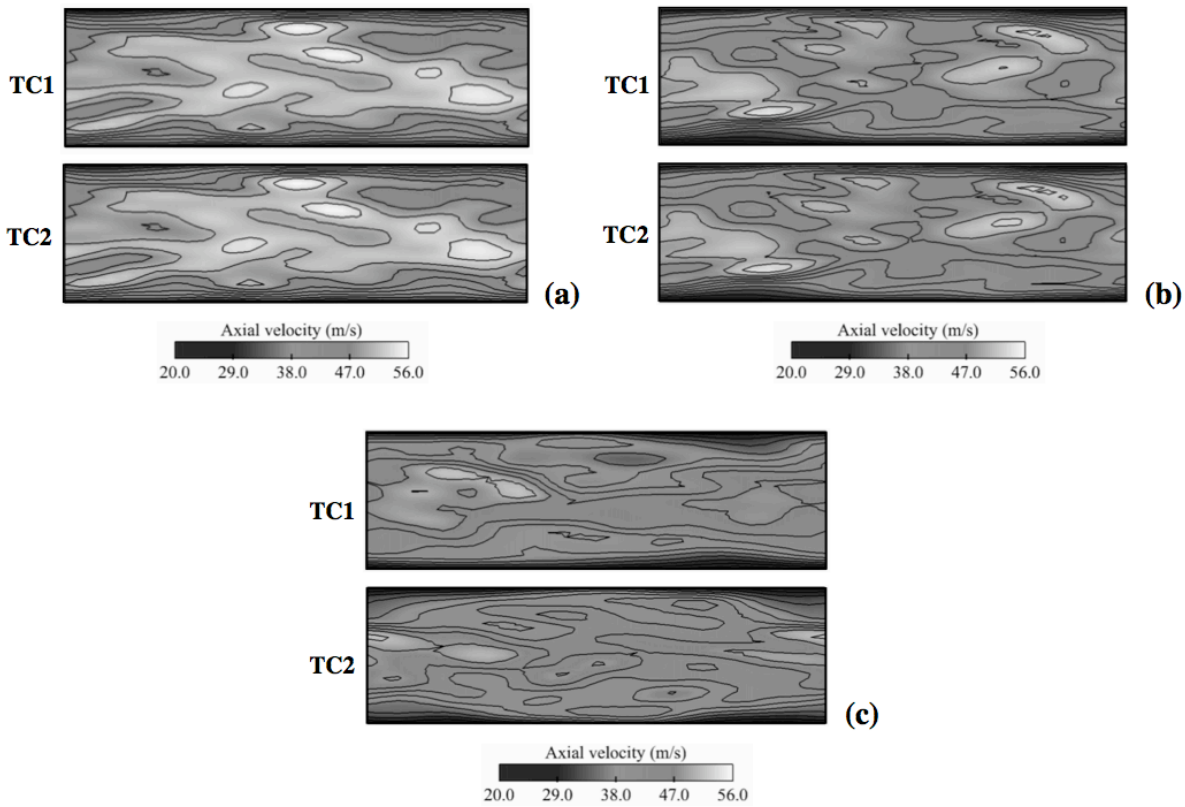


Figure 2.23 - *Instantaneous field of axial velocity in the central plane of the turbulent channel (TC) at (a)* $t^+ = 7.68$ *(b)* $t^+ = 18.43$ *(c)* $t^+ = 26.11$. *Simulations performed with four (TC1) and eight (TC2) processors. From Senoner* et al. *[198]; reprinted by permission of the American Institute of Aeronautics and Astronautics, Inc.*

There are two main reasons behind such differences as briefly explained hereafter:

- the nature of finite precision computations and,
- the true nature of turbulence.

A proper method to quantify these differences is the use of two norms. Let u_1 and u_2 be the axial velocity components of two given instantaneous solutions at the same instant after initialization. These norms are:

$$\begin{aligned} N_{max} &= \max(|u_1(\mathbf{x}) - u_2(\mathbf{x})|) && \text{for } \mathbf{x} \in \Omega \\ N_{mean} &= \left(\frac{1}{V_\Omega}\int_\Omega [u_1(\mathbf{x}) - u_2(\mathbf{x})]^2 d\Omega\right)^{\frac{1}{2}} && \text{for } \mathbf{x} \in \Omega, \end{aligned} \tag{2.96}$$

where Ω and V_Ω denote the computational domain and its volume, respectively. Both norms are expressed in m/s. There, N_{max} gives the maximum local axial velocity difference in the field between two solutions while N_{mean} yields a volume-averaged axial velocity difference between the two solutions. The growth of N_{max} and N_{mean} versus the number of time steps is used as a direct indicator of the divergence of the solutions.

Effects of node reordering in mesh

The cell-vertex scheme used in AVBP has been presented in Subsection 2.2.1. The order used to add residuals at nodes (see Eq. (2.53)) is fixed in a deterministic computation, but what happens if this order is changed? This is briefly analyzed next. The order of this addition can be indirectly controlled by the user. One way is to change the number of cells per group although this parameter mainly concerns parallel computations. Another simple way that can be used in a sequential computation is to use different reordering techniques as presented in Subsection 2.4.5. By changing the ordering of this addition, the residual at a given node is assembled by adding the contributions to a cell residual in a different order but without affecting the flow data. Figure 2.24 shows an example of reordering in the 2D grid introduced in Figs. 2.16-2.17. Note that cells are always spanned in increasing order (from 1 to 4) but the reordering algorithms changes the starting cell and span direction and, as consequence, residuals are not added in the same order (see gray scale and A-D coding).

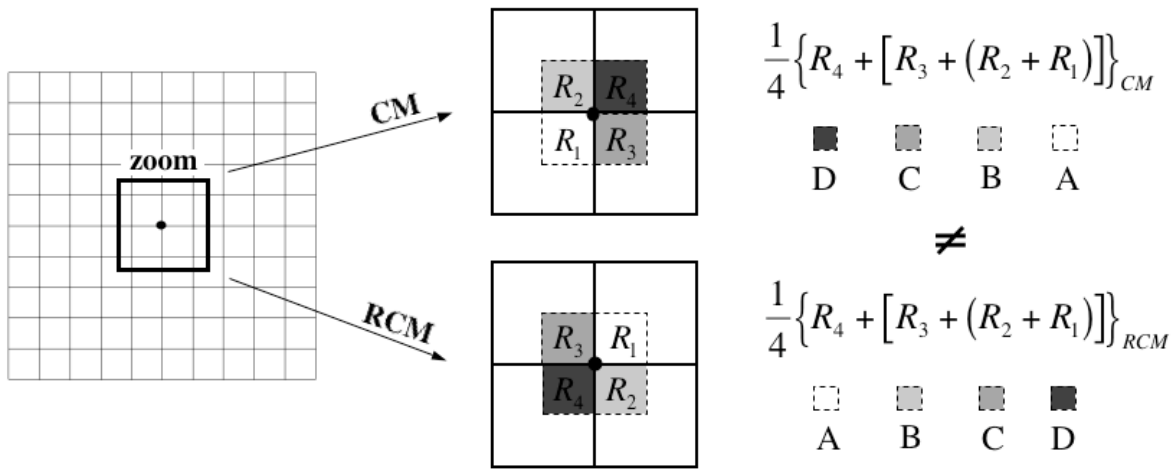


Figure 2.24 - *Example of different node reordering (CM vs RCM) for the central node of a simple 2D grid.*

This node reordering highlights a well-known problem in finite arithmetic: operations do not follow associativity rules as in exact arithmetic and a simple addition of more than two terms like $C+(B+A)$ may yield distinct results than $A+(B+C)$ because of rounding errors [228, 209, 169]. In particular, larger differences in the order of magnitude between the summands [78] may affect LES results after a few tens of thousands of time steps.

Results of the effects of node reordering are shown in Fig. 2.25. Test cases TC5 and TC3 were performed in a single processor and differ only for the reordering technique. These results confirm that rounding errors (and not the parallel character of the code) are the source of the divergence observed between two solutions.

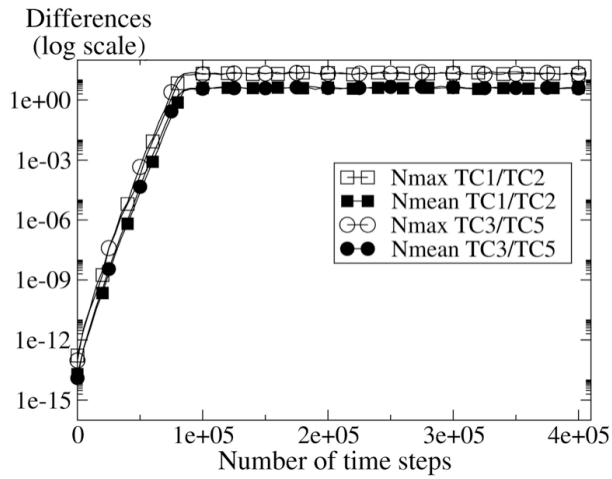


Figure 2.25 - *Differences between solutions vs time step. Squares: different number of processors. Circles: different addition order. From Senoner* et al. *[198]; reprinted by permission of the American Institute of Aeronautics and Astronautics, Inc.*

2.7.1 Finite precision computation

In the next paragraphs we recall some basic notions (following Overton [146]) of finite arithmetic and computer representation of numbers. This knowledge is important to better understand previous results and some specific issues occurring in simulation on parallel computing.

The idea of representing numbers using power of 10 was used by many ancient peoples, e.g., the Hebrews, the Greeks, the Romans and the Chinese. The reason for the decimal choice is the simple biological fact that humans have 10 fingers[11]. The system used worldwide today is the decimal, or base 10, which requires the representation of the numbers from 0 to 9. This notation is called *positional* because the meaning of the number is understood from the position of the symbols (or digits) of the numbers. Zero is needed, for example, to distinguish 601 from 61. Although, decimal representation is convenient for people, it is not particularly convenient for computers. The binary, or base 2, system is much more useful: in this, every number is represented as a string of *bits*[12], each of which is either 0 or 1. Each bit corresponds to a different power of 2, just as each digit of a decimal number corresponds to a different power of 10. Although the binary system was not widely used before the computer age, the idea of representing numbers as sums of powers of 2 is far from new. It was used as the basis for a multiplication algorithm described in the *Rhind Mathematical Papyrus* [27], written nearly four millennia ago. An example of decimal (Eq. (2.97)) and binary (Eq. (2.98)) representation of the integer 71 is included here:

$$(71)_{10} = \mathbf{7} \times 10^1 + \mathbf{1} \times 10^0 \tag{2.97}$$

$$(1000111)_2 = \mathbf{1} \times 2^6 + \mathbf{0} \times 2^5 + \mathbf{0} \times 2^4 + \mathbf{0} \times 2^3 + \mathbf{1} \times 2^2 + \mathbf{1} \times 2^1 + \mathbf{1} \times 2^0 \tag{2.98}$$

The word *expansion* is sometimes used instead of representation so we can say that both of these expansions are finite. The representation of integers is straightforward but the expansion of real numbers is not so simple since it may not be finite. To better understand this difference, an

[11] The word *digit* derives from the Latin word for finger.
[12] The word *bit* is an abbreviation for *binary digit*.

example of the expansion of two real numbers is presented hereafter:

$$\frac{11}{2} = (101.1)_2 = \mathbf{1} \times 2^2 + \mathbf{0} \times 2^1 + \mathbf{1} \times 2^0 + \mathbf{1} \times 2^{-1} \tag{2.99}$$

$$\begin{aligned} \frac{1}{10} &= (0.0001100110011...)_2 = \\ &= \mathbf{0} \times 2^0 + \mathbf{0} \times 2^{-1} + \mathbf{0} \times 2^{-2} + \mathbf{0} \times 2^{-3} + \mathbf{1} \times 2^{-4} + \mathbf{1} \times 2^{-5} + \mathbf{0} \times 2^{-6} + \mathbf{0} \times 2^{-7} + \\ &\quad \mathbf{1} \times 2^{-8} + \mathbf{1} \times 2^{-9} + \mathbf{0} \times 2^{-10} + \mathbf{0} \times 2^{-11} + \mathbf{1} \times 2^{-12} + \mathbf{1} \times 2^{-13}... \end{aligned} \tag{2.100}$$

The first real number (Eq. (2.99)) can be represented with a finite expansion. On the contrary, the binary representation of the second one (Eq. (2.100)) is not finite and the computer should approximate this number by the closest one that it is able to handle. A real number is called a *floating point number* if it can be stored *exactly* on the computer given the floating point representation scheme. Its representation on the computer is done with the exponential (or scientific) notation:

$$x = \pm S \times 2^E, \qquad \text{where} \qquad 1 \leq S < 2. \tag{2.101}$$

The numbers S and E are called the *significand* and the *exponent*. The binary expansion of the significand is written as:

$$S = (b_0.b_1b_2b_3...) \qquad \text{with} \qquad b_0 = 1. \tag{2.102}$$

Applying this to Eq. (2.99) yields:

$$\frac{11}{2} = (1.011)_2 \times 2^2. \tag{2.103}$$

If a number x is not a floating point number, it must be **rounded** before it can be stored on the computer and used in any other operation. The number of bits in the significand that can be stored for each floating point number is called *precision*. The gap between 1 and the next larger floating point number is called *machine epsilon*, $\epsilon = 2^{-(p-1)}$. The IEEE[13] standard for binary floating-point arithmetic (ANSI/IEEE Std 754-1985 [91]) specifies four formats for representing floating-point values: single-precision, double-precision, single-extended precision (not commonly used) and double-extended precision (usually implemented with 80 bits), see Table 2.6.

Format	**Bits**	**Precision**	**Machine epsilon**	
Single	32-bit	$p = 24$	$\epsilon = 2^{-23}$	$\approx 1.2 \times 10^{-7}$
Single-extended	≥43-bit	$p \geq 32$	$\epsilon = 2^{\geq -31}$	$\approx 4.7 \times 10^{-10}$
Double	64-bit	$p = 53$	$\epsilon = 2^{-52}$	$\approx 2.2 \times 10^{-16}$
Double-extended	≥79-bit	$p \geq 64$	$\epsilon = 2^{\geq -63}$	$\approx 1.1 \times 10^{-19}$
Quadruple	128-bit	$p = 113$	$\epsilon = 2^{-112}$	$\approx 1.9 \times 10^{-34}$
Quadruple-extended	80-bit	$p = 64$	$\epsilon = 2^{-64}$	$\approx 5.4 \times 10^{-20}$

Table 2.6 - *Floating-point formats characteristics.*

[13] For Institute of Electrical and Electronics Engineers.

Although the *quadruple-precision* name and the particular parameters of its format are not specified in the IEEE-754 standard, it is a legally derived IEEE-754 format because its parameters are specific subset elements within the bounds of those specified for the *extended double-precision* format. Like the *quadruple-precision* format, Intel's *extended-precision* format is a legal IEEE-754 format derived from the *extended double-precision* format.

Effects of machine precision

The effects of machine precision on AVBP computations have also been investigated and results are detailed in Appendix D: *Effects of machine precision.* Figure 2.26 shows that solution differences for single, double and quadruple precision runs start from the respective machine accuracies and increase exponentially with the same growth rate, before reaching identical difference levels for all three machine precisions. These results show that higher precision computations cannot prevent the exponential divergence of trajectories but only delay it.

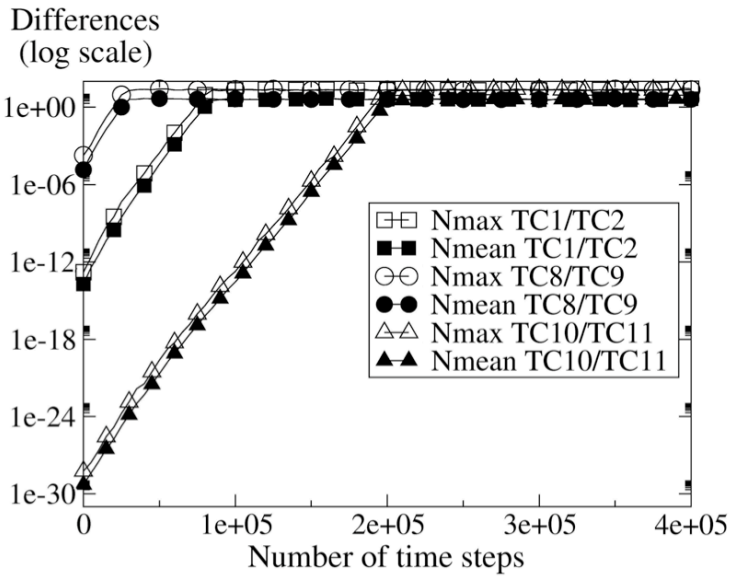


Figure 2.26 - *Differences between solutions vs time step. Squares: double precision. Circles: simple precision. Triangles: quadruple precision. From Senoner* et al. *[198]; reprinted by permission of the American Institute of Aeronautics and Astronautics, Inc.*

In double precision, Eq. (2.103) would be stored by the computer as:

0	10000000001	0110000000000000000000000000000000000000000000000000

with 1 bit used to store the sign of the number, 11 bits for the exponent and 52 bits for the significand.

$\pm$	$a_1a_2a_3...a_{11}$	$b_1b_2b_3...b_{52}$

Every floating-point number is a real number, but few real numbers have floating-point equivalents. Consequently, floating-point operations ($+,-,\times,/$) are generally thought of as being composed of the corresponding real operation followed by a rounding step which chooses a floating-point

number to approximate the real result ($\oplus$,$\ominus$,$\otimes$,$\oslash$). The representation of integer and real numbers in a line will help us to understand what happens. Figure 2.27 presents a line where every point corresponds to a number, integer and reals, but only a few are marked. Making a zoom through this line up to machine precision, we can observe the limit between the representation of two floating-point numbers (Fig. 2.28).

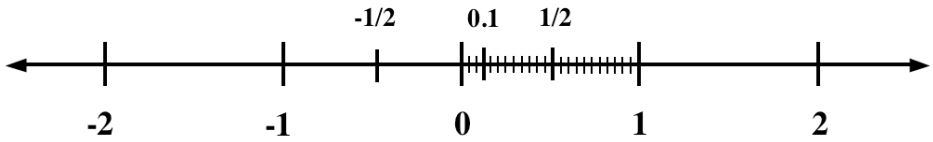


Figure 2.27 - *Numbers represented in a line.*

Let A, B and C be three floating-point numbers in double precision. Let B and C be two consecutive floating-point numbers.

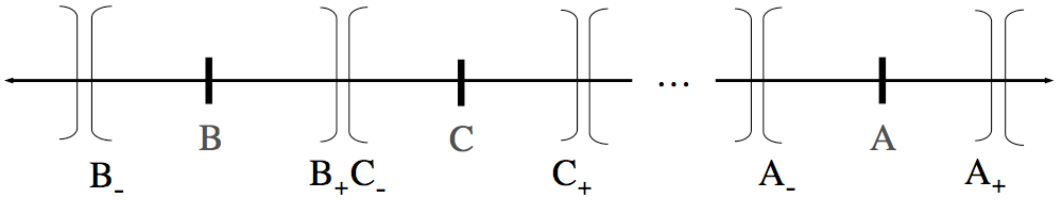


Figure 2.28 - *Zoom up to finite precision in the number representation line.*

The decimal representation of these three numbers is:

$$\begin{aligned}
(A)_{10} &= 0.5431661834275266359517431169479095843 \times 10^{-1} \\
(B)_{10} &= 0.5429813767367863\mathbf{0032991776488415780 48} \times 10^{-1} \\
&\qquad\qquad\qquad\qquad\qquad\qquad \Big\downarrow \textit{Next floating-point number} \\
(C)_{10} &= 0.5429813767367863\mathbf{6971885680395644158 12} \times 10^{-1}
\end{aligned} \tag{2.104}$$

Their binary expansion:

$$\begin{aligned}
(A)_2 &= (1.10111100111101100011010001101101001010100111110010 11) \times 2^{-5} \\
(B)_2 &= (1.10111100110011110111001010110001010110010101010\mathbf{111 11}) \times 2^{-5} \\
&\qquad\qquad\qquad\qquad\qquad\qquad \Big\downarrow \textit{Next binary} \\
(C)_2 &= (1.10111100110011110111001010110001010110010101010\mathbf{100 00}) \times 2^{-5}
\end{aligned} \tag{2.105}$$

The difference between B and C in double precision is of the order of 0.7×10^{-17} ($\approx \epsilon/2^{-5}$) on a Compaq AlphaServer SC. This means that a small perturbation, slightly higher than this value could have some effect in the representation of two numbers of this order of magnitude. An example is presented hereafter. The upper limit of B (B_+) and the lower limit of C (C_-) are represented in base 10.

$$(B_+)_{10} = 0.5429813767367863350208904131948576767 \times 10^{-1} \tag{2.106}$$

$$(C_-)_{10} = 0.5429813767367863350208904131948576768 \times 10^{-1} \tag{2.107}$$

The difference between both limits is in the last digit. Let us assume that rounding is done with the default rounding mode *rounding to nearest*, i.e., we do not consider other rounding types like *round down*, *round up* or *round towards zero*. Then, the result of an arithmetic operation on two floating-point numbers greater than B and smaller than B_+ will produce B as the floating-point number in the destination format. On the contrary, if the result of this operation is lower than C and higher than C_-, the rounding operation will produce C as the floating-point number in the destination format.

A direct consequence of this is the lack of associativity presented in Fig. 2.24 since small differences between the addition of $A+B$ and $A+C$ could produce a different result with the addition of a new summand. Then, any small perturbation of a given reference state may exhibit a separation of trajectories for LES. These results are presented next.

Effects of initial conditions

The effects of a small perturbation on initial conditions lead to different instantaneous results after a certain time with an effect similar to domain partitioning. These results are detailed in Appendix D (*Influence of initial conditions*) and presented in Fig. 2.29. The only difference between TC3 and TC4 is that, in TC4, the initial solution is identical to TC3, except for one random point where a single 10^{-16} m/s perturbation is applied to the streamwise velocity component.

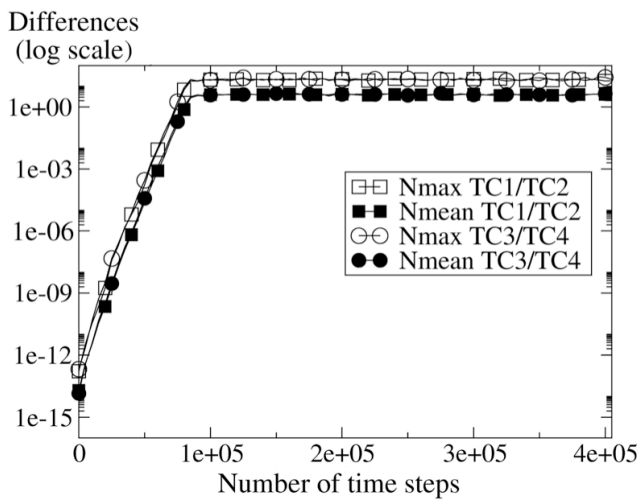


Figure 2.29 - *Differences between solutions vs time step. Squares: different number of processors. Circles: different initial conditions. From Senoner* et al. *[198]; reprinted by permission of the American Institute of Aeronautics and Astronautics, Inc.*

2.7.2 True nature of turbulence

We have seen in the previous section that the turbulent channel flow simulations are very sensitive to a change in the number of processors, the node reordering, the machine precision and the initial conditions too, showing a rapid divergence of the solutions. But, do the rounding errors affect so much to LES simulations or is there something more? The following section tries to answer to this question by quantifying the influence of turbulence.

Influence of turbulence

A mandatory test is the comparison of previous turbulent channel solutions with a laminar flow. Run parameters of the laminar Poiseuille flow are presented in Table 2.7.

Run ID	N^o proc.	Initial condition	Precision	Node reordering	CFL
LP1	4	Fixed	Double	CM	0.7
LP2	8	Fixed	Double	CM	0.7

Table 2.7 - *Summary of laminar runs for the Poiseuille flow (LP).*

Figure 2.30 shows the evolutions of N_{max} and N_{mean} vs time step for runs TC1/TC2 and LP1/LP2. It can be observed that results for the turbulent channel diverge rapidly. The initial exponential divergence observed for TC1/TC2 is typical of chaotic systems [93], while the later saturation happens when the separations are large enough to consider both solutions uncorrelated from one another. The consequence of this divergence is that the influence of the initial conditions is lost very quickly. On the other hand, the difference between LP1 and LP2 hardly increases and levels off when reaching values on the order of 10^{-12} m/s.

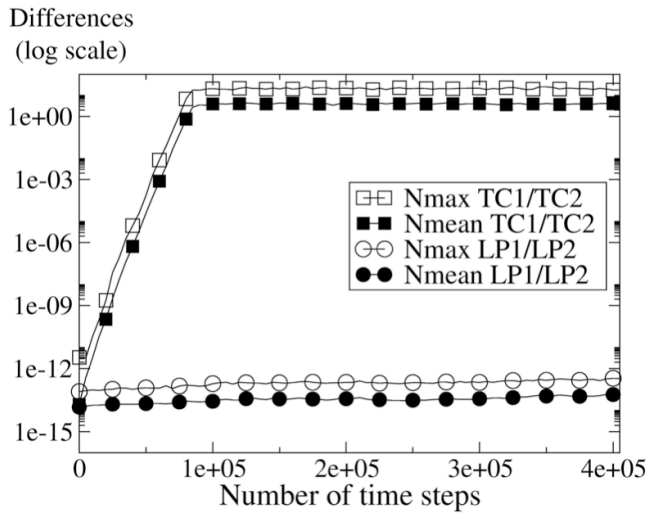
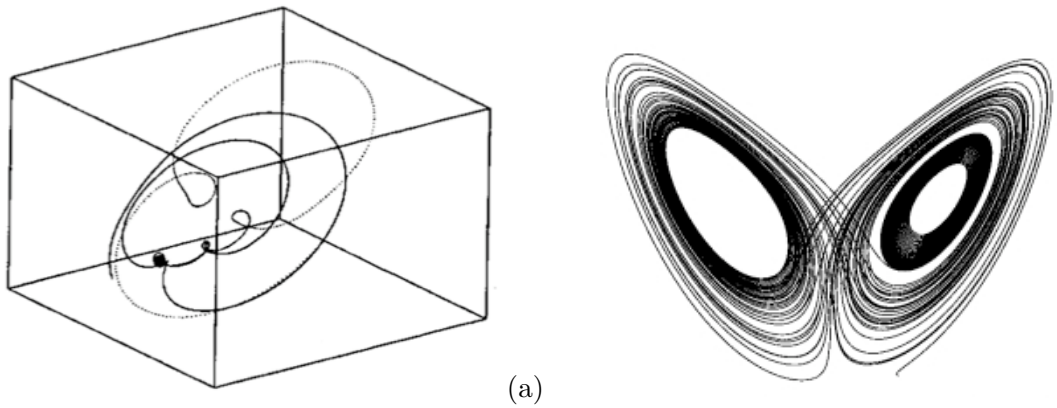


Figure 2.30 - *Differences between solutions vs time step. Squares: turbulent channel flow. Circles: laminar Poiseuille flow. From Senoner* et al. *[198]; reprinted by permission of the American Institute of Aeronautics and Astronautics, Inc.*

This behaviour is expected because there is only one stable solution for the Poiseuille flow for infinite times and, accordingly, laminar flows do not induce exponential divergence of trajectories. On the contrary, the turbulent flow acts as an amplifier for rounding errors generated by the fact that the mesh is decomposed differently in TC1 and TC2. The source of this difference is the new node reordering obtained for both decompositions. As mentioned in previous sections, this implies a different ordering when adding the contributions to a cell residual for nodes inside the subdomains, but mainly at partition interfaces. This perturbation roughly starts at machine accuracy at a few points in the flow and grows continuously if the flow is turbulent. This simple test case confirms that the turbulent character of the flow is the source of the divergence of solutions.

The growth rate of the differences between solutions in simulations TC1 and TC2 cannot be estimated in a simple manner. A description for the determination of growth rates of trajectory separation in two-dimensional vortical flows is given by Leith [113] and is briefly summarised in the following. A description of vortices as points with associated circulations and negligible viscosity is assumed. Under these hypotheses, a set of linearized ordinary differential equations can be derived to evaluate the time evolution of the distance between two neighbouring flowfield trajectories differing by an arbitrary infinitesimal perturbation $\delta(t)$ (see Fig. 2.31 (a)). This system admits exponential solutions, the growth rates of which are determined by the real part of the eigenvalues. The evolution of inviscid/conservative systems conserves volume in phase space. As the real part of the eigenvalues describes the separation of trajectories in time, it represents a measure of the evolution of the volume in phase space. Thus, the sum of the real parts vanishes and at least one of them has to be positive. At this stage, the number of degrees of freedom of the system imposes topological constraints on the trajectories and can prevent their separation, but a few degrees of freedom suffice for such systems to exhibit chaotic behaviour, as demonstrated by the famous Lorenz attractor[14] [122, 123] (see Fig. 2.31 (b)).



(a) (b)

Figure 2.31 - *(a) Two trajectories starting close to each other at the points marked with circles. Since there are no restrictions on lines crossing each other when the dimension of the embedding space is at least three, trajectories with three or more degrees of freedom can intertwine and diverge exponentially, allowing chaotic behaviour (from [93]). (b) A plot of the Lorenz attractor for values* $\rho = 28$, $\sigma = 10$ *and* $\beta = 8/3$.

When considering dissipative two- or three-dimensional turbulent flows, a well-defined phase space does not exist. Therefore, predictability analysis is based on the evolution of the energy spectrum of two realizations of a given velocity field differing by a perturbation $\delta(\mathbf{u})$ but having the same statistics. It is possible to derive equations for the evolution of the error energy spectrum and define predictability times using simulations of decaying turbulence [131]. Therefore, a simple estimate of the growth rate from flow parameters a priori does not seem possible. However, one may suppose that, independently of the spatial distribution and amplitude of perturbations applied to a given turbulent flowfield, the separation of trajectories for various simulations yields similar exponential growth rates, as confirmed in previous tests (see Figs. 2.25, 2.26 and 2.29). Moreover, it is a purely physical phenomenon and, though induced by rounding errors, the growth rate does not depend on numerical parameters such as machine precision.

[14] Named for Edward N. Lorenz, is a 3-dimensional structure corresponding to the long-term behaviour of a chaotic flow, noted for its butterfly shape. The equations that govern the Lorenz attractor are: $dx/dt = \sigma(x - y)$; $dy/dt = x(\rho - z) - y$ and $dz/dt = xy - \beta z$.

2.8 Summary and conclusions

This chapter have made an overall description of the main features of the CFD solver used during this thesis: AVBP. A summary of the more important characteristics and conclusions is presented below:

- Section 2.1 describes the governing equations for LES used in AVBP to resolve the set of compressible Navier-Stokes equations for non-reacting flows. The models available in AVBP for the subgrid stress tensor are also included in this section.

- The cell-vertex method used for the discretization of the governing equations is presented in Section 2.2. The two more used numerical schemes (Lax-Wendroff and TTGC) with AVBP are described in detailed in this section since they will be used in the test cases presented in this thesis.

- Section 2.3 briefly described the importance of boundary conditions in AVBP. No mention of all the boundary conditions available in the code is done.

- Section 2.4 described the partitioning strategy used in the AVBP code. The concept of static load balancing is introduced together with the description of the three initial partitioning algorithms available (RCB, RIB and RGB). An analysis of performance of these algorithms is done through three different grid sizes in order to study their behaviour while increasing the number of partition subdomains. The main objective being the detection of possible problems (e.g., high CPU time requirements) related to these algorithms. To this concern, the RGB partitioning algorithm seems to be the worst choice for calculation in massively parallel machines due to its increasing exponential CPU time requirements.

  Any of the three mentioned algorithms deal with multi-constraint partitioning options. With Lagrangian calculations in perspective and for calculations in massively parallel machines, the choice of a new partitioning algorithm was necessary. The new partitioning algorithm implemented in AVBP is the multilevel $k-way$ partitioning described in [96]. This algorithm is part of the software package METIS and the comparative study with the current partitioning algorithms has shown an important reduction of the CPU time used to partition the test grid and a reduction in the number of duplicated nodes resulting for the partition. Results of this section points out the influence and the importance of a good choice of partition algorithms as a way to reduce CPU time and memory requirements of a simulation.

  This section also addressed another issue of partitioning that is data reordering techniques. An example of node reordering was presented with a simple 2D grid and its influence in the CPU time of the simulation was demonstrated with a 3.2 million cells grid. An additional role played by node reordering in LES simulations is shown in Section 2.7.

- The flow solver AVBP is based on the unstructured and hybrid grid technology. Unstructured methods are distinguished by their flexibility for handling complicated geometries. Section 2.5 presented the type of elements that can be used for grid generation. The element-to-node pointer is introduced. This array is used for indirect addressing and indicates the relationship between the element (or cells) and the nodes (or vertices). This pointer simplifies the implementation and allows to concentrate the computational intensive tasks in loops over elements.

Because a large part of the work in constructing most numerical discretizations takes place at the level of the elements, AVBP disposes of a grouping strategy. The element arrays are reordered into chunks of small groups which are chosen to minimize memory requirements and to optimize efficient cache usage. The two types of groups were also presented in this section.

The knowledge of basic data structure of AVBP is necessary for the comprehension of the strategy used to develop the data structures of the Lagrangian module.

- Section 2.6 presented the main characteristics of the parallel strategies available in AVBP. The master-slave paradigm and the SPMD approach were detailed together with the communication protocol used. The concepts of scalability, speedup and CPU time ratio were also introduced. These notions will be used in the chapters dedicated to the validation of the Lagrangian module.

- In Section 2.7 we remembered that the key property of chaotic systems is their sensitivity to initial conditions. Two trajectories with initial conditions which differ only by a small amount diverge exponentially, and soon become completely decorrelated. It is as if the trajectories, free from the need of remaining close to each other, close to diverge in arbitrary ways. Any sufficiently turbulent flow computed in LES exhibits significant sensitivity to small perturbations, leading to instantaneous solutions which can be totally different. On the contrary, laminar flows are almost insensitive to these parameters even for periodic simulations.

  The divergence of solutions observed is due to two combined facts: (1) the exponential separation of trajectories in turbulent flows, and (2) the different propagation of rounding errors induced by domain partitioning and scheduling of operations.

  The sensitivity of instantaneous large-eddy simulation fields to different parameters has been studied in this thesis (see Appendix D). Rounding error propagation is activated due to the modification of various parameters of the run, such as number of processors, initial condition or changes in addition ordering of cell residuals for cell-vertex methods. Working with higher precision in finite arithmetic does not suppress the divergence of solutions, but delays it.

  A direct consequence of these divergences is that any change in the code lines affecting the propagation of rounding errors will have a similar effect. This implies that the validation of an LES code after modifications may only be based on statistical fields. Therefore, comparing instantaneous solutions is not a proper validation method for LES.

Chapter 3

Numerical implementation of the Lagrangian module

This chapter presents the main features of the Lagrangian module implementation in the AVBP solver for the treatment of the dispersed phase. Section 3.1 presents the governing equations of particle motion. The general model of the forces acting on the particle equation of motion are first recalled, then the governing equations retained for this study are given. An efficient implementation of these equations in the code depends on the data structure considered. Section 3.2 presents briefly the basic data structure used in the Lagrangian module. Lagrangian variables are stored separately in two arrays of integer and real type. The pointers used to locate particles crossing interfaces between processors are also described. Particles must be tracked at each time step inside the grid to advance them. It is known that the elements of unstructured grids can be of arbitrary shape. Section 3.3 presents the criteria chosen to locate particles in the elements available in AVBP. The different search algorithms used to detect a set of cells where the localisation criterion will be checked are summarised in Section 3.4. The different situations in which particles need to be located are also discussed in this section. Once particles are located inside a cell, they collect information from the fluid before they change position. The interpolation algorithms implemented to calculate the fluid properties at the particle position are presented in Section 3.5. In some cases (e.g., two-way coupling), particles are also supposed to exchange information with the gaseous or carrier phase. The coupling method between the gaseous and the dispersed phase is presented and validated in Section 3.6. To conclude, a model for injecting particles at a point source or a disk is described in Section 3.7.

3.1 Particle equations of motion

The description of the interaction forces between the carrier phase and an isolated particle is a well known problem that has been extensively studied in the literature. A detailed overview of these forces can be found in the following books [32, 35]. Particle trajectories are given by:

$$\frac{dx_{p,i}}{dt} = u_{p,i} \tag{3.1}$$

$$m_p \frac{du_{p,i}}{dt} = \sum F_{p,i} \tag{3.2}$$

with $x_{p,i}$, $u_{p,i}$ the particle position and velocity components, m_p the mass of the particle and $\sum F_{p,i}$ the set of forces acting on the particle. The forces acting on a single particle are briefly recalled below. The particle equation of motion retained for this study is presented at the end of the section.

3.1.1 Drag force

First works on drag forces date back to 19th century. A fundamental relation for the drag force related to a sphere was derived by Stokes [211]. It assumes that the Reynolds number of the particle, given by the ratio of diameter and the relative velocity to dynamic viscosity, $R_p = d_p\,|u_{g,i} - u_{p,i}|/\nu_g$ is much smaller than one ($R_p \ll 1$). Assuming "steady-state", the drag force which acts on the particle or droplet in a uniform pressure field when there is no acceleration of the relative velocity between the particle and the conveying fluid is:

$$F_{D,i} = \frac{1}{2}\,\rho_g\,C_D\,A\,|u_{g,i} - u_{p,i}|\,(u_{g,i} - u_{p,i}), \tag{3.3}$$

with C_D the drag coefficient, A the representative area of the droplet and $u_{g,i}$ and $u_{p,i}$ the velocities of the gaseous phase and the droplet or particle, respectively. In general, the drag coefficient will depend on the particle shape and orientation with respect to the flow as well as on the flow parameters such as Reynolds number, Mach number, turbulence level and so on. The classic Stokes drag coefficient is:

$$C_D = \frac{24}{Re_p}. \tag{3.4}$$

First studies were focused on solid and isolated spheres without evaporation. The studies by Oseen [145] resulted in the extension of the Stokes drag coefficient to include first-order inertial effects. This expression is often called "Oseen's drag coefficient":

$$C_D = \frac{24}{Re_p}(1 + \frac{3}{16}Re_p^{0.687}), \tag{3.5}$$

which is valid up to Reynolds number of 5. A decade later, Oseen's student, Faxen [57] studied the flow of spheres close to solid boundaries and extended the theory of the transient flow around a sphere to non-uniform flows. His work resulted in the introduction of new terms in the expression of the hydrodynamic force, which are now called the "Faxen terms".

One of the most used expressions, even nowadays, is the empirical correlation from Schiller & Naumann [191]:

$$C_D = \frac{24}{Re_p}(1 + 0.15Re_p^{0.687}) \qquad \text{for} \quad Re_p < 1000, \tag{3.6}$$

which is a correction of the Stokes expression.

Other authors have proposed similar correlations:

$$\text{Clift [32]:} \quad C_D = \frac{24}{Re_p}(1 + 0.1935 Re_p^{0.6305}) \qquad Re_p \leqslant 1000 \tag{3.7}$$

$$\text{Rudinger [32]:} \quad C_D = \frac{24}{Re_p}(1 + \frac{1}{6} Re_p^{\frac{2}{3}}) \qquad Re_p \leqslant 1000 \tag{3.8}$$

$$\text{White [32]:} \quad C_D = \frac{24}{Re_p} + (\frac{6}{1 + Re_p^{0.5}}) + 0.4 \qquad Re_p \leqslant 200 \tag{3.9}$$

$$\text{Mulholland [140]:} \quad C_D = \frac{24}{Re_p}(1 + 0.11 Re_p^{0.687}) \qquad 25 \leqslant Re_p \leqslant 300 \tag{3.10}$$

$$\text{Virepinte [221]:} \quad C_D = \frac{24}{Re_p}(1 + 0.12 Re_p^{0.687}) \qquad 20 \leqslant Re_p \leqslant 120. \tag{3.11}$$

All these correlations have a quite similar behaviour (Fig. 3.1). Note that the expressions proposed by Mulholland *et al.* [140] and Virepinte [221] have been obtained experimentally for droplets. The value of the drag coefficient for a droplet is lower than for a solid particle since droplets are affected by internal circulation and deformation. These phenomena modify their behaviour as displayed in Fig. 3.1 and therefore, the corresponding correlation should be considered as dependent on their physical nature. The effect of droplet deformation on drag has been quantified by Helenbrook & Edwards [84] who performed detailed simulations of axisymmetric liquid drops in a uniform gaseous stream. Even if the Schiller & Naumann [191] correlation is the most used, the effects produced by internal circulation and droplet deformation are already considered in Lagrangian simulations of realistic gas turbine combustors [133].

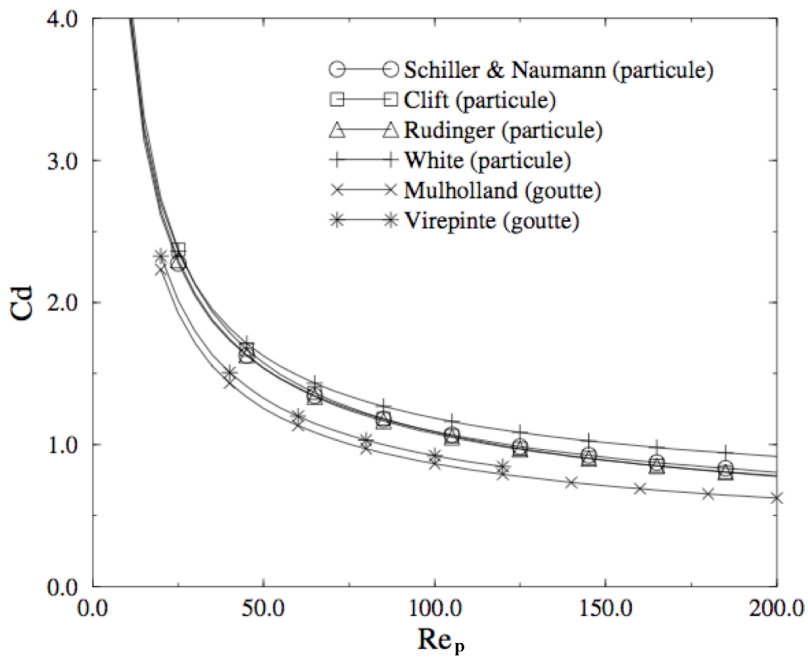


Figure 3.1 - *Comparison of different correlations of the drag coefficient for a solid (particule) or liquid (goutte) isolated particle. From Massol [127].*

Previous correlations and others [32, 35] are applied in the case of isolated particles and therefore they do not take into account the modification of the drag coefficient due to the surrounding particles, when the number of particles is high, the carrier phase flow is also altered by the particle presence. How much the carrier phase flow is affected depends on volume, mass loading and on the particle diameters. In these conditions, a more precise analysis is necessary. A complete study of the flow behaviour of regular arrays of fixed particles can be consulted in Massol [127].

The correlation of the drag coefficient used in this thesis is Eq. (3.6) since all the application test cases are done with solid particles (see Chapters 4 and 5). The introduction of other expressions more adapted to droplets behaviour is straightforward and it is left to future developments concerning turbulent combustion with evaporating droplets.

3.1.2 Other forces

The drag force is the most important one in sprays but particle trajectories can also be affected by other forces that are briefly summarised hereafter.

Added mass

When a body is accelerated through a fluid, there is a corresponding acceleration of the fluid which is at the expense of work done by the body. This additional work relates to the virtual mass effect. Therefore, the added or virtual mass force is a result of the fluid surrounding a particle being accelerated. It has a tendency to keep the particle from being accelerated in any direction. For a sphere, it can be expressed in the following form:

$$F_{am,i} = \frac{\rho_g V_p}{2} \left(\frac{Du_{g,i}}{Dt} - \frac{du_{p,i}}{dt} \right), \tag{3.12}$$

where V_p is the particle volume and D/Dt is the material derivative of the gas velocity.

All bodies accelerating in a fluid are affected by added mass, but since the added mass depends on the density of the fluid, the effect is often neglected for dense bodies falling in much less dense fluids. For situations where the density of the fluid is comparable to or greater than the density of the body, the added mass can often be greater than the mass of the body and neglecting it can introduce significant errors in the simulation.

In this thesis it is assumed that the particle or droplet density is substantially larger than the carrier phase flow ($\rho_p/\rho_g \sim 10^3$), and therefore, this force is neglected.

Pressure gradient and buoyancy force

The effect of the local pressure gradient gives rise to a force in the direction of the pressure gradient. By assuming constant pressure gradient over the volume of the particle one has:

$$F_{pres,i} = -\nabla p V_p. \tag{3.13}$$

The pressure gradient produced by hydrostatic pressure is:

$$\nabla p = -\rho_g g \vec{e}_z, \tag{3.14}$$

where z is the direction opposed to gravity, g. The corresponding pressure force is

$$F_{pres} = \rho_g g V_p, \tag{3.15}$$

which states that the force is equal to the weight of the fluid displaced. This is known as Archimedes principle. The buoyancy force for the particle equation of motion can then be written as:

$$F_{b,i} = V_p(\rho_p - \rho_g) g_i. \tag{3.16}$$

Basset force

If the particle has a changing relative velocity, additional forces should be considered. One of them is related to the fluid viscosity and is called Basset or history force. The Basset force term describes the force due to the lagging boundary layer development with changing relative velocity (acceleration) of bodies moving through a fluid [35]. The Basset term accounts for viscous effects and addresses the temporal delay in boundary layer development as the relative velocity changes with time. This force is difficult to implement and is commonly neglected for practical reasons [130], however, it can be substantially large when the body is accelerated at a high rate [217, 94]:

$$F_{B,i} = \frac{3}{2} d_p^2 \sqrt{\pi \rho_g \mu_g} \int_0^t \frac{\frac{Du_{g,i}}{Dt} - \frac{du_{p,i}}{dt}}{\sqrt{t - t'}} dt'. \tag{3.17}$$

Reeks & McKee [175] have shown that the Basset term has to be modified to include the case when there is an initial velocity. The term becomes

$$F_{B,i} = \frac{3}{2} d_p^2 \sqrt{\pi \rho_g \mu_g} \left[\int_0^t \frac{\frac{Du_{g,i}}{Dt} - \frac{du_{p,i}}{dt}}{\sqrt{t - t'}} dt' + \frac{u_{g,i} - u_{p,i}}{\sqrt{t}} \right]. \tag{3.18}$$

3.1.3 Equations retained for this study

The particle equation of motion including the drag, the pressure (buoyancy) force, added mass force, and the Basset force is often referred to as the Basset-Boussinesq-Oseen (BBO) equation, who originally derived it. It is probably the most common equation to solve for the path of a particle. A rigorous derivation of the equation of motion of small particles (with diameter comparable or smaller than the Kolmogorov length-scale) in nonuniform flows has been performed by Maxey & Riley [129]. The study of Maxey & Riley [129] has been considered by many as the definitive study on the equation of motion of a solid sphere under creeping flow conditions. The resulting form of the transient equation of motion encompasses the unsteady and non-uniform fluid motion as well as body forces. Their final expression is the Lagrangian equation of the motion of the sphere. In a lesser-known paper, published on the same year, Gatignol [70] also derived a very similar expression for the Lagrangian equation of motion of a solid sphere. For the case of particles much heavier than the fluid, Elghobashi & Truesdell [51] have shown that the only significant forces are the Stokes drag, buoyancy and Basset forces. They also report that the Basset force was always an order of magnitude smaller than the drag and buoyancy forces.

For gas-particle flows where the ratio of the dispersed phase density is much larger than the gaseous phase density ($\rho_p/\rho_g \sim 10^3$), the BBO equation can be simplified to

$$\frac{dx_{p,i}}{dt} = u_{p,i} \tag{3.19}$$

$$\frac{du_{p,i}}{dt} = \frac{\tilde{u}_{g,i} - u_{p,i}}{\tau_p} + g_i, \tag{3.20}$$

where $\tilde{u}_{g,i}$ is the fluid velocity at the position of the particle assuming that the flow field is locally undisturbed by the presence of this particle (Gatignol [70], Maxey & Riley [129]). In a first approximation, the velocity is assumed to be equal to the interpolated filtered velocity at the position of the particle (Wang & Pletcher [224], Yamamoto *et al.* [230], Apte *et al.* [6]). The effect of the subgrid fluid turbulence is assumed to be negligible owing to the large inertia of the solid particles (Fede & Simonin [58]). The particle relaxation time τ_p is defined as

$$\tau_p = \frac{\rho_p d_p^2}{18\mu_g C_D}. \tag{3.21}$$

As previously mentioned, for the drag coefficient we retain the correlation proposed by Schiller & Naumann [191] (Eq. (3.6)).

A linear interpolation algorithm (see Section 3.5) is used to compute the fluid velocity at the position of the particle. In cases where the particle relaxation time is much larger than the time scale of filtered velocity fluctuations such a linear interpolation is supposed to be sufficiently accurate to resolve particle motions (Simonin, personal communication, April 11, 2007).

The influence of the particles on the gas phase is taken into account by using one of the two methods compared in Boivin *et al.* [19] to numerically implementing the fluid-particle coupling. The first one is the so-called Particle-in-Cell (PIC) method, where the coupling force is calculated as the average of all the forces in the control volume surrounding a grid point. The second method, rather than a summation of forces around a point is based on a projection operation. In this method, each point force is distributed to the neighbouring grid nodes based on the cell volumes (as in Squires & Eaton [206]) or on the distances between particle and their nearest grid nodes (as employed by Elghobashi & Truesdell [52]). This last option is the one retained for implementation in this thesis (see Section 3.6). A slightly different projection method has been proposed by Sundaram & Collins [213]. They show that, if the weights used to interpolate the fluid velocity onto particle position are also used to transfer the coupling force back to the flow grid, then the local kinetic energy of the fluid-particle interaction is conserved. However, this method needs in general a high-order interpolation scheme to calculate the fluid velocity at particle positions. Lower-order schemes have large truncation errors at small length-scales and may not provide an accurate representation of the fluid velocity. A comparison of this last projection method and the commonly used with projection onto neighbouring nodes is presented by Narayanan *et al.* [142]. For the particle-laden mixing layer that they studied, no significant advantage was observed with this new method. The effect of high-order projection methods was not studied in this thesis since the use of higher-order interpolation algorithms would also be needed.

The particle equations of motion are advanced in time using the same Δt of the gaseous phase (see Section 2.2). Therefore, we consider that particle relaxation times are always greater than the minimum of the gaseous time step and they cannot cross more than one cell by time step.

3.2 Data structure

The two main criteria that guide the choice of particle data structure are efficiency and clarity. Given a high number of particles that are supposed to be tracked, using too many superfluous parameters obviously increases the memory requirements and the computational cost, whereas using too few parameters may impact negatively the comprehension of the code or even worst, can limit the range of application. This section briefly presents the main data structures that have been implemented in the Lagrangian module of AVBP.

3.2.1 Lagrangian variables

In Subsection 2.1.2, we presented the conservative variables stored for the gaseous phase; the variables associated with each particle are separated in two arrays depending on their data type: integer or real. The integer data type contains the following information:

- **The particle identifier**, id_p: this number is used to label a particle, for example for the reconstruction of particle trajectories with post-processing tools. It is not directly used during the simulation except for debugging purpose.
- **The processor number (containing the particle)**, np_p: this information is used during communications of particles that cross the interfaces between processors. In addition, if this data is known at the beginning of the simulation, the solver will consider that a previous search has been performed and avoids the initial search step (see Subsection 3.4) leading to a corresponding gain in time.
- **The cell number (containing the particle)**, nc_p: this parameter is one of the most important since it allows the localisation of a particle inside the computational domain. It is used during the simulation to improve efficiency of the search algorithms by limiting the guest cell where a particle is searched when it leaves the current cell. Since particles change position each time step, being able to locate them quickly is crucial.
- **The kind of particle**, st_p: this parameter identifies the different types of particles: tracers (fluid particles), solid particles or droplets (heating and evaporating)[1].

The real data type contains the following information:

- **The particle position**, $x_p,\ y_p\ [, z_p]$: these are the particle coordinates in two or three dimensions. The initial particle position is an input parameter and the new particle position is obtained by solving Eq. (3.20).
- **The particle velocity**, $u_p,\ v_p\ [, w_p]$: these are the velocity values of the particle in two or three dimensions. These values are usually read as input parameters but they can be reset at the beginning of the simulation if particles need to be initialised with the fluid velocity.

[1] Note that if particles and droplets are used in the same simulation they are considered to have the same density. Multicomponents simulations are not the object of this thesis but introducing a variable that carries this information is straightforward.

- **The particle radius**, r_p: this variable contains the information of the particle radius. Polydispersion is then automatically taken into account via this simple information.
- **The particle temperature**, T_p: this variable contains the information of the particle temperature. This information is not used for tracers or solid particles, only droplets can heat and evaporate.

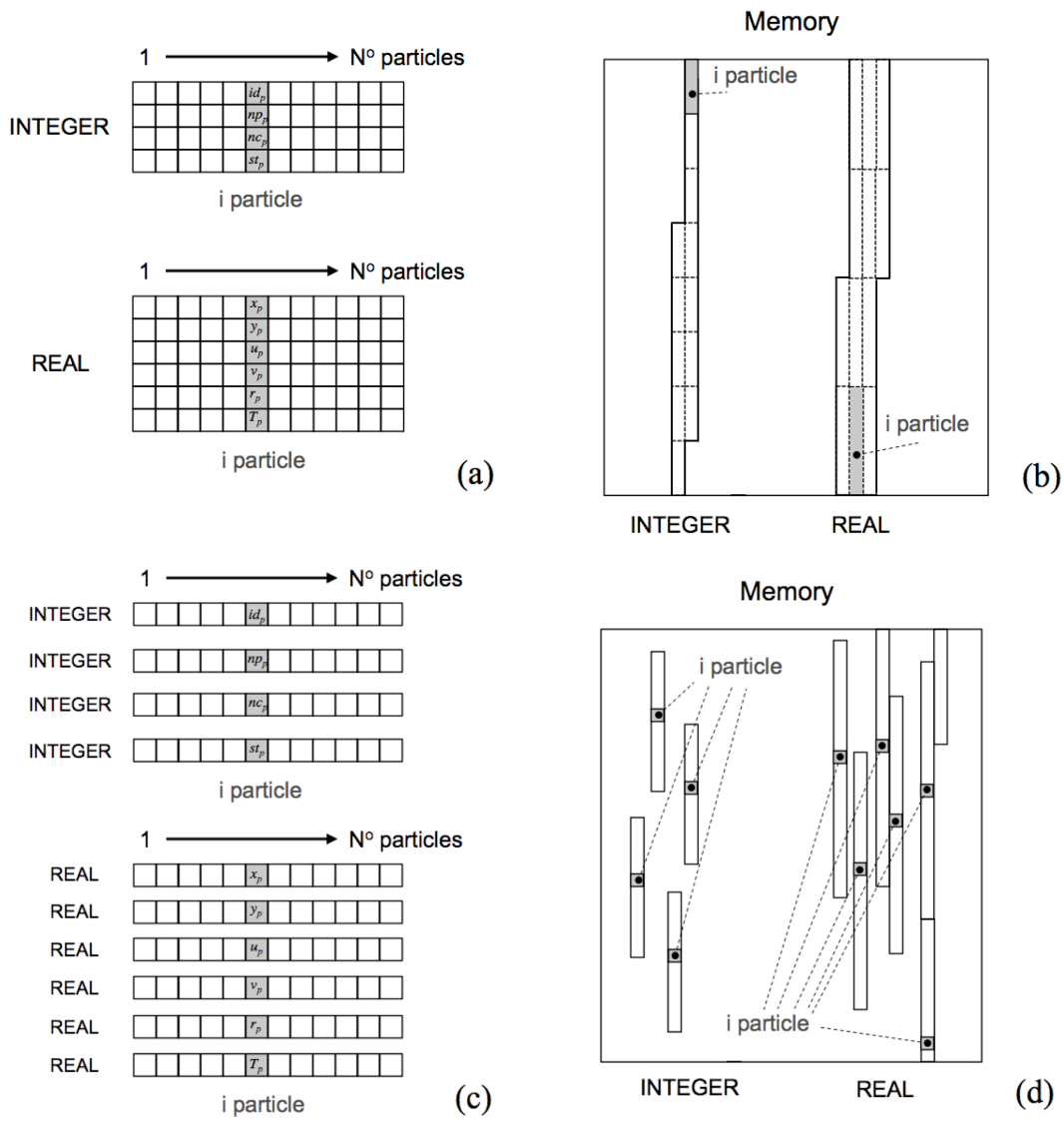


Figure 3.2 - *(a) One integer and one real array used to store particle information; (b) example of memory storage of previous integer and real arrays; (c) several integer and real arrays, one for each particle variable; (d) example of memory storage for the different arrays. In all figures, data of an 'i' particle are highlighted in gray.*

Figure 3.2 (a) sketches the data-storage technique retained for particle variables (for simplicity, only the 2D case is shown). Integer and real data type are stored in two-dimensional arrays: the first dimension contains the data of a single particle (particle identifier, cell number, etc); the second dimension indicates the number of particles. An example of the memory storage representation associated to this structure (for Fortran arrays, variables are stored in columns) is shown

in Fig. 3.2 (b); it is really clear that the set of parameters related to each particle are quickly accessible since they are close in memory. This choice is motivated by the fact that real data type are used at the same time since particle position, velocities, radius and temperature are updated even in the same subroutine.

Other solvers may use a different data structure. Figure 3.2 (c) displays a set of integer and real arrays: each one is a one-dimensional array and contains the same variable for all the particles (e.g., particle x-coordinate, particle radius, etc.). However, this kind of organization implies an increase in memory cache defaults since every time that a particle variable is needed there will be jumps in memory to recover the information (Fig. 3.2 (d)). This choice of data structure was mainly motivated in the past by the use of vectorial computers which were extremely fast to perform simple loops if data was cache friendly. To do this, variables and loops were adapted to be written in a particular way. Current platforms are more 'sensitive' to cache misses because of their low memory storage capabilities and therefore less suitable to this kind of storage. The principle being the increment of the number of processors required for the simulation in case of memory problems.

3.2.2 Cell-number pointer

Subsection 2.5.2 describes the grouping strategy used in Avbp. This technique is done once at the beginning of the simulation and remains unchanged during the run. This method implies that all the elements of the grid are grouped in chunks of cells in order to update variables that are stored in grid nodes in an efficient manner. Loops over grid elements are therefore the underlying structures of the code. Then, the variables stored in every node of each cell are updated via gather/scatter operations (see Fig. 2.2).

Concerning the particle treatment, making loops over the grid elements to locate particles would increase CPU time since elements without particles should also be checked. For example, in simulations where particles are concentrated in a particular region (e.g., near the fuel injector) some processors will keep on covering void zones. A more suitable strategy consists of making loops over the number of particles. This choice is motivated by the fact that particles change cell and subdomain frequently. The gain of time obtained for the gaseous phase by grouping cells would be inefficient to the particle treatment. A high number of cache misses would be produced in particle arrays due to jumps in memory since particles arrays are not ordered following a pattern related to cell grouping. Ordering strategies to make these arrays cache-friendly would also be expensive since these arrays can change every time step and they should be ordered during the simulation.

Considering the solution of loops over particles implies that the task of particle tracking is supposed to be efficient. In other case, this possibility would not be retained. However, there is no data structure available in Avbp to provide a cell number. Obviously, the reason is that this kind of information is not really needed for current simulations but for particle tracking it is crucial. To mitigate this lack, one of the first priorities was the creation of a data structure to obtain the cell number while the particle is moving. Due to the static nature of Avbp (see Subsection 2.4.1) this structure is created only once, at the beginning of the simulation. This array is two-dimensional: the first dimension points to the beginning address of a cell and the second dimension indicates the number of vertices of the cell. This last dimension is needed because of the hybrid structure of Avbp. While grouping cells, the type of element (triangles, quadrilateral, etc) is always the same in each group (see Subsection 2.5.2), groups of different type of elements are mixed since the 'ielno' pointer (see Subsection 2.5.1) is ordered to consider first the owned cells

(cells without any node in contact with other processors or cells type 1) and next the interface cells (cells with at least one node in the interface or cells type 2). Therefore, in a two-dimensional grid with triangles and quadrilateral, cells of type 1 and 2 are first stored for groups of triangles and then for quadrilateral ones. An example of this is presented hereafter. Figure 3.3 (a) shows a simple 2D hybrid grid with six quadrilaterals and twelve triangles. Figure 3.3 (b) presents this grid partitioned in two subdomains; each one with the same number of cells. For the sake of clarity, only the subdomain with a different type of elements is discussed (subdomain 1). The local node number after reordering (see Subsection 2.4.5) is written in gray near each node. The order in which the cells are covered during the simulation is included in the centre of the cells. As it can be observed, this subdomain contains five quadrilateral cells of type 1, and three triangles and one quadrilateral of type 2.

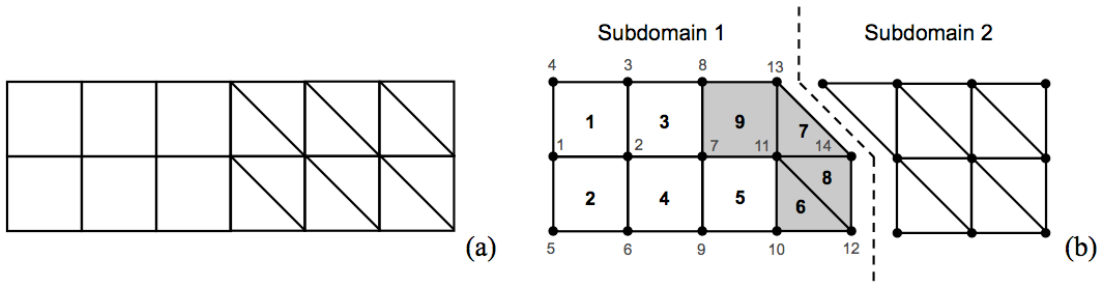


Figure 3.3 - *(a) Example of a simple 2D hybrid grid with six quadrilaterals and twelve triangles; (b) Result of partitioning the hybrid grid in two subdomains. The local nodes and cells numbers (after reordering) are highlighted for the subdomain containing different type of elements.*

The cell-number pointer created by the Lagrangian module for this subdomain will contain the following information for the processor containing subdomain 1:

Beginning of the cell:	1	5	9	13	17	21	24	27	30	34

This is a one-dimensional array of size: number of cells + 1. Each column corresponds to the beginning position of a cell. The last element is needed since the number of vertices of the cell is obtained by subtracting two consecutive cells: e.g., the 8th and 9th cells have 3 $(30 - 27 = 3)$ and 4 $(34 - 30 = 4)$ vertices, respectively. This kind of storage can be found in some index arrays used to store sparse graphs in CSR format since it avoids a supplementary dimension with the information of the type of element.

3.2.3 Number of interface element pointer

To simplify the management of particles crossing subdomains, a new array is necessary. Particles crossing interfaces are clustered in a set of arrays, as many as the number of neighbouring subdomains supposed to receive a particle. This information is obtained using this pointer which is used to gather each particle to the corresponding subdomains arrays depending on the cell where it is located. This array is created at the beginning of the simulation and its size is much smaller than the arrays presented in the previous sections. On the contrary, it can vary from one processor to another more than other arrays since its size depends on the number of interface cells (type 2) and in the number of adjacent subdomains. This difference comes from the fact that partitioning algorithms do not try to balance the number of adjacent partitions but the number of cells by subdomain (see Subsection 2.4.2), creating indirectly a slight imbalance in this case.

Figure 3.4 (a) shows an example of a simple 2D grid partitioned in four subdomains. In gray, cells with one or more nodes in the subdomain interface. The number inside these cells indicates how many adjacent partitions are in contact with the cell. In most cases there is only one adjacent subdomain by cell but while increasing the number of partitions, the number of cells in contact with more than one subdomain increases. Figure 3.4 (b) presents an example of memory storage of this array. The first row indicates how many adjacent partitions are in contact with this cell. The rest of the components are the numbers of these adjacent partitions.

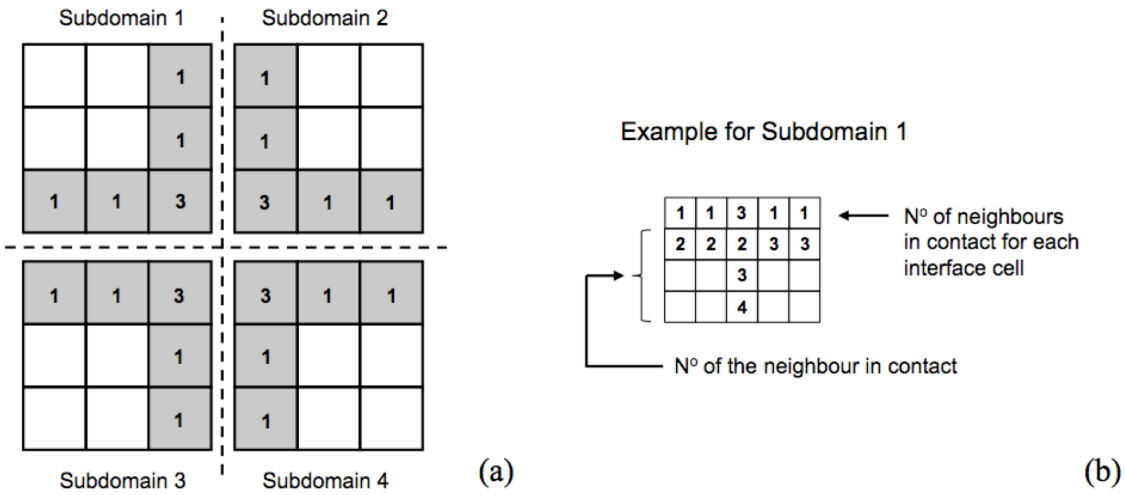


Figure 3.4 - *(a) Example of a simple 2D grid partitioned in four subdomains, find in gray cells of type 2; (b) Example of the interface pointer for subdomain 1.*

This data structure could be improved by adapting its size to store only the useful information, specially since the number of cells with more than one node in contact with several subdomains is quite low. The choice of the current data structure is motivated by the idea of keeping simplicity in the array manipulation and usage. As its size is quite small (if we compare it with other arrays used within AVBP) this extra memory is not considered a problem. However, for future simulations in massively parallel machines where the number of adjacent partitions increases, new developers are encouraged to improve this array to reduce memory requirements.

3.3 Locating particles in elements of arbitrary shape

Locating particles in a generalized-coordinate structured code is straightforward. For example in a two-dimensional case with uniform grid spacing, the cell (i_c, j_c) where a particle is located can be easily calculated as:

$$i_c = \text{Int}\left\{\frac{(x_p - x_{min})}{(x_{max} - x_{min})}\, N_x\right\} + 1, \qquad j_c = \text{Int}\left\{\frac{(y_p - y_{min})}{(y_{max} - y_{min})}\, N_y\right\} + 1, \tag{3.22}$$

where $\mathbf{x_p}$ is the current location of the particle, $\mathbf{x}_{min}$ and $\mathbf{x}_{max}$ denotes the range of coordinate values for the current mesh, and N_x, N_y the number of cells in the x and y directions. $Int\{\quad\}$ denotes truncation to the lowest integer part.

With this simple equation, we know that the particle is located inside a cell and also the cell where it is located. This is not the case in unstructured grids because of the (rather) arbitrary shape of the grid elements which makes difficult to get a simple criterion to locate particles. This section tries to give an answer to the criterion used to know if a particle is located inside a cell or control volume. The solution adopted and implemented in the Lagrangian module should be able to deal with the kind of elements presented in Fig. 2.19. In the literature, several researchers have already tried to solve this problem: Seldner & Westermann [196], Westermann [227] and Löhner [120]. They describe approaches to locate particles in particle-in-cell codes that are briefly explained below:

- **Calculation of areas or partial volumes**

  One approach to determine whether a particle lies inside a grid cell is based on the calculation of areas or partial volumes. The nodes of the grid element are joined to the particle location, and the volumes of the resulting sub-cells are compared to that of the control volume. An example is presented in Fig. 3.5 with a quadrilateral cell. If the particle lies inside the control volume, the sum of the sub-cell volumes will be equal to the total volume (Fig. 3.5 (a)). If the particle is outside the control volume, the sum of the sub-cell volumes is greater than the total volume (Fig. 3.5 (b)). This method amounts to compute the sub-cell volumes and the comparison with the total volume of the cell for each particle searched. The main shortcoming of this approach is that it fails drastically for highly skewed meshes due to round-off inaccuracies in the computation of partial volumes.

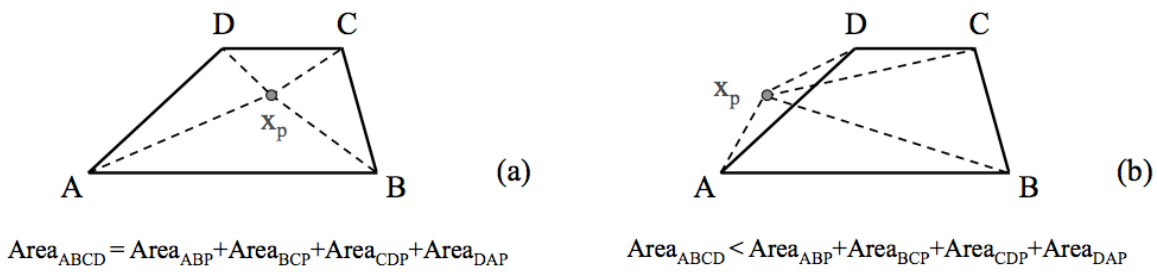


Figure 3.5 - *Calculation of areas to detect if (a) a particle lies inside the quadrilateral ABCD; (b) the particle is outside the quadrilateral.*

- **Evaluation of shape-functions**

  A second approach is to evaluate the shape-function values of the particle with respect to the coordinates of the points belonging to the element:

$$\mathbf{x_p} = \sum_i N^i \mathbf{x_i}. \tag{3.23}$$

  For triangles in $2D$ and tetrahedra in $3D$, there are two equations for three shape-functions and three equations for four shape-functions, respectively. The sum-property of shape-functions,

$$\sum_i N^i = 1 \tag{3.24}$$

yields the missing equation, making it possible to evaluate the shape-functions from the following system of equations:

$$\left\{ \begin{array}{c} x_p \\ y_p \\ z_p \\ 1 \end{array} \right\} = \left\{ \begin{array}{ccc} x_1 & x_2 & x_3 \\ y_2 & y_2 & y_3 \\ z_3 & z_2 & z_3 \\ 1 & 1 & 1 \end{array} \right\} \cdot \left\{ \begin{array}{c} N^1 \\ N^2 \\ N^3 \\ N^4 \end{array} \right\}, \tag{3.25}$$

Then, the particle is inside the element if and only if

$$\min(N^i, 1 - N^i) \geqslant 0, \quad \forall i. \tag{3.26}$$

For other types of elements (quadrilateral, hexahedra, etc) more nodes than equations are encountered. Therefore, in order to determine if a particle is inside an element, the easiest way is to split the element into triangles or tetrahedra and evaluate each of these sub-elements. If the particle happens to be in any of them, it is inside the element. However, the drawback of this procedure is that it is computationally expensive since it requires the evaluation of all sub-elements before particle detection.

- **Comparison of face-normal vectors**

  The third approach projects the particle location onto the faces of the grid element and compares these vectors with face-normals for all faces (Fig. 3.6). If the particle lies inside the cell, the projected vectors point the same direction as the face-normals. This technique is very accurate even for highly skewed elements. In addition, if the condition is not verified on a face, there is no need to continue checking the rest of the faces inside this cell, which reduces dramatically the overall CPU time of the algorithm.

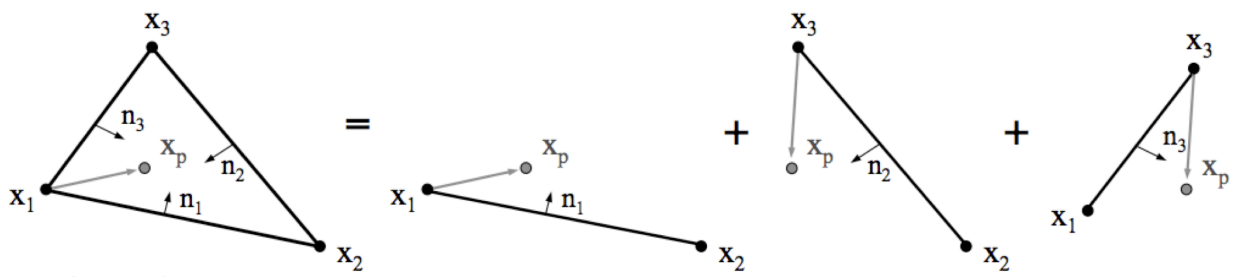


Figure 3.6 - *Comparison of particle location vectors and face-normals vectors of the grid element.*

This last option has already been used in other simulations of Lagrangian particle-laden flows on unstructured grids (Apte *et al.* [7], Haselbacher *et al.* [82] call it the "in-cell test") and is the one chosen to be implemented in the Lagrangian module. On one hand, this solution can deal with any kind of element shape due to its face treatment, and on the other hand, there is no need to check the whole element to know if a particle is located inside it. In case of failure of any of the face detection, a new cell can be checked with a resulting gain in CPU time since fewer arithmetic operations are performed. A search algorithm is then required to efficiently select the control volume to which the criterion should be applied.

3.4 Search algorithms for particles on unstructured grids

In Lagrangian simulations, particles change position at each time step of the simulation. But before they can contribute information to the grid (e.g., two-way coupling) or sample field information from it (interpolation of gaseous-phase properties), their new containing cell must be identified. Therefore, they must be traced through the grid. As previously mentioned, in regular, uniform Cartesian grids, the new cell can be detected with Eq. (3.22). Locating particles on a mesh becomes more and more complicated as soon as any irregularity is introduced in the grid. Even for rectangular grids with non-uniform spacing, Eq. (3.22) is no longer valid. The study of search algorithms in this thesis is motivated by the need of using an efficient way to locate particles on the unstructured and hybrid grids used in AVBP.

A search algorithm is an algorithm that takes a problem as input and returns a solution to the problem, usually after evaluating a number of possible solutions. The set of all possible solutions to a problem is called the search space. Search algorithms can be classified as: uninformed, list, tree, graph or informed search algorithms, for example. Informed search algorithms use heuristic functions that take advantage of the structure of the search space to try to reduce the amount of time spent searching. List search algorithms examine each element of a list in order to find one element of a set by some key. Tree search algorithms use a class of hierarchical data structures based on the principle of recursive decomposition. A node is taken from a data structure, its successors examined and added to the data structure. By manipulating this data structure, the tree is explored in different orders to find the element searched. Search algorithms, specially hierarchical data structures, are becoming increasingly important representation techniques in the domains of computer graphics, image processing, computational geometry, geographic information systems, robotics and software-based interpolation of unstructured field data.

In the following, a (non-exhaustive) list of search algorithms is presented. Some of them are used within the Lagrangian module implemented in this thesis and others are just presented since they are used with other particle tracking schemes presented in the literature. Then, Subsections 3.4.1 - 3.4.4 detail the different search strategies implemented in this module:

- **Brute force**

  Brute-force search algorithm uses the simplest, most intuitive method to search a particle in a grid. This approach simply loops over all the elements of the grid and applies a localisation criterion (in our case the one described in Section 3.3) to locate a particle in a cell. This algorithm is simple to implement, and will always find a solution if it exists. However, its cost is proportional to the number of candidate solutions, which, in many practical problems, tends to grow very quickly as the size of the problem increases. Therefore, brute-force search is typically used when the problem size is limited, or when the simplicity of implementation is more important than speed. It is also used when there are problem-specific heuristics that can be used to reduce the set of candidate solutions to a manageable size. For example, the modified brute-force approach evaluates the closest point of the mesh to the particle location and only considers the elements surrounding that point. Should this attempt (which in general is very successful) fail, the elements surrounding all the close points are considered. If this also fails the search region is enlarged or the brute-force method is applied. Another ways of subdividing the search space can be achieved with the use of Cartesian background grids (bins) or tree structures to reduce the number of elements queried, at the expense of creating and storing additional search tables. Some of them are presented next.

- **Bins**

  The idea is to superimpose the irregular foreground grid on a regular background grid. The elements of the foreground mesh that cover each cell of the Cartesian mesh are stored in a linked list. An example of a Cartesian background grid and some particles is presented in Fig. 3.7. Given the new particle position x_p, the cell of the Cartesian background grid are obtained from Eq. (3.22). Then, all elements covering this cell are checked to find the new host element for the particle. The main shortcoming of this approach is the inefficiency and inaccuracy that arises when meshes with large variations in element size are employed. If the number of cells employed for the construction of the background grid is too small, the number of elements to be checked may become excessive. If, on the other hand, the number of cells employed for the background grid is too large, the storage overhead may be prohibitive.

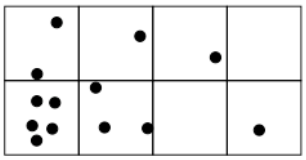


Figure 3.7 - *Example of space division with a Cartesian background grid to locate particles (dots).*

- **Quad/octrees**

  To circumvent the problem for meshes with large variations in element size, a hierarchy of Cartesian meshes can be used. This is most easily accomplished with quad trees (2D) or octal trees (3D) (Finkel & Bentley [63], Samet [188], Knuth [101, 102]). A *quadtree* (octree) is a tree data structure in which each internal node has up to four (eight) children. This structure is based on the principle of recursive decomposition. The decomposition may be done into equal parts on each level (i.e., regular polygons and termed a *regular decomposition*), or it may be governed by the input. The resolution of the decomposition (i.e., the number of times that the decomposition process is applied) may be fixed beforehand, or it may be governed by properties of the input data. An alternative to the quadtree representation is to use a decomposition method that is not regular (i.e., rectangles of arbitrary size rather than squares). This alternative has the potential of requiring less space. The main shortcoming of this procedure is the additional complexity in coding and the essentially scalar nature of the procedure. A tree-based scheme introduces additional indirect addressing that makes it all but impossible to vectorize.

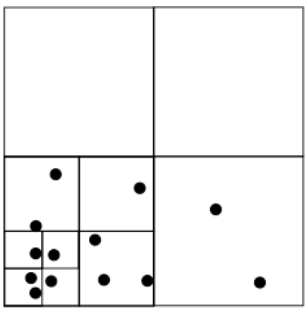


Figure 3.8 - *Example of space division with a quadtree to locate particles (dots).*

- **Digital trees**

  Digital trees are recursive data structures commonly used for searching and sorting operations. They are constructed by defining a root, and assigning an element to one of two branches based upon whether the bounding box of such element satisfies some geometric condition. By following this procedure for all elements in a list, a tree is built up in such a way that, when searching for the location of an element, each search step ideally reduces the number of elements (N) to be checked by a factor of two, resulting in search times that can scale with $\mathcal{O}(logN)$. Digital tree algorithms are particularly attractive for unstructured mesh generation, in that setup times and tree storage requirements are modest - proportional to the number of elements - and thus trees can be easily modified as new elements are introduced. A popular variant is the alternating digital tree (ADT) algorithm [21], which provides well-defined branch criteria for the general case of n-dimensional meshes. One variation of this algorithm is: the augmented spatial digital tree (ASDT) [59].

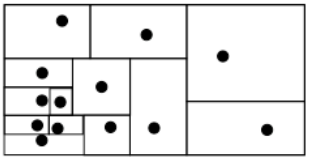


Figure 3.9 - *Example of space division with a ADT to locate particles (dots).*

- **Neighbor to neighbour**

  In this approach, the closest node to the particle is first identified using some of the methods described above, and then only elements surrounding that node are searched. It is easy to see how this can substantially reduce the number of element required; however, this saving is achieved at the not negligible cost of calculating distances from the particle to each node in the mesh. Moreover, the nearest node will not necessarily be connected to the element containing the particle. Thus, it is usually necessary to query at least one extra layer of elements surrounding the nearest node to ensure robustness. In addition to its obvious simplicity, an advantage of nearest neighbour searches is the reduced setup and storage requirements: the search table simply consists of a vector of nodes pointing to another vector of connected elements. This algorithm is usually used when a particle is supposed to cross several cells in one time step (as it can be the case in implicit-time advancement that allows large time steps) and it is considered very efficient if the particle can be found in less than 10 attempts [120] (Fig. 3.10). A comparison of performance of the modified brute-force and the known-vicinity algorithm can be consulted in Apte *et al.* [7], where a version of the known-vicinity algorithm of Löhner [120] is used to track the evolution of particles in a coaxial-jet combustor. More recently, Haselbacher *et al.* [82] proposed an efficient and robust version of the algorithm of Löhner & Ambrosiano [121] applied to a solid-propellant rocket simulation which improves the search algorithm for particles crossing between processors and avoids limitations to small particle displacements.

The following paragraphs detail the different search strategies used to track particles in Avbp. The choice of different algorithms is motivated by the reduction of memory and CPU time requirements depending on the situation. Four different cases can be distinguished and the search algorithms used are explained hereafter:

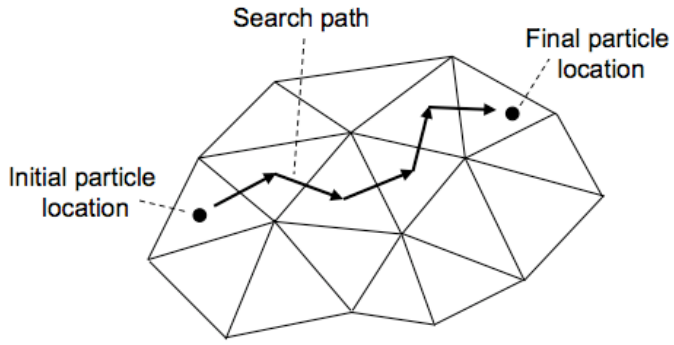


Figure 3.10 - *Sketch of the neighbour to neighbour algorithm. Initial and final particle location are represented by black dots and the search path of the particle by arrows.*

- Searching particles for the first time.
- Searching injected particles.
- Searching particles during the simulation.
- Searching particles crossing boundaries between processors.

3.4.1 Searching particles for the first time

In a Lagrangian simulation, particles can be injected at each time step or read from a file at the beginning of the simulation. In this case, two options are possible: (i) the file contains the solution of a previous simulation; (ii) the file is generated by a pre-processing tool. This subsection treats this last point which may be the most time consuming.

When a particle data file is generated by a pre-processing tool, the cell number and the processor where the particle is located are unknown. Thus, particles must be searched for the first time to obtain this information and if the number of particles and the number of cells to span is high, the search algorithm can be extremely time consuming (see Subsection 4.5.1). The algorithm used in the Lagrangian module to search particles for the first time is based on a quad/octree search algorithm. The subroutines were provided by F. Collino, consultant researcher in the Electromagnetism Computational Project of CERFACS and were adapted to be used in AVBP. As previously mentioned, these algorithms need a data structure to create the tree of the background grid. The idea is to construct a boxes structure where each box contains a particle or the centroid of an element of the mesh. For a given particle p, instead of searching the element in which the particle is located in, by inspecting all the elements, the search is restricted to the elements located in B_p (the box where the particle p is) as well as the neighbouring boxes of B_p. The construction of the boxes is made recursively. The dimension of the data structure depends on the number of elements and particles, and it is limited by two parameters: (i) the maximum edge length of the elements of each partition; (ii) an input parameter used to rescale the former. The first parameter depends on cell dimensions, therefore, grids with highly anisotropic elements may imbalance the search operation. Fig. 3.11 shows an example of this problem: the length of the largest edge of subdomain 3 is controlled by big cells of the coarsen region (right zone of Fig. 3.11 (a)). The first level of the Cartesian background grid created is presented in Fig. 3.11 (b). The data structure of the tree is smaller than the one of subdomains 1 and 2, however, the number of elements to check

inside each background grid cell is higher and so is, the time spent to locate particles. The second parameter is constant and fixed by the user. The higher the value of this parameter, the smaller the memory requirements for the tree structure used by the search algorithm and the slower the search is. This trade-off between memory and time provides a better control of particles tracking in cases where any of them could be restrictive (its value is an integer equal or higher than one).

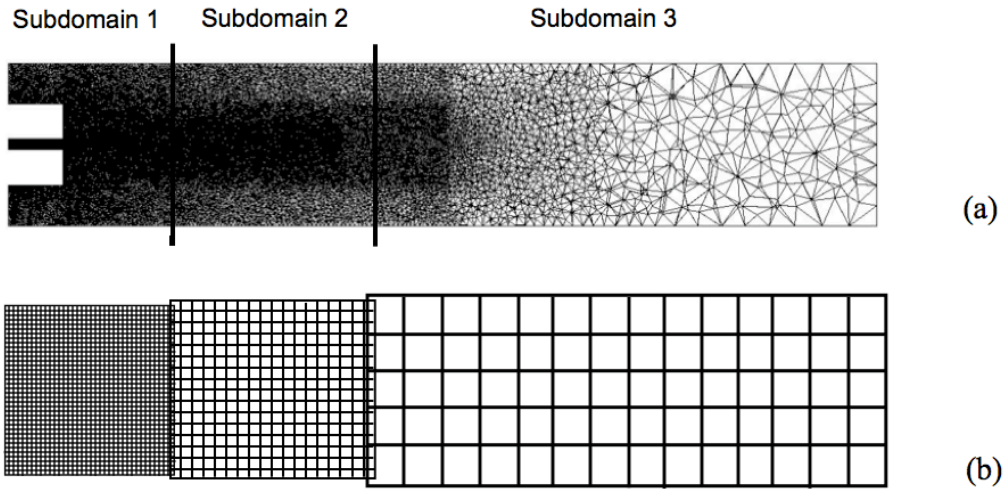


Figure 3.11 - *(a) Example of limits of a high anisotropic mesh partitioned in three subdomains; (b) Result of the first level of box sizes used to locate particles. Dimensions of the Cartesian background grids of each subdomain are functions of the cell edge lengths.*

Once each particle is located in a cell via this algorithm, the cell number is stored in the particle integer-data array to be used during the simulation. All the data structures related to this tree are deallocated to free memory that could be necessary for the whole simulation. An example of the use of this search is found in the HIT simulation presented in Chapter 4. An input particle file is created with particles position and velocity, however, the cell number is initialised to zero since this value is unknown at the beginning of the simulation and therefore, so a first search is needed.

3.4.2 Searching injected particles

In the cases where particles are injected (see Section 3.7), the search algorithm used to locate particles is a modified brute force which searches in the subset of cells close to the injection area. This region is determined at the beginning of the simulation and contains a reduced number of cells, allowing a fast search and low memory requirements. Figure 3.12 shows an example of the determination of the injection cells. The position and dimensions of the injection area are given by the user at the beginning of the simulation as input data. Note that this region can cover a zone where different subdomains are connected, which increases the difficulty of the implementation in parallel architectures. Each partition performs a loop over all the nodes to detect the ones at the intersection between the domain and the injection region. Then, these nodes are stored in a list. The numbers of the cells containing injection nodes are obtained via the cell-number pointer (see Subsection 3.2.2) and stored in an array that will be used during the simulation.

Injection in the Lagrangian module is done by the root processor (in the context of the master-slave paradigm) or by the master processor (in the SPMD approach since root and master are the

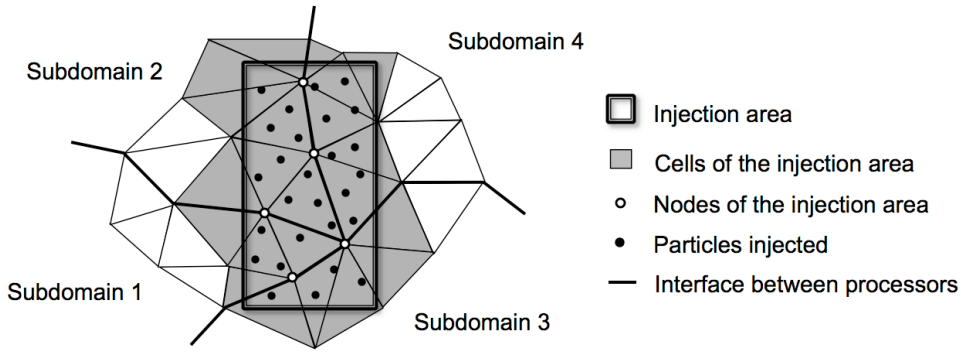


Figure 3.12 - *Example of injection area located near the intersection of several partitions.*

same). After detection of injection cells by each partition (Fig. 3.13 (a)), the numbers of processors sharing the injection zone are recovered in a reduction operation and stored by the root in an array. Injection can be done at each time step during the simulation or every certain number of time steps specified by the user. The root processor injects particles and sends all data (position, velocity, etc.) only to the set of processors containing the injection cells (Fig. 3.13 (b)). This communication is always done between the same processors but the number of particles may vary depending on the injection parameters. After reception, each partition searches particles only in the set of injection cells to find the cell number to be used in the following time step. An important point during the injection step is the ability to reduce the injection area to the lowest number of cells, so is, to avoid a possible load imbalance between the processors containing injection cells and the others due to processors extra memory and time requirements to compute injection.

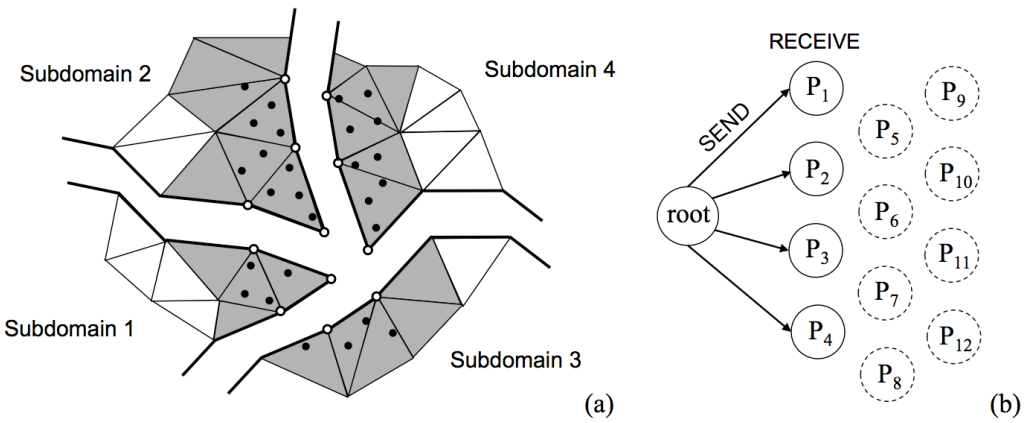


Figure 3.13 - *(a) Illustration of the injection cells of four subdomains with particles superimposed; (b) communication between the root (or master) processor and the injection processors only.*

3.4.3 Searching particles during the simulation

During the simulation, a cell number associated with each particle is stored in the integer-data array presented in Subsection 3.2.1. This number corresponds to the cell where the particle is located before it changes its position. Therefore, given a particle location and the associated cell, the search algorithm determines the cell that contains the new particle position. The procedure is divided into several steps described below.

The first step consists of checking the current cell in order to know if the particle is still inside of it, which is usually the case since particles displacements are small due to CFL time step restriction, In case of failure, the algorithm extents its search to the surrounding cells. This is the second step and only cells that contain at least one node in common with the current particle cell are considered, reducing the search to a first level of cells. To that end, a pointer containing the number of elements connected to a given node has been created at the beginning of the simulation. This procedure is based on the known-vicinity algorithm which allows a significant improvement in speed by only checking the elements that cover the immediate neighbourhood. Should this search fail, the third step consists of checking the cell number in order to see if the cell is in the interface between two or more subdomains. If it is the case, particle is considered to have exited the subdomain and data are stored in separate arrays to be sent to the adjacent partitions (see Subsection 3.4.4). Should this also fail, the particle is considered lost in the last step and a message is displayed to notify a problem in the search algorithm.

Figure 3.14 illustrates the second step of this particle-location problem. Assuming that a particle is no longer located in the current cell (dark gray cell), the first operation consists of the detection of the surrounding cells (light gray cells). Then, a loop over these cells is performed to detect the new cell containing the particle by evaluating the localisation criterion (see Section 3.3).

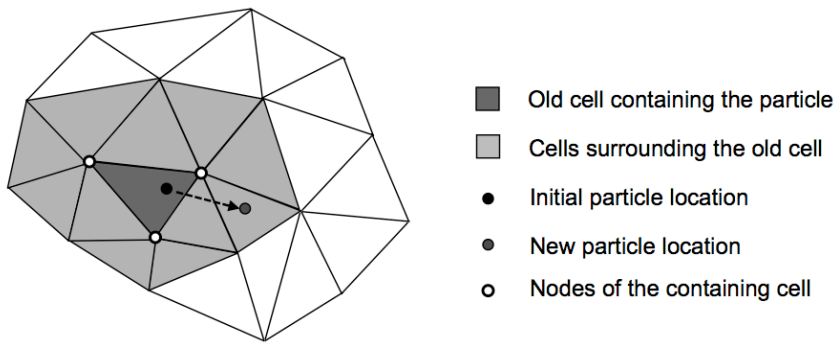


Figure 3.14 - *Sketch of a step of the search algorithm used during the simulation.*

Several improvements are possible for this algorithm. One may, in a first pass, evaluate the closest point to the given cell and only consider the elements surrounding that point instead of considering all the elements surrounding all the nodes, as pointed out by Löhner [120]. Another possibility consists of detecting the face of the cell intersected by the particle trajectory, jumping from neighbour to neighbour until the particle is found. This last option performs very well and also does not limit particles to small displacements [120, 7, 82]. Future developments of the Lagrangian module should focus on the implementation of these functionalities.

3.4.4 Searching particles crossing boundaries between processors

A major challenge in the parallelization of particle-tracking algorithms is that the number of particles to be communicated each time step can vary both in space and time. Data structures dealing with this dynamic situation should be efficient enough to avoid slowing down the whole computation or worst, reducing the speedup, especially in massively parallel computations. In addition, the parallelization must be able to treat correctly particles which cross one or more partition boundaries between the old and the new positions. Figure 3.15 shows an example of this exchange and the following paragraph try to clarify how it is done.

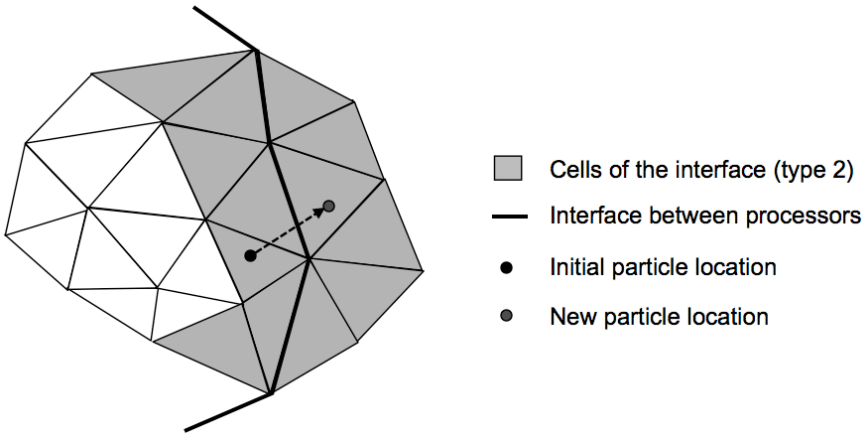


Figure 3.15 - *Illustration of particle crossing between processors.*

As mentioned in the previous subsection, particles not found and located in the interface cells (type 2) are treated in a particular manner. A data structure is implemented to help in this task (see Subsection 3.2.3). When a particle is supposed to cross an interface, this pointer provides a list with the numbers of the adjacent partitions of the containing cell. Then, the counter of particles to be sent is incremented for each adjacent partition of the list and particle data are stored in separate arrays, one for each subdomain where the particle is supposed to be. Once the loop over all particles is finished and particles are either located or flagged for communication, the number of particles to be communicated between partitions is determined by a reduction operation. Then, a send/receive operation is performed between adjacent partitions if the number of particles to be exchanged is greater than zero. The buffer used to store particles exchanged is reallocated as soon as a new set of particles is received. These contributions are added in the same order of reception to reduce communication time and to avoid accumulation of messages waiting to be free. After the reception of all messages, new particles are searched by each partition only in the set of interface cells. Particles found are added at the end of the integer and real particle data arrays.

