Università degli Studi di Padova

Dipartimento di Tecnica e gestione dei Sistemi Industriali

Corso di Laurea Magistrale in Ingegneria

dell'Innovazione del Prodotto

The effective elastic stiffness of glass-rubber granular mixtures: numerical study and experimental comparison

Relatori
Prof. Filippo Berto
Prof. Vanessa Magnanimo

Correlatore
MSc. Kianoosh Taghizadeh Bajgirani

Laureando
Giacomo Baccini

Anno Accademico 2015-2016

To my family,
to Francesca,
to Kianoosh and all MSM people

Abstract

Understanding the mechanical behaviour of dense granular materials is important in many fields, such as soil mechanics, material science and physics.
Of particular interest is the elastic regime of granular materials, as related for example with earthquakes problems, landslides initiation or wave propagation. Difficulties in this type of studies arise from the discrete, disordered nature of granular materials. A multi-scale approach is needed in order to fully understand their behavior, as the response at macroscale is related to the kinematics at particle level.
The Discrete Element Method (DEM) is a modern numerical tool that allows to track individual particles. These move following Newton's second law, after microscopic contact properties between particles are defined, along with external displacement/loads. Thanks to the micro-macro transition, the macroscopic, stress-strain behavior of the granular assembly can be inferred from the particle interactions.
In this thesis DEM and its laws are initially presented.
Then polydisperse packings of spheres are prepared by isotropic compression at various levels of volume fraction and confining pressure.
After preparation, the effective elastic moduli are determined per each sample from the incremental stress response to applied strain-probes.
Two types of contact interactions are considered in the normal direction, namely Hookian and Hertzian, coupled with a frictional Coulomb-type contact law for the tangential part of the force.
Unexpectedly, both bulk and shear moduli decrease with increasing inter-particle contact friction, for samples with the same volume fraction. We explain this by differences in the microstructure (isotropic fabric) that characterize the samples state after preparation.
When comparing the cases of linear and Hertzian laws, it is possible to see that for Hertzian laws both bulk and shear moduli have a non-linear dependance with pressure, contrary to linear model.
As further application, we focus on the behavior of granular mixtures, where stiff and soft particles are combined. The influence of the soft-stiff composition on the effective elastic stiffness on the mixture is investigated.
Bulk and shear moduli are linearly scaled with pressure and isostropic fabric and they decrease with increasing of rubber content.
Finally the results are compared with existing experiments of wave propagation in a triaxial cell.

Sommario

Capire il comportamento meccanico di sistemi composti da materiali granulari, con particolare attenzione al regime elastico, è materia di interesse in molti campi quali la meccanica del terreno, la scienza dei materiali e la fisica.
Le caratteristiche che rendono difficoltosa questa operazione sono la discretezza ed il disordine che caratterizza questo tipo di materiali: sarà necessario utilizzare un approccio multi-scala, ossia capire il comportamento del sistema a livello micro per poi passare a quello macro.
Il Metodo agli Elementi Discreti (*Discrete Element Method - DEM*) è uno strumento di calcolo che permette di definire le proprietà di contatto a livello microscopico e di conseguenza, tramite la *micro-macro transition*, il comportamento macroscopico di un assemblato granulare.
In questo lavoro di tesi verranno innanzitutto presentati il Metodo agli Elementi Discreti e le leggi su cui questo si fonda.
Si procederà quindi alla fase di preparazione dei provini numerici: aggregati granulari composti da particelle sferiche caratterizzate da una polidispersità dimensionale saranno portati per mezzo di una compressione isotropica a vari livelli di frazione solida.
A questo punto si passerà alla fase principale dell'indagine: si imporranno al sistema una deformazione infinitesima ed isotropica ed una infinitesima di taglio in modo da poter valutare, tramite la tensione che viene generata, il modulo di compressione volumetrica (*bulk modulus*) ed il modulo di taglio (*shear modulus*). Durante ogni fase di simulazione è importante ricordare che essendo l'attenzione rivolta al comportamento elastico reversibile del sistema, l'ampiezza della perturbazione applicata per eseguire la deformazione deve essere sufficientemente piccola per evitare riarrangiamenti irreversibili del sistema.
Per quanto riguarda l'interazione tra le particelle, per la definizione della forza normale saranno utilizzati due modelli: quello lineare e quello hertziano. In entrambi i casi per la definizione della forza tangenziale verrà utilizzata una legge di tipo coulombiano.
I sistemi che si andranno a investigare inizialmente sono assemblati di particelle dello stesso materiale con valori del coefficiente d'attrito che andranno da zero fino a valori molto elevati, analizzati con il modello lineare. Inaspettatamente, all'aumentare del valore del coefficiente d'attrito, i moduli di compressione volumetrica e di taglio decrescono a parit di frazione solida. Questo può essere collegato alla differenza nella microstruttura che caratterizza i provini numerici dopo la fase di preparazione.
Successivamente si passerà allo studio di provini numerici composti da particelle morbide (gomma) e particelle rigide (vetro), con composizione variabile. Nelle miscele è possibile notare che i moduli elastici sono linearmente proporzionali alla pressione e decrescono al crescere del quantitativo di gomma nel sistema.
I risultati ottenuti questa seconda tipologia di sistema saranno poi confrontati con quelli conseguiti tramite misure sperimentali.

Symbols

k	Normal spring stiffness
x_e	Spring equilibrium lenght
m	Particle mass
ρ	Density
$\vec{r_i}$	Vectiorial postion of i-th particle
$\dot{\vec{r_i}}$	Vectorial velocity of i-th particle
$\ddot{\vec{r_i}}$	Vectorial accleration of i-th particle
$\vec{n_{ij}}$	Unit vector pointing from particle j to particle i
x_i	x position of i-th particle
$\dot{x_i}, v_i, v_{xi}$	x velocity of i-th particle
$\ddot{x_i}$	x acceleration of i-th particle
y_i	y position of i-th particle
v_{yi}	y velocity of i-th particle
$\vec{f_i}$	force exterted on i-th particle
N_p	Number of particles
λ	Spring compression in 2D case
U	Potential energy
E_k	Kinetic energy
E_{tot}	Total energy
t_f	Simulation total time
Δt	Simulation time step
f^n	Linear normal force
γ	Normal viscosity
$\vec{g}$	Gravity acceleration
I_i	Moment of inertia of i-th particle
$\vec{t_i}$	Total torque of a particle
$\vec{\omega_i}$	Particle angula velocity
δ	Overlap
a_i, R_i	Radius of i-th particle
f_i^c	Total contact force acting on particle i
f^n	Linear normal force
t_c	Half-period of a vibration around an equilibrium position
m_{ij}	Reduced mass
ω	Eigenfrequency of contact
e	Coefficient of restitution
f_H^n	Hertz normal force

k_H	Effective Hertz spring stiffness
d_{eff}	Effective diameter
E^*	Effective Young's modulus
K	Hertz non linear stiffness
γ_H	Hertz contact viscosity parameter
η_H	Hertz viscosity parameter
f^t	Tangential force
k_t	Tangential spring stiffness
γ_t	Tangential viscosity
γ_b	Background viscosity
SP	Scaling parameter
ν	Volume fraction
$< r >$	Mean radius
W	Polydispersity
a_{min}	Lowest radius value
a_{max}	Hightest radius value
v_{min}	Lowest velocity value
v_{max}	Hightest velocity value
$\mathbf{E}$	Strain tensor
σ	Stress tensor
V	Volume
$\mathbf{f}^c$	Contact force
$\mathbf{l}^c$	Branch vector
$\mathbf{F}$	Fabric tensor
V^P	Particle volume of particle P
$\mathbf{n}^c$	Normal unit branch-vector
F_v	Isotropic fabric
C	Coordination number
σ_{xx}	Stress in x direction
σ_{yy}	Stress in y direction
σ_{zz}	Stress in z direction
P	Pressure
P^*	Non-dimensional pressure
ε_{xx}	Strain in x direction
ε_{yy}	Strain in y direction
ε_{zz}	Strain in z direction
ε_{vol}	Isotropic strain
B	Bulk modulus
B^*	Non-dimensional bulk modulus
G	Shear modulus
G^*	Non-dimensional shear modulus

Contents

Chapter 1

Introduction

1.1 Granular materials

The term **granular materials** describes a large number of grains or particles acting collectively as an ensamble.
It is easy to find them in our daily life: cereals, coffee (Fig. 1.1), rice (Fig.1.2), spices, asphalt (Fig. 1.3). There are also many examples in nature like sand (Fig. 1.4), snow or even "dust" clouds in space. These are just a few examples everyday life from which one can easily realize the importance of granular materials.

Figure 1.1: Coffee grains

Figure 1.2: Rice grains

Figure 1.3: Asphalt

Figure 1.4: Desert sand

The industial applications of granular materials are equally important and crucial for the society and our civilization: they represent more than the 75 % of all raw materials.

Some examples are the handling of rocks, gravel (Fig 1.5) and sand in mining and construction industry or the transport and stocking (Fig. 1.6) of grains, powders and pills in agricultural or pharmaceutical companies.
Granular materials are the second most manipulated raw materials after fluids [8].

Figure 1.5: Gravel conveyor belt

Figure 1.6: Silo failure. This event is linked to the different distribution of pressure in a granular material than in liquids.

Despite its semplicity and omnipresence, the physics of granular materials is poorly understood and this fact represents a huge blank for geotechnical studies (like, for example, earthquakes propagation) and overall industrial business:

- it has been extimated that abuout 10 % of the world's energy is used in the processing, storage and transport of granular materials [12];
- in a industrial survey, Ennis *et al.* reported that 40 % of the capacity of industrial plants is wasted because of granular solid problems [9];
- Merrow [27] found that the principal cause of long start-up dalays in chemical plants is solids processing, especially the lack of reliable predictive models and simulations [28].

Therefore this topic represent a possibility for innovation and to solve problems in areas as diverse as natural disasters and unsolved industrial material handling issues which incur extensive economic losses, as seen before.
An interesting feature of granular material is the fact that they can behave as solids, liquids or gases, depending on the way the material is driven.
Trying to model these systems with classical continuum theory, standard numerical methods or design tools cannot always be successful because they ignore the fact that the assemblies consist of discrete objects.
Discrete Element Method (Chap. 3) is a perfect tool to simulate the microscopic evolution of a particle system. The procedure that permits to bridge microscopic system with macroscopic is called "micro-macro transition".

1.2 Elastic moduli

Behaviour of granular materials is highly non-linear and involves irreversibilty (plasticity), possibly already at very small strains, due to rearrangements of the elementary particles

[3, 10, 17, 33].
Many industrial and geotechnical applications that are crucial for our society involve granular system at small strain levels. That is the case of structure designed to be far from failure (*e.g.* shallow foundations or underlying infrastructure), strains in the soil are small and a sound knowledge of the bulk stiffness is essential for the realistic prediction of ground movements [6].
Finite-element analyses of tunnel depend on the model adopted for the pre-failure soil behaviour. When the surface settlement is considered, the importance of modelling non-linear elasticity and the shear modulus characterization become of outmost importance [2].
When looking at natural flows, a complete description of the granular rheology should include an elastic regime [5], so understending deeper the evolution of bulk modulus is a fundamental topic.

1.3 Mixtures

Mixtures have become a key research topic in recent years due to its wide range of applications in engineering. An interesting example is sand-rubber mixture [15, 38] in geotechnical applications:the reuse of waste rubber tyres creates a win-win situation whereby waste rubber tyres which are non-biodegradable are given a new lease of life.
The use of mixtures can change properties of a grandular assemble, like reduce weight of a system but improve the stiffness at the same time: actually it is an open and interesting research.

1.4 Goal

The aim of this thesis work is to investigate the elastic response of a granular system, performing so-called strain probing test along and isotropic deformation (pre-strain) path [17, 25]. In the case of a finite assembly of particles, in simulations, a finite elastic regime can always be detected and the elastic stiffnesses can thus be measured by means of an applied very small strain perturbation.
Firstly a wide range of inter-particle friction and volume fractions is scanned for one material system using linear normal contact force law, in order to understand how the interplay of contact and system properties affects the microstructure and thus the elastic moduli (Chap. 5.3).
Successively one material system is simulated using Hertz contact force law (Chap. 5.4), to apreciate the difference between models.
At conclusion systems composed of various quantities of glass and rubber togheter are scanned using Hertz model (Chap. 6). Results of this last part are compared to experimental ones.

Chapter 2

Introductive Models

DEM method is composed by two phases: definition of force acting on a particle due to the interaction with other ones and then definition of paticles motion.
In this chapter we present some introductive cases based on molecular dynamics.

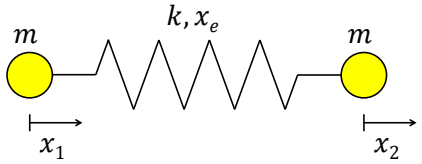
2.1 Molecular Dynamics

Molecular dynamics attempts to model reality as a collection of particles obeying Newtons laws. It is therefore assumed that the motion of particles satisfies [1] :

$$m\ddot{\vec{r}}_i = \vec{f}_i \tag{2.1}$$

where m_i is the mass of the $i-th$ particle with position r_i subject to force f_i.

2.2 Two particles connected with a spring 1D



The figure displays two particles that can move in one dimention connected with a spring. The positions of the particles are x_1 and x_2, they both have mass m and the spring has stiffness k and equilibrium length x_e. Its possible to write down Eq. (2.1) for each particle:

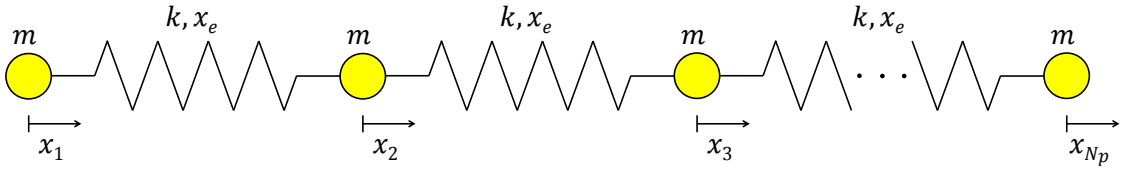
$$m\ddot{x}_i = f_i \qquad\qquad i = 1, 2 \tag{2.2}$$

where

$$f_1 = k[(x_2 - x_1) - x_e],$$
$$f_2 = -k[(x_2 - x_1) - x_e].$$

Given initial positions $x_i(0)$ and velocities $v_i(0)$, $i = 1, 2$, we can use an integration scheme such as Verlet [4] to obtain positions and velocities of the two particles at discrete times $t(n)$, $t(n) = n\Delta t$.

2.3 N_p particles connected with a spring 1D



System of N_p particles, connected with springs, that can move in one direction. The positions of the particles are x_i, $i = 1, \ldots, N_p$; all particles have mass m and the springs have stiffness k and equilibrium lenght x_e. We can write down the equations of motions for each particle and thanks to this we can calculate force acting on every sphere:

$$m\ddot{x}_i = f_i \qquad\qquad i = 1, \ldots, N_p \tag{2.3}$$

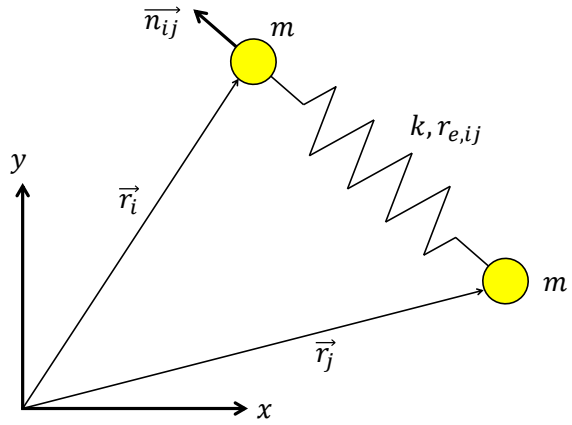
where

$$\begin{aligned}
f_1 &= k[(x_2 - x_1) - x_e], \\
f_2 &= -k[(x_2 - x_1) - x_e] + k[(x_3 - x_2) - x_e], \\
f_3 &= -k[(x_3 - x_2) - x_e] + k[(x_4 - x_3) - x_e], \\
&\vdots \\
f_2 &= -k[(x_2 - x_1) - x_e].
\end{aligned}$$

The expression can be written in a more concisely way:

$$f_i = \begin{cases} k[(x_{i+1} - x_i) - x_e] & i = 1, \\ k[x_{i+1} - 2x_i + x_{i-1}] & 1 < i < N_p, \\ -k[(x_{i+1} - x_i) - x_e] & i = N_p. \end{cases}$$

2.4 Two particles connected with a spring 2D



The magnitude of the force exerted by the spring on the particles, f_{ij}, can be computed by multiplying the spring stiffness, k, with the spring compression, λ. The compression is given by substracting the equilibrium length of the spring between particles i and j, $r_{e,ij}$, from the length of the vector pointing from particle i to particle j, $|r_i - r_j|$ and multiplying this with -1.

$$f_{ij} = k\lambda, \qquad \text{with} \quad \lambda = -(|\vec{r_i} - \vec{r_j}| - r_{e,ij}) \tag{2.4}$$

Note that a positive f_{ij} means the spring is in compression whereas a negative magnitude of the force means the spring is in tension. The force exerted on particle i by the spring between particles i and j,f_{ij}, can be calculated by multiplying the magnitude of the spring force, f_{ij}, with the unit vector pointing to particle i from particle j:

$$\vec{f_{ij}} = f_{ij}\vec{n_{ij}},$$

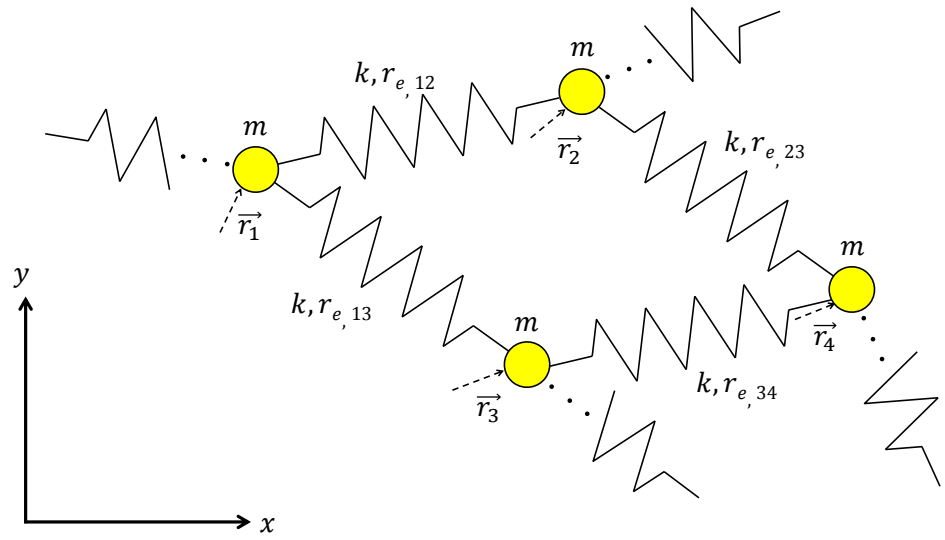
where the unit vector pointing to particle i from particle j is given by the vector pointing from particle i to particle j, divided by its length:

$$n_{ij} = \frac{\vec{r_i} - \vec{r_j}}{|\vec{r_i} - \vec{r_j}|}$$

The force exerted on particle j by the spring between particles i and j, $\vec{f_{ji}}$, is equal in magnitude and opposite in direction to $\vec{f_{ij}}$:

$$\vec{f_{ji}} = -\vec{f_{ij}}$$

2.5 N_p particles connected with a spring 2D



We can express the force on particle i, f_i, as:

$$\sum_{j=1}^{N_p} C_{ij} \vec{f_{ij}} \tag{2.5}$$

$\vec{f_{ij}}$ is the force exerted by the spring connecting particle i and j on particle i, given by Eq. (2.4). C_{ij} is given by:

$$C_{ij} = \begin{cases} 1, & \text{if particles } i \text{ and } j \text{ are connected,} \\ 0, & \text{if particles } i \text{ and } j \text{ are not connected.} \end{cases}$$

2.6 Conservation of energy

In conservative systems such as the ones considered so far the law of conservation of energy should hold. The total energy, E_{tot}, is preserved in time and changes form between potential, U, and kinetic energy, E_k. For particles connected with springs the potential energy is found by summing the potential energy stored in each spring:

$$U = \sum_{i=1}^{N_p-1} \frac{1}{2} k[(x_{i+1} - x_i) - x_e]^2 \qquad \text{for } N_p \text{ particles in 1D,} \tag{2.6}$$

$$U = \sum_{i=1}^{N_p-1} \sum_{j=i+1}^{N_p} C_{ij} \frac{1}{2} k(|\vec{r_i} - \vec{r_j}| - r_{eq,ij})^2 \qquad \text{for } N_p \text{ particles in 2D,} \tag{2.7}$$

The kinetic energy can be computed by summing the kinetic energy of all particles:

$$E_k = \sum_{i=1}^{N_p} \frac{1}{2} m v_i^2 \qquad \text{for } N_p \text{ particles in 1D,} \tag{2.8}$$

$$E_k = \sum_{i=1}^{N_p} \frac{1}{2} m |v_i|^2 \qquad \text{for } N_p \text{ particles in 2D,} \tag{2.9}$$

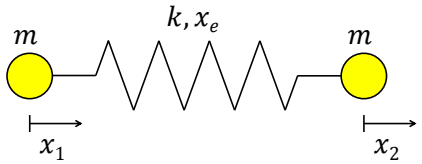
The total energy Etot is the sum of the kinetic and potential energy:

$$E_{tot} = E_k + U \tag{2.10}$$

2.7 Examples

In the following we report some examples to show how to apply interaction laws previously described.

2.7.1 Two masses connected with a spring 1D



We consider two particles with the same mass $m = 1$ connected by a spring of stiffness $k = 30$ and equilibrium lenght $x_e = 10$. The distance between the particles is $x_e + 1$. At the beginning the velocity of particle 1 is 0 and velocity of particle 2 is 1.5. In Figs. 2.1 - 2.5 we plot the evolution of position, velocity, accleration, force and total energy.
We use a Velvet algorithm to solve the problem.
For the solution script see A.1.1.

System parameters		
Mass	m	1
Stiffness	k	30
Equilibrium lenght	x_e	10
Initial conditions		
Position1	x_1	0
Position2	x_2	x_e+1
Velocity1	v_1	0
Velocity2	v_2	1.5
Simulation parameters		
Total time	t_f	1.5
Time step	Δt	0.01

Results

The system is conservative, so in Fig. 2.5 is possible to apreciate conservation of total energy as seen in Fig. 2.6.

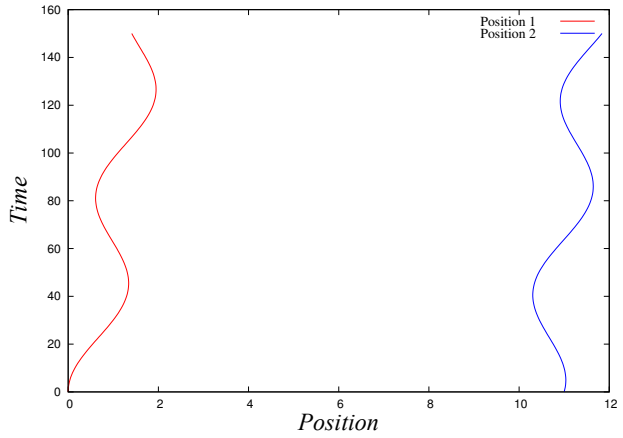


Figure 2.1: Time vs position

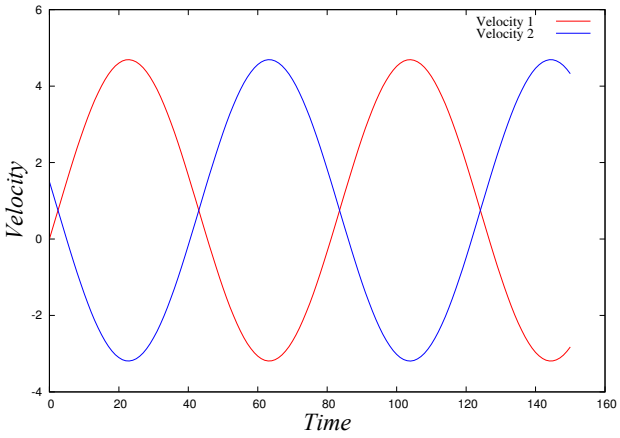


Figure 2.2: Velocity vs time

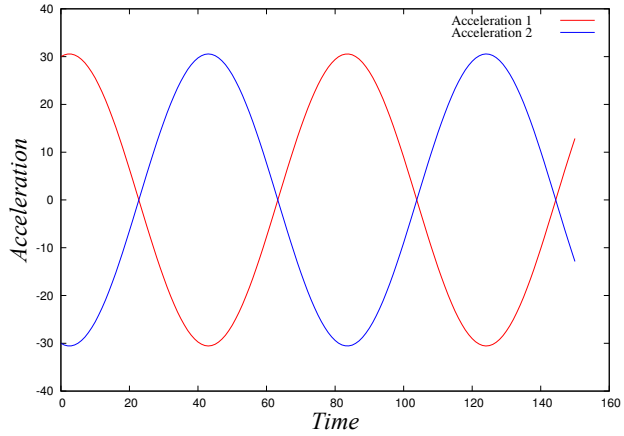


Figure 2.3: Acceleration vs time

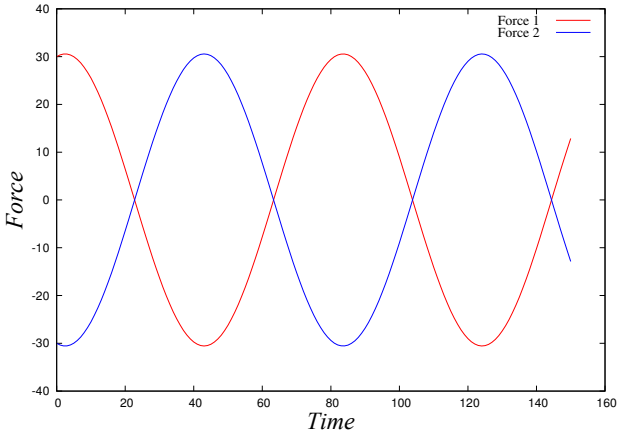


Figure 2.4: Force vs time

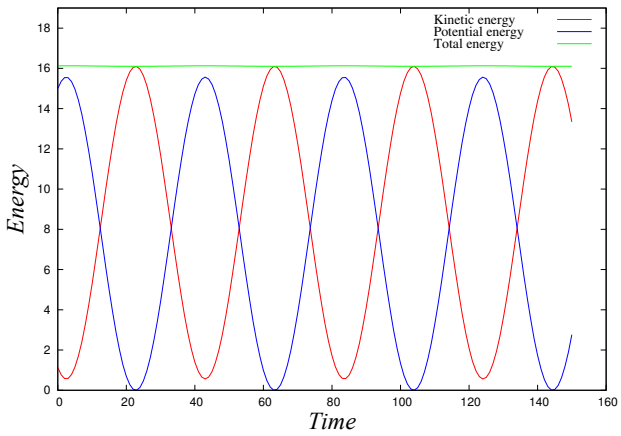


Figure 2.5: Energy vs time

Force analitic solution,$F_{1,an}$, is found multiplying the relative position (x_{rel}) between particles, calculated using solution script (A.1.1), with the stiffness k of the spring: $F_{1,an} =$

$k \cdot x_{rel}$.
We can compare script and analitical results and apreciate that perfectly fit. This means that script solution is correct.

