Master's thesis

NTNU
Norwegian University of Science and Technology
Faculty of Natural Sciences
Department of Physics

Aleksander Gjersvoll

# Simulation methods for granular gas dynamics in periodic three-dimensional systems

Master's thesis in Applied Physics and Mathematics
Supervisor: Tor Nordam
June 2020

NTNU
Norwegian University of
Science and Technology

Aleksander Gjersvoll

# Simulation methods for granular gas dynamics in periodic three-dimensional systems

Master's thesis in Applied Physics and Mathematics
Supervisor: Tor Nordam
June 2020

Norwegian University of Science and Technology
Faculty of Natural Sciences
Department of Physics

NTNU
Kunnskap for en bedre verden

# Preface

This thesis is submitted as the work done for the course TFY4900 - Physics, Master's Thesis (30 ECTS) at NTNU. The course is my final course of the five year master's degree program Applied Physics and Mathematics at NTNU, where I have chosen Applied Physics as my field of study. Upon completion of this course I will achieve a M.Sc. in Physics and Mathematics. The work done for this master's thesis has been carried out through the spring of 2020 in Trondheim. The code written and used for the simulations in this report can be found on github[1], and guidance can be be provided upon request.

The work presented here is a continuation of the work done during the autumn of 2019 for the course TFY4510 - Physics, Specialization Project (15 ECTS) [see 1]. The work done throughout these two courses has been under the guidance of my supervisor, Associate Professor Tor Nordam. His expertise in the field of computational physics, high performance computing and stochastic differential equations have contributed greatly to the results in this thesis. I would like to express my gratitude for all his contributions during the year we have worked together. Our discussions have inspired many new ideas, given me motivation to continue working and solved many of the problems I managed to stumble across this past year. Our weekly meetings have been full of humour and I always felt inspired to work on new ideas afterwards.

This project originates from the exam Tor Nordam created for the course TFY4235 - Computational Physics in 2016. We then created a project based on the foundation from the exam, by conducting numerical simulations of many-particle systems of particles colliding in a box. For the specialization project we focused mainly on results for a molecular gas, while we intend here to study the more complex case of granular gas dynamics.

I would also like to thank my family and friends for the support I have experienced through my time at NTNU. Especially, I would like to express my gratitude for my friends and fellow students in the class of 2015 for the endurance shown during the many hours we have spent learning interesting topics in the area of physics and mathematics and for all the good memories we have made together. As a result of the currently ongoing COVID-19 pandemic I have since the middle of March been working from home. I would also thus like to thank Ida Marie Falnes, who has been in the same situation as me while writing a master's thesis in applied mathematics and I have shared a home office with. Her company and cheerfulness has made working from home possible.

Trondheim, June 2020
Aleksander Gjersvoll

[1] `https://github.com/alekgjer/master_thesis`

# Abstract

In this thesis we study granular gas dynamics, where it has been assumed that the dynamics of a granular gas is determined by instantaneous pairwise inelastic particle collisions, and the collisions are given by a constant coefficient of restitution. A granular gas thus differ from a molecular gas, where the particle collisions are elastic. In order to study granular gas dynamics two different simulations methods have been implemented to conduct numerical studies of a system of particles colliding in a three-dimensional cubic box. To conduct molecular dynamics simulations, with either reflecting or periodic boundary conditions, an event driven simulation has been implemented. Langevin dynamics have also been used by solving different Langevin equations modelling the dynamics of different types of particles.

Molecular dynamics is a numerical method used to study the movement and the dynamics of particles. Instead of solving Newton's equation of motion for each particle, we have in this project utilized the nature of a granular gas to motivate the choice of reducing the molecular dynamics simulation to an event driven simulation, for which one only has to deal with pairwise particle collisions. The event driven simulation has been implemented with a priority queue, an efficient data structure used to easily identify the next collision and store all future collisions.

Langevin dynamics have been used to model the dynamics of different particles as a stochastic differential equation. Numerically, we can solve such equations by applying a time discretization method. The Euler-Maruyama method has been implemented with success to solve the underdamped Langevin equation and underdamped scaled Brownian motion modelling the particles in a molecular and a granular gas respectively.

In order to verify the implemented simulation methods we have computed numerical results for both a molecular and a granular gas, and compared the results with theoretical predictions from kinetic theory and Brownian motion and with published results. The study of Brownian motion consists of computing the mean squared displacement, and comparing with theory given by the velocity autocorrelation function of different systems. The event driven simulation has given results in the areas of system statistics, speed distributions, diffusion and Brownian motion, for which the results have shown a satisfactory agreement with theoretical predictions and published results, verifying the implemented simulation. The numerical solution to the Langevin equations exhibits the same type of behaviour as the event driven simulation, verifying the use of stochastic differential equations to model the dynamics of a granular system.

# Samandrag

I denne masteroppgåva studerar me granulær gassdynamikk, der me har antatt at dynamikken til ein granulær gass er bestemt av momentane parvise uelastiske partikkelkollisjonar, og kollisjonane er gitt av ein konstant restitusjonskoeffisient. Ein granulær gass skiljer seg derfor frå ein molekylær gass, der partikkelkollisjonane er elastiske. For å studere granulær gassdynamikk har to simuleringsmetodar blitt implementert for å gjennomføre numeriske studiar av partiklar som kolliderar i ein tredimensjonal kubisk boks. For å gjennomføre simuleringar av molekylærdynamikk, med enten reflekterande eller periodiske grensar, har ein hendingbasert simulering blitt implementert. Langevindynamikk har også blitt brukt for å løyse ulike Langevinlikningar som modellerar dynamikken til ulike typer partiklar.

Molekylærdynamikk er ein numerisk metode brukt for å studere rørslene og dynamikken til partiklar. I stedet for å løyse Newton sine rørslelikningar for kvar partikkel, har me i dette prosjektet utnytta eigenskapane til ein granulær gass for å motivere ein reduksjon frå molekylærdynamikk til ein hendingbasert simulering, der ein berre må håndtere parvise partikkelkollisjonar. Den hendingsbaserte simuleringa har blitt implementert med ei prioritetskø, som er ein effektiv datastruktur som kan brukast til å identifisere den neste kollisjonen og lagre alle framtidige kollisjonar.

Langevindynamikk har blitt brukt for å modellere dynamikken til ulike partikler som ei stokastisk differensiallikning. Numerisk kan me løyse sånne likningar med å bruke ein tidsdiskretiseringsmetode. Euler-Maruyama metoden har blitt implementert med suksess for å løyse den underdempa Langevinlikninga og den underdempa skalerte Brownske rørslen som modellerar partiklane i ein henhaldsvis molekylær og granulær gass.

For å verifisere dei implementerte simuleringsmetodane har me rekna ut numeriske resultat for både ein molekylær og ein granulær gass, og samanlikna resultata med teoretiske prediksjonar frå kinetisk teori og Brownske rørsler og med publiserte resultat. Vår studie av Brownske rørsler har bestått av å rekne ut den gjennomsnittlege kvadratiske distansen frå startposisjon, og samanlikna med teori gitt av funksjonen for hastigheiten sin autokorrelasjon for ulike system. Den hendigbaserte simuleringsverktøyet har gitt resultat innenfor systemstatistikk, fartsfordelingar, diffusjon og Brownske rørsler, der resultata har vist eit tilfredsstillande samsvar med teori og publiserte resultat, og dermed verifisert implementasjonen. Resultata frå å løyse Langevinlikningar har vist den same åtferda som den hendingbaserte simuleringa, noko som har verifisert bruken av stokastiske differensiallikningar for å modellere dynamikken til eit granulært system.

# Contents

# List of Figures

# List of Tables

# List of Abbreviations

Here we present a list of abbreviations used in the report. In the list we give the abbreviation, the full form, and the page number where the abbreviation is first introduced.

**HPC** High Performance Computing. 55

**MSD** Mean Squared Displacement. 22

**ODE** Ordinary Differential Equation. 31

**PBC** Periodic Boundary Conditions. 40

**RBC** Reflecting Boundary Conditions. 40

**SDE** Stochastic Differential Equation. 23

**UDSBM** Underdamped Scaled Brownian Motion. 32

# Chapter 1

# Introduction

The field of granular materials has seen an increase in interest during the last thirty years. This is mainly due to the increase in available computational power, making it possible to do large scale simulations of granular systems. Earlier, granular materials has been a field where there have been contributions from major names in physics, e.g. Coulomb, Faraday and Rayleigh [see 2, preface]. There exists, however no complete acknowledged theory of granular matter to this date. Granular materials are of interest due to the variety of systems that can be modelled as granular systems, and some examples of such materials are sand, grains, snow and dust. Being ubiquitous in nature, granular materials are important in different industrial applications found in e.g. agriculture, mining, pharmaceutical industry and many more [see 3, pp. 1–2]. One of the known early attempts of gathering knowledge from a stray of different fields in order to study the transport of granular material by a fluid was done by Bagnold looking at the physics of blown sand and desert dunes [see 4].

One key property of a granular material is the ability to behave differently under different circumstances. Take sand for example. Sand in an hourglass flows like a liquid, but standing on a beach you will not fall through the sand. Instead you will likely, after sinking a few centimeters, feel like you are standing on solid structure. Behaviour like this is hard to predict based on the normal convention of looking at matter as either liquid, solid, gas or plasma that can undergo phase transitions. One can thus argue that granular materials should perhaps be considered as a new state of matter [see 3, pp. 1–2]. A motivation for study of granular materials is to understand and predict behaviour of these materials in a satisfactory manner, which would present valid information and insight for industrial application [see 5, pp. 1–2].

There are several difficulties in addressing such an immense and diverse field as granular materials with no prior knowledge in the subject for a master's thesis. First of all it is an impossible task to provide a detailed introduction to granular matter. We will instead use this introduction to focus on the topic we will study, while presenting some sources giving a detailed treatment of granular matter. For a tentative view of granular matter see [6]. For an effort to collect ideas and studies performed in different fields of science, and introducing different concepts to discuss the collective behaviour of granular materials see [7]. We will present the concepts and theory needed to fully understand the results of this thesis, in addition to a detailed introduction to two different ways to study a granular system numerically. The main characteristics of granular materials are that their behaviour are determined by their kinetic and or gravitational energy, and the grains of the material experience dissipative interactions. The latter is what separates the behaviour of granular materials from usual gases and liquids, laying the foundation for some spectacular effects [see 3, preface].

This master's thesis aims to study what is known as granular gases. We will use the definition where one characterizes a granular gas as a many-particle system, where the particle collisions occur dissipatively and the duration of contact is much smaller that the mean flight time. The latter

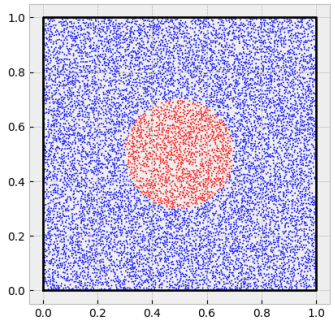
indicates that the particles spend the majority of their time moving with constant velocity between successive collisions. Due to the dissipation of energy a granular gas behaves differently than a molecular gas, where collisions are elastic, leading to non-trivial effects such as cluster formation, anomalous diffusion and many more [2, preface, 3, pp. 4–5]. The energy dissipation leads to a decrease in temperature, and the gas cools in a non-uniform fashion as a result of the collisions [see 8]. The instability of the clustering phase of dissipative gases is most likely one of the reasons why the field of granular materials has seen an increase in interest with the increasing possibilities of computer simulations [see 9].

We will restrict the study of a granular gas to the case where we only consider the dissipative collisions between the particles to determine the dynamics of the system, with no external forces. We will then study a system of hard spheres colliding with a coefficient of restitution, where the coefficient of restitution determine the degree of inelasticity in the system, in order to conduct studies of systems with a varying degree of energy dissipation. In order to study such a system, we will need to derive a collision rule, relating the post-collision velocities to the pre-collision velocities and other particle parameters. Such simple systems have been shown to provide valid results in the area of kinetic theory where we focus on the dynamics of the particles in the system instead of modelling the dynamics of the entire system [see 3, pp. 1–5]. In the absence of external forces, or systems where such forces are negligible compared to the collisions between particles, it is possible to derive a number of theoretical predictions based on kinetic theory, both for molecular and granular gases. The treatment of molecular gases arises from the fields of thermal and statistical physics [see 10, pp. 117–154]. The generalization of such principles used for a system of particles colliding inelastically, namely a granular gas, has provided some interesting results [see 11].

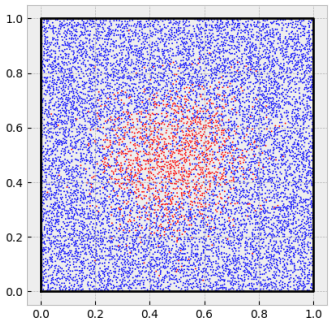
In order to study many-particle systems, such as gases, we are going to need some verification tools, in order to verify if we manage to model the correct dynamics for different systems. We can thus model a molecular gas by letting the collisions be elastic and thus conserving energy. Similarly we can model a granular gas by letting the collisions be inelastic, where energy is dissipated for each collision. Hence we can use these two different types of gases to verify the implemented simulation algorithm. Even though granular gases are of more interest due to the complexity of the system, we started with the study of a molecular gas for the implementation. Even though we expect that the same simulation will capture the correct dynamics for both molecular and granular gases, as elastic collisions are a special case of inelastic collisions, it is always convenient and natural to start with simpler systems before moving on to more complex ones.

Figure 1.1 illustrates the different dynamic behaviour of a molecular gas and a granular gas in a two-dimensional system with reflecting boundaries, from the same initial system given in Figure 1.1a. The two different systems, at the same later time are given in Figure 1.1b and Figure 1.1c. In the figures two colors have been used to indicate two different subsets of particles. By letting the particles behave as in a molecular gas, the two subsets will start to mix in the expected manner. However as one can see, the particles in a granular gas show some of the previously mentioned non-trivial effects, as clusters have started forming in some regions of the box.

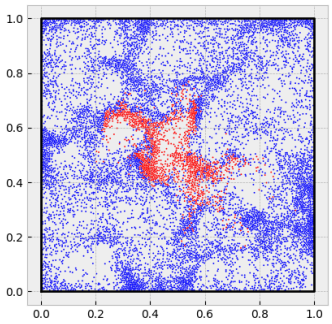
There exist several different simulation methods used to study granular systems, e.g. direct simulation Monte Carlo, Langevin dynamics, and molecular dynamics simulations [see 5]. The latter has played the most important role in studying granular systems [see 7, 8, 12, 13, 14, 15]. The general idea of molecular dynamics is to numerically solve equations of motion, e.g. Newton's, for all particles simultaneously [see 5, pp. 8–9]. Historically molecular dynamics has been used for a diverse variety of systems, and was highly relevant even before we used computers to perform it efficiently. Just imagine the simple case of gravitational forces acing upon the planets in our solar system. Even for a simple case as looking at the gravitational effects on the Earth from the Sun and one other planet we encounter a problem for which there are no explicit solution. The problem is more commonly generalized and known as the $N$-body problem. For such problems we see the simplicity and beauty of molecular dynamics, where we can easily add numerous forces in order to compute the trajectories for our planets [see 16, pp. 1–4]. For an introduction to how we can




**(a)** $t_0 = 0$




**(b)** $t_1 > t_0$




**(c)** $t_1 > t_0$

**Figure 1.1:** Illustration of the different futures for a molecular gas and a granular gas from the same initial system with reflecting boundaries. Figure 1.1a shows the initial system where the particles are uniformly distributed in the box. The particles inside a given radius from the center of the box have been highlighted in order to better see how the particles move in time. Figure 1.1b shows the system at some later time $t_1$ by letting the particles collide elastically, which is the case for molecular gases. When energy is conserved, the red particles will reach the wall given enough time. Figure 1.1c shows the system at the same time $t_1$, but where the particle collisions have dissipated energy, which is the case for granular gases. The loss of energy leads to a decrease in temperature. The gas thus cools in the denser regions, where the collision frequency has been high, and clusters are formed. We also see that for a molecular gas the two different colors will tend to mix, while for a granular gas the colors are still separated in different areas.

use molecular dynamics to study the dynamics of granular materials see [16, pp. 436–445] and [5, pp. 13–131]. We will in this thesis use two of the mentioned simulation methods, namely Langevin dynamics and molecular dynamics simulations, to perform simulations of a molecular and a granular gas.

To better illustrate the needed requirements to solve Newton's equations for a granular system, let us look at what we would need to know about the system. For particle $i$ in a system of $N$ particles, Newton's equation has the familiar form

$$m_i \frac{d^2}{dt^2} \boldsymbol{x}_i(t) = \boldsymbol{F}_i(\boldsymbol{x}_1(t), \boldsymbol{v}_1(t), \ldots, \boldsymbol{x}_N(t), \boldsymbol{v}_N(t)), \tag{1.1}$$

where $m_i$, $\boldsymbol{x}_i(t)$ and $\boldsymbol{v}_i(t)$ is the mass, position and velocity of particle $i$ respectively, $t$ is time and $\boldsymbol{F}_i$ is the force acting upon particle $i$, which in general can be a function of all particle positions and velocities. Here we can see one of the reasons why computational power sparked a newfound interest in granular systems. Since one can not analytically solve a total of $N$ equations on the form of (1.1), it has been difficult to achieve reliable predictions of granular systems without making assumptions on how one can describe the system on a larger scale. Since granular systems do not always act as expected, it has been a challenging task to gain insight in the long term behaviour of such systems. On the other hand, solving Newton's equations numerically is possible to some extent by making assumptions on how the particles affect each other. However, often it is a challenging task to express the force term as a function of the known variables in order to explain the behaviour of a granular system. One option is to use experimental data to investigate the interactions between the particles. Even by achieving approximations of the interactions in the system, it could still be a computationally heavy task to solve (1.1) for each particle [see 5, pp. 135–136]. As we are interested in systems where the only interactions between the particles are given as collisions, we could use a force equal to zero when there are no collisions, and a non-zero force to represent different collisions when the particles come in contact.

A more efficient approach can however be developed by exploiting the nature of a granular gas. The particles in a granular gas spend most of their time moving with constant velocity between successive collisions, and they are rarely in contact with multiple particles. Thus one can argue that the particles in a granular gas only interact though instantaneous pairwise collisions, which can be treated separately since they occur at different times. With this assumption we can reduce the computational task of solving Newton's equations to solving equations for collisions between two particles. This leads to a force-free description of the interactions between the particles in the granular gas, where we do not use nor need an analytical expression for the force acting upon the particles. We can thus create an event driven simulation for the collisions in the system, which is an effective way of studying how a force-free granular gas evolves in time for some densities [see 5, pp. 135–136]. The force-free modelling of a granular gas is a common approach used to study the dynamics of a granular gas, e.g. see [15], reducing the molecular dynamics simulation to a series of pairwise particle collisions, which can use efficient data structures in order to be less computationally expensive. The event driven simulation will follow the idea that initially the particles move with constant velocity, until a particle collides with another. Then the velocity of the colliding particles are updated from the collision rule, before all particles again move with constant velocity until the next collision. The procedure can thus be repeated to study the long term behaviour of the system [see 11, pp. 5–8].

Some experimental data of granular gases show a good agreement with theory and numerical studies, see e.g. [14] and [17], but generally it is a challenging task to retrieve the data of interest without affecting the behaviour of the system [see 5, pp. 2–8]. The sparked interest in the field of granular materials has given some interesting results, but from the difficulty of the systems, one can imagine that there is still more to come in the next years. Non-trivial effects should not be a surprise from the a priori knowledge that a granular gas is a non-equilibrium system due the dissipation of energy. However, by giving the system energy by implementing e.g. vibrating walls, heated walls or

random heating one can try to create equilibrium systems. For a study of vibrated granular gases see [12]. One should note that the area of driven granular materials, e.g. by gravity, vibration or rotation, is used to model and study some of the most applicable uses of granular materials [see 7, pp. 643–646]. Even though a driven granular system produces results very different from a force-free system, it is natural to use this thesis as an introduction to some of the simpler aspects of a granular system, intended to inspire others to work within the interesting field of granular materials. For a list of some of the early attempts of experimental investigation of the kinetic theory of granular gases see [11, pp. 10–12].

The use of event driven simulations to study granular systems does however have its limitations. Some of the more obvious limitations arise from only allowing pairwise interactions, not justifiable for high densities and for systems where multi-particle interactions dominate. On the other hand, event driven simulations are common in a variety of different applications. A recent example is the use of a particle collision simulation to illustrate the effects of social distancing on the spread of disease for the currently ongiong COVID-19 pandemic as seen in `https://www.washingtonpost.com/graphics/2020/world/corona-simulator/` and `https://www.nrk.no/norge/xl/slik-virker-den-nasjonale-koronadugnaden-1.14947139`. For such a case as looking at how disease spreads through a population, it is possible to provide a qualitative illustration of how different countermeasures affects the number of people infected with the disease. Event driven simulations are also commonly used in the fields of Queueing theory and stochastic processes. More generally one can claim that any situation of discrete events[1] occurring in a sequence is a potential area for which an event driven simulation can be used to gain more insight. With more insight it is possible to make better decisions, or to make changes in order to be better prepared for difficult circumstances, such as pandemics. Whereas in this project we will only use collisions as events, the general idea of an event driven simulation, defined as a simulation where time is incremented between successive events, works for any type of event. It is also possible to study systems consisting of multiple different events, e.g. the planning of work schedules for a hospital, the number of different people to hire for a business and many more.

The main objectives are to implement an event driven simulation code for granular gases, and to reproduce previously published results from the literature using the programming language Python[2]. Additionally, we will look at the effects of different boundary conditions, and compare the results of the event driven simulations to the numerical solutions of stochastic differential equations known as Langevin equations. These tools will be verified by looking at and comparing the numerical results with theoretical predictions for molecular and granular gases. The main workload will thus be to write an efficient code, understanding and creating a method to study granular gases, which is often not discussed in great detail in relevant papers. The verification process will consist of a few elementary tests and the reproduction of some known results from statistical physics and kinetic theory. Finally, the simulation code has been applied to the area of Brownian motion, where we compute the so-called mean squared displacement that can be used to describe diffusion processes. The work done for this thesis is thus intended to give a detailed introduction to two different simulation methods possible to use in the study of granular gas dynamics, implemented from scratch.

As mentioned in the preface the work done for this thesis is a continuation of the work done for the specialization project [1]. Achieving reliable results for a molecular gas is not a surprise, as we have already studied some of the properties of a molecular gas, including speed distribution in two dimensions and some aspects of Brownian motion for a molecular gas [1]. Motivated by the lack of agreement between some of the numerical and theoretical predictions in [1] we have made some changes to the implementation. I.e. we have changed the study of particles in a two-dimensional

[1] See `https://en.wikipedia.org/wiki/Discrete-event_simulation` for a variety of applications for discrete-event simulation.

[2] See `https://www.datacamp.com/community/blog/python-scientific-computing-case` for an introduction to why Python is an effective tool used in scientific computing.

system with reflecting boundaries to a study of particles in a three-dimensional system with periodic boundaries. A discussion comparing the results for this thesis with the results in [1] is provided in chapter 4.

This thesis is structured in the following manner. In chapter 2 the theory of collisions and the theory behind the topics we will study in this project are presented. Then, in chapter 3, the numerical modelling used in order to achieve an efficient event driven simulation of a many-particle system is presented in detail. We will also present how we have solved the Langevin equations numerically. Further, in chapter 4, simulation results for different systems are presented, and compared to theory and previously published results. In addition to using the results to verify the implemented approaches, we also provide a discussion concerning the possible errors and the agreement between the numerical results and theory. Chapter 5 provides some thoughts about possible other interesting topics in the area of granular gas dynamics, possible improvements for the implemented event driven simulation, and other possible uses for the implemented event driven simulation. Finally, chapter 6 concludes the work done throughout this master's thesis. In addition, there are also some detailed derivations provided in the appendices to complement the report. Appendix D provides the code used for the most time consuming part of the event driven simulation, which is the computation of which particles a particle will collide with and when for its current trajectory, and illustrates some of the possible optimizations one can implement in Python. At the end of the report there is included a list of symbols which can used to look up different symbols used in the report and on what page you can find their definition.

