



**HAL**  
open science

# Massively parallel numerical simulation using up to 36,000 CPU cores of an industrial-scale polydispersed reactive pressurized fluidized bed with a mesh of one billion cells

Hervé Neau, Maxime Pigou, Pascal Fede, Renaud Ansart, Cyril Baudry, Nicolas Mérigoux, Jérôme Laviéville, Yvan Fournier, Nicolas Renon, Olivier Simonin

## ► To cite this version:

Hervé Neau, Maxime Pigou, Pascal Fede, Renaud Ansart, Cyril Baudry, et al.. Massively parallel numerical simulation using up to 36,000 CPU cores of an industrial-scale polydispersed reactive pressurized fluidized bed with a mesh of one billion cells. *Powder Technology*, 2020, 366, pp.906-924. 10.1016/j.powtec.2020.03.010 . hal-02735376

**HAL Id: hal-02735376**

**<https://hal.science/hal-02735376>**

Submitted on 2 Jun 2020

**HAL** is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.



## Open Archive Toulouse Archive Ouverte





OATAO is an open access repository that collects the work of Toulouse researchers and makes it freely available over the web where possible

This is an author's version published in: <http://oatao.univ-toulouse.fr/25782>

### Official URL:

<https://doi.org/10.1016/j.powtec.2020.03.010>

### To cite this version:

Neau, Hervé  and Pigou, Maxime  and Fede, Pascal  and Ansart, Renaud and Baudry, Cyril and Méricoux, Nicolas and Laviéville, Jérôme and Fournier, Yvan and Renon, Nicolas and Simonin, Olivier  *Massively parallel numerical simulation using up to 36,000 CPU cores of an industrial-scale polydispersed reactive pressurized fluidized bed with a mesh of one billion cells.* (2020) Powder Technology, 366. 906-924. ISSN 00325910

Any correspondence concerning this service should be sent to the repository administrator: [tech-oatao@listes-diff.inp-toulouse.fr](mailto:tech-oatao@listes-diff.inp-toulouse.fr)

# Massively parallel numerical simulation using up to 36,000 CPU cores of an industrial-scale polydispersed reactive pressurized fluidized bed with a mesh of one billion cells

Hervé Neau<sup>a,\*</sup>, Maxime Pigou<sup>a</sup>, Pascal Fede<sup>a</sup>, Renaud Ansart<sup>b</sup>, Cyril Baudry<sup>d</sup>, Nicolas Méricoux<sup>e</sup>, Jérôme Laviéville<sup>e</sup>, Yvan Fournier<sup>e</sup>, Nicolas Renon<sup>c</sup>, Olivier Simonin<sup>a</sup>

<sup>a</sup> Institut de Mécanique des Fluides de Toulouse (IMFT), Université de Toulouse, CNRS, Toulouse, France

<sup>b</sup> Laboratoire de Génie Chimique, Université de Toulouse, CNRS, INPT, UPS, Toulouse, France

<sup>c</sup> UMS CALMIP 3667 Université de Toulouse, CNRS, INPT, INSA, ISAE, UPS, Toulouse, France

<sup>d</sup> Délégation technologies et systèmes d'information, EDF R&D, Palaiseau, France

<sup>e</sup> Fluid Mechanics, Energy and Environment Dpt, EDF R&D, Chatou, France

## ABSTRACT

For the last 30 years, experimental and modeling studies have been carried out on fluidized bed reactors from laboratory up to industrial scales. The application of developed models for predictive simulations has however been strongly limited by the available computational power and the capability of computational fluid dynamics software to handle large enough simulations. In recent years, both aspects have made significant advances and we thus now demonstrate the feasibility of a massively parallel simulation, on whole supercomputers using NEPTUNE\_CFD, of an industrial scale polydispersed fluidized bed reactor. This simulation of an olefin polymerization reactor makes use of an unsteady Eulerian multi fluid approach and relies on a billion cells meshing. This is a worldwide premiere as the obtained accuracy is yet unmatched for such a large scale system. The interest of this work is two fold. In terms of High Performance Computation (HPC), all steps of setting up the simulation, running it with NEPTUNE\_CFD, and post processing results induce multiple challenges due to the scale of the simulation. The simulation ran using 1260 up to 36,000 cores on supercomputers, used 15 millions of CPU hours and generated 200 TB of raw data for a simulated physical time of 25 s. This article details the methodology applied to handle this simulation, and also focuses on computation performances in terms of profiling, code efficiency and partitioning method suitability. Though being by itself interesting, the HPC challenge is not the only goal of this work as performing this highly resolved simulation will benefit chemical engineering and CFD communities. Indeed, this computation enables the possibility to account, in a realistic way, for complex flows in an industrial scale reactor. The predicted behavior is described, and results are post processed to develop sub grid models. These will allow for lower cost simulations with coarser meshes while still encompassing local phenomenon.

### Keywords:

Polydispersed  
Fluidized bed  
Olefin polymerization reactor  
Industrial scale  
Heat transfer  
Computation fluid dynamics  
High performance computing  
HPC  
Massively parallel  
MPI  
Sub-grid model

## 1. Introduction and motivations

Dense particle laden reactive flows are encountered in a wide range of industrial applications, both historical such as coal combustion (CO<sub>2</sub> capture / chemical looping), catalytic polymerization, uranium fluorination or medicine drugs production, and more recently for green applications (solar energy, biomass combustion, ...). Industrially, gas phase polymerization is one of the main approaches to produce polyethylene by catalytic polymerization [1–4]. A typical scheme of the gas solid ethylene polymerization process is shown by Fig. 1. This kind of reactor

operates in a dense phase fluidization regime. The gas, formed by unreacted monomer of ethylene, circulates in the cycle loop and serves to fluidize the granular bed. The gas also carries out the reaction heat, and maintains a stable reactor temperature. Polymer grows on small, high activity, catalyst particles of Geldart groups A/C upon reaching large particles size of Geldart groups B/D [5]. Full size polymer particles are finally withdrawn at the bottom of the reactor in dry granular form.

A goal for industrial communities is to access accurate details about the functioning of fluidized beds to improve reactors design, their efficiency and their scale up. Experimental approaches could be considered but are usually limited to reactors of laboratory and pilot scales. Furthermore, experimental measurements are difficult to obtain, due to their invasive effect coupled with extreme operating conditions in

\* Corresponding author.

E-mail address: neau@imft.fr (H. Neau).

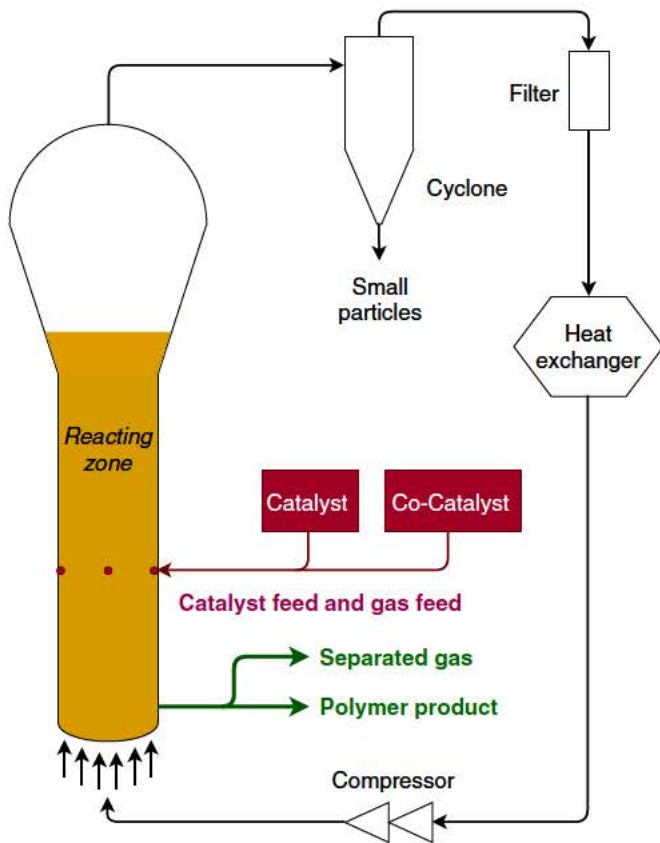


Fig. 1. Industrial polyethylene production process scheme.

terms of temperatures, pressure and high volume fraction of particles. The other possible approach is that of simulating the reactor to fully describe its behavior. This numerical approach also comes with specific challenges in particular in terms of (i) modeling physical phenomena occurring in these reactors, and (ii) performing accurate simulations based on these models.

The development of fluidized beds numerical modeling with particular focus on their hydrodynamics has been a very active topic of research over the last three decades [6–13]. Due to the high number of particles involved in fluidized bed reactors at industrial scale, only Euler-Euler modeling approaches can be reasonably considered to reach tractable simulation. In particular, simulations of fluidized bed reactors have been performed using the software NEPTUNE\_CFD [14–22].

From a physical point of view, the key challenge is to account for all complex and coupled phenomena occurring in reactive multi-scale flows: turbulence, particle fluctuating kinetic energy, fluid-particle interactions, interfacial transfers of mass, momentum and energy as well as chemical reactions. Accounting for all these phenomena in an Euler-Euler framework induces very expensive numerical simulations (high CPU time) at academic scales and even more at industrial scales, in particular due to the 3D unsteadiness. Furthermore, predicting the macroscopic flow properties such as bed expansion, solid flow rate or wall pressure profile in these systems requires a detailed description of small-scale solid clusters [23,24]. To accurately describe and predict the formation of smallest clusters, a very fine mesh having million/billion cells is required.

Recently, computational power of supercomputers has significantly increased [25] along with massively parallel capabilities of simulation software. Therefore, it is nowadays possible to perform realistic 3D simulations of industrial geometries using an unsteady Eulerian multi-fluid approach for turbulent polydisperse reactive particle mixtures [26].

Performing such a simulation requires tackling numerical challenges such as (i) developing suited numerical schemes, (ii) generating highly refined meshes and their partitioning, (iii) ensuring High Performance Computations (HPC) and (iv) handling large data sets and their post-processing.

The purpose of the current work is the numerical simulation of an industrial-scale reactive gas-solid pressurized fluidized bed for catalytic olefin polymerization whose dimensions are 30 m height and 5 m in diameter. This reactor is alike industrial reactors that are usually used for polymerization of olefin and production of polyethylene. The key novelties of this paper are a highly refined meshing and the use of the whole new CALMIP and EDF supercomputers to perform the simulation of this configuration with a mesh size of one billion (1,002,355,456) cells. This massively parallel simulation running over 36,000 CPU cores is a worldwide premiere.

Results from this simulation are analyzed under two different angles. The first one is that of HPC. A special attention is given to massively parallel computation performances of NEPTUNE\_CFD [27] in terms of profiling and code efficiency, obtained speed up, MPI communications and mesh partitioning. The second angle is that of physical results. Thanks to its yet unmatched accuracy, this simulation allows for a fine understanding of complex flows encountered at industrial scale. These results can be considered as reference results which are, at the best of our knowledge, the closest from mesh independency for a large industrial-scale geometry.

Previous studies pointed out the difficulty to use a sufficiently refined mesh to account for the effect of small structures for all scales from laboratory to industrial size installations [28–36] and introduced the need for sub-grid modeling [37–40]. Therefore, our fine simulation may serve as the basis to develop such sub-grid models which will in turn enable low-cost simulations while still accounting for small-scale phenomena. This is the first time that highly resolved simulations can be used to develop such models at an industrial scale.

We first remind in Section 2 the formulation of an  $n$ -fluid Euler model for the description of an industrial multiphase fluidized bed. Section 3 focuses on numerical methods applied to solve the model equations and on parallelization techniques which ensure the good scalability and high performances of NEPTUNE\_CFD. The pre-processing (mesh generation and partitioning) and the simulation case configuration are described in Section 4. Finally, simulation results, both in terms of HPC performances and of industrial fluidized bed behavior predictions, are presented in Section 5 along with their post-processing for sub-grid modeling.

## 2. Mathematical modeling

NEPTUNE\_CFD is not limited in the number of tracked phases. This software simulates a generic  $n$ -fluid Euler model and simultaneously resolves mass, momentum and enthalpy conservation equations. These generic equations are briefly recalled hereafter along with models for momentum and enthalpy transfers that are used for the simulation of the industrial fluidized bed reactor described in Section 4.

The Eulerian  $n$ -fluid approach used is a hybrid approach [41,42]. Transport equations are derived by phase ensemble averaging for the continuous phase and by using Kinetic Theory of Granular Flows (KTGF) supplemented by fluid and turbulent effects for the dispersed phase thanks to joint fluid-particle Probability Density Function (PDF) approach [42]. Here, we actually present a 3-fluid Euler model application, but following approach can be extended directly to  $n$ -fluid applications. As further explained in Section 4, our simulations for the considered fluidized bed track a gas phase (subscript  $g$ ) and two particulate phases for two particle sizes, referred to as fine (subscript  $f$ ) and large particles (subscript  $l$ ). In terms of notations, the subscript  $p$  refers to any particle ( $p \in \{f, l\}$ ) and  $k$  refers to any phase ( $k \in \{g, f, l\}$ ). In the particle transport equations,  $\alpha_p \rho_p$  represents  $n_p m_p$ , the local mass of  $p$  particles per unit volume with  $n_p$  the local number density of

$p$  particle and  $m_p$  the mass of a single  $p$  particle:  $\alpha_p = n_p m_p / \rho_p$  is an approximation of the local volume fraction of particle  $p$  with  $\rho_p$  its density. Hence, gas and particle volume fractions,  $\alpha_g$ ,  $\alpha_f$  and  $\alpha_l$  have to satisfy:

$$\alpha_g + \alpha_f + \alpha_l = 1 \quad (1)$$

Fig. 2 illustrates momentum and enthalpy transfers between the three phases.

### 2.1. Mass transport equation

The mass transport equation reads:

$$\frac{\partial}{\partial t} \alpha_k \rho_k + \frac{\partial}{\partial x_j} \alpha_k \rho_k U_{k,j} = 0 \quad (2)$$

with  $\rho_k$  density of the phase  $k$  and  $U_{k,i}$  the  $i$  component of its velocity. The null right hand side corresponds to no mass transfer between phases.

### 2.2. Momentum transport equation

The evolution of phase momentum is described by the conservation equation:

$$\alpha_k \rho_k \left[ \frac{\partial U_{k,i}}{\partial t} + U_{k,j} \frac{\partial U_{k,i}}{\partial x_j} \right] - \alpha_k \frac{\partial P_g}{\partial x_i} + \alpha_k \rho_k g_i + \sum_{k' \neq k} I_{k' \rightarrow k, i} - \frac{\partial \Sigma_{k,ij}}{\partial x_j} \quad (3)$$

with:

- $P_g$  the mean gas pressure,  $g_i$  the gravity  $i$  component.
- $I_{k' \rightarrow k, i} = -I_{k \rightarrow k', i}$  is the  $i$  component of the momentum transfer term from phase  $k'$  to  $k$ :
  - $I_{g \rightarrow f}$  and  $I_{g \rightarrow l}$  are the gas particle momentum transfer rates modeled using the drag law of Wen and Yu [43] limited by the Ergun [44] equation for dense flows [6,45] after subtracting mean gas pressure contributions (Archimede force).
  - $I_{l \rightarrow f}$  is the momentum transfer rate between particle species [46–48].
- $\Sigma_{k,ij}$  is the effective stress tensor of phase  $k$  made of three parts: (i) the kinetic part, dominant in dilute flows, (ii) the collisional part dominant in dense flows for  $\alpha_p \in [0.15; 0.5]$ , and (iii) the frictional part in quasi static granular flows ( $\alpha_p \geq 0.5$ ). The modeling of the two first terms is derived in the frame of the Kinetic Theory of Granular Flow [49], accounting for the effect of the interstitial fluid [42,50]. So, both kinetic and collisional parts of the effective stress will be functions of

the particle agitation and the fluid particle correlation. The particle kinetic stress tensor is modeled using the Boussinesq approximation [6] extended to polydispersed mixtures [7,19,51]. The quasi static granular flow zones are taken into account in the particle stress tensor by the additional frictional stress tensor [21,52] activated with a threshold value of  $\alpha_p = \alpha_f + \alpha_l \geq 0.5$ .

The fluid turbulence modeling is achieved by the two equations  $k - \varepsilon$  model extended to particle laden flows accounting for additional source terms due to inter phase interactions [53] (see two way coupling by Fig. 2). For the dispersed phase, a coupled transport equation system is solved on particle fluctuating kinetic energy and gas particle fluctuating covariance ( $q_p^2 - q_{gp}$ ).

### 2.3. Enthalpy transport equation

The enthalpy of each phase satisfies the transport equation:

$$\frac{\partial}{\partial t} (\alpha_k \rho_k H_k) + \frac{\partial}{\partial x_j} (\alpha_k \rho_k H_k U_{k,j}) = \frac{\partial}{\partial x_j} \left( \alpha_k \rho_k K_k \frac{\partial H_k}{\partial x_j} \right) + \sum_{k' \neq k} \Pi_{k' \rightarrow k} + S_k^{\text{reac}} \quad (4)$$

where  $H_k$  is the specific enthalpy of phase  $k$ .

