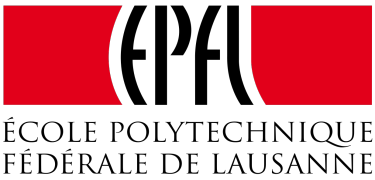
POLITECNICO DI TORINO

Master's degree course in Mathematical Engineering

Numerical simulation of the Freeman-FT4 powder rheometer

An application of the Discrete Element Method



Internal supervisor
Prof. Luigi Preziosi

External supervisor:
Dr. Mark Sawley from EPFL

Candidate
Ombretta Bistoni
Student ID: 240021

Company Tutor
Novartis International AG

Academic year 2017 – 2018

Contents

To my family and my friends

Abstract

The Freeman-FT4 powder rheometer is a universal tool to study powder properties. It is widley used for pharmaceutical purposes, also by the Swiss company Novartis, that collaborated to this work. This thesis concerns the numerical study of the rheometer. The Discrete Element Method is used, because it is particularly indicated to treat granular materials. This method is implemented in a code called GFX, written in Fortran90 by Dr. Mark Sawley from EPFL. Several simulations are done and compared to the experimental results fournished by Novartis. During the first simulations the impact of the particles' number and the cohesion between particles is assessed. The last simulations instead, are done to investigate how the particles property parameters influence the results. Both a qualitative and quantitative post-processing is done to analyze the results obtained. The qualitative analysis consists in the visualization of the results with ParaView, an open-source tool for the visualization of scientific data.
The thesis is structured as follows: in the first two chapters there are the descriptions of the method and code used; in the third part there is the presentation of the case study, the FT4 powder rheometer; the last part is the analysis and discussion of the results obtained.

Acknowledgements

I would like to thank all the people that supported me during these years. It was hard to leave my family, but I met some friends that made me feel like I was at home, firstly in Turin and then in Lausanne. With the constant love of people sourrinding me I was able to reach my goals, and I will be greatful forever for this. A special thank to the people of my family and my oldest friends who travelled 700 km from Rome, just to stand by my side during this day. I cannot say how lucky I am for this. This thesis is dedicated to my family: to my mum Tiziana, my dad Luciano and my brother Oliviero. I tried to do my best to make you proud of me.

Introduction

In recent years there has been an increase in the application of the Discrete Element Method (DEM), originated by *Cundall and Starck* [5], to study properties such as flowability of granular media and powders [36]. Flowability is the ability to flow under specific conditions and can influence industrial processes including powder feeding and mixing, as well as product quality [36]. Pharmaceutical industries are particularly interested in investgating powder's properties such as flowability, since they can affect the raw material of which drugs are composed.
The case-study presented here is indeed a pharmaceutical application. The theme of this theis is in fact the numerical study of the Freeman-FT4 powder rheometer owned by the Swiss pharmaceutical company Novartis. The powder rheometer is a commonly used device for assessing the bulk flow properties of pharmaceutical powders [37].
A numerical study can be useful to anticipate or validate the results obtained by experiments. Nevertheless, this work is not finalized to commercial purposes and an immidiate application, but wants to lay the foundations for a further study.
Since powders can be considered as sets of small particles, they can not be treated as a continuum, and therefore a particle-based method is needed to study them.

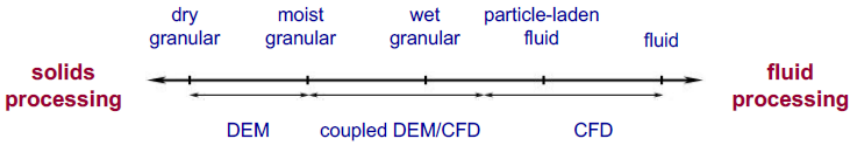


Figure 1: Discrete Element Method (DEM) and Computational Fluid Dynamics (CFD) are used in different scale processings, but can be also coupled. Image source: [2]

DEM simulations are well suited for this type of studies as they can isolate the impact of each particle property on the bulk powder behaviour [37]. DEM is based on a Lagrangian approach: the reference frame moves with the flow and is not fixed. Here we can understand the revolutional point of view of this method: each

numerical particle corresponds to a real particle, with no need of numerical meshes. The particles' motion is described by the second Newton's laws of motion, that are quite easy to implement and are approximated with an explicit first order scheme. The first chapter of this thesis is dedicated to the presentation of the method used, while the second is the description of the code used. The numerical implementation is done with GFX, written in Fortran90 by the supervisor Mark Sawley from École polytechnique fédérale de Lausanne (EPFL). A LSD (linear-spring-dashpot) model is taken into account: the contact forces between particles are made of an elastic and a viscous force. These forces act when two particles collide. Here a soft-particle model is used and therefore the particles are allowed to slightly overlap, but also a hard-particle model where this does not happen exists, even if it is less common. The importance of the contact detection is then clear. A fine search algorithm is used in GFX, but more complicated and efficient algorithms exist in literature (see [1]). Other forces can be added to the LSD model, as an elastic cohesion force between particles.
In the third chapter there is the description of the rheometer and its main features. It is composed by a cylindrical vessel, and a blade that moves downwards and upwards through the powder bed. During these movements this instrument is able to register the force and the torque applied on the blade for each small distance travelled. It is possible then to calculate the total energy, made of the sum of two contributions: the force energy and the torque energy. The calculations of these terms are present as well in the third chapter, together with the geometrical description of the experimental setup. From all the possible experiments that can be done with the FT4 powder rheometer, the following are studied and reproduced: the confined flow test with the measure of the Basic Flowability Energy (BFE), and the unconfined flow test with the measure of the Specific Energy (SE), both dynamic tests. The first test is done with the blade of the rheometer going downwards and forcing the powder to flow. The BFE is therefore a measure of how easly the powder flows when it is forced to by the movement of the blade. The higher the BFE is, the more the powder is resistent to this motion. On the contrary, the unconfined test is done with the blade going upwards. In this case the SE is a measure of the powder's flowability when unconfined, such as during low stress filling, or low shear blending [24].
The last chapter is instead dedicated to the simulations done, together with the discussion of the results. Just one simulation represents the unconfined flow test, since no experimental data about this test were fournished. As it concern the confined flow test, several simulations are done. The first concerns 5977 particles with a radius varying from $0.7mm$ to $0.9mm$, with the blade going downwards and forcing the powder to flow. A comparison with the experimental data fournished by Novartis is done to compare the force, the torque and the energy, together with the value of the BFE. In order to have results more similar to the experimental ones, some changes are made to the model. Firstly, a simulation with more particles is

carried out, exactly with 8 times the previous number, since particles with radii between $0.35mm$ and $0.45mm$ are considered. We will see then the differences that more particles bring to the model.
Since the real powders are cohesive, an elastic cohesion force is added to the model during other simulations, in order to make them more realistic. Three simulations varying the spring cohesion constant of the cohesive force are therefore carried out. The results will be again compared to the experimental ones.
The last simulations interest instead the particle property parameters present in particles' interaction equations. The Coloumb friction limit, the rolling friction coefficient and the coefficient of restitution are changed in order to understan how they influence powder's behaviour. In particular powder's flowability, measured through the BFE value, is assessed.
Together with this quantitative study, a qualitative post-processing is done with the help of ParaView, an open-source application for scientific visualization. This is not only a way to visualize the results obtained, but it is also useful to immediately identify mistakes concerning the simulations. An example of the visualization is given here:

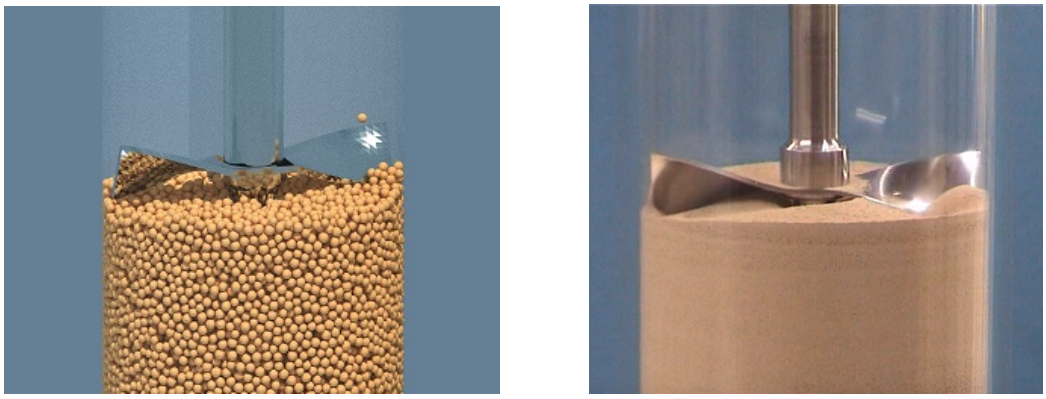


Figure 2: Comparison between a realisitc visualization of the rehometer done with ParaView (left), and a real image (right) taken from [27].

This is a realistic visualization, while the scientific one is present in the last chapter, together with the quantitative post-processing.

Chapter 1

Discrete Element Method

This chapter presents the main features of the Discrete Elements Method (DEM). This method is designed to treat granular material and it is based on the principle that our world is made of particles. Therefore it is widely used in various applications such as food and agriculture, mineral processing, chemical, pharmaceutical industries [2].

1.1 Mesh-less method

The main feature of the Discrete Element Method is that it is implemented without any generation of mesh. The main purpose of a mesh-based procedure is to divide the problem domain into small computational grid cells. Examples of mesh-based methods are the finite difference method, finite element method, finite volume method. Even if not always, generally the mesh-based methods are used with a Eulerian approach, while the particle-based methods (such as DEM) deal more with a Lagrangian approach. The Eulerian description considers a frame that is fixed with respect to the fluid, while the Lagrangian description considers a frame that moves with the fluid:

This underlines the fact that for the method considered here the physical quantities are defined on mobile discrete particles and not at the node or cell centers of a generally fixed computational mesh. The result of this is that the theory behind DEM is semplified with respect to the one used for a mesh-based method, since the numerical particles correspond to real particles.

1.1.1 Advantages and disadvantages

A comparison between the mesh-generated methods and the mesh-free methods is necessary to choose what approach should be used. The discratization of a complex geometry with the generation of a mesh could be largely time-consuming, while it

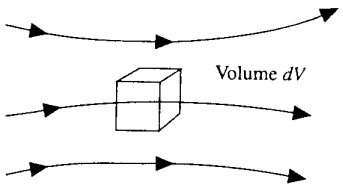


Figure 1.1: With the Eulerian approach the body frame is fixed

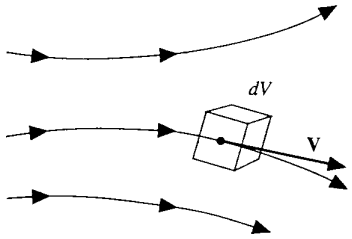


Figure 1.2: With the Lagrangian approach the body frame moves

Figure 1.3: Image source: *Computational Fluid Dynamics* by John Anderson

is easly treated without a mesh. Furthermore the mesh-based approach is not always appropriate for discrete systems. Nevertheless the particle-based methods' validation lacks in many application areas, it is computationally intensive and it is less available in open-source and commercial codes.
As stated by *Nouruzi et al.* in [1] DEM, with its Lagrangian approach, leads to more accurate results if used to study problems that cannot be trated as a continuum. Therefore, the Discrete Element Method was chosen here. Another reason to prefer it in this case is its simplicity to deal with a great number of particles. Since the field quantities are defined for each discrete point in space, the general behaviour is deduced by the average (statistical) quantities.

1.2 The procedure

Here a list of the main steps followed in the solution procedure is reported (see [2]):

1. representation of the domain with particles, that initially contain the boundary conditions
2. discretization of the governing equations
3. determination of neighboring particles in the influence domain
4. calculation of forces exerted on each particle at time t
5. calculation of the new position of particles after a timestep Δt
6. calculation of physical quantities at the time $t + \Delta t$
7. repetition of steps 3-6.

After the initialization of the numerical domain, there is the tracking of position, velocity and spin for each particle. Then the collisions between neighboring particles

are considered in the physical model. Finally, the equations of motion are solved for each particle.

1.3 Physical modelling

The particles have both linear and angular motion and they are allowed to interact following two models: the hard-particle model and the soft-particle model.The description of these models is done following [1].

1.3.1 Hard-particle model

The hard-sphere formulation is based on instantaneus collisions between non-deformable particles. The bodies are allowed to move freely without the possibility of multiple contacts. The velocity and position of the particles before collisions are determined by integration of the Newton's second law of motion:

$$m_i \frac{d\mathbf{v}_i}{dt} = m_i \mathbf{g} + \mathbf{F}_i^{f-p} \tag{1.1}$$

The velocity of the particle i changes thanks to the external forces: the gravitational force and the total fluid-particle interaction force (right-hand side of the equation). This is a generic formulation that takes into account the interaction between the particles and the fluid (multi-phase flows). Some of these forces can be buoyancy, drag and lift. Once that translational and rotational velocities of each particle are calculated, it is possible to determine the velocities after the collisions thanks to the impulse equations:

$$m_i \mathbf{v}_i^1 = m_i \mathbf{v}_i^0 + \mathbf{J} \tag{1.2}$$

$$I_i \boldsymbol{\omega}_i^1 = I_i \boldsymbol{\omega}_i^0 + R_i \mathbf{n}_{ij} \times \mathbf{J} \tag{1.3}$$

Where superscripts 0 and 1 are used for translational and rotational velocities before and after collision, R_i is the radius of the particle i, $\mathbf{n}_{ij}$ is the unit normal perpendicular to the contact plane, $\mathbf{J}$ is the impulse force and I_i is the moment of inertia of particle i: $I_i = \frac{2}{5} m_i R_i^2$. These equations refer to the particle i but the ones for particle j are equal.

All the terms present in (1.2) and (1.3) can be calculated, but since this model is not used in the case-study presented here, the interested reader is referred to [1] for a more detailed description.

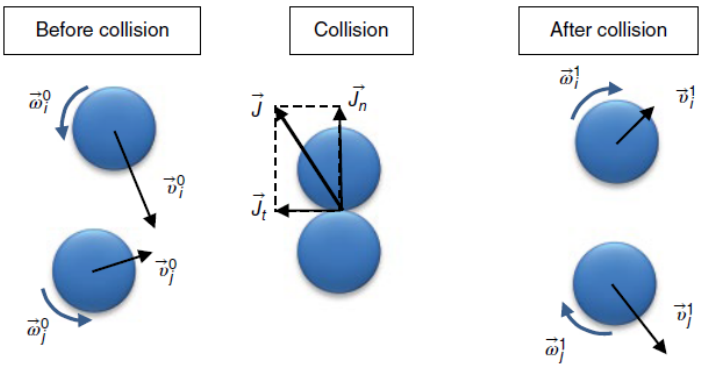


Figure 1.4: Schematic representation of a collision between particles in the hard-sphere scheme. Image source: [1]

1.3.2 Soft-particle model

The soft-sphere formulation is based on continous collisions between deformable particles. During collisions, the particles overlap and after they retain their original shapes, but the maximum overlap must be small if compared to the particles sizes. Multiple collisions are allowed for this model. Since an explicit time integration scheme is used, the time step should be chosen small, such that each particle is influenced only by its sourranding particles. An advantage of using an explicit time scheme for the numerical implementation is that both linear and nonlinear contact force models can be used. The governing equations in this case are the Newton's second law of motion for the center of mass and the Euler's second law for the angular momentum:

$$m_i \frac{d\mathbf{v}_i}{dt} = m_i \frac{d^2\mathbf{x}_i}{dt^2} = \mathbf{F}_i \tag{1.4}$$

$$I_i \frac{d\boldsymbol{\omega}_i}{dt} = I_i \frac{d^2\boldsymbol{\varphi}_i}{dt^2} = \mathbf{T}_i \tag{1.5}$$

where $\mathbf{F}_i$ and $\mathbf{T}_i$ are respectively the total force and torque exerted on particle i, and $\boldsymbol{\varphi}_i$ is the angular position. Moreover, $\mathbf{x}_i$, $\mathbf{v}_i$, m_i, $\boldsymbol{\omega}_i$ and I_i are respectively the position, velocity, mass, angular velocity and moment of inertia of particle i. For each time step, it is possible to calculate the interaction forces and by integration of the equations the new state of the system. The total force and torque are given by:

$$\mathbf{F}_i = \sum_{j \in CL_i} \mathbf{F}_{ij}^{p-p} + \mathbf{F}_i^{ext} \tag{1.6}$$

$$\mathbf{T}_i = \sum_{j \in CL_i} \left(\mathbf{T}_{ij}^t + \mathbf{T}_{ij}^r \right) \tag{1.7}$$

where CL_i represents the list of the particles in contact with the particle i. The force terms are respectively the particle-particle interaction forces and the external forces (as the gravitational force). The torque terms are respectively the tangential torque produced by particle-particle collision and the torque due to rolling resistance.
From now on, the soft-particle model is adopted.

