Particle agglomeration in flows: fast data-driven spatial decomposition algorithm for CFD simulations

Kerlyns Martínez Rodríguez, Mireille Bossy, Christophe Henry

Particle agglomeration in flows: fast data-driven spatial decomposition algorithm for CFD simulations

Kerlyns Martínez Rodríguez[a,b,*], Mireille Bossy[a], Christophe Henry[a]

[a] *Université Côte d'Azur, Inria, CNRS, Sophia Antipolis, France, CaliSto Laboratory*
[b] *Institute of Statistics, University of Valparaíso, Chile.*

Abstract

Computational fluid dynamics simulations in practical industrial/environmental cases often involve non-homogeneous concentrations of particles. In Euler-Lagrange simulations, this can induce the propagation of numerical error when the number of collision/agglomeration events is computed using mean-field approaches. In fact, mean-field statistical collision models allow to sample the number of collision events using a priori information on the frequency of collisions (the collision kernel). Yet, since such methods often rely on the mesh used for the Eulerian simulation of the fluid phase, the particle number concentration within a given cell might not be homogeneous, leading to numerical errors. In this article, we apply the data-driven spatial decomposition (D2SD) algorithm to control such error in simulations of particle agglomeration. Significant improvements are made to design a fast D2SD version, minimizing the additional computational cost by developing re-meshing criteria. Through the application to some practical simulation cases, we show the importance of splitting the domain when computing agglomeration events in Euler/Lagrange simulations, so that within each elementary cell there is a spatially uniform distribution of particles.

Keywords: Agglomeration, Particle-laden flows, Population Balance Equation (PBE), Mesh-independence, Particle-based mesh, Computational Fluid Dynamics (CFD)

1. Introduction

1.1. General context: particle agglomeration

Particle agglomeration is the process whereby solid particles or liquid droplets dispersed in a flow adhere together to form larger structures. This process can also be referred to as aggregation [1, 2] or as coalescence for droplets. The agglomeration process is at play in a range of fields belonging to both natural and applied sciences. For instance, in natural sciences, agglomeration occurs in geo-morphology [3] (e.g. river delta formation), meteorology [4, 5, 6] (e.g. droplet growth in clouds) or astrophysics [7] (e.g. planetoids growth). In applied sciences, agglomeration plays a major role in a number of industrial applications including in waste-water treatment facilities [8, 9], in combustion systems [10, 11], in oil/sludge refining [12] and food industry [13].

From this brief overview of applications, it appears that the process of agglomeration involves a variety of objects (molecules, polymers, solids, droplets, bubbles, etc.) and covers a wide range of temporal and spatial scales (from molecular scales with proteins up to astrophysical scales with planetoids). Depending on the field, this process is also referred to as aggregation (as for solid materials), flocculation (e.g. polymers), coalescence (such as droplets/bubbles) or coagulation (e.g. non-Newtonian fluids).

In this paper, we focus on the prediction of agglomeration of solid particles in the colloidal range (i.e. with a size ranging from a few nanometers up to a few micrometers) that are suspended in a flow. Nevertheless, most of the developments and examples provided in this paper remain relevant for droplets or other objects.

1.2. Modelling of particle agglomeration: existing approaches and limitations

Given the importance of the agglomeration process in many natural and industrial applications, accurate predictions are paramount. This led to the development of a range of models to simulate particle agglomeration. The difficulty that arises is that such simulations require to couple algorithms to solve the fluid motion (e.g. [14]), numerical models for the transport of particles by this flow (see e.g. [15, 16]) as well as algorithms to detect collisions between particles and treat the outcome of such collisions (see e.g. [12, 17]). This brings about multidisciplinary issues related to turbulence, multiphase flow as well as physico-chemical sciences (for the interactions between particles). In the following, some of the existing algorithms are summarised, sorted according to their level of description, starting from microscopic approaches up to macroscopic approaches (see also [17]):

Tracking of n-particles with direct collision detection and treatment. These approaches are based on a microscopic level of

*Corresponding author
Email addresses: kerlyns.martinez-rodriguez@inria.fr (Kerlyns Martínez Rodríguez), mireille.bossy@inria.fr (Mireille Bossy), christophe.henry@inria.fr (Christophe Henry)

description. They usually consist in a coupling of four modelling approaches: (i) a fine-scale simulation of the fluid phase (e.g. with Direct Numerical Simulation DNS); (ii) a fully resolved Lagrangian tracking of the motion of n particles (e.g. explicit solver for translational and rotational equations of motion); (iii) a direct collision-detection algorithm (e.g. with an overlap criterion [18] or with geometric criteria [19]) and (iv) specific scenarios for the outcome of the collision (e.g. relying on energy-based or momentum-based agglomeration models to distinguish between sticking, sliding or rebound collisions [20, 21]).

Such approaches allow to obtain the number of collision events as a direct result of the simulation while providing detailed information on aggregate properties (e.g. shape of agglomerates, spatial/temporal correlations between collisions) [22]. However, they suffer from their high computational costs: the direct collision-detection algorithm (also called fully deterministic approach in [17]) requires $n_p \times (n_p - 1)/2$ examinations of particle pairs that are potential candidates for a collision, where n_p is the number of particles in the volume considered (note that n_p can actually be a fraction of the total number of particles n if a spatial division is used to improve the algorithm efficiency). For that reason, such approaches have been applied only in simple idealised cases.

One-particle (Lagrangian) tracking with given collision frequencies and efficiencies. These methods have a "coarser" level of description. They usually consist in a coupling of four approaches: (i) an Eulerian simulation of a flow (e.g. using RANS models when dealing with turbulent flows); (ii) a one-particle PDF Lagrangian model for the motion of particles [16, 23, 24, 25] (based on stochastic equations); (iii) a detection model for particle collisions (e.g. a probabilistic collision algorithm [17, 26, 27, 28, 29, 30] or a stochastic-deterministic algorithm [17, 31, 32, 33]) and (iv) the use of a collision efficiency to evaluate the proportion of collisions that actually lead to the formation of aggregates. These approaches are thus based on a Bird-like algorithm [34, 35, 36], which relies on a separate treatment of the transport step (transport without interaction) and of the collision step (interaction without transport).

It should be noted here that, in order to avoid the high computational costs associated with the tracking of all real particles, the notion of parcel is often used: one "numerical" parcel is a statistical representation of a large number of real particles. In that case, the collision-detection algorithm is adapted for parcels. Similar formulations have been used recently, replacing the probability of collisions based on the relative motion of pairs of particles/parcels with a probability of collisions evaluated with a mean-field approach [37]. It thus requires a priori information on the collision frequency and collision efficiency (also used in PBE formulations). Besides, other inter-particle collision models for hybrid Euler/Lagrange approaches rely on fictitious collision partners and on kinetic theory for the evaluation of the inter-particle collision probability (thus removing the need to have precise information on the location of surrounding particles) [17, 26, 27].

Population Balance Equations (PBE). PBE are macroscopic approaches that allow to perform fast evaluations of the kinetics of particle agglomeration at large-scale. They describe the evolution in time of mean-field quantities. More precisely, these equations describe the rate of variation of the number density of a population of particles due to agglomeration within a control volume $\mathbb{V}_C$. As particles agglomerate, the distribution of sizes changes according to the following integro-differential equation [38, 39]:

$$\frac{\partial n}{\partial t}(\mathbb{v}, t) = \frac{1}{2}\int_0^{\mathbb{v}} \alpha_{\mathbb{v}-\mathbb{w},\mathbb{w}}\, \beta_{\mathbb{v}-\mathbb{w},\mathbb{w}}\, n(\mathbb{v}-\mathbb{w}, t) n(\mathbb{w}, t) d\mathbb{w} - \int_0^{\infty} \alpha_{\mathbb{v},\mathbb{w}}\, \beta_{\mathbb{v},\mathbb{w}}\, n(\mathbb{v}) n(\mathbb{w}) d\mathbb{w}, \tag{1}$$

where $n(\mathbb{v}, t)$ is the number density of particles of volume $\mathbb{v}$ at time t within the control volume $\mathbb{V}_C$, α is the collision efficiency and $\beta_{\mathbb{v},\mathbb{w}}$ the collision frequency kernel between particles of volume $\mathbb{v}$ and $\mathbb{w}$. The first term on the r.h.s. of Eq. (1) corresponds to the formation of new aggregates of volume $\mathbb{v}$ produced by the collision between pairs of smaller aggregates (such that the sum of their volumes equals $\mathbb{v}$) while the second term on the r.h.s. accounts for the loss of particles of volume $\mathbb{v}$ due to their agglomeration with other particles.

PBE approaches are compatible with the level of description used in standard Computational Fluid Dynamics (CFD) simulations [40, 39] (e.g. Euler-Euler simulations with turbulence models). In particular, a large amount of studies rely on the coupling between standard CFD approaches (e.g. Multiple Size Group model named "MUSIG" or Quadrature-Based Moment Methods named "QMOM") and PBE models to compute the agglomeration. Such approaches have been applied to both homogeneous or inhomogeneous fluid flows [41, 42, 43, 44]. This is due to the fact that CFD computations of the fluid flow naturally handle inhomogeneities in the flow quantities (e.g. velocity) and their effect on the transport of the particles [45, 46, 44]. Such methods can also handle spatially-dependent values of the collision kernel [47].

However, these approaches are based on PBE formulations for particle agglomeration. Hence, they suffer from the limitations of PBE formulations that have been well identified in the literature [39, 48]. This includes the fact that the collision frequency and the collision efficiency have to be provided. At the moment, analytical formulas for α and β are only available in simple situations (e.g. collision due to Brownian motion only, gravity only, shear only).

In the present paper, we focus on hybrid approaches which combine Eulerian simulations of the fluid phase, one-point Lagrangian tracking of particles and a mean-field collision model for agglomeration (see e.g. [37]). This coupling of several approaches induces several consequences when computing particle agglomeration.

a. First, some limitations are actually inherent to the use of a mean-field approach to compute particle agglomeration (as in PBE formulations). In particular, a mean-field approach implicitly assumes that there is no correlation between individual particles [39]. Although such approaches provide

proper information on the mean quantities, recent studies of particle agglomeration have shown that correlations between successive collisions explains the abrupt growth of aggregates observed locally in high vorticity regions of turbulent flows [22]. To remove this limitation, higher-order models are necessary. This is left out of the scope of the present paper.

b. Another consequence of the mean-field approach is that it requires prior knowledge of the collision kernel and collision efficiency. As mentioned before, analytical formulas are available only in simple cases [39]. These analytical formulas have been obtained considering only binary collisions and that the fluid properties entering α and β are constant within each control volume (i.e. cell here) during the time step considered. Such assumptions can be lifted, but it remains to be seen if analytical formulas can still be obtained then. Regardless of these assumptions, the values of α and β can also be fixed with empirical formulas drawn from experimental observations (when possible).

c. The main issue when coupling Lagrangian tracking of particles to mean-field approaches for particle agglomeration is actually related to the need of having a uniform distribution of particles within each control volume $\mathbb{V}_C$ considered (i.e. in each cell of the mesh). This is not an issue in Euler-Euler simulations since the only information available is the particle number concentration in each cell (meaning that particles are implicitly homogeneously distributed in each cell). However, in hybrid Euler-Lagrange approaches, the discrete particle positions mean that this homogeneous distribution might not be respected when using the same mesh as the one used for the Eulerian simulation of the fluid flow.

In this paper, we focus on the uniform distribution assumption, later referred to as the spatially-uniform condition. It means that no information on the particle coordinates is required in the collision step [39]. In practice, this spatially-uniform condition must be fulfilled locally during a time step in each volume element (i.e. computational cell in CFD simulations). However, this assumption does not necessarily hold in Lagrangian simulations of complex flows, where particles can be injected locally in the system or accumulate in specific regions, thus leading to significant gradients of particle concentration at the system scale [32]. As an immediate consequence, numerical errors are quickly propagated in the simulation [49, 50], which result in a need for prior mesh-dependency analysis dedicated to the collision step. In this context, spatial-decomposition algorithms can prove useful in any Euler-Lagrange approach. In fact, the use of two distinct meshes, one for the fluid phase and another one for the dispersed particle phase, is also adapted when computing statistical information on the particle phase. In that case, a mesh adapted to the particle positions will ensure more accurate statistical information on particles (e.g. mean particle concentration, mean particle velocities, etc.).

A variety of attempts have been made to reduce the mesh-dependency of the agglomeration results from hybrid Euler/Lagrange approaches [51, 52, 53, 54, 55]. Initially, the proposed methods focused on computing the agglomeration over a volume comprising the cell (within the mesh used for the CFD simulations) and its neighbours. Other proposed methods generate a specific mesh for the computation of the particle agglomeration (e.g. the No-Time Counter algorithm [51]) by randomly choosing the partitioning orientation, every time step, before splitting the domain, or by introducing an adaptive collision mesh method. More recently, the notion of collision cells has been introduced and used to keep the collision frequency independent of a computational grid [56, 57, 58]: it corresponds to a local spherical cell in which a sufficient number of parcels are present to ensure a low statistical error in the computations. The main drawback of these approaches lies in the fact that they do not rely on precise information coming from the spatial distribution of the particles/parcels. More recently, a new data-driven spatial decomposition (D2SD) algorithm has been proposed to detect non-homogeneous concentrations in a set of particles (point data) within a regular volume [50]. This D2SD algorithm provides a spatial splitting according to the spatial distribution of particles. More precisely, the D2SD algorithm uses as an input data only the information on the location of the center of gravity of each particle. One of the many advantages of the D2SD algorithm is that the parameters leading to the optimal domain decomposition are automatically tuned through the statistical information coming from the data (position of particles). Thus, there is no bias coming from the choice of arbitrary parameter. The algorithm is coupled with a battery of uniformity tests to check whether the input data and decomposed output data satisfy the spatially-uniform condition. Then, the output data are classified in regions according to the number density of particles within it. This means that the particle number concentration is homogeneous in each region of space obtained from the D2SD algorithm. This classification of regions according to the number density provides a spatial decomposition of the domain.

1.3. Objectives of the paper

The present article is a follow-up of the proposed D2SD algorithm [50] that detects the presence of non-uniform spatial distributions of point particles and that comes up with an optimal spatial decomposition respecting locally the uniform distribution. The previous paper [50] was focussed on assessing the robustness and accuracy of the algorithm. Furthermore, the importance of applying the method when computing particle agglomeration in complex 3D industrial/environmental cases has been illustrated in a simple case, and the application of the D2SD algorithm to CFD simulations was left out.

The objective of this paper is thus to adapt the D2SD algorithm in order to respect requirements for standard CFD codes and to illustrate the impact of its use in practical 3D cases. More precisely, this article proposes numerical schemes for an efficient use of D2SD, decreasing the computational cost of D2SD or/and the number of times the algorithm is called during the simulation. Several options are assessed, introducing a criterion to avoid applying the full version of the D2SD algorithm every time step, or simplifying uniformity tests. The main difficulty is to ensure that the adapted algorithm keeps an appropriate balance between its accuracy and its computational costs.

Beyond a particular application, the methodology adopted in [50], and the present article, can be easily adapted to other contexts where a decomposition of the space respecting the spatial homogeneity of a certain variable is aimed (e.g. to compute particle statistics in hybrid Euler/Lagrange approaches). Hence, we want to promote the proposed methodology and its usefulness in more general applications through the analysis of statistical information provided by the data.

1.4. Layout of the paper

To address the above objectives, the paper is organised as follows. First, the full D2SD algorithm is briefly recalled in Section 2.1. Faster versions of the D2SD algorithm to fit the computational efficiency and costs of CFD simulations are detailed in Section 2.2. These faster versions are analysed in terms of accuracy and computational efficiency on a series of numerical experiments that are presented in Section 3. Finally, numerical simulations are performed in practical 3D cases. The results are analysed in Section 4, with a specific emphasis on the accuracy of the results obtained.

2. Adaptation of the D2SD algorithm to CFD simulations

The input information of the D2SD algorithm is a set of positions of the center of gravity of n particles, $\mathbf{X} = \{X^1, X^2, \dots, X^n\}$, in a bounded spatial domain $\mathcal{D}$. The D2SD algorithm detects deviations from the uniformity of the spatial distribution of particle positions, and generates a Cartesian partition of $\mathcal{D}$, satisfying the uniform distribution condition for the point cloud contained in each cell of the partition. In particular, the domain partition identifies each cell according to eight different levels of concentration, which allows one to quantify the difference in the amount of particles in each region of $\mathcal{D}$ and to locate clusters of particles.

For the sake of clarity and self-containment, the D2SD algorithm [50] is briefly recalled in the following. To simplify the presentation of the method and its adaptation, the $\mathcal{D}$ domain is embedded and normalized as the unit cube $\mathcal{D} = [0,1]^d$, with d being the dimension.

2.1. Principle of the D2SD algorithm

The method is presented in the case $d = 3$, its adaptation to cases $d = 1, 2$ being straightforward. The D2S2 algorithm relies on three main ingredients.

► The first one is the histogram estimation of the probability density function (PDF) associated with the set of particle positions $\mathbf{X}$. In order to extract information from this PDF without introducing any external numerical parameter, the choice is made to use the Freedman and Diaconis rule [59] for sizing the histogram bin box $h_1 \times h_2 \times h_3$. This rule ensures a good balance between robustness and computational complexity: in each direction $j = 1, 2, 3$,

$$h_j(\mathbf{X}) = 2\frac{\texttt{iqr}(\mathbf{X}_j)}{n^{3/2}}, \tag{2}$$

where the interquartile range $\texttt{iqr}(\mathbf{X}_j)$ of the sample $(\mathbf{X}_j)$ gives the size of the central sub part of $[0,1]$ containing 50% of the $(\mathbf{X}_j)$ population. With this bin volume size $h_1(\mathbf{X}) \times h_2(\mathbf{X}) \times h_3(\mathbf{X})$, a first Cartesian grid is constructed on $\mathcal{D}$, with a bin number #bins on $[0,1]^3$. This grid is the skeleton of the domain decomposition of the D2SD partition.

The approximate density (i.e. concentration) is computed for each bin $i \in \{1, 2, \dots, \#\text{bins}\}$, with:

$$\texttt{PDF}(i) = \frac{\sum_{j=1}^{n} \mathbf{1}_{X^j \in \texttt{Bin}(i)}}{n\, h_1(\mathbf{X}) h_2(\mathbf{X}) h_3(\mathbf{X})}. \tag{3}$$

Naturally, the values $\texttt{PDF}(i)$ of the empirical PDF will vary (even starting from a uniformly distributed sample).