Figure 3.16 illustrates different communication situations that can be found while exchanging particles between subdomains (this is an example of a four-processor simulation where all processors are connected). Figure 3.16 (a) shows a communication between processor P_1 and processors P_2 and P_4. At the same time, processor P_2 communicates with processors P_3 and P_4 and processor P_4 with P_3. In a different iteration (Figs. 3.16 (b)-(c)), the number of particles and messages to exchange can be completely different. Good efficiency in the treatment of particles crossing subdomains is therefore crucial to reduce memory and time requirements.

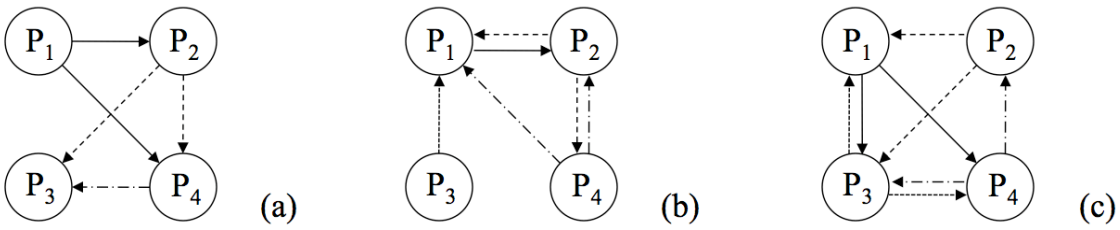


Figure 3.16 - *Different messages with particle data exchanges between processors at different time steps.*

3.5 Interpolation of gas-phase properties

Gas-phase properties, such as fluid density and velocity components, are needed at particle positions in Eq. (3.20). This data is obtained by interpolating the gas-phase variables at the vertices of the cell that contains the particle. As gas-phase properties are only stored at grid nodes, a new array with fluid interpolated quantities at particle positions is needed to store this information for each particle.

A discussion about the different interpolation algorithms (piecewise, linear, polynomial, spline interpolation, etc.) is out of the goal of this thesis and can be consulted in classical textbooks on numerical analysis. In the following, we briefly present the three interpolation algorithms that have been implemented in the Lagrangian module during this thesis (see for example Chapter 3 of [169, 168] for other algorithms and subroutines):

- Taylor series.
- Linear Least-Squares.
- Lagrange polynomials.

3.5.1 Taylor series

The Taylor series is a representation of a function as an infinite sum of terms calculated from the values of its derivatives at a single point. The Taylor series of a real or complex function $f(x)$ that is infinitely differentiable in a neighbourhood of a real or complex number a, is the power series:

$$f(x) = \sum_{n=0}^{\infty} \frac{f^{(n)}(a)}{n!}(x-a)^{(n)}, \tag{3.27}$$

where $n!$ is the factorial of n and $f^{(n)}(a)$ denotes the nth derivative of f at the point a.

The interpolation algorithm retained for this study is a first-order Taylor series. Therefore, Eq. (3.27) yields:

$$f(x) = f(a) + f'(x)(x-a). \tag{3.28}$$

The choice of this algorithm is motivated by the fact that the first derivatives of the gas-phase properties are already calculated and stored in AVBP so there is no need to create extra arrays. The expansion to a second-order interpolation, even if it is straightforward, will need the memory storage and the calculation at each time step of second derivatives, with the corresponding increase in CPU time and memory.

This interpolation function was implemented by V. Moureau during his thesis [137] as a post-processing tool. It is currently used in Arbitrary-Lagrangian Eulerian (ALE) simulations of piston engines to interpolate data of the final and initial grids associated with the different phases of the combustion cycle (intake, compression, combustion/expansion and exhaust). An analysis of the quality of first and second-order interpolation for an homogeneous isotropic turbulence test case is also presented in [137]. The second order shows a slight improvement in the prediction of the kinetic energy spectrum for high frequencies but this improvement is not high enough to justify

the increment in memory and time consumption due to the introduction of new arrays to calculate second derivatives.

To reduce problems related to high memory requirements in AVBP post-processing tools (which are all sequential), a complementary and parallel version of the interpolation tool proposed by V. Moureau has been developed in this thesis and is available in AVBP official versions at run-time. This facility is independent of the Lagrangian module but it is based on the same data structure and follows similar developments. More details of the implementation, including main features, validation test cases and performance, can be consulted in García [68].

3.5.2 Linear Least-Squares

The method of least squares can be used to find an approximate solutions to overdetermined[2] systems. The linear least-square problem can be formulated as:

$$\underset{x}{\text{minimize}} \; \|A\mathbf{x} - \mathbf{b}\|_2, \tag{3.29}$$

where A is an m-by-n matrix, $\mathbf{b}$ is a given element vector and $\mathbf{x}$ is the n element solution vector. For the least-squares interpolation a polynomial function is reconstructed around each point for which an interpolated value is sought. The coefficients of the polynomial are optimised such that the sum of the weighted squares of the interpolation error at the support points of the mesh to interpolate from is minimized.

To solve this problem, several subroutines can be found in the literature [169, 168]. In this thesis we choose to use Linear Algebra PACKage (LAPACK) subroutines. LAPACK [5] is written in Fortran 77 and provides routines for solving systems of simultaneous linear equations, least-squares solutions of linear systems of equations, eigenvalue problems, and singular value problems. In all areas, similar functionality is provided for real and complex matrices, in both single and double precision. More recent versions of LAPACK subroutines in Fortran 90 or 95 (LAPACK3E [4] or LAPACK95 [8]) have not been considered since these packages are less standard and their use may reduce AVBP portability.

The LAPACK algorithm used in this thesis is DGELS[3]. It solves overdetermined linear systems in the least-squares sense and it computes the minimum norm solution to underdetermined systems. Elmroth & Gustavson [53] propose a significantly faster and simpler version of this algorithm. The functionality of their implementation corresponds to that of the LAPACK routine DGELS but it outperforms it for all matrix sizes tested. The improvement is usually 50-100% and it is as high as 400%. The implementation of this new version could be considered in future updates to improve performance.

3.5.3 Lagrange polynomials

Another interpolation algorithm introduced to obtain gas-phase properties at particle position is Lagrange polynomials. The Lagrange interpolating polynomial is the polynomial $P(x)$ of degree $\leqslant (n-1)$ that passes through the n points $(x_1, y_1 = f(x_1))$, $(x_2, y_2 = f(x_2))$, ..., $(x_n, y_n = f(x_n))$,

[2] A system with more equations than unknowns.
[3] See web page http://www.netlib.org/lapack/double/dgels.f.

and is given by

$$P(x) = \sum_{j=1}^{n} P_j(x), \qquad \text{where} \qquad P_j(x) = y_j \prod_{k=1,k\neq j}^{n} \frac{x - x_k}{x_j - x_k}. \tag{3.30}$$

In the case of quadrilateral (2D) and hexahedra (3D) in Cartesian grids, it can be shown that this polynomial formulation is equivalent to the partial volume calculation used to detect if a point is inside an element. This has been retained for the numerical implementation in this thesis.

3.6 Implementation of two-way coupling

3.6.1 Source terms between solvers

In two-phase flow modeling, the filtered Navier-Stokes equations (Eqs. (2.4)-(2.7)) presented in Subsection 2.1.1 need to be modified to take into account particles and droplets mass, momentum and energy exchanges with the gas. This is done by adding in the right-hand side of gas equations the corresponding source terms, $\dot{\rho}_p$, $\dot{F}_{p,i}$, $\dot{Q}_p$ and $\dot{S}_{p,k}$ for mass, momentum, energy and species equations, respectively:

$$\frac{\partial \overline{\rho}}{\partial t} + \frac{\partial}{\partial x_i}(\overline{\rho}\widetilde{u}_i) = \dot{\rho}_p \tag{3.31}$$

$$\frac{\partial \overline{\rho}\widetilde{u}_i}{\partial t} + \frac{\partial}{\partial x_j}(\overline{\rho}\widetilde{u}_i\widetilde{u}_j) = -\frac{\partial}{\partial x_j}[\overline{P}\delta_{ij} - \overline{\tau}_{ij} - \overline{\tau}^t_{ij}] + \dot{F}_{p,i} \tag{3.32}$$

$$\frac{\partial \overline{\rho}\widetilde{E}}{\partial t} + \frac{\partial}{\partial x_j}(\overline{\rho}\widetilde{E}\widetilde{u}_j) = -\frac{\partial}{\partial x_j}[\overline{u_i(P\delta_{ij} - \tau_{ij})} + \overline{q}_j + \overline{q}^t_j] + \overline{\dot{\omega}}_T + \overline{Q}_r + \dot{Q}_p \tag{3.33}$$

$$\frac{\partial \overline{\rho}\widetilde{Y}_k}{\partial t} + \frac{\partial}{\partial x_j}(\overline{\rho}\widetilde{Y}_k\widetilde{u}_j) = -\frac{\partial}{\partial x_j}[\overline{J}_{j,k} + \overline{J}^t_{j,k}] + \overline{\dot{\omega}}_k + \dot{S}_{p,k}. \tag{3.34}$$

These source terms coming from two-way coupling are computed, as detailed elsewhere (Oefelein & Yang [144], Faeth [54], Sankaran & Menon [189], Faeth [55], Chen & Shuen [31], Pannala & Menon [148]). In this thesis, neither evaporation nor combustion are considered, therefore all terms with variations in the particle mass are assumed to be zero ($\dot{m}_p = 0$). This leads directly to $\dot{\rho}_p = 0$, $\dot{S}_{p,k} = 0$ for mass and species coupling source terms. The momentum and energy coupling source terms are then numerically implemented at a grid node m as:

$$\dot{F}_{p,i} = \sum_{n=1}^{N_p} \frac{\rho_p^n V_p^n}{V_m} f_i^n W(x_i^n, x_i^m) \tag{3.35}$$

$$\dot{Q}_p = \sum_i \dot{F}_{p,i} u_{p,i}, \tag{3.36}$$

where N_p is the number of particles, V_m is the fictitious volume surrounding each grid node m, f_i^n is the fluid-particle interaction force (taken into account through drag force) on a single particle located at x_i^n and W is the projection weight of the force onto the grid node m.

Figure 3.17 sketches the coupling force exerted by a particle on the fluid and its projection onto the grid nodes. The weights in the projection operation are constructed to be inversely proportional

to the distances d_j between the particle and the nodes of the cell containing the particle:

$$W_j = \frac{1/d_j}{\sum_{k=1,N} 1/d_k} W_p, \tag{3.37}$$

where j is the index of the cell vertex and N is the number of vertices of the cell.

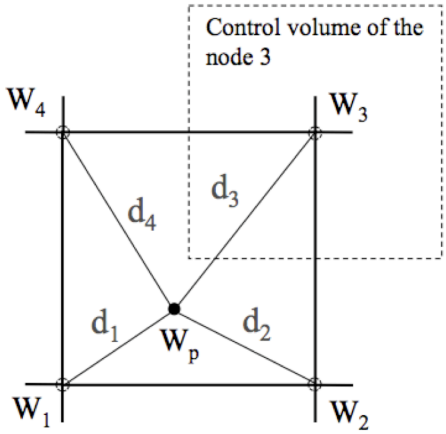


Figure 3.17 - *Example of projection of a particle two-way coupling contribution into the vertices of a quadrilateral cell.*

3.6.2 Validation of two-way coupling implementation

A simple numerical validation of two-way coupling is presented below. A set of particles are introduced with an initial constant velocity ($u_p(0) = 1$ m/s) in a fluid at rest ($u_g(0) = 0$ m/s). Thus, the system of equations that need to be solved for the gas and particle phases is:

$$\frac{du_{g,i}}{dt} = -\frac{1}{(\Delta x)^3}\frac{m_p}{\rho_g}(u_{g,i} - u_{p,i}) \tag{3.38}$$

$$\frac{du_{p,i}}{dt} = \frac{1}{\tau_p}(u_{g,i} - u_{p,i}), \tag{3.39}$$

with τ_p given by Eq. (3.21). Considering as a first approximation that Re_p is constant, the analytical solution of Eqs. (3.38)-(3.39) is:

$$u_{g,i} = u_{p,i}(0)\frac{m_p}{m_p + \rho_g(\Delta x)^3}\left[1 - \exp\left(-\frac{1}{\tau_p}\left(\frac{m_p + \rho_g(\Delta x)^3}{\rho_g(\Delta x)^3}\right)t\right)\right] \tag{3.40}$$

$$u_{p,i} = u_{p,i}(0)\frac{1}{m_p + \rho_g(\Delta x)^3}\left[m_p + \rho_g(\Delta x)^3 \exp\left(-\frac{1}{\tau_p}\left(\frac{m_p + \rho_g(\Delta x)^3}{\rho_g(\Delta x)^3}\right)t\right)\right]. \tag{3.41}$$

The aim of this test is to verify the conservation of the total momentum of the gas/particles system, i.e.,

$$n_p m_p u_p + \rho_g u_g V_m = constant = n_p m_p u_p(0). \tag{3.42}$$

Figure 3.18 (a) shows the computational domain containing 10^3 hexahedra cells. One particle is introduced in the center of each cell with velocity in the x-direction only. Momentum conservation is satisfied as observed in Fig. 3.18 (b). Particles are decelerated by the fluid, which is in turn accelerated by the particles so that Eq. (3.42) is verified at each time t.

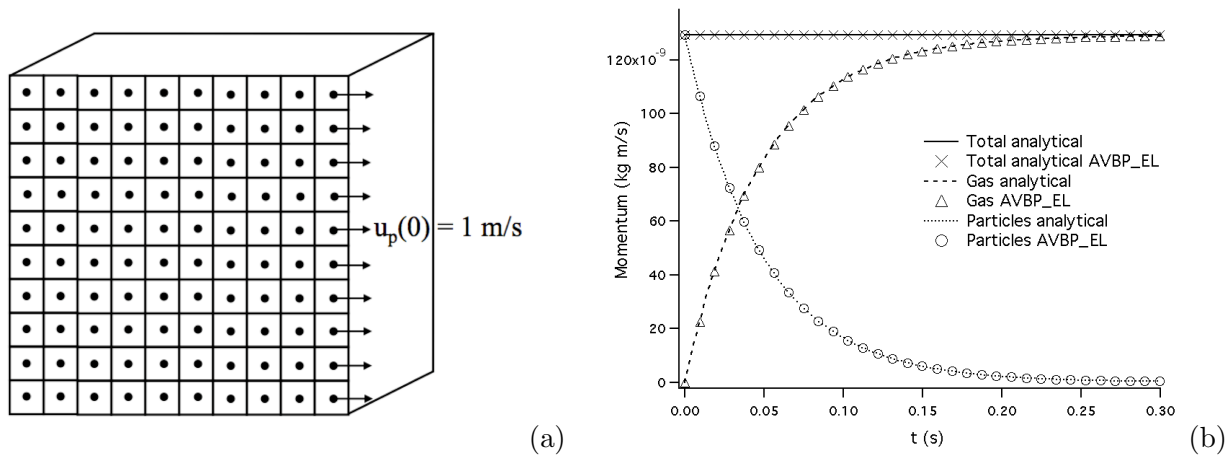


Figure 3.18 - *(a) Mesh used to validate the two-way coupling; one particle is located inside each cell. (b) Momentum conservation versus time.*

3.7 Boundary conditions

The main goal of this thesis was to write and validate the general structure of the Lagrangian module; we did not implement a general treatment of special boundary conditions such as outlets, symmetry, periodicity or particle-wall interactions. In the current status of the code and for all applications presented in this dissertation, particles leaving the outflow section are simply deleted from particle arrays. However, when the particular physics or geometry requires symmetry, periodicity and particle-wall interactions (rebound) to be treated, particle position and velocity directions are modified to account for such conditions. Although this approach is not general, a simple way to integrate these new features efficiently is presented in Appendix A. This suggestion has been retained by the PhD student in charge of this development and will be part of the official Lagrangian version at the beginning of 2009.

The following section describes the inlet boundary condition developed to model particle injection. The final objective is the application to piston engine and gas turbine simulations where the fuel is injected in the combustion chamber at each time step. The developments included in this section focus on the input parameters description. Two types of injection geometries implemented during this thesis are briefly presented in the following. These developments complete the simple case where particle parameters are read from an input data file.

3.7.1 Particle injection

One of the first questions linked to injection models is the choice of the geometry to inject particles. The current developments are detached of usual treatment of AVBP boundary conditions in the sense that the injection geometry is not included within the mesh files. The aim is to keep a certain independence from mesh generation to speed up the dedicated developments introduced by new PhD students, in particular the improvement of this simple injection model via the introduction of new geometries, new particle size distributions and multi-injection capabilities. In the version described to date, the user introduces a set of parameters to determine the injection type and position, then, the solver detects the rest of information needed to start the calculation.

To determine the number of particles N_p that have to be injected per time step two main input parameters are necessary: the particle mass-flow rate $\dot{m}_{p,inj}$ and the type of particle size distribution. In practice, the number of particles injected is obtained from Eq. (3.43):

$$\dot{m}_{p,inj}\,\Delta t = \sum_{n=1}^{N_p} \rho_p^n\,\frac{4}{3}\pi r_{p,n}^3 \tag{3.43}$$

In monodisperse cases, where particles diameter is constant, N_p is then simply given by:

$$N_p = \mathrm{Int}\left\{\frac{\dot{m}_{p,inj}\,\Delta t}{\rho_p\,\frac{4}{3}\pi r_p^3}\right\} \tag{3.44}$$

where r_p is the value of the particle radius introduced as input parameter. In polydisperse cases, where a wide range of particle sizes exists, the number of particles to be injected depends on the particle size distribution and particle diameters are calculated depending on the distribution chosen (see below). Particles mass is added until the stored mass of all particles is smaller than the mass of a single particle. In both cases, the remaining mass is added in the next time step.

Once the number of particles is known, other input parameters help to the calculation of particle variables (position, velocity, etc). Depending on the injection geometry and the particle size distribution, a different set of input parameters are specified. They are briefly explained hereafter:

Injection geometry

- **Point injection: all droplets are injected at the same point.**

  Other parameters specified by the user at the beginning of the simulation are the coordinates of the point where particles are injected and the injection direction. Particle velocity is calculated as the sum of a mean velocity and a random fluctuating component whose amplitude (RMS value) is given as a fraction of the mean velocity. This fluctuation is given as a simple white noise, no turbulence injection for particle phase velocity has been implemented so far. Particle position is then reconstructed with all these data (the algorithm for detection of the cell and the subdomain where the particle is injected has been explained in subsection 3.4.2).

- **Disk injection: droplets are injected over a disk.**

  The following geometry uses the same principle than the previous one but with the difference that a radius and a thickness are specified to delimit disk boundaries for particle injection. Figure 3.19 (a) shows a front view of a disk with 10^4 particles injected.

In both cases, particle temperature and state (tracers, solid particles are droplets) are directly read as input parameters and for both geometries, different particle size distributions can be used: constant and the typical log-normal distribution with one or two peaks.

Particle size distribution

The most general definition of the spread of the particle size distribution is monodisperse or polydisperse. In Lagrangian (or Eulerian) injection modeling, one of the key issues is the capability to

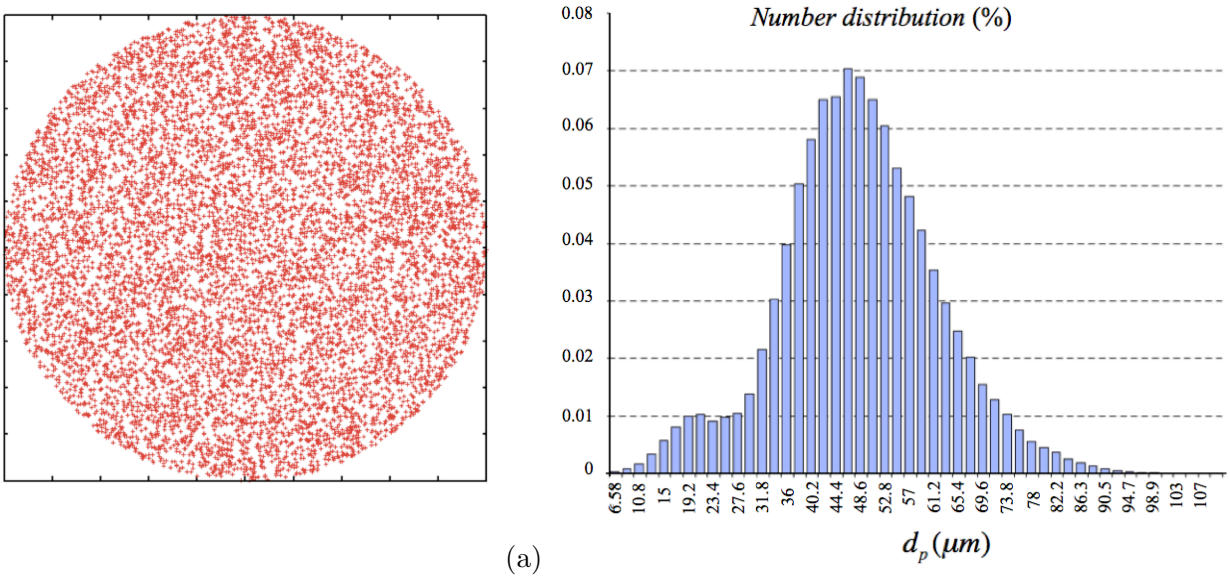


Figure 3.19 - *(a) Front view of* 10^4 *particles injected in a disk geometry. Each cross corresponds to a particle. (b) Particle log-normal distribution with two peaks.*

fit the experimental data sufficiently well, especially in the injection area in order to be as close as possible of the real phenomena.

The distributions of particle or droplet size are usually formulated in terms of particle number or particle mass and the parameters used to describe the distribution are the mean, median, mode and variance[4]. Example distribution functions are the traditional Gaussian, log-normal and Rosin-Rammler [139] (a special case of the Nukiyama-Tanasawa function) distributions and the more recently developed log-hyperbolic distribution. There are numerous other distributions which have been developed for specific purposes. The reader is referred to the abundant literature in this area (e.g., Crowe *et al.* [35], Lefebvre [112]). Below, only the log-normal distribution is presented since is the one used in one of the application tests of this thesis.

- **Log-normal distribution**

  The log-normal distribution is frequently used to represent the size of solid particles. It derives from the normal or Gaussian distribution and is defined as

$$f(x;\mu,\sigma) = \frac{1}{\sqrt{2\pi}\sigma x}\exp\left[-\frac{1}{2}\left(\frac{ln(x)-\mu}{\sigma}\right)^2\right] \tag{3.45}$$

  where σ is the standard deviation and μ is the mean. The log-normal distribution has been used for a long time to describe size distributions of particle properties in atmospheric aerosols, clouds, precipitation and chemical processes. An example of particle number distribution obtained with a log-normal distribution with two peaks is presented in Fig. 3.19 (b) and is used to match the injection profile of the polydisperse case described in Chapter 5 (see Fig. 5.4 (b)).

[4] The variance is a measure of the spread of the distribution; a large variance implying a wide distribution of sizes.

Chapter 4

Application to a homogeneous isotropic turbulence

« Big whorls have little whorls,
that feed on their velocity;
and little whorls have lesser whorls,
and so on to viscosity
(in the molecular sense). »

Lewis Fry Richardson
Weather prediction by numerical processes (1922)

The mixing and dispersion of particles are important subjects of interest in fluid mechanics, not only from an academic point of view but also in an industrial context. The dynamics of particles in turbulence are studied in a wide range of applications, from the atmospheric transport of pollutants, to transport of particles in coal power plants, from the dispersion in gas turbines to fluidized beds in chemical industry.

Two of the basic processes that characterise particle/turbulence interactions are the dispersion of particles by turbulence and the modification of turbulence by particles. The former is usually studied assuming that the properties of the turbulent flow field are not modified by the presence of the particle (one-way coupling). The latter has been studied experimentally (Snyder & Lumley [205], Wells & Stock [226]) and recently by numerical simulation which have the advantage that very detailed statistics can be easily obtained from the trajectory of each particle (Squires & Eaton [207], Elghobashi & Truesdell [51], Deutsch [42], Elghobashi & Truesdell [52], Février *et al.* [61], Moreau [135]). Theoretical analysis on particle dispersion have been initiated by Tchen [216] among others (Reeks [173, 174]), leading to a definition of time and length scales that characterise the behaviour of the particles in turbulence. Analytical methods developed by Maxey [128] were able to predict the well-known phenomenon of preferential concentration, i.e., the accumulation of inertial particles in regions of low vorticity and high strain. These phenomena have been studied by several authors using the Lagrangian approach (Deutsch [42], Elghobashi & Truesdell [52], Février *et al.* [61], Moreau [135]).

In this chapter, the results of a DNS of homogeneous isotropic turbulence (HIT) using the Lagrangian module developed in this thesis are validated by comparison with the results of another Lagrangian solver, NTMIX3D. The code NTMIX3D is a finite difference, fully parallel two-phase flow solver. The gas-phase solver has been mainly developed at CERFACS by Baum & Poinsot [11], Poinsot *et al.* [162], Stoessel *et al.* [210], Cuenot & Poinsot [36], Boughanem & Trouvé [24] whereas the two-phase flows have been investigated by Albrecht *et al.* [2], Vermorel *et al.* [220], Paoli *et al.* [151], Moreau [135], Paoli & Shariff [153] and others. For the gas phase, the fully compressible Navier-Stokes equations are solved with a non-dimensional formulation. Space discretization is performed by the sixth-order compact scheme Lele [114] while time integration is performed by a three-stage Runge-Kutta method. For the dispersed phase, the transport equation for particle trajectories are advanced in time with a three-stage Runge-Kutta method as the gas phase. A fourth-order interpolation of the Lagrangian polynomials provides the fluid velocity sampled along particle trajectories, that is required for drag force computation, condensation/evaporation source terms, etc.

The code NTMIX3D is a pure structured DNS, high-fidelity solver. Comparing the results of AVBP, which is built for LES, to this DNS code is an interesting exercise. Results will show that the third-order spatial scheme (TTGC) performs perfectly well for this test case, providing results which are close to the sixth-order scheme of NTMIX3D. Results will also highlight the effects and importance of the interpolation algorithms for the dispersed phase.

The chapter is organised as follows: Section 4.1 reviews some basic notions about turbulence. Section 4.2 presents the main categories in which the HIT flows are classified. Section 4.3 summarises the theoretical basis of the HIT and particle dispersion quantities that will be used to validate the Lagrangian module. Section 4.4 presents the main parameters used for the single-phase computation and the energy spectrum obtained before introducing the dispersed phase. The results of the two-phase test case are presented in Section 4.5. Last section (Section 4.6) contains a summary of the main conclusions.

4.1 Notions of turbulence

The notion of turbulence is generally accepted nowadays but its rigorous definition is not so simple. Hinze [88] proposed the following: "Turbulent fluid motion is an irregular condition of flow in which the various quantities show a random variation with time and space coordinates, so that statistically distinct average values can be discerned". An important characteristic of turbulence is its ability to transport and mix fluid much more efficiently than a comparable laminar flow. This is well demonstrated by an experiment first reported by Osborne Reynolds [177]. Dye is injected on the centerline of a long pipe in which water is flowing to study the stability of flow in a tube. Figure 4.1 shows a sequence of photographs of the repetition of Reynolds' dye experiment performed by N. H. Johannesen and C. Lowe a century later. The experiment was repeated with the original apparatus, still present at the University of Manchester. In laminar flow a filament of colored water introduced at a bell-shaped entry extends undisturbed the whole length of the glass tube. Transition is seen in the second of the photographs as the speed is increased; the last two photographs show fully turbulent flow.

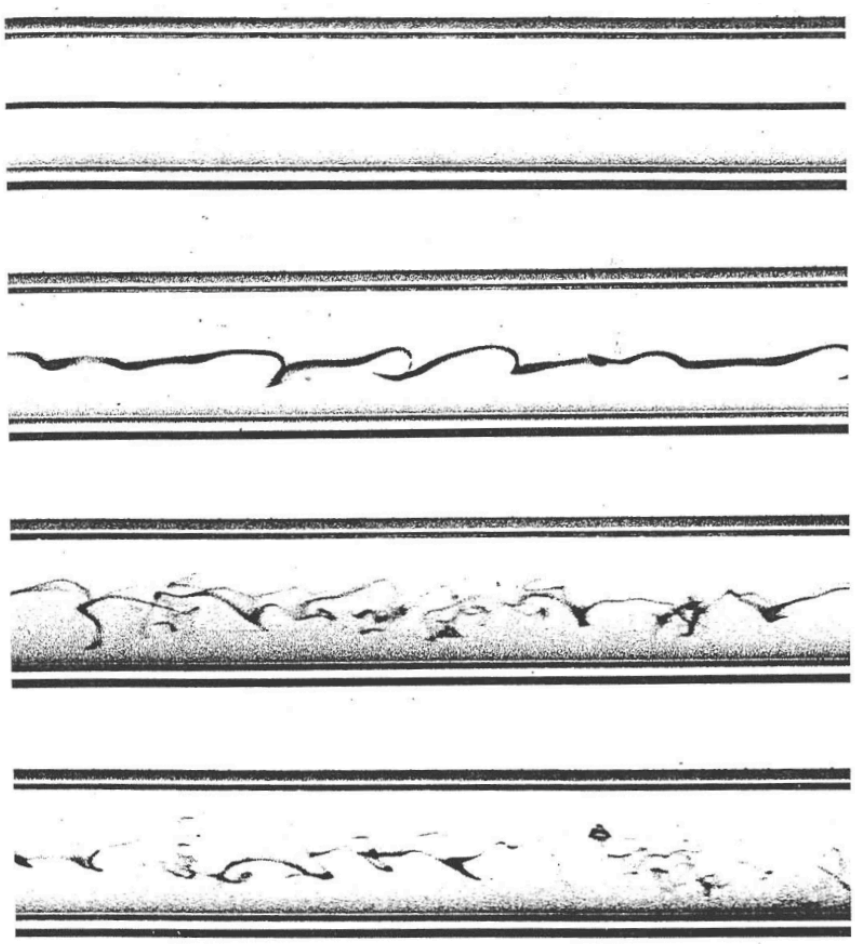


Figure 4.1 - *Sequence of photographs of the repetition of Reynolds' dye experiment by N. H. Johannesen and C. Lowe. Transition is seen in the second of the photographs as the speed is increased; and the last two photographs show fully turbulent flow. From Van Dyke [218].*

As Reynolds [178] later established, this flow is characterised by a single non-dimensional parameter, now known as the Reynolds number, Re:

$$Re = \frac{u_g \, l_g}{\nu_g}, \tag{4.1}$$

where u_g and l_g are characteristic velocity and length scales of the flow, and ν_g is the kinematic velocity of the fluid.

Laminar flows are characterised by low Reynolds numbers. They are generally steady in time and regular in space. Destabilisation of the flow happens when the Reynolds number increases, leading to turbulent flows. Those are unsteady and very chaotic in space. The limit between laminar and turbulent flows is difficult to define since other parameters may influence the transition from laminar to turbulent flows. In Reynolds' experiment, if Re is less than about 2300, the flow is laminar. If, on the other hand, Re exceeds about 4000, then the flow is turbulent[1].

[1] As the Reynolds number is increased, the transition from laminar to turbulent flow occurs over a range of Re, and this range depends on the details of the experiment.

As previously mentioned, turbulent flows must satisfy the condition of irregularity. Variables like velocity, temperature or pressure have a random behaviour and can be decomposed into an ensemble average and a fluctuating part. For velocity, it reads:

$$U = \overline{U} + u, \quad \text{with } \overline{u} = 0. \tag{4.2}$$

Since Dryden & Kuethe [45], the definition of the intensity of the turbulence fluctuations has been given by the root-mean-square (r.m.s.) value:

$$u' = \sqrt{\overline{u^2}}. \tag{4.3}$$

In addition, to describe a turbulent motion quantitatively, it is necessary to introduce the notion of *scale of turbulence*: a certain scale in time and a certain scale in space of the turbulent eddies that characterised such flows. The size of the largest scale turbulent eddies, also referred to as integral length-scale $l_{g,t}$, is mainly linked to the geometry and the boundary conditions. The turbulent energy produced by large eddies is characterised by the turbulent Reynolds number:

$$Re_t = \frac{u'_g \, l_{g,t}}{\nu_g}. \tag{4.4}$$

The turbulent time-scale of the largest scale turbulent eddies then yields:

$$\tau_{g,t} = \frac{l_{g,t}}{u'_{g,t}}. \tag{4.5}$$

Richardson [181] introduced the idea of the *energy cascade* in 1922. This notion says that the kinetic energy enters the turbulence (through the production mechanism) at the largest scales of motion. Then, this energy is transferred (by inviscid processes) to smaller and smaller scales until, at the smallest scales, the energy is dissipated by viscous action. Kolmogorov [103] added to and quantified this picture. In particular, he identified the smallest scales of turbulence to be those that now bear his name. The dissipation ε_g scales independently of ν_g as:

$$\varepsilon_g = \frac{u'^2_{\varepsilon_g}}{\tau_{\varepsilon_g}}, \tag{4.6}$$

where $\tau_{\varepsilon_g} = l_{\varepsilon_g}/u'_{\varepsilon_g}$ and u'_{ε_g} is the velocity fluctuation of the eddies of length l_{ε_g}. The Kolmogorov length, velocity and time scales characterise the smallest and the dissipative scales and are:

$$\eta_k \equiv (\nu_g^3/\varepsilon_g)^{1/4}, \tag{4.7}$$

$$u'_k \equiv (\nu_g \varepsilon_g)^{1/4}, \tag{4.8}$$

$$\tau_k \equiv (\nu_g/\varepsilon_g)^{1/2}. \tag{4.9}$$

The Reynolds number based on the Kolmogorov scales is unity:

$$Re_k = \frac{u'_k \, \eta_k}{\nu_g} = 1. \tag{4.10}$$

By non-dimensional analysis one can easily relate the integral and the Kolmogorov scales as a function of the turbulent Reynolds number as follows:

$$\frac{l_{g,t}}{\eta_k} = Re_t^{\frac{3}{4}}, \tag{4.11}$$

$$\frac{u'_{g,t}}{u'_k} = Re_t^{\frac{1}{4}}, \tag{4.12}$$

$$\frac{\tau_{g,t}}{\tau_k} = Re_t^{\frac{1}{2}}. \tag{4.13}$$

$$\tag{4.14}$$

4.2 Choice of the configuration: particles in decaying turbulence

A field is said to be statistically homogeneous when all statistics are invariant under translations. If the field is also statistically invariant under rotations and reflexions of the coordinate axes, it is besides isotropic. When both conditions are satisfied the flow is referred to as Homogeneous Isotropic Turbulence (HIT) and represents one of the fundamental class of flows that any Direct Numerical Simulation should reproduce. One reason is that it allows to identify and quantify all the turbulent scales previously defined. Another reason is that since this turbulent flow has no boundaries and no preferential direction it is an ideal case for testing turbulent models and subgrid-scale closures for RANS and LES. Within the first works are those of Taylor [214] and Batchelor [10], who conducted theoretical studies about the diffusion of passive scalars in stationary HIT. Since then, a large amount of theoretical studies has been achieved to increase the knowledge in turbulence structures, and in mechanisms of turbulence dissipation and production. Numerical investigations of HIT flows are more recent (1970's), and closely related to the development of powerful computers. In addition, it is a case that allows the study of the dynamics of a coupled gaseous-dispersed phase system.

Although it is an academic problem, mechanisms induced in such two-phase flows arise in a wide variety of industrial applications and therefore the study of particle behaviour in such a case is mandatory before the simulation of complex geometries. In this section, the main categories in which these model flows can be classified are presented:

- **Decaying homogeneous isotropic turbulence**

  This reflects the natural behaviour of the Navier-Stokes equations and no artificial forcing terms are needed. However, this test has the drawback of being not statistically stationary and therefore particle relaxation times need to be chosen carefully, so that the decreasing behaviour of the carrier phase turbulence does not alter the behaviour of the points of interest. One advantage of this setup is that it can be compared to experiments of spatially decreasing grid turbulence. This approach has been used together with Lagrangian particle tracking to determine particle dynamics and particle dispersion (Squires & Eaton [207], Elghobashi & Truesdell [51]).

- **Forced homogeneous isotropic turbulence**

  In this setup the most energetic length-scales are forced (usually) in spectral space (Hunt *et al.* [90]). This setup is well suited to study temporally independent quantities, since

the turbulence is statistically stationary. This approach has been extensively used to study particles in turbulent flows (Février & Simonin [60], Laviéville [109], Deutsch [42], Lundgren [124], Paoli & Shariff [152]).

- **Kinematic simulation (KS)**

  This approach is sometimes called artificial (or cheap) turbulence, since it only provides a divergence-free flow field with the spectral kinetic energy characteristic of the Navier-Stokes equations [66]. The advantage of this approximation is its low numerical cost. Its biggest drawback is that the flow field does not satisfy the complete Navier-Stokes equations. As pointed out by Maxey [128] triple correlations vanish and there is no representation of the energy transfer from large scales to dissipative small scales. Furthermore, it lacks the temporal advection of small scale turbulent motion by large eddies.

In the rest of this chapter, we choose the approach of decaying homogeneous isotropic turbulence. In the present case, the regime is dilute enough to assume one-way coupling and the fluid flow is independent of the particle motion.

4.3 Homogeneous Isotropic Turbulence statistics

4.3.1 Analysis in the physical space

The starting point is the observation that all statistics are independent of the location and orientation of the coordinate system. All the statistical quantities are therefore only a function of the distance r, separating the locations observed. The correlation function of the velocity vectors at points A and B separated by the distance r reads:

$$Q_{g,ij}(r) = \overline{u'_{g,i}(A)\ u'_{g,j}(B)}. \tag{4.15}$$

where $u'_{g,i}(A)$ is the ith component of the fluctuating velocity vector at the location A and the over-bar stands for the ensemble mean which is equivalent to a spatial mean under the assumption of homogeneity. The correlation coefficient is defined as:

$$R_{g,ij}(r) = \frac{Q_{g,ij}(r)}{{u'_g}^2}, \tag{4.16}$$

where ${u'_g}^2 = \frac{1}{3}\ \overline{u'_{g,i}u'_{g,i}} = \frac{1}{3}\ Q_{g,ii}(0) = \frac{2}{3}q_g^2$, where q_g^2 is the turbulent kinetic energy.

Based on Eq.(4.16), the longitudinal, transversal and cross-stream integral length-scales are respectively defined as:

$$L^1_{g,11} = \int_0^\infty R_{g,11}(r,0,0)\ dr, \tag{4.17}$$

$$L^2_{g,22} = \int_0^\infty R_{g,22}(0,r,0)\ dr, \tag{4.18}$$

$$L^3_{g,33} = \int_0^\infty R_{g,33}(0,0,r)\ dr. \tag{4.19}$$

$L^i_{g,ii}$ is a characteristic length of the large scale motions in the turbulent field and is of the same order as the turbulent length-scale $l_t = {u'_g}^3/\varepsilon_g$, with ε_g being the turbulent kinetic energy dissipation rate. The Reynolds number based on $L^i_{g,ii}$ reads:

$$Re_{L^i_{g,ii}} = \frac{u'_g L^i_{g,ii}}{\nu_g}. \tag{4.20}$$

For an incompressible fluid, isotropy imposes:

$$L^1_{g,11} = L^2_{g,22} = L^3_{g,33}. \tag{4.21}$$

4.3.2 Turbulent kinetic energy temporal decaying

Eq. (4.22) defines the turbulent kinetic energy q_g^2 and the dissipation rate ε_g:

$$q_g^2 = \frac{1}{2}\overline{u'_{g,i}u'_{g,i}}, \qquad \varepsilon_g = \nu_g \overline{\frac{\partial u'_{g,i}}{\partial x_j}\frac{\partial u'_{g,j}}{\partial x_i}}. \tag{4.22}$$

If the Navier-Stokes equations are restricted to incompressible flows, those quantities verify two exact equations. For this study, the exact equation, Eq. (4.23), is used for q_g^2 whereas a modeled one, Eq. (4.24), is considered for ε_g.

$$\frac{dq_g^2}{dt} = -\varepsilon_g \tag{4.23}$$

$$\frac{d\varepsilon_g}{dt} = -C_{\varepsilon_g}\frac{\varepsilon_g^2}{q_g^2}, \tag{4.24}$$

where C_{ε_g} is a constant determined experimentally. For high Reynolds number flows, its value is $C_{\varepsilon_g} = 1.96$. There is an analytical solution to this system. The change of variable $\tau_{\varepsilon_g} = q_g^2/\varepsilon_g$ determines the temporal evolution of τ_{ε_g} itself, q_g^2 and ε_g among others:

$$\tau_{\varepsilon_g} = \tau_{\varepsilon_g,0}\left(1 + (C_{\varepsilon_g} - 1)\frac{t}{\tau_{\varepsilon_g,0}}\right), \tag{4.25}$$

$$q_g^2 = q_{g,0}^2\left(1 + (C_{\varepsilon_g} - 1)\frac{t}{\tau_{\varepsilon_g,0}}\right)^{-\frac{1}{C_{\varepsilon_g}-1}}, \tag{4.26}$$

$$\varepsilon_g = \varepsilon_{g,0}\left(1 + (C_{\varepsilon_g} - 1)\frac{t}{\tau_{\varepsilon_g,0}}\right)^{-\frac{C_{\varepsilon_g}}{C_{\varepsilon_g}-1}}, \tag{4.27}$$

where the subscript 0 denotes the initial field. By using these equations, the temporal evolutions of u'_g, l_t and Re_t can then be determined.

4.3.3 Analysis in the spectral space

In Fourier space, or spectral space, a wavenumber k is associated to each length scale l, by $k = 2\pi/l$ [88]. The energy is usually represented with the fluid kinetic energy spectrum or spectral density of

the turbulent kinetic energy $E_g(k,t)$, as a function of the wavenumber and time. There are different expressions for $E_g(k,t)$ and they depend on the domain of validity. Most of them constitute a solution of the exact equation of the spectral density for a particular wavenumber k through:

$$\frac{\partial}{\partial t}E_g(k,t) = T_g(k,t) - 2\nu_g k^2 E_g(k,t) \tag{4.28}$$

where $T_g(k,t)$ is a term that accounts for the interaction between eddies and represents the energy transfer from the largest scales to the smallest ones. Integrating this equation over the wavenumber from zero to infinity gives the equation of the evolution of the turbulent kinetic energy, already defined in Eq. (4.23) [2]:

$$\frac{dq_g^2}{dt} = -\varepsilon_g. \tag{4.29}$$

Two turbulence spectra are available in the AVBP solver to model the turbulent kinetic energy:

- **The Passot-Pouquet spectrum [155] (PP)**

  The spectrum proposed by Passot-Pouquet only considered the largest scales of the turbulence and it does not characterise the smallest scales, i.e., it is only valid for small wavenumbers, $k \ll k_d$, with $k_d = 2\pi/\eta_k$ representing the dissipation scale. Only the big structures are represented and therefore do not require a fine mesh resolution:

  $$E(k) = A\left(\frac{k}{k_e}\right)^4 \exp^{-2(k/k_e)^2} \quad \text{with } A = 16\sqrt{\frac{2}{\pi}}\frac{u'^2}{k_e}, \tag{4.30}$$

  where A is the amplitude of the spectrum, k is the wavenumber and $k_e = 2\pi/l_e$ corresponds to the most energetic wavenumber.

- **The Von Kármán spectrum rectified by Pao [149, 150] (VKP)**

  This spectrum allows a description of both the largest and smallest turbulent scales. It uses explicitly u', ε, k_e and k_d and has the form:

  $$E(k) = \frac{Au'^5}{\varepsilon}\frac{\left(\frac{k}{k_e}\right)^4}{\left(1+\left(\frac{k}{k_e}\right)^2\right)^{17/6}}\exp\left(-\frac{3}{2}\gamma\left(\frac{k}{k_d}\right)^{4/3}\right) \tag{4.31}$$

  with the constants of the model A and γ equals to 1.5.

More information about a parametric study of these spectra for different values of k/k_e can be consulted in the dissertation of Boughanem [23].

4.3.4 Properties of the Lagrangian field

Using the Lagrangian equations of particle transport with Stokes drag (Eqs. (3.1) and (3.2)) a corresponding set of ordinary differential equations for the fluid-particle correlation,

$$q_{gp} = \frac{1}{N_p}\sum_{i=1}^{N_p}\tilde{u}_{g,i}\ u_{p,i} \tag{4.32}$$

[2] The integral over the spectral space of the term $T_g(k,t)$ is zero, it neither produces nor dissipates energy.

and the particle kinetic energy,

$$q_p^2 = \frac{1}{2}\frac{1}{N_p}\sum_{i=1}^{N_p} u_{p,i}\ u_{p,i} \tag{4.33}$$

can be obtained for the dispersed phase and the fluid-particle correlation (where N_p is the total number of particles) and $\tilde{u}_{g,i}$ is the fluid velocity interpolated at particle position, $x_{p,i}$.

Using the definition of the Lagrangian particle velocity the particle dispersion coefficient can be related to the Lagrangian velocity autocorrelation [207] by

$$R^L_{p,ij}(t_1,\tau) = \frac{\overline{u_{p,i}(t_1)\ u_{p,j}(t_1+\tau)}}{(\overline{u_{p,i}(t_1)^2})^{1/2}\ (\overline{u_{p,i}(t_1+\tau)^2})^{1/2}}, \tag{4.34}$$

where t_1 corresponds to a time in the simulation following the peak in the mean-square relative velocity [207].

4.4 Single-phase test case

All simulations for this study were performed with the same grid resolution on a 64^3 uniform grid, the length of the computational domain is $2\pi\ 10^{-3}$ m in the three directions and the cell size is then $\Delta x = 98.1719\ 10^{-6}$ m. Boundary conditions are periodic in all directions. Parallel simulations were performed on a Compaq AlphaServer SC. The main characteristics of the grid are summarised in Table 4.1. All simulations were obtained with the TTGC scheme.

N^o of space dimensions	3
N^o of nodes	274,625
N^o of cells	262,144
Element type	Hexahedra
Length of domain (m)	$2\pi\ 10^{-3}$
Grid spacing (m)	$98.1719\ 10^{-6}$

Table 4.1 - *Characteristics of the mesh used for the HIT simulations.*

4.4.1 Gaseous initial conditions

The initial flow field ($t = 0$ s) has been obtained by first initialising the velocity field with a divergence-free random field, so that the associated kinetic energy $E(k,t)$ follows a Passot-Pouquet spectrum [155] (see Eq. (4.30))[3]. The parameters to initialise the spectrum are: $l_e = 2.2\ 10^{-3}$ m and $u'_g = 34.7$ m/s. The Reynolds appearing in Eqs. (4.4) and (4.20) are $Re_t = 13.4$ and $Re_{L^i_{g,ii}} = 17.8$,

[3] The VKP spectrum allows to represent both the biggest and the most dissipative scales of the turbulent field, and so, is rather physical. However, it is valid only for $Re_{L^i_{ii}} > 100$, or $Re_t > 200$. Considering the grid resolution of this test case, only the PP spectrum can be used to initialise the velocity field.

respectively. In order to obtain a flow field that is solution of the Navier-Stokes equations the initial field is computed for half an eddy turnover time ($t_0 = 1.24942\ 10^{-5}$ s) before particles are introduced in the domain. The PP model spectrum introduced at $t = 0$ does not represent properly the inertial range and the dissipative scales (besides, it is not a solution of the Navier-Stokes equations). The time interval $0 < t < t_0$ is then necessary for the most energetic structures (that are well modeled by the PP spectrum) to transfer energy to the smallest ones, leading to a better energy spectrum. Figure 4.2 shows the shape of the Passot-Pouquet spectrum used to initialise the HIT ($t = 0$, dashed line) and the spectrum at particle insertion ($t = t_0$, solid line).

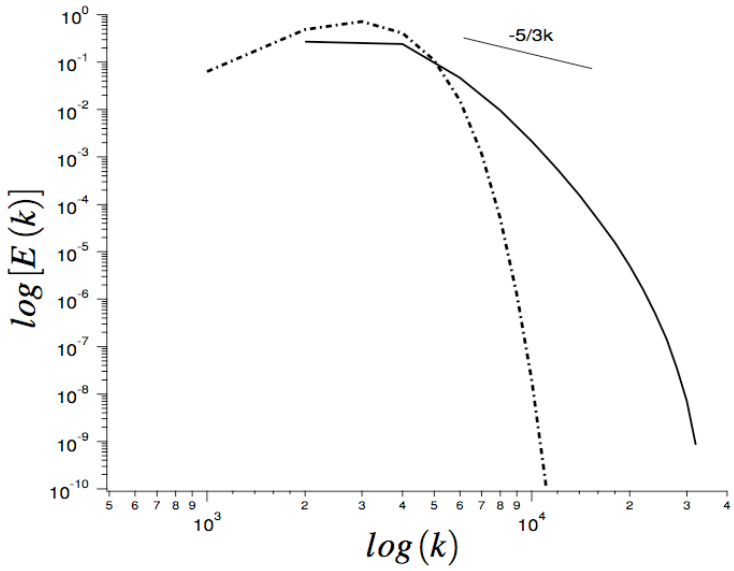


Figure 4.2 - *Spectrum of the turbulent kinetic energy of the carrier phase at time 0 (dotted-dashed line) and at the time particles are injected, t_0 (solid line).*

All simulations are conducted at ambient conditions ($T = 300$ K and $P = 1$ atm) and with a kinematic viscosity $\nu_g = 1.7147\ 10^{-3}$ m^2/s. The parameters of the Passot-Pouquet spectrum used in this case for the initial field and the field at time t_0 are presented in Table 4.2.

t	k_g (m^2/s^2)	u'_g (m/s)	ε_g (m^2/s^3)	τ_{ε_g} (s)	L_{11} (m)	L_{22} (m)	L_{33} (m)
0	1777.10	34.420	6.32 10^7	2.86 10^{-5}	8.81 10^{-4}	8.52 10^{-4}	5.71 10^{-4}
t_0	1137.21	27.534	4.64 10^7	2.45 10^{-5}	9.26 10^{-4}	9.35 10^{-4}	5.57 10^{-4}

Table 4.2 - *Statistical analysis of the gaseous field obtained numerically at times* 0 *and* $t_0 = 1.2494210^{-5}$ *s.*

We limit this simple analysis to the results of the gaseous-phase. More information about the validation of AVBP in DNS of single-phase HIT can be consulted in:

- Kaufmann [98]: where the dissipation rate and the temporal derivative of the kinetic energy are compared for different initial conditions.
- Prière [170]: where the influence of the initial conditions on the energy spectrum, the vorticity and the dissipation are studied.
- Riber [179]: where the effects of spatial resolution are analysed and the dissipative properties of the Lax-Wendroff and TTGC schemes are compared.

4.5 Dispersed-phase test case

The dynamics of the particles in homogeneous isotropic turbulence depends on the Stokes number, namely the ratio between the particle relaxation time and a characteristic time scale of turbulence. In the present study we take the turnover time τ^{ε} as the turbulence timescale, so the Stokes number is $St_{\varepsilon} = \frac{\tau_p}{\tau^{\varepsilon}}$. For all simulations the particle density is $\rho_p = 1916$ kg/m^3 and particle diameter, d_p, was set to 17.3 μm which gives $\tau_p = 1.577\ 10^{-5}$ s and $St_{\varepsilon} = 0.64$. This small value for St_{ε} indicates that the initial velocity of the particles is close to the gaseous phase velocity which proves that initialising the dispersed phase velocity field with the gaseous one is a reasonable assumption.

In all cases presented in this section, particles are homogeneously distributed in the computational domain (see Table 4.1) at the injection time and periodic boundary conditions are used in all directions. Figure 4.3 shows a snapshot (Fig. 4.3 (a)) and a 2D slice (Fig. 4.3 (b)) of the initial particle distribution. This implies that all processors share the same number of particles and therefore the number of cells and particles is naturally balanced between all subdomains with a one-constraint partitioning algorithm.

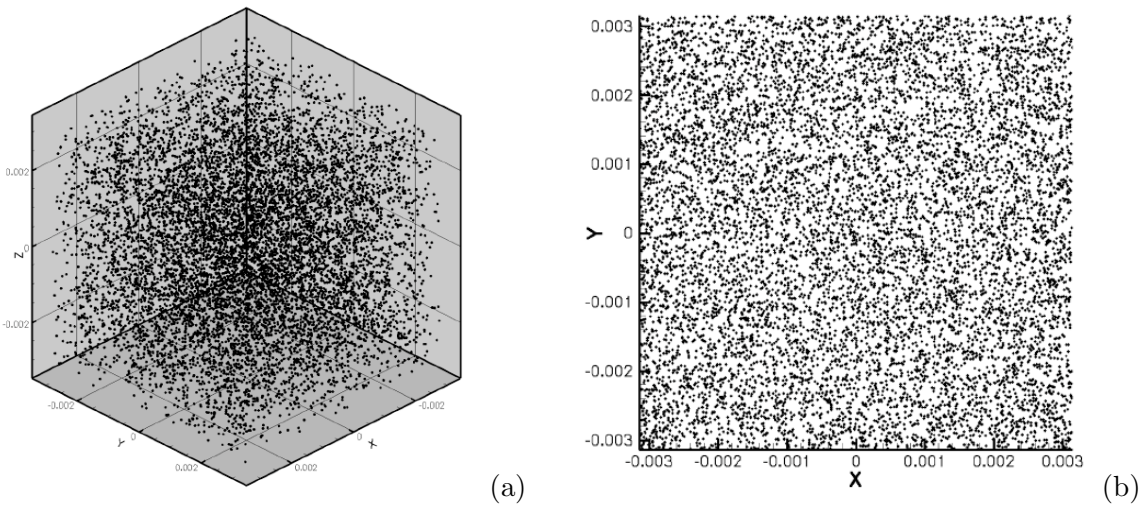


Figure 4.3 - *Snapshot (a) and 2D slice (b) of the initial particle distribution.*

The number of particles trajectories computed at each time step changes depending on the simulation, varying from 262,144 (1 particle per cell) to 2,621,440 (10 particles per cell). The main objective here is to analyse the effect of particles loading on the flow physical properties (e.g., kinetic energy spectrum), as well as on computational performance of the code (e.g., CPU time to locate particles initially).

4.5.1 Performance analysis of particle locations for the first time

In this subsection, the octree algorithm implemented to locate particles for the first time (see Subsection 3.4.1) is compared to the brute force algorithm for various numbers of particles per cell: 1, 5 and 10. The idea of using the load of particles per cell is to relate the number of particles that will be used for sampling. The higher the number of samples the more accurate the statistics obtained in a simulation (see Subsection 5.5.4). For example, 10 particles per cell is a minimum value if proper sampling must be achieved (especially for combustion).

Table 4.3 summarises the CPU time spent to search particles for the first time using the brute force and the octree algorithms for different number of processors. For the three different numbers of particles per cell, each case includes the value of the fastest processor (minimum CPU time) and the lowest processor (maximum CPU time), which is the determining factor in parallel performance.

In this initial search, each processor contains the total number of particles. The higher the number of subdomains in which the mesh is partitioned, the lower the number of cells per subdomain and therefore, the lowest the CPU time needed to locate particles with both algorithms (since the loops over cells are shorter and the memory is lower). In addition, the higher the number of particles, the higher the CPU time needed by the slowest processor to locate them. These results are summarised in Fig. 4.4 where the octree algorithm shows to be noticeably faster than the brute-force algorithm, as expected. For a given simulation, the ratio of the CPU time between the fastest and the slowest processors varies between 1.3 to 2.75 (for the brute-force algorithm) and between 2.0 to 20.63 (for the octree algorithm) which confirms the existence of internal differences in the way the search is performed. With brute-force, all cells are spanned and the time needed to find a particle depends on how far it is from the starting cell. On the other hand, with an octree this dependency is related to the root node of the tree data structure (not the starting cell) and its fast localisation may reduce considerably the search time for some processors, with explains the higher ratio.

BRUTE FORCE	**CPU time (s)**					
N^o of processors	**1 particle/cell**		**5 particles/cell**		**10 particles/cell**	
8	min	582.63	min	3134.57	min	7737.83
	max	741.53	max	6261.82	max	19,593.60
16	min	294.38	min	1564.48	min	4173.23
	max	384.41	max	3083.51	max	10,447.77
24	min	196.45	min	1025.80	min	2089.06
	max	260.31	max	2327.82	max	7221.72
32	min	149.48	min	797.75	min	2190.82
	max	195.38	max	1672.05	max	5511.89

OCTREE	**CPU time (s)**					
N^o of processors	**1 particle/cell**		**5 particles/cell**		**10 particles/cell**	
8	min	12.87	min	179.69	min	1795.05
	max	29.53	max	2717.40	max	12,000.80
16	min	10.02	min	109.96	min	1247.70
	max	18.63	max	1665.13	max	6156.74
24	min	6.95	min	31.62	min	91.12
	max	16.25	max	1267.77	max	5011.63
32	min	8.12	min	71.56	min	721.00
	max	12.44	max	868.80	max	3885.09

Table 4.3 - *Summary of the CPU time to locate particles for the first time with the brute-force (top) and the octree (bottom) algorithms as a function of the number of processors and for a different number of particles per cell (1, 5 and 10). Simulations performed on a Compaq AlphaServer SC.*

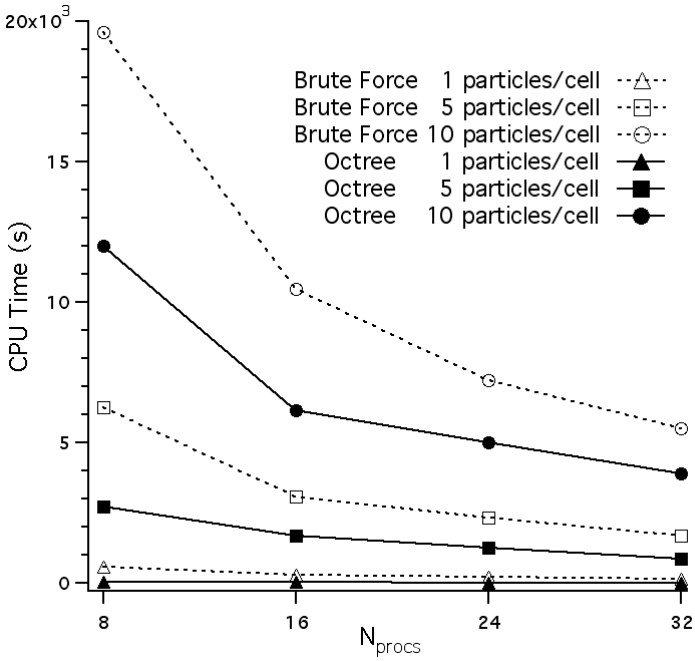


Figure 4.4 - *Comparison of the CPU time to locate particles for the first time with the brute-force (solid line) and the octree (dashed line) algorithms as a function of the number of processors and for a different number of particles per cell: 1 (triangles), 5 (squares) and 10 (circles).*

4.5.2 General results of the two-phase flow simulation

Figure 4.5 reports the temporal evolution of the kinetic energy of the gaseous phase q_f^2 (continuous line with circles), the dispersed phase q_p^2 (dashed line with squares) and the fluid-particle velocity correlation q_{fp} (dotted line with triangles)[4]. Results obtained with AVBP and NTMIX3D are compared. The initial velocity fields and particle distribution are the same for both solvers even if the order of NTMIX3D is 6 while AVBP uses a 3rd order scheme (TTGC). The carrier phase kinetic energy decreases due to viscous dissipation as expected. The figure also shows that the fluid-particle correlation closely follows the energy of the carrier phase. Due to particle inertia the particle kinetic energy decays more slowly than the carrier phase kinetic energy (or the fluid-particle correlation) with a delay of the order of the particle relaxation time. All results show an overall good agreement between the two Lagrangian solvers. No differences in the kinetic energy of the dispersed phase have been observed by increasing the number of particles per cell from 1 to 5 and 10. However, a high number of particles ensures convergence when grid filtered fields are computed from the discrete particle distribution.

For the simulation of decaying isotropic turbulence it is important that Lagrangian statistics for particles be computed only after the particles have become independent of the initial conditions. One measure of this independence is the mean-square relative velocity between fluid and particles, $\overline{(u_{g,i} - u_{p,i})^2}$. Once the mean-square relative velocity has reached a maximum value, it provides one indication that the particles are not overly influenced by the initial conditions (though it is not certain that they are completely independent). Thus, results of the correlations of fluid and particles velocities are computed after this reference value (which is $t = t_1 = 3.3913\ 10^{-5}$ s); and

[4] Lagrange polynomials are used as interpolation algorithm in the AVBP-EL simulation.

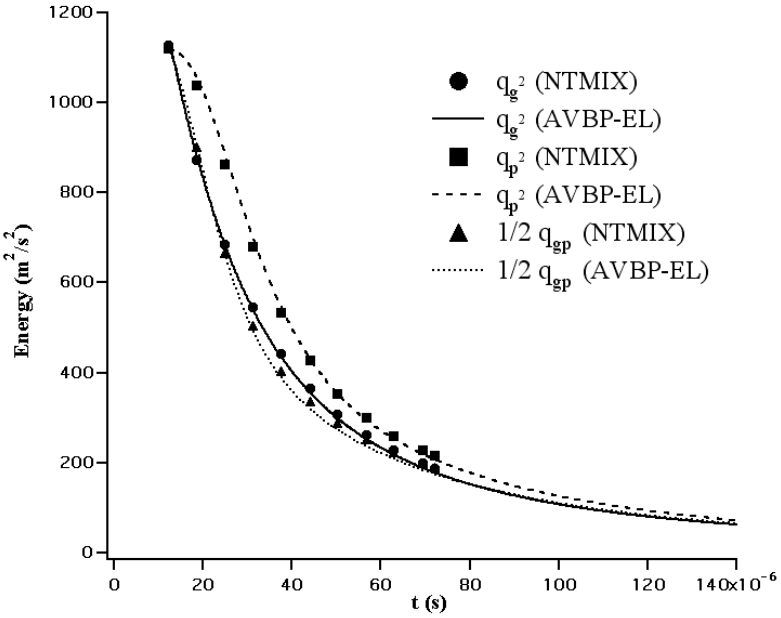


Figure 4.5 - *Temporal evolution of the fluid (triangles), particle (squares) and fluid-particle (circles) velocity correlations in an homogeneous isotropic decaying two-phase turbulence, comparison of results from NTMIX3D (symbols) and AVBP-EL (lines).*

in Fig. 4.6 we report these correlations for the average over $i = j = 1,\ 2$ and 3. As shown in the figure, at the beginning of the simulation particles keep more memory of the previous state than fluid does; i.e., they are more correlated than the fluid but after some time, particle velocities become less correlated than the fluid. The reader is referred to Squires & Eaton [207] for a complete analysis of the influence of particle inertia. In particular, it was there shown that particle velocity autocorrelations increase with particle inertia.

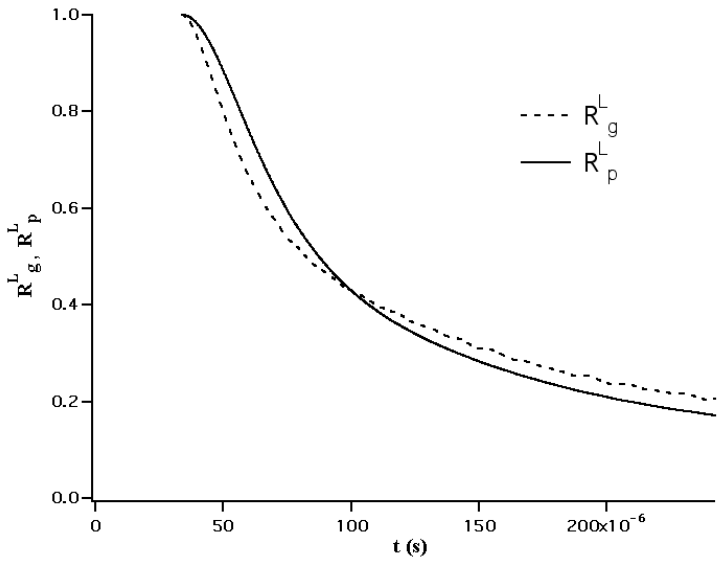


Figure 4.6 - *Autocorrelation functions of the fluid and particles with AVBP-EL. Average over $i = j = 1,\ 2$ and 3.*

4.5.3 Influence of the interpolation algorithm

Figure 4.7 shows the particle kinetic energy profiles with the three interpolation algorithms currently available in the Lagrangian module (see Section 3.5): the first-order Taylor algorithm (solid line), Lagrange polynomials (dashed line) and Linear-Least Squares (dotted line). A small difference (lower than 3%) between the Taylor algorithm and the other algorithms can be observed but differences are quickly reduced and profiles become indistinguishable after some iterations. One reason is that only the data of the closest node to the particle (and its derivatives) are used in the interpolation with Taylor series. On the other hand, Lagrange polynomials and LLS use all vertices of the cell (eight in this case) which provide more information in case of differences between values of each cell vertex. Note that since periodic boundary conditions are used for the gaseous phase there is no loss of accuracy of the interpolation when the particle is close to the boundary.

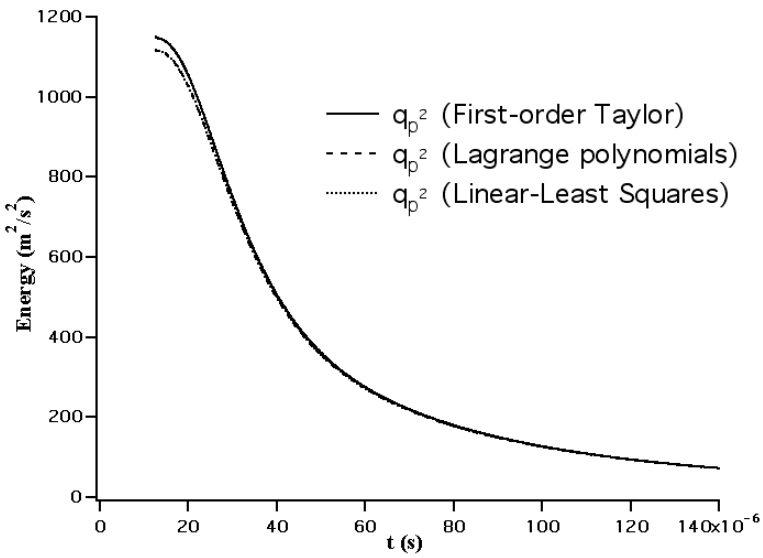


Figure 4.7 - *Temporal evolution of the particle kinetic energy for three different partitioning algorithms: First-order Taylor (solid line), Lagrange polynomials (dashed line) and Linear-Least squares (dotted line).*

However, an important difference has been observed concerning performance. Table 4.7 summarises the CPU time to perform 1500 iterations with the three interpolation algorithms: Lagrange polynomials is the fastest, closely followed by Taylor series interpolation. Surprisingly, calls to the LAPACK function for LLS resolution are extremely slow. The reason may be that the dimension of the matrix to resolve is too small (N^o of vertices of the element $\times$ (N^o of dimensions + 1)) and the number of calls too high (N^o of particles $\times$ N^o of iterations), which limits the interest of its use in future applications.

Interpolation algorithm	**Taylor**	**LLS**	**Lagrange**
CPU time for 1500 it. (s)	7105.48	19932.83	6812.82
CPU time increment	+4%	+192%	—
Efficiency/iteration/node (s)	0.1725 10^{-4}	0.4839 10^{-4}	0.1654 10^{-4}

Table 4.4 - *Summary of the CPU time to perform 1500 iterations on 8 processors and with a load of 10 particles per cell. Simulations performed on a Compaq AlphaServer SC.*

As discussed in Subsubsection 2.7.1, small differences in initial conditions can lead to different instantaneous results after a certain time. An example of this is shown in Fig. 4.8, where two different particle trajectories computed with the three interpolation algorithms are displayed.

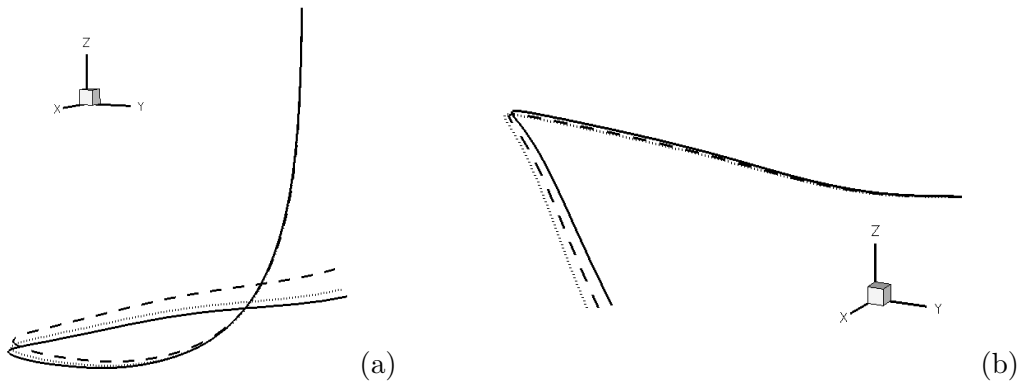


Figure 4.8 - *Example of two different particle trajectories with the three interpolation algorithms: first-order Taylor interpolation (solid); Lagrange polynomials (dashed); Linear-Least Squares (dotted).*

4.5.4 Illustration of preferential concentration

Experimental and numerical evidences indicate that, for some parameter regimes, particle concentrate in low-vorticity regions due to centrifugal effects, a phenomenon called preferential concentration [208, 50]. The basic physics underlying this phenomenon is the fact that inertial particles spin out from the centre of eddies; if the particle and fluid time constants are commensurate, so that the eddy persists on this spin-out timescale, then the particles will concentrate in regions where straining dominates vorticity [50, 206]. The key parameter identified in this kind of investigations is the Stokes number based on the Kolmogorov time scale, $St_K = \frac{\tau_p}{\tau_K}$, where $\tau_K \equiv (\nu_g/\varepsilon_g)^{1/2}$ is the Kolmogorov time scale (see Eq. (4.9)). Preferential concentration is usually observed for particles with $St_K \approx 1$. The slice of particle distribution presented in Fig. 4.9 shows that the Lagrangian module correctly reproduces the effects of preferential concentration: (a) a completely homogeneous particle concentration field corresponding to $St_K = 0$, (b) particle concentration fields at $St_K = 1$ (slice centered at $z = 0$, slice thickness of 1/64 the box length).

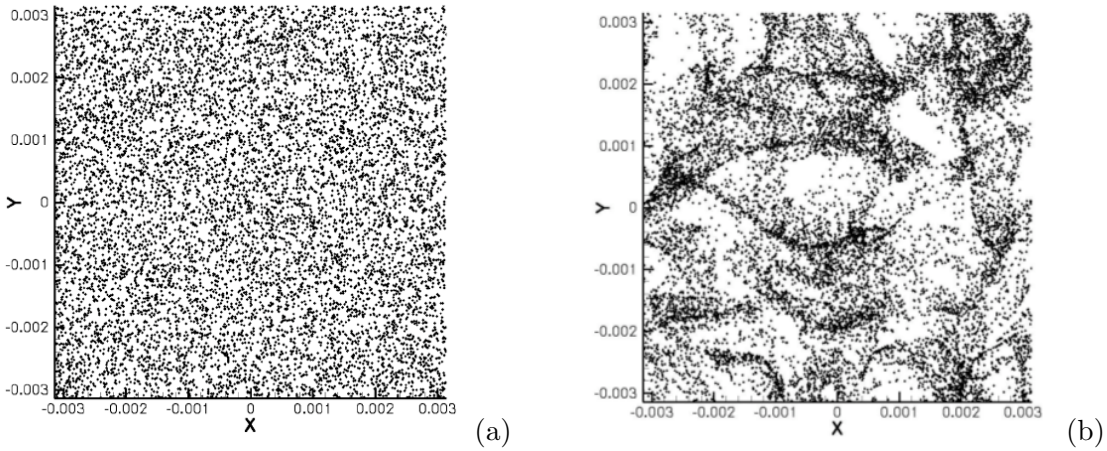


Figure 4.9 - *2D slice showing concentrated particle fields at (a) $St_\eta = 0$ and (b) $St_\eta = 1$.*

Figure 4.10 shows two vorticity snapshots with particles superimposed at two different instants. Regions of accumulation of particles are clearly visible showing that particle collection is done preferentially in regions of low vorticity. As pointed out by Squires & Eaton [208] in a study of forced homogeneous turbulence this implies that turbulence may inhibit rather than enhance mixing of particles.

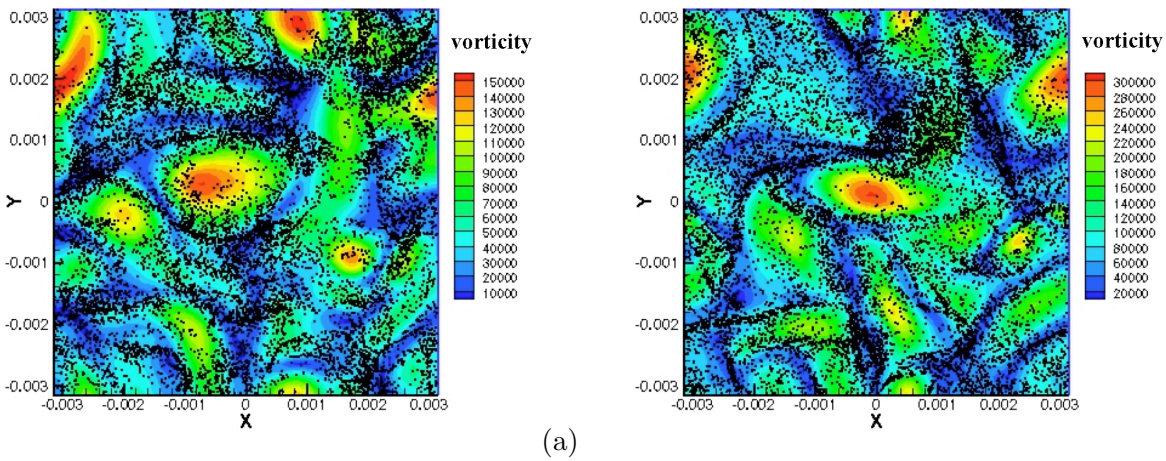


Figure 4.10 - *Vorticity snapshot with particles superimposed at two different instants.*

4.6 Summary and conclusions

The aim of this chapter was to validate the Lagrangian module developed in this thesis with an academic test case. The configuration chosen was particles in an Homogeneous Isotropic Turbulence (HIT) flow. Several aspects of performance and particle behaviour were studied. The following points summarise the main ideas of each section of the chapter.

- Section 4.1 described some basic notions of turbulence (e.g., the Reynolds number [177]) needed to better understand the rest of the chapter. The idea of *energy cascade* developed by Richardson [181] and the main scales of turbulence (integral and Kolmogorov [103]) were also presented.
- Gaseous Homogeneous Isotropic Turbulence (HIT) flows have been considered for a long time to analyse characteristics of turbulence and to propose turbulence models. A large amount of theoretical and numerical studies is available. Although HIT flows are considered an academic problem, mechanisms induced in such two-phase flows, like mixing and particles dispersion, are important subjects of interest in fluid mechanics since these phenomena occur in many technologically important areas, from the atmospheric transport of pollutants to particle dispersion in fluidized beds in chemical industry. Therefore, Section 4.2 focused on the explanation of the choice of the configuration.
- Isotropic turbulence is the simplest type of turbulence since no preference for any specific direction occurs and a minimum number of quantities and relations are required to describe its structure and behaviour. The study of this flow in the physical and spectral spaces was

presented in Section 4.3. In addition, the definition of turbulent kinetic energy (for the gaseous and the dispersed phase) and dissipation were also included.

- Section 4.4 presented the main characteristics of the grid and the parameters used to initialise the energy spectrum to perform the single-phase simulation. The spectrum of the turbulent kinetic energy of the carrier phase at the beginning of the simulation was compared to the one used at particle insertion.

- Section 4.5 presented the results of the two-phase HIT simulation. The first one is a performance analysis of particle location at the beginning of the simulation. The octree algorithm implemented in this thesis was compared to the brute force algorithm for a different number of particles per cell and for a different number of processors. The octree algorithm showed to be noticeably faster than the brute-force algorithm, as expected. Then, the results of the fluid and particle kinetic energies and autocorrelations were analysed. Another subsection showed the differences observed while changing the interpolation algorithm used to calculate fluid properties at particle positions. Results reflect minor differences in Lagrangian properties (like particle kinetic energy) but important differences in performance with the Linear-Least Squares algorithm of the LAPACK library. To conclude, an illustration of preferential concentration is presented. Vorticity snapshots with particles superimposed clearly shows that particle accumulation is done in regions of low vorticity.

Chapter 5

Application to a polydisperse two-phase flow of a confined bluff body

In combustion chambers, the flame resulting from a free jet flow would be too long compared to the dimensions of the chamber, and also very difficult to stabilise. Therefore, most combustion devices are designed so as to anchor the flame at a specific location (Kuo [106], Poinsot & Veynante [165]). The use of a flame holder is often difficult due to the very high temperatures that may damage the device itself. Another possibility is to stabilise the flame behind a sudden expansion like a backward-facing step (Pitz & Daily [160], Poinsot [161]): the flow is strongly decelerated forming a corner recirculation zone, and the recirculating hot gases then provoke the ignition of the incoming fresh gases. As far as aeronautical combustion chambers are concerned, highly swirling flows (Gupta *et al.* [75], Sloan *et al.* [203], Dellenback *et al.* [41]) that pass through a sudden expansion are preferred since they provide a more compact stabilised flame. A central toroidal recirculation zone is created, acting as a flame holder in the centre of the flow, close to the injector tip (Roux *et al.* [182], Hanson & Thomas [79], Pierce & Moin [157]). In such devices, the recirculation zones induce high turbulence levels and high mixing rates, stabilising the flame.

Before computing reactive two-phase flows in such devices, a validation of the turbulent dispersion of the particles in similar flows is needed. The test case chosen to validate the numerical developments of the Lagrangian module is a particle-laden bluff-body configuration from Borée *et al.* [22] where glass beads are injected into a complex recirculating flow. This configuration is typical of an industrial application where the objective is to control the mixing of a fuel (pulverised coal) with the air. The test is performed for non-reacting, non-evaporating sprays. Several reasons motivated the choice of this configuration: the presence of a recirculation zone which allows us to study the dispersion of the particulate phase in a critical configuration where small particles will be captured within the recirculation zone while larges ones will cross it and escape directly into the downstream part of the flow; the relatively simple geometry compared to complex realistic devices, and, most important, the large amount of data available for both gaseous and dispersed phases (the complete data set, including accurate boundary conditions, has been selected for benchmarking at the 'Ninth workshop on two-phase flow predictions' Ishima *et al.* [92] and can be downloaded at the following web site: http://www-mvt.iw.uni-halle.de/english/index.php?bluff_body_flow).

The computational results obtained with the Lagrangian version of AVBP are compared with results from another Lagrangian solver developed at the Stanford University (CDP) and with measurements. Results of the CDP solver for the gaseous and the dispersed phases were provided by Vincent Moureau. The reader is referred to a more complete study of the influence of various parameters (mesh type, numerical convective scheme and inlet boundary conditions) for this single-phase flow that is included in the article accepted for publication in the Journal of Computational Physics (Vol. 228, N° 2, pp. 539-564, 2009) presented in Appendix E.

This chapter is organized as follows: Section 5.1 describes the bluff-body configuration from Borée *et al.* [22]. The computational set-up of the configuration is presented in Section 5.2. Sections 5.3-5.5 compare results obtained with both solvers and measurements for the single-phase simulation, the two-phase monodisperse and the polydisperse simulation, respectively. Section 5.6 is an analysis of code performance and especially of its efficiency on parallel architectures.

5.1 Experimental set-up

The configuration of Borée *et al.* [22] consists of a vertical axisymmetric particle-laden confined bluff-body flow. The data were obtained in a flow loop of EDF-R&D, called Hercule. A sketch of the geometry is presented in Fig. 5.1. Both air and particles are injected in the inner jet whereas air blowers are used to generate the coflow. The measurement zone is located downstream of the inner and annular ducts ($z > 0$), where large recirculation zones are created between the central jet and the coflow due to the geometry. The resulting flow is similar to the flows obtained in industrial combustion devices, where fuel droplets are injected together with air.

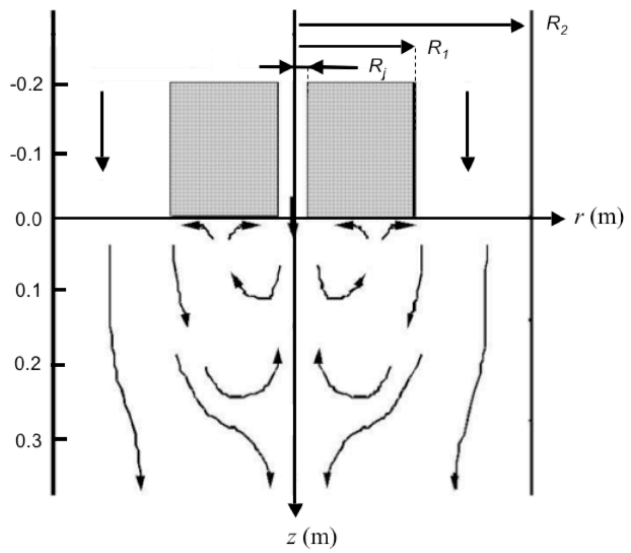


Figure 5.1 - *Sketch of the configuration of Borée* et al. *[22]. The dimensions are:* $R_j = 10$ *mm,* $R_1 = 75$ *mm,* $R_2 = 150$ *mm. The length of the measurement section is* $z = 1.5$ *m.*

The topology of the gas flow mainly depends on the ratio between the velocity in the inner pipe and the velocity in the coflow. As pointed out by Borée *et al.* [22] the characteristics of the single-phase flow were carefully selected to obtain a distinct mean jet stagnation point on the axis of the recirculation zone. Such a single-phase flow has been shown to be very interesting when adding

particles. Indeed, particle inertia as well as fluid-particle interactions are the main mechanisms in such two-phase flows (Simonin [202]). The data are obtained for isothermal conditions using a two-component phase-Doppler anemometer that provides size and velocity measurements. The data include the radial profiles of the following quantities at seven stations along the axis ($z = 3, 80, 160, 200, 240, 320$ and 400 mm) in the measurement zone:

- mean axial and radial velocity components for the carrier and the dispersed phases;
- RMS axial and radial velocity components for both phases;
- particle number density and particle mass flux.

In addition, the axial evolution of the mean axial velocity, RMS axial velocity and RMS radial velocity for air and for some size classes are provided versus z on the centerline $r = 0$.

5.1.1 The gaseous phase

Schefer *et al.* [190] showed that the topology of a turbulent bluff-body flow strongly depends on the ratio $\overline{U}_j/\overline{U}_e$, where $\overline{U}_j$ and $\overline{U}_e$ are bulk velocities in the inner and annular pipes. Depending on this velocity ratio, there are three possible flow topologies. Figure 5.2 illustrates these topologies for three decreasing velocity ratios ($\overline{U}_j/\overline{U}_e = 2.8, 1.4$ and 0.84) and Fig. 5.3 shows the flow topology by seeding the gas flow with small particles. In these cases it can be observed that:

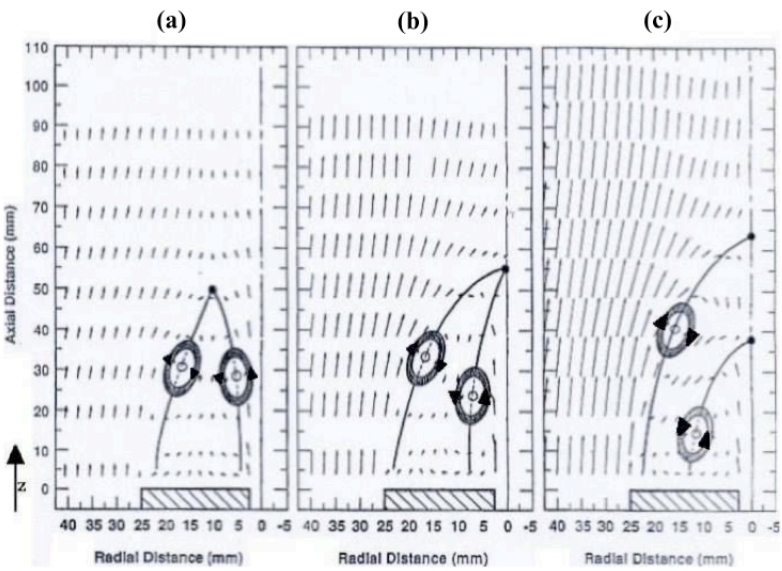


Figure 5.2 - *Plots of the measured mean velocity vectors in a bluff body stabilised methane jet for (a)* $\overline{U}_j/\overline{U}_e = 2.8$, *(b)* $\overline{U}_j/\overline{U}_e = 1.4$, *and (c)* $\overline{U}_j/\overline{U}_e = 0.84$. *Black dots show the location of the stagnation points and round arrays give the direction of rotation of the shear-layer vortices. From Schefer* et al. *[190].*

1. For the highest velocity ratio ($\overline{U}_j/\overline{U}_e = 2.8$), the flow along the axis is similar to a free jet flow. The air flows coming from the inner and annular pipes converge far from the outlet of the inner duct. Two counter-rotating eddies separate the two flows before they converge. There are two stagnation points on both sides of the central jet.

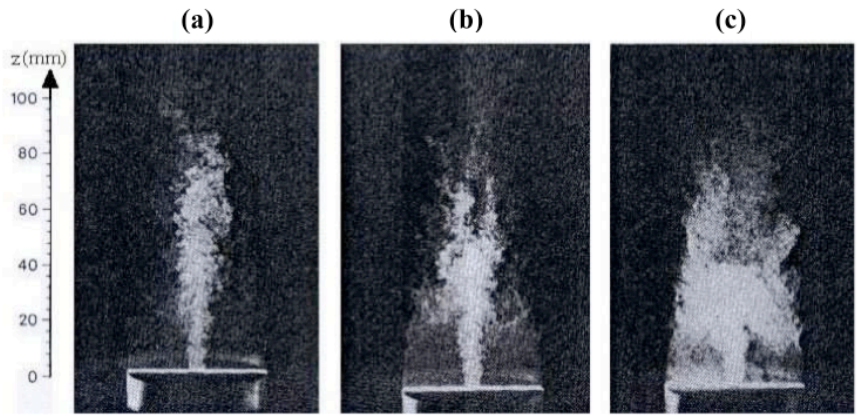


Figure 5.3 - *Mie scattering measurements of the instantaneous particle distribution in a bluff body stabilised methane jet for* $\overline{U}_j/\overline{U}_e = 2.8$ *(a),* $\overline{U}_j/\overline{U}_e = 1.4$ *(b), and* $\overline{U}_j/\overline{U}_e = 0.84$ *(c). From Schefer* et al. *[190]*

2. At lower velocity ratio ($\overline{U}_j/\overline{U}_e = 1.4$), a single stagnation point is formed along the centerline and the flow looks like a non-penetrating jet.

3. Finally, for the smallest velocity ratio ($\overline{U}_j/\overline{U}_e = 0.84$), a second stagnation point appears on the axis whose location is linked to the geometry global parameter. The second stagnation point remains close to the chamber inlet and does not move any longer when the ratio $\overline{U}_j/\overline{U}_e$ becomes lower.

The experiments are conducted at ambient temperature and standard pressure. The length of the measurement section is 1.5 m. The dimensions of the annular outer region are: $R_1 = 75$ mm, $R_2 = 150$ mm (see Fig. 5.1). The external volume flux is kept constant at $Q_e = 780$ m^3/h. The coflow mean velocity is of $\overline{U}_e = 4.1$ m/s (with a maximum velocity of $U_{e,max} = 6$ m/s) and the Reynolds number of the flow is $Re \approx 40{,}000$. The length of the inner jet upstream of the test section is $L_e = 2$ m. With $2L_e/(R_2 - R_1) \approx 54$, the turbulent boundary layer can be considered as established, but the turbulent annular flow is suspected to be under slight development (Comte-Bellot [34]). The radius of the inner pipe is $R_j = 10$ mm and the air volume flux is $Q_j = 3.4$ m^3/h. The jet mean velocity on the axis is $\overline{U}_j = 3.4$ m/s (with a maximum velocity of $U_{j,max} = 4$ m/s).

In the present configuration, the ratio for the gaseous flow is: $\overline{U}_j/\overline{U}_e = 0.83 < 1$. Following Schefer *et al.* [190], this implies the existence of a recirculation zone delimited by two stagnation points along the centerline. Choosing a ratio lower than one creates a complex gas flow behaviour when modeling particle dispersion: depending on their inertia, the particles remain in the recirculation zone delimited by the two stagnation points or leave it (see Section 5.5).

5.1.2 The dispersed phase

The dispersed phase consists of solid particles (glass beads) so that evaporation, coalescence and break-up do not have to be considered. The material density of the glass particle is $\rho_p = 2470\ kg/m^3$. Particles are released in the inner pipe by a particle feeder with a initial particle size distribution covering a wide range of size classes from 20 to 100 μm. Figure 5.4 shows particle distribution, in mass (a) and in number (b). The resulting mean diameters are $d_{p,M} = 63$ μm and $d_{p,N} = 50$ μm, respectively. Two mass flow rates of particles of 1 kg/h and 5 kg/h corresponding

to inner jet mass loading ratio of $M_j = 22\%$ and $M_j = 110\%$, respectively, are studied in Borée *et al.* [22]. Only the moderate mass loading ratio of the inner jet ($M_j = 22\%$) is studied in this thesis since the current version of the Lagrangian module does not dispose of a collision model which would be needed to perform the high mass loading ratio case. In fact, this moderate mass loading ratio corresponds to a solid volume fraction less than 10^{-4} and thus collision effects can be assumed to be negligible in the model.

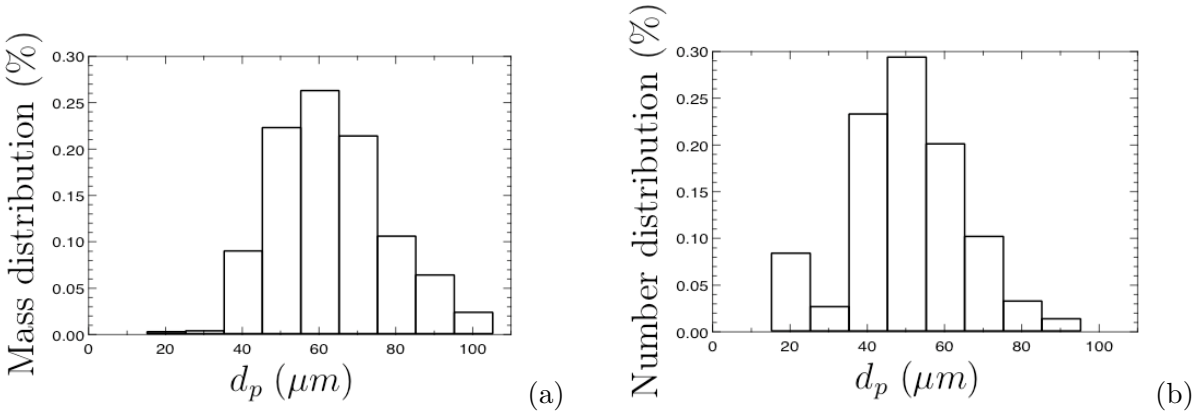


Figure 5.4 - *Initial mass (a), and number (b) distribution of the particle size.*

Assuming Stokes flow around particles gives a particle relaxation time $\tau_p = \rho_p {d_p}^2/18\mu_f$, where ρ_p is the particle density, d_p is the median diameter of the particle size class and μ_f is the fluid dynamic viscosity. The comparison of the particle relaxation time with a characteristic time scale of the fluid most energetic eddies, $\tau_{f,t}$, yields the Stokes number St. To evaluate $\tau_{f,t}$ at the outlet of the inner pipe, the size of the most energetic eddies is estimated as a third of the pipe diameter and their velocity as the maximum fluctuating velocity in the pipe:

$$St = \frac{\tau_p}{\tau_{f,t}} \quad , \quad \tau_{f,t} = \frac{\frac{2R_j}{3}}{u_{f,max}}. \tag{5.1}$$

Table 5.1 presents the particle relaxation time and the characteristic Stokes number of particles depending on their diameter: the smallest particles with diameter $d_p = 20$ μm almost follow the gas flow while the inertia of the largest ones with diameter $d_p = 100$ μm makes them much more independent of the fluid flow.

d_p (μm)	20	30	40	50	60	70	80	90	100
τ_p (ms)	3.1	6.9	12.3	19.2	27.6	37.6	49.1	62.2	76.7
St (−)	0.4	1.0	1.8	2.7	3.9	5.4	7.0	8.9	11.0

Table 5.1 - *Particle relaxation time and Stokes number depending on particle diameter.*

5.2 Computational set-up

The experimental configuration consists of two annular ducts of 2 m length and a chamber of 1.5 m (Fig. 5.5 (a)). The integral turbulence scale of the larges eddies of the coflow is of the order of some

millimeters, and the Kolmogorov length-scale of the order of some micrometers. The time needed to perform a three-dimensional LES simulation depends on the characteristic length-scales that are to be resolved. Regarding the inner pipe, for example, a grid with several thousand millions cells would be necessary to compute flow dynamic and particle motion inside the duct. This means that trying to represent numerically the whole experimental domain would be computationally too expensive to capture a large number of turbulent structures and scales.