1.4 Contact detection

In the previous section the list of neighbouring particles is mentioned. There are several algorithms to determine the list of particles that interact with a certain particle. A possibility is to search for all possible pair-wise interactions, but this would lead to an algorithm with $\mathcal{O}(n^2)$ operations, if n is the number of particles. Therefore the CPU time would rapidly increase for systems with a large quantity of particles. A two-step search procedure is then employed. During the first time step a near neighbour list is constructed, with all the particles below a certain distance from the particle considered. Then, during the second time step, only the particles in this list are checked for possible contacts. If a fine mesh technique for spherical particles is used, then two particles i and j collide if:

$$overlap = R_i + R_j - |\mathbf{x}_i - \mathbf{x}_j| > 0 \tag{1.8}$$

where R_i and R_j are the radii of the two particles and $\mathbf{x}_i$, $\mathbf{x}_j$ thier positions. In the literature many other algorithms for contact detection can be found, in order to study more complex situations like the precence of non-spherical particles, or to increase the computational efficency, see [1, 3, 4]. Nevertheless for the case-study presented here a fine mesh technique is an efficient compromise between simplicity of implementation and computational cost [6].

1.5 Force-displacement law

As already said, the collision of two particles happens when the overlap between them is major then zero:

The normal overlap is defined as:

$$\delta_n = R_i + R_j - |\mathbf{x}_i - \mathbf{x}_j| \tag{1.9}$$

The collision forces are divided into their normal and tangential components, and the normal and tangential vectors are defined as:

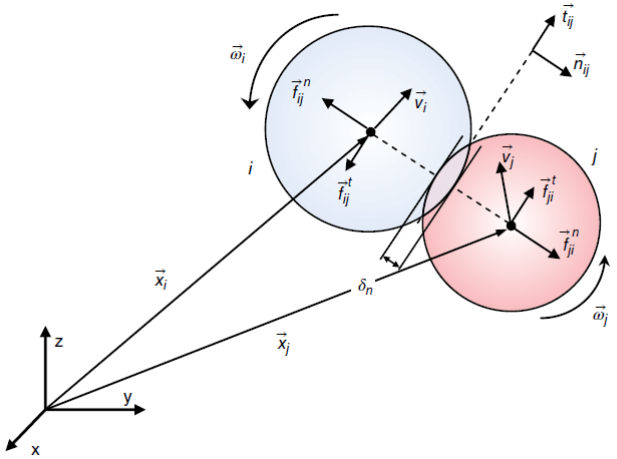


Figure 1.5: Two colliding particles and the forces involved, for the soft-particle model. Image source: [1]

$$\mathbf{n}_{ij} = \frac{\mathbf{x}_j - \mathbf{x}_i}{|\mathbf{x}_j - \mathbf{x}_i|} \qquad \mathbf{t}_{ij} = \frac{\mathbf{v}_{ij}^t}{|\mathbf{v}_{ij}^t|} \tag{1.10}$$

Where $\mathbf{v}_{ij}^t$ is the tangential component of the velocity at the contact point, defined as $\mathbf{v}_{ij}^t = \mathbf{v}_{ij} - \mathbf{v}_{ij}^n$, with:

$$\mathbf{v}_{ij} = \mathbf{v}_i - \mathbf{v}_j + (R_i\boldsymbol{\omega}_i + R_j\boldsymbol{\omega}_j) \times \mathbf{n}_{ij} \tag{1.11}$$

$$\mathbf{v}_{ij}^n = (\mathbf{v}_{ij} \cdot \mathbf{n}_{ij})\mathbf{n}_{ij} \tag{1.12}$$

All the tools necessary to present one of the most common viscoelastic force-displacement law have been mentioned.

1.5.1 LSD model

The linear spring-dashpot (LSD) model was first introduced by Cundall and Starck [5]. It is used to model the total force acting on a particle, given here by $\mathbf{F}_{ij}^T = \mathbf{F}_{ij} + \mathbf{F}_{ij}^c + \mathbf{F}_g^i$, where $\mathbf{F}_{ij}$ is the contact force and $\mathbf{F}_{ij}^c$ is the cohesion force. These components copose the particle-particle interaction forces previously mentioned. $\mathbf{F}_g^i = m_i\mathbf{g}$ is the gravitational force acting on a particle of mass m_i. As well as the total force, this model is used to write the total torque: $\mathbf{T}_{ij}^T = \mathbf{T}_{ij}^t + \mathbf{T}_{ij}^r$, where the previous terms represent respectively the torque due to the tangential contact force and the torque due to rolling resistence.

Contact force

The normal component of the collision force between two particles is made of two terms: an elastic force given by Hook's law and a viscous force. The equation is then:

$$\mathbf{F}_{ij}^n = \mathbf{F}_{el}^n + \mathbf{F}_{diss}^n = -(k_n \delta_n)\mathbf{n}_{ij} - (C_n \mathbf{v}_{rn})\mathbf{n}_{ij} \tag{1.13}$$

where k_n is the normal spring stiffness, C_n is the normal damping coefficient and $\mathbf{v}_{rn} = \mathbf{v}_{ij} \cdot \mathbf{n}_{ij}$ is the relative velocity in normal direction.

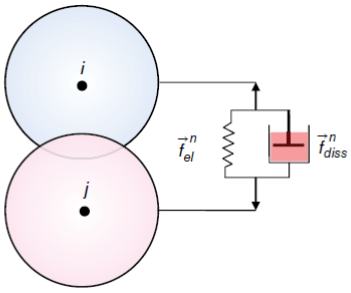


Figure 1.6: A viscoelastic collision model in normal direction. Image source: [1]

The collision time during which the particles remain in contact can be determined by [6]:

$$t_{col} = -\frac{ln(\epsilon)}{\gamma}\sqrt{\frac{m_{red}}{k_n}} \tag{1.14}$$

where ϵ is the normal coefficient of restitution, so the fraction between the relative normal velocity before collision and the relative normal velocity after collision. γ is the coefficient of critical damping, given by:

$$\gamma = -\frac{ln(\epsilon)}{\sqrt{\pi^2 + ln(\epsilon)^2}} \tag{1.15}$$

and m_{red} is the reduced mass: $m_{red} = \frac{m_i m_j}{m_i + m_j}$. The normal damping coefficient is related to the previous mentioned quantities through the following relation:

$$C_n = 2\gamma\sqrt{m_{red} k_n} \tag{1.16}$$

For a real material the normal spring constant k_n is very large, producing a very small collision time. Since very small integration times are not practical for numerical simulations [6], a suitable k_n is chosen, such that the collision time remains of the order of μs, and the normal overlap δ_n doesn't exceed 1% of the particle diameter. We will see that sometimes this overlap limit is not respected during the

simulations, but it will always be in an acceptable range. In [1] it is shown that a non-zero force at the start and at the end of the collision is produced, due to the presence in this model of the viscous term that dissipates the kinetic energy. Since this behaviour is physically unrealistic for non-adhering collision, other models have been taken in consideration, such as the non-linear LSD [1] and the four-parameter linear model (see [7]). We will not treat in details these other models.
The tangential contact force, as the normal component, is given by:

$$\mathbf{F}_{ij}^t = \mathbf{F}_{el}^t + \mathbf{F}_{diss}^t = -(k_t\delta_t)\mathbf{t}_{ij} - (C_t\mathbf{v}_{rt})\mathbf{t}_{ij} \tag{1.17}$$

where k_t and C_t are respectively the tangential spring constant and damping coefficient. $\mathbf{v}_{rt}$ is the relative velocity of particles in the tangential direction and δ_t is the tangential overlap, calculated by integrating the tangential velocity during the contact between particles. The expression in (1.17) is no longer valid if the Coulomb criterion doesn't apply, that is $|\mathbf{F}_{ij}^t| \geq \mu|\mathbf{F}_{ij}^n|$. In this case, the particles begin to slide over each other and the tangential contact force is given by the model used in [5]:

$$\mathbf{F}_{ij}^t = -\mu\mathbf{F}_{ij}^n sgn(\delta_t)\mathbf{t}_{ij} \tag{1.18}$$

where μ is the coefficient of dynamic friction between particles i and j. The total contact force is given by the sum of its normal and tangential components.

Cohesion force

The model of the cohesion force is taken from [6], but this force could be caused by different fenomena such as liquid bridges due to van der Waals forces and electrostatic forces (see [8, 9]). Following [6], a linear spring model similar to the one used to determine the contact force is applied. Its normal component is given by:

$$\mathbf{F}_{ij}^{c,n} = -k_{c,n}\delta_{c,n} \tag{1.19}$$

where $\delta_{c,n}$ is the cohesion overlap between the particles in the normal direction, and $k_{c,n}$ is the normal cohesion spring constant. Since the cohesion force is attractive, it acts in the opposite direction of the repulsive contact force, so $k_{c,n} < 0$. In addition, we have: $|k_{c,n}| << k_n$ while $\delta_{c,n} > \delta_n$. In particular, the cohesion overlap between two particles of radii R_i and R_j, is given by:

$$\delta_{c,n} = \delta_n + \Delta\delta_{c,n} = \delta_n + (\sigma_{c0,i} + \sigma_{c1,i}R_i) + (\sigma_{c0,j} + \sigma_{c1,j}R_j) \tag{1.20}$$

where $\sigma_{c0,i}$,$\sigma_{c1,i}$, $\sigma_{c0,j}$ and $\sigma_{c1,j}$ are constants defining respectively the cohesive zone surrunding each of particles i and j [6].

Torque

As already said, the total torque is made of two terms. As done in [1], the torque due to the tangential component of the contact force is written as

$$\mathbf{T}_{ij}^{t} = R_i \mathbf{n}_{ij} \times \mathbf{F}_{ij} \tag{1.21}$$

This term is the cause of the rotation of the particles, while the second term, that is the rolling resistance torque, opposes to this rotation. To be more clear, the rolling resistance torque is the reason why a sphere rolling on a flat surface gradually decreases its speed until stopping. The rolling resistance torque is formulated by [10]:

$$\mathbf{T}_{ij}^{r} = -\mu_r R_{red} |\mathbf{F}_{ij}^{n}| \hat{\boldsymbol{\omega}}_{ij} \tag{1.22}$$

where μ_r is the coefficient of rolling resistance, $R_{red} = \frac{R_i R_j}{R_i + R_j}$ is the reduced radius, and $\hat{\boldsymbol{\omega}}_{ij}$ is the relative angular velocity of particles:

$$\hat{\boldsymbol{\omega}}_{ij} = \frac{\boldsymbol{\omega}_i - \boldsymbol{\omega}_j}{|\boldsymbol{\omega}_i - \boldsymbol{\omega}_j|} \tag{1.23}$$

As stated in [1], in this model, a constant resistant torque is applied on colliding particles. The minus sign is used to emphasize that this torque opposes the relative rotation of the particles, as we already said. Note that for a spherical particle rolling on a plate ($R_j \to \infty$), $\mathbf{T}_{ij}^{r}$ is proportional to the particle radius R_i.

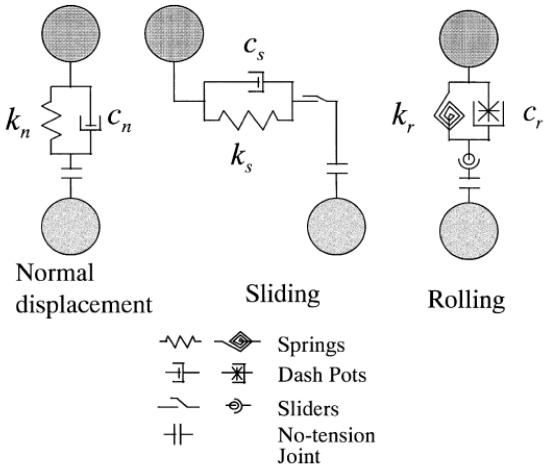


Figure 1.7: Inter-particle contact forces and torques for LSD model. Image source: [13]

Chapter 2

Code used: GFX

In order to transform the DEM theory into numerical results, we have used the code called GFX. It is written in Fortran90 and it is mostly used to deal with simulations of particulate processes.
This chapter is mainley the description of the code, as reported in [6]. In [11] more applications of GFX are studied, such as those in the chemical, food and agricolture, mining and mineral processing industries. We will not write about all the potential capabilities of the code, but only about the features that we used during the simulation procedure.
In GFX, the particle-particle and particle-boundary interactions are detected using a two-step process: spatial sorting to provide a near neighbour list of potential interactions, followed by explicit determination of each member in the list. Referring to Section 1.3.2, we use the soft-particle model to calculate the normal and tangential components of the forces. The equations of motion for the position and spin of the particles are integrated using an explicit time-stepping procedure. A variety of informations can be obtained with a GFX simulation, both qualitative and quantitative. A schematic representation of the structure of the code is reported here, as well as the numerical methods employed inside it.

2.1 Structure

In GFX it is possible to import 2D and 3D geometries, comprised of a collection of distinct objects, that are allowed to have translational and rotational motion. Particles with different sizes and density distributions can be created, and material properties are given to all the objects. Particles can be coloured according to different criteria, such as particles radii, heights, materials, groups. Different output can be exported after the simulation, including visualization files, particle-based quantities, statistical analysis. All these capabilities occur during three different phases that can be summarized with the following scheme:

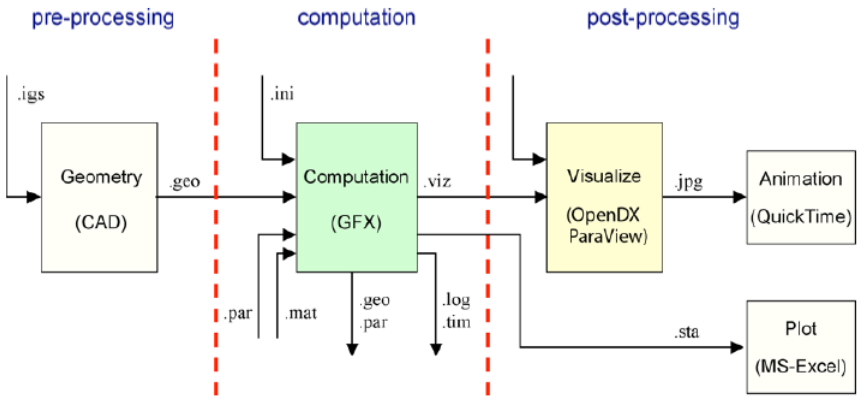


Figure 2.1: Global schematic diagram of GFX functioning. Image source: [6]

2.1.1 Pre-processing

The pre-processing phase consists of two steps: the geometry creation and the surface mesh generation. Both these steps can be done with the GAMBIT graphical user interface (GUI). Nevertheless, it is not possible to import directly the formats exported by the pre-processing software, so conversion into the GFX *geometry* format is necessary. In principle, any reasonable geometry can be used in GFX. The geometry is usually defined as a number of objects. Using CAD software, the geometry can either be created directly or be imported from a standard geometry file. As it concern the surface meshing, each geometry object is decomposed into a number of elements that can be either triangular and/or quadrilateral. Is is generally computationally more efficient for GFX to consider a number of elements not too high. Therefore a coarse surface mesh containing quadrilateral elements is often used. We underline the fact that this is just a 2D-boundary mesh, and that we do not mesh the whole 3D domain. This is the reason why DEM is called a mesh-less method. We refer to [6] for a more datailed description.