► The second main ingredient of D2S2 is a classification criterion of bins according to concentration levels. For this purpose, a suitable choice of a threshold λ is introduced, separating the PDF values (and therefore the domain) into high-concentration values and low-concentration values. The area in between is the ambient (or medium) concentration. Particles are well dispersed in this zone and their positions are uniformly distributed.

To avoid the use of arbitrary parameters, percentile indicators are used as a measure of dispersion of the PDF-values themselves. The 1d-distribution of the PDF-values (called here reduced-PDF) together with its percentiles are computed : Q_x is the level for which x% of the values in $\{\texttt{PDF}(i); i = 1, \dots, \#\text{bins}\}$ are below Q_x. Given that we are targeting to separate high and low particle concentrations, the suitable value for the threshold λ will be found in $[Q_{60}, Q_{100}]$. A region with a low number of particles, as well as a region of 'uniform ambient' concentration, can be identified as follows:

- bins with zero as PDF values are empty
- bins with PDF values below Q_{20} are "under concentrated"
- bins with PDF values in $[Q_{20}, \lambda]$ are "ambient uniform".

The threshold λ plays the role of a contrast measure between low and high concentration. But it is noteworthy that the higher the concentration, the more this contrast must be refined. Conversely, a statistical test on the uniformity of the distribution of positions is less rejected on a low concentration (or small sample size). Thus an optimal threshold λ^* is computed, minimizing a score function (detailed in Appendix B) that quantifies how far a sample X is from being uniformly distributed.

Once λ^* is determined, five regions with a high-concentration of particles can be identified, by computing the percentiles $\{\widetilde{Q}_{20}, \widetilde{Q}_{40}, \widetilde{Q}_{60}, \widetilde{Q}_{80}\}$ of the truncated reduced-PDF $\{\texttt{PDF}(i); i = 1, \dots, \#\text{bins}, \texttt{PDF}(i) > \lambda^*\}$:

- bins with PDF values in $[\widetilde{Q}_{20y}, \widetilde{Q}_{20(y+1)}]$ are of "density level#y", $y \in \{0, 1, 2, 3, 4\}$,

defining the eight density level regions illustrated in Figure 2 and the following ones.

► The third main ingredient of D2S2 is the computation of λ^*as an optimum defined in (B.2). Browsing through all the discrete possible values of λ in the range $[Q_{60}, Q_{80}]$, a first λ_1^* is determined with the minimal score function: for each candidate λ, a synthetic sample $\mathbf{X}^\lambda$ is introduced as follows.

Particles within the bins previously identified as low- or high-particle concentration regions (with threshold λ) are temporarily replaced by synthetic uniformly distributed particles (also called *ghost particles*). The number of ghost particles within a cell is determined using the cell-volume and ambient concentration level (depending on λ), the ambient region stays unchanged. The closer λ is to the optimal value, the closer the new sample $\mathbf{X}^{\lambda}$ (consisting of the ambient particles plus the ghosts) would be to a uniform distribution. Thus, ideally, the sample $\mathbf{X}^{\lambda_1^*}$ would be uniformly distributed.

The next step of the optimization involves checking whether $\mathbf{X}^{\lambda_1^*}$ passes a statistical uniformity test (see Appendix A). If the test is accepted, the concentration levels are determined with $\lambda^* = \lambda_1^*$, and the bin cells are merged according to them producing the final partition. If the uniformity test is rejected, it means that some clusters or voids were missed in the process. Based on the sample $\mathbf{X}^{\lambda_1^*}$, synthetic PDF values $\{\texttt{PDF}^1(i); i = 1, \ldots, \#\text{bins}\}$ are recomputed, keeping unchanged the bin size computed in (2):

$$\texttt{PDF}^1(i) = \frac{\sum_{j=1}^{n} \mathbf{1}_{X^{\lambda_1^* j} \in \texttt{Bin}(i)}}{n\, h_1(\mathbf{X}) h_2(\mathbf{X}) h_3(\mathbf{X})}.$$

A search of a λ_2^* is done minimizing again the score of synthetic sample among the candidates in the $\texttt{PDF}^1$ values in $[Q^1_{60}, Q^1_{80}]$. The idea behind this iterative search is to increase the contrast in the right hand side of the medium zone concentration. The search of a $\lambda^* = \lambda^*_{\text{final}}$ stops when the a posteriori uniformity is accepted on the sample $\mathbf{X}^{\lambda^*_{\text{final}}}$.

The statistical uniformity test is also used at the very first step on the sample $\mathbf{X}$. The D2SD algorithm is started only if this first uniformity test is rejected.

The main steps of D2SD can be summarised as follows (see also Figure B.19):

`step1` applies a battery of statistical tests to decide if the input sample $\mathbf{X}$ is uniformly distributed. In case of acceptance, there is no need for a specific domain partition. In case of rejection, the next step is applied.

`step2` aims to search for an optimal spatial decomposition that respects the local-uniform condition by constructing the PDF approximation and computing the optimal threshold allowing to separate low and high concentrations.

`step3` applies a battery of statistical tests to check whether the last computed spatial decomposition satisfies the spatially-uniform condition. If the test-battery is accepted, D2SD is ended. If it is rejected, the algorithm goes back to `step2` with the corresponding synthetic particle sample.

Once the required steps of the D2SD algorithm have been completed, the output is a spatial decomposition distinguishing between areas of zero concentration, low concentration, medium concentration and high concentration. As required by mean-field statistical approaches, each bin-cell defined with the D2SD algorithm respects the local spatially-uniform condition.

2.2. *Fast version of the D2SD for CFD purposes*

As mentioned in the introduction, some existing CFD software relies on Lagrangian methods for particle tracking coupled to mean-field statistical algorithms for particle agglomeration. Yet, the use of a mean-field approach requires a uniform particle distribution in each volume considered. This means that, in each cell within the mesh and for a given time step, the uniform distribution condition has to be met. In the case of non-homogeneous flows, the value of the fluid properties and particle concentration can vary significantly within the whole domain $\mathcal{D}$ (i.e. from one cell to another). For that reason, the D2SD algorithm appears as a suitable candidate to create a new mesh that meets this condition. This new mesh is thus only related to the position of particles in the whole domain $\mathcal{D}$ and is independent from the mesh used for the fluid simulation. It is only used during the agglomeration step and has no influence on particle transport (which, if necessary, can be still carried out using the mesh used for the fluid simulation). The D2SD algorithm has been shown to properly and accurately capture low, mild, or high non-uniformities in the particle concentration (either void regions or over-concentration [50]). The accuracy of the algorithm has been characterised in a range of 2D and 3D numerical experiments that are representative of realistic situations. When using the D2SD algorithm with mean-field approaches for particle agglomeration, it has also been shown that better results are obtained compared to an arbitrary choice of mesh.

However, these tests have shown that the numerical costs associated with the use of the original D2SD algorithm, hereinafter referred to as `full_D2SD`, can dramatically increase when a large number of particles n is involved. This is mostly related to the numerical costs associated with the uniformity verification steps (`steps1,3`) and the minimization problem in `step2`. To take all the benefit of using the D2SD algorithm in a CFD software, it is necessary to reduce the numerical costs associated with this spatial decomposition since it is expected to be applied every time step (i.e. using the information on particle positions at time t). The solely -but key- issue when designing a fast D2SD algorithm is to avoid losing too much accuracy. For that reason, in the rest of the paper, we will often refer to efficiency measured as the ratio between the computational costs and the accuracy of the algorithm proposed.

Drawing on these computational issues, the aim of this section is to present some decision/selection criteria offering a reduced version of the D2SD algorithm. In particular, we assess here three levels of simplification that are summarised in the following:

`Algo1` Full D2SD with re-meshing criterion.

The idea would be to update the spatial decomposition only when deemed necessary, using the `full_D2SD` algorithm. This implies first to develop a criterion to decide when D2SD needs to be applied again, i.e. when the spatial decomposition with particle positions $\mathbf{X}_t$ at time t becomes too different from the spatial decomposition applied to the sample $\mathbf{X}_{t-s}$ at the time distance s. The criterion is composed of two sub-criteria (see Section 2.2.2 below), one

based on the fluctuation of the dispersion of the particles within the domain $\mathcal{D}$, and the other based on the variation of the particle concentration between regions.

`Algo2` Quick D2SD, no re-meshing criterion.

This option consists of two modifications. First, the uniformity tests in `step1&3` are limited to sample size smaller than a priori value n_{crit}, a critical sample size that can be set according to a desired error level ϵ. The aim of this modification is to lower the computational cost due to uniformity checks. The tolerance level can be chosen knowing that the larger the number n of particles, the lower the relative numerical error within the computation of agglomeration events. In contrast, when n is small, introducing a partition of $\mathcal{D}$ whereas the particles are uniformly distributed will immediately bring a numerical error, and this effect increases when the number of particles decreases. Eliminating this verification must therefore be limited to large sets of particles ($> n_{crit}$). For the case of a uniformly distributed population of particles, it has been numerically shown in [50, Fig. 11] that a relative error of $\epsilon = 10^{-1}$ is obtained by removing `step1,3` with a set of $n_{crit} = 100$ particles, against a relative error of 10^{-2} by applying the `full_D2SD` algorithm. The modified `step1&3` are denoted $\texttt{step1}|_{n_{crit}}, \texttt{3}|_{n_{crit}}$.

Second, the spatial decomposition algorithm in `step2` is applied using chosen a priori information, speeding up the process by simplifying the iterative search of an optimal threshold in `step2`. More precisely, the idea is to choose an a priori value $\hat{\lambda}$ in place of the optimal value λ^*. With this, the D2SD algorithm reduces to implementing the `steps 2` with a fixed $\hat{\lambda}$, together with $\texttt{step 1}|_{n_{crit}}, \texttt{3}|_{n_{crit}}$. The choice for $\hat{\lambda}$ is discussed in Section 2.2.1.

`Algo3` Quick D2SD with re-meshing criterion.

This version combines the two previous options, i.e. a re-meshing criterion is used to avoid applying the spatial decomposition algorithm every time step while using a simplified decomposition with an a priori evaluation of the PDF-threshold $\hat{\lambda}$.

These three levels of simplification to speed-up the computation of the spatial decomposition rely on two main notions: the `quick_D2SD` algorithm and the re-meshing criteria that we detail in the following.

2.2.1. *Quick D2SD: a faster computation of spatial decomposition*

The `quick_D2SD` algorithm requires an "as good as possible" guess value $\hat{\lambda}$ instead of the optimal one λ^*. Within the procedures considered in `step2`, the reduced-PDF (required to classify the regions by levels of concentration) is computed and provides statistical information in order to propose an a priori threshold. In this context, since we know that the optimal threshold is searched within the set $[Q_{60}, Q_{80}] \cap \cup_i \texttt{PDF}(i)$, we choose here

$$\hat{\lambda} := Q_{60}. \tag{4}$$

With this value, the algorithm will continue to classify 5 different concentration levels in $[\hat{\lambda}, \max_i \texttt{PDF}(i)]$. By underestimating λ^* with $\hat{\lambda}$, the decomposition will induce a higher numerical error. Nonetheless, since the 5 high-concentration levels are computed from the percentiles associated with $[\hat{\lambda}, \max_i \texttt{PDF}(i)] \cap \cup_i \texttt{PDF}(i)$, this numerical error will be possibly distributed on the 5 high-concentration levels instead of just one (that is exactly what would happen if $\hat{\lambda}$ overestimates λ^*). This choice will be assessed later in Section 3.

2.2.2. *Re-meshing criterion*

In simulations of particle agglomeration with Euler-Lagrange approaches, the local particle concentration in each cell can vary in time due to three effects: (i) the motion of particles can lead to their departure from the current cell and entrance in another one; (ii) the agglomeration/fragmentation of particles will change the total number of particles in a given cell; (iii) new particles can be created due to source terms or inlet boundaries while particles can leave the domain due to sink terms or outlet boundaries. All these three aspects also play a role in the transport step. However, we focus here on the collision step in a Bird-like approach. Hence, the aim is to evaluate how much the number concentration changes with time. Since the last two possibilities can be related to each other, we propose here a re-meshing criterion based on two separate criteria: a first one on the dispersion of particles already present in the domain $\mathcal{D}$ and a second one on the variation in the particle concentration due to appearance/disappearance of particles.

In both cases, we consider the input pair $(\mathbf{X}_k, \mathtt{M}_k)$, with $\mathbf{X}_k$ the set of particle positions and $\mathtt{M}_k$ its corresponding spatial decomposition at a given time-iteration k of the simulation. Before formalising the re-meshing criterion, we describe below the principle of the parts that compose it.

• *Criterion on particle dispersion fluctuation.*

In order to decide whether a new spatial decomposition is required, the key information is actually a measure of the scattering/gathering fluctuation due to the motion of particles. For that purpose, we resort here to the bin size computed in (2) with $\mathbf{X}_t$ since it is directly related to the information about how the particles are dispersed in each direction throughout their interquartile range. The fluctuations in the particle scattering/gathering between time steps are evaluated by comparing directly the respective number of bins.

Denote $\texttt{N\_bins}(\mathbf{X}_t)$ as the number of bins estimated for the set of positions $\mathbf{X}_t$. If a cluster spreads, the interquartile range value will increase (i.e. the bin size will increase) so the total number of bins will decrease. Conversely, if the cluster collapses, the total number of bins will increase between time-iterations. With this in mind, the idea is to quantify the percentage of lost/won bins between time steps and to use this value as an indicator of the fluctuations in particle dispersion. More precisely, at time $t + \Delta t$, we compute the new bin size with (2), and we evaluate

$$\Delta_t \texttt{N\_bins} = \frac{(\texttt{N\_bins}(\mathbf{X}_{t+\Delta t}) - \texttt{N\_bins}(\mathbf{X}_t))}{\texttt{N\_bins}(\mathbf{X}_t)} \%. \tag{5}$$

Positive (resp. negative) values of Δ_tN_bins represents the percentage of bins won (resp. lost) between the mesh at time $t + \Delta t$ and the mesh at time t. Clearly, if $|\Delta_t$N_bins$| = 0$ or small enough, there is no significant change in the particle global dispersion.

Hence, the criterion on particle dispersion consists in applying the D2SD algorithm at $t + \Delta t$ only if $|\Delta_t$N_bins$| > \epsilon_1\%$, where ϵ_1 is a given tolerance threshold. Otherwise, if $|\Delta_t$N_bins$| \leq \epsilon_1\%$, the spatial decomposition is not changed (i.e. $M_{t+\Delta t} = M_t$).

At this point, it is worth noting that, although the present criterion does depend on the particle displacement (local concentrations vary when particles change cells), it is different from a particle-in-cell CFL condition. The particle-mesh (or particle-in-cell) CFL condition is indeed a usual criterion in hybrid Euler-Lagrange approaches for CFD simulations [60, 61]. It is applied to ensure a proper convergence of the solver for the dynamics of particles, imposing Δt to not move particles more than one cell during a time step (CFL< 1). Here, the criterion on particle dispersion fluctuation is used to determine the frequency of re-meshing for the computation of agglomeration. The criterion rather takes into account concentration fluctuations due to particle displacement (and thus to fluxes). Hence, the criterion does not impose particles to avoid moving particles more than one cell during a time step. However, if CFL< 1, concentration fluctuations are expected to be relatively low statistically speaking (whereas fluctuations can take any value when CFL> 1).

• *Criterion on concentration variation.*

This second indicator measures how much particle concentration changes between two time steps due to sources/sinks/agglomeration events.

Denote $\mathcal{L}$ the set of 8 different levels of concentration discussed in step2, and $\mathcal{D}_\ell$ the region associated with the level ℓ of concentration. We define a probability measure $\mathbb{P}$ on $\mathcal{L}$ as follows:

$$\forall\, h \in \mathcal{L},\ \mathbb{P}(h) = \sum_{\ell \in \mathcal{L}} \frac{\mathtt{Vol}(\mathcal{D}_\ell)}{\mathtt{Vol}(\mathcal{D})} \mathbf{1}_{\{h=\ell\}}.$$

Next, given the mesh $\mathtt{M}_t$, for any sample $\mathbf{X}$, we define the number density of particles at the level $\ell \in \mathcal{L}$ as:

$$\rho_{\mathbf{X}}^{\mathtt{M}_t}(\ell) = \frac{N_{\mathbf{X}}^{\mathtt{M}_t}(\ell)}{N_{\mathbf{X}} \mathtt{Vol}(\mathcal{D}_\ell)}, \tag{6}$$

where $N_{\mathbf{X}}^{\mathtt{M}_t}(\ell)$ is the number of particles from the sample $\mathbf{X}$ that are within the level ℓ of the mesh $\mathtt{M}_t$, and $N_{\mathbf{X}}$ the size of $\mathbf{X}_t$. As a measure of variation of the concentration, we define the coefficient of variation:

$$\rho_{t,t+\Delta t}^{\mathtt{M}_t} := \mathbb{E}\left[\frac{|\rho_{\mathbf{X}_{t+\Delta t}}^{\mathtt{M}_t} - \rho_{\mathbf{X}_t}^{\mathtt{M}_t}|}{|\rho_{\mathbf{X}_{t+\Delta t}}^{\mathtt{M}_t} - \rho_{\mathbf{X}_t}^{\mathtt{M}_t}| + 1} \right], \tag{7}$$

with $\mathbb{E}$ the expectation under the measure $\mathbb{P}$ defined above.

By construction, the coefficient of variation in Formula (7) is lower than 1 and increases with the difference of concentration between the samples $\mathbf{X}_t$ and $\mathbf{X}_{t+\Delta t}$ (seen by the mesh $\mathtt{M}_t$). In other words, if $\rho_{t,t+\Delta t}^{\mathtt{M}_t}$ is close to zero, then no significant variation of concentrations between time steps is detected and we can keep the decomposition $\mathtt{M}_t$ to compute the agglomeration at time $t + \Delta t$. More precisely, if $\rho_{t,t+\Delta t}^{\mathtt{M}} < \epsilon_2$ where ϵ_2 is another tolerance threshold, we keep the mesh $\mathtt{M}_t$. Otherwise, we apply full_D2SD/quick_D2SD to obtain a new spatial decomposition.

The criterion based on particle dispersion fluctuation (5), is less computationally expensive but also less accurate in some cases where the variation in the number of bins is balanced (see test case (c) in Section 3). Meanwhile, the criterion based on the difference of the particle concentration (7) is more accurate but at higher computational costs (as it requires the computation of densities). Therefore, the criterion proposed in this article is based on a two-step verification, starting with the verification of the particle dispersion fluctuation, and, when necessary, checking the particle concentration difference.