Assuming that the heat exchanged by contact during interparticle collisions and by radiation is negligible [14,22], modeled heat transfer to the particle is only accounting for (i) heat exchange by the gas phase, (ii) transport by random velocity fluctuations (kinetic diffusion) and (iii) exothermic source term summarized by:

- For the particle phase, the diffusivity coefficient is obtained as  $K_p = K_p^c$ , where  $K_p^c$  are the contributions due to the transport of the enthalpy by particle fluctuating motion expressed by Laviéville et al. [54].
- For the gas phase, the diffusivity coefficient is obtained as  $K_g = K_g^c + K_g^t$  where  $K_g^c$  and  $K_g^t$  are the contributions to the transport of enthalpy due to gas turbulent dispersion and to the laminar diffusivity.
- The convection/diffusion heat transfer  $\Pi_{g \rightarrow p}$  between the gaseous phase and the particles occurring with a characteristic time scale  $\tau_{gp}^T$  expressed as a function of Nusselt number. The dimensionless number may be expressed by Ranz and Marshall [55] correlation.
- Thermal source terms allow to account for the exothermic reaction. Here,  $S_k = \alpha_k \rho_k Q_k$  with  $Q_k$  a constant per phase specific power (W/kg $_k$ ) dissipated by the reaction.

All presented models are available in the CFD software NEPTUNE\_CFD [27]. Section 3 details numerical and parallelization methods used to perform the simulation.

## 3. Numerical methods

Simulations are performed using the software NEPTUNE\_CFD based on the open source software Code Saturne. This software is a multi phase flow solver developed in the framework of the NEPTUNE project, financially supported by CEA, EDF, IRSN and Framatome. The main numerical characteristics of this software are (i) unstructured meshes with all types of cell, (ii) non matching meshes and/or rotating meshes, (iii) "cell center" type finite volume method, (iv) calculation of localized gradients with reconstruction methods and (v) distributed memory parallelism by domain splitting (mesh partitioning and MPI parallelization). The algorithm core is based on an elliptic fractional step method using iterative linear solvers or direct matrix inversion [56]. The algorithm can account for density variation as a function of pressure and enthalpy during each time step. A parallel multigrid solver resolves the pressure. NEPTUNE\_CFD scalability has already been proven on this kind of configuration up to 2560 cores [26]. Previous runs have been performed using up to 4000 cores with linear speed up and perfect efficiency as long as each CPU core handles at least 25,000 mesh cells.

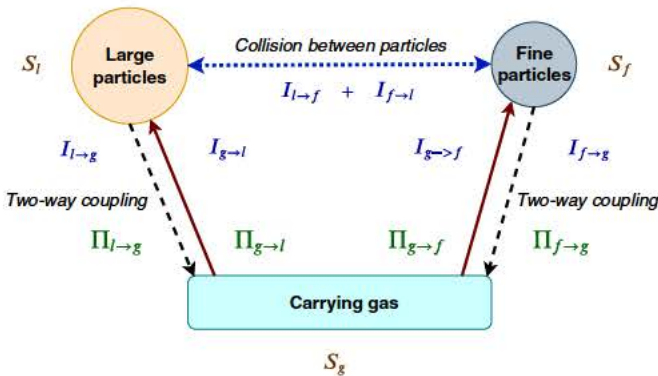


Fig. 2. Illustration of inter-phase modeling coupling. Green: enthalpy transfer terms. Brown: Enthalpy source terms. Blue: Momentum transfer terms. Continuous arrows: one-way coupling. Dashed arrows: two-way coupling. Dotted arrows: four-way coupling. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

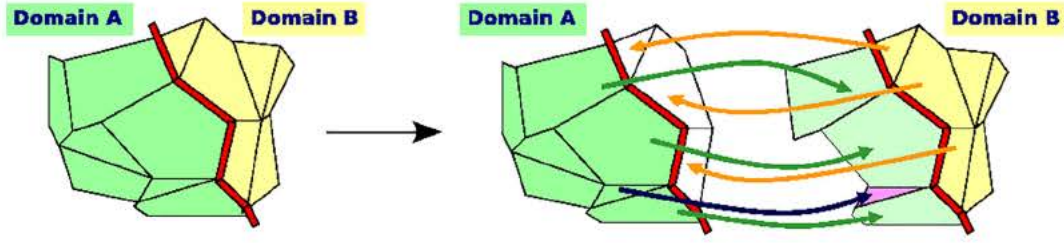


Fig. 3. Ghost cell value exchange principle.

21	22	23	24	25
16	17	18	19	20
11	12	13	14	15
6	7	8	9	10
1	2	3	4	5

Fig. 4. Example of simple 2D mesh and associated global cell ids; partitionned to balance computations and limit inter-domain data exchanges.

### 3.1. Original fractional step method for multi phase systems

The algorithm core is based on an elliptic fractional time step method using iterative solver or direct inversion of matrix of size  $n_{\text{phase}} \times n_{\text{phase}}$  [56,57] to solve all the transport equations of the  $n$  fluid Euler modeling approach. The main originality of the numerical method is the so called “alpha pressure energy cycle” that ensures an accurate separate conservation of mass and energy of each phase while allowing a strong implicit coupling of mass, momentum and energy between all the computed phases (carrying gas, fine and large particles in the case of this paper). A first step predicts separately each phase velocity components, neglecting the volumetric fraction and pressure time variations, and is completed by an implicit inter phase coupling sub step corresponding to a new estimation of the local inter phase momentum transfer terms using  $n_{\text{phase}} \times n_{\text{phase}}$  matrix inversions. The second step solves a coupled system of mass and enthalpy written in order to enforce a very accurate conservation of these quantities and allowing a correction of the phase velocity components accounting for the pressure and volume fractions time variations. This step is called the “alpha pressure energy cycle” because the computed variables are, on one hand, the carrying gas pressure and, on the other hand, the volumetric fraction and enthalpy for each phase. The alpha pressure energy system is resolved using sub cycles which sequentially solve the separate phase transport equations for enthalpy and volumetric fraction and, finally, an elliptic equation for pressure. Convergence criteria of the “alpha pressure energy cycle” is based on the mixture volume conservation accuracy:  $|\sum_k \alpha_k - 1| < \epsilon$  with  $\epsilon = 10^{-8}$  the convergence tolerance.

### 3.2. Computation parallelization

*Code Saturne* architecture and libraries, on which NEPTUNE\_CFD is based, rely on the MPI paradigm, using a classical partitioning with a “ghost cell” approach, where the computational domain is distributed over compute nodes and cores. Values from adjacent domains are “cached” in ghost cells (see Fig. 3) to minimize communications with distant ranks. To restrict communications to a bare minimum, ghost cell exchanges can be limited to cells adjacent through faces, needed for all types of operations (linear solvers, gradient computations, balances) and those adjacent only through vertices, needed only for gradient computations.

Except for some stages of the simulation, the whole execution is distributed, with each process handling only a part of the data. This requires some precautions, but avoids having a single rank (CPU core) which needs to handle the full domain. This is the key to run large models. The adaptation of *Code Saturne* to large runs has been an ongoing effort for several years [58], and a few key elements are provided here.

To handle parallelism at all stages, we use two different data distributions [59]:

- A main “partitioned”, or “compute” distribution, where data is distributed so as to minimize communication volume (see top of Fig. 5);
- A temporary “block”, or “flat” distribution for I/O operations, where data is distributed based on global (unpartitioned) element ids, in successive blocs, so that determination of where a given element’s data resides can be inferred by a simple rule, requiring no communication (see top of Fig. 5).

Whereas the main partitioning distributes data over all ranks, the block distribution may reside on a subset of available ranks, thus avoiding many smaller blocks, which may be useful in some cases such as I/O. Switching from one distribution to another may be done using “all to all” MPI communications. Figs. 4 and 5 illustrate this principle for cell attached data. The same principle is also applied for face attached data with slight adaptation as faces can be shared by multiple compute ranks [59].

Several stages in the computation setup are based on these data distributions: (i) preprocessing (mesh reading, joining and partitioning); (ii) checkpoint write and read operations and (iii) post processing file exports for visualization [60].

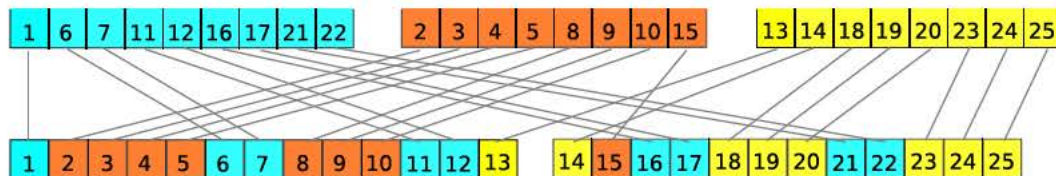


Fig. 5. Data movement between “compute” distribution over 3 ranks, and “flat” distribution over 2 ranks. Colors and indexes refer to the domain partitioning from Fig. 4.

All I/O of large datasets (mesh input, checkpoint/restart, and visualization file output) use MPI IO, based on the block distribution. This allows reading from and writing to single files, which are independent from the partitioning, while avoiding the bottleneck of a single rank requiring global data.

Several important preprocessing algorithms are available in the main solver and are fully parallel, including mesh joining [61], and more recently extrusion, boundary layer insertion and mesh refinement. The preprocessor converting files from external to internal formats is not parallelized yet, which is not too much of an issue, as most available meshing tools are not parallel either. The common strategy is to mesh by pieces, and read and assemble (join) sub meshes in parallel. Once preprocessed, a mesh may be saved using an internal format.

Checkpoint file generation and calculation restart also use a block distribution and parallel IO, as well as postprocessing output to the EnSight gold format [60]. Note that for postprocessing, we can choose to extract only selected subsets of the mesh, to reduce the output volume, and avoid issues with some mesh formats (for example, the EnSight gold mesh format is based on 32 bit numberings, so a single part may not contain slightly more than 2 billion entities). Any combination of sub meshes and outputs to different files may be used, based on a postprocessing mesh (extracts)/writer (file format and output frequency) paradigm.

To allow for more complex postprocessing and further reduce the volume of data produced, *in situ* postprocessing using ParaView's Catalyst subset is also possible [62,63].

### 3.3. Multi grid solver

NEPTUNE\_CFD uses a segregated solver, with solution by increments [64]. For symmetric systems such as the pressure correction, an aggregation based multi grid solver is used, with tailored smoothing based on local mesh characteristics. By default, aggregation is limited to local processes, so the setup stage is quite fast. As the associated matrix coefficients may vary over time, the multi grid hierarchy is rebuilt at each time step. The multi grid algorithm may be used both as a solver or as a preconditioner for a conjugate gradient algorithm, with the latter option usually leading to better performances.

As an option, the PETSc library may also be used. We have observed that for a sampling of single phase, in *Code Saturne* cases, the built in multigrid preconditioner exhibits a less regular convergence than HYPRE. It requires more iterations to converge, but for a lower cost per iteration, leading to slightly better performances. Thus, this algorithm is on par with state of the art reference libraries. In the NEPTUNE\_CFD context, even though the associated matrix is not perfectly symmetric, this multigrid algorithm may also be used for the pressure correction, with slightly different default settings not relying on PETSc.

## 4. Billion mesh simulation setup

The simulated system is a catalytic polydispersed gas solid pressurized fluidized bed reactor for olefin production at industrial scale. The flow inside is a complex multi scale flow with meso and macro structures. The ratio between the reactor diameter and the particle size is very high, up to  $d_R/d_p \approx 6.10^4$  for fine particles, leading to strong interactions between all scales of the flow. This system is unsteady, multi phase (gas solid) and reactive: the polymerization reaction is exothermic.

This section first describes the actual simulated geometry, phases properties and the system operating and boundary conditions. In a second time, it details the procedure applied to generate the mesh and its partitioning which by itself constitutes a HPC challenge.

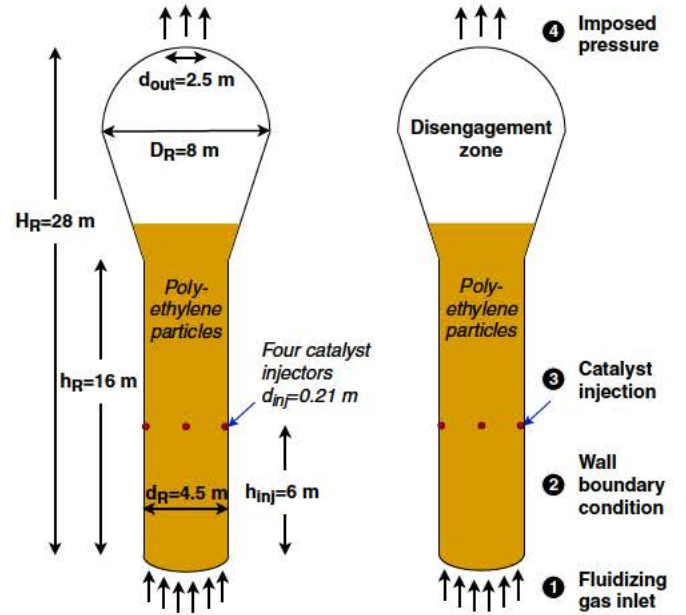


Fig. 6. Gas solid pressurized polymerization reactor scheme (left) and boundary conditions (right).

### 4.1. Industrial scale geometry

The scheme of the reactor is shown by Fig. 6. The reactor is a cylinder with a distributor plate at the bottom and a disengagement zone above.

The characteristic dimensions are: height  $H_R = 28$  m, diameter  $d_R = 4.5$  m, height of cylindrical base  $h_R = 16$  m, and bulb max diameter  $D_R = 8$  m. The reactor is equipped (i) with a gas fluidization inlet at the bottom; (ii) with a gas and particles outlet/evacuation at the top whose diameter is  $d_{out} = 2.5$  m and (iii) with four lateral catalyst and gas inlet located at a height  $h_{inj} = 6$  m. The diameter of each injector is  $d_{inj} = 0.21$  m which corresponds to a circular section  $S_{inj} \approx 0.03$  m<sup>2</sup>.

An actual industrial system would also have outlets at the bottom to extract the final product, by withdrawing large particles. Another aspect that is not taken here into account but would appear in actual reactors is the retrieval of fine particles that escape by elutriation at the top of the fluidized bed using cyclonic separators and their re injection at the bottom of the bed. These aspects are not considered here as the simulation time (about 25 s) is not long enough compared to both the lifetime of polyethylene particles in the reactor and the mean residence time of fine particles (see 5.4). Moreover, outlet and re injection pipes do not have a significant feedback impact on the main hydrodynamics. Another simplified aspect is that the fluidization grid is considered ideal, which implies an homogeneous velocity at the fluidization gas inlet. This homogeneous velocity hypothesis has actually been assessed and validated beforehand in simulations where the distribution chamber below the distribution plate, and the head losses through the plate, were both accounted for (unpublished results).

Note that in this CFD modeling approach, we only consider the hydrodynamic and thermal behavior within the fluidized bed polymerization reactor. Other unit operations of the process are not included in the simulation.

These geometry simplifications were kept from previous numerical studies on the same industrial fluidized bed reactor [65]. Therefore, we do not describe lateral injectors pipes.

### 4.2. Gas and particulate phase properties

The Particle Size Distribution (PSD) is very large ranging from a few microns for catalyst particles up to several millimeters for fully

**Table 1**  
Powder and gas physical properties.

	Gas	Large particles	Fine particles
Density at 24 bars ( $\text{kg} \cdot \text{m}^{-3}$ )	22	850	850
Viscosity $\times 10^5$ at 24 bars ( $\text{Pa} \cdot \text{s}$ )	1.54	–	–
Mean diameter ( $\mu\text{m}$ )	–	1600	80
Restitution coefficient	–	0.9	0.9
Specific heat ( $\text{J} \cdot \text{kg}^{-1} \cdot \text{K}^{-1}$ )	1728	2000	2000
Thermal conductivity ( $\text{W} \cdot \text{m}^{-1} \cdot \text{K}^{-1}$ )	0.04	–	–

grown polymer particles. To account for this PSD in a numerically tractable manner, we choose to describe particles of two characteristic sizes: (i) large particles (1,600  $\mu\text{m}$ ) which are representative of polyethylene particles in the reactor and will yield a realistic fluidized bed hydrodynamics and (ii) fine particles (80  $\mu\text{m}$ ) which are characteristic of injected catalyst particles. The growing aspect of particle sizes is not accounted for, as the simulated time is short compared to particles residence time in the reactor. Therefore, the three present phases are the gaseous phase and the two particulate phases.

One hundred tons of large particles are present from the start of the simulation whilst fine particles are injected through the four catalyst injectors. Both categories of particles are subjected to an exothermic reaction (see Section 2).

The powder and gas material properties are gathered in Table 1. Note that the simulation does not account for the variation of these physical properties according to temperature and pressure. For the gas phase, the reference pressure is 24 bars.