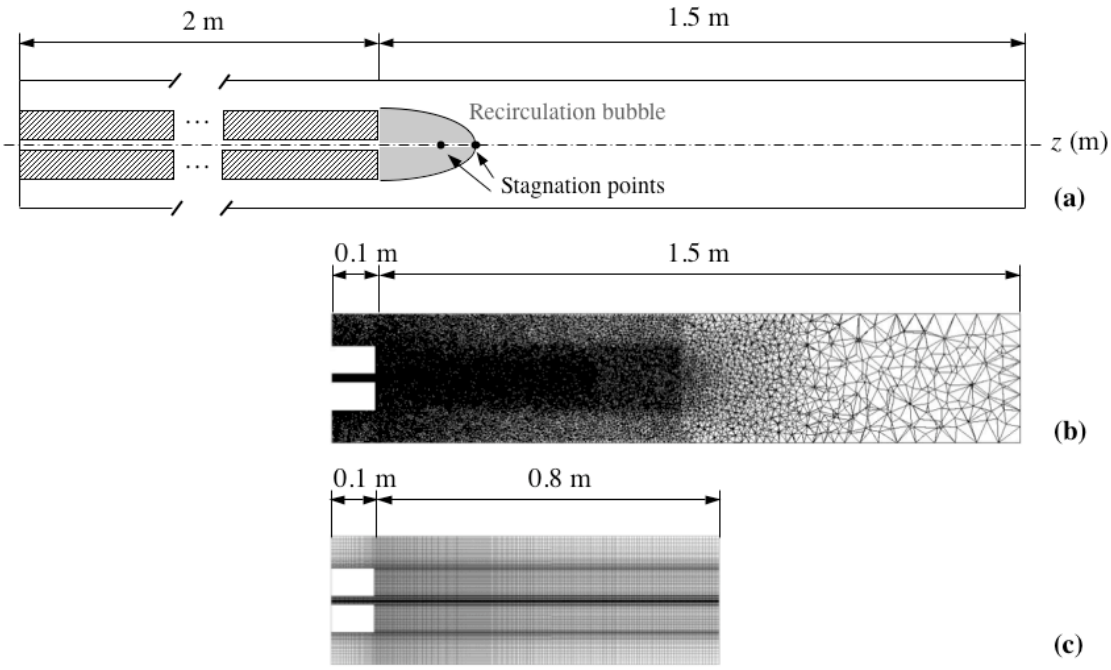


Figure 5.5 - *(a) Sketch of the experimental configuration and longitudinal views of (b) the tetrahedron and (c) the hexahedron-based grids used. Cutting plane:* $x = 0$.

The idea is then to reduce the size of the computational domain without losing the key features of the flow. For example, the location of the second stagnation point (in the vicinity of $z = 0.2$ m) must be ensured to capture the recirculation zone. This point mainly depends on the geometry global diameter, and therefore, the diameters of the inner and annular pipes have not been modified: $R_j = 0.010$ m; $R_1 = 0.075$ m and $R_2 = 0.150$ m (see Fig. 5.1). In contrast, the length of the annular and inner ducts and the chamber have been reduced. Figures 5.5 (b) and (c) present the two different grid types used in this thesis (hexahedron and tetrahedron-based grids). Results of the effects of the grid resolution are detailed in Appendix E. In both grid types, the inner pipes have been shortened to 0.1 m for two reasons:

- On one hand, it is necessary to decrease the length of the ducts: considering the low Reynolds number and the grid resolution in the inner pipe as well as the accuracy of the numerical scheme, it is impossible to wait for natural destabilisation of the gas flow within the pipe.
- On the other hand, the pipes length cannot be decreased down to 0.1 m: the accurate prediction of particle motion in a pipe (or a channel) is still difficult to obtain, especially because of particle-wall interactions (Wang & Squires [223]) and inter-particle collisions (Vance *et al.* [219]). Since these interactions are not accounted for in this work, one has to ensure that the modified pipe length is compatible with the particle relaxation time (evaluated in Section 5.1.2).

The length of the chamber has been reduced from 1.5 m to 0.8 m with the hexahedron-based grid. This is sufficient to capture the main structures of the bluff-body flow without an effect on the second stagnation point. The main motivation for testing a hexahedron-based grid on this configuration is the comparison with CDP that uses hexahedra elements.

According to these simplifications, the volume of the computational domain is considerably reduced allowing a large-eddy simulation with a reasonable computational cost. Results of this configuration with the tetrahedron-based grid will not be presented in this chapter. The reader is referred to Appendix B and C for a more complete study with this grid.

5.2.1 Numerical parameters

The LES solver CDP solves the incompressible Navier-Stokes equations using an implicit formulation. Time integration is based on the fractional-step method (Kim & Moin [100]) while space integration relies on a second-order central scheme which conserves the kinetic energy (Mahesh *et al.* [125], Ham & Iaccarino [77]). The dynamic Smagorinsky model (Germano *et al.* [72]) is used to model the subgrid stress tensor.

The explicit LES solver AVBP solves the compressible Navier-Stokes equations with a third-order scheme for spatial discretization and a Runge-Kutta time advancement, called TTGC (Colin & Rudgyard [33], Moureau *et al.* [138]). For the present case, the WALE (Wall-Adapting Local Eddy-viscosity) model proposed by Ducros *et al.* [47] is used to model SGS tensors.

Fig. 5.6 shows the computational domain and the boundary conditions. Walls are treated as no-slip boundaries. The boundary conditions are handled with the NSCBC formulation (Poinsot & Veynante [165], Moureau *et al.* [138]). All these parameters, together with the averaging time for the simulations are summarised in Table 5.2.

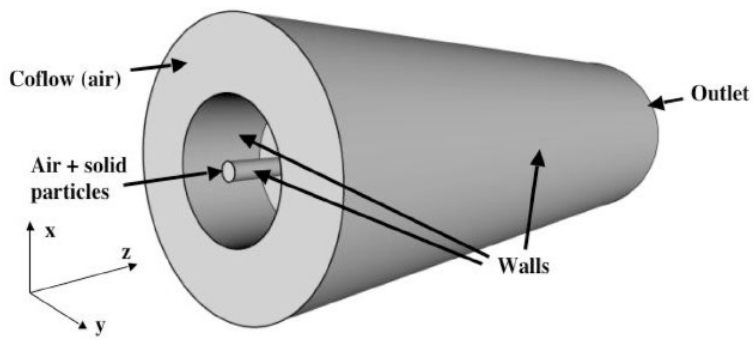


Figure 5.6 - *Boundary conditions presented in the configuration of Borée* et al. *[22].*

Figure 5.7 displays a longitudinal view of the hexahedron-based grid with the seven experimental stations and Table 5.3 summarises parameters of the this grid used for both solvers. Note that a buffer zone is added in the last 100 millimeters to dissipate structures before they leave the domain As already mentioned, CDP solves the incompressible Navier-Stokes equations implicitly whereas AVBP-EL solves them explicitly. The main consequence is that the time step is 35 times larger for CDP, leading to smaller averaging time for AVBP-EL; however, we verified that the convergence was good enough to make comparisons between both codes.

	CDP	**AVBP-EL**
Time step (μs) / CFL	147 / 50	$4,22$ / 0.7
Averaging time (s) without particles / iterations	2.65 / 18,000	0.4642 / 110,000
Averaging time (s) with particles (mono) / iterations	3.97 / 27,000	0.4642 / 110,000
Averaging time (s) with particles (poly) / iterations	11 / 74,000	3.89 / 922,000
Convective scheme	2nd order kinetic energy conserving	TTGC (3rd order)
LES model	Dynamic Smagorinsky	WALE
Wall model	None	None
Inner jet / Coflow Inlet BC	Forcing / No forcing	Forcing / Forcing

Table 5.2 - *Comparison of the parameters and models used for the computations with CDP and AVBP-EL.*

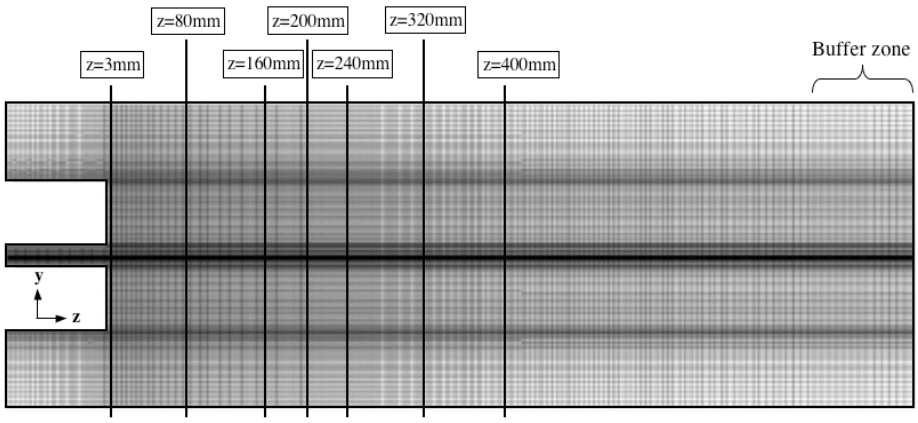


Figure 5.7 - *Longitudinal view of the hexahedron-based grid used by AVBP-EL and CDP. Cutting plane: $x = 0$. The seven experimental stations and the buffer zone are also presented.*

Grid elements	**N° of cells**	**N° of nodes**	$\mathbf{L_e}$ (m)	$\mathbf{L_{chamber}}$ (m)	$\mathbf{min(V_{cell})}$ (m^{-3})
Hexahedra	3,207,960	3,255,085	0.1	0.8	$4.94\ 10^{-10}$

Table 5.3 - *Summary of grid parameters used in all computations.*

Figure 5.8 (a) shows a frontal view of the grid. Zooms (A-B) show that the radial structure is less regular at the centre of the grid (Fig. 5.8 (b) and (d)) than at the near-wall cells. Mean velocity profiles resulting from LES time-averaging are calculated at 16 radial stations (Fig. 5.8 (c)) and further averaged in the azimuthal direction. A total number of 200 angular samples is interpolated at each station.

The influence of the particles on the gas phase is taken into account by using the point-force approximation in the general framework of the particle-in-cell method (PIC) (Boivin *et al.* [19], Vermorel *et al.* [220]). According to Boivin *et al.* [20], such an assumption is valid for small mass loading ratio $\alpha_p \rho_p / \rho_g \leq 1$, with response time larger than the subgrid turbulence characteristic time scale. Modification of the gas subgrid-scale turbulence model by particles is neglected.

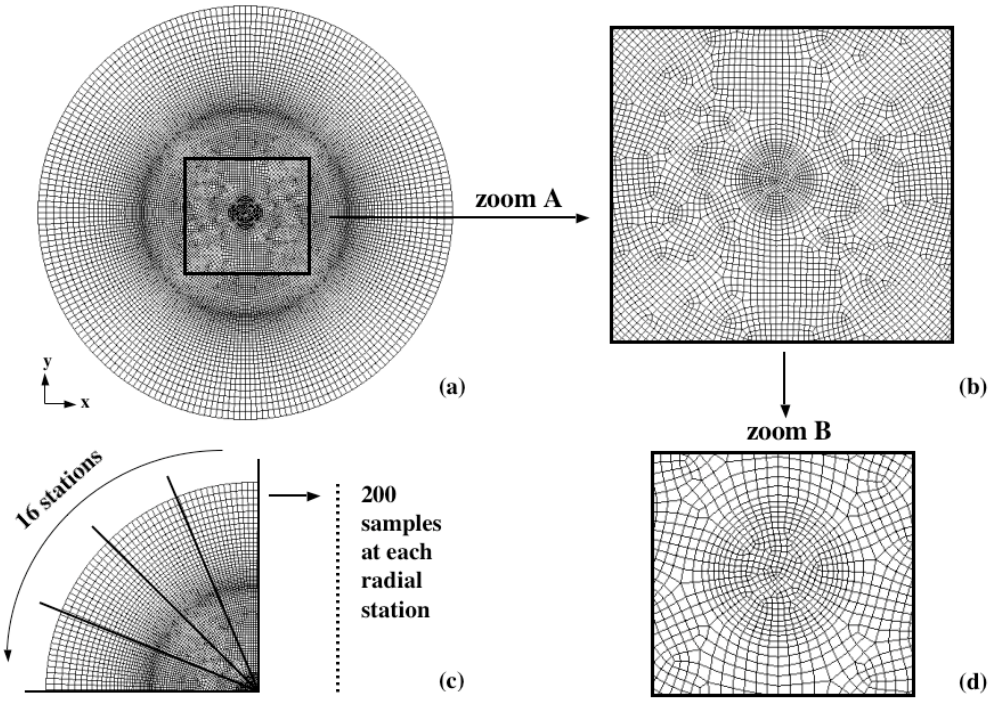


Figure 5.8 - *(a) Y-X view of the hexahedron-based grid; (b) zoom of the previous view; (c) location of the radial stations used in the post-processing of mean variables. Each station contains 200 samples; (d) zoom of the centre of the grid.*

5.3 Results for single-phase flow case

Before discussing results for the two-phase flow simulations, the accuracy of both LES solvers for the single-phase case is evaluated by computing the flow without particles and comparing it with the corresponding data provided by Borée *et al.* [22]. Hereafter, the experimental results are plotted using symbols for comparison with AVBP-EL numerical results which are represented by a solid line and CDP numerical results plotted by a dot-dashed line.

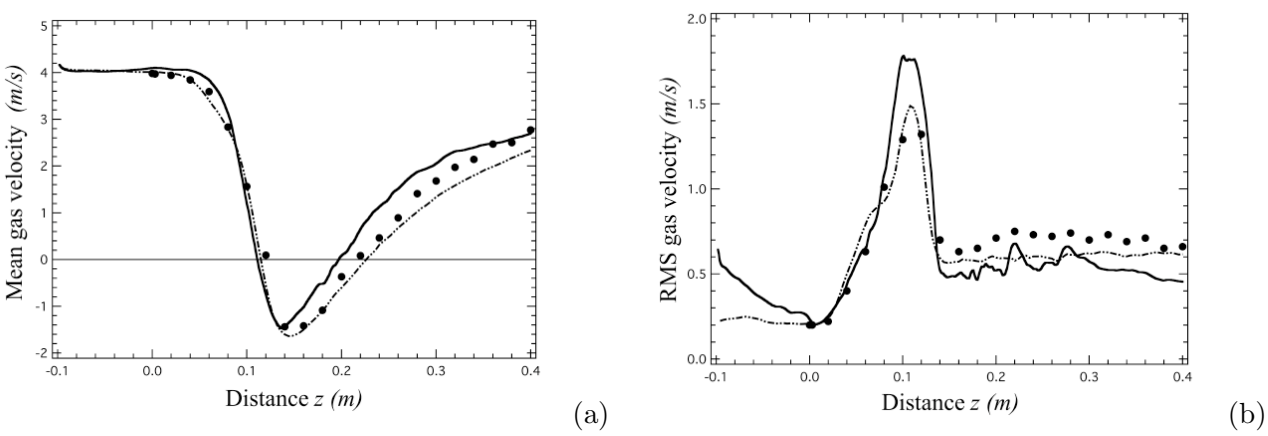


Figure 5.9 - *Axial evolution of mean (a) and RMS (b) gas velocities. Symbols: experiment; solid line: AVBP-EL; dot-dashed line: CDP.*

Figure 5.9 displays mean (a) and RMS (b) gas velocities along the axis. The global agreement between the two codes and experiments is very good and most of the flow physics is captured by both LES solvers. Figure 5.9 (a) shows that the central jet flow is stopped: the mean longitudinal velocity significantly becomes zero around $z = 100$ mm. The width and length of the recirculation zone (evidenced by the negative values of axial velocities on the axis at $z \simeq 120$ and $z \simeq 210$ mm) are well predicted. Minor differences between the results of the two codes are observed in the prediction of the location of the second stagnation point. The first stagnation point at $z = 120$ mm is almost located at the same place but the second one is located 20 mm upstream the experimental value (approx. $z \simeq 190$ mm) in the AVBP-EL simulation. The prediction of stagnation points is known to be a critical issue in bluff-body simulations as they are very sensitive to the ratio between the mean velocity of the inner jet and the coflow. Indeed, the small differences observed between the two codes in the inlet velocities lead to a substantial difference in the position of the second stagnation point. Figure 5.9 (b) shows the evolution of the axial velocity fluctuations predicted by both solvers. The peak-value location is well predicted but the amplitude is better predicted by CDP. This maximum corresponds to the first stagnation point. Downstream of this point, the RMS levels are slightly underestimated but despite these differences the overall agreement is good.

Figs. 5.10-5.11 present the radial profiles of the mean axial and RMS velocities at seven cross-sections along the axis predicted by the two codes and compared to the experimental measurements. Both solvers capture the mean axial component correctly despite the small acceleration of the flow observed at $z = 160$ mm and $z = 200$ mm for AVBP-EL results and a small deceleration downstream $z = 200$ mm for CDP results (see Fig. 5.10). These differences are due to the mismatch in the recirculation position. Regarding the RMS velocities (Fig. 5.11), the agreement between numerical results and experiments is good. A general slight underestimation of the RMS values in the coflow ($r > 0.10$ m) for CDP results can be observed.

Figs. 5.12-5.13 present the radial profiles of the mean and RMS radial velocities, respectively. Despite the low levels of mean radial velocity compared to the dominating axial velocity the two LES codes capture the mean radial velocity fields correctly. The first station ($z = 3$ mm) shows a very low level of the mean radial profile at the output of the annular and inner pipes since almost all the gas velocity comes from the axial component (as observed in the first section of Fig. 5.10). Results at $z = 80$ mm show levels of the order of 0.5 m/s which represent a slight opening of the jet with positive values up to a distance to axis of $r = 60$ mm. Negative values confirm the existence of a region with a recirculating upward flow feeding the initial entrainment in the jet and the recirculation section. The highest levels (positive and negative) of the mean radial velocity profiles are detected in the stations near the recirculation bubble. In the end, we can reasonably state that main differences are always found in the third cross-section ($z = 160$ mm), whereas upstream and downstream, the agreement is very good. As mentioned above, the source of the problem is the prediction of the exact position of the stagnation point as shown in Fig. 5.9 (a): any small variation in this position leads to large changes in profiles measured around this point. The same general behaviour can be observed in Fig. 5.13 except for a discrepancy at the corner of the coflow and the step (first station at $z = 3$ mm) where the RMS velocities are underpredicted by both solvers.

The conclusion of this section is that both codes provide similar results for the gas phase without particles even though they use totally different methods. This indicates that solver independence for the gas is achieved for this case and that dispersed phase computations can be performed with reasonable confidence. The reader is reported to a more detailed analysis of the influences of different parameters (mesh type, numerical convective scheme and inlet BC) for this single-phase flow in the article submitted to the Journal of Computational Physics and included in Appendix E.

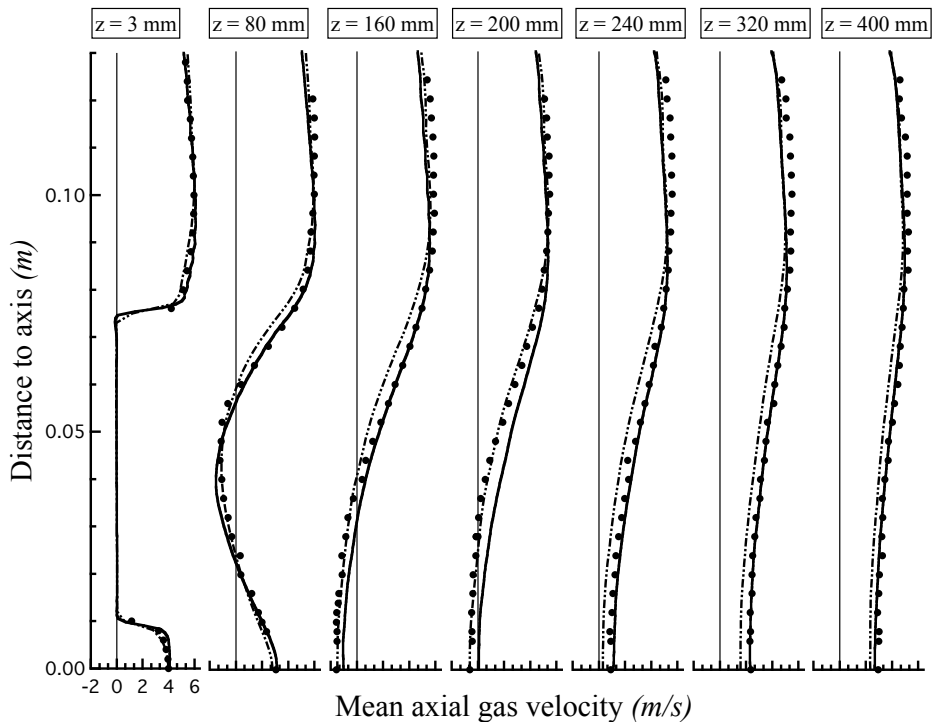


Figure 5.10 - *Radial profiles of mean axial gas velocities at seven stations along z axis. Symbols: experiment; solid line: AVBP-EL; dot-dashed line: CDP.*

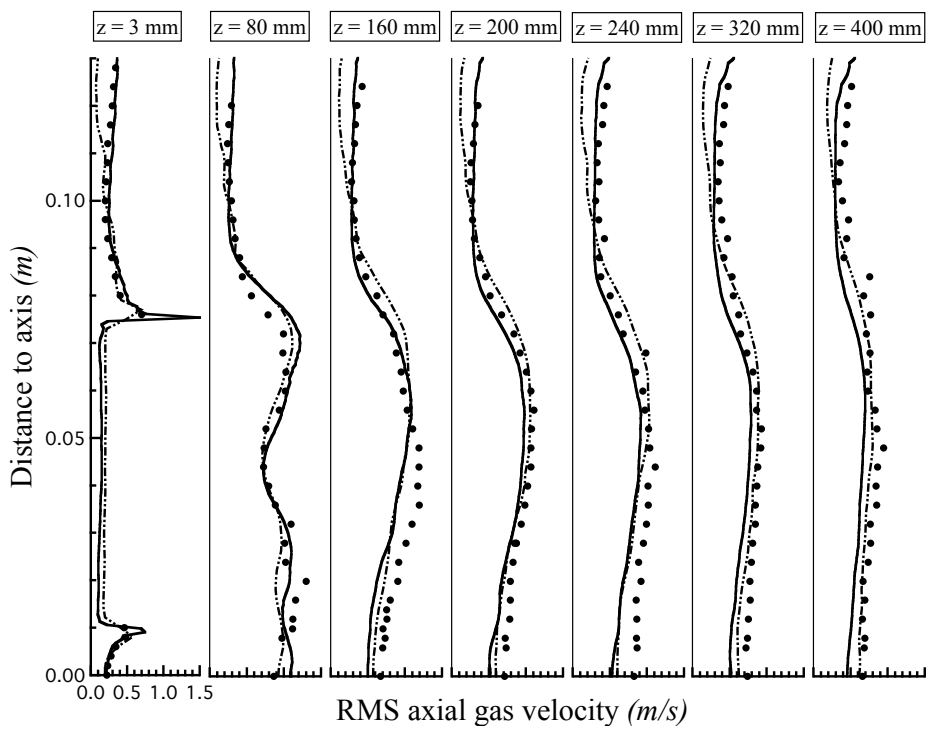


Figure 5.11 - *Radial profiles of RMS axial gas velocities at seven stations along z axis. Symbols: experiment; solid line: AVBP-EL; dot-dashed line: CDP.*

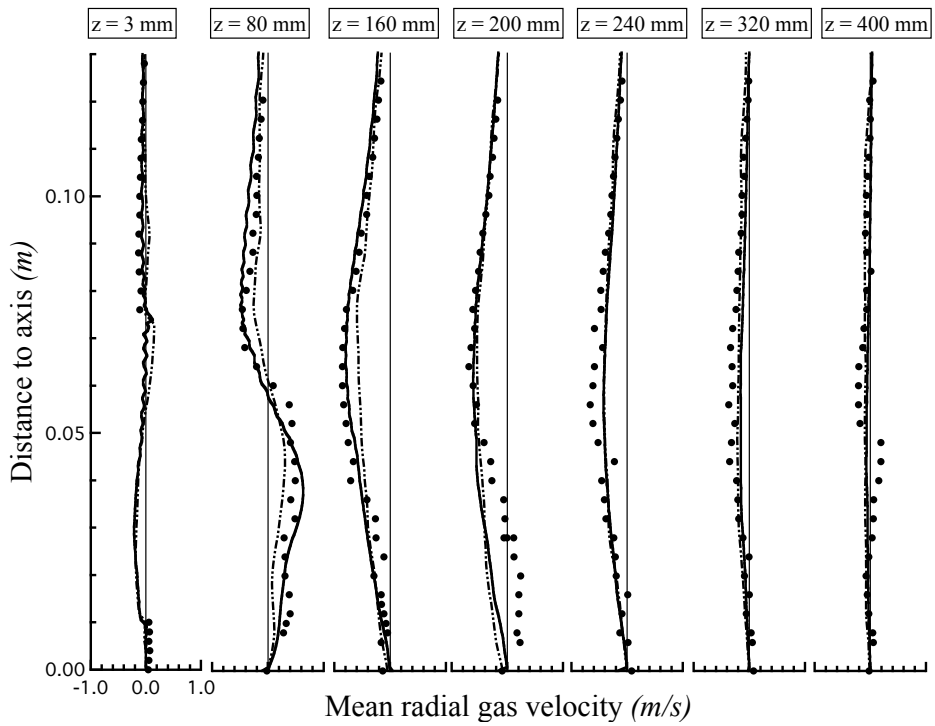


Figure 5.12 - *Radial profiles of mean radial gas velocities at seven stations along z axis. Symbols: experiment; solid line: AVBP-EL; dot-dashed line: CDP.*

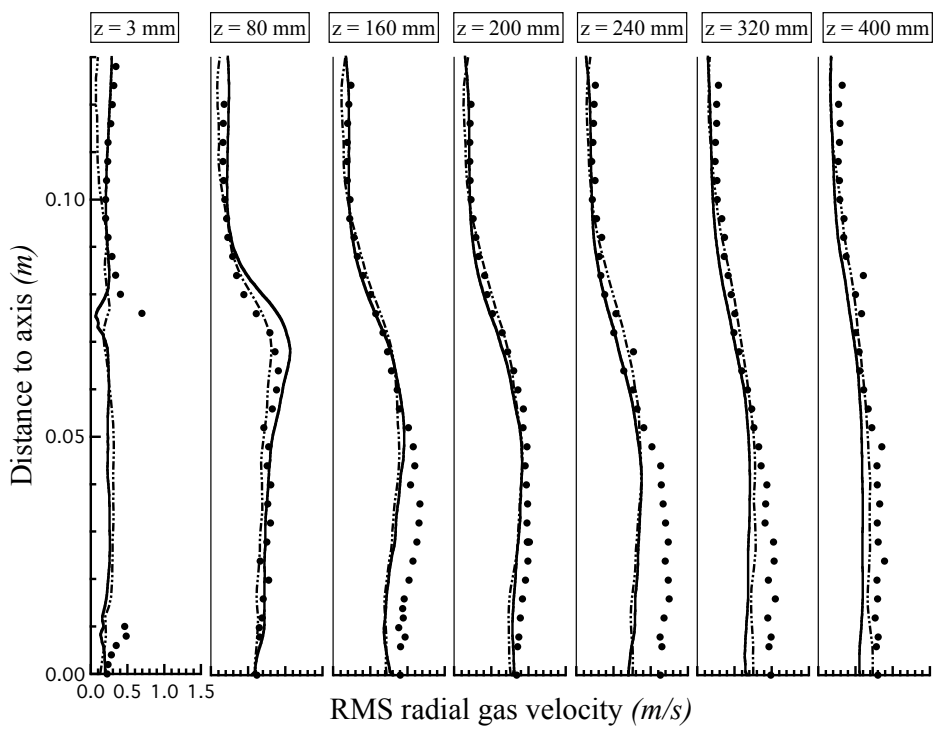


Figure 5.13 - *Radial profiles of RMS radial gas velocities at seven stations along z axis. Symbols: experiment; solid line: AVBP-EL; dot-dashed line: CDP.*

5.4 Results for two-phase flow monodisperse case

Although the distribution of particles introduced experimentally in the inner pipe is polydisperse, a monodisperse distribution is first computed and analysed in this section. One of the reasons is the comparison (mesh and input parameters) with the Eulerian version of AVBP (where the dispersed phase is solved using an Eulerian formulation) whose polydisperse functionalities were not available at this time. These results can be consulted in the articles included in Appendices B, C and E. These articles represent an important step towards the study and comprehension of this bluff-body flow and are completed with the analysis of the polydisperse case included in Section 5.5.

The particle diameter chosen to perform this monodisperse case is $d_p = 60$ μm, which is close to the mean diameter in mass: $d_{p,M} = 63$ μm. Only the moderate mass loading ratio ($M_j = 22\%$) is analysed in this document. As a first approximation, considering a monodisperse distribution at moderate mass loading would be sufficient to capture both the mean flow effects on the gas (two-way coupling) and the dynamics of the 60 μm particle class. However, due to the sensitivity of the gaseous phase to initial mass loading of the inner jet, a polydisperse distribution should be used to study the flow behaviour at high mass loading ($M_j = 110\%$).

An essential part of these LES is the introduction of the particles in terms of position and velocity (Fig. 5.14). The injection planes are not the same for the two codes but the methodology used to inject particles is the same: the mass loading is uniform over the injection section and the speed profile is the experimental one measured at the first measurement station, $z = 3$ mm (see Fig. 5.7). A white noise with an amplitude of the order of 10 percent of the mean velocity is added to the particle mean velocity profiles to match experimental measurements at $z = 3$ mm. The velocity fields for the gas phase change when the particles are injected and the jet penetrates more, changing the position of the recirculation zone (first stagnation point now located at $z = 150$ mm and second mean stagnation point located at approximately $z = 240$ mm) but these effects are not discussed here.

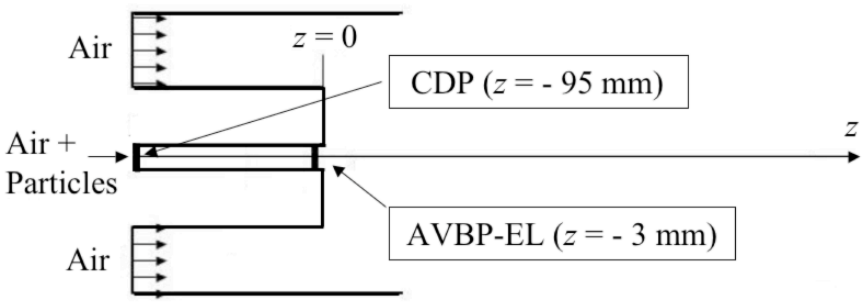


Figure 5.14 - *Injection position for particles in both solvers.*

5.4.1 Gaseous phase

As observed in the previous section, a critical zone of the flow-field is located on the central axis around $z \simeq 150$ mm. Figure 5.15 (a) shows the axial evolution of the mean gas velocity. Both solvers capture well the width of the recirculation zone. However, CDP is able to reproduce it almost at the same place as the measurements. A good agreement is also found for the axial RMS

gas velocities (Fig. 5.15 (b)). Again, a slight underestimation of the RMS results can be observed after $z = 200$ mm but the location and amplitude of the peaks are well predicted by both solvers.

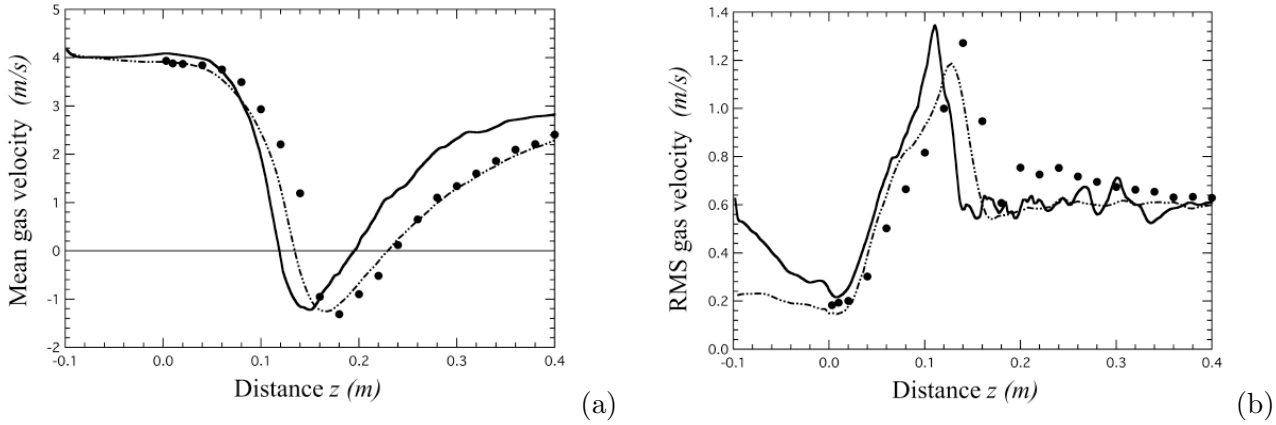


Figure 5.15 - *Axial evolution of mean (a) and RMS (b) gas velocities at $M_j = 22\%$. Symbols: experiment; solid line: AVBP-EL; dot-dashed line: CDP.*

Figure 5.16 shows a snapshot of the instantaneous gas velocity field in the cutting plane $x = 0$ with isocontour line of zero axial gas velocity. Many structures of different sizes are clearly visible. The highest values of the velocity field are obtained in the annular outer region (white flood). The recirculating bubble is well evidenced by negative values (black flood) and the inner jet is also clearly observed. This figure also displays the influence of the buffer zone added after $z = 700$ mm. This zone induces additional numerical dissipation and damps structures, as expected.

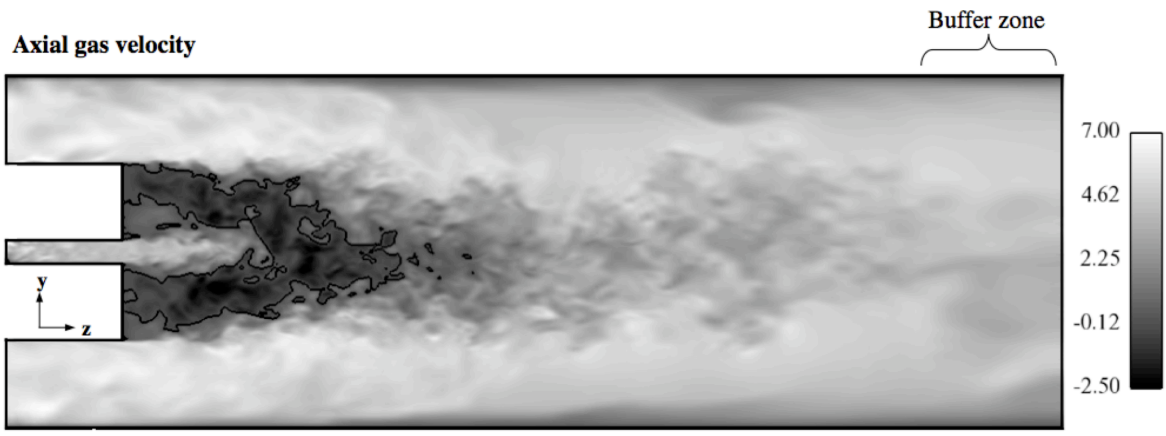


Figure 5.16 - *Instantaneous field of axial gas velocity in the cutting plane $x = 0$ with isocontour line (black) of zero axial gas velocity.*

Results for the radial profiles of the gaseous phase are shown in Figs. 5.17-5.20. The two solvers give quite similar results for the mean and RMS gas velocity profiles after the recirculation zone but a discrepancy between numerical results and the measurements must be pointed out. For example, the predicted mean radial gas velocities (Fig. 5.19) are slightly lower than measurements, even if profiles at the first three cross-sections are in very good overall agreement with experimental results. Despite these differences there is an overall good agreement between the experiments and the LES results.

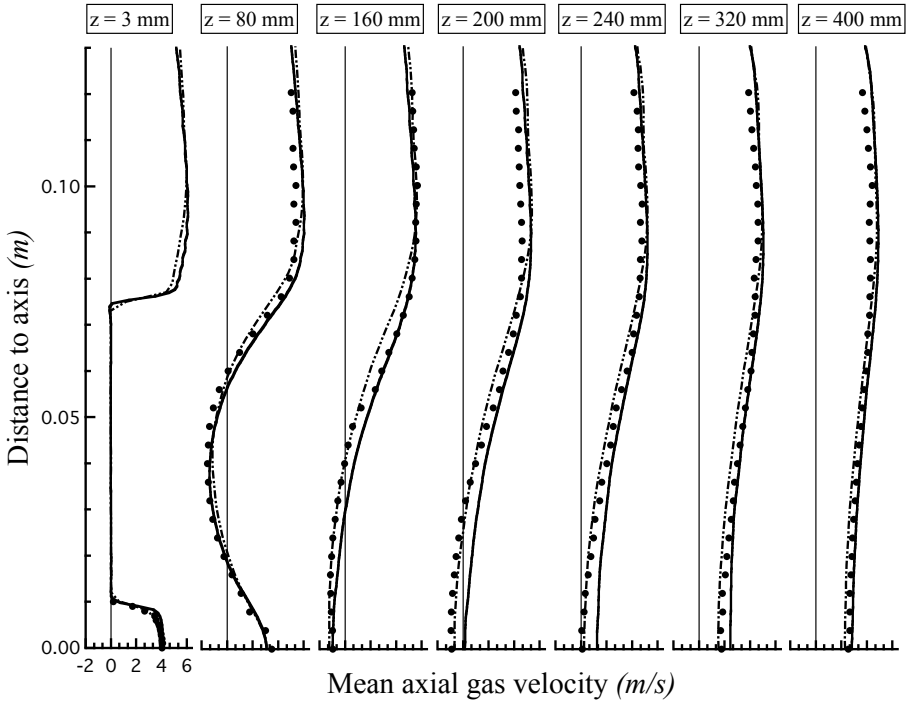


Figure 5.17 - *Radial profiles of mean axial gas velocities at seven stations along z axis at* $M_j = 22\%$. *Symbols: experiment; solid line: AVBP-EL; dot-dashed line: CDP.*

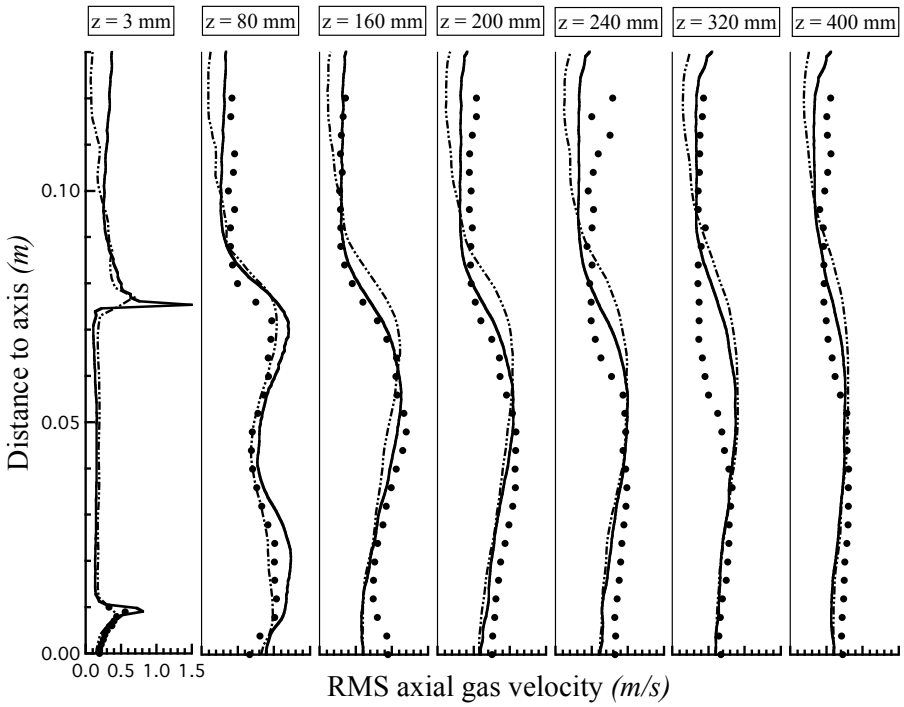


Figure 5.18 - *Radial profiles of RMS axial gas velocities at seven stations along z axis at* $M_j = 22\%$. *Symbols: experiment; solid line: AVBP-EL; dot-dashed line: CDP.*

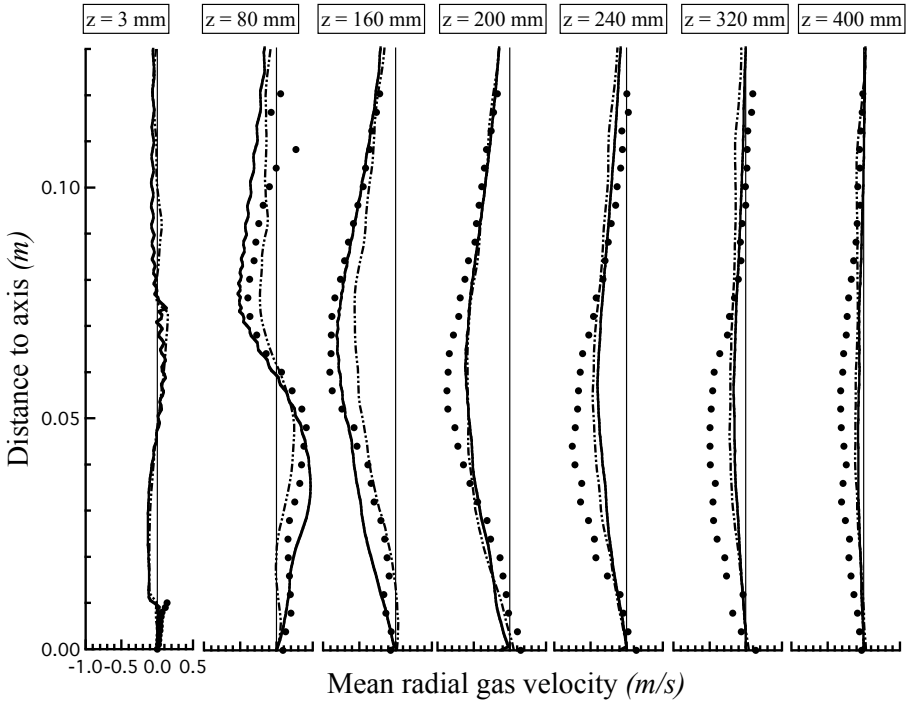


Figure 5.19 - *Radial profiles of mean radial gas velocities at seven stations along z axis at $M_j = 22\%$. Symbols: experiment; solid line: AVBP-EL; dot-dashed line: CDP.*

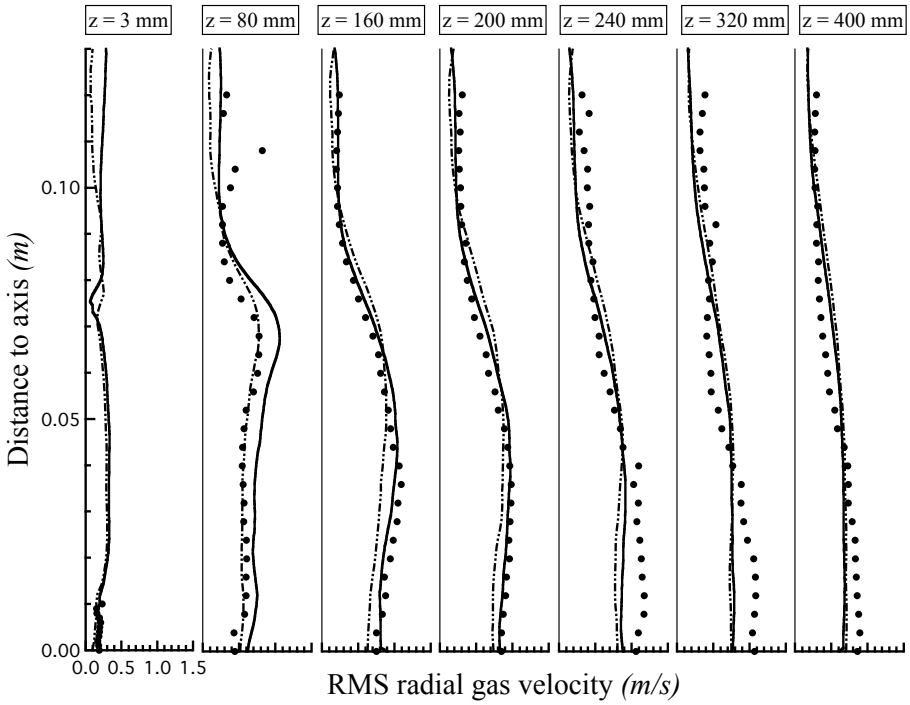


Figure 5.20 - *Radial profiles of RMS radial gas velocities at seven stations along z axis at $M_j = 22\%$. Symbols: experiment; solid line: AVBP-EL; dot-dashed line: CDP.*

5.4.2 Dispersed phase: results for $d_p = 60\ \mu$m

Figure 5.21 (a) shows the mean gas velocity vectors in the configuration of Borée *et al.* [22]. As mentioned in Subsection 5.1.1, the topology of this turbulent bluff-body flow depends on the velocity ratio of the inner and the annular pipes. In the present configuration, this ratio ($\overline{U}_j/\overline{U}_e = 0.83$) is the same as the one defined by Schefer *et al.* [190] for a low velocity ratio (see Fig. 5.2 (c)). The recirculation zone is evidenced by the blue velocity values. The counter-rotating eddies and the two stagnation points (along the z axis) can be distinguished inside this zone. Figure 5.21 (b) displays instantaneous particle distribution for this low velocity ratio. This figure is equivalent to Fig. 5.3 (c) where the gas flow was seeded with small particles to show the flow structure. As highlighted by Schefer *et al.* [190], the instantaneous flow structure differs considerably from the time-averaged behaviour presented in Fig. 5.21 (a) and recirculation patterns rarely appear instantaneously.

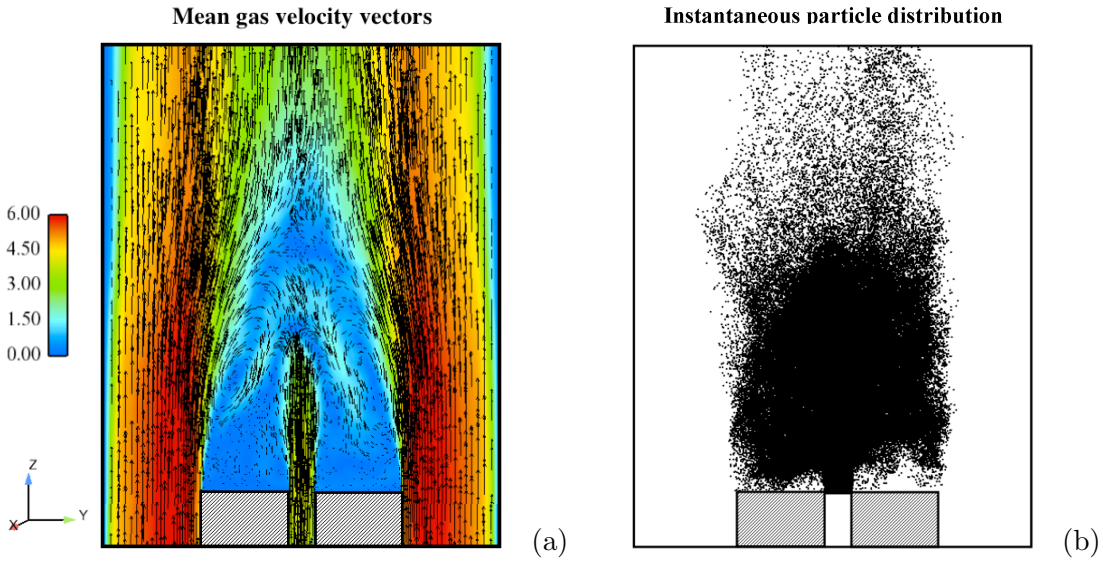


Figure 5.21 - *(a) Mean gas velocity vectors and (b) instantaneous particle distribution in the configuration of Borée* et al. *[22] ($\overline{U}_j/\overline{U}_e = 0.83$).*

Figure 5.22 shows a snapshot of the gas velocity modulus in the cutting plane $x = 0$. The corresponding particle locations for a 4 mm slice are superimposed on the turbulent velocity field[1]. Inside the recirculating zone there is a region where particles accumulate and must stop before turning around to escape from the recirculating flow by the sides. Particles are captured by large-scale structures, convected downstream and spread in the axial and radial directions. Figure 5.23 shows four cross-sections of the instantaneous field of the gas velocity modulus at different stations: (a-d) $z = 80, 160, 200$ and 240 mm, respectively. The corresponding particle locations for a 4 mm slice are superimposed on the turbulent velocity field. A reduction in turbulent velocity field can be observed by comparing the evolution of the central region of the four stations. In the first station, particles are mainly concentrated in the core region and move downstream. External particles are inside the recirculating zone and move upstream entrained by the bluff body to the sides. The internal dynamics of the recirculating zone is responsible for an efficient radial dispersion of the glass beads. Particles tend to be less concentrated in stations (c) and (d) since most of them are still trapped inside the recirculation zone.

[1] See animation on http://www.cerfacs.fr/cfd/movies/BB-AVBP_EL-MG.mov.

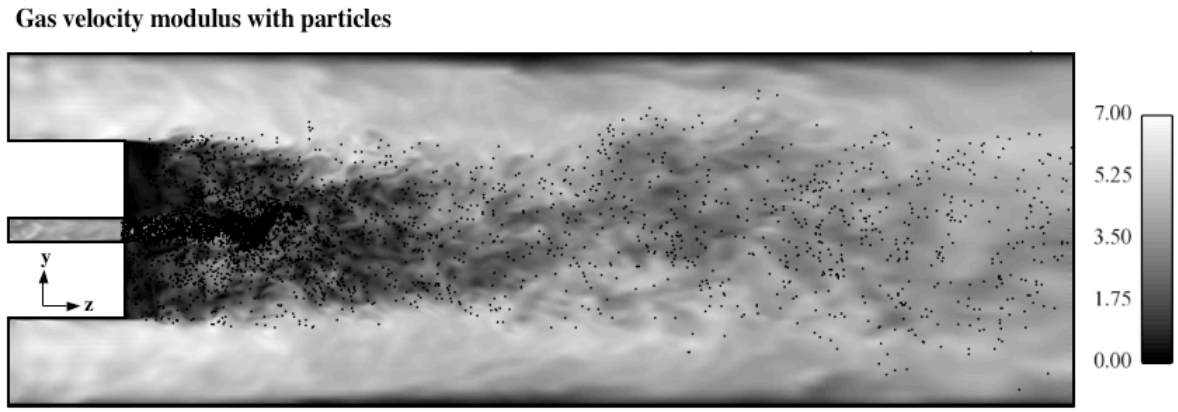


Figure 5.22 - *Instantaneous field of gas velocity modulus with particles superposed in the cutting plane* $x = 0$.

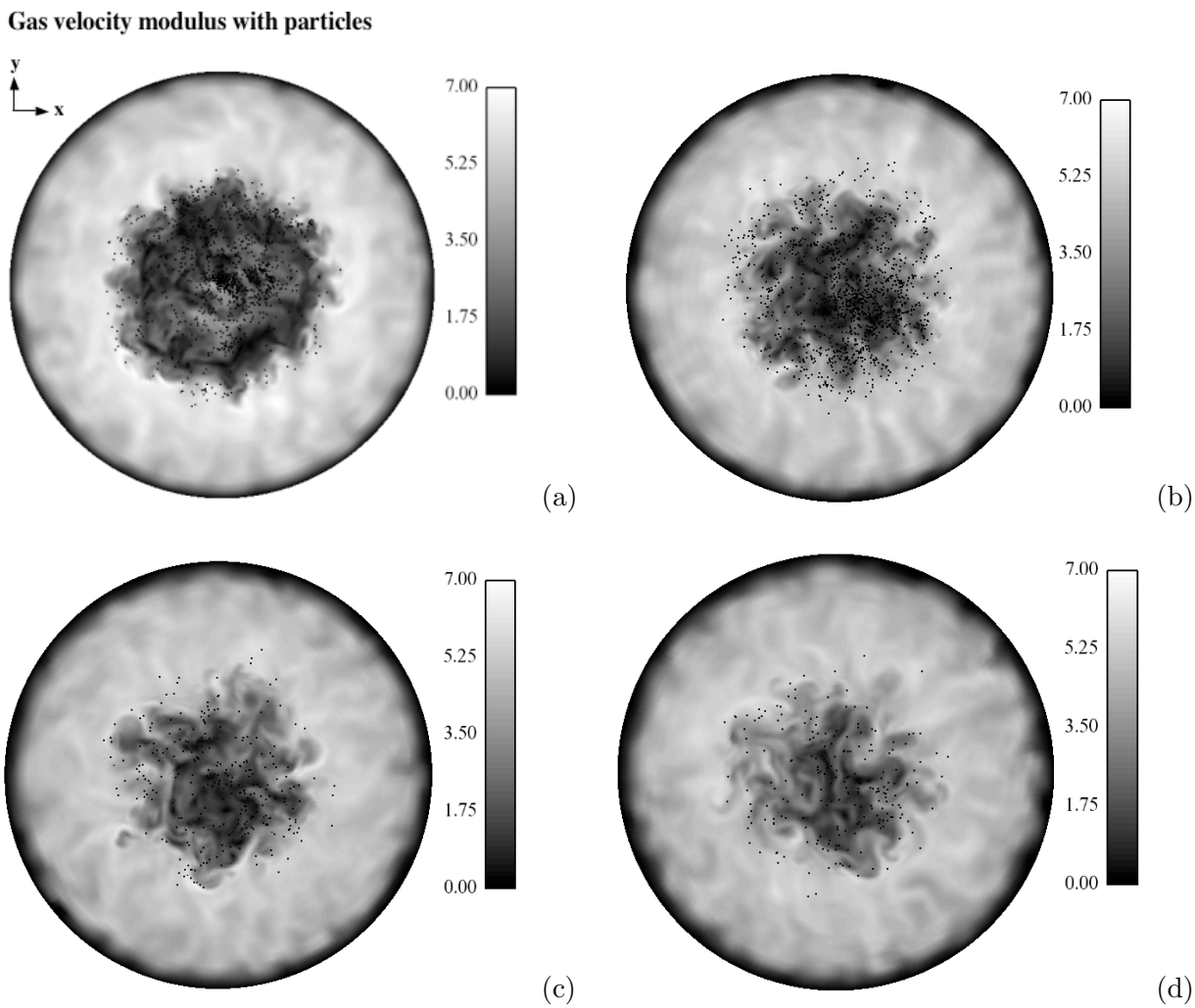


Figure 5.23 - *Cross-sections with the instantaneous field of gas velocity modulus and particles superposed at different stations: (a)* $z = 80$ *mm, (b)* $z = 160$ *mm, (c)* $z = 200$ *mm, (d)* $z = 240$ *mm.*

An example of particle velocities is presented in Fig. 5.24. Particles are colored by their axial velocity. The inner jet region is clearly defined with axial velocities of the order of 3-4 m/s. Particles within the recirculating region are recognized by the negative values of the axial velocity. An important accumulation of particles can be observed at the edge of the jet where flow and particles are roughly stopped. The coflow limits the dispersion of particles to a region where $r < 80$ mm and only a few number of particles are able to cross this zone entrained by large-scale structures. Downstream of the first stagnation point, particles are dispersed more efficiently.

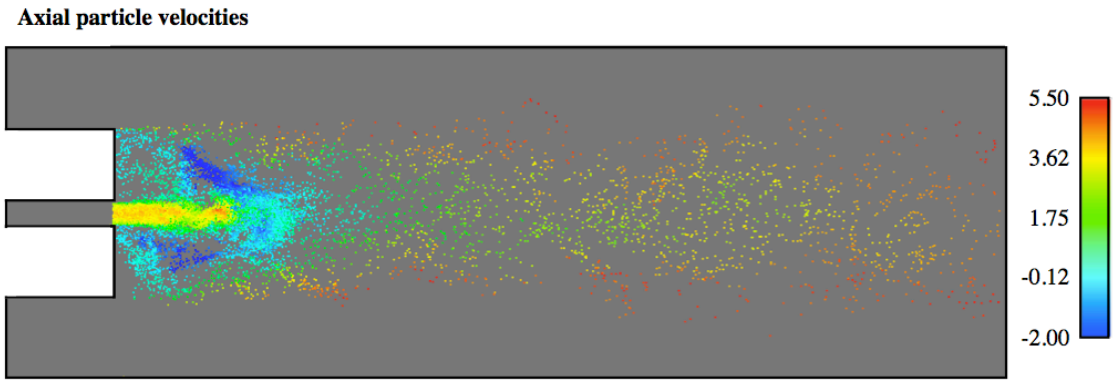


Figure 5.24 - *Instantaneous axial particle velocities in the cutting plane $x = 0$.*

Figs. 5.25 and 5.26 display 40 tracks of 60 μm particles at a moderate mass loading and for different physical times. Left and right panels present the projections of particle pathlines in the $y-z$ and $y-x$ axes, respectively. Figure 5.25 corresponds to the physical time $t = 0.05$ s. Initially, all particles follow the same direction and axial velocity is the dominating component of the flow. Particles are mainly located in a region with equivalent size as the inner jet pipe ($R_j = 10$ mm). The top panel of Fig. 5.26 ($t = 0.12$ s) clearly shows the critical zone located around $z = 150$ mm where particles accumulate and must stop before turning around to escape from the recirculating flows by the sides. In the middle panel we can observe that some particles reach the wall between the inner and annular jets and rebound ($z = 0$). Afterwards, they are trapped again into the recirculation bubble and move downstream. Their movement is completely controlled by the large-scale structures of the flow.

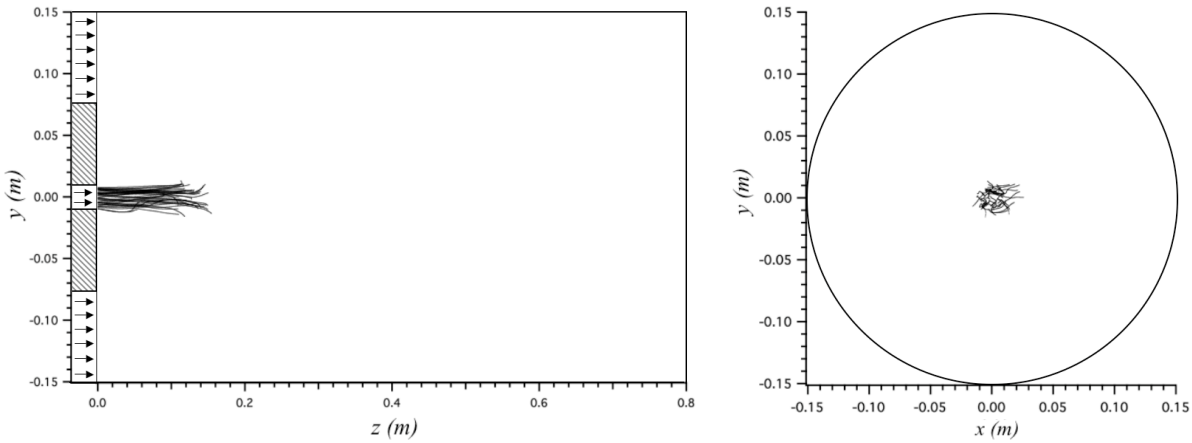


Figure 5.25 - *Some particle trajectories calculated at $M_j = 22\%$ for $d_p = 60$ μm at $t = 0.05$ s. Left: y-z view. Right: y-x view.*

The bottom panel of Fig. 5.26 displays complete particle pathlines for a physical time $t = 0.48$ s. We can observe that all particles have left the chamber keeping a maximum distance from the central axis and staying far from the external walls of the chamber.

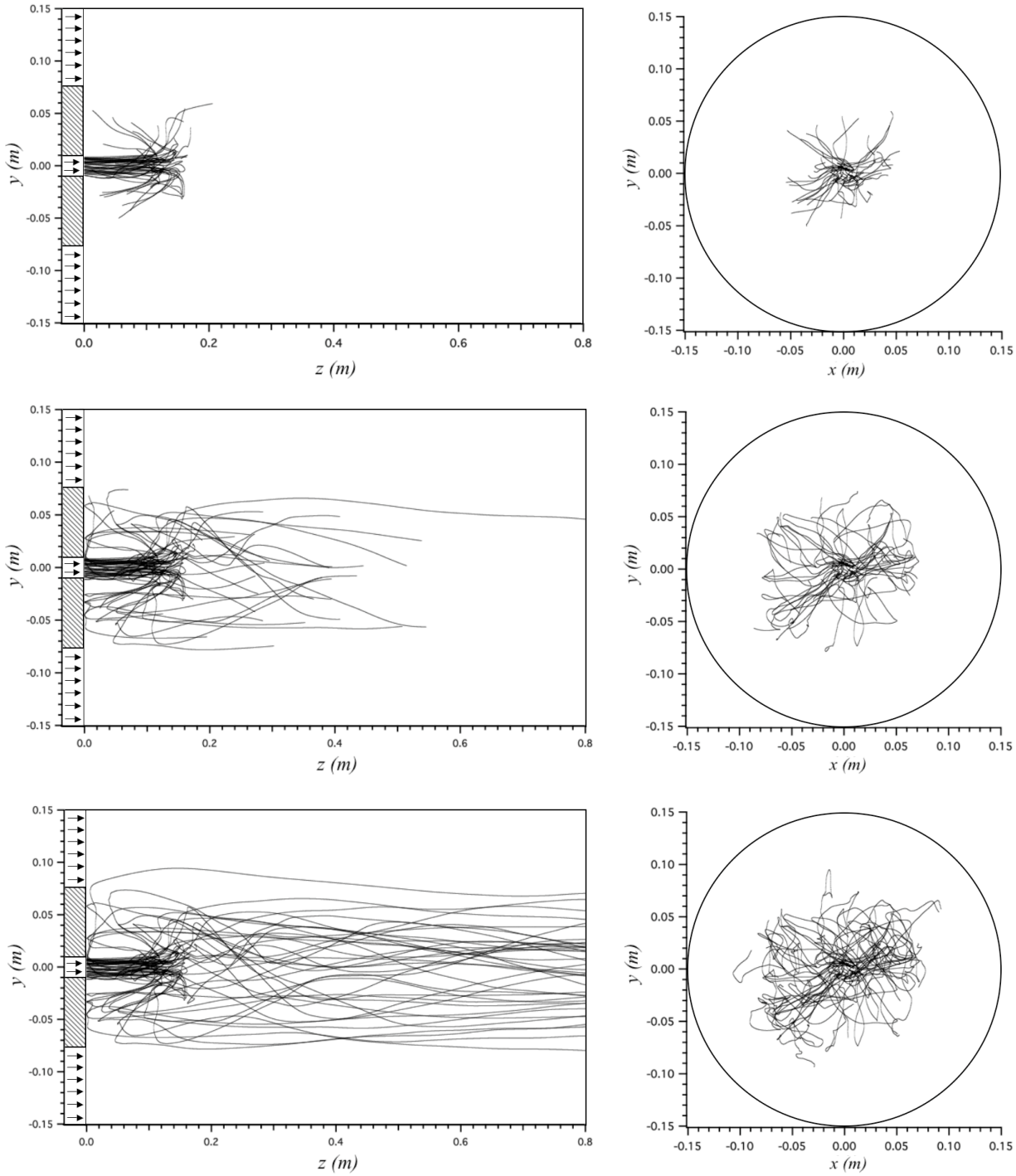


Figure 5.26 - *Some particle trajectories calculated at $M_j = 22\%$ for $d_p = 60$ μm. Top panel: $t = 0.12$ s; middle panel: $t = 0.25$ s; bottom panel: $t = 0.48$ s. Left: y-z view. Right: y-x view.*

The evolution of axial and radial profiles of the dispersed phase is presented hereafter. Figure 5.27 shows the axial evolution of the mean (a) and RMS (b) particle velocities. The solver CDP reproduces well particle velocity levels at the inner jet and it captures very well the mean velocities after the small recirculation zone. On the other hand, AVBP-EL has some difficulties to match the recirculation zone, which is almost absent. This discrepancy seems to be related to the different prediction of the gas flow values and the consequent mismatch of the recirculation zone already observed in Fig. 5.15 (a). On the other hand, RMS profiles obtained in both solvers are quite similar and close to the measurements except at the outlet of the inner jet where the white noise introduced for the dispersed phase seems to be smaller than the natural particle fluctuations values in that place. Particles recover the correct levels of turbulence after 40 mm due to their relative high inertia (St = 3.9, see Table 5.1) and predicted levels and profiles are in good agreement with the experiments downstream.

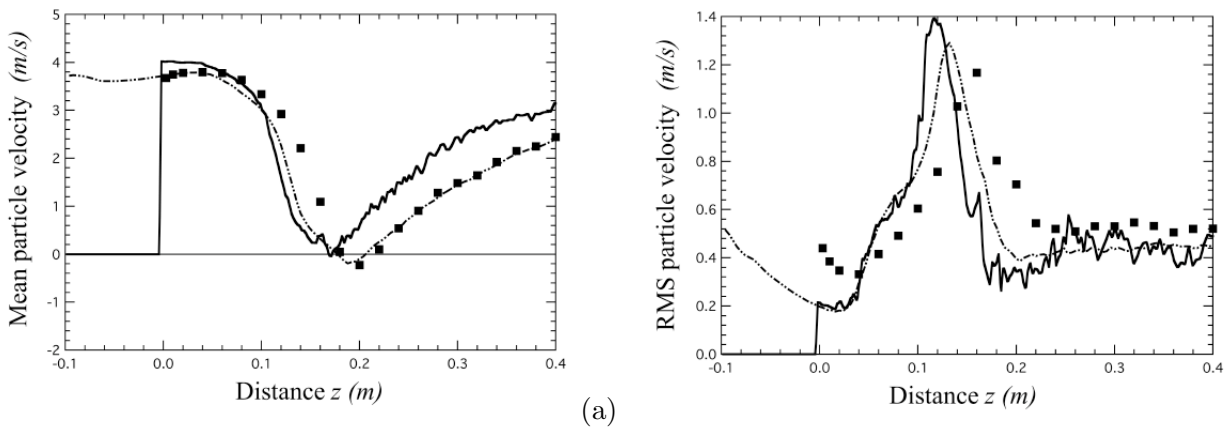


Figure 5.27 - *Axial evolution of mean (a) and RMS (b) particle velocities at $M_j = 22\%$ for $d_p = 60$ μm. Symbols: experiment; solid line: AVBP-EL; dot-dashed line: CDP.*

Results for the radial profiles of the dispersed phase are plotted in Figs. 5.28-5.31. Despite the differences observed in Fig 5.27 to match the recirculation bubble, the overall agreement of both solvers with the experiments is very good for the axial and radial particle profiles. These differences justify the higher velocity values observed in Fig. 5.28 (mainly at $z = 200, 240, 320$ and 400 mm) for the AVBP-EL results. Note however that RMS values of particle velocities in the outer region ($r > 80$ mm) are poorly evaluated because the number of samples is not high enough. The most sensitive quantity is the axial RMS velocity (Fig. 5.29) in a zone where mean velocity values are high and large-scale structures dominate particle movement. A better prediction of RMS profiles in this small area can be obtained by increasing the averaging time so as to get more samples. However, this is not crucial for this analysis. In fact, experimental results finish approximately at this distance also due to the reduced number of samples of this zone. The influence of the number of samples in the mean and RMS velocity profiles will be studied with the polydisperse case in Subsection 5.5.4.

To conclude, we can observe that even if the dispersed phase has been represented with only one class, its dynamics at a moderate mass loading has been quite well captured and the mean flow effects on the gas (through two-way coupling) are taken into account. This leads to good agreement in the comparison between both solvers and the experiments. However, a more complete study of the polydisperse case is presented in the next subsection to highlight eventual differences that could appear in the flow structure and to study particle dispersion of the different classes.

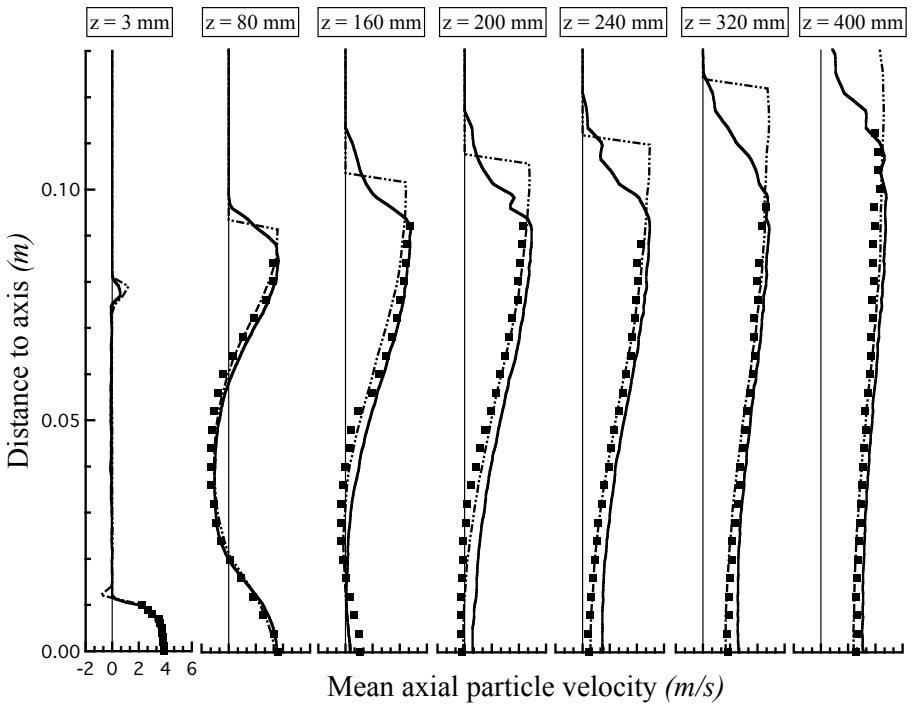


Figure 5.28 - *Radial profiles of mean axial particle velocities at seven stations along z axis at $M_j = 22\%$ for $d_p = 60$ μm. Symbols: experiment; solid line: AVBP-EL; dot-dashed line: CDP.*

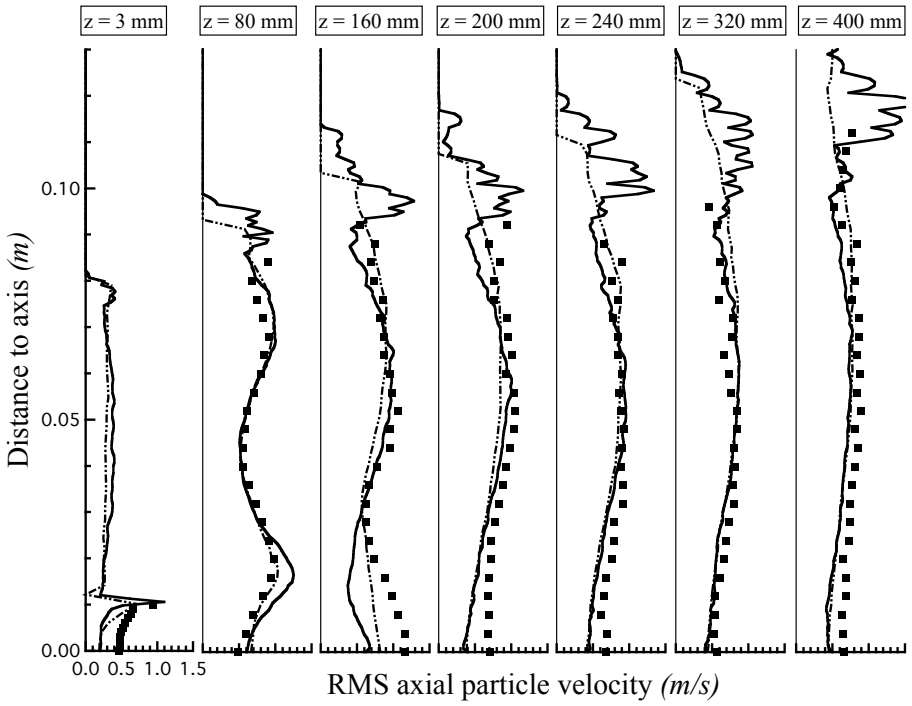


Figure 5.29 - *Radial profiles of RMS axial particle velocities at seven stations along z axis at $M_j = 22\%$ for $d_p = 60$ μm. Symbols: experiment; solid line: AVBP-EL; dot-dashed line: CDP.*

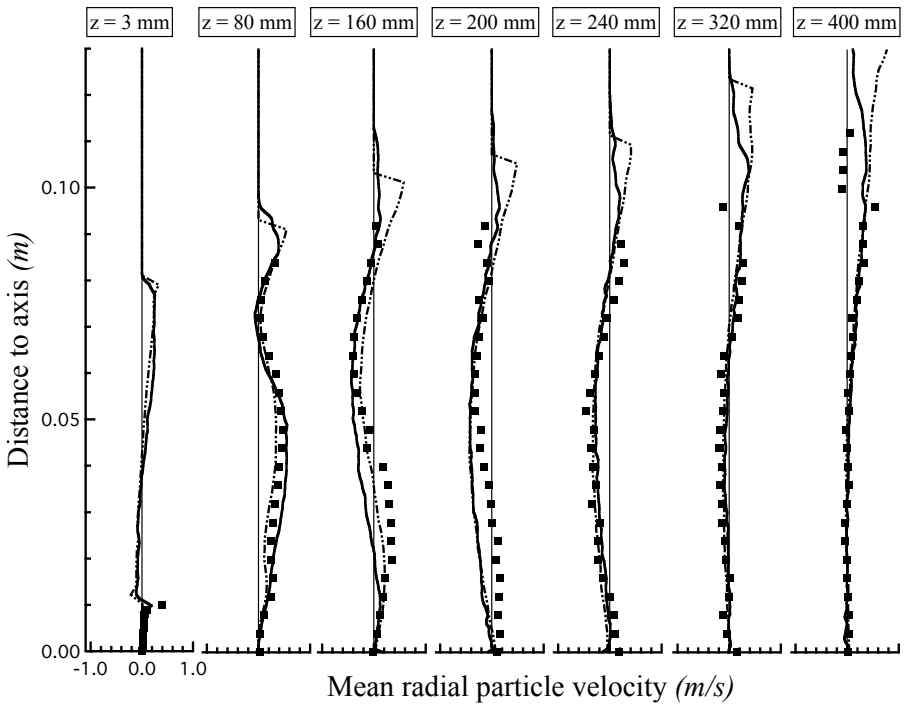


Figure 5.30 - *Radial profiles of mean radial particle velocities at seven stations along z axis at* $M_j = 22\%$ *for* $d_p = 60$ μm. *Symbols: experiment; solid line: AVBP-EL; dot-dashed line: CDP.*

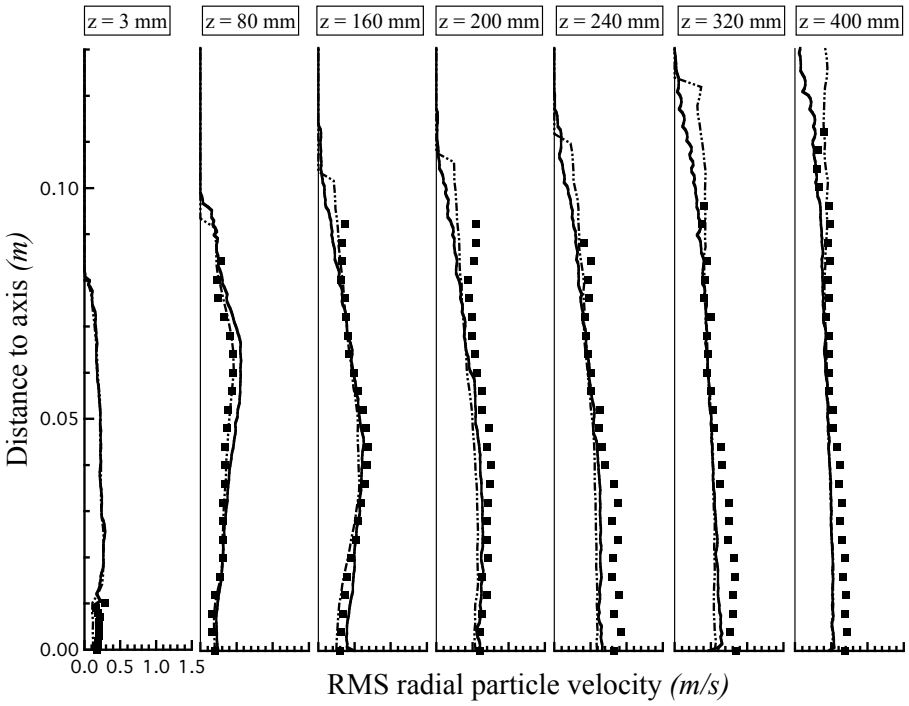


Figure 5.31 - *Radial profiles of RMS radial particle velocities at seven stations along z axis at* $M_j = 22\%$ *for* $d_p = 60$ μm. *Symbols: experiment; solid line: AVBP-EL; dot-dashed line: CDP.*

5.5 Results for two-phase flow polydisperse case

In this section, the downstream evolution of the air and particulate flow fields at a moderate mass loading ($M_j = 22\%$) for a polydisperse test case is presented and discussed. Numerical results of the AVBP-EL and CDP solvers are compared to experiments. Axial and radial profiles of the gaseous and dispersed phases are presented in Subsections 5.5.1-5.5.3. Subsection 5.5.4 discusses the influence of the number of samples on the axial and radial velocity profiles. Particle-dispersion characteristics are analysed in Subsection 5.5.5, focusing on the dependence of particle trajectories and the slip velocities upon particle sizes.

Figure 5.32 shows the initial particle number distribution used by both solvers at the corresponding injection plane. Levels are quite similar to the experimental ones (see Fig. 5.4 (b)) except for the two smallest classes ($d_p = 20$ and 30 μm) but this is assumed to have minor effects in the simulation due to their negligible contribution to the overall mass distribution (see Fig. 5.4 (a)). Both solvers use the same particle injection parameters in order to make easier the comparison of particles profiles. Note however, that the injection planes are not the same (see Fig. 5.14). For the sake of clarity, only 4 classes are analysed in the following: $d_p = 20, 40, 60$ and 80 μm.

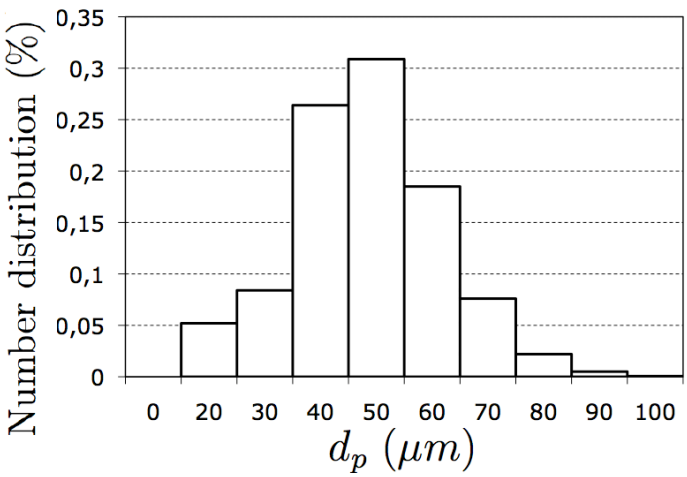


Figure 5.32 - *Initial number distribution of the particle size injected numerically by both solvers.*

5.5.1 Gaseous phase

Figure 5.33 presents the axial evolution of the mean (a) and RMS (b) velocity of the gaseous phase. At first sight, there is a clear similarity between these results and the ones obtained for the LES solvers in the monodisperse case (see Fig. 5.15). The difference observed between AVBP-EL and CDP in the location of the recirculation zone is still the same as in the monodisperse simulation. Again, CDP predicts better its location and the AVBP-EL solver displays a difference in the prediction of the first and second stagnation points located respectively, 40 and 60 mm before the experimental values. As mentioned for the monodisperse calculation, the prediction of stagnation points is a critical issue in bluff-body simulations due to the sensitivity to the ratio between the mean velocity of the inner jet and the coflow. However, an important detail must be highlighted when comparing the monodisperse and the polydisperse simulations: the levels of mean gas velocity

in the recirculation zone are lower than expected (-1 m/s instead of -1.4 m/s for the peaks near $z \approx 180$ mm) with both codes (Fig. 5.33 (a)). This implies a reduction in the size of the recirculation bubble and it has an effect in the axial velocity profiles of the different particle classes (as discussed in the next subsection). It can be observed that the location of the maximum RMS in Fig. 5.33 (b) (first stagnation point) has not changed. The levels of RMS at $z > 180$ mm are lower than the experimental values and lower than the ones obtained for the monodisperse simulation; however, AVBP-EL seems to capture the small variation between 180 mm $< z <$ 280 mm even if the level of turbulence fluctuations is not the same.

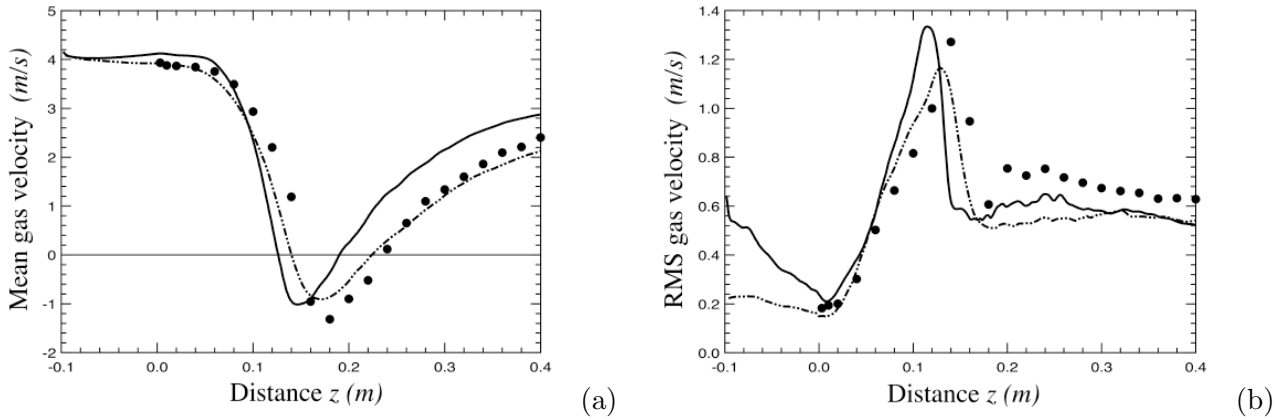


Figure 5.33 - *Axial evolution of mean (a) and RMS (b) gas velocities at $M_j = 22\%$. Symbols: experiment; solid line: AVBP-EL; dot-dashed line: CDP.*

Figs. 5.34-5.37 show the radial profiles of the mean and RMS, axial and radial gas velocities. In Fig. 5.34, the negative values of the mean axial gas velocity profiles at $z = 80$ mm and $z = 160$ mm indicate the location and radial extent of the recirculation zone. The second stagnation point near $z = 240$ mm is visible for the experimental profiles but it is located close to $z = 200$ mm for the AVBP-EL results. Mean radial gas profiles (Fig. 5.36) show only negative values for $z > 160$ mm. This inward flow converges to the centerline where values are close to zero. The reduction of radial velocity values and their convergence indicate also the boundary of the recirculation bubble which is associated to a radial compression. Both solvers have some difficulties to capture the maximum of the negative values in the last four cross-sections, probably due to a lower prediction of pressure values. RMS axial and radial profiles (Figs. 5.35 and 5.37) are similar to the experimental values.

These radial profiles are almost exactly the same as the ones presented in Section 5.4 for the monodisperse case (see Figs. 5.17-5.20). This may lead us to think that considering a monodisperse distribution is sufficient to capture the mean flow effects on the gas for the moderate mass loading. However, they do not reflect the reduction of the recirculation zone observed in Fig. 5.33.

As a technical remark, the averaging time to obtain particle profiles with the AVBP-EL solver in this polydisperse case ($t = 3.89$ s) is almost eight times the one considered for the monodisperse case, $t = 0.4642$ s (see Table 5.2). The number of samples of classes $d_p = 20$ and 80 μm would not be enough for converged statistics if the physical time was equal to the one used in the monodisperse case. To support this statement, results of the radial velocity of mean and RMS axial profiles for three different physical times: $t \approx 0.26$, 1 and 4 s, are presented in Subsection 5.5.4.

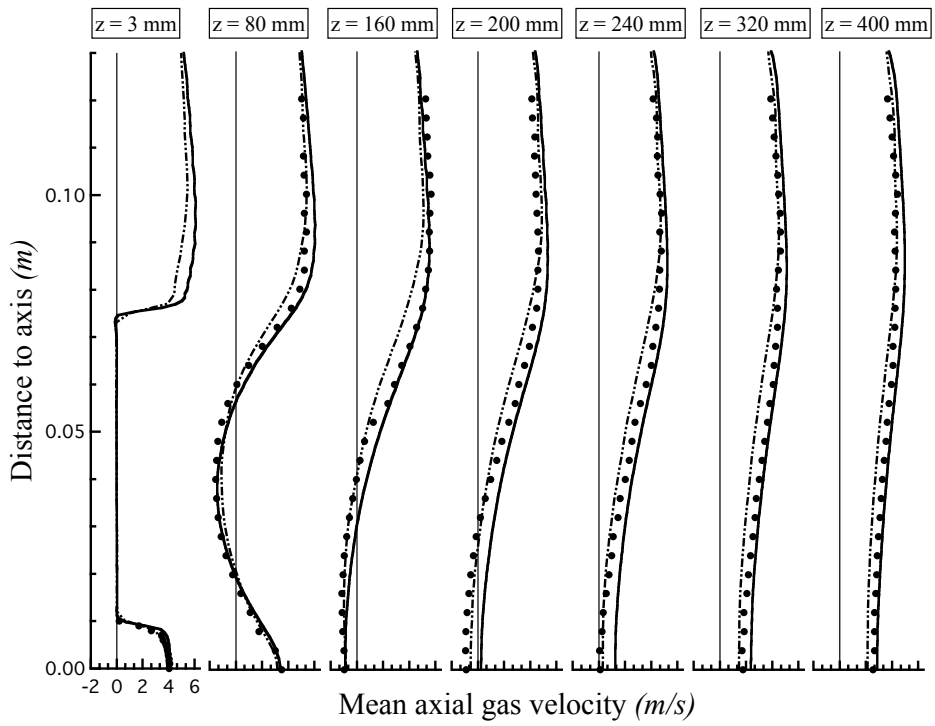


Figure 5.34 - *Radial profiles of mean axial gas velocities at seven stations along z axis at $M_j = 22\%$. Symbols: experiment; solid line: AVBP-EL; dot-dashed line: CDP.*

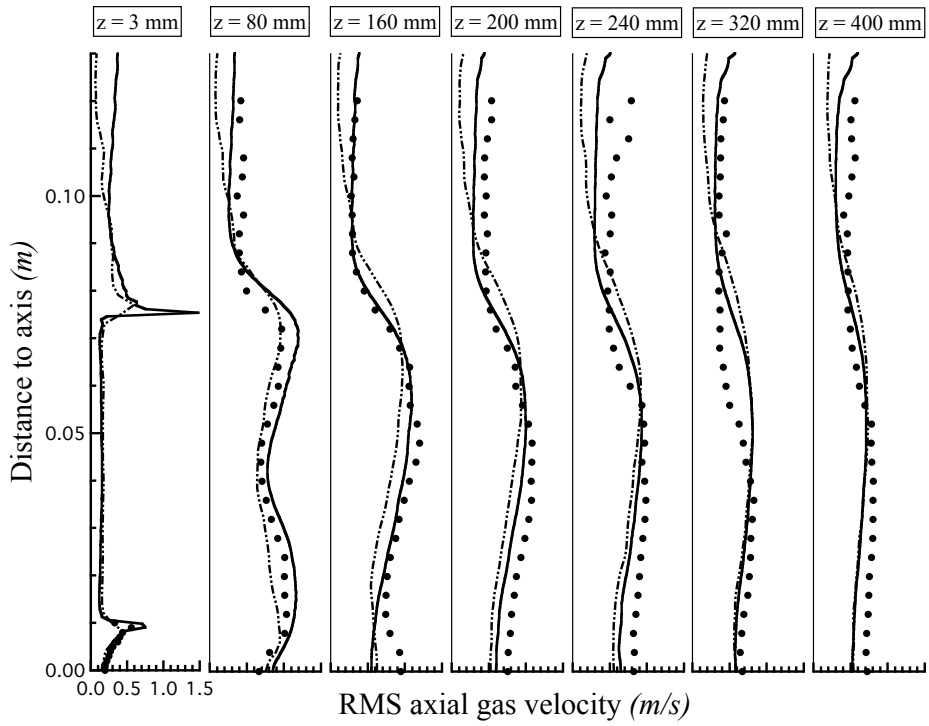


Figure 5.35 - *Radial profiles of RMS axial gas velocities at seven stations along z axis at $M_j = 22\%$. Symbols: experiment; solid line: AVBP-EL; dot-dashed line: CDP.*

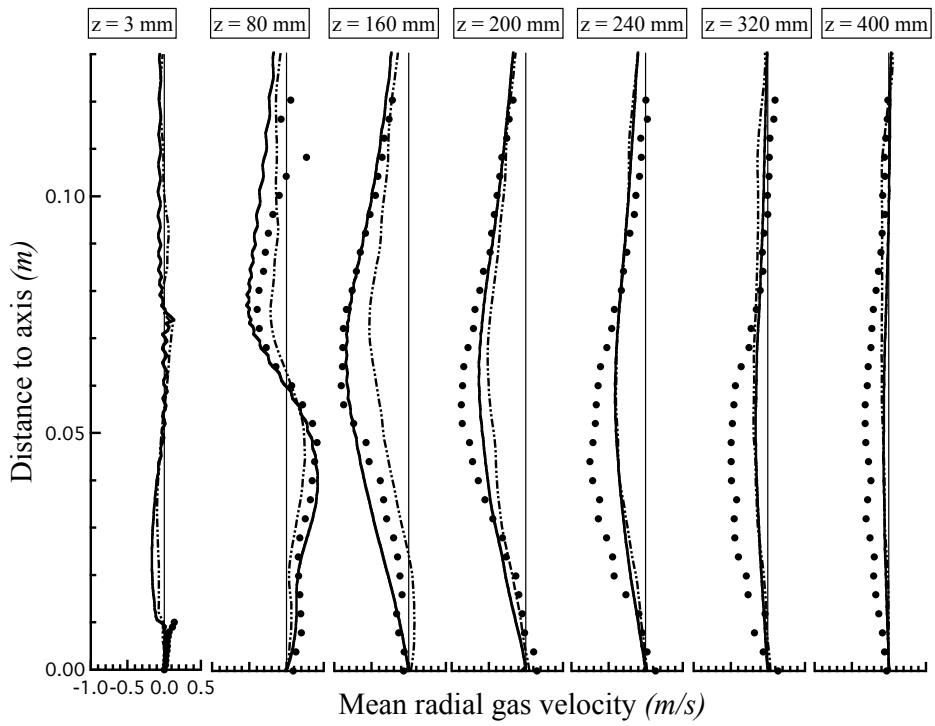


Figure 5.36 - *Radial profiles of mean radial gas velocities at seven stations along z axis at $M_j = 22\%$. Symbols: experiment; solid line: AVBP-EL; dot-dashed line: CDP.*

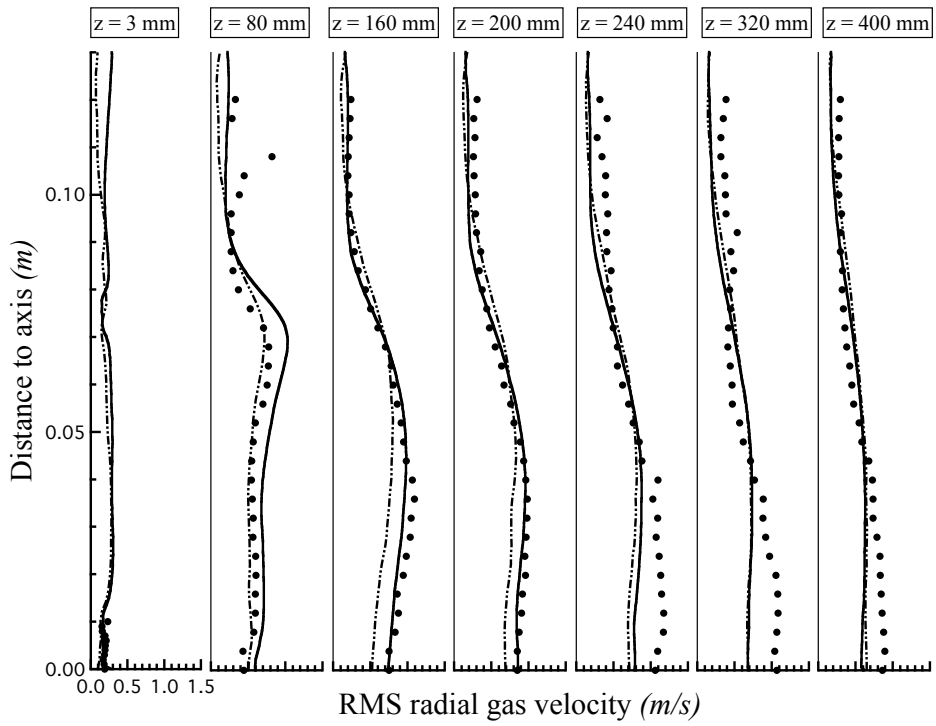


Figure 5.37 - *Radial profiles of RMS radial gas velocities at seven stations along z axis at $M_j = 22\%$. Symbols: experiment; solid line: AVBP-EL; dot-dashed line: CDP.*

5.5.2 Dispersed phase: axial velocity profiles

Regarding the results of the dispersed phase, the motion of the smallest particles with diameter $d_p = 20$ μm is expected to be very different from the largest ones, with diameter $d_p = 80$ μm. While the smallest particles (Fig. 5.38 (a)) almost follow the gas flow (see Fig. 5.33 (a)) the inertia of the largest particles (Fig. 5.41 (a)) decorrelates them from the fluid flow as can be observed in the axial evolution of the mean particle velocities. The reduction in the extent and location of the recirculation zone observed by the particles (near $z = 200$ mm) is evident while comparing Figs. 5.38 (a)-5.41 (a). CDP is able to better capture this recirculation bubble with similar results than the experimental ones, however, the delay in the occurrence of the recirculation zone for the gaseous phase with AVBP-EL (see Fig. 5.33) is still visible in these particle velocity profiles.

Another trace of the different particle inertia effects can be observed near the exit of the inner pipe ($0 < z < 60$ mm) in Figs. 5.38 (b)-5.41 (b). In spite of differences in the injection location (see Fig. 5.14), neither CDP, nor AVBP-EL display the accurate levels of particle fluctuations at the exit of the inner pipe but particle behaviour is the same in both solvers while trying to capture the RMS values in the first millimeters of the jet exit. The smallest particles adapt very quickly to the flow fluctuations, mid-size particles take around 40 mm and the largest ones need more than 60 mm to achieve the same level as the one detected by the experiments.

As mentioned for other graphs of this section, AVBP-EL mean and RMS profiles results look more scattered than CDP profiles (mainly for the classes $d_p = 20$ and 80 μm) due to a lower number of samples. Nevertheless, we emphasize that both solvers present the same differences in the RMS values at $z > 200$ mm where numerical results are 20% lower than experimental ones. Following Borée *et al.* [22], particle velocity fluctuations in this region seems to be controlled by the dragging of large-scale fluid turbulent motion, therefore, the differences observed may be related to an underestimation of these large-eddies effects.

Regarding the differences between the monodisperse and the polydisperse cases, particles with diameter $d_p = 60$ μm show quite similar profiles to the one presented in the monodisperse test case (see Fig. 5.27).