Formally, considering the input $(\mathbf{X}_t, \mathbf{X}_{t+\Delta t}, \mathtt{M}_t, \epsilon_1, \epsilon_2)$, the re-meshing criterion consists of the three following steps (sketched in Fig. 1) coming before step1 in Section 2.1:

step0.a Verify the percentage of the won/lost bins. Compute Δ_tN_bins in Equation (5). If $|\Delta_t$N_bins$| < \epsilon_1\%$ continue to step0.b, otherwise continue to step0.c1.

step0.b Verify the difference of the concentration of particles. Compute $\rho_{t,t+\Delta t}^{\mathtt{M}_t}$ in Equation (7). If $\rho_{t,t+\Delta t}^{\mathtt{M}_t} < \epsilon_2$, then continue to step0.c2, otherwise continue to step0.c1

step0.c1 Update the spatial decomposition. Apply quick_D2SD/full_D2SD to the set of particles position $\mathbf{X}_{t+\Delta t}$ and deduce the decomposition $\mathtt{M}_{t+\Delta t}$.

step0.c2 Keep the previous mesh. Set $\mathtt{M}_{t+\Delta t} = \mathtt{M}_t$.

Note that step0.b does not add computational cost when the decomposition $\mathtt{M}_t$ is good enough for the snapshot $\mathbf{X}_{t+\Delta t}$ since in this case the densities must be computed for the agglomeration step.

Threshold ϵ_1 indicates how much a cluster is allowed to disperse/collapse before changing the mesh. Threshold ϵ_2, on the other hand, indicates how much variation in concentration is tolerated before re-meshing.

As $\rho_{t,t+\Delta t}^{\mathtt{M}} \leq \mathbb{E}[|\rho_{\mathbf{X}_{t+\Delta t}}^{\mathtt{M}_t} - \rho_{\mathbf{X}_t}^{\mathtt{M}_t}|]$, we can imagine that we are looking for a maximum tolerated value for the concentration difference $m = \max_l |\rho_{\mathbf{X}_{t+\Delta t}}^{\mathtt{M}_t}(l) - \rho_{\mathbf{X}_t}^{\mathtt{M}_t}(l)| \in [0, 1]$, and then set:

$$\epsilon_2 = \frac{m}{1 + m}. \tag{8}$$

The values of these parameters will depend on the error allowed by the user. However, in the following section we suggest some values for ϵ_1, ϵ_2 from examples.

We now can summarise the steps of fast-D2SD, for each time-iteration (see also Fig. 1):

steps0+step1$|_{n_{crit}}$+step2(fully/partly)+step3$|_{n_{crit}}$.

Remark. *For simplicity, the mesh generated by the D2SD algorithm is a non uniform Cartesian mesh. However, D2SD can be adapted to more general meshes and to other geometry of*

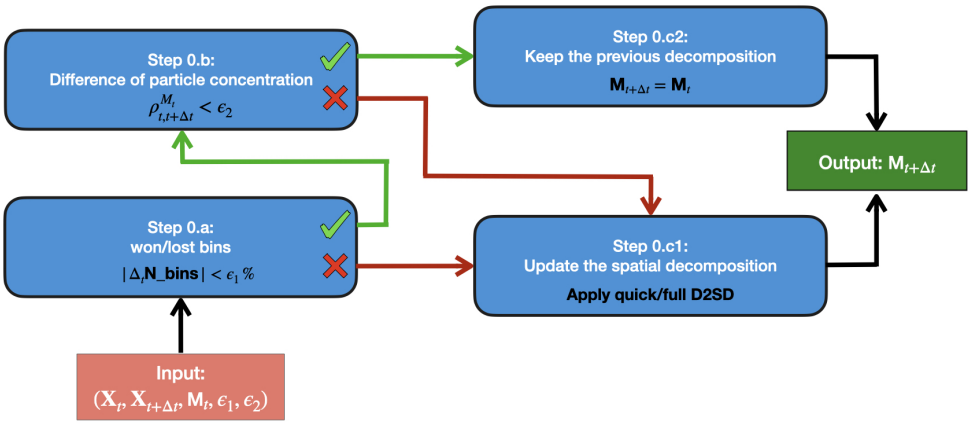


Figure 1: Sketch of the re-meshing criterion.

the domain than the unit cube: for `step2` *the key is to correctly approximate the empirical PDF, for example, with the help of a change of coordinate that puts the domain in correspondence with the cube. The extension of* `steps1,3` *to more general geometries, it is a bit more complex as -usually- uniformity tests are built on simple (mostly rectangular) domains, and generalising them would imply the construction of suitable statistics. Nevertheless, this difficulty can be easily mitigated by embedding the domain $\mathcal{D}$ in a rectangular domain $\widetilde{\mathcal{D}}$, and adding uniformly distributed ghost particles in the complementary part $\widetilde{\mathcal{D}}\backslash\mathcal{D}$ to test uniformity. When it is possible, coordinates transformation can also be an interesting alternative (see e.g. in Section 4.1.2).*

It should also be noted that the ideas and tools driving the D2SD algorithm can be easily adapted to other contexts where there is a need to identify regions of space that are distinguishable from others through some property of the system (in our case the number density of particles).

3. Testing the efficiency of the adapted D2SD algorithm

In this section, we evaluate the accuracy and efficiency of each of the simplification options proposed for the D2SD algorithm. This is done through a series of two-dimensional numerical experiments that consider both common situations and more specific scenarios where one of the indicators may fail.

To quantify the error resulting from the use of an adapted D2SD algorithm instead of the original D2SD algorithm, we introduce the measure $\mathcal{E}(k, \mathtt{M}_0)$. It is defined as the relative error between the agglomerations events computed using an initial mesh $\mathtt{M}_0$ and the agglomeration events computed using the full D2SD algorithm $\mathtt{M}^*_k$, at the time instant k. Mathematically, we define:

$$\mathcal{E}(k, \mathtt{M}_0) = \frac{|\mathcal{A}(\mathtt{M}^*_k) - \mathcal{A}(\mathtt{M}_0)|}{\mathcal{A}(\mathtt{M}^*_k)}, \tag{9}$$

where, for any mesh $\mathtt{M}$ with $\mathcal{L}$ levels,

$$\mathcal{A}(\mathtt{M}) = \sum_{\ell \in \mathcal{L}} \beta \frac{N^{\mathtt{M}}_{\mathbf{X}}(\ell)^2}{\mathtt{Vol}(\mathcal{D}_\ell)} \Delta t, \tag{10}$$

is the number of agglomeration events computed using the mean-field statistical approach for agglomeration on the spatial decomposition obtained (with $\alpha = 1$), Δt is the time step, and $\beta = \beta_{\mathbb{v}_1,\mathbb{v}_1}$ the primary particle collision kernel.

3.1. Testing the quick D2SD algorithm without re-meshing criterion

We start by assessing the accuracy and efficiency of the quick D2SD algorithm, which relies on the 60th percentile as an a priori value for the contrast threshold $\hat{\lambda}$ (instead of the iterative search of the optimal one). For that purpose, we perform 2D simulation in the same five cases investigated in the previous paper, which were selected to represent a range of situations that can happen in CFD simulations [50]:

i. homogeneously distributed population of particles;

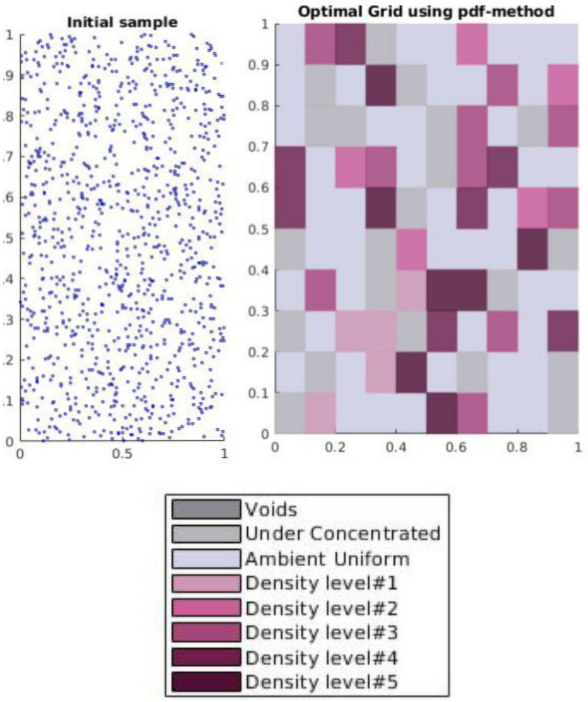


Figure 2: Testing D2SD with $\hat{\lambda} = Q_{60}$ for a population of particles uniformly distributed. In this case $\lambda^* = Q_{60}$.

ii. slightly non-homogeneous distribution;

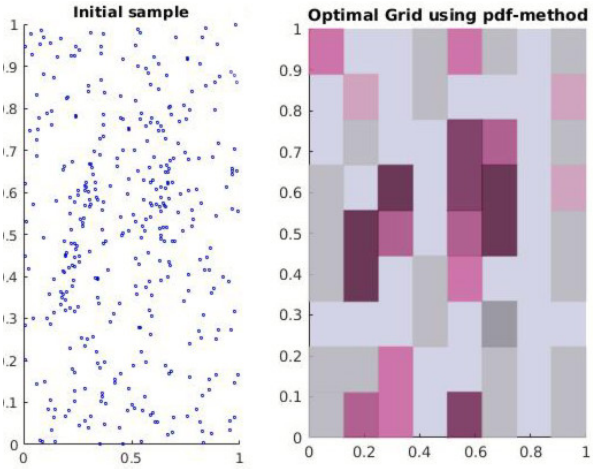


Figure 3: Testing D2SD with $\hat{\lambda} = Q_{60}$ for a population of particles containing clusters of particles with a mild concentration. In this case $\lambda^* = Q_{60}$. (Levels are defined as in Fig. 2.)

iii. medium non-homogeneous distribution;

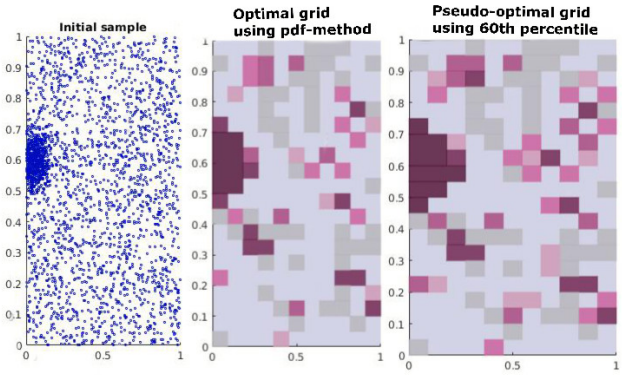


Figure 4: Testing D2SD with $\hat{\lambda} = Q_{60}$ for a population of particles containing clusters of particles with a medium concentration. Plot of the mesh produced with $\hat{\lambda}$ is included since $\lambda^* \neq \hat{\lambda}$. (Levels are defined as in Fig. 2.)

iv. highly non-homogeneous distribution;

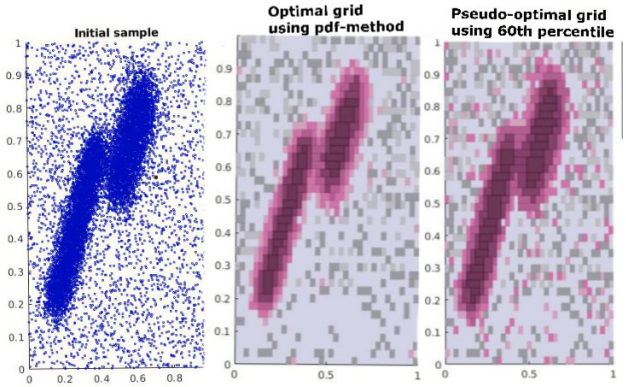


Figure 5: Testing D2SD with $\hat{\lambda} = Q_{60}$ for a population of particles: containing clusters of particles with a high concentration. Plot of the mesh produced with $\hat{\lambda}$ is included since $\lambda^* \neq \hat{\lambda}$. (Levels are defined as in Fig. 2.)

v. symmetric case (with an empty region).

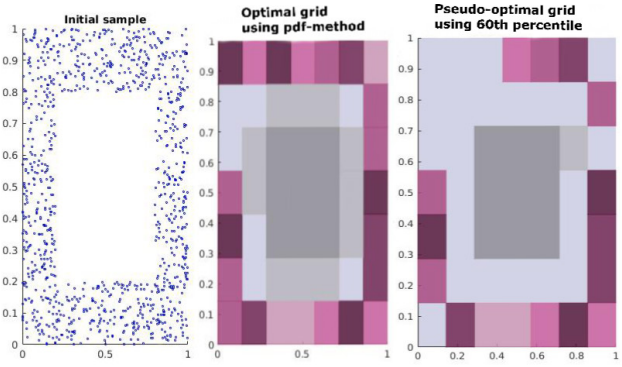


Figure 6: Testing D2SD with $\hat{\lambda} = Q_{60}$ for a population of particles with a symmetric distribution. In this case, for `full_D2SD`, $\lambda_1^* = Q_{60}$ does not pass the uniformity test. The optimal grid is obtained after a second iteration for the computation of λ^*, showing the impact of the iteration criterion. (Levels are defined as in Fig. 2).

Figures 2 - 6 display the five cases considered, showing the spatial distribution (left) together with the optimal mesh decomposition and the mesh decomposition obtained with $\hat{\lambda} = Q_{60}$ (when it differs from the optimal mesh). It is complemented by the results in Table 1, which contains information about: the number of thresholds tested in the `step2.f` of the original

CASE	Thresholds evaluated	$\frac{\lambda^*}{\hat{\lambda}}$	$\frac{\mathcal{A}^*}{\mathcal{A}}$	$\frac{\texttt{CPU}^*}{\texttt{CPU}}$
Homogeneous	3	1	1	1
Slightly non-homogeneous	2	1	1	1
Medium non-homogeneous	3	1.08	1.15	1.51
Highly non-homogeneous	9	1.25	1.01	2.21
Symmetric	4	1	1.019	3.19

Table 1: Testing D2SD with $\hat{\lambda} = Q_{60}$ for a population of particles: uniform distributed, containing clusters of particles with a mild-medium or high concentration or with a symmetric distribution.

D2SD algorithm, the ratio between the optimal threshold and the 60th percentile, the ratio between the number of agglomerations $\mathcal{A}^*$ for the optimal mesh or the number of agglomerations $\mathcal{A}$ for the mesh obtained with $\hat{\lambda} = Q_{60}$, and the ratio between the computational times of the original D2SD algorithm and the quick D2SD method.

The first key result is that the quick D2SD mesh coincides with D2SD (optimal mesh) in the cases of a uniform spatial distribution or a slightly non-homogeneous distribution. This means that the optimal threshold corresponds to the 60th percentile, i.e. $\lambda^* = Q_{60}$. In these cases, the quick D2SD algorithm provides the same results as the D2SD (i.e. without error). In the other cases, the optimal threshold does not coincide with the 60th percentile, the largest difference being reached for the highly non-homogeneous case. Yet, despite this difference in the threshold, the error made in the number of agglomeration events does not exceed 15% (with the biggest difference occurring for the medium non-uniform distribution). In terms of computational costs, the use of the quick D2SD algorithm can speed up the results up to 200% or 300% in some cases compared to the original D2SD algorithm.

To further analyse these results, we evaluate the ratio of the number of agglomeration events $\frac{\mathcal{A}^*}{\mathcal{A}}$ in each of the five cases as a function of the number of initial particles. This allows to assess the balance between the accuracy and efficiency of the quick D2SD algorithm (i.e. numerical error to computational cost). The results are presented in Figure 7, where we can verify how the accuracy of the quick D2SD algorithm stabilises as the number of particles n increases, and the difference in cost increases with n. In general, we can say that $\lambda = Q_{60}$ is a suitable choice for the threshold in order to decrease the computational cost. The only difficulty arises when the number of particles is small, where it may be preferable to apply the original D2SD algorithm.

These tests confirm that the quick D2SD algorithm can be safely used to efficiently compute particle agglomeration in most of the cases of interest, as long as the expected results are within a 15% error range.

3.2. Testing the re-meshing criterion

We assess here the criterion that determines when the D2SD algorithm should be applied again, assuming that it has already been applied in a previous time step. For that purpose, we design test cases that account for both inhomogeneities in the spatial distribution of particles and the motion of particles with time. In Section 3.2.1, we start by describing and testing the D2SD algorithms on each case, suggesting furthermore some values for the thresholds ϵ_1 and ϵ_2. Then, the fast D2SD algorithms are tested on selected cases and their efficiency is assessed in Section 3.2.2.

3.2.1. Setting-up thresholds ϵ_1 and ϵ_2

We have selected three different cases; for each case, we apply the following analysis method: (1) we start with a set of initial particle positions $\mathbf{X}_1$; (2) we apply the D2SD algorithm to obtain the initial optimal decomposition $\texttt{M}_1^*$; (3) particles are displaced during a time step Δt and (4) we apply again the D2SD algorithm (full computation reference) and we compare the results to those obtained with the re-meshing criterion (fast computation). This allows to determine at each time-iteration if an update of the spatial decomposition is required. Finally the number of agglomeration events is computed. This procedure is repeated for ten time-iterations intending to assess the evolution of the error and indicators as the simulation progresses. For each time-iteration k, we compare the two meshes obtained, i.e. $\texttt{M}_k$ versus $\texttt{M}_k^*$. To that extent, we compute the evolution of the indicators $\Delta_{m,k}\texttt{N\_bins}$ (i.e. percentage of won/lost bins) and $\rho_{m,k}^{\texttt{M}_m}$ (difference of particle concentration with respect to the previous and initial mesh) for various time steps $k = 2, \ldots, 10$ and initial time $m = 1, k-1$.

