OpenFOAM® Journal

Volume 5 [Full Papers], Pages 148–179
ISSN: 2753-8168

Particle-Resolved Simulations of Flows through Sphere, Cylinder or Spherocylinder Assemblies: Automatic Workflow

Jelena Mačak*, Nicolas Fintzi, Lionel Gamet, Jean-Lou Pierson, and Mathieu Morin

IFP Energies nouvelles, Rond-point de l'échangeur de Solaize, 69360 Solaize, France
Email address: jelena.macak@protonmail.com

DOI: https://doi.org/https://doi.org/10.51560/ofj.v5.157
Results with version(s): OpenFOAM® v2306
Repository: –

Abstract. We developed an OpenFOAM® application for generation of tri-periodic assemblies of fixed non-overlapping particles, intended for direct numerical simulations with body-fitted unstructured meshes. The particles can be spherical, cylindrical or spherocylindrical, with random or pre-assigned positions and orientations, and mono- or polydisperse. The assemblies are optimized for meshing with `snappyHexMesh`: various meshing errors are minimized by using automatically generated edge meshes, as well as by controlling the interparticle distance and tangentiality to the boundaries. Further, we provide a new pressure boundary condition which improves the accuracy of the resulting hydrodynamic forces. The available post-processing function objects are extended to also calculate stresslets (i.e., resistance to the straining motion), relevant for rheology of suspensions. The workflow is validated against available analytical and numerical data, showing excellent agreement.

With our present contribution, an OpenFOAM® user is able to significantly reduce the pre-processing efforts: typically, packings of solid fractions up to 0.3 are generated in the range of a few seconds to around a minute. This allows for efficient gathering of data needed for formulation of closure laws or for developing machine learning models, relevant for industrial applications such as pneumatic conveying and fluidized beds.

1. Introduction

In this paper, we present an OpenFOAM® application for generation of tri-periodic assemblies of spheres, cylinders or spherocylinders intended for body-fitted direct numerical simulations (DNS). The assemblies can be mono- or polydisperse, and with fixed or random positions and orientations. The goal of our work is to aid in derivation of closure laws for hydrodynamic forces and moments acting on the particles (i.e., drag, lift, torque, stresslet).

1.1. Motivation.

Fluid-particle flows are common in chemical and process industry, often featuring non-spherical particles and polydispersity [1,2]. An increasingly important example is pneumatic conveying of biomass or plastic waste [3]. In order to facilitate waste recovery, pneumatic handling of these highly non-spherical materials needs to be optimized. This is a challenging task due to the lack of knowledge on the relationships between the operating conditions, particle properties and dimensions of the conveying line. Since experimental testing is costly and difficult, numerical simulation presents a convenient alternative to bridge the gaps in knowledge.

Two types of numerical simulations are commonly used for such problems: two-fluid method (TFM) [4–6] and unresolved coupled computational fluid dynamics and discrete element method (CFD-DEM) [7]. TFM treats both fluid and particle phase as continuous interpenetrating media (Euler-Euler description), while CFD-DEM treats the particle phase as discrete parcels (Euler-Lagrange description). Discretization in both types of simulations leaves the details of the flow field between individual particles unresolved. Fluid-particle interaction forces (e.g., drag) therefore need to be included using closure models.

* Corresponding author

Received: 17 December 2024, Accepted: 12 August 2025, Published: 7 December 2025

Both TFM and CFD-DEM have been successfully used to predict flows in conveying systems [3], albeit almost exclusively for flows with spherical particles. Simulations with non-spherical particles are still a challenge due to the lack of closure models for hydrodynamic quantities such as drag, lift and torque.

Direct numerical simulation (DNS) is a popular way to solve the closure problem in fluid-particle flows [1]. In contrast to TFM and CFD-DEM, discretization in DNS resolves the smallest details of the flow field. This enables explicit calculation of fluid-particles interaction terms, and provides a basis for building of closure models.

So far, the majority of DNS work deals with spherical particles. For example, closures for the average drag force acting on assemblies of monodisperse spheres have been derived by Hill et al. [8] and also by Tenneti et al. [9], while Beetstra et al. [10] additionally considered bidisperse systems. More realistic, polydisperse systems have been studied recently, e.g., by Cheng and Wachs [11]. While the previous studies focused on the average forces, the interest has currently shifted towards the force and torque fluctuations as functions of the adjacent particles. Recent work by Siddani and Balachandar [12] demonstrates feasibility of such sophisticated models.

Despite the large body of work dedicated to spherical particles, there are still unexplored topics that could benefit from further DNS studies. For instance, to the best of our knowledge, no closure model currently exists for stresslet. Stresslet physically signifies resistance to the straining motion [13, 14], usually considered in contexts of microhydrodynamics [15]. However, studies by Wang et al. [16] and Gupta et al. [17] demonstrate that there is a strong contribution of stresslet even in turbulent flows. Since stresslets can be connected to the deformation and breakage of particles, inclusion of such a closure could help build more accurate TFM and CFD-DEM models.

On the other hand, few DNS studies deal with non-spherical particles. As a consequence, closure models are unavailable for a large number of common particle shapes. So far, the most studied non-spherical particles are ellipsoids [18–21]. Recent study by Sanjeevi and Padding [22] stands out as it focuses on spherocylinders. Still, the study by Sanjeevi and Padding [22] is limited to spherocylinders of aspect ratio 4. For instance, the standard shape of a capsule in pharmaceutical industry is spherocylindrical with aspect ratios (length/diameter) ranging from 2.26 to 2.63 [23]. Additionally, many axisymmetric particles found in industry are blunt with sharp edges [24, 25] and therefore more accurately represented by cylinders than by spherocylinders. Some examples include catalysts [26, 27], biofuels such as straw, wood chips and wood pellets [28–30], pharmaceutical tablets [31], and recycled solids [32]. Nonetheless, due to the technical difficulties associated with meshing of sharp edges [33], such particles are rarely represented in literature.

With our current contribution, we hope to inspire more research on the cylindrical and spherocylindrical particles, as well as to aid in derivation of more sophisticated closure laws for the spherical particles. Being motivated by the application of pneumatic conveying, our current study is limited to flows with solid fraction below 0.3. The workflow is tested using creeping flow and steady state inertial flow regimes, with the highest considered Reynolds number $Re = 300$ (see section 3 for the definition of Re). However, we demonstrate that the workflow is also compatible with transient solvers and could be applied to arbitrarily high Re.

1.2. **Goals and limitations.** The goal of our work is to streamline the pre- and post-processing of body-fitted DNS of flows through particle assemblies with OpenFOAM®. Specifically, we address the challenges pertaining to 1) generation of particle assemblies, 2) occurrences of various meshing errors, and 3) calculation of hydrodynamic forces and moments. In what follows, we describe how each of these three challenges is treated in the recent literature and we also highlight the main contributions of our study. Limitations of our study are summarized at the end of this section.

1.2.1. *Generation of particle assemblies.* First, we discuss the state-of-the-art approaches to the generation of geometry. To generate random particle positions and orientations one would typically need to use the Monte Carlo method [22] or the discrete element method (DEM) [34–37]. Within the OpenFOAM® framework, this can be achieved with solvers such as `dsmcFoam` and `rarefiedMultiphaseFoam` [38]. However, the available shapes are limited to spheres and the resulting assemblies will feature overlapping particles which can cause meshing errors (more on this in subsubsection 1.2.2). Alternatively, external software for computer-aided design (CAD) can be used. For example, `Porous microstructure generator` [39] is a software specifically developed for creation of 3D porous structures. The resulting 3D geometry can be exported in form of Stereolithography (STL) files, which can be further handled by `snappyHexMesh`. Similarly, `Blender` [40], a computer graphics software, has been used to create the geometry for simulations of packed bed chemical reactors [41, 42]. Still, the approach based on the STL files has some drawbacks. First, the quality of the STL file becomes a critical parameter that affects the

quality of the mesh and therefore the accuracy of the solution [43]. While this can be circumvented by using a higher resolution file, such as OBJ or STEP, the process of resolving the geometry is still slow. This can be particularly tedious for parametric studies which require a large number of simulations. For example, the number of necessary simulations for derivation of a drag closure is typically around 100: Beetstra et al. [10] studied 150 systems while Sanjeevi and Padding [22] studied 97. To make such pre-processing automatic, one would need to devote additional time into writing scripts that would link different software together [44].

With our present contribution, we simplify the above process by eliminating the need for additional software completely. Instead, we generate the geometry directly using the OpenFOAM® library. Our workflow is based on the feature of `snappyHexMesh` to create geometric objects via `searchableSurfaces`. While spheres and cylinders are already available, we also create spherocylinders as combinations of the other two shapes.

Compared to using external CAD software, our procedure is more efficient. The typical execution time for generating a random tri-periodic assembly of 200 particles and with solid fractions up to 0.3 is in the range of a few seconds to around a minute.

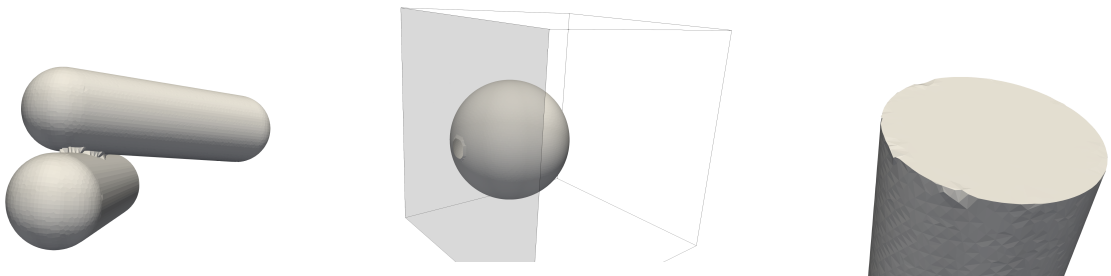
It is important to note here that, being motivated by the application of pneumatic conveying, we did not consider solid fractions higher than ≈ 0.3. Additionally, while there is a possibility to generate polydisperse systems, mixtures of shapes are not supported in our present study. Details on the code implementation are given in section 2, with subsection 2.1 dealing specifically with the generation of random particle assemblies.

1.2.2. *Meshing errors.* The second challenge we tackle is related to avoiding meshing errors in body-fitted unstructured grids created with `snappyHexMesh`. In general, there are two types of grids used for particle assemblies: 1) body-fitted unstructured grid, where only the fluid region is meshed, and 2) body–non-conformal structured grid, where both the fluid and the particle regions are meshed.

The body-fitted approach avoids meshing the particle interior and therefore requires less cells. Additionally, it allows for a smooth representation of the particle surface. The non-conformal approach, on the other hand, introduces artificial roughness due to the stair-stepped representation of the particle surface [45]. This is especially problematic for blunt particles (such as cylinders), since inaccurately described edges lead to inaccurate force calculation [33]. Nonetheless, the non-conformal approach is so far more popular [8, 9, 22], mostly due to the difficulties related to unstructured grids.

Particle packings represent a complex geometry to mesh, so much so that creating an unstructured grid can in some cases take longer than running the simulation [45]. Furthermore, a direct comparison between a body-fitted and a non-conformal simulation of a packed bed of spheres [46] revealed the body-fitted approach to be labour-intensive, slower performing and—due to the presence of skewed cells—slightly less accurate.

However, the time and effort needed to create an unstructured grid are greatly reduced when using currently available automatic meshers such as `snappyHexMesh`. This leaves us to address the accuracy of the body-fitted approach. Specifically, `snappyHexMesh` is known to produce meshing errors in certain cases [43]. Some of the errors we observed are shown in Fig. 1 and they include bridges between nearby particles, distortions of the shape in the vicinity of the boundaries, and rough edges resulting from improper edge capturing.



(a) Bridges between nearby particles

(b) Shape distortion near boundary

(c) Rough edge

Figure 1. Examples of meshing errors

To eliminate the unwanted bridges (Fig. 1a)—without increasing the mesh resolution—we control the minimum allowed distance between the particles (subsubsection 2.1.5). Similarly, the allowed distance/overlap between a particle and a boundary are controlled to avoid errors due to the tangentiality to

the boundary (Fig. 1b). This tangentiality check is described in subsubsection 2.1.4. Particularly prone to meshing errors are the sharp edges, occurring in cylindrical particles (Fig. 1c) or at the intersections between the particles and the boundaries. To avoid jagged appearances of the edges, we use the explicit edge capturing feature available in `snappyHexMesh`. In brief, we analytically calculate the points comprising an edge and feed them to `snappyHexMesh` in form of `eMesh` files. The procedure is described in subsection 2.2.

1.2.3. *Calculation of hydrodynamic forces and moments.* Lastly, we look into calculation of relevant forces acting on the particles. With OpenFOAM®, it is already possible to calculate the hydrodynamic force and torque using the `forces` function object. We extend this function object to also include stresslets [13, 14]. This lesser-known quantity is connected to straining motion and affects the stress within suspensions [15, 47]. Additionally, we implement a pressure drop correction for forces acting on the periodic neighbours, i.e., particles split by periodic boundaries. This correction is explicitly calculated by a novel iterative pressure boundary condition. The boundary condition and the force calculation are described in subsection 2.3 and subsection 2.4, respectively.

In section 3, we demonstrate that the body-fitted DNS is as accurate as the other available methods, including the lattice-Boltzmann method (LBM) [8,22], the immersed boundary method (IBM) [9], and the fictitious domain method (FDM) [48, 49]. Specifically, we achieve excellent agreement for ordered arrays of spheres compared to the LBM and IBM simulations [8,9,50] in subsection 3.1. Next, in subsection 3.2, we compare a simulation of cylindrical particles using OpenFOAM® and a well-validated FDM-based solver PeliGRIFF [48, 49]. The two solvers yield nearly identical forces and torques acting on individual particles. In subsection 3.3, we reproduce the results for the drag force acting on unidirectional random assemblies of spherocylinders as reported in [22]. Here, we also simulate similar systems of cylinders, showing that the blunt shape of the cylinder leads to around 13% higher drag. Lastly, in subsection 3.4, we verify the stresslet calculation against an analytical solution.

1.2.4. *Limitations.* The main limitations of our present study are as follows:

- This study is limited to solid fractions below 0.3. The present algorithm for assembly generation does not allow for relaxation of the system and therefore limits the possible packing densities.
- Treatment of meshing errors for particles in contact is not addressed (instead, we recommend assigning a minimum distance between particles, as described in Appendix B).
- While it is possible to create polydisperse assemblies, assemblies featuring a mixture of particle shapes (e.g., containing spheres and cylinders) are not supported.
- The algorithm for geometry generation is not parallelized.
- In our approach the particles are frozen, meaning that they do not change their position nor orientation during the flow. This allows us to compute closures for the fluid-averaged equations. To study how the evolution of particle cloud affects the flow properties, methods such as resolved CFD-DEM need to be used [51]. With resolved CFD-DEM, further information can be gained: how fluid flow and particle collisions affect the particle dispersion, as well as how particles affect the flow field.
- The pressure boundary condition needs an iterative solver for proper functioning.

2. Computational methodology

This section describes the computational model for pre- and post-processing for body-fitted particle-resolved simulations. The model is implemented in OpenFOAM®-plus (2306). Before describing the workflow, it is important to note that the particles are immobile, i.e., they have fixed positions. Additionally, we will assume a steady state flow. Application of the workflow to transient flows is discussed in Appendix E.

Following Fig. 2, the workflow starts with creating a domain with a uniform background grid using the `blockMesh` application. The cell size of the background grid will define the coarsest cell size in the final unstructured grid.

A particle assembly is generated next with the `initialiseSearchableSurfaces` pre-processing utility. Specifically, this step produces files needed by the other OpenFOAM® applications. These files include `searchableSurface` dictionaries, `eMesh` files and `forcesPressureJump` dictionaries.

Next, using the `snappyHexMesh` application, an unstructured body-fitted grid is created around the particles. The particle surfaces are represented through the `searchableSurface` objects defined in the previous step. If appropriate, sharp edges are explicitly refined using the `eMesh` files.

The pre-processing finishes by imposing a periodic constraint on the domain boundaries with the `createPatch` application.

Finally, the simulation is run with `simpleFoam`. Pressure drop which maintains the target mean velocity is enforced with the new `uniformVelocityPressureJumpAMI` boundary condition. Forces (and moments) acting on the particles are calculated using the new `forcesPressureJump` function object. The input dictionaries for the function object are automatically generated by the `initialiseSearchableSurfaces` application. Pressure correction for the particles at the outlet boundary is computed by the pressure boundary condition.