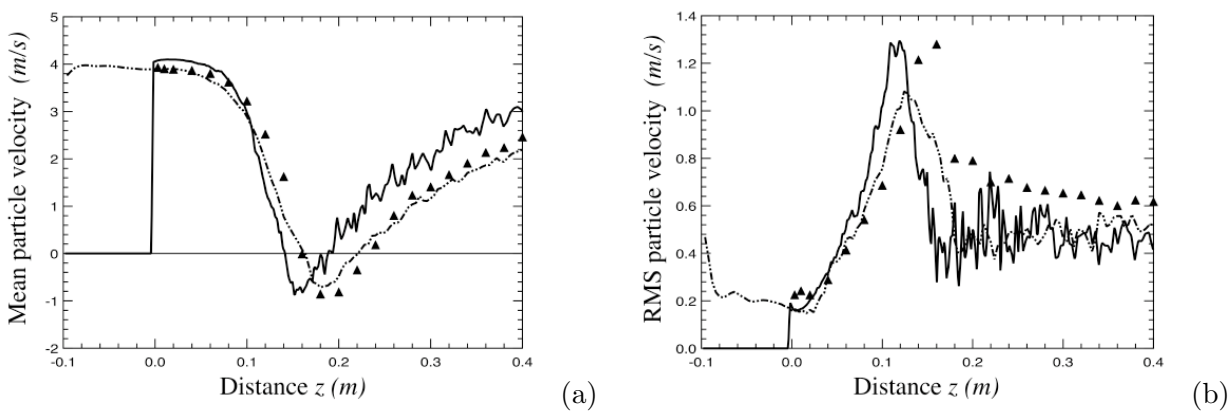


Figure 5.38 - *Axial evolution of mean (a) and RMS (b) particle velocities at $M_j = 22\%$ for $dp = 20$ μm. Symbols: experiment; solid line: AVBP-EL; dot-dashed line: CDP.*

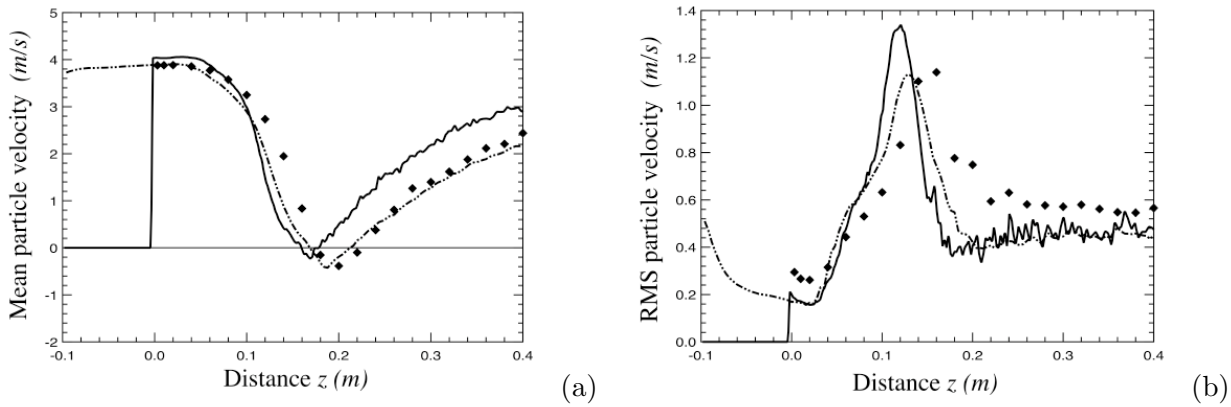


Figure 5.39 - *Axial evolution of mean (a) and RMS (b) particle velocities at $M_j = 22\%$ for $dp = 40$ μm. Symbols: experiment; solid line: AVBP-EL; dot-dashed line: CDP.*

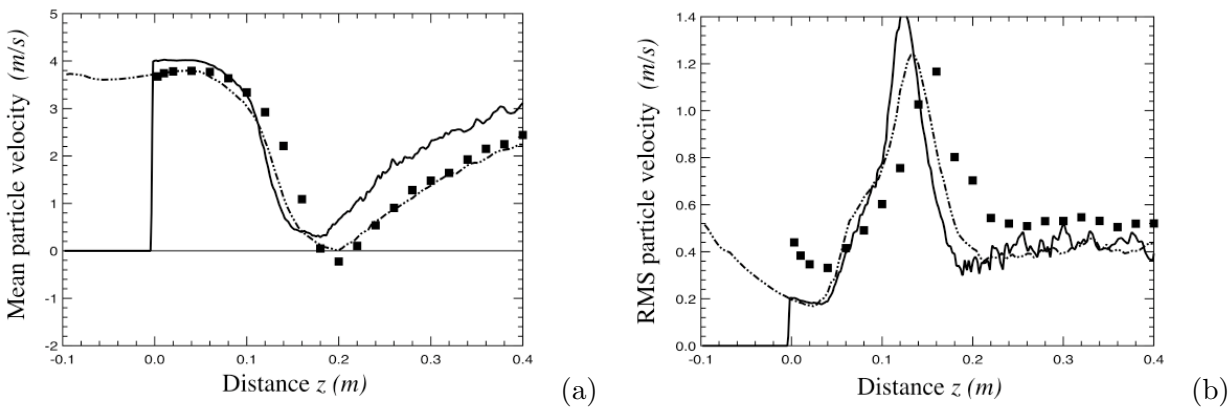


Figure 5.40 - *Axial evolution of mean (a) and RMS (b) particle velocities at $M_j = 22\%$ for $dp = 60$ μm. Symbols: experiment; solid line: AVBP-EL; dot-dashed line: CDP.*

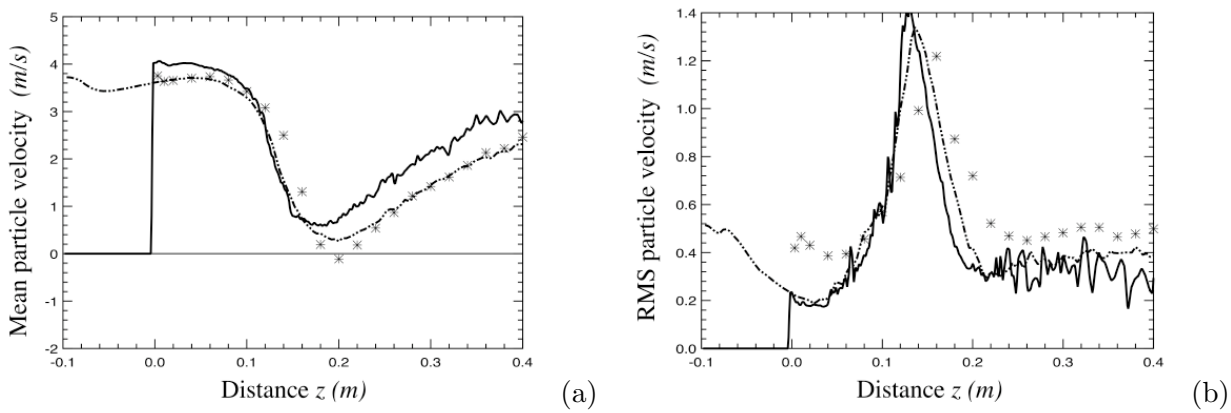


Figure 5.41 - *Axial evolution of mean (a) and RMS (b) particle velocities at $M_j = 22\%$ for $dp = 80$ μm. Symbols: experiment; solid line: AVBP-EL; dot-dashed line: CDP.*

5.5.3 Dispersed phase: radial velocity profiles

In the following subsection, the mean and RMS, axial and radial velocity profiles of the four classes previously mentioned are presented. Again, numerical results obtained with both Lagrangian solvers are compared to experiments.

Figs. 5.42-5.45: $d_p = 20$ μm

Mean axial velocity profiles (Fig. 5.42) of the smallest particle diameter are in good agreement with the experimental ones, with AVBP-EL giving better results at most stations. The lack of numerical samples in the outer region ($r > 80$ mm) is visible in some cross-sections. CDP profiles go farther since the averaging time is three times greater than the AVBP-EL averaging time. This lack of samples is more evident in the RMS axial profiles (Fig. 5.43). It is difficult to capture particles in this region since they are quickly accelerated by the large-scale structured presented in the separated region. Mean radial profiles (Fig. 5.44) of both solvers still underestimate the experimental results as can be expected since these particles behaves almost like the gaseous phase. RMS radial profiles (Fig. 5.45) are also quite similar to the experiments except in the centerline after the second stagnation point like for the gaseous phase.

Figs. 5.46-5.49: $d_p = 40$ μm

Mean axial velocity profiles (Fig. 5.46) are also in good agreement with experiments. The number of samples of the numerical results is greater than for the previous class, covering more experimental data than the smallest particles do. This can also be observed for the RMS results (Fig. 5.47). Results of the radial and RMS particle velocities (Figs. 5.48 and 5.49) are in very good agreement with experimental data and they are lower than for the 20 μm class since 40 μm particles are less controlled by the fluid flow.

Figs. 5.50-5.53: $d_p = 60$ μm

No relevant differences have been observed between these figures and the ones presented in the monodisperse case (see Figs. 5.28-5.31 and comments in Subsection 5.4.2).

Figs. 5.54-5.57: $d_p = 80$ μm

Mean axial velocity profiles (Fig. 5.54) of the largest particle diameter considered here are in good agreement with the experimental data. The lack of numerical samples in the outer region ($r > 70$ mm) is evident but even Borée *et al.* [22] make a reference to the poor statistical converge of large particles to limit duration of data acquisition. Results of the RMS axial particle velocities (Fig. 5.55) show scattered profiles due to this reason. It can be observed that the mean radial particle velocities (Fig. 5.56) are close to zero. This shows that the large particles motion is mainly controlled by the axial velocity component of the bluff-body flow. RMS radial profiles (Fig. 5.57) are also quite similar to the experiments, with highest values mainly on the axis and with some differences after $z > 240$ mm, as stated for the other classes.

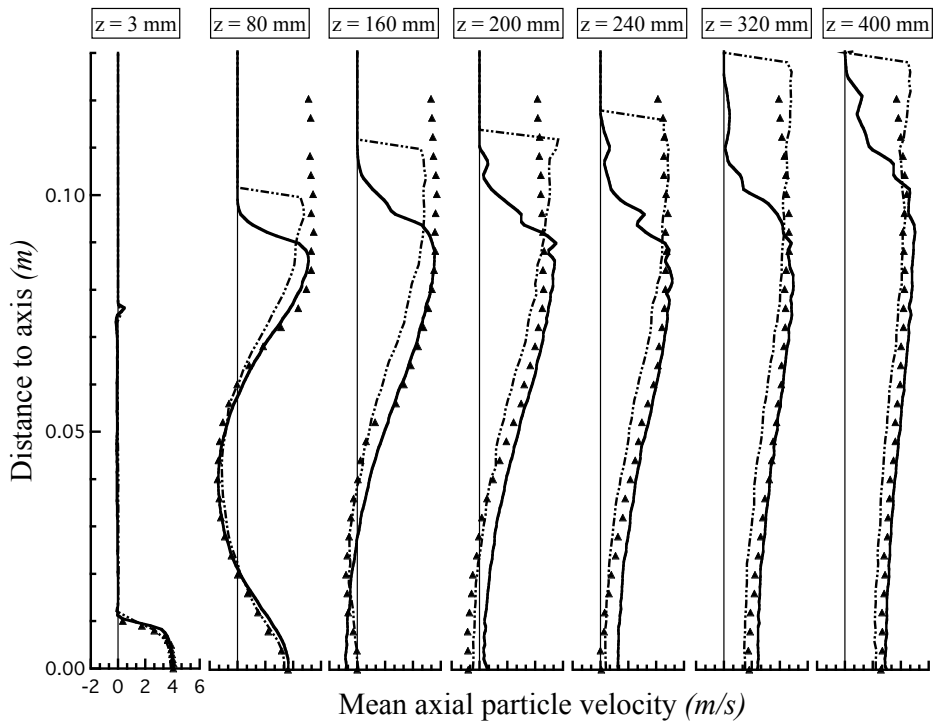


Figure 5.42 - *Radial profiles of mean axial particle velocities at seven stations along z axis at $M_j = 22\%$ for $d_p = 20$ μm. Symbols: experiment; solid line: AVBP-EL; dot-dashed line: CDP.*

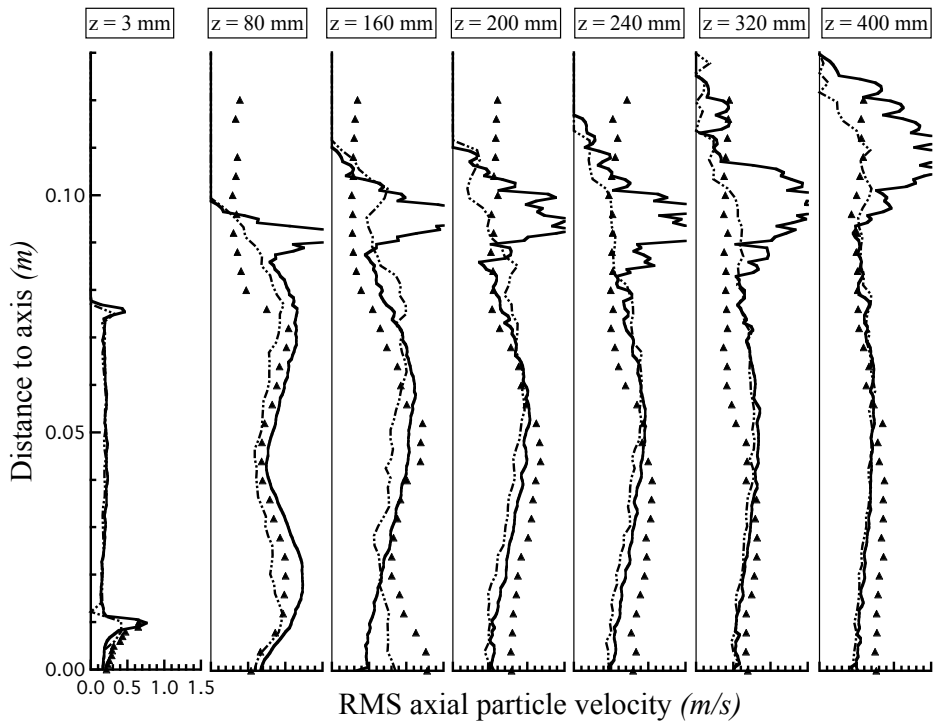


Figure 5.43 - *Radial profiles of RMS axial particle velocities at seven stations along z axis at $M_j = 22\%$ for $d_p = 20$ μm. Symbols: experiment; solid line: AVBP-EL; dot-dashed line: CDP.*

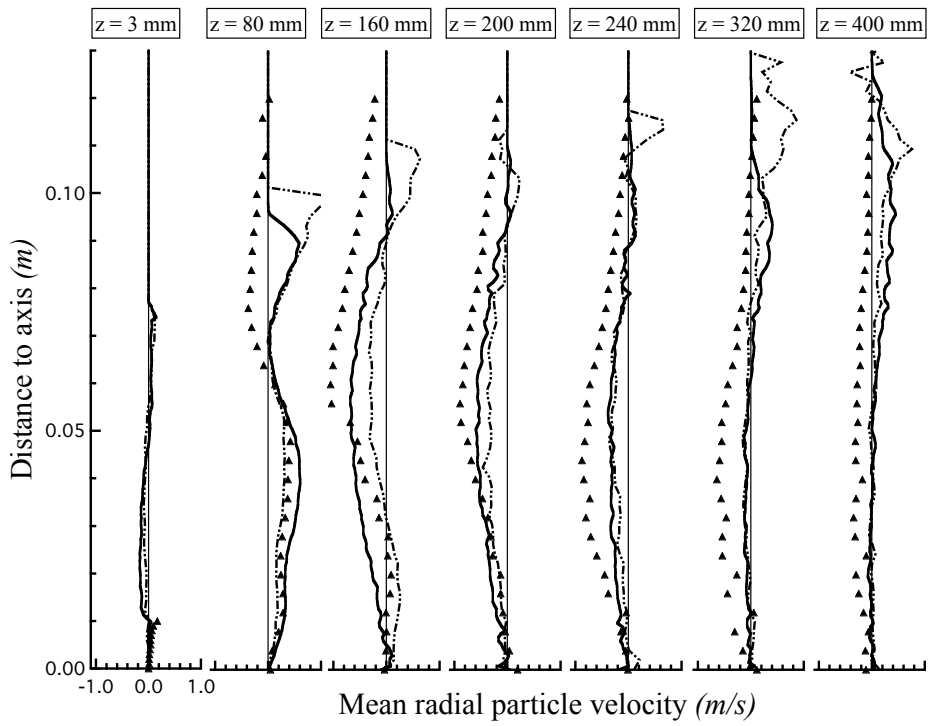


Figure 5.44 - *Radial profiles of mean radial particle velocities at seven stations along z axis at $M_j = 22\%$ for $d_p = 20$ μm. Symbols: experiment; solid line: AVBP-EL; dot-dashed line: CDP.*

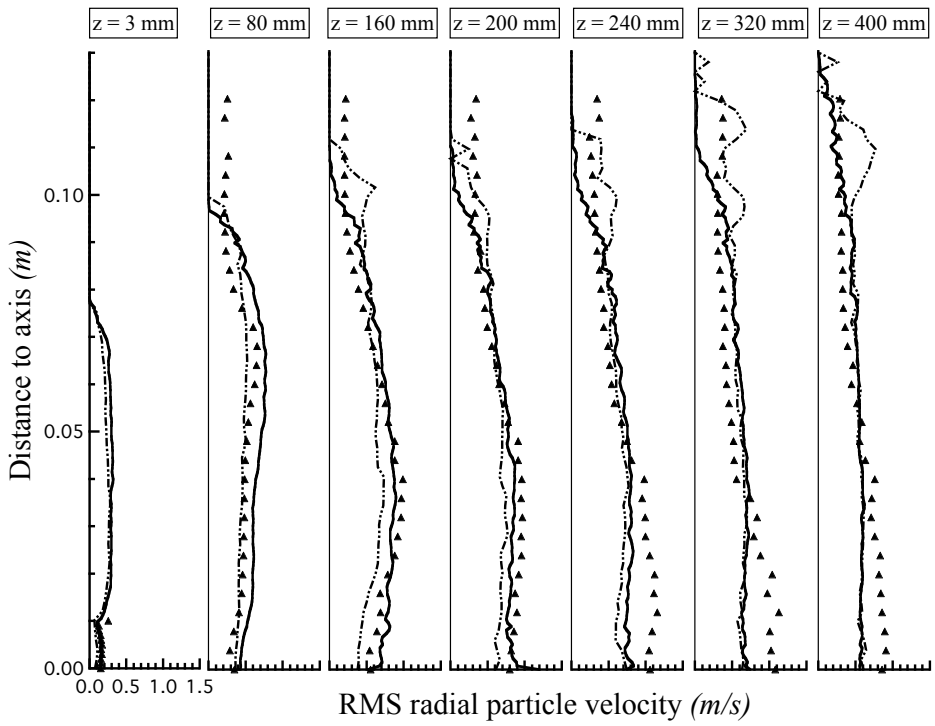


Figure 5.45 - *Radial profiles of RMS radial particle velocities at seven stations along z axis at $M_j = 22\%$ for $d_p = 20$ μm. Symbols: experiment; solid line: AVBP-EL; dot-dashed line: CDP.*

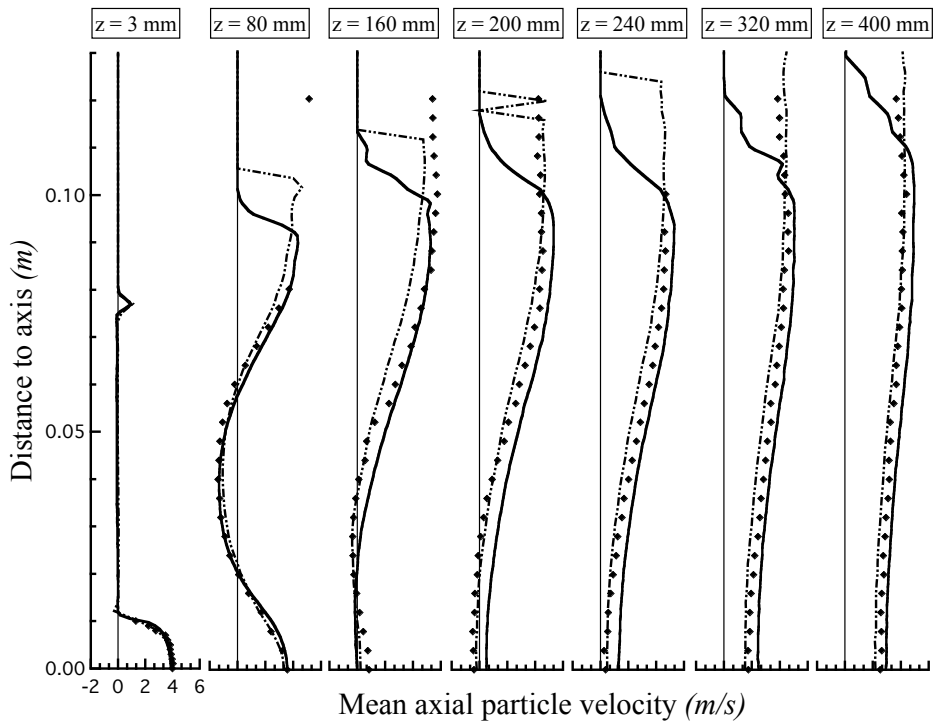


Figure 5.46 - *Radial profiles of mean axial particle velocities at seven stations along z axis at $M_j = 22\%$ for $d_p = 40$ μm. Symbols: experiment; solid line: AVBP-EL; dot-dashed line: CDP.*

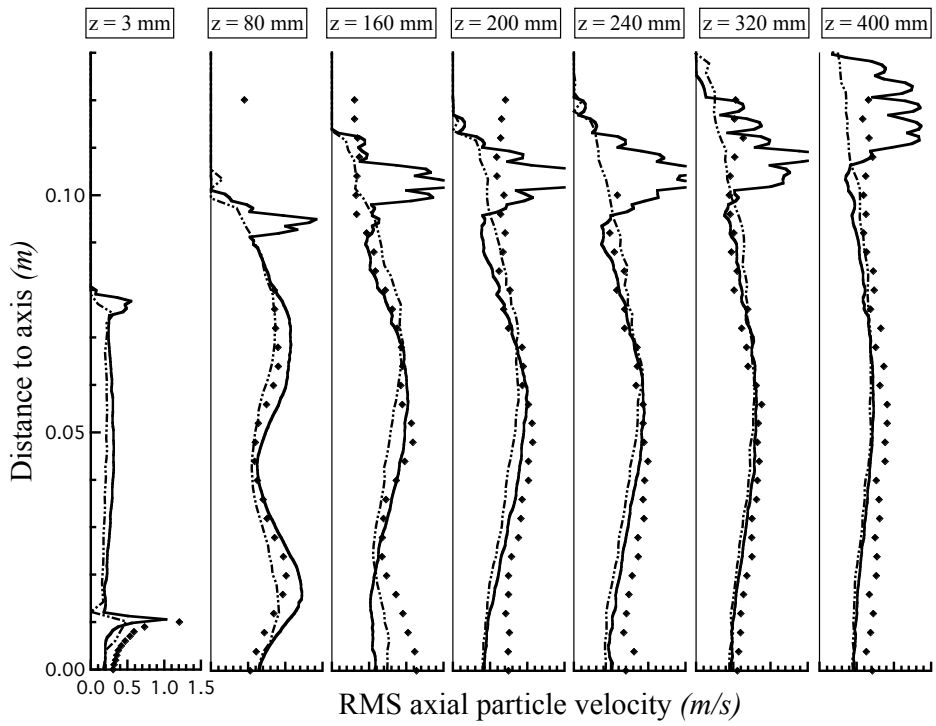


Figure 5.47 - *Radial profiles of RMS axial particle velocities at seven stations along z axis at $M_j = 22\%$ for $d_p = 40$ μm. Symbols: experiment; solid line: AVBP-EL; dot-dashed line: CDP.*

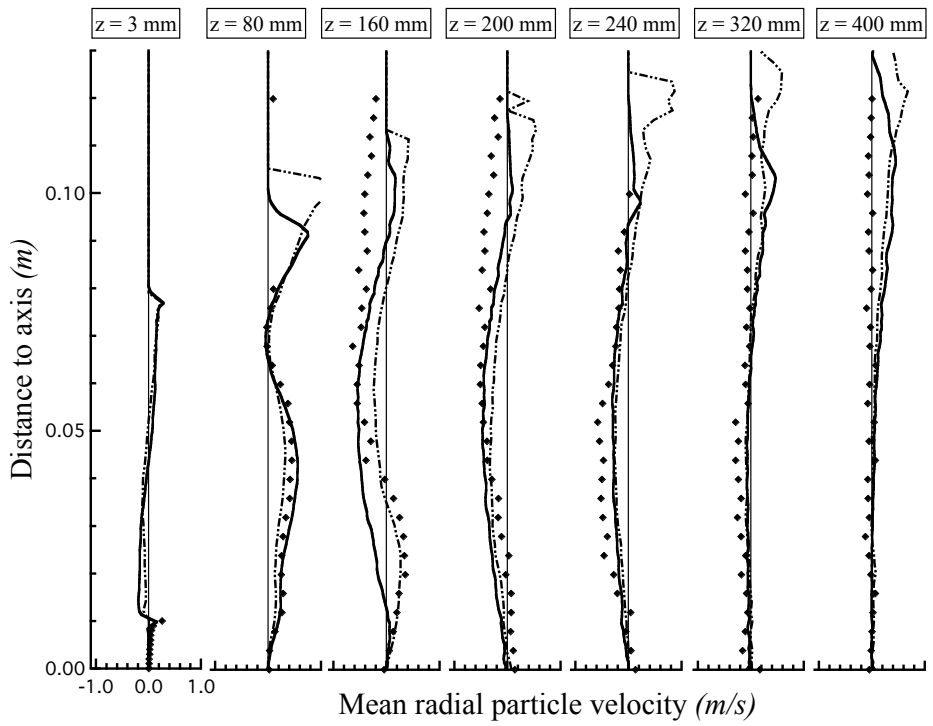


Figure 5.48 - *Radial profiles of mean radial particle velocities at seven stations along z axis at $M_j = 22\%$ for $d_p = 40$ μm. Symbols: experiment; solid line: AVBP-EL; dot-dashed line: CDP.*

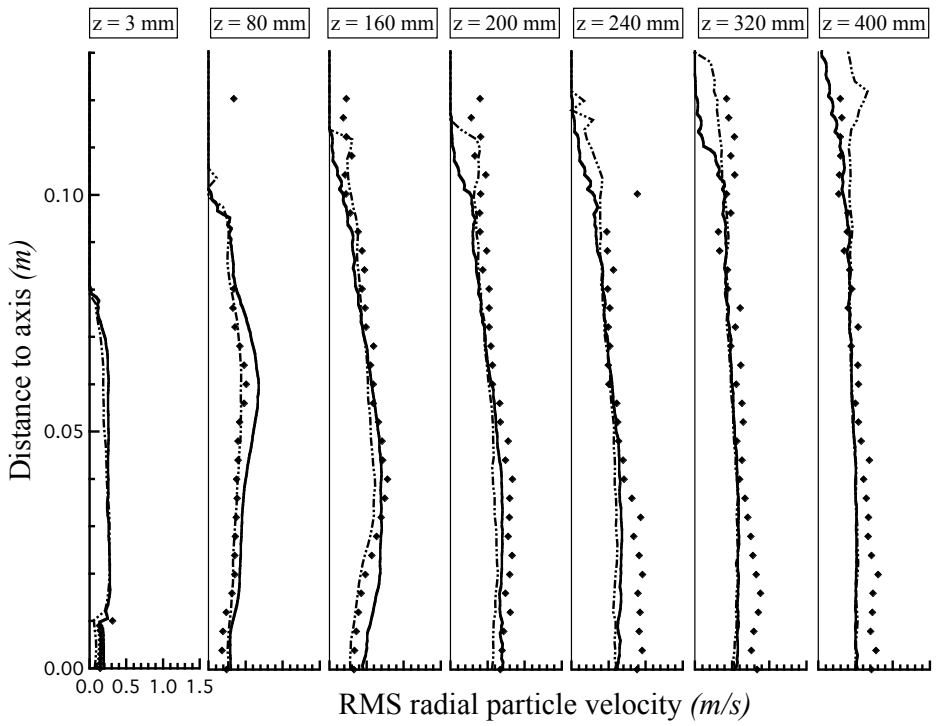


Figure 5.49 - *Radial profiles of RMS radial particle velocities at seven stations along z axis at $M_j = 22\%$ for $d_p = 40$ μm. Symbols: experiment; solid line: AVBP-EL; dot-dashed line: CDP.*

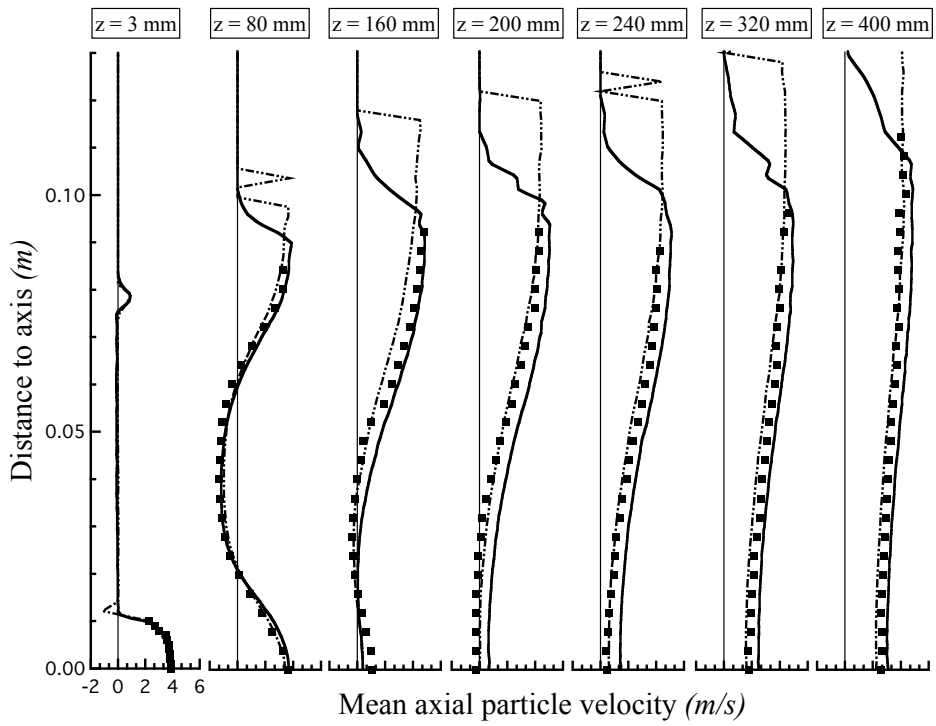


Figure 5.50 - *Radial profiles of mean axial particle velocities at seven stations along z axis at $M_j = 22\%$ for $d_p = 60$ μm. Symbols: experiment; solid line: AVBP-EL; dot-dashed line: CDP.*

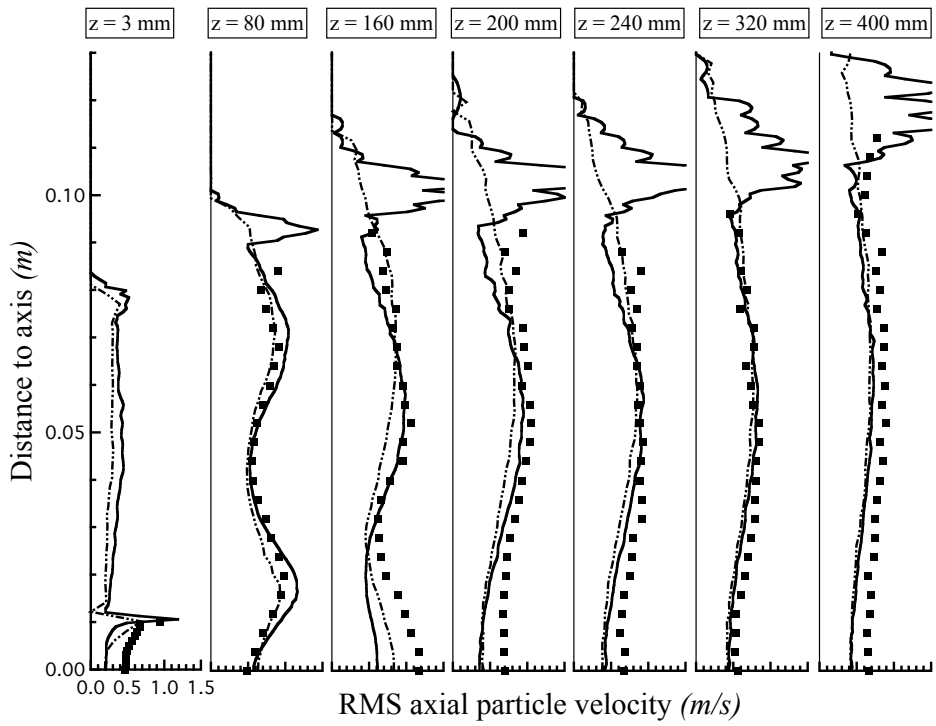


Figure 5.51 - *Radial profiles of RMS axial particle velocities at seven stations along z axis at $M_j = 22\%$ for $d_p = 60$ μm. Symbols: experiment; solid line: AVBP-EL; dot-dashed line: CDP.*

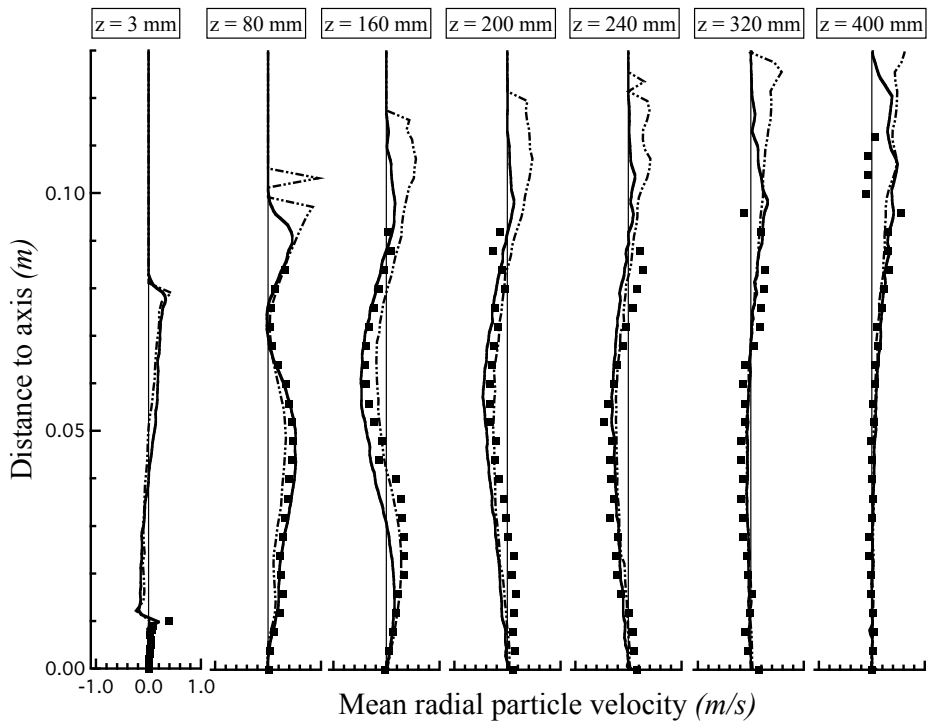


Figure 5.52 - *Radial profiles of mean radial particle velocities at seven stations along z axis at $M_j = 22\%$ for $d_p = 60$ μm. Symbols: experiment; solid line: AVBP-EL; dot-dashed line: CDP.*

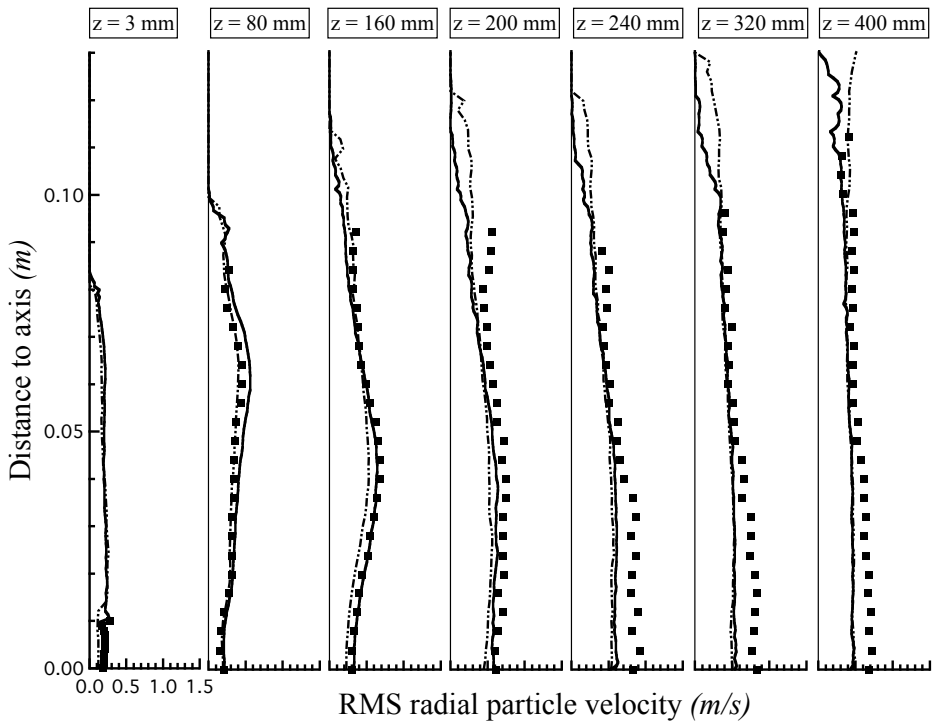


Figure 5.53 - *Radial profiles of RMS radial particle velocities at seven stations along z axis at $M_j = 22\%$ for $d_p = 60$ μm. Symbols: experiment; solid line: AVBP-EL; dot-dashed line: CDP.*

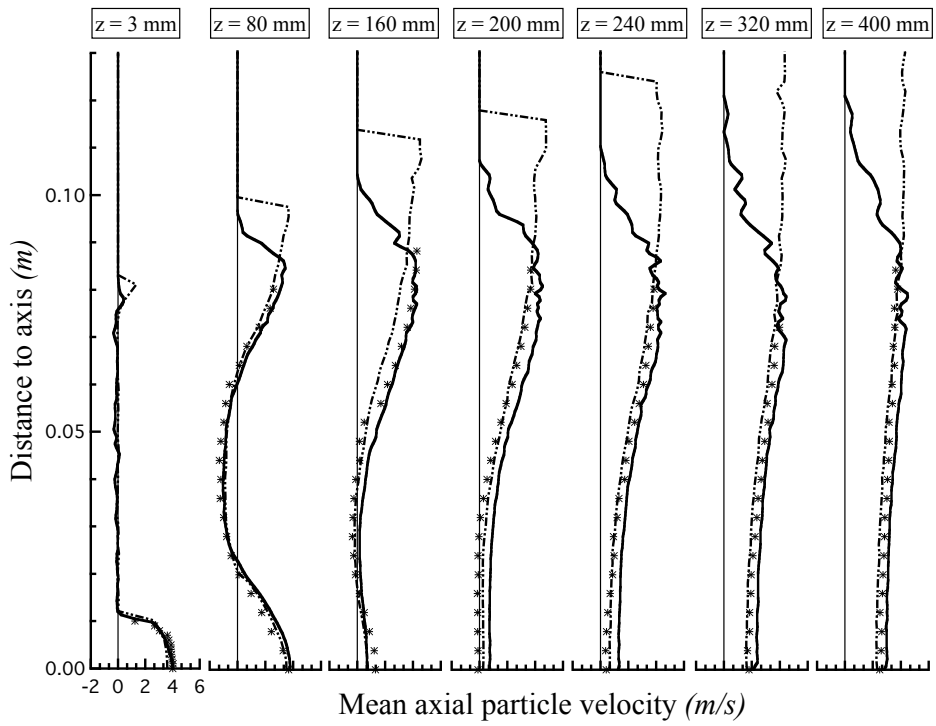


Figure 5.54 - *Radial profiles of mean axial particle velocities at seven stations along z axis at $M_j = 22\%$ for $d_p = 80$ μm. Symbols: experiment; solid line: AVBP-EL; dot-dashed line: CDP.*

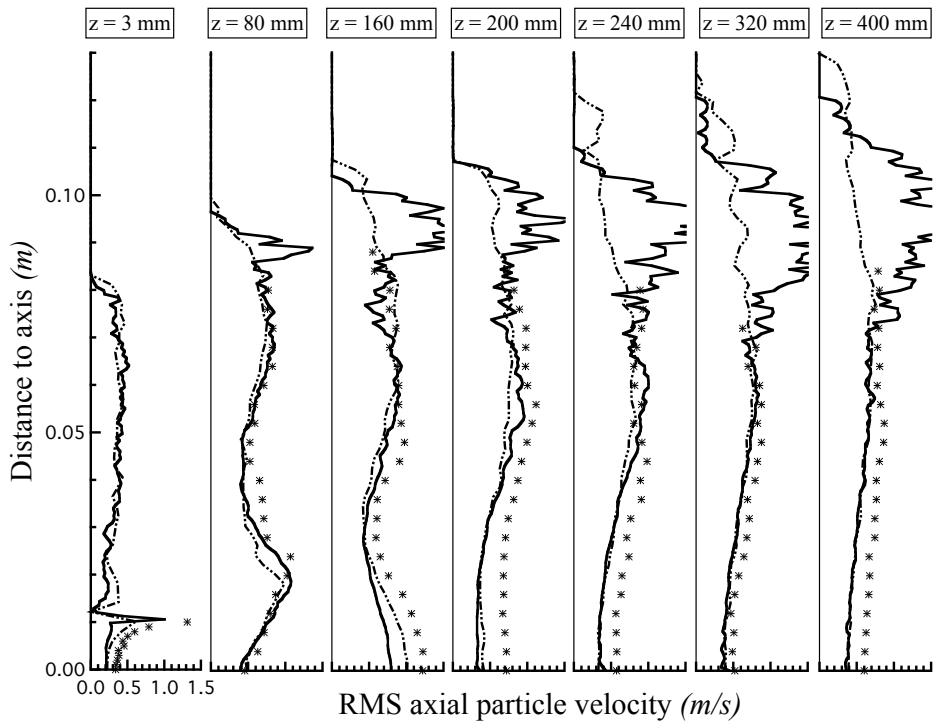


Figure 5.55 - *Radial profiles of RMS axial particle velocities at seven stations along z axis at $M_j = 22\%$ for $d_p = 80$ μm. Symbols: experiment; solid line: AVBP-EL; dot-dashed line: CDP.*

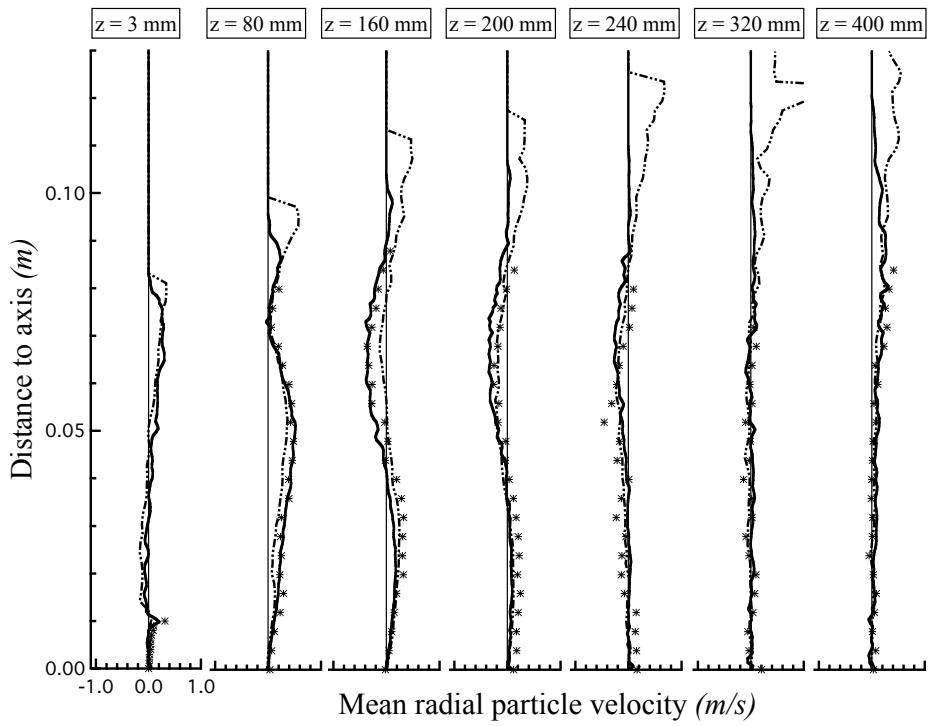


Figure 5.56 - *Radial profiles of mean radial particle velocities at seven stations along z axis at $M_j = 22\%$ for $d_p = 80$ μm. Symbols: experiment; solid line: AVBP-EL; dot-dashed line: CDP.*

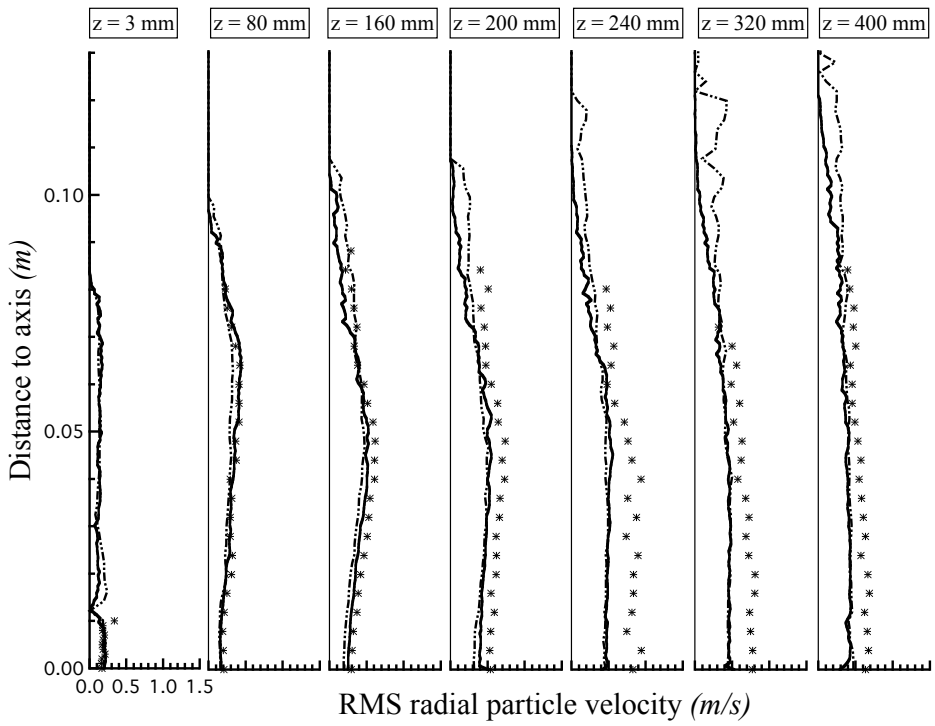


Figure 5.57 - *Radial profiles of RMS radial particle velocities at seven stations along z axis at $M_j = 22\%$ for $d_p = 80$ μm. Symbols: experiment; solid line: AVBP-EL; dot-dashed line: CDP.*

5.5.4 Influence of the number of samples

The following paragraphs discuss the effect of the number of samples on the radial velocity profiles presented in previous subsections. Table 5.4 summarises the physical time of the three simulations compared hereafter. Only the numerical results of the AVBP-EL solver are compared to the experimental data. The number of iterations and the CPU time (in hours) spent for a 32-processor simulation are also included in this table.

Physical time (s)	0.26	1	3.89
N° of time steps	63,000	234,000	922,000
CPU time with 32 processors (h)	≈ 74	≈ 274	≈ 1078

Table 5.4 - *Summary of parameters of AVBP-EL with the hexahedron-based grid on an IBM JS21.*

Figs. 5.58-5.65:

These figures show the mean and RMS axial velocity profiles[2] of classes: $d_p = 20, 40, 60$ and 80 μm. The differences between the three times, and therefore, the influence of an increasing number of samples are evident in all the figures. In most cases, an averaging time of one second appears to be sufficient. The first two cross-sections in the lightest particles (Figs. 5.58 and 5.59) are quite similar since the number of particles used to create the average profiles is high enough and the improvement of results is only visible in the outer region ($z > 80$ mm). On the contrary, mean and RMS profiles in the rest of stations ($z \geq 160$ mm) show an important improvement while increasing the average time, especially in the last stations. Results of the classes: 40 and 60 μm (Figs. 5.60-5.63) are quite similar. Both of them contain an important number of samples after one second of physical time. Again, the first cross-sections are always the less scattered. The number of samples in the heaviest particles (Figs. 5.64 and 5.65) leads to important differences between the three cases. These differences are even stronger than for the lightest particles since their number distribution is one of the lowest.

Figs. 5.66-5.69:

These figures[3] display the number of samples used to calculate average statistics. As expected, profiles at four seconds are close to four times the one second profiles. Results presented in these graphs confirm the tendencies observed in the previous ones: (i) the classes with the highest number of samples are: 40 and 60 μm; (ii) the number of particles captured in the first cross-sections is greater and that is why, mean profiles of Figs. 5.58-5.65 were less scattered. In addition, particles are mainly located near the centerline with a radial dispersion as long as the axial distance to the origin increases. Profiles of the 80 μm particles display a lower radial dispersion than the other classes. As mentioned in previous subsections, their large inertia causes these particles to penetrate more in the axial direction.

[2] Results of the mean and RMS radial velocity profiles are not presented here due to their similarity.
[3] Note that the x-axis do not have the same scale.

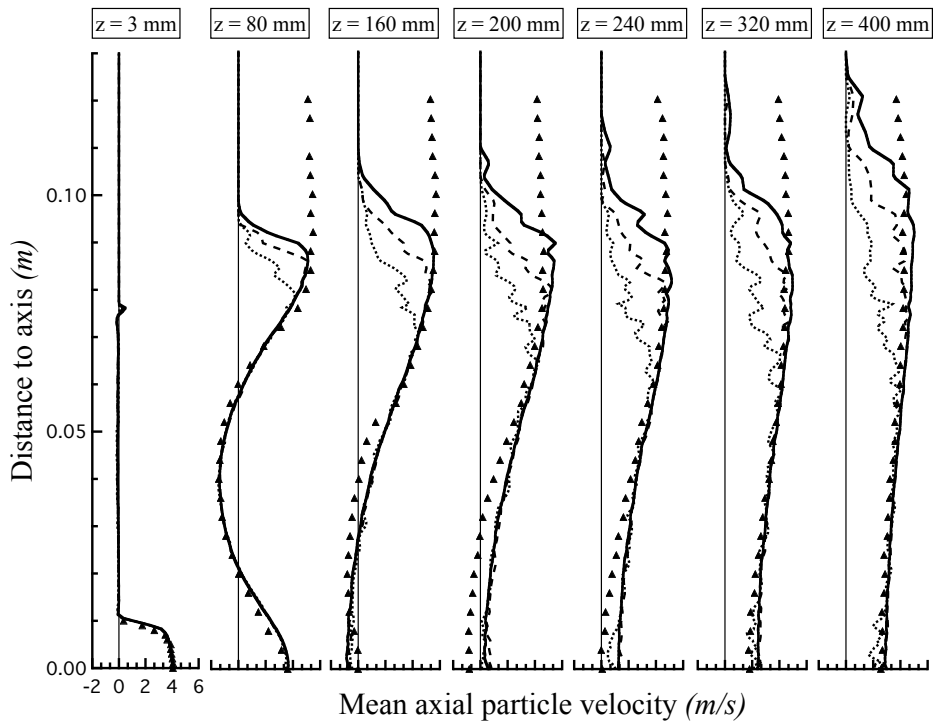


Figure 5.58 - *Radial profiles of mean axial particle velocities at seven stations along z axis at $M_j = 22\%$ for $d_p = 20$ μm. Symbols: experiment; lines: AVBP-EL at $t \approx 0.26$ s (dotted), $t \approx 1$ s (dashed), $t \approx 4$ s (solid).*

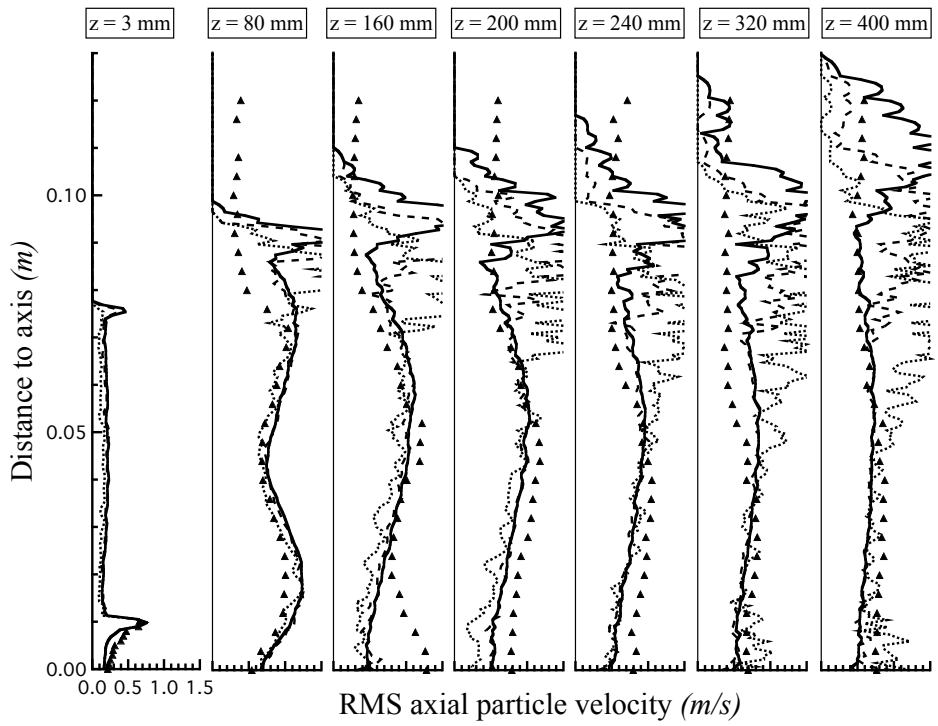


Figure 5.59 - *Radial profiles of RMS axial particle velocities at seven stations along z axis at $M_j = 22\%$ for $d_p = 20$ μm. Symbols: experiment; lines: AVBP-EL at $t \approx 0.26$ s (dotted), $t \approx 1$ s (dashed), $t \approx 4$ s (solid).*

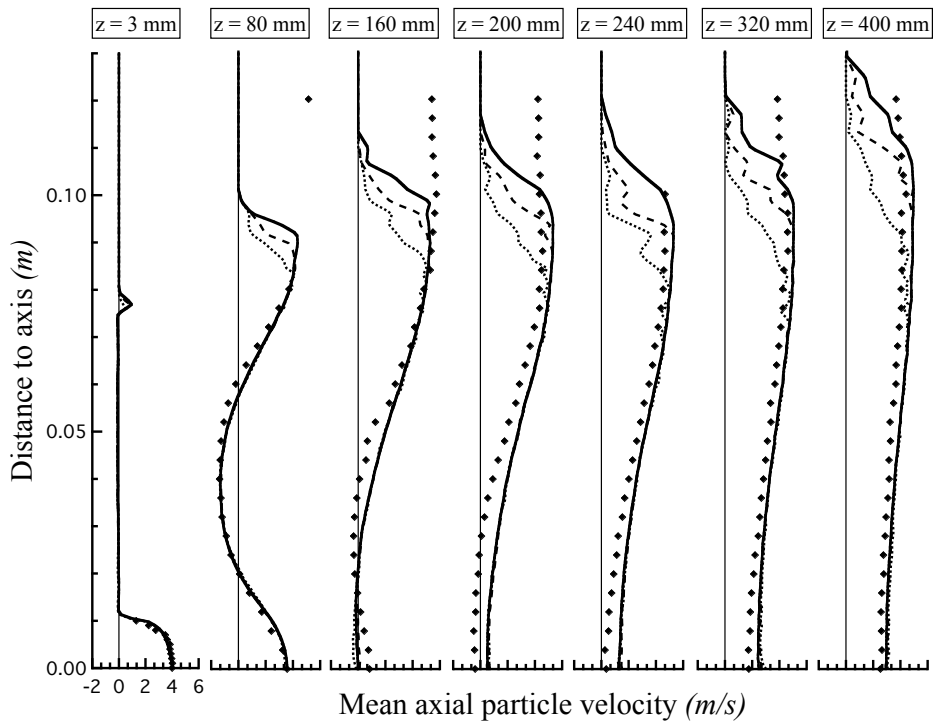


Figure 5.60 - *Radial profiles of mean axial particle velocities at seven stations along z axis at $M_j = 22\%$ for $d_p = 40$ μm. Symbols: experiment; lines: AVBP-EL at $t \approx 0.26$ s (dotted), $t \approx 1$ s (dashed), $t \approx 4$ s (solid).*

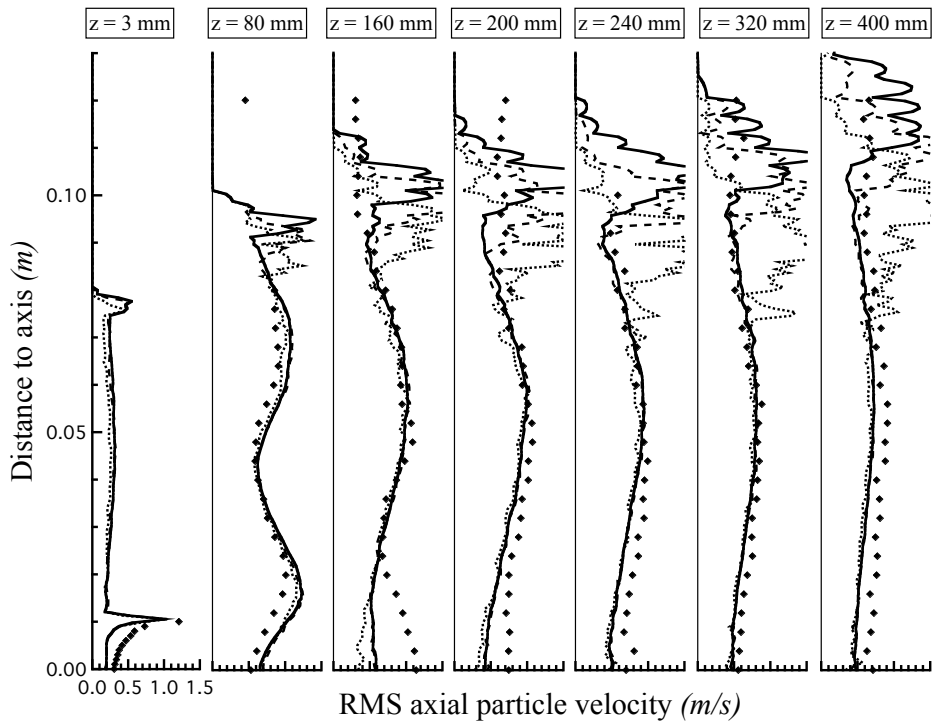


Figure 5.61 - *Radial profiles of RMS axial particle velocities at seven stations along z axis at $M_j = 22\%$ for $d_p = 40$ μm. Symbols: experiment; lines: AVBP-EL at $t \approx 0.26$ s (dotted), $t \approx 1$ s (dashed), $t \approx 4$ s (solid).*

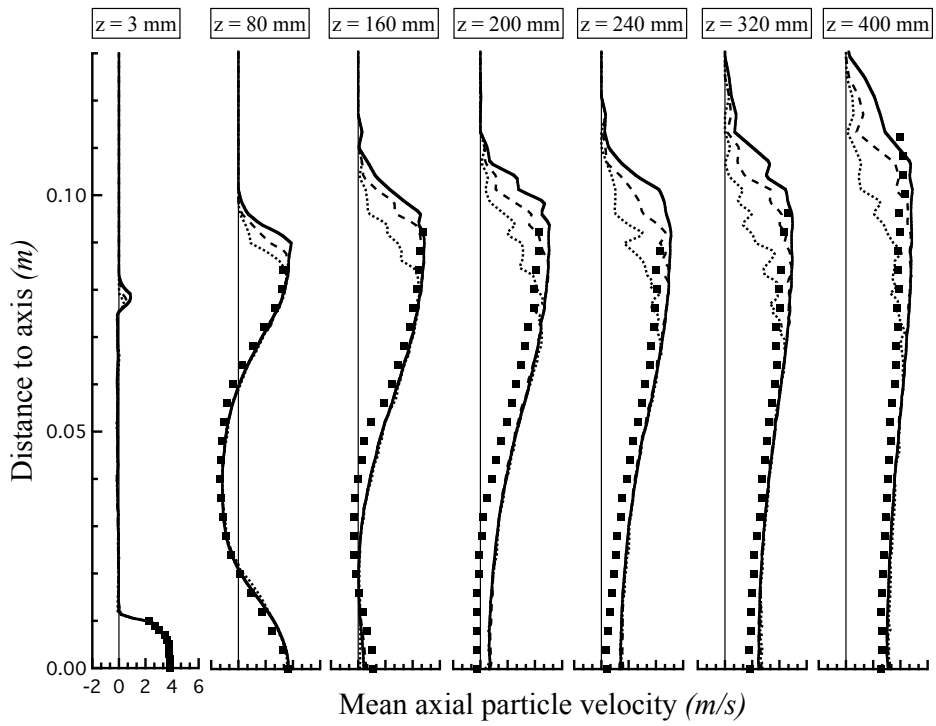


Figure 5.62 - *Radial profiles of mean axial particle velocities at seven stations along z axis at $M_j = 22\%$ for $d_p = 60$ μm. Symbols: experiment; lines: AVBP-EL at $t \approx 0.26$ s (dotted), $t \approx 1$ s (dashed), $t \approx 4$ s (solid).*

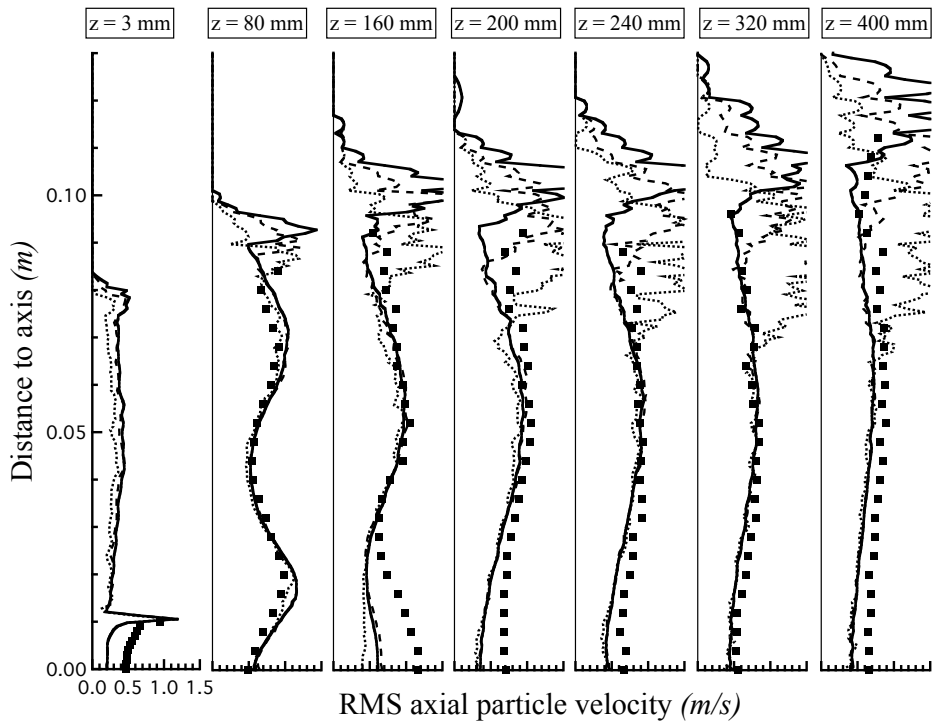


Figure 5.63 - *Radial profiles of RMS axial particle velocities at seven stations along z axis at $M_j = 22\%$ for $d_p = 60$ μm. Symbols: experiment; lines: AVBP-EL at $t \approx 0.26$ s (dotted), $t \approx 1$ s (dashed), $t \approx 4$ s (solid).*

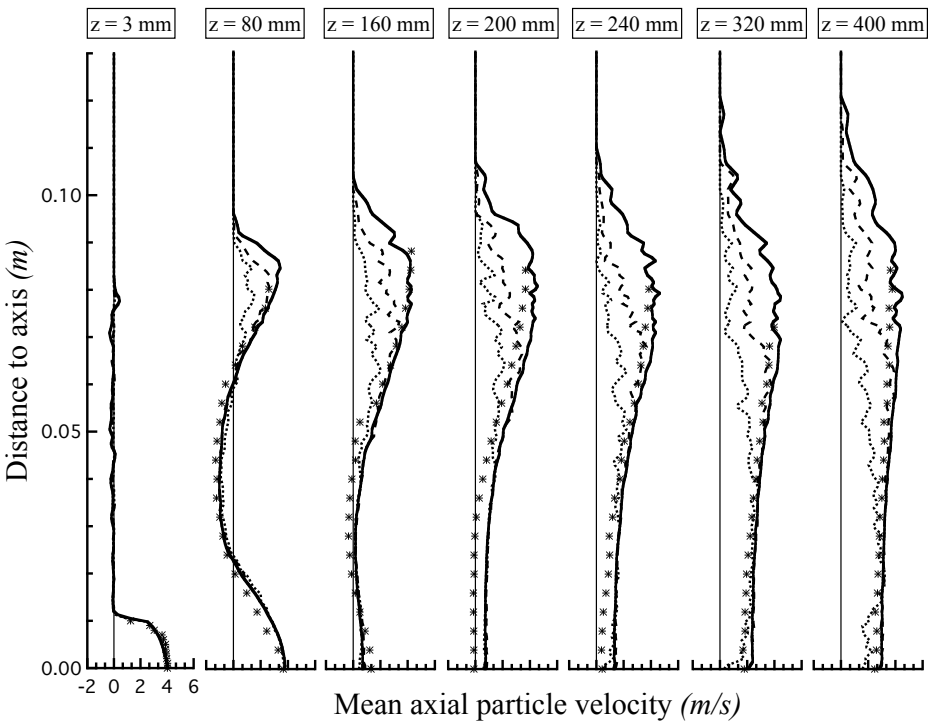


Figure 5.64 - *Radial profiles of mean axial particle velocities at seven stations along z axis at $M_j = 22\%$ for $d_p = 80$ μm. Symbols: experiment; lines: AVBP-EL at $t \approx 0.26$ s (dotted), $t \approx 1$ s (dashed), $t \approx 4$ s (solid).*

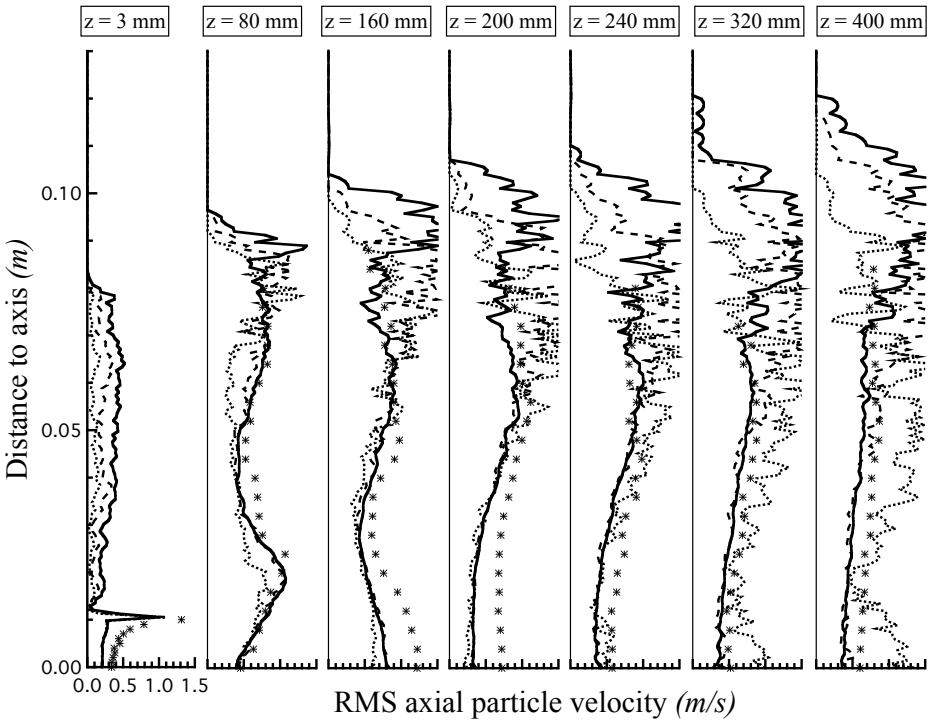


Figure 5.65 - *Radial profiles of RMS axial particle velocities at seven stations along z axis at $M_j = 22\%$ for $d_p = 80$ μm. Symbols: experiment; lines: AVBP-EL at $t \approx 0.26$ s (dotted), $t \approx 1$ s (dashed), $t \approx 4$ s (solid).*

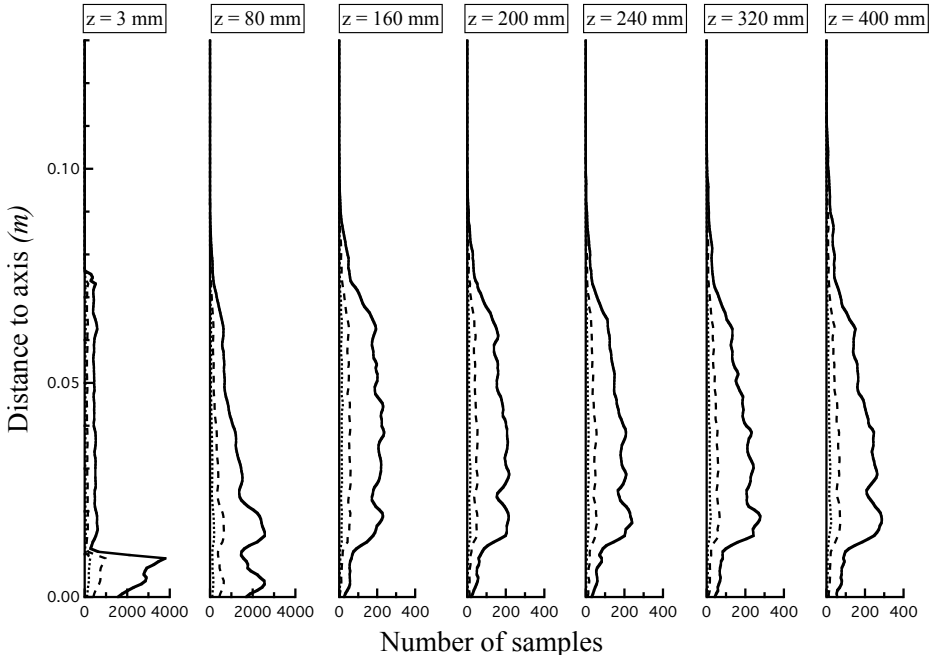


Figure 5.66 - *Number of samples at seven stations along z axis at $M_j = 22\%$ for $d_p = 20$ μm. Lines: AVBP-EL at $t \approx 0.26$ s (dotted), $t \approx 1$ s (dashed), $t \approx 4$ s (solid).*

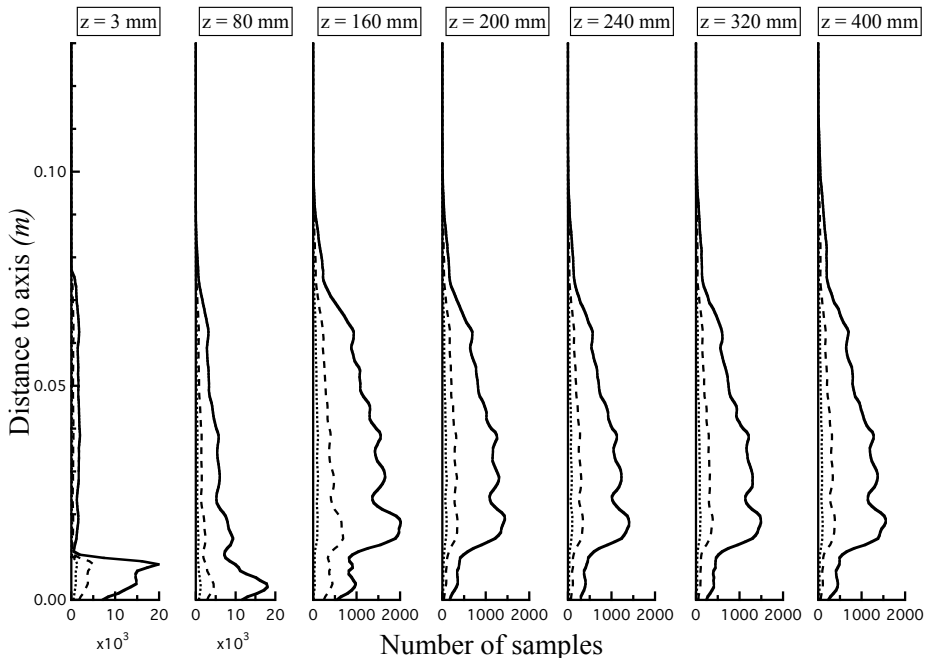


Figure 5.67 - *Number of samples at seven stations along z axis at $M_j = 22\%$ for $d_p = 40$ μm. Lines: AVBP-EL at $t \approx 0.26$ s (dotted), $t \approx 1$ s (dashed), $t \approx 4$ s (solid).*

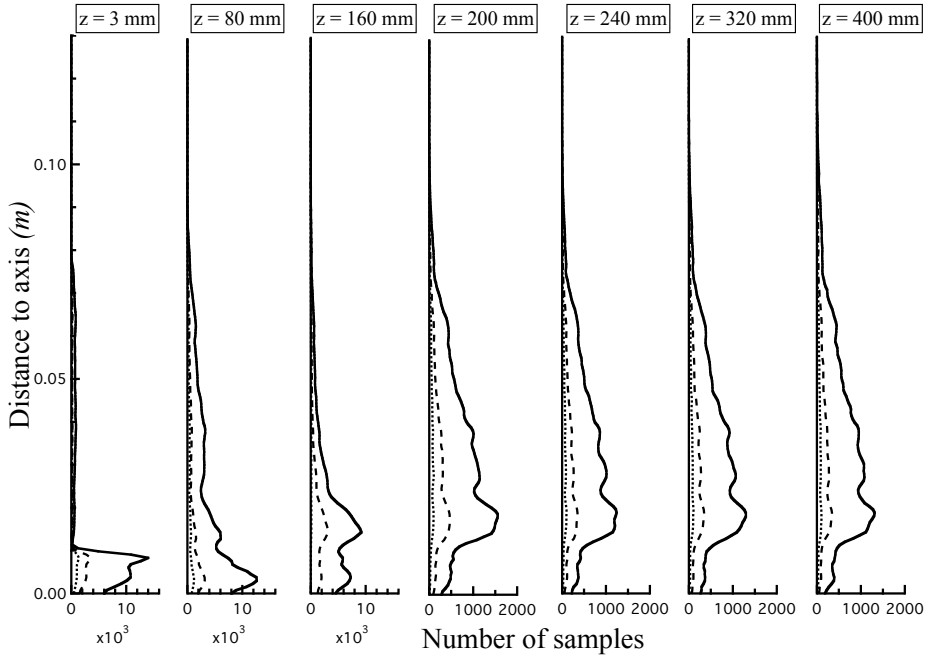


Figure 5.68 - *Number of samples at seven stations along z axis at $M_j = 22\%$ for $d_p = 60$ μm. Lines: AVBP-EL at $t \approx 0.26$ s (dotted), $t \approx 1$ s (dashed), $t \approx 4$ s (solid).*

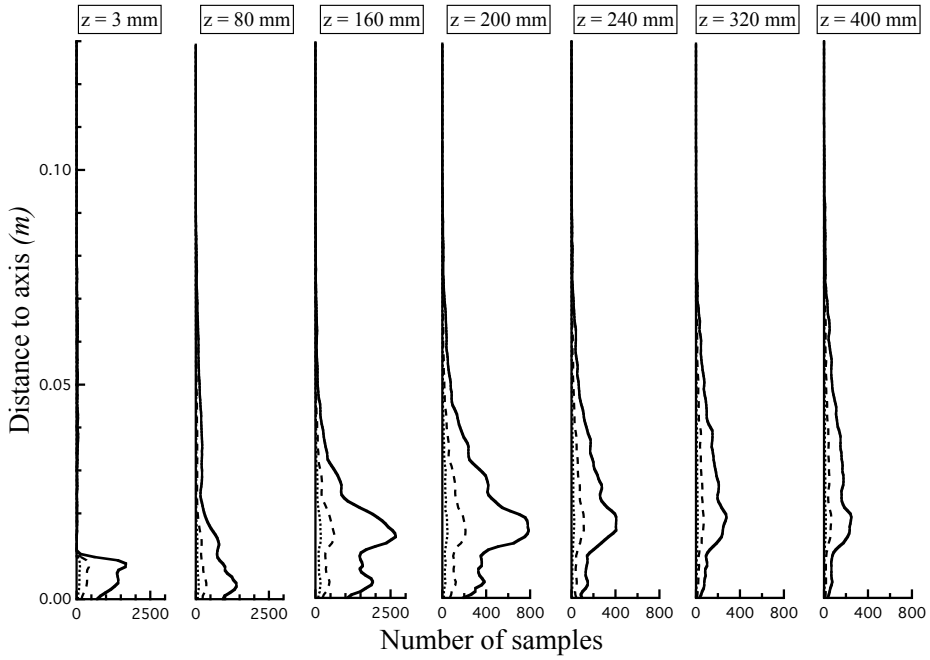


Figure 5.69 - *Number of samples at seven stations along z axis at $M_j = 22\%$ for $d_p = 80$ μm. Lines: AVBP-EL at $t \approx 0.26$ s (dotted), $t \approx 1$ s (dashed), $t \approx 4$ s (solid).*

5.5.5 Particle trajectories

Figure 5.70 (a) shows the probability density function (PDF) of the slip velocities of all particle classes. The peak of the curve is near a value of 0.5 m/s which confirms that a high number of particles follow the fluid motion. The PDFs of the slip velocities of four different particle classes are presented in Fig. 5.70 (b). Smallest particles follow the fluid very close with low values of the slip velocity (≈ 0.2 m/s). As long as the particle diameter increases, the peak of the PDF decreases and the slip velocity associated to it increases, making them more independent of the flow.

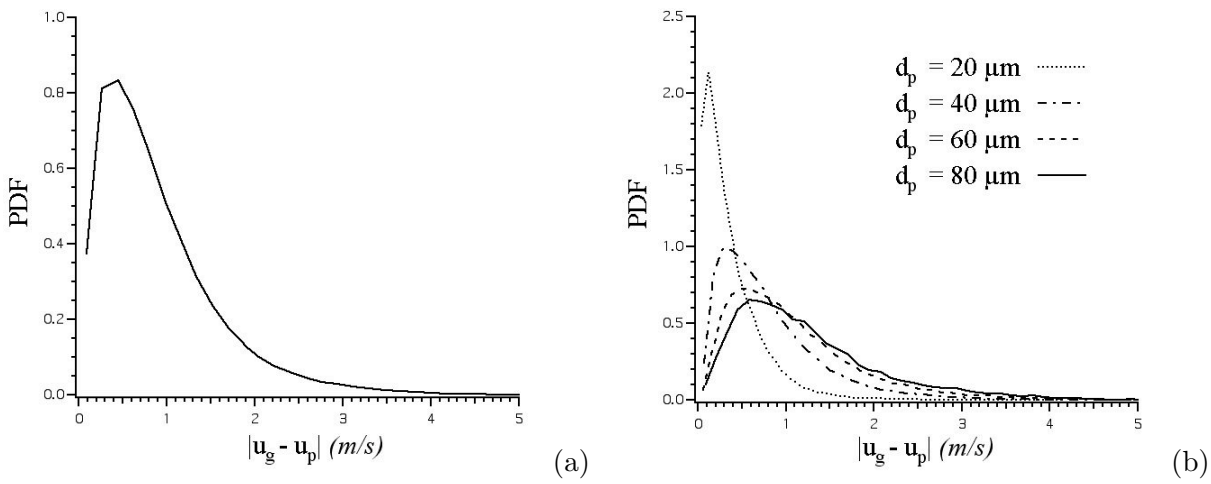


Figure 5.70 - *Probability density function (PDF) of (a) all classes and (b) only four particle classes.*

Figs. 5.71 and 5.72 display 25 tracks of particles with diameters: $d_p = 20, 40, 60$ and 80 μm at a moderate mass loading and for the same physical time: $t = 0.44$ s. Left and right panels present the projections of particle pathlines in the $y - z$ and $y - x$ axes, respectively. Initially, all particles follow the axial direction without turning away from the axis. Due to their different nature, lighter particles respond to the flow faster and are decelerated to zero axial velocity sooner (Fig. 5.71). Furthermore, their trajectories are deviated and more influenced by turbulence as it can be observed from their twisted pathlines inside the recirculation bubble.

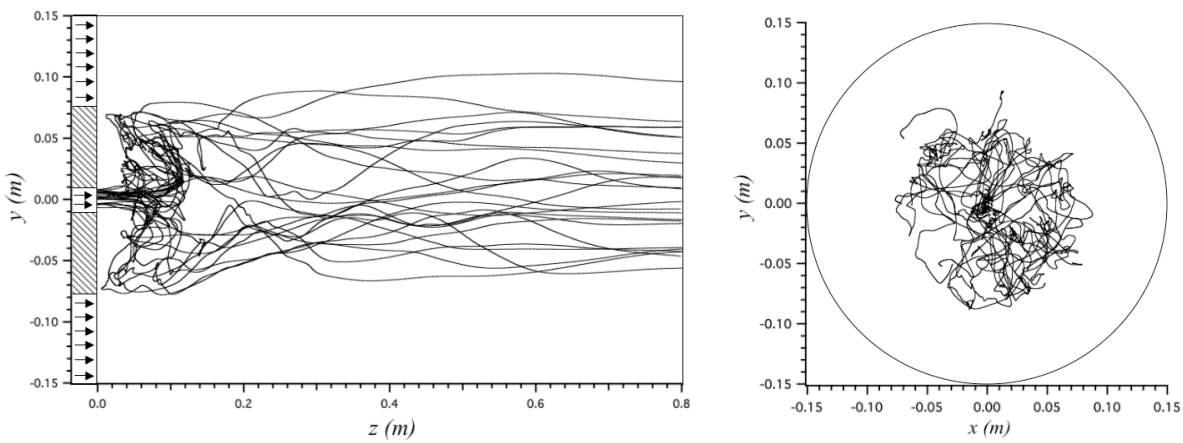


Figure 5.71 - *Some particle trajectories calculated at $M_j = 22\%$ for $d_p = 20$ μm at $t = 0.44$ s. Left: y-z view. Right: y-x view.*

On the contrary, heavier particles (Fig. 5.72) penetrate more into the recirculation bubble due to their large inertia before coming to rest. The highest penetration is clearly presented for 80 μm particles (bottom panel). This class also presents a particularity because these particles do not reach the wall between the inner and the annular jets since they are captured before by large eddies of the coflow boundary and transported downstream.

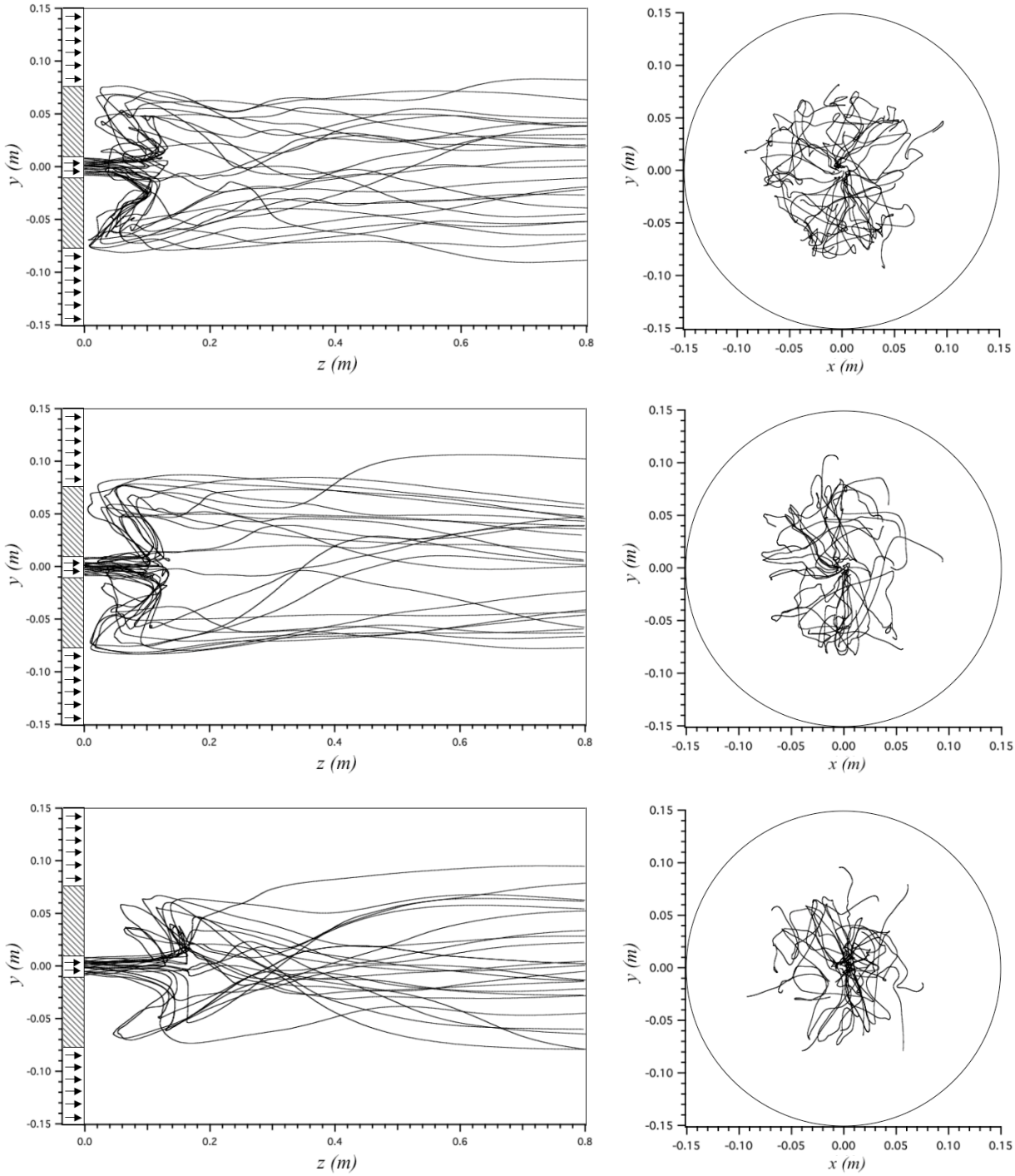


Figure 5.72 - *Some particle trajectories calculated at $M_j = 22\%$ at $t = 0.44$ s. Top panel: $d_p = 40$ μm; middle panel: $d_p = 60$ μm; bottom panel: $d_p = 80$ μm. Left: y-z view. Right: y-x view.*

5.6 Performance

One interesting issue in the development of Lagrangian methods is the analysis of scalability and computational performance. The implementation on massively parallel machines of the dispersed phase with Eulerian approaches is not technically problematic because the flow and the droplets are solved on the same grid using the same spatial discretization scheme. On the other hand, Lagrangian approaches are less well-suited to parallel computers since the two phases must be coupled, which increases the complexity of the implementation. Thus, before implementing a Lagrangian module into a new solver one of the strategies presented in Subsection 2.6.1 should be adopted for the dispersed phase treatment. The most used are:

- Task parallelism in which some processors compute the gaseous phase and others compute the particles or droplets.

- Data parallelism in which particles are computed together with the gas flow on geometrical subdomains mapped on parallel processors. Individual particles or parcels are tracked as they cross the computational domain and must be exchanged between processors when leaving a subdomain to enter an adjacent one.

Particle tracking within an unstructured solver (like AVBP) reveals an additional constraint since particle coordinates cannot be easily used to locate them inside a cell, and time (and memory) consuming searching algorithm must be used. However, for LES it is easy to show that only the strategy based on data parallelism is efficient on large grids because task parallelism would require the communication of very large three-dimensional data sets at each iteration between all processors, increasing the cost of communications and reducing considerably the speedup. It is well known that codes treating particles with a data parallelism technique and based on domain partitioning are difficult to optimize on massively parallel architectures when droplets are clustered in one part of the domain (typically, near the fuel injectors) due to load imbalance. Moreover, the distribution of droplets may change during the computation: for a gas turbine reignition sequence, for example, the chamber is filled with droplets when the ignition begins thus ensuring an almost uniform droplet distribution; these droplets then evaporate rapidly during the computation, leaving droplets only in the near injector regions. This may lead to a poor speedup on a parallel machine if the domain is decomposed in the same way for the entire computation since some processors should compute a high number of particles while others are waiting for this task to finish. As a result, load balancing strategies are required to redecompose the domain by taking into account particles information to preserve a high parallel efficiency [76].

In this section, the notions of scalability and CPU time introduced in Section 2.6 will be applied to analyse performance of the implementation of the Lagrangian module. This scalability study has been performed in a CRAY XD1[4] supercomputer for a number of processors up to 64. Subsection 5.6.1 summarises results of the simulations performed with a one-constraint partitioning algorithm (RIB), i.e., without balancing the number of particles across the subdomains; and subsection 5.6.2 presents the effect of a two-constraint partitioning algorithm which takes into account the particle information while partitioning the grid.

[4] This machine has 58 nodes with 2 processors/node and 2 GB/processor.

5.6.1 Scalability with a one-constraint partitioning algorithm

Results of the moderate mass loading test case has been calculated in two different grids (tetrahedron and hexahedron-based grids) and results are presented hereafter. The total number of particles presented in the domain is of the order of 560,000 and 430,000, respectively. Variations smaller than 0.5% in the number of particles were observed between the beginning and the end of the simulation, which implies that it is statistically stationary. The lower number of particles used in the tetrahedron grid is due to the difference in the length between the two grids (see Fig. 5.5). Even if the length of the chamber is half the dimension of the tetrahedron-based grid, the number of particles is three-fourth since they are mainly concentrated inside the recirculation zone. Figure 5.73 shows a front view of both computational meshes after the partitioning into 32 subdomains by the RIB partitioning algorithm (see Subsection 2.4.2). A more complete study of a simulation with the tetrahedron-based grid can be consulted in Appendix C.

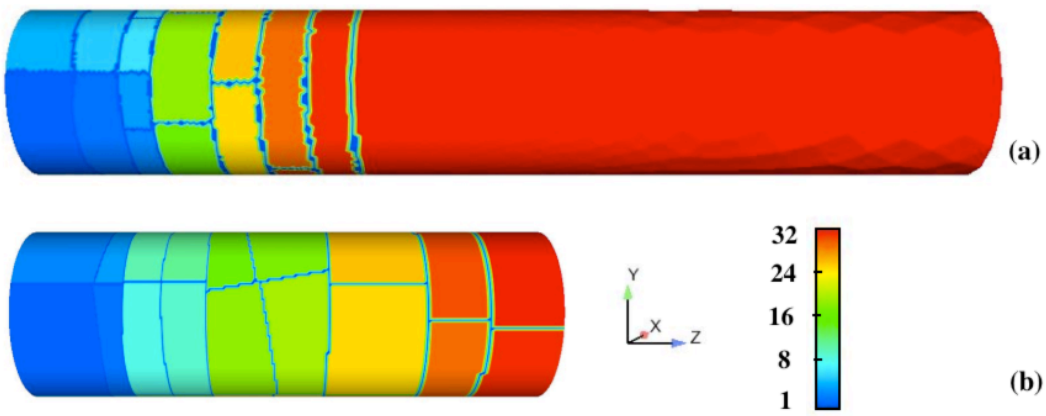


Figure 5.73 - *Front view of a computational mesh divided into 32-subdomains by using the RIB partitioning algorithm for (a) a tetrahedron and (b) a hexahedron-based grid. Configuration of Borée* et al. *[22].*

Figure 5.74 shows the speedup of the single-phase and the monodisperse test cases with the tetrahedron (Fig. 5.74 (a)) and the hexahedron-based grid (Fig. 5.74 (b)). Scaling of the hexahedron grid is reported relative to the 8 processor case ($n = 8$), which was the smallest number of CRAY XD1 processors that could run this problem due to high memory requirements. In both cases, the good scalability of the single-phase is unquestionable. The drop of performance observed in Fig. 5.74 (a) for the two-phase flow simulation is not related to large communications costs between processors but merely to the parallel load imbalance generated by the partitioning algorithm [69], as will be demonstrated in the next subsection. The same simulation with a different grid can lead to a completely different speedup graph, as observed from Fig. 5.74 (b).

The differences between the two speedup graphs can also be explained by plotting the number of nodes (or cells) and particles presented in each processor. As AVBP is based on a *cell-vertex* formulation, comparing the number of nodes to the number of particles is more representative of the computational loading since almost all arrays are dimensioned as a function of the number of nodes per processor. Figure 5.75 reports the number of nodes and particles per processor for a 32-processors simulation with RIB, for the tetrahedron and the hexahedron-based grids, respectively.