Each case and the corresponding results are described in the following:

a. A cluster that disperses uniformly in space.
This illustrates cases where a particle cluster disperses in the domain to reach a homogeneous distribution at longer times, as shown in Figure 8.
The dispersion of the initial cluster in space translates into a strong evolution of the optimum splitting obtained by applying the D2SD algorithm every time step. This is especially true when comparing results with the one got at the 1^{st} time-iteration, where the distribution is still far from a homogeneous one (bottom figures). As a result, the number of won/lost bins compared to the initial time-iteration is always very high. Setting up a reasonable tolerance threshold of $\epsilon_1 = 15\% - 20\%$ won/lost bins between two time-iterations is consistent with the results observed in Fig. 8, drawing on the fact that we expect re-meshing to be unnecessary when a stationary state has been attained (see the right plots of Fig. 8). The same trend is observed for the concentration difference, which is always around 0.5 when comparing it to the initial time-iteration but quickly decreases when comparing to the previous time-iteration, where the concentration difference with respect to the previous/initial time-iteration at the stationary state is around 0.15. Indeed, this is also consistent with a tolerance threshold of $m = 20\%$ of maximum

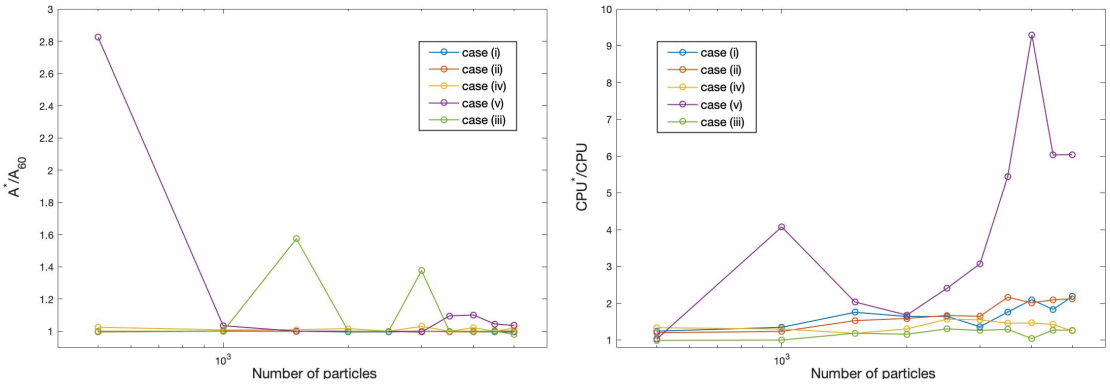


Figure 7: Testing D2SD with $\hat{\lambda} = Q_{60}$ for a population of particles of size $n = \{500\ell : \ell \in \{1, \ldots, 10\}\}$ in the case: homogeneous, slightly, medium and highly non-homogeneous, and symmetric.

difference between concentration at t and $t + \Delta t$ (see Equation (8)) with associated threshold $\epsilon_2 = 1/6$.
Using the threshold of $\epsilon_1 = 15{-}20\%$ for the won/lost bins and $\epsilon_2 = 15 - 20\%$ for the concentration difference, the present results would suggest to re-mesh the first two time-iterations. After the third time-iteration, a steady-state is reached and re-meshing is not likely anymore.

b. Local injection in a homogeneous distribution.
This case is expected to be representative of typical CFD simulations with an injection of particles. As seen in Figure 9 (top), the initial distribution is homogeneous and becomes non-homogeneous with time due to the local injection of particles (this is not compensated by the uniform displacement since the injection is the dominant phenomenon).
This complex evolution of the distribution is well captured by applying the D2SD algorithm every time step (bottom figures). However, as displayed in Figure 9 (right), the won/lost bins with respect to the initial time-iteration increases due to the development of a strongly non-homogeneous distribution. This is especially true in the first few time-iterations, where even the won/lost bins between consecutive time-iterations peaks up to 200%. However, at longer times, both the won/lost bins and concentration difference between consecutive time-iterations reach values similar to the homogeneous distribution in time. This is due to the establishment of a steady-state situation, where the number of particles remains constant in the domain (due to the balance between newly injected particles and particles going out of the square of observation).
The won/lost bins with a threshold of $\epsilon_1 = 15 - 20\%$ provide a quick first acceptance/rebuttal which is completed by the 2nd criterion on the concentration difference with a threshold around $\epsilon_2 = 15 - 20\%$ (when needed). Here, re-meshing is expected to occur during the second and third time-iterations and again around time-iteration 9.

c. Two clusters (expanding/shrinking in space).
This case is representative of a situation where two initial clusters evolve in time (see also Figure 10): one is expanding in space while the other one is shrinking. The rates of expansion and shrinking are designed with opposite values, such that the number of bins remains constant.
As seen in the right plots of Figure 10, the number of won/lost bins always remains below 10% regardless if it is compared to the initial or previous time-iteration. The information on the change in particle positions comes from the concentration difference, which is seen to steadily increase from 0.1 to 0.35 when compared to the initial time-iteration. The concentration difference w.r.t. the previous time-iteration remains relatively low (below 0.16).
This case thus highlights the interest of the two-step process. Here, we expect the `step0.a` (won/lost bin) to be accepted every time, but the concentration difference suggests to re-mesh every few time-iterations because of the change in particle locations (`step0.b`). Thus, resorting to a single-step process (i.e. using information on won/lost bins only) would lead to incorrect results while the two-step process does capture this change. The concentration difference will provide more information on the actual changes in concentrations.

All these cases support the use of a re-meshing criterion based on two indicators, namely the number of won/lost bins and the concentration difference. Drawing on the present tests, we suggest to set the tolerance thresholds as $\epsilon_1 = 15 - 20\%$ and $\epsilon_2 = 1/6$. It is worth noting that these thresholds are expected to work even when the number of particles is small. In fact, in such cases, the creation/removal of one particle will correspond to significant changes in the concentration, meaning that the 2^{nd} criterion will be satisfied. Further testing of the algorithm at very low particle numbers will be done in the near future.

The next step consists of applying dynamically the criteria described by `steps0.a-0.c` in Section 2.2.2 and evaluating the error $\mathcal{E}(k, \mathtt{M}_k)$ made on the computation of particle agglomeration (see Equation (9)), with $\mathtt{M}_k$ the output of `step0` at time-iteration k. This means that the criteria are applied at every time-iteration: if rejected, the current time-iteration becomes the new reference mesh but, if accepted, the reference mesh is left untouched (see Figure 1).

3.2.2. *Efficiency of re-meshing and illustration of the effect of the thresholds*

Here, we analyse the results obtained with the re-meshing criterion applied dynamically. Based on the results of Section

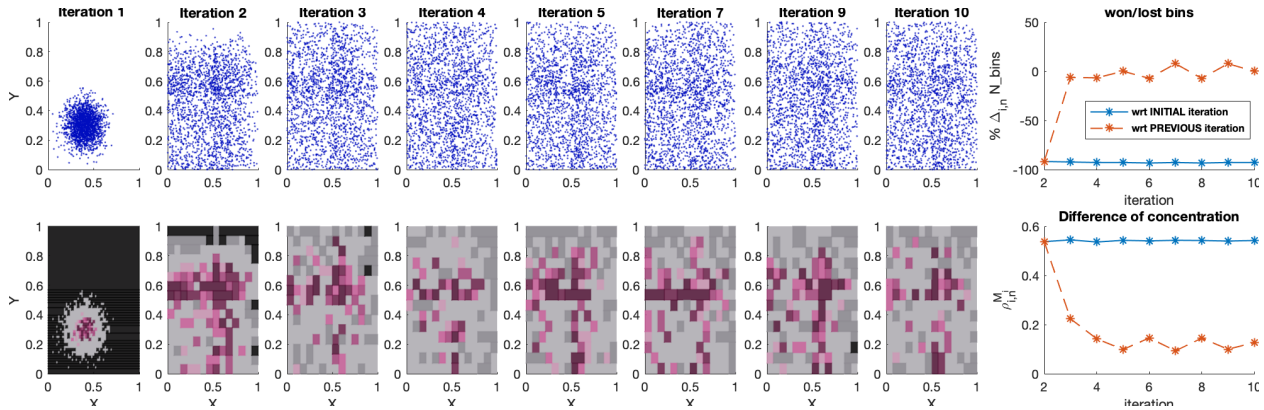


Figure 8: Cluster that disperses uniformly in space, with parameter $\Delta t = 0.5$. Evolution of the distribution in 10 time-iterations (left-top figures) and corresponding `full_D2SD` meshes (left-bottom figures); Levels are as in Fig. 2 except for the empty cells in black for better contrast. Evolution of the won/lost bins $\Delta_{m,k}$`N_bins` (right-top figure) and difference of particle concentration $\rho^{M_m}_{m,k}$ (right-bottom figure) is shown with respect to: initial time-iteration ($m = 1$, in blue) and previous time-iteration ($m = k - 1$, in orange).

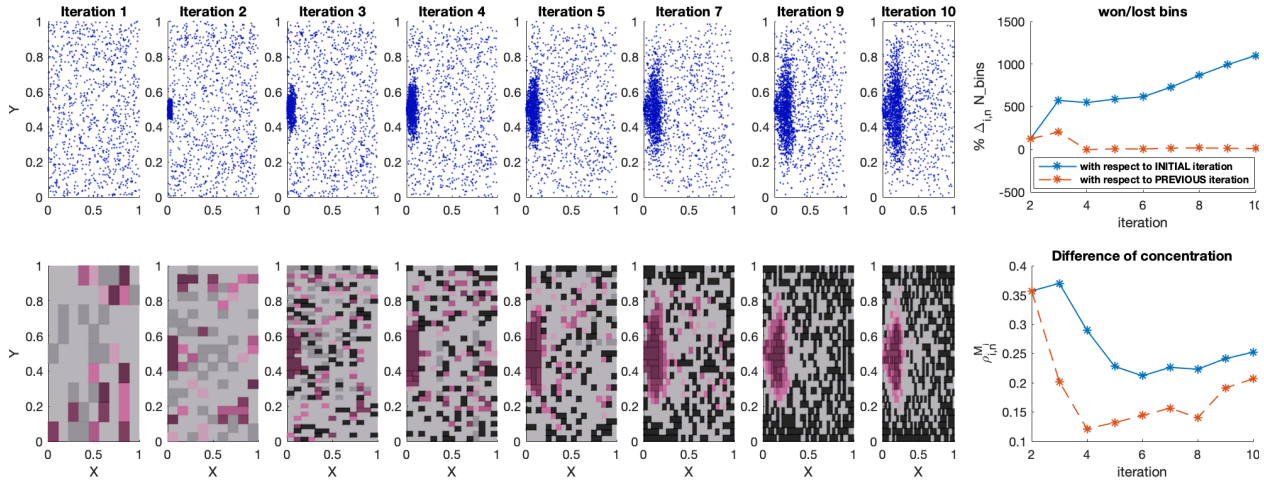


Figure 9: Local injection in a homogeneous distribution, with parameter $\Delta t = 0.1$. Evolution of the distribution in 10 time-iterations (left-top figures) and corresponding `full_D2SD` meshes (left-bottom figures); Levels are as in Fig. 2 except for the empty cells in black for better contrast. Evolution of the won/lost bins $\Delta_{m,k}$`N_bins` (right-top figure) and difference of particle concentration $\rho^{M_m}_{m,k}$ (right-bottom figure) is shown with respect to: initial time-iteration ($m = 1$, in blue) and previous time-iteration ($m = k - 1$, in orange).

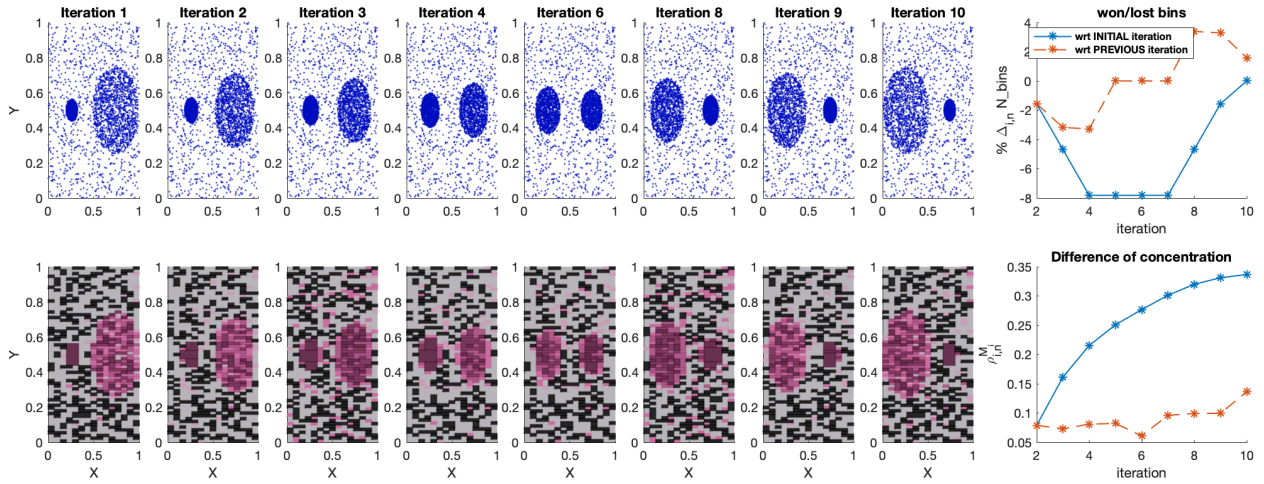


Figure 10: Two clusters (expanding/shrinking in space), with parameter $\Delta t = 0.1$. Evolution of the distribution in 10 time-iterations (left-top figures) and corresponding `full_D2SD` meshes (left-bottom figures); Levels are as in Fig. 2 except for the empty cells in black for better contrast. Evolution of the won/lost bins $\Delta_{m,k}$`N_bins` (right-top figure) and difference of particle concentration $\rho^{M_m}_{m,k}$ (right-bottom figure) is shown with respect to: initial time-iteration ($m = 1$, in blue) and previous time-iteration ($m = k - 1$, in orange).

3.2.1, we set the tolerance thresholds as $\epsilon_1 = 20\%, \epsilon_2 = 1/6$, where the latter corresponds to $m = 20\%$ for the maximum difference $|\rho^{\mathtt{M}_{k-1}}_{\mathbf{X}_{k-1}} - \rho^{\mathtt{M}_{k-1}}_{\mathbf{X}_k}|$, during time-iteration k.

As displayed in Figure 11, for each of these cases and each (time) time-iteration k, we illustrate the evolution of the indicators $\Delta_{k-1,k}\mathtt{N\_bins}$, $\rho^{\mathtt{M}_{k-1}}_{k-1,k}$ and the error $\mathcal{E}(k, \mathtt{M}_k)$ ($\mathtt{M}_k$ being the output of the criterion at time-iteration k). For the sake of clarity, we have also added markers around the time-iteration identifier (see x-axis on plots 11(a) and 11(b)) to highlight every time a re-meshing is suggested by each criterion. Several conclusions can be drawn from this set of figures. First, we observe that the error made in evaluating the number of agglomeration events always remains below 0.4, even in complex cases such as case c. Second, in the cases where the population of particles reaches a state of uniform distribution (last time-iterations in case a.), the criterion suggests keeping the spatial decomposition unchanged (i.e. no re-meshing is applied), in accordance with the small error made on the calculation of agglomerations.

To further analyse how the setup of the thresholds ϵ_1 and ϵ_2 influences the error made on the prediction of agglomeration, we focus on the case of clusters expanding/shrinking (case c). For that purpose, two different pairs (ϵ_1, ϵ_2) of thresholds are used: $\{(10\%, \frac{1}{11}), (20\%, \frac{1}{6})\}$, corresponding respectively to $m = 10\%$ and $m = 20\%$ in Eq. (8). The results obtained are shown in Figure 12: it can be seen that, by tuning the values of ϵ_i, a lower error on the agglomeration prediction is obtained. This is related to the fact that, with lower tolerance thresholds, a higher number of re-meshing events are triggered. Indeed, compared to the error associated with a threshold of 20%, the error associated with a threshold of 10% decreases by almost half (an error of about 0.4 against one of about 0.25). However, a higher accuracy in the results obtained also comes with higher computational costs since more re-meshing occurs. These experiments confirm that the proposed re-meshing criterion is a suitable candidate to increase the computational efficiency of the D2SD algorithm while respecting a user-defined criterion on the error made in the evaluation of particle agglomeration events.

3.3. Testing `quick_D2SD` not every time step

In this section, we assess the accuracy and efficiency of the algorithm combining both the `quick_D2SD` algorithm and the re-meshing criterion. For that purpose, we consider case (c) with clusters expanding/shrinking and we compare the error made between the computation of agglomeration events implementing the `quick_D2SD` versus `full_D2SD` algorithm, both using the re-meshing criterion, i.e. `Algo 3` versus `Algo 1`. The results for $(\epsilon_1, \epsilon_2) = (20\%, 1/6)$ are displayed in Figure 13: we observe that the error propagates as the time-iterations progress but remains within acceptable levels. When applying `Algo 1` (`full_D2SD` with re-meshing criterion), we can observe a decrease of the error made between the computation of agglomeration events. Whereas by applying `Algo 3` (`quick_D2SD` with re-meshing criterion), the error propagates and increases as time-iterations progress. Regarding the computational time, denoting by $\mathtt{CPU}^*$ the cumulative computational time (over all time-iterations) associated with the implementation of the `full_D2SD` algorithm: `Algo 1` was executed in approximately 40%$\mathtt{CPU}^*$, while `Algo 3` took 29%$\mathtt{CPU}^*$. These results underline the great advantage, as far as computational cost is concerned, of the fast versions of the D2SD algorithm proposed in this paper. In particular, it highlights the improvement regarding the computational cost when the `Algo 3` is implemented. However, this gain comes with a loss of accuracy, which can be clearly seen in the three last time-iterations in Figure 13. Therefore, when implementing the quick version of the D2SD algorithm, it is suggested to refresh from time to time the spatial decomposition using the full version in order to stop the propagation of numerical errors.

4. Application to a practical case and validation

In this section, the `quick_D2SD` algorithm is coupled to 3D simulations of particle agglomeration in practical cases. The objectives are two-fold: first, we want to assess the accuracy and efficiency of the method in a validation case; second, we illustrate how the D2SD can be applied in a practical case that is representative of situations of interest in the multiphase flow community.