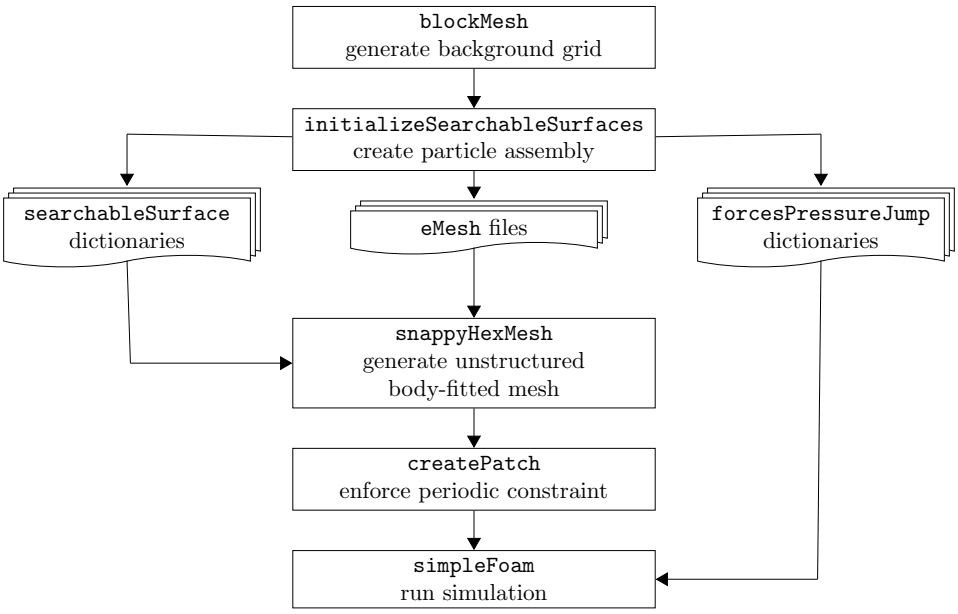
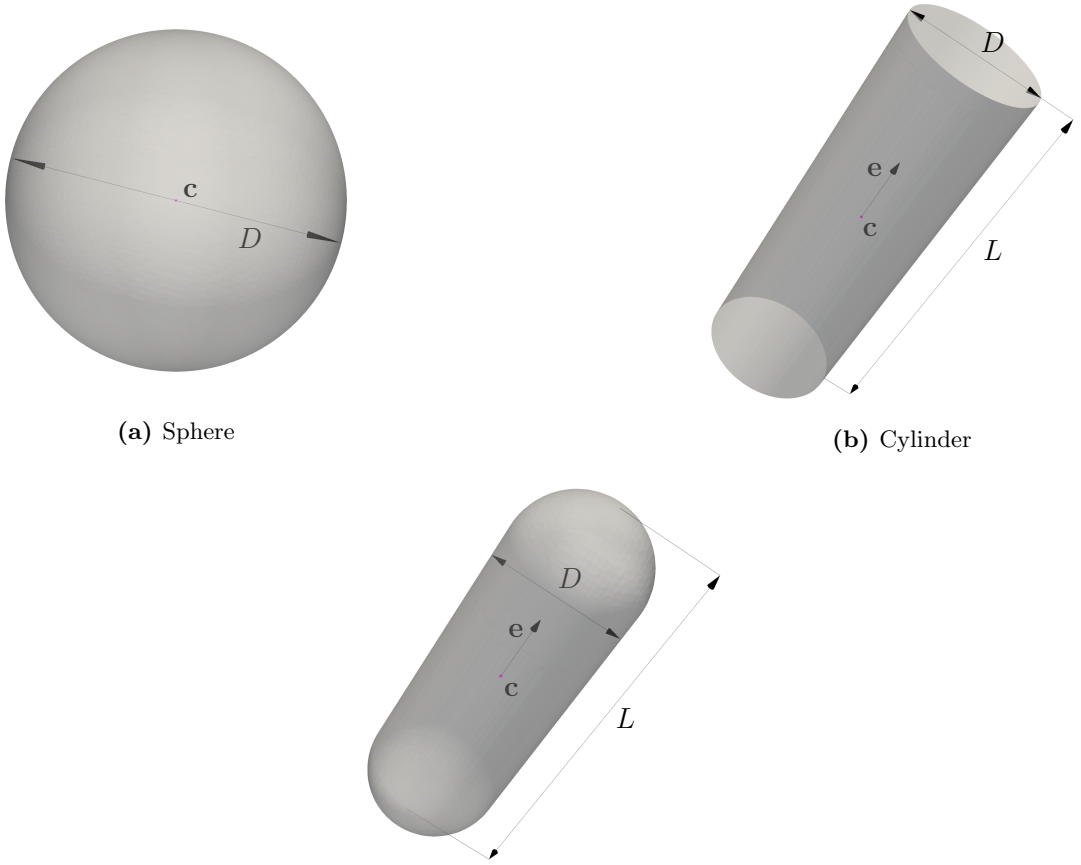


Figure 2. Workflow for DNS of steady state flow through tri-periodic assemblies of particles

Summary of the new developments within the context of OpenFOAM® directory structure is shown below.

```
applications
└── utilities
    └── preProcessing
        └── initialiseSearchableSurfaces
            └── searchableSurfacesFunctions
                ├── surface
                │   ├── cylinder
                │   ├── sphere
                │   ├── spherocylinder
                │   └── surface
                └── surfaceAssembly
                    ├── cylinderAssembly
                    ├── sphereAssembly
                    ├── spherocylinderAssembly
                    └── surfaceAssembly
src
├── finiteVolume
│   └── fields
│       └── fvPatchFields
│           └── derived
│               └── uniformVelocityPressureJumpAMI
└── functionObjects
    └── forcesPressureJump
```

As mentioned, the `initialiseSearchableSurfaces` application creates all files needed for the meshing and post-processing of a particle assembly. Two abstract classes are written to support the application: class `surface` and class `surfaceAssembly`. The subclasses inheriting the `surface` class are the classes `sphere`, `cylinder` and `spherocylinder`. Illustrated in Fig. 3, each class contains data members referring to the geometric properties of the particles such as the position of the centre of the mass $\mathbf{c}$, particle diameter D, orientation vector $\mathbf{e}$, particle length L, aspect ratio $AR = \frac{L}{D}$, etc. The generation of assemblies is handled by the classes inheriting the `surfaceAssembly` class, i.e., `sphereAssembly`, `cylinderAssembly` and `spherocylinderAssembly` class, all containing lists of objects of the titular `surface` subclass.



(a) Sphere

(b) Cylinder

(c) Spherocylinder

Figure 3. Illustration of the main geometric properties of the particles

In order to run the `initialiseSearchableSurfaces` application, the user needs to specify the input parameters in the `searchableSurfacesDict` placed within the `system` directory. The parameters include particle type, number of particles, type of assembly (fixed or random), size distribution, etc. To run the entire case, files created by the `initialiseSearchableSurfaces` application need to be linked in the `snappyHexMeshDict` and `controlDict`, as shown in the supplemental tutorial examples.

Specificities of the generation algorithms for each particle type are given in subsection 2.1. Creation of the `eMesh` files, needed for the explicit edge refinement, is described in subsection 2.2. Details on the steady state incompressible flow solver `simpleFoam` can be found in many references, e.g. [52–54]. The new boundary condition `uniformVelocityPressureJumpAMI` and the new post-processing function object `forcesPressureJump` are described in subsection 2.3 and subsection 2.4, respectively.

We note here that, since our study is limited to solid fractions up to ≈ 0.3, particles do not come into contact. However, for higher solid fractions, contact is unavoidable. Various techniques for handling the contact points can be found in the literature [45, 55, 56]. Potentially, our code could be extended in the future to also enable dense packings and contact point handling.

2.1. **Generation of particle assembly.** This section deals with the algorithm for generation of a particle assembly (Fig. 4). Each step of the algorithm is detailed in the corresponding subsection (subsubsection 2.1.1–subsubsection 2.1.5). Lastly, performance is tested in subsubsection 2.1.6.

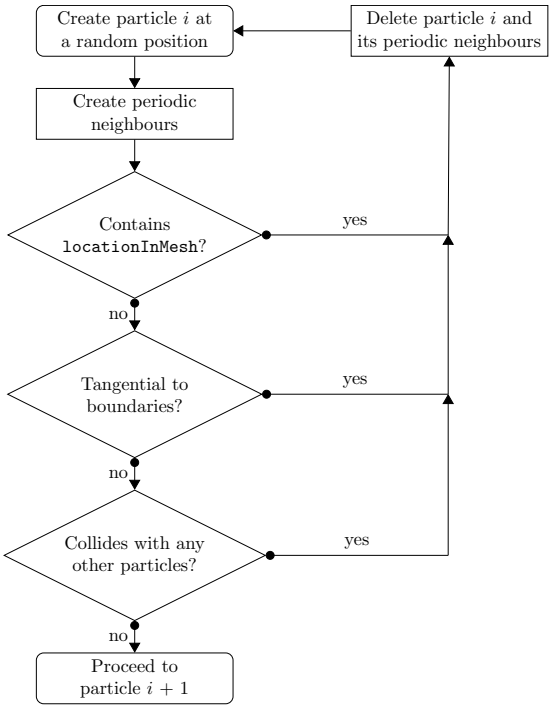


Figure 4. Algorithm for generation of a particle assembly

Being a part of an assembly, the individual particles should be mathematically denoted with a subscript i. However, to simplify the notation, we keep the subscript only where necessary, i.e.: to distinguish particles i and j in a collision check.

2.1.1. *Creating a particle.* To create a particle, its geometric properties need to be defined. As seen in Fig. 3, depending on the particle type, these properties include position (centre of the mass $\mathbf{c}$), orientation ($\mathbf{e}$), diameter (D) and aspect ratio (AR).

As said before, the assemblies can be fixed or random. In the case of a fixed assembly, positions and orientations are simply read from a user-supplied list. To create a particle in a random assembly, a random position $\mathbf{c}$ is generated in the space within the domain boundaries. To generate random orientations, according to [57], we need two random angles ϕ and θ:

$$\phi = \arccos(1 - 2X), \tag{1}$$

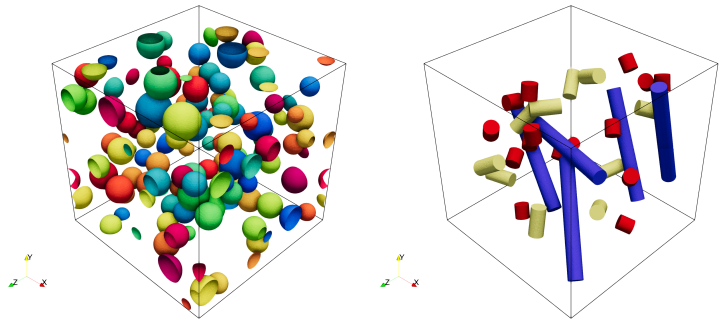
$$\theta = Y\pi, \tag{2}$$

where X and Y are random numbers between 0 and 1. The orientation vector then follows from:

$$\mathbf{e} = \left[\sin\phi\,\cos\theta\,e_{g,x},\ \sin\phi\,\sin\theta\,e_{g,y},\ \cos\phi\,e_{g,z}\right], \tag{3}$$

where $\mathbf{e}_g$ is a user-defined global orientation vector. The global orientation vector prevents rotation in certain directions, e.g.: $\mathbf{e}_g = [1, 0, 0]$ produces a unidirectional assembly, parallel to the x-axis, while $\mathbf{e}_g = [1, 1, 0]$ produces a planar random assembly with rotations contained to the xy-plane. By default, the global orientation vector is set to fully random: $\mathbf{e}_g = [1, 1, 1]$.

Properties defining the particle size (such as diameter or aspect ratio) can be set according to uniform (monodisperse), Gaussian or number-based distribution. Examples of polydisperse assemblies are shown in Fig. 5. Specifically, Fig. 5a shows a polydisperse system of 100 spheres with radius set according to the Gauss normal distribution. Such distribution is achieved using Listing 1 in `searchableSurfacesDict`. Fig. 5b shows a number-based size distribution. Here, 5 cylinders are assigned an aspect ratio of 8, 10 cylinders an aspect ratio of 2 and 15 cylinders an aspect ratio of 1 (Listing 2).



(a) Gaussian distribution **(b)** Number distribution

Figure 5. Examples of polydisperse assemblies

```
radius
{
    sizeDistribution          GaussNormal;
    meanValue                 0.06;
    standardDeviation         0.01;
}
```

Listing 1. Dictionary input for Gaussian normal distribution

```
aspectRatio
{
    sizeDistribution      numberDistribution;
    numberDistribution    (5 10 15);
    value                 (8  2  1);
}
```

Listing 2. Dictionary input for number distribution

2.1.2. *Creating periodic neighbours.* To keep the periodicity of the system, particles which intersect the domain boundaries need periodic neighbours.

Intersection between a spherical particle and a planar boundary is tested using:

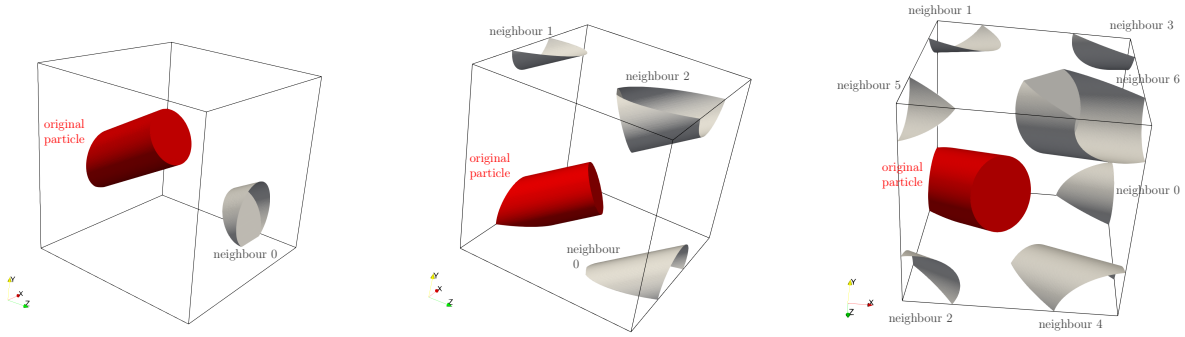
$$|\mathbf{n}_p \cdot (\mathbf{c} - \mathbf{p}_p)| \leq \frac{D}{2}, \tag{4}$$

while for a cylinder, this test is [58]:

$$|\mathbf{n}_p \times (\mathbf{c} - \mathbf{p}_p)| \leq \frac{D}{2}\sqrt{1 - (\mathbf{n}_p \times \mathbf{e})^2} + \frac{L}{2}\,|\mathbf{n}_p \times \mathbf{e}|\,. \tag{5}$$

Intersection of a spherocylinder and a boundary is evaluated using a combination of the above expressions. In the expressions above, $\mathbf{n}_p$ is the unit-normal vector of the boundary and $\mathbf{p}_p$ is an arbitrary point on the boundary (implemented as the centre-point of the boundary patch).

In the case of an intersection of the original particle and the boundary, a neighbouring particle is created at the opposite side of the domain. This new particle has all properties identical to those of the original one, except for the position. The total number of the periodic neighbours is determined by the number of intersecting boundaries: one neighbour is created for an intersection with one boundary, three neighbours for intersections with two boundaries, and seven neighbours for intersections with three boundaries (Fig. 6).



(a) 1 periodic neighbour **(b)** 3 periodic neighbours **(c)** 7 periodic neighbours

Figure 6. Particle with periodic neighbours

It is also possible to create a non-periodic assembly. In this case, all particles that intersect the boundaries are removed. Such an assembly is created by setting `findPeriodicNeighbours` in `searchableSurfacesDict` to `false`. As seen in Fig. 7, the resulting orientation distribution is non-uniform, with particles near boundaries being more likely parallel to the boundaries. This happens even if isotropically random distribution is enforced with Eqn. 3 and allows for a representation of wall-induced perturbations [59].

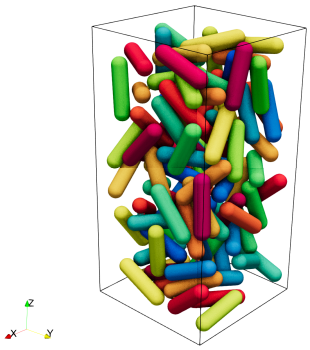


Figure 7. Non-periodic assembly

2.1.3. *Reference point test.* After the periodic neighbours are created, a test is performed to determine if the reference point is contained within the new particles. The reference point is specified under `locationInMesh` setting in `snappyHexMeshDict`, with the purpose of designating the region occupied with the fluid. Particles (and their neighbours) overlapping with this location are therefore removed.

It needs to be addressed here that it would be natural to determine this reference point only after the assembly is created as it would reduce the number of necessary computations. However, compared to the collision detection algorithm, the time needed for this test is negligible. Also, determining the reference point after creating the assembly would slightly complicate the setup of the `snappyHexMeshDict`. Therefore, we specify the reference point prior to generating the assembly.

2.1.4. *Tangentiality check.* To prevent meshing errors for particles near or intersecting the boundary (as seen in Fig. 1b), we perform a tangentiality check. The tangentiality is controlled through two parameters: minimum allowed distance from the boundary (`minBoundaryDistance`) and minimum allowed overlap with the boundary (`minBoundaryOverlap`), both values relative to the particle radius r. As before, if evaluated as tangential, the particle (and its periodic neighbours) are removed from the assembly.

A sphere is considered tangential to a boundary (containing a point $\mathbf{p}_p$ and a normal $\mathbf{n}_p$) if the below expression is true:

$$(1 - \Delta_{mo})\, r \leq |(\mathbf{p}_p - \mathbf{c}) \cdot \mathbf{n}_p| \leq (1 + \Delta_{md})\, r, \tag{6}$$

where Δ_{mo} and Δ_{md} are respectively the aforementioned minimum allowed overlap and minimum allowed distance.

For a spherocylinder, Eqn. 6 is applied on its top and bottom hemisphere.

