User Guide

Xiaoyu Liu, Mostafa Sulaiman, Jari Kolehmainen, Ali Ozel and Sankaran Sundaresan

August, 2019, updated in Feb 2022

Contents

1 Outline **2**

2 Tutorial cases **3**
2.1 Case 1 : Particle-wall collision 4
2.2 Case 2 : Particle-particle collision 8
2.3 Case 3 : Particle-carrier cohesion test 12
2.4 Caes 4 : Particle settling test 15
2.5 Case 5: Carrier-API agglomeration 20
2.6 Case 6: Deagglomeration induced by collision with a wall 23

3 Inhaler Simulation **26**
3.1 Geometry Description 26
3.2 Stage 1: Sedimentation 27
3.3 Stage 2: Inhalation 31
3.4 Data post-processing 40

Abstract

This document is meant to provide detailed instructions for installing and using the model, illustrated with tutorial examples.

1 Outline

Dry powder inhaler (DPI) is an important drug delivery vehicle for treating respiratory conditions. In order for Active Pharmaceutical Ingredients (API) to be delivered to the lung, bigger carrier particles are utilized to first form agglomerate with API particles to facilitate fluidization and then deagglomerate to release API. This code aims to provide a simulation tool that captures the underlying physics behind agglomeration and deagglomeration, which are essential to DPI drug delivery.

The DPI simulations considers both carrier and API particles, so there are two types of particles involved. In a typical DPI application, the total number of particles can be several tens of millions. For example, a full simulation of the case in van Wachem et al.[6][5] would require tracking 26 million particles. Such systems require impractically large computational resources.Thus, we adopt particle-based coarse-grained simulation approach, henceforth referred to as the representative particle approach, that permits to simulate the same system with affordable computational resources. Here a "representative particle" refers to a fictitious particle which replaces many primary particles. When every representative particle contains just one primary particle, one recovers the Computational Fluid Dynamics-Discrete Element Method (CFD-DEM) where every particle is tracked individually.

In the representative particle approach used in this study, every carrier particle is simulated, while only a subset of the API particles, referred to as representative API particles are simulated. Each representative API particles represents N primary API particles. N is also referred to as the "coarsening factor". For example, a coarsening factor $N = 5$ means that each representative API particle represents five primary API particles and that the number of representative API particles being tracked in the simulation is one-fifth of that in the full CFD-DEM simulation. Therefore, such a particle-based coarsening approach can reduce the computational cost required to simulate a realistically sized DPI application. Please refer to Liu et al. 2021 [4] for detailed discussion of this approach.

2 Tutorial cases

The user guide consists of six tutorial cases that aim at explaining the usage of the implemented models in the code before setting up the actual inhaler simulation. For each case we begin with case description followed by case setup with input files, the way to run the case and the expected outcome of the cases. Cases 1 & 2 include contact models, case 3 cohesion model, and case 4 drag model. In each test, we compare the results between a primary API particle (i.e. N = 1) and an representative API particle with different coarsening factors and check if the representative particle approach is giving results similar to those given by simulations using primary particles.

Cases 1-3 and Cases 5-6 use only DEM on LIGGGHTS® platform. To understand standard commands of LIGGGHTS®, readers are referred to the official documentations of LIGGGHTS® (https://www.cfdem.com/media/DEM/doc/Manual.html) and LAMMPS® (https://lammps.sandia.gov/doc/Manual.html). In addition, features/commands that are developed specifically for this code will be highlighted and explained in each test case. Physical properties for these tests are listed in the following table. Because of the small size of API particles, CGS unit system is recommended instead of SI unit system.

	SI Unit		CGS Unit	
	Carrier	API	Carrier	API
Diameter	70×10^{-6} m	5×10^{-6} m	70×10^{-4} cm	5×10^{-4} cm
Density	1520 kg.m^{-3}	1520 kg.m^{-3}	1.52 g.cm^{-3}	1.52 g.cm^{-3}
Young's modulus	5×10^{8} kg.m^{-1}s^{-2}	5×10^{8} kg. m^{-1}s^{-2}	5×10^{9} g.cm^{-1}s^{-2}	5×10^{9} g.cm^{-1}s^{-2}
Poisson's Ratio	0.35	0.35	0.35	0.35

Table 1: Physial Properties for particles

	Carrier-Carrier	Carrier-API	API-API	Carrier-wall	API-wall
Restitution coefficient	0.85	0.85	0.85	0.85	0.85
Friction coefficient	0.45	0.45	0.45	0.45	0.45

Table 2: Physical properties for contact models

Case 4 uses CFD-DEM, where CFD is based on OpenFOAM® platform and the coupling is based on CFDEM-coupling® platform. Readers are referred to the standard documentation for OpenFOAM® (https://www.openfoam.com/documentation/) and CFDEM-coupling® (https://www.cfdem.com/media/CFDEM/docu/CFDEMcoupling_Manual.html). In addition, features/commands specifically developed for this code will be highlighted in the case.

2.1 Case 1 : Particle-wall collision

Case setup

We start with a simple case where a particle collides with a wall. We plot the force for the particle as well as the representative particle. Only the soft-sphere spring-dashpot force generated during contact between the two particles is included in this case, and other forces such as van der Waals force and electrostatic force are not considered. If the collision between the particle/representative particle and the wall appears to be poorly resolved, a smaller time step should be considered.

Simulation details are specified in an input file "in.liggghts_init", shown below. The geometry shown in Figure (1) is generated in the code by importing the "wall1.stl" file in Line 39, which is found in *../mesh*.

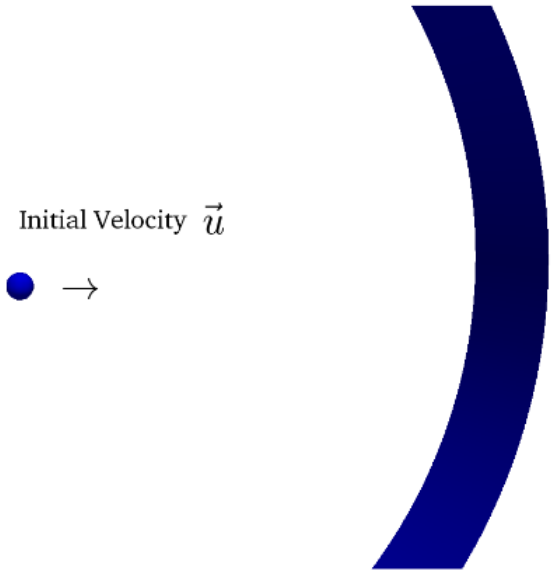


Figure 1: Geometry of the particle-wall collision

Line 40 generates the mesh and specifies the force model for particle-wall collision. Lines 68-71 define the time, y-coordinate, y-component velocity and y-component force of the particle and Line 73 outputs them into the file *output.dat*. We specify the units to be *cgs* in Line 14. In Line 43, we create a particle through *create_atom* of particle type 1 at position $(0.3, 0.3497, 0.0)$.

```
1 create_atoms    1 single 0.3 0.3497 0.0 units box
```

with diameter and density set in Line 44. Line 46 group the particle as "atom1" and Line 47 sets the initial velocity of the group "atom1" to be (0,300,0).

The force models for particle-particle collisions are specified in Line 25. The time step size for the simulation is given in Line 36. In standard LIGGGHTS®, Hertzian model for normal contact forces and tangential contact forces are specified by declaring *model hertz* and *tangential history* respectively. For more information on command "pair_style", please refer to https://www.cfdem.com/media/DEM/docu/pair_style.html. The parameters for Herztian model are specified in Lines 29-32. Since there is only one particle type (Type 1), the total number of particle types should be specified as *1* in Line 19 after *create_box*. Only one value is given for *youngsModulus* in Line 29 and *poissonsRatio* in Line 30, as both are per atom type (*peratomtype*) properties. In Lines 31-33, *coefficientRestitution*, *coefficientFriction* are per atom type pair (*peraomtypepair*), so we first specify the number of atom pairs (which is 1 in our case as we only consider particle collision with a wall), followed by the value for each atom pair.

In addition to the standard pair_styles in LIGGGHTS® standard library, we implemented the coarse-grained version for these two forces (*model hertz_parcel* and *tangential history_parcel*). If coarse-grained versions are

used, the coarsening factor N needs to be specified in Line 54. Since there is only one particle type (Type 1) and *nparcel* is a per atom type property (*peratomtype*), we only specify one coarsening factor (in the example it is specified as 100). Readers are referred to Section 4 of Theory Guide for the force model of the coarse-grained approach.

```
1  # in.liggghts_init file
2  echo both
3
4  #DEFINE VARIABLES
5  variable cutOff equal 1.e-2
6
7  atom_style  granular
8  atom_modify map array sort 0 0
9  comm_modify mode multi vel yes
10
11 boundary  f f f
12 newton    off
13
14 units   cgs
15 processors  * * *
16
17 region    reg block 0 5 -0.4 0.4 -0.4 1.4 units box
18
19 create_box  1 reg
20
21 neighbor  ${cutOff} bin
22 neigh_modify  delay 0
23
24 #pair style
25 pair_style  gran model hertz_parcel tangential history_parcel #Hertzian model
26 pair_coeff  * *
27
28 #Material properties required for new pair styles
29 fix m1 all property/global youngsModulus peratomtype 5.e9
30 fix m2 all property/global poissonsRatio peratomtype 0.35
31 fix m3 all property/global coefficientRestitution peratomtypepair 1 0.85
32 fix m4 all property/global coefficientFriction peratomtypepair 1 0.45
33 #fix m5 all property/global coefficientRollingFriction peratomtypepair 1 0.05
34
35 #timestep
36 timestep          1.e-8
37
38 #walls (liggghts 2.0)
39 fix wall1 all mesh/surface/stress file ../meshes/wall1.stl type 1
40 fix granwalls all wall/gran model hertz_parcel tangential history_parcel mesh n_meshes 1
       meshes wall1
41
42 ##particle insertion##
43 create_atoms    1 single 0.3 0.3497 0.0 units box
44 set             type 1 diameter 5e-4 density 1.52
45
46 group   atom1 id 1
47 velocity  atom1 set 0.0 300 0.0 units box
48
49 group   nve_group region reg
50
51 #apply nve integration to all particles that are inserted as single particles
52 fix   integr all nve/sphere/parcel
53
54 fix   groupPar all property/global nparcel peratomtype 1
55 run     0
56
57 #screen output
58 compute   1 all erotate/sphere
59 thermo_style  custom step atoms ke c_1 vol
60 thermo    1000
61 thermo_modify lost ignore norm no
62 compute_modify  thermo_temp dynamic yes
```

```
63
64 #insert the first particles so that dump is not empty
65 run   1
66 dump    dmp all custom 1 post/dump.part id type x y z ix iy iz vx vy vz fx fy fz radius
67
68 variable    time equal step*dt
69 variable    y1 equal y[1]
70 variable    vy1 equal vy[1]
71 variable    fy1 equal fy[1]
72
73 fix     extra all print 1 "${time} ${y1} ${vy1} ${fy1}" file output.dat screen no
74
75 run   30
```

Case run

In order to run the current case, the user needs to decompress the provided tutorial case and then copy the LIGGGHTS executable that runs the simulation from its source file location ($CFDEM_LIGGGHTS_SRC_DIR) by typing the following commands in a terminal:

```
1 tar -xvf Case1ParcelWallCollision.tar
2 cd Case1ParcelWallCollision
3 cp $CFDEM_LIGGGHTS_SRC_DIR/lmp_fedora_fpic .
```

The user then enters the following command in the same terminal to run the case:

```
1 mpirun -np 1 lmp_fedora_fpic < in.liggghts_init
```

The output results are saved in a file whose format and name are specified in line 73 of the "in.liggghts_init" file. The output file name in this case is "output.dat" and the format is specified as shown: ($time) for time ($y1) for y position, ($vy1) for the y component velocity and ($fy1) for the y component of the force.