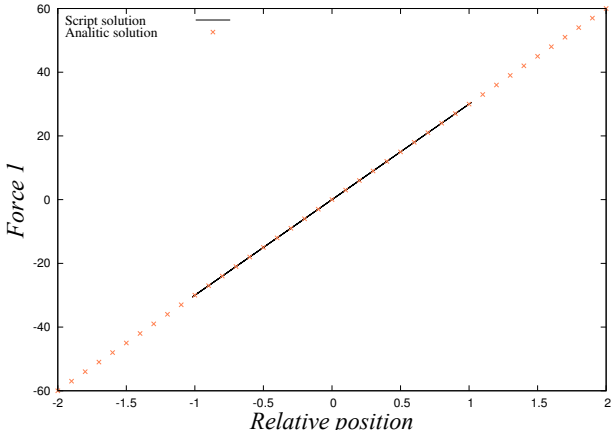
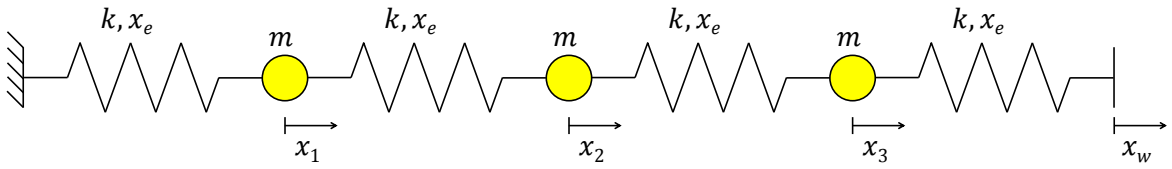


Figure 2.6: Script vs analitic force

2.7.2 Three masses connected with springs 1D



We consider a linear chain of 3 particles of mass $m = 1$ connected to each other by a spring of stiffness $k = 1$. The distance between the particles at the beginning is $x_e = 2$. The first and the last particle are connected to a wall: the one on the left is fixed, the other oneis moved very slowly by a distance $x_w = x_e$ from its equilibrium position and then fixed in that position. In Figs. 2.7 - 2.11 we plot the evolution of position, velocity, accleration, force and total energy. We use a Velvet algorithm to solve the problem.
For the solution script see A.1.2.

System parameters		
Mass	m	1
Stiffness	k	1
Equilibrium lenght	x_e	2
Initial conditions		
Position1	x_1	x_e
Position2	x_2	$2x_e$
Position3	x_3	$3x_e$
Velocity1	v_1	0
Velocity2	v_2	0
Velocity3	v_3	0
Simulation parameters		
Total time	t_f	15
Time step	Δt	0.01

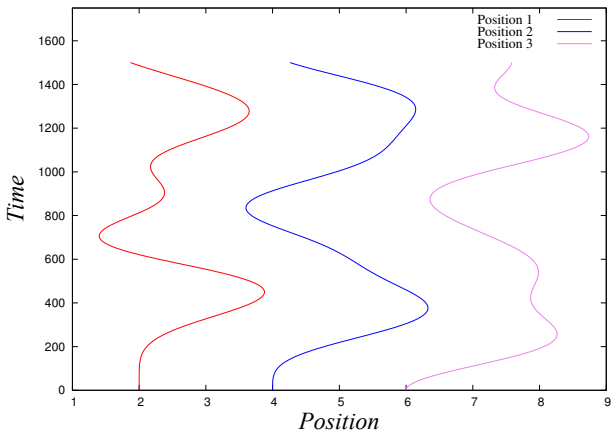


Figure 2.7: Time vs position

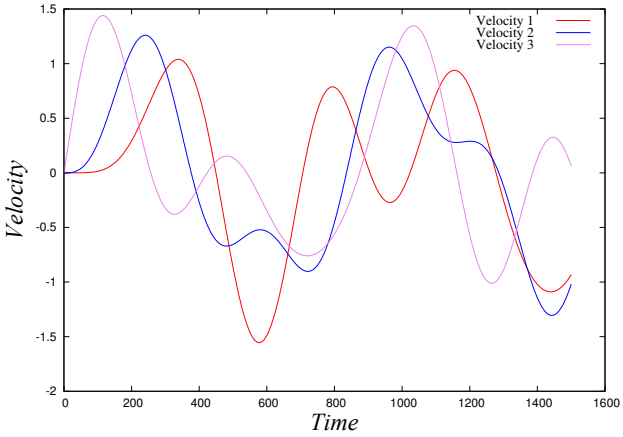


Figure 2.8: Velocity vs time

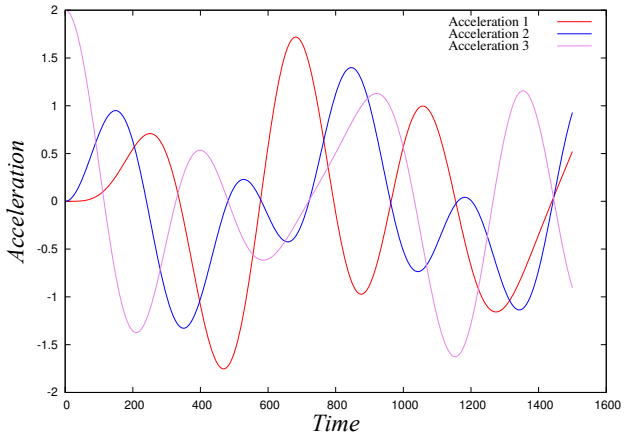


Figure 2.9: Acceleration vs time

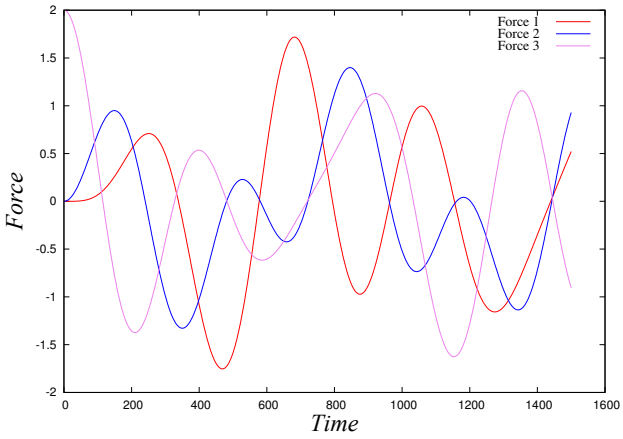


Figure 2.10: Force vs time

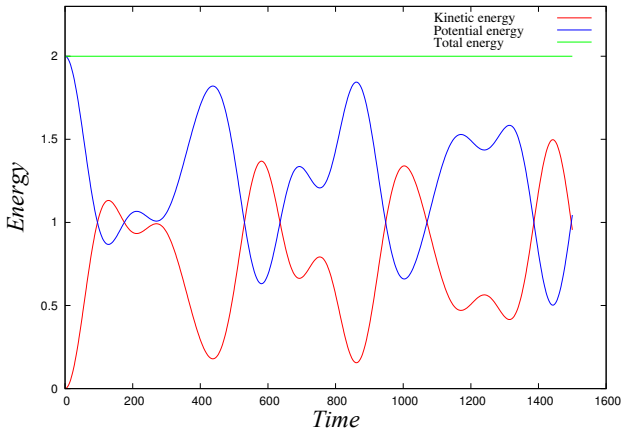
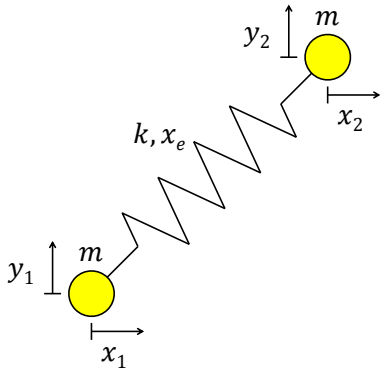


Figure 2.11: Energy vs time

2.7.3 Two masses connected with a spring 2D



We consider two particles with the same mass $m = 1$ connected by a spring of stiffness $k = 1$ and equilibrium lenght $x_e = 10$. The particles move in 2 directions on the plane xy. Initial positions are $x_1 = 0$ and $y_1 = 0$ for particle 1 $x_2 = x_e$ and $y_2 = x_e$ for particle 2. At the beginning the velocity of particle 1 is $v_{x1} = 0$ and $v_{y1} = 0$ and velocity of particle 2 is $v_{x2} = 1$ and $v_{y2} = 1$. In Figs. 2.12 - 2.16 we plot the evolution of position, velocity, accleration, force and total energy. We use a Velvet algorithm to solve the problem.
For the solution script see A.1.3.

System parameters		
Mass	m	1
Stiffness	k	1
Equilibrium lenght	x_e	10
Initial conditions		
Position1x	x_1	0
Position1y	y_1	0
Position2x	x_2	x_e
Position2y	y_2	x_e
Velocity1x	v_{x1}	0
Velocity1y	v_{y1}	0
Velocity2x	v_{x2}	1
Velocity2y	v_{y2}	1
Simulation parameters		
Total time	t_f	10
Time step	Δt	0.01

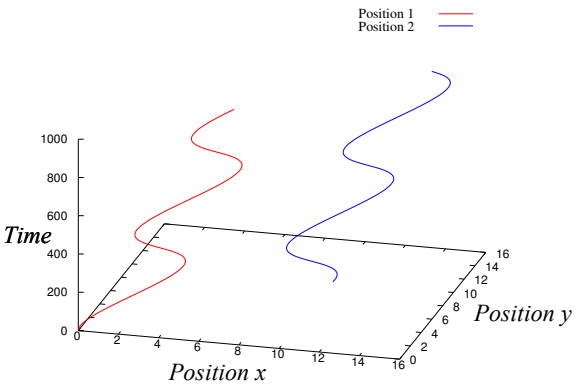


Figure 2.12: Time vs position

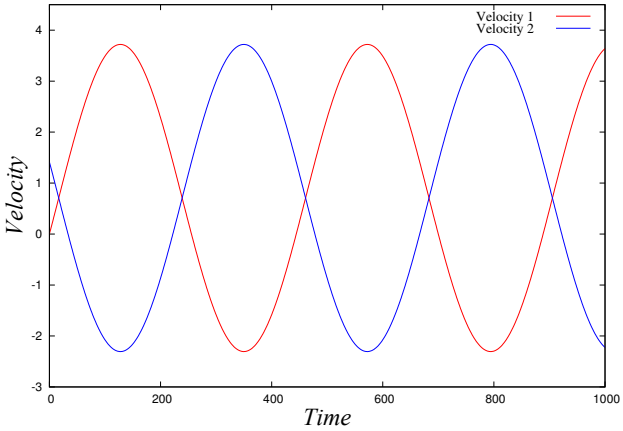


Figure 2.13: Velocity vs time

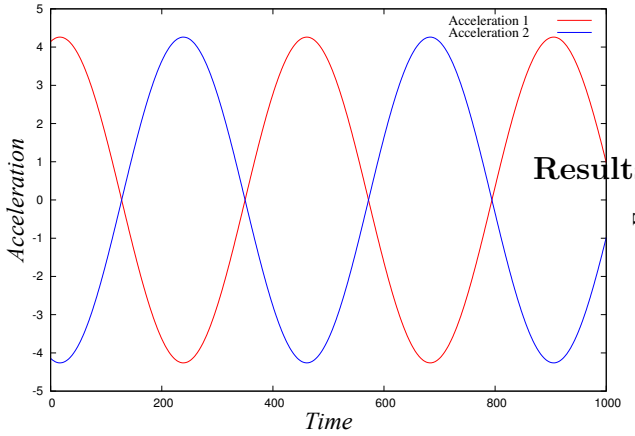


Figure 2.14: Acceleration vs time

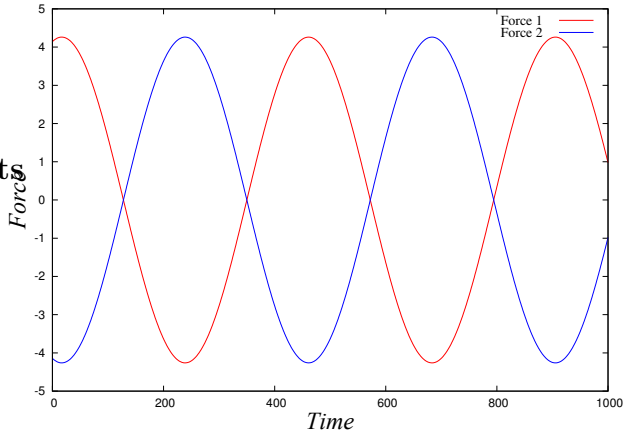


Figure 2.15: Force vs time

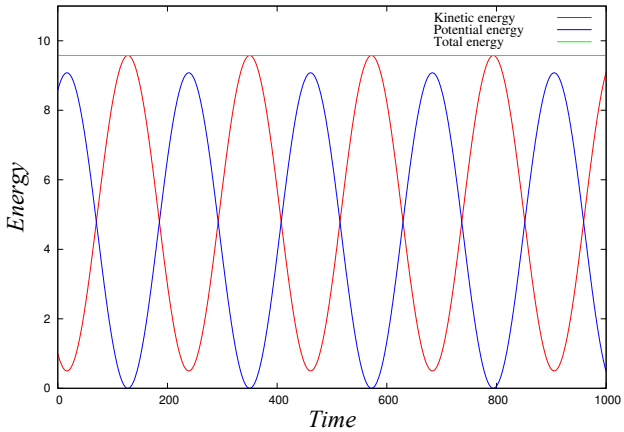
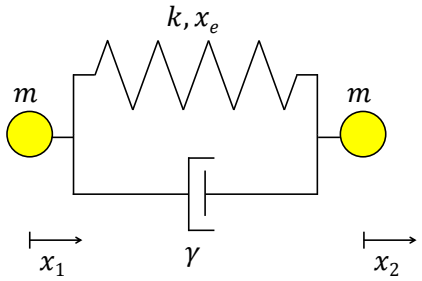


Figure 2.16: Energy vs time

2.8 Damped harmonic oscillator 1D



The model that best suites to simulate the normal contact force of collision of two particles (see Chap. 3) is the damped harmonic oscillator.

It's possible to write Eq. (2.1) for each particle:

$$m\ddot{x}_i = f_i \qquad\qquad i = 1, 2$$

where

$$f_1 = k[(x_2 - x_1) - x_e] + \gamma(v_2 - v_1),$$
$$f_2 = -k[((x_2 - x_1) - x_e) + \gamma(v_2 - v_1].$$

k is the stiffness of the spring, γ is the viscosity of the damper. Given initial positions $x_i(0)$ and velocities $v_i(0)$, $i = 1, 2$, we can use the laws of uniformly acceleratd motion to obtain the positions and velicities of the two particles at discrete times $t(n)$, $t(n) = n\Delta t$.

2.9 Damped harmonic oscillator 2D

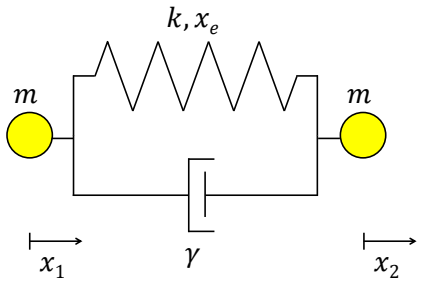
For the two dimensions behaviour decription we use the same procedure seen in Sec. 2.7.3.

2.10 Non-conservation of energy

The damper is a non-conservative element: the total energy, defined in Eq. (2.10) of the system will decrease.

2.11 Examples

2.11.1 Two masses with a spring and a damper



We consider a damped harmonic oscillator composed by two particles with the same mass $m = 1$. The spring has stiffness $k = 30$ and equilibrium lenght $x_e = 10$, the damper has viscosity $\gamma = 1$ The distance between the particles is $x_e + 1$. At the beginning the velocity of particle 1 is 0 and velocity of particle 2 is 1.5.
In Figs. 2.17 - 2.21 we plot the evolution of position, velocity, accleration, force and total energy.
For the solution script see A.2.1.

System parameters		
Mass	m	1
Stiffness	k	30
Equilibrium lenght	x_e	10
Viscosity	γ	1
Initial conditions		
Position1	x_1	0
Position2	x_2	$x_e + 1$
Velocity1	v_1	0
Velocity2	v_2	1.5
Simulation parameters		
Total time	t_f	10
Time step	Δt	0.01

Results

As said in Sec. 2.10 in this system there is dissaption of energy: during the time the physiscal quantities under investigation will decrease (see Fig. 2.21).

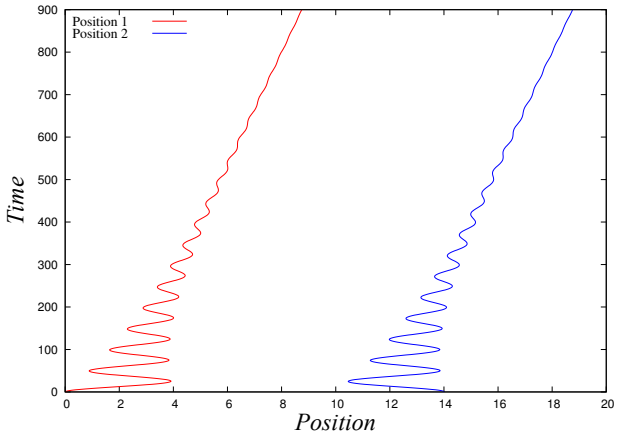


Figure 2.17: Time vs position

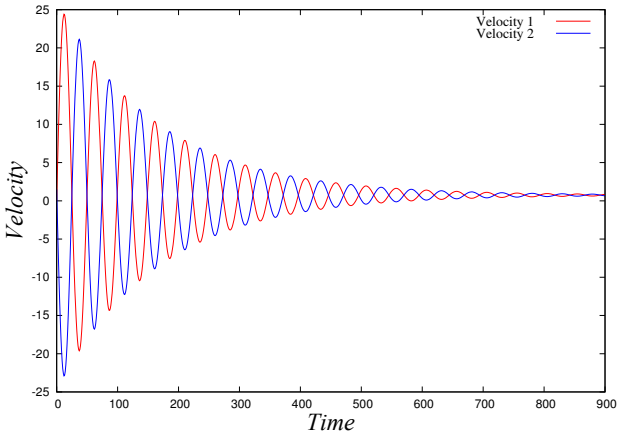


Figure 2.18: Velocity vs time

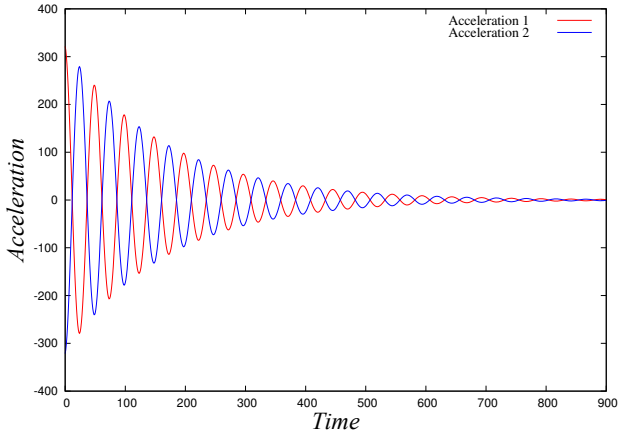


Figure 2.19: Acceleration vs time

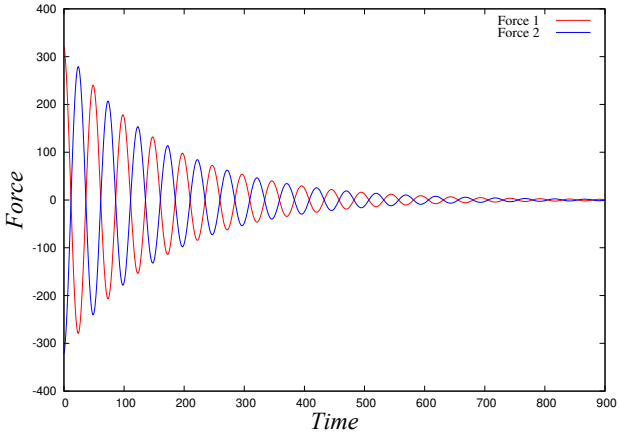


Figure 2.20: Force vs time

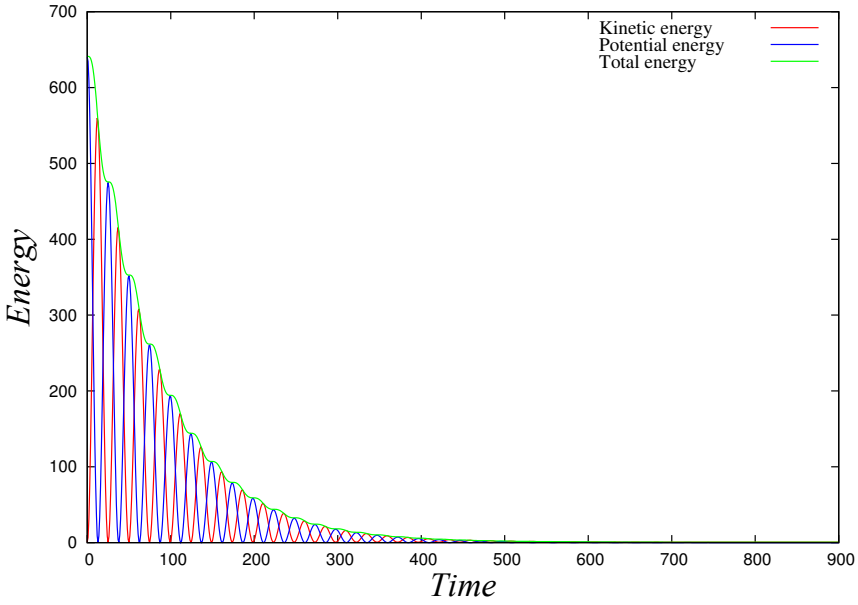


Figure 2.21: Energy vs time

Chapter 3

Discrete Element Method

The approach towards the microscopic understanding of a macroscopic particulate material behaviour is the modelling of particles using so-called discrete element methods (DEM), numerical scheme originally formulated and developed by Cundall *et al.* [7]. Many research works [31, 37, 40, 41] verified the usefulness of this model. Even though millions of particles can be simulated, the possible length of such a particle system is in general too small in order to regard it as microscopic: will be necessary the definition of methods and tools to perform a so-called micro-macro transition. These "microscopic" simulation of a small sample (Representative Volume Element, REV) can be used to derive macroscopic constitutive relations needed to describe the material within the framework of a macroscopic continuum theory. In the particular case of a solid in static equilibrium the REV can be assumed to be homogeneous. It represents a material point in the elastic Cauchy continuum.

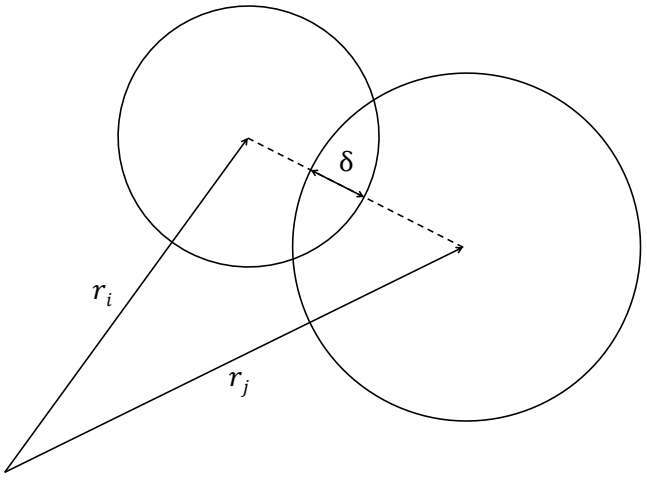


Figure 3.1: Two particles contact with overlap δ

Discrete Element Model (DEM) method is a straightfoward implementation to solve the equations of motion for a system of many interacting particles. For DEM, both normal and tangential interactions are considered for spherical particles. The elementary units of granular material are mesoscopic grains which deform under stress. Since the realistic

modelling of the deformation of the particles is much too complicated, we relate the interaction force to the overlap δ of two particles (Fig. 3.1). Note that the evaluation of the inter-particle forces based on the overlap may not be sufficient to account for the inhomogeneous stress distribution inside the particles.

3.1 Equation of motion

The discrete element model is based on the laws described in 2.1, adding Newton's equations of motion for the rotational degrees of freedom (each particle has tanslational and rotational motion) [22]:

$$m\ddot{\vec{r}}_i = \vec{f}_i + m_i\vec{g} \qquad\qquad I_i\dot{\vec{\omega}}_i = \vec{t}_i \tag{3.1}$$

where m_i is the mass of the $i-th$ particle with position r_i subject to two kind of forces: one due to contacts with other particles ($\vec{f}_i = \sum_c \vec{f}_i^c$) and one due volume (gravity acceleration, $\vec{g}$). I_i is the spherical particles moment of inertia, $\dot{\omega}_i$ its angular velocity and $\vec{t}_i = \sum_c(\vec{l}_i^c + \vec{f}_i^c + \vec{q}_i^c)$ is the total torque, where $\vec{q}_i^c$ are torques/couples at contacts other than due a tangential force, *e.g.*, due to rolling and torsion.