#### 4.3. Operating, boundary and initial conditions

We start with an homogeneous distribution of solid volume fraction in the bed part:

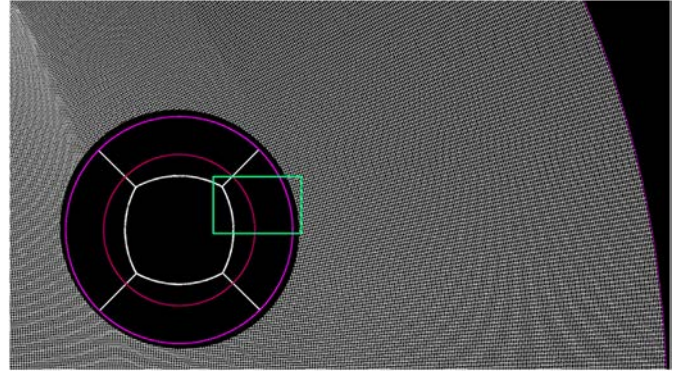
$$\alpha_l \begin{cases} 0.402 & \text{if } z < 18\text{m} \\ 10^{-12} & \text{otherwise} \end{cases}$$

$$\alpha_f = 0$$

Initial gas and particle temperatures are set to 100 °C. Operating conditions are defined by taking into account the exothermic polymerization reaction. Thermal source terms are used to implement this exothermic reaction. Fine particles are 30 times more reactive than large ones per mass unit. Their reaction dissipated powers are  $Q_f = 6.0 \text{ kW/kg}_f$  and  $Q_l = 0.2 \text{ kW/kg}_l$ . Enthalpy source terms for each phase (see Eq. (4) and Fig. 2) are then given by:

**Table 2**  
Boundary conditions.

	Fluidization inlet	Catalyst injectors	Outlet	Wall
Gas	$Q_g = 756 \text{ t/h}$ $V_g = 4U_{mf}$ $k = 1.0 \times 10^4 \text{ m}^2 \cdot \text{s}^{-2}$ $\varepsilon = 1.0 \times 10^3 \text{ m}^2 \cdot \text{s}^{-3}$ $T_g = 50 \text{ }^\circ\text{C}$	$Q_{inj,g} = 19.3 \text{ t/h/injector}$ $\alpha_{inj,g} = 0.99976$ $k = 1.0 \times 10^4 \text{ m}^2 \cdot \text{s}^{-2}$ $\varepsilon = 1.0 \times 10^3 \text{ m}^2 \cdot \text{s}^{-3}$ $T_{inj,g} = 50 \text{ }^\circ\text{C}$	Free flow with imposed gauge pressure.	Friction model.
Fine part.	No-slip.	$Q_{inj,f} = 0.18 \text{ t/h/injector}$ $\alpha_{inj,f} = 2.4 \times 10^5$ $q_p^2 = 5.0 \times 10^5 \text{ m}^2 \cdot \text{s}^{-2}$ $q_{fp} = 1.0 \times 10^4 \text{ m}^2 \cdot \text{s}^{-2}$ $T_{inj,f} = 50 \text{ }^\circ\text{C}$	Free outlet for enthalpy.	Zero flux of enthalpy.
Large part.	No-slip.	No-slip.	Free flow with imposed gauge pressure	No-slip.
			Free outlet for enthalpy.	Zero flux of enthalpy.



**Fig. 7.** Mono-layer O-Grid mesh generated with Simail [67] - 476,928 hexahedral cells - 850 cells along a diameter.

$$S_g = 0.0 \text{ kW/kg}_g \quad (5)$$

$$S_f = \alpha_f \rho_p Q_f \quad (6)$$

$$S_l = \alpha_l \rho_p Q_l \quad (7)$$

For the simulation, the total mass of large particles will be  $M_l = 100,000 \text{ kg}$ , and the total mass of fine particles will be negligible for the time period simulated: about 5.2  $\text{kg}_f$  after 25 s. Therefore, the total reaction dissipated power will be about  $Q_l M_l = 20 \text{ MW}$  which is of the same order of magnitude than in an actual reactor.

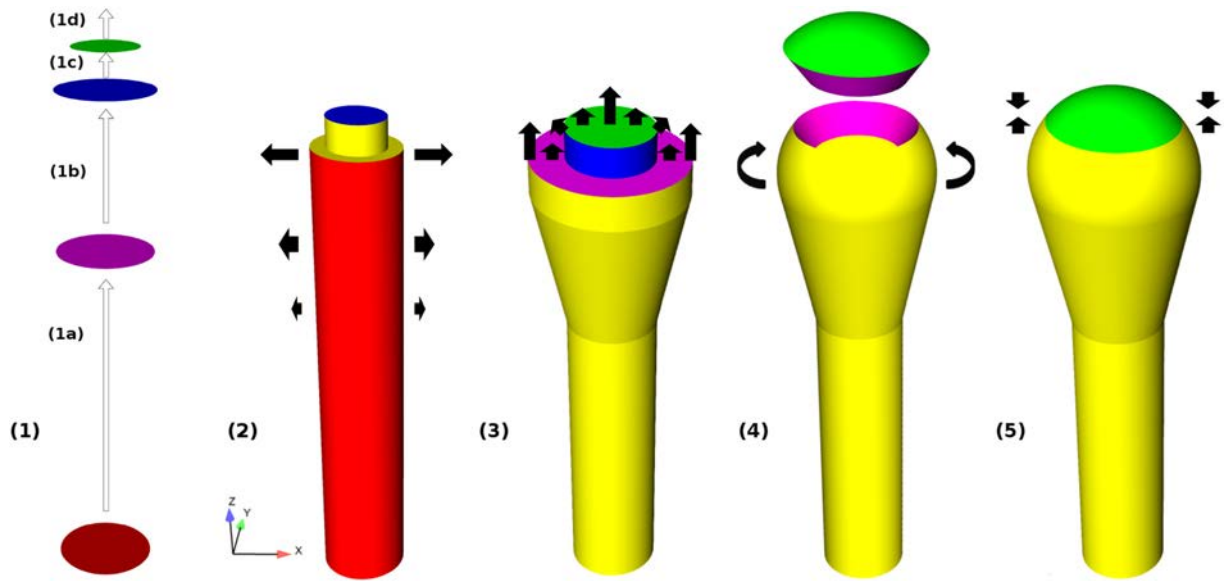
The geometry has four kinds of boundary conditions summarized by Fig. 6 (right). The fluidization plate, through which air was injected at a constant mass flow rate, is modeled by a gas inlet with uniform superficial velocity equal to  $V_g = 0.6 \text{ m/s}$  and set temperature. This boundary was seen as a wall by solid phases which induces a null flux for solid phase enthalpy. The detail of boundary conditions applied for each phase is given in Table 2.

The wall boundary condition was a no slip condition for particles and a friction condition for the gas. The no slip wall condition for particles correspond to a zero flux for random kinetic energy [20]. A Neumann type boundary condition is imposed for gas and solid phase enthalpies on walls.

The top of the reactor is a free outlet with set gauge pressure. Should a depression appear just below this surface, the outlet boundary condition acts as a gas inlet with a set temperature of 100 °C.

To take into account the polymerization reaction, we consider an exothermic reaction through heat source terms on particles. Enthalpy transfers between gas and large particles and between gas and fine particles are taken into account.





**Fig. 8.** 3D meshes transformations with *Code\_Saturne*. (1) mono-layer meshes extrusion; (2) Radial dilation; (3) and (4): bulb formation through vertices coordinates shifting; (5): sub-mesh merging.

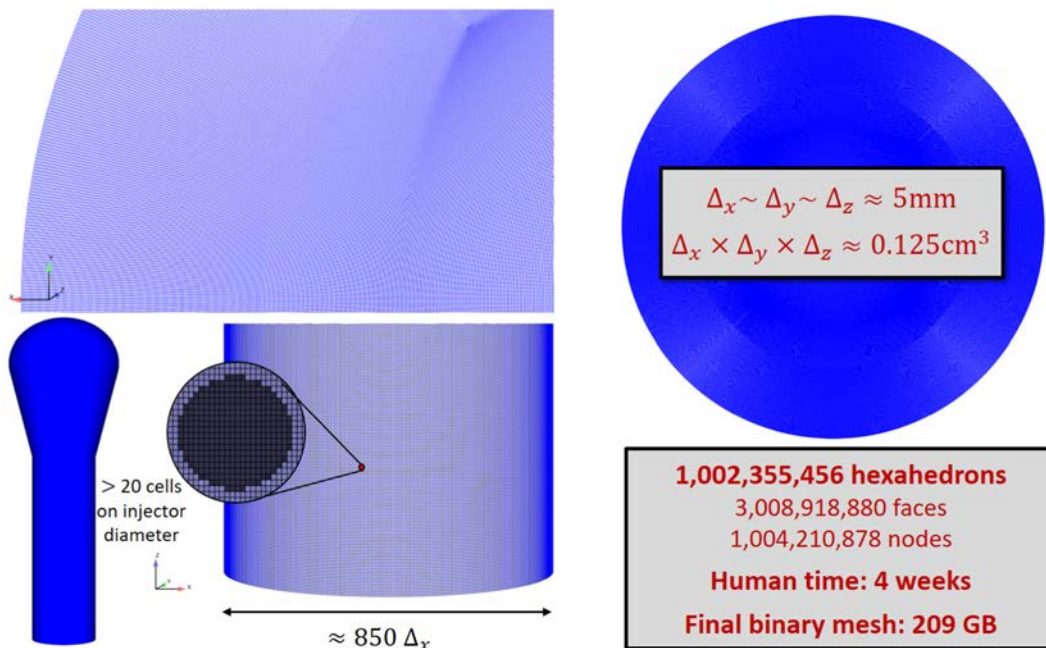
#### 4.4. Mesh construction and mesh partitioning

In this study, the mesh refinement is a key point. We first detail how to generate a billion cells mesh for the industrial scale geometry, which is not feasible using a classical one step mesh generator. This requires more memory than available on most workstations or supercomputers compute nodes. Moreover, only few meshing software are parallelized. The use of supercomputers large memory nodes could be considered, but this approach would be prohibitively long due to the limit in computational power of these nodes, and this approach would not scale to even bigger meshes. The use of a distributed approach is thus required. We then designed and applied a non conventional methodology using a multi step, multi software approach detailed hereafter.

To generate the large mesh used in following simulations, we relied on multiple useful features of *Code\_Saturne* [66] in terms of mesh

modification and extrusion, namely (i) its surface mesh extrude capability; (ii) its mesh merging feature (either matching or non matching) and (iii) its compatibility with user defined mesh transformations. The strategy has been the following:

1. Generate 4 mono layer O Grid type meshes representing horizontal slices of the geometry: This step was performed using the software Simail [67]. Three of these meshes consist in 476,928 hexahedral cells each (see Fig. 7), and the top one contains 253,184 hexahedron. The three first meshes have 850 cells along a diameter.
2. Using the extrude functionality of *Code\_Saturne* [66], form 4 cylinders from the four mono layer meshes (see Fig. 8: (1a): 1451 layers up to  $z = 16m$ , (1b): 471 layers up to  $z = 23m$ , (1c): 99 layers up to  $z = 25m$  and (1d): 152 layers up to  $z = 28m$ ).



**Fig. 9.** Illustration of the final billion cell mesh. The greyed zoomed circle area represents a single injector.

**Table 3**  
Mesh quality parameters.

Billion cell mesh	
Cell Number	1,002,355,456
Interior face number	3,005,213,856
Boundary face number	3,705,024
Vertices number	1,004,210,878
Minimum control volume (m <sup>3</sup> )	$6 \times 10^{-8}$
Maximum control volume (m <sup>3</sup> )	$7 \times 10^{-6}$
Total domain volume (m <sup>3</sup> )	633.0
Orthogonality quality	0 bad cell (0%)
Offset	344 bad cells ( $\ll 0.001\%$ )
Least-Squares Gradient Quality	11,776 bad cells ( $\ll 0.001\%$ )
Cells Volume Ratio	0 bad cell (0%)

- Using user defined mesh transformations, first dilate the mesh radially to obtain the expected diameters, then generate the bulb shape (disengagement zone) by shifting 3D coordinates of all vertices in the three top cylinders.
- Using the matching mesh merging feature of *Code Saturne* [66], merge all 3D meshes into a final single mesh.

These 3D steps are summarized by Fig. 8.

The final mesh contains 1,002,355,456 hexahedrons whose mean volume is about  $0.125 \text{ cm}^3$ , i.e.  $5 \text{ mm}$  of characteristic size (see Fig. 9). The whole mesh connectivity is conformal. The refinement of this mesh is such that each injector of diameter  $d_{inj} = 0.21 \text{ m}$  is defined by more than 350 cells. The mesh file, saved in a double precision binary format is about 209 GB in size.

All operations performed using *Code Saturne* [66] benefited from the parallelization of the code and ran on CALMIP supercomputer (see Section 5.1 hereafter). This mesh creation step requires 90 nodes to have enough memory. The mesh construction step is critical for the simulation quality. It is a quick step in terms of CPU hours, note however that it was actually a long step in terms of dedicated human time: about 4 weeks.

Main characteristics of this mesh are detailed in Table 3. Quality criteria of NEPTUNE\_CFD and *Code Saturne* are defined in their user's manual [66]. The few cells marked as bad by respective criterion are located over the bed, in the top of the bulb. They will not impact the quality of the simulation.

To benefit from HPC capabilities of NEPTUNE\_CFD [27], partitioning of the mesh is required. Each CPU core will handle computations for one

domain and will exchange information to CPU cores handling neighbors domains through MPI communications for ghost cell data. We look for a balanced partitioning (same number of cells in each domain) which also minimizes the number of neighbor domains for each domain, to limit inter domain communications to the bare minimum. The number of ghost cells (halo) is also a highly important metric of partitioning, and should be as small as possible. Other criteria to select a good partitioning algorithm or library are their stability (repeatable and stable) and the computation time required to perform this partitioning. Fig. 10 presents a decomposition in 64 domains of the mesh for two partitionner, PT SCOTCH and Morton method. Each color corresponds to a domain solved by a CPU core.

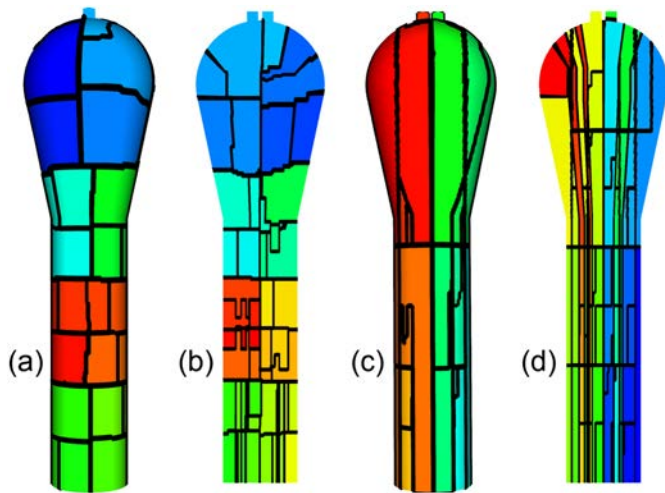
Multiple partitioning libraries are available and have been assessed (see Table 4). Sequential partitioners are not able to handle such a large mesh. We then evaluated five parallel partitioners: PT SCOTCH [68], ParMETIS [69], Morton Curve, Hilbert Curve and Block methods. PT SCOTCH and ParMETIS make use of eponymous libraries, using default options, and are based on graph partitioning. Morton and Hilbert curve approaches are algorithm that use the corresponding space filling curves (see Fig. 11) to sort mesh cells. The obtained curve is then split into the required number of domains which induces perfect domain balance.

With this billion cell mesh, we wanted to create mesh partitioning from 1260 up to 36,000 cores, i.e. to create 1260 up to 36,000 domains.

ParMETIS (Parallel version of METIS version 4.0.2 [69]) was not able to handle such a large mesh over such a high number of cores, it was not stable and simply crashed and is thus discarded from the comparison with other methods. PT SCOTCH could generate a high number of domains, but could not run on many CPU cores either, otherwise the partitioning step would fail. This induces a very slow domain partitioning due to the low computational power of a limited number of cores. Nevertheless, this library generated the best results in terms of neighbor and ghost cells balancing. The domain unbalance was about 20%. Overall, this disequilibrium is compensated by the low number of ghost cells and induced more favorable computation times. Even though this library yields interesting results, the fact that it does not scale robustly for high domain counts ( $>4000$ ) is very limiting. More over, this algorithm is based on a random seed and is thus non reproducible. We then saved the best partitioning among multiple calls to the library and re used it for multiple simulations.

The Block partitioning uses a dummy approach (simply partitioning by slabs based on initial ordering) which may be useful in some cases, but here, the domain unbalance was over 10%, the neighboring unbalance was bad as well as the ghost cell unbalance.