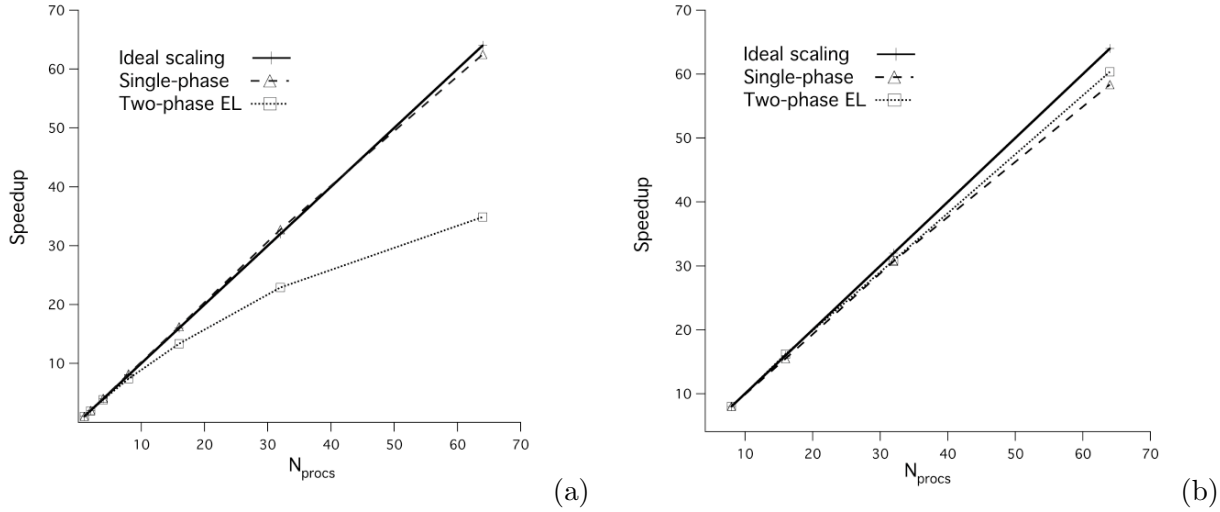


Figure 5.74 - *Speedup of the single-phase and the monodisperse test case with (a) the tetrahedron and (b) the hexahedron-based grid on a CRAY XD1 supercomputer.*

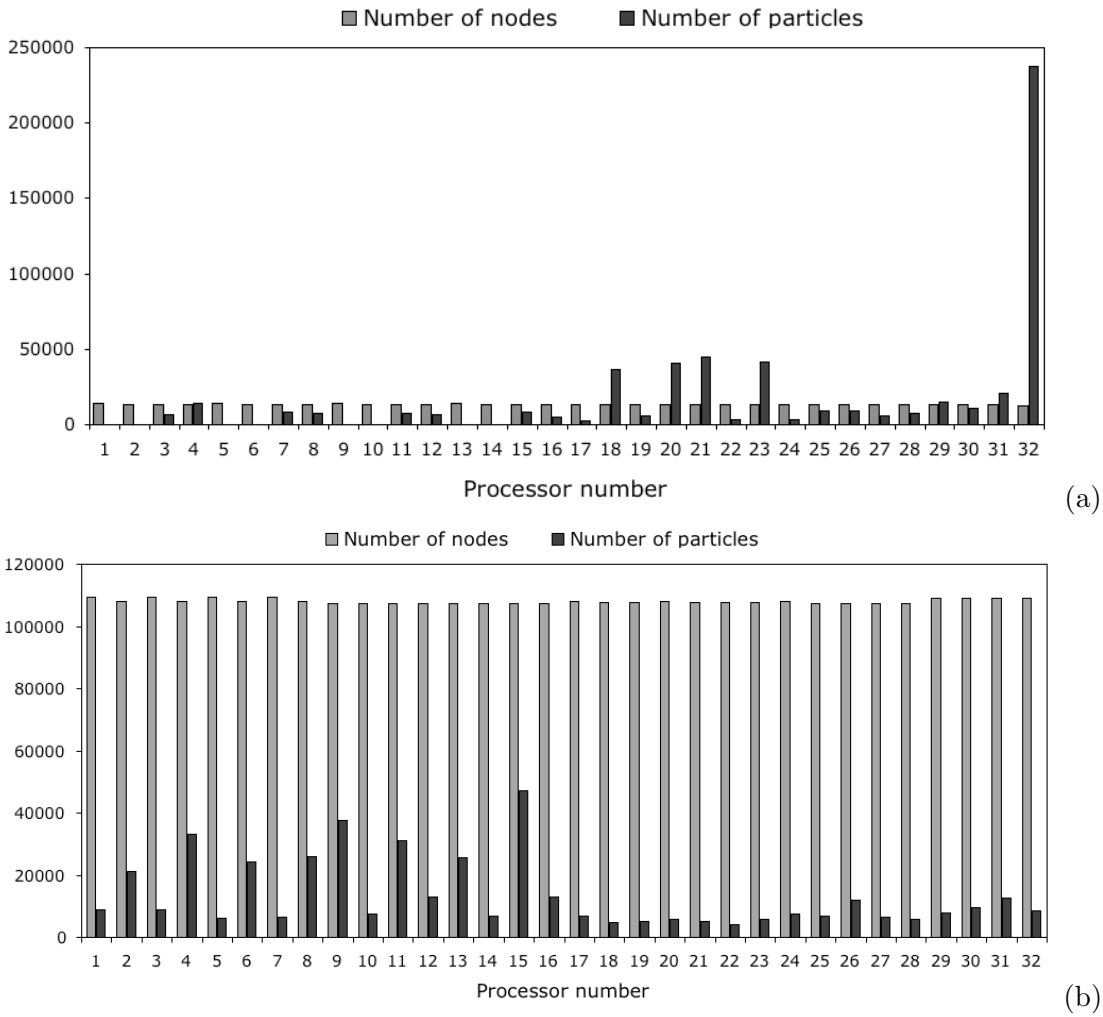


Figure 5.75 - *Number of nodes and particles per processor for a 32-processors simulation by using the RIB partitioning algorithm for the (a) tetrahedron and the (b) hexahedron-based grids.*

As expected from Fig. 5.74, Fig. 5.75 shows an excellent load balancing for the gaseous phase: all processors contain about the same number of nodes. On the contrary, it shows a strong particle load imbalance (Fig. 5.75 (a)) where one single processor contains almost half the total number of particles of the simulation. This fact is related to the grid downstream coarsening which increases significantly the memory requirements and the floating-point operations for this processor. This case confirms the need of load balancing strategies for two-phase flow simulations with a Lagrangian approach. Even if the dispersed phase presents a small load imbalance in Fig. 5.75 (b), it is hidden by the higher computational loading needed to calculate the gaseous phase. This is one of the reasons of the good speedup observed in Fig. 5.74 (b).

Tables 5.5 and 5.6 summarised the CPU time ratios with both grids. Additional time to perform the two-phase flow simulation can vary from 5% (for a sequential simulation) up to 87% (for a 64-processors simulation) with the Lagrangian formulation in the tetrahedron-based grid but it is not higher than 5% with the hexahedron-based grid which confirms the tendencies observed with the speedup graphs. The same simulations with the Eulerian approach has a constant added cost of the order of 80% since this approach is independent of the mass loading. Therefore, at this moderate mass loading the Lagrangian approach proved to be faster than the Eulerian formulation.

N_{procs}	1	2	4	8	16	32	64
Single-phase	1	0.50	0.25	0.12	0.06	0.030	0.016
Two-phase AVBP-EL	1.05	0.54	0.27	0.14	0.08	0.046	0.030

Table 5.5 - *Summary of the CPU time ratios of AVBP-EL with a tetrahedron-based grid on a CRAY XD1 supercomputer.*

N_{procs}	8	16	32	64
Single-phase	1	0.51	0.26	0.137
Two-phase AVBP-EL	1.06	0.524	0.275	0.14

Table 5.6 - *Summary of the CPU time ratios of AVBP-EL with a hexahedron-based grid on a CRAY XD1 supercomputer.*

5.6.2 Scalability with a two-constraint partitioning algorithm

As demonstrated in the previous subsection, partitioning algorithms able to balance only a single quantity can result in good or bad speedup graphs for two-phase flow simulations depending on the application. This problem is not new: in fact, many important types of multi-phase and multi-physics computations require that multiple quantities be load-balanced simultaneously. The critical point is that each processor has the same amount of work from each phase. Two examples are particle-in-cells [225] and contact impact [62] simulations. The aim of this subsection is the implementation of a multi-constraint partitioning algorithm into the Lagrangian module of AVBP to show the effect of particles load balancing. The multi-constraint partitioning algorithm chosen is one of those available inside the METIS package (see Subsection 2.4.4). In the following, only two constraints are used: one to balance the number of grid elements and the other the number of particles, respectively.

Figure 5.76 presents in a simple way the problem encountered in a PIC application and the information recovered by the multi-constraint algorithm to perform the partitioning. A grid containing some particles (represented by small red circles) is displayed on the left panel. The right panel shows the dual graph of the mesh with two weights on each vertex. The first weight represents the work associated with the mesh-based computation (gaseous phase) for the corresponding element[5]. The second weight represents the work associated with the particle-based computation (dispersed phase). This value is estimated by the number of particles that fall within each element. With this information, the multi-constraint algorithm is able to partition the grid by balancing elements and particles between subdomains (the interface is represented by a bold line): ten elements and eight particles on each subdomain.

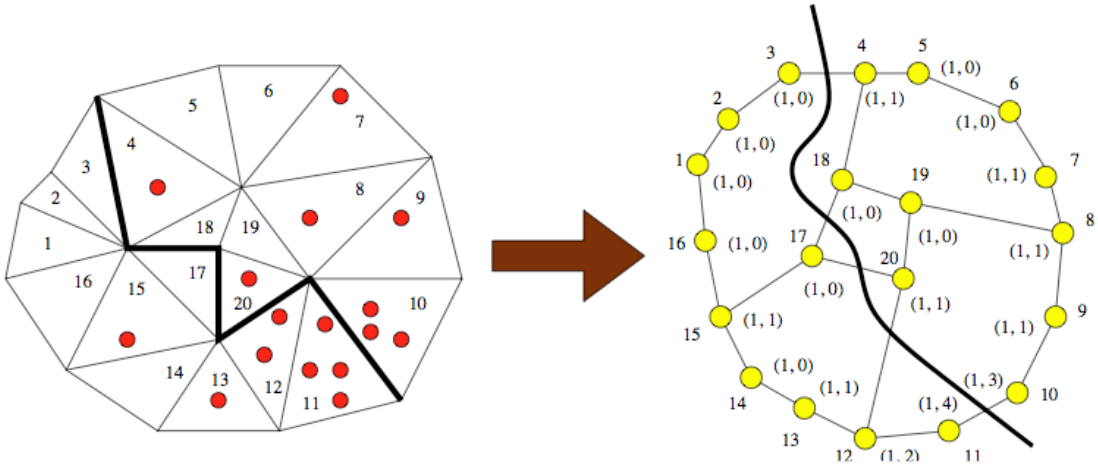


Figure 5.76 - *A dual graph with vertex weight vectors of size two (right) is constructed from the mesh. A multi-constraint partitioning has been computed for this graph, and this partitioning has been projected back to the mesh. From [97].*

Applying the same principle to the imbalance test case observed in Fig. 5.74 (a) for the tetrahedron-based grid we obtain the following mesh partitioned into 32-subdomains (Fig. 5.77). At first sight, it can be observed that this new view of the computational mesh is quite different from the one displayed in Fig. 5.73 (a), illustrating one of the differences between multilevel and geometric partitioning algorithms. Another difference can be found in the number of duplicated nodes: that is a reduction of 2.7% (RIB: 424,163; METIS: 412,603) even when using the new weight imposed by particle treatment (however, the increase or reduction in the number of these new nodes depends on the test case).

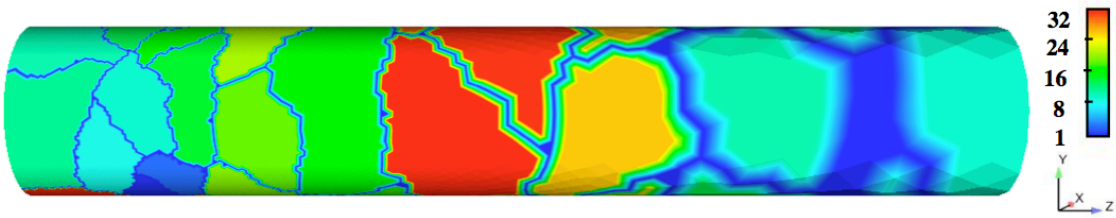


Figure 5.77 - *Front view of a computational mesh divided into 32-subdomains by using a multi-constraint partitioning algorithm (from METIS) to take into account particles. Configuration of Borée* et al. *[22].*

[5] All values are one because we assume that all the elements have the same amount of work associated with them.

Figure 5.78 presents the speedup of a single-phase simulation and two two-phase simulations performed with the Lagrangian version developed during this thesis[6]. The differences between the two two-phase simulations is on the algorithms used to partition the grid:

- A one-constraint recursive bisection algorithm (RIB) explained in Subsection 2.4.2.
- A multi-constraint multilevel recursive bisection algorithm, from the METIS package.

The improvement observed by using an algorithm to balance particles across subdomains is clearly evident. This is a remarking result since load imbalance is an inherent problem to Lagrangian simulations and it has always been considered as a major drawback. The success in the use of multi-constraint partitioning algorithm to improve Lagrangian computation performance on massively parallel machines had already been demonstrated by Ham *et al.* [76]. Results obtained in this thesis confirm the competitive position of Lagrangian formulation compared to its principal competitor, the Eulerian formulation, reducing the disadvantages associated with the Lagrangian formulation.

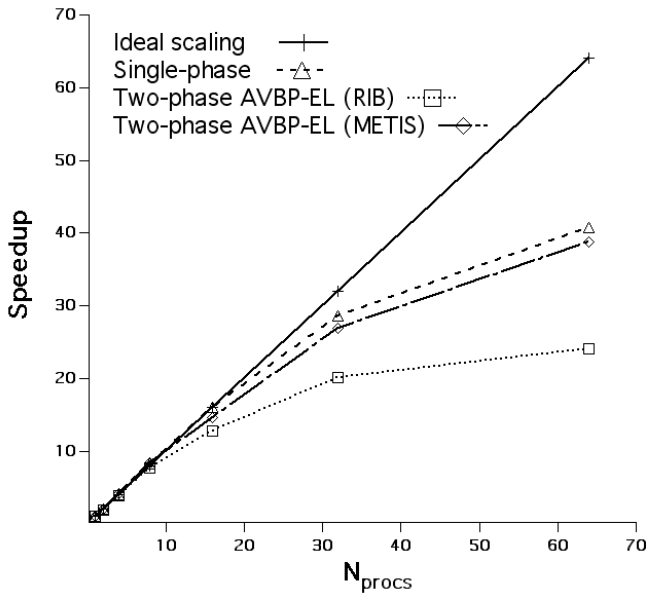


Figure 5.78 - *Speedup of the single-phase and the monodisperse test case with two two-phase simulations: one with the RIB algorithm and the other with a multi-constraint partitioning algorithms from METIS.*

Figure 5.79 reports the number of nodes and particles per processor for a 32-processors simulation with the multi-constraint partitioning algorithm for the tetrahedron-based grid. As expected from the speedup graph, this figure shows an excellent load-balancing for both, gaseous and dispersed phases. Comparing this figure with Fig. 5.75 (a) it can be observed that processor number 32 has reduced in more than 10 times the number of particles to treat. The main consequences of this reduction is that other processors will not wait for this one to finish, increasing the efficiency of the overall computation.

[6] The input files of the single-phase computation and the two-phase AVBP-EL (RIB) computation are the same as the ones used for the simulations presented in Fig. 5.74 (a) but the version of AVBP (for gaseous and dispersed-phase subroutines) is more recent in these last results. However, the differences observed between them are due to a hardware problem detected in this machine after some upgrades. The computer support group (CSG) at CERFACS is working with CRAY to detect the source of the problem.

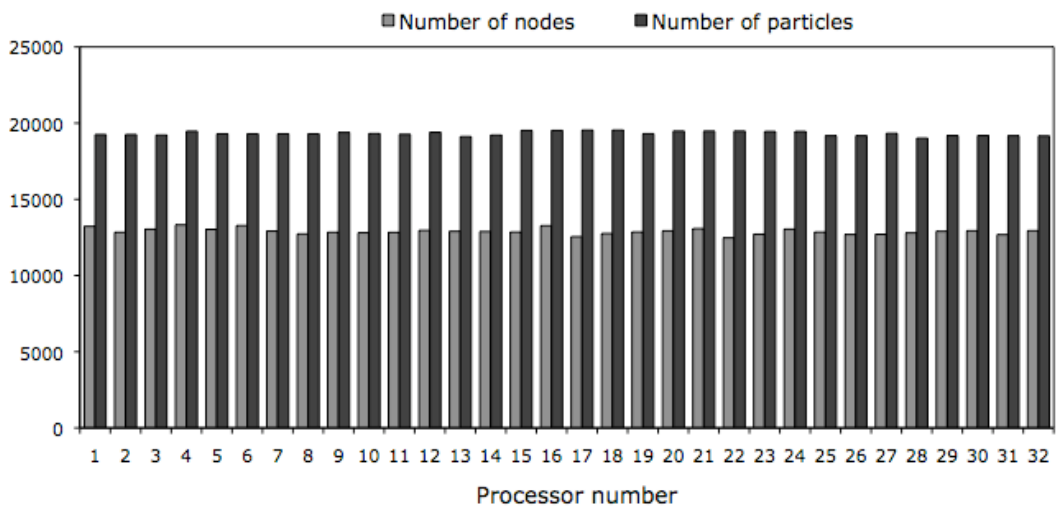


Figure 5.79 - *Number of nodes and particles per processor for a 32-processors simulation by using a two-constraint partitioning algorithm from METIS on the tetrahedron-based grid.*

Table 5.7 summarised the CPU time ratios of the simulations previously mentioned. Additional time to perform the one-constraint two-phase flow computation with 32 processors was 55%. On the contrary, with the use of the two-constraint algorithm, the two-phase computation needs only 16% more than the single-phase computation, which is a reasonable cost to study the moderate mass loading case with this configuration.

N_{procs}	1	2	4	8	16	32	64
Single-phase	1	0.504	0.257	0.122	0.063	0.035	0.025
Two-phase AVBP-EL (RIB)	1.09	0.557	0.278	0.143	0.086	0.054	0.045
Two-phase AVBP-EL (METIS)	1.09	0.533	0.266	0.131	0.075	0.041	0.028

Table 5.7 - *Summary of the CPU time ratios of AVBP-EL with a tetrahedron-based grid on a CRAY XD1 supercomputer.*

5.7 Conclusions

The main conclusions of this chapter are summarised here:

- In this configuration where particles are inertial, but still much dependent on the gas flow, prediction of particles motion strongly depend on the results for the gas phase. Results obtained with the Lagrangian version of AVBP (AVBP-EL) are in good agreement with the experiments, and with the results provided by the LES solver CDP. The accuracy in the single-phase case for radial and axial profiles of mean and fluctuating velocities are as good as the results from CDP except for the location of the recirculation zone which is slightly shifted upstream with the AVBP-EL solver.

- As a general remark, the axial evolution and the radial profiles of the mean and RMS particle velocities of the monodisperse and polydisperse cases are in good agreement with CDP results which validates all developments of the Lagrangian module implemented on the code. Concerning the gaseous results, the monodisperse and the polydisperse cases are very similar. The main important difference is in the levels of the mean gas velocity in the recirculation zone which are lower than expected. This implies a reduction in the size of the recirculation bubble in the polydisperse case. Since there are no differences in CPU time between the monodisperse and the polydisperse cases, trying to simulate this flow with a polydisperse particle distribution is closer to the experiments and the reality.

- The effect of the number of particles on the radial velocity profiles at three physical times ($t = 0.26,\ 1$, and 3.89 (s)) has been studied for the polydisperse case. Mean and RMS particle velocity profiles show an important improvement while increasing the average time, specially for the lightest ($d_p = 20\ \mu$m) and the heaviest particles ($d_p = 80\ \mu$m) since their number distributions are one of the lowest. Results of the classes 40 and 60 μm are less scattered since both of them contain an important number of samples after one second of physical time.

- A scalability study of the AVBP-EL solver has been performed on a CRAY XD1 supercomputer at CERFACS up to 64 processors. No particular problems related to load-balancing have been observed with the hexahedron-based grid and speedup results are very good. This results on two-phase flow simulations without almost any additional cost and open good perspectives for other future Lagrangian test cases in massively parallel machines. Results with the tetrahedron-based grid leads to a poor speedup if the domain is decomposed without considering particles information and a high load imbalance is presented. On the contrary, balancing particles with a multi-constraint partitioning algorithm improves considerably the speedup. As a result, to preserve a high parallel efficiency on massively parallel machines in cases intrinsically imbalance, the use of load balancing strategies is essential. Results obtained in this thesis confirm the competitive position of Lagrangian formulation compared to its principal competitor, the Eulerian formulation, reducing the disadvantages associated with the Lagrangian formulation.

- For the present case with moderate mass loading, the total number of particles per processor is moderate. The CPU and memory requirements to track particles is lower than for the gaseous phase even with the load balancing problem observed with one of the grids studied. Although the additional cost of Eulerian formulations is independent on the mass loading, for such a dilute case, the Lagrangian approach proved to be faster.

Conclusions and perspectives

Conclusions

Understanding combustion phenomena is the key to progress in terms of power supply, better performance and reduction of consumption in most of today industrial devices, but also directly conditions the control of the pollutants formation. Combustion is a highly non-linear and complex process in which chemistry, fluid mechanics, thermodynamics, radiation and phase change are deeply coupled. The first steps in combustion knowledge were obtained experimentally, but the potential of numerical simulation as a tool to investigate these phenomena, has grown significantly in the last few years with the application of direct numerical simulation (DNS) and large-eddy simulation (LES) to two-phase flow combustion. In many industrial devices, fuel is stored in condensed form and injected as a dispersed liquid phase in the combustion chamber where it mixes with the oxidizer and burns usually through a turbulent combustion process. In order to understand the physics of reactive two-phase flows, a Lagrangian formulation is proposed to treat the dispersed phase. One of the motivations of this study is the rapid increase in computing power which opens a new way for simulations that were prohibitive one decade ago.

The objective of the present thesis is the development and validation of a two-phase Lagrangian formulation on a parallel and unstructured solver, named AVBP, for large-eddy simulations of reacting flows. This solver is a parallel Computational Fluid Dynamics (CFD) code that solves turbulent compressible Navier-Stokes equations in two and three space dimensions. The handling of unstructured or hybrid grids is a key feature of AVBP and represents a major challenge in the implementation of the Lagrangian data structure. A related problem analysed in this thesis is the study of new partitioning algorithms to improve performance on massively parallel machines by reducing the size of partitions and the time of the algorithm to partition. An analysis of performance of the current partitioning algorithms is done and the need of a new partitioning algorithm is highlighted. The chosen algorithm is part of the software package METIS which offers multi-constraint partitioning algorithms and parallel facilities. This feature was used to add load-balancing capabilities to the Lagrangian version developed during this thesis. A comparative study between the new partitioning algorithm and those already available has shown a significant reduction of the CPU time used to partition the test grid and a reduction in the number of duplicated nodes resulting for the partition.

The use of a LES solver and different analysis of parallelism, reordering techniques of partitioning algorithms and computer precision lead indirectly to the study of sensitivity of chaotic systems to initial conditions. Any turbulent flow computed in LES exhibits significant sensitivity to small perturbations, leading to instantaneous solutions which can be totally different. On the contrary, laminar flows are almost insensitive to these parameters even for periodic simulations. One reason of

the divergence of solutions is the propagation of rounding errors in a naturally unstable (turbulent) flow induced by domain partitioning and scheduling of operations. The effect of different parameters is studied and results have been the object of a publication in *AIAA Journal* (see Appendix D).

The two-phase flow Lagrangian module was validated in an Homogeneous Isotropic Turbulence (HIT) which allows a simple analysis of several aspects of performance and particle behaviour. Firstly, a performance analysis of particle search algorithm at the beginning of the simulation was performed. The octree algorithm implemented in this thesis was compared to the simple brute force algorithm for a different number of particles per cell, and for a different number of processors. The octree algorithm showed to be noticeably faster than the brute-force algorithm, as expected. Then, the results of the fluid and particle kinetic energies were analysed and compared to another high-resolution Lagrangian solver, NTMIX3D. Results will show that the third-order spatial scheme (TTGC) performs perfectly well for this test case, providing results which are close to those obtained with the sixth-order scheme of NTMIX3D, which validates the Lagrangian formulation.

The second test case was chosen to validate the Lagrangian module in configurations similar to those encountered in real combustion chambers, in particular the turbulent dispersion properties predicted by the code. The test consists in a particle-laden bluff-body configuration from Borée *et al.* [22] where glass beads are injected into a complex recirculating flow. In this configuration where particles are inertial, although much dependent on the gas flow, the prediction of particles motion strongly depends on the gas phase. Results obtained with the Lagrangian formulation are in good agreement with the experiments, and with the results provided by the LES solver CDP developed at Stanford University. As a general remark, the axial evolution and the radial profiles of the mean and RMS particle velocities of the monodisperse and polydisperse cases are in good agreement with CDP results which validates all developments of the Lagrangian module implemented on the code. Concerning the gaseous results, the monodisperse and the polydisperse cases are very similar. The main difference is in the levels of the mean gas velocity in the recirculation zone which are lower than expected. This implies a reduction in the size of the recirculation bubble in the polydisperse case. Since there are no differences in CPU time between the monodisperse and the polydisperse cases, trying to simulate this flow with a polydisperse particle distribution is closer to the experiments and to reality. The effect of the number of particles on the radial velocity profiles at three physical times has been studied for the polydisperse case. Mean and RMS particle velocity profiles show an important improvement while increasing the average time.

A scalability study of the AVBP-EL solver has been performed on a CRAY XD1 supercomputer at CERFACS up to 64 processors. No particular problems related to load-balancing have been observed with the hexahedron-based grid and speedup is very good. Results with the tetrahedron-based grid leads to a poor speedup if the domain is decomposed without considering the information of particles position and a high load imbalance is presented. Nevertheless, balancing particles with a multi-constraint partitioning algorithm improves considerably the speedup. As a result, to preserve a high parallel efficiency on massively parallel machines in highly imbalanced simulations, the use of load balancing strategies is essential. The results obtained in this thesis confirm the competitive position of Lagrangian formulation compared to the Eulerian formulation once such strategies are efficiently implemented. All these results have been the object of a publication in *Journal of Computational Physics* (see Appendix E).

Perspectives

Despite all the work done in this thesis, a number of important developments deserve to be undertaken. Here we mentioned some of them: the introduction of an evaporation model to simulate two-phase reactive flows, the treatment of particle-wall interactions, the introduction of collision and coalescence models, improvement of particle injection (new geometries, new particle size distributions and multi-injection capabilities), introduction of unfiltered gas fluctuating velocity on particle velocity components, improvement of current search algorithms, etc.

During this thesis, at the beginning of 2007, two new PhD students from the FP6 European project ECCOMET (Efficient and clean combustion experts training) start to work on the Lagrangian version giving solution to an important number of these problems:

- F. Jaegle has introduced an evaporating model that has been validated in academic test cases and is currently working on particle-wall features.
- J.-M. Senoner is focused on the improvement of particle injection options.

In both cases, some of their developments are integrating part of the official Lagrangian version and the others will be included in a near future.

Bibliography

[1] K. Akselvoll & P. Moin (1996) Large-eddy simulation of turbulent confined coannular jets. *J. Fluid Mech.* **315**, 387–411.

[2] A. Albrecht, B. Bédat, T. Poinsot & O. Simonin (1999) Direct numerical simulation and modelling of evaporating droplets in homogeneous turbulence: application to turbulent flames. In *15th Annual Conference on Liquid Atomization and Spray System - ILASS Europe*. Toulouse, France.

[3] C. J. Alpert & A. B. Kahng (1995) Recent directions in netlist partitioning: a survey integration. *Integration: the VLSI Journal* **19** (1-2), 1–81.

[4] E. Anderson (2002) LAPACK3E a Fortran 90-enhanced version of LAPACK. *Tech. Rep.*.

[5] E. Anderson, Z. Bai, C. Bischof, S. Blackford, J. Demmel, J. Dongarra, J. Du Croz, A. Greenbaum, S. Hammarling, A. McKenney & D. Sorensen (1999) *LAPACK Users' guide, third edition*. SIAM, Philadelphia.

[6] S. V. Apte, K. Mahesh & T. Lundgren (2003) A Eulerian-Lagrangian model to simulate two-phase/particulate flows. In *Annual Research Briefs*, pp. 161–171. Center for Turbulence Research, NASA Ames/Stanford Univ.

[7] S. V. Apte, K. Mahesh, P. Moin & J. C. Oefelein (2003) Large-eddy simulation of swirling particle-laden flows in a coaxial-jet combustor. *Int. J. Multiphase Flow* **29** (8), 1311–1331.

[8] V. A. Barker, L. S. Blackford, J. J. Dongarra, J. Du Croz, S. Hammarling, M. Marinova, J. Wasniewski & P. Yalamov (2001) *LAPACK95 Users' guide*. SIAM, Philadelphia.

[9] S. T. Barnard & H. D. Simon (1993) A fast multilivel implementation of recursive spectral bisection for partitioning unstructured problems. In *Proc. of the 6th SIAM conference on Parallel Processing for Scientific Computing*, pp. 711–718.

[10] G. K. Batchelor (1949) Diffusion in a field of homogeneous turbulence. Eulerian analysis. *Aust. J. Sci. Res* **2**, 437–450.

[11] M. Baum & T. Poinsot (1992) A direct numerical simulation code for studies of combustion phenomena. *Tech. Rep.* 920201. Laboratoire EM2C du CNRS et de l'École Centrale de Paris.

[12] J. L. Bentley (1975) Multidimensional binary search trees used for associative searching. *Communications of the ACM* **18**, 509–517.

[13] M. J. Berger & S. H. Bokhari (1987) A partitioning strategy for nonuniform problems on multiprocessors. *IEEE Trans. on Computers* **C-36** (5), 570–580.

[14] P. Berkhin (2002) *Survey of clustering data-mining techniques*. Accrue Software, San Jose, USA.

[15] M. Bern & P. Blassmann (2000) Mesh generation. In *Handbook of Computational Geometry* (ed. Sack & Urrutia), chap. 6. Elsevier Science.

[16] J. Bertin (1981) *Graphics and Graphic Information Processing*. Berlin: Walter de Gruyter & Co., originally: *La graphique et le traitemente graphique de l'information*, 1967, translated in English by William J. Berg and Paul Scott.

[17] J. Bertin (1983) *Semiology of Graphics–Diagrams Networks Maps*. Madison: The University of Wisconsin Press., originally: *Semiologue graphique*, 1967, translated in English by William J. Berg.

[18] M. Boileau (2007) Simulation aux grandes échelles de l'allumage diphasique des foyers aéronautiques. Phd thesis, INP Toulouse.

[19] M. Boivin, O. Simonin & K. D. Squires (1998) Direct numerical simulation of turbulence modulation by particles in isotropic turbulence. *J. Fluid Mech.* **375**, 235–263.

[20] M. Boivin, O. Simonin & K. D. Squires (2000) On the prediction of gas-solid flows with two-way coupling using large eddy simulation. *Phys. Fluids* **12** (8), 2080–2090.

[21] J. Bonet & J. Periare (1991) An alternating digital tree (ADT) algorithm for 3D geometric searching and intersection problems. *Int. J. Numer. Meth. Eng.* **31** (1), 1–17.

[22] J. Borée, T. Ishima & I. Flour (2001) The effect of mass loading and inter-particle collisions on the development of the polydispersed two-phase flow downstream of a confined bluff body. *J. Fluid Mech.* **443**, 129–165.

[23] H. Boughanem (1998) Évaluation des termes de transport et de dissipation de surface de flamme par simulation numérique directe de la combustion turbulente. Phd thesis, Université de Rouen.

[24] H. Boughanem & A. Trouvé (1996) Validation du code de simulation directe NTMIX3D pour le calcul des écoulements turbulents réactifs. *Tech. Rep.* 42907. Institut Français du Pétrole.

[25] J. Boussinesq (1877) Essai sur la théorie des eaux courantes. *26ème édn. Acad. Sci. Paris* .

[26] A. N. Brooks & T. Hugues (1982) Streamline upwind/Petrov-Galerkin formulations for convection dominated flows with particular emphasis on the incompressible Navier-Stokes equations. *Comput. Methods Appl. Mech. Eng.* **32**.

[27] A. B. Chace (1979) *The Rhind Mathematical Papyrus*. National Council of Teachers of Mathematics, Reston, VA, The cited quote is from Volume 1, pp. 48-49.

[28] S. de Chaisemartin, L. Fréret, D. Kah, F. Laurent, R. O. Fox, J. Réveillon & M. Massot (2008) Turbulent combustion of polydisperse evaporating sprays with droplets crossing: Eulerian modeling and validation in the infinite Knudsen limit. In *Proc. of the Summer Program*. Center for Turbulence Research, NASA Ames/Stanford Univ.

[29] J. R. Chasnov (1991) Simulation of the Kolmogorov inertial subrange using an improved subgrid model. *Phys. Fluids A* **3** (1), 188–200.

[30] C. H. Chen (2002) Generalized association plots: Information visualization via iteratively generated correlation matrices. *Statistica Sinica* **12**, 7–29.

[31] K. H. Chen & J. S. Shuen (1993) Analysis of transcritical spray phenomena in turbulent mixing layers. *31st AIAA Aerospace Sciences Meeting* (AIAA 93-0108).

[32] R. Clift, J. R. Grace & M. E. Weber (1978) *Bubbles, Drops, and Particles*. Academic Press.

[33] O. Colin & M. Rudgyard (2000) Development of high-order Taylor-Galerkin schemes for unsteady calculations. *J. Comput. Phys.* **162** (2), 338–371.

[34] G. Comte-Bellot (1965) Écoulement turbulent entre deux parois parallèles. *Publ. Sci. Tech. Min. Air.* (419).

[35] C. Crowe, M. Sommerfeld & Y. Tsuji (1998) *Multiphase Flows with Droplets and Particles*.

[36] B. Cuenot & T. Poinsot (1996) Asymptotic and numerical study of diffusion flames with variable Lewis number and finite rate chemistry. *Combust. Flame* **104**, 111–137.

[37] D. E. Culler & A. Gupta J. P. Singh (1999) *Parallel Computer Architecture: A Hardware/Software Approach*. Morgan Kaufmann Publishers.

[38] E. Cuthill & J. McKee (1969) Reducing the bandwidth of sparse symmetric matrices. In *Proceedings of the 24th National Conference of the ACM*, pp. 157–172.

[39] J. Deardorff (1970) A numerical study of three-dimensional turbulent channel flow at large Reynolds numbers. *J. Fluid Mech.* **41**, 453–480.

[40] J. W. Deardorff (1974) Three-dimensional numerical study of the height and mean structure of a heated planetary boundary layer. *Boundary-Layer Meteorology* **7** (1), 81–106.

[41] P. Dellenback, D. Metzger & G. Neitzel (1988) Measurement in turbulent swirling flows through an abrupt axisymmetric expansion. *AIAA Journal* **13** (4), 669–681.

[42] E. Deutsch (1992) Dispersion de particules dans une turbulence homogène isotrope stationnaire calculée par simulation numérique directe des grandes échelles. Phd thesis, EDF-DRD.

[43] J. Donea (1984) Taylor-Galerkin method for convective transport problems. *Int. J. Numer. Meth. Fluids* **20** (1), 101–119.

[44] J. Donea, L. Quartapelle & V. Selmin (1987) An analysis of time discretization in the finite element solution of hyperbolic problems. *J. Comput. Phys.* **70**, 463–499.

[45] H. L. Dryden & A. M. Kuethe (1930) Effect of turbulence in wind tunnel measurements. *Tech. Rep.* 342. Natl. Advisory Comm. Aeronaut.

[46] F. Ducros, P. Comte & M. Lesieur (1996) Large-eddy simulation of transition to turbulence in a boundary layer developing spatially over a flat plate. *J. Fluid Mech.* **326**, 1–36.

[47] F. Ducros, F. Nicoud & T. Poinsot (1998) Wall-adapting local eddy-viscosity models for simulations in complex geometries. In *ICFD*, pp. 293–300. Baines M. J.

[48] I. S. Duff, A. M. Erisman & J. K. Reid (1986) *Direct methods for sparse matrices*. Oxford University Press.

[49] A. E. Dunlop & B. W. Kernighan (1985) A procedure for placement of standard-cell VLSI circuits. *IEEE Trans. on Computers - Aided Design of Integrated Circuits and Systems* **CAD-4** (1), 92–98.

[50] J. K. Eaton & J. R. Fessler (1994) Preferential concentration of particles by turbulence. *Int. J. Multiphase Flow* **20**, 169–209.

[51] S. Elghobashi & G. C. Truesdell (1992) Direct simulation of particle dispersion in a decaying isotropic turbulence. *J. Fluid Mech.* **242**, 655–700.

[52] S. Elghobashi & G. C. Truesdell (1993) On the two-way interaction between homogeneous turbulence and dispersed solid particles. I: Turbulence modification. *Phys. Fluids* **5** (7), 1790–1801.

[53] E. Elmroth & F. Gustavson (2001) A faster and simpler recursive algorithm for the LAPACK routine DGELS. *BIT Numerical Mathematics* **41** (5), 936–949.

[54] G. M. Faeth (1983) Evaporation and combustion of sprays. *Prog. Energy Comb. Sci.* **9**, 1–76.

[55] G. M. Faeth (1987) Mixing, transport and combustion sprays. *Prog. Energy Comb. Sci.* **13** (4), 293–345.

[56] A. Favre (1969) Statistical equations of turbulent gases. In *Problems of hydrodynamics and continuum mechanics*, pp. 231–266. Philadelphia: SIAM.

[57] H. Faxen (1922) Der Widerstand gegen die Bewegung einer starren Kugel in einer zähen Flüssigkeit, die zwischen zwei parallelen ebenen Wänden eingeschlossen ist. *Annalen der Physik* **68**, 89–119.

[58] P. Fede & O. Simonin (2006) Numerical study of the subgrid fluid turbulence effects on the statistics of heavy colliding particles. *Phys. Fluids* **18** (045103).

[59] Y. T. Feng & D. R. J. Owen (2002) An augmented spatial digital tree algorithm for contact detection in computational mechanics. *Int. J. Numer. Meth. Eng.* **55**, 159–176.

[60] P. Février & O. Simonin (2000) Statistical and continuum modelling of turbulent reactive particulate flows. Part 2: Application of a two-phase second-moment transport model for prediction of turbulent gas-particle flows. In *Theoretical and Experimental Modeling of Particulate Flows, Lecture Series 2000-06, von Karman Institute for Fluid Dynamics, Rhode Saint Genese (Belgium).*

[61] P. Février, O. Simonin & K. Squires (2005) Partitioning of particle velocities in gas-solid turbulent flows into a continuous field and a spatially uncorrelated random distribution: Theoretical formalism and numerical study. *J. Fluid Mech.* **533**, 1–46.

[62] J. Fingberg, A. Basermann, G. Lonsdale, J. Clinckermaillie, J. Gratien & R. Ducloux (2000) Dynamic load-balancing for parallel structural mechanics simulations with DRAMA. In *ECT2000*.

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

[64] M. J. FLynn (1972) Some computer organizations and their effectiveness. *IEEE Trans. on Computers* **C-21** (9), 948–960.

[65] L. Fréret, F. Laurent, S. de Chaisemartin, D. Kah, R. O. Fox, P. Vedula, J. Reveillon, O. Thomine & M. Massot (2008) Turbulent combustion of polydisperse evaporating sprays with droplet crossing: Eulerian modeling of collisions at finite Knudsen and validation. In *Proc. of the Summer Program*. Center for Turbulence Research, NASA Ames/Stanford Univ.

[66] J. C. H. Fung, J. C. R. Hunt, N. A. Malik & R. J. Perkins (1992) Kinematic simulation of homogeneous turbulence by unsteady random Fourier modes. *J. Fluid Mech.* **236**, 281–318.

[67] M. García (2003) Analysis of precision differences observed for the AVBP code. *Tech. Rep.* TR/CFD/03/84. CERFACS, Toulouse, France.

[68] M. García (2007) Interpolateur parallèle de solutions. Projet ANR CAMPAS, Livrable L3 CR/CFD/07/149. CERFACS, Toulouse, France.

[69] M. García, Y. Sommerer, T. Schönfeld & T. Poinsot (2005) Evaluation of Euler/Euler and Euler/Lagrange strategies for large eddy simulations of turbulent reacting flows. In *ECCOMAS Thematic Conference on Computational Combustion*.

[70] R. Gatignol (1983) The Faxén formulae for a rigid particle in an unsteady non-uniform Stokes flow. *J. Méc. Théor. Appl.* **1** (2), 143–160.

[71] A. George (1971) Computer implementation of the finite element method. *Tech. Rep.* STAN-CS-208. Stanford University, Department of Computer Science.

[72] M. Germano, U. Piomelli, P. Moin & W. Cabot (1991) A dynamic subgrid-scale eddy viscosity model. *Phys. Fluids* **3** (7), 1760–1765.

[73] N. Gibbs, W. Poole & P. Stockmeyer (1976) A comparison of several bandwidth and profile reduction algorithms. *ACM Trans. Math. Software* **2**, 322–330.

[74] L. Giraud, P. Noyret, E. Sevault & V. Van Kemenade (1995) IPM - User's guide and reference manual. *Tech. Rep.* TR/PA/95/01. CERFACS, Toulouse, France.

[75] A. K. Gupta, D. G. Lilley & N. Syred (1984) *Swirl flows*. Abacus Press.

[76] F. Ham, S. V. Apte, G. Iaccarino, X. Wu, M. Herrmann, G. Constantinescu, K. Mahesh & P. Moin (2003) Unstructured LES of reacting multiphase flows in realistic gas turbine combustors. In *Annual Research Briefs*, pp. 139–160. Center for Turbulence Research, NASA Ames/Stanford Univ.

[77] F. Ham & G. Iaccarino (2004) Energy conservation in collocated discretization schemes on unstructured meshes. In *Annual Research Briefs*, pp. 3–14. Center for Turbulence Research, NASA Ames/Stanford Univ.

[78] G. Hanrot, V. Lefèvre, D. Stehlé & P. Zimmermann (2007) Worst cases for a periodic function with large arguments. In *Proceedings of the 18th IEEE Symposium on Computer Arithmetic* (ed. Peter Kornerup & Jean-Michel Muller), pp. 133–140. Los Alamitos, CA: IEEE Computer Society Press.

[79] R. J. Hanson & A. Thomas (1984) Flame development in swirling flows in closed vessels. *Combust. Flame* **55**, 255–277.

[80] R. von Hanxleden & L. R. Scott (1991) Load balancing on message passing architectures. *J. Parallel and Distributed Computing* **13** (3), 312–324.

[81] J. A. Hartigan (1972) Direct clustering of a data matrix. *Journal of the American Statistical Association* **67** (337), 123–129.

[82] A. Haselbacher, F. M. Najjar & J. P. Ferry (2007) An efficient and robust particle-localization algorithm for unstructured grids. *J. Comput. Phys.* **225** (2), 2198–2213.

[83] D. C. Haworth & K. Jansen (2000) Large-eddy simulation on unstructured deforming meshes: towards reciprocating IC engines. *Comput. Fluids* **29**, 493–524.

[84] B. T. Helenbrook & C. F. Edwards (2002) Quasi-steady deformation and drag of uncontaminated liquid drops. *Int. J. Multiphase Flow* **28** (10), 1631–1657.

[85] B. Hendrickson & K. Devine (2000) Dynamic load balancing in computational mechanics. *Comput. Methods Appl. Mech. Eng.* **2-4** (184), 485–500.

[86] B. Hendrickson & R. Leland (1993) A multilevel algorithm for partitioning graphs. *Tech. Rep.* SAND93-1301. Sandia National Laboratories, Albuquerque, NM.

[87] B. Hendrickson & R. Leland (1995) The Chaco user's guide, version 2.0. *Tech. Rep.* SAND95-2344. Sandia National Laboratories, Albuquerque, NM.

[88] J. O. Hinze (1987) *Turbulence*. McGraw-Hill Classic Textbook Reissue.

[89] C. Hirsch (2007) Finite volume method and conservative discretization with an introduction to finite element method. In *Numerical Computation of internal & external flows: Fundamentals of Computational Fluid Dynamics, second edition*, chap. 5, pp. 203–248. New York: John Wiley & Sons.

[90] J. C. R. Hunt, J. C. Buell & A. A. Wray (1987) Big whorls carry little whorls. In *Proc. of the Summer Program*, pp. 77–94. Center for Turbulence Research, NASA Ames/Stanford Univ.

[91] IEEE (1985) *IEEE Standard for Binary Floating-Point Arithmetic*. ANSI/IEEE Std 754-1985. Institute of Electrical and Electronics Engineers, New York.

[92] T. Ishima, J. Borée, P. Fanouillère & I. Flour (1999) Presentation of a two phase flow data base obtained on the flow loop Hercule. In *Proc. of the 9th Workshop on Two-Phase Flow Predictions*. Merseburg, Germany.

[93] J. Jiménez (2004) Turbulence and vortex dynamics. *Tech. Rep.*. École Polytechnique, Madrid.

[94] R. W. Johnson (1998) *The handbook of fluid dynamics*. CRC Press.

[95] G. Karypis & V. Kumar (1998) A fast and high quality multilevel scheme for partitioning irregular graphs. *SIAM Journal on Scientific Computing* **20** (1), 359–392.

[96] G. Karypis & V. Kumar (1998) Multilevel algorithms for multi-constraint graph partitioning. *Tech. Rep.* 98-019. University of Minnesota, Department of Computer Science/Army HPC Research Center.

[97] G. Karypis, K. Schloegel & V. Kumar (2003) *Parmetis: Parallel Graph Partitioning and Sparse Matrix Ordering Library*. Univ. of Minnesota, Department of Computer Science and Engineering, USA, (version 3.1).

[98] A. Kaufmann (2004) Vers la simulation des grandes échelles en formulation Euler/Euler des écoulements réactifs diphasiques. Phd thesis, INP Toulouse.

[99] J. de Keyser & D. Roose (1992) Grid partitioning by inertial recursive bisection. *K. U. Leuven, Dept. of Computer Science, Belgium* **Report TW 174**.

[100] J. Kim & P. Moin (1985) Application of a fractional-step method to incompressible Navier-Stokes equations. *J. Comput. Phys.* **59** (2), 308–323.

[101] D. E. Knuth (1997) *The Art of Computer Programming: Fundamental Algorithms*. Addison-Wesley, Reading, MA, third edition, Vol. 1.

[102] D. E. Knuth (1998) *The Art of Computer Programming: Sorting and Searching*. Addison-Wesley, Reading, MA, second edition, Vol. 3.

[103] A. N. Kolmogorov (1941) The local structure of turbulence in incompressible viscous fluid for very large Reynolds numbers. *C.R. Acad. Sci. USSR* **30**, 301.

[104] R. H. Kraichnan (1976) Eddy viscosity in two and three dimensions. *J. Atmos. Sci.* **33**, 1521–1536.

[105] V. Kumar (2000) Graph partitioning for dynamic, adaptive and multi-phase computations. In *Workshop "Resource Recovery"*. Material from IMA Talks.

[106] K. K. Kuo (1986) *Principles of Combustion*. New York: John Wiley.

[107] N. Lamarque (2007) Schémas numériques et conditions limites pour la simulation aux grandes échelles de la combustion diphasique dans les foyers d'hélicoptère. Phd thesis, INP Toulouse.

[108] J. Lavedrine (2008) Simulations aux grandes échelles de l'écoulement diphasique dans des modèles d'injecteur de moteurs aéronautiques. Phd thesis, INP Toulouse.

[109] J. Laviéville (1997) Simulations numériques et modélisation des interactions entre l'entrainement par la turbulence et les collisions interparticulaires en écoulement gaz-solide. Phd thesis, Université de Rouen.

[110] P. D. Lax & B. Wendroff (1960) Systems of conservation laws. *Commun. Pure Appl. Math.* **13**, 217–237.

[111] P. D. Lax & B. Wendroff (1964) Difference schemes for hyperbolic equations with high order of accuracy. *Commun. Pure Appl. Math.* **17**, 381–398.

[112] A. H. Lefebvre (1989) *Atomization and Sprays*. Taylor & Francis.

[113] C. Leith (1971) Atmospheric predictability and two-dimensional turbulence. *J. Atmos. Sci.* **28** (2), 145–161.

[114] S. Lele (1992) Compact finite difference schemes with spectral like resolution. *J. Comput. Phys.* **103**, 16–42.

[115] A. Leonard (1974) Energy cascade in large eddy simulations of turbulent fluid flows. *Adv. Geophys.* **18** (A), 237–248.

[116] J. G. Lewis (1982) The Gibbs-Poole-Stockmeyer and Gibbs-King algorithms for reordering sparse matrices. *ACM Trans. Math. Software* **8** (2), 190–194.

[117] D. K. Lilly (1967) The representation of small-scale turbulence in numerical simulation experiments. In *Proceedings of the IBM Scientific Computing Symposium on Environmental Sciences*, pp. 195–210. Yorktown Heights, USA.

[118] D. K. Lilly (1992) A proposed modification of the Germano sub-grid closure method. *Phys. Fluids* **4** (3), 633–635.

[119] W.-H. Liu & A. H. Sherman (1976) Comparative analysis of the Cuthill-McKee and the reverse Cuthill-McKee ordering algorithms for sparse matrices. *SIAM Journal of Numerical Analysis* **13** (2), 198–213.

[120] R. Löhner (1995) Robust Vectorised Search Algorithms for Interpolation on Unstructured Grids. *J. Comput. Phys.* **118** (2), 380–387.

[121] R. Löhner & J. Ambrosiano (1990) A Vectorised Particle Tracer for Unstructured Grids. *J. Comput. Phys.* **91** (1), 22–31.

[122] E. N. Lorenz (1963) Deterministic nonperiodic flow. *J. Atmos. Sci.* **20** (2), 130–141.

[123] E. N. Lorenz (1993) *The essence of chaos*. University of Washington Press, Chapters 1-2.

[124] T. S. Lundgren (2003) Linear forced isotropic turbulence. In *Annual Research Briefs*, pp. 461–473. Center for Turbulence Research, NASA Ames/Stanford Univ.

[125] K. Mahesh, G. Constantinescu & P. Moin (2004) A numerical method for large-eddy simulation in complex geometries. *J. Comput. Phys.* **197** (1), 215–240.

[126] P. J. Mason (1994) Large-eddy simulation: a critical review of the technique. *Quarterly Journal of the Royal Meteorology Society* **120 (A)** (515), 1–26.

[127] A. Massol (2004) Simulation d'écoulement autour de réseaux de particules sphériques. Phd thesis, INP Toulouse.

[128] M. R. Maxey (1987) The gravitational settling of aerosol particles in homogeneous turbulence and random flow fields. *J. Fluid Mech.* **174**, 441–465.

[129] M. R. Maxey & J. J. Riley (1983) Equation of motion for a small rigid sphere in a nonuniform flow. *Phys. Fluids* **26** (4), 883–889.

[130] R. Mei, R. J. Adrian & T. J. Hanratty (1991) Particle dispersion in isotropic turbulence under Stokes drag and Basset force with gravitational settling. *J. Fluid Mech.* **225**, 481–495.

[131] O. Métais & M. Lesieur (1986) Statistical probability of decaying turbulence. *Journal of Atmospheric Science* **43** (9), 857–870.

[132] P. Moin (2004) Large eddy simulation of multi-phase turbulent flows in realistic combustors. *Prog. Comput. Fluid Dynamics* **4**, 237–240.

[133] P. Moin & S. V. Apte (2006) Large-eddy simulation of realistic gas turbine combustors. *Am. Inst. Aeronaut. Astronaut. J.* **44** (4), 698–708.

[134] P. Moin & J. Kim (1982) Numerical investigation of turbulent channel flow. *J. Fluid Mech.* **118**, 341–377.

[135] M. Moreau (2006) Modélisation numérique directe et des grandes échelles des écoulements turbulents gaz-particules dans le formalisme eulérien mésoscopique. Phd thesis, INP Toulouse.

[136] J.-B. Mossa (2005) Extension polydisperse pour la description Euler-Euler des écoulements diphasiques réactifs. Phd thesis, INP Toulouse.

[137] V. Moureau (2004) Simulation aux grandes échelles de l'aérodynamique interne des moteurs à piston. Phd thesis, École Centrale Paris.

[138] V. Moureau, G. Lartigue, Y. Sommerer, C. Angelberger, O. Colin & T. Poinsot (2005) High-order methods for DNS and LES of compressible multi-component reacting flows on fixed and moving grids. *J. Comput. Phys.* **202** (2), 710–736.

[139] R. A. Mugele & H. D. Evans (1951) Droplet size distribution in sprays. *Ind. Engr. Chem.* **43** (6), 1317–1324.

[140] J. A. Mulholland, R. K. Srivastava & J. O. L. Wendt (1988) Influence of droplet spacing on drag coefficient in nonevaporating, monodisperse streams. *AIAA Journal* **26** (10), 1231–1237.

[141] J.-D. Müller (2008) *A User's guide to hip*. CERFACS, Toulouse, France, (version 1.23 "Kaa").

[142] C. Narayanan, D. Lakehal & G. Yadigaroglu (2002) Linear stability analysis of particle-laden mixing layers using Lagrangian particle tracking. *Powder Technology* **125** (2-3), 122–130.

[143] F. Nicoud (1999) Defining wave amplitude in characteristic boundary conditions. *J. Comput. Phys.* **149** (2), 418–422.

[144] J. C. Oefelein & V. Yang (1996) Analysis of transcritical spray phenomena in turbulent mixing layers. *34th AIAA Aerospace Sciences Meeting* (AIAA 96-0085).

[145] C. W. Oseen (1910) Über die Stokes'sche Formel und über eine verwandte Aufgabe in der Hydrodynamik. *Arkiv. Mat. Astron. Physik* **6** (29).

[146] M. L. Overton (2001) *Numerical Computing with IEEE Floating Point Arithmetic*. SIAM, Philadelphia.

[147] R. Pandya & F. Mashayek (2002) Two-fluid large-eddy simulation approach for particle-laden turbulent flows. *Int. J. Heat and Mass Transfer* **45**, 4753–4759.

[148] S. Pannala & S. Menon (1998) Large eddy simulations of two-phase turbulent flows. *36th AIAA Aerospace Sciences Meeting* (AIAA 98-0163).

[149] Y. H. Pao (1965) Structure of turbulent velocity and scalar fields at large wavenumbers. *Phys. Fluids* **8**, 1063–1075.

[150] Y. H. Pao (1968) Transfert of turbulent energy and scalar quantities at large wavenumbers. *Phys. Fluids* **11**, 1371–1372.

[151] R. Paoli, J. Hélie & T. Poinsot (2004) Contrail formation in aircraft wakes. *J. Fluid Mech.* **502**, 361–373.

[152] R. Paoli & K. Shariff (2004) Direct numerical simulation of turbulent condensation in clouds. In *Annual Research Briefs*, pp. 305–316. Center for Turbulence Research, NASA Ames/Stanford Univ.

[153] R. Paoli & K. Shariff (2008) Turbulent condensation of droplets: direct simulation and a stochastic model. *J. Atmos. Sci.* **DOI 10.1175/2008JAS2734.1**.

[154] S. Pascaud (2006) Vers la simulation aux grandes échelles des écoulements turbulents diphasiques réactifs: application aux foyers aéronautiques. Phd thesis, INP Toulouse.

[155] T. Passot & A. Pouquet (1987) Numerical simulation of compressible homogeneous flows in the turbulent regime. *J. Fluid Mech.* **181**, 441–466.

[156] F. Pellegrini (2007) Scotch and libscotch 5.0 user's guide. *Tech. Rep.*. ENSEIRB and LaBRI, Université Bordeaux I, Talence, France.

[157] C. D. Pierce & P. Moin (1998) Large eddy simulation of a confined coaxial jet with swirl and heat release. In *29th Fluid Dynamics Conference* (ed. AIAA Paper 98-2892). Albuquerque, NM.

[158] U. Piomelli (1993) High Reynolds number calculations using the dynamic subgrid-scale stress model. *Phys. Fluids A* **5**, 1484–1490.

[159] H. Pitsch & N. Peters (1998) A consistent flamelet formulation for non-premixed combustion considering differential diffusion effects. *Combust. Flame* **114**, 26–40.

[160] R. W. Pitz & J. W. Daily (1983) Combustion in a turbulent mixing layer formed at a rearward facing step. *AIAA Journal* **21**, 1565–1570.

[161] T. Poinsot (1987) Analyse des instabilités de combustion de foyers turbulents prémélangés. Thèse d'état, Université d'Orsay.

[162] T. Poinsot, D. Haworth & G. Bruneaux (1993) Direct simulation and modelling of flame-wall interaction for premixed turbulent combustion. *Combust. Flame* **95** (1/2), 118–133.

[163] T. Poinsot & S. Lele (1992) Boundary conditions for direct simulations of compressible viscous flows. *J. Comput. Phys.* **101** (1), 104–129.

[164] T. Poinsot & D. Veynante (1999) Combustion numérique. *Cours de DEA, ENSEEIHT-IMFT* .

[165] T. Poinsot & D. Veynante (2005) *Theoretical and numerical combustion*. R.T. Edwards, 2nd edition.

[166] A. Pothen, H. D. Simon & K.-P. Liou (1990) Partitioning sparse matrices with eigenvectors of graphs. *SIAM J. Matrix Anal. Appl.* **11** (3), 430–452.

[167] R. Preis & R. Diekmann (1996) The PARTY partitioning-library, user guide - version 1.1. *Tech. Rep.* tr-rsfb-96-024. Dept. of Computer Science, Univ. of Paderborn, Paderborn, Germany.

[168] W. H. Press, S. A. Teukolsky, W. T. Vetterling & B. P. Flannery (1996) *Numerical recipes in Fortran 90 (second edition): the art of parallel scientific computing*. New York, NY, USA: Cambridge University Press.

[169] W. H. Press, S. A. Teukolsky, W. T. Vetterling & B. P. Flannery (2003) *Numerical Recipes in Fortran 77, The Art of Scientific Computing, second edition*. Cambridge: Cambridge University Press.

[170] C. Prière (2005) Simulation aux grandes échelles : application au jet transverse. Phd thesis, INP Toulouse.

[171] L. Quartapelle & V. Selmin (1993) High-order Taylor-Galerkin methods for non-linear multidimensional problems.

[172] A. Quarteroni & A. Valli (1999) *Domain Decomposition Methods for Partial Differential Equations*. Oxford Science Publications.

[173] M. W. Reeks (1991) On a kinetic equation for the transport of particles in turbulent flows. *Phys. Fluids* **3** (3).

[174] M. W. Reeks (1993) On the constitutive relations for dispersed particles in nonuniform flows. I. dispersion in a simple shear flow. *Phys. Fluids* **5** (3), 750–761.

[175] M. W. Reeks & S. McKee (1984) The dispersive effects of Basset history forces on particle motion in a turbulent flow. *J. Fluid Mech.* **27** (7), 1573–1582.

[176] J. Réveillon, C. Pera, M. Massot & R. Knikker (2004) Eulerian analysis of the dispersion of evaporating polydispersed sprays in a statistically stationary turbulent flow. *J. Turb.* **5** (1), 1–27.

[177] O. Reynolds (1883) An experimental investigation of the circumstances which determine the motion of water shall be direct or sinuous, and the law of resistance in parallel channels. *Philos. Trans. R. Soc. London Ser. A* **174**, 935–982.

[178] O. Reynolds (1894) On the dynamical theory of incompressible viscous flows and the determination of the criterion. *Philos. Trans. R. Soc. London Ser. A* **186**, 123–161.

[179] E. Riber (2007) Développement de la méthode de simulation aux grandes échelles pour les écoulements diphasiques turbulents. Phd thesis, INP Toulouse.

[180] E. Riber, M. Moreau, O. Simonin & B. Cuenot (2005) Towards large eddy simulation of non-homogeneous particle laden turbulent gas flows using Euler-Euler approach. In *11th Workshop on Two-Phase Flow Predictions*. Merseburg, Germany.

[181] L. F. Richardson (1922) *Weather Prediction by Numerical Process*. Cambridge University Press.

[182] S. Roux, G. Lartigue, T. Poinsot, U. Meier & C. Bérat (2005) Studies of mean and unsteady flow in a swirled combustor using experiments, acoustic analysis and large eddy simulations. *Combust. Flame* **141**, 40–54.

[183] M. Rudgyard (1990) Cell vertex methods for compressible gas flows. Phd thesis, Oxford University Computing Laboratory.

[184] M. Rudgyard (1993) Cell vertex methods for steady inviscid flow. In *Lectures Series 1993-04*. Von Karman Institute for Fluid Dynamics.

[185] A. Safjan & J. T. Oden (1994) High-order Taylor-Galerkin methods for linear hyperbolic systems. *J. Comput. Phys.* **120**, 206–230.

[186] P. Sagaut (1998) *Introduction à la simulation des grandes échelles*, mathématiques & applications edn. Springer.

[187] P. Sagaut (1998) *Large Eddy Simulation for Incompressible Flows*. Berlin: Springer.

[188] H. Samet (1984) The quadtree and related hierarchical data structures. *J. Comput. Phys.* **16** (2), 187–260.

[189] V. Sankaran & S. Menon (2002) LES of spray combustion in swirling flows. *J. Turb.* **3**, 011.

[190] R. W. Schefer, M. Namazian & J. Kelly (1994) Velocity measurements in turbulent bluff-body stabilized flows. *Am. Inst. Aeronaut. Astronaut. J.* **32** (9), 1844–1851.

[191] L. Schiller & A. Naumann (1935) A drag coefficient correlation. *VDI Zeitung* **77**, 318–320.

[192] K. Schloegel, G. Karypis & V. Kumar (2002) Graph partitioning for high-performance scientific simulations. In *Sourcebook on Parallel Computing* (ed. J. Dongarra, I. Foster, G. Fox, W. Gropp, K. Kennedy, L. Torczon & A. White), chap. 18, pp. 491–541. San Francisco, CA: Morgan Kaufmann.

[193] T. Schönfeld & M. Rudgyard (1998) COUPL and its use within hybrid mesh CFD applications. In *Proc. of the 10th International Conference on Parallel CFD 98* (ed. A. Ecer, D. Emerson, J. Périaux & N. Satofuka). Hsinchu, Taiwan: Elsevier, Amsterdam.

[194] T. Schönfeld & M. Rudgyard (1999) Steady and unsteady flows simulations using the hybrid flow solver AVBP. *AIAA Journal* **37** (11), 1378–1385.

[195] U. Schumann (1975) Subgrid scale model for finite difference simulations of turbulent flows in plane channels and annuli. *J. Comput. Phys.* **18**, 376–404.

[196] D. Seldner & T. Westermann (1988) Algorithms for Interpolation and Localization in Irregular 2D Meshes. *J. Comput. Phys.* **79** (1), 1–11.

[197] L. Selle, G. Lartigue, T. Poinsot, P. Kaufmann, W. Krebs & D. Veynante (2002) Large-eddy simulation of turbulent combustion for gas turbines with reduced chemistry. In *Proc. of the Summer Program*, pp. 333–344. Center for Turbulence Research, NASA Ames/Stanford Univ.

[198] J.-M. Senoner, M. García, S. Mendez, G. Staffelbach, O. Vermorel & T. Poinsot (2008) Growth of Rounding Errors and Repetitivity of Large-Eddy Simulations. *AIAA Journal* **46** (7), 1773–1781.

[199] J. Shi & J. Malik (1997) Normalized cuts and image segmentation. In *Proc. of the 1997 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 731–737. IEEE Computer Society Press.

[200] H. D. Simon (1991) Partitioning of unstructured problems for parallel processing. *Computing Systems in Eng.* **2/3**, 135–148.

[201] H. D. Simon & S.-H. Teng (1997) How good is recursive bisection? *SIAM J. Sci. Comput.* **18** (5), 1436–1445.

[202] O. Simonin (1991) Prediction of the dispersed phase turbulence in particle-laden jets. In *Proc. 1st Int. Symp. Gas-Solid Flows*, pp. 197–206. ASME.

[203] D. G. Sloan, P. J. Smith & L. D. Smoot (1986) Modeling of swirl in turbulent flow systems. *Prog. Energy Comb. Sci.* **12**, 163–250.

[204] J. Smagorinsky (1963) General circulation experiments with the primitive equations: I. The basic experiment. *Mon. Weather Rev.* **91** (3), 99–164.

[205] W. H. Snyder & J. L. Lumley (1971) Some measurements of particle velocity autocorrelation functions in a turbulent flow. *J. Fluid Mech.* **48** (1), 41–71.

[206] K. D. Squires & J. K. Eaton (1990) Particle response and turbulence modification in isotropic turbulence. *Phys. Fluids A* **2** (7), 1191–1203.

[207] K. D. Squires & J. K. Eaton (1991) Measurements of particle dispersion obtained from direct numerical simulations of isotropic turbulence. *J. Fluid Mech.* **226**, 1–35.

[208] K. D. Squires & J. K. Eaton (1991) Preferential concentration of particles by turbulence. *Phys. Fluids A* **3** (5), 1169–1178.

[209] J. S. Stoer & R. Bulirsch (1980) *An introduction to numerical analysis*. Berlin: Springer.

[210] A. Stoessel, M. Hilka & M. Baum (1994) 2D direct numerical simulation of turbulent combustion on massively parallel processing platforms. In *1994 EUROSIM Conference on Massively Parallel Computing*, pp. 793–800. Delft: Elsevier Science B.V.

[211] G. G. Stokes (1851) On the effect of the inertial friction of fluids on the motions of pendulums. *Trans. Cambridge Phil. Soc.* **9**, 8–23.

[212] R. Struijs, I. d'Ast, T. Schönfeld, M. Rudgyard, F. Nicoud, G. Hernandez & O. Colin (1996) AVBP Version 3.0 results on some basic test cases. *Tech. Rep.* TR/CFD/96/10. CERFACS.

[213] S. Sundaram & L. R. Collins (1999) A numerical study of the modulation of isotropic turbulence by suspended particles. *J. Fluid Mech.* **379**, 105–143.

[214] G. I. Taylor (1921) Diffusion by continuous movements. *Proc. London Math. Soc* **20**, 196–211.

[215] V. E. Taylor & B. Nour-Omid (1994) A study of the factorization fill-in for a parallel implementation of the finite element method. *Int. J. Numer. Meth. Eng.* **37** (22), 3809–3823.

[216] C.-M. Tchen (1947) Mean value and correlation problems connected with the motion of small particles suspended in a turbulent fluid. Phd thesis, Technical University of Delft, Netherlands.

[217] P. J. Thomas (1992) On the influence of the Basset history force on the motion of the particle through a fluid. *Phys. Fluids* **4** (9), 2090–2093.

[218] M. Van Dyke (1982) *An Album of Fluid Motion*. The Parabolic Press, Stanford, CA.

[219] M. W. Vance, K. D. Squires & O. Simonin (2006) Properties of the particle velocity field in gas-solid turbulent channel flow. *Phys. Fluids* **18** (063302).

[220] O. Vermorel, B. Bédat, O. Simonin & T. Poinsot (2003) Numerical study and modelling of turbulence modulation in a particle laden slab flow. *J. Turb.* **4, 025**.

[221] J.-F. Virepinte (1999) Étude du comportement dynamique et thermique de gouttes en régime d'interaction dans le cas de jets rectilignes. Phd thesis, École Nationale Supérieure de l'Aéronautique et de l'Espace.

[222] C. Walshaw & M. Cross (2007) JOSTLE: Parallel multilevel graph-partitioning software – an overview. In *Mesh Partitioning Techniques and Domain Decomposition Techniques* (ed. F. Magoules), pp. 27–58. Civil-Comp Ltd., (Invited chapter).

[223] Q. Wang & K. D. Squires (1996) Large eddy simulation of particle-laden turbulent channel flow. *Phys. Fluids* **8** (5), 1207–1223.

[224] W. Wang & R. Pletcher (1996) On the large eddy simulation of a turbulent channel flow with significant heat transfer. *Phys. Fluids* **8** (12), 3354–3366.

[225] J. Watts, M. Rieffel & S. Taylor (1997) A load balancing technique for multi-phase computations. In *Proc. of High Performance Computing '97*, pp. 15–20.

[226] M. R. Wells & D. E. Stock (1983) The effects of crossing trajectories on the dispersion of particles in a turbulent flow. *J. Fluid Mech.* **136**, 31–62.

[227] T. Westermann (1992) Localization schemes in 2D boundary-fitted grids. *J. Comput. Phys.* **101** (2), 307–313.

[228] J. H. Wilkinson (1964) *Rounding Errors in Algebraic Processes*. Englewood Cliffs, NJ: Prentice-Hall.

[229] R. D. Williams (1991) Performance of dynamic load balancing algorithms for unstructured mesh calculations. *Concurrency: Practice, and Experience* **3** (5), 451–481.

[230] Y. Yamamoto, M. Potthoff, T. Tanaka, T. Kajishima & Y. Tsuji (2001) Large-eddy simulation of turbulent gas-particle flow in a vertical channel: effect of considering inter-particle collisions. *J. Fluid Mech.* **442**, 303–334.

[231] H. Zha, X. He, C. Ding, M. Gu & H. Simon (2001) Bipartite graph partitioning and data clustering. In *Proc. of the 10th ACM International Conference on Information and Knowledge Management*, pp. 25–31.

Appendices

Appendix A

Suggestion of treatment of other Lagrangian boundary conditions

Figure A.1 (a) shows an example of hexahedron-based grid where the outlet, and the external and internal walls are highlighted. The idea proposed is to label in a particular way the cells of boundary conditions to check them when relocating particles. An example of this is presented in Figure A.1 (b) where BC cells are colored in grayscale. Cells of the outlet are in dark gray and cells of the different walls are in light gray.

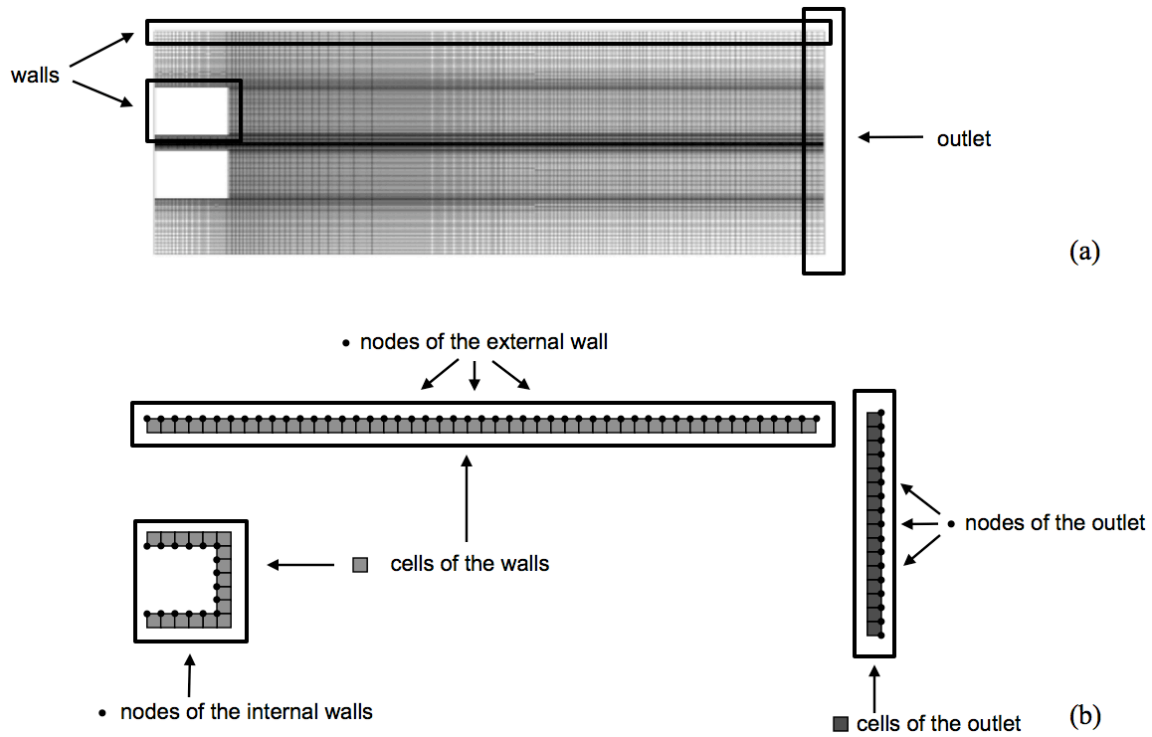


Figure A.1 - *(a) Mesh with outlet and wall positions highlighted; (b) illustration of the cells and nodes of the boundary conditions presented. Two gray scales are used to distinguish the outlet or wall cells.*

This procedure can be extended to any other boundary condition: symmetry, periodicity; or even to distinguish different kind of particle-wall treatments: rebound, films, splash, etc. For development purpose, integer values can be used: zero for cells which are not boundaries, one for outlets, two for walls, three for symmetry, and so on. This allows the use of only one array with dimension of the number of cell of each subdomain.

One advantage of this procedure is that it benefits from the existent data structure of AVBP. Firstly, all the nodes with a boundary condition (also called *patch*) are known at the beginning of the simulation and the cells associated to a particular patch could be flagged by using the same arrays and while recovering information of the patch. The array with the cells labeled could then be filled at this time, reducing memory and time requirements. Secondly, only one array is needed to store information about the particle-cell treatment. Then, the kind of treatment to apply to a particle is accessible with an *if* statement.

Another possibility is to label only the cells of the boundaries to reduce memory storage but this may increase time requirements since for every cell of the boundary and for every particle, a loop should be done to detect if the cell is flagged. In addition and focusing on massively parallel machines, the higher the number of partitions, the lower the number of cells associated to each subdomain and the lower the impact of the array proposed on the whole memory storage.

Appendix B

Proc. of the Summer Program 2006, pp. 197-211

Évaluation des méthodes numériques pour la simulation aux grandes échelles des écoulements diphasiques réactifs turbulents

La combustion turbulente est parfois effectuée avec des réactifs purement gazeux (dans les turbines à gaz industrielles par exemple) mais elle est aussi souvent basée sur l'emploi de réactifs liquides. En effet, pour des raisons de stockage et de manipulation, le carburant est en général injecté sous forme liquide avant d'être brûlé. L'aspect diphasique devient alors essentiel et parfois prédominant face à la combustion. La prédiction de la dispersion du carburant dans les écoulements diphasiques turbulents (la plupart comprenant des zones de recirculation) est donc une question clé dans le cadre de notre étude.

Les méthodes les plus classiques pour la description de la phase dispersée dans ce type d'écoulements sont les approches Euler/Euler (EE) et Euler/Lagrange (EL). Dans la première approche, la phase gazeuse et la phase liquide sont décrites sur le même maillage sur lequel on cherche à résoudre un lot d'équations différentielles partielles. Dans la seconde approche, le gaz est résolu comme toujours sur un maillage Eulérien alors que la phase liquide est simulée au moyen de particules Lagrangiennes. Dans les deux cas, le couplage est nécessaire et intervient au travers d'échanges d'information entre le système fluide et liquide. La plupart des codes RANS (pour Reynolds-averaged Navier-Stokes) utilise les approches EL. Cependant, l'histoire du développement du RANS a montré que les deux approches sont disponibles dans les codes commerciaux. Dans le cas de la simulation aux grandes échelles (LES pour Large-Eddy Simulation) - qui permet une analyse plus complète des phénomènes instables (instabilités, flashback or quenching) à l'intérieur des chambres de combustion - les deux formulations ont été développées dans le but de comparer les résultats numériques dans un cas test de référence disposant d'un ensemble complet de données expérimentales pour un calcul gazeux et un calcul diphasique.

Dans cette étude, le cas test choisi pour la comparaison des deux approches avec la stratégie LES est décrit dans Borée *et al.* [22]. Dans cette configuration, un jet d'air avec des particules de verre est injecté dans la chambre par le biais d'un tube d'injection entouré d'un co-courant d'air. Le rapport de vitesse entre le tube d'injection et le co-courant a été choisi de manière à créer une large zone centrale de recirculation (entre le jet central et le co-courant). Ces tests sont effectués

sans combustion ni évaporation, mais sont obligatoires avant de réaliser ce type de calculs dans des géométries complexes proches de celles des chambres de combustion. Le diamètre des particules de verre est compris entre 20 et 100 microns et leur masse volumique est de 2470 kg/m^3. Cette configuration dispose de deux bases de données complètes pour un cas faiblement chargé en masse ($M_j = 22\%$) et un autre fortement chargé en masse ($M_j = 110\%$). Les comparaisons présentées dans cet article se limitent au calcul faiblement chargé où l'écoulement diphasique consideré est suffisamment dilué pour négliger les collisions inter-particulaires. Les simulations numériques ont été réalisées uniquement avec des particules de 60 microns. Des études non rapportées ici ont montré qu'un calcul monodispersé avec la taille moyenne des particules est suffisant pour capturer l'effet moyen de la phase gazeuse et la dynamique des particules.

Deux codes de calcul différents ont également été testés sur la configuration de Borée *et al.* [22].

- Le solveur implicite CDP (développé à l'Université de Stanford) résout les équations de Navier-Stokes de façon incompressible. L'avancement en temps de CDP est basé sur la méthode de pas de temps fractionné (fractional-step method) et l'avancement en espace est fait par un schéma centré à l'ordre 2. Le modèle de sous-maille utilisé est celui de Smagorinsky dynamique (Germano *et al.* [72]).
- Le code de calcul explicite AVBP (développé au centre de recherche : CERFACS) résout les équations de Navier-Stokes de façon compressible. Le schéma utilisé dans cette étude est à l'ordre 3 en espace avec un avancement en temps de type Runge-Kutta. Le modèle de sous-maille est celui de Smagorinsky et les conditions limites sont du type NSCBC (Poinsot & Veynante [165], Moureau *et al.* [138]).

Les points qui sont étudiés avec la LES de cet écoulement diphasique sont les suivants :

- Comparaison des performances et des temps de calcul des approches EE et EL.
- Évaluation de l'influence du type des cellules du maillage (hexaèdres vs tétraèdres)
- Comparaison des formulations implicite et explicite pour l'avancement en temps.
- Étude des effets des conditions limites sur la phase dispersée.

Après une description détaillée des équations de la phase dispersée dans les deux approches, les résultats des calculs gazeux et diphasique sont présentés en les comparant aux résultats expérimentaux sur sept profils radiaux de vitesses moyenne et fluctuante.

Les résultats montrent que l'écoulement est bien prédit par les deux solveurs. La phase dispersée est bien prédite avec les deux approches, mais la formulation Lagrangienne prédit les valeurs fluctuantes (RMS) plus précisément. En fait, la formulation Eulérienne montre une sous-estimation de l'agitation des particules, et de leur dispersion radiale, qui semble liée au fait que dans cet étude l'effet de l'agitation décorrélée (RUV) de la phase dispersée ne soit pas prise en compte. L'importance des conditions limites d'entrée pour le gaz et la phase dispersée apparait à travers différents cas tests. Dans ce cas faiblement chargé où le nombre de particules reste faible pas rapport au nombre de cellules, l'approche Lagrangienne reste moins coûteuse (en temps et en mémoire) que l'approche Eulérienne.

Evaluation of numerical strategies for LES of two-phase reacting flows

By E. Riber∗, †, M. García∗, V. Moureau, H. Pitsch, O. Simonin† AND T. Poinsot†

Predicting particle dispersion in recirculating two-phase flows is a key issue for reacting flows. In this study, Euler/Euler and Euler/Lagrange LES formulations have been compared in the bluff-body configuration from Borée *et al.* (2001) where glass beads are injected into a complex recirculating flow. These tests are performed for non-reacting, non-evaporating sprays but are mandatory validations before computing realistic combustion chambers. Two different solvers (one explicit and compressible and the other implicit and incompressible) have also been tested on the same configuration. Results show that the gas flow is well predicted by both solvers. The dispersed phase is also well predicted but the Lagrange solver predicts RMS values more precisely. The importance of inlet boundary conditions for the gas and the dispersed phase is revealed through various tests.

1. Motivations and objectives

Today, RANS (Reynolds-averaged Navier-Stokes) equations are routinely solved to design combustion chambers, for both gaseous and liquid fuels. Recently, in order to provide better accuracy for the prediction of mean flows but also to give access to unsteady phenomena occurring in combustion devices (such as instabilities, flashback or quenching), Large-Eddy Simulation (LES) has been extended to reacting flows. The success of these approaches for gaseous flames in the last years (Caraeni *et al.* 2000; Chakravarthy & Menon 2000; Colin *et al.* 2000; Forkel & Janicka 2000; Pitsch & Duchamp de la Geneste 2002; Mahesh *et al.* 2004; Selle *et al.* 2004; Sommerer *et al.* 2004; Moureau *et al.* 2005; Roux *et al.* 2005; Poinsot & Veynante 2005) is a clear illustration of their potential. LES gives access to the large scales structures of the flow reducing the importance of modeling, and naturally capturing a significant part of the physics controlling these flames. Even though LES has already demonstrated its potential for gaseous flames, its extension to two-phase flames is still largely to be done. First, the physical submodels required to describe the atomization of a liquid fuel jet, the dispersion of solid particles, their interaction with walls, evaporation and combustion are as difficult to build in LES as in RANS because they are essentially subgrid phenomena. Second, the numerical implementation of two-phase flow LES remains a challenge. The equations for both the gaseous and the dispersed phases must be solved together at each time step in a strongly coupled manner. This differs from classical RANS where the resolution of the two phases can be done in a weak procedure, bringing first the gas flow to convergence, then the solid particles and finally iterating until convergence of both phases. Finally, in the context of parallel super-computing, numerical efficiency is an additional constraint. For single-phase flows, efficient and accurate solvers have been developed and speedups of

† IMF Toulouse, UMR CNRS/INPT/UPS, 31400 Toulouse, France
∗ CERFACS, CFD team, 42 Av. Gaspard Coriolis, 31057 Toulouse, France

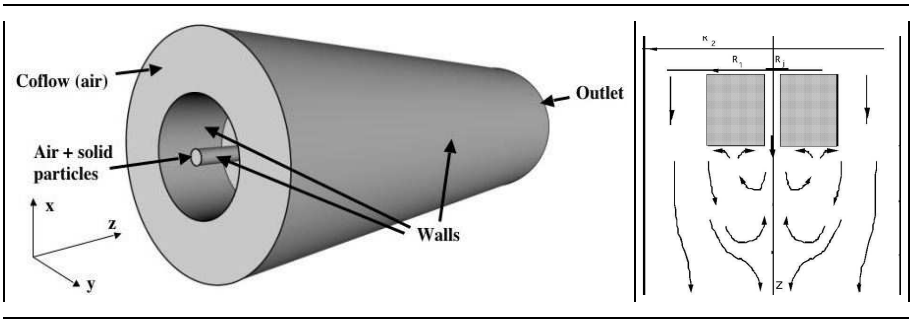


FIGURE 1. Configuration of Borée (Borée *et al.* 2001). The dimensions are : $R_j = 10$ mm, $R_1 = 75$ mm, $R_2 = 150$ mm. The total length of the experiment is 1.5 m.

the order of 5000 are not uncommon (www.cerfacs.fr/cfd/parallel.html). Maintaining a similar parallel efficiency for a two-phase flow solver while representing the main physics of the flow raises additional questions.

In LES of two-phase flows, physics and numerics interact strongly: the first question is to choose a paradigm to describe the two-phase flow. Most RANS codes use Euler/Lagrange (EL) methods in which the flow is solved using an Eulerian method and the particles are tracked using a Lagrangian approach. An alternative technique is to use two-fluid models in which both the gas and the dispersed phases are solved using an Eulerian method (Euler/Euler or EE). The history of RANS development has shown that both EE and EL are useful and either is found today in most commercial codes. For LES, both EE and EL formulations are being developed and the focus of this study is to test them in a reference case where complete sets of solutions for gas and dispersed phase are available. This exercise is performed here without evaporation or combustion.

2. Configuration and work objectives

In this study two solvers developed at CTR and CERFACS are used to investigate some critical issues for LES of two-phase flows on massively parallel computers:

- an implicit incompressible solver (CDP) using a EL formulation. For this study, hexahedron-based grids are used in CDP.
- an explicit compressible solver (AVBP) using both EL and EE formulations (Kaufmann *et al.* 2003). For the present study, AVBP is used on tetrahedron-based grids.

These solvers are used to study a bluff-body configuration (Borée *et al.* 2001) where a jet of air and solid particles are injected in a coflow of air (Fig. 1). The jet velocity on the axis is 4 m/s and the coflow maximum velocity is 6 m/s. The experiment is designed to provide large recirculation zones between the central jet and the coflow. The dispersed phase consists of solid particles (glass beads with diameter ranging from 20 to 100 microns with a mean value of 60 microns) so that evaporation, coalescence and break up do not have to be considered. The material density of the glass particle is $\rho_p = 2470$ kg.m^{-3}. The mass loading ratio of particles in the inner jet is 0.22 corresponding to a solid volumetric fraction less than 10^{-4}. Thus collision effects will be assumed to be negligible in the modelling approaches.

The issues which can be studied are still very relevant for LES of two-phase flows:

- Compare performances and cost of EE and EL approaches.

- Evaluate influence of mesh type (hexahedra vs tetrahedra).
- Compare implicit and explicit formulations for time advancement.
- Study effects of boundary conditions for the dispersed phase: should the particle velocities at the domain inlet be modulated to account for turbulence or not?

3. Description of solvers and models

Numerical methods used in both LES solvers for the gas phase have been extensively described in the literature (Moureau *et al.* 2005; Selle *et al.* 2004; Schmitt *et al.* 2006; Mahesh *et al.* 2004; Ham & Iaccarino 2004) and will only be summarized here.
The LES solver CDP solves implicitly the incompressible Navier-Stokes equations. The time integration of CDP is based on the fractional-step method (Kim & Moin 1985) and the space integration relies on a second-order central scheme which conserves the kinetic energy (Mahesh *et al.* 2004; Ham & Iaccarino 2004). The dynamic Smagorinsky model (Germano *et al.* 1991) is used to model the subgrid stress tensor.
The explicit LES solver AVBP solves the compressible Navier-Stokes equations with a third-order scheme for spatial differencing and a Runge-Kutta time advancement (Colin & Rudgyard 2000; Moureau *et al.* 2005). For the present case, the Smagorinsky model is used to model SGS tensors. Walls are treated using the law-of-the-wall formulation of Schmitt *et al.* (2006). The boundary conditions are handled with the NSCBC formulation (Poinsot & Veynante 2005; Moureau *et al.* 2005).
The influence of the particles on the gas phase is taken into account in the EL simulations by using the point-force approximation in the general framework of the particle-in-cell method (PIC) (Boivin *et al.* 1998; Vermorel *et al.* 2003), with standard single-phase subgrid turbulence modelling approaches. According to Boivin *et al.* (2000), such an assumption is valid for small mass loading ratio of particles (typically, $\alpha_p \rho_p / \rho_g \leq 1$) with response time larger than the subgrid turbulence characteristic time scale.
The influence of the particles on the gas phase is taken into account through the drag force in the EE simulations. Modification of the gas subgrid-scale turbulence model by the particles is neglected.
This section focuses on techniques used for the dispersed phase.

Euler/Lagrange approach

The dispersed phase consists of particles which are assumed to be rigid spheres with diameter comparable or smaller than the Kolmogorov length scale. If the particle density is much larger than the fluid density, the forces acting on particles reduce to drag and gravity. With these assumptions, the particle equations of motion can then be written for a single particle as:

$$\frac{dx_{p,i}}{dt} = u_{p,i} \tag{3.1}$$

$$\frac{du_{p,i}}{dt} = -\frac{3}{4}\,\frac{\rho_g}{\rho_p}\,\frac{C_D}{d_p}\,|\mathbf{v}_r|\;\mathrm{v}_{r,i} + g_i = -\frac{u_{p,i} - \tilde{u}_{g,i}}{\tau_p} + g_i\ . \tag{3.2}$$

The local drag coefficient in Eq. (3.2) is C_D and may be expressed in terms of the particle Reynolds number Re_p following Schiller & Nauman (1935):

$$C_D = \frac{24}{Re_p}\left[1 + 0.15 Re_p^{0.687}\right], \qquad Re_p = \frac{|\mathbf{v}_r|\;d_p}{\nu_g} \leq 800\ . \tag{3.3}$$

The local instantaneous relative velocity between the particle and the surrounding fluid is $\mathrm{v}_{r,i} = u_{p,i} - \tilde{u}_{g,i}$ where g_i is the gravity vector and $\tilde{u}_{g,i}$ is the filtered fluid velocity at the position of the particle assuming that the flow field is locally undisturbed by the presence of this particle (Gatignol 1983; Maxey & Riley 1983) and that the subgrid fluid velocity seen by the particles is negligible (Fede *et al.* 2006). The particle relaxation time τ_p is defined as the Stokes characteristic time:

$$\tau_p = \frac{4}{3}\,\frac{\rho_p}{\rho_g}\,\frac{d_p}{C_D}\,|\mathbf{v}_r| \tag{3.4}$$

where d_p is the particle diameter, ρ_p is the density of the particle, ν_g is the kinematic viscosity of the fluid at the particle location.

Euler/Euler approach

The treatment of the dispersed phase is based on an Eulerian approach: Eulerian equations for the dispersed phase may be derived by several means. A popular and simple way consists in volume filtering of the separate, local, instantaneous phase equations accounting for the inter-facial jump conditions (Druzhinin & Elghobashi 1999). Such an averaging approach is restrictive because particle sizes and particle distances have to be smaller than the smallest length scale of the turbulence. Besides, they do not account for the Random Uncorrelated Motion (Février *et al.* 2005). In the present study, a statistical approach analogous to kinetic theory (Chapman & Cowling 1939) is used to construct a probability density function (pdf) $\breve{f}_p(\mathbf{c}_p, \mathbf{x}, t)$ which gives the local instantaneous probable number of particles with the given translation velocity $\mathbf{u}_p = \mathbf{c}_p$. The resulting model (Février *et al.* 2005; Moreau *et al.* 2005) leads to equations for the particle number density $\bar{n}_p$ and the correlated velocity $\hat{\mathbf{u}}_p$:

$$\frac{\partial}{\partial t}\bar{n}_p + \frac{\partial}{\partial x_j}\bar{n}_p\hat{u}_{p,j} = 0 \tag{3.5}$$

$$\begin{aligned}\frac{\partial}{\partial t}\bar{n}_p\hat{u}_{p,i} + \frac{\partial}{\partial x_j}\bar{n}_p\hat{u}_{p,i}\hat{u}_{p,j} &= -\frac{\bar{n}_p}{\tau_p}\left(\hat{u}_{p,i} - \hat{u}_{f,i}\right) + \bar{n}_p g_i - \frac{\partial}{\partial x_j}T_{p,ij} \\ &\quad -\frac{\partial}{\partial x_j}\bar{n}_p\widehat{\delta R}^*_{p,ij} - \frac{\partial}{\partial x_i}\frac{2}{3}\bar{n}_p\widehat{\delta\theta}_p\end{aligned} \tag{3.6}$$

where $\bar{n}_p$, $\hat{\mathbf{u}}_p$ and $\widehat{\delta\theta}_p$ are respectively the filtered particle number density, correlated velocity and Random Uncorrelated Energy (RUE). The two first terms of the rhs of Eq. (3.6) are the drag force and gravity effects on large scales, the third one accounts for the SGS effects, the fourth one takes into account the RUE effects and the last one is a dissipation term by RUE. $T_{p,ij}$ stands for the particle subgrid stress tensor:

$$T_{p,ij} = \bar{n}_p(\widehat{u_{p,i}u_{p,j}} - \hat{u}_{p,i}\hat{u}_{p,j}) \tag{3.7}$$

As in fluid anisotherm turbulence, an additional equation on energy is needed. The transport equation of filtered RUE is:

$$\begin{aligned}\frac{\partial}{\partial t}\overline{n}_p\widehat{\delta\theta}_p + \frac{\partial}{\partial x_j}\overline{n}_p\hat{u}_{p,j}\widehat{\delta\theta}_p &= -2\frac{\overline{n}_p}{\tau_p}\widehat{\delta\theta}_p - \frac{2}{3}\overline{n}_p\widehat{\delta\theta}_p\frac{\partial\hat{u}_{p,j}}{\partial x_j} \\ -\overline{n}_p\widehat{\delta R}^*_{p,ij}\frac{\partial\hat{u}_{p,i}}{\partial x_j} &- \frac{1}{2}\frac{\partial}{\partial x_j}\overline{n}_p\widehat{\delta S}_{p,iij} + \Pi_{\delta\theta_p} - \frac{\partial}{\partial x_j}Q_{p,j}\end{aligned} \tag{3.8}$$

The first rhs term is the RUE destruction by drag force, the second one is a RUE-dilatation term, the third one is a production term by filtered Random Uncorrelated Velocity (RUV) tensor, the next one is the diffusion by filtered RUV third correlation tensor. $\Pi_{\delta\theta_p}$ and $Q_{p,j}$ are respectively production and diffusion terms by subgrid scales:

$$\Pi_{\delta\theta_p} = \left(\overline{\breve{n}_p \delta R_{p,ij} \frac{\partial \breve{u}_{p,i}}{\partial x_j}} - \bar{n}_p \delta \widehat{R_{p,ij}} \frac{\partial \hat{u}_{p,i}}{\partial x_j}\right) \quad \text{and} \quad Q_{p,i} = \bar{n}_p \left(\widehat{u_{p,i}\delta\theta_p} - \hat{u}_{p,i}\widehat{\delta\theta}_p\right) \tag{3.9}$$

Closure of filtered RUV terms

Using an equilibrium assumption, Kaufmann (2004) model $\delta R^*_{p,ij}$ by a viscous term and $\delta S_{p,iij}$ by a diffusive term similar to Fick's law. For LES approach these models are adapted by replacing non filtered quantities by filtered ones leading to (Moreau *et al.* 2005):

$$\widehat{\delta R}^*_{p,ij} = -\hat{\nu}_{RUM}\left(\frac{\partial \hat{u}_{p,i}}{\partial x_j} + \frac{\partial \hat{u}_{p,j}}{\partial x_i} - \frac{\partial \hat{u}_{p,k}}{\partial x_k}\frac{\delta_{ij}}{3}\right) \quad \text{and} \quad \frac{1}{2}\widehat{\delta S}_{p,iij} = -\hat{\kappa}_{RUM}\frac{\partial \widehat{\delta\theta}_p}{\partial x_j} \tag{3.10}$$

where the RUM viscosity, $\hat{\nu}_{RUM}$, and the RUM diffusion coefficient, $\hat{\kappa}_{RUM}$, are given by:

$$\hat{\nu}_{RUM} = \frac{\tau_p}{3}\widehat{\delta\theta}_p \quad \text{and} \quad \hat{\kappa}_{RUM} = \frac{10}{27}\tau_p\widehat{\delta\theta}_p \tag{3.11}$$

Subgrid terms modeling

By analogy to single phase flows (Moin *et al.* 1991; Vreman *et al.* 1995), Riber *et al.* (2005) propose a viscosity model for the SGS tensor $T_{p,ij}$. The trace-free SGS tensor is modeled using a viscosity assumption (compressible Smagorinsky model), while the subgrid energy is parametrized by a Yoshizawa model (Yoshizawa 1986):

$$T_{p,ij} = -C_S 2\Delta_f^2 \bar{n}_p |\hat{S}_p|\left(\hat{S}_{p,ij} - \frac{\delta_{ij}}{3}\hat{S}_{p,kk}\right) + C_I 2\Delta_f^2 \bar{n}_p |\hat{S}_p|^2 \delta_{ij} \tag{3.12}$$

where $\hat{S}_p$ is the filtered particle strain rate tensor, $|\hat{S}_p|^2 = 2S_{p,ij}S_{p,ij}$ and Δ_f the filter characteristic length. The model constants have been evaluated in a priori tests (Riber *et al.* 2006) leading to the values $C_S = 0.02$, $C_I = 0.012$.