For a cylinder, the tangentiality check involves several steps. First, it is determined whether the cylinder is intersecting the boundary using Eqn. 5. For a non-intersecting cylinder (see Fig. 8a), the minimum distance from the boundary is evaluated using:

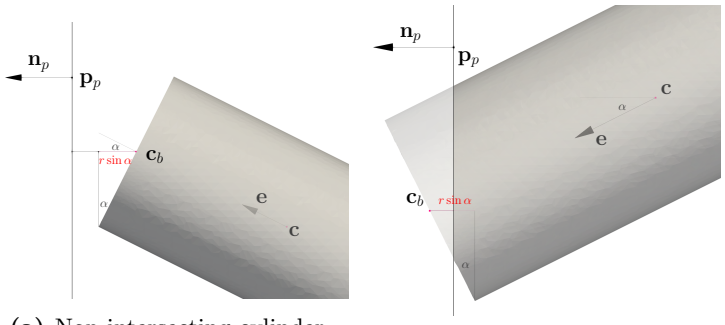
$$|(\mathbf{p}_p - \mathbf{c}_b) \bullet \mathbf{n}_p| - r \sin\alpha \leq \Delta_{md} r, \tag{7}$$

where $\mathbf{c}_b$ is the centre of the top or bottom cylinder base and α is the angle between the boundary and the cylinder:

$$\alpha = \arccos |\mathbf{n}_p \bullet \mathbf{e}| \,. \tag{8}$$

If the cylinder intersects the boundary (see Fig. 8b), the test for the minimum allowed overlap will depend on the angle α:

$$\begin{aligned} \text{if } \ 0 < \alpha < \frac{\pi}{2} &\implies (1 - \Delta_{mo})\, r \sin\alpha \leq |(\mathbf{p}_p - \mathbf{c}_b) \bullet \mathbf{n}_p| \leq r \sin\alpha, \\ \text{if } \ \alpha = 0 &\implies |(\mathbf{p}_p - \mathbf{c}_b) \bullet \mathbf{n}_p| \geq (1 - \Delta_{mo})\, r, \\ \text{if } \ \alpha = \frac{\pi}{2} &\implies |(\mathbf{p}_p - \mathbf{c}_b) \bullet \mathbf{n}_p| \leq \Delta_{mo} r. \end{aligned} \tag{9}$$



(a) Non-intersecting cylinder **(b)** Intersecting cylinder

Figure 8. Tangentiality check for a cylindrical particle

Selection of the optimal values for Δ_{mo} and Δ_{md} will depend on the considered system and the grid size. The test case described in subsection 3.3 can be used as a rough guideline: for a system with solid fraction 0.2 and a cell length of $\approx D/40$, we used $\Delta_{md} = 0.1$ and $\Delta_{mo} = 0.25$. For a similar system, but with a finer mesh, these values could be reduced. More information on selection of optimal parameters can be found in Appendix B.

2.1.5. *Collision check.* The last step of the assembly generation algorithm is the collision check. This removes particles that overlapping with the already existing particles in the domain.

For two spherical particles, i and j, the collision check is based on the distance between their centres:

$$|\mathbf{c}_i - \mathbf{c}_j| \leq r_i + r_j + \epsilon. \tag{10}$$

To prevent meshing errors, such as formation of bridges (Fig. 1a), the minimum allowed distance between the particles is controlled with the ϵ parameter. For spheres, this value is:

$$\epsilon = \frac{r_i + r_j}{2} \Delta_{p-p}, \tag{11}$$

where Δ_{p-p} is the user-defined minimum relative gap between the particles. A procedure for selection of optimal Δ_{p-p} for the considered case is discussed in Appendix B.

The collision check for two cylinders, i and j, depends on the angle between their orientation vectors. If the two cylinders are parallel ($\mathbf{e}_i \times \mathbf{e}_j \equiv 0$), it is sufficient to test [60]:

$$\begin{aligned} \frac{L_i + L_j + 2\epsilon}{2} &< |\mathbf{\Delta}_{ij} \bullet \mathbf{e}_i| \,, \\ r_i + r_j + \epsilon &< |\mathbf{\Delta}_{ij} \times \mathbf{e}_i| \,, \end{aligned} \tag{12}$$

where $\mathbf{\Delta}_{ij} = \mathbf{c}_j - \mathbf{c}_i$ is the distance between their centres. The minimum allowed distance ϵ is defined as:

$$\epsilon = \min\left(\frac{r_i + r_j}{2}, \frac{L_i + L_j}{4}\right) \Delta_{p-p}, \tag{13}$$

with Δ_{p-p} being the relative gap as described before.

Otherwise, for two non-parallel cylinders, the collision check is a prohibitive task [60]. In the interest of efficiency, we first perform a quick check using the method of separation of axes [60], and then—if needed—we employ the algorithm for spherocylinders [61, 62] (see Appendix A).

The method of separation of axes states that two cylinders are separated (not colliding) if there exists any axis $\mathbf{d}$ such that:

$$f(\mathbf{d}) = (r_i + \epsilon)\,|\mathbf{d} \times \mathbf{e}_i| + (r_j + \epsilon)\,|\mathbf{d} \times \mathbf{e}_j| + \frac{L_i + 2\epsilon}{2}\,|\mathbf{d} \bullet \mathbf{e}_i| + \frac{L_j + 2\epsilon}{2}\,|\mathbf{d} \bullet \mathbf{e}_j| - |\mathbf{d} \bullet \boldsymbol{\Delta}_{ij}| < 0. \tag{14}$$

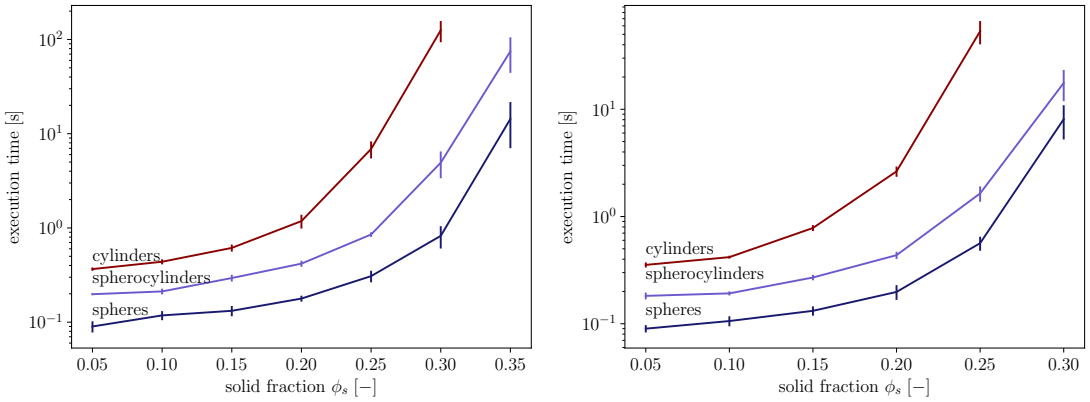
As recommended in [60], we test the above expression for $\mathbf{d} \equiv \boldsymbol{\Delta}_{ij}, \mathbf{e}_i \times \mathbf{e}_j, \mathbf{e}_i$ and $\mathbf{e}_j$. The testing stops as soon as Eqn. 14 is evaluated as true and the new cylinder is kept.

While finding a separation axis proves that the cylinders are not colliding, not finding one does not imply a collision. There is potentially an infinite number of separation axes $\mathbf{d}$ that could be tested [60]. Therefore, if no separation axis is found after four tests, we employ the algorithm for spherocylinders [61, 62] (see Appendix A). Here we approximate the cylinder with a spherocylinder of an equal shaft-length, i.e., $AR_{sc} = AR_c + 1$, where sc denotes the spherocylinder and c cylinder. However, it needs to be noted that this approximation could occasionally remove non-colliding cylinders.

2.1.6. *Performance test.* To illustrate the behaviour of our packing algorithm, we use an example of a random tri-periodic assembly with 200 particles (Fig. 9). For non-spherical particles, we use aspect ratio of 2. Generally, the aspect ratio will limit the possible density of a random packing, since denser packings tend to exhibit a directional preference [22, 59].

Since we use a rather simplistic method for generating particle assemblies, we are only able to produce moderately dense systems. Imposing no minimum interparticle distance ($\Delta_{p-p} = 0$, subsubsection 2.1.5) and no tangentiality check (subsubsection 2.1.4), we are able to achieve solid fraction of 0.35 for spheres and spherocylinders, and of 0.3 for cylinders (Fig. 9a). However, if meshed, these assemblies would likely feature meshing errors. Adapting the assembly for coarse meshes, i.e.: minimum interparticle distance set to $\Delta_{p-p} = 0.1$ and tangentiality-related quantities set to $\Delta_{md} = 0.1$ and $\Delta_{mo} = 0.25$, causes the maximum solid fractions to drop to 0.3 for spheres and spherocylinders, and to 0.25 for cylinders (Fig. 9b). Selection of optimal parameters for meshing is discussed in Appendix B.

More advanced algorithms that allow for higher solid fractions are available in the literature [63, 64] and could be implemented in the future. For now, the code works for solid fractions $\phi_s \leq 0.3$. As seen in Fig. 9 (tested on Intel®Core™i7-9850H CPU @ 2.60GHz using one core), it typically takes less than a minute to generate a packing with $\phi_s \leq 0.25$ and less than 10 seconds for $\phi_s \leq 0.2$.



(a) $\Delta_{p-p} = 0$, no tangentiality testing

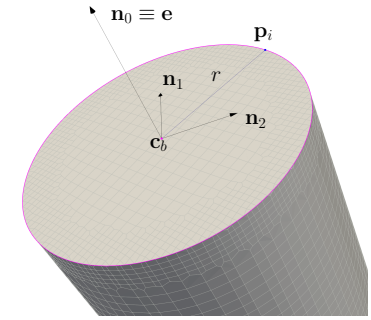
(b) $\Delta_{p-p} = 0.1$, $\Delta_{md} = 0.1$, $\Delta_{mo} = 0.25$

Figure 9. Average execution time for creating a tri-periodic assembly with 200 particles. The averages are calculated using 5 independent runs. Error bars represent the standard deviation. Tested on Intel®Core™i7-9850H CPU @ 2.60GHz.

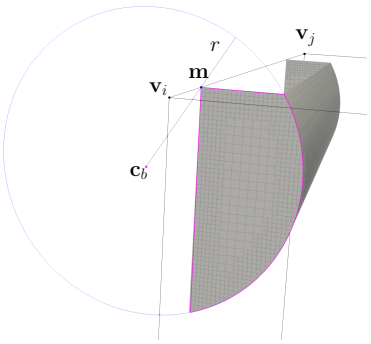
2.2. **Treatment of sharp edges.** The automatic mesh generation is highly sensitive to the occurrence of sharp edges in geometry (Fig. 1c). In order to avoid meshing errors, we explicitly define the edge points via `eMesh` files. The `eMesh` files inform `snappyHexMesh` of the edge location, leading to a precise discretization. This also allows for additional mesh refinement near and at the edge. Such refinement is strategically important, as it can reveal the details of the flow field near abrupt changes of the interstitial space.

We deal with two types of edges. The first type is inherent to the geometric shape of the particles, here appearing only in cylinders, at the joinings of the shaft and the bases. Details on the edge meshes for the cylinder bases are given in subsubsection 2.2.1. The second type of edges occurs at the intersections of particles and periodic boundaries. These intersection meshes are particularly important for the stability of the simulation, as they ensure conformity between the neighbouring periodic boundaries. Mathematically, the boundaries are treated as planes characterized by a point $\mathbf{p}_p$ (centre of the patch) and an outward-pointing unit-normal vector $\mathbf{n}_p$. The procedures for finding the intersection edges of spherical and cylindrical particles are described in subsubsection 2.2.2 and subsubsection 2.2.3, respectively. Spherocylinders simply inherit the procedures from the other two shapes.

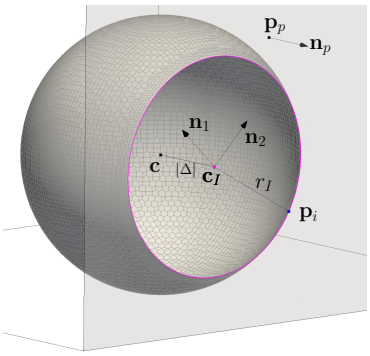
Examples of various edge meshes are shown in Fig. 10.



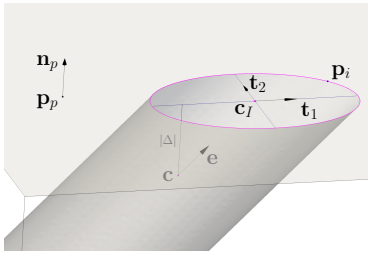
(a) Cylinder base with an orthonormal basis



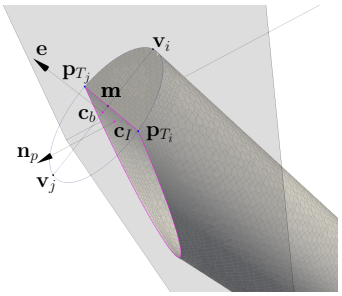
(b) Cylinder base intersected by a domain edge



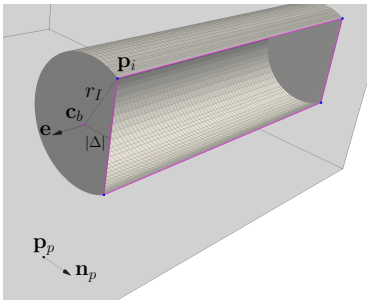
(c) Circular intersection of a spherical particle and a planar boundary



(d) Elliptical intersection of a cylindrical particle and a planar boundary



(e) Elliptical intersection trimmed by the cylinder base



(f) Rectangular intersection of a cylindrical particle and a planar boundary

Figure 10. Types of edge meshes

2.2.1. *Cylinder edges.* Cylindrical particles need explicitly defined edges at the circular bases. Following the notation from Fig. 10a, a point $\mathbf{p}_i$ on the cylinder edge is calculated using:

$$\mathbf{p}_i = \mathbf{c}_b + \cos\theta_i r\mathbf{n}_1 + \sin\theta_i r\mathbf{n}_2, \tag{15}$$

where $\mathbf{c}_b$ is the centre of the circular base, r is the cylinder radius, and θ_i is an angle between 0 and 2π. Number of points describing the edge is user-defined, with the default value set to 360 (in the future, the code could be improved to automatically determine the resolution of the edge mesh based on the local grid size). Vectors $\mathbf{n}_1$ and $\mathbf{n}_2$ create an orthonormal basis with the cylinder orientation vector $\mathbf{e}$.

To find the vector $\mathbf{n}_1$, we use the Gram-Schmidt process [65]:

$$\mathbf{n}_1 = \frac{\mathbf{n}_1^1 - (\mathbf{n}_1^1 \cdot \mathbf{n}_0)\,\mathbf{n}_0}{|\mathbf{n}_1^1 - (\mathbf{n}_1^1 \cdot \mathbf{n}_0)\,\mathbf{n}_0|}. \tag{16}$$

Here, $\mathbf{n}_0$ is equal to the orientation vector $\mathbf{e}$ and $\mathbf{n}_1^1$ is set to $[1,1,1]$ in general case or to $[0,1,1]$ if $\mathbf{n}_0$ is parallel to $[1,1,1]$.

Vector $\mathbf{n}_2$ then simply follows from:

$$\mathbf{n}_2 = \mathbf{n}_1 \times \mathbf{n}_0. \tag{17}$$

The situation where the circular base is intersected by one of the domain edges requires a special attention. As seen in Fig. 10b, there will be a point $\mathbf{m}$ on the base interior which needs to be added to the list of the edge points. This point is found using the Plücker formula for a meet of a line and a plane [66, 67]:

$$\mathbf{m} = \frac{(\mathbf{n}_P \times \mathbf{C}_m) - (-(\mathbf{p}_P \cdot \mathbf{n}_P)\,\mathbf{C}_d)}{\mathbf{n}_P \cdot \mathbf{C}_d}. \tag{18}$$

In the equation above, the plane is defined with a point $\mathbf{p}_P$ and a normal $\mathbf{n}_P$. Since we are interested in the plane containing the cylinder base, we use $\mathbf{p}_P \equiv \mathbf{c}_b$ and $\mathbf{n}_P \equiv \mathbf{e}$. The line intersecting the base is the domain edge with vertices $\mathbf{v}_i$ and $\mathbf{v}_j$, which is in Plücker coordinates defined as:

$$\begin{aligned} \mathbf{C}_d &= \mathbf{v}_j - \mathbf{v}_i, \\ \mathbf{C}_m &= \mathbf{v}_i \times \mathbf{v}_j. \end{aligned} \tag{19}$$

2.2.2. *Intersection edges for spherical particles.* Intersection of a sphere and a plane is circular in shape (Fig. 10c). The points of the intersection edge are therefore found using an equation similar to Eqn. 15:

$$\mathbf{p}_i = \mathbf{c}_I + \cos\theta_i r_I \mathbf{n}_1 + \sin\theta_i r_I \mathbf{n}_2 + \boldsymbol{\epsilon}. \tag{20}$$

Here, the capital subscript I denotes the intersection values, i.e., centre $\mathbf{c}_I$ and radius r_I of the intersection curve, while the lower subscript i denotes points iterated between angles 0 and 2π. Vectors $\mathbf{n}_1$ and $\mathbf{n}_2$ form an orthonormal basis with the normal to the boundary patch, i.e., $\mathbf{n}_0 \equiv \mathbf{n}_p$ in Eqns. 16 and 17. The optional quantity $\boldsymbol{\epsilon} = -\Lambda r \mathbf{n}_p$ pushes the edge mesh inside the domain for a small fraction of the particle radius. This prevents the edge from being inadvertently omitted during the meshing. The default value for Λ is 0. We used $\Lambda = 10^{-4}$ in the test cases in section 3.