The temporal evolution of y-coordinate Y is plotted in Figure(2) and y-component of force experienced by the particle is plotted in Figure(3) for coarsening factors N = 1 and 100. Notice that the particle trajectories of these two cases are the same, while the force experienced by the representative particle with N = 100 is 100 times that experienced by the particle with N = 1. For more details about outputs the reader is referred to LIGGGHTS® public documentation: https://usermanual.wiki/Pdf/Manual.1338160442.pdf

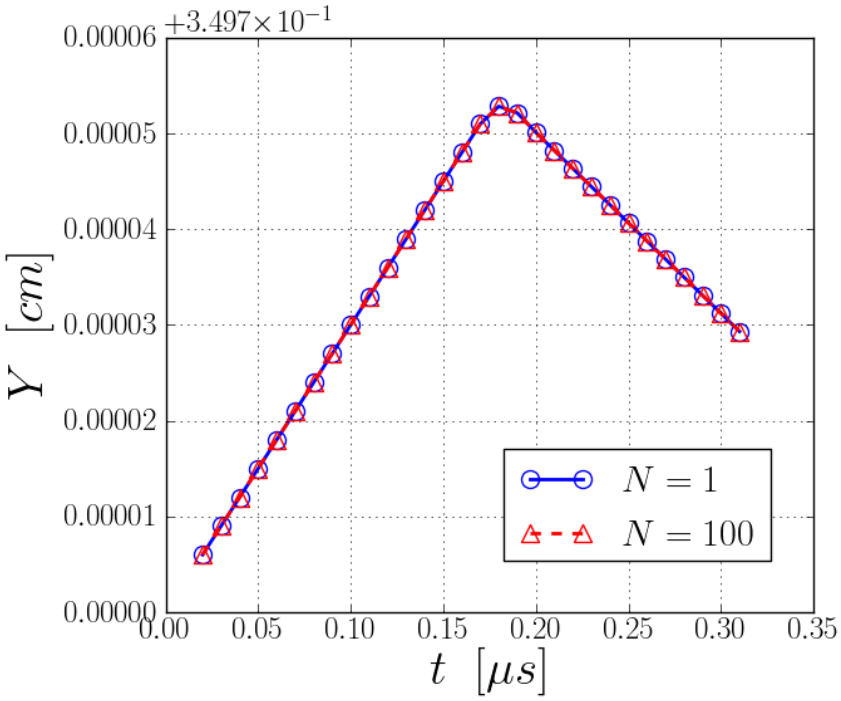
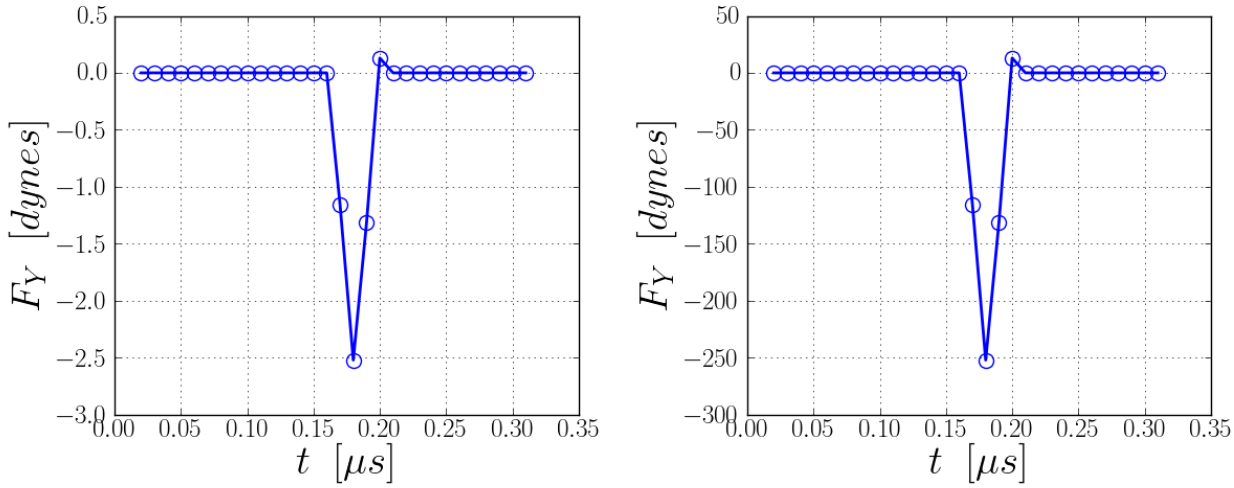


Figure 2: Temporal evolution of y-coordinate for coarsening factors N = 1 and N = 100



(a) (b)

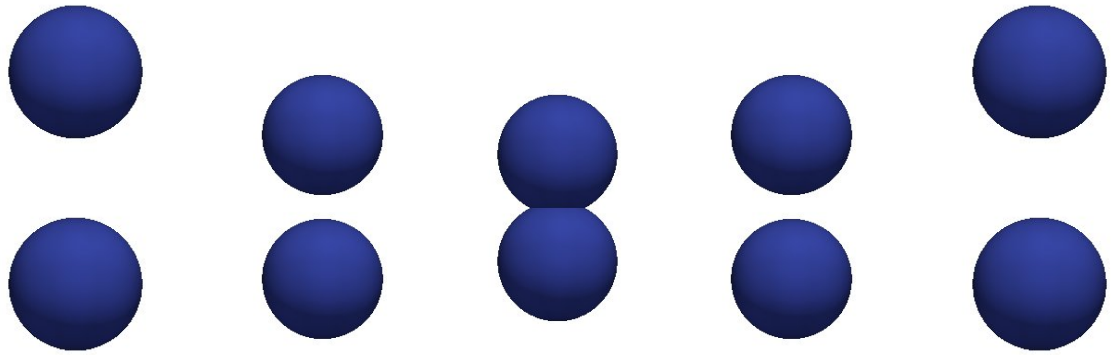
Figure 3: Temporal evolution of y-component force for coarsening factors (a) N = 1 and (b) N = 100

2.2 Case 2 : Particle-particle collision

Case setup

This case shows another example about contact forces where collision takes place between two particles. Only the soft-sphere spring-dashpot force generated during contact between the two particles is included in this case, and other forces such as van der Waals force and electrostatic force are not considered. This configuration aids in deciding an appropriate time step for simulation. If the collision test between two particles appears to be poorly resolved, a smaller time step should be considered.

Similar to Case 1, Line 39 specifies the time step to be $1 \times 10^{-8} s$. Line 42 creates a particle at position (0.05, 0.05, 0.05) with velocity (0, 0, 100) specified in Line 47. Line 43 creates a particle at position (0.05, 0.05, 0.051) with velocity (0, 0, -100) specified in Line 50. The reader can refer to the input file "in.liggghts_init" for all details. The z-component of particle coordinates, velocities and forces are defined in Lines 72-77. Line 80 outputs these data into the file *output.dat*.



(a) $t = 0$ (b) $t = 200dt$ (c) $t = 300dt$ (d) $t = 400dt$ (e) $t = 600dt$

Figure 4: Illustration of the particle-particle collision case, dt $= 1 \times 10^{-8} s$

```
1  # in.liggght_init file
2  echo both
3
4  #DEFINE VARIABLES
5  variable cutOff equal 1.e-2
6
7
8
9  atom_style  granular
10 atom_modify map array sort 0 0
11 comm_modify mode multi vel yes
12
13 boundary  f f f
14 newton    off
15
16 units   si
17 processors  * * *
18
19
20 region    reg block 0 0.1 0 0.1 0 0.1 units box
```

```
21
22 create_box  1 reg
23
24 neighbor  ${cutOff} bin
25 neigh_modify  delay 0
26
27 #pair style
28 pair_style  gran model hertz_parcel tangential history_parcel #Hertzian model
29 pair_coeff  * *
30
31 #Material properties required for new pair styles
32 fix m1 all property/global youngsModulus peratomtype 5.e9
33 fix m2 all property/global poissonsRatio peratomtype 0.35
34 fix m3 all property/global coefficientRestitution peratomtypepair 1 0.85
35 fix m4 all property/global coefficientFriction peratomtypepair 1 0.45
36 fix m5 all property/global coefficientRollingFriction peratomtypepair 1 0.1
37
38 #timestep, gravity
39 timestep          1.e-8
40
41 ##particle insertion##
42 create_atoms    1 single 0.05 0.05 0.05 units box
43 create_atoms    1 single 0.05 0.05 0.051 units box
44 set             group all type 1 diameter 5e-4 density 1.52
45
46 group   atom1 id 1
47 velocity  atom1 set 0.0 0.0 100 units box
48
49 group   atom2 id 2
50 velocity  atom2 set 0.0 0.0 -100 units box
51
52 group   nve_group region reg
53
54 #apply nve integration to all particles that are inserted as single particles
55 fix   integr all nve/sphere/parcel
56
57 fix   groupPar all property/global nparcel peratomtype 100
58 run     0
59
60 #screen output
61 compute   1 all erotate/sphere
62 thermo_style  custom step atoms ke c_1 vol
63 thermo    1000
64 thermo_modify lost ignore norm no
65 compute_modify  thermo_temp dynamic yes
66
67 #insert the first particles so that dump is not empty
68 run   1
69 dump    dmp all custom 1 post/dump*.part id type x y z ix iy iz vx vy vz fx fy fz radius
70
71 variable    time equal step*dt
72 variable    z1 equal z[1]
73 variable    z2 equal z[2]
74 variable  vz1 equal vz[1]
75 variable  vz2 equal vz[2]
76 variable  fz1 equal fz[1]
77 variable  fz2 equal fz[2]
78 variable  overlap equal (0.0005-(z[2]-z[1]))/0.0005
79
80 fix     extra all print 1 "${time} ${overlap} ${z1} ${z2}" file output.dat screen no
81
82 run   1000
```

Case run

In order to run the current case, the user needs to decompress the provided test case and then copy the LIGGGHTS executable from the source file location ($CFDEM_LIGGGHTS_SRC_DIR) by typing the fol-

lowing commands in a terminal:

```
tar -xvf Case2ParcelParcelCollision.tar
cd Test2ParcelParcelCollision
cp $CFDEM_LIGGGHTS_SRC_DIR/lmp_fedora_fpic .
```

The user then enters the following command in the same terminal to run the case:

```
mpirun -np 1 lmp_fedora_fpic < in.liggghts_init
```

The temporal evolution of z-coordinate of both particles are plotted for coarsening factor N = 1 and 100 in Figure(5). Notice that the trajectories in both cases are the same. The temporal evolution of the overlapping distance given by Eq(1) is plotted in Figure(6).

$$\delta = \frac{d - (z_1 - z_2)}{d} \tag{1}$$

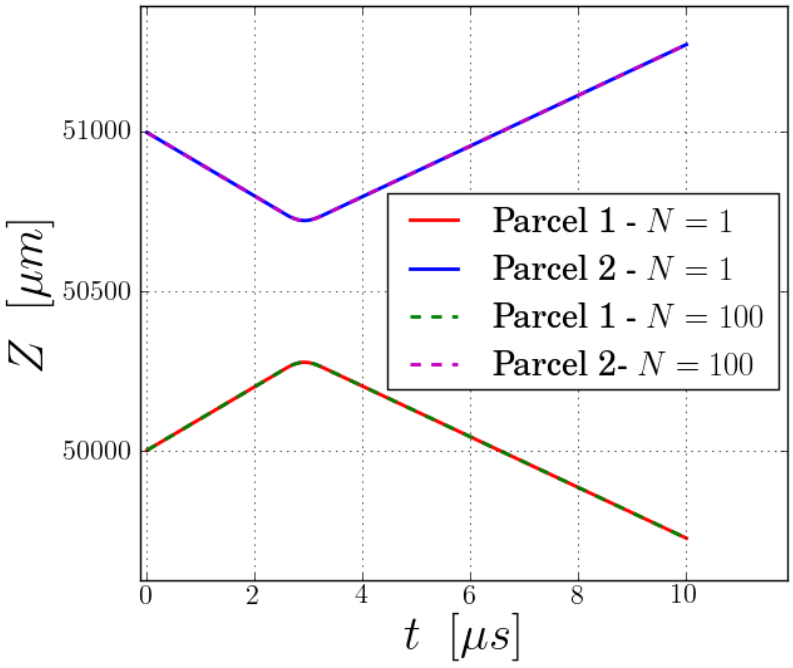


Figure 5: Temporal evolution of particle positions for coarsening factors N = 1 and N = 100

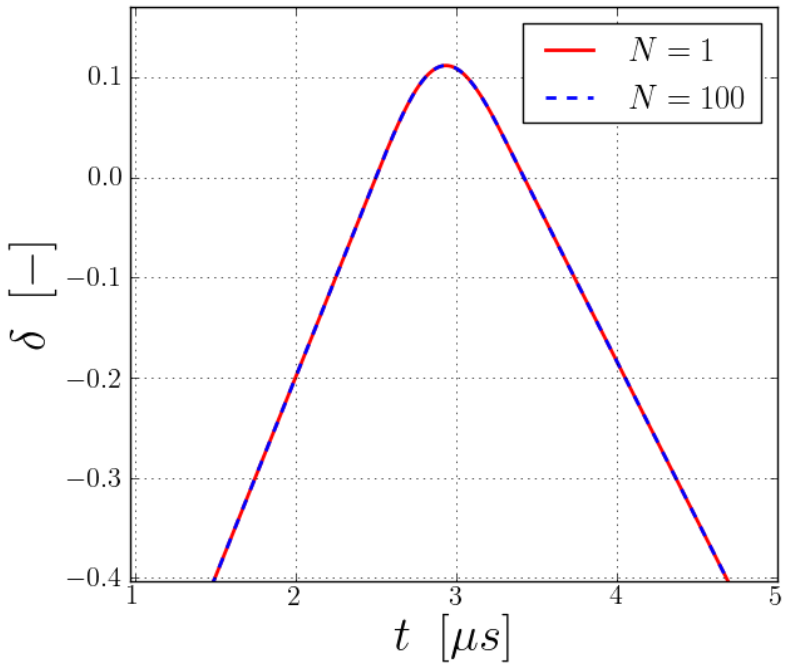


Figure 6: Temporal evolution of the overlapping distance for coarsening factors N = 1 and N = 100

2.3 Case 3 : Particle-carrier cohesion test

Case setup

This case shows an example about employing van der Waals' cohesion model as well as its coarse-grained version, which are essential for the agglomeration/deagglomeration process in DPI simulations.