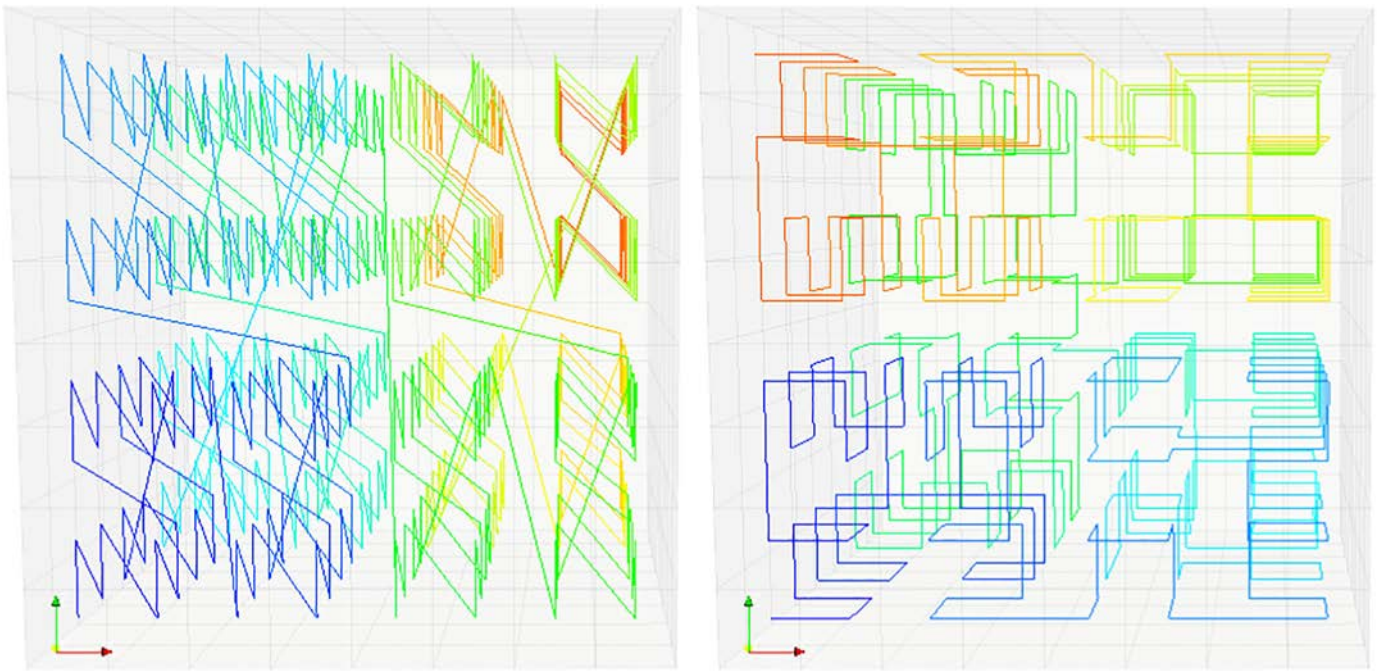
Morton curve and Hilbert curve algorithms behave very similarly: stable, efficient, perfect balance for the chosen criteria (same number of local cells per domain  $\pm 1$  cell). The Morton curve led to slightly better



**Fig. 10.** Illustration of mesh partitioning of the geometry into 64 domains. (a) PT-SCOTCH - exterior view. (b) PT-SCOTCH - centered cut plane. (c) Morton - exterior view. (d) Morton - centered cut plane.

**Table 4**  
Partitioner comparison. Stability indicates whether the partitioning step succeeded. Partitioning time, and the duration of a reference simulation that ran on obtained partitionings (CFD CPU time) are normalized by the results for Morton method.

	Morton	PT-SCOTCH	Hilbert	Block
Stability	good	limited to few cores	good	good
Partitioning time	1.0	4.7	1.0	3.0
Domain unbalance	$\ll 1\%$	$\approx 20\%$	$\ll 1\%$	$\approx 10\%$
Neighbor unbalance	good	excellent	good	bad
min-max with 50 nodes	10–43	5–23	10–52	–
Ghost cell unbalance	good	very good	good	bad
min-max with 50 nodes	54,271–115,996	31,200–63,732	53,213–118,784	–
CFD CPU time	1.0	1.0	1.2	3.0



**Fig. 11.** (left) Illustration of the Morton (or Z) space-filling curve, used to sort mesh cells by the Morton partitioner. (right) Illustration of the Hilbert space-filling curve, used to sort mesh cells by the Hilbert partitioner.

**Table 5**  
Supercomputer main characteristics.

Olympe - peak performance 1.365 Pflop/s			
13,392 cores (2.3GHz)			
374 compute nodes	Cores / node Processor - GPU	RAM/node	1 node
360 bi-socket	2 × 18 - Intel® Xeon® Gold Skylake 6140	192 GB	2.65 TF
12 bi-socket + GPU	2 × 18 - Skylake 6140-4 GPU NVIDIA V100	384 GB	33.8 TF
2 bi-socket memory	2 × 18 cores SKYLAKE 6140	1536 GB	2.65 TF
Gaia - peak performance 3.05 Pflop/s			
127th TOP500 (06/2019) - 42,912 cores (2.3GHz)			
1224 compute nodes	Cores / node Processor - GPU	RAM/node	1 node
1192 bi-socket	2 × 18 - Intel® Xeon® Gold Skylake 6140	192 GB	2.65TF
32 bi-socket + GPU	2 × 4 Skylake 5122-2 GPU NVIDIA V100	384 GB	18 TF
48 bi-socket memory	2 × 18 - Intel® Xeon® Gold 6140	384 GB	
32 bi-socket memory	2 × 4 - Xeon Gold 5122	384 GB	

results than the Hilbert curve in terms of neighboring unbalance, ghost cell unbalance and simulation CPU time.

For large meshes, the space filling curve partitioning is thus preferred, with Morton being the best choice here. This is consistent with observations on many other large cases handled with *Code Saturne*.

NEPTUNE\_CFD allows creating all needed mesh partitioning before simulation runs, using a pre processing mode. This is useful mainly to allow partitioning from a different number of ranks, and is often faster than recomputing the partitioning at each restart (depending on the performance of the parallel I/O subsystem). The different partitioning rank mappings are exported and can be re used according to CPU core number wished for simulation. Using this functionality we created all mesh partition maps before performing any simulation. Depending on the number of CPU cores for a set simulation, we read the adequate partition map during the simulation preprocessing step.

## 5. Results and discussion

### 5.1. Hardware and software resources

Numerical simulations of the present work have been made on the supercomputer Olympe at the Toulouse University

Computing Center<sup>2</sup> and on the supercomputer Gaia of EDF R&D. Main characteristics of these supercomputers are summarized in Table 5.

The supercomputer Olympe is a BullSequana X1000, last generation of the European vendor ATOS aimed to Exascale computations. This cluster is made of 374 compute nodes interconnected with an infiniband fabric (EDR 100 GB/s) in a fat tree topology (blocking factor 2:1). Processors are Intel® “Skylake” 6140 (18 cores per socket@ 2.3 GHz), 13,392 cores as a whole in the cluster. All nodes have a bi socket architecture, *i.e.* two processors of 18 cores each. There are three kinds of compute nodes: 360 classical bi socket nodes, 12 nodes with GPU accelerator and 2 large memory nodes. All simulations ran on classical bi socket partition, except the 362 nodes simulations which used some GPU nodes but without using GPU accelerators. For this compute partition, nodes memory reaches 192 GB of RAM, for a total amount of 69 TBytes of distributed RAM. The overall peak performance is 1.365 Pflop/s. For storage a Lustre file system with 40 GB/s of bandwidth and 1.5 PB of space is connected to nodes through an infiniband fabric.

<sup>2</sup> [www.calmip.univ-toulouse.fr](http://www.calmip.univ-toulouse.fr)

The supercomputer Gaia is a Atos Bull Cluster, also last generation of the European vendor ATOS aimed to Exascale computations. Gaia's architecture is close to that of Olympe, but is three times bigger in terms of nodes and peak performances. Gaia overall peak performance is about 3.05 Pflop/s which ranks it 127th of TOP500 citeptop500 [25] as of June 2019. The local storage is based on IBM Spectrum Scale (GPFS) and DDN SFA14KE with 200 GB/s of bandwidth and 16 PB of space. The parallel file system is connected to nodes through Intel OPA v1 and the total core number is 42,912.

Presented simulation benefited from the opportunity offered by pre-production operations associated with the installation of Olympe and Gaia supercomputers. These installation phases allowed using these systems in an exclusive and dedicated manner. During the set up phase, strong interactions with CALMIP and EDF R&D Scientific Information System architects and HPC engineers were necessary to circumvent all technical issues due to the fact that supercomputers were at this time newly installed. The size of the simulation made it an excellent benchmark for these two supercomputers and allowed fine tuning and validating them for IO, MPI, filesystems and node stability.

Major issues were two folds: Input/Output for one, and runtime for more than 10,000 processes for the other. It is noteworthy that memory footprint of the simulation was very low for such a scale, and was not an issue. Indeed, less than 10 TB of distributed RAM was necessary, compare to the capacity of Olympe mentioned above (69 TB of distributed RAM). Due to the expected duration of the simulation, we needed to frequently write checkpoint file to restart the simulation and to avoid losing CPU computation time in case of software or hardware failure. The size of the simulation lead to a checkpoint restart file of 1.34 TBytes. On both supercomputers, libraries and software have been compiled using Intel Compiler 18.2 and relied on the MPI library IntelMPI 18.2. We used the capability of the software NEPTUNE\_CFD [27] to write and read in parallel (MPI IO) accordingly to the domain decomposition (partitioning). As we used the whole Olympe machine, this would have put a lot of pressure on the underlying file system (*i.e.* Lustre, characteristics mentioned above). Nonetheless, Olympe and Gaia exhibited a great stability for reading and writing operations and with good performances. For the runtime performance we experienced more difficulties. On Olympe, we encountered locks in communications with simulation above a threshold, namely over 5760 MPI processes (160 nodes). To circumvent the problem we had to finetune the Intel MPI library that we linked against NEPTUNE\_CFD to handle inter processes communications. This tuning is quickly detailed in section 5.3. As we found the right parameterization on Olympe, we were able to run successfully above 5760 MPI processes and reach the upper bound of 13,032 MPI processes (362 compute nodes). To sum up Olympe simulations consumed 5 millions of core hours, and generated sets of data of 120 TB.

On Gaia, we have been able to reach the upper bound of 36,000 MPI processes (1000 nodes). Simulations which ran on Gaia consumed 10 millions of core hours and generated sets of data of 200 TB. Gaia benefited from previous experience on Olympe and no MPI issues were faced at large scales, though the same finetuning as been applied. Gaia being in installation phase, some issues related to OPA connections were also detected during NEPTUNE\_CFD [27] installation. Using OPA, for performance reasons, we choose to use unpopulated simulations using 35 cores out of 36 on each node.

## 5.2. Run organization and data management

Numerous successive simulations using checkpoint/restart were required to reach the total simulated time of 25 s due to some issues related to the scale of this simulation. These aspects are further discussed in Section 5.3.

A critical aspect that has been considered before running the simulation is that of data export and postprocessing. On top of the raw challenge of running such large simulations, one goal of this study is to

obtain a dataset characteristic of the functioning of an industrial fluidized bed at the smallest scales currently achievable. The way these results may be used is discussed in Section 5.5 but generating these results is a whole challenge by itself.

In terms of data volume: NEPTUNE\_CFD solves 22 PDEs. Exporting all variables in EnSight Gold Format (single precision) would take 200 GiB per time step. The simulation mean time step is about  $2 \times 10^{-4}$  s, which corresponds to 125,000 timesteps for the whole 25 s simulation. It would then be impossible to save results either over the whole domain and/or for all timesteps. In the selected approach, 10 variables are exported at a rate of 40 exports per simulated second in 13 volumic and surfaces zones. Selected export areas are:

- 4 thick planes along the radial direction;
- 7 thick planes at specific heights;
- a whole cylindrical volume encompassing the catalyst injectors;
- the reactor external surface.

Despite reducing significantly the size of exports, the generated volume is still significant and very difficult to handle, manage and post process.

The Lustre file system is scalable, has high performances and allows parallel I/O operations. When using additional library for I/O (*i.e.* MPI IO in this case, but also HDF5 or parallel netCDF), reading and writing can be done in parallel from several nodes into single shared file. We used a technique called file striping to increase I/O performance. The Lustre file system is made up of an underlying set of I/O servers and disks called Object Storage Servers (OSSs) and Object Storage Targets (OSTs) respectively. A file is said to be striped when I/O operations access multiple OSTs concurrently. We enabled 4 possible concurrent accesses to each file, which allowed increasing the available I/O bandwidth significantly.

## 5.3. Computational performances (HPC)

The first learning from this billion mesh simulation is about NEPTUNE\_CFD computational performances. NEPTUNE\_CFD is powered by Code Saturne which was already known for its peta scale performances [58,59]. During this simulation, we have been able to evaluate NEPTUNE\_CFD HPC from 35 nodes up to 1000 nodes, *i.e.* from 1224 cores up to 36,000 cores on two supercomputers of last generation. The following analysis is based on the averaged CPU time value of 2 runs for each core number and we considered the mean value. No significant difference of CPU time is observed. NEPTUNE\_CFD scalability has been evaluated on a restart simulation of 200 iterations after a transient step. To obtain an evaluation of computing performances, undisturbed by I/O operations, the first and last iterations are discarded from the analysis. During these remaining 198 iterations, no disk access occurs except for printing parallel performances (negligible overhead). Table 6 summarizes the effective CPU time required per timestep (averaged over the 198 iterations). Computation elapsed time is very close to this time. On Olympe, we evaluated scalability from 35 up to 362 nodes and on Gaia, we started at 80 nodes up to 1000 nodes. As the two supercomputers use same processors, their performances can be directly compared despite slightly different architectures.

This study corresponds to a strong scaling study. Strong scaling is defined as how the solution time varies with the number of cores for a fixed total problem size. It is hard to achieve a good strong scaling at larger process counts since the communication overhead increases in proportion to the number of processes used. The speedup is defined as the ratio between the elapsed time to execute a program on reference node number (here 1260 cores on Olympe) and on a set of concurrent  $n$  nodes ( $n \times 36$  cores) with  $n > 35$ . As architectures of both supercomputers are similar, the reference time from Olympe is also used to analyse speed up on Gaia. The efficiency is defined as the ratio between speedup and  $n$  the number of nodes.

Table 6  
NEPTUNE\_CFD effective CPU time - scalability.

Olympe supercomputer (CALMIP)		
Node number	Core number	Effective CPU time (s)
35	1260	181.25
40	1440	161.40
45	1620	135.07
50	1800	121.18
75	2700	81.17
100	3600	61.50
125	4500	52.08
150	5400	42.05
175	6300	35.79
200	7200	31.58
225	8100	29.18
275	9900	24.02
330	11,880	23.43
362	13,032	23.15
Gaia supercomputer (EDF)		
Node number	Core number	Effective CPU time (s)
80	2880	72.91
160	5760	37.01
320	11,520	20.21
640	23,040	12.65
800	28,800	11.60
1000	36,000	11.37

$$\text{Speedup} = \frac{T_{\text{ref}}}{T_n} \frac{T_{35 \text{ nodes}}}{T_{n \text{ nodes}}} \text{ with } n > 35 \text{ nodes} \quad (8)$$

$$\text{Efficiency} = \frac{\text{Speedup}}{n} \quad (9)$$

where  $T_{\text{ref}}$  is the effective core CPU time on the reference node number (here 35) and  $T_n$  is the effective core CPU time on  $n$  nodes ( $n > 35$ ). NEPTUNE\_CFD effective CPU time per core, speedup and efficiency of the calculation are depicted by Figs. 12, 13 and 14 respectively. Fig. 12 exhibits a seemingly hyperbolic evolution of cpu time per iteration against core number which is expected for code strong scaling and demonstrates the good scalability of NEPTUNE\_CFD. The solid red line on Figs. 13 and 14 represents the ideal speedup and efficiency respectively.

Figs. 12 and 14 underline that NEPTUNE\_CFD scalability and parallel performances are excellent. The speedup is ideal up to 12,000 cores and

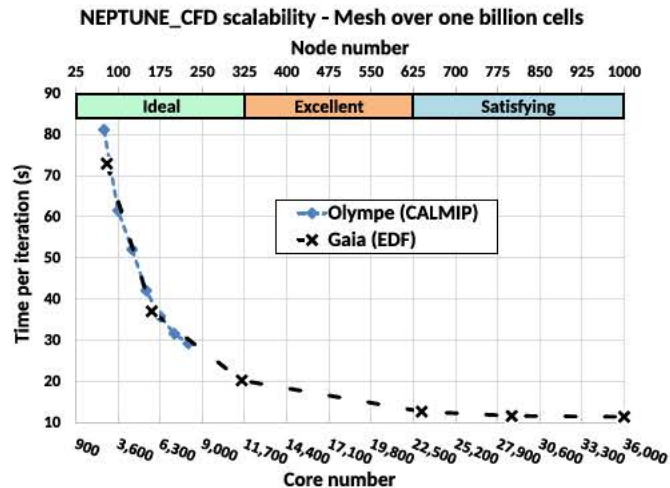


Fig. 12. NEPTUNE\_CFD scalability on billion mesh. Performances are almost identical between Olympe and Gaia, though slightly better for Gaia.