The centre of the intersection circle $\mathbf{c}_I$ is calculated from:

$$\mathbf{c}_I = \mathbf{c} + \Delta \mathbf{n}_p, \tag{21}$$

where $\mathbf{c}$ is the centre of the mass of the sphere, and Δ is the signed distance between the sphere and the boundary:

$$\Delta = (\mathbf{p}_p - \mathbf{c}) \cdot \mathbf{n}_p. \tag{22}$$

Last, the radius of the intersection follows from:

$$r_I = \sqrt{r^2 - \Delta^2}. \tag{23}$$

2.2.3. *Intersection edges for cylindrical particles.* The shape of an intersection of a cylinder and a plane depends on the angle between them: if the cylinder axis is parallel to the plane, the intersection is rectangular; if the cylinder axis is normal to the plane, the intersection is circular; otherwise, the intersection is elliptical. This elliptical intersection can be full or trimmed by either one or both cylinder bases [58].

To find the points of the elliptical and circular intersections (Fig. 10d), we use the following equation:

$$\mathbf{p}_i = \mathbf{c}_I + \cos\theta_i r_{I,1} \mathbf{t}_1 + \sin\theta_i r_{I,2} \mathbf{t}_2 + \boldsymbol{\epsilon}, \tag{24}$$

where $\mathbf{c}_I$ is the intersection centre, $r_{I,1}$ and $r_{I,2}$ are respectively the major and minor radius of the ellipse, and $\mathbf{t}_1$ and $\mathbf{t}_2$ are respectively the major and minor axis direction. The optional parameter $\boldsymbol{\epsilon} = -\Lambda \max(D, L)\mathbf{n}_p$ pushes the edge mesh inside the domain for a small fraction of the cylinder maximal dimension, either the diameter or the length. As described previously, this is to aid `snappyHexMesh`, since edges that are directly at the boundaries are considered to be outside of the domain and thus get omitted during the meshing.

The major radius of the ellipse is found from:

$$r_{I,1} = \frac{r}{|\mathbf{n}_p \cdot \mathbf{e}|}, \tag{25}$$

while the minor radius is simply that of the cylinder:

$$r_{I,2} = r. \tag{26}$$

Generally, the major axis direction can be calculated using:

$$\mathbf{t}_1 = \frac{\mathbf{e} - (\mathbf{n}_p \cdot \mathbf{e})\,\mathbf{n}_p}{|\mathbf{e} - (\mathbf{n}_p \cdot \mathbf{e})\,\mathbf{n}_p|}. \tag{27}$$

However, if the cylinder axis is normal to the plane ($|\mathbf{n}_p \cdot \mathbf{e}| = 1 \implies |\mathbf{e} - (\mathbf{n}_p \cdot \mathbf{e})\,\mathbf{n}_p| = 0$), the intersection is circular and $\mathbf{t}_1$ is found from Eqn. 16 with $\mathbf{n}_0 \equiv \mathbf{e}$.

The minor axis direction follows from the cross product between the plane normal and the major axis direction:

$$\mathbf{t}_2 = \mathbf{n}_p \times \mathbf{t}_1. \tag{28}$$

Finally, the ellipse centre is:

$$\mathbf{c}_I = \mathbf{c} + \frac{\mathbf{e}\Delta}{\mathbf{n}_p \cdot \mathbf{e}}, \tag{29}$$

where Δ is the signed distance between the cylinder centre $\mathbf{c}$ and the point on the boundary $\mathbf{p}_p$ as in Eqn. 22.

Before iterating through the edge points with Eqn. 24, we test whether the ellipse is trimmed by any of the cylinder bases (Fig. 10b). For this we use the Plücker equation for a meet between a line and a plane. In Eqn. 18, the plane is now defined with the patch normal $\mathbf{n}_P \equiv \mathbf{n}_p$ and with the ellipse centre $\mathbf{p}_P \equiv \mathbf{c}_I$. The line spans the diameter of the cylinder base, with the following vertices:

$$\mathbf{v}_{i,j} = \mathbf{c}_b \pm (\mathbf{t}_2 \times \mathbf{e})\, r. \tag{30}$$

After transforming the line to Plücker coordinates with Eqn. 19, the meet $\mathbf{m}$ follows from Eqn. 18.

If the distance between the meet $\mathbf{m}$ and the cylinder base centre $\mathbf{c}_b$ is shorter than the cylinder radius, the ellipse is trimmed. The trim points are then:

$$\mathbf{p}_{T_{i,j}} = \mathbf{m} \pm \sqrt{r^2 - |\mathbf{c}_b - \mathbf{m}|^2}\mathbf{t}_2. \tag{31}$$

We convert the trim points into angles using [68]:

$$\theta_{T_{i,j}} = \arctan\left[\frac{r_{I,1}\left(\left(\mathbf{p}_{T_{i,j}} - \mathbf{c}_I\right) \cdot \mathbf{t}_2\right)}{r_{I,2}\left(\left(\mathbf{p}_{T_{i,j}} - \mathbf{c}_I\right) \cdot \mathbf{t}_1\right)}\right]. \tag{32}$$

The intersection points can now be calculated from Eqn. 24 for angles θ_i in the interval demarcated by the equation above.

Lastly, the intersection of a cylinder and a plane is rectangular if the cylinder axis is parallel to the plane (Fig. 10f). In this case, the vertices of the rectangular intersection are:

$$\mathbf{p}_i = \mathbf{c}_b - \Delta\mathbf{n}_p \pm r_I\,(\mathbf{n}_p \times \mathbf{e}) + \boldsymbol{\epsilon}, \tag{33}$$

where point $\mathbf{c}_b$ is the centre of the top or the bottom base and Δ is the signed distance between the point on the plane $\mathbf{p}_p$ and $\mathbf{c}_b$ (Eqn. 22). The distance r_I is found from Eqn. 23. As before, $\boldsymbol{\epsilon}$ is in the opposite direction of the plane normal and of magnitude equal to a small fraction of the cylinder maximum dimension (i.e., diameter or length).

2.3. **Pressure boundary condition.** To establish a flow in a tri-periodic domain, we developed an iterative pressure boundary condition which adjusts the pressure jump to the imposed mean velocity.

OpenFOAM® already offers a possibility to specify a pressure jump at the periodic boundaries, i.e., `uniformJump` boundary condition. However, this boundary condition is impractical in situations where the pressure jump is not known a-priori. Another recommended approach [69, 70] is to impose the target mean velocity at the boundaries using a source term (via `patchMeanVelocityForce` function object), while setting the cyclic constraint on the pressure. However, this enforces equal pressure at the inlet and outlet boundaries, affecting the accuracy of the force calculation (see Appendix C).

Our boundary condition is derived from the aforementioned uniform pressure jump boundary condition [71]. Here, instead of a pressure jump value, we impose the value of the target mean velocity through the domain. The actual mean velocity in the domain $\bar{\mathbf{u}}$, projected onto the direction of the target velocity $\mathbf{u}$, is calculated as:

$$\bar{\mathbf{u}} = \frac{\sum_i \left(\frac{\mathbf{u}}{|\mathbf{u}|} \cdot \mathbf{u}_i\right) V_i}{\sum_i V_i}, \tag{34}$$

where $\mathbf{u}_i$ and V_i are respectively the velocity and volume of cell i.

The pressure jump is consequently updated according to the ratio of the current and the target mean velocity:

$$\Delta p^n = \Delta p^o \left(1 - \alpha\left(1 - \frac{|\mathbf{u}|}{|\bar{\mathbf{u}}|}\right)\right), \tag{35}$$

with superscript n denoting the new value, superscript o denoting the value from the previous iteration, and α being a user-defined relaxation factor (default value set to 1). As shown in Fig. 11, the pressure boundary condition needs to be used within a context of an iterative solver, such as `simpleFoam`.

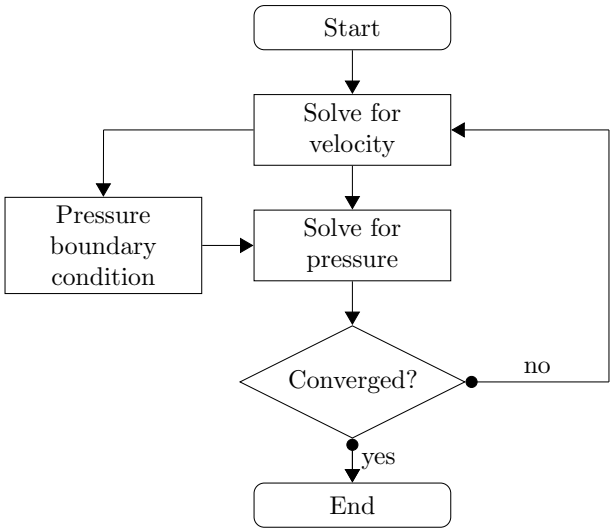


Figure 11. Pressure boundary condition within SIMPLE algorithm

For the initial iteration, an initial value for the pressure jump $(\Delta p)_{init}$ needs to be provided. Using an informed guess for the initial pressure jump (e.g., Ergun pressure drop [72] for problems with particle assemblies), the number of iterations until convergence can be reduced. An example of the convergence behaviour can be seen in subsection 3.1. The influence of the initial pressure jump is examined in Appendix D. Using the boundary condition with `pimpleFoam` is described in Appendix E.

2.4. **Force calculation.** To calculate the forces acting on the particles, we extended the `forces` function object [73] with two new features: 1) pressure correction for periodic neighbours and 2) calculation of all first moment components.

2.4.1. *Pressure correction for periodic neighbours.* Following notation from [14], the hydrodynamic force acting on a particle with surface S is:

$$\mathbf{f} = \oint_S \boldsymbol{\sigma} \cdot \mathbf{n} dS, \tag{36}$$

where $\boldsymbol{\sigma}$ is the stress tensor and $\mathbf{n}$ the unit-normal vector at the particle surface.

In the `forces` function object [73], the above force is split into a pressure and a viscous contribution:

$$\mathbf{f} = \mathbf{f}_p + \mathbf{f}_\mu. \tag{37}$$

The pressure contribution is found from:

$$\mathbf{f}_p = \sum_i \rho_i \mathbf{s}_{f,i} \left(p_i - p_{ref} \right), \tag{38}$$

where subscript i denotes the face-centre values of the boundary cells, ρ is the fluid density, $\mathbf{s}_f$ is the cell face area vector (parallel to the normal at the cell face), p is the local pressure and p_{ref} is the reference pressure value, supplied by the user.

The equation above calls for a special treatment when applied to particles in periodic domains. In such simulations, we are often interested in the total force acting on a single particle. To calculate this force, contributions from particles intersecting the periodic boundaries need to be summed up, as if they were parts of a single particle. However, such summation leads to an unbalanced force if a pressure drop exists between the boundaries. To rectify this, we add a pressure drop correction to the force acting on particles crossing the outlet boundary:

$$[\mathbf{f}_p]_{outlet} = \sum_i \rho_i \mathbf{s}_{f,i} \left(p_i - p_{ref} + \Delta p \right). \tag{39}$$

The pressure drop correction Δp is read from the pressure jump boundary condition detailed in subsection 2.3. With the above addition of Δp, the particle at the outlet boundary effectively experiences the same pressure as the particle at the inlet boundary, correcting the force imbalance that would occur otherwise (for an example see Appendix C).

The viscous contribution from Eqn. 37 is not sensitive to the periodicity and it remains calculated as [73]:

$$\mathbf{f}_{\mu} = \sum_{i} \mathbf{s}_{f,i} \left(\mu \mathbf{R}_{dev} \right), \tag{40}$$

where μ is the dynamic viscosity and $\mathbf{R}_{dev}$ the deviatoric stress tensor.

It is important to note here that the force $\mathbf{f}$ is the total fluid-to-particle force, including the effects due to the local pressure gradient. In order to obtain a force closure consistent with the ensemble/volume averaging theory [7], the pressure gradient contribution needs to be removed. As shown in [74], this force due to the flow alone (without pressure gradient) is simply found from:

$$\mathbf{f}_{flow} = \mathbf{f} \left(1 - \phi_s \right), \tag{41}$$

where ϕ_s is the solid fraction.

2.4.2. *First moment calculation.* Apart from the hydrodynamic force, OpenFOAM® also calculates the hydrodynamic torque (called `moment` in `forces` function object) [73]. However, as detailed in [14], the torque mathematically represents only the antisymmetric part of the first moment of the force.

From the symmetric part of the first moment, we can also calculate the hydrodynamic stresslet [13]. Physically, stresslet represents the resistance to the straining motion [14]. While (unlike force and torque) it does not affect the Lagrangian motion of the particles, stresslet affects the suspension mechanics. For example, stresslet is used in calculation of Einstein's viscosity [75] for dilute suspensions of spheres, and recently, it is studied in contexts of microorganisms [15] and shear-thinning [47]. There is also compelling evidence that stresslet affects turbulent flows [16,17], and could be important in a wide range of applications.

In indicial notation, the first moment of the hydrodynamic force (Eqn. 36) is [14]:

$$M_{ij} = \oint_S \sigma_{ik} n_k r_j dS, \tag{42}$$

where $\boldsymbol{\sigma}$ is the stress tensor, $\mathbf{n}$ the unit-normal at the particle surface, and $\mathbf{r}$ the position vector. The position vector is implemented as the distance of the cell centres from the geometric centre of the particles.

As mentioned before, tensor $\mathbf{M}$ can be re-written as a sum of its symmetric $\mathbf{S}$ and antisymmetric part $\mathbf{A}$:

$$M_{ij} = S_{ij} + A_{ij}, \tag{43}$$

$$S_{ij} = \frac{1}{2} \oint_S \left(\sigma_{ik} r_j + \sigma_{jk} r_i \right) n_k dS, \tag{44}$$

$$A_{ij} = \frac{1}{2} \oint_S \left(\sigma_{ik} r_j - \sigma_{jk} r_i \right) n_k dS = -\frac{1}{2} e_{ijk} t_k. \tag{45}$$

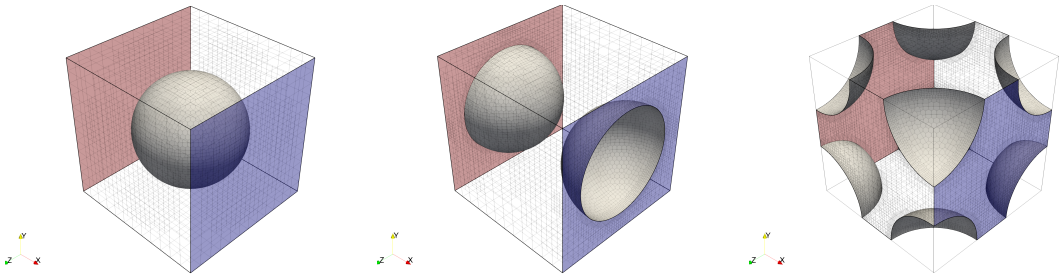
Here, the hydrodynamic stresslet $\mathbf{S}$ is equal to the symmetric part of the first moment, while the hydrodynamic torque $\mathbf{t}$ is found as the Hodge dual of the antisymmetric tensor $\mathbf{A}$. Verification of the stresslet calculation is given in subsection 3.4.

3. **Verification**

This section describes the verification cases. Spherical, cylindrical and spherocylindrical particles are considered respectively in subsection 3.1, subsection 3.2 and subsection 3.3. At the end, in subsection 3.4, we also verify the calculation of stresslet with the analytical solution for a single sphere in simple shear flow.

3.1. **Spherical particles.** For our first test case, we consider spherical particles in a simple cubic array. This case is well-studied in literature: there is a semi-analytical solution for the drag force in Stokes flow [76] and there are several numerical results for the drag force at moderate-Re–flow [8,9,50].

Our simulation domain is depicted in Fig. 12. The domain is cubic, with flow parallel to the x-axis and with enforced periodicity in all three directions. Due to the periodicity, it is sufficient to simulate only one sphere. To verify the force summation for the periodic neighbours (subsubsection 2.4.1), we consider three cases: 1) the sphere is placed in the middle of the domain (Fig. 12a), 2) the centre of the sphere coincides with the centre of the inlet boundary, resulting in a periodic neighbour at the opposite (outlet) boundary (Fig. 12b), 3) the centre of the sphere coincides with one of the domain vertices, resulting in 7 periodic neighbours (Fig. 12c). The three cases are referred to as *full-*, *half-*, and *eight-part–sphere*, respectively.



(a) full-sphere **(b)** half-sphere **(c)** eight-part–sphere

Figure 12. Simple cubic array in tri-periodic computational domain. Flow is in the direction of x-axis, from the red towards the blue boundary. Example shown for solid fraction 0.201.

At the inlet and outlet boundaries we impose the pressure boundary condition from subsection 2.3 with the target mean fluid velocity $\mathbf{u}$. No-slip condition is set at the sphere surface.

Definitions of the relevant parameters, following [8, 76], are given in Tab. 1. Sphere diameter, fluid density and dynamic viscosity are kept constant and, for simplicity, set to $D = 1$ m, $\rho = 1\ \frac{\text{kg}}{\text{m}^3}$, and $\mu = 1$ Pas.