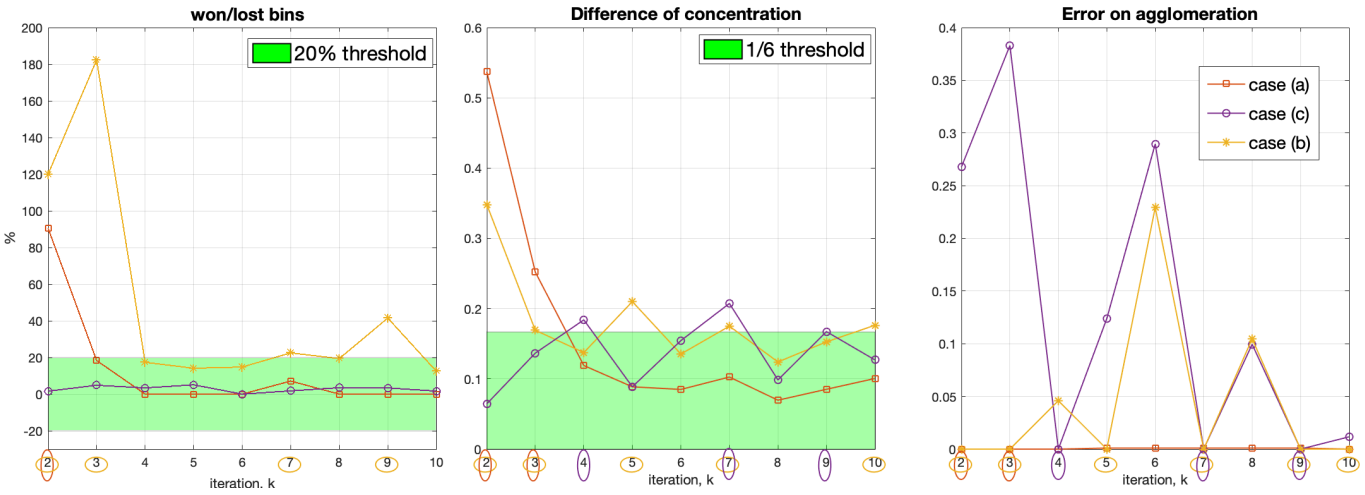
For that purpose, the following section is divided into two subsections. In Section 4.1, `quick_D2SD` is applied to a validation case, which consists of the simulation of particle agglomeration undergoing diffusive motion but with a non-homogeneous spatial distribution of particles. In Section 4.2, the `quick_D2SD` algorithm is applied to a practical case representative of particle dispersion in an atmospheric boundary layer with an obstacle.

4.1. Validation case

The `quick_D2SD` is first applied and compared to numerical results obtained with micro-scale simulations for the agglomeration of particles undergoing diffusive motion. Since the aim is to validate the D2SD algorithm, we focus our attention on a case involving non-homogeneous repartition of particles, here with a local injection. The case studied together with the micro-scale model and theoretical expectations on the rate of agglomeration are presented in Section 4.1.1 while the use of the `quick_D2SD` algorithm and the comparison to these results are discussed in Section 4.1.2.

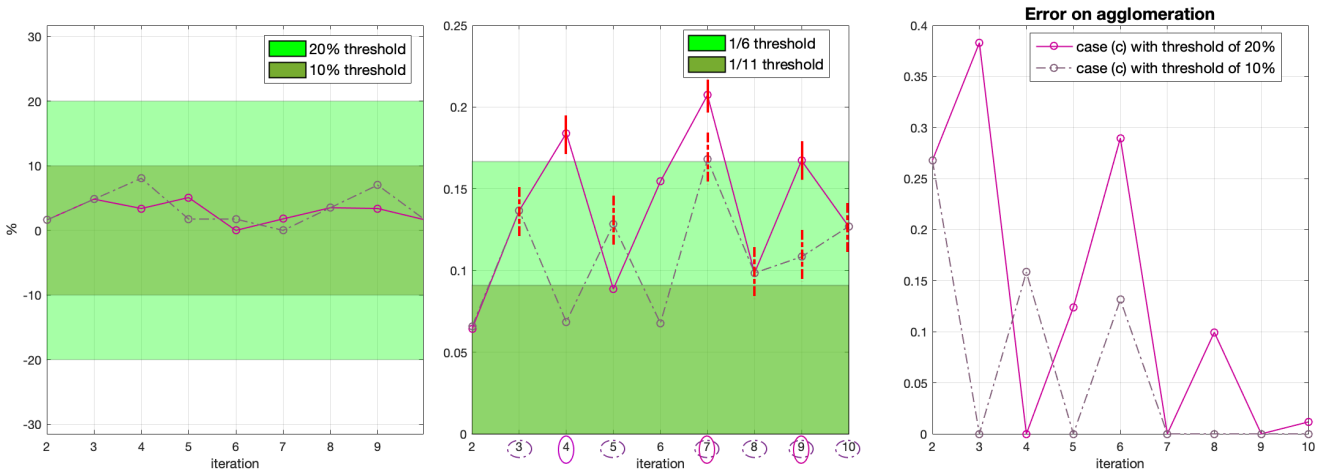
4.1.1. Micro-scale simulation of particle agglomeration with local injection

Case studied: local particle injection. A case of non-homogeneous particle repartition has been selected in order to highlight the interest of using the D2SD algorithm in such complex cases, as well as to assess the accuracy of the `quick_D2SD` algorithm. We have opted here to simulate the case of particles undergoing diffusive motion since a theoretical estimation of the agglomeration rate can be computed in that case. More precisely, we focus on the non-homogeneous case of particles within a spherical domain of radius R_s, but with a higher concentration around the sphere centre. Since the aim of these micro-scale simulations is to extract from the results statistical information on the agglomeration rate, the case studied has to be a steady-state. For that reason, we did not set-up a simulation in a sphere with rebound conditions on the edges. In that case, even if particles are initially non-homogeneously distributed, the purely



(a) Evolution of $\Delta_{k-1,k}$N_bins (b) Evolution of $\rho^{\mathtt{M}_{k-1}}_{k-1,k}$ (c) Evolution of $\mathcal{E}(k, \mathtt{M}_k)$

Figure 11: Illustration of the studied cases: dispersing cluster (case a), injection of particles (case b) and extraction/expansion of clusters (case c). Plots for the evolution of the won/lost bins $\Delta_{k-1,k}$N_bins and difference of concentration $\rho^{\mathtt{M}_{k-1}}_{k-1,k}$ are shown using thresholds $(\epsilon_1, \epsilon_2) = (20\%, \frac{1}{6})$. Additionally, the corresponding error on agglomeration events is plotted in Fig. 11(c)



(a) Evolution of $\Delta_{k-1,k}$N_bins (b) Evolution of $\rho^{\mathtt{M}_{k-1}}_{k-1,k}$ (c) Evolution of $\mathcal{E}(k, \mathtt{M}_k)$

Figure 12: Illustration of the extraction/expansion of clusters (case c). Plots for the evolution of the won/lost bins $\Delta_{k-1,k}$N_bins and difference of concentration $\rho^{\mathtt{M}_{k-1}}_{k-1,k}$ are shown using thresholds $(\epsilon_1, \epsilon_2) = (20\%, \frac{1}{6})$ (light green) and $(\epsilon_1, \epsilon_2) = (10\%, \frac{1}{11})$ (dark green). Additionally, the corresponding error on agglomeration events is plotted in Fig. 12(c)

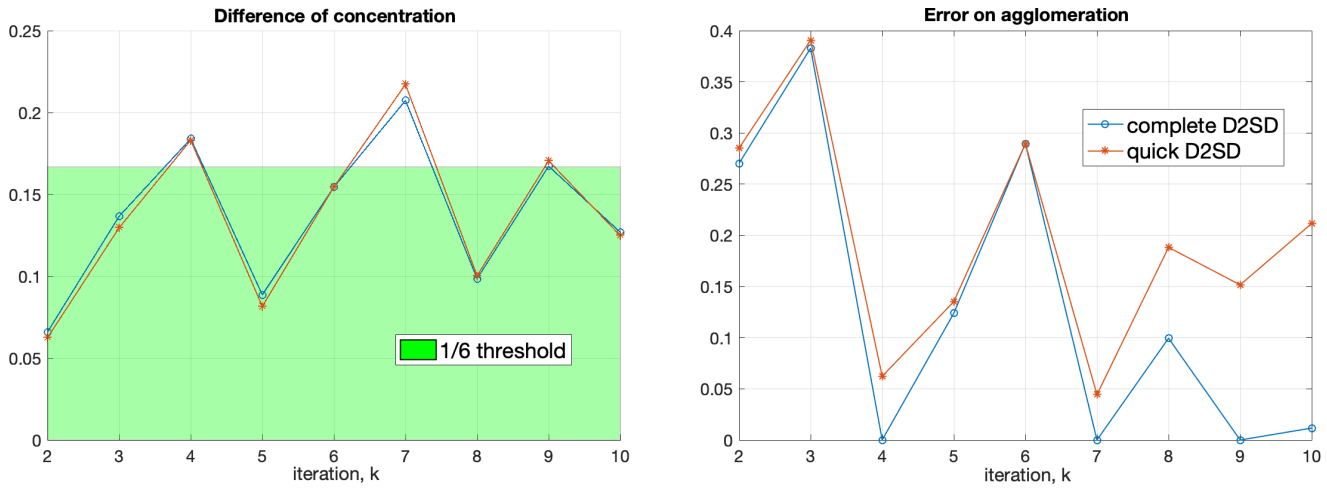


Figure 13: Evolution of the difference of particle concentration and error on agglomeration events in the case (f.): expansion/shrinking of clusters. Results for `Algo 3` (`quick_D2SD` plus re-meshing criterion) are plotted in orange while the results for `Algo 1` (`full_D2SD` plus re-meshing criterion) are plotted in blue.

diffusive motion will result in a steady state corresponding to a case of homogeneously distributed particles within the sphere (see case a in Section 3.2.1). We rather opted for a continuous injection of particles near the sphere centre, which results in a non-homogeneous steady-state where particles are more concentrated near the sphere centre. Particles reaching the domain boundary are removed from the simulation. When a collision between two particles is detected, one of the colliding partners is removed from the simulation (this corresponds to the sticky case where particles adhere to each other upon colliding).

Theoretical collision rate. As mentioned previously, we have focused on this case since the collision rate can be estimated theoretically (details about the derivation of the collision rate are provided in Appendix C). Here, the average number of particles evolves as

$$\frac{\mathrm{d}}{\mathrm{d}t}\langle N(t)\rangle = -\frac{\beta}{\mathtt{Vol}(\mathcal{D})}\,\langle N(t)\,(N(t)-1)\rangle \tag{11}$$

where the collision rate β is given by

$$\beta = \frac{8\pi R_p^2\,\sigma}{\sqrt{2\pi\gamma}}. \tag{12}$$

It should be noted here that this formula for the collision rate was obtained, assuming that particles are homogeneously distributed within the domain. Actually, in this case, the same formula holds, although particles are not distributed homogeneously: this is because in each circular region around the centre of the domain, the concentration can be considered as locally homogeneous. This means that the same collision rate holds and that a higher number of collisions occurs in regions with higher concentrations (since the collision frequency is multiplied by the local particle concentration to compute the number of collision events).

Microscopic Langevin simulations. We perform here numerical simulations of particles undergoing Langevin diffusive motion using the following equations:

$$\begin{aligned} d\mathbf{X}_i &= \mathbf{V}_i dt, \\ d\mathbf{V}_i &= -\gamma\mathbf{V}_i dt + \sigma dB_i(t), \end{aligned} \tag{13}$$

with the particle position $\mathbf{X}_i$, the fluid velocity $\mathbf{V}_i$, the friction γ, the noise amplitude σ and a family of independent Brownian motion $(B_i(t); t \geq 0)_i$. All the particles are considered to have the same radius R_p. N_{inject} particles are injected in the domain at the sphere centre every N_{iter} time-iterations. Particles reaching the domain boundary (here at $R_s = 0.5$) are eliminated from the simulation. This allows one to obtain a steady-state once the number of particles entering the domain is compensated by the number of particles exiting the domain. The motion of particles is computed with a time step small enough so that the average particle displacement in a time step is close to the particle radius, i.e. $|\mathbf{X}_i(t+\Delta t) - \mathbf{X}_i| \simeq R_p$. This allows to detect the collision between two particles using geometric arguments assuming a ballistic regime within the time step (i.e. the motion of a particle within one time step follows a straight line). When such a collision is detected, one of the partners is removed from the simulation. This choice corresponds to the case of sticky collisions (i.e. a collision does lead to agglomeration). The collision rate is extracted from these simulations by recording the number of collisions N_{coll} that occurred during a certain amount of time t_{simu}. Since the particles are not homogeneously distributed in the domain, the collision rate is analysed as a function of the distance from the sphere centre (where particles are injected at a given frequency). For that purpose, we define a number of spherical rings according to their distance from the centre, here comprised between $r\,R_s/50$ and $(r+1)\,R_s/50$, with $r = 0, \ldots, 49$. Then, drawing on Eq. (11) which gives the evolution of average number of particles in the domain, the collision rate is estimated within each ring over a given simulation time t_{simu}:

$$\beta_{approx.}(r) = \frac{N_{coll}(r)}{t_{simu} N_p(r)(N_p(r)-1)} \tag{14}$$

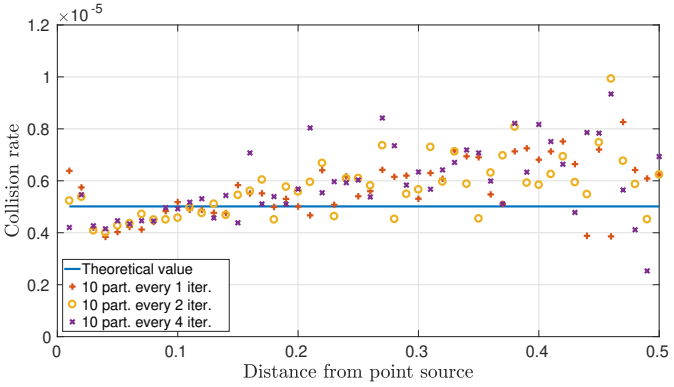


Figure 14: Comparison between the theoretical value of the collision rate and numerical estimations obtained with fine Langevin simulations using various particle concentrations.

where $N_p(r)$ is the number of particles averaged in time in the ring labelled r, and $N_{coll}(r)$ the number of collisions averaged in time in the same ring.

Results obtained. Figure 14 displays the comparison between the theoretical value of the collision rate and the results obtained with the numerical simulations as a function of the distance from the point source (where particles are injected locally). The coefficients σ and γ are respectively fixed to 1 and 1. As expected from standard mean-field formulations, the collision rate appears to be independent of the distance from the point source, despite the variation in the particle concentration. The results are less accurate as the distance from the point source increases since the number of particles present in these regions is lower (leading to higher statistical noise).

4.1.2. Simulations with `quick_D2SD`

The next step consists in coupling the `quick_D2SD` algorithm to standard CFD simulations. Here, the particle positions generated by the fine Langevin simulations are used as an input data for the `quick_D2SD` algorithm every time step. This allows one to test the accuracy and efficiency of the `quick_D2SD` algorithm.

The `quick_D2SD` algorithm has been applied to 1000 consecutive iterations sampled from the simulation with 10 particles injected every 4 time steps. We have used a spatial transformation proposed in [50], that allows to analyse the homogeneity in the particle concentration using spherical coordinates instead of Cartesian ones (which better fit the present case). Sometimes it is desirable/more appropriate to make use of non-Cartesian meshes; this can be achieved by the use of non-Cartesian coordinates. However, when changing coordinates, it is mandatory that the transformation preserves the uniform density, otherwise `steps1,3` have to be modified properly.

The results obtained for the identification of high concentration regions from `quick_D2SD` are illustrated in Fig. 15: it can be seen that clusters of particles are clearly identified near the sphere centre while other regions further from the centre can also be identified as having a concentration slightly higher than the average one.

Using Eq. (10), we compare the number of collision events computed using the D2SD algorithm over the whole spherical domain to the one directly extracted from fine Langevin simulations and the one provided by applying the mean-field formulation on the whole domain. The results are shown in Table 2, indicating that the evaluation of the number of collision events is greatly improved using the D2SD algorithm compared to the results obtained using the mean-field approach over the whole domain. In fact, the mean-field formulation applied over the whole domain leads to a significant underestimation of the number of collision events (around 75%). Meanwhile, the error made using the D2SD algorithm is much smaller: it increases slightly using the `quick_D2SD` algorithm with re-meshing criterion instead of the `full_D2SD` algorithm (from 18.5 % to 20.2 %) but the computational costs are reduced by a factor 3 with the `quick_D2SD` algorithm.

CASE	**Langevin simulation**	`full_D2SD`	`quick_D2SD`	**Mean-field one cell**
Nb. coll. events	523	426	417	130

Table 2: Values of the number of collision events over 1000 time-iterations computed using the fine Langevin simulation, the `full_D2SD` or `quick_D2SD` algorithm with re-meshing and the mean-field approach over the whole domain (one cell only).

4.2. Practical case

Case description.. The chosen practical test case corresponds to the flow in a neutral boundary layer on which a square obstacle is placed. The dimension of the obstacle along the y-direction (i.e. span-wise) is considered to be much larger than the other characteristic lengths, meaning that the flow can be simplified to a 2D flow around an obstacle. This case is actually representative of flows over large obstacles in atmospheric dispersion studies (e.g. flow over a building). This case has been chosen due to its relevance in the atmospheric dispersion community and also since the flow obtained is highly non-homogeneous.