3.2 Contact force laws

Two spherical particles, i with radius a_i and j with radius a_j, interact only if they are in contact. This means their overlap is positive:

$$\delta = (a_i + a_j) - (\vec{r_i} - \vec{r_j}) \cdot \vec{n} > 0, \tag{3.2}$$

where $\vec{n} = \vec{n_{ij}} = (\vec{r_i} - \vec{r_j})/|\vec{r_i} - \vec{r_j}|$ is the unit vector pointing from j to i. The total force acting on particle i due to the contact c with particle j is $\vec{f}^c$ and it can be decomposed in a normal and in a tangential component:

$$\vec{f}^c := \vec{f}_i^c = f^n\vec{n} + f^t\vec{t}.$$

3.2.1 Linear normal contact force law

For the definition of normal contact force the simplest model available is the linear normal contact model, which considers a linear repulsive and a linear dissipative force:

$$f^n = k\delta + \gamma v_n. \tag{3.3}$$

This model is a damped harmonic oscillator (as seen previously in 2.8) composed of a spring with stiffness k_n and a damper with viscosity γ. v_n is the relative velocity in normal direction: $v_n = -\vec{v_{ij}} \cdot \vec{n} = -(\vec{v_i} - \vec{v_j}) \cdot \vec{n} = \dot{\delta}$. An important advantage of using the damped harmonic oscillator is that the half-period of a vibration around an equilibrium position can be computed:

$$t_c = \frac{\pi}{\omega} \qquad\qquad \text{with } \omega = \sqrt{(k/m_{ij}) - \eta_0^2} \tag{3.4}$$

where ω is the eigenfrequency of the contact, $\eta_0 = \gamma_0/(2m_{ij})$ is the rescaled damping coefficient and $m_{ij} = m_i m_j/(m_i + m_j)$ is the reduced mass. Solving (3.4) is possible to obtain the coefficient of restitution

$$e = -v_n'/v_n = \exp(-\pi\eta_0/\omega) = \exp(-\eta_0 t_c) \tag{3.5}$$

which quantifies the ratio of relative velocities after and before the collision.

The half-period t_c is also important by an operative point of view: the integration of the equations of motion is stable only if the integration time-step Δt_{DEM} is much smaller than t_c. A consequent operative advice is to use a dissipation coefficient γ_0 neither too weak nor too strong: in the first case t_c will be too large, in the second too short.

3.2.2 Hertzian normal contact force law

Hertzian normal contact force law without damping

In order to formulate a more refined force than Eq. (3.3), one can use the results of the Hertz theory of elastic contact [11, 14, 19, 26, 32] :

$$f_H^n = k_H \delta^{3/2} \tag{3.6}$$

where $k_H = \sqrt{d_{eff}} E^*$ is the effective spring stiffness, with $d_{eff} = (2d_i d_j)/(d_i + d_j) = (4R_i R_j)/(R_i + R_j)$ as effective diameter and E^* effective Young's modulus, defined as (keeping in mind that $E = 2G(1+\nu)$):

$$E^* = \begin{cases} \dfrac{E}{3(1-\nu^2)} = \dfrac{2G}{3(1-\nu^2)} & \text{for single species i=j,} \\ \left[\dfrac{3}{2}\left(\dfrac{1-\nu_i^2}{E_i} + \dfrac{1-\nu_j^2}{E_j}\right)\right]^{-1} = \left[\dfrac{3}{2}\left(\dfrac{1-\nu_i}{2G_i} + \dfrac{1-\nu_j}{2G_j}\right)\right]^{-1} & \text{for i} \neq \text{j.} \end{cases} \tag{3.7}$$

Eq. (3.6) can be also written as:

$$f_H^n = K(\delta)\delta, \tag{3.8}$$

with $K = \sqrt{d_{eff}\delta} E^*$ non linear stiffness dependent on the compression level trough the overlap δ.

Note that with Eq. (3.6), the collision time t_c is no longer independent of v_n^i [19]:

$$t_c = 3.21 \left(\frac{m_{ij}}{k_H}\right)^{2/5} \left(v_n^1\right)^{-1/5} \tag{3.9}$$

This means that there is no intrinsic timescale to collisions. The choice of the numerical time step Δt must depend on the maximum relative velocities expected during the simulation to ensure satisfactory numerical accuracy.

Hertzian energy

The kinetic energy of the system, as seen in Eq. 2.8 and Eq. 2.9, is the sum over all single particle kinetic energy.

Potential energy for Hertz is defined as:

$$U = \frac{2}{5}\sum_{c=1}^{M}\sqrt{d_{eff}}E^{*}\delta^{5/2} = \frac{2}{5}\sum_{c=1}^{M}k_H\delta^{5/2} = \frac{2}{5}\sum_{c=1}^{M}K\delta^{2} \tag{3.10}$$

where M is the total number of contacts c.

Total energy is the sum of kinetic and potential energies, as defined in Eq. (2.10).

Hertzian normal contact force law with damping - first definition

In order to obtain a dissipative Hertz-type force, a viscous damping term was added to the Hertz force in an *ad hoc* fashion in some studies [20, 29, 30] :

$$f_H^n = k_H\delta^{3/2} + \gamma_n\dot{\delta} \tag{3.11}$$

However, as Taguchi [36] pointed out, this force leads to collisions that become more elastic as the impact velocity increases, contrary to the experimental evidence: $(1-e) \propto v^{i-1/5}$ [23]. For low impacts velocities, where the Hertz results for elastic contacts should be regained, force 3.11 produces a coefficient of restitution that approaches zero.

Hertzian normal contact force law with damping - second definition

Kuwabara and Kono [18] and Brillantov *et al.* [34] extended the original Hertz approach assuming the material to be viscoelastic instead of elastic. They derive:

$$f_H^n = k_H\delta^{3/2} + \gamma_H\delta^{1/2}\dot{\delta} \tag{3.12}$$

where k_H is the effective spring stiffness seen in Eq. 3.6 and $\gamma_H = \eta_H\sqrt{d_{eff}}$ is the respective contact viscosity parameter, with η_H Hertz viscosity parameter.

3.2.3 Plastic deformation normal contact force law

An approach guided by the picture of plastic deformation was presented by Walton and Braun [39]. They assumed that thre are different spring constants, k_1 and k_2, for the loading and unloading part of the contact:

$$f^n = \begin{cases} k_1\delta & \dot{\delta} \geq 0 \text{ (loading)} \\ k_2(\delta - \delta_0) & \dot{\delta} < 0 \text{ (unloading)} \end{cases} \tag{3.13}$$

where δ_0 is the value of δ where the unloading curve intersects the abscissa under the given circumstances, or the permanent plastic deformation. In this model $e = \sqrt{k_1/k_2}$. This model is presented for completeness, but will be never used in this thesis work.

3.2.4 Tangential contact force laws

Tangential forces are linked to friction, which is generated by the relative motion of the two particles in contact with each other: this is a source of energy dissipation. Friction could be static or dynamic and it is modeled according to the Coulomb friction law:

$$\begin{aligned} f_c^s &= \mu_s f^n && \text{for static friction} \\ f_c^d &= \mu_d f^n && \text{for dynamic friction} \end{aligned} \tag{3.14}$$

where (μ_s) is static and (μ_d) dynamic friction coefficient.
Static friction occurs when two particles do not involve micro-slip at the contact surface. In the case of static friction, the friction force (tangential force) f^t exerted between the surfaces of two particles where the particles having no relative motion can not exceed the value given by Coulomb's law using static friction coefficient (i.e., $f^t \leq \mu_s f^n$).
A the linear visco-elastic contact model is used for the tangential direction component of force:

$$f^t = k_t \delta^t + \gamma_t \dot{\delta}^t \tag{3.15}$$

where k_t is the tangential spring stiffness , γ_t is the tangential contact viscosity parameter, δ^t is the displacement in tangential direction and $\dot{\delta}^t$ is the relative velocity in the tangential direction [21].
However, kinetic friction happens when the tangential component of force is exceeding the maximum value of static force ($\mu_s f^n$), hence two particles start to slide against each other.

3.2.5 Background friction

The dissipation mode seen until now is suitable for a two paticles system, but in the bulk material, where there are many particles interacting with each other, it is very inefficient for long-wavelenght cooperative modes of motion. Therefore, there is the necessity to define an additional damping with the background, so the total force acting on particle i is

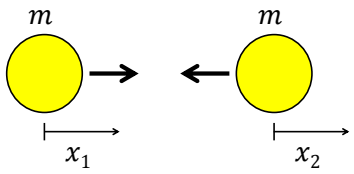
$$\vec{f_i} = \sum_j (f^n \vec{n} + f^t \vec{t}) - \gamma_b \vec{v_i} \tag{3.16}$$

with the background viscosity γ_b.

3.3 Examples

In this section we study the collision between two particles in one and two directions, with and without damping. All these examples will be solved using linear normal contact force law, one comparison between linear and hertzian model will be done in Sec. 3.3.5.

3.3.1 Two particles 1D without damping



We consider two particles with the same mass $m = 1$ and the same radius $R_1 = R_2 = 10$. Material has stiffness $k = 1$ and there is no damping or friction. Initial positions are $x_1 = 0$ for particle 1 and $x_2 = 50$ for particle 2. At the beginning velocities are $v_1 = 2$ (particle 1) and $v_2 = -2$ (particle 2).
In Figs. 3.2 - 3.19 we plot the evolution of position, velocity, acceleration, force and total energy, evaluated using the normal contact force law (3.3).
For the solution script see A.3.1.

System parameters		
Mass	m	1
Radius particle 1	R_1	10
Radius particle 2	R_2	10
Stiffness	k	1
Initial conditions		
Position1x	x_1	0
Position2x	x_2	55.5
Velocity1x	v_{x1}	2
Velocity2x	v_{x2}	-2
Simulation parameters		
Total time	t_f	20
Time step	Δt	10^{-3}

Results

In Fig. 3.2 it is possible to appreciate the trajectory of the centre of mass of the two particles and the hightest value of overlap δ.
Because of the absence of damping, the two particles exchange all their own velocities after the impact (Fig. 3.3).
Acceleration (Fig. 3.4 and force (Fig. 3.5) have the typical behaviour of impulsive phenomena, like impact between two particles.
The system is conservative: there is not dissipation of total energy (see 3.19), that, except the impact interval, coincide with kinetic energy.

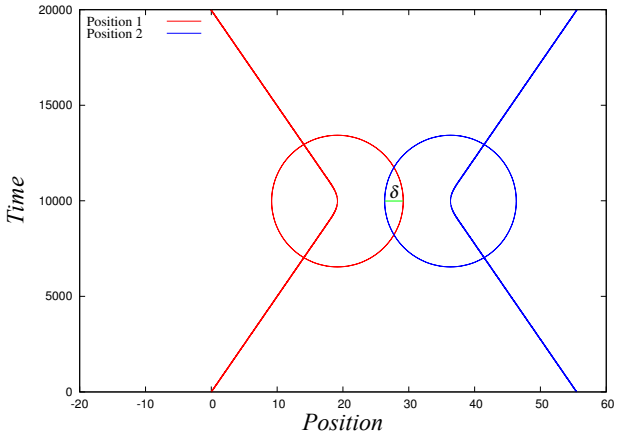


Figure 3.2: Time vs position

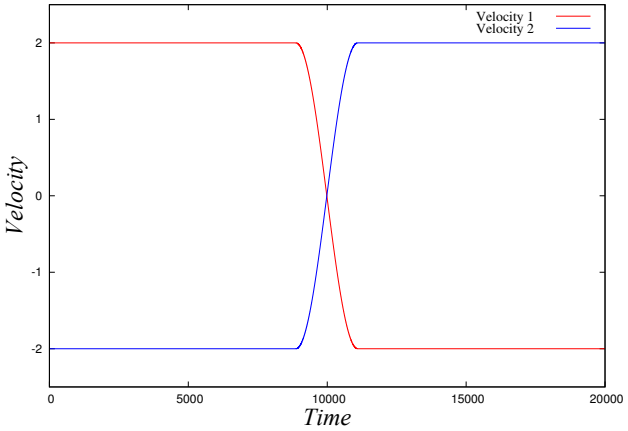


Figure 3.3: Velocity vs time

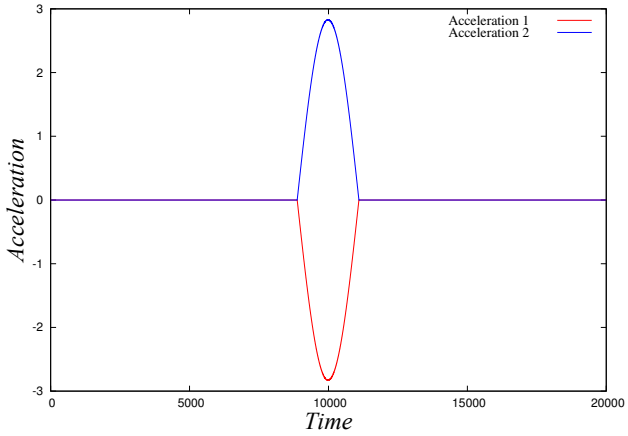


Figure 3.4: Acceleration vs time

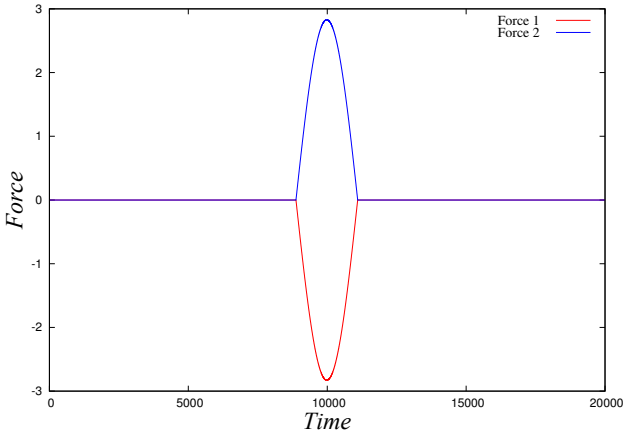


Figure 3.5: Force vs time

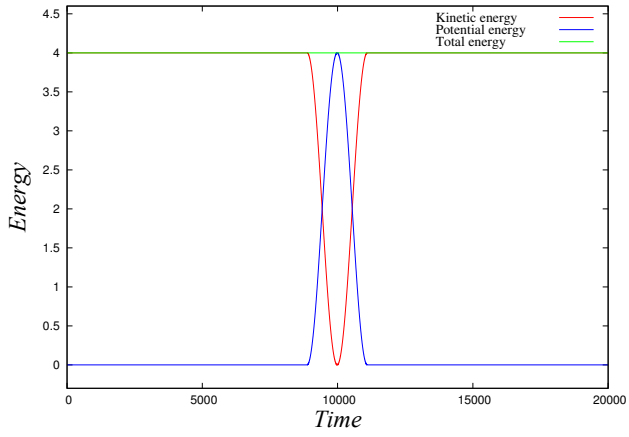


Figure 3.6: Energy vs time

It is interesting to compare the script solution of the problem with the analitic one (see Fig. 3.7 and Fig. 3.8), calculated starting from the evaluation of the starting and ending point of the contact obtained thanks to my own simple DEM code. Between these points we define two linear vectors,$x_{an,2}$ and $x_{an,1}$, and so it is possible to eavluate the analitic

solution of the problem:

$$F_{an} = k \cdot \delta_{an} = k \cdot [(R_1 + R_2) - (x_{an,2} - x_{an,1})].$$

Analitic solution is calculated for compression and decompression phase. The two solutions perfectly fit.
It is very important to note that due to the absence of damping, force has the same shape in both compression and decompression phase: there is not an hysteretic behaviour.

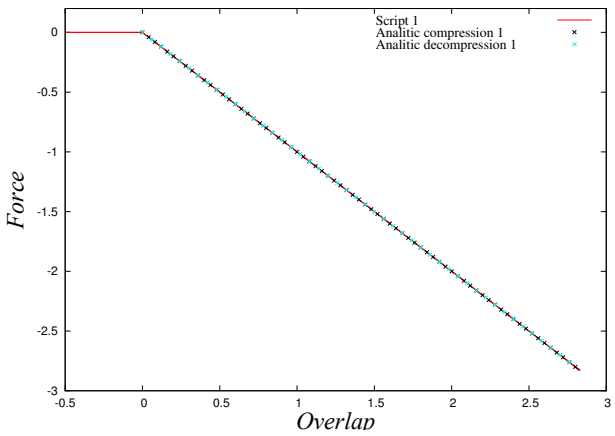


Figure 3.7: Script vs analitic force 1

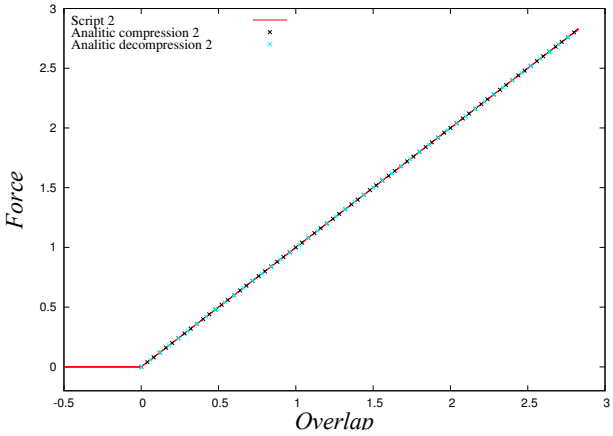
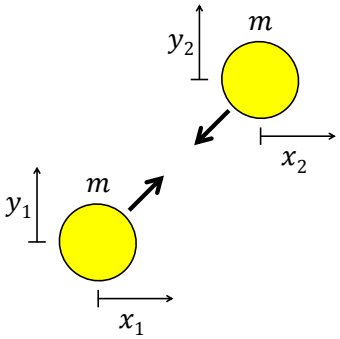


Figure 3.8: Script vs analitic force 2

3.3.2 Two particles 2D without damping



We consider two particles with the same mass $m = 1$ and the same radius $R_1 = R_2 = 10$ located in the plane xy (2D promblem). Material has stiffness $k = 1$ and there is no damping or friction. Initial positions are $x_1 = 0$ and $y_1 = 0$ for particle 1, $x_2 = 50$ and $y_2 = 50$ for particle 2. At the beginning the velocities of the components of particle 1 are $v_{x1} = 2$ and $v_{y1} = 2$ and velocities of particle 2 are $v_{x2} = -2$ and $v_{y2} = -2$.
In Figs. 3.9 - 3.13 we plot the evolution of position, velocity, acceleration, force and total energy.
For the solution script see A.3.2.

System parameters		
Mass	m	1
Radius particle 1	R_1	10
Radius particle 2	R_2	10
Stiffness	k	1
Initial conditions		
Position1x	x_1	0
Position1y	y_1	0
Position2x	x_2	50
Position2y	y_2	50
Velocity1x	v_{x1}	2
Velocity1y	v_{y1}	2
Velocity2x	v_{x2}	-2
Velocity2y	v_{y2}	-2
Simulation parameters		
Total time	t_f	15
Time step	Δt	10^{-4}

Results

We can see that, because of the absence of damping, the behaviour of particles is the same seen in Sec. 3.3.1.

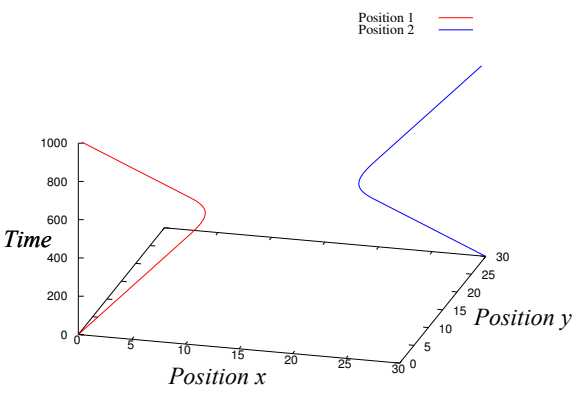


Figure 3.9: Time vs position

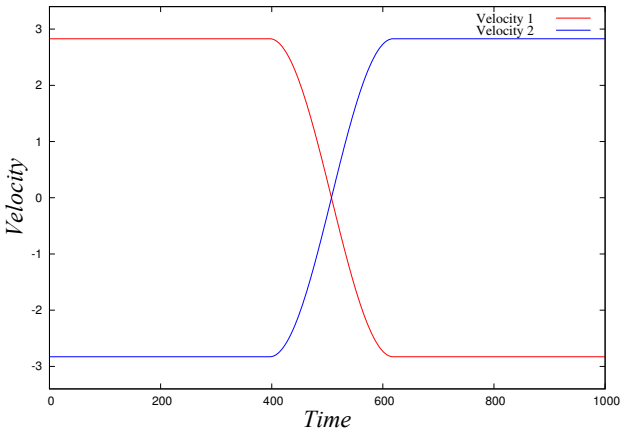


Figure 3.10: Velocity vs time

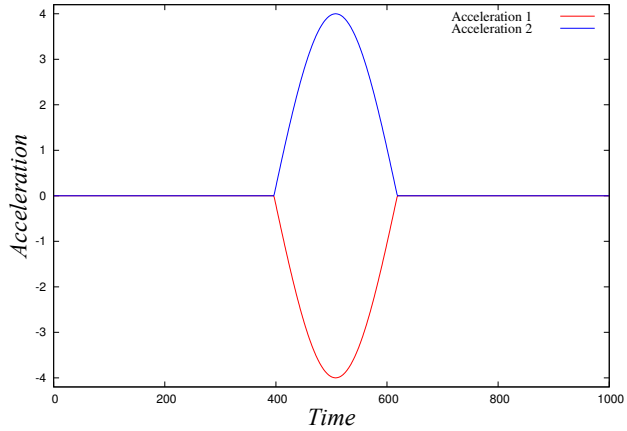


Figure 3.11: Acceleration vs time

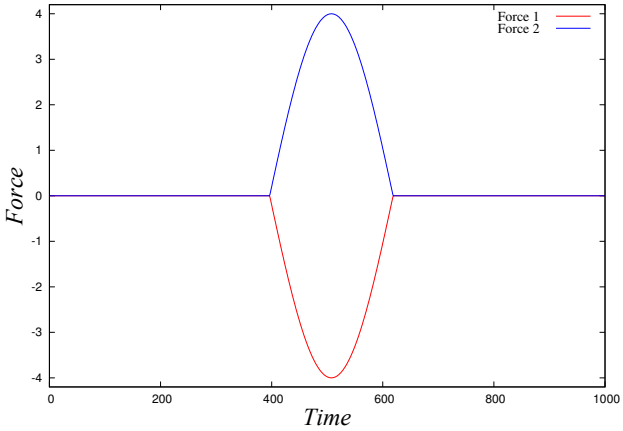


Figure 3.12: Force vs time

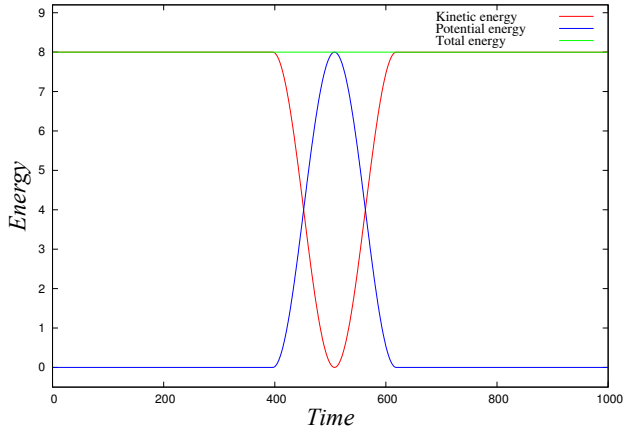
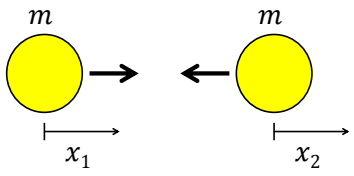


Figure 3.13: Energy vs time

3.3.3 Two particles 1D with damping



We consider two particles with the same characteristics of the exercise seen before (*Two particles 1D without damping*) except the stiffness that is $k = 100$. In this case we consider also the damping: material has a viscosity $\gamma = 1$.
In Figs. 3.14 - 3.18 we plot the evolution of position, velocity, acceleration, force and total energy.
For the solution script see A.3.3.

System parameters		
Mass	m	1
Radius particle 1	R_1	10
Radius particle 2	R_2	10
Stiffness	k	100
Viscosity	γ	1
Initial conditions		
Position1x	x_1	0
Position2x	x_2	30
Velocity1x	v_{x1}	2
Velocity2x	v_{x2}	-2
Simulation parameters		
Total time	t_f	5
Time step	Δt	10^{-4}

Results

Analyzing the evolution of position in Fig. 3.14 we see that, due to the viscosity, the trajectory of each particle changes after the collision.
About velocity (Fig. 3.15), after the collision the value of each velocity is less than before because of the damper.
The evolution of acceleration (Fig. 3.16) and force (Fig. 3.17) shows the typical attractive behaviour of the damper in the decompression phase.
The system is not conservative, so a part of the total energy (Fig. 3.18) is dissipated after the collision.