# Chapter 2

# Theory

In the following chapter, several important principles and equations are presented. These derivations are key in understanding some of the results from the simulations which will be presented later in chapter 4. The theory presented here is general, but some of the topics are related to the numerical modelling which is discussed in greater detail in chapter 3.

## 2.1 Collisions

There are two different main types of collisions, elastic and inelastic. The difference between these collisions is that during elastic collisions kinetic energy is conserved, while kinetic energy is lost during an inelastic collision. How the velocity of a particle changes due to a collision with another particle is based on two principles. The first principle is conservation of linear momentum, which is a vector quantity. The second principle is related to the energy of the system, which is a scalar quantity. Note that we ignore any effects of rotation. For an inelastic collision one can introduce a coefficient of restitution, $\xi$, which is a measure of how much energy is lost during a collision. The coefficient of restitution is thus a measure of the degree of inelasticity in the system. There exist several expressions for the coefficient of restitution, e.g. for a one-dimensional collision

$$\xi = -\frac{v_j' - v_i'}{v_j - v_i}, \tag{2.1}$$

where $v_i'$ and $v_j'$ are the velocities after the collision, while $v_i$ and $v_j$ are the velocities before the collision for particle $i$ and particle $j$. See Appendix A for a derivation of the expression in (2.1) for an elastic collision. A direct interpretation of (2.1) is a ratio of relative velocities after and before a collision. One should note that the velocity is a function of time, $v(t)$, but we will drop the time notation for simplicity and only look at the velocity after and before a collision, distinguishable from the use of $'$. The numerical value of $\xi$ leads to the following different types of collision

- $\xi = 1$, elastic collision,
- $0 < \xi < 1$, inelastic collision,
- $\xi = 0$, perfectly inelastic collision.

There exist several ways of deriving the equations stating how the velocity of a particle changes as a result of a collision, often referred to as a collision rule. There are some assumptions needed in order to make the following derivations valid. When deriving the equations stating the velocity of the particles after the collision, we assume that the particles collide as two hard spheres, which are not deformed in the process. In addition, we assume that the particles can not rotate (or equivalently

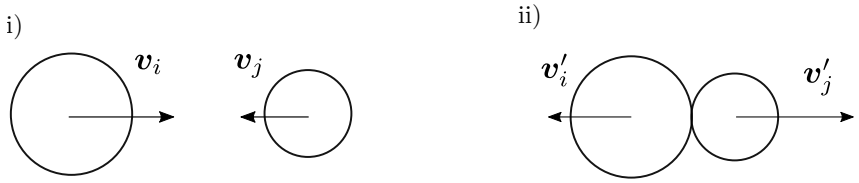



**Figure 2.1:** Illustration of a one-dimensional system containing particle $i$ and particle $j$. The figure shows the two particles before colliding in i), and right after colliding in ii). The velocities before the collision are given as $\boldsymbol{v}_i$ and $\boldsymbol{v}_j$. After the collision the velocities are given as $\boldsymbol{v}_i'$ and $\boldsymbol{v}_j'$.

that they are frictionless), and the collision is instantaneous. These assumptions make the collision into an event where the change in energy is a result of a new speed in a new direction. Since the equations for an elastic collision can be achieved by setting $\xi = 1$, only the inelastic case will be derived. Even though the derivation in one dimension is trivial, the derivation is given because we will need it for the three-dimensional case. In the derivation the notation will be similar as in (2.1), where $'$ is used to indicate velocities after the collision.

Intuitively the change in velocity for particle $i$ colliding with particle $j$ will depend on the difference in position and velocity. Thus there exist some helpful quantities, here given for the case of a three-dimensional system

$$\Delta \boldsymbol{x}_{ij} = \boldsymbol{x}_j - \boldsymbol{x}_i = [x_j - x_i, y_j - y_i, z_j - z_i], \tag{2.2a}$$

$$\Delta \boldsymbol{v}_{ij} = \boldsymbol{v}_j - \boldsymbol{v}_i = [v_{xj} - v_{xi}, v_{yj} - v_{yi}, v_{zj} - v_{zi}], \tag{2.2b}$$

$$R_{ij}^2 = |\Delta \boldsymbol{x}_{ij}|^2 = (x_j - x_i)^2 + (y_j - y_i)^2 + (z_j - z_i)^2, \tag{2.2c}$$

where $\Delta \boldsymbol{x}_{ij}$ is a vector from the center of particle $i$ to the center of particle $j$, $\Delta \boldsymbol{v}_{ij}$ is the vector stating the difference between the velocity of particle $j$ and particle $i$, also known as the relative velocity, and $R_{ij}$ is the distance between the centers of the particles.

### 2.1.1 Inelastic collision in one dimension

Consider the system given in Figure 2.1, with two particles colliding in one dimension. For such a system there are two unknowns after the collision, $v_i'$ and $v_j'$. Conservation of momentum provides the following relation

$$m_i v_i + m_j v_j = m_i v_i' + m_j v_i'. \tag{2.3}$$

From (2.1) one can get the following expression for $v_j'$

$$v_j' = (v_i - v_j)\xi + v_i'. \tag{2.4}$$

By inserting (2.4) into (2.3) and rearranging terms we get the following velocity for particle $i$ after the collision

$$v_i' = \frac{m_j v_j (1 + \xi) + v_i (m_i - m_j \xi)}{m_i + m_j}. \tag{2.5}$$

In order to obtain the velocity of particle $j$ after the collision we insert (2.5) into (2.4), leading to the following expression

$$v_j' = \frac{m_i v_i (1 + \xi) + v_j (m_j - m_i \xi)}{m_i + m_j}. \tag{2.6}$$

A validation of the resulting velocities derived in (2.5) and (2.6) can be achieved by inserting the expressions into the right hand side of (2.1).

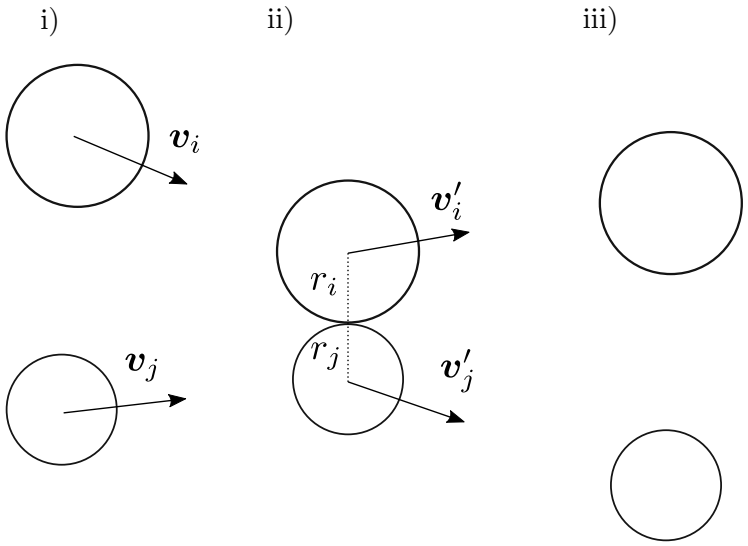



**Figure 2.2:** Illustration of a three-dimensional system containing particle $i$ and particle $j$. The figure shows the two particles before colliding in i), right after colliding in ii) and some time after colliding in iii). The figure uses the same notation as Figure 2.1. We see that during the collision $R_{ij} = r_i + r_j$, where $r_{(i/j)}$ is the radius of particle $(i/j)$.

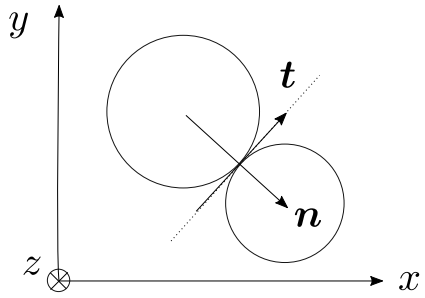



**Figure 2.3:** The coordinate system given by $\boldsymbol{n}$ and $\boldsymbol{t}$ is used in the derivation of the velocities after a collision for two particles in three dimensions. The figure illustrates the collision moment where the normal vector, $\boldsymbol{n}$, is the unit vector pointing along $\Delta\boldsymbol{x}_{ij}$, and $\boldsymbol{t} \perp \boldsymbol{n}$.

### 2.1.2 Inelastic collision in three dimensions

Consider the system shown in Figure 2.2, with two particles colliding in three dimensions. For such a system there are six unknowns, $\boldsymbol{v}_i' = [v_{xi}', v_{yi}', v_{zi}']$ and $\boldsymbol{v}_j' = [v_{xj}', v_{yj}', v_{zj}']$. The conservation of momentum gives one equation along each axis. There is also one equation regarding the energy, which in total gives four equations. However, four equations cannot uniquely determine six unknowns. We can solve this problem by using a new coordinate system. The new system needs to have a property reducing the number of unknowns. This can be achieved by creating a coordinate system where the collision occurs only along one of the axes. An example of a possible new system is shown in Figure 2.3. The axes is given by the normalized directional vectors

$$\boldsymbol{n} = \frac{\Delta \boldsymbol{x}_{ij}}{R_{ij}}, \tag{2.7a}$$

$$\boldsymbol{t} = [-n_y, n_x, 0], \tag{2.7b}$$

where $\Delta \boldsymbol{x}_{ij}$ and $R_{ij}$ are the quantities presented in (2.2) and $\boldsymbol{t} \perp \boldsymbol{n}$. In three dimensions there exist an infinite number of different vectors perpendicular to $\boldsymbol{n}$. As we will see shortly it does not actually matter which one we choose. Note that in the instant of the collision, we have $|\Delta \boldsymbol{x}_{ij}| = R_{ij} = r_i + r_j$, where $r_i$ and $r_j$ is the radius of particle $i$ and particle $j$. In order to derive the solution, we first have to decompose the velocity vectors. The velocity vectors before the collision in the new coordinate system are given as

$$\boldsymbol{v}_i = v_{ni}\boldsymbol{n} + v_{ti}\boldsymbol{t}, \tag{2.8a}$$

$$\boldsymbol{v}_j = v_{nj}\boldsymbol{n} + v_{tj}\boldsymbol{t}, \tag{2.8b}$$

where $v_{n(i/j)} = \boldsymbol{v}_{i/j} \cdot \boldsymbol{n}$ and $v_{t(i/j)} = \boldsymbol{v}_{i/j} \cdot \boldsymbol{t}$. The velocity vectors after the collision are given as

$$\boldsymbol{v}_i' = v_{ni}'\boldsymbol{n} + v_{ti}'\boldsymbol{t}, \tag{2.9a}$$

$$\boldsymbol{v}_i' = v_{nj}'\boldsymbol{n} + v_{tj}'\boldsymbol{t}, \tag{2.9b}$$

with the same notation as in (2.8). A collision in three dimensions is equivalent to a one-dimensional collision along $\boldsymbol{n}$, while there are no forces acting along $\boldsymbol{t}$. We then achieve the simple result for the components along $\boldsymbol{t}$, $v_{t(i/j)}' = v_{t(i/j)}$. The collision along $\boldsymbol{n}$ can then be solved by using the derived result from the one-dimensional case. We insert the result from (2.5) for $v_{ni}'$ into the right hand side of (2.9a) to get the following

$$\begin{aligned}
\boldsymbol{v}_i' &= \frac{m_j v_{nj}(1+\xi) + v_{ni}(m_i - \xi m_j)}{m_i + m_j}\boldsymbol{n} + v_{ti}\boldsymbol{t} \\
&= \frac{(1+\xi)m_j(v_{nj} - v_{ni})}{m_i + m_j}\boldsymbol{n} + v_{ni}\boldsymbol{n} + v_{ti}\boldsymbol{t} \\
&= \boldsymbol{v}_i + \frac{(1+\xi)m_j(\boldsymbol{v}_j - \boldsymbol{v}_i)\cdot\boldsymbol{n}}{m_i + m_j}\boldsymbol{n} \\
&= \boldsymbol{v}_i + \left((1+\xi)\frac{m_j}{(m_i+m_j)}\frac{\Delta \boldsymbol{v}_{ij}\cdot\Delta\boldsymbol{x}_{ij}}{R_{ij}^2}\right)\Delta\boldsymbol{x}_{ij},
\end{aligned} \tag{2.10}$$

where (2.2b) and (2.8a) has been used to simplify the expression and transform the result back into the regular coordinate system. The derivation for $\boldsymbol{v}_j'$ is done similarly by inserting the results from (2.6) for $v_{nj}'$ into the right hand side of equation (2.9b). The resulting expression is on the form

$$\boldsymbol{v}_j' = \boldsymbol{v}_j - \left((1+\xi)\frac{m_i}{(m_i+m_j)}\frac{\Delta \boldsymbol{v}_{ij}\cdot\Delta\boldsymbol{x}_{ij}}{R_{ij}^2}\right)\Delta\boldsymbol{x}_{ij}. \tag{2.11}$$

The expressions for $\boldsymbol{v}_i'$ and $\boldsymbol{v}_j'$ given in (2.10) and (2.11) show that the velocities after collision depend on the ratio of one mass over the total mass of the two particles, the difference in velocity, the difference in position and the coefficient of restitution. As proposed, the quantities in (2.2) are present in the end result. The expressions in (2.10) and (2.11) are reduced to (2.5) and (2.6) for a one-dimensional system as given in Figure 2.1, as expected.

Another interesting property to note is that from the equations above we can derive that the energy dissipated from a collision between two particles with equal mass is equal to the expression

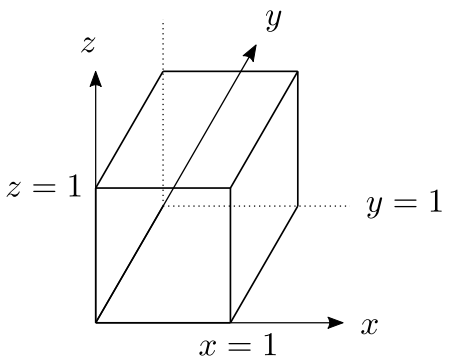



**Figure 2.4:** Illustration of the cubic box with boundaries at $x = 0$, $x = 1$, $y = 0$, $y = 1$, $z = 0$ and $z = 1$, which is the system we will use to contain $N$ particles colliding with a coefficient of restitution.

$\frac{1}{4}m(1-\xi^2)(\frac{\Delta \boldsymbol{v}_{ij}\cdot\Delta \boldsymbol{x}_{ij}}{R_{ij}})^2$, which reduces to zero for elastic collisions. The derivation is presented in Appendix B, and will later be used in the derivation of how the energy of a granular gas decreases as a function of time due to the inelastic collisions between the particles. The expressions in (2.10) and (2.11) constitute the collision rule applied in this project, where the post-collision velocities are given by the particle parameters and the pre-collision velocities. Note that the expressions in (2.10) and (2.11) are correct in any number of dimensions, where the dimensions are decided by the dimension of $\Delta \boldsymbol{x}_{ij}$ and $\Delta \boldsymbol{v}_{ij}$ and the velocities. This is the collision rule derived for smooth inelastic hard spheres in [see 3, pp. 5–8], and have been used by many authors, e.g. see [15].

From Eqs. (2.10) and (2.11) we can also get an expression for the coefficient of restitution in three dimensions[1]

$$\xi = -\frac{\Delta \boldsymbol{v}'_{ij}\cdot\Delta \boldsymbol{x}_{ij}}{\Delta \boldsymbol{v}_{ij}\cdot\Delta \boldsymbol{x}_{ij}}, \tag{2.12}$$

which resembles the expression in a one-dimensional system given in (2.1). The coefficient of restitution in multiple dimensions can thus be interpreted as the ratio of the relative velocities along the normal axis between the particle centers, which is as expected since in multiple dimensions a collision is reduced to a one-dimensional collision along the normal axis.

In addition to the collision rule we need to know the time until a particle collides with another particle. When we know the time until a particle-particle collision and how to update the velocities of the colliding particles we can implement a collision numerically. As we will perform simulations of particles colliding in a cubic box with boundaries at $x = 0$, $x = 1$, $y = 0$, $y = 1$, $z = 0$ and $z = 1$, the system in Figure 2.4, we will also look at the situation where a particle interacts with a wall. The interactions with the walls can be modelled in different ways, either by implementing hard reflecting walls or periodic boundaries.

In order to provide some short notation of the boundaries we will use the term vertical walls to represent the boundaries at $x = 0$ and $x = 1$, horizontal walls to represent the boundaries at $y = 0$ and $y = 1$, and the top/bottom wall to represent the boundaries at $z = 0$ and $z = 1$. Even though we will be using three-dimensional systems to perform numerical studies, we will use two-dimensional systems for visualization and illustration purposes. The naming scheme for the walls is a result of the latter as we want a two-dimensional system, which is a slice of the box in Figure 2.4 along the $z$-axis, to be confined by the horizontal and vertical walls.

[1]The expression can be derived in a simple manner by looking at the expression for $\Delta \boldsymbol{v}'_{ij} = \boldsymbol{v}'_j - \boldsymbol{v}'_i$ and inserting the collision rule in Eqs. (2.10) and (2.11), before finally multiplying the equation with $\Delta \boldsymbol{x}_{ij}$.

### 2.1.3 The time until a particle-particle collision

In order to compute the time until a particle collides with another particle we have to take into account the movement of both particles in order to know if there is a future collision on their current trajectories. To obtain the time of a collision, we need to solve an equation to determine if their trajectories will bring them into contact. Contact between particle $i$ and $j$ occurs when the distance between their centers, $R_{ij}$, is equal to $r_i + r_j$ as illustrated in Figure 2.2. Let $\boldsymbol{x}_i'$ and $\boldsymbol{x}_j'$ be the positions of the particles at the time of collision and let the collision occur at time $t + \Delta t^*$.

The position at the time of collision is also given by their position and their velocity at time $t$, since the velocity is constant until the collision. Thus, we have the following relation

$$\boldsymbol{x}_i' = \boldsymbol{x}_i + \boldsymbol{v}_i \Delta t^* \tag{2.13a}$$

$$\boldsymbol{x}_j' = \boldsymbol{x}_j + \boldsymbol{v}_j \Delta t^*, \tag{2.13b}$$

where $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ are the positions at time $t$, and $\boldsymbol{v}_i$ and $\boldsymbol{v}_j$ are the velocities at time $t$. We can write the square distance between the particle centers at the time of the collision as

$$R_{ij}^2 = (r_i + r_j)^2 = |\boldsymbol{x}_j' - \boldsymbol{x}_i'|^2. \tag{2.14}$$

We can derive an equation for $\Delta t^*$, which only has a solution if the particles collide, by inserting (2.13) into (2.14). We then obtain the following expression

$$R_{ij}^2 = |\boldsymbol{x}_j + \Delta t^* \boldsymbol{v}_j - (\boldsymbol{x}_i + \Delta t^* \boldsymbol{v}_i)|^2 = |\Delta \boldsymbol{x}_{ij} + \Delta t^* \Delta \boldsymbol{v}_{ij}|^2, \tag{2.15}$$

where the same notation as in (2.2) has been used. By expanding the expression in (2.15) we obtain the following second order equation for $\Delta t^*$,

$$(\Delta t^*)^2 \Delta \boldsymbol{v}_{ij}^2 + 2\Delta t^* (\Delta \boldsymbol{x}_{ij} \cdot \Delta \boldsymbol{v}_{ij}) + (\Delta \boldsymbol{x}_{ij}^2 - R_{ij}^2) = 0. \tag{2.16}$$

Introducing the coefficients $a = \Delta \boldsymbol{v}_{ij}^2$, $b = 2(\Delta \boldsymbol{x}_{ij} \cdot \Delta \boldsymbol{v}_{ij})$ and $c = (\Delta \boldsymbol{x}_{ij}^2 - R_{ij}^2)$, the two possible solutions to (2.16) have the familiar form

$$\Delta t^* = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a} = \frac{-\frac{b}{2} \pm \sqrt{(\frac{b}{2})^2 - ac}}{a}. \tag{2.17}$$

By reintroducing the expressions for the coefficients $a$, $b$ and $c$ we achieve the following expression by using the solution in (2.17)

$$\Delta t^* = \frac{-(\Delta \boldsymbol{x}_{ij} \cdot \Delta \boldsymbol{v}_{ij}) \pm \sqrt{d}}{\Delta \boldsymbol{v}_{ij}^2}, \tag{2.18}$$

where $d$ is given by

$$d = (\Delta \boldsymbol{x}_{ij} \cdot \Delta \boldsymbol{v}_{ij})^2 - \Delta \boldsymbol{v}_{ij}^2 (\Delta \boldsymbol{x}_{ij}^2 - R_{ij}^2). \tag{2.19}$$

First of all, one should note the characteristics needed for equation (2.18) to have a valid solution. $\Delta \boldsymbol{x}_{ij}$ is the distance vector at time $t$, thus $\Delta \boldsymbol{x}_{ij}^2 > R_{ij}^2$ which implies $\sqrt{d} \leq |\Delta \boldsymbol{x}_{ij} \cdot \Delta \boldsymbol{v}_{ij}|$. The physical interpretation of $\Delta t^*$, as the earliest non-negative collision time, leads to the following end result of (2.18)

$$\Delta t^* = \begin{cases} \infty & \text{if } \Delta \boldsymbol{x}_{ij} \cdot \Delta \boldsymbol{v}_{ij} \geq 0, \\ \infty & \text{if } d \leq 0, \\ -\frac{\Delta \boldsymbol{x}_{ij} \cdot \Delta \boldsymbol{v}_{ij} + \sqrt{d}}{\Delta \boldsymbol{v}_{ij}^2} & \text{otherwise,} \end{cases} \tag{2.20}$$

where $\Delta t^* = \infty$ is used to indicate that particle $i$ and particle $j$ will not collide on their current trajectory and that collision can be ignored. After determining which and when two particles will collide, the velocity of the particles after the collision is given by the collision rule in Eqs. (2.10) and (2.11).