In the Stokes flow regime, the solid fraction is varied between 0.027 and 0.45, while Re is kept constant at 0.01. For the moderate-Re–flow, solid fraction is kept constant at 0.201. Other parameters, such as the domain length (l), superficial velocity ($\mathbf{v}$) and average fluid velocity ($\mathbf{u}$), follow from the relations in Tab. 1.

Table 1. Definitions of relevant parameters for a simple cubic array of spheres according to [8, 76]

Parameter	Definition
Reynolds number	$Re = \frac{\rho r \lvert\mathbf{v}\rvert}{\mu}$
Superficial velocity	$\mathbf{v} = (1 - \phi_s)\,\mathbf{u}$
Solid fraction	$\phi_s = \frac{4}{3}\pi \left(\frac{r}{l}\right)^3$
Dimensionless drag force	$F_D = \frac{f_x}{6\pi\mu r \lvert\mathbf{v}\rvert}$

The mesh resolution needs to be sufficiently fine to capture the boundary layer at the particle surface. The boundary layer thickness is proportional to $D/\sqrt{Re_D}$ [9], where Re_D is the Reynolds number based on the particle diameter D. It should be emphasized here that the definition of the Reynolds number in Tab. 1, according to [8, 76], is based on the particle radius, and therefore amounts to half of Re_D. In order to have at least 3 cells across the boundary layer, we impose a cell length of $D/24$ for $Re < 100$ ($Re_D < 50$), and $D/40$ for $Re \geq 100$ ($Re_D \geq 50$).

To achieve such a resolution in an unstructured mesh, we fix the background mesh resolution (`blockMesh`) to $D/6$ and $D/10$, for $Re < 100$ and $Re \geq 100$, respectively. The surface refinement level is set to 2, reducing the cell length at the sphere surfaces 2^2 times. Configurations with periodic neighbours have additional edge refinement of level 3 at the locations of intersections with periodic boundaries. Between each refinement level, there are 4 layers of cells (i.e., `nCellsBetweenLevels` $= 4$).

Simulations are run with the steady-state solver `simpleFoam`. We consider the simulations converged when the residual values drop below $1 \cdot 10^{-6}$ for $Re < 100$, or below $1 \cdot 10^{-5}$ for $100 \geq Re \geq 250$. After $Re > 250$, the residuals stop decreasing with iterations, pointing to a possible onset of unsteady flow. To illustrate the convergence behaviour, Fig. 13 shows the relative change of pressure jump Δp (Eqn. 35, with initial value $(\Delta p)_{init} = 1$ and relaxation factor $\alpha = 0.2$) against the number of iterations for $Re = 100$.

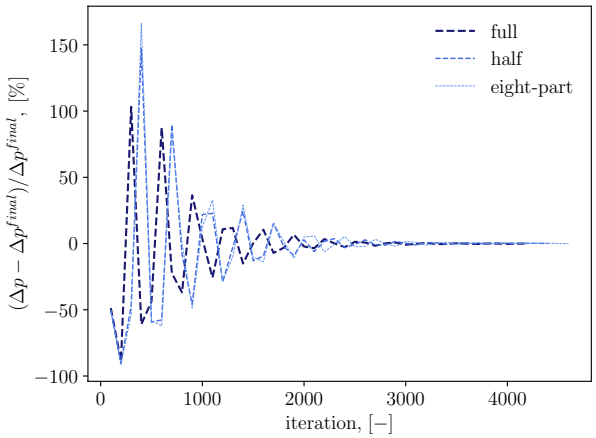
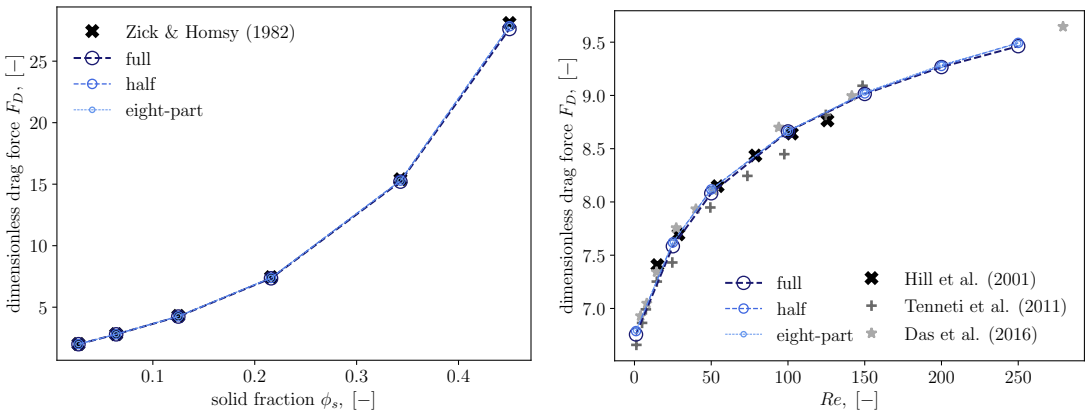


Figure 13. Relative change of pressure jump at $Re = 100$

The results are shown in Fig. 14. In the Stokes flow (Fig. 14a), all three configurations achieved excellent agreement with the semi-analytical solution from [76]. The mean relative deviations for the full-, half-, and eight-part–sphere are respectively 1.14%, 0.7%, and 0.8%. Slightly better agreement of the configurations with periodic neighbours can be attributed to the additional mesh refinement at the intersection edges.

Agreement with the available numerical data is also achieved for the moderate-Re–flow (Fig. 14b). On average, the relative difference between our results and the data from Hill et al. [8], Tenneti et al. [9], and Das et al. [50] stays below 2%. It is important to note that while we used body-fitted DNS, Hill et al. [8] used LBM, and both Tenneti et al. [9] and Das et al. [50] used IBM (different implementations). Good agreement with the results acquired using various methods confirms the validity of our drag force calculation. In the next section, we test all components of the hydrodynamic force and torque.



(a) Stokes flow ($Re = 0.01$) **(b)** Moderate-Re–flow ($\phi_s = 0.201$)

Figure 14. Simulation results for a spherical particle in simple cubic array. Stokes flow results are compared to the semi-analytical data from Zick and Homsy [76]. Moderate-Re–flow results are compared to the numerical data from Hill et al. [8], Tenneti et al. [9] and Das et al. [50].

3.2. **Cylindrical particles.** To test the simulation of cylindrical particles, we use a well-validated fictitious-domain–based solver PeliGRIFF [48, 49].

Our test case contains 16 randomly oriented cylindrical particles placed in a cubic, tri-periodic domain (Fig. 15). Exact positions and orientations of the cylinders can be found in the related case files. The rest of the general parameters is summarized in Tab. 2.

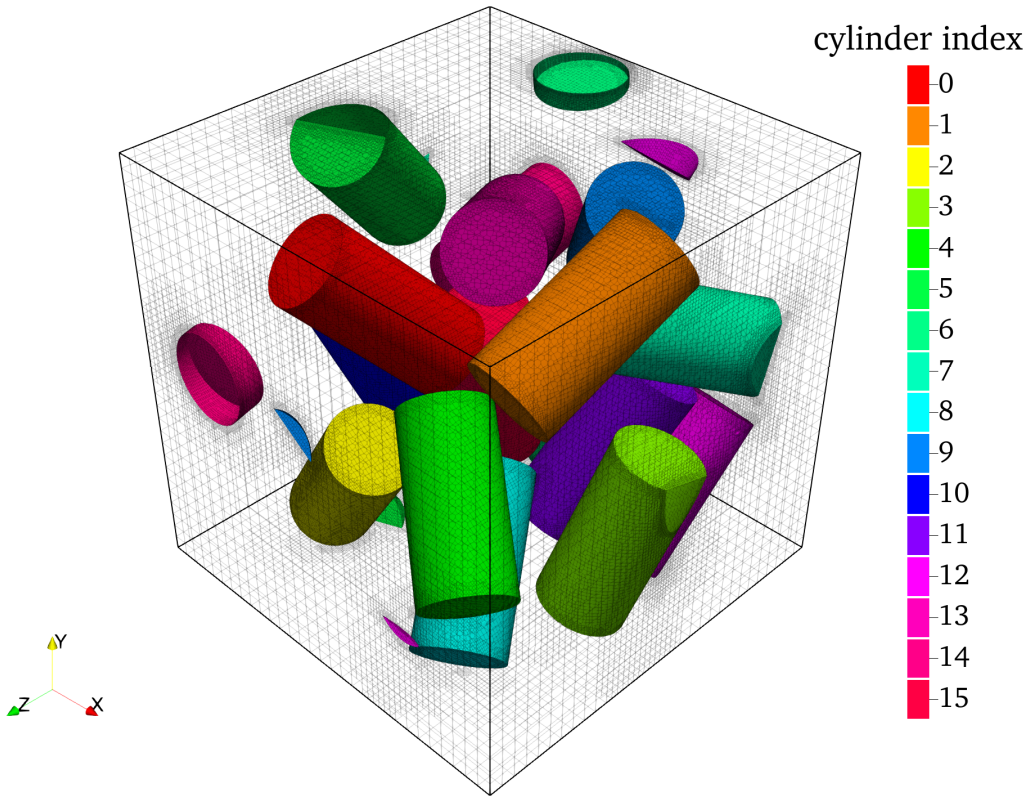


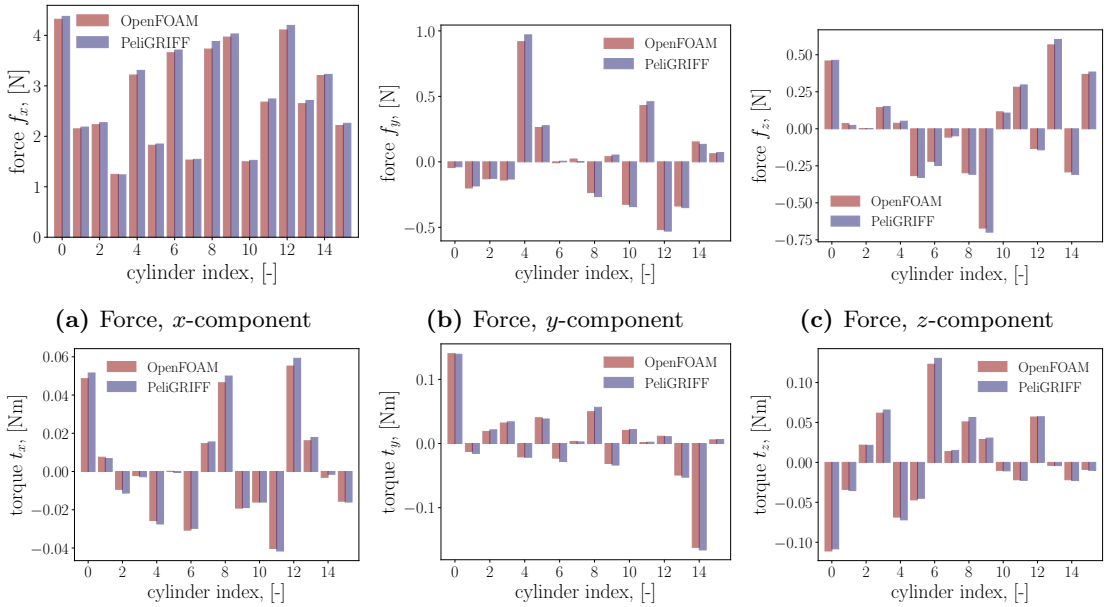
Figure 15. Simulation domain for cylindrical particles

Table 2. Simulation parameters for cylindrical particles

Parameter	Value
Cylinder diameter D	0.2 m
Cylinder aspect ratio AR	2
Superficial velocity $\|\mathbf{v}\|$	0.1 $\frac{\text{m}}{\text{s}}$
Density ρ	1000 $\frac{\text{kg}}{\text{m}^3}$
Reynolds number $Re = \frac{\rho D\|\mathbf{v}\|}{\mu}$	20
Domain length l	1 m
Minimum point of the domain	(0.2, 0.2, 0.2) m
Maximum point of the domain	(1.2, 1.2, 1.2) m

In the OpenFOAM® simulation, the size of the background mesh is set to 1/5 of the cylinder diameter. The refinement level is set to 2 for the surfaces, and to 3 for the edges and gaps, leading to a final, unstructured mesh with ≈ 650k cells. Further increase in the mesh fidelity (background mesh cell reduced to $D/6$) had little influence on the results, yielding on average less than 0.5% relative difference for the drag force. In the PeliGRIFF simulation, the mesh is uniform with 20 cells per cylinder diameter. The spatial numerical scheme is second order accurate.

As seen in Fig. 16, nearly identical results are computed for each cylinder, both in terms of the hydrodynamic force f_i and torque t_i. The drag force (i.e., component f_x, Fig. 16a) is found to have the most significant impact, being several times greater in magnitude than the other components. Here, the agreement between the results is excellent, with the mean relative difference below 2% and the maximum difference below 4%. The largest discrepancy overall is found in the components tending to zero (for example, f_y for cylinder 7 or t_x for cylinder 5). Excluding such small values from the consideration, the relative difference on average stays below 2% for all force and torque components.

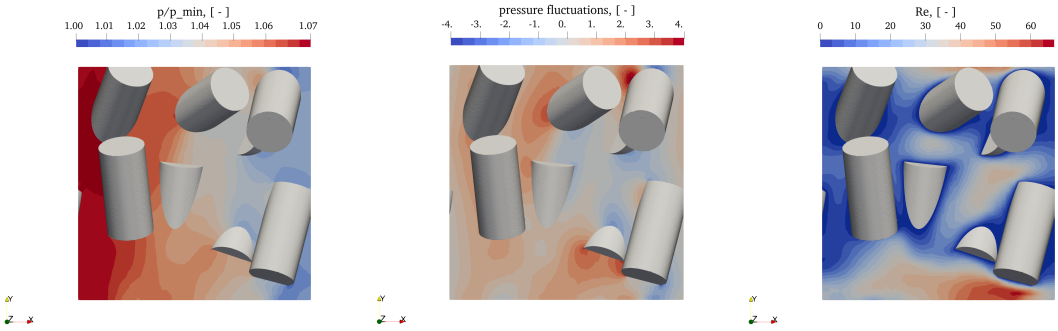


(a) Force, x-component **(b)** Force, y-component **(c)** Force, z-component

(d) Torque, x-component **(e)** Torque, y-component **(f)** Torque, z-component

Figure 16. Components of the hydrodynamic force and torque acting on individual cylinders in a tri-periodic domain. Comparison of the simulation results produced by OpenFOAM and by PeliGRIFF.

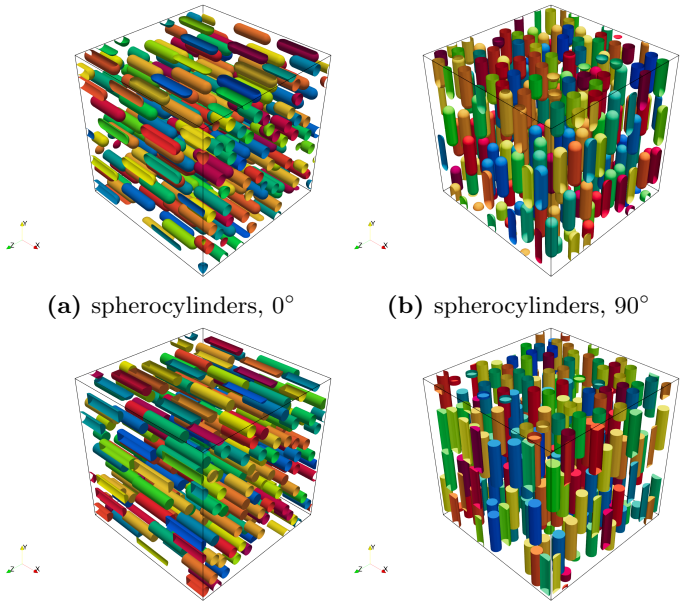
Converged dimensionless pressure and velocity fields are shown in Fig. 17 (sampled at the middle of the domain).



(a) Pressure **(b)** Pressure fluctuations **(c)** Velocity

Figure 17. Dimensionless pressure and velocity fields sampled at the middle of the domain. Pressure fluctuations are normalized as: $\left(p - \Delta p \cdot \frac{x}{l}\right) / \left(\frac{|\mathbf{v}|^2}{2}\right)$, where Δp is the pressure jump between boundaries [Pa m^3/kg] and x the distance.

3.3. **Spherocylindrical particles.** For our penultimate test case, we consider tri-periodic packings of 200 spherocylinders, as studied by Sanjeevi and Padding [22]. The spherocylinders are of aspect ratio 4, and of orientation either parallel to the incoming velocity vector (0° configuration, Fig. 18a), or normal to it (90° configuration, Fig. 18b). Additionally, we consider similar configurations of cylindrical particles, of the same aspect ratio (Figs. 18c and 18d). The unidirectional random assemblies are generated by setting the global orientation vector (Eqn. 3) to $[1, 0, 0]$ in 0° configurations and to $[0, 1, 0]$ in 90° configurations.



(a) spherocylinders, 0° **(b)** spherocylinders, 90°

(c) cylinders, 0° **(d)** cylinders, 90°

Figure 18. Configurations used for comparison of drag force in assemblies of spherocylinders and cylinders. Each domain contains 200 particles of aspect ratio 4. Solid fraction is 0.2. Inlet velocity is parallel to the x-axis.

As in the previous test cases, we define the relevant quantities following the reference work [22]. These are summarized in Tab. 3, while Tab. 4 summarizes parameters that are kept constant in our simulations.