2.1.2 Computation

To perform the numerical simulation, GFX needs to know about the geometry, the physcal properties and the numerical parameters. All these informations are provided with input files.

Input files

The parameters for each input file are grouped into sub-sections.

- **Control.in**
  Define. Here the physical parameters are chosen. Firstly the contact model,

that is the spring-dashpot, and then the activation of particle spin, rolling friction, particle interaction and where needed cohesive forces.
Initialize. Here one can either define the geometrical limits of the bounding box, or read them from the geometry file. The bounding box is referred here to the simulation domain. Similarly, also the particle initialization can either done here or read from the particle file. Finally, the particles can be devided in groups here.
Solve. Informations about the time duration of the simulation, the initial integration time step and the algorithm used to update near neighbour list are mentioned here.
Export parameters. As the title suggests, here there are the parameters that we will find in the output files: the particle radius, colour, velocity components, spin magnitude, normal force, the temporal statistical quantities such as the total force and power.
An example of a *Control.in* file can be found in Appendix A.1.

- **Geometry.in**
  Objects. Here the geometry properties of the objects are reported. For each object a list of parameters is written: the number of elements composing the object, the type (delated, neutral, active or invisible), the material number, the motion type (translational + rotational in our case) and the name.
  Elements. For each element we have: its number, the number of the object that contain the element, the element type (3=triangular, 4=quadrilateral), the number of nodes that constitue the element (4 numbers for a quadrilateral element).
  Nodes. For each node, here there are its coordinates (x, y, z).
  Geometrical. Geometrical limits of simulation domain, and status of control volume (0=fixed throughout the simulation, 1=updated).

- **Particle.in**
  Particles. For each particle there are: coordinates of position, velocity and spin, radius, material, type (1=solid sphere), colour, cluster.

- **Material.in**
  Materials. For each material, the corresponding parameters are listed: the mass density, the normal spring constant, the ratio of tangential to normal spring constants, the ratio of tangential to normal damping constants, the Coulomb frictional limit, the coefficient of restitution, the coefficient of rolling resistance, the normal spring constant for cohesive interaction, the maximum distance between surfaces over which cohesive forces act, the maximum distance between surfaces over which cohesive forces act expressed as a ratio of

the particle radius and the coefficient of rolling resistance for cohesive interaction.
An example of a *Material.in* file can be found in Appendix A.2.

Output files

The output files of a simulation with GFX are: the visualization files, the geometry and particle files, the log file containing the selected information throughout the simulation, the timing file containing the wall-clock time, the statistics files containing averaged values of different quantities. The geometry and particle output files have the same format as the corresponding input files. They can therefore be used to restart the simulation from a previously computed solution.
The output files are organized as follow.

- **Standard output file**.
  An example of the standard output file can be found in Appendix B.1.

- **Ensight 6 file**.
  The Ensight 6 format consists of a case file and the data files. The case file is written in ascii (formatted) format. An example of the case file is in Appendix B.2. The data files are organized as follow:

  Geometry file
  geometry_000i.geo refers to the timestep number i and contains for each node, its coordinates and for each element, its nodes.

  Particle file
  particle_000i.pos, *particle_000i.rad*, *particle_000i.col*, *particle_000i.spn*, *particle_000i.vel*, *particle_000i.frn* refer to the timestep number i and contain for each particle respectively the coordinates of its position, the radius, the colour, the spin, the components of its velocity and the normal component of the total force acting on it.

- **Log file**
  It is a *.dat* file in which the input values are reported in the following order: general, particles, boundary objects, physical modelling, material properties, numerical parameters. Other interesting output that can be found here are, for each time step, the number of particles, the average overlap between particles, the translational and rotational velocity and the particle energy. These last parameters can be used to verify if a simulation has worked well, with physically meaningful values.

- **Timing file**
  Here we can find all the relevant informations about the timing. The total elapsed wall-clock time is devided into the time spent for initialization, computation, post-processing, termination. A list of all computational steps is present here, with the time spent for each step.

- **Statistics file**
  Only the temporal file is taken into account. It is a *.dat* file in which for each time step the three components and the magnitude of the total force, as well as the power are listed.

2.1.3 Post-processing

As mentioned in [6], during the post-processing phase, the simulation data is analyzed to extract the required physical information. Both qualitative and quantitative post-processings could be done.

Qualitative

The *Ensight.case* output file form GFX is imported in ParaView. ParaView is an open-source, multi-platform application for the visualization of scientific data [12]. Several features of this application have been investigated, in order to make the scientific visualization of the simulations done. The version used is ParaView 4.4.0, together with ParaView 5.5.2 for a particular realistic visualization that couldn't have be done with the previous version (the one present in the Introduction). We refer directly to Chapter 4 for more informations about the qualitative results obtained.

Quantitative

The statistcs files exported by GFX provide global information on the simulation properties. As already said, the *log.dat* file has been used to check the validity of the simulation. The files are in simple tabular format and can therfore be easily processed with Microsoft Excel and Matlab R2017b. The *temporal.dat* files have been instead used to compare the simulations data with the experimental results. Also for this part we refer to Chapter 4.

2.2 Numerical method

We would like to better understand how GFX deal with the numerics of our model. We recall Subsection 1.3.2 and Section 1.4 to see how the equations have been integrated in the code and what fine search algorithm has been used.

2.2.1 Integration of equations

The equations of motion for particle i expressing the conservation of linear and angular motion are:

$$m_i \frac{d\mathbf{v}_i}{dt} = m_i \frac{d^2\mathbf{x}_i}{dt^2} = \mathbf{F}_i \tag{2.1}$$

$$I_i \frac{d\boldsymbol{\omega}_i}{dt} = I_i \frac{d^2\boldsymbol{\varphi}_i}{dt^2} = \mathbf{T}_i \tag{2.2}$$

where $\mathbf{x}_i$ is the position of particle i, $\boldsymbol{\omega}_i$ its angular velocity and I_i its moment of inertia. The above equations of motion are integrated in GFX using an explicit first-order scheme: the forward Euler differencing scheme. For the sake of simplicity we omit the i index in the equation (2.1). The second equation of motion is approximated with the same scheme, thus we do not explicitly write it.
We suppose to know the position $\mathbf{x}(t_k)$ and the velocity $\mathbf{v}(t_k)$ of a certain particle at time t_k. From (2.1), we have:

$$\mathbf{a} = \frac{d\mathbf{v}}{dt} = \frac{\mathbf{v}(t_{k+1}) - \mathbf{v}(t_k)}{h} + \mathcal{O}(h) = \frac{\mathbf{F}}{m} + \mathcal{O}(h) \tag{2.3}$$

Where $t_{k+1} = t_k + h$ and h is the time step. We have indicated with $\mathbf{F} = \mathbf{F}(\mathbf{x}(t_k), \mathbf{v}(t_k), t_k)$. At the istant later t_{k+1} we have:

$$\mathbf{v}(t_{k+1}) = \mathbf{v}(t_k) + \frac{h}{m}\mathbf{F} + \mathcal{O}(h^2) \approx \mathbf{v}(t_k) + \frac{h}{m}\mathbf{F} \tag{2.4}$$

To find the equation for the position, we consider the Taylor expansion truncated at the second order:

$$\mathbf{x}(t_{k+1}) = \mathbf{x}(t_k) + (t_{k+1} - t_k)\left(\frac{d\mathbf{x}}{dt}\right)_{t_k} + \frac{1}{2}(t_{k+1} - t_k)^2\left(\frac{d^2\mathbf{x}}{dt^2}\right)_{t_k} + \mathcal{O}(h^3) \approx$$

$$\approx \mathbf{x}(t_k) + h\mathbf{v}(t_k) + \frac{1}{2}h^2\frac{\mathbf{F}}{m} \tag{2.5}$$

Since we are expressing position and velocity at time t_{k+1} in terms of the physical quantities at the previous time, this is called an *explicit* method. Reporting what have been said in [14], the approximation of $\mathbf{v}(t_{k+1})$ is $\mathcal{O}(h^2)$, so each step induces quadratic error. This is the *localized truncation error*, that is caused by the troncation of the Taylor series at the first order. If we integrate in time with $\mathcal{O}\left(\frac{1}{h}\right)$ steps, then the total error behaves as $\mathcal{O}(h)$. This estimate represents *global truncation error*, and thus the forward Euler scheme is called "first-order accurate". The relation written in (2.5) is exact for an infinitely short time period, and approximately true for a short time period. Therefore, to have an enough good approximation of the

solution, a carefully choice of the time step must be done. To conclude, we observe that we need initial conditions to find a solution. If we have the initial values of position and velocity at $t = 0$, we can then step through time and generate the entire trajectory of each particle. The initialization of particles is done in GFX with a preliminary simulation. Particles are created randomly by GFX on a fictive Cartesian mesh above the geometry. A small downard velocity is given to those particles until they fill the geometrical domain. Then the *particle_000i.dat* output is copied and used as an input file for the real simulation. This is possible since the GFX input and output files have the same format. In particular, the mentioned file will contain the particles initial coordinates, while the initial velocities are set to zero. For the visualization of the initialization test, we refer to Chapter 4.

2.2.2 Contact detection

As already mentioned in Section 1.4, a neighboring cell algorithm is implemented in GFX, since a control of all the possible interactions between particles would lead to a quadratic procedure in terms of number of particles, and therfore to a high time and CPU consuming.
The fine search algorithm starts with the division of the workspace into a grid of cells. Then two steps are followed: the particles inside a particle cell and inside the neighbor cells, are inserted in a list; then only particles inside this list are checked for possible contact. If we consider sphere particles, then a contact occur when

$$overlap = R_i + R_j - |\mathbf{x}_i - \mathbf{x}_j| > 0. \tag{2.6}$$

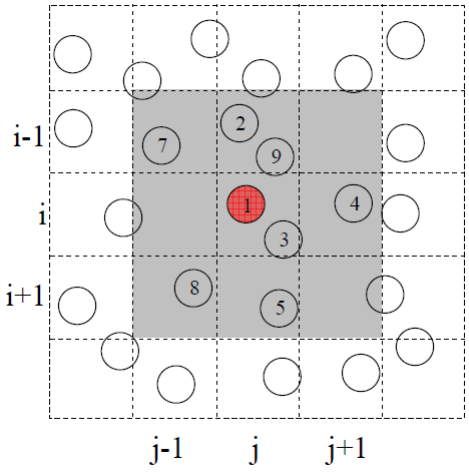


Figure 2.2: 2D representation of grid cells used for contact detection. Image source: [15]

In this case, the grey cells represent the neighbor cells of particle 1. In the three-dimensional case, the granular media is divided into cubic cells of the size not less than the diameter of the largest particle [16]. Since in our simulation we deal with spherical particles with compact packing, the easiest algorithm about contat detection is used. However, a brief review of other algorithms is done for the sake of completness.

Other algorithms

Norouzi et al. developed in [1] the NBS (no binary search) algorithm, in order to avoid identifying each contact twice. According to the NBS, contact tests should be done between the target cell and 4 neighboring cells in 2D space and 13 neighboring cells in 3D:

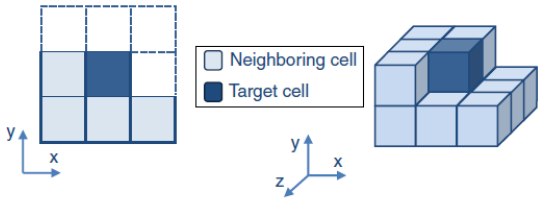


Figure 2.3: Neighboring cells and target cell for NBS algorithm in 2D and 3D. Image source: [1]

In order to map the spheres onto cubic cells, a three-dimensional array should be used. Therefore, the RAM requirement is a cubic function of simulation domain size [1]. For a small system with dense packing, where the number of particles is comparable to the number of cells, this is not a problem. However, for large systems with loose packing much of the RAM space remains useless. To avoid this, *Munjiza and Andrew* [17] and *Munjiza* [18] introduced connected lists along each axis. Instead of using a 3D array, three 1D arrays were chosen, one for each axis: X-lists, Y-lists, Z-lists. Considering a sphere in the cell (x_i, y_i, z_i), to find possible contacts five sets of Z-lists should be constructed at the same time. Three of them belong to columns x_{i-1}, x_i, x_{i+1} of the layer y_{i-1}, and two of them to columns x_{i-1} and x_i of the layer y_i. Furthermore, there should be two sets of X-lists, one for layer y_i and one for layer y_{i-1}. Looking at Figure 2.3, the cell in (x_i, y_i, z_i) is the dark blue one, and the others are the light blue ones. The RAM requirement in this case is a linear function of the simulation domain length [1].
As noticed by *Norouzi et al.* [1], the cell size in NBS and NBS-Munjiza algorithms should be at least as large as the largest sphere. Therefore, if one consider a system with varying size spheres and keep the size of the cells as the largest one, the algorithms loose their efficency. In fact, the average number of spheres in a non-empty cell would increase and there would be a lot of unusless contact checks

between spheres that are not in contact. To overcome this problem, *Peters et al.* [19] proposed a hierarchical search algorithm for this type of systems. The idea is to give a hierarchy containing several grid levels, each of which with a fixed size. Large spheres are assigned to grid level with large cell size and small spheres to grid level with small cell size. In this way, the problem of the contact search algorithm is divided into several smaller contact search problems [1]. To detect a contact, firstly the neighbor cells of the same level are checked, and then the neighbor cells of the other levels.
The algorithm mentioned until now are the so-colled *Cell-Based Algorithms*, since the simulation domain is divided into cells. There are also other type of algorithms to deal with contact detection. We refer to [20] to have an example of *Sort-Based Algorithms*, in which mapping of particles is performed by sorting arrays rather then linked lists. An example of *Tree-Based Broad Search Algorithm* is instead analyzed in [1]. The tree structure lends itself well to contact search and positioning of the particles.
We notice that we have mentioned only spheres, but algorithms thought to treat non-spherical particle can also be found in literature: [1, 21, 22, 23].

Chapter 3

FT4 Powder Rheometer

The FT4 powder rheometer of Freeman Technology is an instrument used to analyze the rheology, or flow properties, of powders.

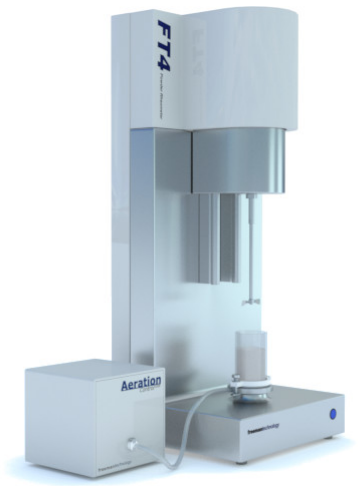


Figure 3.1: FT4 powder rheometer. Image source: [24]

We are going to mention just some of the parts that compose the rheometer: the vessel (sometimes called simply cylinder) and the blade.
This instrument measures the resistance of the powder to flow. This is achieved with a blade moving downwards and upwards the powder bed: the harder the powder resists to the blade movement, the harder it is to get the powder to flow [24]. The FT4 measures both the rotational and vertical resistances, in the form of torque and force.