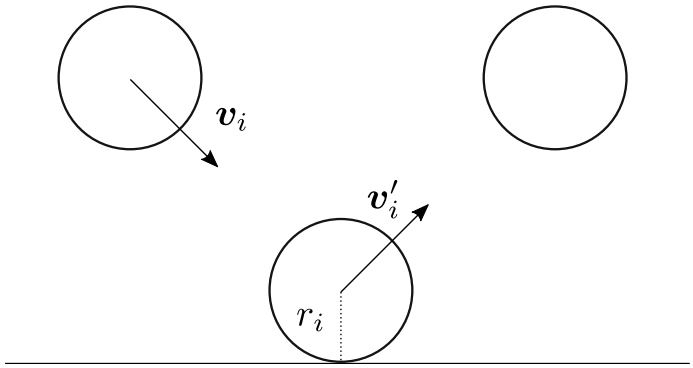



**Figure 2.5:** Illustration of particle $i$ colliding with a wall. The figure uses the same notation as Figure 2.2.

### 2.1.4 Particle-wall collision

Assume a system as illustrated in Figure 2.5, where particle $i$ moving the velocity $\boldsymbol{v}_i = [v_{xi}, v_{yi}, v_{zi}]$ and radius $r_i$ collides with a hard reflecting wall. For particle $i$, we want to know the time until the collision occurs. Since the particle is moving in three dimensions, from a starting position $\boldsymbol{x}_i = [x_i, y_i, z_i]$, we can treat the collision with a vertical, horizontal wall and top/bottom wall separately, but based on the same idea. The time until a collision with a vertical wall, $\Delta t^*$, is determined by the particle parameters $r_i$, $x_i$ and $v_{xi}$ and is simply computed as the distance travelled in order for the edge of the particle to be in contact with the wall over the speed along that axis. $\Delta t^* = \infty$ will be used to indicate that the particle will not collide with a wall on its current trajectory. Thus, the time until particle $i$ collides with a vertical wall is given by

$$\Delta t^* = \begin{cases} (1 - r_i - x_i)/v_{xi} & \text{if } v_{xi} > 0, \\ (r_i - x_i)/v_{xi} & \text{if } v_{xi} < 0, \\ \infty & \text{if } v_{xi} = 0. \end{cases} \tag{2.21}$$

If a particle has the velocity $\boldsymbol{v_i} = [v_{xi}, v_{yi}, v_{zi}]$ before colliding with a vertical wall, the velocity afterwards, $\boldsymbol{v_i'}$, is given as

$$\boldsymbol{v_i'} = [-\xi v_{xi}, \xi v_{yi}, \xi v_{zi}], \tag{2.22}$$

which is a simplified version of the collision rule for particle-particle collisions. In order to derive the expression in (2.22) one can look at the situation where particle $i$ collides with a wall, which can be approximated as a particle with infinite mass. The expression in (2.10) in the limit $m_j \to \infty$ reduces to $\boldsymbol{v}_i' = [-\xi v_{xi}, v_{yi}, v_{zi}]$ when particle $j$ is a vertical wall. To better see how we end up with this expression let us imagine that the wall in Figure 2.5 is the vertical wall given by $x = 1$[2]. For this situation we get $\frac{m_j}{m_i + m_j} \to 1$, $\Delta \boldsymbol{x}_{ij} = [r_i, 0, 0]$, $\Delta \boldsymbol{v}_{ij} = -\boldsymbol{v}_i$, and $R_{ij} = r_i$. By inserting these quantities into (2.10) the reduction is straightforward. For the limit $m_j \to \infty$ for a vertical wall we found an expression similar to (2.22), but we have chosen to multiply the other components with $\xi$ in order for the particle to maintain the angle relative to the wall after the collision. That is, the particle obeys the law of reflection. The wall gets momentum from the collision, but the momentum is assumed to be negligible, which holds for a particle with infinite mass.

The time until a collision with a horizontal wall is determined by the particle parameters $r_i$, $y_i$

[2]It does not matter which vertical wall we choose as the sign of $\Delta \boldsymbol{x}_{ij}$ is squared in the end.

and $v_{yi}$. The time until collision for particle $i$ is similar to equation (2.21), and given by

$$\Delta t^* = \begin{cases} (1 - r_i - y_i)/v_{yi} & \text{if } v_{yi} > 0, \\ (r_i - y_i)/v_{yi} & \text{if } v_{yi} < 0, \\ \infty & \text{if } v_{yi} = 0. \end{cases} \tag{2.23}$$

The velocity after colliding with a horizontal wall is similar to equation (2.22) and is given by

$$\boldsymbol{v_i'} = [\xi v_{xi}, -\xi v_{yi}, \xi v_{zi}]. \tag{2.24}$$

The expression (2.24) is derived with the same procedure as for (2.22).

The time until a collision with a top/bottom wall is determined by the particle parameters $r_i$, $z_i$ and $v_{zi}$. The time until collision for particle $i$ is similar to (2.21) and (2.23), and given by

$$\Delta t^* = \begin{cases} (1 - r_i - z_i)/v_{zi} & \text{if } v_{zi} > 0, \\ (r_i - z_i)/v_{zi} & \text{if } v_{zi} < 0, \\ \infty & \text{if } v_{zi} = 0. \end{cases} \tag{2.25}$$

The velocity after colliding with a top/bottom wall is similar to (2.22) and (2.24) and is given by

$$\boldsymbol{v_i'} = [\xi v_{xi}, \xi v_{yi}, -\xi v_{zi}]. \tag{2.26}$$

The expression (2.26) is derived with the same procedure as for (2.22) and (2.24).

## 2.2 Coefficient of restitution

There exists different ways to model a granular gas, depending on how one decides to model the coefficient of restitution of the system. A granular gas is a complex system, but the simple introduction of a coefficient of restitution for the pairwise collisions in a granular gas is an effective way to model such a system under the assumption that the dynamics of the system is given only by instantaneously pairwise collisions [see 11, pp. 19–20]. The simplest approximation is to assume that the coefficient of restitution is a constant $\in [0, 1]$ for all pairwise collisions between the particles in the system. A more complex case, referred to as viscoelastic particles, is used to study a granular gas where the coefficient of restitution is a function of the relative velocity of the colliding particles [see 11, pp. 23–26]. The latter is true for realistic particles, and makes intuitive sense from the fact that the deformation and forces acting upon the particles during a collision should depend on various particle parameters [see 11, p. 20]. In this project we will however for simplicity use a constant coefficient of restitution for the simulations. In addition we will throughout this report introduce some of the main different results for the case of viscoelastic particles compared to a system with a constant coefficient of restitution. Viscoelastic particles is one of topics that could be explored further, and are included in chapter 5.

## 2.3 Inelastic collapse

There are some challenges related to the numerical study of many-particle systems, such as gases. In the study of a granular gas, we are introduced to some problems not found in a molecular gas due to the different behaviour of inelastic and elastic collisions. For successive inelastic collisions, a phenomenon called inelastic collapse can occur, which is the case when the number of collisions per time goes towards infinity. Inelastic collapse was first discovered in a one-dimensional model, but have been shown to be present in two dimensions as well [see 18, p. 114]. We will illustrate the

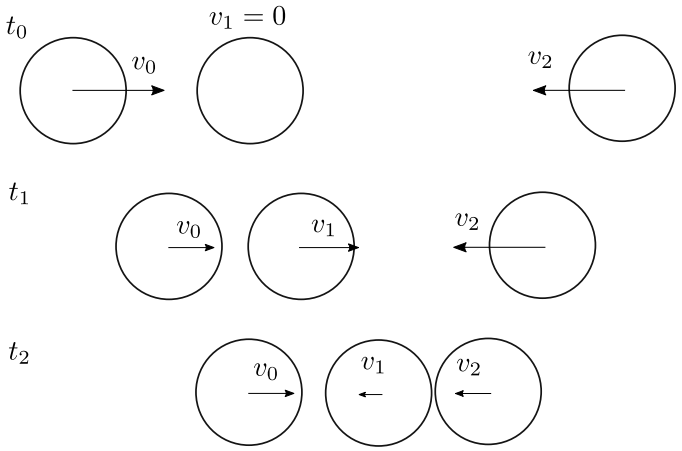



**Figure 2.6:** Illustration of the inelastic collapse problem. The figure shows a one-dimensional system with three identical particles undergoing inelastic collisions with a coefficient of restitution $\xi$. The initial system is given at time $t_0$. The first collision will be between the left particle and the middle particle leading to a situation as given at $t_1 > t_0$. The next collision will be between the middle particle and the right particle. For certain values of $\xi$ one can end up in a situation as given at $t_2 > t_1$. Such a sequence of alternating collisions between the particle in the middle and the particles on the sides can result in an infinite number of collisions in a finite amount of time before the kinetic energy is dissipated and the relative motion disappear.

inelastic collapse problem in a simple one-dimensional system with three identical particles given in Figure 2.6. Inserting $m_i = m_j = m$ into the collision rule in one dimension presented in Eqs. (2.5) and (2.6) gives the following collision rule

$$v_i' = \frac{1+\xi}{2} v_j + \frac{1-\xi}{2} v_i, \tag{2.27a}$$

$$v_j' = \frac{1+\xi}{2} v_i + \frac{1-\xi}{2} v_j. \tag{2.27b}$$

The initial system in presented at time $t_0$ in Figure 2.6, where the middle particle is at rest and the two other particles start with the same speed $v_0$. At the two later times, given by $t_1$ and $t_2 > t_1$ in Figure 2.6, the velocity of the particles has been computed as a result of the alternating collisions between the left and right particle with the middle particle, and are given in Table 2.1. From the expression of $v_2$ at time $t_2$ in Table 2.1 we can deduce that $\xi < \sqrt{5} - 2$ in order to achieve a situation as illustrated in Figure 2.6. This iterative scheme can be generalized and in order for the alternating sequence of collision to continue even further and lead to inelastic collapse it has been shown that $\xi \leq 7 - 4\sqrt{3} \approx 0.0718$ for three particles [see 11, pp. 36–40]. Such low values of $\xi$ are not commonly applied in the study of granular gases. However, for a higher number of particles, inelastic collapse can occur in a much wider range for $\xi$ [see 5, pp. 177–179]. It is thus possible for inelastic collapse to occur for values of $\xi$ not far from the elastic limit.

To the authors understanding inelastic collapse is a one-dimensional effect which occurs in higher dimensions if one gets an approximately one-dimensional chain of particles colliding under the right circumstances. Such behaviour is impossible to predict, and we thus need to handle the inelastic collapse in order for the simulations to be correct. It seems natural that the choice of boundary conditions impacts the probability of an inelastic collapse to occur. For reflecting boundaries it is possible for particles to get pressed towards and away from a wall simultaneously. As the particles cannot escape such a situation it is possible for one-dimensional chains to occur. On the other hand, it seems natural to assume that periodic boundaries reduce the possibility of such chains by letting

**Table 2.1:** Table of the velocities of the three particles in Figure 2.6 at the different times $t_0 < t_1 < t_2$. The velocities of the particles have been calculated from the collision rule given in (2.27).

| Time | $v_0$ | $v_1$ | $v_2$ |
|---|---|---|---|
| $t_0$ | $v_0$ | $0$ | $-v_0$ |
| $t_1$ | $\frac{1-\xi}{2}v_0$ | $\frac{1+\xi}{2}v_0$ | $-v_0$ |
| $t_2$ | $\frac{1-\xi}{2}v_0$ | $\frac{v_0}{4}(-\xi^2-2\xi-1)$ | $\frac{v_0}{4}(\xi^2+4\xi-1)$ |
| $\dots$ | | | |

the particles "escape" though the boundaries. The impact of boundary conditions in the simulation of particles colliding in a box is discussed in more detail in section 3.4.

For the set of particles involved in an inelastic collapse, the energy of the relative motion will be completely consumed by the dissipative collisions. As a result, the particles will move as a cluster with a common velocity after the collapse [see 11, p. 36]. Initially this effect does not seem to hold any distinct problem, but the effect causes several numerical problems since the collapse consists of an infinite number of collisions in a finite time [see 11, p. 40]. Inelastic collapse has been been shown to be present in two dimensions for a wide range of values for $\xi$ [see 19]. It is reasonable to assume that the possibility of inelastic collapse occurring in a three-dimensional system is significantly lower than for a two-dimensional system due to the increase in possible trajectories. Inelastic collapse is not present for viscoelastic particles since the coefficient of restitution approaches unity for low relative velocities between colliding particles [see 11, p. 40].

## 2.4 Maxwell-Boltzmann distribution

A system of many particles colliding with each other in a defined area while conserving energy will eventually reach equilibrium, which is the case for a molecular gas. The properties of a system in equilibrium are given by its temperature $T$. In equilibrium the particles will have a velocity distribution given by the Maxwell-Boltzmann distribution. The Maxwell-Boltzmann distribution states that a velocity component, e.g. $v_x$, of a particle has the following probability density

$$f_v(v_x) = \sqrt{\frac{m}{2\pi k_B T}} \exp\left(-\frac{m v_x^2}{2k_B T}\right), \tag{2.28}$$

where $k_B$ is the Boltzmann constant and $m$ is the mass of the particle [see 10, pp. 117–134]. We will now in turn derive the Maxwell-Boltzmann speed distribution in both two and three dimensions before computing some simple expectation values from the speed distributions.

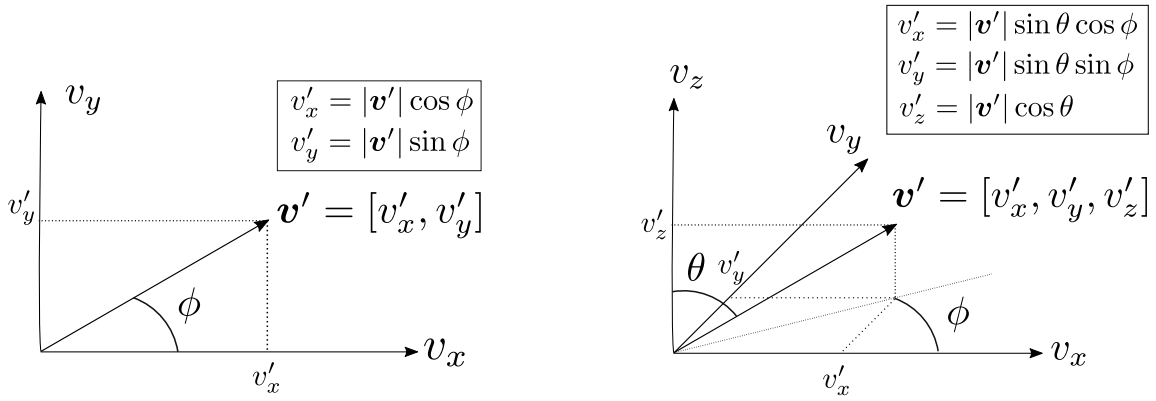
### 2.4.1 Speed distribution in two dimensions

The integral of the probability density function for the velocity components of a two-dimensional system is equal to unity and is given by

$$\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} dv_x dv_y f_v(v_x) f_v(v_y), \tag{2.29}$$

where $f_v(v_x)$ is the velocity distribution given in (2.28). The expression in (2.29) can be used to derive an expression for the speed distribution by introducing polar coordinates and considering the azimuthal symmetry. The transformation from Cartesian to polar coordinates is illustrated in Figure 2.7a. By using the following relations

$$dv_x dv_y = v dv d\phi,$$
$$v_x^2 + v_y^2 = v^2,$$




**(a)** Polar coordinates

**(b)** Spherical coordinates

**Figure 2.7:** Illustration of the different transformations one can introduce for a two-dimensional and a three-dimensional system. Due to the spherical symmetry of the Maxwell-Boltzmann velocity distribution one can apply these transformation to derive an expression for the speed distribution. Polar coordinates is used for two-dimensional systems where a velocity vector $\boldsymbol{v} = [v_x, v_y]$ can be described by the polar coordinates ($v$, $\phi$). Spherical coordinates is used for three-dimensional systems where a velocity vector $\boldsymbol{v} = [v_x, v_y, v_z]$ can be described by the spherical coordinates ($v$, $\phi$, $\theta$).

where $v$ is the speed of the particle and $\phi$ is the azimuth angle, in (2.29) we obtain an expression on the form

$$\int_0^\infty P(v)dv,$$

where $P(v)$ is the probability density function of the speed distribution. As illustrated in Figure 2.7a we can transform the integral by introducing polar coordinates. From Figure 2.7a we see that the following limits provide the same contributions

$$\begin{aligned} -\infty \leq v_x \leq \infty \\ -\infty \leq v_y \leq \infty \end{aligned} \Rightarrow \begin{aligned} 0 \leq v \leq \infty \\ 0 \leq \phi \leq 2\pi. \end{aligned}$$

The integral in (2.29) can thus be written as

$$\int_0^\infty dv \frac{m}{2\pi k_B T} v \exp\left(-\frac{mv^2}{2k_B T}\right) \int_0^{2\pi} d\phi,$$

where the angular part is trivial due to the azimuthal symmetry. In two dimensions the normalized Maxwell-Boltzmann speed distribution is thus on the form

$$P_{2D}(v) = \frac{m}{k_B T} v \exp\left(-\frac{mv^2}{2k_B T}\right). \tag{2.30}$$

### 2.4.2 Speed distribution in three dimensions

The derivation of the speed distribution in three dimensions is done in a similar fashion as in two dimensions. The difference arises from the transformation to spherical coordinates. We now want to look at the integral of the probability density function for the velocity components of a three-dimensional system, which is equal to unity and given by

$$\int_{-\infty}^\infty \int_{-\infty}^\infty \int_{-\infty}^\infty dv_x dv_y dv_z f_v(v_x) f_v(v_y) f_v(v_z). \tag{2.31}$$

The expression for the speed distribution can be derived by introducing spherical coordinates and considering the spherical symmetry. The transformation from Cartesian to spherical coordinates is illustrated in Figure 2.7b. By using the following relations

$$dv_x dv_y dv_z = v^2 \sin\theta dv d\phi d\theta,$$
$$v_x^2 + v_y^2 + v_z^2 = v^2,$$

where $\theta$ is the inclination angle, in (2.31) we get the probability density function of the speed distribution, $P(v)$. As illustrated in Figure 2.7b we can transform the integral by introducing spherical coordinates. From Figure 2.7b we see the following limits provide the same contributions

$$\begin{aligned} -\infty \leq v_x \leq \infty \quad & 0 \leq v \leq \infty \\ -\infty \leq v_y \leq \infty \Rightarrow \quad & 0 \leq \phi \leq 2\pi \\ -\infty \leq v_z \leq \infty \quad & 0 \leq \theta \leq \pi. \end{aligned}$$

The integral in (2.31) can thus be written as

$$\int_0^\infty dv \left(\frac{m}{2\pi k_B T}\right)^{\frac{3}{2}} v^2 \exp\left(-\frac{mv^2}{2k_B T}\right) \int_0^{2\pi} d\phi \int_0^\pi d\theta \sin\theta,$$

where the angular parts are trivial due to the spherical symmetry. In three dimensions the normalized Maxwell-Boltzmann speed distribution is thus on the form

$$P_{3D}(v) = \left(\frac{m}{2\pi k_B T}\right)^{\frac{3}{2}} 4\pi v^2 \exp\left(-\frac{mv^2}{2k_B T}\right). \tag{2.32}$$

### 2.4.3 Expectation values

The expressions for the speed distributions in Eqs. (2.30) and (2.32) can be used to compute expectation values for the speed and the square speed of the particles in a molecular gas. The definition of an expectation value given the distribution of the value can be written as

$$\langle v^\alpha \rangle = \int_0^\infty dv v^\alpha P(v), \tag{2.33}$$

where $P(v)$ is the speed distribution and $\alpha$ is a number. Certain parameters of interest for a distribution is the mean value, $\mu$, and the standard deviation, $\sigma$. The mean value can be derived from (2.33) by inserting $\alpha = 1$. The standard deviation [see 20, p. 78] is given as $\sqrt{Var(v)} = \sqrt{\langle v^2 \rangle - \langle v \rangle^2}$, which can be derived from (2.33) as a result of the expression for the mean value and the mean square value. The expressions for the expectation values giving the mean value, the mean square value and the standard deviation in two and three dimensions based on the speed distribution given in (2.30) and (2.32) are presented in Table 2.2. We also note that the results for the values of $\langle v^2 \rangle$ in Table 2.2 propose a relation between the average energy and temperature, which can be generalized by the equipartition theorem, which will be presented in section 2.5.

## 2.5 Kinetic gas theory

The energy in the many-particle system we will study is uniquely determined by its kinetic energy. This is the case when one ignores gravitational effects and neglects the possibility of making the particles rotate as a result of a collision. The energy of the system is given by

$$\mathcal{E} = \sum_{i=1}^{N} \frac{1}{2} m_i v_i^2, \tag{2.34}$$

**Table 2.2:** Some useful expectation values and the standard deviation, $\sigma$, in two and three dimensions of the Maxwell-Boltzmann speed distribution.

| Dimensions | $\langle v \rangle$ | $\langle v^2 \rangle$ | $\sigma$ |
|---|---|---|---|
| 2 | $\sqrt{\frac{\pi}{2}\frac{k_BT}{m}}$ | $2\frac{k_BT}{m}$ | $\sqrt{\frac{4-\pi}{2}\frac{k_BT}{m}}$ |
| 3 | $\sqrt{\frac{8}{\pi}\frac{k_BT}{m}}$ | $3\frac{k_BT}{m}$ | $\sqrt{\frac{3\pi-8}{\pi}\frac{k_BT}{m}}$ |

where $m_i$ is the mass of particle $i$, $v_i$ is the speed of particle $i$ and $N$ is the total number of particles. The equipartition theorem [see 21, pp. 66–67] states that every quadratic term in the energy function contributes with $k_BT/2$ to the average kinetic energy in thermal equilibrium. An example of a quadratic term is $v_x^2$ for a velocity component in the kinetic energy. The average kinetic energy is thus proportional to the thermal energy. For a three-dimensional system of particles with equal mass, an expression for the average kinetic energy per particle in equilibrium can be derived from the energy function given in equation (2.34). The derivation leads to the following

$$\langle E \rangle = \frac{\langle \mathcal{E} \rangle}{N} = \frac{1}{2}\frac{m}{N}\langle \sum_{i=1}^{N} v_i^2 \rangle = \frac{1}{2}m\langle v^2 \rangle = \frac{1}{2}m\langle v_x^2 + v_y^2 + v_z^2 \rangle = \frac{1}{2}m(\langle v_x^2 \rangle + \langle v_y^2 \rangle + \langle v_z^2 \rangle) = \frac{3}{2}k_BT, \quad (2.35)$$

where $\langle E \rangle$ is the average kinetic energy of the particles. Using the relation given in (2.35) we can compute the temperature of a gas of particles in equilibrium from the average kinetic energy of the particles. In order to achieve an equilibrium state in the system the energy must be constant. This can be achieved by setting $\xi = 1$ indicating elastic collisions or by giving the system additional energy equal to the energy dissipated from the collisions at all times. From now on the quantity $k_BT$ which is a energy, will be referred to as a temperature $T$, which is equivalent to setting $k_B$ equal to unity. Note that for a two-dimensional system $\langle E \rangle = T$.

A molecular gas, where energy is conserved, will thus evolve with a constant temperature $T$. The same relation between the average kinetic energy of the particles and the temperature given in (2.35) is also commonly used for a granular gas, where the temperature is referred to as the granular temperature [see 11, p. 51]. Due to the dissipative nature of a granular gas however, the energy and thus the temperature decays as a function of time. The evolution of temperature for a granular gas is known as Haff's law, which is presented in detail in section 2.6.

## 2.6 Haff's law

A granular gas is a many-particle system where the particle collisions occur dissipatively. The average kinetic energy of the particles will thus decay as a function of time. How fast the energy decays depends on how much energy is dissipated from a collision and the number of collisions, which depends on various system parameters. The evolution of the energy of a granular gas is commonly known as Haff's law, based on different properties of a granular system introduced by Haff in 1983 [see 22]. The average kinetic energy is given as a function of temperature from the relation in (2.35) and as commonly used for a granular gas we will use the granular temperature $T$ when deriving Haff's law. For a granular gas, the dissipated energy per collision on average, $\Delta T'$, follows the relation

$$\Delta T' \propto (1 - \xi^2)T, \quad (2.36)$$

where we have used that $\langle E \rangle \propto T$. The relation in (2.36) is derived and argued for in Appendix B, and is equal to the original statement used by Haff [see 22, p. 410]. Another derivation of (2.36) can be found in [11, p. 52].

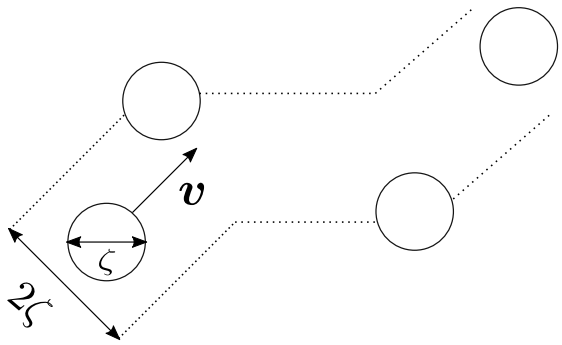



**Figure 2.8:** Illustration of the free path for a particle moving with the velocity $\boldsymbol{v}$. In a three-dimensional system, the particle will collide with other particles if they are inside the volume given by the moving cylinder inside the two dashed lines which are related to the diameter, $\zeta$, and the speed, $|\boldsymbol{v}|$, of the particle. The free path is used to compute the number of collisions in the system. In the figure, the other particles are at rest. If that is not the case one has to use the relative velocity between the particles in the calculation of the number of collisions.