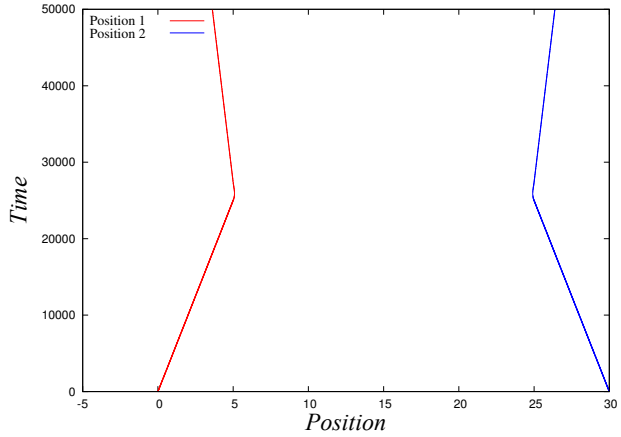


Figure 3.14: Time vs position

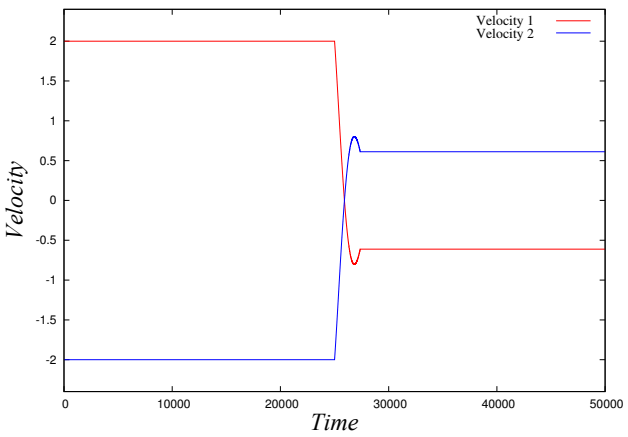


Figure 3.15: Velocity vs time

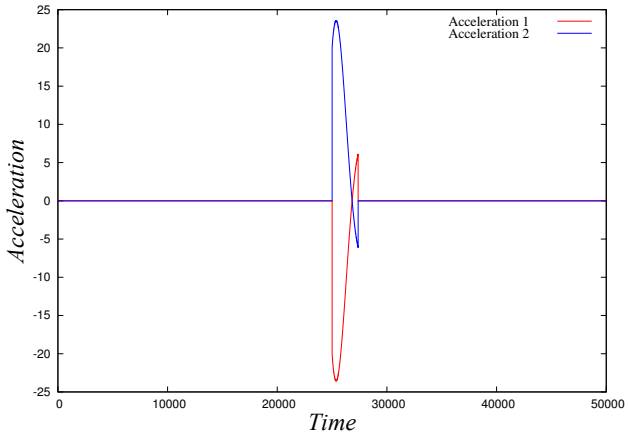


Figure 3.16: Acceleration vs time

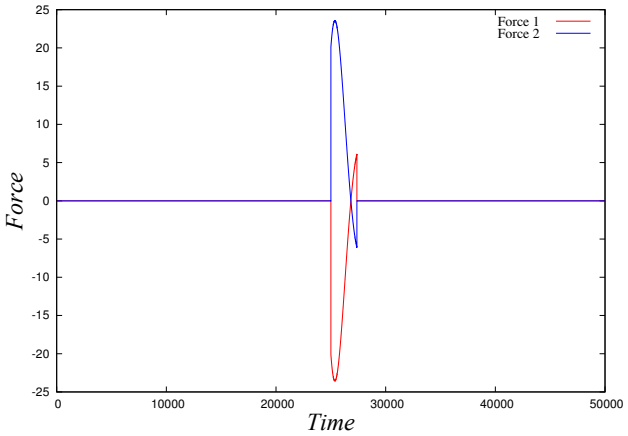


Figure 3.17: Force vs time

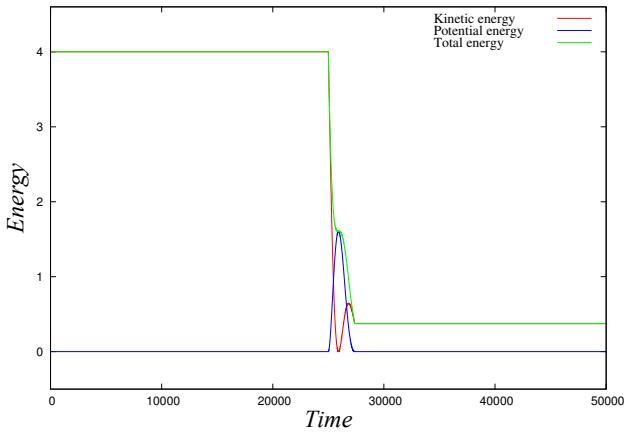


Figure 3.18: Energy vs time

Due to damping, the force presents a hysteretic behaviour and it is interesting to compare this with a system with the same charateristics and initial settings, but without the damper: it has the typical linear shape (Fig. 3.19).

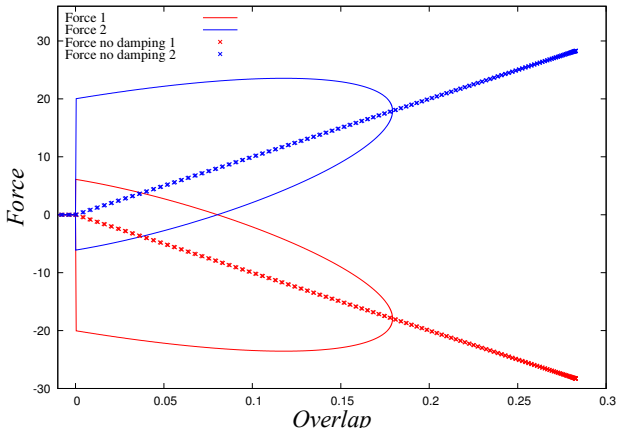
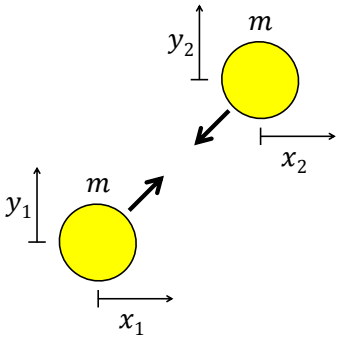


Figure 3.19: Force behaviour with and without damping vs overlap

3.3.4 Two particles 2D with damping



Consider two particles with the same geometrical dimensions and masses of *Two particles 2D without damping*, but in this case stiffness is $k = 100$ and viscosity is $\gamma = 1$.
In Figs. 3.20 - 3.24 we plot the evolution of position, velocity, acceleration, force and total energy.
For the solution script see A.3.4.

System parameters		
Mass	m	1
Radius particle 1	R_1	10
Radius particle 2	R_2	10
Stiffness	k	100
Viscosity	γ	1
Initial conditions		
Position1x	x_1	0
Position1y	y_1	0
Position2x	x_2	50
Position2y	y_2	50
Velocity1x	v_{x1}	2
Velocity1y	v_{y1}	2
Velocity2x	v_{x2}	-2
Velocity2y	v_{y2}	-2
Simulation parameters		
Total time	t_f	15
Time step	Δt	10^{-4}

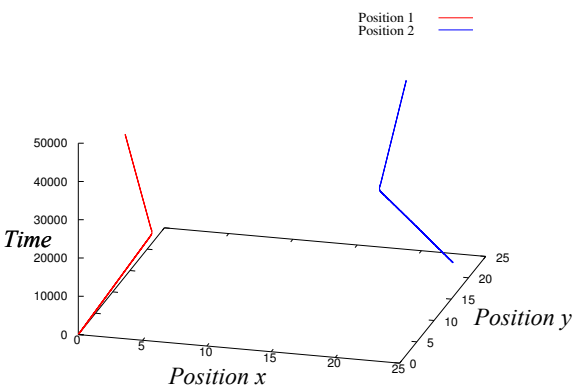


Figure 3.20: Time vs position

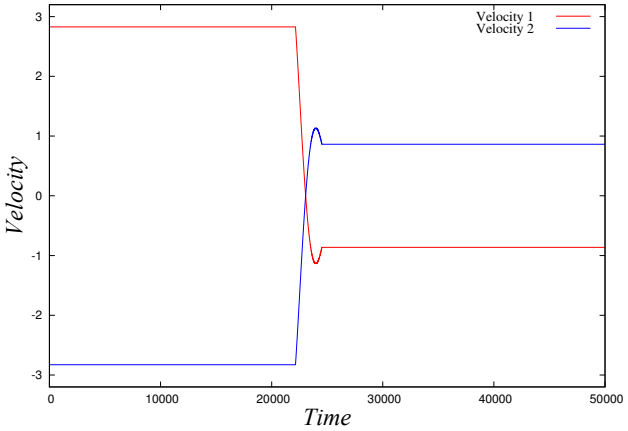


Figure 3.21: Velocity vs time

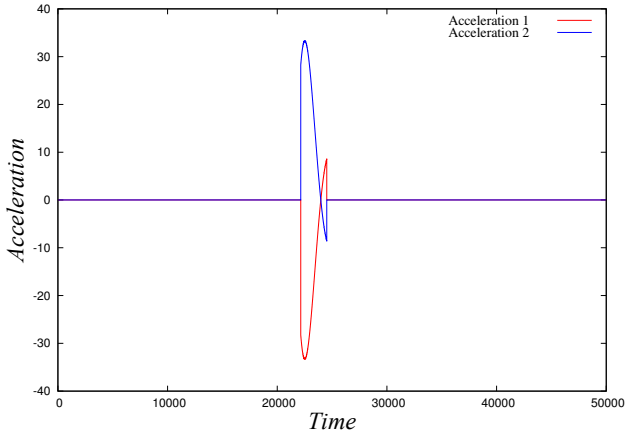


Figure 3.22: Acceleration vs time

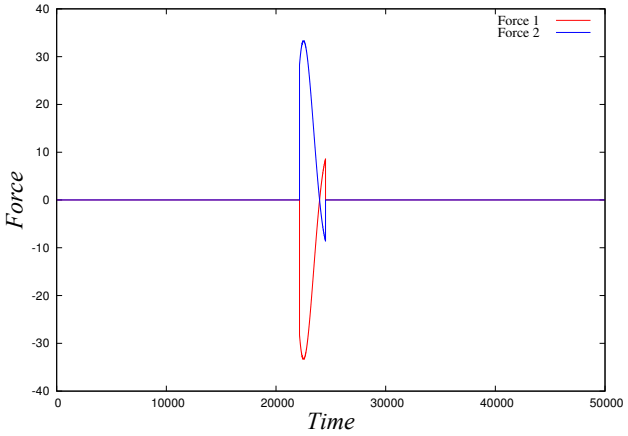


Figure 3.23: Force vs time

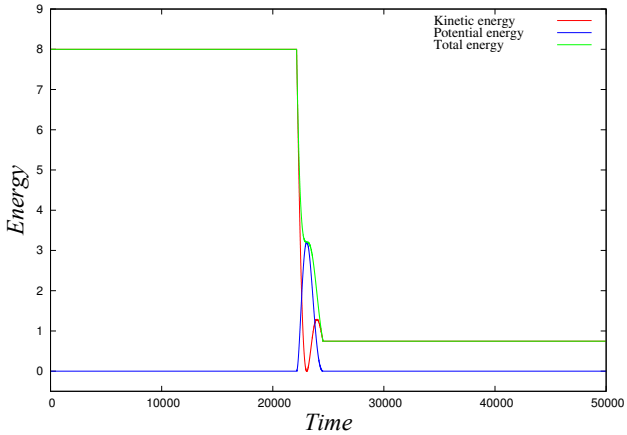
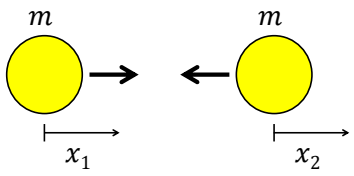


Figure 3.24: Energy vs time

3.3.5 Two particles 1D without damping - Linear vs Hertz model



We consider two glass particles with the same density $\rho = 2540[kg/m^3]$ and the same radius $R_1 = R_2 = 1[mm]$. Material has shear modulus $G = 29[GPa]$ and Poisson's ratio $\nu = 0.2$. Initial positions are $x_1 = 48.891[mm]$ for particle 1 and $x_2 = 50.9[mm]$ for particle 2. At the beginning velocities are $v_1 = 1[m/s]$ (particle 1) and $v_2 = -1[m/s]$ (particle 2).
We show a comparison of normal force behaviour using linear model (Eq. (3.2.2)) and Hertz model (Eq. (3.6)).
For the solution script see A.3.5.

System parameters		
Density	ρ	$2540[kg/m^3]$
Radius particle 1	R_1	10
Radius particle 2	R_2	10
Shear modulus	G	$29[GPa]$
Poisson's ratio	ν	0.2
Initial conditions		
Position1x	x_1	0
Position2x	x_2	50
Velocity1x	v_{x1}	2
Velocity2x	v_{x2}	-2
Simulation parameters		
Total time	t_f	$1.5 \cdot 10^{-5}[s]$
Time step	Δt	$10^{-8}[s]$

Stiffness k_{linear} for the linear model is found from the slope of Hertz force at a typical overlap value (40 % of δ_{max}), where δ_{max} is the maximum overlap found using Hertz model (Fig. 3.25).

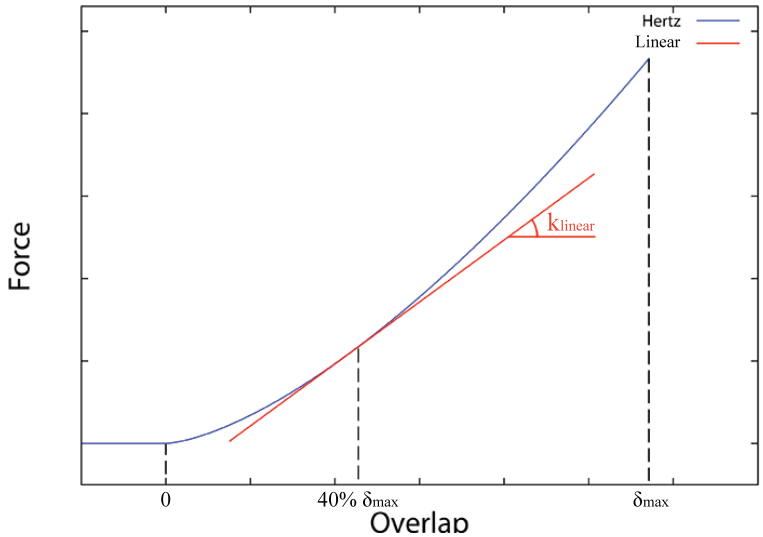


Figure 3.25: Evaluation of linear stiffness k_{linear} starting from Hertz model.

Results

In Fig. 3.26 and Fig. 3.27 it is possible to apreciate the dependence of Hertz model on the overlap: slope is continuously changing. On the other hand slope for linear model is constant.

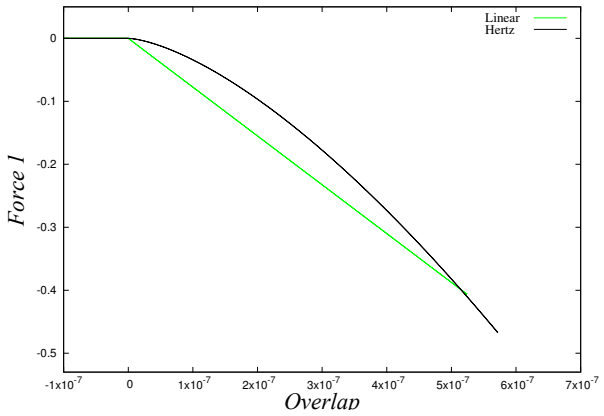


Figure 3.26: Linear and hertzian force on particle 1 vs overlap

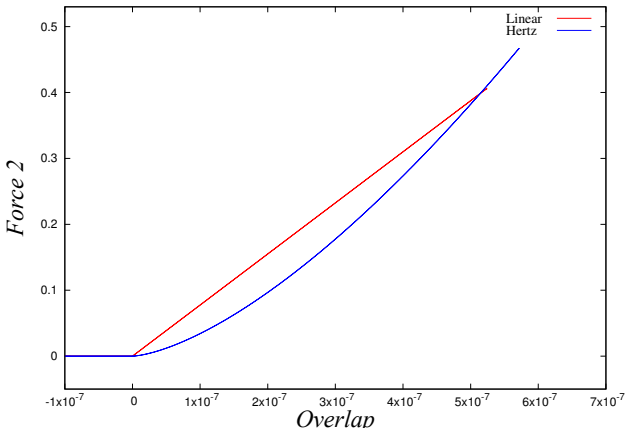


Figure 3.27: Linear and hertzian force on particle 2 vs overlap

Chapter 4

MSM Code

For simulations in house DEM code developed by prof. Stefan Luding will be used. This code is based on the same fundamentals seen in Chap. 3. Advanced features have been introduced in order to simulate random aggregate of spheres in various complex situations. In the following we will refer to this code as MSM code.

4.1 Scaling parameters

The DEM model used in MSM code does not have a built-in system unit. Every value used is dimensionless and becomes dimensional when it is scaled with the proper fundamental units. Therefore there is the need to define scaling parameters [16], starting from the foundamental units expressed in the SI-system:

Mass	m_u	kilograms $[kg]$
Lenght	x_u	meters $[m]$
Time	t_u	seconds $[s]$

Because of the small dimesions of particles and the short contact time of the systems that will be simulated, for code simulations the following values will be used:

	Reality		**Code**
Mass	$1[\mu gr]$	$\leftrightarrow$	1
Lenght	$1[mm]$	$\leftrightarrow$	1
Time	$1[\mu s]$	$\leftrightarrow$	1

It means, *e.g.*, that if in the code I read that a particle has mass $m_c = 2$ it means that, according to the value of foundamental units that I chose, the real mass is $m_u = 2\mu gr$. It is now possible to define scaling parameters, SP , that are the constant that permit the transition from dimensional (X_u) to adimensional (X_c) values and vice versa:

$$X_c = SP \cdot X_u \tag{4.1}$$

4.1.1 Scaling parameters table

The scaling parameters for the foundamental units are:

- Mass, m, $[kg]$

$$[kg] = 10^9[\mu gr] \Rightarrow SP_m = 10^9$$

- Lenght, x, $[m]$

$$[m] = 10^3[mm] \Rightarrow SP_x = 10^3$$

- Time, t, $[s]$

$$[s] = 10^6[\mu s] \Rightarrow SP_t = 10^6$$

All the other units can be defined starting from the foundamental:

- Density, ρ, $[kg/m^3]$

$$\frac{[kg]}{[m^3]} = \frac{10^9[\mu gr]}{(10^3[mm])^3} = \frac{10^9[\mu gr]}{10^9[mm^3]} = \frac{[\mu gr]}{[mm^3]} \Rightarrow SP_\rho = 1$$

- Velocity, v, $[m/s]$

$$\frac{[m]}{[s]} = \frac{10^3[m]}{10^6[\mu s]} = 10^{-3}\frac{[mm]}{[\mu s]} \Rightarrow SP_v = 10^{-3}$$

- Linear contact stiffness, k, $[kg/s^2]$

$$\frac{[kg]}{[s^2]} = \frac{10^9[\mu gr]}{(10^6[\mu s])^2} = \frac{10^9[\mu gr]}{10^{12}[\mu s^2]} = 10^{-3}\frac{[\mu gr]}{[\mu s^2]} \Rightarrow SP_k = 10^{-3}$$

- Linear viscosity, γ, $[kg/s]$

$$\frac{[kg]}{[s]} = \frac{10^9[\mu gr]}{10^6[\mu s]} = 10^3\frac{[\mu gr]}{[\mu s]} \Rightarrow SP_\gamma = 10^3$$

- Hertz viscosity parameter, η_H, $[kg/\sqrt{m}s]$

$$\frac{[kg]}{[\sqrt{m}][s]} = \frac{10^9[\mu gr]}{(10^{3/2}[\sqrt{mm}])(10^6[\mu s])} = 10^{3/2}\frac{[\mu gr]}{[\sqrt{mm}\mu s]} \Rightarrow SP_{\eta_H} = 10^{3/2}$$

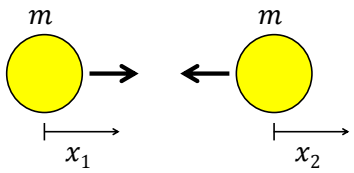
It is convenient to collect all the scaling paramter that will be used for our simulations in one table:

Property	**Symbol**	**SI unit**	**Scaling parameter**	X_u	X_c
Mass	m	$[kg]$	10^9	1	10^9
Lenght	x	$[m]$	10^3	1	10^3
Time	t	$[s]$	10^6	1	10^6
Density	ρ	$[kg/m^3]$	1	1	1
Velocity	v	$[m/s]$	10^{-3}	1	10^{-3}
Linear stiffness	k	$[kg/s^2]$	10^{-3}	1	10^{-3}
Linear viscosity	γ	$[kg/s]$	10^3	1	10^3
Hertz viscosity	η_H	$[kg/\sqrt{m}s]$	$10^{3/2}$	1	$10^{3/2}$

4.2 Comparison between my MATLAB®script and MSM code examples

Before going on with the simulation, it will be interesting compare different examples solved with my MATLAB®script and with MSM code and then calculate the hightest relative error.

4.2.1 Two particles 1D without damping



We consider two glass beads with the same radius, $R_1 = R_2 = 10^{-3}[m]$, moving in x direction. The density of the material is $\rho = 2000[kg/m^3]$, its stiffness is $k = 10^8$ and there is not damping. Initial positions are $x_1 = 0.048891[m]$ and $x_2 = 0.0509[m]$ and initial velocities are $v_1 = 1[m/s]$ and $v_2 = -1[m/s]$. We Compare the results obtained using my MATLAB®script (based on Velvet algorithm) and MSM code in Fig. 4.1 and Fig. 4.2.
For the solution script see A.4.1.

Before starting is important to define the input values for the script and the code using the scaling parameters defined in Sec. 4.1.

System parameters				
		MATLAB®script		**MSM code**
Density	ρ	2000	$\leftrightarrow$	2000
Radius particle 1	R_1	10^{-3}	$\leftrightarrow$	1
Radius particle 2	R_2	10^{-3}	$\leftrightarrow$	1
Stiffness	k	10^8	$\leftrightarrow$	10^5
Initial conditions				
		MATLAB®script		**MSM code**
Position1x	x_1	0.048891	$\leftrightarrow$	48.891
Position2x	x_2	0.0509	$\leftrightarrow$	50.9
Velocity1x	v_1	1	$\leftrightarrow$	10^{-3}
Velocity2x	v_2	-1	$\leftrightarrow$	-10^{-3}
Simulation parameters				
		MATLAB®script		**MSM code**
Total time	t_f	10^{-5}	$\leftrightarrow$	10
Time step	Δt	10^{-9}	$\leftrightarrow$	10^{-3}

Results

It is possible to evaluate that the evolution of position defined with my MATLAB®script perfetcly fits with the one obtained with MSM code for each particle.

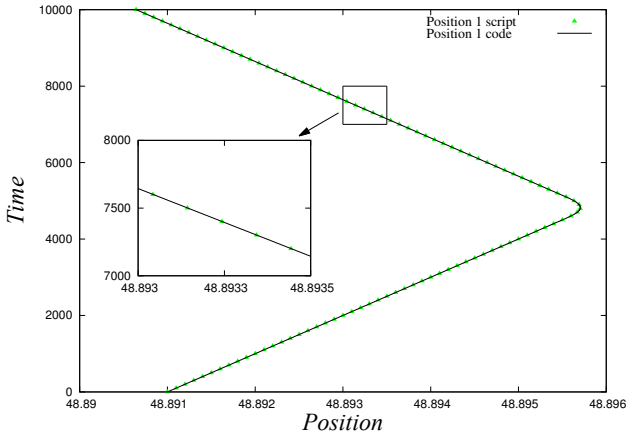


Figure 4.1: x position of particle 1

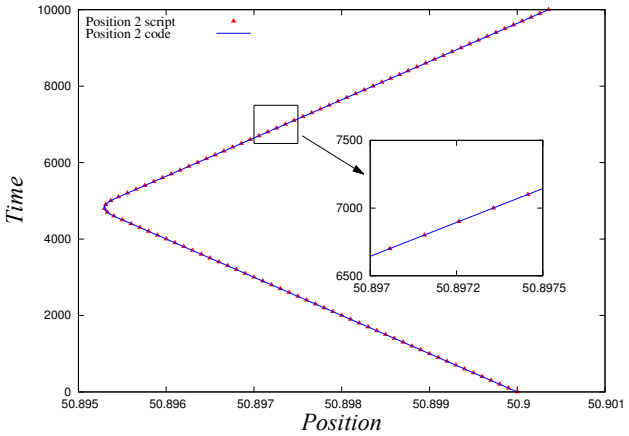


Figure 4.2: x position of particle 2

Hightest relative error	
x_1	$2.0454 \cdot 10^{-8}$
x_2	$1.9648 \cdot 10^{-8}$

Same examples for only y and only z directions have results that are comparable with this example.

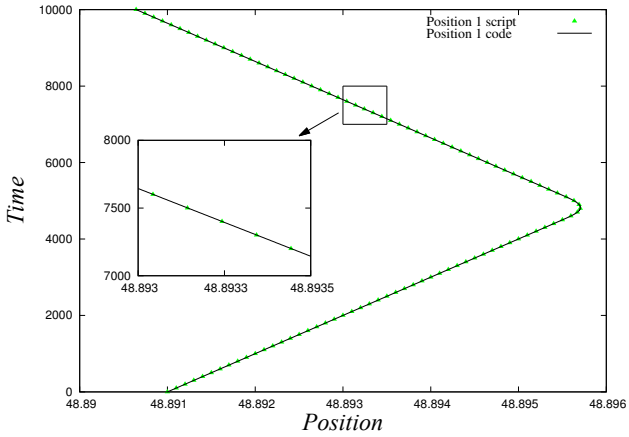


Figure 4.3: y position of particle 1

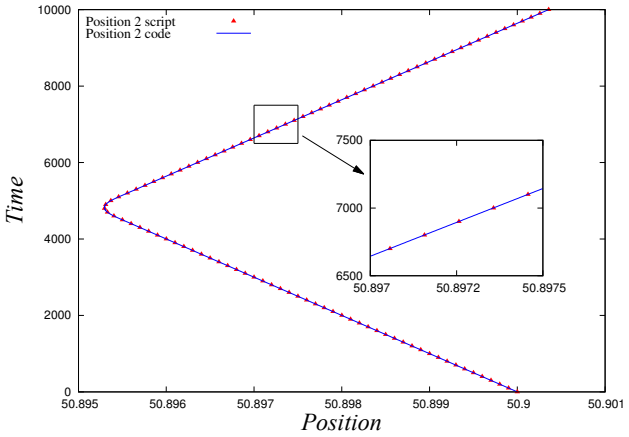


Figure 4.4: y position of particle 2

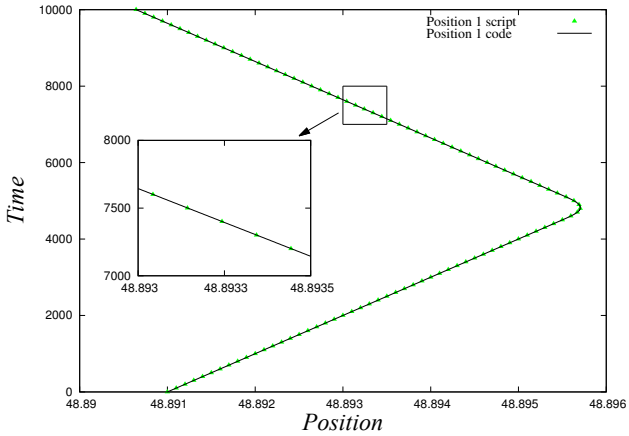


Figure 4.5: z position of particle 1

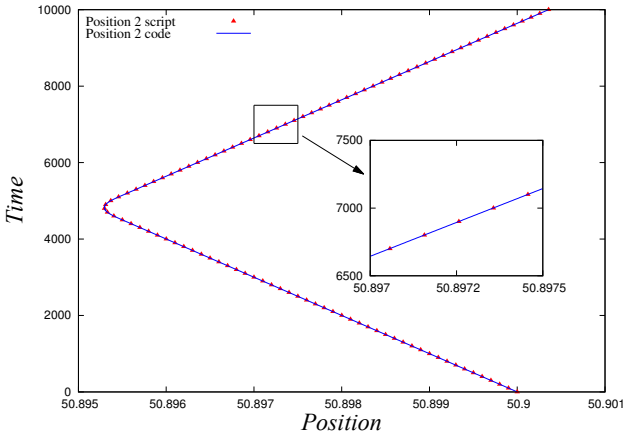
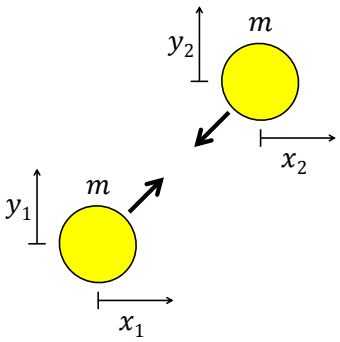


Figure 4.6: z position of particle 2

Hightest relative error	
y_1	$2.0454 \cdot 10^{-8}$
y_2	$1.9648 \cdot 10^{-8}$
z_1	$2.0454 \cdot 10^{-8}$
z_2	$1.9648 \cdot 10^{-8}$

4.2.2 Two particles 2D without damping



We consider two glass beans both with radius $R_1 = R_2 = 10^{-3}[m]$, moving in the xy plane. The density of the material is $\rho = 2000[kg/m^3]$, its stiffness is $k = 10^8$ and there is not damping. Initial coordinate for particle 1 are $x_1 = 0.049288[m]$ and $y_1 = 0.049288[m]$ and for particle 2 are $x_2 = 0.050712[m]$ $y_2 = 0.050712[m]$. Velocity initial components are $v_{1x} = 1[m/s]$ and $v_{1y} = 1[m/s]$; $v_{2x} = -1[m/s]$ and $v_{2y} = -1[m/s]$.We Compare the results obtained using my MATLAB®script (based on Velvet algorithm) and MSM code in Fig. 4.7 and Fig. 4.8.
For the solution script see A.4.2.