Simulation set-up. The geometry used in numerical simulations is the same as the one used in a previous paper [62] (see also Fig. 16): the domain consists in a rectangular box with a length of 5 m, a height of 1 m and a thickness of 0.1 m while the square obstacle is located in the middle of the channel with a fixed size of 0.1 m. The mesh is made up of 798 400 square cells (with a fixed thickness equal to the domain thickness). The boundary conditions are chosen as follows (more details can be found in [62]): the inlet condition is set to be representative of the flow features typical of rough boundary layers (with a log-law for the scaling of the average velocity $U_{f,x}$ along the direction z); an outlet condition is imposed on the other side of the box; a symmetry condition is applied at the top while a smooth wall condition is applied at the bottom surfaces. The simulation is run resorting to a Reynolds Stress Model for turbulence (more precisely a Rotta

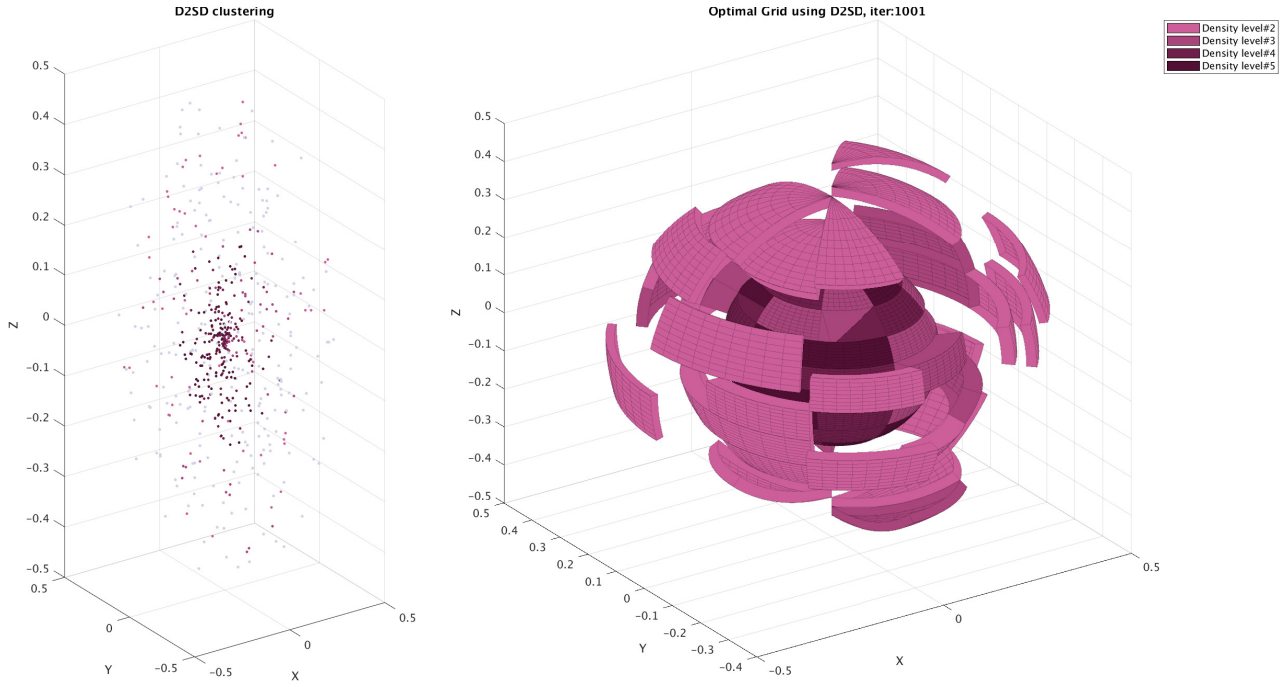


Figure 15: 3D plots showing the particles positions coloured according to their clustering region (left) and the spatial decomposition given by the `quick_D2SD` algorithm using particle positions at a given time-iteration of the Langevin simulation (here 1001).

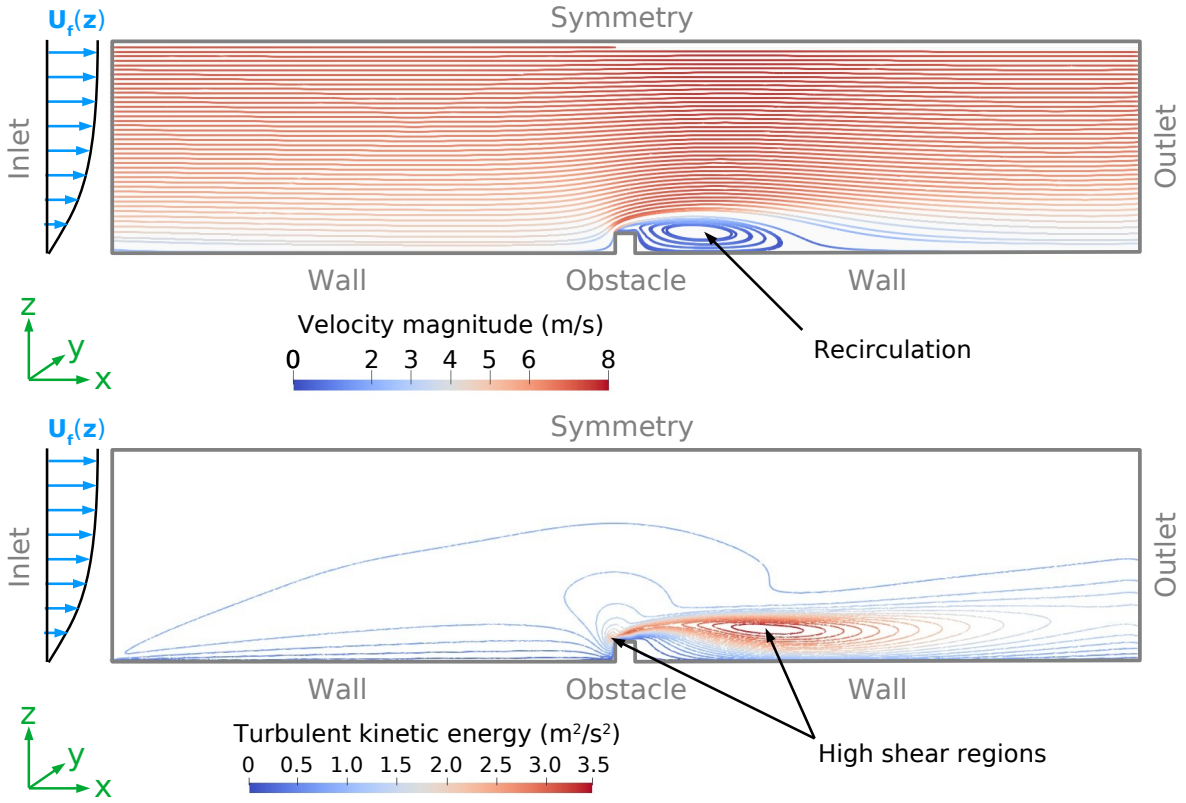


Figure 16: Flow around an obstacle within a boundary layer: sketch of the geometry used (including boundary conditions) together with the isocontours of the fluid velocity (top) and the isocontours of the turbulent kinetic energy (bottom).

model) and using the open-source CFD software *Code_Saturne* [63].

Fluid flow. Numerical simulations of the fluid phase have been run until a statistically steady-state was reached. The results of the fluid velocity and of the turbulent kinetic energy are shown in Fig. 16, which shows several features. First, the isocontours of the magnitude of the fluid velocity reveal the complex flow structure, especially in the wake of the obstacle where a recirculation region is clearly visible (actually a similar but smaller recirculation occurs just above the obstacle). Second, the isocontours of the turbulent kinetic energy (obtained by computing half of the trace of the Reynolds stress tensor in the simulation) show high levels of turbulent kinetic energy just before the obstacle and in the region separating the main longitudinal flow from the recirculation region in the wake of the obstacle. These two regions actually correspond to high-shear regions. These features demonstrate the strongly non-homogeneous flow that results from the impact of an incoming flow stream on the obstacle.

Particle tracking. Particles are then tracked within this steady fluid flow (i.e. the fluid flow computed previously is used for initialization and remains fixed during particle tracking). For that purpose, we resort to the stochastic Lagrangian module implemented in the open-source CFD software *Code_Saturne* (details about the stochastic Lagrangian model are available in [64]). Particles with a fixed diameter of 10 or 100 μm and a density of 1000 kg.m^{-3} are continuously injected through the inlet boundary. The number of particles injected every time step has been varied between 5 and 50. Injected particles are uniformly distributed across the whole injection area, independently of the velocity profile. This particle injection method leads to a higher number of particles near the wall: particles in the upper region move indeed faster than particles near the surface. As a result, the number concentration of particles before the obstacle is not a uniform distribution (as would be expected from an injection that respects the flow rate). This injection method has been retained here since it allows to test if the D2SD algorithm is able to detect non-uniform number concentrations in complex geometries. In line with the boundary conditions for the fluid phase, particles are assumed to rebound on the bottom wall surfaces (meaning that no deposition occurs) while a symmetry condition is applied on the top boundary and an outlet condition is imposed on the downstream boundary. The simulation is ran using a fixed time step of 0.01 s for 20 s. The simulation time has been chosen to be long enough so that a stationary regime is obtained for the number of particles in the domain. This regime is typically reached after 15 s and the total number of particles in the domain varies between roughly 8,300 particles (for 10 particles injected per time step) and 42,000 (for 50 particles injected per time step). A snapshot of particle positions is displayed in Fig. 17, where several features can be seen: first, in the region upstream of the obstacle, the particles are uniformly distributed along the longitudinal direction while the particle concentration decreases linearly with the distance from the wall (this is due to the injection of particles uniformly on the inlet area, regardless of the velocity profile); second, 100 μm particles tend to accumulate just before and around the obstacle while they are much less present in the wake of the obstacle; third, 10 μm particles appear to be much more homogeneously distributed throughout the domain. These observations are actually mainly related to the particle inertia: small 10 μm particles behave almost like fluid particles due to their low inertia (meaning that they do not accumulate in non-homogeneous fluid regions) while particles with a high inertia tend to accumulate in regions around the obstacle and deplete low-energy regions (e.g. the recirculation in the wake of the obstacle).

Agglomeration. Drawing on these particle-laden flow simulations, we now investigate the role of particle agglomeration in such complex cases. For that purpose, the `quick_D2SD` algorithm has been coupled to the CFD simulations to analyse the spatial repartition of particles in the domain and compute the corresponding number of collisions. The number of collisions obtained is compared to the one obtained with a mean-field formula considering either the whole domain as a single cell or using the mesh used in the CFD computation.

As mentioned in the introduction, one of the difficulties that arise when computing particle collision and agglomeration in complex flows lies in the definition of the collision frequency for a mean-field approach [39]. In the present case, since no information is available experimentally or numerically for the collision frequency, we first consider a simple case to illustrate the interest of the D2SD approach: particles are assumed to agglomerate only through Brownian motion (i.e. independently of the local flow properties). In that case, the collision frequency between particles of sizes R_i and R_j is simply given by [39, 2]:

$$\beta = \frac{4\,k_B\,T_f}{3\,\mu_f}\frac{\left(R_i + R_j\right)^2}{R_i\,R_j}, \tag{15}$$

where T_f is the fluid temperature, k_B the Boltzmann constant, μ_f the fluid kinematic viscosity. Since the fluid temperature is homogeneous across the whole domain, the collision frequency is constant and the number of collisions is only driven by the spatial repartition of particles. In the present case, we have retained a value of $\beta = 5 \times 10^{-8}$ for purely illustrative purposes (which corresponds to a fixed fluid temperature across the whole fluid domain).

The spatial decomposition obtained with the D2SD algorithm for both sizes (10 and 100 μm) is shown in Figure 18 for 20 particles injected per time step: the void region in the wake of the obstacle is clearly visible in the case of large inertial particles. The results for particle agglomeration are summarised in Table 3. Several conclusions can be drawn from these results. First, the computation of collisions using the mesh used for the CFD calculation is not appropriate and leads here to an overestimation of the number of agglomeration events. This is due to the fact that the mesh has been designed to capture the flow field around the obstacle, leading to very small sizes (here squares of 2.5 mm with a thickness of 100 mm). As a result,

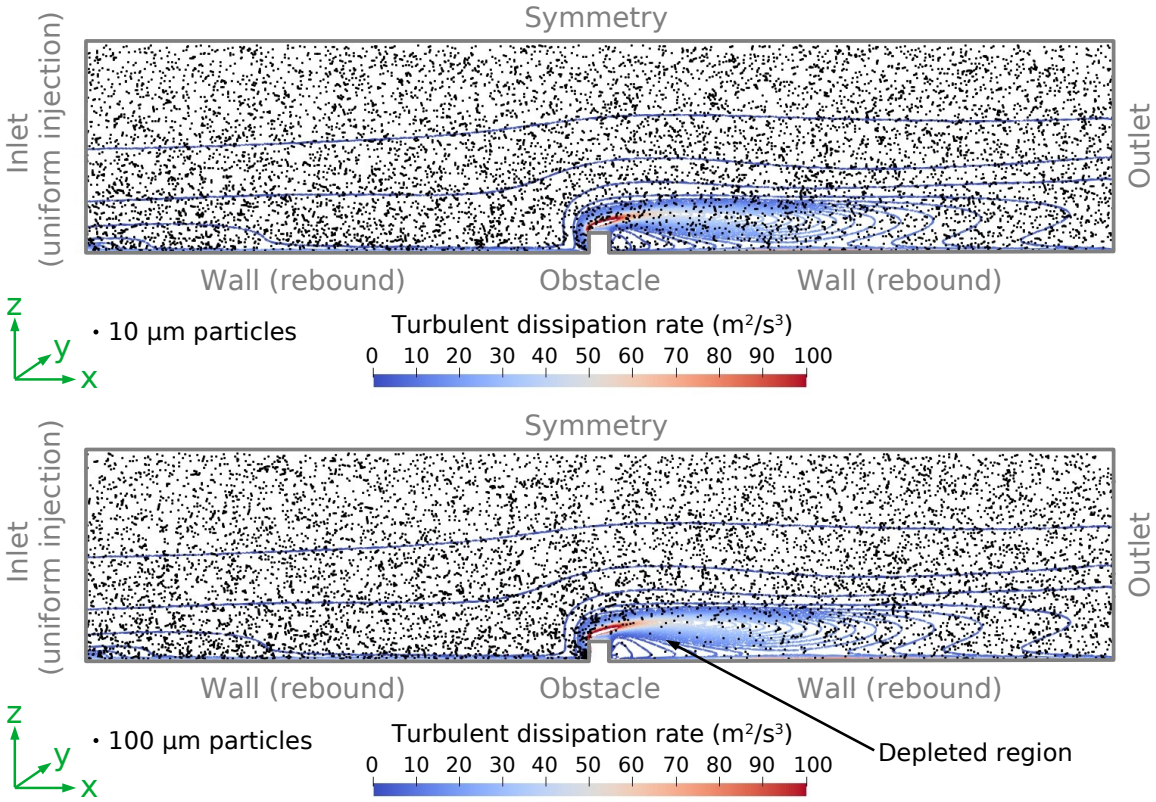
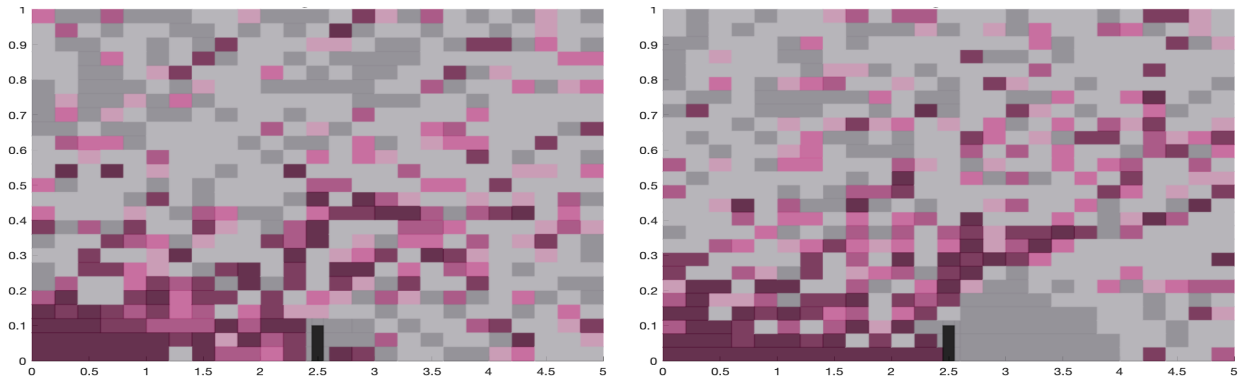


Figure 17: Two-phase flow simulations around an obstacle within a boundary layer: snapshot of the positions of 10 μm particles (top) and 100 μm particles (bottom) at a given time (here 20 s) together with the isocontours of the fluid turbulent dissipation rate.



(a) 10 μm particles

(b) 100 μm particles

Figure 18: Mesh obtained from the `quick_D2SD` applied to a turbulent particle-laden flow around an obstacle (black rectangle in both graphs): case of 10 μm particles with 20 particles injected each time step (left); case of particles of 100 μm particles (right). Levels are as in Fig. 2.

most cells are actually populated by less than 1 particle, meaning that agglomeration can only occur in the cells where two or more particles are present. However, when such a situation happens, the small cell volume immediately leads to a local but significant over-estimation of the number of collision events. Adding all contributions of these cells with two or more particles leads to an over-estimation of the number of events. At this point, it is worth noting that the results using the CFD mesh could also severely under-estimate the actual number (if all cells were not populated by more than one particle). Second, the number of collisions calculated using `quick_D2SD` is the same as the one with the `full_D2SD` algorithm. This further confirms the accuracy and efficiency of the `quick_D2SD`algorithm compared to the `full_D2SD` algorithm. Besides, the average number of particles present in a single D2SD cell is roughly equal to 600, meaning that a large enough sample is used when computing the number of agglomeration events (this contributes to lower the computational error). Third, the efficiency of the algorithm can be analysed by comparing the computational times. This is displayed in Table 3, which provides the ratio `CPU/CPU*` of computational times for each method, considering `CPU*` the computational time for the mean-field approach using the CFD mesh as a referential time. From this, we can observe that the `quick_D2SD` algorithm reduces the computational cost by $74-78\%$ with respect to its full version. On the other hand, the `quick_D2SD` algorithm has an associated cost approximately 11 times the cost using a mean-field approach on CFD mesh. Naturally, the computation of the number of agglomerations using the CFD mesh is faster than the computation after the construction of any dedicated mesh; nevertheless, we have already seen that this advantage in computational cost comes with a significant degradation of the accuracy of the computed number of aggregations. Finally, it is worth pointing out that the results obtained with the `quick_D2SD` algorithm is comparable to the one obtained with a single cell (which is roughly 4.000 times faster). With such a result, it can be tempting to resort to a single cell. However, the issue of a single cell is that it does not provide information on where the aggregates will be formed. In fact, with a single cell, aggregates can be formed anywhere in the domain with the same probability. In reality, the formation of aggregates does depend on the local number concentration (aggregates should never be formed in void regions, but should be formed more frequently in regions with a higher-than-average number concentration). This information will be key in long-term simulations, i.e. when such models are used to predict the evolution of the aggregate sizes with time.