In order to estimate the number of collisions in a gas, we have to look at the necessary conditions for particles to collide. Assume a system as illustrated in Figure 2.8, with a single particle moving, with the velocity $\boldsymbol{v}$ and diameter $\zeta = 2r$, towards a number of particles with the same diameter at rest. The particle will collide with another particle if $|\Delta \boldsymbol{x}| < \zeta$. This results in a collision if the center of a particle at rest is inside the volume given by the dashed lines in Figure 2.8. The volume inside the dashed lines in the figure is given by the moving cylinder with a radius equal to the diameter of the particle. The number of collisions is given by the number of particles in the volume given by the dashed lines. The dashed lines are separated by a distance $2\zeta$ for a gas of particles with equal radius. If we introduce the number density, $n$, given as $N/V$, where $V$ is the volume, we can write the average number of collisions in a time $\Delta t$, $N_c(\Delta t)$, as

$$N_c(\Delta t) = \pi \zeta^2 \cdot n \langle v \rangle \Delta t, \tag{2.37}$$

where $\langle v \rangle$ is the average speed of the particle. Eq. (2.37) gives the volume of the moving cylinder in Figure 2.8 with the cross-section $\pi\zeta^2$ times the density of particles in that volume. The result in (2.37) is based on a system where all other particles is at rest. This is not true for a system of particles colliding in a box. Therefore we have to use the average relative speed, $\langle u \rangle$, when considering the number of collisions for the particle. As shown in [10, pp. 136–137], $\langle u \rangle = \sqrt{2}\langle v \rangle$, which holds for a system having a speed distribution given by the Maxwell-Boltzmann distribution in any dimension. The speed distribution of a granular gas is not expected to follow the Maxwell-Boltzmann distribution due to the decay of energy, but the average relative velocity is of the same order as the average velocity [see 22, p. 410]. From the previous argument, we can thus conclude that

$$N_c(\Delta t) \propto r^2 n \langle v \rangle \Delta t \propto r^2 n \sqrt{\frac{T}{m}} \Delta t, \tag{2.38}$$

where the last step is based on that the kinetic energy is of the same order as the square average velocity [see 11, p. 52].

The dissipated energy in a granular gas, $\Delta T$ per time $\Delta t$, can thus be estimated as the energy dissipated per collision on average times the number of collisions in the system per time. We can thus write the following relation

$$\frac{\Delta T}{\Delta t} = \frac{\Delta T'}{\Delta t} N_c(\Delta t) = \tau_0' T^{3/2}, \tag{2.39}$$

where $\tau_0' \propto r^2 n(1-\xi^2)\frac{1}{\sqrt{m}}$ is a constant. In the limit $\Delta t \to 0$, (2.39) gives a separable differential equation for the average kinetic energy of the granular gas. We can solve the introduced differential equation with an initial energy and thus an initial temperature $T(t=0) = T_0$, leading to

$$T(t) = \frac{T_0}{(1 + t/\tau_0)^2}, \tag{2.40}$$

where $\tau_0^{-1} \propto r^2 n(1-\xi^2)\sqrt{\frac{T_0}{m}}$ is a constant related to system parameters. Note that for viscoelastic particles Haff's law gives a different decay of temperature [see 11, p. 53]. For a three-dimensional granular gas the exact expression for the constant $\tau_0^{-1}$ [see 11, p. 116] is given as

$$\tau_0^{-1} = \frac{1}{6}(1-\xi^2)\tau_c(0)^{-1}, \tag{2.41}$$

where $\tau_c(t)^{-1} \propto \sqrt{T(t)/m}$ is the mean collision time. The mean collision time, $\tau_c(t)$, is given as

$$\tau_c(t)^{-1} = 4\sqrt{\pi} g_2(\zeta)\zeta^2 n\sqrt{\frac{T(t)}{m}}, \tag{2.42}$$

where $\zeta$ is the diameter of the particles [see 11, p. 139] and $g_2(\zeta)$ is the contact value of the equilibrium pair correlation function for hard spheres [see 11, p. 59]. The contact value of the equilibrium correlation function for hard spheres is given as

$$g_2(\zeta) = \frac{2-\eta}{2(1-\eta)^3}, \tag{2.43}$$

where $\eta$ is the packing fraction, also known as the particle volume density. The packing fraction for a system of particles with equal radius, $r$, is given as

$$\eta = \frac{4}{3}\pi r^3 n = \frac{\pi\zeta^3}{6}, \tag{2.44}$$

which is simply the total volume of all the particles relative to the volume of the system. Eq. (2.44) is then a measure of how much of the volume in the system is occupied by the particles. Inserting the mean collision time in (2.42) at $t = 0$ into (2.41) we achieve that $\tau_0$, which can be interpreted as a characteristic timescale of the evolution of the granular temperature, can be written as

$$\tau_0^{-1} = \frac{2}{3}(1-\xi^2)\sqrt{\pi} g_2(\zeta) n\zeta^2\sqrt{\frac{T_0}{m}}. \tag{2.45}$$

Haff's law, given in (2.40), has been shown to be an accurate description for a granular gas in the homogeneous cooling state in both experimental and numerical studies [see 8, 14, 18]. The homogeneous cooling state is a term used to describe the state where a force-free granular gas cools due the inelastic collisions between particles and the particles are still uniformly distributed in the system [see 11, p. 51]. Due to the dissipative nature of the particle collisions, the uniform distribution of particles breaks down due to the formation of clusters. The formation of clusters was one of the effects which initiated the recent scientific interest in granular gases [see 11, p. 223]. For the remaining theory we will assume that the system is at all times in the homogeneous cooling state, as done in [11]. From the results in chapter 4 we will see that this assumption is justified.

## 2.7 Diffusion

There are several different properties which can be of interest when looking at a collection of particles. When working with concentrations of particles and particles who can move, to some degree freely,

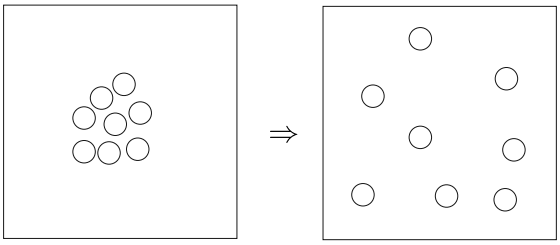


**Figure 2.9:** Illustration of a diffusion process, where the particle concentration is spread out.

one common approach is to study the problem as a diffusion[3] process. Diffusion is a process where a quantity, e.g. concentration or temperature, is driven by a gradient to spread out. The gradient drives the system to restore balance in the diffusing quantity by driving the quantity from regions of high concentration to regions of lower concentration as illustrated in Figure 2.9. Diffusion is a well studied subject in the field of physics, and can be modelled in several ways.

By limiting the focus to the position of a single particle in a box filled with particles, how the position evolves as a function of time can in addition to diffusion be modelled as a random walk or as Brownian motion. Random walk can be argued to be a valid approach if the particle moves around in a random manner. Brownian motion is a model where a particle moves randomly around due to collisions with other particles, which are discussed in greater detail in later sections. One typical quantity of interest for all these models is the so-called Mean Squared Displacement (MSD). It is known that the MSD of a set of particles in a diffusion process in one dimension will show the following property

$$\langle (x(t) - x_0)^2 \rangle = 2Dt, \tag{2.46}$$

where $D$ is the diffusion coefficient, $x_0$ is the initial position and $x(t)$ is the position at time $t$. This result was famously derived by Einstein in 1905 [see 23, pp. 556–559]. The relation given in (2.46) states that the MSD is linear in time. This is also a known result for random walks [see 10, pp. 152–153]. Numerical simulations show that the MSD of a molecular gas in two dimensions shows the linear trend proposed in (2.46) at long times, thus making it possible to achieve an estimation of the diffusion coefficient from numerical simulations [see 24, pp. 140–145].

Molecular dynamics of particles colliding without any external forces can be though of as a diffusion process with inertia, since it is not possible for a particle to suddenly change its velocity without colliding with another particle. Even if it collides with another particle, the velocity after colliding is related to the velocity prior to the collision, as derived in (2.10) and (2.11). One of the assumptions in the derivation of (2.46) is based on that one can assume the motion of a particle at different times to be independent [see 23, p. 556]. Therefore one should not expect the MSD to be linear in time for all times for a simulation of particles colliding in a box.

Whereas an MSD linear in time is widely regarded as normal diffusion, many systems have been discovered to exhibit what is now considered to be anomalous diffusion. Anomalous diffusion is often represented by the following power law of the MSD

$$\langle (x(t) - x_0)^2 \rangle \propto t^{\alpha}, \tag{2.47}$$

where $\alpha$ is a positive number. The value of $\alpha$ in (2.47) is used to characterize the different regions of diffusion. Anamalous diffusion is normally split in two different regions, where the term subdiffusion is used for $0 < \alpha < 1$ and superdiffusion is used for $\alpha > 1$ [see 25, 26, 27, 28].

[3]For an introduction to the diffusion equation and how diffusion is used to model the development of particle density see [10, pp. 148–153].

In addition, there also exist a variety of systems which exhibit what is known as ultraslow diffusion. Ultraslow diffusion is represented by the following logarithmic power law of the MSD

$$\langle (x(t) - x_0)^2 \rangle \propto \ln^{\alpha}(t), \tag{2.48}$$

where different values of $\alpha$ is obtained for different systems. A force-free granular gas with a constant coefficient of restitution, which is the topic studied in this thesis, is a system for which the dynamics are given by (2.48) at long times with $\alpha = 1$ [11, pp. 137–141, 25, pp. 1–2, 28, pp. 1–2]. A granular gas of viscoelastic particles is an example of a system with subdiffusive properties, where the MSD $\propto t^{1/6}$ [11, pp. 142–143, 15, p. 21794, 25, pp. 6–9]. We will thus see that the dynamics of the system of particles we will study in a box, and thus the dependence of time for the MSD will be different for a molecular gas and a granular gas. This should not be very surprising, given the fact that the dissipation of kinetic energy should provide different future dynamics leading to a smaller displacement. The negative spiral of continuously dissipating energy have an self increasing effect which makes it reasonable for the MSD to be logarithmic.

An commonly used model for systems exhibiting anomalous diffusion is known as scaled Brownian motion [see 28, 29, 30]. We will later use underdamped scaled Brownian motion as a model for the particles in a granular gas [see 15, 25], while we will for a molecular gas use a model known as the underdamped Langevin equation [see 25, p. 3].

## 2.8 Brownian motion

Brownian motion was first described by the biologist Robert Brown in 1827, during investigations of pollen [see 31, p. 47]. These observations were later used by Albert Einstein to provide a physical explanation for the movement of such particles suspended in fluids as a result of interactions with the particles in the fluid [see 23]. Brownian motion can thus be modelled as in Figure 2.10, with a large Brownian particle in a sea of smaller particles. On a microscopic level we know that the dynamics of the system in Figure 2.10 is given by collisions between particles, which is given by the collision rule in Eqs. (2.10) and (2.11). The microscopic description is what we aim to study using the event driven simulation of molecular dynamics. On a macroscopic level it is possible to model Brownian motion as a Stochastic Differential Equation (SDE), which is the topic of section 2.9.

We will in this project however only look at a simplification of Brownian motion where all particles are equal. Equal is here used to indicate that the mass and radius of all the particles in the system will be equal. This is not uncommon practice and exploits the fact that we can get ensemble averages by using the data of all the particles in a simulation [15]. The ensemble average is used to get the correct mean behaviour. While each particle gets a unique trajectory which does not necessarily give the expected result for e.g. diffusion, the ensemble average should converge towards the theoretical predictions. For a study of diffusion of a larger Brownian particle in a molecular gas see [24], and in a granular gas of viscoelastic particles see [27].

### 2.8.1 Diffusion coefficient

As one might expect, the diffusion coefficient is important when deriving the MSD. The self-diffusion coefficient, $D(t)$, for a granular gas of equal particles with a constant $\xi$ is given as

$$D(t) = \frac{4\mathcal{D}_0(t)}{(1+\xi)^2}, \tag{2.49}$$

where $\mathcal{D}_0(t)$ is the Enskog self-diffusion coefficient, given as

$$\mathcal{D}_0(t) = \frac{3}{8}\frac{1}{ng_2(\zeta)\zeta^2}\sqrt{\frac{T(t)}{m\pi}}, \tag{2.50}$$

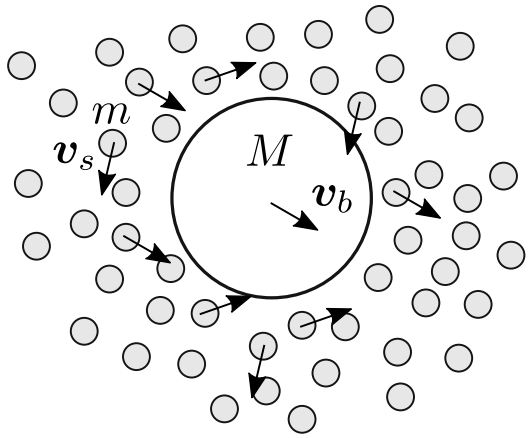



**Figure 2.10:** Illustration of a large Brownian particle, with mass $M$ and velocity $\boldsymbol{v}_b$, in a sea of smaller particles, with mass $m$ and velocity $\boldsymbol{v}_s$. On a macroscopic level, the dynamics of the Brownian particle is decided by friction and interactions with the particles in the fluid. On a microscopic level, the dynamics of the Brownian particle is determined by the collisions between the Brownian particle and the smaller particles. The macroscopic description can be modelled as a stochastic differential equation, addressed in section 2.9. The microscopic description is what we aim to study with the event driven simulation of molecular dynamics.

with the same definitions as in (2.42) [see 11, p. 162]. Self-diffusion is a term used to emphasize that the diffusion coefficient does only take into account the effects of the collisions between the particles in the system, since there are no other forces acing upon the particles. We will for simplicity refer to the self-diffusion coefficient as the diffusion coefficient, which is also called the diffusivity. The time dependence of (2.49) is given as the square root of the time dependence of the temperature, which for a granular gas is given by Haff's law in (2.40). For simplicity, we thus rewrite (2.49) into the following form

$$D(t) = \frac{D_0}{1 + t/\tau_0}, \tag{2.51}$$

where $D_0$ is a constant. The expression for $D_0$ is obtained by inserting (2.50) into (2.49) at time $t_0 = 0$ and is given as

$$D_0 = \frac{3}{2} \frac{1}{n g_2(\zeta) \zeta^2 (1+\xi)^2} \sqrt{\frac{T_0}{m\pi}}. \tag{2.52}$$

The diffusion coefficient for a gas of particles can also be written in the following form

$$D(t) = \frac{T(t)}{m\gamma(t)}, \tag{2.53}$$

where $\gamma(t)$ is the friction coefficient, sometimes also referred to as the damping coefficient. $\gamma(t)$ is also given as the inverse velocity autocorrelation time, which will appear in the derivations of the MSD [11, 25], making the relation in (2.53) useful later. The relation in (2.53) is known and used by many authors, see [11, pp. 137–148, 15, p. 21792, 25, p. 5]. From the relation in (2.53) and the definition of the diffusivity in (2.51) the friction coefficient can be written as

$$\gamma(t) = \frac{\gamma_0}{1 + t/\tau_0}, \tag{2.54}$$

where $\gamma_0$ is the initial friction coefficient. Note that the friction coefficient in (2.54) display the same time dependence as the diffusivity. We can thus relate the initial coefficients in the following way

$$\gamma_0 = \frac{T_0}{mD_0}, \tag{2.55}$$

from the two different expressions for the diffusivity in (2.51) and (2.53). The decay of the diffusivity and the friction coefficient is natural due to the decay of the kinetic energy in a granular gas. The relation between the temperature, the friction coefficient and the diffusion coefficient is also referred to as the Einstein[4]-Smoluchowski-Sutherland relation [see 25, p. 3].

An important note to make here is that for a molecular gas, we recover normal diffusivity with a constant diffusion coefficient given by (2.52) for $\xi = 1$ as a result of a constant temperature. In addition to the physical explanation, this can be seen mathematically as $\tau_0^{-1} = 0$ for $\xi = 1$. Thus, the time dependence in (2.51), (2.54) and in Haff's law in (2.40) vanish as expected. The dynamics of such a system is then determined by $D(t) = D_0$, $\gamma(t) = \gamma_0$ and $T(t) = T_0$.

### 2.8.2 Mean squared displacement

The coefficients for the diffusivity and the friction coefficient are given for a three-dimensional system, due to the fact that we will consider three-dimensional simulations of gases. The derivations of the MSD will be given for a one-dimensional system for simplicity, and the extension to higher dimensions is trivial. A system of particles in a force-free system have no reason to behave differently in any direction. Due to symmetry we should thus observe the following in three dimensions

$$\begin{aligned}
\langle(\boldsymbol{x}(t) - \boldsymbol{x}_0)^2\rangle =& \langle(\boldsymbol{x}(t) - \boldsymbol{x}_0) \cdot (\boldsymbol{x}(t) - \boldsymbol{x}_0)\rangle, \\
=& \langle|\boldsymbol{x}(t) - \boldsymbol{x}_0||\boldsymbol{x}(t) - \boldsymbol{x}_0|\rangle, \\
=& \langle\sqrt{(x(t) - x_0)^2 + (y(t) - y_0)^2 + (z(t) - z_0)^2}^2\rangle, \\
=& \langle(x(t) - x_0)^2 + (y(t) - y_0)^2 + (z(t) - z_0)^2\rangle, \\
=& \langle 3(x(t) - x_0)^2\rangle, \\
=& 3\langle(x(t) - x_0)^2\rangle,
\end{aligned} \tag{2.56}$$

where $\boldsymbol{x}(t) = [x(t), y(t), z(t)]$ is the position at time $t$, $\boldsymbol{x}_0 = \boldsymbol{x}(t = 0)$ and $\langle(x(t) - x_0)^2\rangle$ is the MSD in one dimension. The expressions for the MSD must thus be multiplied with 3 in order to compare with the simulation results or vice-versa.

In order to derive the MSD of a particle in one dimension we need to know the position of the particle. The difference between the position at time $t$, $x(t)$, and the initial position $x_0$ is given by the following integral

$$x(t) - x_0 = \int_0^t dt' v(t'), \tag{2.57}$$

where $v(t)$ is the velocity as a function of time. The expression in (2.57) can be used to compute the MSD by the following expression

$$\langle(x(t) - x_0)^2\rangle = \langle\left(\int_0^t dt' v(t')\right)^2\rangle = \langle\int_0^t dt_1 v(t_1) \int_0^t dt_2 v(t_2)\rangle = \int_0^t dt_1 \int_0^t dt_2 \langle v(t_1)v(t_2)\rangle, \tag{2.58}$$

where $\langle v(t_1)v(t_2)\rangle$ is known as the velocity autocorrelation function. Due to the symmetry of the velocity autocorrelation function the intregral in (2.58) can be simplified to

$$\langle(x(t) - x_0)^2\rangle = 2\int_0^t dt_1 \int_{t_1}^t dt_2 \langle v(t_1)v(t_2)\rangle, \tag{2.59}$$

[4]The relation was introduced by Einstein as what is now considered to be the Stokes-Einstein result [see 23, pp. 554–556].

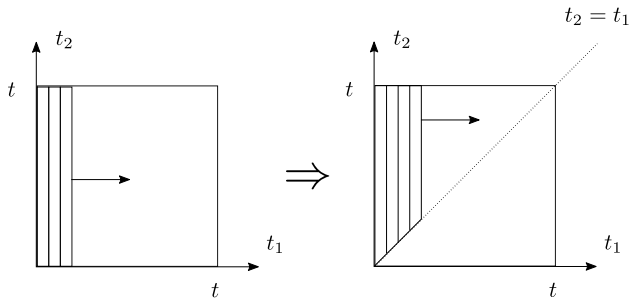



**Figure 2.11:** Due to the symmetry in the velocity autocorrelation function we can use the change of integration limits in the figure to change which area to integrate over. We then reduce the integral over a square to two times the integral over a triangle where $t_2 \geq t_1$, which turns out to be very convenient.

where it is assumed that $t_2 \geq t_1$ [see 11, p. 125]. The transformation is illustrated in Figure 2.11. The expression in (2.59) will be convenient later when deriving expressions for the MSD as the velocity autocorrelation function usually depends on $|t_2 - t_1|$ and not explicitly on $t_1$ or $t_2$. For a given expression of the velocity autocorrelation function, we can thus compute an analytical expression for the MSD [see 11, p. 125].

We can also use the expression for the MSD as a function of the velocity autocorrelation function to derive an expression for the diffusion coefficient as a function of the velocity autocorrelation function by combining Einstein's result in (2.46) and (2.59). For an equilibrium system the velocity autocorrelation function only depends on $t_2 - t_1$, which we will denote $\tau$ [see 11, p. 125]. As a result of the previous statement we can rewrite the velocity autocorrelation function in the following way

$$\langle v(t_1)v(t_2)\rangle = \langle v(0)v(\tau)\rangle. \tag{2.60}$$

If we now use $\tau$ as a integration variable instead of $t_2$, in addition to the relation in (2.60), the expression in (2.59) becomes

$$\langle (x(t) - x_0)^2\rangle = 2\int_0^t dt_1 \int_0^{t-t_1} d\tau \langle v(0)v(\tau)\rangle. \tag{2.61}$$

The integrand in (2.61) is not a function of $t_1$ anymore, and we can thus change the integration order to perform the trivial integral over $t_1$. The change of integration order is illustrated in Figure 2.12. We can thus obtain the following expression for the MSD

$$\langle (x(t) - x_0)^2\rangle = 2t\int_0^t d\tau \langle v(0)v(\tau)\rangle(1 - \frac{\tau}{t}). \tag{2.62}$$

As we want the long term behaviour of the MSD to coincide with Einstein's result in (2.46) we get the following expression for the diffusion coefficient

$$D = \int_0^\infty dt \langle v(0)v(t)\rangle, \tag{2.63}$$

where we have rewritten $\tau$ as $t$. The expression for the diffusion coefficient in (2.63) is known as a Green-Kubo relation, sometimes referred to as a fluctuation-dissipation relation [11, p. 126, 32].

The derivation above is given for an equilibrium system. A granular gas is not an equilibrium system as there are dissipative interactions in the system. Due to the dissipative interactions the diffusion coefficient is no longer constant as discussed earlier. The concept can however be generalized

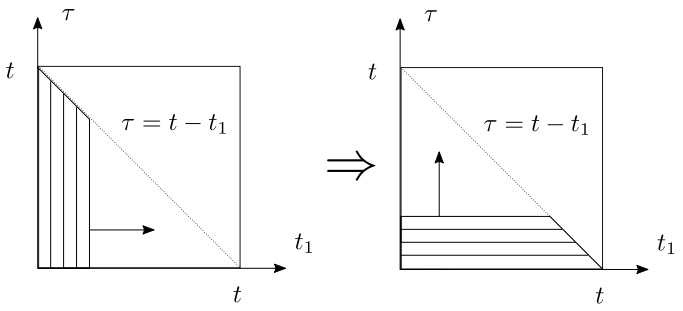



**Figure 2.12:** Illustration of the change of integration order in the derivation of the diffusion coefficient as a function of the velocity autocorrelation function.

for a granular gas, where the long term behaviour of the MSD can be computed from the following expression

$$\langle (x(t) - x_0)^2 \rangle = 2 \int_0^t dt' D(t'), \tag{2.64}$$

where the time dependence of the diffusion coefficient, $D(t)$, determines the time dependence of the MSD [see 11, pp. 125–126]. However due to (2.64) only capturing the long term behaviour the more general expression in (2.59) will be the main tool used to derive an expression for the MSD. We can however verify that due to constant diffusivity the MSD of a molecular gas is linear in time from (2.64) at long times. The time dependence of the diffusivity for a granular gas in (2.51) leads to an MSD which shows a logarithmic dependence of time for long times. The logarithmic dependence arises from (2.64) as $D(t) \propto t^{-1}$ for $t \gg \tau_0$.