Before starting is important to define the input values for my MATLAB®script and MSM code using the scaling parameters defined in 4.1.

System parameters				
		MATLAB® script		**Code**
Density	ρ	2000	$\leftrightarrow$	2000
Radius particle 1	R_1	10^{-3}	$\leftrightarrow$	1
Radius particle 2	R_2	10^{-3}	$\leftrightarrow$	1
Stiffness	k	10^{8}	$\leftrightarrow$	10^{5}
Initial conditions				
		MATLAB® script		**MSM code**
Position1x	x_1	0.049288	$\leftrightarrow$	49.288
Position1y	y_1	0.049288	$\leftrightarrow$	49.288
Position2x	x_2	0.050712	$\leftrightarrow$	50.712
Position2x	y_2	0.050712	$\leftrightarrow$	50.712
Velocity1x	v_{1x}	1	$\leftrightarrow$	10^{-3}
Velocity1y	v_{1y}	1	$\leftrightarrow$	10^{-3}
Velocity2x	v_{2x}	-1	$\leftrightarrow$	-10^{-3}
Velocity2x	v_{2y}	-1	$\leftrightarrow$	-10^{-3}
Simulation parameters				
		MATLAB® script		**MSM code**
Total time	t_f	10^{-5}	$\leftrightarrow$	10
Time step	Δt	10^{-9}	$\leftrightarrow$	10^{-3}

Results

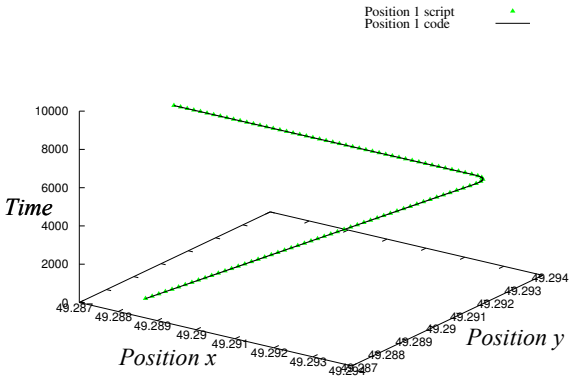


Figure 4.7: Position of particle 1

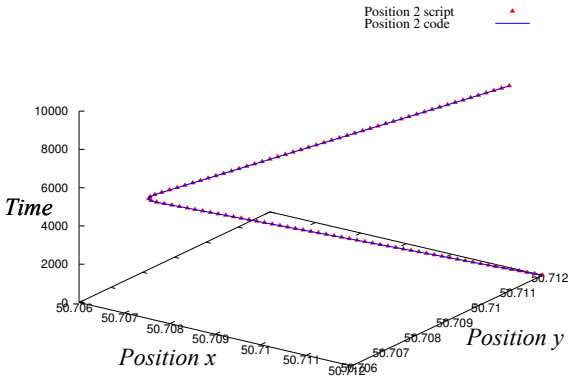
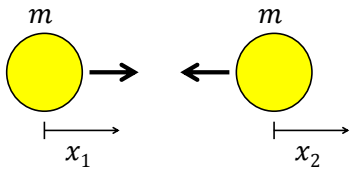


Figure 4.8: Position of particle 2

Hightest relative error	
x_1	$2.0289 \cdot 10^{-8}$
y_1	$2.0289 \cdot 10^{-8}$
x_2	$1.9721 \cdot 10^{-8}$
y_2	$1.9721 \cdot 10^{-8}$

4.2.3 Two particles 1D with damping



We consider two particles made with the same characteristics of the exercise seen before (*Two particles 1D without damping*). In this case we consider also the damping: material has a viscosity $\gamma = 1[kg/s]$. Initial positions and velocities are the same of previous exercise. We compare the results obtained using my MATLAB®script and MSM code in Fig. 4.9 and Fig. 4.10.
For the solution script see A.4.3.

We define input values for my MATLAB®script and MSM code thanks to the scaling parameters (see Sec. 4.1).

System parameters				
		MATLAB®script		**MSM code**
Density	ρ	2000	$\leftrightarrow$	2000
Radius particle 1	R_1	10^{-3}	$\leftrightarrow$	1
Radius particle 2	R_2	10^{-3}	$\leftrightarrow$	1
Stiffness	k	10^{8}	$\leftrightarrow$	10^{5}
Viscosity	γ	1	$\leftrightarrow$	10^{3}
Initial conditions				
		MATLAB®script		**MSM code**
Position1x	x_1	0.048891	$\leftrightarrow$	48.891
Position2x	x_2	0.0509	$\leftrightarrow$	50.9
Velocity1x	v_1	1	$\leftrightarrow$	10^{-3}
Velocity2x	v_2	-1	$\leftrightarrow$	-10^{-3}
Simulation parameters				
		MATLAB®script		**MSM code**
Total time	t_f	10^{-5}	$\leftrightarrow$	10
Time step	Δt	10^{-9}	$\leftrightarrow$	10^{-3}

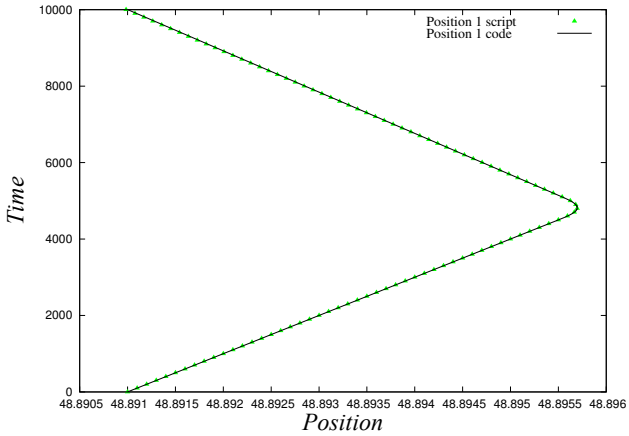


Figure 4.9: x position of particle 1

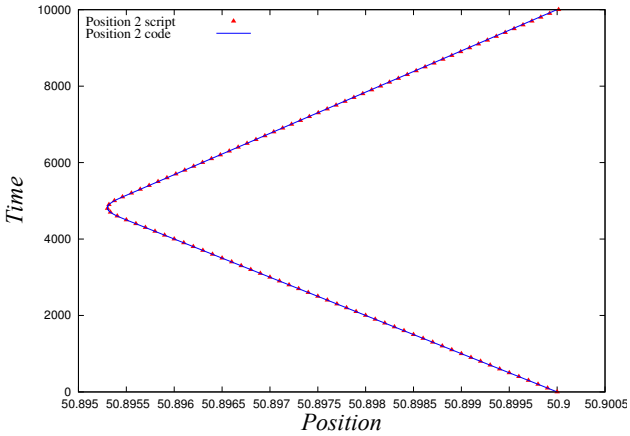
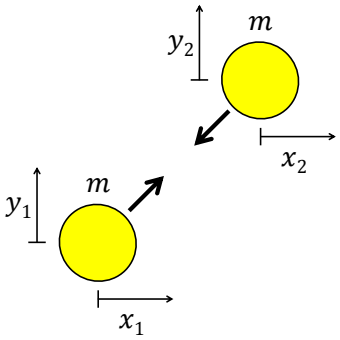


Figure 4.10: x position of particle 2

Hightest relative error script vs code	
x_1	$3.7248 \cdot 10^{-7}$
x_2	$3.5778 \cdot 10^{-7}$

4.2.4 Two particles 2D with damping



We consider 2 particles in $x - y$ plane with same characteristics of material, initial positions and initial velocities of the previous exercise *Two particles 2D without damping*. In this case we consider the damping, with a material viscosity $\gamma = 1$. We compare results obtained using my MATLAB®script and MSM code in Fig. 4.11 and Fig. 4.12.
For the solution script see A.4.4.

We define input values for my MATLAB®script and MSM code thanks to the scaling parameters (see 4.1).

System parameters				
		MATLAB® script		**MSM code**
Density	ρ	2000	$\leftrightarrow$	2000
Radius particle 1	R_1	10^{-3}	$\leftrightarrow$	1
Radius particle 2	R_2	10^{-3}	$\leftrightarrow$	1
Stiffness	k	10^{8}	$\leftrightarrow$	10^{5}
Viscosity	γ	1	$\leftrightarrow$	10^{3}
Initial conditions				
		MATLAB® script		**MSM code**
Position1x	x_1	0.049288	$\leftrightarrow$	49.288
Position1y	y_1	0.049288	$\leftrightarrow$	49.288
Position2x	x_2	0.050712	$\leftrightarrow$	50.712
Position2x	y_2	0.050712	$\leftrightarrow$	50.712
Velocity1x	v_{1x}	1	$\leftrightarrow$	10^{-3}
Velocity1y	v_{1y}	1	$\leftrightarrow$	10^{-3}
Velocity2x	v_{2x}	-1	$\leftrightarrow$	-10^{-3}
Velocity2x	v_{2y}	-1	$\leftrightarrow$	-10^{-3}
Simulation parameters				
		MATLAB® script		**MSM code**
Total time	t_f	10^{-5}	$\leftrightarrow$	10
Time step	Δt	10^{-9}	$\leftrightarrow$	10^{-3}

Results

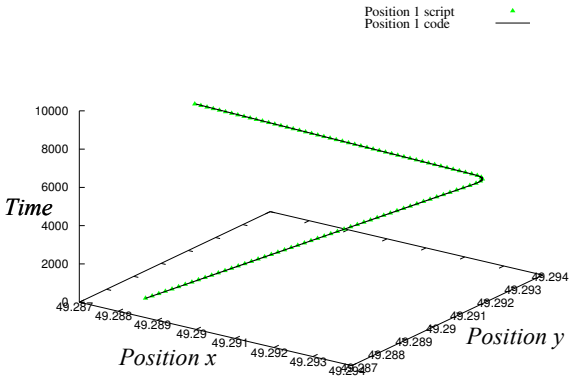


Figure 4.11: Position of particle 1

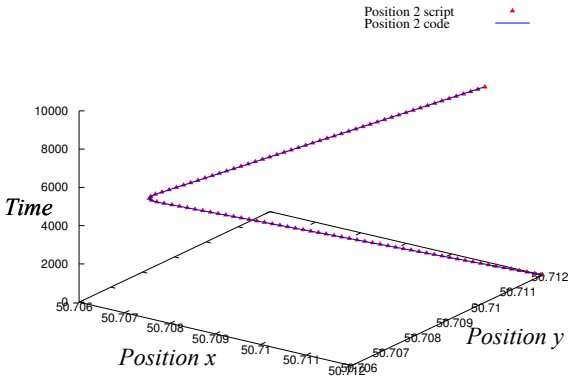


Figure 4.12: Position of particle 2

Biggest relative error	
x_1	$3.4118 \cdot 10^{-7}$
y_1	$3.4118 \cdot 10^{-7}$
x_2	$3.3161 \cdot 10^{-7}$
y_2	$3.3161 \cdot 10^{-7}$

Chapter 5

DEM simulation of random aggregate of spheres

All simulations made in this thesis work are performed following the same procedure: first of all position and velocity of each particle must be defined, then the sample is prepared to the final phase, after that we can obtain data for evaluation of bulk and shear modulus.

5.1 System parameters

Before starting with the simulations is important to define system parameters.

- Volume fraction, ν

$$\nu = \frac{\sum_{i=1}^{N_p} V_i}{V_{box}} \tag{5.1}$$

where V_i is the volume of the $i-th$ particle in a system of N_p particles inside a box with volume V_{box}.The system shows gas-like behaviour at low volume fraction, as it behaves like liquid close to jamming point and it becomes like a solid above jamming point.
Jamming fraction, ν_{jam}, is a point where system behaviour changes from liquid-like to solid-like, so it is a very delicate point. Jamming point depends upon different parameters, inter-particle contact friction and strain rate of particles are two important parameters.

- Mean radius, $< r >$

$$< r >= \frac{\sum_{i=1}^{N_p} a_i}{N_p} \tag{5.2}$$

where a_i is the radius of the $i-th$ particle in a system of N_p.

- Polydispersity, W

$$W = \frac{a_{max}}{a_{min}} \tag{5.3}$$

where r_{max} is the maximum and r_{min} is the minimum value of the radius in a system of N_p particles. $W = 1$ means that all particles having the same dimension.

5.2 Macroscopic (tensorial) quantities

DEM allows a very detailed description of granular materials including contact forces and exact position of the particles. However, the amount of data generated during a DEM simulation is a lot. Therefore, averaging method is employed to link between microscopic and macroscopic quantities.
Here, we define averaged tensorial macroscopic quantities that provide information about state of packing and reveal interesting elastic features [35].

- Strain tensor, $\mathbf{E}$
  By speaking about the strain tensor, we refer to the external (global) strain that we apply to the sample. This tensor is necessary to define the isotropical infinitesimal strain (ε_{vol}): $\varepsilon_{vol} = tr(-\mathbf{E})/3 = tr(-\dot{\mathbf{E}}dt)/3$.

- Stress tensor, σ

$$\sigma = \frac{1}{V} \sum_{c \in V} \mathbf{l}^c \otimes \mathbf{f}^c \tag{5.4}$$

averaged over all contacts in the volume V, with the dyadic product between the contact force $\mathbf{f}^c$ and the branch vector $\mathbf{l}^c$, where the contribution of the kinetic fluctuation energy has been neglected. The isotropic component of the stress is the pressure P

$$P = \frac{tr(\sigma)}{3}. \tag{5.5}$$

- Fabric tensor, $\mathbf{F}$

$$\mathbf{F} = \frac{1}{V} \sum_{P \in V} V^P \sum_{c \in P} \mathbf{n}^c \otimes \mathbf{n}^c \tag{5.6}$$

weighted according to V^P, the particle volume of particle P, for all the particles inside the averaging volume V, the normal unit branch-vector $\mathbf{n}^c$ pointing from the center of particle P to contact c. The isotropic fabric, F_v, is proportional to the volume fraction ν and the coordination number C [17]:

$$F_v = tr(\mathbf{F}) = g_3 \nu C \tag{5.7}$$

with a function g_3 of moments of the size distribution and $g_3 \approx 1.22$ for polydispersity $W = 3$.

5.3 Elastic moduli, linear normal contact force law

In this section, we study the incremental response of cohesionless granular materials during different deformation modes, to obtain the effective bulk and shear modulus of material. For the sake of simplicity, the linear contact model (Sec. 3.2.1) for the normal component of force is used and Coulomb friction (Sec. 3.2.4) for the tangential components.

5.3.1 Preparation procedure

Sample preparation is a foundamental procedure for every physical or numerical experiment to obtain reproducible and reliable results, especially when friction is involved.
Collision of two typical particles is considered to be elastic with a restitution coefficient of $e = 0.92$, which corresponds to a contact duration $t_c = 0.64[\mu s]$. Simulation time step is chosen to be 1/50 of contact duration in order to ensure that all collisions having an enough time.
Standard simulation parameters and numerical values used into simulations are summarized in Tab. 5.1.

System parameters				
		SI units		**Numerical**
Number of particles	N_p	$21^3 = 9261$	$\leftrightarrow$	9261
Mean radius	$< r >$	$10^{-3}[m]$	$\leftrightarrow$	1
Polydispersity	W	3	$\leftrightarrow$	3
Density	ρ	$2000[kg/m^3]$	$\leftrightarrow$	2000
Normal stiffness	k	$10^8[kg/s^s]$	$\leftrightarrow$	10^5
Tangential stiffness	k_t	$2 \cdot 10^7[kg/s^s]$	$\leftrightarrow$	$2 \cdot 10^4$
Normal damping	γ	$1[kg/s]$	$\leftrightarrow$	10^3
Tangential damping	γ_t	$0.2[kg/s]$	$\leftrightarrow$	200
Background damping	γ_b	$0.1[kg/s]$	$\leftrightarrow$	100
Time step	Δt	$10^{-8}[s]$	$\leftrightarrow$	0.01

Table 5.1: Simulation parameters

Sample preparation consists of following steps (see Fig. 5.2), each one composed by 4000 timesteps:

- Randomization
  Particles are generated randomly with random velocities at low volume fraction ($\nu = 0.3$) in a periodic 3D box (Fig. 5.1). It is necessary to leave enough space and time for particles to exchange places and randomize themselves.

- Isotropic compression (1)
  The first step of the preparation is to compress the granular gas isotropically, until a volume fraction below the jamming fraction ($\nu = 0.5$).

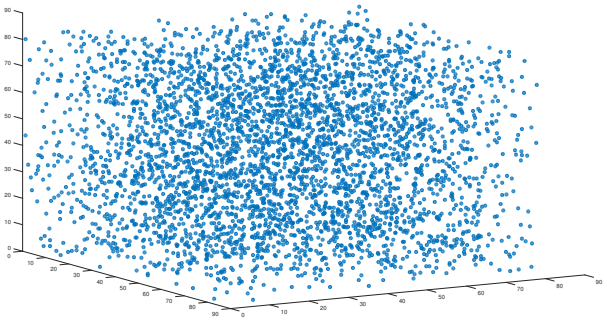


Figure 5.1: Initial configuration of samples. Particles (blue dots) are generated randomly into a periodic 3D box at low volume fraction $\nu = 0.3$.

- Relaxation (2)
  After the compression, the system is relaxed at a constant volume fraction ($\nu = 0.5$) to allow the paricles to dissipate their kinetic energy and to achieve a zero-pressure static configuration.

- Isotropic compression (3)
  Isotropic compression is applied on the periodic box to reach a desired volume fraction, $\nu_{max} = 0.82$.

- Isotropic decompression (4)
  Isotropic decompression until $\nu = 0.5$. This phase defines the starting configurations of the next step and also identifies the value of the jamming fraction.

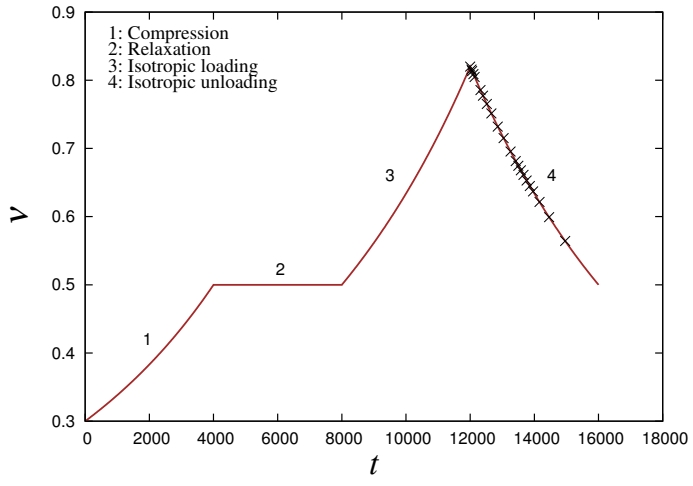


Figure 5.2: Evolution of volume fraction as a function of time during sample preparation: (1) a granular gas is homogeneously compressed from $\nu = 0.3$ to $\nu = 0.5$; and (2) relaxed at $\nu = 0.5$; (3) the sample is compressed from $\nu = 0.5$ to $\nu = 0.82$; finally, the sample is decompressed from $\nu = 0.82$ to $\nu = 0.5$. Black crosses 'X' represent the chosen configurations for further tests.

5.3.2 Elastic moduli

Here, we study the incremental response of cohesionless granular materials during different deformation modes. Calculation of bulk and shear modulus is implemented starting from various configurations along the decompression branch (phase D in Fig. 5.2) of preparation. After the selection of the beginning configurations, a sufficient relaxation is applied that allows particles to achieve a static configuration in mechanical equilibrium. Relaxed configurations are ready to be applied under different deformation modes (isotropic compression or pure shear).
The relaxed samples are compressed isotropically by applying small perturbations. The incremental stress responses can be calculated, therefore, the effective bulk modulus B^* is measured using Eq. (5.11).
On the other hand, pure shear (compression - decompression) can be applied to the relaxed samples and effective shear modulus G^* is obtained using Eq. (5.12).

Before going on, we should define the bulk modulus and shear modulus thanks to the parameters linked to them. Using non dimensional moduli permits to define easily characteristics of each material using proper scaling parameters.

- Pressure, P

$$P = \frac{\sigma_{xx} + \sigma_{yy} + \sigma_{zz}}{3}. \tag{5.8}$$

- Non-dimensional pressure, P^*

$$P^* = \frac{P}{k^*} \tag{5.9}$$

  where $k^* = k/(2 < r >)$.

- Volumetric strain, ε_{vol}

$$\varepsilon_{vol} = \frac{\varepsilon_{xx} + \varepsilon_{yy} + \varepsilon_{zz}}{3} \tag{5.10}$$

It is possible to define the non dimesional bulk modulus, B^*, of the granular assembly as the ratio between an incremental pressure and volumetric strain:

$$B^* = \frac{\delta P^*}{3\delta\varepsilon_{vol}} \tag{5.11}$$

Non dimensional shear modulus for xy plane G^*_{xy} is defined as the ratio of the change in difference of stresses to the change in difference of strains with $\delta(\varepsilon_{xx} = -\delta(\varepsilon_{yy}$.

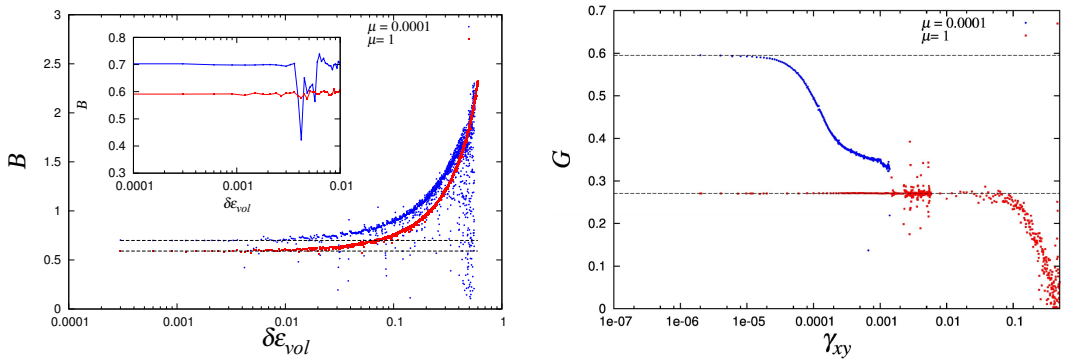
$$G^*_{xy} = \frac{\delta(\sigma_{xx} - \sigma_{yy})^*}{2\delta(\varepsilon_{xx} - \varepsilon_{yy})} \tag{5.12}$$

where $(\sigma_{xx} - \sigma_{yy})^* = (\sigma_{xx} - \sigma_{yy})/k^*$.

It should be noted that if packings are isotropically prepared then $G^*_{xy} = G^*_{zx} = G^*_{yz}$.

5.3.3 Evolution of elastic moduli

Because of the interest about the elastic response of granular packings, first of all is necessary to identify the elastic regime, the marginal regime and the plastic regime [17]. Elastic regime is characterized by a pratically constant value of elastic moduli, so it is the horizontal part in Fig. 5.3(a) and Fig. 5.3(b). Plastic regime starts when there are particle rearrangements, for this reason is important to keep the infinitesimal strain step small enough.
In Fig. 5.3(a) is possible to see that bulk modulus stays pratically constant for small amplitudes ($\delta\varepsilon_{vol} < 10^{-4}$) and the regime can be considered to be elastic [24]. By increasing the amplitudes of the pertubation, $\delta\varepsilon_{vol}$, B starts to increase non-linearly. Furthere, difference between reversible and irrevesible part becomes clear in both figures. Fig. 5.3(a) and Fig. 5.3(b) show that the elastic regime is wider when the friction coefficient is larger, due to existence of a stronger force network while friction is higher (*i.e.* higher tangential forces). [13].