Table 3. Definitions of relevant parameters used in simulations of spherocylindrical particles according to [22]

Parameter	Spherocylinders	Cylinders
Diameter of a volume-equivalent sphere	$D_{eq} = D\sqrt[3]{\frac{3AR-1}{2}}$	$D_{eq} = D\sqrt[3]{\frac{3AR}{2}}$
Reynolds number	$Re = \rho D_{eq}\lvert\mathbf{v}\rvert/\mu$	
Dimensionless drag force	$F_D = f_x\,(1-\phi_s)\,/\,(3\pi\mu D_{eq} v_x)$	

Table 4. Simulation parameters for spherocylindrical and cylindrical particles

Parameter	Value
Particle diameter D	1 m
Particle aspect ratio AR	4
Solid fraction ϕ_s	0.2
Number of particles N_p	200
Fluid density ρ	1 $\frac{\text{kg}}{\text{m}^3}$
Dynamic viscosity μ	1 Pas

The mesh resolution is specified by setting the background mesh size to 6 cells per particle diameter, surface refinement level to 3 and edge refinement level to 4. This leads to at least 46 cells per particle diameter. The final mesh contains around 15 million cells for spherocylinders and around 29.5 million cells for cylinders. The finer resolution for cylinders is partially due to the presence of sharp edges, which increases the number of cells, but also due to the larger domain needed to achieve the same solid fraction. To avoid meshing errors, the minimum distance between the particles is set to 30% of particle radius, minimum distance from the boundaries is set to 10% of the radius, and minimum allowed overlap with a boundary is set to 25% of the radius.

While the initialization of particle assemblies took around 2 s, the meshing on average took less than 30 min for spherocylinders and less than 45 min for cylinders. The average convergence time for the slowest simulation ($Re = 100$) is around 2 h for spherocylinders and 5 h for cylinders (Tab. 5). We used Intel®Xeon®Gold 6126 CPU @ 2.60GHz processor with 24 cores.

Table 5. Execution times using Intel®Xeon®Gold 6126 CPU @ 2.60GHz processor with 24 cores

Case	assembly generation	meshing	$Re = 1$	$Re = 100$
spherocylinders, 0°, run 1	1.58 s	28 min	103 min	130 min
spherocylinders, 0°, run 2	1.4 s	25 min	98 min	114 min
spherocylinders, 0°, run 3	1.38 s	27 min	96 min	110 min
spherocylinders, 90°, run 1	1.97 s	27 min	113 min	140 min
spherocylinders, 90°, run 2	1.51 s	27 min	105 min	123 min
spherocylinders, 90°, run 3	1.84 s	26 min	119 min	155 min
cylinders, 0°, run 1	1.04 s	46 min	199 min	250 min
cylinders, 0°, run 2	1.07 s	47 min	203 min	256 min
cylinders, 0°, run 3	1.53 s	39 min	209 min	246 min
cylinders, 90°, run 1	1.3 s	38 min	171 min	246 min
cylinders, 90°, run 2	1.24 s	40 min	191 min	256 min
cylinders, 90°, run 3	1.17 s	45 min	171 min	246 min

The results are shown in Fig. 19. The dimensionless drag force is based on the mean force acting on an individual particle. Three runs are performed for each configuration. The maximum relative deviations between the runs—due to the differences in random configurations—are 2.56% for spherocylinders and 4.35% for cylinders. Comparing to the drag fit for spherocylinders from [22, 77], the average relative difference is 1.9% for 0° and 4.7% for 90° configuration. The maximum deviation of 5.72% is found for 90° configuration at $Re = 100$.

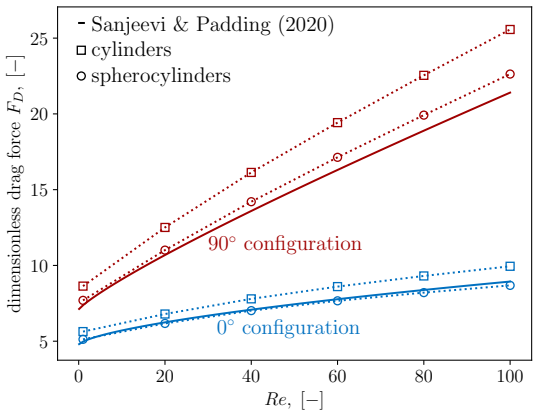
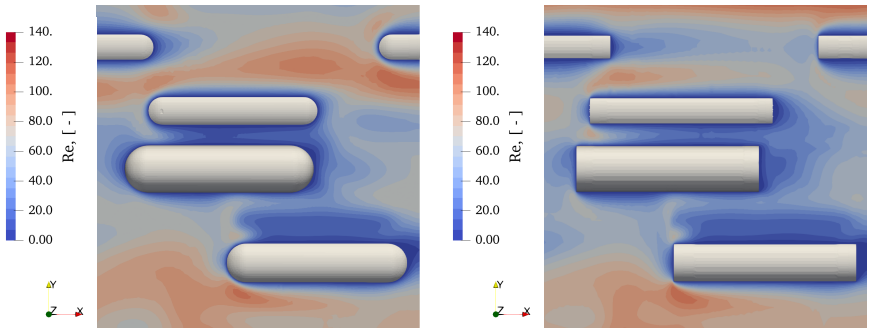


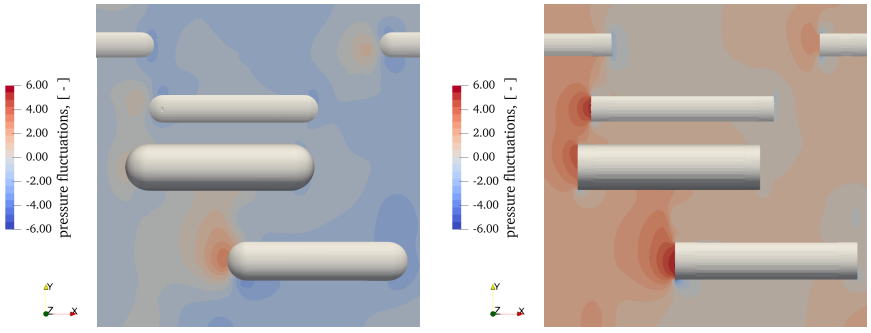
Figure 19. Drag force in uniaxial random assemblies of spherocylinders and cylinders with aspect ratio 4 and solid fraction 0.2. Simulation data is based on the mean force acting on an individual particle in tri-periodic assemblies with 200 particles. Three independent runs were performed for each configuration. Reference data for spherocylinders is taken from [22].

After verifying the simulation, we can move on to the comparison of spherocylindrical and cylindrical particles. As expected, the cylinders experience higher drag due to their blunt shape [24, 25]. As shown in Figs. 20 and 21, the sharp edges of the cylinders cause flow separation. This induces higher pressure fluctuations and therefore higher drag. On the other hand, spherocylinders are more streamlined, and therefore experience less drag. Specifically, the offset of the simulation results for cylinders and spherocylinders stays around 13% (Fig. 19). This finding suggest that, for particles of aspect ratio 4, the drag fit for spherocylinders could also be used for cylinders if amended with a correction factor.



(a) spherocylinders **(b)** cylinders

Figure 20. Detail of the dimensionless velocity field (Re) for the average $Re = 60$. Snapshots of a smaller test case with 40 particles.



(a) spherocylinders **(b)** cylinders

Figure 21. Detail of the dimensionless pressure fluctuation field for the average $Re = 60$. Pressure fluctuations are normalized as: $\left(p - \Delta p \cdot \frac{x}{l}\right) / \left(\frac{|\mathbf{v}|^2}{2}\right)$, where Δp is the pressure jump between boundaries [Pa m^3/kg] and x the distance. Snapshots of a smaller test case with 40 particles.

3.4. **Stresslet verification.** Last, to test our calculation of the stresslet, we use the analytical solution for a single sphere in simple shear flow [14].

The test case consists of a sphere placed at the centre of a cubic domain (Fig. 22a). Velocities of equal magnitudes, but of opposite directions, are imposed at the boundaries normal to the y-axis, such that the field far away from the sphere is:

$$\mathbf{u}^\infty = \begin{bmatrix} y & 0 & 0 \end{bmatrix}. \tag{46}$$

No-slip velocity constraint is set at the surface of the sphere, while cyclic constraint is imposed at the remaining boundaries. Meshing is similar as in the previous cases, i.e.: there are 6 cells across the sphere diameter in the background mesh, and the surface refinement level is set to 3.

Under the assumption of Stokes flow, the motion can be decomposed into straining and rotation, both of equal magnitudes. Since we are interested in the stresslet, we focus only on the rate-of-strain tensor:

$$E_{ij}^\infty = \frac{1}{2}\left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i}\right). \tag{47}$$

Using Eqn. 46, the only non-zero components of the above tensor are $E_{xy}^\infty = E_{yx}^\infty = 0.5$.

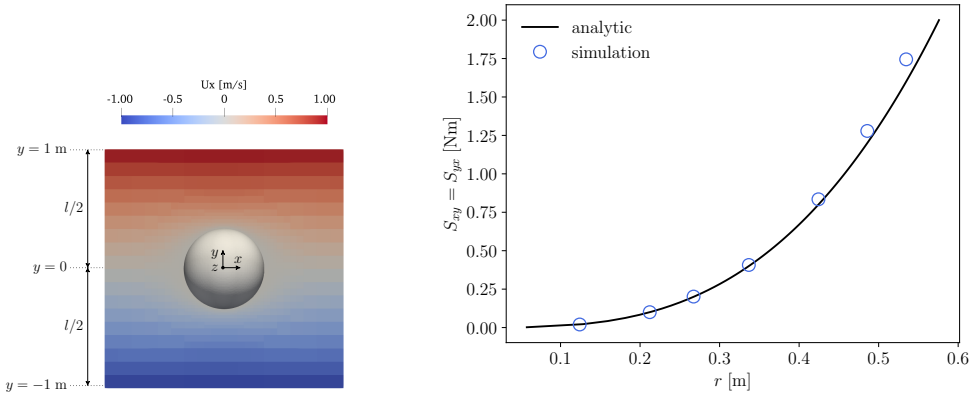
As shown in [14], the stresslet components can be found from:

$$S_{ij} = \frac{20\pi}{3}\mu r^3 E_{ij}^\infty, \tag{48}$$

where r is the sphere radius. For the rest of the parameters—similar as in [78]—we use a dynamic viscosity of $\mu = 1$ Pas and a total length of the domain edge of $l = 2$ m. The sphere size varies according to the imposed solid fraction.

Comparison of the simulation results and Eqn. 47 is shown in Fig. 22b. The graph shows the non-zero stresslet components in relation to the particle radius. In the range of tested solid fractions (0.001 to

0.08), the mean relative deviation from the analytical solution is 3.17%. For small particles ($r < 0.4$ m, i.e., $\phi_s < 0.04$), the deviation stays below 2%. Maximum error of 9.04% is found for the largest particle. This is expected since the influence of the boundary layer around the larger spheres reaches the domain boundaries and invalidates the simple shear assumption. Nonetheless, excellent agreement found for the small particles verifies the correctness of our stresslet calculation.



(a) Steady-state flow field for solid fraction 0.02

(b) Comparison of the analytical solution for the stresslet [14] and the simulation results

Figure 22. Single sphere in simple shear flow

4. Conclusion

We developed an OpenFOAM® workflow for automatic pre- and post-processing for body-fitted DNS of flows through assemblies of spheres, cylinders or spherocylinders. The workflow allows for a fast generation of moderately dense assemblies ($\phi_s \leq 0.3$) of non-overlapping particles, featuring periodicity, size distribution, and random positions and orientations. Common meshing errors are minimized through controlling the distance between the particles, eliminating the tangentiality to the boundaries, and explicitly defining the edge meshes. We improve the accuracy of the force calculation with an iterative pressure boundary condition. This boundary condition enforces a pressure jump such that the target mean velocity is established in the domain. Post-processing is expanded to include all components of the first moment of the force.

Our method is carefully verified against available numerical data, showing excellent agreement with the results from LBM, IBM and FDM simulations. Additionally, we show a direct comparison between the drag force in uniaxial random assemblies of cylinders and spherocylinders of aspect ratio 4 and solid fraction 0.2. The less streamlined shape of a cylinder leads to around 13% greater mean drag force compared to a spherocylinder of the same aspect ratio. While this finding suggests that both types of particles could be described using a similar closure law, more research needs to be done in order to expand the range of studied solid fractions, aspect ratios and Reynolds numbers. Taking into account the efficiency of OpenFOAM® for this type of simulations—geometry generation typically takes a few seconds, meshing takes less than an hour and simulation converges in a few hours—it is feasible to undertake such a study with a relatively low computational effort.

In the future, the following aspects of our work could be improved:

- A more advanced algorithm should be implemented to allow generation of dense packings;
- For dense packings, meshing of particles in contact needs to be addressed;
- New classes representing other particles shapes could be added, as well as classes that would enable representing a mixture of particle shapes;
- Efficiency of the algorithm could be further improved by parallelization.

The numerical methodology presented in this work will be applied in future studies to close the averaged equations of motion for systems involving non-spherical particles. Such particles are commonly used in industrial processes. However, their irregular geometries—particularly when elongated or flattened—pose challenges in operations such as pneumatic conveying. To better understand the hydrodynamic behaviour of these materials, we will conduct direct numerical simulations of fluid flow through assemblies of randomly oriented cylindrical particles. Additionally, the current methodology enables investigation into the inertial rheology of the bulk phase. In particular, it allows for the computation of terms such

as stresslet, which are needed in determining the bulk stress in particle suspensions. Beyond that, the workflow can be applied to similar analyses of spherical and spherocylindrical particles, as well as of polydisperse systems.

Acknowledgements

This work is a part of the PHOBARS project (Pneumatic Handling Of Bio And Recycled Solids) with funding received from the ANR (ANR-21-CE50-0032).

Author Contributions: Conceptualisation, J.M., L.G., J-L.P. and M.M.; methodology, J.M. and N.F.; software, J.M., N.F. and L.G.; validation, J.M. and J-L.P.; formal analysis, J.M.; investigation, J.M.; writing—original draft preparation, J.M.; writing—review and editing, L.G., J-L.P. and M.M.; visualisation, J.M.; supervision, L.G., J-L.P. and M.M.; project administration, M.M.; funding acquisition, M.M. All authors have read and agreed to the published version of the manuscript.

Appendix A. Collision between spherocylinders

Here we briefly describe the collision algorithm for spherocylindrical particles. The collision check is based on an algorithm for finding the shortest distance between two rods [61], adapted for spherocylinders in [62].

In the first step, the following quantities are calculated for two non-parallel spherocylinders with centres $\mathbf{c}_i$ and $\mathbf{c}_j$:

$$a = \mathbf{a}_i \cdot \mathbf{a}_i, \tag{49}$$
$$b = -\mathbf{a}_i \cdot \mathbf{a}_j, \tag{50}$$
$$c = \mathbf{a}_j \cdot \mathbf{a}_j, \tag{51}$$
$$d = \mathbf{a}_i \cdot (\mathbf{c}_i - \mathbf{c}_j), \tag{52}$$
$$e = -\mathbf{a}_j \cdot (\mathbf{c}_i - \mathbf{c}_j), \tag{53}$$
$$f = ac - b \cdot b, \tag{54}$$

where $\mathbf{a} = \frac{1}{2}(L - D)\,\mathbf{e}$ is the half-length of the shaft of the spherocylinder. Using the above expressions, quantities T, t, S and s are found:

$$T = \frac{bd - ae}{f}, \tag{55}$$
$$t = F(T), \tag{56}$$
$$S = \frac{bt - d}{a}, \tag{57}$$
$$s = F(S), \tag{58}$$

where:

$$F(x) = \begin{cases} -1, & \text{for } x \leq -1, \\ 1, & \text{for } x \geq 1, \\ x, & \text{for } -1 < x < 1, \end{cases} \tag{59}$$

If $s \neq S$, T and t need to be re-evaluated using:

$$T = \frac{-bs - e}{c}, \tag{60}$$

and Eqn. 56.

Next, vectors $\mathbf{s}_i$ and $\mathbf{s}_j$ follow from:

$$\mathbf{s}_i = \mathbf{c}_i + s\mathbf{a}_i, \tag{61}$$
$$\mathbf{s}_j = \mathbf{c}_j + t\mathbf{a}_j. \tag{62}$$

Finally, the collision is evaluated using:

$$|\mathbf{s}_i - \mathbf{s}_j| \leq r_i + r_j + \epsilon, \tag{63}$$

where the minimum allowed distance ϵ is defined as Eqn. 13.

Collision of two parallel spherocylinders is tested using Eqn. 12.