This test uses *cgs* unit system. Line 49 creates a carrier particle (ID=1, type=1) initialized at position (0.05, 0.05, 0.05) with velocity (0, 0, 0). Line 50 creates an API particle (ID=2,type= 2) initialized at position (0.05, 0.05, 0.05375) with velocity (0, 0, 0) so that these two particles just touch each other. Notice in Line 27, in addition to contact forces, cohesion force model is also included as *cohesion vdw/gu/parcel*. This is the coarse-grained version for the VDW model developed by Gu et al., 2016 [3].

For CFD-DEM simulations, particles are often softened for computational expediency (lower the Young's modulus, bigger the simulation time step). To ensure that softening does not alter the simulation results, one has to modify the standard van der Waals'(VDW) force model to account for the lowered Young's modulus. Our LIGGGHTS version has an implementation of VDW forces with an option for the stiffness correction (*stiffnessScaling*) in Line 11, which can be computed as

$$stiffnessScaling = \frac{1}{(\frac{Y_{sim}}{Y_{real}})^{0.4}} \tag{2}$$

where Y_{sim} is the user-specified Young's modulus in the simulation, Y_{real} the actual Young's modulus of the particle. Lines 30-34 specify the model parameter for the VDW model. The definition of each parameter can be found in the van der Waals' force part of the theory guide. Since these are per atom type pair (*peratomtypepair*) properties, we first specify the number of atom pairs (2 in this case), followed by values for each pair in the order of Type1-Type1, Type1-Type2, Type2-Type1, Type2-Type2.
The overlap between the carrier particle and the API particle is computed in Line 82, and Line 84 writes the overlap into a data file "output.dat". Please refer to the following input file "in.liggghts_init" for detail.

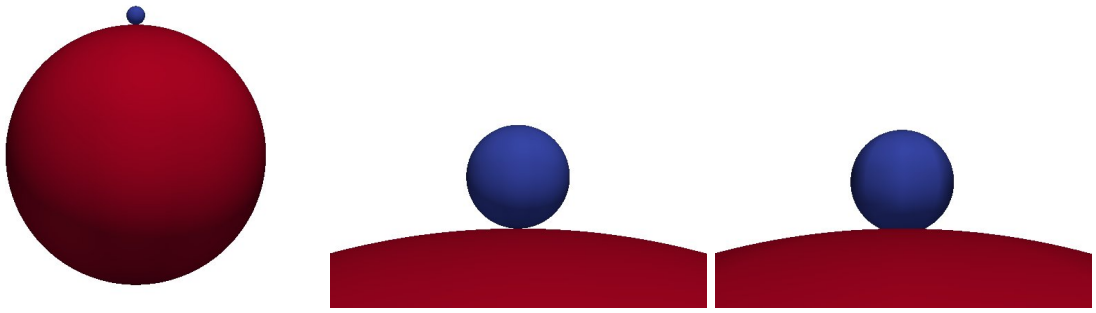


Figure 7: Illustration of the overlap due to cohesion between a carrier particle and an API particle

```
1  # in.liggght_init file
2  echo both
3
4  #DEFINE VARIABLES
5  variable cutOff equal 1.e-2
6
7  atom_style  granular
8  atom_modify map array sort 0 0
9  comm_modify mode multi vel yes
10
11 stiffnessScaling 3.177672
12
13 boundary  f f f
14 newton    off
15
16 units   cgs
```

```
processors  * * *

region    reg block 0 0.1 0 0.1 0 0.1 units box

create_box  2 reg

neighbor  ${cutOff} bin
neigh_modify  delay 0

#pair style
pair_style  gran model hertz_parcel tangential history_parcel cohesion vdw/gu/parcel #
    Hertzian model
pair_coeff  * *

#Material properties required for VDW model
fix     vdw1 all property/global cohesionEnergyDensity peratomtypepair 2 5.0e-12 5.0e-12 5.0
    e-12 5.0e-12 # erg = 1e-7 J
fix         vdw2 all property/global sMin peratomtypepair 2 1.0e-7 1.0e-7 1.0e-7 1.0e-7
fix             vdw3 all property/global sMins peratomtypepair 2 6.276e-7 6.227e-7 6.227e-7
    6.077e-7
fix             vdw4 all property/global sO peratomtypepair 2 4.494e-7 4.445e-7 4.445e-7
    4.295e-7


#Material properties required for new pair styles
fix         m1 all property/global youngsModulus peratomtype 5.e9 5.e9
fix         m2 all property/global poissonsRatio peratomtype 0.35 0.35
fix         m3 all property/global coefficientRestitution peratomtypepair 2 0.85 0.85 0.85
    0.85
fix         m4 all property/global coefficientFriction peratomtypepair 2 0.45 0.45 0.45 0.45
fix                             m5 all property/global coefficientRollingFriction
    peratomtypepair 2 0.1 0.1 0.1 0.1

#timestep, gravity
timestep          1.e-8
#fix       gravi all gravity 0.0 vector 0.0 0.0 -1.0

##particle insertion##
create_atoms    1 single 0.05 0.05 0.05 units box
create_atoms    2 single 0.05 0.05 0.05375 units box
set             type 1 diameter 70e-4 density 1.52
set             type 2 diameter 5e-4 density 1.52

group   nve_group region reg

#apply nve integration to all particles that are inserted as single particles
fix   integr all nve/sphere/parcel

#set coarsening factor
fix   groupPar all property/global nparcel peratomtype 1 COARSEN_FACTOR
run     0

#screen output
compute   1 all erotate/sphere
thermo_style  custom step atoms ke c_1 vol
thermo    1000
thermo_modify lost ignore norm no
compute_modify  thermo_temp dynamic yes

#insert the first particles so that dump is not empty
run   1
dump    dmp all custom 1 post/dump.part id type x y z ix iy iz vx vy vz fx fy fz radius


variable        time equal step*dt
variable        z1 equal z[1]
variable        z2 equal z[2]
variable  vz1 equal vz[1]
```

```
variable   vz2 equal vz[2]
variable   fz1 equal fz[1]
variable   fz2 equal fz[2]
variable   overlap equal (0.00375-(z[2]-z[1]))/0.00025

fix      extra all print 1 "${time} ${overlap} ${z1} ${z2}" file output.dat screen no

run    10000
```

Case run

In order to run the current case, the user needs to decompress the provided case and then copy the LIGGGHTS executable from the source file location ($CFDEM_LIGGGHTS_SRC_DIR) by typing the following commands in a terminal:

```
tar -xvf Case3CarrierAPIcohesion.tar
cd Case3CarrierAPIcohesion
cp $CFDEM_LIGGGHTS_SRC_DIR/lmp_fedora_fpic .
```

The user then enters the following command in the same terminal to run the case:

```
mpirun -np 1 lmp_fedora_fpic < in.liggghts_init
```

The temporal evolution of overlap are computed for different coarsening factors N = 1, 10 and 100, which are essentially the same.

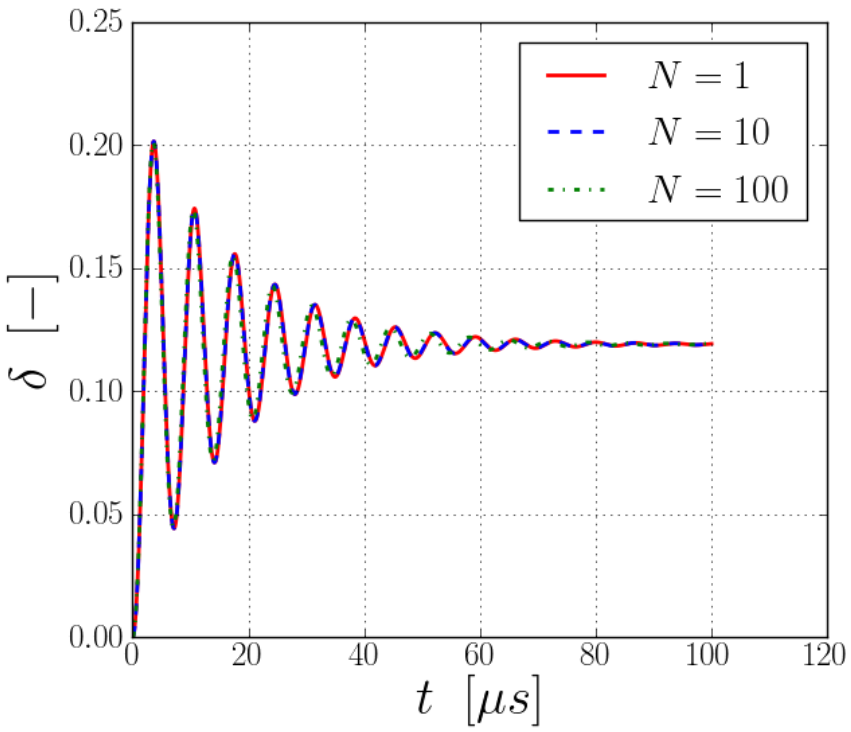


Figure 8: Temporal evolution of z-coordinate for coarsening factors (a) N = 1 and (b) N = 10 (c) N = 100

2.4 Caes 4 : Particle settling test

Case setup

In this case we employ the drag model as well as its coarse-grained version where we perform CFDEM-coupling® (LIGGGHTS® + OpenFOAM®). The case dimensions are in SI unit as specified in Line 14. Line 43 creates an API particle at position $(0.01, 0.01, 0.0012)$, with initial velocity (0,0,0). Due to gravity specified in Line 37, the particle will drop to the bottom wall created in Line 40. For more information on how to create primitive walls in LIGGGHTS®, please refer to https://www.cfdem.com/media/DEM/docu/fix_wall_gran.html.