Both the torque and the force contribute to the calculation of the total energy, that is then composed by two terms. There are two ways to test the powder flowability. The confined flow test is done with the blade going downwards, and

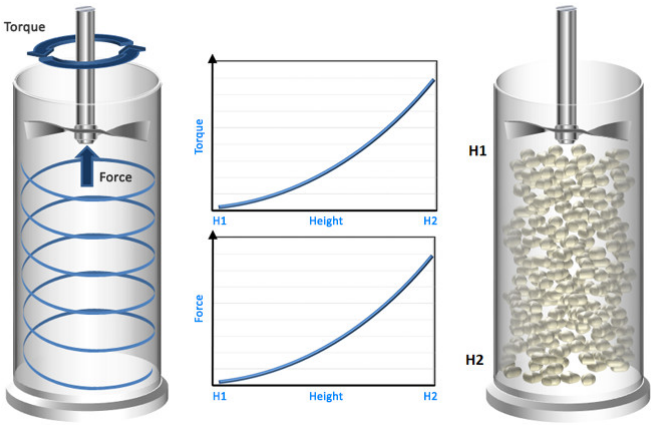


Figure 3.2: The FT4 measures torque and force while it is operating. Image source: [24]

the powder is forced to flow because it is confined by the bottom of the test vessel. The powder's flowability when forced to flow is called BFE, the Basic Flowability Energy. the second test is the unconfined flow test and it is done with the blade going up. The powder is not forced this time because the vessel is not closed at the top. The powder's flowability in this case is called SE, the Specific Energy.

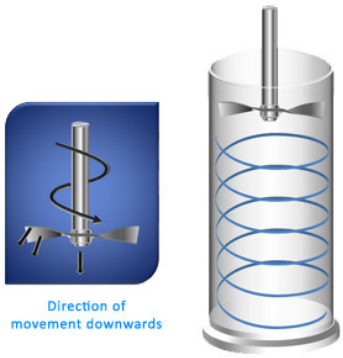


Figure 3.3: Representation of a confined flow test. Image source: [24]

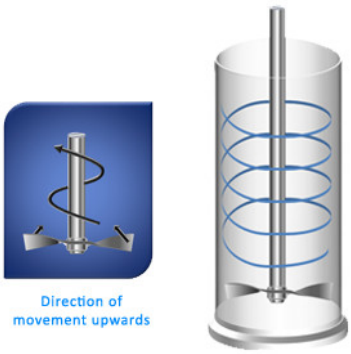
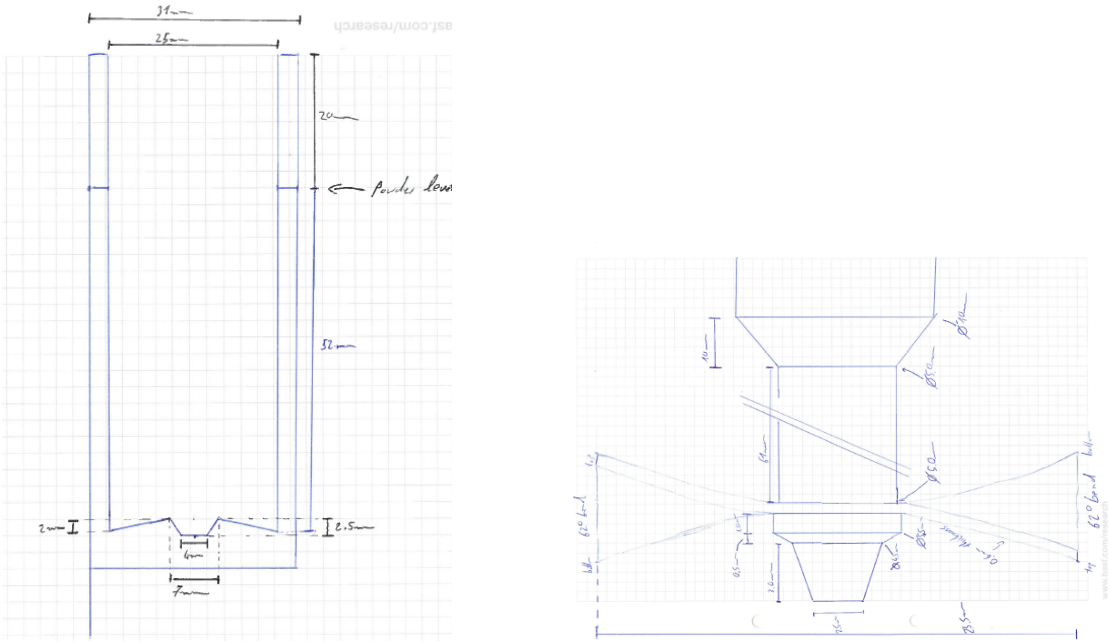


Figure 3.4: Representation of an unconfined flow test. Image source: [24]

For a description of the other functionalities of this instrument, we refer to [24]. Since the experimental data fournished by Novartis concerned the BFE, we were able to compare our simulation results with the experimental ones only in the case of a confined flow.

3.1 Experimental setup

The Freeman Technology developed powder rehometers of different size. The geometry of our case study was fournished by Novartis. The rheometer studied here is made of a helix blade with helix angle of 5° and a diameter of 23.5 mm. The cylinder is 72 mm high, with a diameter of 25 mm, and it is filled with the powder until the height of 52 mm, as reported in the Figure 3.5.



(a) Cylinder (b) Helix blade

Figure 3.5: Draws of the rheometer geometry done by Novartis.

The TipSpeed is set to 100 mm/s. With this data, it is possible to calculate the rotational speed of the blade, with this formula: $\mathbf{v} = \boldsymbol{\omega} \times \mathbf{r}$, where $\mathbf{v}$ is the TipSpeed. Taking the module of this expression, we have:

$$\omega = \frac{v}{r} = \frac{100mm/s}{11.75mm} = 8.5106382\frac{1}{s} \tag{3.1}$$

where $r = 11.75mm$ is the radius of the helix (half of the diameter).

3.1.1 Experimental calculations

The experiment done by Novartis is the confined flow test mentioned before. The aim of this test is to calculate the total flow energy required to move the blade through the sample from the top to the bottom of the powder column [24]. As

already said, we need to calculate the two terms due to the force and torque applied on the blade.

$$E_{tot} = E_F + E_T \tag{3.2}$$

Since the force and torque constantly change, we need to calculate the total energy for each small distance travelled by the blade. We call these small distances $x_0, x_1, .., x_n$ where x_0 is the first distance travelled, e.g. $x_0 = h_0 - h_1$ where h_0 and h_1 are respectively the first and second heights measured during the experiment. To clarify the procedure, we report the values:

$$h_0 = 55.05mm, \text{where the blade starts}$$

$$h_1 = 54.8mm, \text{second value registerd by the rehometer}$$

$$x_0 = h_0 - h_1 = 0.25mm$$

Therefore, for each x_i equation (3.2) becomes $E^i_{tot} = E^i_F + E^i_T$. They have calculated the force energy as follow:

$$E^i_F = \frac{F_i + F_{i+1}}{2} x_i \tag{3.3}$$

where F_i and F_{i+1} are respectively the moduli of the forces calculated by the rehometer at height h_i and h_{i+1}. The equation (3.3) comes from the definition of energy:

$$E^i_F = \int_\gamma \mathbf{F}_i \cdot d\mathbf{s} \tag{3.4}$$

where the integral is done between two consecutive heights, and is approximated by the mean value of the force. Similarly, the torque energy is given by:

$$E^i_T = \frac{|T_i + T_{i+1}|}{2} \alpha_i \tag{3.5}$$

where T_i and T_{i+1} are respectively the torques calculated by the rehometer at height h_i and h_{i+1}. The equation (3.5) comes from the definition of torque as $\mathbf{T}_i = \mathbf{r} \times \mathbf{F}_i$, that can be used in (3.4) to have:

$$E^i_T = \int_{\theta_1}^{\theta_2} \mathbf{T}_i \cdot d\boldsymbol{\theta}. \tag{3.6}$$

This is due to the fact that the infinetesimal linear displacement $d\mathbf{s}$ is related to a corresponding angular displacement $d\boldsymbol{\theta}$ and the radius vector $\mathbf{r}$ as follow: $d\mathbf{s} = \boldsymbol{\theta} \times \mathbf{r}$. Going back to equation (3.5), the infinetesimal angle travelled α_i is given by:

$$\alpha_i = \frac{cos(5°)}{sin(5°)} \frac{360}{2\pi r} x_i \tag{3.7}$$

In fact, the angular travelled in radiants is given by $\alpha_i = \frac{a}{r}$, where a is the angular way and r is the radius. To convert it into degrees: $\alpha_i = \frac{a}{r}\frac{360}{2\pi}$. Finally, to calculate the angular way: $a = x_i\frac{cos(5°)}{sin(5°)}$, where 5° is the helix angle. We report a scheme to understand this calculation:

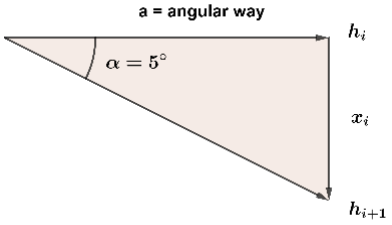


Figure 3.6: Schematic representation of the motion of the helix

Since the simulation data are taken for each infinetesimal time step instead of each infinetesimal height travelled, there is a small difference in the calculation of the angle travelled for the simulation results. Indeed, the angle travelled is given by:

$$\alpha_i = \frac{a}{r} = \frac{v\Delta t}{r}\frac{cos(5°)}{sin(5°)} = \frac{\omega r\Delta t}{r}\frac{cos(5°)}{sin(5°)} = \omega\Delta t\frac{cos(5°)}{sin(5°)} \tag{3.8}$$

where Δt is the infinetesimal time step, and ω is the magnitude of the angular velocity. We do not need the conversion into degrees here.
Going back to the quantity of interest, the Basic Flowability Energy, its value is then given by:

$$BFE = \sum_{i=0}^{n} E^i_{tot} = \sum_{i=0}^{n}(E^i_F + E^i_T) \tag{3.9}$$

where n is the number of space intervals considered.
We can also calculate the axial velocity of the blade (in modulus):

$v_n = vsin(5°)$ where v is the TipSpeed. Therefore, the axial velocity is $v_n = 100\frac{mm}{s}0.0871 = 8.7155\frac{mm}{s}$.

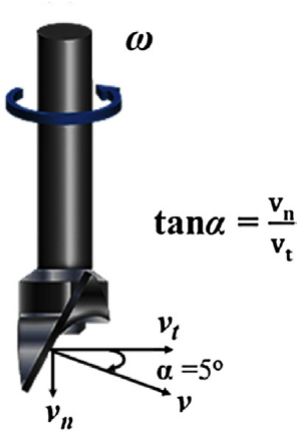


Figure 3.7: Blade image with helix angle α, tip speed v, axial velocity v_n and rotational velocity v_t. Image source: [25]

Chapter 4

Numerical results

This chapter is dedicated to the numerical results obtained during this work. They are both qualitative and quantitative. The qualitative results concern the visualization of the simulations with the use of ParaView 5.4.1. ParaView is an open-source, multi-platform scientific data analysis and visualization tool that enables analysis and visualization of extremely large datasets [26]. The *Ensight.case* file is used with this purpose. The quantitative analysis is mainly a comparison with the experimental data fournished by Novartis, and is done alanyzing the *.dat* output files. The scope is to correctly repropose the experiment done. Several simulations were done with this purpose. The first simulation is done with 5977 particles, the amount of particles needed to fill $52mm$ of the vessel. The second simulation is done with smaller particles, in particular each particles has half of the prevoius radius. To fill the same volume as before, 8 times the previous number of particles was used, so 47816. Both these tests are done with the blade going downwards, so they are confined flow tests. The third simulation, on the contrary, is an unconfined flow test, with the blade going upwards and with 5977 particles. In other three simulations a choesive force between the particles is inserted in the model. Three different choesive constants have been tested. Finally, the last simulations concern the study of some particle parameters, but we refer to Section 4.2 for more details.
The simulations are done with the ACCES computer service of EPFL, and we report here the main features about it, together with some parameters concerning the first two simulations:

We can see that the simulation with 5977 particles lasted about 10 hours, in comparison with the simulation with 47816 particles that lasted instead about 95 hours, so almost 4 days. This is the reason why for the other simulations 5977 particles have been chosen.
For the material properties we refer to Appendix A, where a list of the parameters used for the particles, the blade and the cylinder is present.

Parameters used	**ACCES server details**
Particle radius: $r_{min} = 0.7mm$, $r_{max} = 0.9mm$	accespc2.epfl.ch
Number of particles: $n = 5977$	Processors CPU: 2 quad-core Intel Xeon E5-2643 (3.3 GHz, 8.0 GT/s)
Traslational velocity of the blade: $v = -8.71557427mm/s$	Memory: 64 GB DDR3 1600 MHz
Angular velocity around the rotational axis: $\omega = 8.5106383s^{-1}$	Disk space: 8 TB total
Time duration of simulation: $t = 6.25s$	Operating system: Linux Ubuntu 14.04
Initial timestep: 2 $10^{-6}s$	Total wall-clock time: $t = 36257.952s$

Table 4.1: Parameteres used during the simulation with 5977 particles

Parameters used	**ACCES server details**
Particle radius: $r_{min} = 0.35mm$, $r_{max} = 0.45mm$	accespc2.epfl.ch
Number of particles: $n = 47816$	Processors CPU: 2 quad-core Intel Xeon E5-2643 (3.3 GHz, 8.0 GT/s)
Traslational velocity of the blade: $v = -8.71557427mm/s$	Memory: 64 GB DDR3 1600 MHz
Angular velocity around the rotational axis: $\omega = 8.5106383s^{-1}$	Disk space: 8 TB total
Time duration of simulation: $t = 6.25s$	Operating system: Linux Ubuntu 14.04
Initial timestep: 1 $10^{-6}s$	Total wall-clock time: $t = 343197.716s$

Table 4.2: Parameteres used during the simulation with 47816 particles

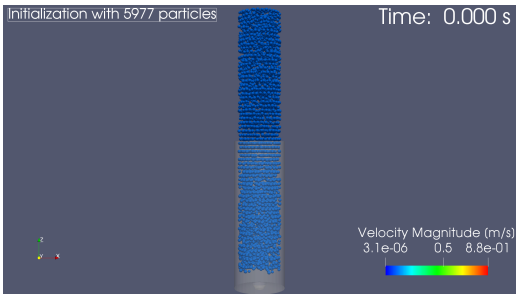
4.1 Qualitative results

Once a simulation is correctly done, an Ensight.case file, together with the other output files concerning the geometry and the particles, are ready to be visualized. The simulations done with GFX, as already mentioned, consit of two phases: the initialization and the simulation.

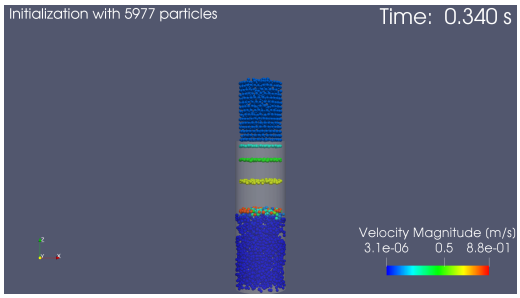
4.1.1 Initialization

In the initialization only the cylinder is considered in the geometry, by setting to -1 the *bc_type* of the blade and the shaft in *Geometry.in*, the geometry input file. The particles are randomly positioned inside a smaller cylinder, as visualized in Figure

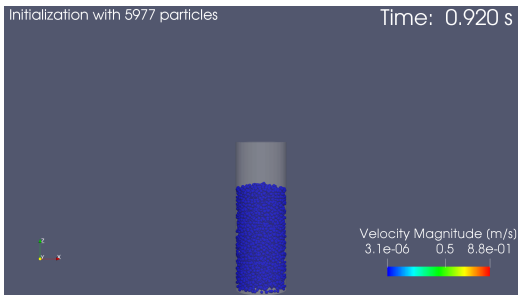
4.1. A small downward velocity (with z-component equal to $-0.1m/s$) is given to them until all the particles have reached their final positions.



(a) t=0.000 s



(b) t=0.340 s



(c) t=0.920 s

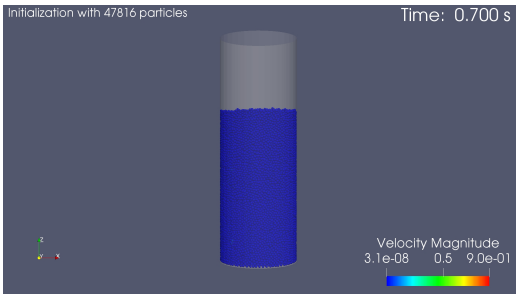
Figure 4.1: Initialization with 5977 particles for different times.