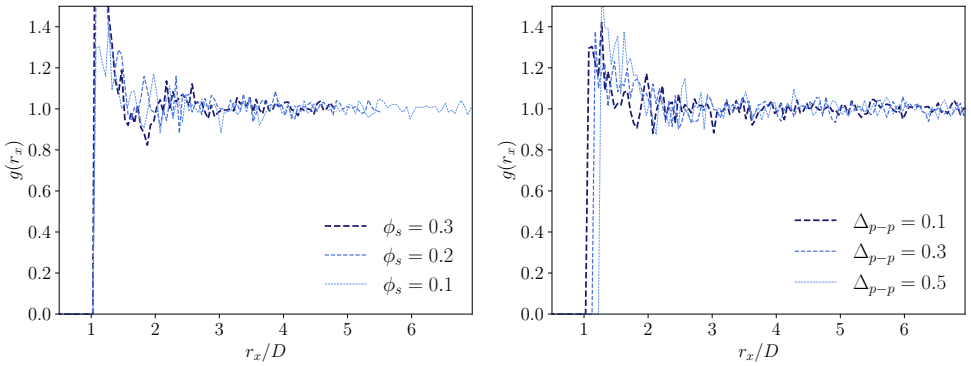
Appendix B. Selection of optimal parameters for meshing

To minimize meshing errors, we introduced parameters that control the minimum distance between the particles Δ_{p-p}, as well as parameters that control minimum distance and overlap in relation to the boundaries, Δ_{mo} and Δ_{md}. As described in subsubsection 2.1.5 and subsubsection 2.1.4, these values are defined relative to the particle size. For example, for spheres, these values represent a fraction of the radius. The optimal parameters for the mesh will highly depend on the considered case. Nonetheless, as a general starting point, we recommend the following procedure:

(1) Decide on the minimum cell size according to the target Reynolds number: there should be at least 3 cells in the boundary layer, and the boundary layer thickness is proportional to $D/\sqrt{Re}$ [9]. Additionally, even in the creeping flow regime, it is recommended to have at least 40 cells across the particle diameter D [9].
(2) Set the distance between the particles Δ_{p-p} (as well as the distance between particles and boundaries Δ_{md}) such that there are at least 7 cells in-between.
(3) Ensure that `maxGlobalCells` in `snappyHexMeshDict` is sufficiently large to cover the overall target number of cells (this depends on the number of particles and the imposed refinement levels).

For relatively coarse meshes, we recommend large Δ_{p-p}, i.e., from 0.1 to 0.3. In subsection 3.3 we achieved good agreement with the reference data [22] using $\Delta_{p-p} = 0.3$. However, we advise against increasing Δ_{p-p} beyond 0.3 as it might adversely influence the microstructure. Microstructure of granular media can be characterized using pair distribution function $g(r_x)$ [14,79]. The pair distribution function indicates likelihood of finding another particle at a radial distance r_x from the reference particle. According to the hard sphere assumption [14,79], the value of $g(r_x)$ is 0 at $r_x < D$ and 1 at $r_x \gg D$. Such a distribution is valid for particles which interact with an infinite potential when colliding. As shown in Fig. 23a, using $\Delta_{p-p} = 0.1$ produces a similar pair distribution function as the hard sphere theory. However, it is important to note that increasing Δ_{p-p} also increases the range of $g(r_x) = 0$ (Fig. 23b) causing a larger departure from the hard sphere assumption.

It is important to note that the effects of the microstructure on the average flow properties are poorly understood. For example, the procedure for finding the pair distribution of cylindrical and spherocylindrical particles is, to the best of our knowledge, currently unavailable, although a similar procedure does exist for ellipsoids [80].



(a) $\Delta_{p-p} = 0.1$ **(b)** $\phi_s = 0.1$

Figure 23. Influence of solid fraction ϕ_s and imposed interparticle distance Δ_{p-p} on the pair distribution function. Examples shown for monodisperse assembly with 1000 spheres.

For the minimum distance from the boundary Δ_{md}, we recommend selecting a value similar to Δ_{p-p}. It is more difficult to devise a procedure for the minimum overlap with the boundary Δ_{mo}. Based on experience, we recommend using $\Delta_{mo} = 0.25$, at least for grid sizes comparable to cases in subsection 3.3. One thing to consider is that some overlap needs to be allowed for tri-periodic systems. No overlap and a large minimum distance from the boundary will produce a non-uniform solid fraction, with a dense region in the centre of the domain and a dilute region near the boundaries. In turn, this leads to inaccurate mean forces based on the overall solid fraction.

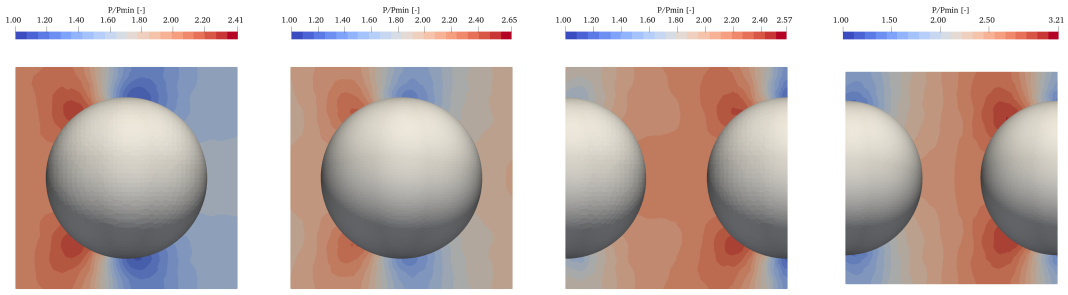
Appendix C. On the necessity for a pressure jump boundary condition

In subsection 2.3, we introduce an iterative boundary condition for the pressure jump between the periodic boundaries. The purpose of this boundary condition is to enforce the flow through a periodic domain. Alternatively, it is reported [69, 70] that the flow can be established using a source term in the momentum equation. This is achieved by setting the desired mean velocity with the `patchMeanVelocityForce` function object, while imposing the cyclic boundary condition for the pressure.

While the approach with the momentum source appears to yield correct results for some types of periodic problems [69, 70], we find it to be inaccurate when applied to force calculation. We demonstrate this with the previously described test cases for spheres (subsection 3.1) and cylinders (subsection 3.2). All parameters are kept as before, except now the flow is enforced using a momentum source.

First, we consider the simple cubic array of spheres in *full* and *half* configuration (Fig. 24). As seen in Fig. 25, the difference in the drag force for a sphere fully enclosed within the boundaries is minimal, around 1.6%. However, the average relative difference between the two methods for the half-sphere is 42%. This large deviation is due to the force imbalance which occurs when summing up the contributions from periodic neighbours.

While the force imbalance can be rectified by adding a pressure correction to one of the periodic neighbours (see subsubsection 2.4.1), there is currently no obvious way to do so for the momentum source approach since the cyclic constraint equalizes the pressure at the inlet and outlet boundaries (Fig. 24).



(a) full sphere, pressure jump **(b)** full sphere, momentum source **(c)** half sphere, pressure jump **(d)** half sphere, momentum source

Figure 24. Snapshots of non-dimensionalized pressure field at $Re = 100$

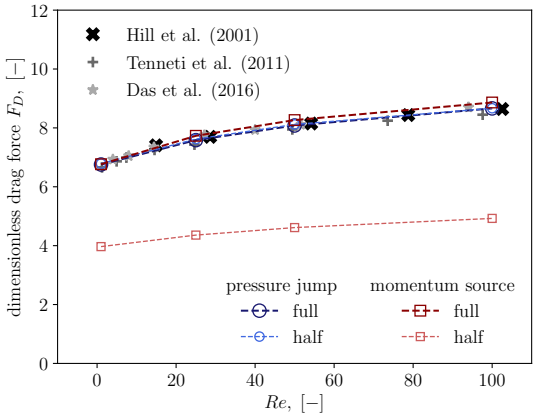


Figure 25. Comparison between the two approaches of establishing flow in a periodic domain: pressure jump boundary condition and source term in the momentum equation. OpenFOAM® results are compared to the numerical data from Hill et al. [8], Tenneti et al. [9] and Das et al. [50].

Next, we calculate the difference from the PeliGRIFF simulation for cylindrical particles. While the mean difference for the drag force was less than 2% when using the pressure jump boundary condition

(subsection 3.2), now, with the momentum source approach, the difference increased to 10.9%. Even when excluding the cylinders that need pressure jump correction (cylinders 3 and 7), the results are still noticeably less accurate, with the mean error of 9% and maximum error of 17%. Similar trends can be observed also for the other components of hydrodynamic force and torque. However, for brevity, we here show only the results for the drag force in Fig. 26.

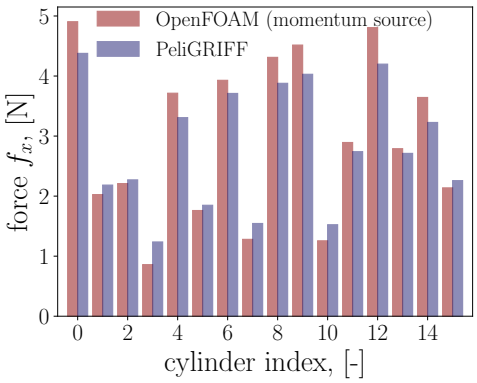


Figure 26. Drag force acting on cylinders (subsection 3.2). Comparison of OpenFOAM® simulation with momentum source and PeliGRIFF simulation.

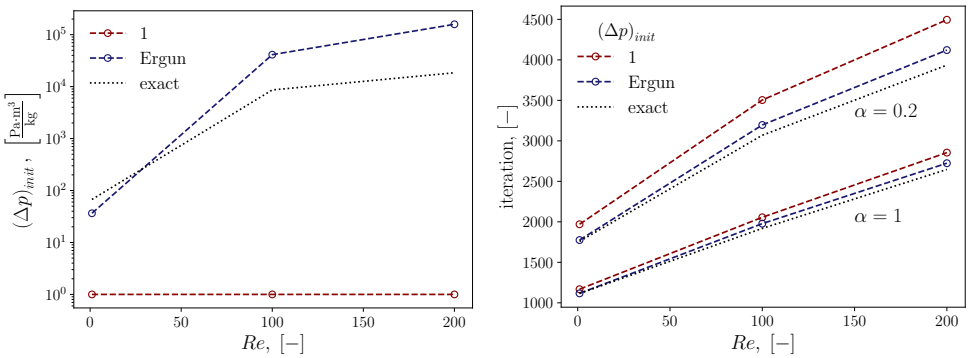
Appendix D. Initial pressure jump selection

This section helps the user define the initial pressure jump $(\Delta p)_{init}$ needed for the pressure jump boundary condition (subsection 2.3). To show how the initial pressure jump guess affects the convergence, we use the simple cubic array test case in full sphere configuration (subsection 3.1, Fig. 12a). As shown in Fig. 27a, we compare three different cases for $(\Delta p)_{init}$. In the first case (red line), $(\Delta p)_{init}$ is set simply to unity. In the second case (blue line), $(\Delta p)_{init}$ is calculated according to the Ergun's pressure drop [72]:

$$(\Delta p)_{init} = \frac{150\nu l}{D^2}|\mathbf{v}|\frac{\phi_s^2}{(1-\phi_s)^3} + \frac{1.75}{D}|\mathbf{v}|^2\frac{\phi_s}{(1-\phi_s)^3}, \tag{64}$$

where l is the length of the cubic domain, ν kinematic viscosity, D particle diameter, $\mathbf{v}$ superficial velocity, and ϕ_s solid fraction. The equation is normalized with fluid density, as usual for incompressible solvers like `simpleFoam`. In the third case (black line), we use the converged value of the pressure jump as the initial value. Additionally, we test two values for the relaxation factor α (Eqn. 35): 0.2 and 1.

As seen in Fig. 27b, if the initial guess for $(\Delta p)_{init}$ is close to the final pressure jump, the number of iterations until convergence will be reduced. However, this reduction is not dramatic and a reasonable time until convergence can be expected even with $(\Delta_p)_{init}$ set to an arbitrary value.



(a) initial value **(b)** number of iterations until convergence

Figure 27. Influence of initial pressure jump selection on the convergence. Example shown for simple cubic array in full sphere configuration at $\phi_s = 0.201$.

Appendix E. Using the workflow with transient solvers

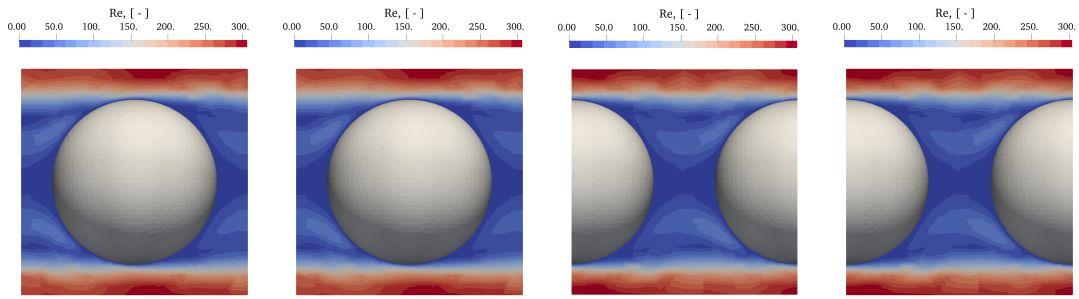
In our present study, we focus on steady state flows, which can be efficiently handled with `simpleFoam`. However, since transient effects are important in many applications, we also briefly discuss how to use the workflow in such cases. The main difficulty is that the pressure boundary condition (subsection 2.3) requires an iterative solver and is therefore incompatible with non-iterative transient flow algorithms such as PISO [81]. Nonetheless, the boundary condition works with transient solvers such as `pimpleFoam`, which allow for iterations between the time steps [82]. Alternatively, the workflow can be used even with non-iterative solvers if a different boundary condition is applied, e.g., `uniformJumpAMI`. In this case, user specifies a constant pressure jump as the means to drive the flow. To solve the force imbalance over the periodic neighbours (see Appendix C), the pressure jump correction needs also to be explicitly assigned in `searchableSurfacesDict` (as shown in the related case files). As the input pressure jump value for `uniformJumpAMI` boundary condition, we used the final Δp^{final} obtained with the iterative pressure boundary condition.

Next, we test the two aforementioned boundary conditions with `pimpleFoam`, using the simple cubic array case in full- and half-sphere configuration (subsection 3.1) at solid fraction 0.201 and with imposed Reynolds number of 300. It is important to mention that we were unable to reach convergence for $Re = 300$ with `simpleFoam` and the iterative pressure boundary condition. Nonetheless, `pimpleFoam` with 100 iterations per time step (i.e., `nOuterCorrectors`) and a time step of 0.001 s (mean Courant number of ≈ 15) reached steady state at the time of approximately 0.1 s. The resulting dimensionless drag forces in Tab. 1 were sampled at the time 0.3 s. The final pressure jump obtained with the iterative pressure boundary condition (subsection 2.3) was used as the input for the cases with constant pressure jump boundary condition (i.e., `uniformJumpAMI`). Both boundary conditions produced nearly identical dimensionless drag forces F_D (Tab. 6), as well as final velocity (Fig. 28) and pressure fields (Fig. 29).

The highest Re considered in literature [50] is 280, with the reported dimensionless drag force of 9.65. The solutions in Tab. 6 deviate from this at most by 1.15%. This additionally verifies the applicability of the workflow to `pimpleFoam`.

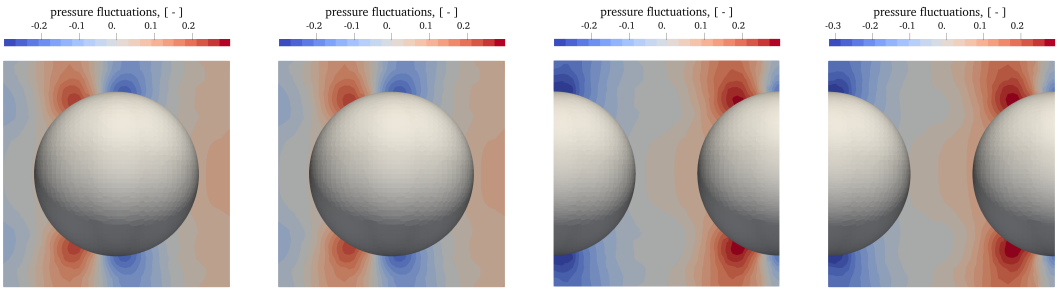
Table 6. Final dimensionless drag force sampled at time $t = 0.3$ s for simple cubic array test case at $Re = 300$. Solutions computed with `pimpleFoam` and with an iterative and a constant pressure jump boundary condition. The final pressure jump Δp^{final} obtained with the iterative pressure boundary condition is imposed as the input for the constant pressure jump boundary condition.

	full sphere, iterative pressure jump	full sphere, constant pressure jump	half sphere, iterative pressure jump	half sphere, constant pressure jump
F_D	9.66, [−]	9.64, [−]	9.76, [−]	9.75, [−]
$\frac{\Delta p^{final}}{0.5\rho\lvert\mathbf{v}\rvert^2}$	0.161, [−]	–	0.163, [−]	–



(a) full sphere, iterative pressure jump **(b)** full sphere, constant pressure jump **(c)** half sphere, iterative pressure jump **(d)** half sphere, constant pressure jump

Figure 28. Snapshots of dimensionless velocity field for $Re = 300$ at $t = 0.3$ s. Simulations run using `pimpleFoam` with an iterative pressure boundary condition (subsection 2.3) and a constant pressure jump boundary condition (`uniformJumpAMI`).



(a) full sphere, iterative pressure jump **(b)** full sphere, constant pressure jump **(c)** half sphere, iterative pressure jump **(d)** half sphere, constant pressure jump

Figure 29. Snapshots of dimensionless pressure fluctuations field for $Re = 300$ at $t = 0.3$ s. Pressure fluctuations are normalized as: $\left(p - \Delta p^{final} \cdot \frac{x}{l}\right) / \left(\frac{|\mathbf{v}|^2}{2}\right)$, where Δp^{final} is the pressure jump between the boundaries [Pa m^3/kg] and x the distance. Simulations run using `pimpleFoam` with an iterative pressure boundary condition (subsection 2.3) and a constant pressure jump boundary condition (`uniformJumpAMI`).