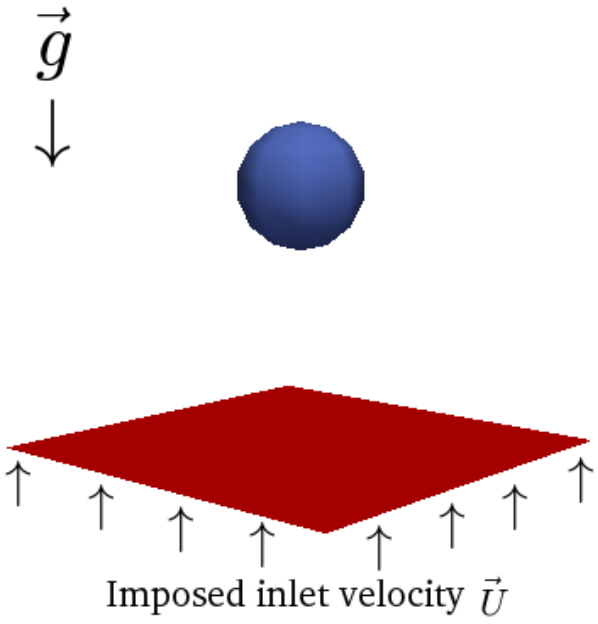


Figure 9: Illustration of particle settling on a wall

The coarsening factor is specified in Line 57 and contact forces model between particles in Line 26, contact forces between particle and wall in Line 40, gravity in Line 37. Line 54 specifies the coarsening factor for integrating the momentum equation, so that the mass of a representative particle is equal to N primary particles. Notice in this case, Lines 49-51 specify the CFD-DEM coupling in the code. In simulation setup, CFD-DEM and CFD-DPM coupling are handled in identical fashion. We refer the reader for the input files "in.liggghts_init" and "couplingProperties". Lines 71-73 specify the z-component of particle coordinate, velocity and force. Line 74 saves them into the file "output.dat"

```
1  # in.liggghts_init file
2  echo both
3
4  #DEFINE VARIABLES
5  variable cutOff equal 1.e-2
6
7  atom_style  granular
8  atom_modify map array sort 0 0
9  comm_modify mode multi vel yes
10
11 boundary  f f f
12 newton    off
13
```

```
units    si
processors  * * *


region     reg block 0 0.0255 0 0.0255 0 0.0255 units box

create_box  1 reg

neighbor  ${cutOff} bin
neigh_modify  delay 0

#pair style
pair_style  gran model hertz_parcel tangential history_parcel #Hertzian model
pair_coeff  * *

#Material properties required for new pair styles
fix        m1 all property/global youngsModulus peratomtype 1.e7
fix        m2 all property/global poissonsRatio peratomtype 0.22
fix        m3 all property/global coefficientRestitution peratomtypepair 1 0.5
fix        m4 all property/global coefficientFriction peratomtypepair 1 0.5

#timestep, gravity
timestep          1.e-5
fix      gravi all gravity/parcel 9.81 vector 0.0 0.0 -1.0

#walls (liggghts 2.0)
fix        bottomwall  all wall/gran model hertz_parcel tangential history_parcel primitive
    type 1 zplane 0.0

##particle insertion##
create_atoms     1 single 0.01 0.01 0.0012 units box
set              group all diameter 2.e-3 density 2400

group   nve_group region reg

#cfd coupling
fix              fcpus all cpus debug no polyhedron yes compress yes
fix              cfd all couple/cfd couple_every 1 mpipun
fix          cfd2 all couple/cfd/force

#apply nve integration to all particles that are inserted as single particles
fix   integr all nve/sphere/parcel

#set coarsen factor for the parcel
fix   groupPar all property/global nparcel peratomtype 1
run      0

#screen output
compute   1 all erotate/sphere
thermo_style  custom step atoms ke c_1 vol
thermo    1000
thermo_modify lost ignore norm no
compute_modify  thermo_temp dynamic yes

#insert the first particles so that dump is not empty
dump    dmp all custom 100 ../DEM/post/dump*.part id type x y z ix iy iz vx vy vz fx fy fz
    radius

variable        time equal step*dt
variable        z equal z[1]
variable  vz equal vz[1]
variable  fz equal fz[1]
fix      extra all print 1 "${time} ${z} ${vz} ${fz}" file ../output.dat screen no
```

Please refer to section 4 of theory guide on how drag force *WenYuRepresentativeDrag* is modeled for a representative particle. For drag force that couples CFD and DEM, the coarsening factor can be specified in the file couplingProperties Line 99 found in CFD/constant/. The following is the file couplingProperties.

```
/*--------------------------------------------------------------------------*\
```

```
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  1.4                                   |
|   \\  /    A nd           | Web:      http://www.openfoam.org               |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/


FoamFile
{
    version          2.0;
    format           ascii;

    root             "";
    case             "";
    instance         "";
    local            "";

    class            dictionary;
    object           couplingProperties;
}

// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

//===========================================================================//
// sub-models & settings

modelType A; // A or B

couplingInterval 1; //1000;

voidFractionModel divided;//centre;//bigParticle;//

locateModel engine;//standard;

meshMotionModel noMeshMotion;

regionModel allRegion;

IOModel basicIO; //trackIO; //

dataExchangeModel twoWayMPIpu;//twoWayMPI;//twoWayFiles;//oneWayVTK;//

averagingModel dense;//dilute;//

clockModel standardClock;//off; //

smoothingModel off;

probeModel off;

insulator on;

forceModels
(
    WenYuRepresentativeDrag
);

momCoupleModels
(
    implicitCouple
    explicitCoupleSource
);

turbulenceModelType RASProperties;//LESProperties;

//===========================================================================//
// sub-model properties
```

```
implicitCoupleProps
{
    velFieldName "U";
    granVelFieldName "Us";
    voidfractionFieldName "voidfraction";
}
explicitCoupleProps
{
}

WenYuDragProps
{
    velFieldName "U";
    granVelFieldName "Us";
    densityFieldName "rho";
    voidfractionFieldName "voidfraction";
    interpolation;
    verbose;
}

WenYuRepresentativeDragProps
{
    velFieldName "U";
    granVelFieldName "Us";
    densityFieldName "rho";
    voidfractionFieldName "voidfraction";
    interpolation;
    verbose;
    npart 1;
    dpart 2.e-3;
    rhopart 2400;
}

twoWayMPIpuProps
{
//    liggghtsPath "../DEM/in.liggghts_restart";
    liggghtsPath "../DEM/in.liggghts_init";
//    liggghtsPath "../DEM/in.liggghts_resume";
    cpusPath "../DEM/in.cpus";
}

dividedProps
{
    alphaMin 0.3;
    scaleUpVol 1.0;
}

engineProps
{
        treeSearch true;
}

// ************************************************************************* //
```

Case run

In order to run the current case, the user needs to decompress the tutorial case by typing the following commands in a terminal:

```
tar -xvf Case4ParcelSettle.tar
cd Case4ParcelSettle
```

The user then enters the following command in the same terminal to run the case:

```
./parCFDDEMrun.sh
```

The output results of time vs z-coordinates are plotted in Figure(10). The trajectories are similar for both cases with different coarsening factor N = 1, 10.

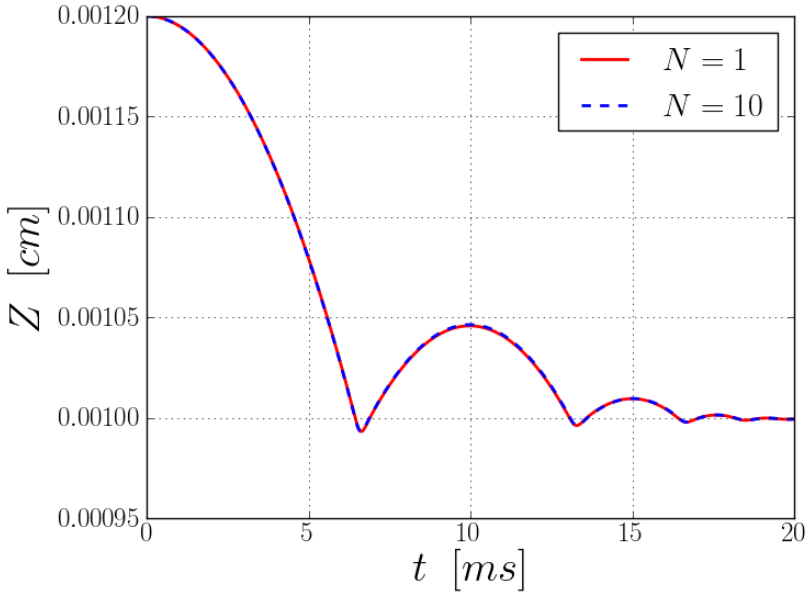


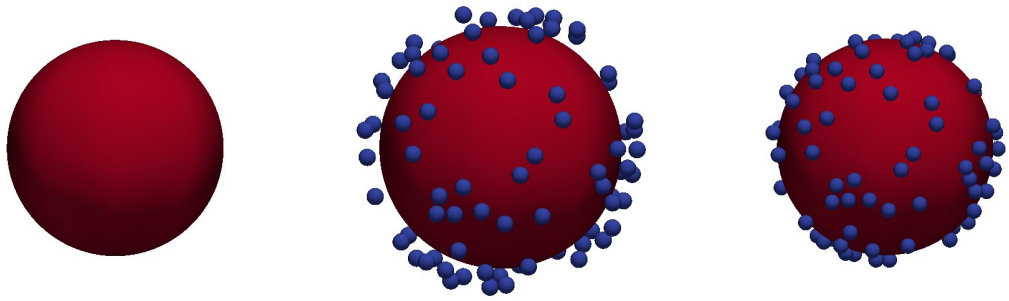
Figure 10: Temporal evolution of z-coordinate for coarsening factors N = 1 and N = 10

2.5 Case 5: Carrier-API agglomeration

Case setup

In this case, we demonstrate how to create an agglomerate out of single carrier particle and multiple API particles. We use the CGS unit system in this case. We will use two types of particles, carrier and API. In Line 63, we create a particle at $(0.05, 0.05, 0.02)$ and assign type 1 to it using *create_atoms 1*. In Line 64, we assign diameter and density to all type 1 particles, which is only one particle in this case, as shown in Figure 11a.

In Line 56 we define the region where API particles are inserted. In Line 65, we assign type 2 to API particles using *atom_type 2*. Subsequently, in line 70 we insert particle type 2 into this region, using the particle distribution defined in lines 65 and 68. The result is shown in Figure 11b. In line 92 we create a restart file every 1000000 steps in the current directory, which may be used to for future simulations (Case 6 will show an example on how to use restart file to start a simulation). In line 93 we set the number of iteration for the current simulation (1000000 steps). The carrier particle and API particles form an agglomerate due to VDW forces, as shown in Figure 11c. In Line 77, the command *compute agglomerate* determines whether any pair of particles are in contact. If they are agglomerated, the *c_agglomerate* in Line 90 for both particles will give the same ID.



(a) Insert the carrier particle (b) Insert API particles (c) Agglomerate formed

Figure 11: Illustration of forming an agglomerate with single carrier particle and multiple API particles

```
1  echo both
2
3  #DEFINE VARIABLES
4  variable dPrimBig equal 0.007
5  variable dPrimSmall equal 0.0005
6  variable rhoParticle equal 1.52
7  variable volfrac equal 0.6
8  variable rPrimBig equal ${dPrimBig}/2.0
9  variable rPrimSmall equal ${dPrimSmall}/2.0
10 variable cutOff equal ${dPrimSmall}*2
11
12 atom_style  granular
13 atom_modify map array sort 0 0
14 comm_modify mode multi vel yes
15
16 stiffnessScaling 3.177672
17
18 boundary  f f f
19 newton    off
20
21 units   cgs # was si
22 processors  * * *
23
24
25 region    reg block 0 0.1 0 0.1 0 0.1 units box
26
```

```
create_box  2 reg

neighbor  ${cutOff} bin
neigh_modify  delay 0

#pair style
pair_style  gran model hertz_parcel tangential history_parcel cohesion vdw/gu/parcel #
    Hertzian model
pair_coeff  * *

#Material properties required for VDW model
fix     vdw1 all property/global cohesionEnergyDensity peratomtypepair 2 5.0e-12 5.0e-12 5.0
    e-12 5.0e-12 # erg = 1e-7 J
fix         vdw2 all property/global sMin peratomtypepair 2 1.0e-7 1.0e-7 1.0e-7 1.0e-7
fix             vdw3 all property/global sMins peratomtypepair 2 6.276e-7 6.227e-7 6.227e-7
    6.077e-7
fix             vdw4 all property/global sO peratomtypepair 2 4.494e-7 4.445e-7 4.445e-7
    4.295e-7

#Material properties required for new pair styles
fix         m1 all property/global youngsModulus peratomtype 5.e9 5.e9
fix         m2 all property/global poissonsRatio peratomtype 0.35 0.35
fix         m3 all property/global coefficientRestitution peratomtypepair 2 0.85 0.85 0.85
    0.85
fix         m4 all property/global coefficientFriction peratomtypepair 2 0.45 0.45 0.45 0.45
fix                             m5 all property/global coefficientRollingFriction
    peratomtypepair 2 0.05 0.05 0.05 0.05

#timestep
timestep          1.e-8

#walls (liggghts 2.0)
fix       bottomwall  all wall/gran model hertz_parcel tangential history_parcel primitive
    type 1 zplane 0.0

##particle insertion##
region      regPart sphere 0.05 0.05 0.02 0.0042 units box
group     nve_group region regPart

group   nve_group region reg

#DEFINE COLLECTION OF ATOMS BELONGING TO A GROUP
#define particle to be inserted
create_atoms    1 single 0.05 0.05 0.02 units box
set     type 1 diameter 70.e-4 density 1.52
fix     pts2 all particletemplate/sphere 2 atom_type 2 density constant ${rhoParticle}
    radius constant ${rPrimSmall} volume_limit 1.e-14

#define distribution of particles for insertion
fix     pdd1 all particledistribution/discrete 66 1 pts2 1.0

fix       ins all insert/pack seed 11103 distributiontemplate pdd1 vel constant 0 0 0
    insert_every once overlapcheck yes all_in yes volumefraction_region ${volfrac} region
    regPart

#apply nve integration to all particles that are inserted as single particles
fix   integr all nve/sphere/parcel

#set coarsen factor for the parcel
fix   groupPar all property/global nparcel peratomtype 1 1
compute   agglomerate all agglomerate/atom 1e-10

run     0

#screen output
compute   1 all erotate/sphere
thermo_style  custom step atoms ke c_1 vol
thermo    1000
```

```
thermo_modify lost ignore norm no
compute_modify  thermo_temp dynamic yes

#insert the first particles so that dump is not empty
run    0
dump    dmp all custom 100000 ./post/dump*.part id type x y z ix iy iz vx vy vz fx fy fz
    radius c_agglomerate

restart   1000000 liggghts.restart
run    1000000
```