The subgrid diffusion term in the filtered RUE is modeled by an eddy-diffusivity model:

$$Q_{p,j} = -\frac{\bar{n}_p C_S 2\Delta_f^2 |\hat{S}_p|}{P_{r,p}^{SGS}}\frac{\partial \widehat{\delta\theta}_p}{\partial x_j} \tag{3.13}$$

with the particle turbulent Prandtl number $P_{r,p}^{SGS} = 0.8$. The subgrid production of filtered RUE term $\Pi_{\delta\theta_p}$ acts like a dissipation term in the subgrid energy equation. Using an equilibrium assumption on the particle correlated subgrid energy and neglecting diffusion terms leads to:

$$-\frac{\bar{n}_p}{\tau_p}\left(\frac{T_{p,kk}}{\bar{n}_p} - q_{fp,SGS}\right) + \Pi_{\delta\theta_p} - T_{p,ij}\frac{\partial \hat{u}_{p,i}}{\partial x_j} = 0 \tag{3.14}$$

where the subgrid covariance is $q_{fp,SGS} = \widehat{u_{p,k}u_{f,k}} - \hat{u}_{p,k}\hat{u}_{f,k}$. To first order, the drag force term can be neglected and $\Pi_{\delta\theta_p}$ can be modeled by: $\Pi_{\delta\theta_p} \approx T_{p,ij}\partial\hat{u}_{p,i}/\partial x_j$ with the SGS tensor modeled by Eq. (3.12). This model ensures that the correlated energy dissipated by subgrid effects is fully transfered into RUE.

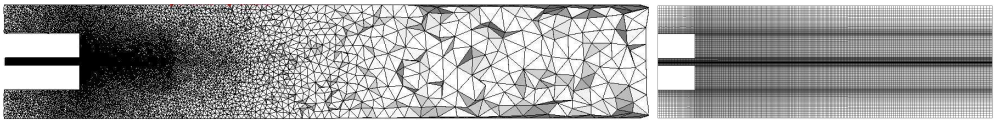


FIGURE 2. Grids used by AVBP (Tetrahedra, left) and CDP (Hexahedra, right).

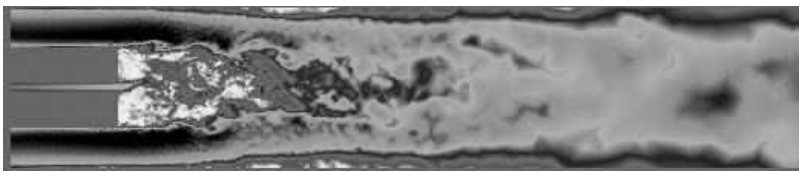


FIGURE 3. Instantaneous field of velocity modulus (AVBP). Maximum value (black): 6 m/s. Minimum value: 0.

	CDP	AVBP
Grid type	Hexahedra	Tetrahedra
Number of cells / nodes	3207960 / 3437576	2058883 / 367313
Time step (microseconds) / CFL	147 / 50	3,2 / 0.7
Averaging time (s) / Iterations	2,65 / 18000	1,03 / 320000
LES model / Wall model	Dynamic Smagorinsky/None	Smagorinsky/Law-of-the-wall

TABLE 1. Summary of parameters and models used for the 'no-particles' computation.

4. Comparison of gas flow without particles

Before discussing results for the dispersed phase, the accuracy of the LES solvers for the gas phase is evaluated by computing the flow without particles and comparing it to the same data provided in Borée *et al.* (2001). The two codes (AVBP and CDP) are used on two different grids (Fig. 2) (see summary in Table 1). A typical snapshot of the velocity field (modulus) in the central plane is displayed in Fig. 3 for an AVBP result (CDP fields are very similar). The complex structure of the recirculating flow is obvious: on the axis, the flow is recirculating down to $z = 200$ mm. On the sides of the channel, the flow also seems to separate from $z = 50$ mm to $z = 400$ mm.

Figures 4 to 7 present the results of the two LES codes along with the experimental measurements. The two LES solvers capture most of the flow physics: the axial mean and RMS velocities (Fig. 4 and 5) agree with the LDV measurements. The length of the recirculation zone (evidenced by the negative values of axial velocities on the axis) is well predicted. In the coflow, the RMS values predicted by LES are too low because no turbulence is injected at the inlet of the domain for these computations.

The mean radial velocity levels (Fig. 6) remain small (less than 1 m/s) and the two LES codes capture the radial velocity fields correctly except at the corner of the coflow and the step (first station at $z = 3$ mm) where the RMS velocities are underpredicted by CDP (Fig. 7). The stagnation point (around $z = 170$ mm) is a delicate zone where both

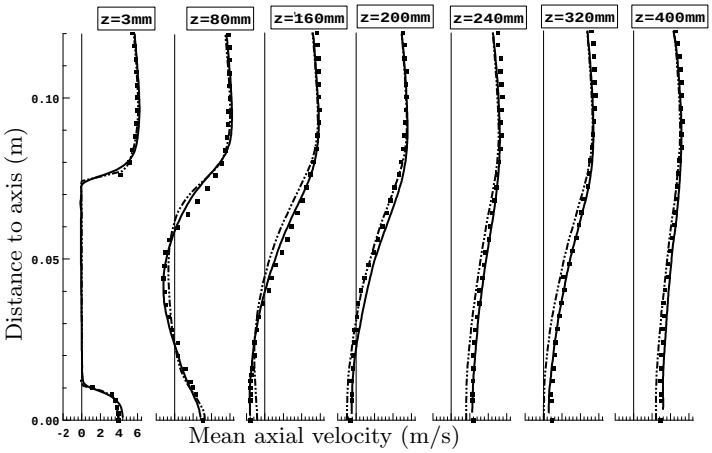


FIGURE 4. Radial profiles of mean axial gas velocities at 7 stations along z axis. Symbols: experiment; solid line: AVBP; dot-dashed line: CDP.

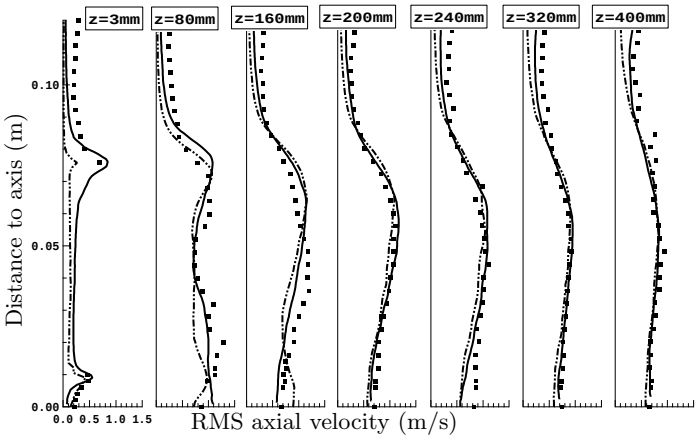


FIGURE 5. Radial profiles of RMS axial gas velocities at 7 stations along z axis. Symbols: experiment; solid line: AVBP; dot-dashed line: CDP.

codes have difficulties. The source of this problem is the exact position of the stagnation point: any small mismatch in this position leads to large changes in profiles measured around this point. Upstream and downstream of this point, the agreement is very good. The overall result is that both codes provide similar results even though they use totally different grids and methods. This indicates that grid independence for the gas is achieved for this test case and that tests for the dispersed phase can be performed with reasonable confidence.

5. Results for two-phase flow cases

This section presents results for the 22 percent mass loading of the central jet, obtained with three different computations summarized in Table 2. The grids and the time steps

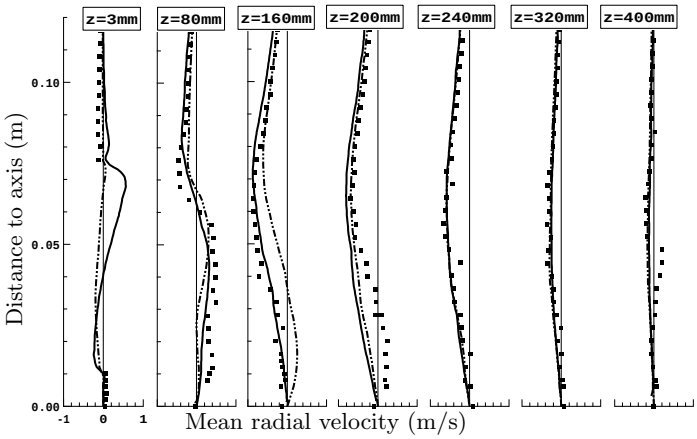


FIGURE 6. Radial profiles of mean radial gas velocities at 7 stations along z axis. Symbols: experiment; solid line: AVBP; dot-dashed line: CDP.

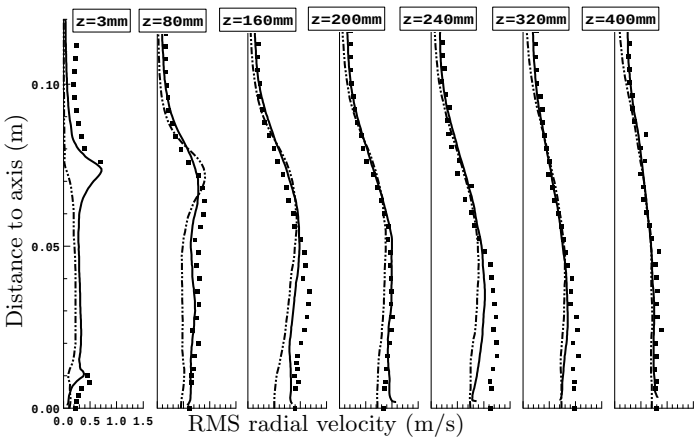


FIGURE 7. Radial profiles of RMS radial gas velocities at 7 stations along z axis. Symbols: experiment; solid line: AVBP; dot-dashed line: CDP.

used in AVBP and CDP are the same as in Table 1†. In all computations presented here, the injected particles have a size of 60 microns. Separated studies which are not reported here, using the Lagrangian solver and multidisperse particles or 60 microns particles only have shown that using a monodisperse distribution of size was very close to the 22 percent case of Borée *et al.* (2001) and was sufficient to capture both the mean flow effects on the gas (through two-way coupling) and the dynamics of the 60 microns class.

An essential part of these LES is the introduction of the particles in terms of position and velocity (Fig. 8). The injection planes are not the same for all codes. The methodologies used to inject the particles are also different to evaluate their impact on results. In AVBP-EE, both the mass loading and the mean velocity imposed in the injection plane

† For these runs, the RUM model is not used and the $\widehat{\delta\theta}_p$ term in Eq. (3.6) is set to zero.

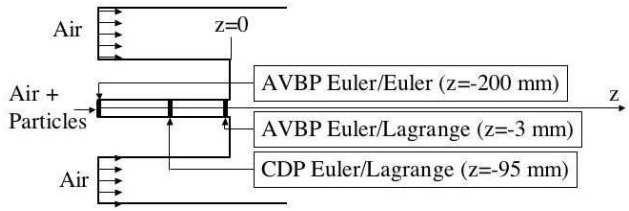


FIGURE 8. Injection position for particles.

	CDP Euler/Lagrange	AVBP Euler/Lagrange	AVBP Euler/Euler
Averaging time (s)	4	0.43	0.64
Particle mean speed	Exp. profile	Exp. profile	Exp. profile
Turbulent fluctuations	White noise (10 %)	White noise (12 %)	Zero
Particle distribution	Homogeneous	Homogeneous	Exp. profile

TABLE 2. Summary of parameters and models used for the particle injection (22 percent mass loading computation). The particles are injected in the central tube.

($z = -200$ mm) are the ones measured experimentally at $z = 3$ mm. No turbulent fluctuations are introduced. In AVBP-EL and in CDP, the mass loading is homogeneous over the injection section and the injection speed profile is also the experimental one measured at $z = 3$ mm. In AVBP-EL and in CDP, a white noise (amplitude of the order of 10 percent of the mean velocity) is added to the particle mean velocity profiles to match experimental measurements at $z = 3$ mm.

The velocity fields for the gas phase change when the particles are injected but these effects are limited and are not discussed here. Figures 9 to 12 show velocity fields for particles obtained with the three codes along with the measurements of Borée. The agreement between the experiments and the three LES sets of data is good. An interesting result is that AVBP-EL (dashed line) and AVBP-EE (solid line) provide extremely similar results showing that the EE approach is able to compute such a flow and to provide results which are equivalent in precision to an EL computation.

The best results are obtained with CDP and injection of turbulence on the gas phase. A convenient way to look at the results is to consider the central z axis of the configuration: a critical zone is located around $z = 160$ mm where the stagnation point for the gas is. This is also a zone where particles accumulate and must stop before turning around to escape from the recirculating flows by the sides. Figure 13 shows fields of gas velocity and of local volume fraction of solid particles for AVBP-EE on the left and CDP on the right side. Both solvers capture the zone where the solid particles accumulate. Local droplet accumulation is also observed upstream of the stagnation point within the central jet. However, the EE computation presented in Fig. 13 shows a droplet flow which stops slightly before the EL computation.

This can be quantified by plotting mean velocities along the axis for the gas (Fig. 14) and for the solid particles (Fig. 15). On this axis, the results provided by CDP are excellent while the two AVBP results match but are slightly off the experimental results.

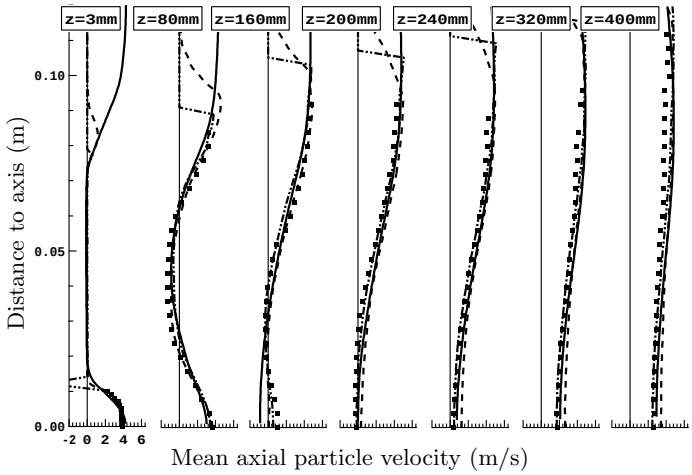


FIGURE 9. Radial profiles of mean axial particle velocities at 7 stations along z axis. Symbols: experiment; solid line: AVBP-EE; dashed line: AVBP-EL; dot-dashed line: CDP.

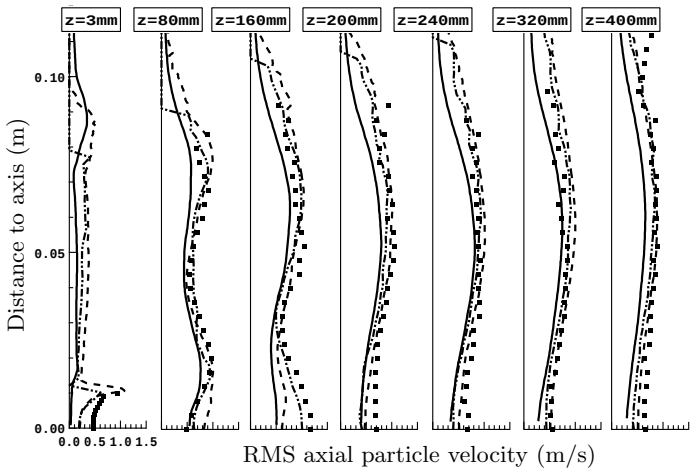


FIGURE 10. Radial profiles of RMS axial particle velocities at 7 stations along z axis. Symbols: experiment; solid line: AVBP-EE; dashed line: AVBP-EL; dot-dashed line: CDP.

The cause of this discrepancy was investigated through various tests during the project and was identified as the absence of turbulence injected on the gas phase in the central duct in AVBP: a direct verification of this effect is that in the two AVBP computations (solid and dashed lines), the gas and the particle velocities in the central duct increase between $z = -200$ and $z = 0$ mm, indicating that the flow is relaminarizing. This also demonstrates the importance of injecting not only the proper mean profile for the gas velocity but also fluctuations with a reasonably well-defined turbulent spectrum as done in CDP. Additional tests also reveal that the injection of white noise on the particle velocities has a very limited effect on the results.

Figures 16 and 17 display axial profiles of RMS velocities for the gas and the particles.

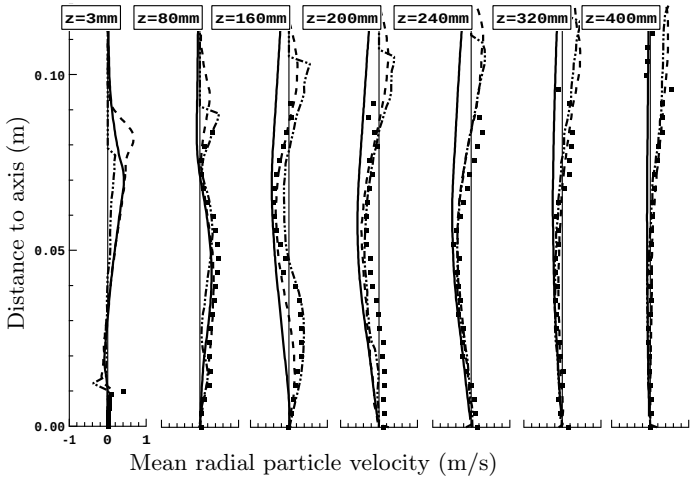


FIGURE 11. Radial profiles of mean radial particle velocities at 7 stations along z axis. Symbols: experiment; solid line: AVBP-EE; dashed line: AVBP-EL; dot-dashed line: CDP.

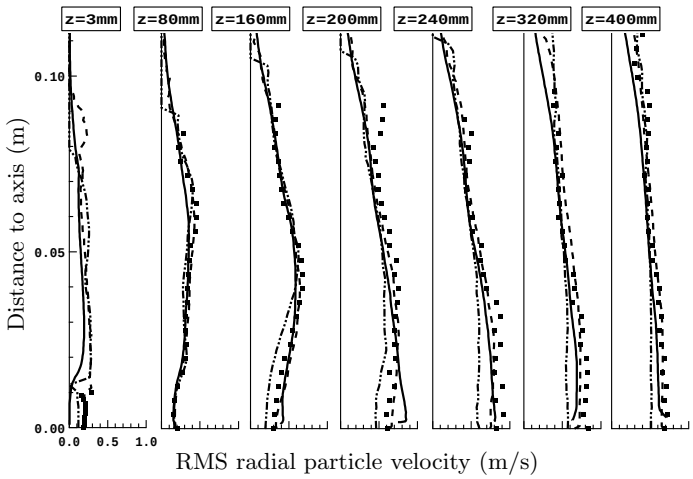


FIGURE 12. Radial profiles of RMS radial particle velocities at 7 stations along z axis. Symbols: experiment; solid line: AVBP-EE; dashed line: AVBP-EL; dot-dashed line: CDP.

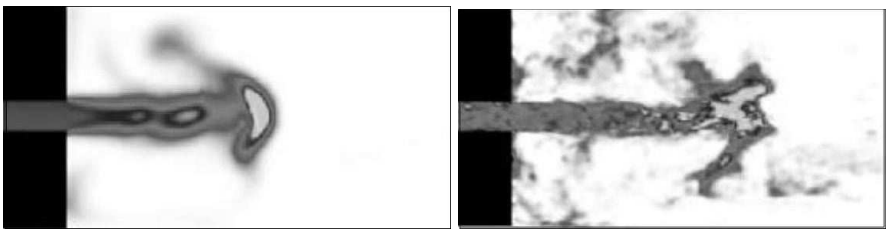


FIGURE 13. Instantaneous volume fraction in the central plane. Maximum value (black): 0.0002. Minimum value (white): 0. Right: values obtained from averaging the Lagrangian simulation in CDP. Left: output from AVBP-EE.

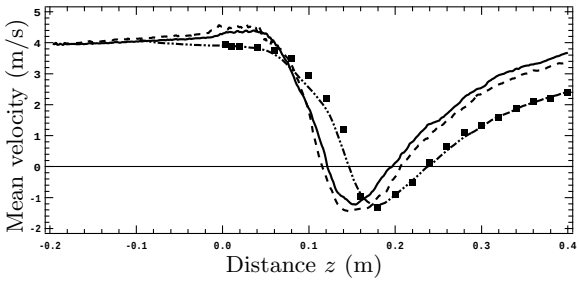


FIGURE 14. Axial profiles of mean gas velocities. Symbols: experiment; solid line: AVBP-EE; dashed line: AVBP-EL; dot-dashed line: CDP.

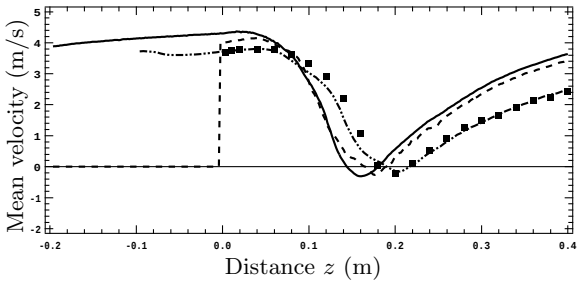


FIGURE 15. Axial profiles of mean particle velocities. Symbols: experiment; solid line: AVBP-EE; dashed line: AVBP-EL; dot-dashed line: CDP.

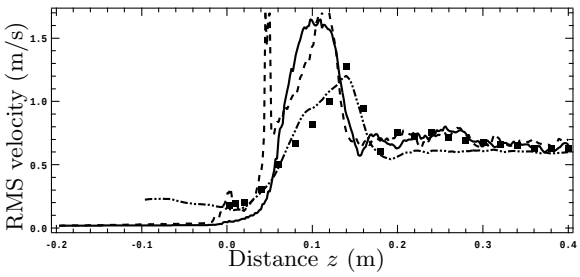


FIGURE 16. Axial profiles of RMS gas axial velocities. Symbols: experiment; solid line: AVBP-EE; dashed line: AVBP-EL; dot-dashed line: CDP.

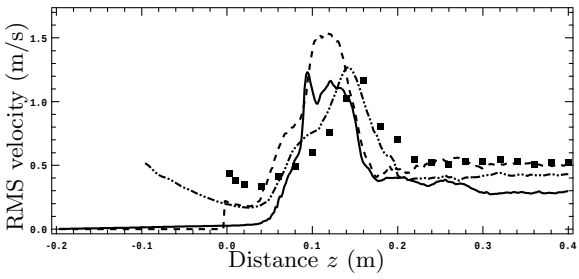


FIGURE 17. Axial profiles of RMS particle axial velocities. Symbols: experiment; solid line: AVBP-EE; dashed line: AVBP-EL; dot-dashed line: CDP.

These plots confirm that the position where the maximum levels of gas and particle turbulence are found on the axis is shifted towards the jet inlet and is too intense for both AVBP computations.

6. Performances and conclusions

For the present test case (mass loading of 22 percent), the total number of particles present in the domain for the Lagrange codes is of the order of 600000. For such a small number of particles, the computing power required by the Lagrangian solvers compared to the power required for the gas flow remains low: the added cost due to the particles is small and no load balancing problem is observed. The EE formulation added cost (of the order of 80 percent) is independent of the mass loading so that, for the present problem, the EL formulations proved to be faster.

In terms of results quality, the EL and the EE results with AVBP are very close showing that both formulations lead to equivalent results in this situation. An important factor controlling the quality of the results is the introduction of turbulence on the gas flow in the injection duct: without these turbulent fluctuations, the results are not as good on the axis in terms of positions of the recirculation zones.

Acknowledgment

The help of J. Borée in providing the experimental data and analyzing the results is gratefully acknowledged.

REFERENCES

Boivin, M., Simonin, O. & Squires, K. 2000 On the prediction of gas-solid flows with two-way coupling using large eddy simulation. *Phys. Fluids* **12** (8).

Boivin, M., Simonin, O. & Squires, K. 1998 Direct numerical simulation of turbulence modulation by particles in isotropic turbulence. *J. Fluid. Mech.* **375**, 235-263.

Borée, J., Ishima, T. & Flour, I. 2001 The effect of mass loading and inter-particle collisions on the development of the polydispersed two-phase flow downstream of a confined bluff body. *J. Fluid Mech.* **443**, 129-165.

Caraeni, D., Bergstrom, C. & Fuchs, L. 2000 Modeling of liquid fuel injection, evaporation and mixing in a gas turbine burner using large eddy simulation. *Flow Turb. and Combustion* **65**, 223-244.

Chakravarthy, V. & Menon, S. 2000 Subgrid modeling of turbulent premixed flames in the flamelet regime. *Flow Turb. and Combustion* **65**, 133-161.

Chapman, S. & Cowling, T. 1939 (digital reprint 1999) *The Mathematical Theory of Non-Uniform Gases*. Cambridge Mathematical Library Edn. Cambridge University Press.

Colin, O., Ducros, F., Veynante, D. & Poinsot, T. 2000 A thickened flame model for large eddy simulations of turbulent premixed combustion. *Phys. Fluids* **12** (7), 1843-1863.

Colin, O. & Rudgyard, M. 2000 Development of high-order Taylor-Galerkin schemes for unsteady calculations. *J. Comp. Phys.* **162** (2), 338-371.

Druzhinin, O.A. & Elghobashi, S. 1999 On the decay rate of isotropic turbulence laden with microparticles. *Phys. Fluids* **11** (3), 602-610.

Fede, P., Villedieu, P., Simonin, O. & Squires, K. 2006 Stochastic modeling of the subgrid fluid velocity fluctuations seen by inertial particles. In *Proc of the Summer Program*. Center for Turbulence Research, NASA Ames/Stanford Univ.

Février, P., Simonin, O. & Squires, K. 2005 Partitioning of particle velocities in gas-solid turbulent flows into a continuous field and a spatially uncorrelated random distribution: Theoretical formalism and numerical study. *J. Fluid Mech.* **533**, 1-46.

Forkel, H. & Janicka, J. 2000 Large-eddy simulation of a turbulent hydrogen diffusion flame. *Flow Turb. and Combustion* **65** (2), 163-175.

Gatignol, R. 1983 The Faxén formulae for a rigid particle in an unsteady non-uniform Stokes flow. *Journal de mécanique théorique et appliquée*, **1** (2), 143-160.

Germano, M., Piomelli, U., Moin, P. & Cabot, W. 1991 A dynamic subgrid-scale eddy viscosity model. *Phys. Fluids* **3** (7), 1760-1765.

Ham, F. & Iaccarino, G. 2004 Energy conservation in collocated discretization schemes on unstructured meshes. In *Annual Research Briefs*, pp. 3-14. Center for Turbulence Research, NASA Ames/Stanford Univ.

Kaufmann, A. 2004 Vers la simulation des grandes échelles en formulation Euler/Euler des écoulements réactifs diphasiques. PhD thesis.

Kaufmann, A., Simonin, O., Cuenot, B., Poinsot, T. & Helie, J. 2003 Dynamics and dispersion in 3D unsteady simulations of two phase flows. In *Supercomputing in Nuclear Applications*. Paris: CEA.

Kim, J. & Moin, P. 1985 Application of a fractional-step method to incompressible Navier-Stokes equations. *J. Comp. Phys.* **59** (2), 308-323.

Mahesh, K., Constantinescu, G. & Moin, P. 2004 A numerical method for large-eddy simulation in complex geometries. *J. Comp. Phys.* **197** (1), 215-240.

Maxey, M. & Riley, J. 1983 Equation of motion for a small rigid sphere in a nonuniform flow. *Phys. Fluids* **26** (4).

Moin, P., Squires, K., Cabot, W. & Lee, S. 1991 A dynamic subgrid-scale model for compressible turbulence and scalar transport. *Phys. Fluids* **A 3** (11), 2746-2757.

Moreau, M., Bédat, B. & Simonin, O. 2005 A priori testing of subgrid stress models for euler-euler two-phase LES from euler-lagrange simulations of gas-particle turbulent flow. In *18th Ann. Conf. on Liquid Atomization and Spray Systems*. ILASS Americas.

Moureau, V., Lartigue, G., Sommerer, Y., Angelberger, C., Colin, O. & Poinsot, T. 2005 High-order methods for dns and les of compressible multicomponent reacting flows on fixed and moving grids. *J. Comp. Phys.* **202** (2), 710-736.

Pitsch, M. & Duchamp de la Geneste, L. 2002 Large eddy simulation of premixed turbulent combustion using a level-set approach. *Proc. of the Comb. Institute* **29**, 2001-2005.

Poinsot, T. & Veynante, D. 2005 *Theoretical and numerical combustion* R.T. Edwards, 2nd edition.

Riber, E., Moreau, M., Simonin, O. & Cuenot, B. 2005 Towards large eddy simulations of non-homogeneous particle laden turbulent gas flows using euler-euler approach. In *11th Workshop on Two-Phase Flow Predictions*. Merseburg, Germany.

Riber, E., Moreau, M., Simonin, O. & Cuenot, B. 2006 Development of Euler-Euler LES Approach for Gas-Particle Turbulent Jet Flow. In *Proc. Symposium Fluid-Particle Interactions in Turbulence*. 2006 ASME Joint U.S. European Fluids Engineering Summer Meeting. Miami, FEDSM2006-98110.

Roux, S., Lartigue, G., Poinsot, T., Meier, U. & Bérat, C. 2005 Studies of mean and unsteady flow in a swirled combustor using experiments, acoustics analysis and large eddy simulations. *Combust. Flame* **141**, 40-54.

Schiller, L. & Nauman, A. 1935 A drag coefficient correlation. *V.D.I. Zeitung* **77**, 318-320.

Schmitt, P., Poinsot, T., Schuermans, B. & Geigle, K. 2006 Large-eddy simulation and experimental study of heat transfer, nitric oxide emissions and combustion instability in a swirled turbulent high pressure burner. *J. Fluid Mech.* **in print**.

Selle, L., Lartigue, G., Poinsot, T., Koch, R., Schildmacher, K.-U., Krebs, W., Prade, B., Kaufmann, P. & Veynante, D. 2004 Compressible large-eddy simulation of turbulent combustion in complex geometry on unstructured meshes. *Combust. Flame* **137** (4), 489-505.

Sommerer, Y., Galley, D., Poinsot, T., Ducruix, S., Lacas, F. & Veynante, D. 2004 Large eddy simulations and experimental study of flashback and blow-off in a lean partially premixed swirled burner. *J. Turb.* **5**.

Vermorel, O., Bédat, B., Simonin, O. & Poinsot, T. 2003 Numerical study and modelling of turbulence modulation in a particle laden slab flow. *J. Turb.* **4, 025**.

Vreman, B., Geurts, B. & Kuerten, H. 1995 Subgrid modeling in les of compressible flow. *Appl. Sci. Res.* **54**, 191-203.

Yoshizawa, A. 1986 Statistical theory for compressible turbulent shear flows, with the application to subgrid modeling. *Phys. Fluids.* **29** (7), 2152-2164.

Appendix C

6th Int. Conference on Multiphase Flow, Leipzig, S3_Fri_A_62 (2007)

Comparaison des deux approches LES Euler/Euler et Euler/Lagrange pour la prédiction d'un écoulement confiné gaz-solide du type " bluff-body "

Aujourd'hui, l'approche numérique par résolution des équations de Navier-Stokes moyennées (RANS pour Reynolds-averaged Navier-Stokes) est régulièrement utilisée comme outil de conception de chambres de combustion, fonctionnant aussi bien avec des combustibles gazeux que des combustibles liquides. Avec le RANS, l'écoulement moyen est résolu alors que les structures turbulentes sont toutes modélisées. Récemment, afin d'offrir une meilleure prédiction des valeurs moyennes de l'écoulement mais également afin de donner accès à des phénomènes instables qui se produisent a l'intérieur des chambres de combustion, la simulation aux grandes échelles (LES pour Large-Eddy Simulation) a été étendue aux écoulements réactifs. Le succès de ces approches pour les flammes monophasiques dans les dernières années est une illustration claire de leur potentiel. La LES donne accès aux grandes structures de l'écoulement en réduisant l'importance de la modélisation et capture ainsi une partie importante de la physique qui contrôle ces flammes. Même si la LES a déjà démontré son potentiel sur des flammes monophasiques, son extension aux flammes diphasiques reste à explorer. Tout d'abord, la physique utilisée pour décrire l'atomisation d'un jet liquide, la dispersion des particules solides, leur interaction avec les parois, leur évaporation et combustion est aussi difficile à modéliser pour la LES que pour le RANS car il s'agit essentiellement de phénomènes de sous-maille. Deuxièmement, l'implantation numérique des écoulements diphasiques pour la LES reste un défi. Les équations pour la phase gazeuse et la phase dispersée doivent être résolues ensemble à chaque pas de temps d'une manière fortement couplée. Par ailleurs, dans le contexte des supercalculateurs parallèles, l'efficacité numérique est une contrainte supplémentaire et le maintien d'une efficacité du parallélisme comparable à celle des calculs gazeux n'est pas si évident pour un calcul diphasique.

Dans cette étude, les résultats des simulations numériques LES des écoulements turbulents chargés en particules sont comparés aux résultats expérimentaux de la configuration étudiée par Borée *et al.* [22] où des particules de verre sont injectées dans un écoulement turbulent comportant une zone de recirculation. Ces tests sont effectués sans combustion ni évaporation, mais sont obligatoires avant de réaliser ce type de calculs dans des géométries complexes proches de celles des chambres de combustion. L'ensemble complet de mesures expérimentales est disponible

en ligne à l'adresse suivante : http://www-mvt.iw.uni-halle.de/english/index.php?bluff_body_flow. Ces données ont été utilisés pour le test de performance du '9ième workshop pour la prédiction des écoulements diphasiques' (Ishima *et al.* [92]). Dans cette configuration, un jet d'air avec des particules de verre est injecté dans la chambre par le biais d'un tube d'injection entouré d'un co-courant d'air. Le rapport de vitesse entre le tube d'injection et le co-courant a été choisi afin de créer une large zone centrale de recirculation entre le jet central et le co-courant. Le diamètre des particules de verre est compris entre 20 et 100 microns et leur masse volumique est de 2470 kg/m^3. Cette configuration dispose de deux bases de données complètes pour un cas faiblement chargé en masse ($M_j = 22\%$) et un autre fortement chargé en masse ($M_j = 110\%$). Les comparaisons présentées dans cet article se limitent au calcul faiblement chargé où l'écoulement diphasique considéré est suffisamment dilué pour négliger les collisions interparticulaires. Les simulations numériques ont été réalisées uniquement avec des particules de 60 microns. Des études non rapportées ici ont montré qu'un calcul monodisperse avec la taille moyenne en masse de particules est suffisant pour capturer l'effet moyen de la phase gazeuse et la dynamique des particules.

Les méthodes les plus classiques pour la description de la phase dispersée dans ces écoulements sont les approches Euler/Euler (EE) et Euler/Lagrange (EL). Dans la première approche, un même maillage et utilisé pour résoudre l'ensemble des équations différentielles partielles qui decrivent la phase gazeuse et la phase liquide. Dans la seconde approche, le gaz est résolu sur un maillage Eulérien alors que la phase liquide est simulée au moyen de particules Lagrangiennes. Dans les deux cas, le couplage est nécessaire et intervient au travers d'échanges d'information entre le système fluide et liquide. Le solveur utilisé pour cette étude, AVBP, est un code parallèle explicite qui résout les équations de Navier-Stokes en 3D de façon compressible sur de maillages non-structurés et hybrides. Ce solveur contient à la fois les approches Eulérienne et Lagrangienne pour la description de la phase dispersée. Le schéma utilisé dans cette étude est d'ordre 3 en espace avec un avancement en temps de type Runge-Kutta. Le modèle de sous-maille est celui de Smagorinsky [204] et les conditions limites sont du type NSCBC (Poinsot & Veynante [165], Moureau *et al.* [138]).

Cet article est organisé de la façon suivante : d'abord une description de la configuration et des objectifs du travail est présentée. Ensuite, le solveur utilisé pour les simulations numériques est décrit, suivi d'une présentation détaillée des équations de la phase gazeuse et de la phase dispersée (pour les deux approches). Les résultats des calculs gazeux et diphasique sont présentés en comparaison avec les résultats expérimentaux sur sept profils radiaux de vitesses moyenne et fluctuante. Les résultats montrent que l'écoulement est bien prédit. La phase dispersée est bien prédite avec les deux approches, mais la formulation Lagrangienne prédit les valeurs fluctuantes (RMS) plus précisément. La dernière section de l'article contient une analyse des performances (speedup et temps CPU) de calcul gazeux et diphasique. Le coût supplémentaire du calcul diphasique avec l'approche Eulérienne par rapport a un calcul gazeux est invariablement estimé autour de 80% et ne dépend pas de la charge en masse. Par rapport au calcul diphasique Eulérien, dans le cas où le nombre de particules reste faible par rapport au nombre de cellules du maillage, l'approche EL se révéle plus rapide jusqu'à 64 processeurs. Les principales raisons de la chute des performances au-delà de 64 processeurs sont liées à l'anisotropie du maillage tétraédrique utilisé et à l'absence de prise en compte des particules au moment du découpage du maillage. Ces deux faits ensemble vont générer dans ce cas un grand déséquilibrage de charges entre les processeurs qui peuvent arriver à des situations où un seul processeur contient la moitie de particules du domaine.

6th International Conference on Multiphase Flow,
ICMF 2007, Leipzig, Germany, July 9 – 13, 2007

Comparison between Euler/Euler and Euler/Lagrange LES approaches for confined bluff-body gas-solid flow prediction

Marta García[1], Eleonore Riber[1,2], Olivier Simonin[2] and Thierry Poinsot[2]

[1] CERFACS, 42, av. Gaspard Coriolis, Toulouse, 31057, France

[2] IMF Toulouse, UMR CNRS/INPT/UPS, Allée du Professeur Camille Soula, Toulouse, 31400, France

garcia@cerfacs.fr, riber@cerfacs.fr, simonin@imft.fr, poinsot@imft.fr

Keywords: Large-eddy simulation, gas-solid flows, bluff-body flow, Eulerian and Lagrangian approaches

Abstract

In this study, Euler/Euler and Euler/Lagrange LES predictions of particle-laden turbulent flows are compared for the bluff-body configuration from Borée *et al.* (2001) where glass beads are injected into a complex recirculating flow. These tests are performed for non-reacting, non-evaporating sprays but are mandatory validations before computing realistic combustion chambers. The numerical code used for this study is a parallel explicit CFD code that solves the 3D compressible Navier-Stokes equations on unstructured and hybrid grids. This solver contains both Euler/Euler and Euler/Lagrange formulations. Results show that the gas flow and the dispersed phase are well predicted but the Lagrangian approach predicts RMS values more precisely. The importance of inlet boundary conditions for the gas is revealed.

Introduction

Today, RANS (Reynolds-averaged Navier-Stokes) equations are routinely solved to design combustion chambers, for both gaseous and liquid fuels. Recently, in order to provide better accuracy for the prediction of mean flows but also to give access to unsteady phenomena occurring in combustion devices (such as instabilities, flashback or quenching), Large-Eddy Simulation (LES) has been extended to reacting flows. The success of these approaches for gaseous flames in the last years (Caraeni *et al.* 2000; Colin *et al.* 2000; Selle *et al.* 2004; Roux *et al.* 2005; Poinsot & Veynante 2005) is a clear illustration of their potential. LES gives access to the large scales structures of the flow reducing the importance of modelling, and naturally capturing a significant part of the physics controlling these flames. Even though LES has already demonstrated its potential for gaseous flames, its extension to two-phase flames is still largely to be done. First, the physical submodels required to describe the atomization of a liquid fuel jet, the dispersion of solid particles, their interaction with walls, evaporation and combustion are as difficult to build in LES as in RANS because they are essentially subgrid phenomena. Second, the numerical implementation of two-phase flow LES remains a challenge. The equations for both the gaseous and the dispersed phases must be solved together at each time step in a strongly coupled manner. This differs from classical RANS where the resolution of the two phases can be done in a weak procedure, bringing first the gas flow to convergence, then the solid particles and finally iterating until convergence of both phases. Finally, in the context of parallel super-computing, numerical efficiency is an additional constraint. For single-phase flows, efficient and accurate solvers have been developed and speedups of the order of 5000 are not uncommon (http://www.cerfacs.fr/cfd/parallel.html). Maintaining a similar parallel efficiency for a two-phase flow solver while representing the main physics of the flow raises additional questions.

In LES of two-phase flows, physics and numerics interact strongly: the first question is to choose a paradigm to describe the two-phase flow. Most RANS codes use Euler/Lagrange (EL) methods in which the flow is solved using an Eulerian method and the particles are tracked using a Lagrangian approach. An alternative technique is to use two-fluid models in which both the gas and the dispersed phases are solved using an Eulerian method (Euler/Euler or EE) (Reeks 1991; Février & Simonin 1999). The history of RANS development has shown that both EE and EL are useful and either is found today in most commercial codes. For LES, both EE and EL formulations are being developed and the focus of this study is to test them in a reference case where complete sets of solutions for gas and dispersed phase are available. This exercice is performed here without evaporation or combustion.

Nomenclature

C_D	drag coefficient
C_I, C_S	model constants
C_v	specific heat at constant volume ($J\ kg^{-1}\ K^{-1}$)
d_p	particle diameter (m)
e_g	internal energy ($m^2\ s^{-2}$)
E_g	total energy ($m^2\ s^{-2}$)
$f_{c,i}$	coupling force ($kg\ m^{-2}\ s^{-2}$)

$\breve{f}_p$	probability density function ($s^3\ m^{-6}$)
g	gravitational constant ($m\ s^{-2}$)
n_p	particle number density (m^{-3})
N_{procs}	number of processors
p	pressure ($N\ m^{-2}$)
Pr	Prandtl number
$q_{g,j}$	heat tranfer vector ($J\ m^{-2}\ s^{-1}$)
$q_{gp,SGS}$	subgrid covariance ($m^{-2}\ s^{-2}$)
Q	diffusion term ($J\ m^{-2}\ s^{-1}$)
$\mathcal{R}$	air gas constant ($J\ kg^{-1}\ K^{-1}$)
r	radial direction (m)
Re	Reynolds number
S	strain rate tensor (s^{-1})
t	time (s)
T	temperature (K)
$\mathcal{T}$	stress tensor ($kg\ m^{-1}\ s^{-2}$)
u_i	velocity vector, i=1,2,3 ($m\ s^{-1}$)
v_r	local instantaneous relative velocity ($m\ s^{-1}$)
x_i	position vector, i=1,2,3 (m)
z	axial direction (m)

Greek letters

α	volume fraction
δ_{ij}	Kronecker delta
$\delta\theta_p$	Random Uncorrelated Energy (RUE) ($m^2\ s^{-2}$)
$\delta R_{p,ij}$	Random Uncorrelated Velocity (RUV) tensor ($m^2 s^{-2}$)
$\delta S_{p,iij}$	RUV third correlation tensor ($m^{-3}\ s^{-3}$)
Δ_f	filter characteristic length (m)
η	dynamic viscosity ($kg\ m^{-1}\ s^{-1}$)
κ	diffusion coefficient ($m^2\ s^{-1}$)
ν	kinematic viscosity ($m^2\ s^{-1}$)
$\Pi_{\delta\theta_p}$	production term by subgrid scales ($m^{-1}\ s^{-3}$)
ρ	density ($kg\ m^{-3}$)
τ_p	particle relaxation time (s)
$\tau_{g,ij}$	viscous stress tensor ($kg\ m^{-1}\ s^{-2}$)
ϕ	azimuthal direction (m)

Subscripts

g	gas phase
i,j,k	index of coordinates directions
p	particle (dispersed phase)
RUM	Random Uncorrelated Motion
SGS	subgrid-scale

Symbols

$\bar{\cdot}$	LES-filtered quantity
$\tilde{\cdot}$	gas Favre LES-filtered quantity
$\hat{\cdot}$	particle Favre LES-filtered quantity
$\breve{\cdot}$	mesoscopic quantity

Configuration and work objectives

In the present study, two approaches developed at CERFACS within the same solver are used to investigate some critical issues for LES of two-phase flows on massively parallel computers. The explicit compressible solver AVBP is used with both EE (Kaufmann *et al.* 2003) and EL formulations on the same tetrahedron-based grid.

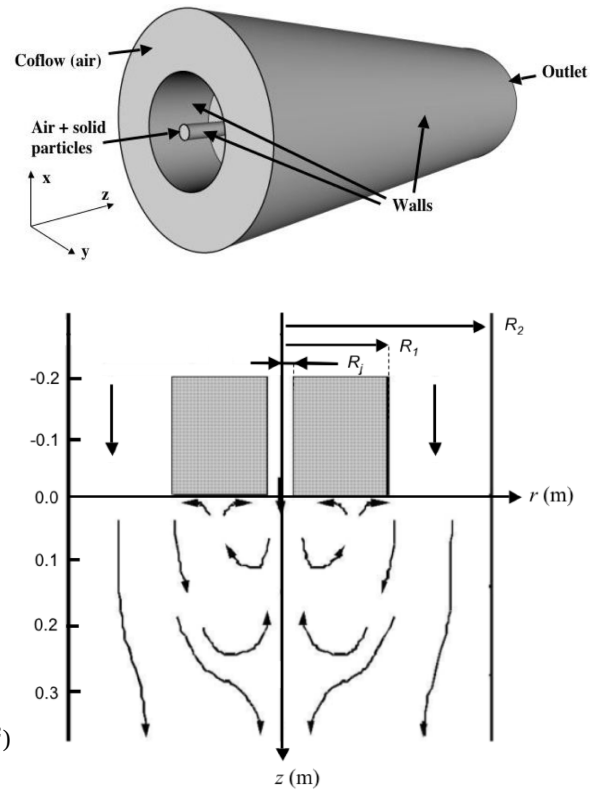


Figure 1: Configuration of Borée *et al.* (2001). The dimensions are : $R_j = 10$ mm, $R_1 = 75$ mm, $R_2 = 150$ mm. The total length of the experiment is 1.5 m.

Both approaches are used to study a bluff-body configuration (Borée *et al.* 2001) where a jet of air and solid particles are injected in a coflow of air (see the sketch in Fig. 1). The jet velocity on the axis is 4 m/s and the maximum coflow velocity is 6 m/s. The experiment is designed to provide large recirculation zones between the central jet and the coflow. The dispersed phase consists of solid particles (glass beads with diameter ranging from 20 to 100 microns with a mean value of 60 microns) so that evaporation, coalescence and break up do not have to be considered. The material density of the glass particle is $\rho_p = 2470$ kg m^{-3}. The mass loading ratio of particles in the inner jet is 0.22 corresponding to a solid volume fraction smaller than 10^{-4}. Thus collision effects are assumed to be negligible in the modelling approaches.

Measurements are performed by a two-component phase-Doppler anemometer (PDA). The origin is set at the edge of the bluff body and at the centre of the inner jet (see Fig. 1). The flow will be described using a cylindrical coordinate system (z, r, ϕ) to indicate the axial (downward), radial and azimuthal directions. Single-phase data are provided in tabulated form at different cross-sections within the jet, in the annular direction and along the z axis. The radial profiles of mean and RMS particle velocities for each size classes are provided in tabulated form at 7 cross-sections of the z axis ($z = 3, 80, 160, 200, 240, 320$ and 400 mm) and along the z axis up to 500 mm. The complete data set, including

accurate boundary conditions, at moderate mass loading (22 percent) has been selected for benchmarking at the 'Ninth workshop on two-phase flow predictions' (Ishima *et al.* 1999) and can be obtained at the following web site: http://www-mvt.iw.uni-halle.de/english/index.php?bluff_body_flow.
Despite of the relative simplicity, this test case contains a number of issues relevant for LES of two-phase flows. These include (i) the comparison of performances and CPU cost for EE and EL approaches and (ii) the analysis of the inlet boundary condition on the dispersed phase solution (turbulent modulation).

Description of the solver

The AVBP solver is a finite volume code based on a cell-vertex formulation. It solves the laminar and turbulent compressible Navier-Stokes equations in two and three space dimensions for hybrid and unstructured grids. Steady state or unsteady flows can be simulated, furthermore it takes into account the variations of molecular weights and heat capacities with temperature and mixture composition. A third-order scheme for spatial differencing and a Runge-Kutta time advancement (Colin & Rudgyard 2000; Moureau *et al.* 2005) is used for the present work. The Smagorinsky model is used to model the subgrid stress tensor. Walls are treated using the law-of-the-wall formulation by Schmitt *et al.* (2007). The boundary conditions are handled with the NSCBC formulation (Poinsot & Veynante 2005; Moureau *et al.* 2005).

The following sections briefly describe the governing equations solved by AVBP for the gaseous and dispersed phases.

Gaseous phase

The filtered conservation equation for gas-phase density, $\bar{\rho}_g$, momentum, $\tilde{u}_{g,i}$, and total energy $\tilde{E}_g = \tilde{e}_g + \frac{1}{2}\tilde{u}_{g,j}^2$ (with $\tilde{e}_g = C_v\tilde{T}_g$, the internal energy, C_v the specific heat at constant volume and $\tilde{T}_g$ the temperature) read:

$$\frac{\partial \bar{\rho}_g}{\partial t} + \frac{\partial(\bar{\rho}_g \tilde{u}_{g,j})}{\partial x_j} = 0 \tag{1}$$

$$\frac{\partial(\bar{\rho}_g \tilde{u}_{g,i})}{\partial t} + \frac{\partial(\bar{\rho}_g \tilde{u}_{g,i}\tilde{u}_{g,j})}{\partial x_j} + \frac{\partial \bar{p}_g}{\partial x_i} - \frac{\partial \bar{\tau}_{g,ij}}{\partial x_j} = \frac{\partial \mathcal{T}_{g,ij}}{\partial x_j} + f_{c,i} \tag{2}$$

$$\frac{\partial(\bar{\rho}_g \tilde{E}_g)}{\partial t} + \frac{\partial(\tilde{u}_{g,j}(\bar{\rho}_g \tilde{E}_g + \bar{p}_g))}{\partial x_j} - \frac{\partial(\bar{\tau}_{g,ij}\tilde{u}_{g,i})}{\partial x_j} + \frac{\partial \bar{q}_{g,j}}{\partial x_j} = \frac{\partial(\mathcal{T}_{g,ij}\tilde{u}_{g,i})}{\partial x_j} + \frac{\partial Q_{g,j}}{\partial x_j} + f_{c,j}\tilde{u}_{g,j}. \tag{3}$$

The left-hand-side (LHS) of Eqs. 1-3 contains all resolved (filtered) variables (being $\bar{\tau}_{g,ij}$ and $\bar{q}_{g,i}$ the viscous stress tensor and the heat transfer vector, while pressure is obtained from the equation of state $\bar{p}_g = \bar{\rho}_g \mathcal{R}\tilde{T}_g$). The right-hand-side (RHS) of Eqs. 2 and 3 contains the SGS terms $\mathcal{T}_{g,ij}$ and $Q_{g,i}$, which are reconstructed using eddy-viscosity concepts (with turbulent viscosity obtained from Smagorinsky model). The last terms in Eqs. 2 and 3, $f_{c,i}$ and $f_{c,j}\tilde{u}_{g,j}$, denote respectively, the coupling force and energy applied to the fluid by all particles.

Dispersed phase: Euler/Lagrange approach

The dispersed phase consists of particles which are assumed to be rigid spheres with diameter comparable or smaller than the Kolmogorov length scale. As the particle density is much larger than the fluid density ($\rho_p/\rho_g = 2470$), the forces acting on particles reduce to drag and gravity. Under these assumptions, the particle equations of motion can then be written for a single particle as:

$$\frac{dx_{p,i}}{dt} = u_{p,i} \tag{4}$$

$$\frac{du_{p,i}}{dt} = -\frac{3}{4}\frac{\rho_g}{\rho_p}\frac{C_D}{d_p}|\mathbf{v}_r|\,\mathrm{v}_{r,i} + g_i = -\frac{u_{p,i} - \tilde{u}_{g,i}}{\tau_p} + g_i \tag{5}$$

with g_i the gravity vector. The local drag coefficient in Eq. (5) is C_D and may be expressed in terms of the particle Reynolds number Re_p following Schiller & Nauman (1935):

$$C_D = \frac{24}{Re_p}\left[1 + 0.15 Re_p^{0.687}\right] \tag{6}$$

$$Re_p = \frac{|\mathbf{v}_r|\,d_p}{\nu_g} \leq 800 \tag{7}$$

where d_p is the particle diameter and ν_g is the kinematic viscosity of the gas phase. The local instantaneous relative velocity between the particle and the surrounding fluid is $\mathrm{v}_{r,i} = u_{p,i} - \tilde{u}_{g,i}$, where $\tilde{u}_{g,i}$ is the fluid velocity at the position of the particle assuming that the flow field is locally undisturbed by the presence of this particle (Gatignol 1983; Maxey & Riley 1983). In first approximation, the velocity is assumed to be equal to the interpolation of the filtered velocity at the position of the particle (Wang & Squires 1996; Yamamoto *et al.* 2001; Apte *et al.* 2003). The effect of the subgrid fluid turbulence is assumed to be negligible owing to the large inertia of the solid particles (Fede & Simonin 2006). The particle relaxation time τ_p is defined as the Stokes characteristic time:

$$\tau_p = \frac{4}{3}\frac{\rho_p}{\rho_g}\frac{d_p}{C_D\,|\mathbf{v}_r|}. \tag{8}$$

The influence of the particles on the gas phase is taken into account in the EL simulations by using the point-force approximation in the general framework of the particle-in-cell method (PIC) (Boivin *et al.* 1998; Vermorel *et al.* 2003), with standard single-phase subgrid turbulence modelling approaches. According to Boivin *et al.* (2000), such an assumption is valid for small mass loading ratio of particles (typically, $\alpha_p\rho_p/\rho_g \leq 1$) with response time larger than the subgrid turbulence characteristic time scale. Modification

of the gas subgrid-scale turbulence model by the particles is neglected. A linear interpolation algorithm is used to compute the fluid velocity at the position of the particle. If particle relaxation time is much larger than the time scale of filtered velocity fluctuations (as in the present case of 22 percent mass loading), such a linear interpolation is found to be sufficiently accurate to resolve particle motions (see e.g. Fede & Simonin (2006)).

Dispersed phase: Euler/Euler approach

Eulerian equations for the dispersed phase can be derived using several approaches. A popular and simple way consists in volume filtering of the separate, local, instantaneous phase equations accounting for the inter-facial jump conditions (Druzhinin & Elghobashi 1999). Such an averaging approach is restrictive because particle sizes and particle distances have to be smaller than the smallest length scale of the turbulence. Besides, they do not account for the crossing of particle trajectories or Random Uncorrelated Motion (RUM), shown by Février *et al.* (2005), which may appear when the particle relaxation time is larger than the Kolmogorov time scale. In the present study, a statistical approach analogous to kinetic theory (Chapman & Cowling 1939) is used to construct a probability density function (pdf) $\breve{f}_p(\mathbf{c}_p, \mathbf{x}, t)$ which gives the local instantaneous probable number of particles with the given translation velocity $\mathbf{u}_p = \mathbf{c}_p$. The resulting model (Février *et al.* 2005; Moreau *et al.* 2005) leads to equations for the particle number density $\bar{n}_p$ and the correlated velocity $\hat{\mathbf{u}}_p$:

$$\frac{\partial}{\partial t}\bar{n}_p + \frac{\partial}{\partial x_j}\bar{n}_p\hat{u}_{p,j} = 0 \tag{9}$$

$$\frac{\partial}{\partial t}\bar{n}_p\hat{u}_{p,i} + \frac{\partial}{\partial x_j}\bar{n}_p\hat{u}_{p,i}\hat{u}_{p,j} = -\frac{\bar{n}_p}{\tau_p}\left(\hat{u}_{p,i} - \hat{u}_{g,i}\right)$$
$$+\bar{n}_p g_i - \frac{\partial}{\partial x_j}\mathcal{T}_{p,ij} - \frac{\partial}{\partial x_j}\bar{n}_p\widehat{\delta R}^*_{p,ij} - \frac{\partial}{\partial x_i}\frac{2}{3}\bar{n}_p\widehat{\delta\theta}_p \tag{10}$$

where $\bar{n}_p$, $\hat{\mathbf{u}}_p$ and $\widehat{\delta\theta}_p$ are respectively the filtered particle number density, correlated velocity and Random Uncorrelated Energy (RUE). The two first terms of the RHS of Eq. (10) are the drag force and gravity effects on large scales, the third one accounts for the subgrid-scale (SGS) effects, the fourth one takes into account the dissipation effects induced by the RUM and the last one is a particle-pressure term proportional to the RUE. $\mathcal{T}_{p,ij}$ stands for the particle subgrid stress tensor:

$$\mathcal{T}_{p,ij} = \bar{n}_p(\widehat{u_{p,i}u_{p,j}} - \hat{u}_{p,i}\hat{u}_{p,j}). \tag{11}$$

As in fluid non-isotherm turbulence, an additional equation for energy is needed. The transport equation of filtered RUE is:

$$\frac{\partial}{\partial t}\overline{n}_p\widehat{\delta\theta}_p + \frac{\partial}{\partial x_j}\overline{n}_p\hat{u}_{p,j}\widehat{\delta\theta}_p = -2\frac{\overline{n}_p}{\tau_p}\widehat{\delta\theta}_p - \frac{2}{3}\overline{n}_p\widehat{\delta\theta}_p\frac{\partial\hat{u}_{p,j}}{\partial x_j}$$
$$-\overline{n}_p\widehat{\delta R}^*_{p,ij}\frac{\partial\hat{u}_{p,i}}{\partial x_j} - \frac{1}{2}\frac{\partial}{\partial x_j}\overline{n}_p\widehat{\delta S}_{p,iij} + \Pi_{\delta\theta_p} - \frac{\partial}{\partial x_j}Q_{p,j}. \tag{12}$$

The first RHS term is the RUE destruction by drag force, the second one is a RUE-dilatation term, the third one is a production term by filtered Random Uncorrelated Velocity (RUV) tensor, the next one is the diffusion by filtered RUV third correlation tensor. $\Pi_{\delta\theta_p}$ and $Q_{p,j}$ are respectively production and diffusion terms by subgrid scales:

$$\Pi_{\delta\theta_p} = \left(\overline{\breve{n}_p\delta R_{p,ij}\frac{\partial\breve{u}_{p,i}}{\partial x_j}} - \bar{n}_p\widehat{\delta R}_{p,ij}\frac{\partial\hat{u}_{p,i}}{\partial x_j}\right) \tag{13}$$

$$Q_{p,j} = \bar{n}_p\left(\widehat{u_{p,j}\delta\theta_p} - \hat{u}_{p,j}\widehat{\delta\theta}_p\right). \tag{14}$$

The particle source term in the gas phase momentum Eq. 2 is equal to minus the drag term in the particle phase Eq. 10.

Closure of filtered RUV terms

Assuming small anisotropy of the RUM, Simonin *et al.* (2002) model $\delta R^*_{p,ij}$ by a viscous term and Kaufmann *et al.* (2005) model $\delta S_{p,iij}$ by a diffusive term similar to Fick's law. For LES approach these models are adapted by replacing non filtered quantities by filtered ones leading to (Moreau *et al.* 2005):

$$\widehat{\delta R}^*_{p,ij} = -\hat{\nu}_{RUM}\left(\frac{\partial\hat{u}_{p,i}}{\partial x_j} + \frac{\partial\hat{u}_{p,j}}{\partial x_i} - \frac{\partial\hat{u}_{p,k}}{\partial x_k}\frac{\delta_{ij}}{3}\right) \tag{15}$$

$$\frac{1}{2}\widehat{\delta S}_{p,iij} = -\hat{\kappa}_{RUM}\frac{\partial\widehat{\delta\theta}_p}{\partial x_j} \tag{16}$$

where the RUM viscosity, $\hat{\nu}_{RUM}$, and the RUM diffusion coefficient, $\hat{\kappa}_{RUM}$, are given by:

$$\hat{\nu}_{RUM} = \frac{\tau_p}{3}\widehat{\delta\theta}_p \quad \text{and} \quad \hat{\kappa}_{RUM} = \frac{10}{27}\tau_p\widehat{\delta\theta}_p. \tag{17}$$

Subgrid terms modeling

By analogy to single phase flows (Moin *et al.* 1991; Vreman *et al.* 1995), Riber *et al.* (2005) propose a viscosity model for the SGS tensor $\mathcal{T}_{p,ij}$. The trace-free SGS tensor is modeled using a viscosity assumption (compressible Smagorinsky model), while the subgrid energy is parametrized by a Yoshizawa model (Yoshizawa 1986):

$$\mathcal{T}_{p,ij} = -C_S 2\Delta_f^2\bar{n}_p|\hat{S}_p|\left(\hat{S}_{p,ij} - \frac{\delta_{ij}}{3}\hat{S}_{p,kk}\right)$$
$$+ C_I 2\Delta_f^2\bar{n}_p|\hat{S}_p|^2\delta_{ij} \tag{18}$$

where $\hat{S}_p$ is the filtered particle strain rate tensor, $|\hat{S}_p|^2 = 2S_{p,ij}S_{p,ij}$ and Δ_f the filter characteristic length. The model constants have been evaluated in a priori tests (Riber *et al.* 2006) leading to the values $C_S = 0.02$, $C_I = 0.012$.
The subgrid diffusion term in the filtered RUE is modeled by an eddy-diffusivity model:

$$Q_{p,j} = -\frac{\bar{n}_p C_S 2\Delta_f^2|\hat{S}_p|}{Pr_{p,SGS}}\frac{\partial\widehat{\delta\theta}_p}{\partial x_j} \tag{19}$$

with the particle turbulent Prandtl number $Pr_{p,SGS} = 0.8$. The subgrid production of filtered RUE term $\Pi_{\delta\theta_p}$ acts like a dissipation term in the subgrid energy equation. Using an

equilibrium assumption on the particle correlated subgrid energy and neglecting diffusion terms leads to:

$$-\frac{\bar{n}_p}{\tau_p}\left(\frac{\mathcal{T}_{p,kk}}{\bar{n}_p} - q_{gp,SGS}\right) + \Pi_{\delta\theta_p} - \mathcal{T}_{p,ij}\frac{\partial \hat{u}_{p,i}}{\partial x_j} = 0 \quad (20)$$

where the subgrid covariance is $q_{gp,SGS} = \widehat{u_{p,k}u_{g,k}} - \hat{u}_{p,k}\hat{u}_{g,k}$. To first order, the drag force term can be neglected and $\Pi_{\delta\theta_p}$ can be modeled by: $\Pi_{\delta\theta_p} \approx \mathcal{T}_{p,ij}\partial\hat{u}_{p,i}/\partial x_j$ with the SGS tensor modeled by Eq. (18). This model ensures that the correlated energy dissipated by subgrid effects is fully transfered into RUE to be finally dissipated by friction with the fluid.

Comparison of gas flow without particles

Before discussing results for the dispersed phase, the accuracy of the LES solver for the gas phase is evaluated by computing the flow without particles and comparing it to the same data provided in Borée *et al.* (2001). The grid used with the code AVBP is presented in Fig. 2 and some parameters of the simulation are summarized in Table 1.

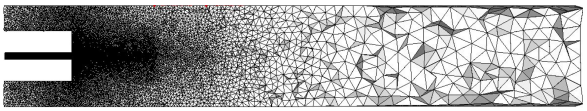


Figure 2: Geometry of the computational domain. Grid elements used: tetrahedra.

Grid type	Tetrahedra
Number of cells / nodes	2,058,883 / 367,313
Time step (μs) / CFL	3.2 / 0.7
Averaging time (s) / Iterations	1.03 / 320,000
LES model	Smagorinsky
Wall model	Law-of-the-wall

Table 1: Summary of parameters and models used in AVBP for the gas-flow computation without particles.

A typical snapshot of the velocity field (modulus) in the central plane is displayed in Fig. 3. The figure shows the complex structure of the recirculating flow: on the axis, the flow is recirculating down to $z = 200$ mm. On the sides of the channel, the flow also separates from $z \approx 50$ mm to $z \approx 400$ mm.

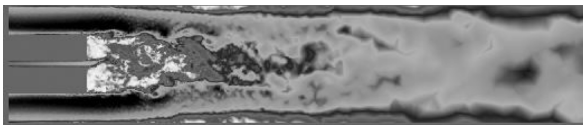


Figure 3: Instantaneous field of velocity modulus. Maximum value (black): 6 m/s. Minimum value (white): 0 m/s.

In Figs 4 to 7, the radial profiles (averaged in the azimuthal direction) of mean and RMS velocities obtained by AVBP are compared with the experimental values at 7 stations of the z axis ($z = 3, 80, 160, 200, 240, 320$ and 400 mm). The LES solver captures most of the flow physics: the axial mean and RMS velocities (Fig. 4 and 5) agree with the measurements. The length of the recirculation zone (evidenced by the negative values of axial velocities on the axis) is well predicted. In the coflow, the RMS values predicted by LES are too low because no turbulence is injected at the inlet of the domain for these computations.

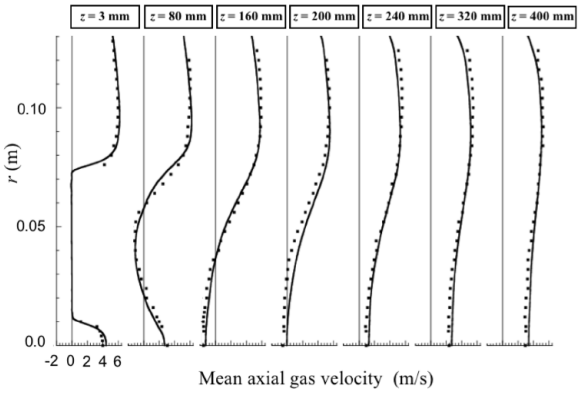


Figure 4: Radial profiles of mean axial gas velocities at 7 stations along z axis. Symbols: experiment; solid line: AVBP.

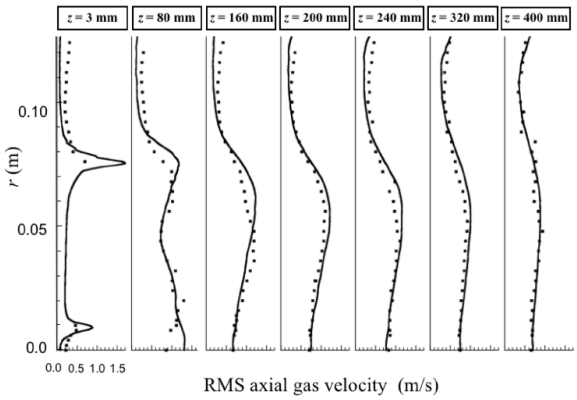


Figure 5: Radial profiles of RMS axial gas velocities at 7 stations along z axis. Symbols: experiment; solid line: AVBP.

The mean radial velocity levels (Fig. 6) remain small (less than 1 m/s) and the LES code captures the radial velocity fields correctly (Fig. 7). The particle mean stagnation point (around $z = 160$ mm) is a delicate zone where the AVBP solver has some difficulties. The source of this problem is the exact position of the stagnation point: any small mismatch in this position leads to large changes in profiles measured around this point. Upstream and downstream of this point, the agreement is very good.

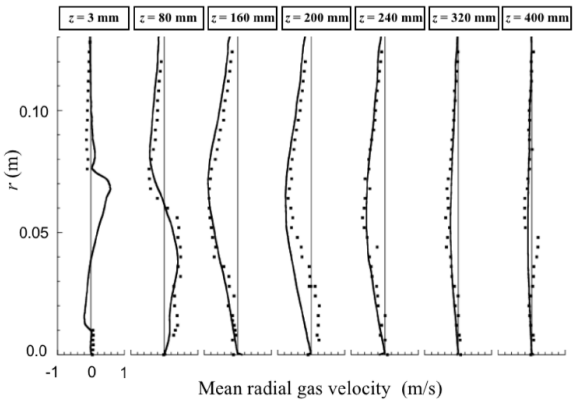


Figure 6: Radial profiles of mean radial gas velocities at 7 stations along z axis. Symbols: experiment; solid line: AVBP.

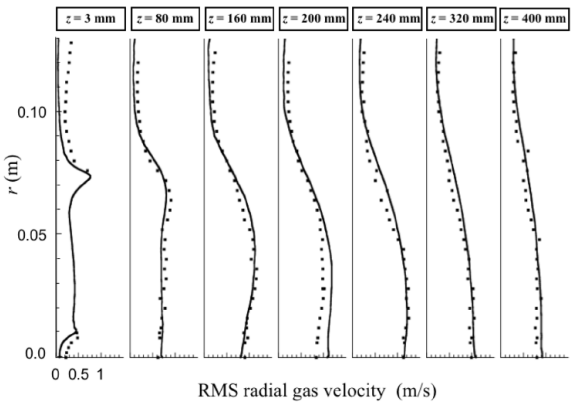


Figure 7: Radial profiles of RMS radial gas velocities at 7 stations along z axis. Symbols: experiment; solid line: AVBP.

The code exhibits an overall good agreement with experimental results. This indicates that tests for the dispersed phase can be performed with reasonable confidence.

Results for two-phase flow cases

This section presents the results for the 22 percent mass loading of the central jet, obtained with two different computations summarized in Table 2[1]. The grid and the time step used are presented in Table 1. In all computations presented here, the injected particles have a size of 60 microns. Separated studies which are not reported here, using another Lagrangian solver and multidisperse particles or 60 microns particles have shown that using a monodisperse distribution of size was very close to the 22 percent case of Borée *et al.* (2001) and was sufficient to capture both the mean flow effects on the gas (through two-way coupling) and the dynamics of the 60 microns class.

[1]For these runs, the RUM model is not used and the $\widehat{\delta\theta}_p$ term in Eq. (10) is set to zero.

	EE	EL
Averaging time (s)	0.64	0.80
Particle mean speed	Exp. profile	Exp. profile
Turbulent fluctuations	Zero	White noise (12%)
Particle distribution	Exp. profile	Homogeneous

Table 2: Summary of parameters and models used for the particle injection (22 percent mass loading computation). The particles are injected in the central tube.

An essential part of these LES is the introduction of the particles in terms of position and velocity. The injection planes are not the same for both approaches (Fig. 8). The methodologies used to inject the particles are also different to evaluate their impact on results. In EE, both the mass loading and the mean velocity imposed in the injection plane ($z = -200$ mm) are the ones measured experimentally at $z = 3$ mm. No turbulent fluctuations are introduced. In the EL formulation, the mass loading is homogeneous over the injection section and the injection speed profile is also the experimental one measured at $z = 3$ mm. In the EL formulation, a white noise (amplitude of the order of 12 percent of the mean velocity) is added to the particle mean velocity profiles to match experimental measurements at $z = 3$ mm.

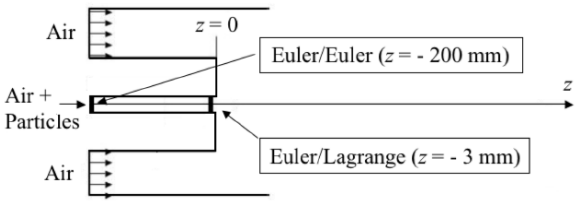


Figure 8: Injection position for particles.

The velocity fields for the gas phase change when the particles are injected but these effects are limited and are not discussed here. Figures 9 to 12 show velocity fields for particles obtained with both approaches along with the measurements of Borée. The agreement between the experiments and the two LES sets of data is good. An interesting result is that EE (solid line) and EL (dashed line) provide similar results showing that the EE approach is able to reproduce the mean-flow properties predicted by the EL computation. On the other hand, Figs. 10 and 12 show that EL formulation predicts particle RMS velocity more precisely. This is consistent with the fact that, when no RUM model is used, the EE approach underestimates turbulent fluctuations of particle velocity. Recent studies by Riber *et al.* (2006) have shown that when these contributions are considered, particle velocity fluctuations are correctly predicted.
A convenient way to look at the results is to consider the central z axis of the configuration: a critical zone is the stagnation point for the gas located around $z = 160$ mm. This is also a zone where particles accumulate and must stop before

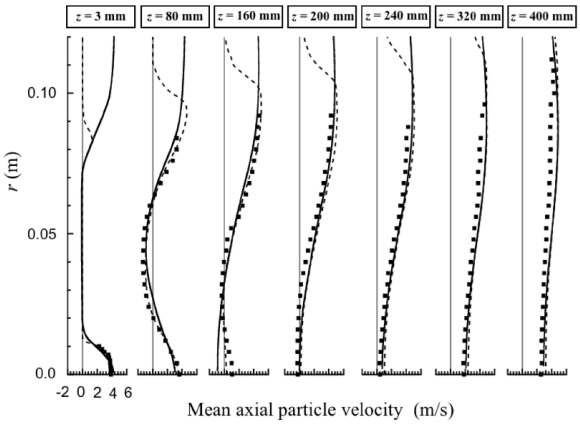


Figure 9: Radial profiles of mean axial particle velocities at 7 stations along z axis. Symbols: experiment; solid line: EE; dashed line: EL.

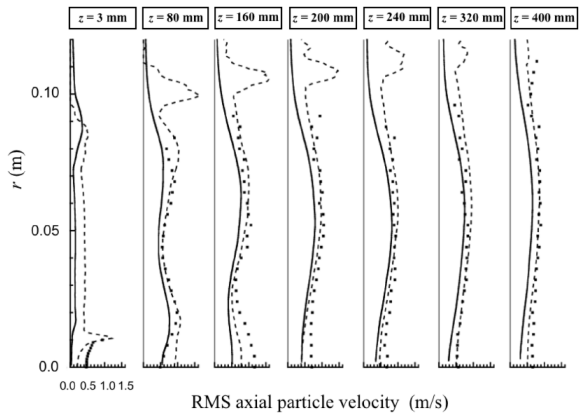


Figure 10: Radial profiles of RMS axial particle velocities at 7 stations along z axis. Symbols: experiment; solid line: EE; dashed line: EL.

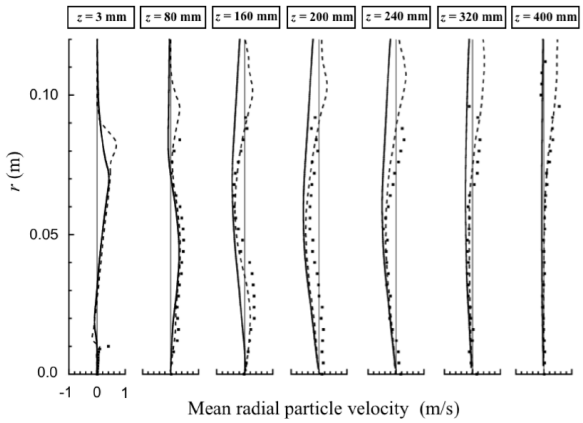


Figure 11: Radial profiles of mean radial particle velocities at 7 stations along z axis. Symbols: experiment; solid line: EE; dashed line: EL.

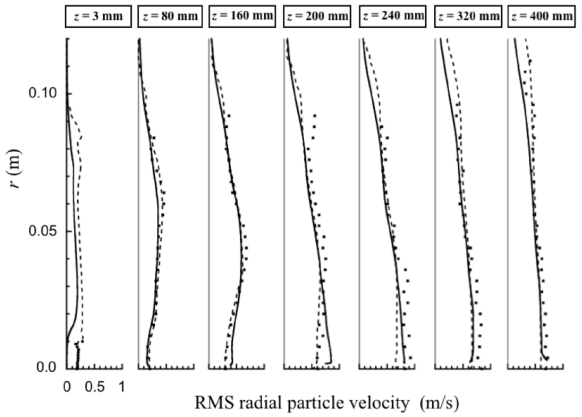


Figure 12: Radial profiles of RMS radial particle velocities at 7 stations along z axis. Symbols: experiment; solid line: EE; dashed line: EL.

turning around to escape from the recirculating flows by the sides. Figure 13 shows field of local volume fraction of solid particles for the EE computation. Local droplet accumulation is also observed upstream of the stagnation point within the central jet.

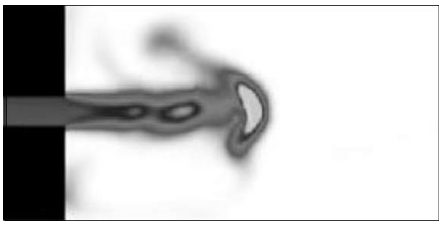


Figure 13: Instantaneous volume fraction in the central plane from Euler-Euler simulation.

This can be quantified by plotting mean velocities along the axis for the gas (Fig. 14) and for the solid particles (Fig. 15). On this axis, both AVBP results match but are slightly off the experimental results. The cause of this discrepancy was investigated through various tests and was identified as the absence of turbulence injected on the gas phase in the inner jet: a direct verification of this effect is that in both computations (EE: solid and EL: dashed lines), the gas and the particle velocities in the central duct increase between $z = -200$ and $z = 0$ mm, indicating that the flow is relaminarizing. This also demonstrates the importance of injecting not only the proper mean profile for the gas velocity but also fluctuations with a reasonably well-defined turbulent spectrum. Additional tests also reveal that the injection of white noise on the particle velocities has a very limited effect on the results.

Figures 16 and 17 display axial profiles of RMS velocities for the gas and the particles. These plots confirm that the position where the maximum levels of gas and particle turbulence are found on the axis is shifted towards the jet inlet and is too intense for both computations.

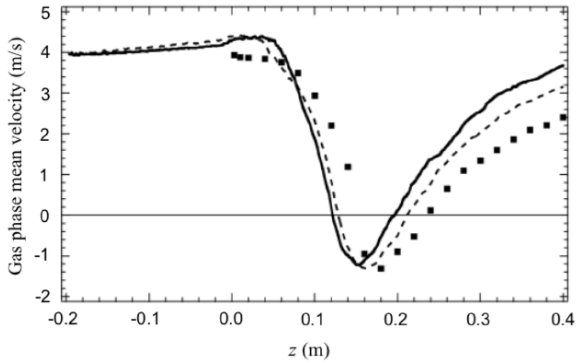


Figure 14: Axial profiles of mean gas velocities. Symbols: experiment; solid line: EE; dashed line: EL.

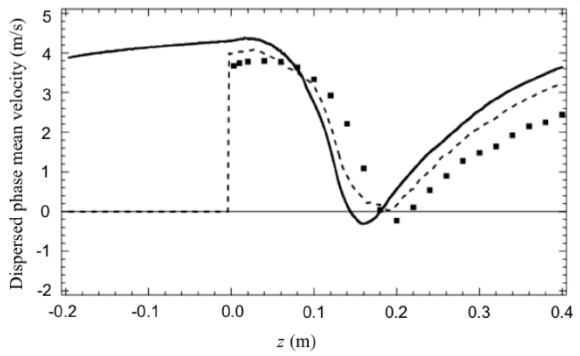


Figure 15: Axial profiles of mean particle velocities. Symbols: experiment; solid line: EE; dashed line: EL.

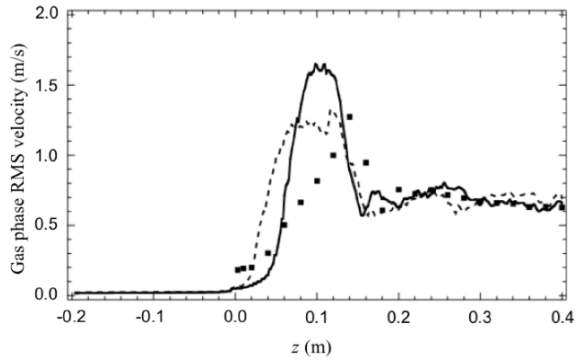


Figure 16: Axial profiles of RMS gas axial velocities. Symbols: experiment; solid line: EE; dashed line: EL.

Analysis of code scalability

In terms of code implementation EE techniques are naturally parallel because the flow and the droplets are solved using the same solver (Kaufmann 2004). On the other hand, the EL approach is not well-suited to parallel computers since two different solvers must be coupled, which increases the complexity of the implementation on a parallel computer. In this case, two methods can be used for LES:

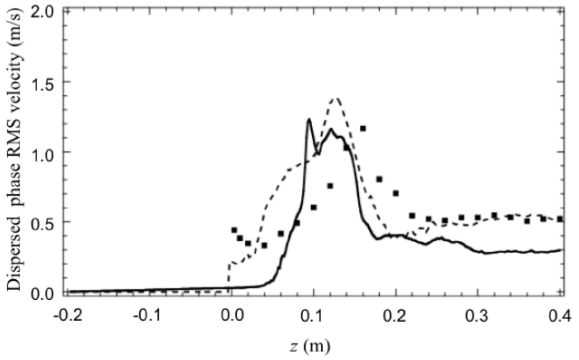


Figure 17: Axial profiles of RMS particle axial velocities. Symbols: experiment; solid line: EE; dashed line: EL.

1. Task parallelization in which some processors compute the gas flow and others compute the droplets flow.
2. Domain partitioning in which droplets are computed together with the gas flow on geometrical subdomains mapped on parallel processors. Droplets must then be exchanged between processors when leaving a subdomain to enter an adjacent domain.

For LES, it is easy to show that only domain partitioning is efficient on large grids because task parallelization would require the communication of very large three-dimensional data sets at each iteration between all processors. However, codes based on domain partitioning are difficult to optimize on massively parallel architectures when droplets are clustered in one part of the domain (typically, near the fuel injectors). Moreover, the distribution of droplets may change during the computation: for a gas turbine reignition sequence, for example, the chamber is filled with droplets when the ignition begins thus ensuring an almost uniform droplet distribution; these droplets then evaporate rapidly during the computation, leaving droplets only in the near injector regions. This leads to a poor speedup on a parallel machine if the domain is decomposed in the same way for the entire computation. As a result, dynamic load balancing strategies are required to redecompose the domain during the computation itself to preserve a high parallel efficiency (Ham *et al.* 2003).

In this section, the scalability of the EL model is analyzed by means of two basic parameters used to measure the efficiency of parallel implementation: the speedup and the reference single-phase CPU time ratio. The former is defined as the ratio between the CPU time of a simulation with 1 processor and the CPU time of a simulation with a given number of processors, N_{procs}:

$$Speedup = \frac{T_{run}(1)}{T_{run}(N_{procs})} \ . \tag{21}$$

The latter is defined as the ratio between the CPU time of a simulation with a given number of procs and the CPU time of the reference single-phase simulation with 1 processor:

$$CPU\ time\ ratio = \frac{T_{run}(N_{procs})}{T_{single-phase}(1)}\ . \qquad (22)$$

Note that the speedup of the EE model can be considered as good as the single-phase computation since the dispersed phase uses the same parallelization applied to the gaseous phase. The EE formulation additional cost is of the order of 80 percent for this test case since the computational cost does not depend on the number of particles.

A scalability study of the EL simulation has been performed in a CRAY XD1 supercomputer at CERFACS for a number of processors up to 64. Table 3-4 and Figs. 18-19 summarize these results for this case (inner jet mass loading of 22 percent) with a total number of particles present in the domain of the order of $600{,}000$.

N_{procs}	1	2	4	8	16	32	64
Ideal scaling	1	2	4	8	16	32	64
Single-phase	1	2.01	4.06	8.2	16.2	32.7	62.5
Two-phase EL	1	1.92	3.85	7.4	13.3	22.9	34.9

Table 3: Summary of the speedup of the EL approach. Supercomputer: CRAY XD1.

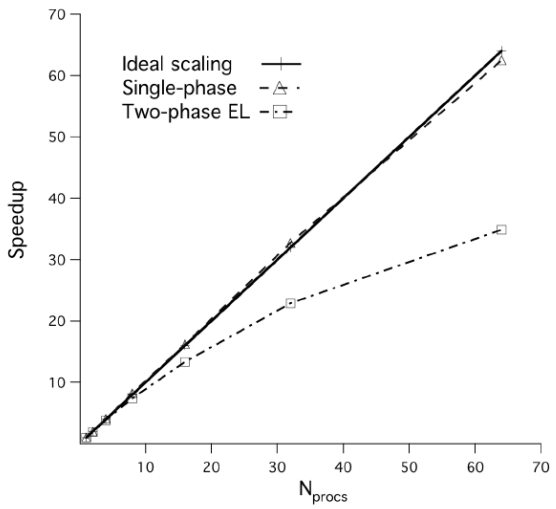


Figure 18: Speedup of the single-phase and the two-phase EL simulation. Supercomputer: CRAY XD1.

The drop of performances shown in Fig 18 is not related to large communications costs between processors as it might be thought at first sight but merely to the parallel load imbalance generated by the partitioning algorithm (Garcia *et al.* 2005). This effect can be observed by plotting the number of nodes, cells and particles presented in each processor. Figure 20 reports the number of nodes and cells presented per processor for a 32-partition simulation by using a recursive inertial bisection (RIB) partitioning algorithm. It shows an excellent load-balancing for the gaseous phase: all processors contains about the same number of cells ($\approx 64{,}500/$processor) and nodes ($\approx 13{,}000/$processor). On the other hand, Fig. 21 shows a huge particle load imbalance where one single processor contains almost half the total number of particles of the simulation. This increases significantly the memory requirements ($\approx$ 20 times the number of nodes) and the floating-point operations for this processor. This points out the need of dynamic load balancing for two-phase flow simulations with a Lagrangian approach, for example, by using multi-constraint partitioning algorithms which take into account particle loading on each processor (Ham *et al.* 2003).

N_{procs}	1	2	4	8	16	32	64
Single-phase	1	0.50	0.25	0.12	0.06	0.030	0.016
Two-phase EL	1.05	0.54	0.27	0.14	0.08	0.046	0.030

Table 4: Summary of the CPU time ratios of the EL approach. Supercomputer: CRAY XD1.

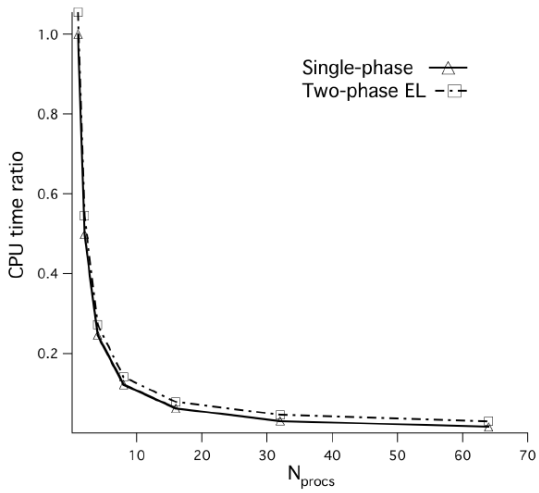


Figure 19: CPU Time ratio of the single-phase and the two-phase EL simulation. Supercomputer: CRAY XD1

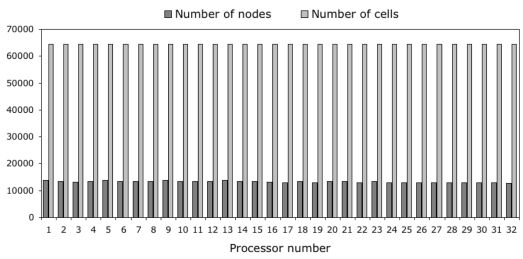


Figure 20: Number of cells and nodes per processor for a 32-partition by using a recursive inertial bisection (RIB) partitioning algorithm.

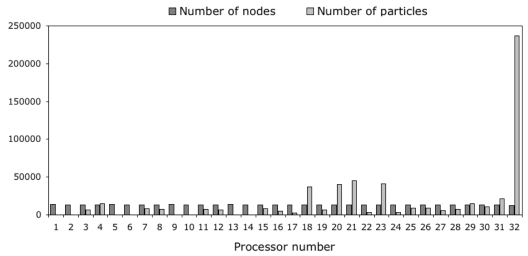


Figure 21: Number of nodes and particles per processor for a 32-partition by using a recursive inertial bisection (RIB) partitioning algorithm.

Conclusions and perspectives

For the present test case (mass loading of 22 percent), the total number of particles present in the domain for the Lagrange codes is of the order of 600,000. For such a small number of particles, the computing power required by the Lagrangian solvers compared to the power required for the gas flow remains low: the additional cost due to the particles is small even with the load balancing problem observed when increasing the number of parallel processors. The EE formulation additional cost (of the order of 80 percent) is independent of the mass loading, so that, for such a dilute case, the EL formulations proved to be faster up to 64 processors.
In terms of results quality, the EL and the EE results implemented into the AVBP solver are very close showing that both formulations lead to equivalent results in this situation. An important factor controlling the quality of the results is the introduction of turbulence on the gas flow in the injection duct: without these turbulent fluctuations, the results are not as good on the axis in terms of positions of the recirculation zones. In addition, the absence of RUV contribution considered in the present case evidences an underestimation of turbulent fluctuations for the EE results to be taken into account in future works. Future developments of the Lagrangian module of the AVBP solver will be devoted to the integration of a particle/mesh load balancing capabilities to improve scalability of the EL simulations.

Acknowledgements

The help of Pr J. Borée in providing the experimental data and analyzing the results is gratefully acknowledged.

References

Apte, S.V., Mahesh, K., Moin, P. & Oefelein, J.C. Large-eddy simulation of swirling particle-laden flows in a coaxial-jet combustor. International Journal of Multiphase Flow, Volume 29, 1311 – 1331 (2003).

Boivin, M., Simonin, O. & Squires, K. Direct numerical simulation of turbulence modulation by particles in isotropic turbulence. Journal of Fluid Mechanics, Volume 375, 235 – 263 (1998)

Boivin, M., Simonin, O. & Squires, K. On the prediction of gas-solid flows with two-way coupling using large eddy simulation. Physics of Fluids, Volume 12, Number 8, 2080 – 2090 (2000).

Borée, J., Ishima, T. & Flour, I. The effect of mass loading and inter-particle collisions on the development of the poly-dispersed two-phase flow downstream of a confined bluff body. Journal of Fluid Mechanics, Volume 443, 129 – 165 (2001)

Caraeni, D., Bergstrom, C. & Fuchs, L. Modeling of liquid fuel injection, evaporation and mixing in a gas turbine burner using large eddy simulation. Flow, Turbulence and Combustion, Volume 65, 223 – 244 (2000)

Chapman, S. & Cowling, T. The Mathematical Theory of Non-Uniform Gases. Cambridge Mathematical Library Edition. Cambridge University Press (1939) (digital reprint 1999)

Colin, O., Ducros, F., Veynante, D. & Poinsot, T. A thickened flame model for large eddy simulations of turbulent premixed combustion. Physics of Fluids, Volume 12, Number 7, 1843 – 1863 (2000)

Colin, O. & Rudgyard, M. Development of high-order Taylor-Galerkin schemes for unsteady calculations. Journal of Computational Physics, Volume 162, Number 2, 338 – 371 (2000)

Druzhinin, O.A. & Elghobashi, S. On the decay rate of isotropic turbulence laden with microparticles. Physics of Fluids, Volume 11, Number 3, 602 – 610 (1999)

Fede, P. & Simonin, O. Numerical Study of Subgrid Fluid Turbulence Effects on the Statistics of Heavy Colliding Particles. Physics of Fluids, Volume 18, 045103 (17 pages) (2006)

Février, P. & Simonin, O. Development and Validation of a Fluid and Particle Turbulent Stress Transport Model in Gas-Solid Flows. Proc. 9th Workshop on Two-Phase Flow Predictions, Merseburg, M. Sommerfeld (Editor), 77 – 85 (1999)

Février, P., Simonin, O. & Squires, K. Partitioning of particle velocities in gas-solid turbulent flows into a continuous field and a spatially uncorrelated random distribution: Theoretical formalism and numerical study. Journal of Fluid Mechanics, Volume 533, 1 – 46 (2005)

García, M., Sommerer, Y., Schönfeld T. & Poinsot, T. Evaluation of Euler-Euler and Euler-Lagrange strategies for Large-Eddy Simulations of turbulent reacting flows. ECCOMAS thematic conference on computational combustion, Lisbon, Portugal (2005)

Gatignol, R. The Faxén formulae for a rigid particle in an unsteady non-uniform Stokes flow. Journal de mécanique théorique et appliquée, Volume 1, Number 2, 143 – 160 (1983)

Ham, F., Apte, S., Iaccarino, G., Wu, X., Herrmann, M., Constantinescu, G., Mahesh, K. & Moin, P. Unstructured LES of reacting multiphase flows in realistic gas turbine combustors.

In Annual Research Briefs. Center for Turbulence Research, NASA Ames/Stanford Univ, 139 – 160 (2003)

Ishima, T., Borée, J., Fanouillère, P. & Flour, I., Presentation of a two phase flow data based obtained on the flow loop Hercule. Proc. 9th Workshop on Two-Phase Flow Predictions, Merseburg, M. Sommerfeld (Editor), 3 – 18 (1999)

Kaufmann, A., Simonin, O., Cuenot, B., Poinsot, T. & Helie, J. Dynamics and dispersion in 3D unsteady simulations of two phase flows. In Supercomputing in Nuclear Applications. Paris: CEA. (2003)

Kaufmann, A. Vers la simulation des grandes échelles en formulation Euler/Euler des écoulements réactifs diphasiques. PhD thesis (2004)

Kaufmann, A., Hélie, J., Simonin, O., & Poinsot, T. Comparison between Lagrangian and Eulerian Particle Simulations Coupled with DNS of Homogeneous Isotropic Decaying Turbulence. Proceedings of the Estonian Academy of Sciences. Engineering, Volume 11, Number 2, 91 – 105 (2005)

Maxey, M. & Riley, J. Equation of motion for a small rigid sphere in a nonuniform flow. Physics of Fluids, Volume 26, Number 4, 883 – 889 (1983)

Moin, P., Squires, K., Cabot, W. & Lee, S. A dynamic subgrid-scale model for compressible turbulence and scalar transport. Physics of Fluids A, Volume 3, Number 11, 2746 – 2757 (1991)

Moreau, M., Bédat, B. & Simonin, O. A priori testing of subgrid stress models for euler-euler two-phase LES from euler-lagrange simulations of gas-particle turbulent flow. In 18th Annual Conference on Liquid Atomization and Spray Systems. ILASS Americas. (2005)

Moureau, V., Lartigue, G., Sommerer, Y., Angelberger, C., Colin, O. & Poinsot, T. High-order methods for dns and les of compressible multicomponent reacting flows on fixed and moving grids. Journal of Computational Physics, Volume 202, Number 2, 710 – 736 (2005)

Poinsot, T. & Veynante, D. Theoretical and numerical combustion. R.T. Edwards, 2nd edition (2005)

Reeks, M. W. On a kinetic equation for the transport of particles in turbulent flows. Physics of Fluids A, Volume 3, 446 – 456 (1991)

Riber, E., Moreau, M., Simonin, O. & Cuenot, B. Towards large eddy simulations of non-homogeneous particle laden turbulent gas flows using euler-euler approach. In 11th Workshop on Two-Phase Flow Predictions. Merseburg, Germany. (2005)

Riber, E., Moreau, M., Simonin, O. & Cuenot, B. Development of Euler-Euler LES Approach for Gas-Particle Turbulent Jet Flow. In Proceedings of Symposium Fluid-Particle Interactions in Turbulence. ASME Joint U.S. European Fluids Engineering Summer Meeting. Miami, FEDSM2006-98110 (2006)

Roux, S., Lartigue, G., Poinsot, T., Meier, U. & Bérat, C. Studies of mean and unsteady flow in a swirled combustor using experiments, acoustics analysis and large eddy simulations. Combustion and Flame, Volume 141, 40 – 54 (2005)

Schiller, L. & Nauman, A. A drag coefficient correlation. V.D.I. Zeitung, Volume 77, 318 – 320 (1935)

Schmitt, P., Poinsot, T., Schuermans, B. & Geigle, K. Large-eddy simulation and experimental study of heat transfer, nitric oxide emissions and combustion instability in a swirled turbulent high pressure burner. Journal of Fluid Mechanics, Volume 570, 17 – 46 (2007)

Selle, L., Lartigue, G., Poinsot, T., Koch, R., Schildmacher, K.-U., Krebs, W., Prade, B., Kaufmann, P. & Veynante, D. Compressible large-eddy simulation of turbulent combustion in complex geometry on unstructured meshes. Combustion and Flame, Volume 137, Number 4, 489 – 505 (2004)

Simonin, O., Février, P., & Laviéville, J. On the Spatial Distribution of Heavy-Particle Velocities in Turbulent Flow : from Continuous Field to Particulate Chaos Journal of Turbulence, Volume 3, 040 (2002)

Vermorel, O., Bédat, B., Simonin, O. & Poinsot, T. Numerical study and modelling of turbulence modulation in a particle laden slab flow. Journal of Turbulence, Volume 4, Number 025, 1 – 39 (2003)

Vreman, B., Geurts, B. & Squires, K. Subgrid modeling in les of compressible flow. Applications Scientific Research, Volume 54, 191 – 203 (1995)

Wang, Q. & Squires, K. Large eddy simulation of particle-laden turbulent channel flow . Physics of Fluids, Volume 8, Number 5, 1207 – 1223 (1996)

Yamamoto, Y., Potthoff, M., Tanaka, T., Kajishima, T. & Tsuji, Y. Large-eddy simulation of turbulent gas-particle flow in a vertical channel: effect of considering inter-particle collisions. Journal of Fluid Mechanics, Volume 442, 303 – 334 (2001)

Yoshizawa, A. Statistical theory for compressible turbulent shear flows, with the application to subgrid modeling. Physics of Fluids, Volume 29, Number 7, 2152 – 2164 (1986)

Appendix D

AIAA Journal, Vol. 46, N° 7, pp. 1773-1781 (July 2008)

Croissance des erreurs d'arrondi et reproductibilité de la Simulation aux Grandes Échelles

La Simulation aux Grandes Échelles (LES) est devenue un outil très efficace pour la prédiction des écoulements turbulents réactifs et non-réactifs. La grande force de la LES par rapport à la méthode classique par résolution des équations de Navier-Stokes moyennées (RANS) est que, comme dans le cas de la Simulation Numérique Directe (DNS), la LES résout les grandes structures de la turbulence au lieu de les modéliser. Cette caractéristique implique que, comme pour la DNS, la LES est sujette au phénomène de séparation des trajectoires : la solution de l'écoulement obtenue par la LES est très sensible à des petites perturbations de l'état de référence. Cela a pour conséquence de limiter le temps de prédictibilité (temps pour qu'une perturbation appliquée sur les petites échelles de la turbulence ait un effet sur les grandes échelles) de la LES/DNS puisque les conditions initiales d'une simulation visant à reproduire un phénomène naturel sont toujours affectées par les incertitudes de mesure.