References

[1] N. G. Deen, E. A. J. F. Peters, J. T. Padding, and J. A. M. Kuipers, "Review of direct numerical simulation of fluid–particle mass, momentum and heat transfer in dense gas–solid flows," *Chemical Engineering Science*, vol. 116, pp. 710–724, 2014.

[2] J. Zhao, S. Zhao, and S. Luding, "The role of particle shape in computational modelling of granular matter," *Nature Reviews Physics*, vol. 5, pp. 505–525, 2023.

[3] S. Kuang, M. Zhou, and A. Yu, "CFD-DEM modelling and simulation of pneumatic conveying: a review," *Powder Technology*, vol. 365, pp. 186–207, 2020.

[4] T. B. Anderson and R. Jackson, "A fluid mechanical description of fluidized beds," *Industrial and Engineering Chemistry Fundamentals*, vol. 6, pp. 527–539, 1967.

[5] R. Jackson, "Locally averaged equations of motion for a mixture of identical spherical particles and a Newtonian fluid," *Chemical Engineering Science*, vol. 52, pp. 2457–2469, 1997.

[6] D. Z. Zhang and A. Prosperetti, "Averaged equations for inviscid disperse two-phase flow," *Journal of Fluid Mechanics*, vol. 267, pp. 185–219, 1994.

[7] Z. Y. Zhou, S. B. Kuang, K. W. Chu, and A. B. Yu, "Discrete particle simulation of particle-fluid flow: model formulations and their applicability," *Journal of Fluid Mechanics*, vol. 661, pp. 482–510, 2010.

[8] R. J. Hill, D. L. Koch, and A. J. C. Ladd, "Moderate-Reynolds-number flows in ordered and random arrays of spheres," *Journal of Fluid Mechanics*, vol. 448, pp. 243–278, 2001.

[9] S. Tenneti, R. Garg, and S. Subramaniam, "Drag law for monodisperse gas–solid systems using particle-resolved direct numerical simulation of flow past fixed assemblies of spheres," *International Journal of Multiphase Flow*, vol. 37, pp. 1072–1092, 2011.

[10] R. Beetstra, M. A. van der Hoef, and J. A. M. Kuipers, "Drag force of intermediate Reynolds number flow past mono- and bidisperse arrays of spheres," *AIChE journal*, vol. 53, pp. 489–501, 2007.

[11] Z. Cheng and A. Wachs, "Hydrodynamic force and torque fluctuations in a random array of polydisperse stationary spheres," *International Journal of Multiphase Flow*, vol. 167, p. 104524, 2023.

[12] B. Siddani and S. Balachandar, "Point-particle drag, lift, and torque closure models using machine learning: Hierarchical approach and interpretability," *Physical Review Fluids*, vol. 8, p. 014303, 2023.

[13] G. K. Batchelor, "The stress system in a suspension of force-free particles," *Journal of Fluid Mechanics*, vol. 41, pp. 545–570, 1970.

[14] E. Guazzelli and J. F. Morris, *A Physical Introduction to Suspension Dynamics*. Cambridge University Press, 2012.

[15] E. Lauga and S. Michelin, "Stresslets induced by active swimmers," *Physical Review Letters*, vol. 117, p. 148001, 2016.

[16] G. Wang, M. Abbas, and E. Climent, "Effect of Reynolds number and concentration on modulation of turbulence by finite size neutrally buoyant particles," in *Proceedings of the 9th International Conference on Multiphase Flow (ICMF 2016)*, Florence, Italy, May 2016, p. 0.

[17] A. Gupta, H. J. H. Clercx, and F. Toschi, "Computational study of radial particle migration and stresslet distributions in particle-laden turbulent pipe flow," *The European Physical Journal E*, vol. 41, p. 34, 2018.

[18] L. W. Rong, Z. Y. Zhou, and A. B. Yu, "Lattice-Boltzmann simulation of fluid flow through packed beds of uniform ellipsoids," *Powder Technology*, vol. 285, pp. 146–156, 2015.

[19] L. He, D. K. Tafti, and K. Nagendra, "Evaluation of drag correlations using particle resolved simulations of spheres and ellipsoids in assembly," *Powder Technology*, vol. 313, pp. 332–343, 2017.

[20] Z. Cao, D. K. Tafti, , and M. Shahnam, "Development of drag correlation for suspensions of ellipsoidal particles," *Powder Technology*, vol. 369, pp. 298–310, 2020.

[21] Z. Cao and D. K. Tafti, "Characterization of lift force and torque in prolate ellipsoid suspensions," *Powder Technology*, vol. 405, p. 117553, 2022.

[22] S. K. P. Sanjeevi and J. Padding, "Hydrodynamic forces on monodisperse assemblies of axisymmetric elongated particles: Orientation and voidage effects," *AIChE Journal*, vol. 66, pp. 1–20, 2020.

[23] LFA Capsule Fillers, "Capsule size guide," https://www.lfacapsulefillers.com/capsule-size-chart, accessed: June 2025.
[24] M. R. Lekkala, M. Latheef, J. H. Jung, A. Coraddu, H. Zhu, N. Srinil, B.-H. Lee, and D. K. Kim, "Recent advances in understanding the flow over bluff bodies with different geometries at moderate Reynolds numbers," *Ocean Engineering*, vol. 261, p. 111611, 2022.
[25] J. G. Leishman, *Introduction to Aerospace Flight Vehicles*. Embry-Riddle Aeronautical University, 2023. [Online]. Available: https://eaglepubs.erau.edu/introductiontoaerospaceflightvehicles/
[26] D. Cheng, S. Wang, and J. A. M. Kuipers, "Modeling study of gas-liquid mass transfer enhancement by cylindrical catalyst particle," *Chemical Engineering Science*, vol. 160, pp. 80–84, 2017.
[27] Y. Shen, W. G. Borghard, and M. S. Tomassone, "Discrete element method simulations and experiments of dry catalyst impregnation for spherical and cylindrical particles in a double cone blender," *Powder Technology*, vol. 318, pp. 23–32, 2017.
[28] J. Jägers, S. Wirtz, and V. Scherer, "Experimental analysis of wood pellet degradation during pneumatic conveying processes," *Powder Technology*, vol. 359, pp. 282–291, 2020.
[29] L. Massaro Sousa, B. Amblard, F. Montjouvet, and S. Tebianian, "Characterization of a pressurized feeder for biomass injection into gas-solid systems," *Chemical Engineering Research and Design*, vol. 175, pp. 171–181, 2021.
[30] H. Rajabnia, O. Orozovic, K. Williams, A. Lavrinec, D. Ilic, M. Jones, and G. Klinzing, "Investigating the relationship between the time constant ratio and plug-flow behaviour in the pneumatic conveyance of biomass material," *Processes*, vol. 11, p. 1697, 2023.
[31] Z. Liu, H. Ma, L. Zhou, Y. Liu, Z. Huang, X. Liao, and Y. Zhao, "DEM-DDM investigation of the tablet coating process using different particle shape models," *Industrial & Engineering Chemistry Research*, vol. 62, pp. 829–840, 2022.
[32] K. Ragaert, L. Delva, and K. van Geem, "Mechanical and chemical recycling of solid plastic waste," *Waste Management*, vol. 69, pp. 24–58, 2017.
[33] N. Fintzi, L. Gamet, and J.-L. Pierson, "Inertial loads on a finite-length cylinder embedded in a steady uniform flow," *Physical Review Fluids*, vol. 8, p. 044302, 2023.
[34] B. Haddadi, C. Jordan, H. R. Norouzi, and M. Harasek, "Investigation of the pressure drop of random packed bed adsorbers," *Chemical Engineering Transactions*, vol. 52, p. , 2016.
[35] V. Pozzobon, J. Colin, and P. Perré, "Hydrodynamics of a packed bed of non-spherical polydisperse particles: A fully virtual approach validated by experiments," *Chemical Engineering Journal*, vol. 354, pp. 126–123, 2018.
[36] V. Sassanis, L. Gamet, M. Rolland, R. Ma, and V. Pozzobon, "Numerical determination of the volumetric heat transfer coefficient in fixed beds of wood chips," *Chemical Engineering Journal*, vol. 417, p. 128009, 2020.
[37] E. Noël and D. Teixeira, "New framework for upscaling gas-solid heat transfer in dense packing," *International Journal of Heat and Mass Transfer*, vol. 189, p. 122745, 2022.
[38] Z. Cao, M. B. Agir, C. White, and K. Kantis, "An open source code for two-phase rarefied flows: rarefiedMultiphaseFoam," *Computer Physics Communications*, vol. 276, p. 108339, 2022.
[39] D. Niblett, M. Mamlouk, and O. E. Godinez-Brizuela, "Porous Microstructure Generator," 2022, 10.25405/data.ncl.20448471.
[40] "Blender," https://www.blender.org/, accessed: July 2024.
[41] G. Boccardo, F. Augier, Y. Haroun, D. Ferre, and D. L. Marchisio, "Validation of a novel open-source work-flow for the simulation of packed-bed reactors," *Chemical Engineering Journal*, vol. 279, pp. 809–820, 2015.
[42] G. Boccardo, E. Crevacore, A. Passalacqua, and M. Icardi, "Computational analysis of transport in three-dimensional heterogeneous materials," *Computing and Visualization in Science*, vol. 23, p. 15, 2020.
[43] D. Gisen, "Generation of a 3D mesh using snappyHexMesh featuring anisotropic refinement and near-wall layers," in *Proceedings of the 11th International Conference on Hydroscience & Engineering*, 2014.
[44] J. Fernengel, F. Habla, and O. Hinrichsen, "Scripting as an approach to automated CFD simulation for packed bed catalytic reactor modeling," *Chemie Ingenieur Technik*, vol. 90, pp. 685–689, 2018.
[45] B. D. Wood, X. He, and S. V. Apte, "Modeling turbulent flows in porous media," *Annual Review of Fluid Mechanics*, vol. 52, pp. 171–203, 202O.
[46] J. Finn and S. V. Apte, "Relative performance of body fitted and fictitious domain simulations of flow through fixed packed beds of spheres," *International Journal of Multiphase Flow*, vol. 56, pp. 54–71, 2013.
[47] B. S. Neo and E. S. G. Shaqfeh, "Stresslet in a dilute suspension of rigid spheres in an Oldroyd-B fluid," *Physical Review Fluids*, vol. 9, p. 033301, 2024.
[48] A. Wachs, "PeliGRIFF, a parallel DEM-DLM/FD direct numerical simulation tool for 3D particulate flows," *Journal of Engineering Mathematics*, vol. 71, pp. 131–155, 2011.
[49] A. Wachs, A. Hammouti, V. G., and M. Rahmani, "Accuracy of finite volume/staggered grid distributed Lagrange multiplier/fictitious domain simulations of particulate flows," *Computers & Fluids*, vol. 115, pp. 154–172, 2015.
[50] S. Das, N. G. Deen, and J. A. M. Kuipers, "Immersed boundary method (IBM) based direct numerical simulation of open-cell solid foams: Hydrodynamics," *AIChE Journal*, vol. 63, pp. 1152–1173, 2016.
[51] S. Elghobashi, "Particle-laden turbulent flows: direct simulation and closure models," *Applied Scientific Research*, vol. 48, pp. 301–314, 1991.
[52] S. V. Patankar, *Numerical heat transfer and fluid flow*. Hemisphere Publishing Corporation, 1981.
[53] H. Jasak, "Error analysis and estimation for the finite volume method with applications to fluid flows," Ph.D. dissertation, Imperial College London, 1996.
[54] F. Moukalled, L. Mangani, and M. Darwish, *The Finite Volume Method in Computational Fluid Dynamics*. Springer Cham, 2016.
[55] A. G. Dixon, M. Nijemeisland, and E. H. Stitt, "Systematic mesh development for 3D CFD simulation of fixed beds: Contact points study," *Computers and Chemical Engineering*, vol. 48, pp. 135–153, 2013.
[56] S. Rebughini, A. Cuoci, and M. Maestri, "Handling contact points in reactive CFD simulations of heterogeneous catalytic fixed bed reactors," *Chemical Engineering Science*, vol. 141, pp. 240–249, 2016.

[57] A. Ang, "Generating uniform unit random vectors in R^n," https://angms.science/doc/RM/randUnitVec.pdf, accessed: July 2024.
[58] D. Eberly, "Intersection of a cylinder and a plane," https://www.geometrictools.com/Documentation/IntersectionCylinderPlane.pdf, 2007, accessed: January 2024.
[59] P. Niegodajew, A. P. Durajski, P. Rajca, K. M. Gruszka, and M. Marek, "Experimental and numerical study on the orientation distribution of cylindrical particles in random packed beds," *Chemical Engineering Journal*, vol. 432, p. 134043, 2022.
[60] D. Eberly, "Intersection of cylinders," https://www.geometrictools.com/Documentation/IntersectionOfCylinders.pdf, 2000, accessed: January 2024.
[61] C. Vega and S. Lago, "A fast algorithm to evaluate the shortest distance between rods," *Computers & Chemistry*, vol. 18, pp. 55–59, 1994.
[62] L. Pournin, M. Weber, M. Tsukahara, J. A. Ferrez, M. Ramaioli, and T. M. Liebling, "Three-dimensional distinct element simulation of spherocylinder crystallization," *Granular Matter*, vol. 7, pp. 119–126, 2005.
[63] W. S. Jodrey and E. M. Tory, "Computer simulation of close random packing of equal spheres," *Physical Review A*, vol. 32, pp. 2347–2351, 1985.
[64] J. Zhao, S. Li, R. Zou, and A. Yu, "Dense random packings of spherocylinders," *Soft Matter*, vol. 8, p. 1003, 2012.
[65] D. C. Lay, S. R. Lay, and J. J. McDonald, *Linear Algebra and Its Applications*. Pearson, 2016.
[66] M. C. L. Belon, "Java: Intersection of two ellipse segments transformed into 3D space," https://stackoverflow.com/questions/54552409/java-intersection-of-two-ellipse-segments-transformed-into-3d-space, 2019, accessed: July 2024.
[67] Y.-B. Jia, "Plücker coordinates for lines in the space," https://faculty.sites.iastate.edu/jia/files/inline-files/plucker-coordinates.pdf, 2022, accessed: July 2024.
[68] P. W. Collingridge, "Finding the angle around an ellipse," https://www.petercollingridge.co.uk/tutorials/computational-geometry/finding-angle-around-ellipse/, accessed: July 2024.
[69] S. Harikrishnan, "Implementing streamwise periodic boundary condition in OpenFOAM," Indian Institute of Technology Madras, Tech. Rep., 2019.
[70] M. Coe and D. Holland, "A cyclic heat transfer solver for OpenFOAM," *OpenFOAM Journal*, vol. 3, pp. 225–251, 2023.
[71] S. V. Patankar, C. H. Liu, and E. M. Sparrow, "Fully developed flow and heat transfer in ducts having streamwise-periodic variations of cross-sectional area," *Journal of Heat Transfer*, vol. 9, pp. 180–186, 1977.
[72] S. Ergun, "Fluid flow through packed columns," *Chemical Engineering Progress*, vol. 48, pp. 89–94, 1952.
[73] "OpenFOAM: User Guide v2112," https://www.openfoam.com/documentation/guides/latest/doc/guide-fos-forces-forces.html, accessed: November 2023.
[74] Y. Tang, E. A. J. F. Peters, J. A. M. Kuipers, S. H. L. Kriebitzsch, and M. A. van der Hoef, "A new drag correlation from fully resolved simulations of flow past monodisperse static arrays of spheres," *Chemical Engineering Journal*, vol. 61, pp. 688–698, 2014.
[75] A. Einstein, "Eine neue Bestimmung der Moleküldimensionen," *Annalen der Physik*, vol. 19, pp. 289–306, 1906.
[76] A. A. Zick and G. M. Homsy, "Stokes flow through periodic array of spheres," *Journal of Fluid Mechanics*, vol. 115, pp. 13–26, 1982.
[77] S. K. P. Sanjeevi, J. A. M. Kuipers, and J. Padding, "Drag, lift and torque correlations for non-spherical particles from Stokes limit to high Reynolds numbers," *International Journal of Multiphase Flow*, vol. 106, pp. 325–337, 2018.
[78] A. S. Sangani, A. Acrivos, and P. Peyla, "Roles of particle-wall and particle-particle interactions in highly confined suspensions of spherical particles being sheared at low Reynolds numbers," *Physics of Fluids*, vol. 23, p. 083302, 2011.
[79] K. K. Rao and P. R. Nott, *An Introduction to Granular Flow*. Cambridge University Press, 2008.
[80] J. Talbot, D. Kivelson, M. P. Allen, G. T. Evans, and D. Frenkel, "Structure of the hard ellipsoid fluid," *The Journal of Chemical Physics*, vol. 92, pp. 3048–3057, 1990.
[81] R. I. Issa, A. D. Gosman, and A. P. Watkins, "The computation of compressible and incompressible recirculating flows by a non-iterative implicit scheme," *Journal of Computational Physics*, vol. 62, pp. 66–82, 1991.
[82] C. J. Greenshields and H. Weller, *Notes on Computational Fluid Dynamics: General Principles*. CFD Direct Limited, 2022.