Case run

In order to run the current case, the user needs to decompress the provided case and then copy the LIGGGHTS executable from the source file location ($CFDEM_LIGGGHTS_SRC_DIR) by typing the following commands in a terminal:

```
tar -xvf Case5CarrierAPIAgglomerate.tar
cd Case5CarrierAPIAgglomerate
cp $CFDEM_LIGGGHTS_SRC_DIR/lmp_fedora_fpic .
```

The user then enters the following command in the same terminal to run the case:

```
mpirun -np 1 lmp_fedora_fpic < in.liggghts_init
```

To see how many API particles are attached to the carrier, the user can check the dump file at time step T by executing

```
vi post/dumpT.part
```

and see how many API particles have the same *c_agglomerate* as the carrier particle.
Below is the top part of a sample dumpfile, showing the particle data of the first 10 particles out of all 98 particles.

```
ITEM: TIMESTEP
1 1 0.05 0.0499999 0.0199998 0 0 0 0 0 -25 0.000467902 0.00379602 0.043924 0.0035 1
2 2 0.051215 0.0518241 0.0169574 0 0 0 0 0 -25 0.00317102 0.00476095 -0.00794053 0.00025 1
3 2 0.0481342 0.0507649 0.0231612 0 0 0 0 0 -25 -0.00486937 0.00199661 0.00825068 0.00025 1
4 2 0.0472392 0.051758 0.0181702 0 0 0 0 0 -25 -0.00720529 0.00458855 -0.00477519 0.00025 1
5 2 0.0488164 0.0492614 0.0234804 0 0 0 0 0 -25 -0.00308903 -0.00192736 0.00908371 0.00025 1
6 2 0.0531428 0.0515612 0.0186786 0 0 0 0 0 -25 0.0085712 0.00900827 0.0012767 0.00025 1
7 2 0.0525319 0.0521223 0.0182263 0 0 0 0 0 -25 0.00660803 0.00553918 -0.00462869 0.00025 1
8 2 0.0526443 0.0480277 0.0217828 0 0 0 0 0 -25 0.00690123 -0.00514712 0.00465315 0.00025 1
9 2 0.0490638 0.0470527 0.0178789 0 0 0 0 0 -25 -0.00244335 -0.00769175 -0.00553539 0.00025
    1
10 2 0.0529133 0.0523275 0.0196054 0 0 0 0 0 -25 0.0111052 0.00219275 -0.00139321 0.00025 1
```

c_agglomerate is the last entry of each row. The type of Atom 1 is 1. Thus Atom 1 is the carrier particle. As seen from the file, *c_agglomerate* of Atom 2-10 are all 1. This means that they are attached to the carrier. To visualize the agglomerate, the user should first type the following command to convert particle data into VTK files.

```
lpp post/dump*.part
```

The user then loads the VTK file into a visualization software. In the case of ParaView®, the user should use the Glyph option and choose particle radius to be the Glyph size. Please refer to the user manual of ParaView® for detail https://www.paraview.org/paraview-guide/.

2.6 Case 6: Deagglomeration induced by collision with a wall

Case setup

In this case, we will demonstrate the collision of a carrier-API agglomerate with a wall by restarting from a saved restart file.

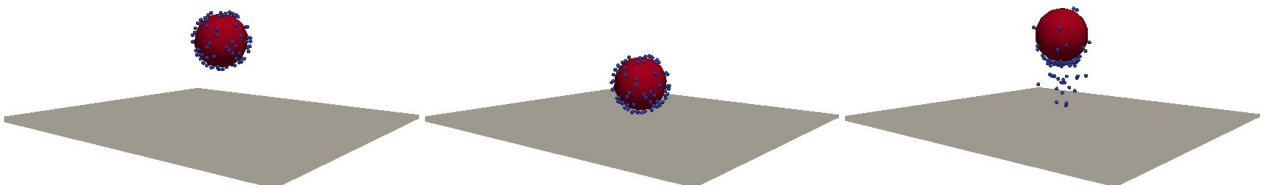


Figure 12: Illustration of deagglomeration induced by collision with a wall

In Line 24, the case reads the restart file saved from the previous test. Line 49 generates a plane wall for the collision. Line 58 assigns velocity $(0, 0, -25)$ to all the particles. Line 73 runs the simulation for 100000 time steps. Readers are referred to the *in.liggghts_restart* for details.

```
1  # file in.liggghts_restart
2  echo both
3
4  #DEFINE VARIABLES
5  variable dPrimBig equal 0.007
6  variable dPrimSmall equal 0.0005
7  variable rhoParticle equal 1.52
8  variable volfrac equal 0.3
9  variable rPrimBig equal ${dPrimBig}/2.0
10 variable rPrimSmall equal ${dPrimSmall}/2.0
11 variable cutOff equal ${dPrimSmall}*2
12
13 atom_style  granular
14 atom_modify map array sort 0 0
15 comm_modify mode multi vel yes
16
17 stiffnessScaling 3.177672
18
19 boundary  f f f
20 newton    off
21
22 units   cgs # was si
23 processors  * * *
24
25 read_restart  ./restart/liggghts.restart.1000000
26
27 region    reg block 0 0.1 0 0.1 0 0.1 units box
28
29 neighbor  ${cutOff} bin
30 neigh_modify  delay 0
31
32 #Material properties required for VDW model
33 fix     vdw1 all property/global cohesionEnergyDensity peratomtypepair 2 1.0e-12 1.0e-12 1.0
       e-12 1.0e-12 # erg = 1e-7 J
34 fix         vdw2 all property/global sMin peratomtypepair 2 1.0e-7 1.0e-7 1.0e-7 1.0e-7
35 fix             vdw3 all property/global sMins peratomtypepair 2 6.276e-7 6.227e-7 6.227e-7
       6.077e-7
36 fix             vdw4 all property/global sO peratomtypepair 2 4.494e-7 4.445e-7 4.445e-7
       4.295e-7
37
38 #Material properties required for new pair styles
39 fix         m1 all property/global youngsModulus peratomtype 5.e9 5.e9
```

```
fix         m2 all property/global poissonsRatio peratomtype 0.35 0.35
fix         m3 all property/global coefficientRestitution peratomtypepair 2 0.85 0.85 0.85
    0.85
fix         m4 all property/global coefficientFriction peratomtypepair 2 0.45 0.45 0.45 0.45
fix                              m5 all property/global coefficientRollingFriction
    peratomtypepair 2 0.1 0.1 0.1 0.1

#timestep, gravity
timestep          1.e-8
fix      gravi all gravity/parcel 981 vector 0.0 0.0 -1.0

#walls (liggghts 2.0)
fix       bottomwall  all wall/gran model hertz_parcel tangential history_parcel primitive
    type 1 zplane 0.0

#apply nve integration to all particles that are inserted as single particles
fix    integr all nve/sphere/parcel

#set coarsen factor for the parcel
fix    groupPar all property/global nparcel peratomtype 1 1
compute    agglomerate all agglomerate/atom 1e-10

velocity   all set 0 0 -25 units box

run      0

#screen output
compute    1 all erotate/sphere
thermo_style  custom step atoms ke c_1 vol
thermo     1000
thermo_modify lost ignore norm no
compute_modify  thermo_temp dynamic yes

#insert the first particles so that dump is not empty
run    0
dump     dmp all custom 1000 ./post/dump*.part id type x y z ix iy iz vx vy vz fx fy fz
    radius c_agglomerate

run    100000
```

Case run

In order to run the current case, the user needs to decompress the provided case and then copy the LIGGGHTS executable from the source file location ($CFDEM_LIGGGHTS_SRC_DIR) by typing the following commands in a terminal:

```
tar -xvf Case6AgglomerateWallCollision.tar
cd Case6AgglomerateWallCollision
cp $CFDEM_LIGGGHTS_SRC_DIR/lmp_fedora_fpic .
```

The user then enters the following command to copy the restart file from Case 5 directory.

```
cp ../Case5CarrierAPIAgglomerate/liggghts.restart.1000000 restart/
```

The user enters the following command in the same terminal to run the case:

```
mpirun -np 1 lmp_fedora_fpic < in.liggghts_restart
```

To check how many API left the agglomerate, the user can check the dump file at time step T by executing

```
vi post/dumpT.part
```

and see how many API particles have different *c_agglomerate* from the carrier particle.
Below is the top part of a sample dumpfile, showing the particle data of the first 10 particles out of all 98 particles.

```
ITEM: TIMESTEP
1100000
```

```
ITEM: NUMBER OF ATOMS
98
ITEM: BOX BOUNDS ff ff ff
0.0000000000000000e+00 1.0000000000000001e-01
0.0000000000000000e+00 1.0000000000000001e-01
0.0000000000000000e+00 1.0000000000000001e-01
ITEM: ATOMS id type x y z ix iy iz vx vy vz fx fy fz radius c_agglomerate
1 1 0.0497195 0.0498843 0.0130366 0 0 0 -0.774672 -0.323404 24.4592 -0.00228509 0.000258565
    -0.00135076 0.0035 1
2 2 0.0507039 0.050386 0.00945316 0 0 0 0.835998 0.437379 25.0197 2.73169e-05 -3.40164e-05
    2.30902e-05 0.00025 1
3 2 0.0475607 0.0508429 0.0101241 0 0 0 13.1216 -5.8488 12.3403 5.78296e-05 -9.62065e-05
    -6.17087e-05 0.00025 1
4 2 0.0517377 0.0503148 0.00560168 0 0 0 7.13328 -3.74719 8.45078 -4.84847e-05 -3.65589e-06
    -8.35757e-05 0.00025 4
5 2 0.046571 0.0478937 0.0126048 0 0 0 0.571411 0.399996 11.3103 -8.87029e-06 -4.36057e-05
    0.000305241 0.00025 1
6 2 0.0534553 0.0501822 0.0131673 0 0 0 -0.718836 -5.43061 34.5088 -1.96382e-05 0.000337868
    -0.000310115 0.00025 1
7 2 0.0511694 0.0504482 0.00962462 0 0 0 0.7398 0.445167 25.2478 8.27238e-06 3.11419e-05
    -2.01688e-06 0.00025 1
8 2 0.0519452 0.0485371 0.010336 0 0 0 0.422102 0.191486 25.2344 -1.52247e-05 3.49406e-05
    -5.93126e-05 0.00025 1
9 2 0.0501442 0.0491365 0.00938664 0 0 0 0.801024 0.441511 24.5213 0.00012235 -7.32036e-05
    -3.31549e-05 0.00025 1
10 2 0.0479347 0.0529445 0.0118072 0 0 0 -12.041 -4.43487 30.582 -9.69703e-05 0.000117032
    0.000447192 0.00025 1
```

As seen from the file, *c_agglomerate* for Atom 4 is no longer the same as that for Atom 1. Thus, Atom 4 has detached from the carrier particle.

3 Inhaler Simulation

After going through the series of micro-scale tests, we now present an actual inhaler simulation in this section. The case is adapted from [6][5]. Both studies track all the carrier particles while treating API particles as passive scalars. Through various micro-scale DEM tests that track both carrier particles and API particles, they formulated kinetic models for API transport due to carrier-carrier collision, carrier-wall collision, API detachment and reattachment due to gas dynamics. In the following section, we will present the simulation using a different approach: instead of doing Eulerian coarsening, we will do Lagrangian coarsening by utilizing the representative particle model discussed in Section 4 of the theory guide as well as Liu et al. 2021 [4].

The case is compressed in *CaseInhaler.tar* file. Execute the following commands to enter the case folder.

```
tar -xvf CaseInhaler.tar
cd CaseInhaler
```

3.1 Geometry Description

The inhaler prototype is known as the screen-haler[6], with the geometry outlined in Figure 13.