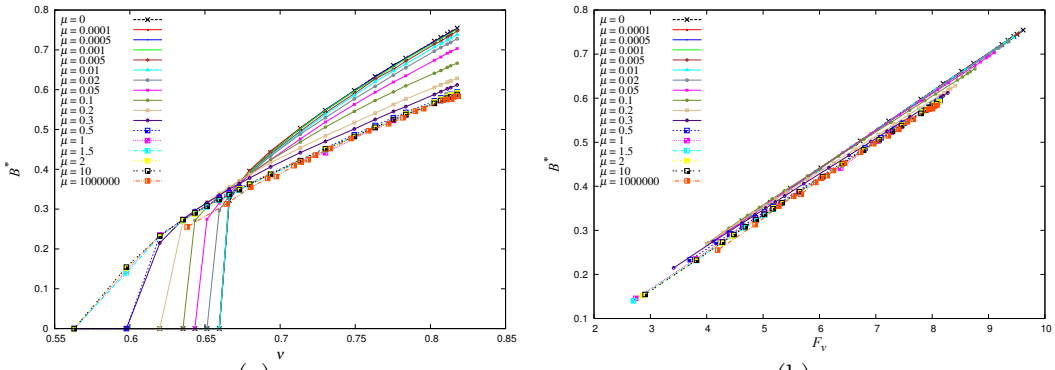


(a) (b)

Figure 5.3: Evolution of bulk modulus B^* (a) and shear modulus G^* (b) for a configuration at $\nu = 0.82$ and coefficient of friction 0.0001 and 1.

Effect of inter-particle contact friction on the bulk modulus

The variation of adimensional bulk modulus B^* verus the volume fraction ν for packings with different friction coefficients μ is illustrated in Fig. 5.4(a). Bulk modulus always increases with increasing density and this behaviuor is slower for packings with high friction. We can relate this characteristic to a higher average number of contacts for samples prepared with low friction at the same volume fraction.
When the bulk modulus is plotted not againts volume fraction, but against the isotropic fabric F_v (Fig. 5.4(b)), the data approximately collapse on an unique curve, implying a general relation between bulk stiffness and isotropic micro-structure.The coefficient of friction has no direct influence on the bulk modulus as sliding is not activated in the elastic regime for isotropic pre-strain, but rather it effects B^* indirectly through the preparation that leads to a different state variable F_v.

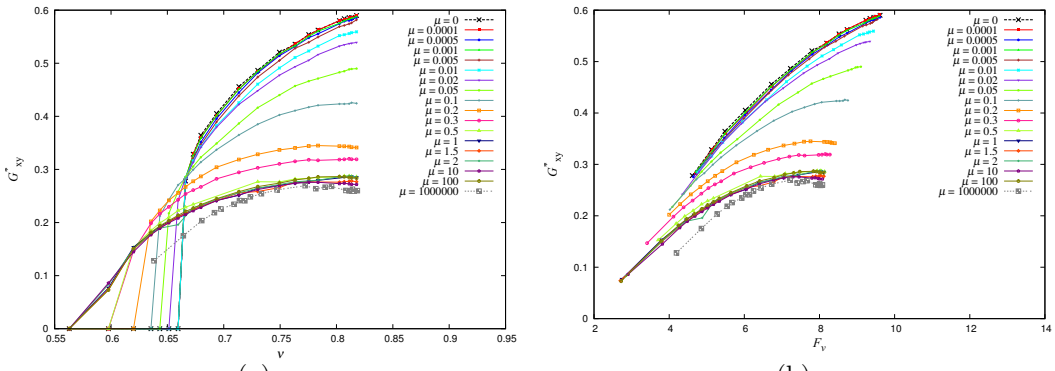


(a) (b)

Figure 5.4: (a) Evolution of adimensional bulk modulus B^* with volume fraction ν for different coefficients of friction μ, as shown in the legend, (b) Evolution of adimensional bulk modulus B^* versus isotropic fabric F_v for different coefficients of friction μ, as shown in the legend.

Effect of inter-particle contact friction on the shear modulus

In Fig. 5.5(a) we show the evolution of adimensional shear modulus G^* with volume fraction ν and in Fig. 5.5(b) G^* is represented versus isotropic fabric F_v.
As we can see, by increasing the value of friction coefficient μ, the value of G^* starts to decrease at the same value of volume fraction. When shear modulus G^* is plotted against isotropic fabric F_v, it is seen that the data does not collapse on an uniqe curve (unlike bulk modulus 5.4(b)) which means that shear modulus depends upon other microscopic parameters (like jamming point).



(a) (b)

Figure 5.5: (a) Evolution of adimensional shear modulus G^* with volume fraction ν for different coefficients of friction μ, as shown in the legend, (b) Evolution of adimensional shear modulus G^* versus isotropic fabric F_v for different coefficients of friction μ, as shown in the legend

5.4 Elastic moduli, Hertz normal contact force law

5.4.1 Sample definition

In this section, the Hertz contact model (Sec. 3.2.2) is used for the normal component and Coulomb friction (Sec. 3.2.4) for the tangential force.
Standard simulation parameters and numerical values used into simulations are summarized in Tab. 5.2.

System parameters				
		Reality		**Code**
Number of particles	N_p	$16^3 = 4096$	$\leftrightarrow$	4096
Mean radius	$< r >$	$2 \cdot 10^{-3}[m]$	$\leftrightarrow$	2
Polydispersity	W	3	$\leftrightarrow$	3
Density	ρ	$2540[kg/m^3]$	$\leftrightarrow$	2540
Effective Young's modulus	E^*	$2.4167 \cdot 10^{10}[Pa]$	$\leftrightarrow$	$2.4167 \cdot 10^4$
Tangential stiffness	k_t	$10^7[kg/s^s]$	$\leftrightarrow$	10^4
Hertz viscosity parameter	η_H	$221.35[kg/\sqrt{m}s]$	$\leftrightarrow$	7000
Tangential damping	γ_t	$1.4[kg/s]$	$\leftrightarrow$	1400
Background damping	γ_b	$0.1[kg/s]$	$\leftrightarrow$	100
Time step	Δt	$10^{-8}[s]$	$\leftrightarrow$	0.01

Table 5.2: Simulation parameters

Tangential stiffness k_t is evaluated as 3/7 of hertzian normal stiffness k_n, which is the slope of Hertz force at 40 % of maximum overlap found in the collision of two typical particles.

5.4.2 Preparation procedure

Preparation procedure is the same described in Sec. 5.3.1. Each phase is composed of 20000 timesteps.

5.4.3 Evolution of elastic moduli

In this section we report the evolution of bulk B (Fig. 5.6) and shear G (Fig. 5.7) modulus versus pressure P and isostropic fabric F_v for Hertz models.
EMT theroy predicts that B and G are scaled with 1/3 of pressure in case of frictionless particles, but we can see that this theory breaks down in case of frictional particles: in our case we have 0.4 dependance for B and 0.35 dependance for G.
Like linear model, B scales linearly with F_v and G does not.

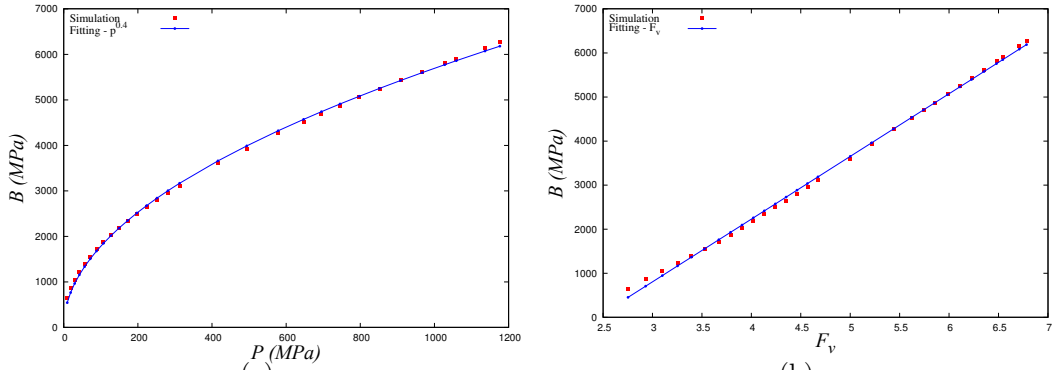


Figure 5.6: Evolution of bulk modulus B versus (a) pressure P and (b) isostropic fabric F_v

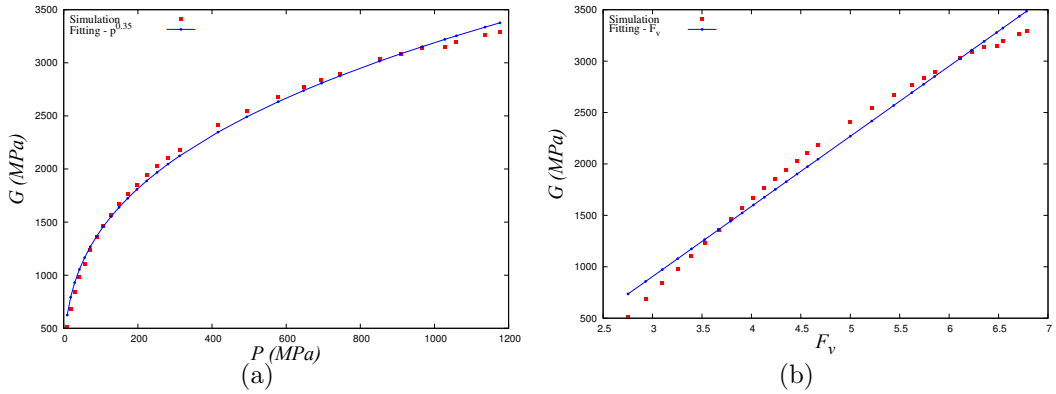


Figure 5.7: Evolution of adimensional shear modulus G^* versus (a) adimensional pressure P^* and (b) isostropic fabric F_v

Chapter 6

Granular Mixtures

Aim of this chapter is to study effects of different composition on the elastic moduli. Here, two different type of material are considered: glass (stiff) and rubber (soft).

6.1 Numerical simulation

In order to investigate the effect of adding rubber to glass volume, different glass-rubber mixtures were generated. Rubber fraction in generated specimens was varied from zero to 100 percent, shown in Tab 6.1.

Rubber %	**Glass particles**	**Rubber particles**	**Total particles**
0	4739	0	4739
5	4502	264	4766
10	4265	529	4794
15	4028	793	4821
18	3886	952	4838
20	3791	1058	4849
25	3554	1322	4876
28	3412	1480	4892
30	3317	1587	4904
32	3222	1693	4915
35	3080	1851	4931
38	2938	2010	4948
40	2843	2116	4959
45	2606	2381	4987
50	2369	2369	4738
55	2132	2910	5042
60	1895	3174	5069
70	1421	3703	5124
80	948	4232	5180
90	474	4761	5235
100	0	5290	5290

Table 6.1: Rubber fraction and number of particles

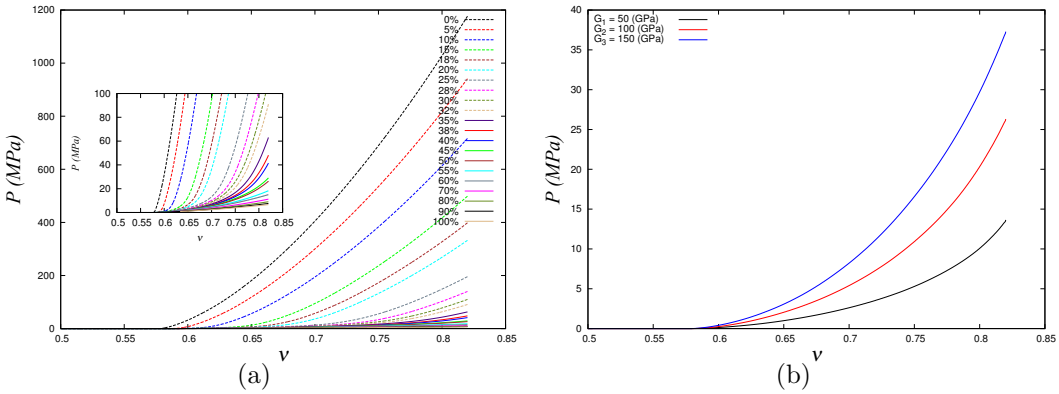
To prepare samples, the preparation procedure described in Sec. 5.3.1 is used. Hertzian contact law is employed to define the interaction between particles, since this model is more reliable rather linear model. Material parameters used in simulations are shown in Tab. 6.2. Glass is the same used in Sec. 5.4 for material paramenters see Tab. 5.2. To test the effect of softness, we use three different types of rubber with same density ρ and Poisson's ratio ν, but different shear moudlus: $G_1 = 5 \cdot 10^7[Pa]$, $G_2 = 10 \cdot 10^7[Pa]$ and $G_3 = 15 \cdot 10^7[Pa]$. This means that we have different values of Effective Young's modulus E^* and tangential stiffness k_t. Hertz viscosity parameters for each type of rubber are choosen to give a restitution coefficient of $e = 0.7$.

System parameters				
		Reality		**Code**
Mean radius	$< r >$	$2 \cdot 10^{-3}[m]$	$\leftrightarrow$	2
Polydispersity	W	3	$\leftrightarrow$	3
Density	ρ	$1270[kg/m^3]$	$\leftrightarrow$	1270
Effective Young's modulus 1	E_1^*	$6.6667 \cdot 10^7[Pa]$	$\leftrightarrow$	66.667
Effective Young's modulus 2	E_2^*	$1.3333 \cdot 10^8[Pa]$	$\leftrightarrow$	$1.3333 \cdot 10^2$
Effective Young's modulus 3	E_3^*	$2 \cdot 10^8[Pa]$	$\leftrightarrow$	200
Tangential stiffness 1	k_{t1}	$25 \cdot 10^3[kg/s^s]$	$\leftrightarrow$	25
Tangential stiffness 2	k_{t2}	$50 \cdot 10^3[kg/s^s]$	$\leftrightarrow$	50
Tangential stiffness 3	k_{t3}	$75 \cdot 10^3[kg/s^s]$	$\leftrightarrow$	75
Hertz viscosity parameter 1	η_{H1}	$28.46[kg/\sqrt{m}s]$	$\leftrightarrow$	900
Hertz viscosity parameter 2	η_{H2}	$44.27[kg/\sqrt{m}s]$	$\leftrightarrow$	1400
Hertz viscosity parameter 3	η_{H3}	$53.76[kg/\sqrt{m}s]$	$\leftrightarrow$	1700
Tangential damping 1	γ_t	$0.18[kg/s]$	$\leftrightarrow$	180
Tangential damping 2	γ_t	$0.28[kg/s]$	$\leftrightarrow$	280
Tangential damping 3	γ_t	$0.34[kg/s]$	$\leftrightarrow$	340
Background damping	γ_b	$0.1[kg/s]$	$\leftrightarrow$	100
Time step	Δt	$10^{-8}[s]$	$\leftrightarrow$	0.01

Table 6.2: Simulation parameters

6.2 The effect of rubber

After preparation of samples, the pressure of samples are calculated during decompression phase and are plotted against volume fraction for different amount of rubber. Fig. 6.1(a) shows that the value of pressure starts to decrease by adding soft particles (rubber) into samples. Due to high deformation ability of rubber, samples with high amount of rubber are deformed easily (*i.e.* less amount of pressure required). Since glass particles are pretty stiff, samples with less amount of rubbers require high pressure.
Further, we studied the effect of rubber stiffness (Fig. 6.1(b)). As can be seen in Fig. 6.1(b), increasing the stiffness of rubber causes to increase of pressure for the sample where rubber content is 50 %.

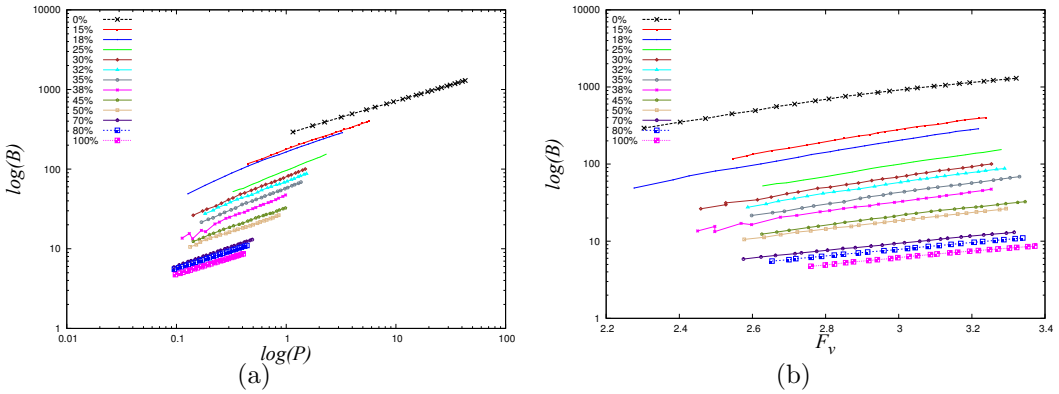


(a) (b)

Figure 6.1: Evolution of pressure P versus (a) different rubber percentage and (b) different rubber stiffness

6.2.1 Evolution of elastic moduli

In this section we report the evolution of bulk B (Fig. 6.2) and shear G (Fig. 6.3) modulus versus pressure P and isotropic fabric F_v for different sample compositions.
We can apreciate that both muduli are linearly scaled with P and F_v with a same slope.
We see also that by increasing of rubber content, B and G decrease.
Note that muduli plots versus P are log-log and plots versus F_v only moduli are logaritmic.



(a) (b)

Figure 6.2: (a) Evolution of bulk modulus B with pressure P for different rubber percentages, as shown in the legend, (b) Evolution of bulk modulus B versus isotropic fabric F_v for different rubber percentages, as shown in the legend.

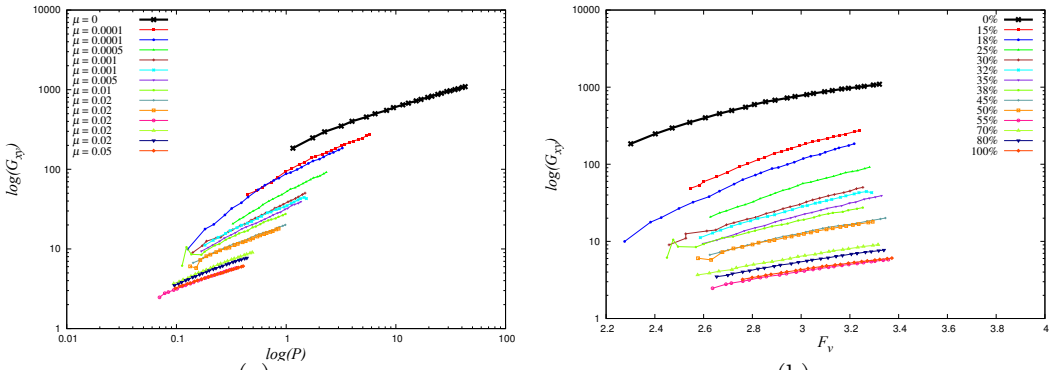


Figure 6.3: (a) Evolution of shear modulus G with pressure P for different rubber percentages, as shown in the legend, (b) Evolution of shear modulus B versus isotropic fabric F_v for different rubber percentages, as shown in the legend.

Experimental comparison

At the end, our simulation results compared with existing experimental data of Kim and Santamarina [15]. Fig. 6.a in [15] shows the same behaviour as we obtained for the bulk modulus (Fig. 6.2), and our shear modulus plots (Fig. 6.3) is consistent with Fig. 9.a in [15]. This comparison is validating our simulation results and protocol.

Chapter 7

Conclusions

In a triaxial box, elastic moduli are measured, which describe the incremental elastic response of relaxed granular materials to applied small strain perturbations (isotropic compression or pure shear). The tested states have experienced different deformation hisotry, since the particles have different properties already during preparation of the tests, like the coefficient of inter-particle contact friction or particle stiffness. In this thesis we have focused on the effect of friction and mixture of particles on the macroscopic elastic moduli over a wide range of volume fractions. We used two different models for normal force: linear model for fictional packings and Hertz model for mixtures.
We found that, in samples with different friction, bulk modulus always increases with increasing volume fraction and this behaviour is slower for packings with high friction. When we plot bulk modulus against isotropic fabric, the data approximately collapse on an unique curve: this means that there is a general relation between bulk stiffness and isotropic micro-structure. Shear modulus has the same behaviour of bulk when plotted against volume fraction, but different versus isotropic fabric, which means that other microscopic parameters are involved.
In the last chapter, we studied the effect of particle stiffness where two different type of particles (soft and stiff) were mixed. As it has been shown, while the number of soft particles increases, required pressure to compress the sample decreases.
Finally, the elastic moduli were calculated for samples with different granular composition. When the elastic moduli were plotted against pressure in a log-log scale plot, it was observed that all different composition follow the same trend with having a unique slope. To complement our observations and conclusions, we compared the simulation data with an existing experimental data. The experimental data confirm that our protocol is correct.

Bibliography

[1] *Lecture notes, Advanced Programming in Engineering*, chapter 3. MSM, University of Twente, 2014/2015.

[2] T. I. Addenbrooke, D. M. Potts, and A. M. Puzrin. The influence of pre-failure soil stiffness on the numerical analysis of tunnel construction. *Pre-failure Deformation Behaviour of Geomaterials*, page 307, 1998.

[3] J. P. Bardet. Numerical simulations of the incremental responses of idealized granular materials. *International Journal of Plasticity*, 10(8):879–908, 1994.

[4] D. Beeman. Some multistep methods for use in molecular dynamics calculations. *Journal of Computational Physics*, 20(2):130–139, 1976.

[5] C. S. Campbell. Granular material flows – An overview. *Powder Technol.*, 162:208–229, 2006.

[6] C. R. I. Clayton. Stiffness at small strain: research and practice. *Géotechnique*, 61(1):5–37, 2011.

[7] P. A. Cundall and O. D. L. Strack. A discrete numerical model for granular assemblies. *Géotechnique*, 29(1):47–65, 1979.

[8] P. G. de Gennes. Granular Matter: a tentative view. *Rev.simMod.simPhys.*, 71:5374, 1999.

[9] B. J. Ennis, J. Green, and R. Davies. The legacy of neglect in the US. *Chemical engineering progress*, 90(4):32–43, 1994.

[10] J. D. Goddard. Nonlinear elasticity and pressure-dependent wave speeds in granular media. *Proc. Roy. Soc. London*, 430:105–131, 1990.

[11] H. Hertz. Uber die Berührung fester elastischer Körper. *Journal für die reine und angewandte Mathematik*, 92(156-171):22, 1882.

[12] O. I. Imole. *Discrete Elements Simulations and Experiments: Toward Applications for Cohesive Powders*. PhD thesis, University of Twente, 2014.

[13] O. I. Imole, N. Kumar, V. Magnanimo, and S. Luding. Hydrostatic and Shear Behavior of Frictionless Granular Assemblies under Different Deformation Conditions. *Kona Powder Part. J.*, 30(30):84–108, 2013.

[14] K. L. Johnson. *Contact Mechanics*. Cambridge Univ. Press, Cambridge, 1989.

[15] H. K. Kim and J. C. Santamarina. Sand-rubber mixtures (large rubber chips). *Canadian Geotechnical Journal*, 45(10):1457–1466, 2008.

[16] N. Kumar. *Micro-Macro and jamming transition in granular materials*. PhD thesis, University of Twente, 2014.

[17] N. Kumar, S. Luding, and V. Magnanimo. Macroscopic model with anisotropy based on micro–macro information. *Acta Mechanica*, 225(8):2319–2343, 2014.

[18] G. Kuwabara and K. Kono. Restitution Coefficient in a Collision between two Spheres. *Jpn. J. Appl. Phys.*, 26(8):1230–1233, 1987.

[19] L. D. Landau and E. M. Lifschitz. *Elastizit"atstheorie*. Akademie Verlag Dresden, Berlin, 1989.

[20] J. Lee. Density waves in the flows of granular media. *Phys. Rev. E*, 49(1):281, 1994.

[21] S. Luding. Cohesive, frictional powders: contact models for tension. *Granul. Matter.*, 10(4):235–246, 2008.

[22] S. Luding. Introduction to discrete element methods: Basics of contact force models and how to perform the micro-marco transition to continuum theory. *Eur. J. Envir. Civ. Eng.*, 12(7-8):785–826, 2008.

[23] S. Luding, E. Clément, A. Blumen, J. Rajchenbach, and J. Duran. Onset of convection in molecular dynamics simulations of grains. *Phys. Rev. E*, 50:R1762–R1765, 1994.

[24] V. Magnanimo and L. La Ragione. A micromechanical numerical analysis for a triaxial compression of granular materials. In *POWDERS AND GRAINS 2013: Proceedings of the 7th International Conference on Micromechanics of Granular Media*, volume 1542, pages 1234–1237. AIP Publishing, 2013.

[25] V. Magnanimo, L. La Ragione, J. T. Jenkins, P. Wang, and H. A. Makse. Characterizing the shear and bulk moduli of an idealized granular material. *Europhys. Lett.*, 81:34006, 2008.

[26] H. A. Makse, N. Gland, D. L. Johnson, and L. Schwartz. Granular packings: Nonlinear elasticity, sound propagation, and collective relaxation dynamics. *Physical Review E.*, 70(6):061302, 2004.

[27] E. Merrow. Estimating startup times for solids-processing plants. *Chemical engineering*, 95:89–92, 1988.

[28] C. S. O'Hern, S. A. Langer, A. Liu, and S. R. Nagel. Force distributions near jamming and glass transitions. *Phys. Rev. Lett..*, 86:111–114, 2001.

[29] T. Pöschel. Granular Material Flowing Down an Inclined Chute: A Molecular Dynamics Simulation. *J. Phys. II*, 3:27, 1993.

[30] G. H. Ristow. Simulating Granular Flow with Molecular Dynamics. *J. Phys. I*, 2(6):649, 1992.

[31] M. H. Sadd, Q. M. Tai, and A. Shukla. Contact Law effects on Wave Propagation in Particulate Materials using distinct element modeling. *Int. J. Non-Linear Mechanics*, 28(2):251–265, 1993.

[32] J. Shäfer, S. Dippel, and D. E. Wolf. Force Schemes in Simulations of Granular Materials. *J. Phys. I (France)*, 6(1):5–20, 1996.

[33] L. Sibille, F. Nicot, and Donzé.

[34] F. Spahn, J. M. Hertzsch, and N. V. Brilliantov. The role of particle collisions for the dynamics in planetary rings. *Chaos, Solitons Fractals*, 5:1945, 1995.

[35] K. Taghizadeh, N. Kumar, V. Magnanimo, and S. Luding. Understanding the effects of inter-particle contact friction on the elastic moduli of granular materials. In *IOP Conference Series: Earth and Environmental Science*, volume 26, page 012008. IOP Publishing, 2015.

[36] Y. Taguchi. New Origin of a Convective Motion: Elastically Induced Convection in Granular Materials. *Phys. Rev. Lett.*, 69(9):1367, 1992.