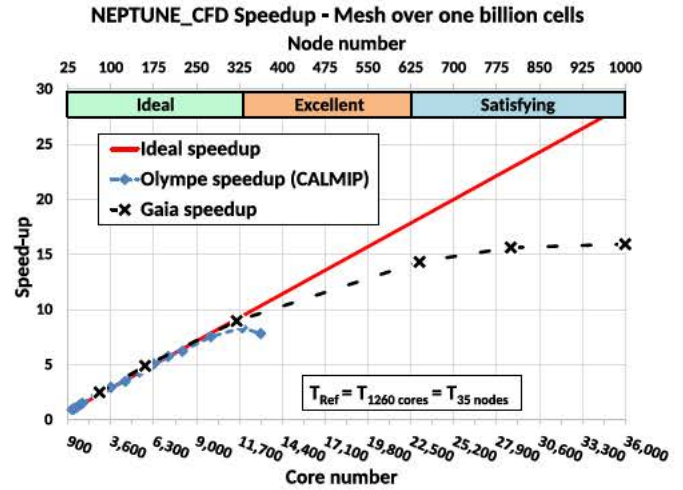


Fig. 13. NEPTUNE\_CFD speed-up on billion mesh. The green "Ideal" area exhibits linear behavior, the orange "Excellent" area shows a slightly sub-linear gain, and the blue "Satisfying" area exhibits a speed-up close to stagnation. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

remains very good up to 22,000 cores where the efficiency is about 80%. Even using 36,000 cores the efficiency remains satisfying, over 55%.

Considering the mesh cell number per core, that means the scalability is perfect for this problem while the cell number per core is over 70,000 and excellent while the cell number per core is over 45,000 cells. Note that this scalability metrics is disputable and difficult to generalize as it depends in fact on the number of operations performed per core relative to the amount of data exchanged between cores. If there were more equations to solve (more species, more complex turbulence models, more phases, ...), the number of operations per core would increase and the minimum number of mesh cell to have a good scalability would decrease. In the same time, the volume of data exchanged between cores would increase.

One metric that is interesting to characterize is the number of second of physical time per core hour CPU. On average, for this billion computation, this metric was about  $1.7 \mu\text{s}/h_{\text{core CPU}}$ . From a performance point of view, NEPTUNE\_CFD keeps a good efficiency even for high number of nodes. However, when aiming at efficient energetic usages, going beyond 625 nodes is not interesting and goes against good environmental practices. The energetic cost does not worth the low associated performance gain.

As stated above, this simulation ran during a period of tuning for both supercomputers. One of the significant key to solve encountered issues was to tune the IntelMPI library through environment variables, to handle communication on such a large scale. These issues are not detailed here, because they are not characteristic of a production functioning.

In terms of MPI communications, a detailed analysis showed an important part of collective communications: MPI\_ALL\_REDUCE when the number of cores increases very high. On this billion simulation, NEPTUNE\_CFD parallel performances are very sensitive to the memory bandwidth up to 600 nodes. Then with large number of MPI processes, the memory size per core decreases, and the solver becomes more sensitive to the CPU frequency along with network limitations.

Figs. 13 and mainly 14 underline an interesting behavior of NEPTUNE\_CFD which exhibits a super linear speed up up to 200 nodes on Olympe and 300 nodes on Gaia. Such a behavior is not necessarily surprising in a context of strong scaling. By reducing the problem size for each CPU core, we improve the residency of data on CPU cache thus improving significantly computation performances. This may explain this super linear speed up. Unfortunately, the context of mesochallenges did not offer the opportunity to profile more accurately

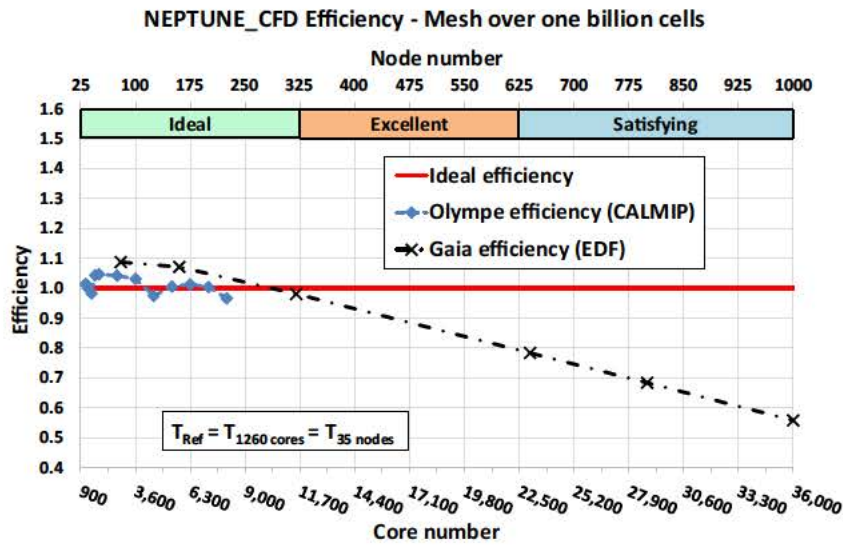


Fig. 14. NEPTUNE\_CFD efficiency on billion mesh. The "Ideal" area exhibits a super-linear efficiency; in following areas, the efficiency decreases linearly from 100 to 1000 nodes.

the simulation through dedicated tools. We could not access metrics about memory usage, RAM to cache transfers, and data residency which would be required to better understand the origin of this super linear behavior. Nonetheless, this acceleration seems reproducible on two different supercomputers and deserves to be more thoroughly investigated.

The conclusion of this part is that NEPTUNE\_CFD is able to use massive HPC resources at least up to 36,000 cores and conserves high efficiency up to 23,000 cores. While the number of mesh cell per core is over 70,000 the speed up is ideal and the efficiency is over 80% while this number is over 45,000. A good trade off between speed up and

electrical/HPC resources consumption is to use up to 22,500 cores, which implies at least 45,000 mesh cells per core.

#### 5.4. Simulation analysis

To help analyzing the simulation results, we first remind main characteristic times associated to this fluidized bed reactor. For this continuous reactor, the longest characteristic time is that of polyethylene production or large particles mean residence time in the reactive bed which is about 2 to 4 h. Fine particles and the gas mean residence times are respectively around 45 s and

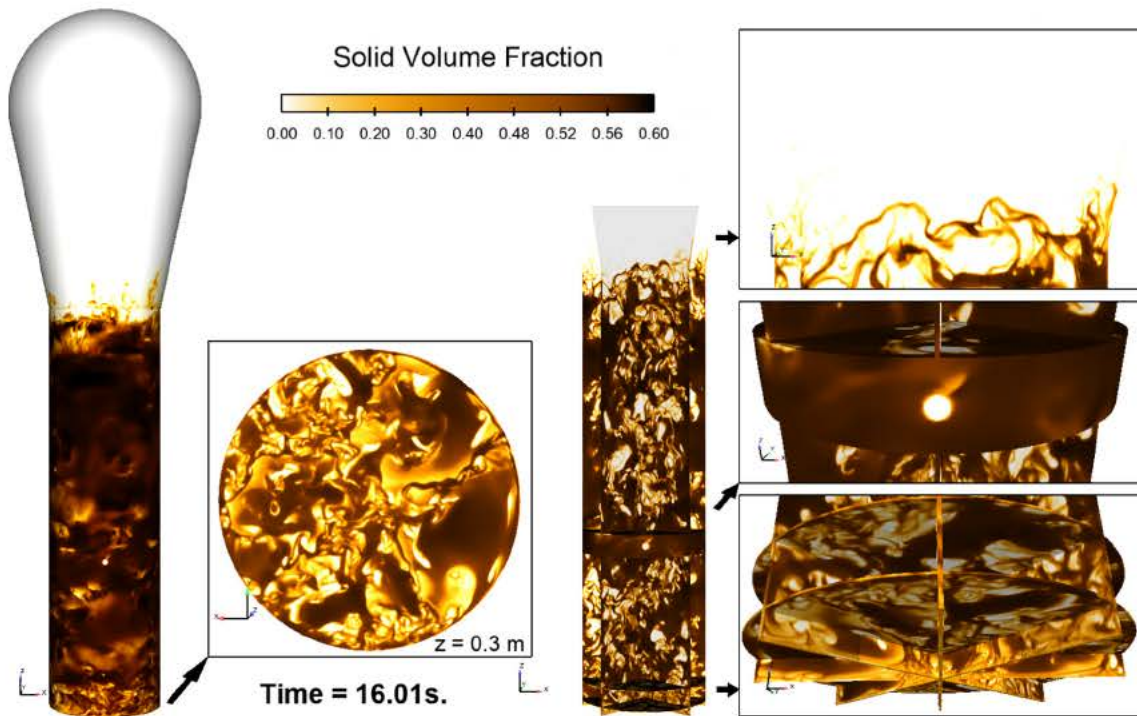
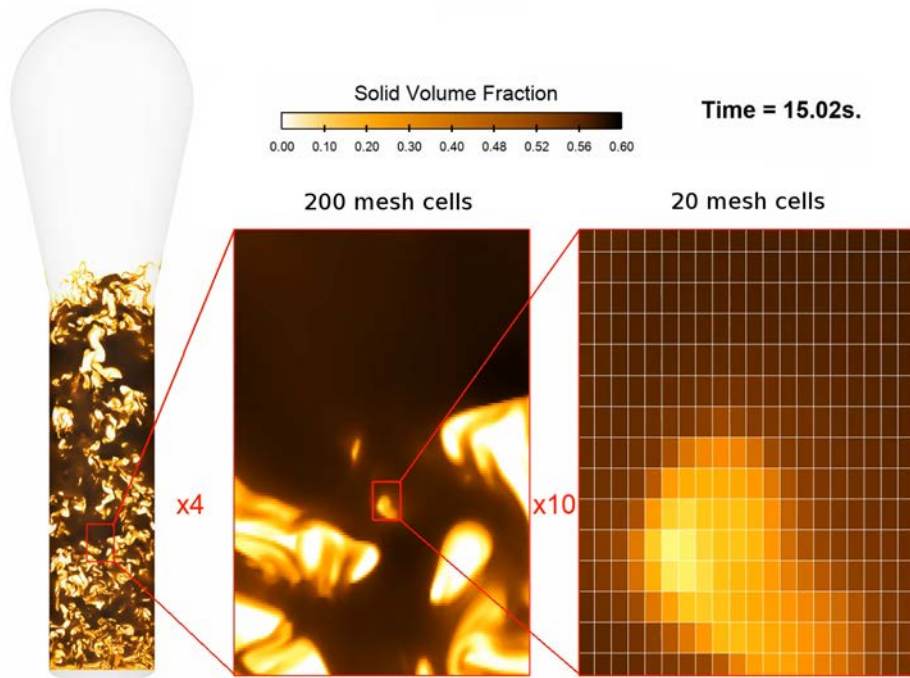


Fig. 15. Instantaneous large particle volume fraction after 16 s of simulation. From left to right: (i) values closest to the exterior surface of the reactor; (ii) a bottom cut plane just above the gas fluidization inlet (at an height  $z = 0.3$  m); (iii) cut planes and cylinder in the dense fluidized bed and (iv) three zooms on the bottom part, the injectors area and on the top limit of the fluidized bed.



**Fig. 16.** Instantaneous large particle volume fraction on a cut plane with successive  $\times 4$  and  $\times 10$  zooms. White lines on the right sub-figure are the actual mesh contours. Note that the color has not been interpolated between mesh cells, hence the pixelated aspect of the right-most sub-plot.

30s. Depending on their size, particle clusters have a life time ranging between 0.5s up to 5s which is in accordance with frequency of apparition of bubbles  $\leq 5$  Hz as usually observed in the literature.

As stated previously, the simulation ran “only” for 25s of physical time. In the first 5 to 10 s of transient regime, the bed is destabilized and the simulation loses track of initial conditions. Therefore, we have at best 15s of useful data with the behavior of a bubbling fluidized bed reactor. This is not enough to access converged mean values and explains why we only focus on instantaneous values fields and unsteady values. Accessing meaningful statistics would require at least 200s. Note however that, as far as clusters formation prediction is concerned, they are fully resolved as the simulation timestep ( $\delta t \approx 2 \times 10^{-4}$ s) is significantly lower than their life time. Considering previously described characteristic times, over the short simulated period, the total mass of particles can be considered stationary. Nevertheless, the simulated last 15s are representative of hydrodynamics unsteadiness.

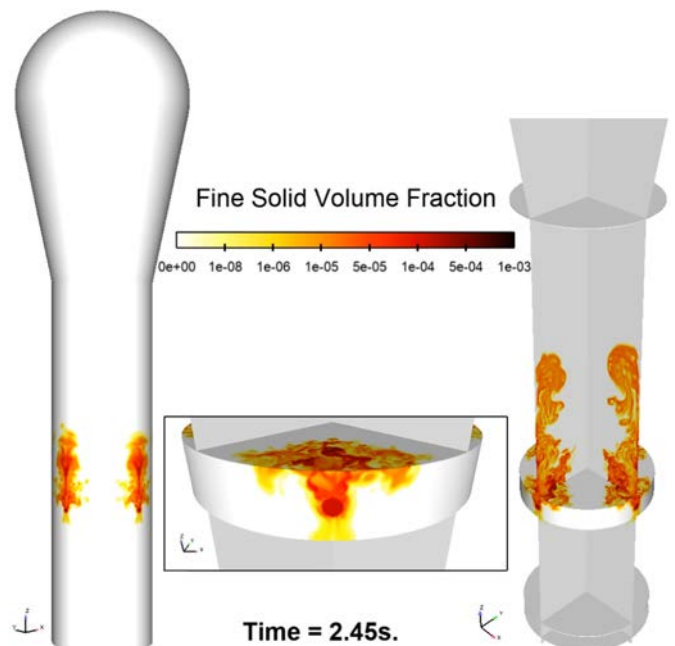
Fig. 15 presents the large particle solid fraction after 15s. The poly ethylene volume fraction is high with a mean value inside the bed around  $\langle \alpha_l \rangle \approx 0.45$ . Close to the wall, this value is higher between 0.56 and 0.6. In contrast, at the injector locations, the large particle volume fraction is null due to the gas and fine particles injection. At the top of the fluidized bed, we can see very small clusters and thread like structures. Only such a highly detailed computation could capture these small scale structures.

Fig. 16 also shows the large particle volume fraction but on a cut plane at the center of the reactor with two recursive zooms. Note the high local gradients of solid volume fraction  $\alpha_l$ . Along only a few cell,  $\alpha_l$  ranges between 0.0 and 0.6 which is close to the maximum solid volume fraction 0.64. We captured very small size structures such as bubbles and clusters (see Fig. 16) whose size is significantly larger than the mesh characteristic size. Though we may not claim that this simulation is independent from the mesh refinement, we got as close as currently possible from a fully resolved industrial scale reactor simulation.

If we focus on catalyst particles (see Fig. 17), the initial catalyst volume fraction is null and catalyst is injected through the four injectors. At the beginning the catalyst particles rise in the reactor and penetrate

inside the dense fluidized bed, then after 10 s, catalyst particles are present everywhere in the reactor. Note that their volume fraction reaches up to  $\alpha_f = 10^{-3}$  whereas they are injected with a lower fraction of  $\alpha_{inj} = 2.4 \times 10^{-5}$  which illustrates cluster formation (see Fig. 18).

One can observe a good dispersion of fine particles in the reactor with (i) an accumulation close to injectors, (ii) intermediate concentrations over these injectors and up to the top of the fluidized bed and (iii) low values at the bottom and in the bulb. If the simulation ran a little while longer, we would expect to see fine particles going out of the reactor by elutriation even if the catalyst velocity in the disengagement zone is lower than in the fluidized bed.