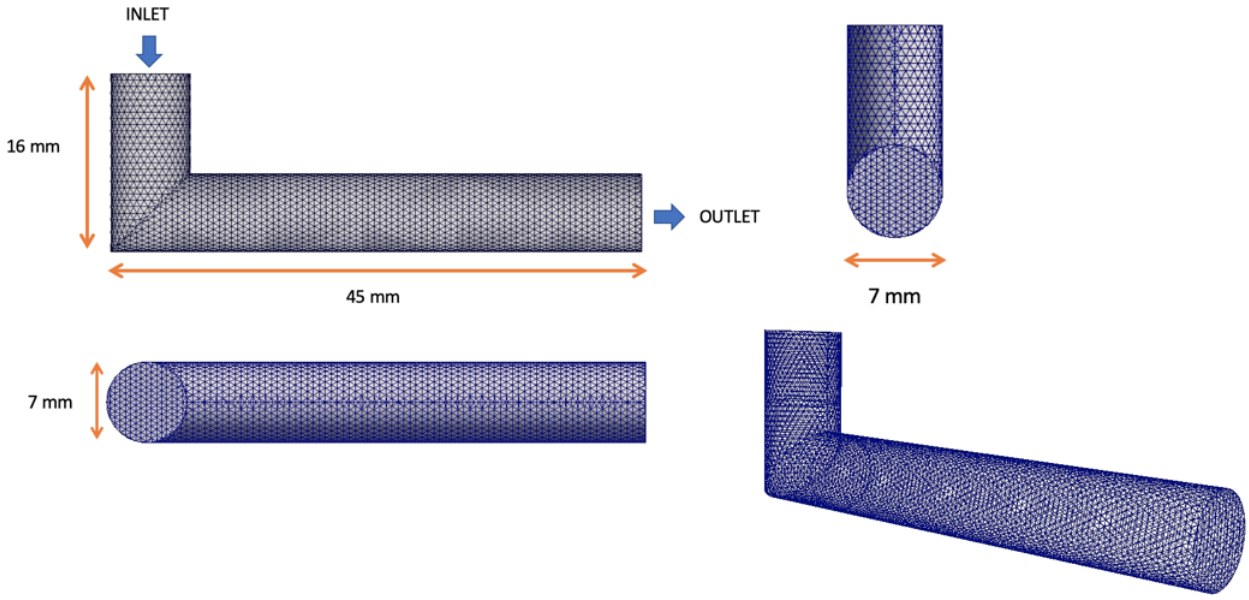
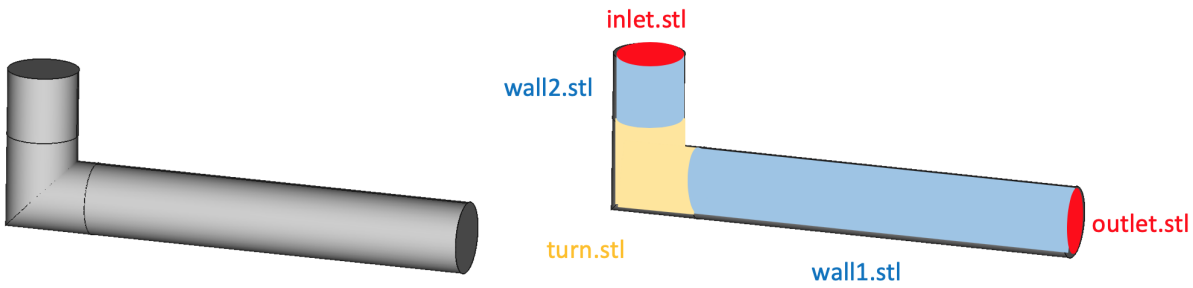


Figure 13: Three-view diagram of the inhaler prototype

The 3D model for the inhaler prototype can be constructed using any CAD software. Figure 14a illustrates a sample 3D model generated using FreeCAD®[1]. The 3D model must be saved in stl format. In this case, it has been saved as *volume.stl*. For the purpose of defining boundary conditions, the surface of the 3D model may be decomposed to different surface patches in the CAD software. Figure 14b is an example decomposition for this geometry. Each patch needs to be saved in a separate stl file.



(a) 3D model: volume.stl (b) Patch decomposition

Figure 14: 3D model and patch decomposition

These stl files can then be passed into an OpenFOAM® utility known as snappyHexMesh to generate the mesh for CFD. For detail on usage of snappyHexMesh, please refer to the official guide https://cfd.direct/openfoam/user-guide/v6-snappyhexmesh/. The following video provides step-by-step guidance on how to generate mesh from stl files using snappyHexMesh https://www.youtube.com/watch?v=ObsFQUiVi1U[2]. The generated mesh and patches will be automatically placed in *CaseInhaler/CFD/constant/polymesh/* folder by the utility and is ready to be used by CFD.
For DEM, stl files can be directly imported in LIGGGHTS input files such as *in.liggghts_init* and *in.liggghts_restart*. In the following example, Line 1-4 import *inlet.stl*, *wall1.stl*, *turn.stl*, *wall2.stl*. Line 5 creates the mesh from the four imported files. The force models for wall-collision are also specified here as *model hertz_parcel tangential history_parcel rolling_friction cdt*.

```
1 fix inlet all mesh/surface/stress file YOUR_MESH_DIRECTORY/inlet.stl type 1
2 fix wall1 all mesh/surface/stress file YOUR_MESH_DIRECTORY/wall1.stl type 1
3 fix turn all mesh/surface/stress file YOUR_MESH_DIRECTORY/turn.stl type 1
4 fix wall2 all mesh/surface/stress file YOUR_MESH_DIRECTORY/wall2.stl type 1
5 fix granwalls all wall/gran model hertz_parcel tangential history_parcel rolling_friction
    cdt mesh n_meshes 4 meshes inlet wall1 turn wall2 restart no
```

3.2 Stage 1: Sedimentation

We would like to first create a pile of carrier-API agglomerates via sedimentation. Line 34 specifies the forces for particle-particle interaction. These include contact forces, cohesion forces and rolling friction. Line 62 specifies the forces for particle-wall interaction, which include contact forces and rolling friction. This can be done in a manner similar to Cases 5 and 6. Line 65 defines the region to be inserted with particles. The region is a *cone* with its axis along *y* direction. The x-coordinate and z-coordinate of the cone axis are 4.3 and 0 respectively. The radii of the low end and high end of the cone are 0.08 and 0.01 respectively, while the bounds of the cone in y-coordinate are -0.34 and -0.235 respectively. Please see LIGGGHTS® documentation for "region" command for detail https://www.cfdem.com/media/DEM/docu/region.html.
Lines 70-71 define the particles to be inserted, where properties like *rhoParticle* and *rPrimBig*, rPrimSmall are defined in Lines 5-11. Line 74 defines the distribution for particle insertion, where the volume fraction of carrier particle is 99.5% and the volume fraction of API particle 0.5%. Or, in other words, the region is filled to 30 volume percent with particles, out of which 99.5% is carrier and 0.5% API particles. Line 76 inserts the particle at solid volume fraction *${volfrac}*, which is 0.3 as specified in Line 8. Line 72 specifies the coarsening factor for each particle type, *1* for Type 1 (Carrier) and *10* for Type 2 (API).
Since the number of particles inserted in this case is around 18,000, a much larger particle number than the previous micro-scale cases introduced in Section 2, multiple processors are required for this parallel run. Line 37 distributes the number of particles in each processor every 1000 steps. Please refer to https://lammps.sandia.gov/doc/fix_balance.html for details about the load balancing feature of LAMMPS®. Line 100 creates a restart file every 1000000 steps, which can be used as the starting state of particles for Stage 2: Inhalation.

```
1 #in.liggghts_init file
2 echo both
```

```
3
4  #DEFINE VARIABLES
5  variable dPrimBig equal 0.007
6  variable dPrimSmall equal 0.0005
7  variable rhoParticle equal 1.52
8  variable volfrac equal 0.3
9  variable rPrimBig equal ${dPrimBig}/2.0
10 variable rPrimSmall equal ${dPrimSmall}/2.0
11 variable cutOff equal ${dPrimSmall}*2
12
13 atom_style  granular
14 atom_modify map array sort 0 0
15 comm_style  tiled
16 comm_modify mode single vel yes
17
18 stiffnessScaling 3.177672
19
20 boundary  f f f
21 newton    off
22
23 units   cgs # was si
24 processors    * * *
25
26 region    reg block -0.5 5.0 -0.4 0.4 -0.4 1.4 units box
27
28 create_box  2 reg
29
30 neighbor  ${cutOff} bin
31 neigh_modify  delay 0
32
33 #pair style
34 pair_style  gran model hertz_parcel tangential history_parcel cohesion vdw/gu/parcel
       rolling_friction cdt
35 pair_coeff  * *
36
37 fix     load0 all balance 1000 1.1 shift xyz 1000 1.1
38
39 #Material properties required for VDW model
40 fix     vdw1 all property/global cohesionEnergyDensity peratomtypepair 2 5.0e-12 5.0e-12 5.0
       e-12 5.0e-12 # erg = 1e-7 J
41 fix         vdw2 all property/global sMin peratomtypepair 2 1.0e-7 1.0e-7 1.0e-7 1.0e-7
42 fix             vdw3 all property/global sMins peratomtypepair 2 6.276e-7 6.227e-7 6.227e-7
       6.077e-7
43 fix             vdw4 all property/global sO peratomtypepair 2 4.494e-7 4.445e-7 4.445e-7
       4.295e-7
44
45 #Material properties required for new pair styles
46 fix         m1 all property/global youngsModulus peratomtype 5.e9 5.e9
47 fix         m2 all property/global poissonsRatio peratomtype 0.35 0.35
48 fix         m3 all property/global coefficientRestitution peratomtypepair 2 0.85 0.85 0.85
       0.85
49 fix         m4 all property/global coefficientFriction peratomtypepair 2 0.45 0.45 0.45 0.45
50 fix                            m5 all property/global coefficientRollingFriction
       peratomtypepair 2 0.05 0.05 0.05 0.05
51
52 #timestep, gravity
53 timestep       1.e-08
54 fix   gravi all gravity/parcel 981 vector 0.0 0.0 -1.0
55
56 #walls (liggghts 2.0)
57 fix   inlet all mesh/surface/stress file ../meshes/inlet.stl type 1
58 fix wall1 all mesh/surface/stress file ../meshes/wall1.stl type 1
59 fix turn all mesh/surface/stress file ../meshes/turn.stl type 1
60 fix   wall2 all mesh/surface/stress file ../meshes/wall2.stl type 1
61 #fix  outlet all mesh/surface/stress file ../meshes/outlet.stl type 1
62 fix   granwalls all wall/gran model hertz_parcel tangential history_parcel rolling_friction
       cdt mesh n_meshes 4 meshes inlet wall1 turn wall2 restart no
63
```

```
##particle insertion##
region       regPart cone z 3.9 0.0 0.08 0.01 -0.34 -0.235 units box
group      nve_group region regPart

#DEFINE COLLECTION OF ATOMS BELONGING TO A GROUP
#define particle to be inserted
fix        pts1 all particletemplate/sphere 1 atom_type 1 density constant ${rhoParticle}
    radius constant ${rPrimBig} volume_limit 1.e-14
fix        pts2 all particletemplate/sphere 2 atom_type 2 density constant ${rhoParticle}
    radius constant ${rPrimSmall} volume_limit 1.e-14

#define distribution of particles for insertion
fix        pdd1 all particledistribution/discrete 66 2 pts1 0.995357 pts2 0.004643

fix          ins all insert/pack seed 86 distributiontemplate pdd1 vel constant 0 0 0
    insert_every once overlapcheck yes all_in yes volumefraction_region ${volfrac} region
    regPart

variable       time equal step*dt

#apply nve integration to all particles that are inserted as single particles
fix    integr all nve/sphere/parcel

fix      print0 all print/nlocal print_step 1000
fix    groupPar all property/global nparcel peratomtype 1 10
compute    agglomerate all agglomerate/atom 1e-10

run      0

#screen output
compute    1 all erotate/sphere
thermo_style  custom step atoms ke c_1 vol
thermo     1000
thermo_modify lost ignore norm no
compute_modify  thermo_temp dynamic yes

#insert the first particles so that dump is not empty
run    0
dump     dmp all custom 100000 ./post/dump*.part id type x y z ix iy iz vx vy vz fx fy fz
    radius c_agglomerate

restart    1000000 liggghts.restart
run    10000000
```

Case Run

In order to run the current case, the user needs to decompress the provided case and then copy the LIGGGHTS executable from the source file location ($CFDEM_LIGGGHTS_SRC_DIR) by typing the following commands in a terminal:

```
cd CaseInhaler/Sedimentation
cp $CFDEM_LIGGGHTS_SRC_DIR/lmp_fedora_fpic .
```