[37] C. Thornton and C. W. Randall. Applications of theoretical contact mechanics to solid particle system simulation. In *Micromechanics of granular media*, Amsterdam, 1988. Elsevier.

[38] J. R. Valdes and T. M. Evans. Sand-rubber mixtures: experiments and numerical simulations. *Canadian Geotechnical Journal*, 45(4):588–595, 2008.

[39] O. R. Walton and R. L. Braun. Viscosity, granular-temperature, and stress calculations for shearing assemblies of inelastic, frictional disks. *Journal of Rheology*, 30:949–980, 1986.

[40] O. R. Walton, D. M. Maddix, T. R. Butkovich, and F. E. Heuzé. Redirection of dynamic compressive Waves in Materials with nearly orthogonal and random joint sets. In M Massoudi and K R Rajagopal, editors, *AMD*, volume 117, 1991.

[41] J. R. Williams. Contact analysis of large numbers of interacting bodies using discrete modal methods for simulating material failure on the microscopic scale. *Eng. Comput.*, 5, 1988.

Appendix A

MATLAB

A.1 Scripts for examples 2.7

A.1.1 Two masses with spring

```
%Set system parameters
k = 1; %stifness of the spring
m = 1; %mass
xe = 2; %equilibrium lenght
%Set initial conditions
x0 = 0; x4 = 5*xe; %position of the walls
x10 = xe; x20 = 2*xe; x30 = 3*xe; %initial position of the particles
v10 = 0; v20 = 0; v30 = 0; %initial velocity of the particles
x_zero = [x10 x20 x30]; %initial position vector
v_zero = [v10 v20 v30]; %initial velocity vector
%Set simulation parameters
tf =15; %time of simulation
delta_t =0.01; %time step
Nt = ceil(tf/delta_t); %number of time increments

%Initialize the loop variables
for n = 1 : Nt+1
    if n == 1
        x(n,:) = [x0, x_zero, x4];
        v(n,:) = [0, v_zero, 0];
        for j = 1 : 5
            if j == 1
                f(n,j) = 0;
            elseif j >=2 & j<=4
                f(n,j) = (k * (x(n,j+1) - 2 * x(n,j) + x(n,j-1)));
            else
                f(n,j) = 0;
            end
        end
        a(n,:) = f(n,:) ./ m;
        alfa(n,:) = x(n,:) + (delta_t * v(n,:)) + (0.5 * a(n,:) * ...
        (delta_t^2)); %uniformly accelerated motion
    else
        x(n,:) = alfa(n-1,:);
```

```
        for j = 1 : 5
            if j == 1
                f(n,j) = 0;
            elseif j >=2 & j<=4
                f(n,j) =(k * (x(n,j+1) - 2 * x(n,j) + x(n,j-1)));%Verlet
            else
                f(n,j) = 0;
            end
        end
        a(n,:) = f(n,:) ./ m;
        alfa(n,:) = (2 * x(n,:)) - x(n-1,:) + ((delta_t^2) * a(n,:));
        v(n,:) = (alfa(n,:) - x(n-1,:))/(2 * delta_t);
    end
end

%Analitical solution
x_rel = x(:,2) - x(:,1) - xe;
x_an = -2:0.1:2;
F1 = k * x_an;

%Conservation of energy
for n = 1 : Nt+1
    Ek1(n,1) = 0.5 * m * (v(n,2))^2; %kinetic energy of mass1
    Ek2(n,1) = 0.5 * m * (v(n,3))^2; %kinetic energy of mass2
    Ek3(n,1) = 0.5 * m * (v(n,4))^2; %kinetic energy of mass2
    Ek(n,1) = Ek1(n,1) + Ek2(n,1) + Ek3(n,1); % system kinetic energy
    Ep(n,1) = (0.5 * k * (x(n,2) - x(n,1) - xe)^2) + ...
    (0.5 * k * (x(n,3) - x(n,2) - xe)^2) + ...
    (0.5 * k * (x(n,4) - x(n,3) - xe)^2) + ...
    (0.5 * k * (x(n,5) - x(n,4) - xe)^2); % system potential energy
    Etot(n,1) = Ek(n,1) + Ep(n,1); %total energy
end
```

A.1.2 Three masses with spring

For the simulation of the two fixes walls, they are considered as two particles that can not move.

```
%Set system parameters
k = 1; %stifness of the spring
m = 1; %mass
xe = 2; %equilibrium lenght
%Set initial conditions
x0 = 0; x4 = 5*xe; %position of the walls
x10 = xe; x20 = 2*xe; x30 = 3*xe; %initial position of the particles
v10 = 0; v20 = 0; v30 = 0; %initial velocity of the particles
x_zero = [x10 x20 x30]; %initial position vector
v_zero = [v10 v20 v30]; %initial velocity vector
%Set simulation parameters
tf =15; %time of simulation
delta_t =0.01; %time step
Nt = ceil(tf/delta_t); %number of time increments
```

```
%Initialize the loop variables
for n = 1 : Nt+1
    if n == 1
        x(n,:) = [x0, x_zero, x4];
        v(n,:) = [0, v_zero, 0];
        for j = 1 : 5
            if j == 1
                f(n,j) = 0;
            elseif j >=2 & j<=4
                f(n,j) = (k * (x(n,j+1) - 2 * x(n,j) + x(n,j-1)));
            else
                f(n,j) = 0;
            end
        end
        a(n,:) = f(n,:) ./ m;
        alfa(n,:) = x(n,:) + (delta_t * v(n,:)) + (0.5 * a(n,:) * ...
        (delta_t^2)); %uniformly accelerated motion
    else
        x(n,:) = alfa(n-1,:);
        for j = 1 : 5
            if j == 1
                f(n,j) = 0;
            elseif j >=2 & j<=4
                f(n,j) =(k * (x(n,j+1) - 2 * x(n,j) + x(n,j-1)));%Verlet
            else
                f(n,j) = 0;
            end
        end
        a(n,:) = f(n,:) ./ m;
        alfa(n,:) = (2 * x(n,:)) - x(n-1,:) + ((delta_t^2) * a(n,:));
        v(n,:) = (alfa(n,:) - x(n-1,:))/(2 * delta_t);
    end
end

%Conservation of energy
for n = 1 : Nt+1
    Ek1(n,1) = 0.5 * m * (v(n,2))^2; %kinetic energy of mass1
    Ek2(n,1) = 0.5 * m * (v(n,3))^2; %kinetic energy of mass2
    Ek3(n,1) = 0.5 * m * (v(n,4))^2; %kinetic energy of mass2
    Ek(n,1) = Ek1(n,1) + Ek2(n,1) + Ek3(n,1); %system kinetic energy
    Ep(n,1) = (0.5 * k * (x(n,2) - x(n,1) - xe)^2) + ...
    (0.5 * k * (x(n,3) - x(n,2) - xe)^2) + ...
    (0.5 * k * (x(n,4) - x(n,3) - xe)^2) + ...
    (0.5 * k * (x(n,5) - x(n,4) - xe)^2); %system potential energy
    Etot(n,1) = Ek(n,1) + Ep(n,1);
end
```

A.1.3 Two masses with spring 2D

```
%Set system parameters
k =1; %stiffness of the spring
m =1; %mass
xe =10; %equilibrium lenght
```

```
%Set initial conditions
x10 = 0  ; y10 = 0 ; x20 =xe ; y20 = xe ; %particles initial position
vx10 =0; vy10= 0; vx20 = 1; vy20 = 1; %particles initial velocity
x_zero = [x10 y10 x20 y20]; %initial position vector
v_zero = [vx10 vy10 vx20 vy20]; %initial velocity vector
%Set simulation parameters
tf =10; %time of simulation
delta_t =0.01; %time step
Nt = ceil(tf/delta_t); %number of time increments

%Initialize the loop variables
for n = 1 : Nt+1
    if n == 1
        x(n,:) = x_zero; %position
        v(n,:) = v_zero; %velocity
        x_p(n,:) = [(x(n,1)^2 + x(n,2)^2)^(1/2) ...
        (x(n,3)^2 + x(n,4)^2)^(1/2)]; %total position
        v_p(n,:) = [((v(n,1)^2 + v(n,2)^2)^(1/2))*sign(v(n,1)) ...
        ((v(n,3)^2 + v(n,4)^2)^(1/2))*sign(v(n,3))]; %total velocity
        f(n,:) = (k * (x_p(n,2) - x_p(n,1) - xe)) * [1 -1]; %force
        a_p(n,:) = f(n,:) ./ m; %total acceleration of each particle
        alfa(n,:) =x_p(n,:)+(delta_t * v_p(n,:))+(0.5 * a_p(n,:) * ...
        (delta_t^2)); %uniformly accelerated motion
    else
        x_p(n,:) = alfa(n-1,:);
        f(n,:) = (k * (x_p(n,2) - x_p(n,1) - xe)) * [1 -1];
        a_p(n,:) = f(n,:) ./ m;
        alfa(n,:) = (2 * x_p(n,:)) - x_p(n-1,:) + ...
        ((delta_t^2) * a_p(n,:)); %Verlet
        v_p(n,:) = (alfa(n,:) - x_p(n-1,:))/(2 * delta_t); %Verlet
    end
end

%Conservation of energy
for n = 1 : Nt+1
    Ek1(n,1) = 0.5 * m * (v_p(n,1))^2; %kinetic energy of mass1
    Ek2(n,1) = 0.5 * m * (v_p(n,2))^2; %kinetic energy of mass2
    Ek(n,1) = Ek1(n,1) + Ek2(n,1); %system kinetic energy
    Ep(n,1) = 0.5*k*(x_p(n,2)-x_p(n,1)-xe)^2; %system potential energy
    Etot(n,1) = Ek(n,1) + Ep(n,1);
end
```

A.2 Scripts for examples 2.11

It's important to note that if there is viscosity (damping case) it's impossible to use the Verlet method, due to its definition of velocity. I decided to use the classical physical laws of motion.

A.2.1 Two masses with a spring and a damper

```
%Set system parameters
```

```
k =80; %stiffness of the spring
gamma=1; %damping
m =1; %mass
xe =10; %equilibrium lenght
%Set initial conditions
x10 =0; x20 =xe + 1; %initial position of the particles
v10 =0; v20 =1.5; %initial velocity of the particles
x_zero = [x10 x20]; %initial position vector
v_zero = [v10 v20]; %initial velocity vector
%Set simulation parameters
tf =10; %time of simulation
delta_t =0.01; %time step
Nt = ceil(tf/delta_t); %number of time increments

%Initialize the loop variables
for n = 1 : Nt+1
    if n == 1
        x(n,:) = x_zero; %position
        v(n,:) = v_zero; %velocity
        f(n,:) = ((k .* (x(n,2) - x(n,1)- xe)) + ...
        (gamma.*(v(n,2) - v(n,1))))*[1 -1]; %force
        a(n,:) = f(n,:) ./ m; %acceleration
        alfa(n,:) = x(n,:) + (delta_t * v(n,:)) + (0.5 * a(n,:) * ...
        (delta_t^2)); %uniformly accelerated motion
        beta(n,:) = v(n,:) + (delta_t * a(n,:)); %accelerated motion
    else
        x(n,:) = alfa(n-1,:);
        v(n,:) = beta(n-1,:);
        f(n,:) = ((k * (x(n,2) - x(n,1)- xe)) + ...
        (gamma .* (v(n,2) - v(n,1)))) * [1 -1];
        a(n,:) = f(n,:) ./ m;
        alfa(n,:) = x(n,:) + (delta_t * v(n,:)) + (0.5 * a(n,:) * ...
        (delta_t^2)); %uniformly accelerated motion
        beta(n,:) = v(n,:) + (delta_t * a(n,:)); %accelerated motion
    end
end

%Conservation of energy
for n = 1 : Nt+1
    Ek1(n,1) = 0.5 * m * (v(n,1))^2; %kinetic energy of mass1
    Ek2(n,1) = 0.5 * m * (v(n,2))^2; %kinetic energy of mass2
    Ek(n,1) = Ek1(n,1) + Ek2(n,1); %system kinetic energy
    Ep(n,1) =0.5*k*(x(n,2) - x(n,1) - xe)^2; %system potential energy
    Etot(n,1) = Ek(n,1) + Ep(n,1);
end
```

A.2.2 Two masses with a spring and a damper 2D

```
%Set system parameters
k =80; %stiffness of the spring
gamma=1; %damping
m =1; %mass
xe =10; %equilibrium lenght
```

```
%Set initial conditions
x10 = 0  ; y10 = 0 ; x20 =xe ; y20 = xe ; %particles initial position
vx10 =0; vy10= 0; vx20 = 1; vy20 = 1; %particles initial velocity
x_zero = [x10 y10 x20 y20]; %initial position vector
v_zero = [vx10 vy10 vx20 vy20]; %initial velocity vector
%Set simulation parameters
tf =5; %time of simulation
delta_t =0.01; %time step
Nt = ceil(tf/delta_t); %number of time increments

%Initialize the loop variables
for n = 1 : Nt+1
    if n == 1
        x(n,:) = x_zero; %position
        v(n,:) = v_zero; %velocity
        x_p(n,:) = [(x(n,1)^2 + x(n,2)^2)^(1/2) ...
        (x(n,3)^2 + x(n,4)^2)^(1/2)]; %total position
        v_p(n,:) = [((v(n,1)^2 + v(n,2)^2)^(1/2))*sign(v(n,1)) ...
        ((v(n,3)^2 + v(n,4)^2)^(1/2))*sign(v(n,3))]; %total velocity
        f(n,:) = ((k .* (x_p(n,2) - x_p(n,1)- xe)) + ...
        (gamma .* (v_p(n,2) - v_p(n,1)))) * [1 -1]; %force
        a_p(n,:) = f(n,:) ./ m; %total acceleration of each particle
        alfa(n,:) = x_p(n,:) + (delta_t * v_p(n,:)) + ...
        (0.5 * a_p(n,:) * (delta_t^2)); %uniformly accelerated motion
        beta(n,:) = v_p(n,:) + (delta_t * a_p(n,:)); %accelerated motion
    else
        x_p(n,:) = alfa(n-1,:);
        v_p(n,:) = beta(n-1,:);
        f(n,:) = ((k .* (x_p(n,2) - x_p(n,1)- xe)) + ...
        (gamma .* (v_p(n,2) - v_p(n,1)))) * [1 -1]; %force
        a_p(n,:) = f(n,:) ./ m;
        alfa(n,:) = x_p(n,:) + (delta_t * v_p(n,:)) + ...
        (0.5 * a_p(n,:) * (delta_t^2)); %uniformly accelerated motion
        beta(n,:) = v_p(n,:) + (delta_t * a_p(n,:)); %accelerated motion
    end
end

%Conservation of energy
for n = 1 : Nt+1
    Ek1(n,1) = 0.5 * m * (v_p(n,1))^2; %kinetic energy of mass1
    Ek2(n,1) = 0.5 * m * (v_p(n,2))^2; %kinetic energy of mass2
    Ek(n,1) = Ek1(n,1) + Ek2(n,1); %system kinetic energy
    Ep(n,1) =0.5*k*(x_p(n,2)-x_p(n,1)-xe)^2;%system potential energy
    Etot(n,1) = Ek(n,1) + Ep(n,1);
end
```

A.3 Scripts for examples 3.3

For the calculation of potential energy in particle collision is foundamental to remember that if overlap is negative ,$\delta < 0$, there is no contact between particles and so any accumulation of potential energy. It's important to note that if there is viscosity (damping case) it's impossible to use the Verlet method, due to its definition of velocity. I decided to use the classical physical laws of motion.

A.3.1 Two particles 1D without damping

```
close all
clear all
clc
%Set system parameters
k =1; %material stiffness
m=1; %mass of particle
R1=10; %radius of particle 1
R2=10; %radius of particle 2
%Set initial conditions
x10=0; x20=50; %initial position of the two particles
v10 =2; v20 =-2; %initial velocity of the particles
x_zero = [x10 x20]'; %initial position vector
v_zero = [v10 v20]'; %initial velocity vector
%Set simulation parameters
tf =15; %time of simulation
delta_t =0.1; %time step
Nt = ceil(tf/delta_t); %number of time increments

%Initialize the loop variables
for n = 1 : Nt+1
    if n == 1
        x(n,:) = x_zero; %position
        v(n,:) = v_zero; %velocity
        over(n,1) = (R1 + R2) - (x(n,2) - x(n,1));%definition of overlap
        if over(n,1) <= 0
            f(n,:) = [0 0];
            a(n,:) = f(n,:) ./ m;
            alfa(n,:) = x(n,:)+(delta_t * v(n,:));%linear uniform motion
            beta(n,:) = v(n,:); %there is not acceleration
        else
            f(n, :) = (k * over(n,1)) * [-1 1];
            a(n,:) = f(n,:) ./ m;
            alfa(n,:) = x(n,:) + (delta_t * v(n,:)) + ...
            (0.5 * a(n,:) * (delta_t^2)); %uniformly accelerated motion
            beta(n,:) = v(n,:) + (delta_t * a(n,:)); %accelerated motion
        end
    else
        x(n,:) = alfa(n-1,:);
        over(n,1) = (R1 + R2) - (x(n,2) - x(n,1));%definition of overlap
        if over(n,1) <= 0
            v(n,:) = beta(n-1,:);
            f(n,:) = [0 0];
            a(n,:) = [0 0];
            alfa(n,:) = x(n,:)+(delta_t * v(n,:));%linear uniform motion
            beta(n,:) = v(n,:); %there is not acceleration
        else
            f(n, :) = (k * over(n,1)) * [-1 1];
            a(n,:) = f(n,:) ./ m;
            alfa(n,:) = (2 * x(n,:)) - x(n-1,:) +...
            ((delta_t^2) * a(n,:)); %Verlet
            beta(n,:) = (alfa(n,:) - x(n-1,:))/(2 * delta_t); %Verlet
            v(n,:) = beta(n,:);
```

```
        end
    end
end

%Analitical Solution
x11 = 15 : 0.1 : 16.4;
x12 = 16.4 : -0.1 : 15;
x21 = 35 : -0.1 : 33.6;
x22 = 33.6 : 0.1: 35;
x_rel1 = x21 - x11;
x_rel2 = x22 - x12;
over1 = (R1+R2) - (x_rel1);
over2 = (R1+R2) - (x_rel2);
F1 = -k * over1;
F2 = -k * over2;

%Conservation of energy
for n = 1 : Nt+1
    Ek1(n,1) = 0.5 * m * (v(n,1))^2; %kinetic energy of particle 1
    Ek2(n,1) = 0.5 * m * (v(n,2))^2; %kinetic energy of particle 2
    Ek(n,1) = Ek1(n,1) + Ek2(n,1); %kinetic energy of the system
    if over(n,1) <= 0
        Ep(n,1) = 0;
    else
        Ep(n,1) = 0.5 * k * (over(n,1))^2;
    end
    Etot(n,1) = Ek(n,1) + Ep(n,1); %total energy of the system
end
```

A.3.2 Two particles 2D without damping

```
%Set system parameters
k =1; %material stiffness
m=1; %mass of particle
R1=10; %radius of particle 1
R2=10; %radius of particle 2
%Set initial conditions
x10 = 0  ; y10 = 0 ; x20 =50 ; y20 = 50 ; %particles initial position
vx10 = 2; vy10= 2; vx20 = -2; vy20 = -2; %particles initial velocity
x_zero = [x10 y10 x20 y20]; %initial position vector
v_zero = [vx10 vy10 vx20 vy20]; %initial velocity vector
%Set simulation parameters
tf =15; %time of simulation
delta_t =0.1; %time step
Nt = ceil(tf/delta_t); %number of time increments

%Initialize the loop variables
for n = 1 : Nt+1
    if n == 1
        x(n,:) = x_zero; %position
        v(n,:) = v_zero; %velocity
        x_p(n,:) = [(x(n,1)^2 + x(n,2)^2)^(1/2) ...
        (x(n,3)^2 + x(n,4)^2)^(1/2)]; %total position
```

```
        v_p(n,:) = [((v(n,1)^2 + v(n,2)^2)^(1/2))*sign(v(n,1)) ...
        ((v(n,3)^2 + v(n,4)^2)^(1/2))*sign(v(n,3))]; %total velocity
        over(n,1) = (R1 + R2) - (x_p(n,2) - x_p(n,1));%overlap
        if over(n,1) <= 0
            f(n,:) = [0 0]; %force
            a_p(n,:) = f(n,:) ./ m; %acceleration
            alfa(n,:)=x_p(n,:)+(delta_t*v_p(n,:));%linear uniform motion
            beta(n,:) = v_p(n,:); %there is not acceleration
        else
            f(n, :) = (k * over(n,1)) * [-1 1];
            a_p(n,:) = f(n,:) ./ m;
            alfa(n,:) = x_p(n,:) + (delta_t * v_p(n,:)) + ...
            (0.5 * a_p(n,:) * (delta_t^2));%uniformly accelerated motion
            beta(n,:) =v_p(n,:)+(delta_t * a_p(n,:));%accelerated motion
        end
    else
        x_p(n,:) = alfa(n-1,:);
        over(n,1) = (R1 + R2)-(x_p(n,2) - x_p(n,1));%overlap
        if over(n,1) <= 0
            v_p(n,:) = beta(n-1,:);
            f(n,:) = [0 0];
            a_p(n,:) = [0 0];
            alfa(n,:)=x_p(n,:)+(delta_t*v_p(n,:));%linear uniform motion
            beta(n,:) = v_p(n,:); %there is not acceleration
        else
            f(n, :) = (k * over(n,1)) * [-1 1];
            a_p(n,:) = f(n,:) ./ m;
            alfa(n,:) = (2 * x_p(n,:)) - x_p(n-1,:) + ...
            ((delta_t^2) * a_p(n,:)); %Verlet
            beta(n,:) = (alfa(n,:) - x_p(n-1,:))/(2 * delta_t); %Verlet
            v_p(n,:) = beta(n,:);
        end
    end
end

%Conservation of energy
for n = 1 : Nt+1
    Ek1(n,1) = 0.5 * m * (v_p(n,1))^2; %kinetic energy of particle 1
    Ek2(n,1) = 0.5 * m * (v_p(n,2))^2; %kinetic energy of particle 2
    Ek(n,1) = Ek1(n,1) + Ek2(n,1); %kinetic energy of the system
    if over(n,1) <= 0
        Ep(n,1) = 0;
    else
        Ep(n,1) = 0.5 * k * (over(n,1))^2;
    end
    Etot(n,1) = Ek(n,1) + Ep(n,1); %total energy of the system
end
```

A.3.3 Two particles 1D with damping

```
%Set system parameters
k = 100; %material stiffness
gamma = 1; %material viscosity
```

```
m = 1; %mass of particles
R1= 10; %radius of particle 1
R2= 10; %radius of particle 2
%Set initial conditions
x10= 0; x20= 30; %initial position of the two particles
v10 = 2; v20 = -2; %initial velocity of the particles
x_zero = [x10 x20]'; %initial position vector
v_zero = [v10 v20]'; %initial velocity vector
%Set simulation parameters
tf = 5; %time of simulation
delta_t = 0.0001; %time step
Nt = ceil(tf/delta_t); %number of time increments

%Initialize the loop variables
for n = 1 : Nt+1
    if n == 1
        x(n,:) = x_zero;
        v(n,:) = v_zero;
        over(n,1) = (R1 + R2) - (x(n,2) - x(n,1)); %overlap
        if over(n,1) <= 0
            f(n,:) = [0 0];
            a(n,:) = f(n,:) ./ m;
            alfa(n,:) =x(n,:)+(delta_t*v(n,:)); %linear uniform motion
            beta(n,:) = v(n,:); %there is not acceleration
        else
            f(n, :) =((k * over(n,1))+(gamma*(v(n,1) - v(n,2))))*[-1 1];
            a(n,:) = f(n,:) ./ m;
            alfa(n,:) = x(n,:) + (delta_t * v(n,:)) + ...
            (0.5 * a(n,:) * (delta_t^2)); %uniformly accelerated motion
            beta(n,:) = v(n,:)+(delta_t * a(n,:));%accelerated motion
        end
    else
        x(n,:) = alfa(n-1,:);
        v(n,:) = beta(n-1,:);
        over(n,1) = (R1 + R2) - (x(n,2) - x(n,1)); %overlap
        if over(n,1) <= 0
            f(n,:) = [0 0];
            a(n,:) = [0 0];
            alfa(n,:) = x(n,:) + (delta_t * v(n,:)); %uniform motion
            beta(n,:) = v(n,:); %there is not acceleration
        else
            f(n, :) =((k * over(n,1))+(gamma*(v(n,1)-v(n,2))))*[-1 1];
            a(n,:) = f(n,:) ./ m;
            alfa(n,:) = x(n,:) + (delta_t * v(n,:)) + ...
            (0.5 * a(n,:) * (delta_t^2)); %uniformly accelerated motion
            beta(n,:) = v(n,:) + (delta_t * a(n,:));%accelerated motion
        end
    end
end

%Conservation of energy
for n = 1 : Nt+1
    Ek1(n,1) = 0.5 * m * (v(n,1))^2; %kinetic energy of particle 1
    Ek2(n,1) = 0.5 * m * (v(n,2))^2; %kinetic energy of particle 2
    Ek(n,1) = Ek1(n,1) + Ek2(n,1); %kinetic energy of the system
    if over(n,1) <= 0
```

```
        Ep(n,1) = 0;
    else
        Ep(n,1) = 0.5 * k * (over(n,1))^2;
    end
    Etot(n,1) = Ek(n,1) + Ep(n,1); %total energy of the system
end
```