**Fig. 17.** Instantaneous catalyst particle volume fraction after 2.5 s.

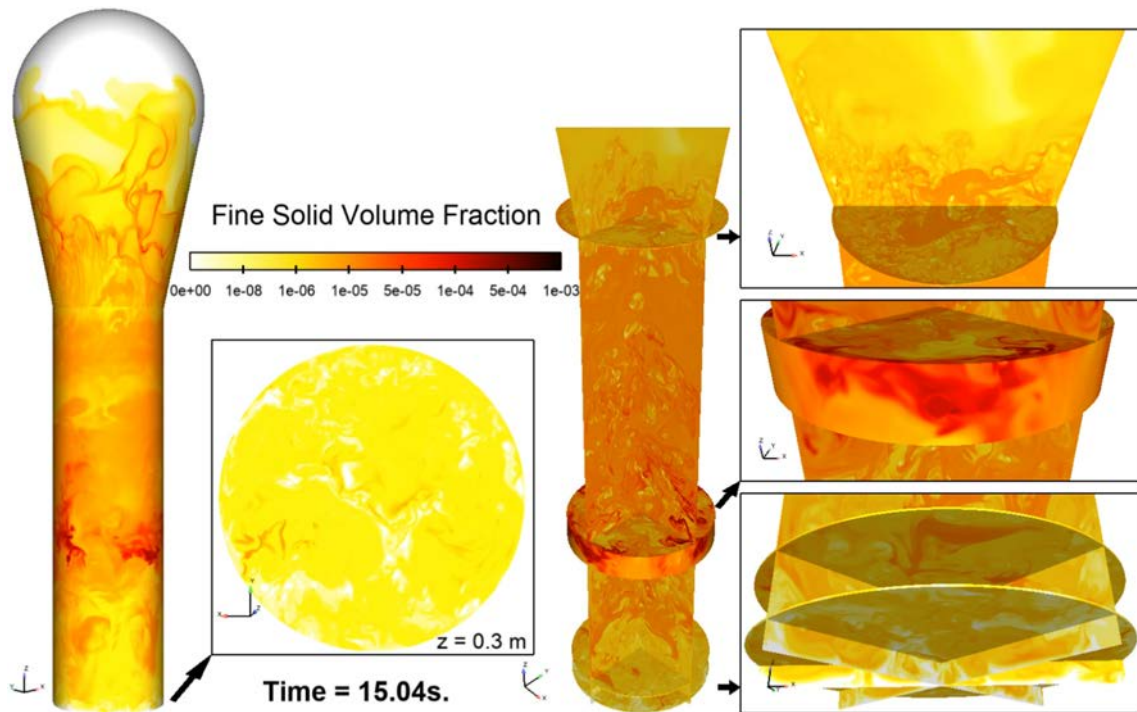


Fig. 18. Instantaneous catalyst particle volume fraction after 15 s.

As illustrated by Fig. 19, large particles are mainly segregated in high density clusters (high peak on the right hand side at  $\alpha_l \in [0.5; 0.6]$ ). A significant part of the volume is almost completely depleted from large particles ( $\alpha_l \in [0; 0.05]$ ). In between ( $\alpha_l \in [0.05; 0.50]$ ), a flat distribution of local volume fraction is observed.

One very interesting result is the thermal aspect. We took into account an exothermic reaction to be representative of the polymerization reaction through enthalpy source terms on particulate phases and gas heats up through particle gas enthalpy transfers. Fig. 20 plots the gas temperature (on the left) and the fine particle temperature. Fig. 21

shows axial temperature profiles for all three phases, at the center of the reactor, and near the wall on a  $x = 0$  plane. Near the fluidization inlet, gas has the set temperature of 50 °C and particles have temperature in between 50 and 98 °C. Close to injectors at height  $z = 6.0m$ , both gas and fine particles have a temperature of 50 °C and large particle temperature drops around 94 °C. Nevertheless, quickly after entering the reactor, all phase temperatures reach a narrow range between 97 °C and 101.6 °C. Being careful with the temperature scale, we can observe on Fig. 20 that a fluidized bed is an excellent mixer which yields a good temperature homogenisation. Despite a

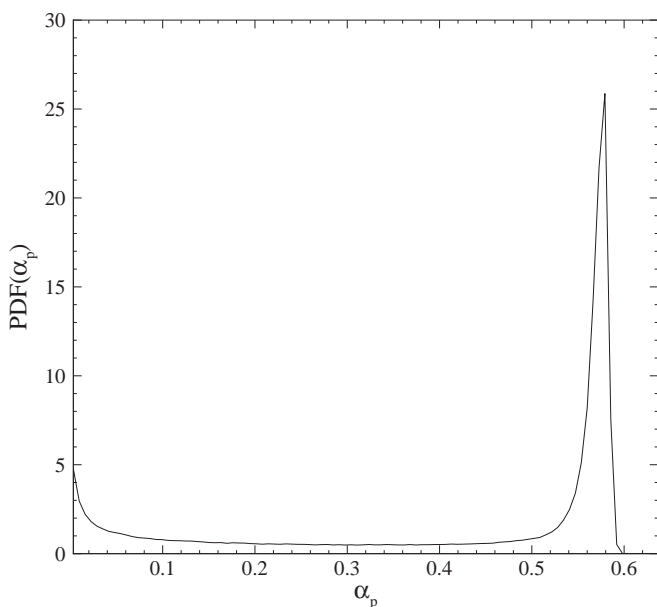


Fig. 19. Probability Density Function of large particles volume fraction. The data have been extracted from one vertical cut plane at time  $t = 16.5$  s.

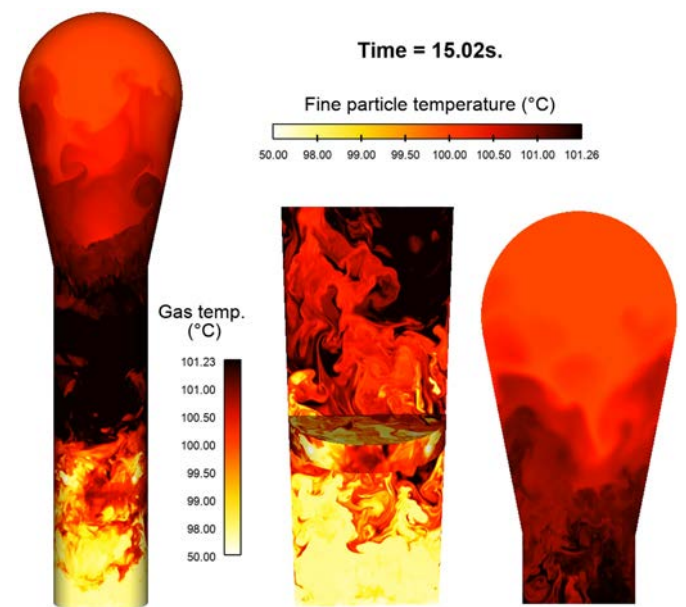


Fig. 20. Gas and fine particle temperatures after 15 s. Note the color-scale that is only linear between 98.00 °C and 101.23 °C.



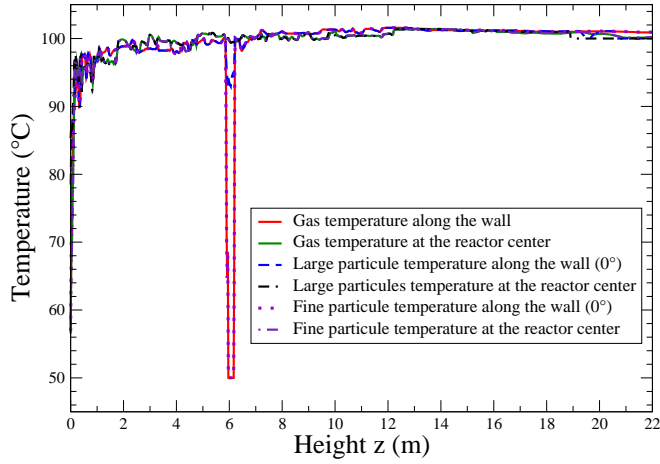


Fig. 21. Axial temperature profiles for all three phases at the center of the reactor and near the wall on a  $x = 0$  plane.

total produced power of 20MW, and catalyst particles that are very reactive, the range between min and max temperatures in the bulk of the reactor is about a few degrees ( $\approx 4^\circ\text{C}$ ).

Maximum gas and fine particles temperatures are observed right above injectors where  $\alpha_f$  is the highest, as fine particles are the most reactive (see section 2 and Eqs. (6) and (7)) and will have had a long enough residence time to heat up. It is important to notice that inside the dense fluidized bed, high local temperature gradient may appear which would not be noticeable on coarser meshes. Nevertheless, the time scale for temperature evolution is significantly higher than hydrodynamics time scales. Reaching a thermic equilibrium would require simulating several hundreds of seconds.

All these transient fields seem to represent the expected physics of an actual fluidized bed. We are able to simulate at industrial scale a reactive polydispersed pressurized fluidized bed reactor with an highly detailed mesh.

To be more quantitative, we can focus on wall pressure profiles, measured at  $t = 16.5\text{ s}$  as shown by Fig. 22. These profiles exhibit a linear slope with a maximum pressure drop of 63,000 Pa. A simple force balance, neglecting the friction force of particle on the wall, allows checking the total mass of particle:

$$M_p = \frac{P_{\max} S_R}{g} = 102,100 \text{ kg} \quad (10)$$

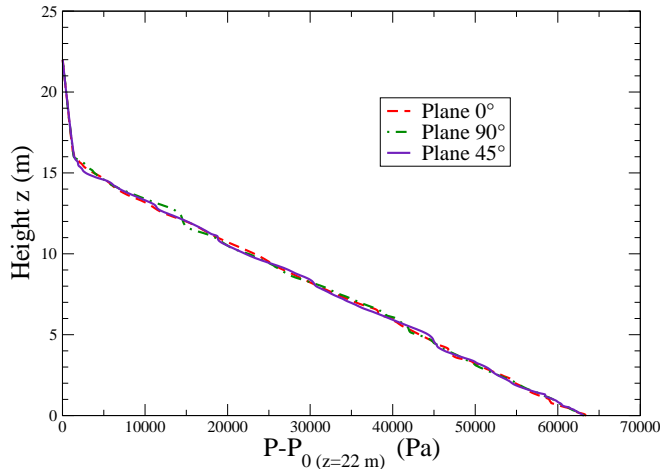


Fig. 22. Instantaneous pressure profiles along the wall at  $t = 16.5\text{ s}$  for vertical planes at different angles.

with  $S_R$  the section of the reactor cylindrical part. This total mass is consistent with the initial mass of large particles ( $M_{l,0} = 100\text{ t}$ ). The influence of injectors in planes at  $0^\circ$  and  $90^\circ$  does not disturb these profiles significantly. The height of the bed can be deduced by extending the bed pressure profile up to the height axis, thus obtaining a bed height of  $\approx 16.5\text{ m}$ .

### 5.5. From highly resolved simulations to sub grid modeling approaches

In the present paper we show that, nowadays, it is possible to perform a numerical simulation of an industrial scale geometry using a very fine mesh. However, such a highly detailed numerical simulation can only be performed with a very well parallelized efficient code and it requires a huge amount of computational resources. For engineers and researchers, computational resources are limited and the number of accessible cores as well.

Since several years researchers have identified the effect of the small scale solid structures present in particulate flows which are not accounted for when the numerical simulation is performed with coarse mesh [35,37]. Agrawal et al. [70], Igci et al. [23], Parmentier et al. [30] and Özel et al. [28] show that, in dense and in circulating fluidized beds, when small scale solid structures are neglected, the first order effect appears on the drag force term (here in the term  $I_{g \rightarrow p,i}$  from Eq. (3)). Basically, the drag force is overestimated with a coarse mesh. As a consequence, in dense fluidized beds, the use of a coarse mesh leads to an over prediction of bed height and in circulating fluidized beds the solid mass flux is overestimated. To overcome this effect and then to correctly predict the dynamics of particulate flows, researchers develop a new approach called Filtered Two Fluid Model (FTFM) [38–40]. Such an approach is widely inspired from the Large Eddy Simulation (LES) for single phase flows.

The use of a coarse mesh is similar to a filter applied on PDE equations. For a given function  $f(\mathbf{r})$  let us define the filtered value as  $f(\mathbf{x}) = \int f(\mathbf{r})G(\mathbf{x}-\mathbf{r})d\mathbf{r}$  where  $G$  is the filter kernel. From this the filtered solid volume fraction reads

$$\alpha_p(\mathbf{x}) = \int \alpha_p(\mathbf{r})G(\mathbf{x}-\mathbf{r})d\mathbf{r} \quad (11)$$

and the gas and particle velocities

$$\alpha_g(\mathbf{x})U_{g,i}(\mathbf{x}) = \int \alpha_g(\mathbf{r})U_{g,i}(\mathbf{r})G(\mathbf{x}-\mathbf{r})d\mathbf{r} \quad (12)$$

$$\alpha_p(\mathbf{x})U_{p,i}(\mathbf{x}) = \int \alpha_p(\mathbf{r})U_{p,i}(\mathbf{r})G(\mathbf{x}-\mathbf{r})d\mathbf{r}. \quad (13)$$

In the following  $\alpha_k$  and  $U_{k,i}$  are called the computed variables because those are the ones computed when a coarse mesh is employed. When such a filtering procedure is performed on the mathematical model given in Section 2, it leads to additional terms. As in single phase flow, the particle Reynolds stress can be decomposed as

$$\alpha_p \widetilde{U_{p,i}U_{p,j}} = \alpha_p U_{p,i}U_{p,j} + \tau_{ij}^{SGS} \quad (14)$$

where the first term on the right hand side is the particle Reynolds stress expressed with the computed variables and  $\tau_{ij}^{SGS}$  is the subgrid stress tensor that requires a model. However from a highly resolved numerical simulation it is possible to measure  $\tau_{ij}^{SGS}$ . Indeed, in the cylinder part of the reactor the mesh is nearly uniform and cartesian. Hence, we have extracted the data from a slice and a discrete spatial filter has been applied. Several filter widths  $\Delta_f$  have been applied up to  $64\Delta_{DNS}$  where  $\Delta_{DNS}$  is the cell size of the highly resolved simulation. The results are shown by Fig. 23 where it can be seen that the instantaneous spatial average of shear components are nearly null. In contrast, the vertical

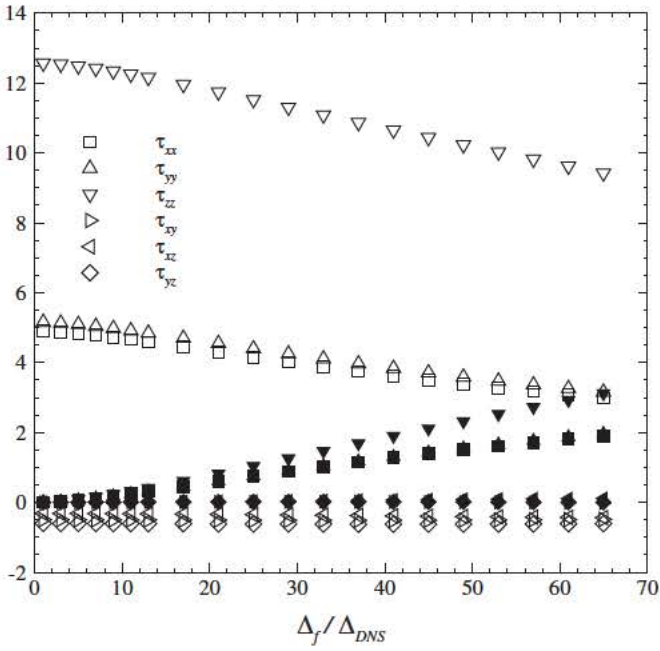


Fig. 23. Computed (open symbols) and subgrid (black-filled symbols) Reynolds stress tensor components normalized by  $\alpha_p \rho_p V_f^2$  with respect to the filter size  $\Delta_f$  (here  $\Delta_{DNS}$  is the size of the highly-resolved mesh). Extracted from one instantaneous field and in the cylinder part of the reactor.

component,  $\tau_{zz}^{sgs}$ , is larger than  $\tau_{xx}^{sgs}$  and  $\tau_{yy}^{sgs}$ . These last two values are found to be nearly identical. Fig. 23 shows that the computed averaged Reynolds stress components decrease when the filter width  $\Delta_f$  is increasing. As expected, the subgrid contributions is found increasing when increasing the filter width.

As already mentioned, recent works put the focus on the modeling of the drag force term because it has been found as of first order. When applying the filter on the drag term, it leads to

$$I_{g \rightarrow p, i} = -\frac{\alpha_p \rho_p}{\tau_{gp}^F} \widetilde{(U_{p, i} - U_{g, i})}. \quad (15)$$

The filtered drag term is decomposed in two terms. The first is written in terms of the computed variables,

$$I_{g \rightarrow p, i}^{comp} = -\frac{\alpha_p \rho_p}{\tau_{gp}^F} (U_{p, i} - U_{g, i}) \quad (16)$$

and a subgrid contribution  $I_{g \rightarrow p, i}^{sgs}$  that requires to be modeled. Obviously the filtered drag writes

$$I_{g \rightarrow p, i} = I_{g \rightarrow p, i}^{comp} + I_{g \rightarrow p, i}^{sgs}. \quad (17)$$

Fig. 24 shows the evolution of the spatial averaged computed and subgrid drag terms measured from the highly resolved numerical simulation. Here again the vertical component is dominant. As already shown by Agrawal et al. [70], Igci et al. [23] and Schneiderbauer [36], the computed drag force is found increasing when the filter width is increasing. It confirms that the use of a coarse mesh leads to an overestimation of the drag.