Un aspect souvent ignoré est que les perturbations numériques qui se produisent en LES conduisent à un autre problème de prédictibilité : des simulations faites avec des conditions initiales strictement identiques peuvent pourtant produire des résultats différents au bout d'un certain temps. Ces perturbations numériques ont différentes sources :

- Les erreurs d'arrondi sont la première source de bruit aléatoire dans n'importe quel calcul en précision finie dans le sens où ils constituent un élément de forçage inévitable pour les équations de Navier-Stokes et peuvent conduire à une variabilité importante des résultats. L'étude de la croissance d'erreur dans les calculs en précision finie est un sujet important en mathématiques appliquées, mais n'a trouvé que peu d'applications dans la mécanique des fluides multidimensionnelle en raison de la complexité des solveurs utilisés en dynamique des fluides numérique (CFD).
- En raison des grandes ressources informatiques qu'ils requièrent, les calculs LES modernes reposent en grande partie sur le calcul parallèle. Dans la plupart des cas, ces codes parallèles utilisent des techniques de découpage de domaine. Le traitement des interfaces entre

processeurs peut alors être une autre source de "bruit" dans les équations de Navier-Stokes. En effet, même dans les codes explicites où l'algorithme est indépendant du nombre de processeurs, un ordre de sommation différent lors de la reconstruction d'une valeur nodale aux interfaces des partitions peut induire des erreurs de non-associativité. Par exemple, dans les codes explicites cell-vertex utilisant des maillages non-structurés, le résidu au nœud est obtenu par addition des résidus pondérés des cellules voisines. Bien évidemment, en arithmétique exacte le résultat de cette addition est indépendant de l'ordre de l'addition, mais ceci n'est plus vrai en arithmétique finie avec les erreurs d'arrondi. Ainsi, les additions de plus de deux termes peuvent produire des résultats différents à cause de l'arithmétique flottante. Par exemple, les erreurs d'arrondi lors des calculs $(a + b) + c$ et $a + (b + c)$ peuvent être différentes, en particulier s'il existe de grandes différences entre les ordres de grandeur des termes de l'addition. Après quelques dizaines de milliers d'itérations, le résultat LES peut en être sensiblement affecté. Étant donné que la propagation de ces erreurs d'arrondi est induite par la réception non-déterministe des messages aux interfaces des partitions, ce type de comportement peut se produire pour tout code CFD parallèle non-structuré, quel que soit le schéma numérique utilisé. En conséquence, les résultats d'une simulation peuvent varier selon le nombre de processeurs utilisés, voir même entre deux exécutions parallèles consecutives sur un même nombre de processeurs. Le cas des codes implicites en temps ou en espace, comme les schémas compacts, n'est pas considéré ici. En effet, pour de tels schémas, les méthodes utilisées pour résoudre un système linéaire à chaque itération dépendent naturellement du nombre de processeurs. Par conséquent, la propagation des erreurs d'arrondi n'est pas la seule raison pour laquelle les solutions obtenues avec des nombres de processeurs différents varient.

- Même sur un calcul séquentiel, les paramètres internes de l'algorithme de découpage peuvent se coupler avec des erreurs d'arrondi pour forcer la solution LES. Par exemple, l'utilisation de l'algorithme Cuthill-McKee (CM) ou de son inverse (RCM) peut mener à une réorganisation différente des nœuds et donc potentiellement produire le même effet qu'une simple perturbation initiale, c'est-à-dire provoquer la divergence de la solution.

- Enfin, les options de compilation, en particulier celles qui touchent à l'optimisation du code et, de toute évidence celles qui affectent les opérations de troncature, sont une quatrième source de variabilité pour la LES. Les tests des options d'optimisation n'ont pas été effectués dans ce travail. Ces tests présenteraient un intérêt certain mais devraient être fait avec précaution car des options trop agressives peuvent directement influer sur l'ordre des opérations et modifier la physique même du calcul, conduisant ainsi à des résultats erronés.

La solution d'un calcul LES/DNS à un instant donné change inévitablement lorsque les erreurs d'arrondi ne sont pas exactement identiques et les solutions LES/DNS sont connues pour n'avoir de sens que d'un point de vue statistique. Pour une utilisation pratique de la LES/DNS, cela n'est pas sans difficulté car cela signifie que l'exécution de la même simulation sur deux machines différentes ou sur la même machine mais avec un nombre différent de processeurs est équivalent à l'introduction d'une perturbation des conditions initiales et peut donc conduire après un certain temps à des résultats instantanés différents. Dans le cas d'écoulements stationnaires en moyenne, les statistiques ne devraient pas dépendre de ces changements et les profils moyens devraient être identiques. Cependant, lorsque l'objectif de la LES est l'étude de phénomènes instationnaires tel que l'allumage dans une chambre de combustion, savoir que les résultats dépendent de ces paramètres est non seulement troublant mais également un inconvénient en terme d'exploitation industrielle.

Cet article traite de ces problématiques et tente de répondre à plusieurs questions simples d'intérêt pour tout utilisateur de la LES : comment la solution instantanée d'un calcul LES dépend-elle du nombre de processeurs utilisés ? Et de la condition initiale ? Et des détails de l'algorithme ?

La première section donne un exemple des effets du nombre de processeurs dans un cas simple : un écoulement turbulent dans un canal rectangulaire calculé avec un code explicite de LES. Cet exemple montre que même dans un code explicite, faire deux fois le même calcul sur une même machine, avec un nombre de processeurs différent peut conduire à l'obtention de solutions instantanées totalement différentes. La deuxième section donne une description systématique des effets des erreurs d'arrondi pour trois types d'écoulements : un écoulement de Poiseuille laminaire, un canal turbulent et une chambre de combustion à géométrie complexe. Dans tous les cas, la différence entre deux solutions instantanées obtenues en changeant le nombre de processeurs, la condition initiale ou la réorganisation des nœuds est quantifiée en terme de normes entre les deux solutions. Les effets du pas de temps et de la précision de la machine (simple, double et quadruple) sont également étudiés dans cette section. Ces résultats montrent que seuls les écoulements turbulents sont caractérisés par une forte sensibilité à ces paramètres. Ces résultats confirment que la LES reflète la véritable nature de la turbulence dans la mesure où elle peut amplifier exponentiellement dans le temps des perturbations infinitésimales des conditions initiales. Ils révèlent également une limitation de la LES en termes de validation et de prédiction des phénomènes instationnaires.

AIAA JOURNAL
Vol. 46, No. 7, July 2008

Growth of Rounding Errors and Repetitivity of Large-Eddy Simulations

Jean-Mathieu Senoner,* Marta García,* Simon Mendez,* Gabriel Staffelbach,† and Olivier Vermorel†
Centre Européen de Recherche et de Formation Avancée en Calcul Scientifique, 31057 Toulouse Cedex 01, France
and
Thierry Poinsot‡
Institut de Mécanique des Fluides, 31400 Toulouse, France

DOI: 10.2514/1.34862

This paper studies the propagation of rounding errors in large-eddy simulation and shows that instantaneous flowfields produced by large-eddy simulation are partially controlled by these rounding errors and depend on multiple parameters: number of processors used for parallel simulation (even in an explicit code), changes in initial conditions (even of the order of machine accuracy), machine precision (simple, double, or quadruple), etc. Using a laminar Poiseuille pipe flow, a fully developed turbulent channel flow, and a complex burner geometry as test cases, results show that only turbulent flows exhibit a high sensitivity to these parameters. These results confirm that large-eddy simulation reflects the true nature of turbulence insofar as it may exponentially amplify infinitely small perturbations on initial conditions in time. However, they highlight an often overlooked limitation of large-eddy simulation in terms of validation and prediction of unsteady phenomena.

Introduction

LARGE-EDDY simulation (LES) has become the most efficient tool to predict nonreacting [1,2], as well as reacting, turbulent flows [3–8]. The main strength of LES compared with classical Reynolds-averaged methods is that, like direct numerical simulation (DNS) [9–11], LES explicitly captures large-scale unsteady motions due to turbulence, instead of modeling them. This feature implies that like DNS, LES is also subject to the separation of trajectories [12,13]: the flow solution exhibited by LES is very sensitive to any small perturbations of a given reference state. This limits the predictability time of LES/DNS because the initial conditions of a simulation aiming to reproduce a natural phenomenon are always affected by measurement uncertainties, and the determination of predictability times of numerical simulations has been an important field of investigation [14,15]. An often ignored aspect is that the numerical perturbations occurring in LES lead to another predictability issue: simulations started with strictly identical initial conditions may yield different results after a certain time. These numerical perturbations have different sources:

1) Rounding errors are the first source of random noise in any finite precision computation; they constitute an unavoidable forcing for the Navier–Stokes equations and may lead to LES variability. The study of error growth in finite precision computations is an important topic in applied numerical mathematics [16–19] but has found few applications in multidimensional fluid mechanics because of the complexity of the codes used in computational fluid dynamics (CFD).

2) Because of its large computational resource requirements, modern LES heavily relies on parallel computing. Therefore, in codes using domain partitioning, i.e., most of them, the treatment of interfaces is an additional "noise" source in the Navier–Stokes equations. Even in explicit codes, where the algorithm is independent of the number of processors, the different summation orders with which a nodal value is reconstructed at partition interfaces may induce nonassociativity errors. For example, in explicit codes on unstructured meshes using cell vertex methods [20], the residual at one node is obtained by adding the weighted residuals of the surrounding cells. Of course, the exact result of this addition is independent of the addition ordering but this is not true for rounding errors. Therefore, additions of more than two summands may yield distinct results for floating-point accumulation. For example, the rounding errors in $(a + b) + c$ and in $a + (b + c)$ may be different, in particular if there are large differences in orders of magnitude between the summands [21] and, after a few tens of thousands time steps, the LES result may be affected. Because the propagation of these rounding errors is induced by nondeterministic message arrival at partition interfaces, such behavior may occur for any unstructured parallel CFD code, regardless of the numerical schemes used. As a consequence, the simulation output might change when run on a different number of processors. The case of implicit codes in time [2,3,22] or in space, such as compact schemes [23–25], is not considered here; for such schemes, the methods [26,27] used to solve the linear system appearing at each time step depend on the number of processors. Therefore, the propagation of rounding errors is not the only reason why solutions obtained with different numbers of processors vary.

3) Even on a single processor computation, internal parameters of the partitioning algorithm may couple with rounding errors to force the LES solution. For example, a different reordering of nodes using the Cuthill–McKee (CM) or the reverse Cuthill–McKee (RCM) algorithm [28,29] may produce the same effect as a simple perturbation and can be the source of solution divergence.

4) Finally, compilation options, in particular those affecting code optimization, and obviously those affecting truncation operations, are a fourth source of LES variability. Different optimization options of the compiler are not tested in the following. Such tests would certainly be of interest, although care must be taken because too

Received 28 September 2007; revision received 21 February 2008; accepted for publication 25 February 2008. Copyright © 2008 by the American Institute of Aeronautics and Astronautics, Inc. All rights reserved. Copies of this paper may be made for personal or internal use, on condition that the copier pay the $10.00 per-copy fee to the Copyright Clearance Center, Inc., 222 Rosewood Drive, Danvers, MA 01923; include the code 0001-1452/08 $10.00 in correspondence with the CCC.

*Ph.D. Student, Computational Fluid Dynamics Team.
†Senior Research Fellow, Computational Fluid Dynamics Team.
‡Research Director, Institut National Polytechnique de Toulouse and Centre National de la Recherche Scientifique. Associate Fellow AIAA.

aggressive optimization options can affect scheduling of operations in a physically wrong manner and lead to erroneous results.

The solution of a given LES/DNS at a certain instant unavoidably changes when the rounding errors are not exactly identical, and LES/DNS solutions are known to have a meaning only in a statistical manner [30]. It is, however, a real difficulty in the practical use of LES/DNS because it means that running the same simulation on two different machines or one machine with a different number of processors is equivalent to a perturbation of initial conditions and can lead to different instantaneous results after a certain time. For steady flows in the mean, statistics should not depend on these changes and mean profiles should be identical. However, when the objective of the LES is the study of unsteady phenomena, such as ignition or quenching in a combustor [31,32], knowing that results depend on these parameters is certainly a sobering thought and a drawback in terms of industrial exploitation. This paper tries to address these issues and answer a simple question which is of interest for all practitioners of LES: How does the instantaneous solution produced by LES depend on the number of processors used to run the simulation? On the initial condition? On internal details of the algorithm?

The first section gives an example of the effects of the number of processors in a simple case: a rectangular turbulent channel flow computed with a fully explicit LES code [33]. This example shows that even in an explicit code, running a simulation twice on a different number of processors can lead to totally different instantaneous solutions. The second section then gives a systematic description of the effects of rounding errors in three flows: a turbulent channel flow, a laminar Poiseuille flow, and a complex burner flow. For all cases, the difference between two instantaneous solutions obtained by changing either the number of processors, the initial condition, or the node ordering is quantified in terms of norms between the two solutions. The effects of time step and machine precision (simple, double, and quadruple) are also investigated in this section. All simulations have been performed on an IBM JS21 supercomputer.

The numerical solver uses a cell vertex formulation, i.e., the discrete values of the conserved variables are stored at the cell vertices, whereas fluxes are obtained by averaging along the cell edges [20]. A compact conservative formulation of the compressible Navier–Stokes equations is considered:

$$\frac{\partial \mathbf{w}}{\partial t} + \nabla \cdot \mathcal{F} = \mathbf{0} \tag{1}$$

where $\mathbf{w}$ represents the vector of conservative variables, and $\mathcal{F}$ is the tensor of both viscous and inviscid fluxes. The Green–Gauss theorem is used to compute the numerical residual in each computational cell K_e:

$$\mathbf{r}_e := \frac{1}{V_e} \oint_{\partial K_e} \mathcal{F}_h \cdot \mathbf{n} \, \mathrm{d}S \tag{2}$$

where $\mathcal{F}_h$ and V_e, respectively, denote a suitable numerical approximation of the flux $\mathcal{F}$ and the volume of the computational cell K_e.

In the semidiscrete form, the scheme then writes:

$$\frac{\partial \mathbf{w}_j}{\partial t} = \frac{1}{V_j} \sum_{e \in D_j} \mathbf{D}_{j,e} V_e \mathbf{r}_e \tag{3}$$

where $\mathbf{D}_{j,e}$ is the distribution matrix that weighs the residual of cell K_e to the node j, and thus depends on the numerical scheme. V_j is the volume of the dual cell associated with the node j. The spatial discretization described here can be combined to explicit time-stepping approaches, such as Runge–Kutta, to obtain a fully discretized scheme. The scheme used here for all simulations is the Lax–Wendroff scheme [33,34]. The fully discretized scheme writes

$$\frac{\mathbf{w}_j^{n+1} - \mathbf{w}_j^n}{\Delta t} = \frac{1}{V_j} \sum_{e \in D_j} \left(\frac{1}{n_v^e} \mathbf{I} + \frac{\Delta t}{2 V_e} \mathcal{A}_e^n \cdot \mathbf{n}_{j,e} \right) V_e \mathbf{r}_e \tag{4}$$

where n_v^e, $\mathcal{A}_e^n$, and $\mathbf{n}_{j,e}$, respectively, denote the number of vertices of the cell K_e, the Jacobian matrices of the cell K_e, and the normal vector associated with the dual cell of the node j on the cell K_e. Additional tests were performed using a third-order scheme in space and time [35], resulting in the same conclusions.

Effects of Number of Processors on Fully Developed Turbulent Channel Flow LES

The first example is the LES of a rectangular fully developed turbulent channel flow with channel dimensions $75 \times 25 \times 50$ mm (Fig. 1). A pressure gradient is applied to a periodic channel flow and random disturbances are added to pass transition to turbulence. There are no boundary conditions except for the walls. The Reynolds number is $Re_\tau = \delta u_\tau / \nu = 1500$, where δ is half the channel height and u_τ the friction velocity at the wall, $u_\tau = (\tau_{\mathrm{wall}}/\rho)^{1/2}$ with τ_{wall} being the wall stress. The mesh contains 30^3 hexahedral elements, and it is not refined at walls where a law-of-the-wall [8] is used. The first grid point is at a reduced distance $y^+ = y u_\tau / \nu \approx 100$ from the wall. The subgrid model is the Smagorinsky model, the value of the constant is $C_S = 0.18$. The Courant–Friedrichs–Lewy (CFL) number λ controlling the time step Δt is $\lambda = \max[(u + c)\Delta t / \Delta]$, where u is the local convective velocity, c the speed of sound, and Δ the mesh size. The chosen value of the CFL number is $\lambda = 0.7$. For all simulations discussed next, the initial condition corresponds to a snapshot of the flow at a given instant, long after turbulence was initialized, so that it is fully established. The domain partitioning method is perfectly equivalent on any number of processors. The recursive inertial bisection [36,37] algorithm has been used to partition the grid, and the Cuthill-McKee algorithm is considered as the default node reordering strategy on subdomains.

Figures 2–4 show fields of axial velocity in the central plane of the channel at three instants after the run initialization. Two simulations performed on, respectively, four (TC1) and eight processors (TC2) with identical initial conditions are compared. The characteristics of all presented simulations are displayed in Tables 1 and 2. The specific times correspond to (in wall units) $t^+ = 7.68$, $t^+ = 18.43$, and $t^+ = 26.11$, respectively, where $t^+ = u_\tau t / \delta$. Obviously, the two flowfields observed at $t^+ = 7.68$ are identical. However, at $t^+ = 18.43$, differences start to become visible. Finally, at $t^+ = 26.11$, the instantaneous flowfields obtained in TC1 and TC2 are totally different. Even though the instantaneous flowfields are different, statistics remain the same: mean and root mean square axial velocity profiles averaged over $t^+ \approx 60$ are identical for both simulations, as can be seen in Figs. 5 and 6.

This very simple example illustrates the main question of the present work: Is the result of Figs. 2–4 reasonable? If it is not a simple programming error (the next section will show that it is not), can other parameters produce similar effects?

Sensitivity of Laminar and Turbulent Flows to Small Perturbations

To understand how LES can produce diverging instantaneous results such as those shown in the previous section, simple tests were performed to investigate the effects of various aspects of the methodology: 1) laminar/turbulent baseline flow, 2) number of processors, 3) initial condition, 4) node ordering, 5) time step,

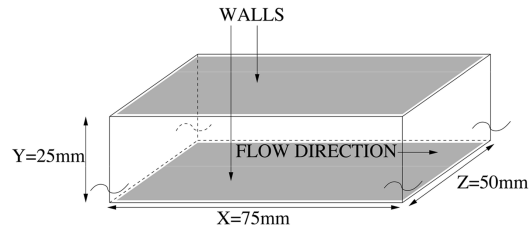
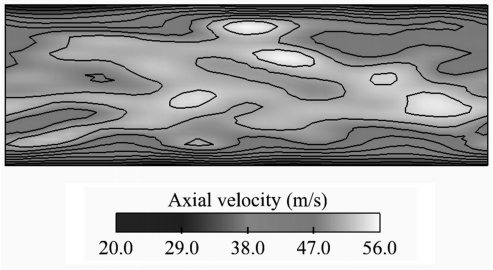
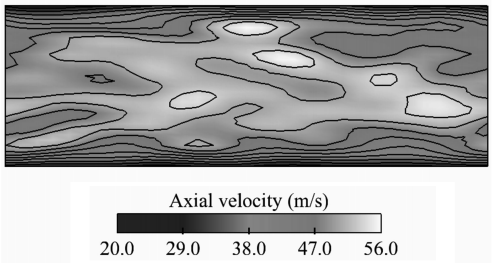


Fig. 1 Schematic of periodic channel. Upper and lower boundaries consist of walls, all other boundaries are pairwise periodic.

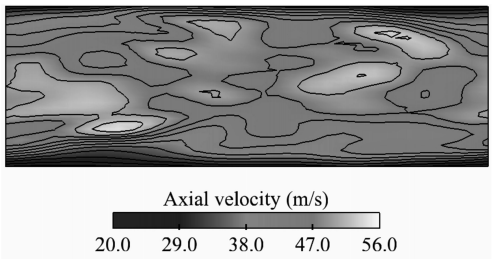


a) Run with 4 processors

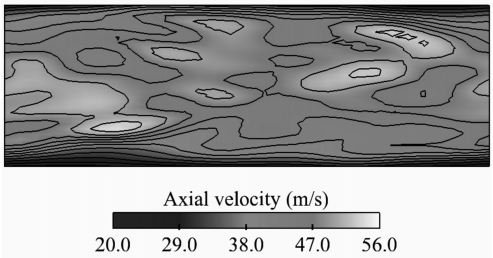


b) Run with 8 processors

Fig. 2 Instantaneous field of axial velocity in the central plane of the channel at $t^+ = 7.68$: a) run TC1, b) run TC2.

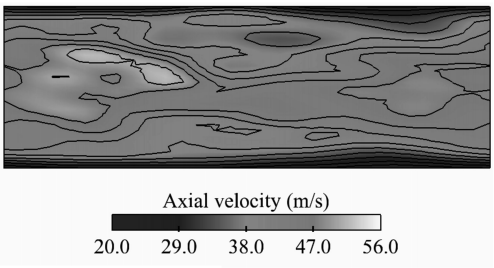


a) Run with 4 processors

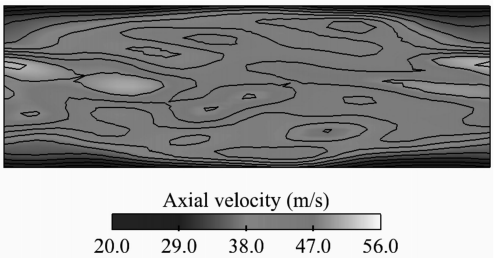


b) Run with 8 processors

Fig. 3 Instantaneous field of axial velocity in the central plane of the channel at $t^+ = 18.43$: a) run TC1, b) run TC2.



a) Run with 4 processors



b) Run with 8 processors

Fig. 4 Instantaneous field of axial velocity in the central plane of the channel at $t^+ = 26.11$: a) run TC1, b) run TC2.

6) floating-point representation according to the Institute of Electrical and Electronics Engineers (IEEE) standard, and 7) investigation of a more realistic configuration with nonperiodic boundary conditions.

For these tests, the objective is to quantify the differences between two LES solutions produced by a couple of simulations in Tables 1 and 2. Let u_1 and u_2 be the axial velocity components of two given instantaneous solutions at the same instant after initialization. A proper method to compare the latter is to use the following norms:

$$N_{\max} = \max(|u_1(\mathbf{x}) - u_2(\mathbf{x})|) \quad \text{for } \mathbf{x} \in \Omega \tag{5}$$

$$N_{\text{mean}} = \left(\frac{1}{V_\Omega}\int_\Omega [u_1(\mathbf{x}) - u_2(\mathbf{x})]^2 \, \mathrm{d}\Omega\right)^{\frac{1}{2}} \quad \text{for } \mathbf{x} \in \Omega \tag{6}$$

where Ω and V_Ω, respectively, denote the computational domain and its volume. Both norms are expressed in meters per second. $N_{\max}$ provides the maximum local axial velocity difference in the field between two solutions, whereas N_{mean} yields a volumetrically averaged axial velocity difference between the two solutions. The growth of $N_{\max}$ and N_{mean} vs the number of time steps will be used as a direct indicator for the divergence of the solutions.

Fully Deterministic LES?

First, it is useful to indicate that performing any of the LES of Table 1 twice on the same machine with the same number of

Table 1 Summary of LES runs (fully developed turbulent channel flow)

Run ID	No. processors	Initial conditions	Precision	Graph ordering	CFL λ
TC1	4	Fixed	Double	CM	0.7
TC2	8	Fixed	Double	CM	0.7
TC3	1	Fixed	Double	CM	0.7
TC4	1	Modif.	Double	CM	0.7
TC5	1	Fixed	Double	RCM	0.7
TC6	4	Fixed	Double	CM	0.35
TC7	8	Fixed	Double	CM	0.35
TC8	4	Fixed	Simple	CM	0.7
TC9	8	Fixed	Simple	CM	0.7
TC10	28	Fixed	Quadr.	CM	0.7
TC11	32	Fixed	Quadr.	CM	0.7

Table 2 Summary of laminar runs (Poiseuille flow)

Run ID	No. processors	Initial conditions	Precision	Graph ordering	CFL λ
LP1	4	Fixed	Double	CM	0.7
LP2	8	Fixed	Double	CM	0.7

processors, the same initial conditions, and the same partition algorithm, leads to exactly the same solution: N_{max} and N_{mean} being zero to machine accuracy. In that sense, the LES remains fully deterministic. However, this is true only if the order of operations at interfaces is not determined by the order of message arrival so that summations are always carried out in the same order. Otherwise, the randomness induced by the nondeterministic order of message arrival is enough to induce diverging solutions. Note that nondeterministic message arrival is usually implemented in parallel codes to improve performance, and that blocking messages order can severely affect the overall simulation cost.

Influence of Turbulence

The first test is to compare a turbulent channel flow studied in the previous section and a laminar flow. A three-dimensional Poiseuille flow was used as a test case. The Poiseuille computation is performed on a pipe geometry with 361×26 points. The flow is laminar and the Reynolds number based on the bulk velocity and diameter is approximately 500. The boundary conditions are set periodic at the inlet/outlet and no slip at the duct wall; a constant axial pressure gradient is imposed in the entire domain. Run parameters of the laminar Poiseuille flow are displayed in Table 2.

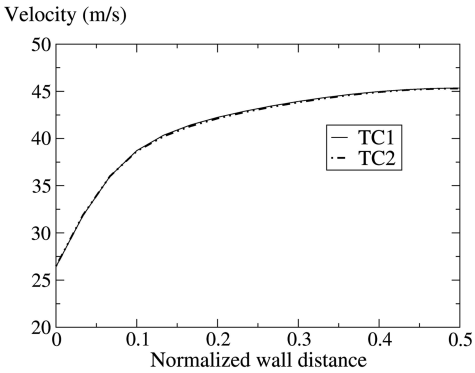


Fig. 5 Comparison of the mean velocity profiles for TC1 (4 processors) and TC2 (8 processors) over half-channel height.

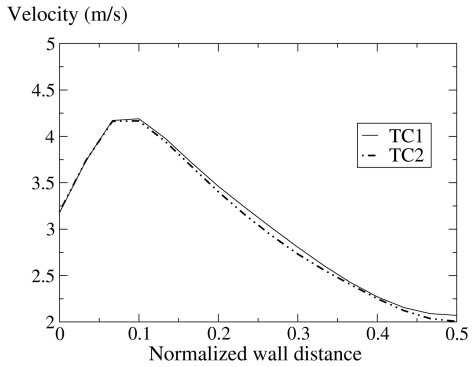


Fig. 6 Comparison of the root mean square velocity profiles for TC1 (4 processors) and TC2 (8 processors) over half channel height.

Figure 7 shows the evolutions of N_{max} and N_{mean} vs time step for runs TC1/TC2 and LP1/LP2. Note that the first point of the graph is the evaluation of the difference after one time step. The only parameter tested here is a change of the number of processors. As expected from the snapshots of Figs. 2–4, the turbulent channel flow simulations are very sensitive to a change in the number of processors, and the solutions of TC1 and TC2 diverge rapidly, leading to a maximum difference of 20 m/s and a mean difference of 3–4 m/s after 90,000 time steps. On the other hand, the difference between LP1 and LP2 hardly increases and levels off when reaching values on the order of 10^{-12} m/s, despite the periodic boundary conditions. This is expected because there is only one stable solution for the Poiseuille flow for infinite times and, accordingly, laminar flows do not induce exponential divergence of trajectories. However, this simple test case confirms that the turbulent character of the flow is the source of the divergence of solutions. This phenomenon must not be confused with the growth of a hydrodynamic mode, which is induced by the bifurcation in phase space of an equilibrium state of a given physical system. Obviously, such an equilibrium state does not exist for a fully developed turbulent channel flow. Moreover, the stagnation of absolute and mean differences between TC1/TC2 simply implies that after 90,000 time steps solutions have become fully uncorrelated and should not be misinterpreted as the saturation of an exponentially growing mode.

The basic mechanism leading to Figs. 2–4 is that the turbulent flow acts as an amplifier for rounding errors generated by the fact that the mesh is decomposed differently in TC1 and TC2. The source of this difference is the new node reordering obtained for both decompositions. This implies a different ordering when adding the contributions to a cell residual for nodes inside the subdomains, but mainly at partition interfaces. This random noise roughly starts at machine accuracy (Fig. 7) at a few points in the flow and grows continuously if the flow is turbulent.

The growth rate of the differences between solutions in simulations TC1 and TC2 cannot be estimated in a simple manner. A description for the determination of growth rates of trajectory separation in two-dimensional vortical flows is given by Leith [14], and is briefly summarized in the following. A description of vortices as points with associated circulations and negligible viscosity is assumed. Under these hypotheses, a set of linearized ordinary differential equations can be derived to evaluate the time evolution of the distance between two neighboring flowfield trajectories differing by an arbitrary infinitesimal perturbation $\delta(t)$. This system admits

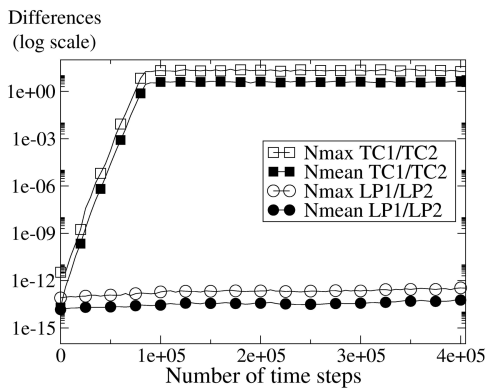


Fig. 7 Differences between solutions vs time step. Squares: turbulent channel flow. Circles: laminar Poiseuille flow.

exponential solutions, the growth rates of which are determined by the real part of the eigenvalues. The evolution of inviscid/conservative systems conserves volume in phase space. As the real part of the eigenvalues describes the separation of trajectories in time, it represents a measure of the evolution of the volume in phase space. Thus, the sum of the real parts vanishes and at least one of them has to be positive. At this stage, the number of degrees of freedom of the system imposes topological constraints on the trajectories and can prevent their separation, but a few degrees of freedom suffice for such systems to exhibit chaotic behavior, as demonstrated by the famous Lorenz attractor [38]. When considering dissipative two- or three-dimensional turbulent flows, a well-defined phase space does not exist. Therefore, predictability analysis is based on the evolution of the energy spectrum of two realizations of a given velocity field differing by a perturbation $\delta(\mathbf{u})$ but having the same statistics. It is possible to derive equations for the evolution of the error energy spectrum and define predictability times using simulations of decaying turbulence [15].

Therefore, a simple estimate of the growth rate from flow parameters a priori does not seem possible. However, one may suppose that, independently of the spatial distribution and amplitude of perturbations applied to a given turbulent flowfield, the separation of trajectories for various simulations yields similar exponential growth rates, which is confirmed in the following. Moreover, it is a purely physical phenomenon and, though induced by rounding errors, the growth rate should not depend on numerical parameters such as machine precision or time step. These aspects are addressed in forthcoming sections.

Influence of Initial Conditions

The previous section has shown that turbulence combined with a different domain partitioning (i.e., a different number of processors) is sufficient to lead to totally different instantaneous flow realizations. It is expected that a perturbation in initial conditions will have a similar effect as domain partitioning. This is verified in runs TC3 and TC4 which are run on a single processor, thereby eliminating issues linked to parallel implementation. The only difference between TC3 and TC4 is that, in TC4, the initial solution is identical to TC3, except at one random point where a single 10^{-16} m/s perturbation is applied to the streamwise velocity component. Simulations with different locations of the perturbation were run to ensure that the position did not affect results.

Figure 8 shows that the growth rate of the difference between TC3 and TC4 is exactly the same as the one observed between TC1 and TC2 (also displayed in Fig. 8): two solutions starting from a very slightly perturbed initial condition diverge as fast as two solutions starting from the same solution but running on different numbers of processors. Note that the difference between runs TC1 and TC2 comes from the accumulation of rounding errors along the interface between subdomains at each time step, whereas TC3 and TC4 differ only through the initial condition: the sequence of floating-point operations is exactly the same in TC3 and TC4. Still, the differences between TC3 and TC4 increase as fast as those between TC1 and TC2; this confirms that a turbulent flow amplifies any difference in the same manner, whether it is due to different sequences of rounding errors or to a perturbation of the initial conditions.

Effects of Node Ordering in Mesh

It has already been indicated that performing the same simulation twice (with the same number of processors and same initial conditions) leads to exactly the same result. However, this is only true as long as exactly the same code is used. It is not verified anymore as soon as a modification affecting rounding errors is done in the code. At this point, so many factors affecting rounding errors can be cited that a general discussion is pointless. This paper focuses on fully explicit codes and on one example only: the order used to add residuals at nodes in a cell vertex scheme. This order is controlled by the developer. For simulation TC5, the ordering of this addition was changed (reverse Cuthill-McKee algorithm); the residual at a given mesh node was assembled by adding the contributions to a cell residual in a different order. This change does not affect the flow data. In TC5, the node residual in a regular tetrahedral mesh is obtained by $1/4\{R_1 + [R_2 + (R_3 + R_4)]\}$ where R_i are the residuals of the cells surrounding the node and by $1/4\{R_4 + [R_3 + (R_2 + R_1)]\}$ in TC3. It has an effect, however, on rounding errors, and the cumulated effects of this nonassociativity error are what this test tries to demonstrate. TC5 and TC3 are performed with the same initial condition and run on one processor only. The only difference is the node reordering strategy.

As shown by Fig. 9, the differences between TC5 and TC3 are again similar to those observed between TC1 and TC2 (obtained by changing the number of processors). This confirms that rounding errors (and not the parallel character of the code) are the source of the solution divergence. It also shows that any modification of the code could lead to such a divergence, suggesting that repeating an LES simulation with a modified code will probably never yield the same instantaneous flowfields, potentially leading to discussions on the validity of the modifications.

Effects of Time Step

It is interesting to verify that numerical aspects do not influence the growth rate of the solutions difference and that the growth rate is only determined by the physical and geometrical parameters of the configuration. On that account, simulations TC6 and TC7 are performed with a time step reduced by a factor of two compared with simulations TC1 and TC2. TC6 and TC7 are carried out on, respectively, four and eight processors. The norms between TC6 and TC7 are displayed in Fig. 10 and compared with the norms between TC1 and TC2. From the preceding explanations, similar growth rates are expected when comparing the growth rates over physical time.

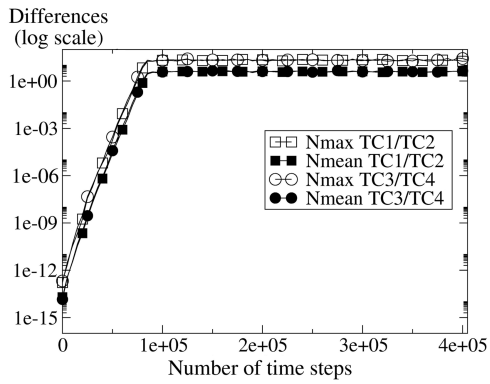


Fig. 8 Differences between solutions vs time step. Squares: different number of processors. Circles: different initial conditions.

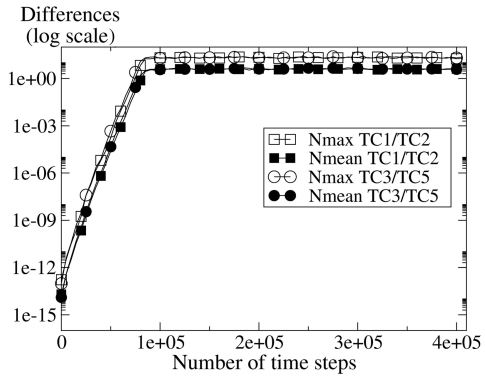


Fig. 9 Differences between solutions vs time step. Squares: different number of processors. Circles: different addition order.

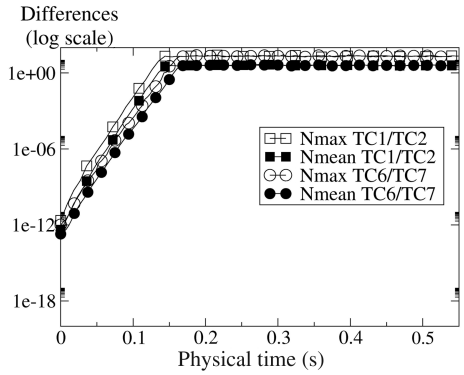


Fig. 10 Differences between solutions vs physical time. Squares: time step Δt. Circles: time step $\Delta t/2$.

The growth rates observed in Fig. 10 are indeed very similar. The slight difference is probably due to the variation of the numerical dispersion and dissipation properties of the scheme with the CFL number [39].

Effects of Machine Precision

A last test to verify that the divergence between solutions is not due to a programming error, but depends primarily on rounding errors, is to perform the same computation with simple/quadruple precision instead of double precision. Simulations TC1 and TC2 were repeated using simple precision in runs TC8 and TC9 (Table 1) and quadruple precision in TC10 and TC11. To compensate for the increase in computational time for quadruple precision simulations, roughly a factor of 10 compared with double precision, TC10 and TC11 are carried out on, respectively, 28 and 32 processors to yield a reasonable computing time. Results are displayed in Fig. 11 and compared with the difference between TC1 and TC2.

Figure 11 shows that the solutions differences for TC8/TC9 and TC10/TC11 roughly start from the respective machine accuracies (differences of 10^{-6} for simple precision after one time step, differences of 10^{-30} for quadruple precision after one time step) and increase exponentially with the same growth rate, before reaching identical difference levels for all three cases. This shows that higher precision computations cannot prevent the exponential divergence of trajectories but only delay it.

Propagation of Rounding Errors on More Realistic Configuration

The previous example corresponds to a periodic turbulent channel flow where perturbations cannot leave the computational domain.

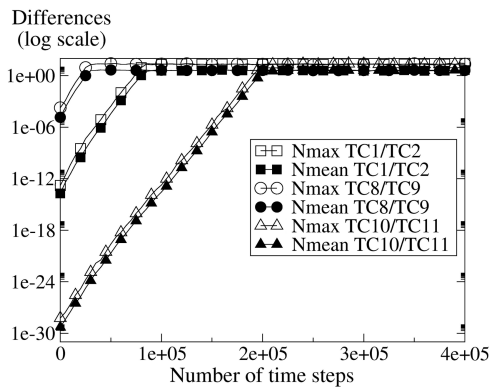


Fig. 11 Differences between solutions vs time step. Squares: double precision. Circles: simple precision. Triangles: quadruple precision.

They can therefore be amplified indefinitely and this might explain the divergence of solutions observed in Figs. 2–11. To verify whether the previously discussed divergence phenomena are independent of the configuration and can also occur in nonperiodic flows, a more realistic configuration is tested.

The chosen configuration is the nonreacting flow in a complex swirled combustor including a plenum, a swirler, and a combustion chamber. The Reynolds number at the inlet of the combustion chamber (based on an equivalent radius of the swirler) is approximately 5200. To avoid the specification of the boundary condition at the outlet of the combustion chamber for further acoustic analysis, the atmosphere around the outlet has also been meshed (Fig. 12). Boundary conditions are summarized in Fig. 13. All solid boundaries are modeled using adiabatic wall laws. The inlet, coflow, and outlet boundary conditions rely on characteristic decomposition according to Moureau et al. [33] and Poinsot and Lele [40]. The inlet injects an air mixture (N_2, O_2) with 15 g/s at $T = 298$ K. The coflow imposed on the left sidewall aims to mimic air entrainment due to the outgoing flow of the combustion chamber. The coflow velocity is set to $u_z = 0.1$ m/s and $T = 298$ K.

The only parameter changing in the two simulations is the number of processors, and the parameters of the run are specified in Table 3.

A divergence of solutions similar to the periodic turbulent channel flow can be observed in Fig. 14.

Instantaneous fields of axial velocity for both runs in the central plane at $t = 32.4$ ms (80,000 time steps) after initialization show that instantaneous flowfields are uncorrelated (Fig. 15). The differences remain mostly confined to the highly turbulent zone at the inlet of the combustion chamber. This region is volumetrically small compared with the entire computed geometry, in particular the atmospheric region, which is why the mean difference curve remains low compared with the turbulent channel flow. The maximal local difference reaches values of 45 m/s.

Figure 16 displays the evolution of axial velocity for the two simulations for a point located at 20 cm of the chamber inlet plane, on the chamber axis. As expected, the signals diverge suddenly due to the exponential amplification of rounding errors, leading to uncorrelated signals after approximately 30 ms ($t = 0$ s corresponds to initialization).

These results clearly demonstrate that rounding errors may propagate until full uncorrelation, even for configurations with inlet/

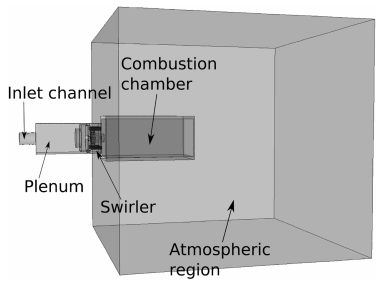


Fig. 12 Schematic of the complex burner geometry.

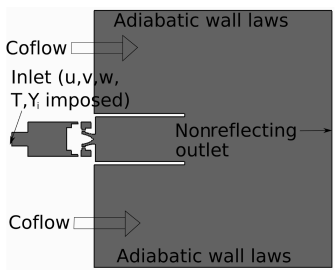


Fig. 13 Midplane cut with boundary-condition specification.

Table 3 Summary of complex burner geometry runs

Run ID	No. processors	Initial conditions	Precision	Graph ordering	CFL λ
CB1	28	Fixed	Double	CM	0.7
CB2	32	Fixed	Double	CM	0.7

outlet boundary conditions, and that it is not an artifact due to the periodicity of the turbulent channel flow (Fig. 1). The presence of recirculation zones in the complex burner geometry, where part of the fluid constantly remains in the computational domain, may be an explanation for this behavior, but this aspect requires further investigation.

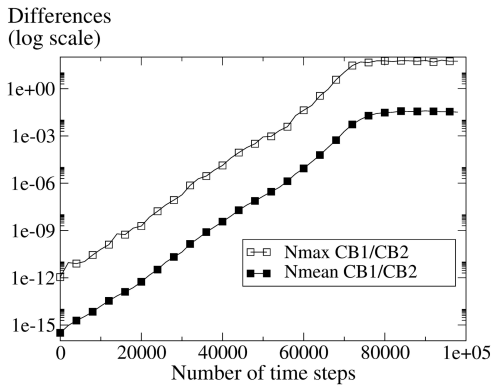


Fig. 14 Differences between CB1 and CB2 (different number of processors) vs time step.

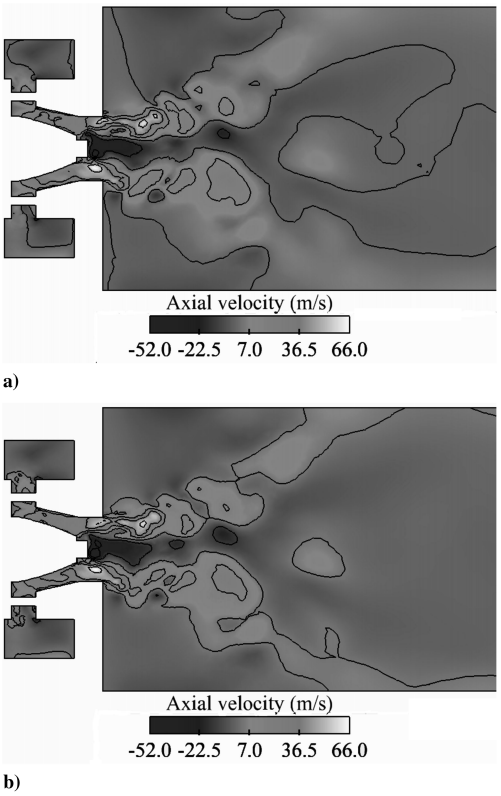


Fig. 15 Instantaneous field of axial velocity in the central plane of the burner at $t = 32.4$ ms: a) run CB1, b) run CB2.

Conclusions

This work focused on the sensitivity of instantaneous large-eddy simulation fields to different rounding error propagation due to situations where various parameters of the run, such as number of processors, initial condition, time step, and changes in addition ordering of cell residuals for cell vertex methods, are modified. The baseline simulation used for the tests was a fully developed periodic turbulent channel flow, but a complex burner geometry displayed a similar behavior. The conclusions are as follows:

1) Any sufficiently turbulent flow computed in LES exhibits significant sensitivity to these parameters, leading to instantaneous solutions which can be totally different. As expected, laminar flows are almost insensitive to these parameters even for periodic simulations.

2) The divergence of solutions is due to two combined facts: 1) the exponential separation of trajectories in turbulent flows, and 2) the different propagation of rounding errors induced by domain partitioning and scheduling of operations. More generally, any change in the code lines affecting the propagation of rounding errors will have a similar effect. This implies that the validation of an LES code after modifications may only be based on statistical fields. This makes error detection in LES codes much more difficult than in classical codes; comparing instantaneous solutions is not a proper validation method for LES.

3) Small changes in initial conditions (of the order of machine accuracy at one point of the flow only) produce similar divergence of solutions.

4) Working with higher precision machines does not suppress the divergence of solutions, but delays it.

Converged statistics reflect the fact that most possible realizations of a turbulent flow have been taken. It is therefore clear that the propagation of rounding errors does not affect statistics of large-eddy and direct numerical simulations. However, instantaneous values may a priori only be used for times during which the differences between two runs with identical initial solution remain negligible with respect to a suitable error norm. The increase in floating-point representation delays the divergence of solutions, but the increase in computational costs appears too severe for practical applications. Another option consists of the use of software which inhibits the

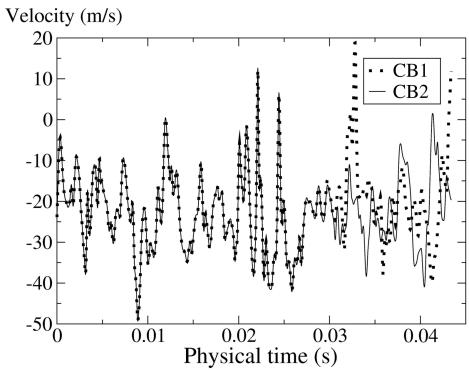


Fig. 16 Evolution of axial velocity at a point located on the chamber axis for CB1 and CB2 over time.

propagation of rounding errors. This would allow the increase of the computational predictability time of a given simulation. The ability to include such software in complex computational fluid dynamics codes must be investigated and the increase in computational expense is again a crucial aspect. More generally, these results demonstrate that the concept of numerical quality in LES will require much more detailed studies and tools than what has been used up to now for Reynolds-averaged simulations.

Acknowledgments

We thank F. Chaitin-Chatelin for helpful discussions. The first author gratefully acknowledges the funding by the European Community in the framework of the Marie Curie Early Stage Research Training Fellowship (contract number MEST-CT-2005-020426).

References

[1] Sagaut, P., *Large Eddy Simulation for Incompressible Flows*, 1st ed., Springer, Berlin, 2002.

[2] Mahesh, K., Constantinescu, G., and Moin, P., "Numerical Method for Large-Eddy Simulation In Complex Geometries," *Journal of Computational Physics*, Vol. 197, No. 1, 2004, pp. 215–240. doi:10.1016/j.jcp.2003.11.031

[3] Di Mare, F., Jones, W. P., and Menzies, K., "Large Eddy Simulation of a Model Gas Turbine Combustor," *Combustion and Flame*, Vol. 137, No. 3, 2004, pp. 278–295. doi:10.1016/j.combustflame.2004.01.008

[4] Poinsot, T., and Veynante, D., *Theoretical and Numerical Combustion*, 2nd ed., R. T. Edwards, Philadelphia, 2005.

[5] Pitsch, H., "Large Eddy Simulation of Turbulent Combustion," *Annual Review of Fluid Mechanics*, Vol. 38, Jan. 2006, pp. 453–482. doi:10.1146/annurev.fluid.38.050304.092133

[6] El-Asrag, H., and Menon, S., "Large Eddy Simulation of Bluff-Body Stabilized Swirling Non-Premixed Flames," *Proceedings of the Combustion Institute*, Vol. 31, No. 2, 2007, pp. 1747–1754. doi:10.1016/j.proci.2006.07.251

[7] Duwig, C., Fuchs, L., Griebel, P., Siewert, P., and Boschek, E., "Study of a Confined Turbulent Jet: Influence of Combustion and Pressure," *AIAA Journal*, Vol. 45, No. 3, 2007, pp. 624–661. doi:10.2514/1.26352

[8] Schmitt, P., Poinsot, T. J., Schuermans, B., and Geigle, K., "Large-Eddy Simulation and Experimental Study of Heat Transfer, Nitric Oxide Emissions and Combustion Instability in a Swirled Turbulent High Pressure Burner," *Journal of Fluid Mechanics*, Vol. 570, 2007, pp. 17–46. doi:10.1017/S0022112006003156

[9] Poinsot, T., Candel, S., and Trouvé, A., "Application of Direct Numerical Simulation to Premixed Turbulent Combustion," *Progress in Energy and Combustion Science*, Vol. 21, No. 6, 1995, pp. 531–576. doi:10.1016/0360-1285(95)00011-9

[10] Moin, P., and Mahesh, K., "DNS: A Tool in Turbulence Research," *Annual Review of Fluid Mechanics*, Vol. 30, Jan. 1998, pp. 539–578. doi:10.1146/annurev.fluid.30.1.539

[11] Vervisch, L., and Poinsot, T., "Direct Numerical Simulation of Non Premixed Turbulent Flames," *Annual Review of Fluid Mechanics*, Vol. 30, Jan. 1998, pp. 655–692. doi:10.1146/annurev.fluid.30.1.655

[12] Tennekes, H., and Lumley, J. L., *First Course in Turbulence*, 1st ed., MIT Press, Cambridge, MA, 1972.

[13] Lesieur, M., *Turbulence in Fluids*, 1st ed., Martinus–Nijhoff, Dordrecht, The Netherlands, 1987, pp. 230–231.

[14] Leith, C., "Atmospheric Predictability and Two-Dimensional Turbulence," *Journal of the Atmospheric Sciences*, Vol. 28, No. 2, 1971, pp. 145–161. doi:10.1175/1520-0469(1971)028<0145:APATDT>2.0.CO;2

[15] Métais, O., and Lesieur, M., "Statistical Predictability of Decaying Turbulence," *Journal of the Atmospheric Sciences*, Vol. 43, No. 9, 1986, pp. 857–870. doi:10.1175/1520-0469(1986)043<0857:SPODT>2.0.CO;2

[16] Stoer, J. S., and Bulirsch, R., *Introduction to Numerical Analysis*, 1st ed., Springer, Berlin, 1980.

[17] Chaitin-Chatelin, F., and Frayssé, V., *Lectures on Finite Precision Computations*, 1st ed., Society for Industrial and Applied Mathematics, Philadelphia, 1996.

[18] Overton, M. L., *Numerical Computing with IEEE Floating Point Arithmetic*, 1st ed., Society for Industrial and Applied Mathematics, Philadelphia, 2001.

[19] Vignes, J., "Discrete Stochastic Arithmetic for Validating Results of Numerical Software," *Numerical Algorithms*, Vol. 37, Nos. 1–4, Dec. 2004, pp. 377–390. doi:10.1023/B:NUMA.0000049483.75679.ce

[20] Schönfeld, T., and Rudgyard, M., "Steady and Unsteady Flow Simulations Using the Hybrid Flow Solver AVBP," *AIAA Journal*, Vol. 37, No. 11, 1999, pp. 1378–1385.

[21] Hanrot, G., Lefévre, V., Stehlé, D., and Zimmermann, P., "Worst Cases for a Periodic Function with Large Arguments," *Proceedings of the 18th IEEE Symposium on Computer Arithmetic*, edited by P. Kornerup, and J.-M. Muller, IEEE Computer Society Press, Los Alamitos, CA, 2007, pp. 133–140.

[22] Freitag, M., and Janicka, J., "Investigation of a Strongly Swirled Premixed Flame Using LES," *Proceedings of the Combustion Institute*, Vol. 31, No. 1, 2007, pp. 1477–1485. doi:10.1016/j.proci.2006.07.225

[23] Lele, S., "Compact Finite Difference Schemes with Spectral-Like Resolution," *Journal of Computational Physics*, Vol. 103, No. 1, 1992, pp. 16–42. doi:10.1016/0021-9991(92)90324-R

[24] Abarbanel, S. S., and Chertock, A. E., "Strict Stability of High-Order Compact Implicit Finite-Difference Schemes: The Role of Boundary Conditions for Hyperbolic PDEs, 1," *Journal of Computational Physics*, Vol. 160, No. 1, 2000, pp. 42–66. doi:10.1006/jcph.2000.6420

[25] Sengupta, T. K., Ganerwal, G., and Dipankar, A., "High Accuracy Compact Schemes and Gibbs' Phenomenon," *Journal of Scientific Computing*, Vol. 21, No. 3, 2004, pp. 253–268. doi:10.1007/s10915-004-1317-2

[26] Saad, Y., "Flexible Inner-Outer Preconditioned GMRES Algorithm," *Journal of Scientific Computing*, Vol. 14, No. 2, 1993, pp. 461–469. doi:10.1137/0914028

[27] Fraysse, V., Giraud, L., and Gratton, S., "Set of Flexible-GMRES Routines for Real and Complex Arithmetics," Centre Européen de Recherche et de Formation Avancée en Calcul Scientifique TR/PA/98/20, 1998.

[28] Cuthill, E., and McKee, J., "Reducing the Bandwidth of Sparse Symmetric Matrices," *Proceedings of the 24th National Conference of the ACM*, Vol. 1, Assoc. for Computing Machinery, New York, 1969, pp. 157–172.

[29] Liu, W.-H., and Sherman, A. H., "Comparative Analysis of the Cuthill-Mckee and the Reverse Cuthill-Mckee Ordering Algorithms for Sparse Matrices," *SIAM Journal on Numerical Analysis*, Vol. 13, No. 2, 1976, pp. 198–213. doi:10.1137/0713020

[30] Pope, S. B., "Ten Questions Concerning the Large-Eddy Simulation of Turbulent Flows," *New Journal of Physics*, Vol. 6, No. 1, 2004, p. 35. doi:10.1088/1367-2630/6/1/035

[31] Sommerer, Y., Galley, D., Poinsot, T., Ducruix, S., Lacas, F., and Veynante, D., "Large Eddy Simulation and Experimental Study of Flashback and Blow-Off in a Lean Partially Premixed Swirled Burner," *Journal of Turbulence*, Vol. 5, No. 1, Article 37, 2004. doi:10.1088/1468-5248/5/1/037

[32] Boileau, M., Staffelbach, G., Cuenot, B., Poinsot, T., and Bérat, C., "LES of an Ignition Sequence in a Gas Turbine Engine," *Combustion and Flame*, 2008 (to be published).

[33] Moureau, V., Lartigue, G., Sommerer, Y., Angelberger, C., Colin, O., and Poinsot, T., "High-Order Methods for DNS and LES of Compressible Multi-Component Reacting Flows on Fixed and Moving Grids," *Journal of Computational Physics*, Vol. 202, No. 2, 2005, pp. 710–736. doi:10.1016/j.jcp.2004.08.003

[34] Hirsch, C., *Numerical Computation of Internal and External Flows: Computational Methods for Inviscid and Viscous Flows*, 1st ed., Vol. 2, Wiley, New York, 1988, pp. 224–306.

[35] Colin, O., and Rudgyard, M., "Development of High-Order Taylor-Galerkin Schemes for Unsteady Calculations," *Journal of Computational Physics*, Vol. 162, No. 2, 2000, pp. 338–371. doi:10.1006/jcph.2000.6538

[36] Williams, R. D., "Performance of Dynamic Load Balancing Algorithms for Unstructured Mesh Calculations," *Concurrency, Practice and Experience*, Vol. 3, No. 5, 1991, pp. 457–481. doi:10.1002/cpe.4330030502

[37] Taylor, V. E., and Nour-Omid, B., "Study of the Factorization Fill-In for a Parallel Implementation of the Finite Element Method," *International Journal for Numerical Methods in Engineering*, Vol. 37, No. 22, 1994,

pp. 3809–3823.
doi:10.1002/nme.1620372205

[38] Lorenz, E. N., "Deterministic Nonperiodic Flow," *Journal of the Atmospheric Sciences*, Vol. 20, No. 2, 1963, pp. 130–141.
doi:10.1175/1520-0469(1963)020<0130:DNF>2.0.CO;2

[39] Hirsch, C., *Numerical Computation of Internal and External Flows: Fundamentals of Numerical Discretization*, 1st ed., Vol. 1, Wiley, New York, 1988, pp. 307–319.

[40] Poinsot, T., and Lele, S., "Boundary Conditions for Direct Simulations of Compressible Viscous Flows," *Journal of Computational Physics*, Vol. 101, No. 1, 1992, pp. 104–129.
doi:10.1016/0021-9991(92)90046-2

P. Givi
Associate Editor

Appendix E

Journal of Computational Physics, Vol. 228, N° 2, pp. 539-564 (2009)

Évaluation des méthodes numériques pour la Simulation aux Grandes Échelles des écoulements diphasiques particulaires réactifs turbulents

La prédiction de la dispersion du carburant dans les écoulements diphasiques turbulents est une partie importante des écoulements réactifs car le carburant est en général injecté sous forme liquide avant d'être brûlé. Même si la simulation aux grandes échelles (LES) a déjà démontré son potentiel sur des flammes monophasiques dans les dernières années, son extension aux flammes diphasiques reste à explorer. Tout d'abord, la physique utilisée pour décrire l'atomisation d'un jet liquide, la dispersion des particules solides, leur interaction avec les parois, leur évaporation et combustion est difficile à modéliser car il s'agit essentiellement de phénomènes de sous-maille. Deuxièmement, l'implantation numérique des écoulements diphasiques pour la LES reste un défi. Enfin, dans le contexte du calcul massivement parallèle, l'efficacité numérique représente une contrainte supplémentaire et le maintien d'une efficacité du parallélisme comparable à celle des calculs gazeux n'est pas si évident pour un calcul diphasique.

Cet article est une compilation d'une partie des résultats de la thèse de Riber [179] concernant le chapitre d'analyses et validation des résultats du calcul gazeux et diphasique monodispersé sur la configuration de Borée *et al.* [22] où des particules de verre sont injectées dans un écoulement turbulent comportant une zone de recirculation. Les résultats des simulations numériques LES avec l'approche Euler/Euler (EE) sont comparés avec les résultats expérimentaux et aussi avec les résultats numériques de deux codes de calculs avec une formulation Lagrangienne. Un de ces codes de calculs contient la version Lagrangienne développée pendant cette thèse.

Cet article est organisé de la façon suivante : d'abord la sélection de la configuration est présentée. L'objectif à court terme est de faire des simulations LES diphasiques sur des géométries complexes avec évaporation et combustion. La validation de la dispersion des particules dans un écoulement dont les caractéristiques sont similaires mais avec une complexité moindre est donc nécessaire. De ce point de vue et avec des applications aéronautiques en perspective, la configuration étudiée par Borée *et al.* [22] est très intéressante. Par ailleurs, elle dispose d'une grande quantité de données de calculs gazeux et diphasiques. Une description du banc expérimental est présentée au début de la section. Elle inclut en même temps de l'information sur les caractéristiques

des écoulements de type bluff body. Le rapport de vitesse entre le tube d'injection et le co-courant va donner lieu à des écoulements très différents. Dans ce cas, l'écoulement a deux points de stagnation dans l'axe avec une large zone centrale de recirculation entre le jet central et le co-courant. Le diamètre des particules de verre est compris entre 20 et 100 microns et leur masse volumique est de 2470 kg/m^3. Les comparaisons présentées dans cet article se limitent au cas faiblement chargé en masse ($M_j = 22\%$) où l'écoulement diphasique considéré est suffisamment dilué pour négliger les collisions interparticulaires. Les principaux objectifs de cet article sont résumés dans cette section :

- étude de l'influence du type des cellules du maillage (hexaèdres vs tétraèdres),
- évaluation de l'influence du schéma numérique,
- comparaison des formulations implicite et explicite pour l'avancement en temps,
- comparaison de la précision des approches Euler/Euler et Euler/Lagrange.

La section 3 commence par une description des deux solveurs de calcul utilisés dans cette étude : le solveur explicite AVBP qui contient les formulations Eulérienne et Lagrangienne pour le traitement de la phase dispersée développé au centre de recherche CERFACS, en France; et le solveur implicite CDP qui traite la phase dispersée avec l'approche Lagrangienne, développé au Centre de Recherche de la Turbulence (CTR) de l'Université de Stanford. Une description détaillée des équations utilisées pour les formulations EE et EL est incluse dans cette section.

La section 4 commence avec la présentation des dimensions du domaine de calcul et détaille les principales caractéristiques des deux maillages sur lesquels ont été effectués les tests de l'influence du type de cellule. Ensuite, les conditions aux limites pour les entrées, les sorties et le traitement aux parois sont présentées. La principale différence étant dans l'injection de turbulence du tube d'amenée. Une fois la présentation des cas tests faite, les résultats des différentes simulations sont analysés en comparant sur sept profils axiaux et radiaux les vitesses moyenne et fluctuante.

La section 5 présente les différentes coupes d'injection de particules dans le tube centrale. Les résultats des calculs diphasiques avec les approches EE et EL sont montrés en comparaison avec les résultats expérimentaux sur sept profils radiaux de vitesses moyenne et fluctuante. Des comparaisons entre les deux codes, avec la même approche (EL), sont ainsi incluses dans cette partie.

Les conclusions sont incluses dans la section 6.

Journal of Computational Physics 228 (2009) 539–564

Contents lists available at ScienceDirect

Journal of Computational Physics

journal homepage: www.elsevier.com/locate/jcp

Evaluation of numerical strategies for large eddy simulation of particulate two-phase recirculating flows

E. Riber [a,b,*], V. Moureau [c], M. García [b], T. Poinsot [a], O. Simonin [a]

[a] IMFT-UMR 5502, allée du Professeur Camille Soula, 31400 Toulouse, France
[b] CERFACS, 42, avenue Gaspard Coriolis, 31057 Toulouse Cedex 01, France
[c] CTR, Stanford University, Stanford, CA 94305-3030, USA

ARTICLE INFO

Article history:
Received 18 December 2007
Received in revised form 5 June 2008
Accepted 1 October 2008
Available online 10 October 2008

Keywords:
LES
Two-phase recirculating flows
Particles
Euler/Lagrange
Euler/Euler

ABSTRACT

Predicting particle dispersion in recirculating two-phase flows is a key issue for reacting flows and a potential application of large eddy simulation (LES) methods. In this study, Euler/Euler and Euler/Lagrange LES approaches are compared in the bluff body configuration from Borée et al. [J. Borée, T. Ishima, I. Flour, The effect of mass loading and inter-particle collisions on the development of the polydispersed two-phase flow downstream of a confined bluff body, J. Fluid Mech. 443 (2001) 129–165] where glass beads are injected into a complex recirculating flow. These tests are performed for non-reacting, non-evaporating sprays but are mandatory validations before computing realistic combustion chambers. Two different codes (one explicit and compressible and the other implicit and incompressible) are also tested on the same configuration. Results show that the gas flow is well predicted by both codes. The dispersed phase is also well predicted by both codes but the Lagrangian approach predicts root-mean-square values more accurately than the Eulerian approach. The effects of mesh, solvers and numerical schemes are discussed for each method.

© 2008 Elsevier Inc. All rights reserved.

1. Introduction

Today, RANS (Reynolds-averaged Navier–Stokes) equations are routinely solved to design combustion chambers, for both gaseous and liquid fuels. Recently, large eddy simulation (LES) has been extended to reacting gaseous flows in order to give access to unsteady phenomena occurring in combustion devices (such as instabilities, flashback or quenching), and to provide better accuracy for the prediction of mean flows. The success of these approaches for gaseous flames in the last years [2–12] is a clear illustration of their potential. LES gives access to the large scales structures of the flow, reducing the importance of modeling, and capturing a significant part of the physics controlling these flames.

Even though LES has already demonstrated its potential for gaseous flames, its extension to two-phase flames is still largely to be done. First, the physical submodels required to describe the atomization of a liquid fuel jet, the dispersion of fuel droplets, their interaction with walls, evaporation and combustion are as difficult to build in LES as in RANS because they are essentially subgrid phenomena. Second, the numerical implementation of two-phase flow LES remains a challenge: the equations for both the gaseous and the dispersed phases must be solved together at each time step in a strongly coupled manner. This differs from classical RANS where the resolution of the two phases can be done in a weak procedure, bringing first the gas flow to convergence, then the solid particles and iterating until convergence of both phases. Finally, in the

* Corresponding author. Address: CERFACS, 42, avenue Gaspard Coriolis, 31057 Toulouse Cedex 01, France. Tel.: +33 561 19 3115; fax: +33 561 19 3000.
E-mail address: riber@cerface.fr (E. Riber).

0021-9991/$ - see front matter © 2008 Elsevier Inc. All rights reserved.
doi:10.1016/j.jcp.2008.10.001

context of parallel supercomputing, numerical efficiency is an additional constraint. For single-phase flows, efficient and accurate solvers have been developed and speedups of the order of 5000 are not uncommon [13]. Representing the main physics of two-phase flows while maintaining a similar parallel efficiency for two-phase flow solvers raises additional questions: particle/mesh load imbalance is a crucial issue in Euler/Lagrange simulations, as discussed for instance in [14,15].

In LES of reacting two-phase flows, physics and numerics interact strongly: the first question is to choose a paradigm to describe the two-phase flow. In dilute particle-laden flows, most RANS codes use Euler/Lagrange (EL) methods in which the gas flow is solved using an Eulerian method and the particles are tracked using a Lagrangian approach. An alternative technique is to use Euler/Euler (EE) methods in which both phases are solved using an Eulerian approach. The history of RANS development has shown that the EL and EE methods both show advantages and disadvantages depending on the application. Consequently, both approaches are found today in most commercial codes. For LES, both EE and EL approaches are being developed and the focus of this study is to test and compare them in a reference case where complete sets of experimental results for gas and dispersed phase are available. This exercise is performed here without evaporation or combustion: the droplets are replaced by solid particles.

2. Flow configuration and work objectives

2.1. Choice of the configuration

In combustion chambers, the flame resulting from a free jet flow would be too long compared to the dimensions of the chamber, and also very difficult to stabilise. Therefore, most combustion devices are designed so as to anchor the flame at a specific location. The use of a flame holder is often difficult due to the very high temperatures that may damage the device itself. Another possibility is to stabilise the flame behind a sudden expansion like a backward-facing step: the flow is strongly decelerated forming a corner recirculation zone, and the recirculating hot gases then provoke the ignition of the incoming fresh gases. As far as aeronautical combustion chambers are concerned, highly swirling flows that pass through a sudden expansion are preferred since they provide a more compact stabilised flame. A central toroidal recirculation zone is created, acting as a flame holder in the center of the flow, close to the injector tip. In such devices, the recirculation zones induce high turbulence levels and high mixing rates, stabilising the flame and reducing pollutant emissions.

Before computing reactive two-phase flows in such devices, a validation of the turbulent dispersion of the particles in similar flows is needed. Indeed, the accurate description of the fuel droplet motion is crucial to determine first the resulting fuel vapor distribution, and then the combustion mode and the pollutant emissions among others. With aeronautical applications in prospect, the bluff body flow from Borée et al. [1] is a very interesting configuration. First, it exhibits the same flow structures as combustion chambers, with corner recirculation zones and stagnation points. To predict their location, a precise description of the large structures and the intermittency of the fluid flow is required [16–18]. Second, the particle motion is complex: depending on their inertia, the fuel droplets are captured in the recirculation zones or cross them. Since they are then vaporised, they directly determine the gaseous fuel field, and consequently, the burner efficiency as well as the pollutant emissions [19,20]. Then, as very few particles reach the external walls of the chamber, the particle–wall interactions can be neglected in this configuration, which simplifies the comparison between both methods. Indeed, modeling particle motion in wall-bounded flows is still a challenge despite multiple studies using either the EL approach [21–24] or the EE approach [25–27]. Furthermore, in a hot combustion chamber where liquid fuel is injected, the fuel droplets often evaporate rapidly and the interaction between particles and solid walls is not a crucial mechanism. If not, a liquid film develops on the walls, which requires specific modeling [28]. Finally, a large amount of data is provided by Borée et al. [1], including mean and fluctuating quantities for both phases, which allows to validate not only the gas LES models, but also the dispersed phase modeling.

2.2. Description of the experimental setup

The configuration of Borée et al. [1] consists of a vertical axisymmetrical particle-laden confined bluff body flow (see Fig. 1) on the flow loop Hercule of EDF – R&D.[1] Air blowers are used to generate the coflow whereas both air and glass beads are injected in the inner pipe. The measurement zone is located at $z = 0$ downstream of the inner and annular ducts, where large recirculation zones are created between the central jet and the coflow. Thus, the resulting flow is very similar to the ones obtained in industrial combustion devices, where fuel droplets are injected together with air. Hereafter, the symbols $\cdot_f$ and $\cdot_p$ denote the fluid and the particles.

2.2.1. The gas phase

Schefer et al. [17] show that the topology of a turbulent bluff body flow strongly depends on the ratio $\overline{U}_{f,I}/\overline{U}_{f,C}$, where $\overline{U}$ is a bulk velocity and the subscripts I and C denote the inner pipe and the coflow, respectively. Depending on the velocity ratio $\overline{U}_{f,I}/\overline{U}_{f,C}$, there are three possible flow topologies [17]. In Fig. 2, sketches of mean gas velocity vectors are drawn for three

[1] Électricité de France – Recherche & Développement.

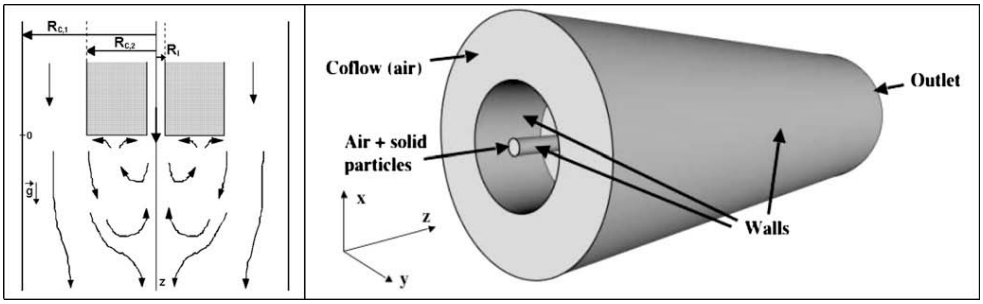


Fig. 1. Configuration of Borée et al. [1]. The dimensions are: R_I = 0.010 m, $R_{C,1}$ = 0.075 m, $R_{C,2}$ = 0.150 m. The ducts and the chamber are respectively 2 m and 1.5 m long.

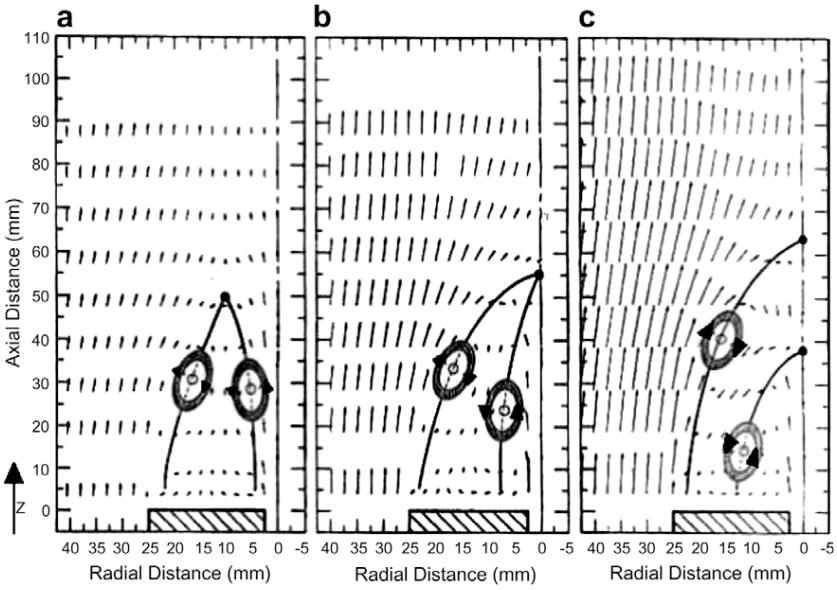


Fig. 2. Plots of the measured mean velocity vectors in a bluff body stabilised methane jet for $\overline{U}_{f,1}/\overline{U}_{f,C} = 2.8$ (a), $\overline{U}_{f,1}/\overline{U}_{f,C} = 1.4$ (b), and $\overline{U}_{f,1}/\overline{U}_{f,C} = 0.84$ (c). Black dots show the location of the stagnation points and round arrays give the direction of rotation of the shear layer vortices. From Schefer et al. [17].

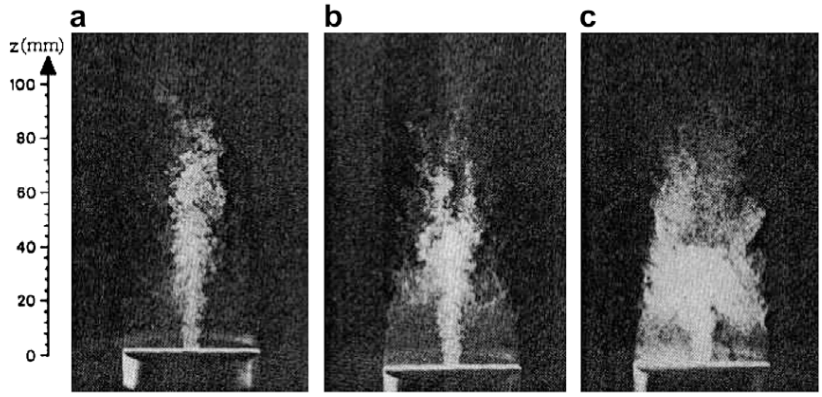


Fig. 3. Mie scattering measurements of the instantaneous particle distribution in a bluff body stabilised methane jet for $\overline{U}_{f,1}/\overline{U}_{f,C} = 2.8$ (a), $\overline{U}_{f,1}/\overline{U}_{f,C} = 1.4$ (b), and $\overline{U}_{f,1}/\overline{U}_{f,C} = 0.84$ (c). From Schefer et al. [17].

decreasing velocity ratios ($\overline{U}_{f,I}/\overline{U}_{f,C} = 2.8, 1.4$ and 0.84) whereas in Fig. 3, the gas flow is seeded with small particles to give access to the flow topology for the same three velocity ratios:

(1) For the highest velocity ratio ($\overline{U}_{f,I}/\overline{U}_{f,C} = 2.8$), the flow along the axis is similar to a free jet flow. The air flows coming from the inner and annular pipes converge far from the outlet of the inner duct. Two counter-rotative eddies separate the two flows before they converge. There are two stagnation points on both sides of the central jet.
(2) At lower velocity ratio ($\overline{U}_{f,I}/\overline{U}_{f,C} = 1.4$), a single stagnation point is formed along the centerline and the flow looks like a non-penetrating jet.
(3) Finally, for the smallest velocity ratio ($\overline{U}_{f,I}/\overline{U}_{f,C} = 0.84$), a second stagnation point appears on the axis whose location is linked to the geometry global parameter. The second stagnation point remains close to the chamber inlet and does not move any longer when the ratio $\overline{U}_{f,I}/\overline{U}_{f,C}$ becomes lower.

The experiments of Borée et al. [1] are conducted at ambient temperature and standard pressure: $T_f = 293$ K and $P_f = 10^5$ Pa. The radius of the inner pipe is $R_I = 10$ mm. The air volume flux of the inner pipe is $Q_{f,I} = 3.4$ m^3 h^{-1}, which corresponds to a mean velocity $\overline{U}_{f,I} = 3.4$ m s^{-1}, whereas the maximum velocity in the inner duct reaches $U_{f,I}^{\max} = 4$ m s^{-1}. Although the Reynolds number is rather moderate ($Re_I = 2R_I\overline{U}_{f,I}/\nu_f \approx 4500$, where ν_f is the kinematic viscosity of the fluid), the ratio $U_{f,I}^{\max}/\overline{U}_{f,I} = 1.18$ at the outlet of the inner pipe is however consistent with developed turbulent pipe flow.

The dimensions of the annular outer region are: $R_{C,1} = 75$ mm, $R_{C,2} = 150$ mm. The air volume flux in the coflow is $Q_{f,C} = 780$ m^3 h^{-1}, which corresponds to mean and maximum velocities equal to: $\overline{U}_{f,C} = 4.1$ m s^{-1} and $U_{f,C}^{\max} = 6$ m s^{-1}. The associated Reynolds number of the annular jet is $Re_C = 2(R_{C,2} - R_{C,1})\overline{U}_{f,C}/\nu_f \approx 40,000$. The main characteristics of the configuration are summarised in Table 1.

The velocity ratio $\overline{U}_{f,I}/\overline{U}_{f,C}$ considered by Borée et al. [1] is 0.83. Following Schefer et al. [17], there are two stagnation points along the centerline. Choosing a ratio lower than one creates a complex gas flow behavior when modeling particle dispersion: depending on their inertia, the particles remain in the recirculation zone delimited by the two stagnation points or leave it.

2.2.2. The dispersed phase

In Borée et al. [1], polydisperse glass particles with material density $\rho_p = 2470$ kg m^{-3}, are injected in the inner pipe only through a particle feeder. Two mass fluxes of glass beads, $Q_{p,I}^1 = 1$ kg h^{-1} and $Q_{p,I}^2 = 5$ kg h^{-1}, have been used experimentally to study the influence of particle inertia on the two-phase flow as well as the role of inter-particle interactions. The corresponding mass loading in the inner duct then varies from $M_{p,I}^1 = 22\%$ to $M_{p,I}^2 = 110\%$. In this study, only the lowest mass loading case is considered.

Particle diameter covers a wide range of size classes from $d_p = 20$ μm to $d_p = 100$ μm. Fig. 4 shows the initial particle distribution, in mass (M) and in number (N). The resulting mean diameters are respectively $d_{p,M} = 63$ μm and $d_{p,N} = 50$ μm. Char-

Table 1
Characteristics of the gas phase at the outlet of the inner and annular pipes.

Gas phase	Inner pipe	Annular pipe
Radius (m)	0.010	0.075
Volume flux (m^3 h^{-1})	3.4	780
Mean velocity (m s^{-1})	3.4	4.1
Max velocity (m s^{-1})	4.	6.
Reynolds number (–)	4500	40,000

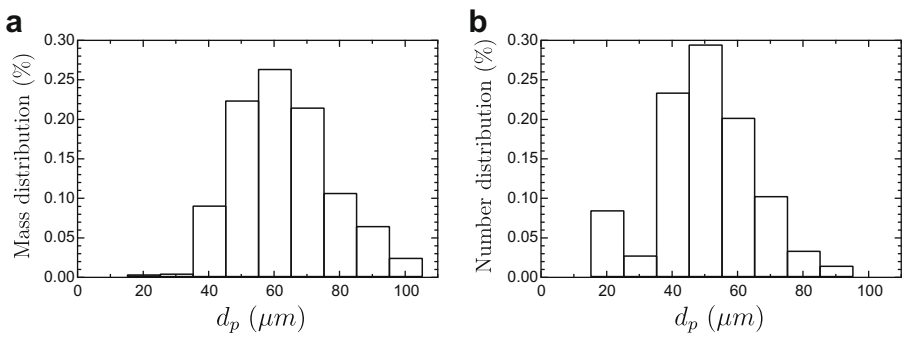


Fig. 4. Experimental initial mass (a), and number (b) distribution of the particle size.

acterising the particle distribution using a particle Doppler anemometry (PDA) method [29] requires special care: the glass beads should remain spherical and the inter-particle collision should not induce any particle break-up. Therefore, Borée et al. [1] repeated the measurements and used a microscope to verify the shape and size of particles.

2.2.3. Characteristic time scales of the two-phase flow

Table 2 gives the particle relaxation time τ_p depending on the particle diameter d_p. In a first approximation, Stokes flow around the particles is assumed so that τ_p reads

$$\tau_p = \frac{\rho_p d_p^2}{18\mu_f}, \tag{1}$$

where μ_f is the dynamic viscosity of the fluid.

The comparison of the particle relaxation time with a characteristic time scale of the fluid most energetic eddies $\tau_{f,t}$, yields the Stokes number St. To evaluate $\tau_{f,t}$ at the outlet of the inner pipe, the length of the most energetic eddies is estimated as a third of the pipe diameter and their velocity as the maximum fluctuating velocity in the pipe:

$$St = \frac{\tau_p}{\tau_{f,t}}, \quad \text{where } \tau_{f,t} = \frac{\frac{2R_I}{3}}{u_{f,I}^{\max}} \approx 7 \text{ ms}. \tag{2}$$

Table 2 presents the characteristic Stokes number of the particles depending on their diameter: the smallest particles with diameter d_p = 20 μm almost follow the gas flow while the inertia of the biggest ones with diameter d_p = 100 μm makes them much more independent of the fluid flow.

As underlined by Schefer et al. [30], the recirculation zones and the stagnation points are related to the inner and outer pipe flow characteristics. However, in the inner pipe (whose diameter is small), the motion of the particles is complex, due to interactions with the fluid and the walls as well as inter-particle collisions. To quantify whether particles have time to adapt to changes in gas mean velocity within the inner duct, the particle relaxation time is compared to their transit time T_p^T in the pipe:

$$T_p^T = \frac{L_I}{U_{f,I}^{\max}} = 500 \text{ ms}, \tag{3}$$

where L_I = 2 m is the length of the inner duct that particles travel across and $U_{f,I}^{\max}$ is the inner maximum gas velocity. Compared to the particle relaxation time, the particle transit time of all particles in the inner pipe remains large. Thus, the particles have time to adapt to fluid turbulence before they reach the outlet of the inner pipe.

2.3. Objectives of the simulations

In this study two unstructured codes developed at CERFACS and CTR are used to investigate the configuration of Borée et al. [1]:

(1) An explicit compressible code (AVBP) using both EL and mesoscopic EE [31–33] approaches. For the present study, AVBP is used on both hexahedron-based and tetrahedron-based grids.
(2) An implicit incompressible code (CDP) using a EL formulation. For this study, hexahedron-based grids are used in CDP.

The dispersed phase consists of solid particles so that evaporation, coalescence and break-up do not have to be considered. Accounting for polydispersion in the bluff body configuration using EL approaches is straightforward: particles with different diameters are injected at the inlet of the chamber. When using EE approaches, there are two main methods. On the one hand, polydispersion can be accounted for by introducing a particle size distribution [34]. On the other hand, EE approaches can be extended to polydispersion using a multi-class method: the set of particulate equations is resolved for a finite number of particle classes depending on their diameter. The main disadvantage of this method is its computational cost which drastically increases with the number of classes to be considered. In the specific case of Borée et al. [1] experiments, separate studies (not reported here) using Lagrangian simulations and polydisperse particles or 60 μm particles only have shown that using a monodisperse distribution of size was very close to the 22% case of Borée et al. [1] and was sufficient to capture both the mean flow effects on the gas (through two-way coupling) and the dynamics of the 60 μm class. Therefore, in this work, particle distribution is assumed monodisperse with particle diameter close to the initial mean diameter in mass: d_p = 60 μm. Only the low mass loading case (M_p = 22%) of Borée et al. [1] is studied. Since the particle volume fraction

Table 2
Experimental particle relaxation time and Stokes number depending on particle diameter.

d_p (μm)	20	30	40	50	60	70	80	90	100
τ_p (ms)	3.1	6.9	12.3	19.2	27.6	37.6	49.1	62.2	76.7
St (–)	0.4	1.0	1.8	2.7	3.9	5.4	7.0	8.9	11.0

α_p is about 1% at the inlet, inter-particle collisions are found to play a crucial role in the injection tube [35]. Nevertheless, the dilution effect being very effective, collision effect may be neglected in the modeling approaches for the numerical simulation of the downstream two-phase flow. Furthermore, the two-way coupling is taken into account through drag force. However, its impact on the gas phase has been shown of minor importance in the 22% mass loading case [1]. This case allows to study the following points:

(1) evaluate influence of mesh type (hexahedra vs tetrahedra);
(2) evaluate influence of numerical convective scheme (second and third order);
(3) compare implicit and explicit formulations for time advancement;
(4) compare accuracy of EE and EL approaches.

3. Description of solvers and models

3.1. Gas flow solvers

Numerical methods used in both LES solvers for the gas phase have been extensively described in the literature ([9,7,36] for CDP and [8,12] for AVBP) and will only be summarised here.

The explicit LES solver AVBP solves the compressible Navier–Stokes equations with a second-order finite-volume Lax–Wendroff scheme or a third-order finite-element scheme TTGC [37,9]. The WALE model [38] that predicts the right scaling for the fluid turbulent viscosity when approaching a solid boundary (i.e., $\nu_{f,t} = 0$) is used to model the subgrid stress tensor.

The LES solver CDP solves implicitly the incompressible Navier–Stokes equations. The time integration of CDP is based on the fractional-step method [39] and the space integration relies on a second-order central scheme that conserves the kinetic energy [7,36]. The dynamic Smagorinsky model [40] is used to model the subgrid stress tensor.

The boundary conditions for each solver are detailed in Section 4.3.

3.2. Dispersed phase flow solvers

The particles are assumed to be rigid spheres with diameter comparable or smaller than the Kolmogorov length-scale. If the particle density is much larger than the fluid density, the forces acting on particles reduce to drag and gravity [41,42].

3.2.1. Euler/Lagrange approach

In EL simulations, the influence of the particles on the gas phase momentum and energy equations is taken into account by using the point-force approximation in the general framework of the particle-in-cell method (PIC) [43–47], with standard single-phase subgrid turbulence modeling approaches. According to Boivin et al. [48], such an assumption is valid for small mass loading ratio (typically, $\alpha_p\rho_p/\rho_f \leqslant 1$) with response time larger than the subgrid turbulence characteristic time scale. In particular, the coupling force exerted by each particle on the fluid is projected onto the grid nodes. The weights in the projection operation are inversely proportional to the distances between the particle and the nodes of the containing cell.

With these assumptions, the particle equations of motion can then be written for a single particle as

$$\frac{dx_{p,i}}{dt} = u_{p,i}, \tag{4}$$

$$\frac{du_{p,i}}{dt} = -\frac{3}{4}\frac{\rho_f}{\rho_p}\frac{C_D}{d_p}|\mathbf{v}_r|v_{r,i} + g_i = -\frac{u_{p,i} - \tilde{u}_{f,i}}{\tau_p} + g_i. \tag{5}$$

The local drag coefficient in Eq. (5) is C_D and may be expressed in terms of the particle Reynolds number Re_p following Schiller and Nauman [49]:

$$C_D = \frac{24}{Re_p}[1 + 0.15Re_p^{0.687}], \quad Re_p = \frac{|\mathbf{v}_r|d_p}{\nu_f} \leqslant 800, \tag{6}$$

where τ_p is the particle relaxation time defined as

$$\tau_p = \frac{4}{3}\frac{\rho_p}{\rho_f}\frac{d_p}{C_D|\mathbf{v}_r|}. \tag{7}$$

The local instantaneous relative velocity between the particle and the surrounding fluid is $\mathbf{v}_{r,i} = u_{p,i} - \tilde{u}_{f,i}$, where $\tilde{u}_{f,i}$ is the fluid velocity at the position of the particle assuming that the flow field is locally undisturbed by the presence of this particle [42,50]. In first approximation, the velocity is assumed to be equal to the interpolation of the filtered velocity at the position of the particle [21,51,52]. The effect of the subgrid fluid turbulence is assumed to be negligible owing to the large inertia of the solid particles [53]. A linear interpolation algorithm is used to compute the fluid velocity at the position of the particle.

3.2.2. Euler/Euler approach

Eulerian equations for the dispersed phase may be derived by several means. A popular and simple way consists in volume filtering of the separate, local, instantaneous phase equations accounting for the inter-facial jump conditions [54]. Such

an averaging approach is restrictive because the filter length must be both larger than the inter-particle distance (to define continuous mean fields of particle number density and particle velocity), and smaller than the smallest length-scale in the particle velocity field to ensure unicity of the particle velocity in the filtering volume. Moreover, Druzhinin [54] assumes that all the particles located in this filtering volume have the same velocity, which is untrue when the particle relaxation time is larger than the smallest turbulent time scale [32]. An alternative to the two-fluid approach is the Eulerian equilibrium approach proposed by Ferry and Balachandar [55] that reduces the number of transport equations to be solved: only the transport equation for particle number density is solved while the particle velocity is given by the expansion in τ_p of the local fluid quantities first proposed by Maxey [41] and then extended by Druzhinin [56], and Ferry and Balachandar [55]. Both approaches show good agreement with EL results when focusing on preferential concentration of low-inertia particles in HIT flows [41,57]. However, the errors in the predictions of the particle field increase with the particle response time [58].

Though, the crucial assumption of the above approaches, the particle velocity uniqueness at a given position, fails when the particle relaxation time is larger than the Kolmogorov time scale, due to the crossing of particle trajectories. To overcome this difficulty, Février et al. [32], proposed a probability density function (PDF) approach based on a conditional ensemble average of the particle properties for a given turbulent fluid flow realization. In such an approach, any discrete particle velocity may be separated into two contributions: an Eulerian velocity field, the mesoscopic velocity field shared by all the particle realizations, and a Lagrangian random distribution, the random uncorrelated velocity (RUV), spatially uncorrelated and which accounts for the particle trajectory crossing. The conditional particle velocity PDF $\breve{f}_p(\mathbf{c}_p, \mathbf{x}, t)$ gives the local instantaneous probable number of particles with the given translation velocity $\mathbf{u}_p = \mathbf{c}_p$, and obeys a Boltzmann-type kinetic equation accounting for external forces acting on the particles and inter-particle collisions. The moments of the particle PDF are mesoscopic Eulerian quantities which obey transport equations derived by integration from the kinetic equation, following the same methodology than for the derivation of the Navier–Stokes equations in the frame of kinetic theory [59]. So, Février et al. [32] derived transport equations for particle number density $\bar{n}_p$, mesoscopic velocity $\breve{\mathbf{u}}_p$ and random uncorrelated kinetic energy (RUE) $\delta\breve{\theta}_p$ and Simonin [60] proposed, as a first approximation, a viscosity assumption to model the random uncorrelated kinetic stresses. The mesoscopic approach was evaluated using a priori test from discrete particle simulation (DPS) coupled with DNS or LES of forced homogeneous isotropic turbulence [61] and fully-developed channel flow [22]. Then, by analogy with the gas phase, a LES filter is applied to the equations for the mesoscopic quantities [62,33], which leads to equations for the filtered (in the LES sense) particle number density $\bar{n}_p$ and the filtered correlated velocity $\hat{\mathbf{u}}_p$ using the mass-weighted averaging $\widehat{n_p u_{p,j}} = \bar{n}_p \hat{u}_{p,j}$ [63]:

$$\frac{\partial}{\partial t}\bar{n}_p + \frac{\partial}{\partial x_j}\bar{n}_p \hat{u}_{p,j} = 0, \tag{8}$$

$$\frac{\partial}{\partial t}\bar{n}_p \hat{u}_{p,i} + \frac{\partial}{\partial x_j}\bar{n}_p \hat{u}_{p,i}\hat{u}_{p,j} = -\frac{\bar{n}_p}{\tau_p}(\hat{u}_{p,i} - \hat{u}_{f,i}) + \bar{n}_p g_i - \frac{\partial}{\partial x_j}T_{p,ij} - \frac{\partial}{\partial x_j}\bar{n}_p \widehat{\delta R^*_{p,ij}} - \frac{\partial}{\partial x_i}\frac{2}{3}\bar{n}_p \widehat{\delta\theta}_p, \tag{9}$$

where $\bar{n}_p$, $\hat{\mathbf{u}}_p$, $\widehat{\delta R^*_{p,ij}}$ and $\widehat{\delta\theta}_p$ are respectively the filtered particle number density, the particle correlated velocity, the second-order uncorrelated particle velocity correlation tensor deviatoric, and the particle RUE. The first two terms of the right-hand side (RHS) of Eq. (9) are the drag force and gravity effects on large scales, the third one accounts for the subgrid-scale (SGS) effects, the fourth one is a diffusion term and the last one represents the pressure effect due to RUE. $T_{p,ij}$ stands for the particle subgrid stress tensor:

$$T_{p,ij} = \bar{n}_p(\widehat{u_{p,i}u_{p,j}} - \hat{u}_{p,i}\hat{u}_{p,j}). \tag{10}$$

3.2.3. Filtered RUV and subgrid term modeling

Assuming small anisotropy of the RUM, Simonin et al. [64] model $\delta R^*_{p,ij}$ by a viscous term. For LES approaches this model is adapted by replacing non-filtered quantities by filtered ones leading to [62]:

$$\widehat{\delta R^*_{p,ij}} = -\hat{\nu}_{p,\mathrm{RUM}}\left(\frac{\partial \hat{u}_{p,i}}{\partial x_j} + \frac{\partial \hat{u}_{p,j}}{\partial x_i} - \frac{2}{3}\frac{\partial \hat{u}_{p,k}}{\partial x_k}\delta_{ij}\right), \tag{11}$$

where the RUM viscosity $\hat{\nu}_{p,\mathrm{RUM}}$ is obtained in the framework of the kinetic theory of particulate flows [61]:

$$\hat{\nu}_{p,\mathrm{RUM}} = \frac{\tau_p}{3}\widehat{\delta\theta}_p. \tag{12}$$

For the SGS tensor $T_{p,ij}$, Riber et al. [65] propose a viscosity model by analogy with single-phase flows [66,67]. The trace-free SGS tensor is modeled using a viscosity assumption (compressible Smagorinsky model), while the subgrid energy is parametrized by a Yoshizawa model [68]:

$$T_{p,ij} = -C_S 2\Delta_f^2 \bar{n}_p|\widehat{S}_p|\left(\widehat{S}_{p,ij} - \frac{\delta_{ij}}{3}\widehat{S}_{p,kk}\right) + C_I 2\Delta_f^2 \bar{n}_p|\widehat{S}_p|^2\delta_{ij}, \tag{13}$$

where $\widehat{S}_p$ is the filtered particle strain rate tensor, $|\widehat{S}_p|^2 = 2S_{p,ij}S_{p,ij}$ and Δ_f the filter characteristic length. The model constants have been evaluated in a priori tests [69] leading to the values $C_S = 0.02$, $C_I = 0.012$.