We will now go into more details for the MSD of a molecular gas, before moving on to a granular gas. We will use an expression for the velocity autocorrelation function exploiting the fact that a molecular gas is an equilibrium system. For a granular gas, we must perform some rigorous mathematical treatment in order to achieve the correct velocity autocorrelation function.

### 2.8.3 Molecular gas

A molecular gas is an equilibrium system and the velocity autocorrelation function can thus be derived in a simple fashion. The velocity autocorrelation function for a molecular gas [see 11, pp. 136–137] is given as

$$\langle v(t_1) v(t_2) \rangle = \langle v^2 \rangle_{t_1} \exp\left(-\gamma_0 |t_2 - t_1|\right), \tag{2.65}$$

where $\gamma_0$ is the inverse velocity autocorrelation time, given as the initial friction coefficient. The expression in (2.65) for $t_2 \geq t_1$ and due to the equipartition theorem where $\langle v^2 \rangle = T/m = T_0/m$ can be written as

$$\langle v(t_1) v(t_2) \rangle = \frac{T_0}{m} \exp\left(-\gamma_0 (t_2 - t_1)\right). \tag{2.66}$$

As a verification of the velocity autocorrelation function we can insert (2.66) into the Green-Kubo relation in (2.63) and compute the diffusion coefficient. From this we obtain $D = T_0/(m\gamma_0)$, the same relation as we introduced for the diffusion coefficient in (2.53).

By inserting the velocity autocorrelation function in (2.66) into (2.59) we achieve the following double integral expression for the MSD

$$\langle (x(t) - x_0)^2 \rangle = 2 \frac{T_0}{m} \int_0^t dt_1 \int_{t_1}^t dt_2 \exp\left(-\gamma_0 (t_2 - t_1)\right). \tag{2.67}$$

Solving the integrals in (2.67) leads to an expression for the MSD given as

$$\langle (x(t) - x_0)^2 \rangle = 2D_0 t + \frac{2D_0}{\gamma_0} \left( \exp\left(-\gamma_0 t\right) - 1 \right), \tag{2.68}$$

where $D_0 = T_0/(m\gamma_0)$ as given by (2.53). As we will later see, the result in (2.68) is the same as we will achieve for the underdamped Langevin equation [see 25, p. 3]. It is often of interest to look at the expression for the MSD in different time limits. The expression in (2.68) behaves differently at $t \ll 1/\gamma_0$ compared to $t \gg 1/\gamma_0$. For the limit $t \gg 1/\gamma_0 \implies \exp\left(-\gamma_0 t\right) \to 0$, we obtain an MSD linear in time. For low values of $t$ we can use a Taylor expansion for the exponential term, where

$$\exp\left(-\gamma_0 t\right) \approx 1 - \gamma_0 t + \frac{t^2 \gamma_0^2}{2} + \mathcal{O}(t^3),$$

leading to an MSD equal to $\frac{T_0}{m} t^2$, which is the ballistic period where all particles initially move with a constant speed without being affected by other particles [see 25, p. 3]. The asymptotic behaviour of the MSD can thus be summarized with

$$\langle (x(t) - x_0)^2 \rangle \propto \begin{cases} t & \text{if } t \gg 1/\gamma_0 \\ t^2 & \text{if } t \ll 1/\gamma_0 \end{cases} \tag{2.69}$$

where we in the limit $t \gg 1/\gamma_0$ get the same time dependence of the MSD as derived by Einstein in (2.46), which is the result for the MSD at long times for normal diffusion.

### 2.8.4 Granular gas

Due to the inelastic collisions occurring in a granular gas, the velocity autocorrelation function is not as straightforward to derive as for a molecular gas. We can map a granular gas to a molecular gas with the time transformation $t \to \tau$, where $\tau$ is a timescale where the granular gas is stationary and thus the energy is constant. This is achieved with the transformation $d\tau = dt/\tau_c(t)$, where $\tau_c(t)$ is the mean collision time given in (2.42) [see 11, pp. 146–147].

First, we will for simplicity rewrite $\tau_c(t)$ in the following manner,

$$\tau_c(t) = A\sqrt{T(t)} = A\sqrt{T_0} \frac{1}{1 + t/\tau_0}, \tag{2.70}$$

where Haff's law (2.40) has been used in the last step and $A = 4\sqrt{\frac{\pi}{m}} g_2(\zeta) \zeta^2 n$, and $\tau_0$ is the characteristic time decay of the granular temperature. The new timescale $\tau$ is thus given as

$$\tau(t) = A\sqrt{T_0}\tau_0 \ln\left(1 + \frac{t}{t_0}\right). \tag{2.71}$$

Using the definition of $\tau_0$ from (2.45) we can rewrite (2.71) into

$$\tau(t) = \frac{6}{1 - \xi^2} \ln\left(1 + t/\tau_0\right), \tag{2.72}$$

which will be convenient later due to the velocity autocorrelation time in the new timescale.

We will also rewrite the velocity autocorrelation function to the following

$$\langle v(t_1) v(t_2) \rangle = v_T(t_1) v_T(t_2) \langle c(t_1) c(t_2) \rangle, \tag{2.73}$$

where $v_T(t) = \sqrt{2T(t)/m}$ is the thermal speed and $c = v(t)/v_T(t)$ [see 11, p. 146]. The trick used for the velocity autocorrelation function for a molecular gas can now be utilized in the timescale $\tau$

for $\langle c(\tau_1)c(\tau_2)\rangle$. Note that $\langle c^2\rangle = 1/2$ since $v_T^2(t) = 2T(t)/m$. The velocity autocorrelation function of the scaled velocities in the new timescale can thus be written as

$$\langle c(\tau_1)c(\tau_2)\rangle = \frac{1}{2}\exp\left(-\hat{\gamma_0}|\tau_2 - \tau_1|\right), \tag{2.74}$$

where $\hat{\gamma_0} = \gamma_0 t_c(0)$ is the inverse velocity autocorrelation time in the new timescale, and $\tau_1 = \tau(t_1)$ and $\tau_2 = \tau(t_2)$. We also note that $\tau(t)\tau_c(0) = \tau_0 \ln(1 + t/\tau_0)$, which simplifies the expression in (2.74). In order to use the expression in (2.59) for the MSD we need an velocity autocorrelation function given for $t_2 \geq t_1$. From the definition of $\tau(t)$ in (2.71) we see the following relation where $t_2 \geq t_1 \implies \tau(t_2) \geq \tau(t_1)$, for which $|\tau_2 - \tau_1| = (\tau_2 - \tau_1)$. Eq. (2.74) can now be written as a function of normal time as

$$\begin{aligned}
\langle c(t_1)c(t_2)\rangle &= \frac{1}{2}\exp\left(-\gamma_0\tau_c(0)(\tau_2(t_2) - \tau_1(t_1))\right) \\
&= \frac{1}{2}\exp\left(-\gamma_0\tau_0(\ln(1 + t_2/\tau_0) - \ln(1 + t_1/\tau_0))\right) \\
&= \frac{1}{2}\exp\left(-\gamma_0\tau_0(\ln(1 + t_2/\tau_0))\right)\exp\left(\gamma_0\tau_0(\ln(1 + t_2/\tau_0))\right) \\
&= \frac{1}{2}(1 + t_2/\tau_0)^{-\gamma_0\tau_0}(1 + t_1/\tau_0)^{\gamma_0\tau_0}.
\end{aligned} \tag{2.75}$$

Inserting (2.75) into (2.73) gives us the following expression for the velocity autocorrelation function

$$\langle v(t_1)v(t_2)\rangle = v_T(t_1)v_T(t_2)\frac{1}{2}(1 + t_2/\tau_0)^{-\gamma_0\tau_0}(1 + t_1/\tau_0)^{\gamma_0\tau_0}. \tag{2.76}$$

Inserting the definition of the thermal velocity and the definition of the granular temperature given in Haff's Law in (2.40) leads to the final expression for the velocity autocorrelation function

$$\langle v(t_1)v(t_2)\rangle = \frac{T_0}{m}(1 + t_2/\tau_0)^{-\gamma_0\tau_0 - 1}(1 + t_1/\tau_0)^{\gamma_0\tau_0 - 1}, \tag{2.77}$$

which is the one-dimensional equivalent to the three-dimensional expression used for the correlation function of a granular gas used in [15]. We will now use the expression in (2.77) to derive an expression for the MSD of a granular gas. Inserting (2.77) into (2.59) leads to the following integral expression for the MSD

$$\langle (x(t) - x_0)^2\rangle = 2\frac{T_0}{m}\int_0^t dt_1 \int_{t_1}^t dt_2 (1 + t_2/\tau_0)^{-\gamma_0\tau_0 - 1}(1 + t_1/\tau_0)^{\gamma_0\tau_0 - 1}. \tag{2.78}$$

By solving the integrals in (2.78) we achieve the following expression for the MSD of a granular gas

$$\langle (x(t) - x_0)^2\rangle = 2D_0\tau_0\left(\ln(1 + t/\tau_0) + \frac{1}{\gamma_0\tau_0}\left[(1 + t/\tau_0)^{-\gamma_0\tau_0} - 1\right]\right), \tag{2.79}$$

which turns out to be quite different to the expression for a molecular gas given in (2.68), as expected. For the expression in (2.79) it is natural to derive the asymptotic behaviour of the MSD in the limits $t \ll \tau_0$ and $t \gg \tau_0$. For the long time limit we will use that $(1 + t/\tau_0) = (t + \tau_0)/\tau_0 \to t/\tau_0$. Under this approximation it is clear that $(1 + t/\tau_0)^{-\gamma_0\tau_0} \to 0$ and the MSD will show an logarithmic dependence of time. For the low time limit we will again use Taylor expansions. By using the following approximations

$$\begin{aligned}
\ln(1 + t/\tau_0) &\approx \frac{t}{\tau_0} - \frac{t^2}{2\tau_0^2} + \mathcal{O}(t^3), \\
(1 + t/\tau)^{-\gamma_0\tau_0} &\approx 1 - \gamma_0 t + \frac{\gamma_0 t^2(\gamma_0\tau_0 + 1)}{2\tau_0} + \mathcal{O}(t^3),
\end{aligned}$$

we see that (2.79) reduces to $\frac{T_0}{m}t^2$, which again is known as the ballistic period. The system is in the ballistic period until collisions has started to occur, thus it is natural that the nature of the collision does not affect the MSD for such short times. The previous arguments are intuitive since there is no difference between a molecular and granular gas before any collisions have occurred. The asymptotic behaviour of the MSD for a granular gas with a constant $\xi$ is thus given as

$$\langle (x(t) - x_0)^2 \rangle \propto \begin{cases} \ln(t) & \text{if } t \gg \tau_0 \\ t^2 & \text{if } t \ll \tau_0 \end{cases} \tag{2.80}$$

where the logarithmic development for the MSD is a result of the decaying energy, and temperature of a granular gas. We have thus shown that a granular gas exhibits what is known as ultraslow diffusion for a constant coefficient of restitution [see 11, 15, 25, 28].

## 2.9 Brownian motion as a stochastic differential equation

The following section will provide detailed derivations starting from a macroscopic description of the dynamics of a single particle in a gas as illustrated in Figure 2.10. We aim here to see how different SDEs can be used as a simplified model of both a molecular and granular gas in order to compare the simulation results with the results obtained for the numerical solution of an SDE. As we will later see in chapter 4 the use of SDEs to model the dynamics of a single particle equal to the other particles in the gas provide similar results as the event driven simulation.

### 2.9.1 Langevin equation

The motion of the single particle can be modelled by a Langevin equation, originally suggested by Paul Langevin in 1908 [see 33]. A Langevin equation is an SDE stating how the velocity of the Brownian particle changes in time.

In one dimension, the Langevin equation of motion is given as

$$\frac{dv(t)}{dt} = -av(t) + b\Gamma(t), \tag{2.81}$$

where $a$ and $b$ are two coefficients used to model the strength of the friction and the random term, and $\Gamma(t)$ is a random force due to the effect on the Brownian particle from colliding with the other particles [see 32, p. 258]. The expression in (2.81) is simply Newton's equation of motion, where the movement of the particle is affected by a friction force and a random force. The equipartition theorem states that in an equilibrium state $\langle v^2(t) \rangle = T_0/m$ for the Brownian particle in a molecular gas. For this to hold there must exist a relation between $a$ and $b$ resulting from the different nature of the forces acting upon the particle. The friction term tries to drive the system to a stop, while the random term keeps the system alive by giving contributions to the velocity. A relation of this type can be found in several physical systems, where thermal equilibrium at long times is achieved for systems containing a dissipative term and a noise term. These kinds of relations are known from the fluctuation-dissipation theorem, where one tries to give explanations of why certain physical systems behave the way they do in thermal equilibrium [see 32]. Note also that even though a granular gas is not an equilibrium system, we can still use what is called a time local fluctuation-dissipation relation. This is a result of using the granular temperature, $T$, where the following relation is used: $\langle v^2(t) \rangle = T(t)/m = D(t)\gamma(t)$ given by the relation in (2.53) [see 25, pp. 3–5].

By comparing simulation results for Brownian motion given by an SDE and the event driven simulation of particles colliding, it is possible to validate the use of event driven molecular simulations to investigate properties on a macroscopic scale. The random force is used to describe the effects of the collisions on a macroscopic scale. We know however that the effects of the collisions are not

random at all on a microscopic scale, where they are given by the collision rule of the system. We are thus in a situation where the collisions must give the same effect on the Brownian particle as the random force in order to provide similar results for the two different simulation methods. Before we start a rigorous treatment of different Langevin equations we will provide a introduction to SDEs and how they are used to model Brownian motion.

### 2.9.2 Stochastic differential equation

An SDE is an differential equation of a stochastic variable, which contains a stochastic process, or more commonly known as a random phenomenon [see 31, p. 1]. A model for Brownian motion as an SDE can be written in the following stochastic differential form

$$dY_t = a(t, Y_t)dt + b(t, Y_t)dW_t, \tag{2.82}$$

where $Y_t$ is the velocity at time $t$ written as a stochastic variable, $W_t$ is a Wiener process, and $a$ and $b$ in general can be functions of the velocity and time. The term containing $a$ is often referred to as the drift, while the term containing $b$ and the Wiener process is often referred to as the diffusion. The expression in (2.82) is an example of an Itô process, which is a common way of expressing an SDE [see 34, p. XXI]. Due to the randomness of a Wiener process, it is a known that a Wiener process is almost surely a nowhere differentiable function of time [see 34, pp. 40–44 and 68–74]. As a result of the previous statement we often see SDEs written as a stochastic integral equation [see 34, p. 104]. The SDE modelling Brownian motion in (2.82) as a stochastic integral equation is given by

$$Y_t = Y_{t_0} + \int_{t_0}^{t} a(s, Y_s)ds + \int_{t_0}^{t} b(s, Y_s)dW_s. \tag{2.83}$$

The position of the Brownian particle is given by the velocity of the particle. Thus the position of the particle must follow the SDE

$$dX_t = Y_t dt, \tag{2.84}$$

where $X_t$ is the position at time $t$ given as a stochastic variable. In integral form, (2.84) can be written as

$$X_t = X_{t_0} + \int_{t_0}^{t} Y_s ds. \tag{2.85}$$

By using the expressions given in (2.83) and (2.85) it is possible to compute the position and the velocity and thus modelling the dynamics of the Brownian particle. This can be achieved numerically by applying a time discretization method in order to solve SDEs.

### 2.9.3 Euler-Maruyama scheme

The simplest time discretization method for SDEs is the Euler-Maruyama approximation. The Euler-Maruyama method can be considered to be the stochastic version of the Euler method for an Ordinary Differential Equation (ODE), and they share some similarities. The difference in the methods originates from the term representing the Wiener process. For simplicity we will use equidistant discretization times, where $\Delta t$ is given as the timestep value, and thus the difference in time between two computations of the velocity and position. The Euler-Maruyama approximation gives the following iterative scheme for $X_t$ and $Y_t$

$$X_{n+1} = X_n + Y_n \Delta t, \tag{2.86a}$$

$$Y_{n+1} = Y_n + a(t_n, Y_n)\Delta t + b(t_n, Y_n)\Delta W, \tag{2.86b}$$

where $\Delta W \sim \mathcal{N}(0, \Delta t)$ and the subscript $n$ is used to indicate the stochastic variables at time $t_n = n\Delta t$ [see 34, pp. 305–307]. Here we have used $\mathcal{N}(\mu, \sigma^2)$ to present a normal distributed

parameter with mean value $\mu = 0$ and variance $\sigma^2 = \Delta t$. The iterative scheme in (2.86) can be used to find the position and the velocity of the Brownian particle at different times from an initial position, $X_0$, and an initial velocity, $Y_0$, with an resolution in time given by $\Delta t$. The dynamics of a single particle will be decided by a unique Wiener process and each realization of the iterative scheme in (2.86) will give very different trajectories. In order to compute the MSD we must thus perform the Euler-Maruyama method for a total of $N$ particles to compute ensemble averages. The different coefficients deciding the values of $a$ and $b$ is given for a three-dimensional system. We will thus need to perform the Euler-Maruyama method for a three-dimensional system. This is achieved by changing $X$ to $\boldsymbol{X} = [X_x, X_y, X_z]$, $Y$ to $\boldsymbol{Y} = [Y_x, Y_y, Y_z]$ and $\Delta W$ to $\Delta \boldsymbol{W} = [\Delta W_x, \Delta W_y, \Delta W_z]$ where $\Delta W_x \neq \Delta W_y \neq \Delta W_z$, but all are $\sim \mathcal{N}(0, \Delta t)$. With this notation, $x$, $y$ and $z$ are used as subscripts to indicate that we have e.g. the position or the velocity along the different Cartesian coordinate axes.

### 2.9.4 Higher order schemes

Higher order schemes for solving SDEs have been considered, but compared to the results from the Euler-Maruyama method in chapter 4 they did not show any significant improvement. The SDEs which will be considered have a diffusion coefficient, $b$, not dependent on $Y_t$. Higher order methods such as the Milstein scheme or Strong Taylor approximations simplifies greatly for a constant $b(t, Y_t) = b(t)$ [see 34, pp. XXVII–XXIX]. It seems for us natural to assume that when some of the higher order terms in higher order methods disappear we do not see the improvement that one should see. For such simple SDEs as the model for Brownian motion given in (2.82) the Euler-Maruyama method provides satisfactory results.

We will now use two different SDEs to model the behavior of the particles in a gas. The first one can be used to approximate the behaviour of a molecular gas, and is known a underdamped Langevin equation [25]. The second one can be used to approximate the behaviour of a granular gas and is known as Underdamped Scaled Brownian Motion (UDSBM) [25].

### 2.9.5 Underdamped Langevin equation

The underdamped Langevin equation is given as the following second order ODE

$$\frac{d^2x(t)}{dt^2} + \gamma_0 \frac{dx(t)}{dt} = \sqrt{2D_0}\gamma_0 \Gamma(t), \tag{2.87}$$

which is an example of Newton's equation of motion with two forces modelling the dynamics of the particles in a molecular gas. The first one is friction, with a constant friction coefficient, the second term on the right hand side in (2.87). The second force is given by a random term, $\Gamma(t)$, used to describe the effect of the particle collisions, and is assumed to exhibit the following properties

$$\langle \Gamma(t) \rangle = 0, \tag{2.88a}$$

$$\langle \Gamma(t)\Gamma(t') \rangle = \delta(t - t'). \tag{2.88b}$$

For such a system the diffusion coefficient and friction coefficient is assumed to be constant. The random term is approximated as a Wiener process in the solution of the underdamped Langevin equation as an SDE. In order to create an iterative scheme for the underdamped Langevin equation using the Euler-Maruyama method we must first rewrite (2.87) into two first order ODEs, which are given as

$$\frac{dx(t)}{dt} = v(t), \tag{2.89a}$$

$$\frac{dv(t)}{dt} = -\gamma_0 v(t) + \sqrt{2D_0}\gamma_0 \Gamma(t). \tag{2.89b}$$

Then we rewrite the two first order ODEs in (2.89) into two first order SDEs written in integral form as

$$X_t = X_{t_0} + \int_{t_0}^{t} Y_s ds, \tag{2.90a}$$

$$Y_t = Y_{t_0} - \gamma_0 \int_{t_0}^{t} Y_s ds + \sqrt{2D_0}\gamma_0 \int_{t_0}^{t} dW_s. \tag{2.90b}$$

Comparing (2.90) with the general SDE used to describe Brownian motion we get the following values for the coefficients $a$ and $b$

$$a(t, Y_t) = -\gamma_0 Y_t, \tag{2.91a}$$

$$b(t, Y_t) = \sqrt{2D_0}\gamma_0. \tag{2.91b}$$

Inserting the values for the coefficients $a$ and $b$ in (2.91) into (2.86) we get the following iterative scheme for the underdamped Langevin equation with the Euler-Maruyama approximation

$$X_{n+1} = X_n + Y_n \Delta t, \tag{2.92a}$$

$$Y_{n+1} = Y_n - \gamma_0 Y_n \Delta t + \sqrt{2D_0}\gamma_0 \Delta W. \tag{2.92b}$$

It is also possible to derive an expression for the velocity autocorrelation function based on the solution to the Langevin equation of motion in (2.89b). The velocity autocorrelation function can as earlier discussed be used to derive an expression for the MSD of particles, whose dynamics follow the underdamped Langevin equation in (2.87). The value for the coefficient $b = \sqrt{2D_0}\gamma_0$ is a result of the fluctuation-dissipation theorem [32]. In order to show this we will start from a more general ODE for the velocity

$$\frac{dv(t)}{dt} + \gamma_0 v(t) = q\Gamma(t), \tag{2.93}$$

where $q$ is the constant to determine. Eq. (2.93) can be handled as a linear ODE. First we want to use $H$ as a help variable. $H$ is given as

$$H = \int_0^t dt'(-\gamma_0) = -\gamma_0 t.$$

We then multiply (2.93) with the integration factor $\exp(-H) = \exp(\gamma_0 t)$, leading to

$$\frac{dv(t)}{dt}\exp(\gamma_0 t) + \gamma_0 v(t)\exp(\gamma_0 t) = q\Gamma(t)\exp(\gamma_0 t), \tag{2.94}$$

where we now see that the left hand side of (2.94) is equal to $\frac{d}{dt}(v(t)\exp(\gamma_0 t))$. We then rewrite (2.94) into the following form

$$\frac{d}{dt}[v(t)\exp(\gamma_0 t)] = q\Gamma(t)\exp(\gamma_0 t). \tag{2.95}$$

Integrating both sides of (2.95) using $v(t = 0) = v_0$ as an initial condition leads to the following expression

$$v(t)\exp(\gamma_0 t) - v_0 = q\int_0^t dt'\Gamma(t')\exp(\gamma_0 t'). \tag{2.96}$$

The expression in (2.96) can be rewritten as a solution for the velocity given as

$$v(t) = v_0\exp(-\gamma_0 t) + q\int_0^t dt'\Gamma(t')\exp(-\gamma_0(t - t')). \tag{2.97}$$

Due to a non-analytical random term $\Gamma(t)$ is it not possible to derive a further expression for the velocity than what has been attempted in (2.97). Using the moments of $\Gamma(t)$ from (2.88) it is however possible to derive an expression for the velocity autocorrelation function from (2.97). The velocity autocorrelation function $\langle v(t_1)v(t_2)\rangle$ is found by inserting the velocity in (2.97) at the times $t_1$ and $t_2$ giving the following expression

$$\begin{aligned}\langle v(t_1)v(t_2)\rangle =&\langle\Big[v_0\exp\left(-\gamma_0 t_1\right)+q\int_0^{t_1}dt'\Gamma(t')\exp\left(-\gamma_0(t_1-t')\right)\Big]\Big[v_0\exp\left(-\gamma_0 t_2\right)+\\ &q\int_0^{t_2}dt''\Gamma(t'')\exp\left(-\gamma_0(t_2-t'')\right)\Big]\rangle.\end{aligned}\tag{2.98}$$