The user then executes the following command in the same terminal to run the case on 14 processors. The number of processors allocated for this *mpirun* case can by modified by changing the number following the flag *-np*.

```
mpirun -np 14 lmp_fedora_fpic < in.liggghts_init
```

Visualization

In order to check if the sediment has formed as expected, the user may visualize the data with visualization software. Here we show an example with ParaView® to visualize the case by following the steps outlined below:

- use command *lpp* to convert particle data stored in *dump.part* files to vtk files, by executing the following commands

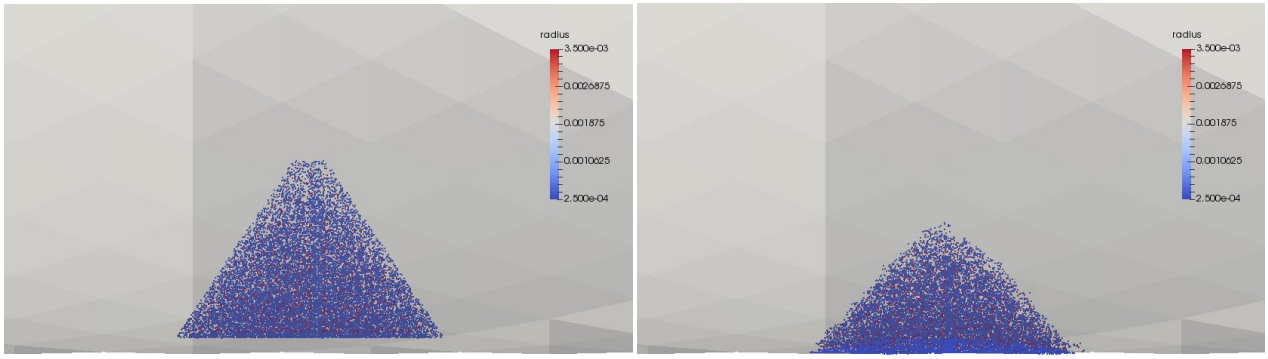
```
openfoam22
lpp post/*.part
```

- open ParaView® by executing

```
paraview
```

The following link provides a helpful quick start to ParaView® http://www.bu.edu/tech/support/research/training-consulting/online-tutorials/paraview/. The complete user manual can be found at https://www.paraview.org/paraview-guide/
Load *volume.stl* and all the vtk files into ParaView®. Figures 15 and 16 illustrate the ParaView® visualizations at time step = 0 and 5000000.

Figure 15: Visualization at step 0



(a) Visualization at step 0 (Zoomed in)

(b) Visualization at step 5000000 (Zoomed in)

Figure 16: Visualization of sedimentation

Figure 16a is a zoomed-in version of the particle-inserted region shown in Figure 15. Figure 16b shows the state of the particles after 5 million time steps (0.05s physical time). In Line 85, the command *compute agglomerate* determines whether any pair of particles are in contact. If they are agglomerated, the *c_agglomerate* for both particles will give the same number. In order to check whether the particles have formed an agglomerate, the user can check the dump file that corresponds to this time step (i.e. *dump5000000.part*) and see if the majority of the particles have the same *c_agglomerate* number.

3.3 Stage 2: Inhalation

DEM

We would like to continue our simulation from the sedimentation formed in Stage 1. To do so, we need to utilize the restart file generated in Stage 1. Execute the following commands to go to the *DEM* folder.

```
cd DEM
```

Below are the *in.liggghts_restart* file in the DEM folder. As mentioned earlier in Case 6, Line 26 reads the particle data stored in *liggghts.restart.5000000*.
The user cannot redefine *pair_style*, i.e. particle-particle interaction forces, as they are already defined in the restart file during the sedimentation simulation. However, the material properties for contact forces (Line 42-45), the material properties for cohesion model (Line 36-39), can be modified here.
Line 53 to 56 imports the wall from STL files, while Line 58 define the particle-wall interactions which include contact forces and rolling friction. For a restart case, the argument for *restart* needs to be *yes*.

```
#in.liggghts_restart file
echo both

#DEFINE VARIABLES
variable dPrimBig equal 0.007
variable dPrimSmall equal 0.0005
variable rhoParticle equal 1.52
variable volfrac equal 0.3
variable rPrimBig equal ${dPrimBig}/2.0
variable rPrimSmall equal ${dPrimSmall}/2.0
variable cutOff equal ${dPrimSmall}*2

atom_style  granular
atom_modify map array sort 0 0
comm_style  tiled
comm_modify mode single vel yes

stiffnessScaling 3.177672

boundary  f f f
```

```
newton     off

units    cgs # was si
processors     * * *

read_restart   ../CFD/restart/liggghts.restart.5000000

region     reg block -0.5 5.0 -0.4 0.4 -0.4 1.4 units box

neighbor  ${cutOff} bin
neigh_modify  delay 0

fix      load0 all balance 1000 1.1 shift xyz 1000 1.1

#Material properties required for VDW model
fix      vdw1 all property/global cohesionEnergyDensity peratomtypepair 2 1.0e-12 1.0e-12 1.0e-12 1.0e-12 # erg = 1e-7 J
fix          vdw2 all property/global sMin peratomtypepair 2 1.0e-7 1.0e-7 1.0e-7 1.0e-7
fix              vdw3 all property/global sMins peratomtypepair 2 6.276e-7 6.227e-7 6.227e-7 6.077e-7
fix              vdw4 all property/global sO peratomtypepair 2 4.494e-7 4.445e-7 4.445e-7 4.295e-7

#Material properties required for new pair styles
fix          m1 all property/global youngsModulus peratomtype 5.e9 5.e9
fix          m2 all property/global poissonsRatio peratomtype 0.35 0.35
fix          m3 all property/global coefficientRestitution peratomtypepair 2 0.85 0.85 0.85 0.85
fix          m4 all property/global coefficientFriction peratomtypepair 2 0.45 0.45 0.45 0.45
fix                            m5 all property/global coefficientRollingFriction peratomtypepair 2 0.05 0.05 0.05 0.05

#timestep, gravity
timestep        1.e-08
fix    gravi all gravity/parcel 981 vector 0.0 0.0 -1.0

#walls (liggghts 2.0)
fix    inlet all mesh/surface/stress file ../meshes/inlet.stl type 1
fix  wall1 all mesh/surface/stress file ../meshes/wall1.stl type 1
fix  turn all mesh/surface/stress file ../meshes/turn.stl type 1
fix    wall2 all mesh/surface/stress file ../meshes/wall2.stl type 1
#fix   outlet all mesh/surface/stress file ../meshes/outlet.stl type 1
fix    granwalls all wall/gran model hertz_parcel tangential history_parcel rolling_friction cdt mesh n_meshes 4 meshes inlet wall1 turn wall2 restart yes

#cfd coupling
fix              fcpus all cpus debug no polyhedron yes compress no
fix              cfd all couple/cfd couple_every 1000 mpipun
fix          cfd2 all couple/cfd/force
fix    cfd3 all couple/cfd/turbulentDispersion fluidViscosity 1.511e-01 delta 0.02

variable       time equal step*dt

#apply nve integration to all particles that are inserted as single particles
fix    integr all nve/sphere/parcel

fix      print0 all print/nlocal print_step 1000
fix    groupPar all property/global nparcel peratomtype 1 10
compute    agglomerate all agglomerate/atom 1e-10

run      0

#screen output
compute    1 all erotate/sphere
thermo_style  custom step atoms ke c_1 vol
thermo     1000
thermo_modify lost ignore norm no
compute_modify  thermo_temp dynamic yes
```

```
83
84 #insert the first particles so that dump is not empty
85 run    0
86 dump    dmp all custom 50000 ../DEM/post/dump*.part id type x y z ix iy iz vx vy vz fx fy fz
       radius c_agglomerate
87
88 run    0
89
90 restart    1000000 liggghts.restart
```

CFD

Now execute the following command to go to the *CFD* folder.

```
1 cd ../CFD/
```

The user executes the following command to copy the restart files generated from Stage1: Sedimentation to *CFD/restart*.

```
1 cp ../Sedimentation/liggghts.restart.* restart/
```

First, the user needs to define the boundary condition, which is located in *CFD/0*. The following file is an example boundary condition for gas-phase velocity, which can be found in following file *U*. The boundary conditions for patch wall1 (Line 25) , wall2 (Line 31), turn (Line 37) are (000), while the boundary condition for the outlet is *zeroGradient*. For detail on the standard boundary conditions in OpenFOAM®, please refer to the following link https://www.openfoam.com/documentation/user-guide/standard-boundaryconditions.php.

```
1  /*--------------------------------*- C++ -*----------------------------------*\
2  | =========                 |                                                 |
3  | \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
4  |  \\    /   O peration     | Version:  1.6                                   |
5  |   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
6  |    \\/     M anipulation  |                                                 |
7  \*---------------------------------------------------------------------------*/
8  FoamFile
9  {
10     version     2.0;
11     format      ascii;
12     class       volVectorField;
13     location    "0";
14     object      U;
15 }
16 // * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //
17
18 dimensions      [0 1 -1 0 0 0 0];
19
20 internalField   uniform (0 0 0);
21
22 boundaryField
23 {
24
25     wall1
26     {
27         type            fixedValue;
28         value           uniform (0 0 0);
29     }
30
31     wall2
32     {
33         type            fixedValue;
34         value           uniform (0 0 0);
35     }
36
37     turn
38     {
39         type            fixedValue;
```

```
40          value           uniform (0 0 0);
41      }
42
43      inlet
44      {
45        type    uniformFixedValue;
46        uniformValue  tableFile;
47        tableFileCoeffs
48        {
49          dimenstions [0 1 -1 0 0 0 0];
50          fileName  "./ramp";
51          outOfBounds clamp;
52          interpolationScheme linear;
53        }
54      }
55
56      outlet
57      {
58        type    zeroGradient;
59      }
60
61  }
62
63
64  // ************************************************************************* //
```

Lines 43 to 54 of the file *U* defines the boundary condition for the inlet by reading from the file *ramp*, which is shown below. Line 2 defines the velocity at time *0.0* to be *(0 0 0)*. Line 3 defines the velocity at time *0.2* to be *(0 0 -3000)*. Line 52 of the file *U* specifies the interpolation scheme to be *linear*, which means that the gas-phase velocity at the inlet at any time between $t = 0.0$ and $t = 0.2$ will be determined by linear interpolation. Line 51 of the file *U* means that for $t < 0$, the velocity will be clamped at constant *(0 0 0)* and similarly for $t > 0.2$, velocity will be constant at *(0 0 -3000)*.

```
1  (
2  (0.0 (0 0 0))
3  (0.2 (0 0 -3000))
4  )
```

The inlet velocity profile defined in the file is illustrated below.

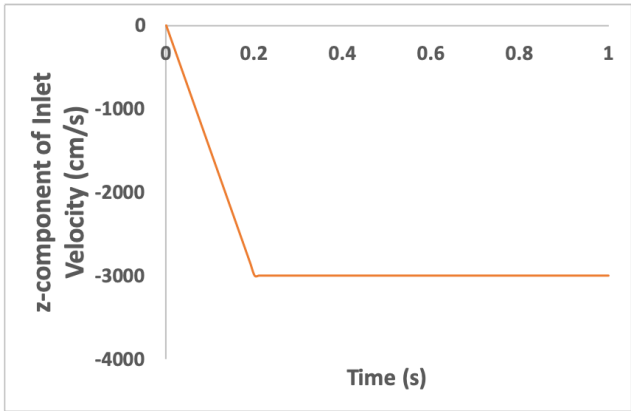


Figure 17: Inlet velocity profile

CFD-DEM coupling properties are specified in the file *constant/couplingProperties*, which is shown below. Lines 53 to 57 specifies the gas-solid interaction model for the simulation. In this case, WenYu drag model (Line 55) and gradP force model (Line 56) are specified. In this code, turbulent dispersion is manifested as a stochastic drag force in addition to the WenYu drag force. Thus, we incorporated the dispersion force due to turbulent dispersion into the WenYu drag model. Please refer to the theory guide for detail of the turbulent

dispersion model implemented in this code.

Line 82 specifies the properties for model *WenYuTurbulentDispersionDrag*. Notice that Line 90 "*parcel*" specifies that this is using the coarse-grained version of WenYu drag model. Line 91 specifies the particle type for API as *2*. which has to be consistent with the particle type defined in DEM. Line 92 specifies the coarsening factor as *10*. Line 93 and 94 specify the diameter and density of API.