A.3.4 Two particles 2D with damping

```
%Set system parameters
k =100; %material stiffness
gamma = 1; %material viscosity
m=1; %mass of particles
R1=10; %radius of particle 1
R2=10; %radius of particle 2
%Set initial conditions
x10 = 0  ; y10 = 0 ; x20 =30 ; y20 = 30 ; %particles initial position
vx10 = 2; vy10= 2; vx20 = -2; vy20 = -2; %particles initial velocity
x_zero = [x10 y10 x20 y20]; %initial position vector
v_zero = [vx10 vy10 vx20 vy20]; %initial velocity vector
%Set simulation parameters
tf =6; %time of simulation
delta_t =0.001; %time step
Nt = ceil(tf/delta_t); %number of time increments

%Initialize the loop variables
for n = 1 : Nt+1
    if n == 1
        x(n,:) = x_zero; %position
        v(n,:) = v_zero; %velocity
        x_p(n,:) = [(x(n,1)^2 + x(n,2)^2)^(1/2) + ...
        (x(n,3)^2 + x(n,4)^2)^(1/2)]; %total position
        v_p(n,:) = [((v(n,1)^2 + v(n,2)^2)^(1/2))*sign(v(n,1)) + ...
        ((v(n,3)^2 + v(n,4)^2)^(1/2))*sign(v(n,3))]; %total velocity
        over(n,1) = (R1 + R2) - (x_p(n,2) - x_p(n,1)); %overlap
        if over(n,1) <= 0
            f(n,:) = [0 0]; %force
            a_p(n,:) = f(n,:) ./ m; %acceleration
            alfa(n,:)=x_p(n,:)+(delta_t*v_p(n,:));%linear uniform motion
            beta(n,:) = v_p(n,:); %there is not acceleration
        else
            f(n, :)=((k*over(n,1))+(gamma*(v_p(n,1)-v_p(n,2))))*[-1 1];
            a_p(n,:) = f(n,:) ./ m;
            alfa(n,:) = x_p(n,:) + (delta_t * v_p(n,:)) + ...
            (0.5 * a_p(n,:) * (delta_t^2));%uniformly accelerated motion
            beta(n,:) = v_p(n,:)+(delta_t*a_p(n,:));%accelerated motion
        end
    else
        x_p(n,:) = alfa(n-1,:);
        v_p(n,:) = beta(n-1,:);
        over(n,1) = (R1 + R2) - (x_p(n,2) - x_p(n,1)); %overlap
        if over(n,1) <= 0
            f(n,:) = [0 0];
```

```
            a_p(n,:) = [0 0];
            alfa(n,:)=x_p(n,:)+(delta_t*v_p(n,:));%linear uniform motion
            beta(n,:) = v_p(n,:); %there is not acceleration
        else
            f(n, :)=((k*over(n,1))+(gamma*(v_p(n,1)-v_p(n,2))))*[-1 1];
            a_p(n,:) = f(n,:) ./ m;
            alfa(n,:) = x_p(n,:) + (delta_t * v_p(n,:)) + ...
            (0.5 * a_p(n,:) * (delta_t^2));%uniformly accelerated motion
            beta(n,:) = v_p(n,:)+(delta_t*a_p(n,:));%accelerated motion;
        end
    end
end

%Conservation of energy
for n = 1 : Nt+1
    Ek1(n,1) = 0.5 * m * (v_p(n,1))^2; %kinetic energy of particle 1
    Ek2(n,1) = 0.5 * m * (v_p(n,2))^2; %kinetic energy of particle 2
    Ek(n,1) = Ek1(n,1) + Ek2(n,1); %kinetic energy of the system
    if over(n,1) <= 0
        Ep(n,1) = 0;
    else
        Ep(n,1) = 0.5 * k * (over(n,1))^2;
    end
    Etot(n,1) = Ek(n,1) + Ep(n,1); %total energy of the system
end
```

A.3.5 Two particles 1D without damping - Linear vs Hertz model

```
%Set system parameters
ro = 2540; %density of the two particles [kg/m^3]
R1= 0.001; %radius of particle 1 [m]
R2= 0.001; %radius of particle 2 [m]
G = 29e9; %material shear modulus [kg/ms^2]
nu = 0.2; %Poisson coefficient
m = (4/3) * pi * (R1^3) * ro; %mass of the two particles [kg]
%Hertz formula coefficients
E = 2*G * (1 + nu); %Young modulus
E_star = E/(3*(1 - (nu^2))); %effective Young modulus
d_eff = (4*R1*R2)/(R1+R2); %effective diameter
%Set initial conditions
x10= 0.048891; x20= 0.0509; %initial position of the two particles [m]
v10 = 1; v20 = -1; %initial velocity of the particles [m/s]
x_zero = [x10 x20]'; %initial position vector
v_zero = [v10 v20]'; %initial velocity vector
%Set simulation parameters
tf = 1.5e-5; %time of simulation [s]
delta_t = 1e-08; %time step [s]
Nt = ceil(tf/delta_t); %number of time increments

%Hertz model
for j = 1 : Nt+1
```

```
    if j == 1
        xh(j,:) = x_zero;
        vh(j,:) = v_zero;
        overh(j,1) = (R1 + R2) - (xh(j,2) - xh(j,1));%definition of overlap
        if overh(j,1) <= 0
            fh(j,:) = [0 0];
            ah(j,:) = fh(j,:) ./ m;
            alfah(j,:)=xh(j,:)+(delta_t*vh(j,:));%linear uniform motion
            betah(j,:) = vh(j,:); %there is not acceleration
        else
            fh(j, :)=(E_star*(d_eff^(1/2))*(overh(j,1)^(3/2)))*...
            [-1 1];%Hertz force
            ah(j,:) = fh(j,:) ./ m;
            alfah(j,:) = xh(j,:)+(delta_t * vh(j,:))+(0.5 * ah(j,:) * ...
            (delta_t^2)); %uniformly accelerated motion
            betah(j,:)=vh(j,:)+(delta_t*ah(j,:)); %accelerated motion
        end
    else
        xh(j,:) = alfah(j-1,:);
        vh(j,:) = betah(j-1,:);
        overh(j,1) = (R1 + R2) - (xh(j,2) - xh(j,1)); %definition of overlap
        if overh(j,1) <= 0
            fh(j,:) = [0 0];
            ah(j,:) = [0 0];
            alfah(j,:) = xh(j,:) + (delta_t * vh(j,:)); %uniform motion
            betah(j,:) = vh(j,:); %there is not acceleration
        else
            fh(j, :) = (E_star * (d_eff^(1/2)) * (overh(j,1)^(3/2)))*[-1 1];
            ah(j,:) = fh(j,:) ./ m;
            alfah(j,:) = xh(j,:)+(delta_t * vh(j,:))+(0.5 * ah(j,:) * ...
            (delta_t^2));
            betah(j,:) = vh(j,:) + (delta_t * ah(j,:));
        end
    end
end

%Definition of k

k = (3/2)*(E_star)*(d_eff^(1/2))*(((40/100)*max(overh))^(1/2)); %slope = k_h

%linear model
for n = 1 : Nt+1
    if n == 1
        x(n,:) = x_zero; %position
        v(n,:) = v_zero; %velocity
        over(n,1) = (R1 + R2) - (x(n,2) - x(n,1)); %definition of overlap
        if over(n,1) <= 0
            f(n,:) = [0 0];
            a(n,:) = f(n,:) ./ m;
            alfa(n,:) = x(n,:) + (delta_t * v(n,:)); %linear uniform motion
            beta(n,:) = v(n,:); %there is not acceleration
        else
            f(n, :) = (k * over(n,1)) * [-1 1];
            a(n,:) = f(n,:) ./ m;
            alfa(n,:) = x(n,:) + (delta_t * v(n,:)) + (0.5 * a(n,:) * ...
            (delta_t^2)); %uniformly accelerated motion
```

```
            beta(n,:) = v(n,:)+(delta_t * a(n,:)); %accelerated motion
        end
    else
        x(n,:) = alfa(n-1,:);
        over(n,1) = (R1 + R2) - (x(n,2) - x(n,1)); %definition of overlap
        if over(n,1) <= 0
            v(n,:) = beta(n-1,:);
            f(n,:) = [0 0];
            a(n,:) = [0 0];
            alfa(n,:) = x(n,:) + (delta_t * v(n,:)); %linear uniform motion
            beta(n,:) = v(n,:); %there is not acceleration
        else
            f(n, :) = (k * over(n,1)) * [-1 1];
            a(n,:) = f(n,:) ./ m;
            alfa(n,:)=(2*x(n,:))-x(n-1,:)+((delta_t^2)*a(n,:)); %Verlet
            beta(n,:) = (alfa(n,:) - x(n-1,:))/(2 * delta_t); %Verlet
            v(n,:) = beta(n,:);
        end
    end
end
```

A.4 Script for comparison examples 4.2

A.4.1 Two particles 1D without damping

```
%Set system parameters
k =1e8; %material stiffness [kg/s^2]
R1=0.001; %radius of particle 1 [m]
R2=0.001; %radius of particle 2 [m]
ro =2000; %density of the two particles [kg/m^3]
m = (4/3) * pi * (R1^3) * ro; %mass of the two particles [kg]
%Set initial conditions
x10= 0.048891; x20= 0.0509; %initial position of the two particles [m]
v10 = 1; v20 = -1; %initial velocity of the particle10s [m/s]
x_zero = [x10 x20]'; %initial position vector
v_zero = [v10 v20]'; %initial velocity vector
%Set simulation parameters
tf = 1e-5; %time of simulation [s]
delta_t = 1e-9; %time step [s]
Nt = ceil(tf/delta_t); %number of time increments

%Initialize the loop variables
for n = 1 : Nt+1
    if n == 1
        x(n,:) = y_zero;
        v(n,:) = v_zero;
        over(n,1) = (R1 + R2) - (y(n,2) - x(n,1)); %overlap
        if over(n,1) <= 0
            f(n,:) = [0 0];
            a(n,:) = f(n,:) ./ m;
            alfa(n,:)=x(n,:)+(delta_t * v(n,:));%linear uniform motion
            beta(n,:) = v(n,:); %there is not acceleration
```

```
        else
            f(n, :) = (k * over(n,1)) * [-1 1];
            a(n,:) = f(n,:) ./ m;
            alfa(n,:) = x(n,:) + (delta_t * v(n,:)) + ...
            (0.5 * a(n,:) * (delta_t^2)); %uniformly accelerated motion
            beta(n,:) = v(n,:) + (delta_t * a(n,:));
        end
    else
        x(n,:) = alfa(n-1,:);
        over(n,1) = (R1 + R2) - (x(n,2) - x(n,1)); %overlap
        if over(n,1) <= 0
            v(n,:) = beta(n-1,:);
            f(n,:) = [0 0];
            a(n,:) = [0 0];
            alfa(n,:)=x(n,:)+(delta_t * v(n,:));%linear uniform motion
            beta(n,:) = v(n,:); %there is not acceleration
        else
            f(n, :) = (k * over(n,1)) * [-1 1];
            a(n,:) = f(n,:) ./ m;
            alfa(n,:) = (2 * x(n,:)) - x(n-1,:) + ...
            ((delta_t^2) * a(n,:)); %Verlet
            beta(n,:) = (alfa(n,:) - x(n-1,:))/(2 * delta_t); %Verlet
            v(n,:) = beta(n,:);
        end
    end
end

%Comparison with the code
%Evaluation of the number of files
fnames = dir('c3d*');
numfids = length(fnames); %Number of c3d files
%Definition of a matrix of position of particle 1
for i = 1:numfids
    Matrix1(i,:)= dlmread(fnames(i).name, ' ',[1 0 1 2]); %position 1
end
%Definition of a matrix of position of particle 2
for i = 1:numfids
    Matrix2(i,:)= dlmread(fnames(i).name, ' ',[2 0 2 2]); %position 2
end

%Calculation error
x1_script_err = x1_script(1:2:Nt+1,1);
for i = 1 : numfids
    x1_err(i,:) = abs(x1_script_err(i,1) - x1_code(i,1))./...
    abs(x1_script_err(i,1)); %relative error of particle 1 position
end
x2_script_err = x2_script(1:2:Nt+1,1);
for i = 1 : numfids
    x2_err(i,:) = abs(x2_script_err(i,1) - x2_code(i,1))./...
    abs(x2_script_err(i,1)); %relative error of particle 2 position
end
x1_err_max = max(x1_err);%maximum relative error of particle 1 position
x2_err_max = max(x2_err);%maximum relative error of particle 2 position
```

A.4.2 Two particles 2D without damping

```
%Set system parameters
k =1e8; %material stiffness [kg/s^2]
R1=0.001; %radius of particle 1 [m]
R2=0.001; %radius of particle 2 [m]
ro =2000; %density of the two particles [kg/m^3]
m = (4/3) * pi * (R1^3) * ro; %mass of the two particles [kg]
%Set initial conditions
x10=0.049288;y10=0.049288;x20=0.050712;y20=0.050712;%initial position[m]
vx10 = 1; vy10= 1; vx20 = -1; vy20 = -1; %initial velocity [m/s]
x_zero = [x10 y10 x20 y20]; %initial position vector
v_zero = [vx10 vy10 vx20 vy20]; %initial velocity vector
%Set simulation parameters
tf = 1e-5; %time of simulation [s]
delta_t = 1e-9; %time step [s]
Nt = ceil(tf/delta_t); %number of time increments

%Initialize the loop variables
for n = 1 : Nt+1
    if n == 1
        x_in(n,:) = x_zero; %position
        v(n,:) = v_zero; %velocity
        x_p(n,:) = [(x_in(n,1)^2 + x_in(n,2)^2)^(1/2) ...
        (x_in(n,3)^2 + x_in(n,4)^2)^(1/2)]; %total position
        v_p(n,:) = [((v(n,1)^2 + v(n,2)^2)^(1/2))*sign(v(n,1)) ...
        ((v(n,3)^2 + v(n,4)^2)^(1/2))*sign(v(n,3))]; %total velocity
        over(n,1) = (R1 + R2) - (x_p(n,2) - x_p(n,1)); %overlap
        if over(n,1) <= 0
            f(n,:) = [0 0]; %force
            a_p(n,:) = f(n,:) ./ m; %acceleration
            alfa(n,:)=x_p(n,:)+(delta_t*v_p(n,:));%linear uniform motion
            beta(n,:) = v_p(n,:); %there is not acceleration
        else
            f(n, :) = (k * over(n,1)) * [-1 1];
            a_p(n,:) = f(n,:) ./ m;
            alfa(n,:) = x_p(n,:) + (delta_t * v_p(n,:)) + ...
            (0.5 * a_p(n,:) * (delta_t^2));%uniformly accelerated motion
            beta(n,:)=v_p(n,:)+(delta_t * a_p(n,:));%accelerated motion
        end
    else
        x_p(n,:) = alfa(n-1,:);
        over(n,1) = (R1 + R2) - (x_p(n,2) - x_p(n,1)); %overlap
        if over(n,1) <= 0
            v_p(n,:) = beta(n-1,:);
            f(n,:) = [0 0];
            a_p(n,:) = [0 0];
            alfa(n,:)=x_p(n,:)+(delta_t*v_p(n,:));%linear uniform motion
            beta(n,:) = v_p(n,:); %there is not acceleration
        else
            f(n, :) = (k * over(n,1)) * [-1 1];
            a_p(n,:) = f(n,:) ./ m;
            alfa(n,:) = (2 * x_p(n,:)) - x_p(n-1,:) + ...
            ((delta_t^2) * a_p(n,:)); %Verlet
```

```
            beta(n,:) = (alfa(n,:) - x_p(n-1,:))/(2 * delta_t); %Verlet
            v_p(n,:) = beta(n,:);
        end
    end
end

%Comparison with the code
%Evaluation of the number of files
fnames = dir('c3d*');
numfids = length(fnames); %Number of c3d files
%Definition of a matrix of position of particle 1
for i = 1:numfids
    Matrix1(i,:)= dlmread(fnames(i).name, ' ',[1 0 1 2]); %position 1
end
%Definition of a matrix of position of particle 2
for i = 1:numfids
    Matrix2(i,:)= dlmread(fnames(i).name, ' ',[2 0 2 2]); %position 2
end

%Errors calculation
%x position
x1_script_err = x1_script(1:2:Nt+1,1);
for i = 1 : numfids
    x1_err(i,:) = abs(x1_script_err(i,1) - x1_code(i,1))./...
    abs(x1_script_err(i,1)); %relative error of x1 position
end
x2_script_err = x2_script(1:2:Nt+1,1);
for i = 1 : numfids
    x2_err(i,:) = abs(x2_script_err(i,1) - x2_code(i,1))./...
    abs(x2_script_err(i,1)); %relative error of x2 position
end
x1_err_max = max(x1_err); %maximum relative error of x1 position
x2_err_max = max(x2_err); %maximum relative error of x2 position
%y position
y1_script_err = y1_script(1:2:Nt+1,1);
for i = 1 : numfids
    y1_err(i,:) = abs(y1_script_err(i,1) - y1_code(i,1))./...
    abs(y1_script_err(i,1)); %relative error of y1 position
for i = 1 : numfids
    y2_err(i,:) = abs(y2_script_err(i,1) - y2_code(i,1))./...
    abs(y2_script_err(i,1)); %relative error of y2 position
end
y1_err_max = max(y1_err); %maximum relative error of y1 position
y2_err_max = max(y2_err); %maximum relative error of y2 position
```

A.4.3 Two particles 1D with damping

```
%Set system parameters
k =1e8; %material stiffness [kg/s^2]
gamma = 1; %material viscosity [kg/s]
R1=0.001; %radius of particle 1 [m]
R2=0.001; %radius of particle 2 [m]
ro =2000; %density of the two particles [kg/m^3]
```

```
m = (4/3) * pi * (R1^3) * ro; %mass of the two particles [kg]
%Set initial conditions
x10= 0.048891; x20= 0.0509; %initial position of the two particles [m]
v10 = 1; v20 = -1; %initial velocity of the particle10s [m/s]
x_zero = [x10 x20]'; %initial position vector
v_zero = [v10 v20]'; %initial velocity vector
%Set simulation parameters
tf = 1e-5; %time of simulation [s]
delta_t = 1e-9; %time step [s]
Nt = ceil(tf/delta_t); %number of time increments

%Initialize the loop variables
for n = 1 : Nt+1
    if n == 1
        x(n,:) = y_zero;
        v(n,:) = v_zero;
        over(n,1) = (R1 + R2) - (y(n,2) - x(n,1)); %overlap
        if over(n,1) <= 0
            f(n,:) = [0 0];
            a(n,:) = f(n,:) ./ m;
            alfa(n,:)=x(n,:)+(delta_t * v(n,:));%linear uniform motion
            beta(n,:) = v(n,:); %there is not acceleration
        else
            f(n, :) = (k * over(n,1)) * [-1 1];
            a(n,:) = f(n,:) ./ m;
            alfa(n,:) = x(n,:) + (delta_t * v(n,:)) + ...
            (0.5 * a(n,:) * (delta_t^2)); %uniformly accelerated motion
            beta(n,:) = v(n,:) + (delta_t * a(n,:));
        end
    else
        x(n,:) = alfa(n-1,:);
        over(n,1) = (R1 + R2) - (x(n,2) - x(n,1)); %overlap
        if over(n,1) <= 0
            v(n,:) = beta(n-1,:);
            f(n,:) = [0 0];
            a(n,:) = [0 0];
            alfa(n,:)=x(n,:)+(delta_t * v(n,:));%linear uniform motion
            beta(n,:) = v(n,:); %there is not acceleration
        else
            f(n, :) = (k * over(n,1)) * [-1 1];
            a(n,:) = f(n,:) ./ m;
            alfa(n,:) = (2 * x(n,:)) - x(n-1,:) + ...
            ((delta_t^2) * a(n,:)); %Verlet
            beta(n,:) = (alfa(n,:) - x(n-1,:))/(2 * delta_t); %Verlet
            v(n,:) = beta(n,:);
        end
    end
end

%Comparison with the code
%Evaluation of the number of files
fnames = dir('c3d*');
numfids = length(fnames); %Number of c3d files
%Definition of a matrix of position of particle 1
for i = 1:numfids
    Matrix1(i,:)= dlmread(fnames(i).name, ' ',[1 0 1 2]); %position 1
```

```
end
%Definition of a matrix of position of particle 2
for i = 1:numfids
    Matrix2(i,:)= dlmread(fnames(i).name, ' ',[2 0 2 2]); %position 2
end

%Calculation error
x1_script_err = x1_script(1:2:Nt+1,1);
for i = 1 : numfids
    x1_err(i,:) = abs(x1_script_err(i,1) - x1_code(i,1))./...
    abs(x1_script_err(i,1)); %relative error of particle 1 position
end
x2_script_err = x2_script(1:2:Nt+1,1);
for i = 1 : numfids
    x2_err(i,:) = abs(x2_script_err(i,1) - x2_code(i,1))./...
    abs(x2_script_err(i,1)); %relative error of particle 2 position
end
x1_err_max = max(x1_err);%maximum relative error of particle 1 position
x2_err_max = max(x2_err);%maximum relative error of particle 2 position
```

A.4.4 Two particles 2D with damping

```
%Set system parameters
k =1e8; %material stiffness [kg/s^2]
gamma = 1; %material viscosity [kg/s]
R1=0.001; %radius of particle 1 [m]
R2=0.001; %radius of particle 2 [m]
ro =2000; %density of the two particles [kg/m^3]
m = (4/3) * pi * (R1^3) * ro; %mass of the two particles [kg]
%Set initial conditions
x10=0.049288;y10=0.049288;x20=0.050712;y20=0.050712;%initial position[m]
vx10 = 1; vy10= 1; vx20 = -1; vy20 = -1; %initial velocity [m/s]
x_zero = [x10 y10 x20 y20]; %initial position vector
v_zero = [vx10 vy10 vx20 vy20]; %initial velocity vector
%Set simulation parameters
tf = 1e-5; %time of simulation [s]
delta_t = 1e-9; %time step [s]
Nt = ceil(tf/delta_t); %number of time increments

%Initialize the loop variables
for n = 1 : Nt+1
    if n == 1
        x_in(n,:) = x_zero; %position
        v(n,:) = v_zero; %velocity
        x_p(n,:) = [(x_in(n,1)^2 + x_in(n,2)^2)^(1/2) ...
        (x_in(n,3)^2 + x_in(n,4)^2)^(1/2)]; %total position
        v_p(n,:) = [((v(n,1)^2 + v(n,2)^2)^(1/2))*sign(v(n,1)) ...
        ((v(n,3)^2 + v(n,4)^2)^(1/2))*sign(v(n,3))]; %total velocity
        over(n,1) = (R1 + R2) - (x_p(n,2) - x_p(n,1)); %overlap
        if over(n,1) <= 0
            f(n,:) = [0 0]; %force
            a_p(n,:) = f(n,:) ./ m; %acceleration
            alfa(n,:)=x_p(n,:)+(delta_t*v_p(n,:));%linear uniform motion
```

```
            beta(n,:) = v_p(n,:); %there is not acceleration
        else
            f(n, :) = (k * over(n,1)) * [-1 1];
            a_p(n,:) = f(n,:) ./ m;
            alfa(n,:) = x_p(n,:) + (delta_t * v_p(n,:)) + ...
            (0.5 * a_p(n,:) * (delta_t^2));%uniformly accelerated motion
            beta(n,:)=v_p(n,:)+(delta_t * a_p(n,:));%accelerated motion
        end
    else
        x_p(n,:) = alfa(n-1,:);
        over(n,1) = (R1 + R2) - (x_p(n,2) - x_p(n,1)); %overlap
        if over(n,1) <= 0
            v_p(n,:) = beta(n-1,:);
            f(n,:) = [0 0];
            a_p(n,:) = [0 0];
            alfa(n,:)=x_p(n,:)+(delta_t*v_p(n,:));%linear uniform motion
            beta(n,:) = v_p(n,:); %there is not acceleration
        else
            f(n, :) = (k * over(n,1)) * [-1 1];
            a_p(n,:) = f(n,:) ./ m;
            alfa(n,:) = (2 * x_p(n,:)) - x_p(n-1,:) + ...
            ((delta_t^2) * a_p(n,:)); %Verlet
            beta(n,:) = (alfa(n,:) - x_p(n-1,:))/(2 * delta_t); %Verlet
            v_p(n,:) = beta(n,:);
        end
    end
end

%Comparison with the code
%Evaluation of the number of files
fnames = dir('c3d*');
numfids = length(fnames); %Number of c3d files
%Definition of a matrix of position of particle 1
for i = 1:numfids
    Matrix1(i,:)= dlmread(fnames(i).name, ' ',[1 0 1 2]); %position 1
end
%Definition of a matrix of position of particle 2
for i = 1:numfids
    Matrix2(i,:)= dlmread(fnames(i).name, ' ',[2 0 2 2]); %position 2
end

%Errors calculation
%x position
x1_script_err = x1_script(1:2:Nt+1,1);
for i = 1 : numfids
    x1_err(i,:) = abs(x1_script_err(i,1) - x1_code(i,1))./...
    abs(x1_script_err(i,1)); %relative error of x1 position
end
x2_script_err = x2_script(1:2:Nt+1,1);
for i = 1 : numfids
    x2_err(i,:) = abs(x2_script_err(i,1) - x2_code(i,1))./...
    abs(x2_script_err(i,1)); %relative error of x2 position
end
x1_err_max = max(x1_err); %maximum relative error of x1 position
x2_err_max = max(x2_err); %maximum relative error of x2 position
%y position
```

```
y1_script_err = y1_script(1:2:Nt+1,1);
for i = 1 : numfids
    y1_err(i,:) = abs(y1_script_err(i,1) - y1_code(i,1))./...
    abs(y1_script_err(i,1)); %relative error of y1 position
for i = 1 : numfids
    y2_err(i,:) = abs(y2_script_err(i,1) - y2_code(i,1))./...
    abs(y2_script_err(i,1)); %relative error of y2 position
end
y1_err_max = max(y1_err); %maximum relative error of y1 position
y2_err_max = max(y2_err); %maximum relative error of y2 position
```

A.5 Script for preparation procedure 5.3.1

```
%Input values
ni = 0.3; %volume fraction of the system
r_m = 1; %mean radius
W = 3; %polydispersity
Np = 21^3; %number of particles
t = 0; %initial time
v_min = -1e-4; %lowest velocity value
v_max = 1e-4; %hightest velocity value
xi=0; %last number of particle line

%Definition of maximum and minimum radius
r_min = (2*r_m)/(W+1); %definition of minimum radius
r_max = W * r_min; %definition of maximum radius
%Definition of the randomic radius of each particle
r(:,1) = r_min + ((r_max - r_min) * rand(Np,1));
%Definition of the total volume of the particles
for n = 1:Np
    Vp(n,1) = (4/3)*pi*(r(n,1))^3; %volume of each particle
end
Vtot = sum(Vp); %total volume of the particles
%Evaluation of dimension of the box
L = (Vtot/ni)^(1/3); %dimension of the box

%Give random position to particles
for i= 1:Np
    x(:,1) = L * rand(Np,1); %definition of x position
    y(:,1) = L * rand(Np,1); %definition of y position
    z(:,1) = L * rand(Np,1); %definition of z position
end

%Give random velocity to particles
for i= 1:Np
    vx(:,1) = v_min + ((v_max - v_min) * rand(Np,1)); %x velocity
    vy(:,1) = v_min + ((v_max - v_min) * rand(Np,1)); %y velocity
    vz(:,1) = v_min + ((v_max - v_min) * rand(Np,1)); %z velocity
end
```