As a result of the properties of the random term (2.98) reduces to

$$\begin{aligned}\langle v(t_1)v(t_2)\rangle =&\langle v_0^2\rangle\exp\left(-\gamma_0(t_1+t_2)\right)+\\ &q^2\int_0^{t_1}dt'\int_0^{t_2}dt''\langle\Gamma(t')\Gamma(t'')\rangle\exp\left(-\gamma_0(t_2+t_1-t'-t'')\right),\end{aligned}\tag{2.99}$$

as the cross terms are equal to zero as a result of (2.88a). Inserting (2.88b) into (2.99) and using the definition of the Dirac delta function in one of the integrals simplifies the expression further. We then obtain the following

$$\langle v(t_1)v(t_2)\rangle=\langle v_0^2\rangle\exp\left(-\gamma_0(t_1+t_2)\right)+q^2\int_0^{t_1}dt'\exp\left(-\gamma_0(t_2+t_1-2t')\right),\tag{2.100}$$

where we have chosen to use the Dirac delta for the integral over $dt''$. From the integral in (2.100) we obtain an expression given as

$$\langle v(t_1)v(t_2)\rangle=\langle v_0^2\rangle\exp\left(-\gamma_0(t_1+t_2)\right)+\frac{q^2}{2\gamma_0}\exp\left(-\gamma_0(t_2-t_1)\right)-\frac{q^2}{2\gamma_0}\exp\left(-\gamma_0(t_2+t_1)\right).\tag{2.101}$$

If we had chosen to use the Dirac delta for the integral of $dt'$ we would have obtained a middle term with $\exp\left(-\gamma_0(t_1-t_2)\right)$. This leads to the conclusion that the correct physics would need to use the absolute difference between the times. In equilibrium we also assume that $t_1+t_2\gg 1/\gamma_0$ leading to the following expression for the velocity autocorrelation function

$$\langle v(t_1)v(t_2)\rangle=\frac{q^2}{2\gamma_0}\exp\left(-\gamma_0|t_2-t_1|\right).\tag{2.102}$$

The expression in (2.102) for $t_2=t_1=t$ leads to $\langle v^2(t)\rangle=q^2/(2\gamma_0)$, which can also be expressed as $T/m=T_0/m$ due to the conservation of energy and the equipartition theorem. We then obtain

$$\langle v(t_1)v(t_2)\rangle=\frac{T_0}{m}\exp\left(-\gamma_0|t_2-t_1|\right),\tag{2.103}$$

as the expression for the velocity autocorrelation function based on the underdamped Langevin equation in (2.89b). The result in (2.103) is equivalent to the expression used for a molecular gas in (2.66). We can also determine the expression for $q$ due to the equipartition theorem. Earlier we used the following relation

$$\frac{q^2}{2\gamma_0}=\frac{T_0}{m}.\tag{2.104}$$

From the expression in (2.104) we achieve the following expression for $q$,

$$\begin{aligned}q&=\sqrt{\frac{2\gamma_0 T_0}{m}},\\ q&=\sqrt{2\gamma_0^2 D_0},\\ q&=\sqrt{2D_0}\gamma_0,\end{aligned}\tag{2.105}$$

where the last step was done by using $T_0/m = D_0\gamma_0$, given from the relation in (2.53). For the expression for $q$ in (2.105) we achieve that (2.93) is equal to (2.89b) as expected. Since the velocity autocorrelation function for the underdamped Langevin equation is equal to the velocity autocorrelation function for a molecular gas, we should obtain similar results for the MSD by solving the underdamped Langevin equation with the Euler-Maruyama method in (2.92) as for event driven simulations of a molecular gas.

### 2.9.6 Underdamped Scaled Brownian Motion

UDSBM is given by the following second order ODE

$$\frac{d^2x(t)}{dt^2} + \gamma(t)\frac{dx(t)}{dt} = \sqrt{2D(t)}\gamma(t)\Gamma(t), \tag{2.106}$$

where $\Gamma(t)$ is again assumed to follow the properties in (2.88). We also see that UDSBM is another example of Newton's equation of motion, where we now have forces with coefficients that depend on time. Comparing UDSBM with the underdamped Langevin equation we see that they are similar, but the diffusivity and the friction is a function of time for UDSBM. This difference is expected since for UDSBM energy is not conserved and thus expected to decrease as a function of time. This link between UDSBM and the underdamped Langevin equation provides the same connection as a granular and a molecular gas, where the former is a system with a time dependent diffusivity and friction coefficient and the latter is a system with a constant diffusivity and friction coefficient. From this one can argue that the case of UDSBM is naturally used as a possible way to model a granular gas. We will use the expression for the diffusivity and the friction coefficient for a granular gas given in (2.51) and (2.54), where the time dependency is a result of the inelastic collisions. Similar as for the underdamped Langevin equation, (2.106) can be written as two first order ODEs as

$$\frac{dx(t)}{dt} = v(t), \tag{2.107a}$$

$$\frac{dv(t)}{dt} = -\gamma(t)v(t) + \sqrt{2D(t)}\gamma(t)\Gamma(t). \tag{2.107b}$$

We then rewrite (2.107) into two first order SDEs in integral form as

$$X_t = X_{t_0} + \int_{t_0}^{t} Y_s ds, \tag{2.108a}$$

$$Y_t = Y_{t_0} - \int_{t_0}^{t} \gamma(s)Y_s ds + \int_{t_0}^{t} \sqrt{2D(s)}\gamma(s)dW_s. \tag{2.108b}$$

Comparing (2.108) with the general SDE used to describe Brownian motion we get the following values for the coefficients $a$ and $b$

$$a(t, Y_t) = -\gamma(t)Y_t \tag{2.109a}$$

$$b(t, Y_t) = \sqrt{2D(t)}\gamma(t) \tag{2.109b}$$

Inserting the values for the coefficients $a$ and $b$ in (2.109) into (2.86) we get the following iterative scheme for UDSBM with the Euler-Maruyama approximation

$$X_{n+1} = X_n + Y_n\Delta t, \tag{2.110a}$$

$$Y_{n+1} = Y_n - \gamma_n Y_n \Delta t + \sqrt{2D_n}\gamma_n \Delta W, \tag{2.110b}$$

where $\gamma_n = \gamma(t_n)$, $D_n = D(t_n)$ and $t_n = n\Delta t$. The iterative scheme in (2.110) is equivalent to the iterative scheme used in [25, pp. 8–9].

A similar procedure as the one introduced for the underdamped Langevin equation can be utilized to compute the velocity autocorrelation function for UDSBM from the ODE given in (2.107b). We will here use the result, and refer to Appendix C for the derivation. The velocity autocorrelation function derived from (2.107b) is given as [see 25, p. 7]

$$\langle v(t_1)v(t_2)\rangle = \frac{T_0\gamma_0\tau_0}{m(\gamma_0\tau_0-1)}(1+t_1/\tau_0)^{\gamma_0\tau_0-2}(1+t_2/\tau_0)^{-\gamma_0\tau_0}. \tag{2.111}$$

In the limit $\gamma_0\tau_0 \gg 1$, the expression for the velocity autocorrelation function for UDSBM in (2.111) can be compared with the velocity autocorrelation function for a granular gas in (2.77). We also note that in the same limit, for $t_1 = t_2 = t$, (2.111) gives us a similar dependence on time as Haff's law in (2.40). However since the expressions for the velocity autocorrelation functions are not equal the results obtained for the MSD with the Euler-Maruyama approximation for UDSBM can differ from the MSD achieved from event driven simulations of a granular gas. Thus we will derive the MSD expression obtained from the velocity autocorrelation function for UDSBM presented in (2.111). Inserting (2.111) into (2.59) we obtain the following expression for the MSD

$$\langle (x(t)-x_0)^2\rangle = 2\frac{T_0\gamma_0\tau_0}{m(\gamma_0\tau_0-1)}\int_0^t dt_1(1+t_1/\tau_0)^{\gamma_0\tau_0-2}\int_{t_1}^t dt_2(1+t_2/\tau_0)^{-\gamma_0\tau_0}. \tag{2.112}$$

By solving the integrals in (2.112) we obtain the following expression for the MSD

$$\langle (x(t)-x_0)^2\rangle = \frac{2D_0\gamma_0^2\tau_0^2}{(\gamma_0\tau_0-1)^2}[\tau_0\ln(1+t/\tau_0) + \frac{\tau_0}{\gamma_0\tau_0-1}\left((1+t/\tau_0)^{-\gamma_0\tau_0+1}-1\right)], \tag{2.113}$$

which again, as expected, is similar to the result for a granular gas in (2.79) in the limit $\gamma_0\tau_0 \gg 1$. The expression for the MSD in (2.113) shows the same asymptotic behaviour in the limits as proposed for a granular gas in (2.80), which makes it understandable that UDSBM can be used as an approximation of a granular gas for an appropriate choice of parameters. From these expressions it seems that we do not expect identical results for the MSD from the Euler-Maruyama approximation of UDSBM and the simulation results of the event driven simulation of a granular gas, but similar types of results exhibiting the same asymptotic behaviour for the timescales $t \gg \tau_0$ and $t \ll \tau_0$.

By capturing the dynamics of a granular gas, UDSBM serve as a practical and a simple way of achieving theoretical results for a granular gas. Even though a force-free description of a granular gas can be simulated with an event driven simulation, the addition of forces, which can be necessary in order to achieve realistic systems, is not straightforward. For some forces, the idea of an event driven simulation seem almost impossible, leaving us with the choice of solving Newton's equations of motion, which has proven to be difficult for many-particle systems. The addition of forces to UDSBM is however possible and does not effect the Euler-Maruyama scheme more than by adding an appropriate term. For a study of ultraslow scaled Brownian motion in a confined area see [28].

## 2.10 Ergodicity

Ergodicity is a theorem in statistical physics which states that the long term time average of a physical quantity is equal to the ensemble average [see 15, 25, 26], as originally stated by Boltzmann as the ergodic hypothesis [see 35]. For a system of particles to be ergodic, the following property of the MSD must hold

$$\langle\overline{\delta^2(\Delta)}\rangle = \langle (x(\Delta)-x_0)^2\rangle, \tag{2.114}$$

where $\langle\overline{\delta^2(\Delta)}\rangle$ is the time averaged MSD, and $\Delta$ is the lag time. The MSD presented in this report so far, is the ensemble MSD, and we will continue to only use MSD when discussing the ensemble MSD only. The time averaged MSD of a time series is often given as a function of the lag time as

$$\langle\overline{\delta^2(\Delta)}\rangle = \frac{1}{t-\Delta}\int_0^{t-\Delta} dt'\langle[(x(t'+\Delta)-x_0)-(x(t')-x_0)]^2\rangle, \tag{2.115}$$

where $t$ is the total length of the time series. The expression in (2.115) is often used for experiments and computer simulations, where the time average is achieved as a sum of square differences of the positions at different times [see 26, pp. 24130–24131]. Note that compared to others, we have added the position relative to the starting position to easier see the connection to the ensemble MSD, while the two terms of $x_0$ will cancel in order to get the more commonly used expression for the time averaged MSD in [15, 25, 26]. As shown in [15, 25] the expression in (2.115) can be used to show the non-ergodic behaviour of a granular gas, and the ergodic behaviour of a molecular gas.

In order to make it clear how it is possible to use (2.115) to show whether or not a system is ergodic from the MSD, we must expand the integrand. The integrand in (2.115) can be rewritten as

$$\begin{aligned}\langle[(x(t'+\Delta)-x_0)-(x(t')-x_0)]^2\rangle =&\langle(x(t'+\Delta)-x_0)^2\rangle+\langle(x(t')-x_0)^2\rangle\\ &-2\langle(x(t')-x_0)(x(t'+\Delta)-x_0)\rangle,\end{aligned} \tag{2.116}$$

where we recognize the first and second term as the MSD at the time $t'+\Delta$ and $t'$ respectively. The last term can be derived in a similar fashion as the MSD in (2.58). We can thus write the third term in (2.116) as

$$\langle(x(t'+\Delta)-x_0)(x(t')-x_0)\rangle = \int_0^{t'} dt_1 \int_0^{t'+\Delta} dt_2 \langle v(t_1)v(t_2)\rangle, \tag{2.117}$$

which differs from (2.58) due to the limits of the integral over $t_2$. By splitting the integral over $t_2$ into two parts we can get the MSD at time $t'$ in addition to an additional term $A$. The expression in (2.117) can thus be written as

$$\langle(x(t'+\Delta)-x_0)(x(t')-x_0)\rangle = \langle(x(t')-x_0)^2\rangle + \int_0^{t'} dt_1 \int_{t'}^{t'+\Delta} dt_2 \langle v(t_1)v(t_2)\rangle, \tag{2.118}$$

where we will call the second term of the right hand side of (2.118) $A$. We now achieve the following expression for the time averaged MSD by inserting (2.118) into the integrand in (2.116) of the expression in (2.115)

$$\langle\overline{\delta^2(\Delta)}\rangle = \frac{1}{t-\Delta}\int_0^{t-\Delta} dt'(\langle(x(t'+\Delta)-x_0)^2\rangle - \langle(x(t')-x_0)^2\rangle - 2A). \tag{2.119}$$

In order to compute an analytical expression for the time averaged MSD from (2.119) we must first compute $A$ from an expression for the velocity autocorrelation function, before using analytical expressions for the MSD at the times $t'+\Delta$ and $t'$.

## 2.11 Central limit theorem

In order to compute certain quantities, such as the MSD we need to compute averages over multiple runs in order to achieve estimates for the mean behaviour of the particles in a gas. The average value should approach the theoretical expectation value by taking the average value over enough runs. This can be formalized by the Lindeberg-Lévy central limit theorem. Suppose that the average is taken over a sequence of $M$ random variables, $A_i$ where $i \in [1,\ldots,M]$. These random variables are drawn from the same distribution where $\mu$ is the mean and $\sigma^2 < \infty$ is the variance. Let $\mu_M$ be given as the computed average value, also known as the sample average, given as

$$\mu_M = \frac{1}{M}\sum_{i=1}^{M} A_i. \tag{2.120}$$

The Lindeberg-Lévy central limit theorem states that

$$\frac{\sqrt{M}}{\sigma}(\mu_M - \mu) \sim \mathcal{N}(0, 1), \tag{2.121}$$

where $\sigma$ is the generally unknown standard deviation of the true distribution [see 20, p. 357]. The error, which is the difference between the computed mean value and the true mean, is thus given as

$$e_M = \mu_M - \mu \sim \mathcal{N}(0, \frac{\sigma^2}{M}), \tag{2.122}$$

where $e_M$ is the error for the average value computed from $M$ values. The expression in (2.122) states that the standard deviation of the error is equal to $\sigma/\sqrt{M}$. By assuming that the standard deviation of the true distribution is equal to the estimated standard deviation calculated from a set of measured data points, we can use the result in (2.122) to compute an estimation of the error in the computed average value.

# Chapter 3

# Numerical modelling

In the following section the numerical modelling used in this project in order to study particles colliding in a three-dimensional box is presented in detail. Some of the numerical modelling have already been introduced in chapter 2, but here we will present a detailed introduction. We will use the majority of this chapter to discuss how the event driven simulation has been implemented and what kind of setup is needed to conduct studies of many-particle systems. We will also include how we have solved the SDEs modelling particles in a molecular and a granular gas using the Euler-Maruyama method.

## 3.1 Overview

Most of the results in chapter 4 are based on an event driven simulation[1] of molecular dynamics, consisting of particles colliding in a cubic box with boundaries at $x = 0$, $x = 1$, $y = 0$, $y = 1$, $z = 0$ and $z = 1$, illustrated in Figure 2.4. An event driven simulation is a simulation method where the idea is to increment time between successive events, which for the case of particles colliding in a box are valid collisions. As we simplify the dynamics of a many-particle system of particles in a box to instantaneous pairwise particle collisions we have a system suited for an event driven simulation as we know how to handle the collisions based on a collision rule and how to update the system between collisions as the particles have a constant velocity between collisions. The box is therefore an $L \times L \times L$ box, where $L$ is the length of the system and the volume is equal to unity. By setting $L = 1$ we make all other length parameters dimensionless, since they can be seen as a ratio of the length relative to the length of the box. The particles are modelled as hard spheres in a three-dimensional system. Each particle is thus described by eight parameters, a position $\boldsymbol{x}_i = [x_i, y_i, z_i]$, a velocity $\boldsymbol{v}_i = [v_{xi}, v_{yi}, v_{zi}]$, a mass $m_i$ and a radius $r_i$. The box of particles will contain a total of $N$ particles, colliding with a coefficient of restitution $\xi$, which is assumed to be constant for a given system. Setting $\xi = 1$ will simulate a molecular gas, where the particle collisions are elastic. In order to simulate a granular gas, we have to use $0 < \xi < 1$, indicating that the particle collisions are inelastic. The position of the particles will change as time is incremented by moving with constant speed until the next collision in the system. When two particles collide, their velocity is updated from the collision rule in Eqs. (2.10) and (2.11).

Due to the amount of constant parameters and the need to easily have access to the parameters of all particles, this numerical project is a good area for object-oriented programming. The written code contains three main classes, ParticleBox, Simulation and SDESolver. ParticleBox and Simulation is used together to conduct the event driven simulation, while SDESolver is used to numerically solve

[1] see `https://algs4.cs.princeton.edu/61event/` for an introduction on how to use event driven simulation to study particle collisions.

the underdamped Langevin equation and UDSBM with the Euler-Maruyama method. ParticleBox has the variables, arrays and functionality for the system of particles, and Simulation is used to perform the event driven simulation. As mentioned in the preface, all the code used to achieve the results in this thesis can be found on github[2]. From an initial setup for positions, velocities, mass and radius for all particles, the system is incremented in time by moving from event to event. The parameters of the particles are saved as arrays where the index can be used to identify particles. In that manner one can easily identify and update e.g. the velocity or the position of a given particle. The indexing of particles is not explicitly important, but we have to use it to update the velocity of the correct particles during a collision and extract the correct particles when computing different quantities. Working in the high-level programming language Python, one should whenever possible avoid for-loops by doing operations on the entire array, known as vector operations. In Python this can be achieved by using wrappers to compiled languages provided by third party modules such as NumPy and SciPy.

In the setup we have used there are four different types of objects that a particle can collide with. These four consist of another particle, a horizontal wall and a vertical wall and a top/bottom wall. Remember that we use vertical walls to represent the boundaries at $x = 0$ and $x = 1$, horizontal walls to represent the boundaries at $y = 0$ and $y = 1$, and the top/bottom wall to represent the boundaries at $z = 0$ and $z = 1$. The common result of all types of collisions is that the particle(s) involved get a new velocity based on the difference in velocity and mass. In order to model a system of particles colliding in a confined area we need to implement boundary conditions, determining how the particles behave when they interact with the walls. For this thesis we have chosen to implement Reflecting Boundary Conditions (RBC) and Periodic Boundary Conditions (PBC), where the former is a way to implement hard walls and the latter is used to neglect boundary effects. With two different boundary conditions we can choose the most suitable one for different applications. For RBC the particles will collide with the walls. For the case of PBC, we will utilize the wall collisions as a measure to implement a periodic system where particles leaving on one side reappear on the opposite side. The area of boundary conditions will be discussed in greater detail later in section 3.4. The implementation and use of force-free collisions gives the project a similar structure as as the simple algorithm presented in [5, pp. 135–189] and in [11, pp. 269–282].

It is possible to conduct simulation of particles in a two-dimensional square box by setting all $z_i$ equal to each other and all $v_{zi}$ equal to zero. A particle will never get a $v_{zi} \neq 0$ as a result of the collision rule in Eqs. (2.10) and (2.11). Thus the particles will be confined to a area with only vertical and horizontal walls as they never gain a velocity towards the top/bottom walls. In this case the particles are represented as hard disks. Two-dimensional simulations will be used for visualization and verification, while the main results are achieved through simulations in three dimensions. The two-dimensional simulations will be referred to as pseudo two-dimensional since they use the code for the three-dimensional simulation, but due to certain simplifications they are correct for two-dimensional systems as well. Three-dimensional systems have been preferred since the derivation of some of the principles given in the theory are known to cause some problems in two dimensions, such as the convergence of the integral of the velocity autocorrelation function [see 11, p. 162]. The use of three-dimensional systems have also been preferred in order to compare results with earlier studies of the MSD in both granular and molecular gases [see 15, 25, 28].

## 3.2 Event driven simulation

An event driven simulation is a systematic approach of letting time move forward until the next event multiple times until a given stopping criterion. A flow chart representation of an event driven simulation is given in Figure 3.1, where an event queue is used to store all future events in the system. We will conduct an event driven simulation of a force-free many-particle system where

[2] `https://github.com/alekgjer/master_thesis`

the particles interact through instantaneous pairwise collisions. We will thus need to create some efficient procedures to perform the different steps in the flow chart in Figure 3.1. The main steps of the simulation can thus be described by the following steps

- Initialization
  - Give each particle parameter values for mass, radius, initial position and initial velocity.
  - Iterate through each particle, calculate if and when it will collide with another object (wall or particle) and store all the collision times.
  - Identify the earliest collision and start the simulation loop.
- Loop
  - Move all particles forward in time with constant velocity until the earliest collision.
  - For the particle(s) involved in the collision, calculate new velocities from the collision rule.
  - For the particle(s) involved in the collision, calculate if and when they will collide with another object and store all the collision times.
  - Identify the new earliest collision.
  - Check if the collision is still valid. A valid collision is a collision which will occur since the involved particles have not been involved in other collisions since the collision was computed. As soon as a particle collides with an object, all of the previously computed collisions for that particle become invalid since the particle will not follow the same trajectory as before the collision. There will at all times exist a high number of invalid collisions since a collision is not removed before the simulation has checked if the collision is valid in this step.
  - If the collision is rejected, discard it and identify the new earliest. Repeat until a valid collision has been found.
  - Repeat the loop until the stopping criterion is reached.

The alternative to an event driven simulation is a time driven simulation. A time driven simulation follows the same main procedure as an event driven simulation, but time is incremented by a fixed timestep value instead of moving time forward between events. In a time driven simulation one must identify the events occurring before the next timestep and handle the events in the correct order. One of the reasons why the event driven simulation has been chosen above the time driven simulation, is the possibility of many collisions in a small amount of time which can make it computationally expensive to reach the end of the timestep[3]. Other situations where a time driven simulation can be disadvantageous are cases where the time until the next collision are much higher than the timestep value, which often is the case for long time scales of inelastic systems. The systematic approach for an event driven simulation is a better fit for the way we have chosen to implement the collisions. By only updating the positions at the time where something has occurred, we reduce the amount of times we have to compute new positions and update the simulation time.

An important thing to note is that by choosing an event driven simulation, one has to add functionality to compute quantities, e.g. the MSD, at times where there are no collisions and with a given resolution in time. There should ideally be no difference between an event driven and a time driven simulation. An additional remark which needs to be made is that the time scale of the simulation will depend on the speed of particles. For an event driven simulation it is thus possible to change the speed of the particles without adjusting some of the other parameters.

[3]Here we mean a time driven simulation where one for instance has decided to simulate for $M$ timesteps. If an inelatic collapse for instance occurs during the last timestep the simulation will not be completed. Whereas for an event driven simulation where we want to simulate until the average number of collisions per particle is equal to a given threshold, that criterion can be reached even if inelastic collapse occurs, as long as the simulation do not break down.