To further analyse how the results evolve with the number of particles present in the domain, we compare the simulations obtained with 1 to 50 particles injected every time step. The results are summarised in Table 4. First, it can be seen that the estimations using the D2SD are generally similar to the one with a mean-field approach over a single cell covering the whole domain, although slightly higher. This can be attributed to the fact that the particle repartition in the domain only weakly deviates from the uniformity assumption for 10 μm particles and that particles are mostly homogeneously distributed in the vast majority of the domain (except near the obstacle especially

CASE	`full_D2SD`	`quick_D2SD`	**Mean-field on a single cell**	**Mean-field on CFD mesh**
10 μm size				
Mean agglo. $\mathcal{A}$	29.9	30.0	28.5	58.3
`CPU/CPU*`	42.47	11.11	$2.90e-3$	1
100 μm size				
Mean agglo. $\mathcal{A}$	33.2	30.1	28.0	61.8
`CPU/CPU*`	54.25	11.87	$6.23e-3$	1

Table 3: Values of the mean number of agglomeration events per time step computed using `full_D2SD` or `quick_D2SD` with re-meshing, the mean-field formula over the CFD mesh and the mean-field formula over the whole domain (one cell only). Additionally, we include the ratio `CPU/CPU*` of computational times for each method, considering `CPU*` the computational time for the mean-field on CFD mesh, as referential time. The value of the agglomeration kernel is $\beta = 5 \times 10^{-8}$.

for 100 μm particles). We also know from our previous paper that the violation of the uniformity assumption can lead to mis-estimations of the agglomeration [50, 49]. Second, the evolution with the number of particles tends to follow the same trend as the one with a mean-field approach over a single cell (i.e. proportional to N^2). For `quick_D2SD`, the largest deviation from this law occurs for the lowest number of particles. It is probably related to the fact that the numerical error increases as the number of particles decreases (since there are less particles per cell). For the mean-field approach on the CFD mesh, the evolution also deviates from the N^2 law at low numbers of particles. This is due to a purely numerical artifact: more and more cells are actually populated by 1 or less particles (meaning that agglomeration does not occur in these cells) leading to a mesh-dependent result. At this stage, it is also important to recall that we compare here an average number of collision events obtained with a single cell with those obtained using the D2SD-mesh. In the present case, the number of particles in the whole domain (typically around 10^4 up to 10^5) might appear to be quite low. This will undoubtedly result in statistical fluctuations due to the low number of particles. However, in reality, when running simulations to compute the evolution of aggregate sizes with time, the statistical error made in each cell will be reduced due to the averaging of the results in each cell over a large number of temporal iterations. For that reason, we have decided not to increase the number of particles in the domain here (which can also be increased by adapting the D2SD algorithm to treat parcels instead of particles).

A more realistic choice for the collision kernel will be to resort to the kernel for the collision between particles in a turbulent flow. One possibility is then to use the kernel derived by Saffman and Turner for collisions in homogeneous isotropic turbulence [39, 2]:

$$\beta = \frac{4}{3}\sqrt{\frac{\epsilon}{\nu_f}}\left(R_i + R_j\right)^3, \tag{16}$$

where ν_f is the fluid dynamic viscosity and ϵ the turbulent dissipation rate. In that case, the collision kernel is not homogeneous across the whole simulation domain since the turbulent dissipation rate is not constant (see the isocontours of ϵ on Fig. 17).

CASE		quick_D2SD	Mean-field on a single cell	Mean-field on CFD mesh
Part. size	Nb. injected	Mean agglo. $\mathcal{A}$		
10 μm	1	0.06	0.07	0.11
	5	1.91	1.79	3.93
	10	7.48	7.06	14.0
	20	30.0	28.5	58.3
	50	187.3	179.4	359.8
100 μm	1	0.08	0.07	0.26
	5	1.89	1.71	3.96
	10	7.48	6.89	14.2
	20	30.1	28.0	61.8
	50	186.8	175.3	380.1

Table 4: Comparison of the values of the mean number of agglomeration events $\mathcal{A}$ per time step obtained for simulations with 1, 5, 10, 20 and 50 particles injected every time step. Values are computed using the quick_D2SD algorithm with re-meshing, the mean-field approach over the CFD mesh and the mean-field approach over the whole domain (one cell only). Values of the agglomeration kernel are $\beta = 5 \times 10^{-8}$ for 10 μm and 100 μm particles.

This means that applying the faster D2SD algorithm — as it has been described — is not enough to properly define spatial regions with homogeneous repartition of particles and homogeneous fluid properties. In practice, this implies that the D2SD algorithm should also include requirements for a homogeneous turbulent dissipation rate here. A first option to respect this additional constraint is to apply a two-stage D2SD algorithm: one stage for spatial-homogeneity and a second stage for homogeneity in terms of fluid properties (here the energy dissipation rate). Another option would be to further split the spatial regions defined by the D2SD algorithm according to the isocontours of the turbulent dissipation rate. Such developments are left out of the scope of the present paper and will be performed later. Nevertheless, the main conclusion that can be drawn from this example is the following: computing particle collision and/or agglomeration in complex flows with hybrid Euler/Lagrange approaches requires extensive developments to identify regions with both a uniform spatial repartition of particles and homogeneous fluid properties (or whatever properties are actually required as inputs in the collision frequency α).

5. Conclusion

In this paper, a fast version of the data-driven spatial decomposition (D2SD) algorithm has been developed. As for the original D2SD algorithm, the fast D2SD allows to detect non-homogeneous particle concentrations within a regular volume, relying solely on the information coming from the set of particles (without requiring other input parameters). The quick_D2SD algorithm developed here combines two simplifications to reduce the computational costs associated with the full_D2SD algorithm. First, a simplified decomposition is applied with an a priori evaluation of the PDF-threshold that is used to classify regions according to their concentration level, which is taken here as $\hat{\lambda} = Q_{60}$. Second, a re-meshing criterion is used to avoid applying the spatial decomposition algorithm every time step in a CFD simulation. More precisely, the re-meshing criterion is composed of two tests to check if the spatial decomposition becomes too different from the one obtained at a previous time step: one test is based on the fluctuation of the dispersion of the particles within the domain $\mathcal{D}$ and another test is based on the variation of the particle concentration.

The accuracy and efficiency of the proposed adaptations of the original D2SD algorithm have been tested on a number of situations involving a range of initial distribution of particles (homogeneous or non-homogeneous) and various particle motion representing practical situations, mostly 2-dimensional examples for visualisation reasons. These test-cases allowed to set-up the thresholds (ϵ_1, ϵ_2) that are required in the quick_D2SD algorithm, and analyse their effect on the overall accuracy. In particular, the results obtained support the use of a re-meshing criterion based on two indicators coupled to a faster D2SD algorithm, since it accurately reproduces the results obtained with the original D2SD algorithm applied every time step, even in complex situations.

This faster D2S2 algorithm has been tested in two 3D cases. First, the case of diffusive particles injected locally within a spherical domain has been considered as a validation case. For that purpose, the results obtained with the D2SD algorithm have been compared to the results given by fine Lagrangian simulations (based on a Langevin model). It was shown in particular that the D2SD algorithm allows to significantly improve the prediction of the number of agglomeration events compared to a simple evaluation using a mean-field formulation over the entire domain (which severely underestimates the number of agglomeration events). As illustrated in [50], similar results are expected (underestimating or overestimating the number of agglomeration events) when an arbitrary mesh is chosen for the collision step. Second, the flow around an obstacle has been used to illustrate the interest of the method in practical CFD cases involving complex geometries. It was shown that the results obtained with the quick_D2SD algorithm are less dependent on the mesh used (compared to the CFD mesh or to a single-cell) and that the computational costs of the quick_D2SD become more compatible with other CFD computations. We also mention that D2SD can be adapted to non-Cartesian geometry as previously discussed in [50].

These results support the need to apply spatial decomposition techniques when resorting to hybrid Euler/Lagrange approaches coupled to mean-field approaches for the computation of particle agglomeration (such as the mean-field approach). However, these results also shed light on the limitations of such mean-field approaches: mean-field formulations indeed require that the population of particles is uniformly distributed in the volume considered and that other properties are constant (e.g. fluid velocity or turbulent dissipation as seen in Section 4.2), while neglecting any spatial/temporal correlation between collisions. Despite the apparent simplicity of mean-field formulations, these simulations show that complex algorithms have to be designed in order to use such formulations in CFD simulations of complex non-homogeneous flows.

Besides, in order to consider only the error coming from the

selection of the mesh, we have considered only the treatment of primary particles in the present paper. However, the D2SD algorithm (and therefore its simplified versions) can be easily adapted to the assessment of parcels. In the latter case, the steps to be adapted will be those in which the probability density function is approximated, considering the statistical weight of each parcel as if the composing particles were present in the same cell. All other steps of the algorithm remain the same. Further developments are also needed to be able to use complex non-Cartesian meshes and possibly unstructured meshes. Such developments will be key when applying the method to realistic cases, where the number of real particles is so high (e.g. 10^{12} or more) that it is not appropriate to explicitly track the motion of each individual particles but rather to resort to the tracking of parcels.

Appendix A. Statistical uniformity test

The first step of D2SD consists of checking if the input sample $\mathbf{X}$ corresponds to the case of uniformly-distributed particles, i.e. whether or not we need an appropriate spatial decomposition. For that purpose, several statistical uniformity tests (considered in the classical literature of goodness of fit) could be applied. However, any uniformity test will naturally have a statistical error associated with it, stemming from the randomness of the observed sample $\mathbf{X}$, and the sample size n. In order to reduce the number of false positives (the number of times we decide to decompose the domain given that the particles are actually uniformly distributed), we apply a set of uniformity tests, and see if enough of them reject the uniformity of $\mathbf{X}$. The total number of tests applied is optional. In our case, after some numerical experiments performed in [50], we consider the set: three different discrepancy tests [65] (used for measuring if a given set of points is uniformly scattered) and a Henze-Zirkler normality test [66] applied to the sample $\mathbf{Y} := \left\{\Phi^{-1}(X^1), \Phi^{-1}(X^2), \ldots, \Phi^{-1}(X^n)\right\}$, with Φ the cumulative distribution function of a standard normal random variable. The latter is based on the inverse transformation sampling method, where, in order to generate random numbers from any probability distribution with cumulative distribution function (CDF) F, the transformation $F^{-1}(\mathcal{U}[0,1]^d)$ is considered, with $\mathcal{U}[0,1]^d$ being the uniform distribution on the unitary cube $[0,1]^d$. Then, for $F = \Phi$, if $\mathbf{X}$ is uniformly distributed in $\mathcal{D}$, we would have that $\mathbf{Y}$ is a sample from a standard Gaussian random variable. The reason for considering Gaussian variables is that, given its usefulness in applications, it has taken most of the attention in the multidimensional case, compared to specialised tests for uniformity. Additionally, a Pearson test (see e.g. [67]) is applied in order to check the independence of the sample (more details can be found in [50]).

Then, considering the results of the different tests and their associated uncertainty, a majority voting criterion is implemented, i.e. we accept the spatially-uniform condition for $\mathbf{X}$ if more than half of the tests (in our case -at least- three tests) accept the uniformity.

Appendix B. The measure score-function to uniformity

In order to find the optimal threshold λ^*, we consider a measure, henceforth called the *score function*, that quantifies how far a sample is from being uniformly distributed. For that purpose, we resort to the score function defined by the distance of $\mathbf{X}$ to the boundary of the domain $\mathcal{D} = [0,1]^d$ [68], remarkably used for testing multivariate uniformity:

$$d_b(\mathbf{X}, \mathcal{U}) := \int_0^1 |G_{n,\mathbf{Z}}(z) - H_{\partial\mathcal{D},\mathcal{U}}(z)|\, dz, \tag{B.1}$$

where $G_{n,\mathbf{Z}}$ stands for the empirical cumulative distribution function of the sample $\mathbf{Z} = \{Z^1, Z^2, \ldots, Z^n\} \in \mathbb{R}^n$, with $Z^i = 2d(X^i, \partial\mathcal{D})$, and $H_{\partial\mathcal{D},\mathcal{U}}(z) = 1-(1-z)^d$, for $0 \leq z \leq 1$. Here, $d(X^i, \partial\mathcal{D})$ denotes the Euclidean distance between the point X^i in $\mathcal{D}$ and the boundary of the domain $\partial\mathcal{D}$ (see [68] for further details).

The ideal or reference score $S_{\texttt{ideal}}$ is obtained by averaging the score associated with a family of uniformly distributed sample $\mathbf{U}$ of size n. The λ^* search is a minimization problem aims to decrease the distance between the ideal score value $S_{\texttt{ideal}}$ and the score $S(\lambda) = d_b(\mathbf{X}^\lambda, \mathcal{U})$ associated with the synthetic sample construct in `step2` ofthe D2SD algorithm:

$$\lambda^* = \texttt{argmin}_\lambda \mathbf{E}(\mathbf{X}^\lambda). \tag{B.2}$$

The associated error to minimize is

$$\mathbf{E}(\mathbf{X}^\lambda) = \frac{|S_{\texttt{ideal}} - \langle S(\lambda)\rangle_{\text{synthetic}}|}{S_{\texttt{ideal}}},$$

where the bracket $\langle S(\lambda)\rangle_{\text{synthetic}}$ denotes the average of the score over several samples of the synthetic observations. The latter is done to account for the uncertainty of ghost particle injection.

Appendix C. Collision rate of Brownian particles

Let us call $N(t)$ the number of particles present in a spherical domain at time t for a given realisation of the initial positions and of the noises. The mean number of collisions n_{coll} between times t and $t + \Delta t$ is given by the average number of pairs of particles that approach each other within a distance equal to the sum of their radii, that is $2R_p$.

Suppose that there is a reference particle i at $\mathbf{X}_i(t)$ and another one, j, at $\mathbf{X}_j$ that approaches i with a velocity $\mathbf{V}_j(t) - \mathbf{V}_i(t)$ given by Eq 13. As displayed in Fig. C.20, particle j will collide with i between t and $t + \Delta t$ if two conditions are satisfied: (i) the radial component of its relative velocity along the line connecting the two spheres is negative $W = [\mathbf{V}_j(t) - \mathbf{V}_i(t)] \cdot \mathbf{r}/|\mathbf{r}|$ (with $\mathbf{r} = \mathbf{X}_j(t) - \mathbf{X}_i(t)$); (ii) particle j is located at a distance less than $2R_p + |W|\Delta t$ from i. Assuming that particles are uniformly distributed in the sphere $[0, R_s]$, the probability that particle j collides with i given its velocity difference $W < 0$ is given by

$$\text{Proba(coll} \mid W) = \frac{4\pi\,(2R_p)^2\,|W|\,\Delta t}{\texttt{Vol}(\mathcal{D})}. \tag{C.1}$$

with $\texttt{Vol}(\mathcal{D}) = 4\pi R_s^3/3$ the volume of the domain.

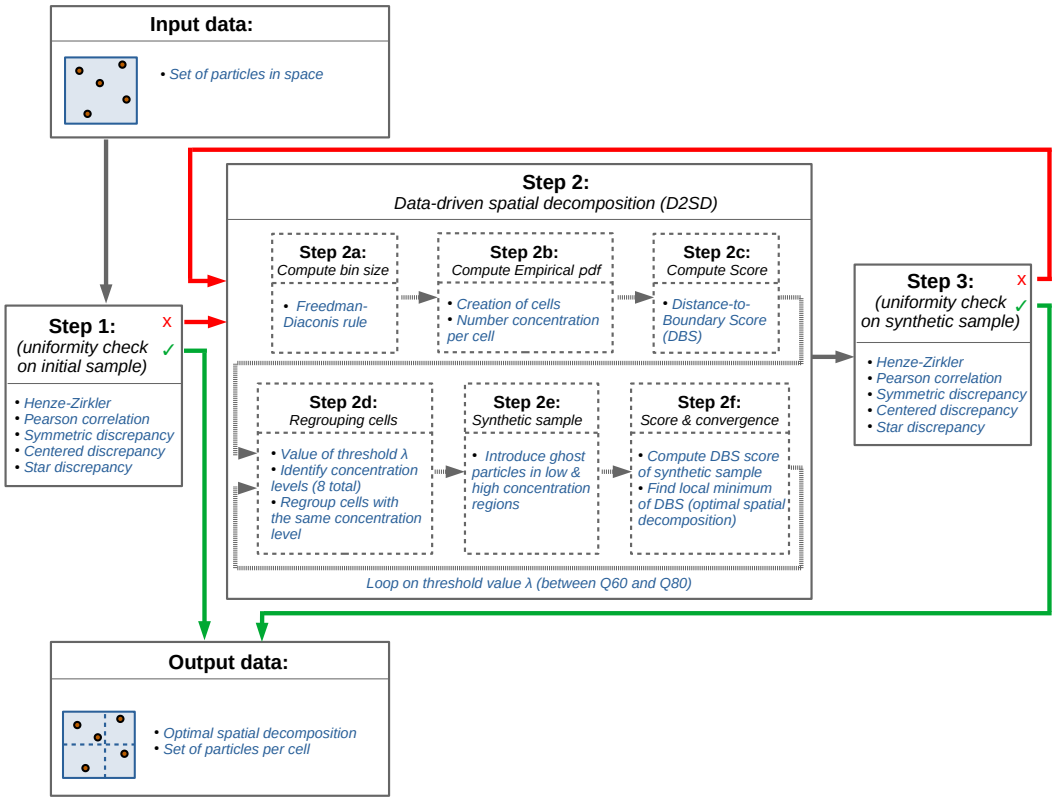


Figure B.19: Sketch of the D2SD algorithm. Figure taken from [50].

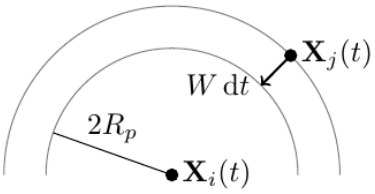


Figure C.20: 2D sketch of the criterion for a particle j to collide with a particle i fixed at the center.