As it concern the simulation with 5977 particles, the radii of the spheres are chosen uniformly between $0.7mm$ and $0.9mm$. Obviously for the simulation with 47816 particles, the radii vary between $0.35mm$ and $0.45mm$. There are several ways to colour the particles in ParaView. In Figure 4.2 we can find some examples: particles coloured according to their velocity, colour index (depending on their height) and radii.

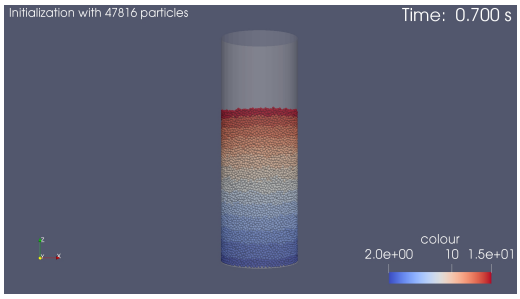
Once the initialization is completed, the output file *particle_000i.dat*, referred to the last time step i, is copied into the input *Particle.in* file and used to run the simulation, beginning where the initialization was stopped.

4.1.2 Simulation

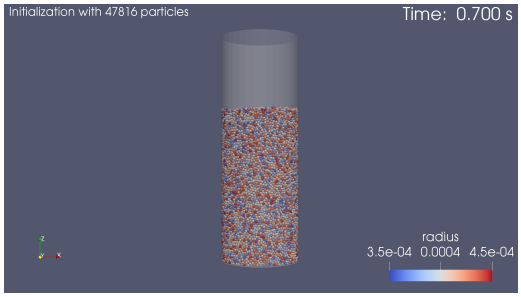
Once the particles have been correctly positioned inside the cylinder, the simulation can start. To do this, the blade and shaft must be switched on, by setting to 1 their *bc_type* in the *Geometry.in* input file. The z-component of particles velocity is set to 0, as the other 2 components, since they have no need of falling downwards anymore. Unfortunately, from a visualization point of view a video would be the best way to see the blade moving downwards through the cylinder. Since this is



(a) Particle velocity



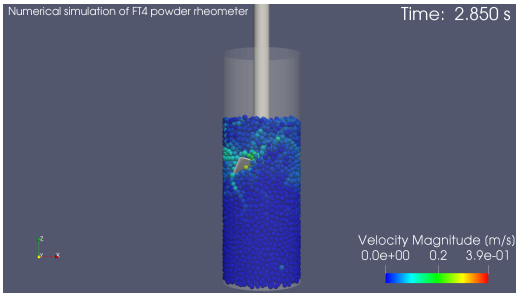
(b) Particle colour



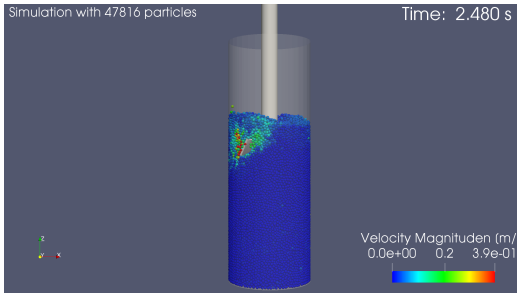
(c) Particle radius

Figure 4.2: Initialization with 47816 particles coloured with different criteria.

not possible here, we show just some pictures taken during this motion. Surely, taking smaller particles lead to a more realistic representation:



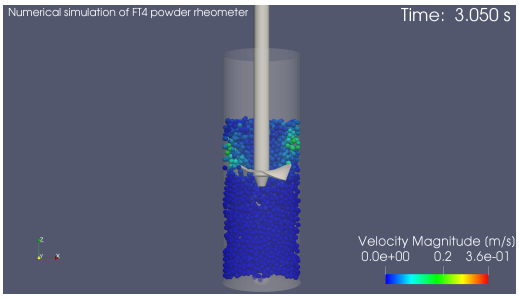
(a) 5977 particles



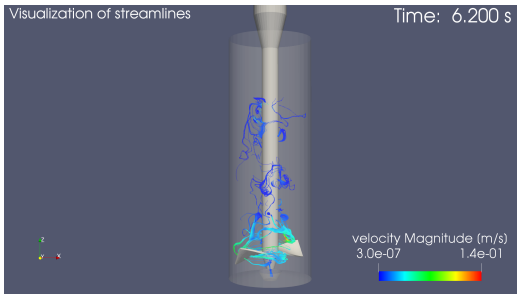
(b) 47816 particles

Figure 4.3: Comparison between simulations with different number of particles.

In Figure 4.4 and 4.5 is shown the use of some ParaView filters. The first pictures show a cutted portion of the cylinder, in order to show what is happening inside. To achieve this view, the Clip filter has been used (with z-axis as normal), together with the Glyph filter to visualize the particles. This last filter is actually always used, it allows to show a small sphere around the point that represents the particle center. For the last pictures a Stream Tracer filter has been used. To use this filter,

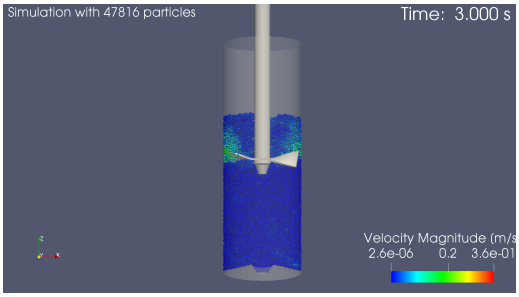


(a) Clip filter

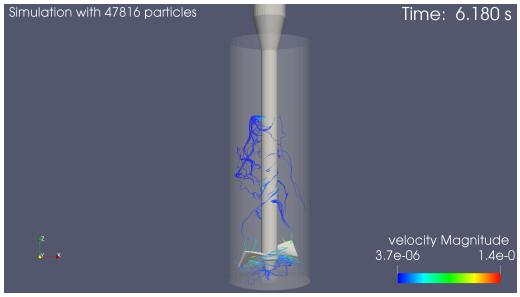


(b) Stream Tracer filter

Figure 4.4: ParaView filters applied to the simulation with 5977 particles.



(a) Clip filter

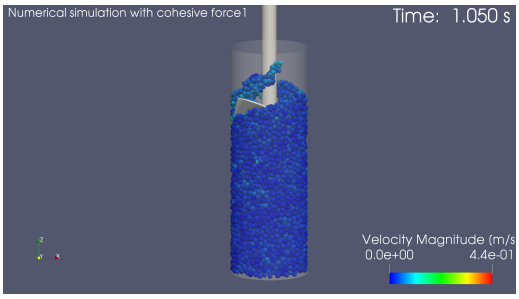


(b) Stream Tracer filter

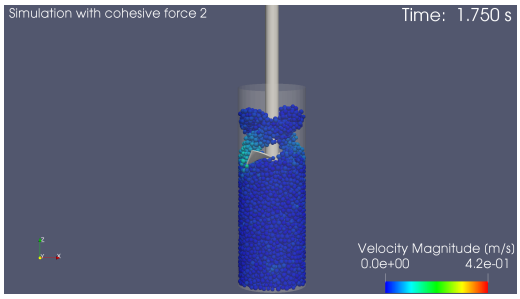
Figure 4.5: ParaView filters applied to the simulation with 47816 particles.

a 3D structured grid is necessary, so a Delaunay 3D filter was previously applied. After this, the streamlines can be visualized.
No qualitative results about the unconfined flow test is showed, since if we consider only images, there are not any differences with the simulation with 5977 particles. The last qualitative analysis is done on the three simulations that have been done to study the cohesion between particles. According to (1.19), a cohesive force has been added between particles. Since the model used here is an elastic force, the parametric study is done on the normal cohesion spring constant. The three simulations are with $k_{c,n}$ respectively equal to -1000 N/m, -500 N/m and -100 N/m. The impact of this added force is immediatly visible:

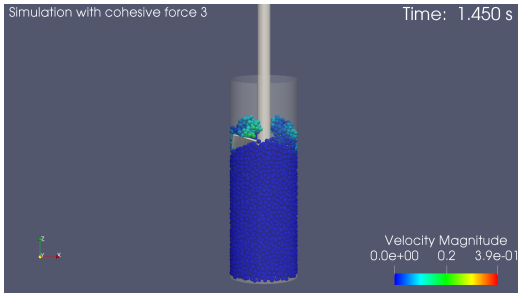
From Figure 4.6 we can see that there is more cohesion between particles, respect to the previous simulations. In particular, a higher spring cohesion constant corresponds to a more 'compact' configuration, as one can expect. From a visual point of view, we can already say that this model is not realistic, expecially with an high value of $k_{c,n}$. This intuition is supported by the fact that during the first two simulations, as the blade goes downwards, some particles exit the cylinder because the cohesion is too high. This consideration is also confirmed by the analysis done during the quantitave post-processing, but we are going to discuss about it in the



(a) $k_{c,n} = -1000N/m$



(b) $k_{c,n} = -500N/m$



(c) $k_{c,n} - 100N/m$

Figure 4.6: Simulations with 5977 particles and cohesive forces with different elastic cohesive constant values.

next Section.

4.2 Quantitative results

From a quantitative point of view, we are able to compare the simulation and the experimental results. In particular, the main quantities analyzed are: the force, the torque and the energy. All these quantities are meant as statistical values, averaged on all the particles. The way to calculate them has been already discussed in Chapter 3. We are going to divide the analysis with respect to the simulations carried out: the confined flow tests (for both 5977 and 47816 particles), the unconfined flow test, the cohesion tests (we call them like this) and the parametric tests. The plots are done with Matlab R2017b, used to read the Excel files with the data of the simulations.

4.2.1 Confined flow tests

Thi is the first simulation runned. For $t = 6.25s$ the blade goes downwards through the vessel, forcing the powder to flow. Once the simulation is finished, we have to check if the final number of particles is the same as the initial number. This can be

done with the *log.dat* output file, where this information is recorded for each time step. Another check that can be done concerns the averaging and maximum overlap of the particles. From the theory, we know that the overlap should not exceed 1% of the particle diameter. The maximum particle-particle overlap is 0.92%, while the maximum particle-element overlap is 7% that is an higher value, but since the average value is 0.03% maximum, this is not an issue.
To compare the simulation results with the experimental results, we have considered the *temporal.dat* output file. This file is structured as follow: for each time step there is a list of the three components of the force, the magnitude of the force and the power exherted by the particles on the blade. All these quantities are averged on all the particles. Since the power is given by $P = \mathbf{T} \cdot \boldsymbol{\omega}$, we were able to calculate the torque, by dividing it for the angular velocity. Since the quantities studied are statistical, a smoothed plot is used. This is done with the Matlab function *'loess'*, a locally weighted non-parametric regression fitting that uses a 2nd order polynomial [30]. The following plots are the comparisons of force, torque and energy:

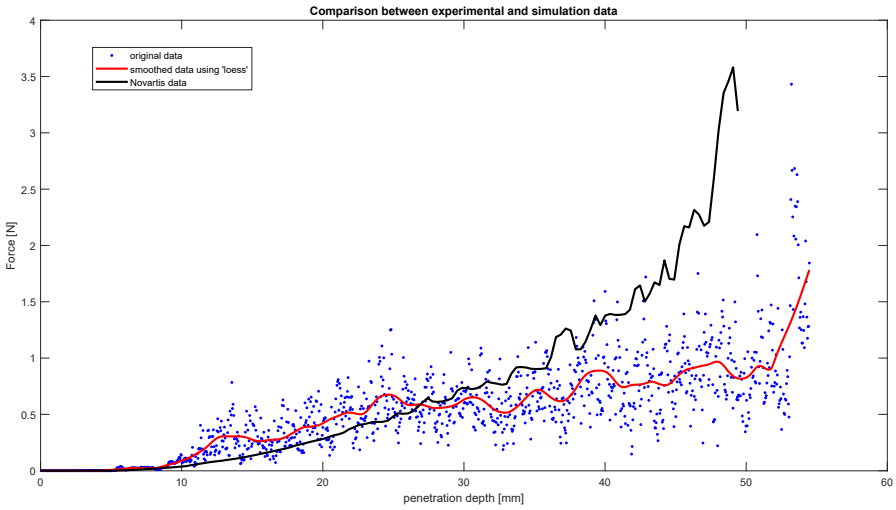


Figure 4.7: Plot of the force calculated during the simulation with 5977 particles, compared to the experimental data.

As we can see, there is a good agreement between the simulation and the experimental data. There are more data about the simulaiton, since the simulation is done until the blade reaches the bottom of the vessel, while during the experiment the blade is stopped $5mm$ before. The biggest difference inside the force plot is that from a penetration depth of about $30mm$ until the end, the simulation estimate a lower force with respect to the experiment. This trend is also present in the torque plot, but is less pronounced and manifests only after penetration depth of $45mm$. As it concern the energy plot we can see a trand that is common for

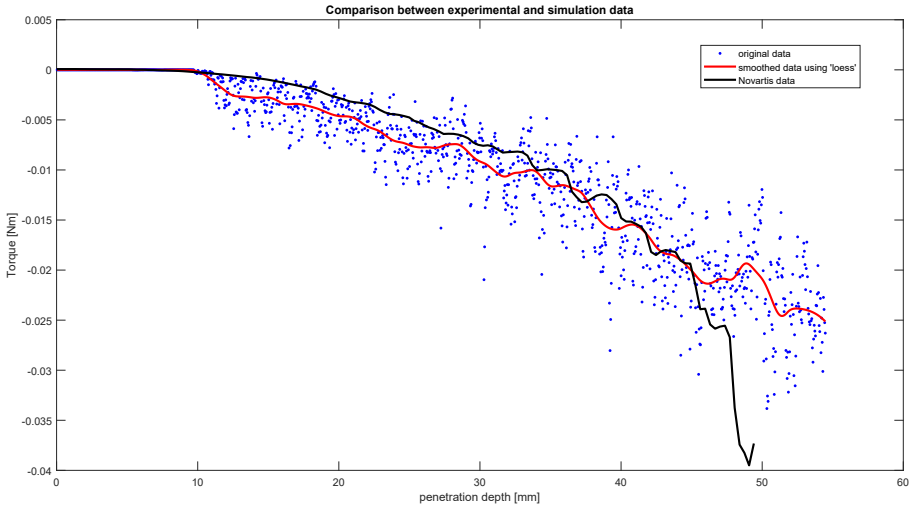


Figure 4.8: Plot of the torque calculated during the simulation with 5977 particles, compared to the experimental data.

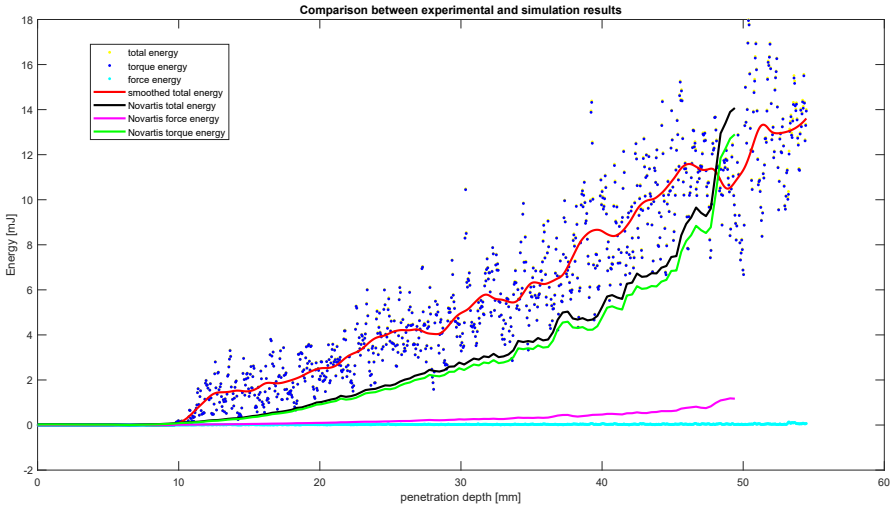


Figure 4.9: Plot of the total energy and its components calculated during the simulation with 5977 particles, compared to the experimental data.

all the simulations, and also for the experimental results: the energy is basically given by its torque component, while its force component does not influence it a lot. The major disagreement is in the energy plot, from which we can see that the energy calculated during the simulation is major then the experimental one. This disagreement manifests itself also in the calculation of the BFE:

$$BFE_{exp} = 406mJ \qquad BFE_{5977} = 6379mJ \tag{4.1}$$

Even if this value is one order of magnitude higher, we should not desperate. Indeed, the features of the simulation considered here are far away from the reality. Just thinking about a sphere with a diameter of $1.6mm$ compared to a real powder particle, allows to understand the differences present. To better understand how much the particles size influences the results, a second simulation was runned. The radius of the particles for this new simulation is chosen to be a half of the previous one, for a total of 8 times the number of particles used before. With the new amount of particles, the total time taken for the simulation has increased a lot, as reported in Tabel 4.2. The results have been compared to the experimental ones and to the results of the simulation with 5977 particles.