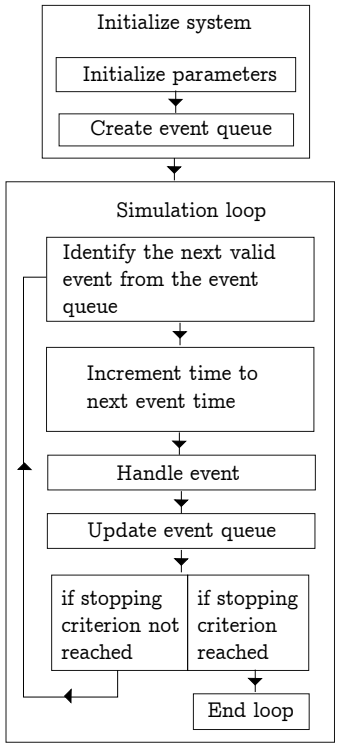



**Figure 3.1:** Simplified illustration of the flow chart used in an event driven simulation where an event queue has been used to store future events.

### 3.2.1 Output

In order to verify the implementation of the simulation we need to compute data from the simulations. We have two possible options depending on the quantity of interest. E.g. if we are interested in the speeds of an equilibrium state to see if we get the Maxwell-Boltzmann distribution from section 2.4, we can compute the speed of the particles after the simulation is done. If we are interested in a quantity as a function of time, e.g. the MSD, we must perform and save an output from the simulation where the quantity of interest is computed. It is also convenient to update certain parameters at each output in order to have a set of simulation statistics which can be helpful to verify that the implemented simulation works in the intended manner. The choice of the output timestep, $\Delta t$, depends on what and how often we want to compute a quantity. We will now focus on the computation of the MSD of a granular gas, and how we inspired by the very different behaviour of the MSD on short and long times have created an alternative to compute a quantity with a constant resolution in time.

We expect to see an MSD scaling logarithmically with time for a granular gas as derived in chapter 2, but we do not want to compute the MSD with a unnecessary high rate as we need to capture the ballistic period. As a natural result of the dissipative collisions the number of collisions per time will decrease as a function of time, for which a high output rate can be more time consuming than the actual event driven simulation for long times. In order to achieve the right asymptotic behaviour of the MSD for a granular gas in (2.80) for both $t \gg \tau_0$ and $t \ll \tau_0$ we would need to choose a relatively small value for the output timestep which are used to compute quantities with a given resolution in time. The following procedure has thus been implemented in order to achieve the correct asymptotic behaviour without computing similar values for the MSD many times. We create a set of values for when we want to compute the MSD, on a logarithmic scale in time. A set of logarithmically spaced times will produce a set of points where the time difference between two times increases as time increases, leading to many points with a high resolution for low times and fewer points for long time scales. This procedure exploits the strength of the event driven simulation where for long times the number of collisions per time decreases, and we would thus get a high number of values for the MSD which would be similar. The choice of logarithmically spaced times makes it possible to present the results with evenly distributed points on a logarithmic plot, which is preferred to display both the short and the long asymptotic behaviour of the MSD in chapter 4.

## 3.3 Priority queue

The approach described for the event driven simulation in section 3.2 is used to calculate all possible collisions and store the collision times. To efficiently make use of all collision times a data structure called a priority queue has been utilized to store the collision information. A priority queue is a structure which can efficiently add elements and return the element with the highest priority in the structure. We can thus store all collisions in a priority queue, where their priority is given by their collision time. For an event driven simulation, we use what is called a min-priority queue, where lower times are given higher priorities. The earliest collision will thus obtain the highest priority, which can easily be retrieved by performing the operation called extract-min. Extract-min consists of extracting and returning the earliest element in the min-priority queue. In the process, extract-min also removes the element from the queue and maintains the min-priority structure in the updated queue. The time scaling of the operations of adding an element to the queue and extract-min are given by the upper limit $\mathcal{O}(\log n_q)$, where $n_q$ is the number of elements in the queue. The main steps of the simulation in an event driven simulation is thus performed by calling extract-min to identify the next valid collision and then adding the new possible collisions to the queue [see 36, pp. 162–164]. For this project, the library heapq[4] in Python has been used as a priority queue.

[4] see `https://docs.python.org/2/library/heapq` for the documentation.

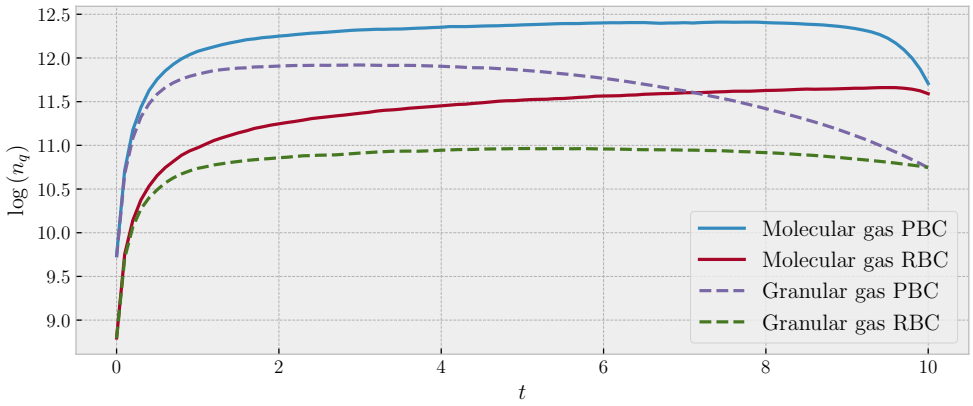



**Figure 3.2:** Plot of the natural logarithm of the number of elements in the priority queue, $n_q$, as a function of time, $t$, for simulations of a molecular gas and a granular gas with $\xi = 0.8$ for the same system of $N = 1000$ particles with a radius $r = 0.025L$. The plot is used as an example to illustrate the scaling of $n_q$ for a molecular gas with PBC (blue line), a molecular gas with RBC (red line), a granular gas with PBC (purple dashed line) and a granular gas with RBC (green dashed line), and the results for the different simulations are given as the average of 4 different runs.

The number of future collisions in the priority queue quickly grows as a function of time as illustrated for a molecular gas and a granular in Figure 3.2. In addition, the number of collisions in the priority queue is naturally a function of the packing fraction of the system. The time scaling of the operations in a priority queue is essential to keep the simulation possible in a reasonable amount of time. This is illustrated for both a molecular gas and a granular gas in Figure 3.2, where the number of possible collisions $n_q \gg N$ at all times and $n_q$ increases as a function of time. In this project all possible collisions are added to the priority queue, even though only the early ones are likely to occur. The later ones are added to account for the situations where the early collisions are not valid due to interference by neighbouring particles. In this manner, the priority queue is often used to discard collisions, but due to the time scaling of extract-min, the priority queue can handle a high number of collisions without becoming too slow.

There are several modifications that can be implemented in order to keep $n_q$ under some control. The most obvious is that if we know how long we want to do a simulation, it is possible to neglect collisions occurring after the maximum time limit, as can be seen as the small drop at the end in Figure 3.2. For a stopping criterion based on a maximum time limit this is a simple and natural improvement. The use of a maximum time limit is especially helpful for simulations of granular gases where particles with low energy use a long time to move across the box and collide with the particles furthest away. We have used a cutoff time given as 1.01 times the maximum time limit. How quickly the number of elements in the queue scale will in addition depend on the implemented boundary conditions, which will be made clear in section 3.4. We also note that from comparing a molecular gas and a granular gas in Figure 3.2 it is clear that the dissipation of energy leads to fewer possible future collisions before the cutoff time.

## 3.4 Boundary conditions

An important topic in the area of numerical simulations, which we have not yet discussed in great detail, is boundary conditions. In order to conduct a simulation of particles colliding in a finite region, given as a three dimensional cubic box, we need to implement a type of boundary conditions

to keep the particles from escaping the box. The choice of boundary conditions will also determine and impact the dynamics of the particles. Boundary conditions often sound like a small detail, but in general it has a huge impact in different applications, e.g. computational fluid dynamics [see 37]. Kinetic theory is often derived without considering boundary conditions, i.e. for large systems [see 11, p. 10]. In order for the simulations to thus capture the correct dynamics we must be certain that the particle data we use coincide with the previous statement. This issue is addressed in more detail in section 3.7. One should however note that this is often the idealization used for theoretical predictions, which numerical simulations cannot match as a result of restrictions occurring in the modelling.

For particles colliding in a three-dimensional box without any external forces we have two natural possible options, RBC and PBC. Both of these options provide some advantages and disadvantages. The illustration of the boundary conditions and the method of applying them is given for a two-dimensional system, whereas they are implemented for a three-dimensional system. They will however work for the pseudo two-dimensional simulation as well, but are not implemented in the most efficient manner. The non-optimized implementation is a result of, as we will later see, the needed number of copies of the system in order to correctly use PBC, which depends on the dimension of the system. In short RBC is implemented as hard walls which the particles will collide with, while for PBC a particle leaving the system of one side will re-enter on the opposite side. The collisions with the walls are added to the priority queue together with the particle-particle collisions in order to get the next valid collision at all times. Resulting from this implementation we add all future collisions for the particle(s) involved in a collision after each valid collision.

Before we go into RBC and PBC with further details we would like to present some of the quantities which will depend greatly on the boundary conditions. These quantities have in common that they are related to the movement of the particles, which will depend on the boundary conditions. An example of such a quantity is the MSD, which we want to use to verify that the numerically obtained MSD of a system of particles follow the theory for both molecular and granular gases as given in chapter 2. Periodic boundaries can give an increasing MSD for all times, while reflecting boundaries will limit the time scale of the computations. The latter is a result of the situations where the red particles in Figure 1.1a start to approach the edges of the system. For such situations the displacement of the particles will not continue to show the same trend as earlier, due to particles bouncing off the walls. The time estimate for these situations to occur is based on the speed of the particles and the particle density, and can not be estimated in a reliable manner for a general system. In practice, we could achieve a time estimate by plotting the MSD and identifying when the displacement stops increasing. One should however note that the real time estimate is earlier. Due to the limiting area, the particles can not escape, making it plausible that reflecting boundaries affect the behaviour of the system even before the particles in the middle have reached the walls.

Another topic which was briefly introduced in the section about Haff's law in section 2.6 is that in a granular system for long time scales clusters will form, for which the system is no longer in the homogeneous cooling state. On these time scales the time evolution of the MSD is not valid due to the granular temperature not following Haff's law. However is it important to note that during this project we have seen that the change from two-dimensional to three-dimensional systems has greatly increased the possible time scale of the simulations. This is again a manifestation of the increase of possible trajectories in three dimensions compared to two dimensions. For similar densities the amount of collisions will be much higher in two dimensions, making it plausible that a two-dimensional system leaves the homogeneous cooling state earlier than a three-dimensional system. The increased collision frequency near the wall from RBC will additionally make clusters appear close to the walls, earlier than what would be most likely the case for PBC [see 5, pp. 169–173]. For the time scale of the simulations in chapter 4 we have observed cluster formation in two dimensions, both for RBC and PBC while for three-dimensional systems with PBC, which is needed for correct computation of the MSD, we have not seen an strong indication that clusters have formed. Cluster formation is one of the topics included as possible further topics for a numerical study of

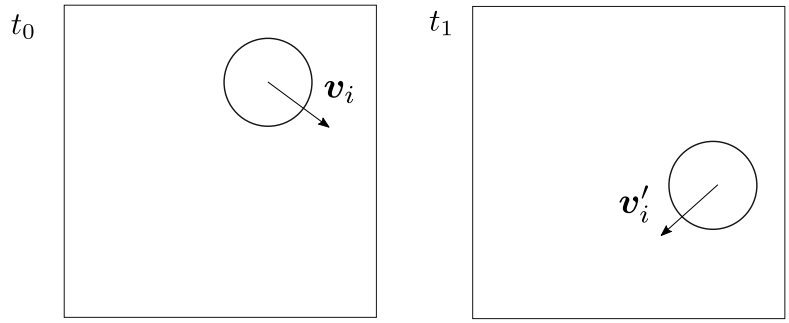



**Figure 3.3:** Illustration of a particle $i$ in a system at two times $t_1 > t_0$ with reflecting boundaries. The figure uses the same notation as Figure 2.2.

granular gas dynamics in chapter 5.

### 3.4.1 Reflecting boundary conditions

The idea of RBC is illustrated in Figure 3.3. For the system of particles in a box RBC can physically be interpreted as hard walls, for which the particles will bounce off. This might be a reasonable approach for simulations where the spread of the particles are limited to a finite region, e.g. looking at air particles in a room. In regards of diffusion properties, the MSD has been shown to vary with system size for PBC in two dimensions [see 24, pp. 141–142]. As a result of such an investigation of the MSD and for simplicity, RBC was originally considered for this project. RBC was utilized for the simulations of a two-dimensional system used in the Specialization Project [1], and was originally utilized for the three-dimensional systems as well. However since we are in this project more interested in the long term behaviour of the MSD the choice of RBC did not provide satisfactory results. This is a natural result given how RBC limits the computations of MSD. The collision rule for a particle colliding with a wall was derived earlier in section 2.1, where as shown in Figure 3.3 the particle obtains a new velocity following the interaction with the boundary.

### 3.4.2 Periodic boundary conditions

As an effort to reduce how RBC affected the computations of the MSD, PBC was implemented. The idea of PBC is illustrated in Figure 3.4, where a particle will re-enter the system on the opposite side of the box when the center of the particle reaches a "wall". Here we have used "wall" to emphasize there does not exist a hard wall, but we still use the walls as a collision partner in order to change the position in the correct manner. Thus the wall collisions are still used in the priority queue as earlier, but the collision is now only used to update the position and does not change the velocity of the particle as indicated in Figure 3.4.

PBC is commonly used to neglect boundary effects by copying the system in all directions and thus making an pseudo-infinite system. Then the dynamics of the system is also copied in all directions, but with different and new positions. We can then find out how a system of particles behaves surrounded by identical systems. If such periodicity is reasonable to assume, PBC often successfully models a system for which boundary effects can be neglected. In reality we still only have a system with boundaries as introduced in the overview. To illustrate PBC we copy the entire system in each direction. Then we have a pseudo-infinite system where the position of particle $i$, $\boldsymbol{x}_i'$, is given by

$$\boldsymbol{x}_i' = \boldsymbol{x}_i + \boldsymbol{e}_x kL + \boldsymbol{e}_y lL + \boldsymbol{e}_z mL \tag{3.1}$$

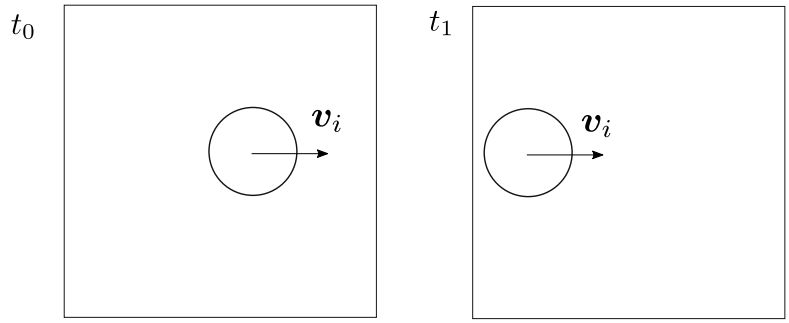



**Figure 3.4:** Illustration of a particle $i$ in a system at two times $t_1 > t_0$ with periodic boundary conditions. When the center of the particle crosses a wall, the particle exits the system and re-enters on the opposite side. For the implementation this counts as a collision, but the velocity of the particle does not change.

where $\boldsymbol{e}_x$, $\boldsymbol{e}_y$ and $\boldsymbol{e}_z$ are the unit vectors along the Cartesian coordinate axes, $\boldsymbol{x}_i$ is the position of the particle in the original system, $k, l, m \in \mathbb{Z}$ and $L$ is the length of the box, equal to unity in the implementation. From a unique set of values for $k, l, m$ we get the offset of the positions of the particles in the copied system. Note that all copies of the particles will have the same velocity as the actual particle. While PBC usually are conceptually easy to understand, it is not always trivial how we can implement them efficiently. The difficulty for us is that we have the original system with boundaries and we do not want to store the data of the copied systems. The latter is a reasonable desire as we can often implement PBC without explicitly using the copied systems for anything other than to explain the idea of PBC.

In order for us to implement PBC we need to remember that we add all future collisions for the particle(s) involved in a collision to the priority queue, which are the next collision with a horizontal, vertical and top/bottom wall and the future collisions with all other particles. Note that after a collision with a "wall", the position of the particle gets updated before we compute future collision. As it turns out we only need to care about the closest neighbouring systems where $k, l, m \in [-1, 0, 1]$ as long as the length of the system is larger than the diameter of the particles. The latter is true as we update the priority queue each time a particle crosses a boundary [see 5, pp. 170–173]. To illustrate why only the closest neighbouring systems are enough let us look at an example. Suppose a three-dimensional equivalent system as given in Figure 3.5, where it turns out that the next particle particle $i$ collides with will be located in the copied system with $k, l, m = [2, 0, 0]$ at the time of the collision and $v_{xi} > 0$. The collision sequence for particle $i$ will then be two collisions with the "wall" given by $x = 1$ before colliding with the other particle. Even though particle $i$ is in the original system, the actual position of particle $i$ is in the correct system. The particle collision are not of interest before they are in the same or neighbouring system as we update the priority queue during the collisions with the "wall". For this simple example we did not actually need to use the copied system for anything. There exists however a type of collisions that occurs due to the periodic boundaries and needs to use the data of the copied systems, that we will discuss after we elaborate how the addition of the system copies affects the numerical implementation.

For a two-dimensional system we will then need 9 copies[5] of the original system as illustrated in Figure 3.5. Even though we only have data for the original system given in bold in Figure 3.5, we need the idea of 8 neighbouring systems to implement PBC correctly. For three-dimensional systems this idea gives 27 copies of the original box, which have been numbered from top left to bottom right as box $0, 1, \ldots, 25, 26$. The box number is then later used to get the correct offset given by unique values for $k, l, m$. Note that from the numbering scheme the original system is then given as box 13.

[5] we will include the original system as a copy since it is included in the implementation with $k, l, m = 0, 0, 0$ due to simplicity

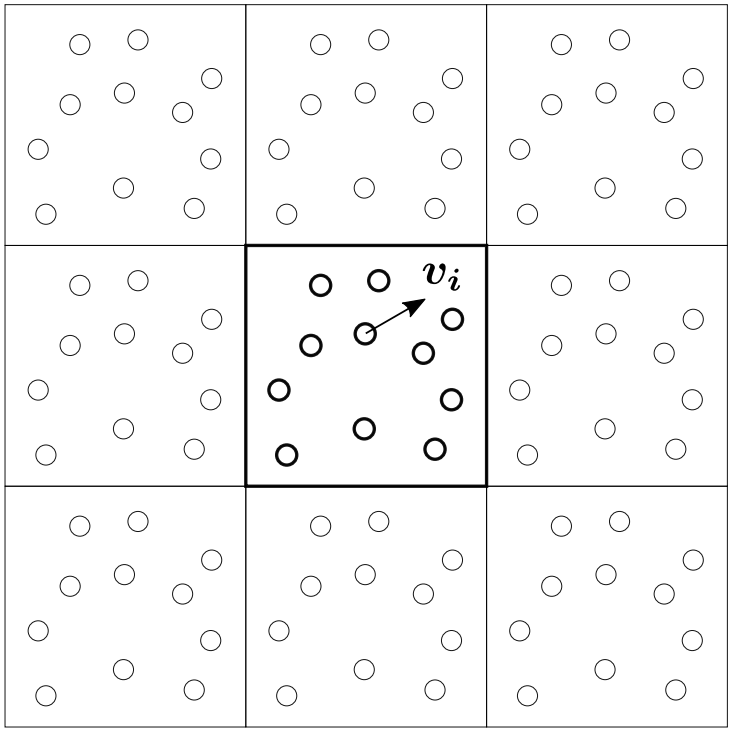



**Figure 3.5:** Method of applying PBC in two dimensions. We take copies of the actual system, presented in bold, and place one copy in each direction. The copies of the particles have different positions, but the same velocities.

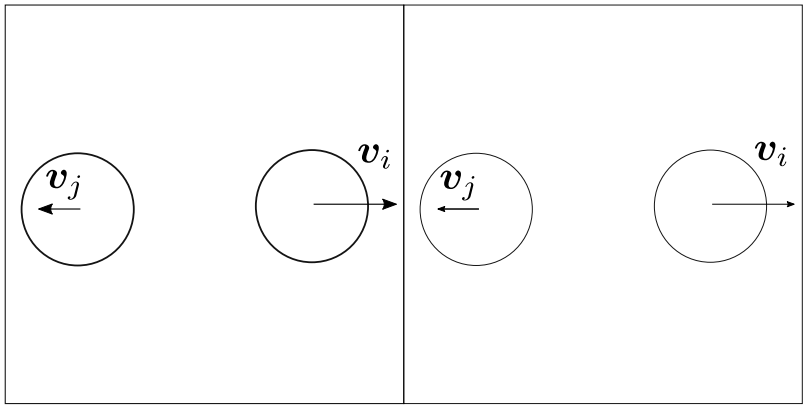



**Figure 3.6:** Illustration of two particles in a system with the copied system to the right. Before the center of particle $i$ crosses the wall the particle $i$ will collide with the copy of particle $j$.

We have modelled that all particles are at all times in the original system with boundaries given by $x = 0, x = 1, y = 0, y = 1, z = 0$ and $z = 1$. Even though it might seem that when a particle exits the system on one side, it re-enters on the other side, it is actually a copy of the particle in a neighbouring system that enters the original system. The particle leaving the system will not actually be confined to the original system any longer. By saving the number of times a particle crosses over to neighbouring systems we will be able to use the actual positions of all particles at later times, and not the positions in the original system. This is essential in getting correct computations of e.g. the MSD, which then can increase for all times. Note that this is a direct result of that the actual positions of the particles are given for the pseudo-infinite system. Another explanation of why we only care about the nearest neighbouring systems can be deduced from the following. When we use the wall collisions as events, we update the priority queue when a particle position is updated. For the new position it is not possible to collide with other neighbouring systems further away before crossing a wall again, which again updates the position and then add new entries to the priority queue. As long as we use the "wall" collision in this manner we only have to use the nearest neighbouring systems.

As illustrated in Figure 3.6, where particle $i$ will collide with the copy of particle $j$ in the neighbouring system to the right, we see a collision between a particle in the original system and a particle in one of the copied systems. If a particle collides with a particle from another box before the center of the particle crosses a "wall", we have to use the offset of the copied system to pretend like the particle in the original system is located in the neighbouring system for $\Delta \boldsymbol{x}_{ij}$ to be correct. This is needed to correctly update the particle velocities, but we do not want to move the particle since it is not necessary. The copies of the original system are only used to compute if the particles in box 13 will collide with a copy before reaching the "walls". Their data are not stored since we are only simulating $N$ particles and not $27N$ particles[6]. The collision in Figure 3.6 must then be correctly computed based on the data of particle $i$ and particle $j$. We must then use the offset of the copied system to the right to create a temporary position for the copy of particle $j$. We then resolve the collision and update the velocity of both particles. Thus the particles will get the correct new velocities without changing the position of any of them. With this implementation we can keep all particles within the boundaries of the original system at all times, while using the number of crossings to get the correct actual positions of all particles.

[6]This concept would be self-increasing and impossible to implement. If we wanted to look at $27N$ particles we would need to include the copies of each of those systems that would give even more particles and so on. One option is that we could use the 27 copies to simulate and only use the results of the particles in the original system, which is equivalent to the idea proposed for RBC by only using the data of the red particles in Figure 1.1a.

An important note to make is that the number of times we achieve a collision between particles in different boxes depends on the packing fraction. For low packing fractions we have noticed in the simulations that PBC are mostly used with the addition of collisions with "walls". For higher densities the case of collisions between particles in different boxes occur more frequently, especially in a granular gas where we can get clusters colliding from two separate sides of the system.

The main difference of PBC compared to RBC can be summarized with the following points.