The probability that particle i has a collision between t and $t + \Delta t$ can be written by summing over all particles $j \neq i$ present at time t and averaging with respect to the velocity difference W. Assuming that particle positions and velocities are completely independent, one obtains the probability $\mathcal{P}^i_{coll\in[t,t+\Delta t]}$ that particle i collides between t and $t + \Delta t$

$$\mathcal{P}^i_{coll\in[t,t+\Delta t]} = -\frac{16\,\pi\,R_p^2}{\mathtt{Vol}(\mathcal{D})}\,(N(t)-1)\,\langle W\,\theta(-W)\rangle\,\Delta t, \qquad \text{(C.2)}$$

where θ denotes the Heaviside function and $\langle\cdot\rangle$ ensemble averages. For velocities following a Langevin process with friction γ and noise amplitude σ, the velocity difference W is a Gaussian process with variance σ^2/γ. One then obtains $\langle W\,\theta(W<0)\rangle = -\sigma/\sqrt{2\pi\gamma}$. As a result, the mean number of collisions $n_{coll\in[t,t+\Delta t]}$ occurring between t and $t + \Delta t$ reads

$$\begin{aligned} n_{coll\in[t,t+\Delta t]} &= \frac{1}{2}\sum_i \mathcal{P}^i_{coll\in[t,t+\Delta t]} \\ &= \frac{8\,\pi\,R_p^2\,\sigma}{\mathtt{Vol}(\mathcal{D})\,\sqrt{2\pi\gamma}}\,N(t)\,(N(t)-1)\,\Delta t. \end{aligned} \qquad \text{(C.3)}$$

We consider the case where each collision leads to the removal of one particle from the simulation. This leads to the following evolution equation for the average number of particles:

$$\frac{\mathrm{d}}{\mathrm{d}t}\,\langle N(t)\rangle = -\frac{\beta}{\mathtt{Vol}(\mathcal{D})}\,\langle N(t)\,(N(t)-1)\rangle \qquad \text{(C.4)}$$

with the collision rate (or collision frequency) β defined by

$$\beta = \frac{8\pi\,R_p^2\,\sigma}{\sqrt{2\pi\gamma}}. \qquad \text{(C.5)}$$

Authors' contributions

The present study was conceived, designed and analysed by KMR, MB and CH, based on the algorithm initiated in [50]. KMR has carried out the adaptation of the D2SD algorithm to CFD simulations as well as testing in simple cases. KMR and CH have carried out the 3D simulations (Langevin system) and the coupling with the quick D2SD algorithm. KMR, MB and CH performed the data analysis. KMR, CH and MB drafted the manuscript. All authors read and approved the manuscript.

Data access

The data that support the findings of this study are available from the corresponding author on request.

Acknowledgement

We acknowledge Jérémie Bec, Martin Ferrand and Jean Pierre Minier for useful and constructive discussions on the method validation. The authors are grateful to the OPAL infrastructure from Université Côte d'Azur for providing resources and support. KMR acknowledges the support of ANID FONDECYT/POSTDOCTORADO N°321011.

References

[1] A. D. McNaught, A. Wilkinson, IUPAC: Compendium of chemical terminology, 2nd ed., online version (2019-) created by S. J. Chalk. Edition, Blackwell Scientific Publications, Oxford, 1997. doi.org/10.1351/goldbook. 1

[2] M. Elimelech, J. Gregory, X. Jia, Particle deposition and aggregation: measurement, modelling and simulation, Butterworth-Heinemann, 2013. doi.org/10.1016/B978-0-7506-7024-1.X5000-6. 1, 18, 20

[3] R. H. Meade, Transport and deposition of sediments in estuaries, Geological Society of America 133 (1) (1972) 91–120. doi.org/10.1130/MEM133-p91. 1

[4] K. Gotoh, Y. Fujii, A fractal dimensional analysis on the cloud shape parameters of cumulus over land, Journal of Applied Meteorology 37 (10) (1998) 1283–1292. doi.org/10.1175/1520-0450(1998)037<1283:AFDAOT>2.0.CO;2. 1

[5] G. Falkovich, A. Fouxon, M. Stepanov, Acceleration of rain initiation by cloud turbulence, Nature 419 (6903) (2002) 151. doi.org/10.1038/nature00983. 1

[6] R. A. Shaw, Particle-turbulence interactions in atmospheric clouds, Annual Review of Fluid Mechanics 35 (1) (2003) 183–227. doi.org/10.1146/annurev.fluid.35.101101.161125. 1

[7] J. Blum, G. Wurm, The growth mechanisms of macroscopic bodies in protoplanetary disks, Annual Review of Astronomy and Astrophysics 46 (2008) 21–56. doi.org/10.1146/annurev.astro.46.060407.145152. 1

[8] M.-F. Pouet, A. Grasmick, Urban wastewater treatment by electrocoagulation and flotation, Water Science and Technology 31 (3-4) (1995) 275–283. doi.org/10.1016/0273-1223(95)00230-K. 1

[9] J. Rubio, M. Souza, R. Smith, Overview of flotation as a wastewater treatment technique, Minerals Engineering 15 (3) (2002) 139–155. doi.org/10.1016/S0892-6875(01)00216-3. 1

[10] M. Bartels, W. Lin, J. Nijenhuis, F. Kapteijn, J. R. Van Ommen, Agglomeration in fluidized beds at high temperatures: Mechanisms, detection and prevention, Progress in Energy and Combustion Science 34 (5) (2008) 633–666. doi.org/10.1016/j.pecs.2008.04.002. 1

[11] L. Gallen, A. Felden, E. Riber, B. Cuenot, Lagrangian tracking of soot particles in LES of gas turbines, Proceedings of the Combustion Institute 37 (4) (2019) 5429–5436. doi.org/10.1016/j.proci.2018.06.013. 1

[12] N. Maximova, O. Dahl, Environmental implications of aggregation phenomena: current understanding, Current Opinion in Colloid & Interface Science 11 (4) (2006) 246–266. doi.org/10.1016/j.cocis.2006.06.001. 1

[13] D. Henning, R. Baer, A. Hassan, R. Dave, Major advances in concentrated and dry milk products, cheese, and milk fat-based spreads, Journal of Dairy Science 89 (4) (2006) 1179–1188. doi.org/10.3168/jds.S0022-0302(06)72187-7. 1

[14] S. B. Pope, Turbulent flows, Cambridge University Press, 2000. 1

[15] C. Henry, J.-P. Minier, G. Lefèvre, Towards a description of particulate fouling: From single particle deposition to clogging, Advances in Colloid and Interface Science 185 (2012) 34–76. doi.org/10.1016/j.cis.2012.10.001. 1

[16] S. Subramaniam, Lagrangian–Eulerian methods for multiphase flows, Progress in Energy and Combustion Science 39 (2-3) (2013) 215–245. doi.org/10.1016/j.pecs.2012.10.003. 1, 2

[17] M. Sommerfeld, L. Pasternak, Advances in modelling of binary droplet collision outcomes in sprays: a review of available knowledge, International Journal of Multiphase Flow 117 (2019) 182–205. doi.org/10.1016/j.ijmultiphaseflow.2019.05.001. 1, 2

[18] M. Chen, K. Kontomaris, J. McLaughlin, Direct numerical simulation of droplet collisions in a turbulent channel flow. Part I: collision algorithm, International Journal of Multiphase Flow 24 (7) (1999) 1079–1103. doi.org/10.1016/S0301-9322(98)00007-X. 2

[19] H. Sigurgeirsson, A. Stuart, W.-L. Wan, Algorithms for particle-field simulations with collisions, Journal of Computational Physics 172 (2) (2001) 766–807. doi.org/10.1006/jcph.2001.6858. 2

[20] M. Breuer, N. Almohammed, Modeling and simulation of particle agglomeration in turbulent flows using a hard-sphere model with deterministic collision detection and enhanced structure models, International Journal of Multiphase Flow 73 (2015) 171–206. doi.org/10.1016/j.ijmultiphaseflow.2015.03.018. 2

[21] N. Almohammed, M. Breuer, Modeling and simulation of agglomeration in turbulent particle-laden flows: A comparison between energy-based and momentum-based agglomeration models, Powder Technology 294 (2016) 373–402. doi.org/10.1016/j.powtec.2015.12.034. 2

[22] J. Bec, S. S. Ray, E. W. Saw, H. Homann, Abrupt growth of large aggregates by correlated coalescences in turbulent flow, Physical Review E 93 (3) (2016) 031102. doi.org/10.1103/PhysRevE.93.031102. 2, 3

[23] J.-P. Minier, E. Peirano, The pdf approach to turbulent polydispersed two-phase flows, Physics Reports 352 (1-3) (2001) 1–214. doi.org/10.1016/S0370-1573(01)00011-4. 2

[24] J.-P. Minier, On Lagrangian stochastic methods for turbulent polydisperse two-phase reactive flows, Progress in Energy and Combustion Science 50 (2015) 1–62. doi.org/10.1016/j.pecs.2015.02.003. 2

[25] J.-P. Minier, Statistical descriptions of polydisperse turbulent two-phase flows, Physics Reports 665 (2016) 1–122. doi.org/10.1016/j.physrep.2016.10.007. 2

[26] M. Sommerfeld, Validation of a stochastic Lagrangian modelling approach for inter-particle collisions in homogeneous isotropic turbulence, International Journal of Multiphase Flow 27 (10) (2001) 1829–1858. doi.org/10.1016/S0301-9322(01)00035-0. 2

[27] C. A. Ho, M. Sommerfeld, Modelling of micro-particle agglomeration in turbulent flows, Chemical Engineering Science 57 (15) (2002) 3073–3084. doi.org/10.1016/S0009-2509(02)00172-0. 2

[28] O. L. Sgrott, M. Sommerfeld, Influence of inter-particle collisions and agglomeration on cyclone performance and collection efficiency, The Canadian Journal of Chemical Engineering 97 (2) (2019) 511–522. doi.org/10.1002/cjce.23371. 2

[29] C. Henry, J.-P. Minier, M. Mohaupt, C. Profeta, J. Pozorski, A. Tanière, A stochastic approach for the simulation of collisions between colloidal particles at large time steps, International Journal of Multiphase Flow 61 (2014) 94–107. doi.org/10.1016/j.ijmultiphaseflow.2014.01.007. 2

[30] C. Henry, J.-P. Minier, J. Pozorski, G. Lefèvre, A new stochastic approach for the simulation of agglomeration between colloidal particles, Langmuir 29 (45) (2013) 13694–13707. doi.org/10.1021/la403615w. 2

[31] P. J. O'Rourke, Collective drop effects on vaporizing liquid sprays, Tech. rep., Los Alamos National Lab., NM (USA) (1981). 2

[32] M. Mezhericher, A. Levy, I. Borde, Probabilistic hard-sphere model of binary particle–particle interactions in multiphase flow of spray dryers, International Journal of Multiphase Flow 43 (2012) 22–38. doi.org/10.1016/j.ijmultiphaseflow.2012.02.009. 2, 3

[33] D. P. Schmidt, C. Rutland, A new droplet collision algorithm, Journal of Computational Physics 164 (1) (2000) 62–80. doi.org/10.1006/jcph.2000.6568. 2

[34] G. Bird, Approach to translational equilibrium in a rigid sphere gas, Physics of Fluids 6 (1963) 1518–1519. doi.org/10.1063/1.1710976. 2

[35] G. A. Bird, J. Brady, Molecular gas dynamics and the direct simulation of gas flows, Vol. 42, Clarendon Press Oxford, 1994. 2

[36] F. J. Alexander, A. L. Garcia, The direct simulation Monte Carlo method, Computers in Physics 11 (6) (1997) 588–593. doi.org/10.1063/1.168619. 2

[37] C. Henry, K. K. Norrfors, M. Olejnik, M. Bouby, J. Luetzenkirchen, S. Wold, J.-P. Minier, A refined algorithm to simulate latex colloid agglomeration at high ionic strength, Adsorption 22 (4-6) (2016) 503–515. doi.org/10.1007/s10450-015-9714-4. 2

[38] D. Ramkrishna, The status of population balances, Reviews in Chemical Engineering 3 (1) (1985) 49–95. doi.org/10.1515/REVCE.1985.3.1.49. 2

[39] D. Ramkrishna, Population balances: Theory and applications to particulate systems in engineering, Elsevier, 2000. 2, 3, 18, 20

[40] D. L. Marchisio, R. O. Fox, Solution of population balance equations using the direct quadrature method of moments, Journal of Aerosol Science 36 (1) (2005) 43–73. doi.org/10.1016/j.jaerosci.2004.07.009. 2

[41] V. Vikas, Z. J. Wang, A. Passalacqua, R. O. Fox, Realizable high-order finite-volume schemes for quadrature-based moment methods, Journal of Computational Physics 230 (13) (2011) 5328–5352. doi.org/10.1016/j.jcp.2011.03.038. 2

[42] D. Li, Z. Li, Z. Gao, Quadrature-based moment methods for the population balance equation: An algorithm review, Chinese Journal of Chemical Engineering 27 (3) (2019) 483–500. doi.org/10.1016/j.cjche.2018.11.028. 2

[43] Y. Liao, D. Lucas, E. Krepper, M. Schmidtke, Development of a generalized coalescence and breakup closure for the inhomogeneous MUSIG model, Nuclear Engineering and Design 241 (4) (2011) 1024–1033. doi.org/10.1016/j.nucengdes.2010.04.025. 2

[44] Y. Liao, Update to the MUSIG model in ANSYS CFX for reliable modelling of bubble coalescence and breakup, Applied Mathematical Modelling 81 (2020) 506–521. doi.org/10.1016/j.apm.2020.01.033. 2

[45] C. Yuan, F. Laurent, R. Fox, An extended quadrature method of moments for population balance equations, Journal of Aerosol Science 51 (2012) 1–23. doi.org/10.1016/j.jaerosci.2012.04.003. 2

[46] A. Buffo, M. Vanni, D. Marchisio, R. Fox, Multivariate quadrature-based moments methods for turbulent polydisperse gas–liquid systems, International Journal of Multiphase Flow 50 (2013) 41–57. doi.org/10.1016/j.ijmultiphaseflow.2012.09.005. 2

[47] G. Bhutani, P. R. Brito-Parada, Analytical solution for a three-dimensional non-homogeneous bivariate population balance equation—a special case, International Journal of Multiphase Flow 89 (2017) 413–416. doi.org/10.1016/j.ijmultiphaseflow.2016.11.005. 2

[48] D. Ramkrishna, M. R. Singh, Population balance modeling: current status and future prospects, Annual Review of Chemical and Biomolecular Engineering 5 (2014) 123–146. doi.org/10.1146/annurev-chembioeng-060713-040241. 2

[49] G. Kasper, On the coagulation rate of aerosols with spatially inhomogeneous particle concentrations, Journal of Colloid and Interface Science 102 (2) (1984) 560–562. doi.org/10.1016/0021-9797(84)90261-3. 3, 20

[50] K. M. Rodríguez, M. Bossy, R. Maftei, S. Shekarforush, C. Henry, New spatial decomposition method for accurate, mesh-independent agglomeration predictions in particle-laden flows, Applied Mathematical Modelling 90 (2020) 582–614. doi.org/10.1016/j.apm.2020.08.064. 3, 4, 5, 6, 8, 16, 20, 21, 22, 23

[51] D. P. Schmidt, C. J. Rutland, Reducing grid dependency in droplet collision modeling, Journal of Engineering for Gas Turbines and Power 126 (2) (2004) 227–233. doi.org/10.1115/1.1564066. 3

[52] S. Hou, D. P. Schmidt, Adaptive collision meshing and satellite droplet formation in spray simulations, International Journal of Multiphase Flow 32 (8) (2006) 935–956. doi.org/10.1016/j.ijmultiphaseflow.2006.02.013. 3

[53] P. Pischke, D. Cordes, R. Kneer, A collision algorithm for anisotropic disperse flows based on ellipsoidal parcel representations, International Journal of Multiphase Flow 38 (1) (2012) 1–16. doi.org/10.1016/j.ijmultiphaseflow.2011.09.002. 3

[54] J. Zhang, J. Mi, H. Wang, A new mesh-independent model for droplet/particle collision, Aerosol Science and Technology 46 (6) (2012) 622–630. doi.org/10.1080/02786826.2011.649809. 3

[55] P. Pischke, R. Kneer, D. P. Schmidt, A comparative validation of concepts for collision algorithms for stochastic particle tracking, Computers & Fluids 113 (2015) 77–86. doi.org/10.1016/j.compfluid.2015.01.018. 3

[56] A. Munnannur, R. D. Reitz, Comprehensive collision model for multidimensional engine spray computations, Atomization and Sprays 19 (7). doi:10.1615/AtomizSpr.v19.i7.10. 3

[57] F. Perini, R. D. Reitz, Improved atomization, collision and sub-grid scale momentum coupling models for transient vaporizing engine sprays, International Journal of Multiphase Flow 79 (2016) 107–123. doi.org/10.1016/j.ijmultiphaseflow.2015.10.009. 3

[58] S. Suo, M. Jia, H. Liu, T. Wang, Development of a new hybrid stochastic/trajectory droplet collision model for spray simulations in internal combustion engines, International Journal of Multiphase Flow 137 (2021) 103581. doi.org/10.1016/j.ijmultiphaseflow.2021.103581. 3

[59] D. Freedman, P. Diaconis, On the histogram as a density estimator: L2 theory, Probability Theory and Related Fields 57 (4) (1981) 453–476. doi:https://doi.org/10.1007/BF01025868. 4

[60] J. Xu, S. Pope, Assessment of numerical accuracy of PDF/Monte Carlo methods for turbulent reacting flows, Journal of Computational Physics 152 (1) (1999) 192–230. doi.org/10.1006/jcph.1999.6241. 7

[61] R. W. Hockney, J. W. Eastwood, Computer simulation using particles, crc Press, 2021. doi.org/10.1201/9780367806934. 7

[62] M. L. Bahlali, C. Henry, B. Carissimo, On the well-mixed condition and consistency issues in hybrid Eulerian/Lagrangian stochastic models of dispersion, Boundary-Layer Meteorology 174 (2) (2020) 275–296. doi.org/10.1007/s10546-019-00486-9. 16

[63] F. Archambeau, N. Méchitoua, M. Sakiz, Code saturne: A finite volume code for the computation of turbulent incompressible flows-industrial applications, International Journal on Finite Volumes 1 (1).hal-01115371. 18

[64] E. Peirano, S. Chibbaro, J. Pozorski, J.-P. Minier, Mean-field/pdf numerical approach for polydispersed turbulent two-phase flows, Progress in Energy and Combustion Science 32 (3) (2006) 315–371. doi.org/10.1016/j.pecs.2005.07.002. 18

[65] J.-J. Liang, K.-T. Fang, F. Hickernell, R. Li, Testing multivariate uniformity and its applications, Mathematics of Computation 70 (233) (2001) 337–355. doi.org/10.1090/S0025-5718-00-01203-5. 22

[66] N. Henze, B. Zirkler, A class of invariant consistent tests for multivariate normality, Communications in Statistics-Theory and Methods 19 (10) (1990) 3595–3617. doi.org/10.1080/03610929008830400. 22

[67] R. A. Fisher, Statistical methods for research workers, Breakthroughs in statistics. Springer, New York, NY (1992) pp. 66-70. doi.org/10.1007/978-1-4612-4380-9_6. 22

[68] J. R. Berrendero, A. Cuevas, F. Vázquez-grande, Testing multivariate uniformity: The distance-to-boundary method, Canadian Journal of Statistics 34 (4) (2006) 693–707. doi.org/10.1002/cjs.5550340409. 22