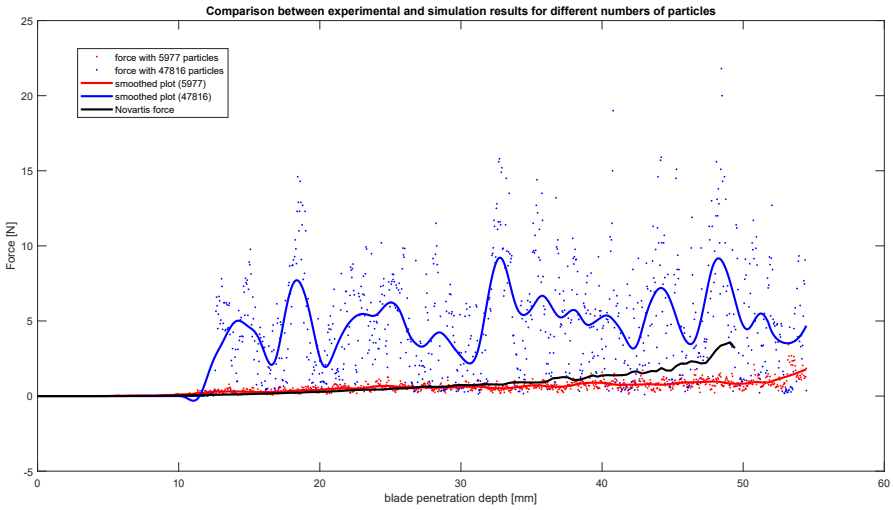


Figure 4.10: Plot of the force calculated during the simulation with 47816 particles, compared to the one of the simulation with less particles and the experimental data.

Form the force plot we can see that the force is higher in this simulation. There are also high peaks and the general path is less regular. This is probably due to the presence of more particles, and therefore more different data. A remark that can be done is that the average force is affected by the change of particle size, becoming bigger as the particle radius decrease. The torque, on the contrary, does not show this difference. Furthermore the torque calculated during the simulation with more particles, vary less, assuming a path that does not decrease as the others (see Figure 4.11). This trand is also present in the energy plot. This leads to a smaller value of the Basic Flowability Energy:

$$BFE_{exp} = 406mJ \qquad BFE_{5977} = 6379mJ \qquad BFE_{47816} = 5519mJ \tag{4.2}$$

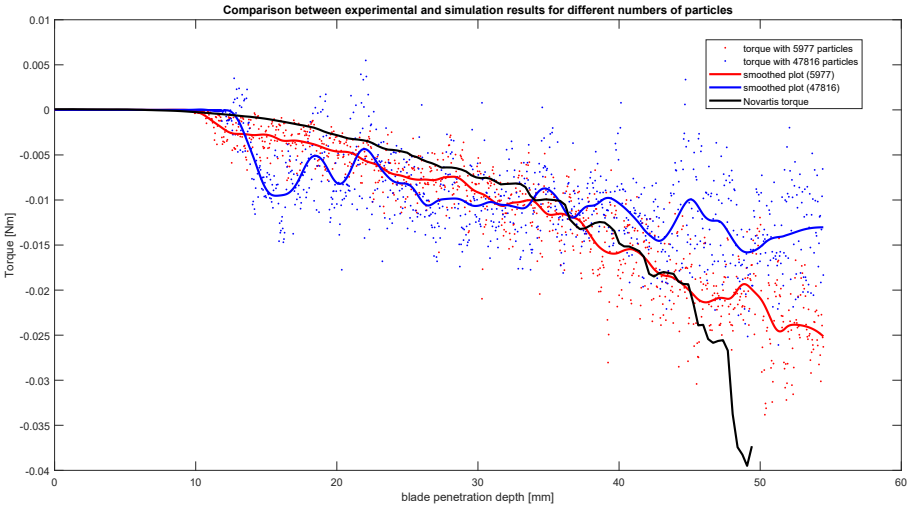


Figure 4.11: Plot of the torque calculated during the simulation with 47816 particles, compared to the one of the simulation with less particles and the experimental data.

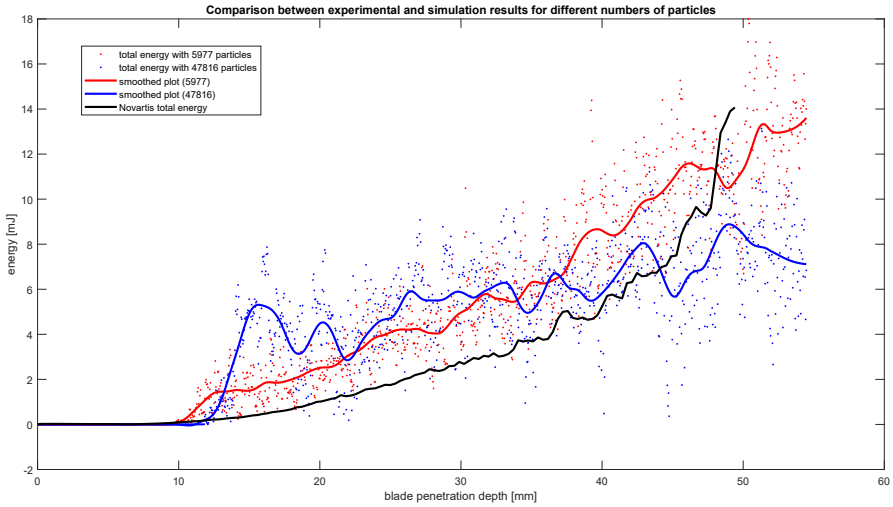


Figure 4.12: Plot of the total energy calculated during the simulation with 47816 particles, compared to the one of the simulation with less particles and the experimental data.

There is a great decrease of the BFE, even if it is still big if compared to the experimental one. This means that if from one hand the particle size contributes

to reduce the BFE, letting then the results to become more similar to the experimental ones, from the other hand it could not be the only factor at stake.

4.2.2 Unconfined flow test

For an unconfined flow test, the quantitiy of interest is not the BFE but the SE, the Specific Energy. Unfortunately, the experimental data concern a confined flow test, with the calculation of the BFE. Therefore, we were not able to compare this test with experimental data. Nevertheless, we can compare the results obtained with the ones concerning the first simulation, with 5977 particles. In fact, the plots regarding the force, the torque and the energy behave similarly, but are inverted with respect to the vertical axis. This is due to the fact that the blade is moving upwards, and therefore its direction is inverted.

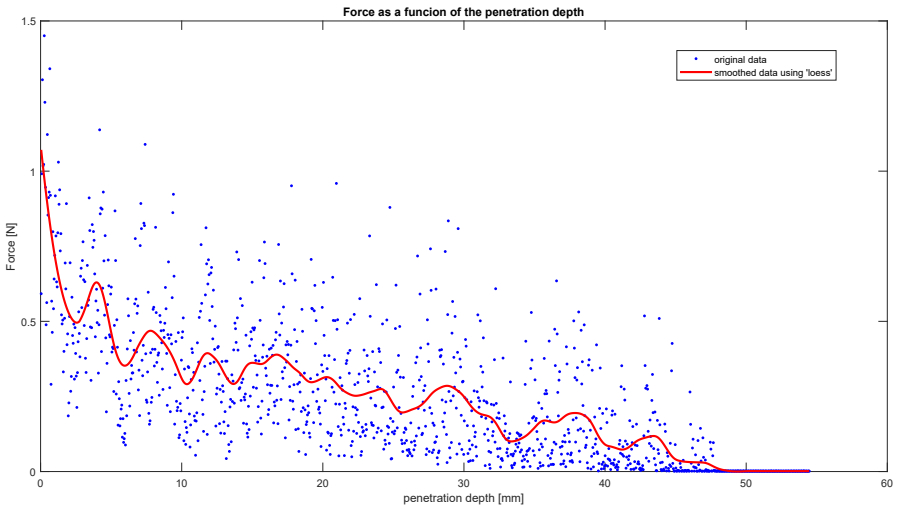


Figure 4.13: Plot of the force calculated during the simulation of an unconfined flow test.

Even if we can not compare these results with experimental data, we have calculated the Specific Energy SE, in the same way of the BFE. We remember that this is the energy measured when the powder is not forced to flow. Since it is measured in mJ/g, the data generated from this upward cycle are then normalized against the mass of powder tested [28]. The resulting value is $SE = \frac{3580.7mJ}{63.8g} \simeq 56\frac{mJ}{g}$. In literature we can find lower values for the SE of lactose used in this type of experiments (see [29]). Nevertheless, this is comprehensible since we are dealing with larger particles and the energy necessary to move them is bigger.

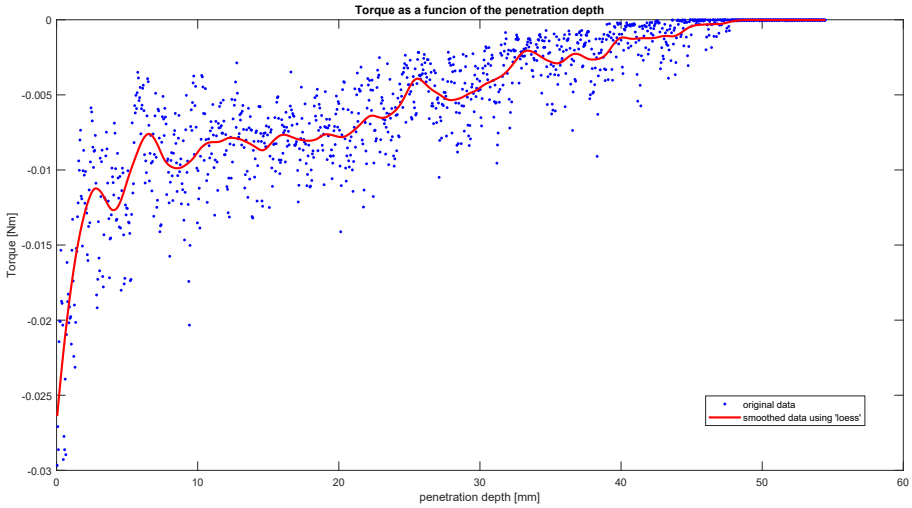


Figure 4.14: Plot of the torque calculated during the simulation of an unconfined flow test.

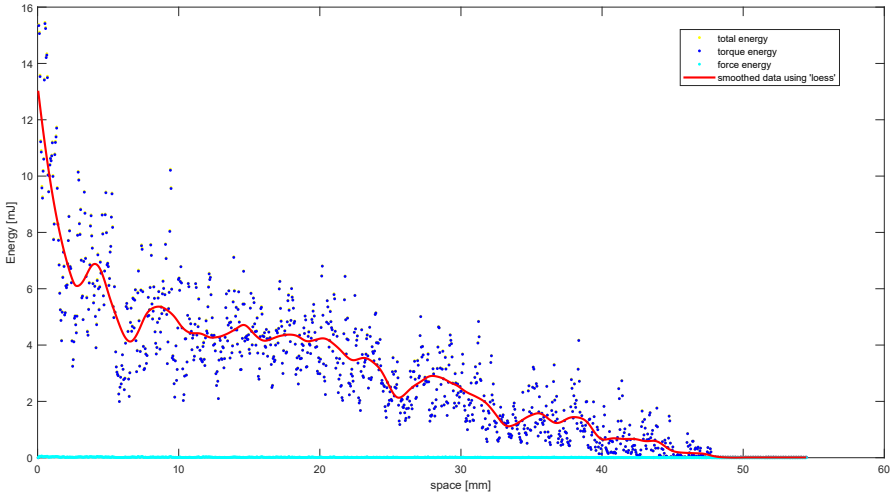


Figure 4.15: Plot of the total energy and its components calculated during the simulation of an unconfined flow test.

4.2.3 Cohesion tests

Three different simulations have been done with three different values of the cohesion spring constant. The simulations have been done with $k_{c,n} = -1000N/m$, $k_{c,n} = -500N/m$, $k_{c,n} = -100N/m$, all with 5977 particles. The comparisons with experimental results have been analyzed:

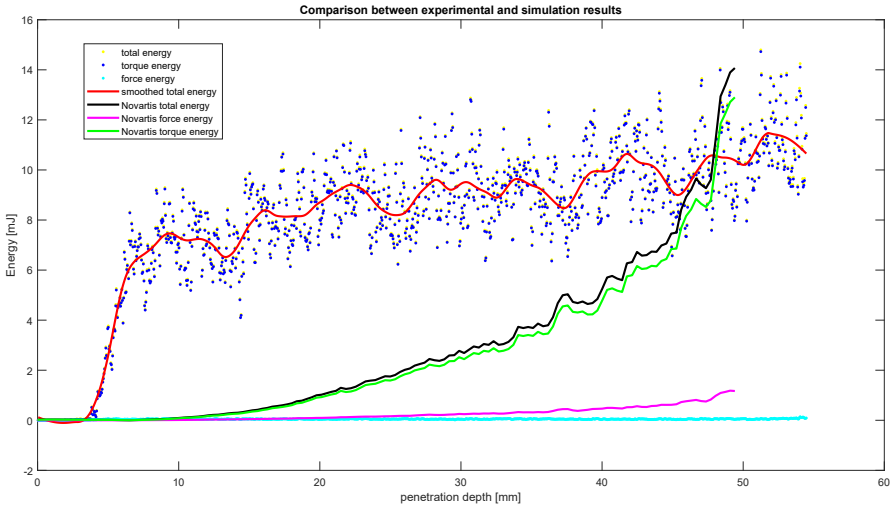


Figure 4.16: Plot of the total energy and its components calculated during the cohesion test 1 with $k_{c,n} = -1000N/m$, compared to the experimental data.

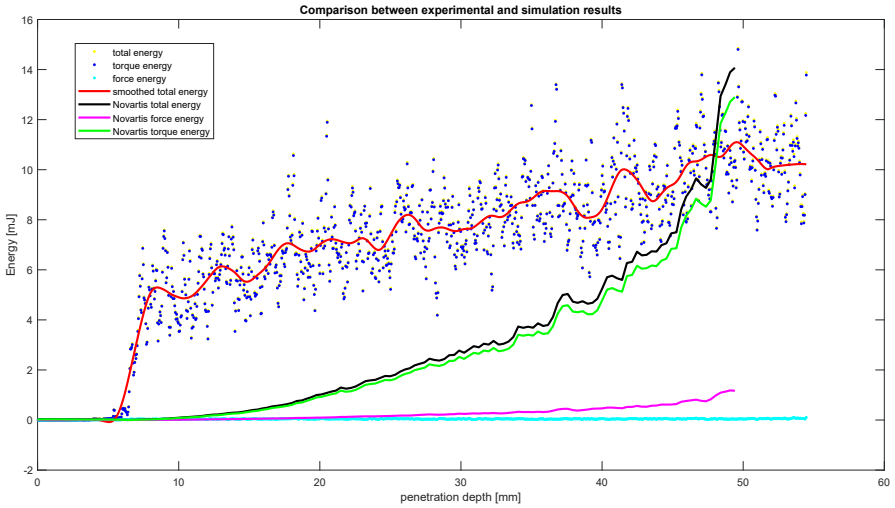


Figure 4.17: Plot of the total energy and its components calculated during the cohesion test 2 with $k_{c,n} = -500N/m$, compared to the experimental data.