To model the subgrid drag term, an approach consists in the introduction of a subgrid drift velocity  $V_{d, i}$  as

$$I_{g \rightarrow p, i} = -\frac{\alpha_p \rho_p}{\tau_{gp}^F} (U_{p, i} - U_{g, i} - V_{d, i}). \quad (18)$$

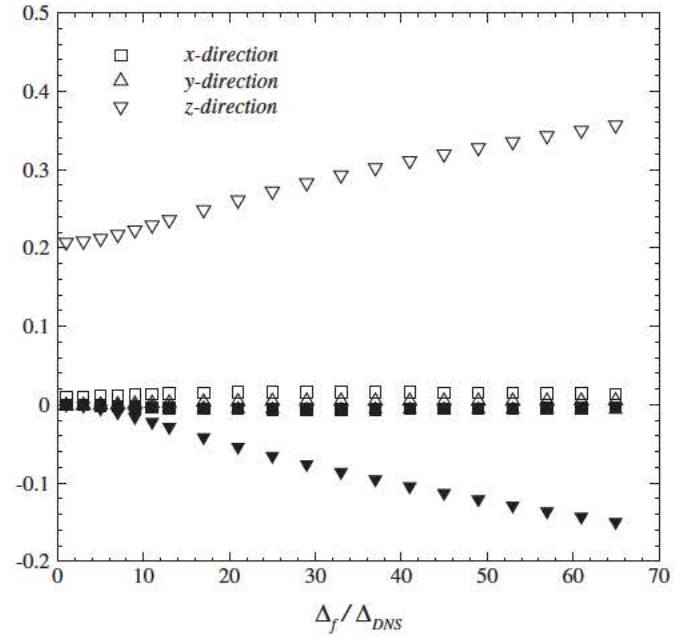


Fig. 24. Computed (open symbols) and subgrid (black-filled symbols) drag terms normalized by the gravity term  $\alpha_p \rho_p g$  with respect to the filter size  $\Delta_f$  (here  $\Delta_{DNS}$  is the size of the highly-resolved mesh).

Under such an assumption, the drift velocity is given by  $V_{d, i} = \widetilde{\alpha_p U_{g, i}} / \alpha_p$  and the subgrid drag term can be rewritten as

$$I_{g \rightarrow p, i}^{sgs} = \frac{\alpha_p \rho_p}{\tau_{gp}^F} V_{d, i}. \quad (19)$$

The modeling of the drift velocity has focussed many attentions over the past years. Following Igci et al. [23], Parmentier et al. [30] and Özal et al. [28], the drift model can be expressed in terms of the computed relative velocity. Such a modeling approach is supported by the strong correlation between the subgrid drift velocity and the relative computed gas particle velocity as shown by Fig. 25. A direct consequence of this correlation is that the subgrid drift velocity may be written as

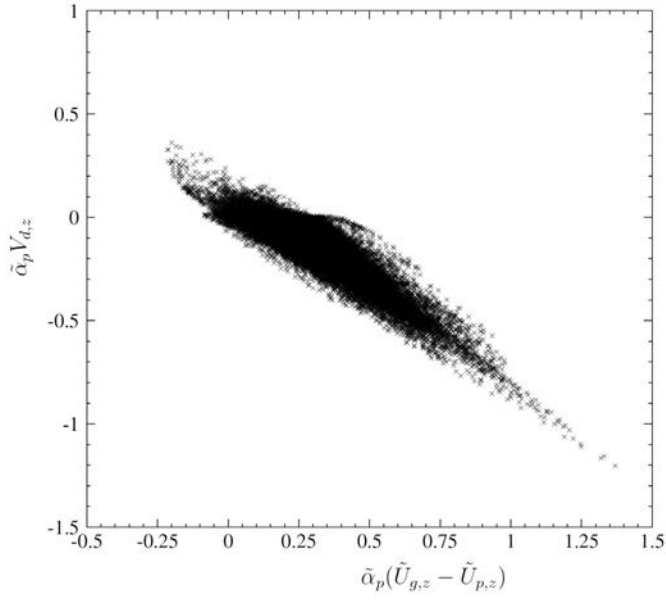
$$V_{d, \beta} = K_{\beta g} g(\Delta_f, \alpha_p) (U_{g, \beta} - U_{p, \beta}) \quad (20)$$

where  $\beta$  is used to indicate that there is no implicit summation. In Eq. (20)  $K_{\beta g}$  is a model constant that can be dynamically adjusted [28,30] like in single phase flow [71]. Finally, the function  $g(\Delta_f, \alpha_p)$  must be modeled but the highly resolved simulation permits a direct measure of this. Indeed it can be obtained from conditional averaging,

$$g(\Delta_f, \alpha_p) = \frac{\langle \alpha_p V_{d, z} | \alpha_p \rangle}{\langle \alpha_p (U_{g, z} - U_{p, z}) | \alpha_p \rangle}. \quad (21)$$

In Eq. (21) it can be noticed that the computation is performed in the vertical direction where the subgrid drag effects are the most important. Fig. 26 shows the function  $g(\Delta_f, \alpha_p)$  with respect to the filtered solid volume fraction and for several filter widths.

Assuming that the effect of the mesh and the effect of the local filtered solid volume fraction are uncorrelated, the function  $g(\Delta_f, \alpha_p)$  can be written as  $g(\Delta_f, \alpha_p) = f(\Delta_f) h(\alpha_p)$  where  $f(\Delta_f)$  represents the effect of the mesh and  $h(\alpha_p)$  the effect of the local value of the particle volume fraction. The last function can be extracted from



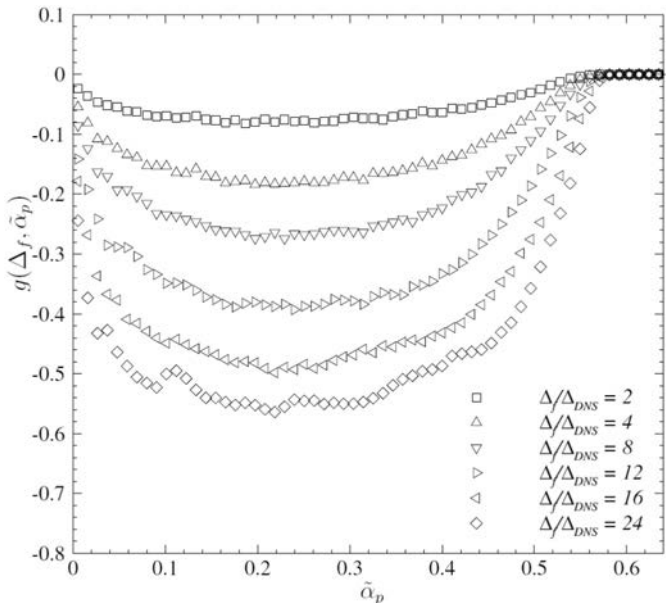
**Fig. 25.** Correlation diagram of the drift velocity in z-direction with respect to the computed gas-particle relative velocity. The filter width is  $\Delta_f/\Delta_{DNS} = 24$ . Only 10% of points are represented.

$$h(\alpha_p) \approx \frac{g(\Delta_f, \alpha_p)}{\int_0^{\alpha_{max}} g(\Delta_f, \alpha_p) d\alpha_p} \quad (22)$$

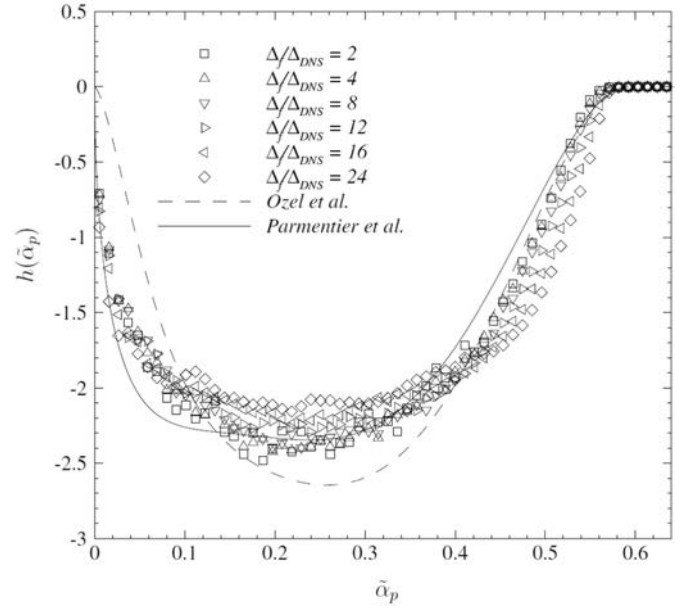
Fig. 27 shows the evolution of  $h(\alpha_p)$  for several values of filter width. The different curves merge into a unique function. In the literature two models have been proposed. First, Parmentier et al. [30] from 2D highly resolved numerical simulation of dense fluidized proposed

$$h(\alpha_p) = \sqrt{\frac{\alpha_p}{\alpha_{max}}(1-u)^2 \left[ 1 - C_{h,2} \left( \frac{\alpha_p}{\alpha_{max}} \right) + C_{h,3} \left( \frac{\alpha_p}{\alpha_{max}} \right)^2 \right]} \quad (23)$$

where  $u = \alpha_p/\alpha_{max}$  and the constant are set to  $C_{h,2} = 1.88$  and  $C_{h,3} =$



**Fig. 26.** Function  $g(\Delta_f, \alpha_p)$  computed with Eq. (21).



**Fig. 27.** Function  $h(\alpha_p)$  with respect to filtered solid volume fraction. The solid line corresponds to Eq. (23) and the dashed line to Eq. (24) both computed with  $\alpha_{max} = 0.592$  and normalized by their own integral.

5.16. On another hand, Özel et al. [28] used 3D highly resolved numerical simulations of a periodical circulating fluidized bed and proposed

$$h(\alpha_p) = \tanh\left(\frac{\alpha_p}{C_{h,1}}\right) \sqrt{\frac{\alpha_p}{\alpha_{max}}(1-u)^2 \left[ 1 - C_{h,2} \left( \frac{\alpha_p}{\alpha_{max}} \right) + C_{h,3} \left( \frac{\alpha_p}{\alpha_{max}} \right)^2 \right]} \quad (24)$$

with  $C_{h,1} = 0.1$ . Fig. 27 shows that  $g(\Delta_f, \alpha_p) / \int_0^{\alpha_{max}} g(\Delta_f, \alpha_p) d\alpha_p$  is independent from the filter width. Also Fig. 27 exhibits that the function proposed by Parmentier et al. [30] is in good accordance with measures from the highly resolved industrial scale numerical simulation.

## 6. Conclusion and outlook

To describe all flow scales in a polymerization fluidized bed reactor, very fine simulations are required. In the present study, one demonstrates that the simulation of a fluidized bed at industrial scale with a billion mesh cells using to 36,000 cores is now achievable. This has been possible thanks to recent advances in the computational power of supercomputers and the development of efficient massively parallel CFD solvers. This is the first time that such a detailed insight view of an industrial reactor has been obtained. Our previous largest simulation used a meshing ten times smaller than the current one [26]. Now, thanks to the achieved size of mesh cells, we may consider simulating complex geometries of industrial reactors. Due to the novelty of performing that large simulations, new challenges appear, both in terms of HPC, and in the exploitation of the simulation results. Tackling these challenges required a close collaboration between research laboratories, supercomputing centers and an industrial partner which develops a massively parallel solver up to date in terms of available models and numerical methods.

HPC challenges occurred at all steps of this project. First, the preprocessing of the simulation was tricky. To generate and partition an unstructured mesh with 1,002,355,456 hexahedral cells, a multi software procedure has been specifically designed. Multiple partitioning methods have been compared in terms of stability, performance and quality of the domain decomposition. Second, the simulation ran on up to 36,000 cores on newly installed supercomputers which had to be fine tuned to handle that massive computations. The simulation

had impressive metrics: more the 200 TB of data have been generated for 25 s of simulated physical time, each checkpoint restart file was about 1.3 TB and reading it on the whole of the supercomputer Olympe required up to 13 min. Postprocessing was to be considered before the simulation ran to limit the volume of generated data. Finally, the transfer and post processing of such a crushingly high volume of data was a challenge by itself.

Though being feasible, this kind of simulation still requires tremendous resources and is not easily performed. Each step (simulation setup, running and post processing) required about a month each, and expensive resources. For this reason and for now, the goal of these large scale simulations must be to generate reference databases. Ideally, such a reference simulation should be longer than the one we performed, to ensure the convergence of averaged results. This was however the best that we could obtain for now, and supplementary advances will be required to reach converged fully resolved simulations. Once reference results are generated, their exploitation may serve to improve the quality of lower cost simulations. This is achievable by exploiting fully resolved simulation results to measure sub grid statistics such as fluid particle drag, velocity or volume fraction variance. Sub grid models [17,29–32] are then tested *a priori* and *a posteriori* against the fully resolved simulation. This allows accounting for the effect of small and meso scale structures even when a coarse mesh is used.

As stated previously, one significant challenge has been the management of significantly large datasets. In terms of actual storage space, I/O operations dedicated time, site to site transfer rates and associated human working time, we again reached the upper limit of what was feasible. A strategy to significantly reduce these issues would be to switch the whole post processing to *in situ* visualization and co processing; *i.e.* post processing performed continuously during the simulation with tools like Catalyst ParaView [72]. By switching toward these co processing approaches, storage footprint would significantly decrease which goes along with good practices for “green” supercomputing practices [73]. For the past few decades, we analyzed computational performances for the simulation of the considered fluidized bed reactor mainly in terms of raw computational power ( $S_{\text{simulated}}/h_{\text{CPU}}$ ). However, in line with concerns for green practices, a better metrics than the one we presented previously would be the consider energetic performances ( $S_{\text{simulated}}/J_{\text{electrical}}$ ).

Overall, this work is a premiere which shows that fully resolved simulations of industrial scale fluidized bed reactors are now possible. They are of academic and industrial interest and should become more and more common in the next few years. Some of the challenges that we encountered should then become more easily tackled with solutions that starts becoming more accessible. With the expected evolution of HPC resources, the numerical cost of this scale of simulation should decrease and reach a reasonable level.

## Declaration of Competing Interest

None.

## Acknowledgments

This work was granted access to the HPC resources of CALMIP supercomputing center under the allocation 2018 mesochallenge and EDF Gaia cluster during EDF Grands Challenges.