3.2.4. Simplified Euler/Euler model

The filtered particle RUE is required twice in the transport equation for filtered particle correlated velocity (Eq. (9)): first in the term representing pressure effects due to particle RUE; second when modeling the filtered second-order uncorrelated particle velocity correlation tensor deviatoric (see Eqs. (11) and (12)). To close these two terms, a transport equation for filtered particle RUE can be solved, as proposed by Février et al. [32]. This has been done by Kaufmann et al. [70] when simulating DNS of particle-laden HIT flows, showing good agreement between Lagrangian and Eulerian results. Still, these models proposed for the unclosed terms due to RUM are very recent and the validity of the viscosity model has recently been questioned by Riber [33] when performing LES in a particle-laden turbulent confined jet flow [71]. Indeed, the resolved particle fluctuations were considerably damped and only the RUM contributed to the particle agitation, which is not realistic. An alternative already tested by Boileau et al. [20] and Riber [33], consists in neglecting the RUM contributions in the transport equations for the dispersed phase. Then, the set of equations reduces to

$$\frac{\partial}{\partial t}\bar{n}_p + \frac{\partial}{\partial x_j}\bar{n}_p\hat{u}_{p,j} = 0, \tag{14}$$

$$\frac{\partial}{\partial t}\bar{n}_p\hat{u}_{p,i} + \frac{\partial}{\partial x_j}\bar{n}_p\hat{u}_{p,i}\hat{u}_{p,j} = -\frac{\bar{n}_p}{\tau_p}(\hat{u}_{p,i} - \hat{u}_{f,i}) + \bar{n}_p g_i - \frac{\partial}{\partial x_j}T_{p,ij}. \tag{15}$$

This simplified Euler/Euler model has been chosen to be evaluated in the bluff body configuration. Note that when using this simplified model, the mean particle velocity and mass flux fields are expected to be correctly predicted while the agitation of the particles should be under-estimated since a part, depending on the particle inertia, is the RUM contribution [33].

4. Comparison of single-phase flow simulations

4.1. Choice of the computational domain

The total volume of the configuration is large, due to the length of the ducts (2 m) and the chamber itself (1.5 m), which is far larger than a typical combustion chamber test-rig. Therefore, calculating the whole geometry would be computationally expensive.

Since the location of the second stagnation point mainly depends on the geometry global diameter, the diameters of the inner and annular pipes have been kept: R_I = 0.010 m; $R_{C,1}$ = 0.075 m; $R_{C,2}$ = 0.150 m. In contrast, the pipes have been shortened to 0.1 m for two reasons. On the one hand, it is necessary to decrease the length of the ducts: considering the low Reynolds number and the grid resolution in the inner pipe as well as the accuracy of the numerical scheme, it is impossible to wait for natural destabilisation of the gas flow within the pipe. Specific inlet boundary conditions are therefore used to help the flow destabilisation (see Section 4.3). On the other hand, the pipes cannot be decreased down to 0.1 m: the accurate prediction of particle motion in a pipe (or a channel) is still difficult to obtain, especially because of particle–wall interactions [21] and inter-particle collisions [22]. Since these interactions are not accounted for in this work, one has to ensure that the modified pipe length stays compatible with the particle relaxation time and the particle transit time evaluated in Section 2.2.2.

Note that the length of the chamber (1.5 m) may also be decreased since the second stagnation point is located in the vicinity of z = 0.4 m and the flow shows very few structures downstream from this point. In this work, the chamber is shortened for one of the grids tested, as specified in Section 4.2.

These simplifications allow to divide the volume of the computational domain by two, which drastically reduces the computational cost of the LES.

4.2. Grids tested

Two grids have been tested on this bluff body configuration to investigate the effects of both resolution and grid type (tetrahedra or hexahedra). Table 3 details the characteristics of the two grids named *gridtet* and *gridhex*. Figs. 5 and 6 present an overview of the grid resolutions in longitudinal and front cutting planes, respectively.

Table 3
Characteristics of the two grids tested.

Name	*gridtet*	*gridhex*
Grid type	Tetrahedra	Hexahedra
Number of nodes	549,369	3,255,085
Number of cells	3,115,898	3,207,960
Length of the pipes (m)	0.1	0.1
Length of the chamber (m)	1.5	0.8
Total volume of the domain (m^3)	0.111	0.062
y^+ in the inner pipe (–)	15	7.5
y^+ in the coflow (–)	64	15

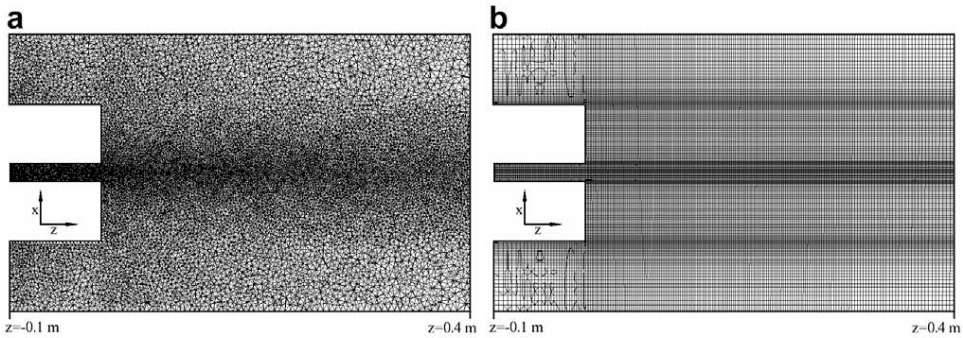


Fig. 5. Longitudinal cutting plane ($y = 0$) of the two grids tested: *gridtet* (a) and *gridhex* (b) detail of the region $z \in [-0.1; 0.4]$ m.

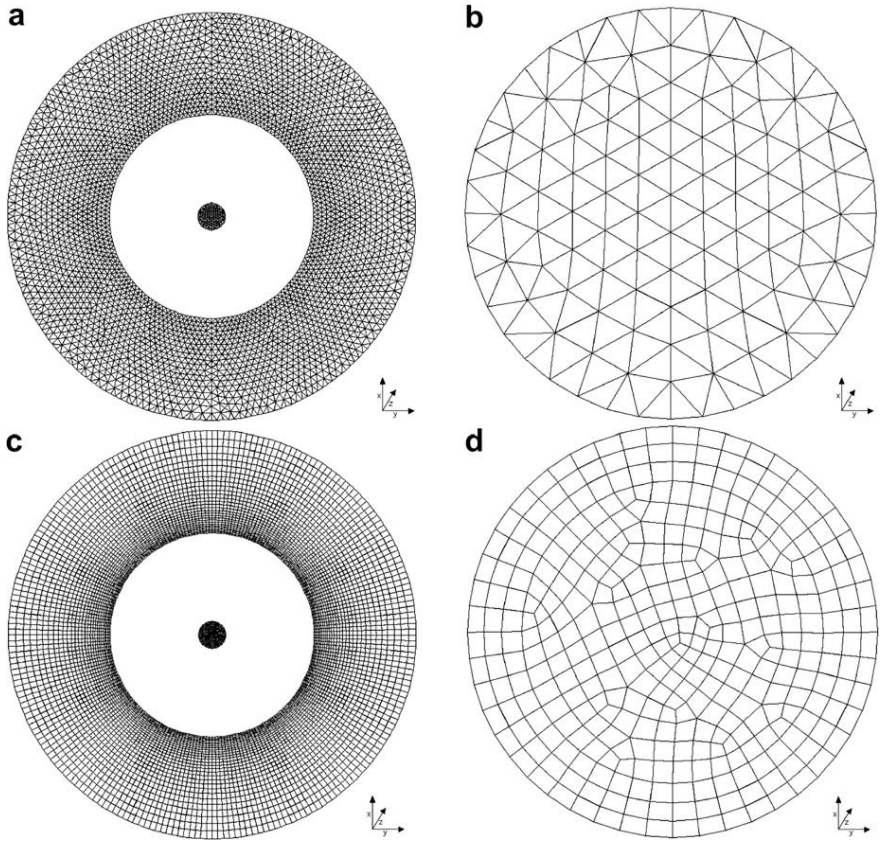


Fig. 6. Global front view (left) and detail of the inner inlet (right) for the two grids tested: *gridtet* (a and b) and *gridhex* (c and d).

There are three main differences between the two grids. First, the length of the chamber is different: according to the experimental setup, the chamber is 1.5 m long for *gridtet* whereas *gridhex* has been shortened down to 0.8 m for two reasons. On the one hand, the second stagnation point is located far upstream from the outlet of the chamber. On the second hand, the grid can be easily coarsened downstream from the second stagnation point to save computational cost when using tetrahedra, which is more difficult with hexahedra. This simplification has been verified to have no influence on the results. Consequently, as summarised in Table 3, the total number of cells is very similar for the two grids, but the total volume is twice larger for *gridtet* than for *gridhex*. The second difference between the two grids deals with the non-dimensional wall distance y^+ in the pipes which is two times smaller in the inner pipe and four times smaller in the coflow in *gridhex* than in *gridtet*. Both grids contain much more cells in the inner pipe than in the outer one. Finally, in the tetrahedron-based grid,

special care has been taken to generate small cells in the recirculation zones, which was unaffordable in the hexahedron-based grid considering the increase in computational cost it would induce.

4.3. Boundary conditions

As the two LES codes use different formulations (see Section 3.1), the boundary conditions shown in Fig. 1 are described separately.

4.3.1. Boundary conditions for the AVBP code

The characteristic boundary conditions developed by Poinsot and Lele [72], and Moureau et al. [9] are used for these simulations.

The outlet is nearly non-reflective at atmospheric pressure: $P_f = 1.013 \times 10^5$ Pa.

The inlet treatment is more complex. Indeed, the first test section where the experimental profiles of mean and fluctuating fluid velocities are known is located at $z = 3$ mm. Then, the main difficulty consists in specifying the boundary conditions at the inlet of the pipes in order to obtain good agreement between the simulations and the experiments at $z = 3$ mm. Considering the Reynolds numbers in both pipes (see Section 2.2.1), typical mean axial velocity profiles of fully-developed turbulent pipes (following the classical 1/7 power-law) are imposed at the inlet of the inner and annular pipes, corresponding to the experimental mass flux. However, the Reynolds numbers in the pipes are too low to expect natural destabilisation of the gas flow, i.e., without any flow forcing. Therefore, a time and space-varying velocity signal at the inlet of both pipes is imposed in order to reproduce the effect of an incoming turbulent field as observed in the experiment. This incoming turbulent signal is constructed using a random flow generation (RFG) algorithm [73,74]. The incoming field consists of a superposition of harmonic functions (50 modes projected in the three directions) with two characteristic length-scales prescribed by user: the most energetic length-scale depends on the pipe diameters while the most dissipative one is directly linked to the grid resolution on the inlet patch. Typical fluctuating profiles of fully-developed pipes are imposed at the inlet of the pipes to match experimental fluctuating profiles at $z = 3$ mm. Forcing the flow in such a way considerably accelerates the establishment of developed turbulent flows. It also ensures the presence of coherent perturbations not warranted with a pure white noise.

Since the WALE subgrid model shows a correct behavior close to the wall [38], no wall modeling is used: non-slip conditions are imposed at the walls that are isothermal at ambient temperature: $T_f = 293$ K.

4.3.2. Boundary conditions for the CDP code

An alternative to the RFG method to generate inlet turbulence in the inner pipe is to compute a well-resolved turbulent pipe separately and to inject it in the bluff body computation. The advantage of this injected turbulence is to be non-synthetic without any parameter to prescribe. This method is used for the inner pipe by computing a periodic pipe with a constant volume forcing in the momentum equations. The forcing is dynamically adjusted to obtain the right mass flux. The computational mesh, which is five diameter long, consists of 2.05 million hexahedra with a y^+ equal to 1.9. Even with this well-resolved mesh, the transition from a laminar to a turbulent regime is achieved by beginning the computation with a Reynolds number of 6000 before decreasing slowly to the target Reynolds number of 4500. Then, instead of recording the velocity on a cut plane, a single instantaneous snapshot of the flow is taken. In this snapshot, the streamwise spatial abscissa is transformed into a time abscissa by dividing it by the mean velocity in the pipe. Finally, the inlet velocity for the bluff body pipe is spatially and temporally interpolated from the transformed snapshot. This method saves CPU time because the stand-alone pipe may be computed during a physical time much shorter than the one needed to converge the bluff body flow. The only drawback is that it deforms the eddies where the mean streamwise velocity is different of the global mean velocity, mainly close to the wall.

For the coflow, no turbulence is injected. Only the mean velocity profile is prescribed by imposing the experimental mean velocity measured at $z = 3$ mm.

Non-slip conditions are imposed at the walls that are isothermal at ambient temperature: $T_f = 293$ K while the outlet is purely convective due to the incompressible equations solved in the CDP code.

4.4. Test cases

In such an industrial-like configuration, the computational cost is often the limiting factor determining the grid resolution and the numerical method to be used. The choice of the numerical scheme (low- or high-order), the grid resolution and type (hexahedra or tetrahedra) and the solver type (implicit or explicit) is not straightforward. Whether results are more accurate with a low-order scheme used on a refined grid or with a high-order scheme coupled with a coarser grid is still an open question, for instance [75]. While Colin and Rudgyard [37] and Vreman [76] for example aim at developing high-order schemes on coarse grids, other authors use second-order schemes on more refined grids [77,7]. Naturally, the computational cost of the simulation is a key point for the final choice. In this section, five cases are analysed to investigate the influence of:

(1) *The* numerical scheme: The TTGC scheme [4], third-order accurate finite-element, is known to provide better results than the second-order finite-volume LW scheme, especially in recirculating flows. The influence of the scheme is shown in Section 4.6 using the same unstructured grid *gridtet*, and the AVBP code.

(2) *The* inlet boundary condition treatment: The relatively low Reynolds number associated with a low grid resolution in the pipes do not allow a natural transition to turbulence. The influence of the inlet forcing using the RFG method described in Section 4.3.1, is analysed in Section 4.7 using the AVBP code with the grid *gridtet* and the TTGC scheme.
(3) *The* grid: Hexahedron-based grids are rarely used to calculate swirled flows since they are said to generate preferential directions. The AVBP code with the TTGC scheme is used to investigate the influence of grid type and grid resolution in Section 4.8.
(4) *The* code: In Section 4.10, the results provided by AVBP and CDP on the hexahedron-based grid *gridhex* are compared with the measurements to evaluate the accuracy of both gas LES solvers.

All cases are summarised in Table 4 for those carried out with AVBP whereas the LES performed with CDP is detailed in Section 4.10. To validate results, the following diagnostics are used:

(1) the Q-criterion of Dubief and Delcayre [78], Hunt et al. [79] and Hussain and Jeong [80] to visualise the flow structures;
(2) the axial variations of mean and root-mean-square (RMS) axial velocities (as plotted in Fig. 3 in Borée et al. [1]) to measure the length of the recirculation zone;
(3) the radial variations of mean and RMS axial velocities at seven stations along the duct axis (z = 3, 80, 160, 200, 240, 320 and 400 mm as represented in Fig. 7) to provide a detailed comparison of LES and experimental fields.

The cylindrical coordinate system (z, r, θ) is used to indicate axial (downward), radial and azimuthal directions. As no mean swirling motion was detected, only the axial and radial velocity components are provided. The mean components resulting from LES-averaging are respectively noted W and U_r whereas the RMS components are w_{rms} and $u_{r,\mathrm{rms}}$.

4.5. Qualitative gas flow analysis

The analysis of the averaged quantities resulting from LES requires a simulation time long enough to ensure convergence, and a sampling time small enough to ensure that the smallest structures can contribute to the averaged solution. In the present configuration, the lowest frequency to be represented is associated to the two counter-rotating structures on each side of the axis as plotted in Fig. 2(c). Considering their size $l_{f,l} \approx 8 \times 10^{-2}$ m, and their mean rotating velocity $U_{f,l} \approx 1$ m s^{-1}, the order of magnitude of the associated time is $\tau_{f,l} \approx 8 \times 10^{-2}$ s. The most energetic eddies in the inner pipe constitute a reasonable choice to determine the order of magnitude of the highest frequencies. Considering their size $l_{f,t} \approx 7 \times 10^{-3}$ m, and their velocity $u'_{f,t} \approx 1.5$ m s^{-1}, the order of magnitude of the associated time scale is $\tau_{f,t} \approx 4.6 \times 10^{-3}$ s. All the LES performed with the explicit solver AVBP have been run for $T_{\mathrm{av,AVBP}} \approx 1$ s while the LES performed with the implicit solver CDP has been run longer: $T_{\mathrm{av,CDP}} \approx 2.5$ s. The time between two samples is $\Delta t_r \approx 1.2 \times 10^{-3}$ s. In both codes, the statistics of the mean fields are then well converged. This is not always the case for the RMS quantities but the overall tendency gives enough information to compare models and solvers.

Table 4
Test cases computed with AVBP and corresponding parameters.

Case	Solver	Grid	Scheme	Inlet treatment
avbp_tet_lw_norfg	AVBP	*gridtet*	LW	None
avbp_tet_ttgc_norfg	AVBP	*gridtet*	TTGC	None
avbp_tet_ttgc_rfg	AVBP	*gridtet*	TTGC	RFG
avbp_hex_ttgc_rfg	AVBP	*gridhex*	TTGC	RFG

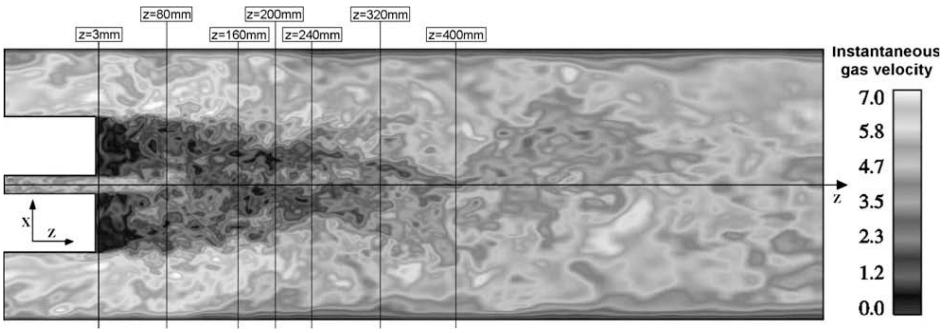


Fig. 7. Instantaneous field of velocity modulus obtained with case *avbp_hex_ttgc_rfg* in the cutting plane $y = 0$. The seven vertical lines represent the experimental stations.

Fig. 7 shows an instantaneous field of the gas velocity modulus in the cutting plane $y = 0$ for the case *avbp_hex_ttgc_rfg*. Many structures of different sizes are visible. The largest ones are linked to the diameter of the coflow, intermediate ones appear in the shear layers and structures coming out of the inner jet are also clearly identified.

The time-averaged fields considerably differ from the instantaneous flow structure. Fig. 8(a)–(c) presents, respectively, the mean field of gas axial velocity, and the RMS fields of gas axial and radial velocities in the cutting plane $y = 0$ for the case *avbp_hex_ttgc_rfg*. The iso-contour line of zero mean axial velocity is added on the three pictures. All fields are rather symmetric, which indicates good convergence of the simulations. As expected, there are two points with zero velocity along the axis, corresponding to distinct inner jet and coflow stagnation points. The peak of axial fluctuations is located at the inner jet stagnation point whereas the radial fluctuations are maximum in the vicinity of the furthest stagnation point. Both axial and radial fluctuations show secondary peaks, respectively in the external shear layer and close to the first stagnation point. At these two specific locations, turbulence is highly anisotropic.

4.6. Influence of the numerical scheme

To evaluate the influence of the numerical scheme on the gas flow, the cases *avbp_tet_lw_norfg* and *avbp_tet_ttgc_norfg* are compared. Both LES are performed with AVBP on the unstructured grid *gridtet*. The RFG method is not used at the inlet of the ducts so that the only difference between the two cases is the numerical scheme.

The qualitative impact of the scheme order on the small structures is clear in Fig. 9 where instantaneous iso-surfaces of Q-criterion are displayed for both LW and TTGC schemes. Both fields exhibit two kinds of coherent structures: some longitudinal vortices come from the inner pipe whereas some others are created in the external shear zone and are rather azimuthal. However, the TTGC scheme provides more numerous and more detailed flow structures than the LW scheme.

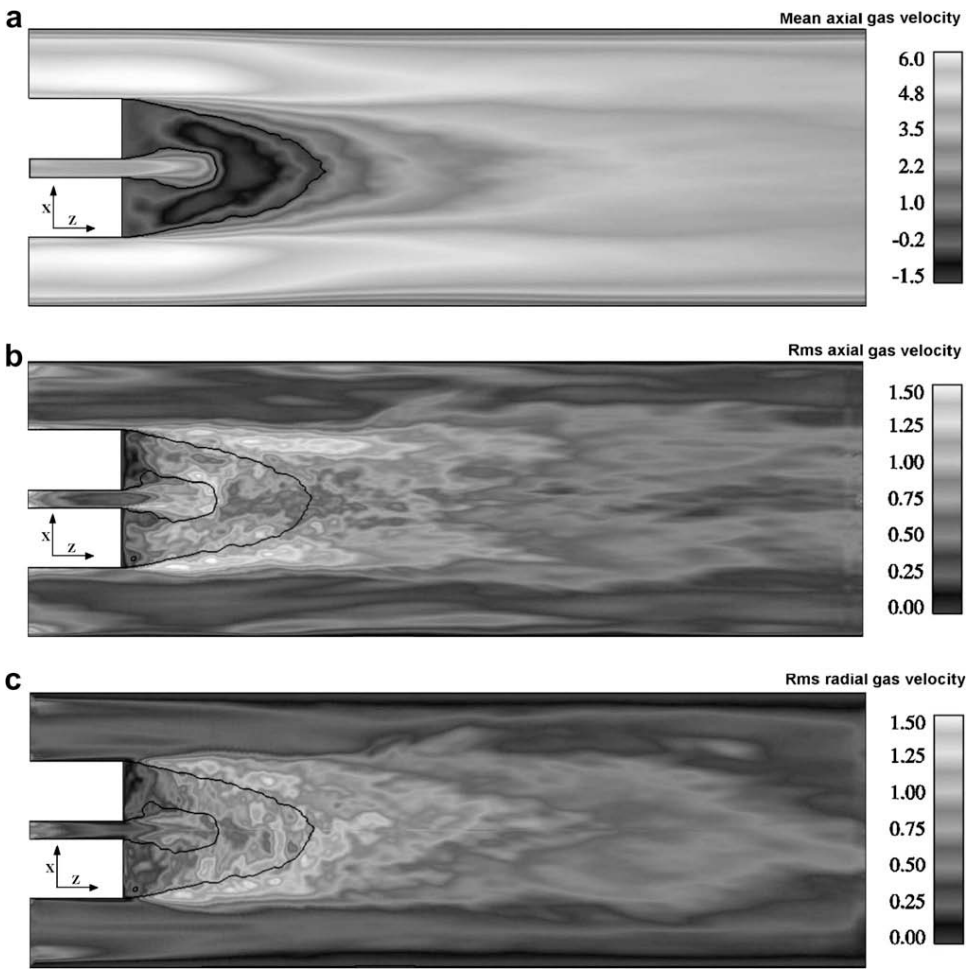


Fig. 8. Mean fields of gas axial velocity (a), and RMS fields of gas axial (b) and radial (c) velocity obtained with case *avbp_hex_ttgc_rfg* in the cutting plane $y = 0$. The black line corresponds to the iso-contour line $\langle W_f \rangle = 0$.

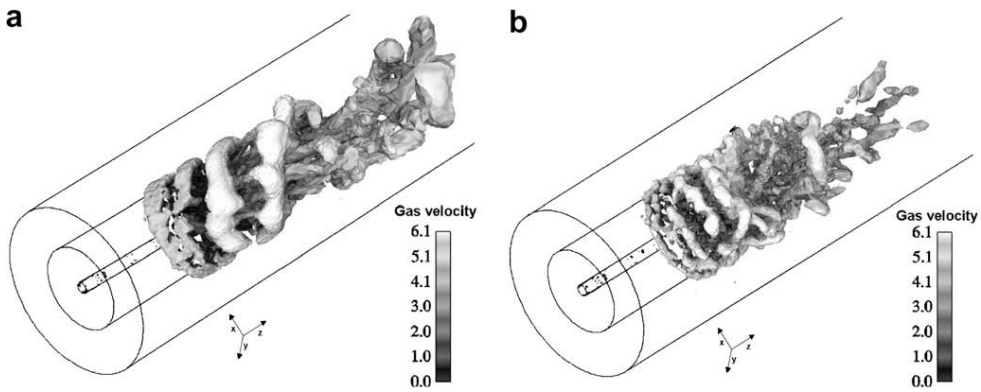


Fig. 9. Instantaneous iso-surfaces of Q-criterion for LW (a), and TTGC (b). The iso-surfaces are colored by instantaneous velocity. (Simulations performed with AVBP.)

These differences are quantified in Fig. 10 where the axial profiles of mean and RMS axial velocities are plotted. When using LW, the two points with zero mean axial velocity that delimit the recirculation zone are located too far downstream from the ducts, and so is the peak of RMS axial velocity. On the contrary, the location of the two stagnation points as well as the peak of RMS axial velocity are well predicted with TTGC. Only the amplitude of this peak is over-estimated. With two-phase flows and even combustion in prospect, delimiting precisely the recirculation zone is of high importance: the particle distribution as well as the flame location and shape greatly depend on the first stagnation point. Consequently, all the LES performed with AVBP and analysed hereafter use the TTGC scheme.

4.7. Influence of the inlet boundary treatment

The runs of Fig. 10 were performed without imposing turbulent velocity fluctuations in the inner pipe and this simplification is questionable. Indeed, the mean axial fluid velocity is shown to increase along the inner pipe ($-0.1 < z < 0$) while there is no fluctuation developing in the duct, showing that the turbulent flow within the central pipe is not correctly captured by the solver. Performing a true LES in the ducts would be computationally expensive because it would require a considerable increase in resolution. Therefore, in this work, the specific inlet boundary treatment described in Section 4.3.1 has been used to feed turbulent fluctuations in the inner pipe.

The direct comparison of the cases *avbp_tet_ttgc_norfg* and *avbp_tet_ttgc_rfg* exhibits the impact of the inlet turbulent forcing method, as shown in Fig. 11. Usually in a pipe flow, the peaks of velocity fluctuations reach almost 10% of the mean velocity on the centerline. Owing to a lack of resolution in the inner pipe, the velocity fluctuations imposed at the inlet of the pipe have deliberately been increased to 15% of the axial mean velocity. The main purpose is to get good agreement between simulations and experiments at the outlet of the duct, i.e., at $z = 3$ mm, which is shown in Fig. 11(b). The consequence of the fluid agitation in the duct itself is a flatter mean axial velocity profile in the pipe: Fig. 11(a) shows that the maximum of mean axial velocity is in better agreement with the experiments at the outlet of the duct when using the RFG method. Note that the accurate prediction of the location of the recirculation zone for the case *avbp_tet_ttgc_norfg* is only due to a fortuitous compensation of errors.

4.8. Influence of the grid

The influence of the grid type as well as the grid resolution is investigated comparing the results provided by the cases *avbp_tet_ttgc_rfg* and *avbp_hex_ttgc_rfg*. As mentioned in Table 4, the only difference between these two LES is the grid.

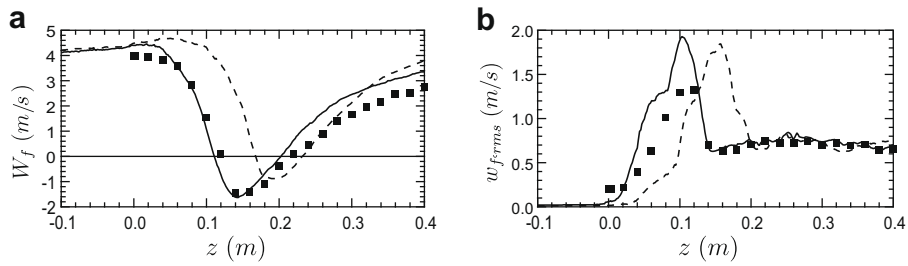


Fig. 10. Axial profiles of mean (a), and RMS (b), axial gas velocity. Symbols: experiment – dashed line: LW – solid line: TTGC. (Simulations performed with AVBP.)

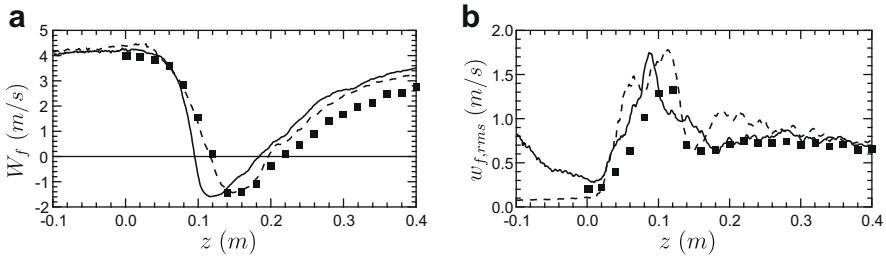


Fig. 11. Axial profiles of mean (a), and RMS (b), axial gas velocity. Symbols: experiment – dashed line: no injected turbulent fluctuations – solid line: turbulent fluctuations injected at inlet. (Simulations performed with AVBP.)

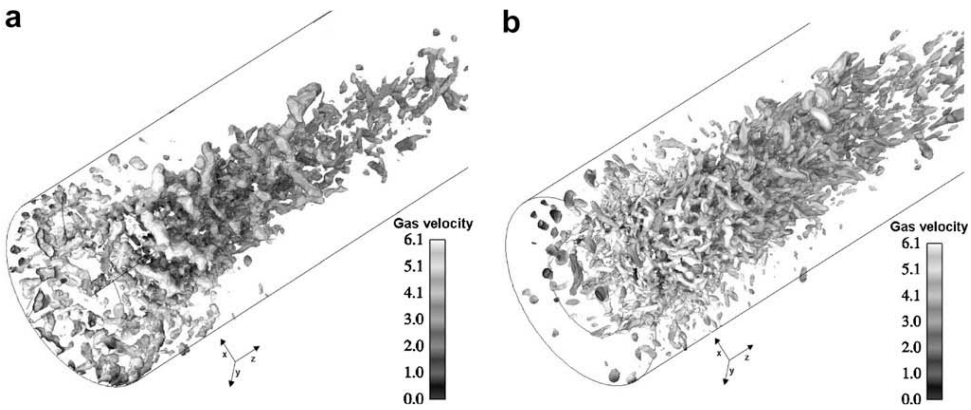


Fig. 12. Instantaneous iso-surfaces of Q-criterion for *gridtet* (a), and *gridhex* (b). The iso-surfaces are colored by instantaneous velocity. (Simulations performed with AVBP.)

The qualitative impact of the grid on the fluid flow topology is shown in Fig. 12 where instantaneous iso-surfaces of Q-criterion are displayed for both cases. Although the two fields exhibit the same kind of longitudinal and azimuthal coherent structures, there are much more, but also much smaller vortices with the hexahedron-based grid. In other words, both grids resolve the large vortices in the same way but the grid *gridhex* resolves much smaller eddies whereas they are dissipated by the grid *gridtet*.

Nevertheless, these differences in flow structures are difficult to quantify when plotting the radial mean and RMS axial velocity at the seven experimental stations as done in Fig. 13. The amplitude of the mean and RMS velocity is very similar for both grids, and in good agreement with the measurements. The most significant discrepancy is located in the central region close to the station $z = 80$ mm where the axial velocity fluctuations are over-estimated with *gridtet*. As a consequence, the inner jet flow penetrates slightly farther in the chamber, improving the prediction of the location of the first stagnation point as well as the length of the recirculation zone.

4.9. Compromise between scheme order and grid

Accounting for the cost efficiency slightly modifies the conclusions drawn in Sections 4.6 and 4.8. Table 5 compares the computational cost of the cases *avbp_tet_lw_norfg*, *avbp_tet_ttgc_rfg* and *avbp_hex_ttgc_rfg* when simulating 0.1 s physical time. Note that the RFG inlet treatment does not notably modify the computational cost so that the case *avbp_tet_ttgc_norfg* is not reported in Table 5. On the one hand, the computational cost of a simulation with TTGC is 2.5 bigger than with LW [4]. On the other hand, using the unstructured grid *gridtet* is about four times cheaper than using the hexahedron-based grid *gridhex*. This figure is to be related to the gain in quality of the predictions choosing the final configuration and grid.

4.10. Influence of the code

The accuracy of both AVBP and CDP codes is finally compared analysing the results provided by the cases *cdp_hex* and *avbp_hex_ttgc_rfg*. For this purpose, the case *avbp_hex_ttgc_rfg* has been chosen, not only because it shows the best agreement with the measurements but also because it is the best candidate for direct comparisons with CDP. Indeed, both codes use here the same hexahedron-based grid. There are however some differences in the parameters used, as summarised in

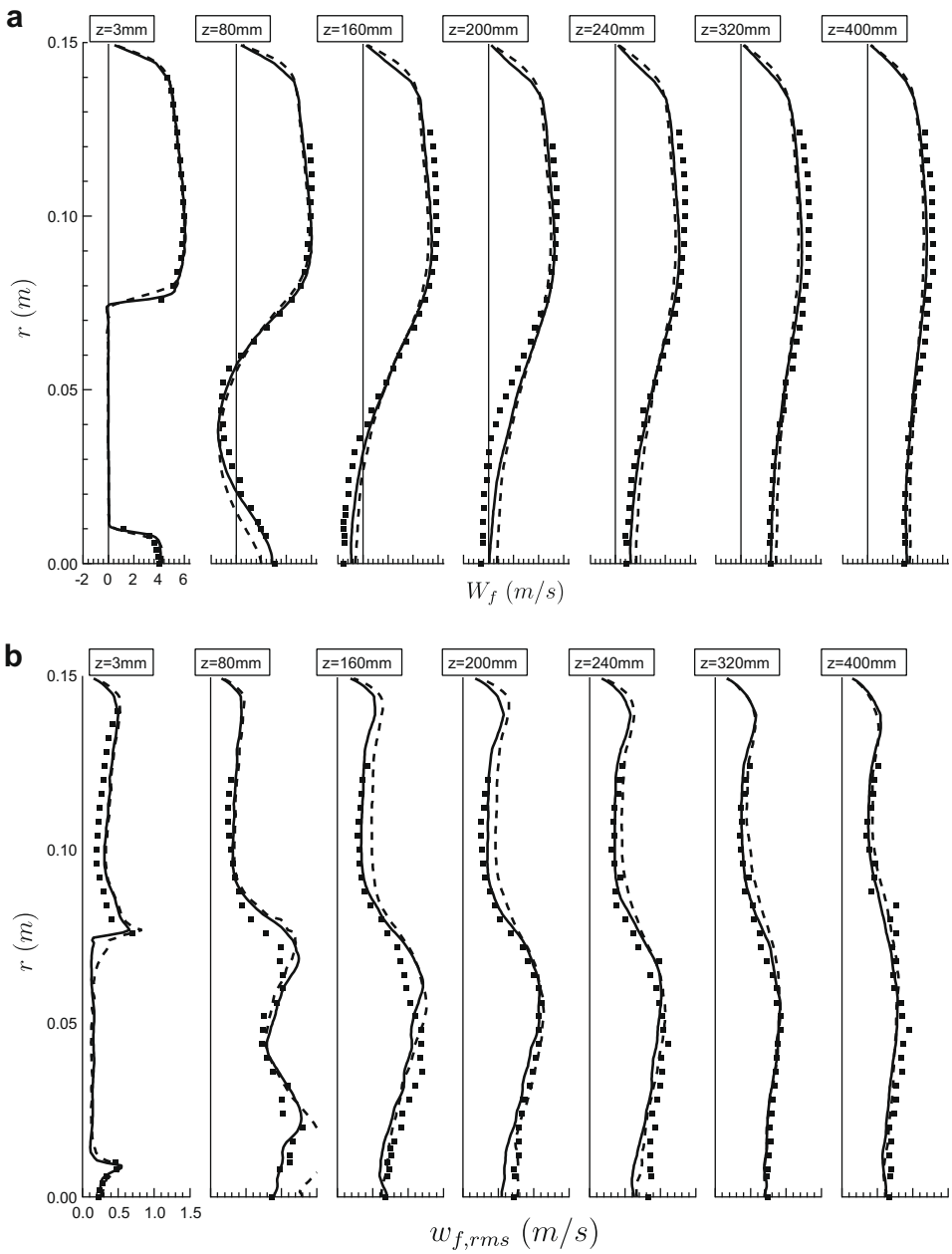


Fig. 13. Radial profiles of mean (a), and RMS (b), axial gas velocity. Symbols: experiment – dashed line: *gridtet* – solid line: *gridhex*. (Simulations performed with AVBP.)

Table 5
Code efficiency for single-phase flow calculations depending on the scheme and the grid. Statistics given for 0.1 s (physical time) computed with TTGC on 16 processors on a CRAY XD1 supercomputer.

Case	Total CPU time (μs)	Efficiency/iteration/node (μs)	Efficiency/iteration/cell (μs)
avbp_tet_lw_norfg	28,527	1.81	0.32
avbp_tet_ttgc_rfg	68,460	4.35	0.77
avbp_hex_ttgc_rfg	235,823	3.06	3.10

Table 6. As already mentioned, CDP solves the incompressible Navier–Stokes equations implicitly whereas AVBP solves the compressible Navier–Stokes equations explicitly. The main consequence is that the time step is 35 times larger for CDP, lead-

Table 6
Comparison of the parameters and models used for the single-phase flow LES performed with AVBP and CDP on the hexahedron-based grid *gridhex*.

Case	avbp_hex_ttgc_rfg	cdp_hex
Solver	AVBP	CDP
Time step (μs)/CFL	4.22/0.7	147/50
Averaging time (s)/iterations	1./192,000	2.65/18,000
Convective	TTGC (third order)	Second-order kinetic energy conserving
Scheme	[4]	[36]
SGS model/wall model	WALE/none	Dynamic Smagorinsky/none
Inner jet/coflow inlet BC	Forcing/forcing	Forcing/no forcing

ing to a reduced computational cost. As a result, the total averaging time is smaller for AVBP, but the convergence has been ensured to be good enough. Another noticeable difference is the treatment of the inlet boundary condition, as detailed in Section 4.3.

Fig. 14 displays mean and RMS gas axial velocities along the axis while Fig. 15 shows the radial profiles of mean and RMS gas axial velocities for the two codes along with the experimental measurements. The global agreement between the two codes and the experiments is very good and most of the flow physics is captured by the two LES codes. The width and the length of the recirculation zone are well predicted (see Fig. 15(a)). As shown in Fig. 14(a), the differences between both codes in predicting the location of the two stagnation points are minor. Focusing on the RMS velocities in Fig. 15(b), the agreement with measurements is also good. The location and the amplitude of the peaks are well predicted, except in the coflow where CDP underpredicts the RMS velocities. The origin of the discrepancy is the treatment of the coflow inlet boundary condition, with no turbulence injected with CDP in the outer duct.

The overall result is that both codes provide very similar results, also close to the measurements, even though they use different numerical methods. This indicates that the accuracy of both codes is good enough to test the dispersed phase with reasonable confidence on this configuration.

5. Comparison of two-phase flow simulations

This section presents the results for the 22% particle mass loading ratio of the central jet. Riber [33] showed that the impact of the dispersed phase on the gas phase is limited at this mass loading ratio: the central jet penetrates slightly further in the chamber, also slightly modifying the location of the recirculation zone. As the differences with the single-phase flow case are minor in the present case, the gas phase results are not discussed and only the results for the dispersed phase are provided hereafter. The validation of the particle dynamics modeling in this recirculating gas flow is done using three different solvers: CDP and AVBP-EL which both calculate the particle motion with the EL approach summarised in Section 3.2.1, and AVBP-EE that uses the simplified mesoscopic EE approach detailed in Section 3.2.2.

An essential part of these two-phase flow LES is the introduction of particles in terms of position and velocity in the central jet. This point is discussed in Section 5.1. Then, the three solvers are compared in two steps in Section 5.2. First, since the gas LES solvers from AVBP and CDP give very similar results on the hexahedron-based grid *gridhex*, the two Lagrangian solvers CDP and AVBP-EL are compared and validated by comparisons with the measurements. Second, the two approaches (EL and mesoscopic EE) are compared using the solvers AVBP-EL and AVBP-EE on *gridhex*. As the gas solver and the grid are exactly the same, a direct comparison of the two methods is proposed.

5.1. Particle injection and test cases

In all cases, the injected particles have a diameter of 60 μm, as justified in Section 2.3. The introduction of these particles in terms of position and velocity is one of the main difficulty in such a two-phase flow configuration. First, the methodology differs depending on the solver used. Second, the injection planes are different for all solvers, as shown in Fig. 16. Thus, the

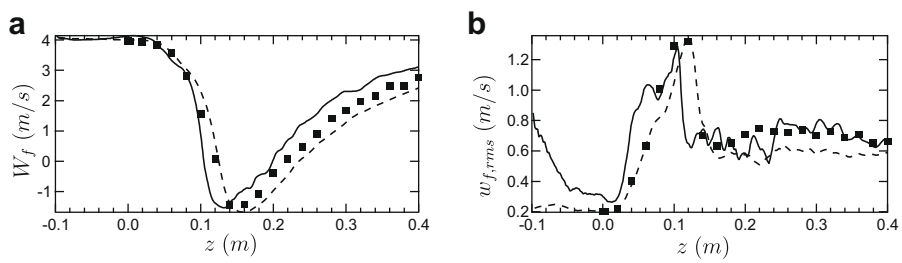


Fig. 14. Axial profiles of mean (a), and RMS (b), axial gas velocity. Symbols: experiment – solid line: AVBP – dashed line: CDP.

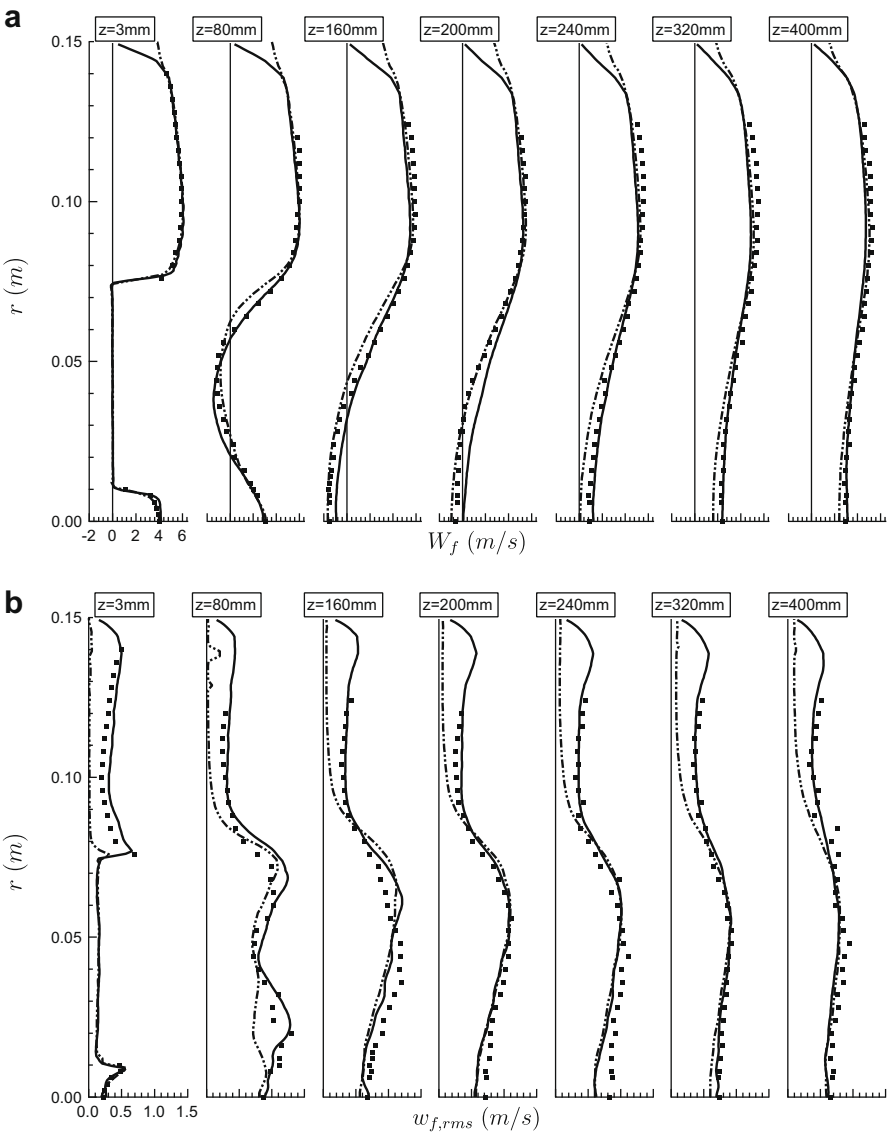


Fig. 15. Radial profiles of mean (a), and RMS (b), axial gas velocity. Symbols: experiment – solid line: AVBP – dot-dashed line: CDP.

impact of the particle injection method on the results can be evaluated. In AVBP-EE, both the particle number density profile and the mean velocity profile are imposed at the inner pipe inlet ($z = -100$ mm) and correspond to the ones measured experimentally at $z = 3$ mm. No turbulent fluctuations are introduced. In AVBP-EL and in CDP, the injection planes are located at $z = -3$ mm and $z = -95$ mm, respectively. The injection speed profile is also the experimental one measured at $z = 3$ mm but the mass loading is homogeneous over the injection section. Furthermore, in AVBP-EL and in CDP, a white noise (amplitude

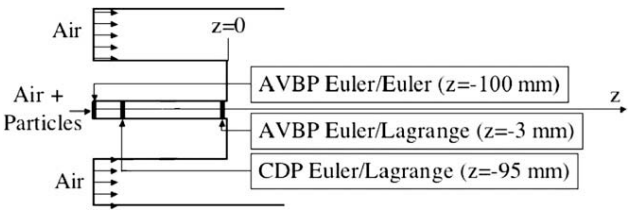


Fig. 16. Location of the particle injection depending on the two-phase flow solver used.

Table 7
Summary of two-phase flow test cases and parameters used for the particle injection.

	cdp_EL_hex	avbp_EL_hex	avbp_EE_hex
Averaging time (s)	4.0	0.5	1.0
Grid	*gridhex*	*gridhex*	*gridhex*
Particle mean axial velocity	Experimental profile	Experimental profile	Experimental profile
Turbulent fluctuations	White noise (10%)	White noise (12%)	Zero
Particle distribution	Homogeneous	Homogeneous	Experimental profile

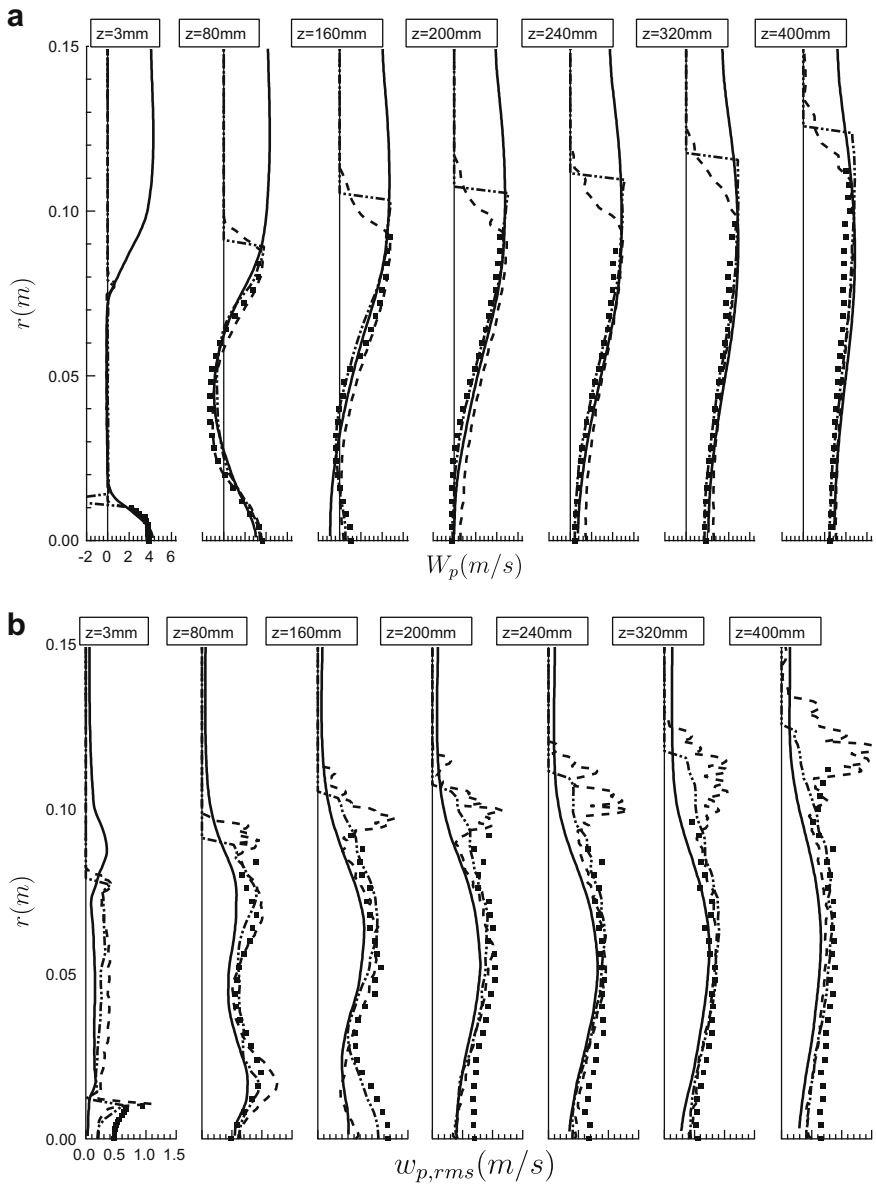


Fig. 17. Radial profiles of mean (a) and RMS (b) axial particle velocity. Symbols: experiment – solid line: AVBP-EE – dashed line: AVBP-EL – dot-dashed line: CDP-EL.

of the order of 10% of the mean velocity) is added to the particle mean velocity profiles to match experimental measurements at $z = 3$ mm.

These differences in particle injection are summarised in Table 7. Although they must be taken into account when comparing the three solvers AVBP-EL, AVBP-EE and CDP, the comparison of the Lagrangian solvers on the one hand, and the EL and mesoscopic EE approaches on the other hand is still relevant. The three LES performed with the three different solvers are also detailed in Table 7. The case *cdp_EL_hex* uses the same gaseous parameters as the case *cdp_hex* whereas the two cases *avbp_EL_hex* and *avbp_EE_hex* are based on the case *avbp_hex_ttgc_rfg* (see Table 6). Due to the implicit formulation of CDP, the total averaging time affordable with CDP is once again larger than with AVBP.

5.2. Results for two-phase flow simulations

Figs. 17 and 18 display the radial profiles of mean and RMS axial and radial particle velocities at the seven stations along the axis while Fig. 19 shows the axial profiles of mean and RMS axial particle velocities. The results of the three cases defined in Table 7 are directly compared with measurements. For the sake of simplicity, the analysis of the results is divided into two

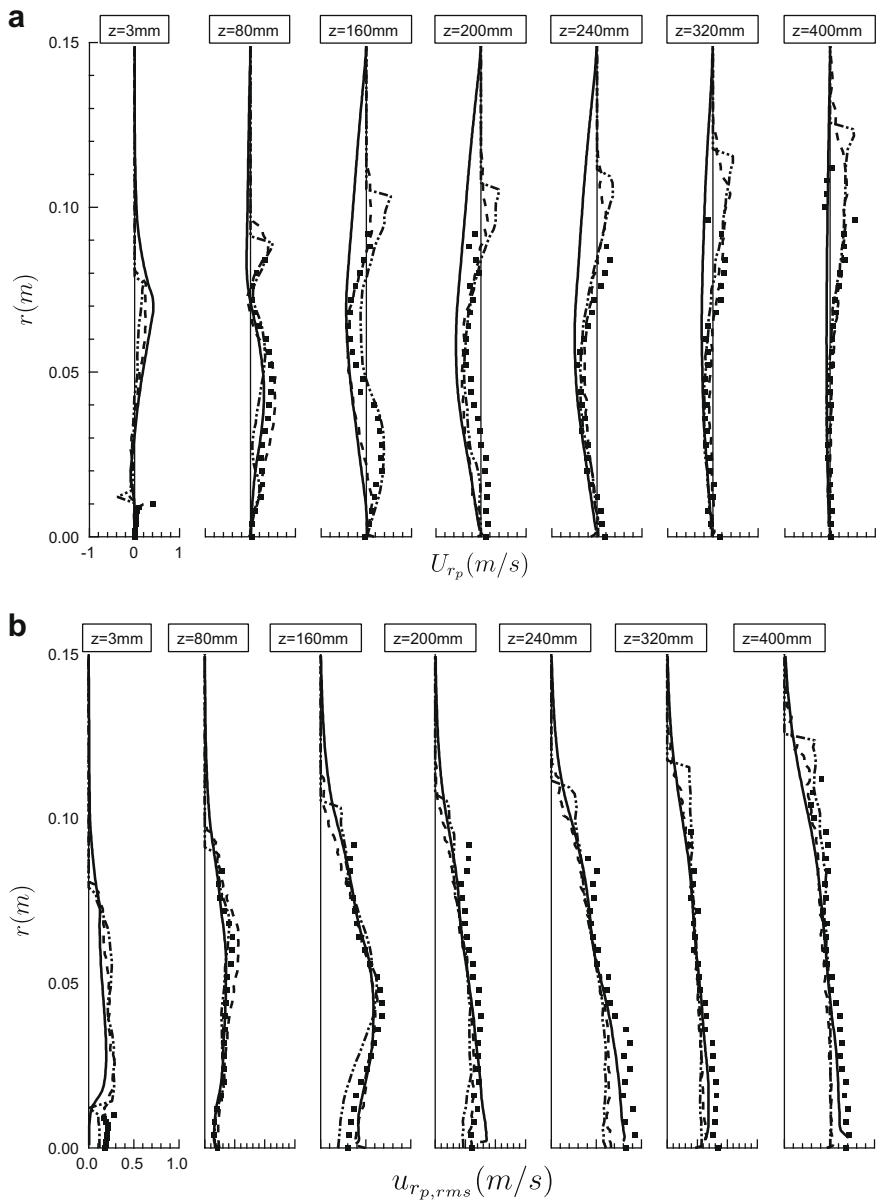


Fig. 18. Radial profiles of mean (a) and RMS (b) radial particle velocity. Symbols: experiment – solid line: AVBP-EE – dashed line: AVBP-EL – dot-dashed line: CDP-EL.

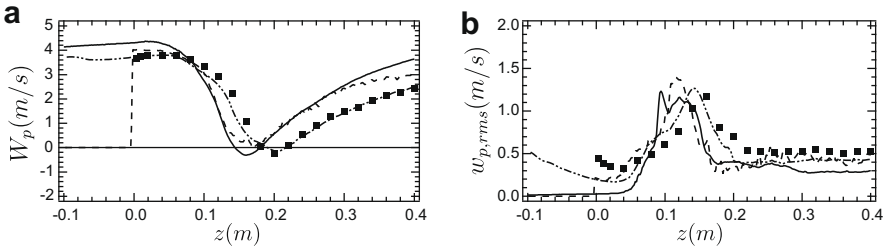


Fig. 19. Axial profiles of mean (a) and RMS (b) axial particle velocity. Symbols: experiment – solid line: AVBP-EE – dashed line: AVBP-EL – dot-dashed line: CDP-EL.

parts: first, the Lagrangian solvers are compared in Section 5.2.1. Second, the EL and mesoscopic EE approaches are directly compared in Section 5.2.2.

5.2.1. Comparison of the two EL solvers

Focusing on the results obtained with the two Lagrangian solvers, Figs. 17–19 show a very good agreement between the two codes, and with the measurements. Better results are generally obtained with CDP but not in all sections. A convenient way to look at the results is to consider the central axis of the configuration: a critical zone is located around $z = 160$ mm, the stagnation point for the gas phase. This is also a zone where particles accumulate and must stop before turning around to escape from the recirculating zone by the sides. As a consequence, this zone is difficult to predict accurately for the dispersed phase and the slight differences in calculating the turbulence in the inner pipe may induce such discrepancies on the dispersed phase.

5.2.2. Comparison of the EL and the EE approaches

In this section, the cases *avbp_EL_hex* and *avbp_EE_hex* are compared. As the two cases use the same grid and exactly the same gas solver, the EL approach and the mesoscopic EE approach can be directly compared.

First, a qualitative comparison is provided in Fig. 20 which displays instantaneous fields of particle number density for the two cases. Note that this quantity is directly available when using the mesoscopic EE approach but is reconstructed from the Lagrangian simulations using a volumic projection method. The two instantaneous fields of particle number density are similar: both approaches show several of particles along the inner jet and the largest one is located close to the second stagnation point. Most of the particles trapped in this region are then released in the gaseous recirculation zones. The two fields show however differences when focusing on the small structures which are more numerous using the EL approach.

Second, Figs. 17–19 show that the two approaches provide extremely similar results showing that the mesoscopic EE approach is able to compute such a flow and to provide results that are equivalent in precision to an EL computation. There are however at least two discrepancies. First, the mean particle axial velocity profiles at $z = 160$ mm show that the particles do not go far enough in the chamber with the mesoscopic EE approach. Nevertheless, this point has been shown to be very delicate to predict for the gas phase and very dependent on the inlet conditions in the pipe. In the present EE computation, no particle velocity fluctuations are imposed at the inlet of the inner pipe with the mesoscopic approach but this may not be crucial. The second main difference deals with the particle agitation that is under-estimated by the mesoscopic EE approach both on the radial and the axial profiles of particle RMS velocities. Actually, this is not surprising since the simplified mesoscopic EE model is used in this work: as the total particle agitation is divided into a correlated and an uncorrelated part, neglecting the RUM contribution in this bluff body flow prevents from predicting the experimental level of particle agitation. Nevertheless, the mean quantities are still well predicted with the mesoscopic EE model.

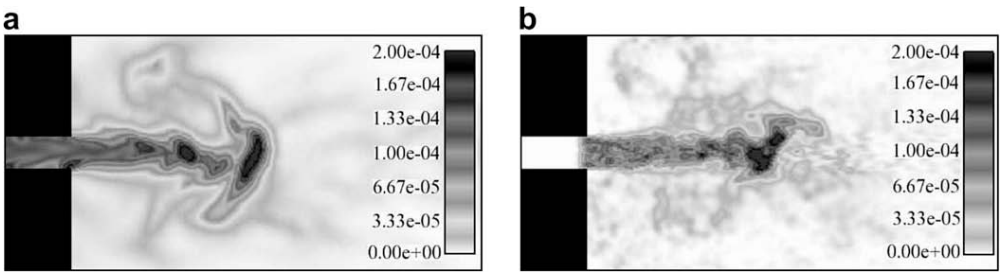


Fig. 20. Instantaneous particle volume fraction field in the central plane obtained with the cases *avbp_EE_hex* (a) and *avbp_EL_hex* (b).

5.2.3. Scalability of the EL approach

One interesting issue in the comparison between Eulerian and Lagrangian methods is the analysis of scalability and computational performance. The implementation on massively parallel machines of the Eulerian approaches is not technically problematic because the flow and the droplets are solved on the same grid using the same spatial discretization scheme. On the other hand, Lagrangian approaches are less well-suited to parallel computers since the two phases must be coupled, which increases the complexity of the implementation. Thus, before implementing a Lagrangian module into a new solver one of these two strategies should be adopted for the dispersed phase treatment:

(1) Task parallelization in which some processors compute the gaseous phase and others compute the droplets.
(2) Domain partitioning in which particles are computed together with the gas flow on geometrical subdomains mapped on parallel processors. Individual particles or parcels are tracked as they cross the computational domain and must be exchanged between processors when leaving a subdomain to enter an adjacent one.

Particle tracking within an unstructured solver reveals an additional constraint since particle coordinates cannot be easily used to locate them inside a cell, and time (and memory) consuming searching algorithm must be used. However, for LES it is easy to show that only the strategy based on domain partitioning is efficient on large grids because task parallelization would require the communication of very large three-dimensional data sets at each iteration between all processors. It is well known that codes based on domain partitioning are difficult to optimize on massively parallel architectures when droplets are clustered in one part of the domain (typically, near the fuel injectors) due to load imbalance. Moreover, the distribution of droplets may change during the computation: for a gas turbine reignition sequence, for example, the chamber is filled with droplets when the ignition begins thus ensuring an almost uniform droplet distribution; these droplets then evaporate rapidly during the computation, leaving droplets only in the near injector regions. This may lead to a poor speedup on a parallel machine if the domain is decomposed in the same way for the entire computation since some processors should compute a high number of particles while other are waiting for this task to finish. As a result, load balancing strategies are required to redecompose the domain by taking into account particles information to preserve a high parallel efficiency [14].

In this section, the scalability of the AVBP-EL solver is analysed using the simulation speedup and the reference single-phase CPU time ratio. The former is defined as the ratio between the CPU time of a simulation with n processors and the CPU time of a simulation with a given number of processors, N_{procs}:

$$\text{Speedup} = \frac{T_{\text{run}}(n)}{T_{\text{run}}(N_{\text{procs}})} * n. \tag{16}$$

The latter is defined as the ratio between the CPU time of a simulation with a given number of processors and the CPU time of the reference single-phase simulation with n processors:

$$\text{CPU time ratio} = \frac{T_{\text{run}}(N_{\text{procs}})}{T_{\text{single-phase}}(n)}. \tag{17}$$

Default value of n is 1 but sometimes it is not possible to run the sequential simulation mainly due to high memory requirements, in which case the speedup and CPU time ratio are reported to a reference parallel simulation. This scalability study has been performed in a CRAY XD1[2] supercomputer for a number of processors up to 64. Results of the moderate mass loading test case for the two grids named *gridtet* and *gridhex* are presented hereafter. The total number of particles presented in the domain is of the order of 560,000 and 430,000, respectively. Variations smaller than 0.5% in the number of particles were observed between the beginning and the end of the simulation, which implies that it is statistically stationary. The lower number of particles used in *gridhex* is due to the difference in the length between the two grids (see Table 3). Even if the length of the chamber is half the dimension of *gridtet*, the number of particles is three-fourths since they are mainly concentrated inside the recirculation zone.

Fig. 21 shows the speedup of the single-phase and the monodisperse test case with *gridtet* (Fig. 21(a)) and *gridhex* (Fig. 21(b)) using AVBP-EL. Scaling of *gridhex* is reported relative to the 8 processor case ($n = 8$), which was the smallest number of CRAY XD1 processors that could run this problem due to high memory requirements. In both cases, the good scalability of the single-phase is unquestionable. The drop of performances observed in Fig. 21(a) for the two-phase flow simulation is not related to large communications costs between processors but merely to the parallel load imbalance generated by the partitioning algorithm [15]. The same simulation with a different grid can lead to a completely different speedup graph, as it can be observed from Fig. 21(b). Note that these graphs do not contain information about the speedup with AVBP-EE. It can be considered as good as the single-phase computation since the dispersed phase uses the same parallelization applied to the gaseous phase.

The differences between the two speedup graphs can be explained by plotting the number of nodes (or cells) and particles presented in each processor. As AVBP is based on a *cell-vertex* formulation, comparing the number of nodes to the number of particles is more representative of the computational loading since almost all arrays are dimensioned as a function of the

[2] This machine has 58 nodes with 2 processors/node and 2 GB/processor.

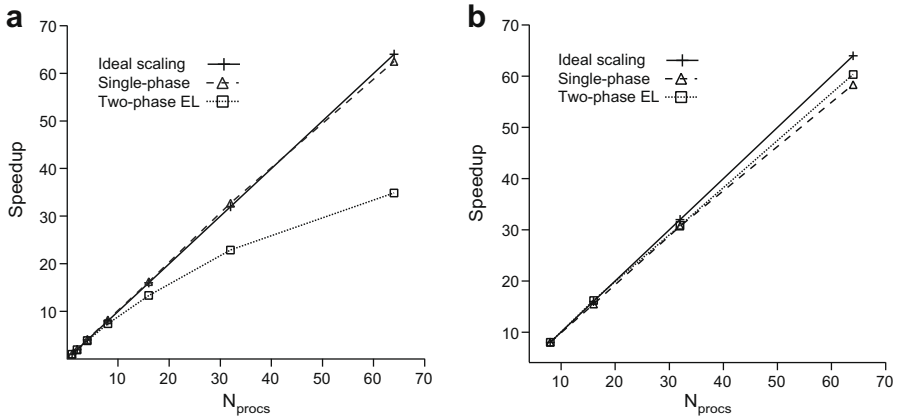


Fig. 21. Speedup of the single-phase and the monodisperse test case with *gridtet* (a) and *gridhex* (b) on a CRAY XD1 supercomputer. (Simulations performed with AVBP-EL.)

number of nodes per processor. Fig. 22 reports the number of nodes and particles per processor for a 32-partition simulation by using a recursive inertial bisection (RIB) [81] partitioning algorithm, with *gridtet* and *gridhex*, respectively. As expected from Fig. 21, Fig. 22 shows an excellent load balancing for the gaseous phase: all processors contain about the same number of nodes. On the contrary, it shows a strong particle load imbalance (Fig. 22(a)) where one single processor contains almost half the total number of particles of the simulation. This fact is related to the grid downstream coarsening which increases significantly the memory requirements and the floating-point operations for this processor. This points out the need of load

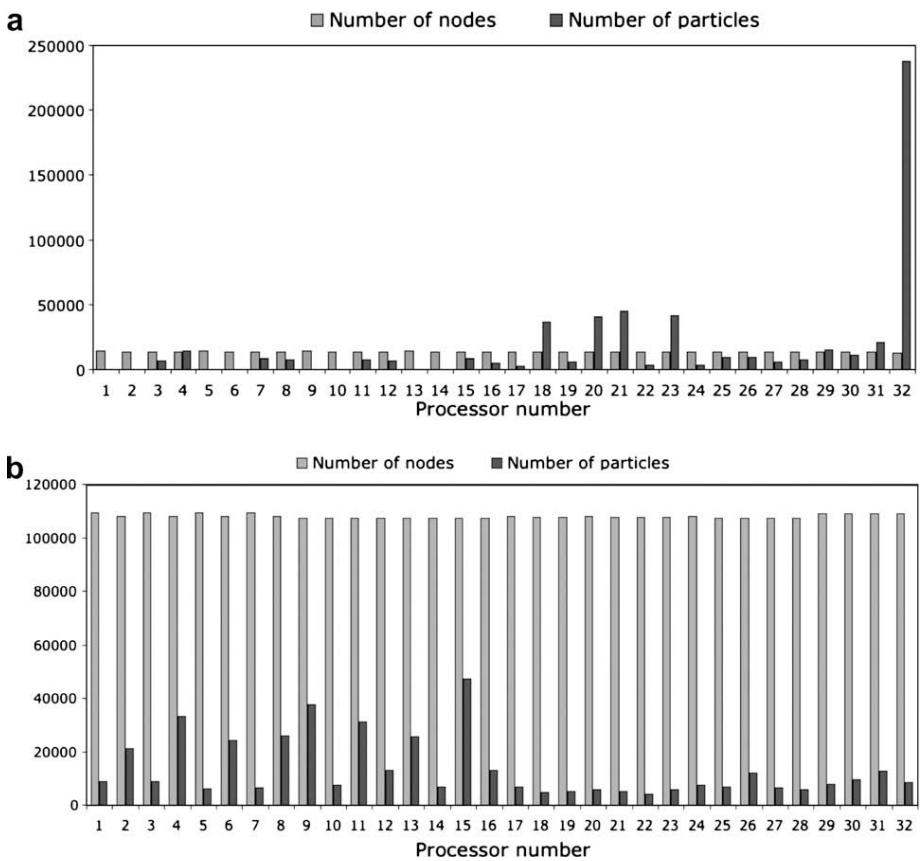


Fig. 22. Number of nodes and particles per processor for a 32-partition by using a recursive inertial bisection (RIB) partitioning algorithm for *gridtet* (a) and *gridhex* (b). (Simulations performed with AVBP-EL.)

Table 8
Summary of the CPU time ratios of AVBP-EL with *gridtet* on a CRAY XD1 supercomputer.

N_{procs}	1	2	4	8	16	32	64
Single-phase	1	0.50	0.25	0.12	0.06	0.030	0.016
Two-phase EL	1.05	0.54	0.27	0.14	0.08	0.046	0.030

Table 9
Summary of the CPU time ratios of AVBP-EL with *gridhex* on a CRAY XD1 supercomputer.

N_{procs}	8	16	32	64
Single-phase	1	0.51	0.26	0.137
Two-phase EL	1.06	0.524	0.275	0.14

balancing strategies for two-phase flow simulations with a Lagrangian approach, for example, by using multi-constraint partitioning algorithms which take into account particle loading on each processor [14]. Even if the dispersed phase presents a small load imbalance in Fig. 22(b), it is hidden by the higher computational loading needed to calculate the gaseous phase. This is the main reason of the good speedup observed in Fig. 21(b).

Tables 8 and 9 summarised the CPU time ratios of AVBP-EL with both grids on a CRAY XD1 supercomputer. Additional time to perform the two-phase flow simulation can vary from 5% up to 87% with the Lagrangian formulation in *gridtet* but it is not higher than 5% with *gridhex* which confirms the tendencies observed with the speedup graphs. All simulations with AVBP-EE has a constant added cost of the order of 80% since this approach is independent of the mass loading. Therefore, at this moderate mass loading the Lagrangian approach proved to be faster than the Eulerian formulation.

5.3. Discussion

The previous results require additional comments:

- The quality of the gas phase prediction is essential for the dispersed phase results.
- The comparison of different solvers and boundary conditions demonstrates the importance of the flow in the central injection pipe: the mean and RMS gas velocity profiles as well as the particle motion directly change the flow field significantly. For example, the location of the stagnation points (see Fig. 14) is extremely sensitive to any change in the inlet pipe boundary conditions. This is a major drawback of this configuration because performing an accurate LES of the two-phase flow within this pipe is still out of reach of present capacities for at least two reasons. First, the gaseous flow within the pipe is difficult to resolve and would require a very fine mesh and sophisticated wall models. Second, the two-phase flow in the pipe is heavily loaded so that computing the motion of the particles within the pipe would require additional models for particle–particle and wall–particle interactions [35]. For the present results, the boundary conditions for the dispersed phase at the tube inlet are approximate solutions and the variety of methods tested for the LES of Table 7 demonstrates that no easy solution was found to model particles injection in the central tube.

Despite these limitations and differences in inlet conditions, results confirm that most of the flow structures are correctly captured and that LES of two-phase flows (using EL or mesoscopic EE approaches) is possible and provides accurate results. To improve on these results, a significant effort will have to be applied to describe the two-phase flow within the injection tube. Note that in a real combustion chamber, the region where such heavily loaded flows are found is very small and limited to the vicinity of the fuel injector.

6. Conclusions

This work has presented a comparison between multiple LES approaches and codes for non-reacting two-phase flows and the experimental results of Borée et al. [1]. This experiment was chosen because it contains multiple complex flow features which are typical of combustion chambers: strong recirculating zones created by a dump geometry, multiple stagnation points, high turbulent Reynolds number. LES and experiments have been compared in terms of radial profiles of mean and RMS axial and radial velocities at seven stations along the axis, and axial profiles of mean and RMS axial velocities on the experiment central axis. LES approaches included both Euler/Euler and Euler/Lagrange models. Two codes (incompressible implicit and compressible explicit) were used to evaluate the effects of the gas solver on the results so that three two-phase flow solvers were used:

- an Euler/Euler explicit compressible solver;
- an Euler/Lagrange explicit compressible solver;
- an Euler/Lagrange implicit incompressible solver.

All computations were performed for monodisperse particle distributions with diameter $d_p = 60$ μm which corresponds to the initial mean experimental diameter in mass. The effects of subgrid-scale model, numerical scheme accuracy and of grid refinement were also investigated. The main conclusions are the following:

- All LES approaches provide high-quality results compared to the PDA data of the configuration of Borée et al. [1], thereby confirming the potential of these methods and their relative insensitivity to the details of the numerical solver. The variations between LES solvers are mainly due to changes in boundary conditions.
- The quality of the gas flow predictions is critical in order to accurately compute the dispersed phase.
- The most critical parameter controlling accuracy is the grid and the convective scheme: using high-order schemes built for LES provides the best results.
- The most critical zone in the flow is the central injection tube which feeds the experiment with a highly loaded air/particles jet. True two-phase LES is impossible in this duct today. Results show that this element controls the flow, especially the positions of stagnation points on the experiment axis. Even though the quality of the present LES results is high in most of the flow, this work confirms that future LES accuracy in combustion chambers will need to include all feeding ducts which represents a significant challenge.
- Both Euler/Lagrange codes provide very similar results while the Euler/Euler approach gives similar mean velocity fields but slightly under-estimates fields of particle agitation. This corresponds to the expected behavior of the present mesoscopic model in which the uncorrelated motion was not taken into account [32,33].
- For the present case with moderate mass loading, the total number of particles per processor is moderate. The CPU and memory requirements to track particles is lower than for the gaseous phase even with the load balancing problem observed with one of the grids studied. Although the additional cost of Eulerian formulations is independent of the mass loading, for such a dilute case, the Lagrangian approach proved to be faster up to 64 processors.

These results confirm the potential of LES approaches for two-phase flows. Nevertheless, there is still much work to be done concerning:

- the effects of the subgrid fluid turbulence on the particle velocity [53];
- the RUM modeling in order to estimate the particle agitation more precisely with the mesoscopic EE approach;
- the wall–particle interactions as well as the particle–particle interactions to simulate the dispersed phase accurately in pipe and channel flows;
- the extension to two-phase reacting flows which has already been initiated [20].

Acknowledgments

Part of this work was performed during the 2006 CTR Summer Program at Stanford and the support of Pr Pitsch and Moin is gratefully acknowledged. Most simulations were performed on the national computing center CINES in Montpellier and on CTR supercomputers.

References

[1] J. Borée, T. Ishima, I. Flour, The effect of mass loading and inter-particle collisions on the development of the polydispersed two-phase flow downstream of a confined bluff body, J. Fluid Mech. 443 (2001) 129–165.
[2] D. Caraeni, C. Bergstrom, L. Fuchs, Modeling of liquid fuel injection, evaporation and mixing in a gas turbine burner using large Eddy simulation, Flow Turbul. Combust. 65 (2000) 223–244.
[3] V. Chakravarthy, S. Menon, Subgrid modeling of turbulent premixed flames in the flamelet regime, Flow Turbul. Combust. 65 (2000) 133–161.
[4] O. Colin, F. Ducros, D. Veynante, T. Poinsot, A thickened flame model for large Eddy simulations of turbulent premixed combustion, Phys. Fluids 12 (7) (2000) 1843–1863.
[5] H. Forkel, J. Janicka, Large Eddy simulation of a turbulent hydrogen diffusion flame, Flow Turbul. Combust. 65 (2) (2000) 163–175.
[6] H. Pitsch, L. Duchamp de la Geneste, Large Eddy simulation of premixed turbulent combustion using a level-set approach, Proc. Combust. Inst. 29 (2002) 2001–2005.
[7] K. Mahesh, G. Constantinescu, P. Moin, A numerical method for large Eddy simulation in complex geometries, J. Comput. Phys. 197 (1) (2004) 215–240.
[8] L. Selle, G. Lartigue, T. Poinsot, R. Koch, K.-U. Schildmacher, W. Krebs, B. Prade, P. Kaufmann, D. Veynante, Compressible large Eddy simulation of turbulent combustion in complex geometry on unstructured meshes, Combust. Flame 137 (4) (2004) 489–505.
[9] V. Moureau, G. Lartigue, Y. Sommerer, C. Angelberger, O. Colin, T. Poinsot, High-order methods for DNS and LES of compressible multi-component reacting flows on fixed and moving grids, J. Comput. Phys. 202 (2) (2005) 710–736.
[10] S. Roux, G. Lartigue, T. Poinsot, U. Meier, C. Bérat, Studies of mean and unsteady flow in a swirled combustor using experiments, acoustic analysis and large Eddy simulations, Combust. Flame 141 (2005) 40–54.
[11] T. Poinsot, D. Veynante, Theoretical and Numerical Combustion, second ed., R.T. Edwards, 2005.
[12] P. Schmitt, T. Poinsot, B. Schuermans, K. Geigle, Large Eddy simulation and experimental study of heat transfer, nitric oxide emissions and combustion instability in a swirled turbulent high pressure burner, J. Fluid Mech. 570 (2007) 17–46.
[13] G. Staffelbach, L. Gicquel, T. Poinsot, Highly parallel large Eddy simulations of multiburner configurations in industrial gas turbines, Lect. Notes Comput. Sci. Eng. – Complex Effects LES 56 (2006) 326–336.
[14] F. Ham, S. Apte, G. Iaccarino, X. Wu, M. Herrmann, G. Constantinescu, K. Mahesh, P. Moin, Unstructured LES of reacting multiphase flows in realistic gas turbine combustors, in: Annual Research Briefs, Center for Turbulence Research, NASA Ames/Stanford Univ., 2003, pp. 139–160.
[15] M. García, Y. Sommerer, T. Schönfeld, T. Poinsot, Evaluation of Euler–Euler and Euler–Lagrange strategies for large-Eddy simulations of turbulent reacting flows, in: ECCOMAS Thematic Conference on Computational Combustion, Lisbon, Portugal, 2005.

[16] L. Chin, R. Tankin, Vortical structures from a two-dimensional nozzle with a bluff body slot, Phys. Fluids A 4 (8) (1992) 1724–1736.
[17] R. Schefer, M. Namazian, J. Kelly, Velocity measurements in turbulent bluff body stabilized flows, AIAA J. 32 (9) (1994) 1844–1851.
[18] M. Namazian, J. Kelly, R. Schefer, Concentration imaging measurements in turbulent concentric-jet flows, AIAA J. 30 (1992) 384–394.
[19] Y. Hardalupas, C. Liu, J. Whitelaw, Experiments with disk stabilised kerosene-fuelled flames, Combust. Sci. Technol. 97 (1994) 157–191.
[20] M. Boileau, S. Pascaud, E. Riber, B. Cuenot, L. Gicquel, T. Poinsot, Large Eddy simulation of spray combustion in gas turbines, Flow Turbul. Combust. 80 (2008) 351–373.
[21] Q. Wang, K. Squires, Large Eddy simulation of particle-laden turbulent channel flow, Phys. Fluids 8 (5) (1996) 1207–1223.
[22] M. Vance, K. Squires, O. Simonin, Properties of the particle velocity field in gas–solid turbulent channel flow, Phys. Fluids 18 (2006). 063302.
[23] M. Picciotto, C. Marchioli, A. Soldati, Characterization of near-wall accumulation regions for inertial particles in turbulent boundary layers, Phys. Fluids 17 (2005). 098101.
[24] C. Marchioli, M. Picciotto, A. Soldati, Particle dispersion and wall-dependent fluid scales in turbulent bounded flow: implications for local equilibrium models, J. Turbul. 7 (60) (2006) 1–12.
[25] M. Sakiz, O. Simonin, Development and validation of continuum particle wall boundary conditions using Lagrangian simulation of a vertical gas–solid channel flow, in: Proceedings of the Eighth International Symposium on Gas-Particle Flows, ASME Fluids Engineering Division Summer Meeting, vol. FEDSM99-7898, 1999.
[26] V. Alipchenkov, L. Zaichik, O. Simonin, A comparison of two approaches to derivation of boundary conditions for continuous equations of particle motion in turbulent flow, High Temperature, Translated from Teplofizika Vysokikh Temperatur 39 (2001) 108–114.
[27] O. Simonin, J. He, Eulerian prediction of the particle behaviour in a turbulent boundary layer, in: M. Sommerfeld (Ed.), Proceedings of the Sixth Workshop on Two-Phase Flow Predictions, Erlangen 1992, Bilateral Seminars of the International Bureau/Forschungszentrum Juelich GmbH, vol. 14, Erlangen, Germany, 1993, pp. 154–165.
[28] G. Desoutter, B. Cuenot, C. Habchi, T. Poinsot, Interaction of a premixed flame with a liquid fuel film on a wall, Proc. Combust. Inst. 30 (2005) 259–266.
[29] W. Bachalo, M. Houser, Phase/doppler spray analyser for simultaneous measurements of drop size and velocity distributions, Opt. Eng. 23 (1984) 583–590.
[30] R.W. Schefer, M. Namazian, J. Kelly, Stabilization of lifted turbulent-jet flames, Combust. Flame 99 (1994) 75–86.
[31] A. Kaufmann, O. Simonin, B. Cuenot, T. Poinsot, J. Hélie, Dynamics and dispersion in 3D unsteady simulations of two phase flows, in: Supercomputing in Nuclear Applications, CEA, Paris, 2003.
[32] P. Février, O. Simonin, K. Squires, Partitioning of particle velocities in gas–solid turbulent flows into a continuous field and a spatially uncorrelated random distribution: theoretical formalism and numerical study, J. Fluid Mech. 533 (2005) 1–46.
[33] E. Riber, Développement de la méthode de simulation aux grandes échelles pour les écoulements diphasiques turbulents, Ph.D. Thesis, INP Toulouse, 2007.
[34] R. Fan, D.L. Marchisio, R.O. Fox, Application of the direct quadrature method of moments to polydisperse gas–solid fluidized beds, Powder Technol. 139 (7) (2004).
[35] N. Caraman, J. Borée, O. Simonin, Effect of collisions on the dispersed phase fluctuation in a dilute tube flow: experimental and theoretical analysis, Phys. Fluids 15 (12) (2003) 3602–3612.
[36] F. Ham, G. Iaccarino, Energy conservation in collocated discretization schemes on unstructured meshes, in: Annual Research Briefs, Center for Turbulence Research, NASA Ames/Stanford Univ., 2004, pp. 3–14.
[37] O. Colin, M. Rudgyard, Development of high-order Taylor–Galerkin schemes for unsteady calculations, J. Comput. Phys. 162 (2) (2000) 338–371.
[38] F. Nicoud, F. Ducros, Subgrid-scale stress modelling based on the square of the velocity gradient, Flow Turbul. Combust. 62 (3) (1999) 183–200.
[39] J. Kim, P. Moin, Application of a fractional-step method to incompressible Navier–Stokes equations, J. Comput. Phys. 59 (2) (1985) 308–323.
[40] M. Germano, U. Piomelli, P. Moin, W. Cabot, A dynamic subgrid-scale Eddy viscosity model, Phys. Fluids 3 (7) (1991) 1760–1765.
[41] M. Maxey, The gravitational settling of aerosol particles in homogeneous turbulence and random flow fields, J. Fluid Mech. 174 (1987) 441–465.
[42] R. Gatignol, The Faxén formulae for a rigid particle in an unsteady non-uniform Stokes flow, J. Méc. Théor. Appl. 1 (2) (1983) 143–160.
[43] M. Boivin, O. Simonin, K. Squires, Direct numerical simulation of turbulence modulation by particles in isotropic turbulence, J. Fluid Mech. 375 (1998) 235–263.
[44] O. Vermorel, B. Bédat, O. Simonin, T. Poinsot, Numerical study and modelling of turbulence modulation in a particle laden slab flow, J. Turbul. 4 (2003) 025.
[45] S. Elghobashi, G. Truesdell, On the two-way interaction between homogeneous turbulence and dispersed solid particles, Phys. Fluids 5 (7) (1993) 1790–1801.
[46] K. Squires, J. Eaton, Particle response and turbulence modification in isotropic turbulence, Phys. Fluids 2 (7) (1990) 1191–1203.
[47] R. Miller, J. Bellan, Direct numerical simulation and subgrid analysis of a transitional droplet laden mixing layer, Phys. Fluids 12 (3) (1999) 650–671.
[48] M. Boivin, O. Simonin, K. Squires, On the prediction of gas–solid flows with two-way coupling using large Eddy simulation, Phys. Fluids 12 (8) (2000) 2080–2090.
[49] L. Schiller, A. Nauman, A drag coefficient correlation, VDI Zeitung 77 (1935) 318–320.
[50] M. Maxey, J. Riley, Equation of motion for a small rigid sphere in a nonuniform flow, Phys. Fluids 26 (4) (1983) 883–889.
[51] Y. Yamamoto, M. Potthoff, T. Tanaka, T. Kajishima, Y. Tsuji, Large-Eddy simulation of turbulent gas-particle flow in a vertical channel: effect of considering inter-particle collisions, J. Fluid Mech. 442 (2001) 303–334.
[52] S. Apte, K. Mahesh, T. Lundgren, A Eulerian–Lagrangian model to simulate two-phase particulate flows, in: Annual Research Briefs, Center for Turbulence Research, NASA Ames/Stanford Univ., 2003, pp. 161–167.
[53] P. Fede, O. Simonin, Numerical study of the subgrid fluid turbulence effects on the statistics of heavy colliding particles, Phys. Fluids 18 (2006) 045103.
[54] O. Druzhinin, S. Elghobashi, On the decay rate of isotropic turbulence laden with microparticles, Phys. Fluids 11 (3) (1999) 602–610.
[55] J. Ferry, E. Balachandar, A fast Eulerian method for disperse two-phase flow, Int. J. Multiphase Flow 27 (2001) 1199–1226.
[56] O. Druzhinin, On the two-way interaction in two-dimensional particle-laden flows – the accumulation of particles and flow modification, J. Fluid Mech. 297 (1995) 49–76.
[57] S. Rani, E. Balachandar, Evaluation of the equilibrium Eulerian approach for the evolution of particle concentration in isotropic turbulence, Int. J. Multiphase Flow 29 (2003) 1793–1816.
[58] S. Rani, E. Balachandar, Preferential concentration of particles in isotropic turbulence: a comparison of the Lagrangian and the equilibrium Eulerian approaches, Int. J. Multiphase Flow 141 (1–2) (2004) 109–118.
[59] S. Chapman, T. Cowling, The Mathematical Theory of Non-Uniform Gases, Cambridge Mathematical Library ed., Cambridge University Press, 1939.
[60] O. Simonin, Gas particules, Cours d'options, École Nationale Superieure d'Electrotechnique, d'Electronique, d'Informatique, d'Hydraulique et des Télécommunications, 2002.
[61] P. Février, O. Simonin, Statistical and continuum modelling of turbulent reactive particulate flows. Part 2: Application of a two-phase second-moment transport model for prediction of turbulent gas-particle flows, in: Theoretical and Experimental Modeling of Particulate Flows, Lecture Series 2000–2006, von Karman Institute for Fluid Dynamics, Rhode Saint Genese, Belgium, 2000.
[62] M. Moreau, B. Bédat, O. Simonin, A priori testing of subgrid stress models for Euler–Euler two-phase LES from Euler–Lagrange simulations of gas-particle turbulent flow, in: 18th Annual Conference on Liquid Atomization and Spray Systems, ILASS Americas, 2005.
[63] A. Favre, Statistical equations of turbulent gases, in: Problems of Hydrodynamics and Continuum Mechanics, SIAM, Philadelphia, 1969, pp. 231–266.
[64] O. Simonin, P. Février, J. Laviéville, On the spatial distribution of heavy-particle velocities in turbulent flow: from continuous field to particulate chaos, J. Turbul. 3 (2002) 040.

[65] E. Riber, M. Moreau, O. Simonin, B. Cuenot, Towards large Eddy simulation of non-homogeneous particle laden turbulent gas flows using Euler–Euler approach, in: 11th Workshop on Two-Phase Flow Predictions, Merseburg, Germany, 2005.
[66] P. Moin, K. Squires, W. Cabot, S. Lee, A dynamic subgrid-scale model for compressible turbulence and scalar transport, Phys. Fluids A 3 (11) (1991) 2746–2757.
[67] B. Vreman, B. Geurts, H. Kuerten, Subgrid modeling in LES of compressible flow, Appl. Sci. Res. 54 (1995) 191–203.
[68] A. Yoshizawa, Statistical theory for compressible turbulent shear flows, with the application to subgrid modeling, Phys. Fluids 29 (7) (1986) 2152–2164.
[69] E. Riber, M. García, V. Moureau, H. Pitsch, O. Simonin, T. Poinsot, Evaluation of numerical strategies for LES of two-phase reacting flows, in: Proceedings of the Summer Program, Center for Turbulence Research, NASA Ames/Stanford Univ., 2006, pp. 197–211.
[70] A. Kaufmann, M. Moreau, O. Simonin, J. Hélie, Comparison between Lagrangian and mesoscopic Eulerian modelling approaches for inertial particles suspended in decaying isotropic turbulence, J. Comput. Phys. 227 (13) (2008) 6448–6472.
[71] K. Hishida, K. Takemoto, M. Maeda, Turbulent characteristics of gas–solids two-phase confined jet, Jpn. J. Multiphase Flow 1 (1) (1987) 56–69.
[72] T. Poinsot, S. Lele, Boundary conditions for direct simulation of compressible viscous flows, J. Comput. Phys. 101 (1992) 104–129.
[73] I. Celik, A. Smirnov, J. Smith, Appropriate initial and boundary conditions for LES of a ship wake, in: Proceedings of the Third ASME/JSME Joint Fluids Engineering Conference, San Francisco, California, USA, vol. FEDSM99-7851, 1999.
[74] A. Smirnov, S. Shi, I. Celik, Random flow generation technique for large Eddy simulations and particle-dynamics modeling, Trans. ASME J. Fluids Eng. 123 (2001) 359–371.
[75] A. Sengissen, T. Poinsot, J. VanKampen, J. Kok, Response of a swirled non-premixed burner to fuel flow rate modulation, Lect. Notes Comput. Sci. Eng. – Complex Effects LES 56 (2006) 337–351.
[76] B. Vreman, Direct and large Eddy simulation of the compressible turbulent mixing layer, Ph.D. Thesis, University of Twente, 1995.
[77] J. Jiménez, P. Moin, The minimal flow unit in near-wall turbulence, J. Fluid Mech. 225 (1991) 213–240.
[78] Y. Dubief, F. Delcayre, On coherent-vortex identification in turbulence, J. Turbul. 1 (2000) 1–22.
[79] J. Hunt, A. Wray, P. Moin, Eddies, stream, and convergence zones in turbulent flows, in: Proceedings of the Summer Program, Center for Turbulence Research, NASA Ames/Stanford Univ., 1988, pp. 193–208.
[80] F. Hussain, J. Jeong, On the identification of a vortex, J. Fluid Mech. 285 (1995) 69–94.
[81] R.D. Williams, Performance of dynamic load balancing algorithms for unstructured mesh calculations, Concurr.: Pract. Exp. 3 (5) (1991) 451–481.

INSTITUT NATIONAL POLYTECHNIQUE DE TOULOUSE

Doctorat d'Université, spécialité *Dynamique des Fluides*

19 Janvier 2009

Marta GARCÍA

Développement et validation du formalisme Euler-Lagrange dans un solveur parallèle et non-structuré pour la simulation aux grandes échelles

De nombreuses applications industrielles mettent en jeu des écoulements gaz-particules, comme les turbines aéronautiques et les réacteurs à lit fluidifié de l'industrie chimique. La prédiction des propriétés de la phase dispersée, est essentielle à l'amélioration et la conception des dispositifs conformément aux nouvelles normes européennes des émissions polluantes. L'objectif de cette thèse est de développer le formalisme Euler-Lagrange dans un solveur parallèle et non-structuré pour la simulation aux grandes échelles pour ce type d'écoulements. Ce travail est motivé par l'augmentation rapide de la puissance de calcul des machines massivement parallèles qui ouvre une nouvelle voie pour des simulations qui étaient prohibitives il y a une décennie. Une attention particulière a été portée aux structures de données afin de conserver une certaine simplicité et la portabilité du code sur des differentes architectures. Les développements sont validés pour deux configurations : un cas académique de turbulence homogène isotrope décroissante et un calcul polydisperse d'un jet turbulent recirculant chargé en particules. L'équilibrage de charges de particules est mis en évidence comme une solution prometteuse pour les simulations diphasiques Lagrangiennes afin d'améliorer les performances des calculs lorsque le deséquilibrage est trop important.

Mots clés : Formalisme Euler-Lagrange, maillages non-structurés, simulation aux grandes échelles, écoulements diphasiques, équilibrage de charges, calcul parallèle.

Development and validation of the Euler-Lagrange formulation on a parallel and unstructured solver for large-eddy simulation

Particle-laden flows occur in industrial applications ranging from droplets in gas turbines to fluidized bed in chemical industry. Prediction of the dispersed phase properties such as concentration and dynamics are crucial for the design of more efficient devices that meet the new pollutant regulations of the European community. The objective of this thesis is to develop an Euler-Lagrange formulation on a parallel and unstructured solver for large-eddy simulation. This work is motivated by the rapid increase in computing power which opens a new way for simulations that were prohibitive one decade ago. Special attention is taken to keep data structure simplicity and code portability. Developments are validated in two configurations: an academic test of a decaying homogeneous isotropic turbulence and a polydisperse two-phase flow of a confined bluff body. The use of load-balancing capabilities is highlighted as a promising solution in Lagrangian two-phase flow simulations to improve performance when strong imbalance of the dispersed phase is present.

Keywords: Euler-Lagrange formulation, unstructured grids, large-eddy simulation, two-phase flows, particle load-balancing, parallel computations.

Centre **E**uropéen de **R**echerche et de **F**ormation **A**vancée en **C**alcul **S**cientifique (CERFACS)
42, avenue Gaspard Coriolis
31057 Toulouse Cedex 1 - France -