Some comments about these three simulations must be done. We can see a different behaviour, not present in the other simulations. There is a peak of the energy even before the blade has touched the particles. This means that just the motion of the blade activate the cohesion between particles. Indeed, this trend is more pronounced in the first two simulations, with a higher cohesion spring

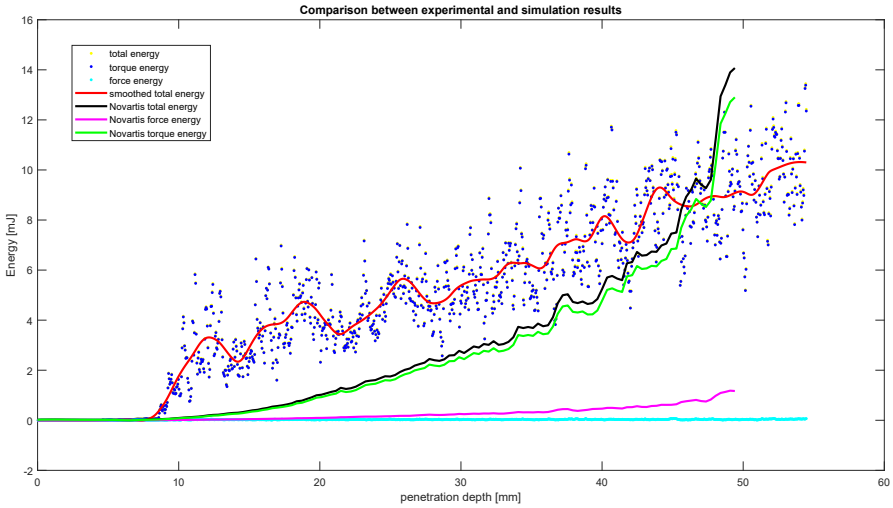


Figure 4.18: Plot of the total energy and its components calculated during the cohesion test 3 with $k_{c,n} = -100N/m$, compared to the experimental data.

constant. Also the general behaviour of the energy is in line with our expectations: generally more cohesive particles need more energy to flow. This is confirmed by the calculation of the BFE of the three tests:

$$BFE_{test1} = 10134mJ \qquad BFE_{test2} = 8738mJ \qquad BFE_{test3} = 6299mJ \tag{4.3}$$

The only thing in disagreement with this, is the fact that the energy referred to the first two experiments grows with a less pronounced slope. Since this behaviour is not present in the last simulation, we can attribuate it to the loose of particles during the simulation. If there are less particles at the end, the energy could not grow and remain basically around the same value of the first part of the simulation. In light of this, and also with the help of the visualization of the results, we can say that the first two simulations are not comparable to the experimental results. The last simulation instead can be compared to it, and also to the other simulations done:

In Figure 4.19 the total energy is analyzed. We have omitted both the force and torque energy components, to have a more readable plot. An interesting result is underlined in this plot, for a penetration depth between $35mm$ and $40mm$. Before this part, both the simulations with 47816 particles and the one with a cohesive force led to an energy that is higher then the energy attributed to the simulation with 5977 particles. After this part, on the contrary, the simulation with less particles produces a higher energy with respect to the others. This means that, for the first part of the simulations the finer and the more cohesive powders need more energy to flow, while for the last part they need less energy.

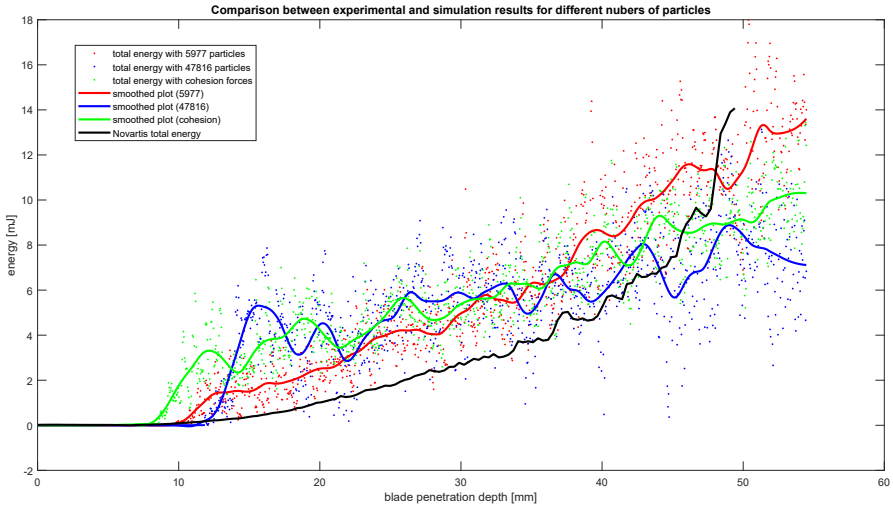


Figure 4.19: Comparison between energies calculated during all the simulations done and experimental data.

4.2.4 Parametric tests

Since the improvements done with more particles and the addition of a cohesive force were limited in terms of BFE calculation, other simulations have been ran. Following the works done in [35, 36, 37], we have changed the parameters concerning the model used. In particular, we are going to analyze how the Coloumb frictional limit μ, the rolling friction coefficient μ_r and the coefficient of restitution ϵ influence the results. We concentrate on the energy and BFE calculation, in order to know if it is lower then the BFE calculated during the previous simulations. The new simulations are done with 5977 particles during a confined flow test. The first two simulations are done with changes on μ and μ_r, the parameters present in Equations (1.18) and (1.22). Since we would like to decrease the BFE, we should decrease μ, μ_r and ϵ to achieve this. In fact, as stated in [35, 36], increasing these parameters means increasing the flow energy and therefore the BFE. The last two simulations were instead done by changing the coefficient of restitution. We notice that this is possible just with low values of μ and μ_r. The results are the following:

As we can see, we have found lower values of BFE. μ has a greater impact with respect to μ_r as we can see from the first two lines. Nevertheless, the best results obtained, the more similar to the experimental value, are achieved during the last two tests. Decreasing both μ and μ_r lead to considerably lower values of BFE. The last two lines show that for low values of μ and μ_r, the coefficient of restitution does not affect the flow energy. Indeed the BFE values found are almost the same, and this is in agreement with what is written in [36]. We remember that a low value of BFE corresponds to a high flowability of the powder, since it resists less

μ	μ_r	ϵ	BFE [mJ]
0.05	0.2	0.3	987
0.75	0.05	0.3	3816
0.05	0.05	0.05	565
0.05	0.05	0.3	567

Table 4.3: Parametric simulations

to the blade's movement.
The BFE is also defined as the area under the energy gradient curve. The energy gradient is the energy for each infinitesimal depth travelled by the blade, and is expressed in $\frac{mJ}{mm}$. We therfore report the graph of the energy gradient for the simulation with $\mu = 0.05$, $\mu_r = 0.05$, $\epsilon = 0.05$, compared with the experiment. The BFE is then the area under these curves:

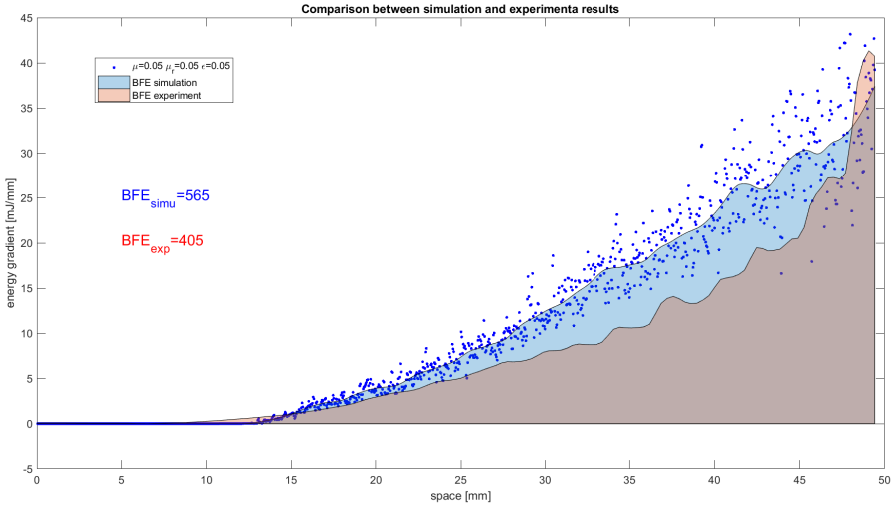


Figure 4.20: Comparison between energy gradients of one parametric simulation and experimental data.

This is the best result achieved compared to the other simulations done.
We should now think about the physical meaning of the parameters considered during these simulations. Firstly, the Coloumb frictional limit also called dynamic friction coefficient or sliding friction coefficient, represents the resistence to the lateral motion of the moving surfaces of two particles in contact. A low value of μ means that there is not a high resistence to motion and the particles are more free to flow. The same reasoning can be applied to the rolling friction coefficient. Indeed, a low value of μ_r causes a less resistence between particles that roll over each other,

and therefore a higher flowability and a lower value of BFE. The coefficient of restitution ϵ is defined as the ratio of the final to the initial relative velocity between two particles during a collision. If $\epsilon = 1$, a perfect elastic collision accurs. Since we have chosen a small value for it, the particles are slowed down afer collisions. To better understand how varying this parameter changes the simulations results, we should run more simulations, since no relevant differences were registered during the last two tests.
The result of this last set of simulations is that particle property parameters are more influential then the number of particles and the presence of cohesion between particles. Indeed the last parametric simulations have led to results more similar to the experimental ones. No qualitative post-processing is done in this case, since these simulations are addressed to the study of BFE.
A further discussion about the simulations is done in the next Section.

4.3 Discussion

As a result, we can say that for almost all the simulations done, the major difference with the experimental data concerned the calculation of BFE. The measured values that we obtained with our first simulations seemed to be dramatically high if compared to the experimental one. Even if this discrepancy has been registered, we report the values of the BFE calculated during a confined flow test with lactose powders. These values has been taken from [32]:

Materials	**Basic Flowability Energy, BFE [mJ]**
Spray dried lactose	1191
Sieved lactose	2413
Coarsely milled lacotse	2164
Finely milled lactose	635

Table 4.4: BFE calculations, taken from [32]

Similar measurements can be found in [34]. We can therefore affirm that the BFE value strongly depends on the material considered, in particular the lactose that is used during pharmaceutical experiments with the FT4 powder rheometer. This assure us to have not found non-physical quantities. Obiouvsly, improvements of the simulations can be done. Since the BFE calculated during the simulation with 47816 particles is lower then the BFE calculated for 5977 particles, a possible improvement could be increasing more the number of particles. Unfortunately, this would lead to a decreasing of the time-step to have stability, and therefore an increasing of the time cost of the simulation. Making too longer simulations means loosing the benefits of the numerical study itself. A possibility to do not

have limitations on the integration time step is to consider an implicit discretization time scheme. Nevertheless, we need to remember that for a LSD model with soft-particles, the integration time step cannot be taken too long, otherwise there would be a higher threshold for the particle overlap, and this is not physically possible. Other models of contact forces for DEM simulations can be used; the most common in literature is the Hertz-Mindlin-Deresiewicz contact model [33]. Also a different contact search algoritm, as seen in Section 2.2.2, could decrease simulation timing and increase results accuracy.
If from one side the Discrete Element Method is designed to study granular material, and is enforced by the freedom to choose the particle properties to better represent the case study considered, from the other side the choose of the model parameters is not straightforward. Some interesting studies about these parameters have been done in [35, 36, 37]. In particular, in [35] *Hare et al.* have shown that an increasing of the sliding friction coefficient from 0.1 to 0.5 causes an increase of the total flow energy, so the BFE. In [37] *Bharadwaj et al.* have instead investigated the sensitivity of the force and torque on the coefficient of restitution, particle sliding and rolling friction coefficients. They state that an increasing of rolling and sliding friction coefficients corresponds to an increasing of both the force and torque values. Finally, in [36] we can read that for the same restitution and friction coefficients, highly cohesive powders show higher flow energy. This result is in line with the BFE values that we have found during the cohesion tests. *Wilkinson et al.* [36] have also done a similar study of [37], but analysing the BFE and SE values insted of force and torque. They have showed that high rolling friction coefficient lead to high BFE, as well as a high restitution coefficient, but with less impact.
Even if these studies were done with different models and simulation setups, we have mentioned them to emphasize the fact that better results can be achieved with a more targeted choice of model parameters. This is what has been done with the last simulations. The parameters taken into account are the Coloumb friction limit or dynamic friction coefficient μ, the rolling friction coefficient μ_r and the coefficient of restitution ϵ. From the simulations done, the result is that for lower μ and μ_r a lower BFE has been found. In particular, μ has a higher impact. No differences were instead found decreasing the coefficient of restitution with the other two parameters already low. This means that ϵ influences less the flowability energy. Actually, to better understand this, and also to validate the results obtained, many others simulations should have been ran. However this work is not aimed at doing a complete parametric study, but is done instead to understand the general behaviour of powders involved in FT4-powder rehometer tests. With this purpose, different simulations have been carried out.

Conclusion

The Discrete Element Method (DEM) is a numerical method for computing the motion of a large number of small particles [38]. Today DEM is becoming widely accepted as an effective method of addressing engineering problems in granular and discontinuous materials, especially in granular flows, powder mechanics, and rock mechanics [31]. Modeling enables us to understand different phenomena, to perform sensitivity analysis on different input parameters and to test different configurations at lower expense compared with experimental methods [1].
This is the reason why we did the numerical simulation of the Freeman-FT4 powder rheometer as an application of DEM, for Novartis. Since the rheometer is a device used to measure the powder's properties, we have investigated how it behaves under different conditions. Several simulations have been carried out with this purpose. The first two simulations have been done with different number of particles, respectively 5977 and 47816. We have found that in both cases the numerical results were in good agreement with the experimental ones. The forces, torques and energies calculated followed well the path of the experimental data. A stronger discrepancy has benn instead observed in the calculations of the BFE, since higher values then the one expected have been found, expecially for the simulation with less particles. This is due to the fact that the blade has travelled with less resistence from a finer powder, producing a lower BFE during the simulation with more particles. Despite that, the value of BFE was still too high to be compared to the experimental one. Morevorer, there was the need to reduce the time step for the simulation with 47816 particles to be stable, increasing the timing and CPU costs. For this reason, the other simulations have been done with the previous number of particles.
As it concern the cohesion tests, three simulations varying the spring cohesion constant of the elastic cohesive force have been carried out. Only the last simulation, with a lower value of the parameter was comparable to the previous simulations, but still with a high BFE if compared to the experimental one. Indeed the first two tests have been done with a too high cohesion constant and thank to ParaView was possible to visualize a non-realistic too cohesive powder.
Another type of test has been also simulated, with the blade going upwards. The results in this case have been not compared to the experimental ones, because it was another type of experiment, but it has been an iteresting validation on the

code for a different test. In this case the value calculated was the Specific Energy, SE that was again higher then the values found in literature.
The conclusion of this first set of simulations was that the number of particles and the cohesion between them influence the flowability of powders, but probably other parameters such as particle properties have a major impact.
An investigation in this sense have been carried out with the last simulations. A parametric study have been done on particle-particle interactions. The parameters considered were the Coloumb friction limit μ, the rolling friction coefficient μ_r and the coefficient of restitution ϵ. The intuition that the particle properties influence more the model, has been confirmed by these simulations. Indeed, a BFE value very similar to the experimental one has been found in the last two simulations. This is comprehensible since low values of μ and μ_r have been chosen, and this means that particles resist less during collisions with each other and therefore the general flowability is higher, producing a lower value of BFE. As it concern the influence of ϵ on powder behaviour and flowability, it was not enlighted during the last two simulations, since its contribute was less impactive, and almost the same value of BFE was found with $\epsilon = 0.3$ and $\epsilon = 0.05$. The study about the contribution of this parameter, as well as the influence of the material parameters concerning the blade and the vessel, could be analyzed during a further work. The qualitative post-processing about the last simulations was omitted, since the interest in this case was focused on finding a suitable BFE value to be compared with the experimental one.
Since this purpose was achieved, and since various improvements can still be made to the model applied, we can be satisfied with the results obtained.