## References

- [1] A. Shamiri, M.A. Hussain, F.S. Mjalli, M.S. Shafeeyan, N. Mostoufi, Experimental and modeling analysis of propylene polymerization in a pilot-scale fluidized bed reactor, *Ind. Eng. Chem. Res.* 53 (21) (2014) 8694–8705, <https://doi.org/10.1021/ie501155h>.
- [2] M. Khan, M. Hussain, Z. Mansourpour, N. Mostoufi, N. Ghasem, E. Abdullah, CFD simulation of fluidized bed reactors for polyolefin production – a review, *J. Ind. Eng. Chem.* 20 (6) (2014) 3919–3946, <https://doi.org/10.1016/j.jiec.2014.01.044>.
- [3] V. Akbari, T.N.G. Borhani, A. Shamiri, R. Aramesh, M.A. Hussain, M.K.A. Hamid, 2D CFD-PBM simulation of hydrodynamic and particle growth in an industrial gas phase fluidized bed polymerization reactor, *Chem. Eng. Res. Des.* 104 (2015) 53–67, <https://doi.org/10.1016/j.cherd.2015.07.016>.
- [4] P. Hui, L. Yuan-Xing, L. Zheng-Hong, Computational uid dynamics simulation of gas–liquid–solid polyethylene fluidized bed reactors incorporating with a dynamic polymerization kinetic model, *Asia Pac. J. Chem. Eng.* 14 (1) (2018) <https://doi.org/10.1002/apj.2265e2265>.
- [5] D. Geldart, Types of gas fluidization, *Powder Technol.* 7 (1973) 285–292.
- [6] A. Gobin, H. Neau, O. Simonin, J. Llinas, V. Reiling, J. Sélo, Fluid dynamic numerical simulation of a gas phase polymerization reactor, *Int. J. Numer. Methods Fluids* 43 (2003) 1199–1220.
- [7] O. Batrak, G. Patino, O. Simonin, I. Flour, T.L. Guevel, E. Perez, Unlike particles size collision model in 3D unsteady polydispersed simulation of circulating fluidized bed, *Circulating Fluidized Bed Technology VIII: Proceedings of the 8th International Conference on Circulating Fluidized Beds, Circulating Fluidized Bed Technology, Hangzhou, China 2005*, pp. 370–377.
- [8] N. Deen, M.V.S. Annaland, M. Van der Hoef, J. Kuipers, Review of discrete particle modeling of fluidized beds, *Chem. Eng. Sci.* 62 (1) (2007) 28–44.
- [9] M. Van der Hoef, M. van Sint Annaland, N. Deen, J. Kuipers, Numerical simulation of dense gas–solid fluidized beds: a multiscale modeling strategy, *Annu. Rev. Fluid Mech.* 40 (2008) 47–70.
- [10] P. Pepiot, O. Desjardins, Numerical analysis of the dynamics of two and three-dimensional fluidized bed reactors using an Euler–Lagrange approach, *Powder Technol.* 220 (2012) 104–121, ISSN 0032-5910 <https://doi.org/10.1016/j.powtec.2011.09.021>.
- [11] J. Capecelatro, O. Desjardins, An Euler–Lagrange strategy for simulating particle-laden flows, *J. Comput. Phys.* 238 (2013) 1–31, ISSN 0021-9991 <https://doi.org/10.1016/j.jcp.2012.12.015>.
- [12] R. Rokkam, A. Sowinski, R. Fox, P. Mehrani, M. Muhle, Computational and experimental study of electrostatics in gas–solid polymerization fluidized beds, *Chem. Eng. Sci.* 92 (2013) 146–156.
- [13] J. Xie, W. Zhong, B. Jin, Y. Shao, Y. Huang, Eulerian–Lagrangian method for three-dimensional simulation of fluidized bed coal gasification, *Adv. Powder Technol.* 24 (1) (2013) 382–392.
- [14] N. Konan, H. Neau, O. Simonin, M. Dupoizat, T. Le Goaziou, 3D unsteady polydispersed simulation of uranium tetrafluoride particles in fluidized bed pilot, *Proc. 20th International Conference On Fluidized Bed Combustion, FBC, 2009, Xian City (China), 2009*.
- [15] P. Fede, H. Neau, O. Simonin, I. Ghouila, 3D Unsteady Numerical Simulation of the Hydrodynamic of a Gas Phase Polymerization Reactor, *Proc. 7th Int. Conference on Multiphase Flow, Tampa (USA), 2010*.
- [16] P. Fede, O. Simonin, I. Ghouila, 3D numerical simulation of polydisperse pressurized gas–solid fluidized bed, *Proc. of AJK2011-FED, ASME-JSME-KSME Joint Fluids Engineering Conference, Hamamatsu, Shizuoka (Japan), 2011*.
- [17] A. Ozel, A. Sarthou, Z. Zeren, P. Fede, O. Simonin, I. Ghouila, J. Chamayou, Numerical simulation of liquid injection into an anisothermal dense fluidized bed, *Proc. 8th International Conference on Multiphase Flow, Jeju (Korea) 2013*.
- [18] Z. Zeren, A. Ozel, A. Sarthou, P. Fede, H. Neau, O. Simonin, J. Chamayou, I. Ghouila, 3D numerical simulation of catalyst injection into a dense fluidized bed, *Proc. 8th International Conference on Multiphase Flow, Jeju (Korea), 2013*.
- [19] F. Fotovat, R. Ansart, M. Hemati, O. Simonin, J. Chaouki, Sand-assisted fluidization of large cylindrical and spherical biomass particles: experiments and simulation, *Chem. Eng. Sci.* 126 (2015) 543–559.
- [20] P. Fede, O. Simonin, A. Ingram, 3D numerical simulation of a lab-scale pressurized dense fluidized bed focussing on the effect of the particle–particle restitution coefficient and particle–wall boundary conditions, *Chem. Eng. Sci.* 142 (2016) 215–235, <https://doi.org/10.1016/j.ces.2015.11.016>.
- [21] L. Bennani, H. Neau, C. Baudry, J. Laviéville, P. Fede, O. Simonin, Numerical simulation of unsteady dense granular flows with rotating geometries, *Chem. Eng. Res. Des.* 120 (2017) 333–347, <https://doi.org/10.1016/j.cherd.2017.01.028>.
- [22] H. Benoit, R. Ansart, H. Neau, P. Garcia Triñanes, G. Flamant, O. Simonin, Three-dimensional numerical simulation of upflow bubbling fluidized bed in opaque tube under high ux solar heating, *AIChE J.* 64 (11) (2018) 3857–3867.
- [23] Y. Igci, A.T. Andrews IV, S. Sundaresan, S. Pannala, T. O'Brien, Filtered two-fluid models for fluidized gas–particle suspensions, *AIChE J.* 54 (2008) 1431–1448.
- [24] J.-F. Parmentier, O. Simonin, O. Delsart, A numerical study of fluidization behavior of Geldart B, A/B and A particles using an Eulerian multifluid modeling approach, *Proc. of the 9th Int. Conference on Circulating Fluidized Beds, Circulating Fluidized Bed Technology IX 2008*, pp. 331–336.
- [25] June 2019 – TOP500 Supercomputer Sites, URL <https://www.top500.org/lists/2019/06/> 2019.
- [26] Z. Hamidouche, E. Masi, P. Fede, R. Ansart, H. Neau, M. Hemati, O. Simonin, Numerical simulation of multiphase reactive flows, in: *Bridging Scales in Modelling and Simulation of Non-Reacting and Reacting Flows. Part I*, Elsevier, 51–124, <https://doi.org/10.1016/bs.ache.2018.01.003>, 2018.
- [27] NEPTUNE CFD Version 4.0.1 User Guide, EDF R&D; Fluid Dynamics, Power Generation and Environment Department Multi-Phase and Reactive Flow Group, 6 quai Watier, 78401 Chatou CEDEX, France, 2017.
- [28] A. Ozel, P. Fede, O. Simonin, Development of filtered Euler–Euler two-phase model for circulating fluidized bed: high resolution simulation, formulation and a priori analyses, *Int. J. Multiphase Flow* 55 (2013) 43–63, ISSN 0301-9322 <https://doi.org/10.1016/j.ijmultiphaseflow.2013.04.002>.

- [29] Y. Igci, S. Sundaresan, Verification of filtered two-fluid models for gas-particle flows in risers, *AIChE J.* 57 (10) (2011) 2691–2707, ISSN 1547-5905 <https://doi.org/10.1002/aic.12486>.
- [30] J.-F. Parmentier, O. Simonin, O. Delsart, A functional subgrid drift velocity model for filtered drag prediction in dense fluidized bed, *AIChE J.* 58 (4) (2012) 1084–1098, <https://doi.org/10.1002/aic.12647>.
- [31] K. Agrawal, W. Holloway, C.C. Milioli, F.E. Milioli, S. Sundaresan, Filtered models for scalar transport in gas-particle flows, *Chem. Eng. Sci.* 95 (0) (2013) 291–300, ISSN 0009-2509 <https://doi.org/10.1016/j.ces.2013.03.017>.
- [32] C.C. Milioli, F.E. Milioli, W. Holloway, K. Agrawal, S. Sundaresan, Filtered two-fluid models of fluidized gas-particle flows: new constitutive relations, *AIChE J.* 59 (9) (2013) 3265–3275, ISSN 1547-5905 <https://doi.org/10.1002/aic.14130>.
- [33] S. Schneiderbauer, S. Puttinger, S. Pirker, Comparative analysis of subgrid drag modifications for dense gas-particle flows in bubbling fluidized beds, *AIChE J.* 59 (11) (2013) 4077–4099, <https://doi.org/10.1002/aic.14155>.
- [34] W. Holloway, S. Sundaresan, Filtered models for bidisperse gas-particle flows, *Chem. Eng. Sci.* 108 (0) (2014) 67–86, ISSN 0009-2509 <https://doi.org/10.1016/j.ces.2013.12.037>.
- [35] S. Schneiderbauer, S. Puttinger, S. Pirker, P. Aguayo, V. Kanellopoulos, CFD modeling and simulation of industrial scale olefin polymerization fluidized bed reactors, *Chem. Eng. J.* 264 (2015) 99–112, <https://doi.org/10.1016/j.cej.2014.11.058>.
- [36] S. Schneiderbauer, A spatially-averaged two-fluid model for dense large-scale gas-solid flows, *AIChE J.* 63 (8) (2017) 3544–3562, ISSN 1547-5905 <https://doi.org/10.1002/aic.15684>.
- [37] S. Cloete, S.T. Johansen, S. Amini, Grid independence behaviour of fluidized bed reactor simulations using the two fluid model: detailed parametric study, *Powder Technol.* 289 (2016) 65–70, <https://doi.org/10.1016/j.powtec.2015.11.011>.
- [38] J.H. Cloete, S. Cloete, F. Mucicchi, S. Radl, S. Amini, The sensitivity of filtered two fluid model to the underlying resolved simulation setup, *Powder Technol.* 316 (2017) 265–277, <https://doi.org/10.1016/j.powtec.2016.11.064>.
- [39] S. Cloete, J.H. Cloete, S. Amini, Hydrodynamic validation study of filtered two fluid models, *Chem. Eng. Sci.* 182 (2018) 93–107, <https://doi.org/10.1016/j.ces.2018.02.032>.
- [40] J.H. Cloete, S. Cloete, S. Radl, S. Amini, On the choice of closure complexity in anisotropic drag closures for filtered two fluid models, *Chem. Eng. Sci.* 207 (2019) 379–396, <https://doi.org/10.1016/j.ces.2019.06.006>.
- [41] S. Morioka, T. Nakajima, Modeling of gas and solid particles 2-phase flow and application to fluidized-bed, *J. Mécanique Théorique et Appliquée* 6 (1) (1987) 77–88.
- [42] O. Simonin, Theoretical and Experimental Modeling of Particulate Flows, Lecture Series 2000–06, von Karman Institute for Fluid Dynamics Rhode Saint Genèse (Belgium): Statistical and Continuum Modelling of Turbulent Reactive Particulate Flows. Part 1: Theoretical Derivation of Dispersed Eulerian Modelling from Probability Density Function Kinetic Equation 2000.
- [43] C. Wen, Y. Yu, Mechanics of fluidization, *Chem. Eng. Symp. Ser.* 62 (1965) 100–111.
- [44] S. Ergun, Fluid flow through packed columns, *Chem. Eng. Prog.* 48 (1952) 89–94.
- [45] O. Simonin, S. Chevrier, F. Audard, P. Fede, Drag force modelling in dilute to dense particle-laden flows with mono-disperse or binary mixture of solid particles, *9th International Conference on Multiphase Flow*, Firenze, Italy, May 22–27, 2016.
- [46] C. Gourdel, O. Simonin, E. Brunier, Two-Maxwellian equilibrium distribution function for the modelling of a binary mixture of particles, in: J. Werther (Ed.), *Circulating Fluidized Bed Technology VI*, DECHEMA, Frankfurt am Main, Germany 1999, pp. 205–210.
- [47] P. Fede, O. Simonin, Application of a perturbed two-maxwellian approach for the modelling of kinetic stress transfer by collision in non-equilibrium binary mixture of inelastic particles, *Proc. 10th Int. Symp. on Gas-Particle Flows*, ASME Fluids Engineering Summer Conference, Houston (USA), 2005.
- [48] L. Zaichik, P. Fede, O. Simonin, V. Alipchenkov, Statistical models for predicting the effect of bidisperse particle collisions on particle velocities and stresses in homogeneous anisotropic turbulent flows, *Int. J. Multiphase Flow* 35 (9) (2009) 868–878, <https://doi.org/10.1016/j.ijmultiphaseflow.2009.05.007>.
- [49] J. Jenkins, M. Richman, Grad's 13-moment system for a dense gas of inelastic spheres, *The Breadth and Depth of Continuum Mechanics*, 647–669, Springer, 1986.
- [50] A. Boëlle, G. Balzer, O. Simonin, Second-order prediction of the particle-phase stress tensor of inelastic spheres in simple shear dense suspensions, *Gas-Particle flows*, ASME FED 28 1995, pp. 9–18.
- [51] P. Fede, O. Simonin, R. Ansart, H. Neau, I. Ghouila, Effect of wall boundary conditions and mesh refinement on the numerical simulation of a pressurized dense fluidized bed for polymerization reactor, *Proc. of the 10th Int. Conference on Circulating Fluidized Beds and Fluidization Technology*, 2011.
- [52] A. Srivastava, S. Sundaresan, Analysis of a frictional kinetic model for gas/particle flows, *Powder Technol.* 129 (2003) 72–85.
- [53] O. Vermorel, B. Bedat, O. Simonin, T. Poinso, Numerical study and modelling of turbulence modulation in a particle laden slab flow, *J. Turbul.* 4 (2019) 025.
- [54] J. Laviéville, E. Deutsch, O. Simonin, Large eddy simulation of interactions between colliding particles and a homogeneous isotropic turbulence field, *ASME Publ. FED* 228 (1995) 347–358.
- [55] W. Ranz, W.R. Marshall, Evaporation from drops, *Chem. Eng. Prog.* 48 (3) (1952) 141–146.
- [56] N. Méchitoua, M. Boucker, J. Laviéville, J. Hérard, S. Pigny, G. Serre, An unstructured finite volume solver for two-phase water/vapor flows modelling based on an elliptic-oriented fractional step method, *Proc. of NURETH-10*, Seoul, Korea, 2003.
- [57] J. Laviéville, E. Quémerais, S. Mimouni, M. Boucker, N. Méchitoua, NEPTUNE CFD V1.0 theory manual, Rapport interne EDF H-181–2006-04377-EN, Rapport NEPTUNE Nept, 2004, L 1 (3).
- [58] Y. Fournier, J. Bonelle, C. Moulinec, Z. Shang, A.G. Sunderland, J.C. Uribe, Optimizing code Saturne computations on petascale systems, *Comput. Fluids* 45 (1) (2011) 103–108, <https://doi.org/10.1016/j.compfluid.2011.01.028>.
- [59] Y. Fournier, J. Bonelle, E. Le Coupance, A. Ribes, B. Lorendeau, C. Moulinec, Recent and upcoming changes in code saturne: computational fluid dynamics HPC tools oriented features, *Fourth International Conference on Parallel, Distributed, Grid and Cloud Computing for Engineering*, Dubrovnik, Croatia, 2015.
- [60] chap. 9 - Ensign Data Formats, Ensign User Manual for Version 10.2, Computational Engineering International, Inc. 2016, pp. 731–950, 2166 N. Salem Street, Suite 101, Apex, NC 27523, USA.
- [61] Y. Fournier, J. Bonelle, P. Vezolle, C. Moulinec, A. Sunderland, An automatic joining mesh approach for computational fluid dynamics to reach a billion cell simulations, in: P. Iványi, B. Topping (Eds.), *Proceedings of the Second International Conference on Parallel, Distributed, Grid and Cloud Computing for Engineering*, Civil-Comp Press, 2011 <https://doi.org/10.4203/ccp.95.63>, paper 63.
- [62] J. Ahrens, B. Geveci, C. Law, ParaView: an end-user tool for large data visualization, *Visualization Handbook*, Elsevier, 2005, ISBN 978-0123875822.
- [63] A. Ribés, B. Lorendeau, J. Jomier, Y. Fournier, In-situ visualization in computational fluid dynamics using open-source tools: integration of catalyst into code saturne, in: J. Bennett, F. Vivodtzev, V. Pascucci (Eds.), *Topological and Statistical Methods for Complex Data*, Springer Berlin Heidelberg, Berlin, Heidelberg 2015, pp. 21–37, ISBN 978-3-662-44900-4.
- [64] N. Méchitoua, Y. Fournier, F. Hülsemann, Improvements of a finite volume based multigrid method applied to elliptic problems, *International Conference on Mathematics, Computational Methods and Reactor Physics (M&C'09)*, American Nuclear Society - ANS, Saratoga Springs, 2009, ISBN 20978-0-89448-069-0.
- [65] H. Neau, P. Fede, J. Laviéville, O. Simonin, High performance computing (HPC) for the fluidization of particle-laden reactive flows, *Proc. 7th International Conference on Multiphase Flow*, 2010.
- [66] Code Saturne Version 5.0.4 User Guide, EDF R&D; Fluid Dynamics, Power Generation and Environment Department Multi-Phase and Reactive Flow Group, 6 quai Watier, 78401 Chatou CEDEX, France, 2017.
- [67] D. SAS, SIMAIL version 8.0 manuel utilisateur, DISTENE, Campus Teratec, 2 rue de la Piquetterie, F-91680 Bruyères-le-Châtel, FRANCE, 2015.
- [68] F. Pellegrini, PT-Scotch version 6.0.4 user guide, Université Bordeaux 1 & LaBRI, UMR CNRS 5800 Bacchus team, INRIA Bordeaux Sud-Ouest, 351 cours de la Libération, 33405 Talence, France, 2014.
- [69] G. Karypis, K. Schloegel, ParMETIS Version 4.0.2 Manual, University of Minnesota, Department of Computer Science and Engineering, Minneapolis, MN, 2013 55455.
- [70] K. Agrawal, P. Loezos, M. Syamlal, S. Sundaresan, The role of mesoscale structures in rapid gas-solid flows, *J. Fluid Mech.* 445 (2001) 151–185.
- [71] M. Germano, Turbulence: the filtering approach, *J. Fluid Mech.* 238 (1992) 325–336.
- [72] A.C. Bauer, H. Abbasi, J. Ahrens, H. Childs, B. Geveci, S. Klasky, K. Moreland, P. O'Leary, V. Vishwanath, B. Whitlock, E.W. Bethel, In Situ Methods, infrastructures, and applications on high performance computing platforms, *Computer Graphics Forum* 35 (3) (2016) 577–597, <https://doi.org/10.1111/cgf.12930>.
- [73] W.-C. Feng, K. Cameron, The Green500 list: encouraging sustainable supercomputing, *Computer* 40 (12) (2007) 50–55, <https://doi.org/10.1109/mc.2007.445>.