Similarly, Line 97 specifies the properties for model *gradPParcelForce*. Line 104 specifies the particle type for API particles, while Lines 105 and 106 specify the coarsening factor and diameter of API. Line 111 specifies the DEM input file for this CFD-DEM coupling simulation. Line 122 specifies the properties for void fraction model *apiParcel*. Notice that Lines 126 and 127 specify the coarsening factor and particle type of API respectively.

```
1  /*--------------------------------------------------------------------------*\
2  | =========                 |                                                |
3  | \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox          |
4  |  \\    /   O peration     | Version:  1.4                                  |
5  |   \\  /    A nd           | Web:      http://www.openfoam.org              |
6  |    \\/     M anipulation  |                                                |
7  \*--------------------------------------------------------------------------*/
8
9
10 FoamFile
11 {
12     version          2.0;
13     format           ascii;
14
15     root             "";
16     case             "";
17     instance         "";
18     local            "";
19
20     class            dictionary;
21     object           couplingProperties;
22 }
23
24 // * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //
25
26 //===========================================================================//
27 // sub-models & settings
28
29 modelType A; // A or B
30
31 couplingInterval 1000;
32
33 voidFractionModel apiParcel;//divided;//centre;//bigParticle;//
34
35 locateModel engine;//standard;
36
37 meshMotionModel noMeshMotion;
38
39 regionModel allRegion;
40
41 IOModel basicIO; //trackIO; //
42
43 dataExchangeModel twoWayMPIpu;//twoWayMPI;//twoWayFiles;//oneWayVTK;//
44
45 averagingModel dense;//dilute;//
46
47 clockModel standardClock;
48
49 smoothingModel off;
50
51 probeModel off;
52
53 forceModels
```

```
(
    WenYuTurbulentDispersionDrag
    gradPParcelForce
);

momCoupleModels
(
    implicitCouple
    explicitCoupleSource
);

turbulenceModelType LESProperties;//RASProperties;//

chargeDensityModel chargeDensityModel;

//=========================================================================//
// sub-model properties

implicitCoupleProps
{
    velFieldName "U";
    granVelFieldName "Us";
    voidfractionFieldName "voidfraction";
}
explicitCoupleProps
{
}

WenYuTurbulentDispersionDragProps
{
    velFieldName "U";
    granVelFieldName "Us";
    densityFieldName "rho";
    voidfractionFieldName "voidfraction";
    interpolation;
    verbose;
    parcel;
    apiType 2;
    npart 10;
    dpart 5.e-4;
    rhopart 1.52;
}

gradPParcelForceProps
{
    pFieldName "p";
    velocityFieldName "U";
    densityFieldName "rho";
    verbose;
    interpolation;
    apiType 2;
    npart 10;
    dpart 5.e-4;
}

twoWayMPIpuProps
{
    liggghtsPath "../DEM/in.liggghts_restart";
//    liggghtsPath "../DEM/in.liggghts_init";
    cpusPath "../DEM/in.cpus";
}

dividedProps
{
    alphaMin 0.3;
    scaleUpVol 1.0;
}

```

```
apiParcelProps
{
    alphaMin 0.3;
    scaleUpVol 1.0;
    nParcel 10;
    apiType 2;
}

engineProps
{
        treeSearch true;
}
// ************************************************************************* //
```

Similar to Stage 1: Sedimentation, Stage 2 should be a parallel run. *system/decomposeParDict* specifies the total number of subdomains for the CFD and their geometrical decomposition, as shown below. Line 18 specifies the number of subdomains (usually should equal the number of processors allocated) for this parallel simulation. Lines 20-26 specifies the spatial decomposition for domains. For detail on spatial decomposition, please refer to the following link https://cfd.direct/openfoam/user-guide/v6-running-applications-parallel/

```
/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  1.6                                   |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    location    "system";
    object      decomposeParDict;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

numberOfSubdomains 14;

method          simple;

simpleCoeffs
{
    n               ( 14 1 1 ); // Decomposition along directions ( nCPU_x nCPU_y nCPU_z )
    nCPU_x*nCPU_y*nCPU_z = numberOfSubdomains
    delta           0.001;
}

// ************************************************************************* //
```

Case Run

After specifying the boundary conditions, CFD-DEM coupling properties and number of processors allocated, the user need to execute the following command in the CFD directory) to carry out the spatial decomposition.

```
openfoam22
decomposePar
```

The number of processors allocated for this parallel simulation can be specified in Line 22 of file *CaseInhaler/parCFDDEMrun.sh*, as shown below

```
#!/bin/bash

#------------------------------------------------------------------------------#
# allrun script for testcase as part of test routine
# run settlingTest CFD part
# Christoph Goniva - Feb. 2011
```

```
#------------------------------------------------------------------------------#

#- source CFDEM env vars
. ~/.bashrc

#- include functions
source $CFDEM_SRC_DIR/etc/functions.sh

#------------------------------------------------------------------------------#
#- define variables
casePath="$(dirname "$(readlink -f ${BASH_SOURCE[0]})")"
logpath=$casePath
headerText="DPIrun"
logfileName="log_$headerText"
solverName="cfdemSolverDPI"
nrProcs="14"
machineFileName="none"    # yourMachinefileName | none
debugMode="off"           # on | off | strict
testHarnessPath="$CFDEM_TEST_HARNESS_PATH"
#------------------------------------------------------------------------------#

#- call function to run a parallel CFD-DEM case
parCFDDEMrun $logpath $logfileName $casePath $headerText $solverName $nrProcs
    $machineFileName $debugMode
```

Then execute the following command to run the simulation.

```
openfoam22
./parCFDDEMrun.sh
```

Visualization

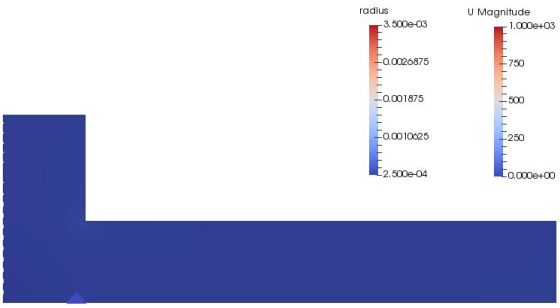
In order to visualize CFD data, the user needs to execute the following commands, in order to reconstruct the full CFD domain from all the subdomains, and then convert to VTK files

```
cd CFD
openfoam22
reconstructPar -noLagrangian
foamToVTK
```

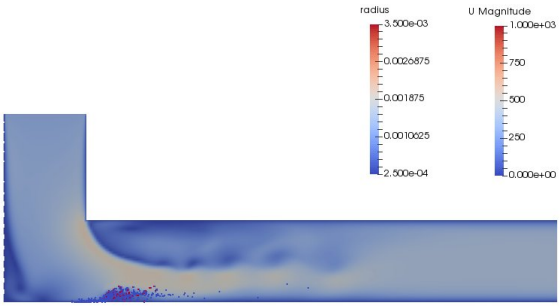
The VTK files for CFD can be found in *CFD/VTK/*. Similarly to Stage 1: Sedimentation, we execute the following command to convert DEM data into VTK files.

```
cd ../DEM/post/
lpp *
```

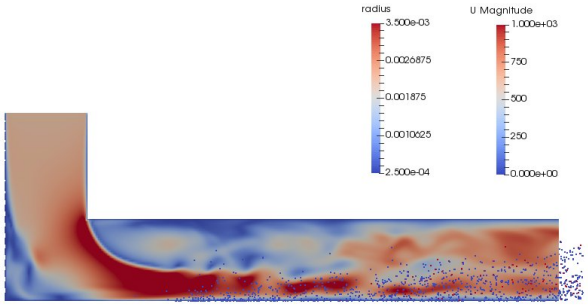
Now the user can load *volume.stl*, VTK files for CFD and VTK files for DEM into the visualization software. Below are some examples using ParaView®.



(a) Visualization at t = 0



(b) Visualization at t = 0.02s



(c) Visualization at t = 0.04s

Figure 18: Visualization of sedimentation

3.4 Data post-processing

DEM data are stored in dump files found in *CaseInhaler/DEM/post/*. They provide useful information for analyzing agglomeration and deagglomeration in DPI simulations, as illustrated in cases 5 and 6. Here we will illustrate an example analysis with a sample postprocessing tool that computes the fine particle fraction, defined as the following.

$$\text{Fine particle fraction} = \frac{\text{Number of API dispersed as singlets}}{\text{Number of API in the system}} \tag{3}$$

The postprocessing tool provided here is for illustration purpose and is specific for analyzing fine particle fraction in this particular geometry. The readers are advised to develop their own postprocessing tool tailored to their specific needs.. Incidentally, we discovered an issue in the postprocessing code, which is not relevant for the purpose of this tutorial.

In the *CaseInhaler* directory, the user executes the following command, in order to copy the postprocessing tools from the *postprocessing* folder to *DEM/post*

```
cp postprocessing/* DEM/post/
```

Execute the following command to preprocess the data. Here the command processes from dump5000000.part, dump5100000.part, dump5200000.part until dump11000000.part, with an increment of 100000 in file name.

```
bash preprocess.sh 5000000 100000 11000000
```

Execute the following command to get a the fine particle fraction within a sampling region.

```
python calculateFPF.py
```

The program will prompt the user to enter the start time, step size and end time, which should correspond to the file name of the dump files. For example, after preprocessing the dump files in the previous command, the start time would be 5000000, step size 100000, end time 11000000. The program also prompts the user to enter parameters that defines a prism-shaped sampling region, as shown in the following example illustrated by Figure 19.

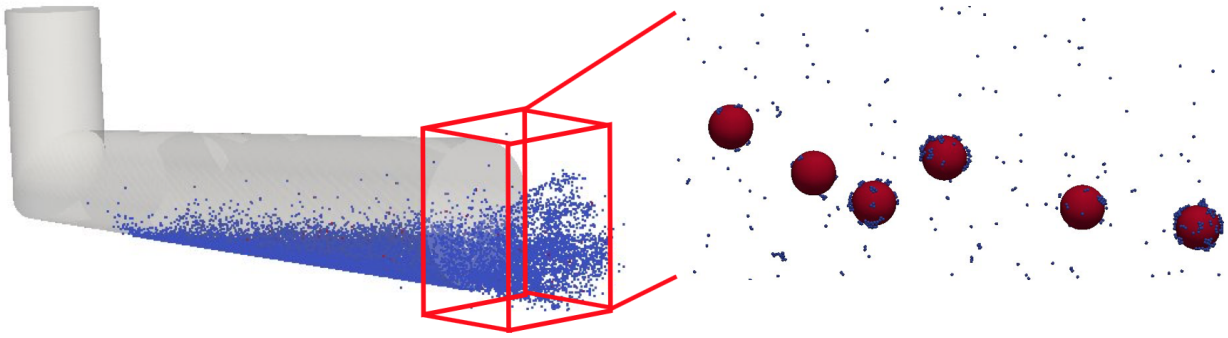


Figure 19: Sampling region at the outlet

```
python fpf.py
Enter start time: 5000000
Enter step size: 100000
Enter end time: 11000000
Enter the sampling region
Enter xmin: -0.5
Enter xmax: 0
Enter ymin: -0.35
Enter ymax: 0.35
```

```
Enter zmin: -0.35
Enter zmax: 0.35
============================================
For dump5000000.part
Total number of carrier particles =  1337
Total number of api parcels =  17100
============================================
Processing time  5000000
Processing time  5100000
Processing time  5200000
...
Processing time  10800000
Processing time  10900000
Within the time and sampling region, fine particle fraction is  0.165420114577
```

References

[1] Freecad®: Your own 3d parametric modeler. https://www.freecadweb.org/. Accessed: 2019-08-12.

[2] Calum Douglas. Openfoam snappyhexmesh tutorial, 2014.

[3] Yile Gu, Ali Ozel, and Sankaran Sundaresan. A modified cohesion model for cfd–dem simulations of fluidization. *Powder Technology*, 296:17–28, 2016.

[4] Xiaoyu Liu, Mostafa Sulaiman, Jari Kolehmainen, Ali Ozel, and Sankaran Sundaresan. Particle-based coarse-grained approach for simulating dry powder inhaler. *International Journal of Pharmaceutics*, 606:120821, 2021.

[5] Duy Nguyen, Johan Remmelgas, Ingela Niklasson Björn, Berend van Wachem, and Kyrre Thalberg. Towards quantitative prediction of the performance of dry powder inhalers by multi-scale simulations and experiments. *International journal of pharmaceutics*, 547(1-2):31–43, 2018.

[6] Berend van Wachem, Kyrre Thalberg, Johan Remmelgas, and Ingela Niklasson-Björn. Simulation of dry powder inhalers: Combining micro-scale, meso-scale and macro-scale modeling. *AIChE Journal*, 63(2):501–516, 2017.