Appendix A

Input files

A.1 Example of Control.in

```
!  Control file for GFX code
!
       1.0        ! version number
!
! 2. DEFINE ---------------------------------------------------------------
!
! 2.1 Physical parameters  ..................................................
!
         1        ! contact model : 1 = spring-dashpot

         F        ! 2D computation
         F        ! particle clustering enabled
         T        ! particle spin enabled
         T        ! rolling friction enabled
         F        ! cohesive forces enabled
         F        ! particle bonding enabled
         F        ! fluid damping enabled
         F        ! fluid buoyancy enabled
         F        ! particle growth enabled
         T        ! particle interaction enabled

   0.   0. -9.81    ! (x,y,z) components of gravity

   0.   0.   0.     ! (x,y,z) velocity components of fluid
      1000.        ! mass density of fluid [kg/m3]
      0.001        ! viscosity of fluid [kg/(ms)]
      1.0          ! coefficient for fluid damping [0.0,1.0]
!
! 2.2 Material parameters  ..................................................
!
         0        ! additional number of materials

! 3. INITIALIZE -----------------------------------------------------------
!
! 3.1 Geometry  .............................................................
!
        1         ! bounding box [i_box] : 1 = read from geometry file;
                  !                        2 = defined below
        0         ! bounding box status :  0 = fixed to below values
                  !                        1 = updated; 2 = periodic
   -5.0   4.0     ! min & max x limits of bounding box [if i_box = 2]
   -2.0   2.0     ! min & max y limits of bounding box [if i_box = 2]
  -10.0  12.0     ! min & max z limits of bounding box [if i_box = 2]
    0   0   0     ! (x,y,z) periodicity (0 = updated; 1 = periodic)
!
! 3.2 Particles  ............................................................
!
        1         ! particle initialization : 1 = read from particle file;
                  !                           2 = create; 3 = read + create
   0.   0.   0.   ! (x,y,z) components of created particle velocity
   0.   0.   0.   ! (x,y,z) components of created particle spin
        T         ! read particle velocity & spin re-initialized to these values
        6         ! created particle colour index
```

```
        T        ! read particle colour re-initialized to this value

     5977        ! maximum number of particles created

        2        ! container shape [i_form] : 1 = box; 2 = cylinder
        3        ! stacking direction : 1 = x; 2 = y; 3 = z
  0.0 0.0  0.01  ! centre coordinates of box/cylinder
  0.01 0.01 1.0  ! length of sides of box              [if i_form = 1]
   0.0126 10.0   ! radius and length of cylinder       [if i_form = 2]
        T        ! random component added to particle position & velocity

        1        ! number of discrete particle groups
     1   1       ! group 1 : size dist. : 1 = constant; 2 = RR ; no. fraction
     1   1       ! group 1 : material, type
  0.0007 0.0009  ! group 1 : radius range (lower & upper values)
!
! 3.3 Particle growth  ..................................................
!
       10        ! update particle growth (timesteps)
    0.005  0.01  ! overlap limits for growth (lower & upper values)
  0.9999 1.0001  ! growth factors (reduction & enlargement)
  0.005  0.025   ! desired particle radius (lower & upper values)
!
! 4. SOLVE ------------------------------------------------------------------
!
! 4.1 Numerical parameters  .............................................
!
      6.25       ! time duration of simulation [s]
    0.0000010    ! initial timestep [s] ;  if < 0 calculate
     -200        ! update near neighbour list (timesteps) ; if < 0 calculate

        1        ! near neighbour list : 1 = cartesian search grid
        1        ! time integration scheme : 1 = first-order
        1        ! parallel partitioning direction
!
! 4.2 Performance parameters [ USE WITH CAUTION ] .......................
!
        T        ! check status of particles each time step
        T        ! check for discarded particles (only if status is checked)
!
! 5. EXPORT -----------------------------------------------------------------

! 5.1 Export parameters  ................................................
!
    0.02         ! visualization export time interval [s]
    0.02         ! logfile export time interval [s]
    1.00         ! restart export time interval [s]
    0.005        ! temporal data averaging / export time interval [s]
  50.0  55.0       ! angular data averaging start and stop times [s]

        2        ! visualization file format : 1 = OpenDX; 2 = Ensight6

        T        ! subroutine profiling enabled
        F        ! binary restart data
        T        ! angular data export enabled
        T        ! periodic angular data export enabled
        F        ! contact visualization enabled
        F        ! bond visualization enabled
!
! 5.2 Visualization quantities (particles)  .............................
!
        T        ! radius
        T        ! colour
        F        ! velocity magnitude
        T        ! velocity components (x,y,z)
        T        ! spin magnitude
        F        ! spin components (x,y,z)
        F        ! coordination number
        F        ! bond number
        T        ! normal force
!
! 5.3 Visualization quantities (geometry)  ..............................
!
        F        ! pressure
        F        ! normal force
        F        ! velocity magnitude
        F        ! velocity components (x,y,z)
!
! 5.4 Temporal statistical quantitites  .................................
!
        T        ! total force, power
        F        ! centre of mass components (x,y,z)
        F        ! average linear momentum
```

```
        F       ! average linear velocity
        F       ! average angular velocity
        F       ! mixing measures GMMI_x,GMMI_y (GMMI_z) based on colour
```

A.2 Example of Material.in

```
!  Materials Database file for Granular Flow Simulation Code
!
    1.0      ! version number
!
!   Number of materials
      3
!
!   imat    name
!           density         ks_norm         ks(t/n)         kd(t/n)
!           mu_c            coef rest       mu_r
!           kc_norm         sigc_0          sigc_1          muc_r
!           ks_bond         kd_bond         sigb_0          sbn_max
!
    1       [particle]
            2500.           100000.         1.              1.
            0.75            0.3             0.2
            -1000.          0.              0.03            0.
            1000000.        50.           5.e-4           4.e8

    2       [blade]
            7500.           100000.         1.              1.
            0.75            0.3             0.2
            -100.           0.              0.              0.
            0.              0.7             0.e-5           1.e10

    3       [cylinder]
            5000.           100000.         1.              1.
            0.75            0.3             0.2
            -100.           0.              0.              0.
            0.              0.7             0.e-5           1.e10
```

Appendix B

Output files

B.1 Example of Standard output file

```
------------------------------------------------------------------

              gfx  -  Granular flow simulation code

                       Serial Version 0.10

                     Granulair Technologies

------------------------------------------------------------------


 ... INITIALIZATION ...

 Reading files from directory : /home/bistoni/Freeman_FT4/Run_blade_vel

 Material database file read (version 1.0)
 - properties for  3 materials read from database
 Control data file read (version 1.0)
 Geometry data file read (version 1.0)
 Particle data file read (version 0.8)

 Non-planar quadrilateral elements detected
 Number of quadrilateral elements split =   704

 3D computation

 Physical properties enabled :
 - particle spin
 - particle rolling friction
 - inter-particle collisions

 Physical properties disabled :
 - particle clustering
 - particle cohesion
 - particle bonds
 - interstitial fluid damping
 - buoyancy due to interstitial fluid
 - particle growth

 Initial number of particles =   5977

 Integration timestep is  2.00000E-06 [s]
 - this value was read from input data file

 Near neighbour list updated initially every   50 timesteps
 - this value may be modified during simulation

 Simulation is initialized at time = 0.00000 [s]

 Visualization particle data written for time = 0.000 [s]
```

```
 ... CALCULATION AND POST-PROCESSING ...

 Temporal statistical data commenced
 Visualization particle data written for time = 0.050 [s]
 Visualization particle data written for time = 0.100 [s]
 Visualization particle data written for time = 0.150 [s]
 Visualization particle data written for time = 0.200 [s]
 Visualization particle data written for time = 0.250 [s]
 Visualization particle data written for time = 0.300 [s]
 .
 .
 .
 Visualization particle data written for time = 5.950 [s]
 Visualization particle data written for time = 6.000 [s]
 Restart data written for time = 6.000 [s]
 Visualization particle data written for time = 6.050 [s]
 Visualization particle data written for time = 6.100 [s]
 Visualization particle data written for time = 6.150 [s]
 Visualization particle data written for time = 6.200 [s]
 Visualization particle data written for time = 6.250 [s]
 Restart data written for time = 6.250 [s]

 gfx successfully completed
```

B.2 Example of Case file

```
# Ensight6 formatted case file for GFX dataset

FORMAT
type:  ensight

GEOMETRY
model:           1            output/geometry_****.geo
measured:        1            output/particle_****.pos

VARIABLE
scalar per measured node:  1  radius    output/particle_****.rad
scalar per measured node:  1  colour    output/particle_****.col
scalar per measured node:  1  spin      output/particle_****.spn
scalar per measured node:  1  force_n   output/particle_****.frn
vector per measured node:  1  velocity  output/particle_****.vel

TIME
time set:                  1
number of steps:           126
filename start number:     1
filename increment:        1
time values:
  0.0000
  0.0500
  0.1000
  0.1500
  0.2000
  0.2500
  0.3000
  0.3500
  .
  .
  .
```

Bibliography

[1] Norouzi H. R., Zarghami R., Sotudeh-Gharebagh R. and Mostoufi N. (2016) - *Coupled CFD-DEM Modeling: Formulation, Implementation and Application to Multiphase Flows*, John Wiley & Sons, Ltd.

[2] Sawley M. (2018) - *Particle-Based Methods*. École polytechnique fédérale de Lausanne, unpublished.

[3] Ferrez J. (2001) - *Dynamic triangulations for efficient 3d simulation of granular materials*. Doctoral thesis, École polytechnique fédérale de Lausanne.

[4] Nezami E. G., Hashash Y.M.A., Zhao D. and Ghaboussi J. (2004) - *A fast contact detection algorithm for 3-D discrete element method*. Computers and Geotechnics, **31**(7), 575-587.

[5] Cundall P.A. and Strack O.D.L. (1979) - *A discrete numerical model for granular assemblies*. Geotechnique, **29**, 47-65.

[6] Sawley M. (2009) - *GFX User Manual*. Granulair Technologies.

[7] Kruggel-Emden H., Simsek E., Rickelt S., Wirtz S., and Scherer V. (2007) - *Review and extension of normal force models for the discrete element method*. Powder Technology, **171**, 157-173.

[8] Jia T., Zhang Y., Chen J. K., and He Y. L. (2012) - *Dynamic simulation of granular packing of fine cohesive particles with different size distributions*. Powder Technology, **218**, 76-85.

[9] Ye M., van der Hoef M. A., and Kuipers J. A. M. (2005) - *The effects of particle and gas properties on the fluidization of Geldart A particles*. Chemical Engineering Science, **60**(16), 4567-4580.

[10] Zhou Y. C., Wright B. D., Yang R. Y., Xu B. H., and Yu A. B. (1999) - *Rolling friction in the dynamic simulation of sandpile formation*. Physica A: Statistical Mechanics and its Applications, **269**(2-4), 536-553.

[11] https://www.granulair.com/index.html [accessed on 17.09.2018]

[12] Henderson A., Ahrens J., and Law C. (2004) - *The ParaView Guide (Vol. 366)*. Clifton Park, NY: Kitware.

[13] Iwashita K., and Oda M. (2000) - *Micro-deformation mechanism of shear banding process based on modified distinct element method*. Powder Technology, **109**(1-3), 192-205.

[14] https://graphics.stanford.edu/courses/cs205a-13-fall/assets/notes/chapter13.pdf [accessed on 19.09.2018]

[15] Wassgren C., Sarkar A. (2008) - *Discrete Element Method (DEM) Course Module*, https://pharmahub.org/resources/113. [accessed on 20.09.2018]

[16] Mirinavičius A., Markauskas D., and Kačianauskas R. (2010) - *Computational Performance of Contact Search during DEM Simulation of Hopper Filling.*

[17] Munjiza A., and Andrews K. R. F. (1998) - *NBS contact detection algorithm for bodies of similar size.* International Journal for Numerical Methods in Engineering, **43**(1), 131-149.

[18] Munjiza A. - (2004) *The Combined Finite-Discrete Element Method*, John Wiley & Sons, Ltd., Chichester.

[19] Peters J.F., Kala R. and Maier R.S. (2009) - *A hierarchical search algorithm for discrete element method of greatly differing particle sizes.* Engineering Computations, **26**(6), 621-634.

[20] Perkins E. and Williams J.R. (2001) - *A fast contact detection algorithm insensitive to object sizes.* Engineering Computations, **18**, 48-61.

[21] Boon C. W., Houlsby G. T. and Utili S. (2013) - *A new contact detection algorithm for three-dimensional non-spherical particles.* Powder technology, 248, 94-102.

[22] Lin X. and Ng T. T. (1995) - *Contact detection algorithms for threeâĂŘdimensional ellipsoids in discrete element modelling.* International Journal for Numerical and Analytical Methods in Geomechanics, **19**(9), 653-659.

[23] Nezami E. G., Hashash Y. M., Zhao D. and Ghaboussi J. (2004) - *A fast contact detection algorithm for 3-D discrete element method.* Computers and geotechnics, **31**(7), 575-587.

[24] https://www.freemantech.co.uk/_powders/ft4-powder-rheometer-universal-powder-tester [accessed on 20.09.2018]

[25] Wilkinson S. K., Turnbull S. A., Yan Z., Stitt E. H. and Marigo, M. (2017) - *A parametric evaluation of powder flowability using a Freeman rheometer through statistical and sensitivity analysis: A discrete element method (DEM) study.* Computers & Chemical Engineering, 97, 161-174.

[26] Ayachit U. (2015) - *The paraview guide: a parallel visualization application.*

[27] https://www.youtube.com/watch?v=9Ai-U3zXBvk [accessed on 24.09.2018]

[28] http://instrumat.ch/wp-content/uploads/2016/08/MRK1812-01-Optimizing-powder-flow-LRLL.pdf [accessed on 24.09.2018]

[29] Zhou Q., Armstrong B., Larson I., Stewart P. J., and Morton D. A. (2010) - *Effect of host particle size on the modification of powder flow behaviours for lactose monohydrate following dry coating.* Dairy Science & Technology, **90**(2-3), 237-251.

[30] https://www.mathworks.com/matlabcentral/fileexchange/55407-loess-regression-smoothing [accessed on 25.09.2018]

[31] https://en.wikipedia.org/wiki/Discrete_element_method [accessed on 25.09.2018]

[32] Freeman R. (2007) - *Measuring the flow properties of consolidated, conditioned and aerated powders - a comparative study using a powder rheometer and a rotational shear cell*. Powder Technology, **174**(1-2), 25-33.

[33] Bharadwaj R., Ketterhagen W. R. and Hancock B. C. (2010) - *Discrete element simulation study of a Freeman powder rheometer*. Chemical Engineering Science, **65**(21), 5747-5756.

[34] Pantaleev S., Yordanova S., Janda A., Marigo M., and Ooi J. Y. (2017) - *An experimentally validated DEM study of powder mixing in a paddle blade mixer*. Powder Technology, 311, 287-302.

[35] Hare C., Zafar U., Ghadiri M., Freeman T., Clayton J. and Murtagh M. J. (2015) - *Analysis of the dynamics of the FT4 powder rheometer*. Powder Technology, 285, 123-127.

[36] Wilkinson S. K., Turnbull S. A., Yan Z., Stitt E. H. and Marigo M. (2017) - *A parametric evaluation of powder flowability using a Freeman rheometer through statistical and sensitivity analysis: A discrete element method (DEM) study*. Computers & Chemical Engineering, 97, 161-174.

[37] Bharadwaj R., Ketterhagen W. R. and Hancock B. C. (2010) - *Discrete element simulation study of a Freeman powder rheometer*. Chemical Engineering Science, **65**(21), 5747-5756.

[38] https://en.wikipedia.org/wiki/Discrete_element_method [accessed on 05/10/2018]