- The use of "walls" to update the position by adding or subtracting the length of the box along the correct axis instead of bouncing off them.
- We need to know when the center of the particle cross the "walls" and not when the edge of the particles is in contact with the walls in order to compute the time of the interaction.
- We need to take 27 copies of the system and add possible collisions with particles from the neighbouring systems for the original particles in addition to the collisions between original particles. As a result we will get several more entries in the priority queue, and we must handle the collision between particles in different boxes as illustrated in Figure 3.6.
- A collision with a "wall" does not lead to a new velocity. Thus the particles can only get a new velocity and a new speed from a collision with another particle.
- We must save the number of crossings to get correct particle positions in the pseudo-infinite system.

## 3.5 Collisions

In order to simulate particles colliding in a three-dimensional box, some assumptions have been made. The possibility of simultaneous collisions between more than two objects has been ignored and collisions are assumed to happen instantaneously. As a result of these assumptions one can study a many-particle system of particles colliding in a box using the event driven simulation in section 3.2. In order to do that we need to choose a way to represent a collision numerically, as there are many properties to consider for a collision.

Intuitively a collision requires two properties, the time until the collision and how the velocities of the colliding objects change as a result of the collision. The equations stating how the velocities of colliding particles are updated are given in the collision rule in Eqs. (2.10) and (2.11). The time until a particle collides with another particle is given in (2.20), and the time until a particle collides with a vertical, horizontal and top/bottom wall is given in (2.21), (2.23) and (2.25) respectively. As we have modelled the walls as particles with infinite mass the collision rule for the collisions between the walls and particles are a simplified version of the general collision rule given in Eqs. (2.22), (2.24) and (2.26). From these equations we can compute the information needed in order to add the collisions to the priority queue.

In this thesis the following way to represent a collision numerically as a tuple has been used

$$(\text{collision time}, \text{colliding objects}, \text{collision count colliding objects}, \text{box colliding objects}), \tag{3.2}$$

where colliding objects $\in$ [particle number, 'hw', 'vw', 'tbw'], where hw, vw and tbw is short for horizontal wall, vertical wall and top/bottom wall respectively, and the collision count is the number of collisions each particle has been involved in. By implementing a collision as a tuple one can utilize the properties of a priority queue based on the collision time, while using the additional information to compute new velocities for the involved particles and to validate the collision. The box of the colliding objects are used to handle PBC in situations as illustrated in Figure 3.6 as discussed earlier. The collision is validated by comparing the collision count at the event time with the collision count at the event computation. If either particle has collided with something else since the collision was

computed, the collision is non-valid, which is easily identified from the collision count. For this project we have chosen the validation process to discard non-valid collisions instead of deleting them from the queue at the moment when they are not valid due to the particle being part of a valid collision. Even though one easily can access the element with the highest priority, deleting random elements from a priority queue is not possible to do in an equally elegant manner. The slow increase in run time for larger input to logarithmic operations can for some cases be preferred over adding non-trivial functionality. One should note however that there exist other data structures which could efficiently remove objects, which should be considered for such an approach. This topic is also discussed as one possible improvement to the implemented event driven simulation in chapter 5.

Now we want to discuss how the collision rules and the time until the collisions differ from RBC to PBC for the interactions with the wall, and how we have implemented the particle-particle collisions efficiently, as the time until a particle collides with another particle is the most time consuming part of the event driven simulation. We emphasize that the simplified use of the computation of the collision times and the collision rule is possible as the objects are not deformed during a collision, and objects bounce off each other instantaneously at the moment of impact.

For simulations using PBC we will use the boundaries as "walls", but they are still counted as collisions in the simulations, even though the velocity of the particle crossing a boundary does not change as illustrated in Figure 3.4. This is necessary in the implementation in order to validate future collisions. For PBC we need to know when the center crosses a "wall" and not when the edge of the particle is in contact with a wall. This can however simply be computed from the already known results in Eqs. (2.21), (2.23) and (2.25) by inserting $r_i = 0$. When the particle center crosses the wall the position along the correct axis is adjusted by adding or subtracting the $L = 1$ along the correct axis. Here have used the correct axis to denote the axis of the boundary that the particle interacts with. E.g., when the center of the particle crosses the vertical wall given by $x = 1$ we subtract $L = 1$ from the $x$-position of the particle. In addition, as we do not update the velocities of the particle, and update the number of crossings we can use PBC to simulate pseudo-infinite systems.

As discussed in section 3.4, during this project we deemed that PBC is better suited for this project than RBC in the study of the MSD. We have however used RBC for some visualization purposes and to compute speed distributions. For the boundaries to affect the system as little as possible we have used elastic walls in the implementation of RBC, where $\xi = 1$ in the collision rule with the wall regardless of the $\xi$ used in the system. This is the same treatment of walls as used in [5, pp. 151–152]. The effect of different boundary conditions and why we have chosen elastic walls is clear from the results for the verification of Haff's law in chapter 4. However the expressions for the collision rule between a particle and a wall have been used to experiment with both inelastic walls, elastic walls and even heated walls which was modelled by $\xi > 1$.

Determining which particles will collide and when they will collide is the most time consuming process in the event driven simulation. This is a result of the many possible collision partners of each particle, and the fact that we must compute when and with whom the particles involved in a collision will collide with again whenever a collision has occurred between two particles. Numerically this is done by computing (2.20). See Appendix D for the code used and an explanation of how this have been solved by using a variety of elegant optimizations possible to use in Python.

## 3.6 TC model

Inelastic collapse can be avoided by introducing the so-called TC model, where one introduces a "duration of contact" for a collision, $t_c$. The model changes the system, such that if a particle is involved in another collision during the duration of contact from the previous collision, the new collision occurs as if $\xi = 1$ [see 18]. Setting the duration of contact $t_c = 0$ is equivalent to not using the TC model. The value of $t_c$ must be set by looking the average time between collisions

and choosing a lower value to give a system where we rarely have the situation where two collisions almost occur at the same time. The TC model can be necessary in a inelastic system due to fact that particles trapped in a small region will be part of many collisions in a small amount of time while dissipating energy. For RBC, the TC model is even more relevant than for PBC due to situations where particles are pushed towards and away from the wall at the same time. These situations occur naturally as for hard reflecting walls the particles will bounce off the wall. If there are particles moving towards the same wall behind the particles that should move away from the wall one-dimensional chains of particles can start forming, giving us a similar situation as in Figure 2.6, but with a wall as the particle to the right. Applying the TC model offers no change for a molecular gas since in that case $\xi = 1$ to begin with.

For two-dimensional simulations with RBC we found that values for $t_c \leq 10^{-3}$ seemed to stop inelastic collapse from occurring. In a three-dimensional system with either boundary conditions we have not experienced any problems which could indicate that inelastic collapse occurs for all tested values for $\xi \in [0.3, 1]$. Due to no inelastic collapse, the TC model has not been used for the results. But due to originally simulating two-dimensional systems and the pseudo two-dimensional simulations we have chosen to include it in the report.

One interesting thing to note is how inelastic collapse manifests itself in a simulation. As introduced in chapter 2 inelastic collapse consist of an infinite number of collisions in a finite time. The inelastic collapse breaks down the simulation due to the priority queue, where there are collisions so close together that the $<$ operation breaks down and it is not possible to identify the earliest collision. Here we experience some of the disadvantages of using the tuple in (3.2) to represent a collision. The default behaviour of heapq in Python is to push new collisions to the priority queue. When the collision occur almost at the same time[7], the behaviour of heapq is to assign a priority for the new collision based on the next quantity in the tuple. I.e. we assign a priority for the new element based on the particle index, the collision count or the box of the colliding objects, which sometimes gives us a complete breakdown when e.g. comparing a particle index with 'hw', 'vw' or 'tbw'. At this point we cannot guarantee that the simulation performs correctly as it is random which collision is considered to be the earliest. Another issue is that we update the particle data by using that the particle have a constant velocity between collisions. When the time difference between updates is very small we most likely do not update the particle data correctly and the simulation cannot be used to compute quantities in a reliable manner.

## 3.7 Statistics

In order to best achieve general trends for the results in chapter 4, the results have been averaged over multiple realizations of a system whenever possible. A realization is here used as a unique simulation of a system. A given system will develop differently for different initial conditions and we will thus use different initial conditions to create unique systems. The number of realizations will thus also be referred to as the number of runs used in the simulations. The use of averaged values is essential in trying to get an accurate image of how different quantities, e.g. speeds, energies and MSD evolve in time for a given system. This is also the case for equilibrium systems since they do not always give the exact equilibrium properties, but the perturbations disappear when we take the average over multiple different runs. Mean results, which are given as the development of an ensemble of particles, denoted by $\langle \dots \rangle$, can in general need multiple runs in order to show the trends proposed by theory.

For the case of a Brownian particle, we need to simulate the system and compute the MSD for the Brownian particle several times in order to achieve the results proposed by theory. This can be a

[7] Almost at the same time is used here for the situations where the difference in time for two collisions is less or equal to the machine epsilon. Machine epsilon for float64, the default number representation used by NumPy arrays, is $\sim 10^{-16}$.

computationally heavy task, knowing that one often could need a high number of realizations before the mean behaviour converges towards the theoretical predictions. However, as we have only studied Brownian motion in systems of identical particles, we have exploited that the desired behaviour are present for several particles[8]. We can thus compute the mean value over a set of particles, also known as an ensemble average, which reduces the amount of runs needed for the results to converge. We can compute the average of different ensemble averages in order to see if the results have converged based on the central limit theorem given in section 2.11, where we assume that each ensemble average are given from the same distribution. As PBC are used to simulate a pseudo-infinte system we can use the data of all the particles, whereas if we used RBC for the MSD we could not use the data of the particles starting close to the boundaries. For the results in chapter 4 we have used RBC to compute speed distributions, as the speed of the particles are not affected by the size of the system, where we can use the data of all the particles as well. As a result we have in the results used the ensemble average of all the particles in the system.

To better illustrate the need to use multiple runs let us look at the case when we compute the MSD. The idea described above is needed due the very different trajectories obtained for the different particles in a systems. All particles in the system will obtain a unique trajectory, giving a unique square displacement based on their initial position. We then compute the MSD as the ensemble average of the square displacements of all the particles. The MSD computed here should ideally be equal to the theoretical predictions, as would likely be the case for a system of infinite particles. As a result of the finite number of particles used in a simulation a perfect match with the theory is unlikely. We thus have two options. The first one is to use a high number of particles, restricted only by memory and the time available for the simulation. The second option, which we have utilized for this thesis, is to take the average of several simulations where we have used a reasonable number of particles. For the latter option how many runs and the number of particles are usually linked strongly in order to obtain satisfactory results, and is a result of trial and error. We thus, instead of computing the MSD of one system of particles, compute the average MSD from different runs. The results can then be presented as the mean MSD with an error estimate based on the central limit theorem (presented in section 2.11), where we assume that all computed MSD in the same point in time are drawn from the same distribution.

For the results in [1] we performed numerical studies of the MSD for molecular gas for both a Brownian particle equal to and different from the other particles. For the case of equal particles we used the average of 50 runs, whereas we used the average of $300-400$ runs for the case of a Brownian particle different from the other particles, modelled by a bigger radius and/or mass. Even though we used many runs for the case of a different Brownian particle, the MSD for long times had not converged completely. Note that as we used two-dimensional simulations with RBC we could only use the data of the particles starting close to the center of the box for the case of equal particles. For the results presented in this thesis we have used the data of all the particles in the system, reducing the number of runs needed for the average value to converge based on the standard deviation of the central limit theorem.

## 3.8 Initial values

In order to achieve reliable results we have to choose suitable initial conditions. The mass and speed of the particles can be chosen freely, but will determine the energy and the time scale of the system. The initial positions have been chosen to be uniformly distributed in the box, while upholding the criterion that no particle can be overlapping with another particle or with any of the boundaries. The initial velocity vector for particle $i$ has been chosen to be

$$\boldsymbol{v_i} = [v_0 \sin\theta\cos\phi, v_0 \sin\theta \sin\phi, v_0 \cos\theta], \tag{3.3}$$

[8] As all particles are equal we imagine that all particles are a Brownian particle of interest, and we thus use the data of all the particles when looking at the behaviour of e.g. the MSD.

where $v_0$ is the initial speed, $\phi$ is a random uniformly distributed angle $\in [0, 2\pi]$ and $\theta$ is another random uniformly distributed angle $\in [0, \pi]$. The procedure in (3.3) generates a Cartesian velocity vector from spherical coordinates with a given length and random angles. For generating initial values for a two-dimensional system we have $\theta = \pi/2$, which generates Cartesian vectors from polar coordinates. The statistics have been achieved by using the same initial positions and the same initial speed, but new velocities during each realization of a system, i.e. new values of $\phi$ and $\theta$ for each particle. We have used $v_0 = \sqrt{2}$ such that a system of particles with masses equal to unity gets the convenient initial average particle energy equal to unity. Note that an initial average particle energy equal to unity gives an initial temperature $T_0 = 2/3$ from the relation in (2.35).

Another approach commonly used in numerical studies of a granular gas is to start from an equilibrium state of a molecular gas where the speed distribution follows the Maxwell-Boltzmann speed distribution given in section 2.4 [see 8, 13, 19]. In order to achieve such an equilibrium state we have simulated a molecular gas, starting with uniformly distributed positions and where the initial velocity vectors are given by (3.3), until the system has reached an equilibrium state. We have then saved the velocity of all the particles in order to use these as the initial velocity vectors when studying the properties of a granular gas. The velocity vectors in the equilibrium state are then shuffled between different runs in order for each realization to be different, where each particle gets a new velocity vector for a new run. The equilibrium state was achieved by simulating a system with RBC until the average number of collisions per particle was equal to $0.2N$, where $N$ is the number of particles in the system, from a set of initial velocity vectors given by (3.3) with $v_0 = \sqrt{2}$. The use of different stopping criteria are discussed in more detail in section 3.9.

The radius of the particles will determine how many particles can fit inside the box without giving overlapping positions. Together with the number of particles, $N$, the radius will determine the packing fraction of the system. For the simulations in this project, $\eta \approx 0.065$ has been used to compare the results with the results in [15]. An important note to make is that for a two-dimensional system the packing fraction is not given by (2.44), but by $\eta = \frac{N}{A}\pi r^2$, where $A = L^2$ is the area of the system. The same packing fraction for a two-dimensional and a three-dimensional system gives very different dynamics due to the increase in possible trajectories for three-dimensional systems, leading to fewer collisions. I.e. the same packing fraction is not comparable for systems of different dimensionality.

Another important thing to note is that for PBC, if we are not careful, we can get dynamics which are dominated by the fact the the whole system moves with a speed in a certain direction. When the whole systems moves we get a contribution to the MSD which are dominating the logarithmic dependence of time for the MSD of a granular gas[9]. This can immediately be noticed by doing simulations in order to compute the MSD. In order for the simulations to be correct, we need to use a reference frame where the displacement is only a result of the interactions between particles. To achieve said reference frame we adjust all particle velocities by subtracting the mean velocity along each axis. This subtle change plays a major role for computations of certain quantities such as the average particle energy and the MSD. The change is actually so small that we still have the case where the initial average particle energy is approximately equal to unity. For the equilibrium state we have used for the results of a granular gas in chapter 4 the difference between unity and the initial average particle energy is $\approx 0.0004$.

## 3.9 Stopping criterion

There are several stopping criteria which are suitable for an event driven simulation. We have already explained one used to create an equilibrium state by doing the event driven simulation

[9]This was a real cause of headache during the initial simulations after the change to PBC. We were not sure whether the PBC was not implemented correctly or if something else was affecting the results. The correction of this behaviour by subtracting the mean velocity along each axis was the idea of Tor Nordam. I can only imagine and be terrified of how much time I could have spent wondering about this problem without his help.

until the average number of collisions per particle is equal to some given limit. A simpler stopping criterion is a limit based on the simulation time, $t$. For a given problem we can thus do the simulation loop as long as the simulation time $t < t_{stop}$. For some types of simulation, for instance looking at equilibrium properties, a better criterion should be implemented in order to make sure that the system has reached equilibrium. One of the strongest arguments for such an implementation is that one often needs a reasonable amount of collisions before the system reaches an equilibrium state. The time, $t$, for this to occur will vary depending on the packing fraction and the speed of the particles. Letting the system evolve until a given number of collisions has occurred is independent of the time scale of the system. Even though only particle-particle collisions help the system approach an equilibrium system, we have not differentiated between particle-particle and particle-wall collisions when counting the number of collisions for simplicity. For studies of such problems the proposed solution has been implemented, which is to let the system evolve in time until the average number of collisions per particle is above a given threshold. The average number of collisions per particle, $\bar{c}$, can be defined as

$$\bar{c} = \frac{1}{N}\sum_{i=1}^{N} c_i, \tag{3.4}$$

where $c_i$ is the number of collisions particle $i$ has been involved in. Since we are already tracking $c_i$ is order to validate collisions, we can easily use the collision count to stop the simulation. Thus, the simulation is run as long as $\bar{c} < \bar{c}_{stop}$, where $\bar{c}_{stop}$ have been chosen from trial and error. For this project, $\bar{c}_{stop}$ has been computed as a function of $N$ with success for some problems. For other problems, like computing statistics, the simulation has been run until $t = t_{stop}$. For cases where one is looking at a system with $\xi < 1$, we could also run a simulation until the average kinetic energy of the particles in the system is below a given threshold, $\langle E \rangle_{stop}$. The stopping criterion used in this project has usually been a result of trial and error, or how long we wanted to run the simulations. For most results we have used a stopping criterion given by $t_{stop}$ in order to neglect collisions occurring after the time limit in the priority queue, which makes a huge difference when looking at the MSD at long time scales. In order to compute speed distributions we have used a stopping criterion based on the average number of collisions per particle.

## 3.10 Parallelization

An important topic for application of numerical techniques on modern computers is parallelization. For this project the library Joblib[10] has been utilized with success to do realizations in parallel. It is also possible to do each realization in parallel by splitting the system into more boxes and only considering neighbouring boxes when adding new collisions to the priority queue [see 38]. The idea of doing a realization in parallel has not been explored in detail, but are listed as one topic of possible further work in chapter 5. Due to the possibility of using High Performance Computing (HPC), doing realizations in parallel is an implementation where we use one of the strengths of HPC and modern computers in general, namely the number of available cores. Using Joblib we have implemented a parallelization scheme where we run each realization with a unique run number. The run number is used when saving the results such that the different cores do not overwrite each other. Since each realization is unique there is no shared memory, which must be argued to be the simplest form of parallel computing. In the beginning of each run we shuffle the velocity vectors from an equilibrium state in order for each realization to be unique. With this implementation it is possible to use all the cores on a processor to do their own realization, which comes in handy both when running local simulations and especially when using HPC. In the implementation we specify the number of runs, $n_r$, and the number of cores, $n_c$, to use when running an event driven simulation. Then $n_c$ realizations are done in parallel until $n_r$ realizations have been simulated.

[10] see `https://joblib.readthedocs.io/en/latest/` for the documentation

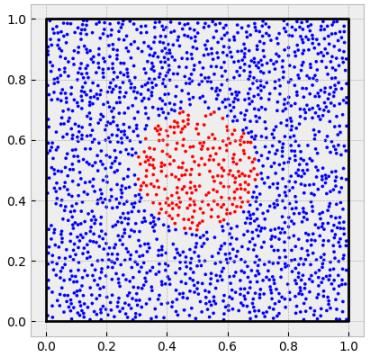



**Figure 3.7:** System consisting of $N = 2000$ particles with equal radius $r = 0.005L$. This is an illustration of an initial condition for the pseudo two-dimensional simulations used in this project where the two-dimensional packing fraction $n\pi r^2 \approx 0.16$. The mass of the particles does not need to be equal, making it possible to look at mixtures of different gases, or if the masses are equal, look at how properties of a single gas evolve in time.

## 3.11 Numerical setup and errors

The main objective of this thesis is to use two different simulation methods to study granular gases. With this in mind we should discuss some of the numerical properties of the implementation, namely the setup and the possible errors. As one might expect there is a high number of variables and parameters which are used for both methods. In order to understand the results in chapter 4 an elaborate introduction to the setup is not needed, as the results should in principle be independent of how we implemented the model. For some purposes it is however useful to be introduced to the implementation if some of the work is relevant for the reader, either as inspiration or simply to provide more details of the numerical modelling. For the interested reader we suggest to read Appendix E, where we present all the variables and parameters used both for the event driven simulation and to solve SDEs describing Brownian motion numerially using the Euler-Maruyama method.

We will now start by looking at the numerical setup and errors of the event driven simulation before moving on to the numerical solution of SDEs.

### 3.11.1 Event driven simulation

In order to simulate a system of particles with an event driven simulation we study a particle setup as given in Figure 3.7. Particles act in a collision as expressed by the collisions rule in (2.10) and (2.11). The time until a collision with another object is computed from (2.21), (2.23) (2.25), or (2.20). Thus a simulation is performed by computing and updating quantities given by the equations listed above while using the priority queue to make sure that we only perform valid collisions, and in the correct order. Figure 3.7 illustrates a typical initial system where the positions are uniformly distributed in the box. Since we have looked at the special case of Brownian motion in three dimensions where all particles are equal we have used three-dimensional equivalent systems to the two-dimensional system illustrated in Figure 3.7.

The model considered for the event driven simulation in this project is very simple, and all the equations can in principle be solved exactly. The errors in this thesis will mainly be due to number representation. This can be a factor for the priority queue, especially during inelastic collapse. Since there can be an arbitrary small time between collisions one can end up in situations where round-

off error decides which collision is considered to be the earliest. In addition, the energy will not disappear completely, as even for perfectly inelastic collisions the particles end up with a non-zero speed, which gives future collisions after a long time.

In addition to purely numerical errors, the model is of course only an approximation of reality, as it considers binary, instantaneous collisions and only a finite number of particles. For many particles we can end up in situations where memory can become a problem, both because of the size of the arrays and the number of elements in the priority queue together with a low value for the output timestep when computing quantities, such as the MSD. This may give some limitations of using the introduced parallelization scheme if each of the cores we want to use need to use a lot of memory.

### 3.11.2 Numerical solution of SDEs

For the numerical solution to SDEs describing Brownian motion one should ideally discuss the error and the convergence of the numerical solution to the exact solution. This is however considered to be outside the scope of this thesis, as we are more interested in the application of the Euler-Maruyama method instead of giving justification of why it works. We will only briefly mention that the numerical solutions of the SDEs converge to the true solutions as $\Delta t \to 0$, and that the ensemble average converges to the true average, almost surely, as $N \to \infty$. For the interested reader, we suggest to see e.g. [31] and [34] for a introduction to various topics of SDEs and the properties of numerical solutions to SDEs.

As for the event driven simulations, there is a tradeoff between computational effort and statistically sound results also when modelling Brownian motion by an SDE. As we want to solve the iterative schemes for a system for $N$ particles at the same time using vector operations, for a three-dimensional system with a constant resolution in time given by timestep $\Delta t$, one can end up in a situation where memory becomes an issue. Memory issues are most prominent by combining a high $N$ with a low timestep value. The results provided in this thesis are at the limit of what we were able to compute locally with 16 GB of RAM. One workaround for this issue is to lower $N$ and compute the average over more runs. One should also note that one does need to save the particle positions at all times in order to compute the MSD, as we have done. The reason for such an implementation is that we also want to show whether or not the particles we try to model with an SDE is an ergodic system, as discussed in section 2.10.

## 3.12 Specifications

The results have been computed in Python with the following specifications. The computer used to run the program has a Intel(R) Core(TM) i7-2600 CPU @ 3.40 GHz. The software used was Python 3.7.5, NumPy 1.17.4, SciPy 1.3.2 and Joblib 0.14.1. The code has also been tested on one of the HPC clusters available at NTNU, called Idun, with the newest available software. The results from running on HPC matched, as expected, the results achieved from running on a local computer. With the higher number of cores available on Idun, we did achieve to run more realizations in parallel at the same time. When running locally we have used 4 cores, but this is only limited by the number of available cores.

# Chapter 4

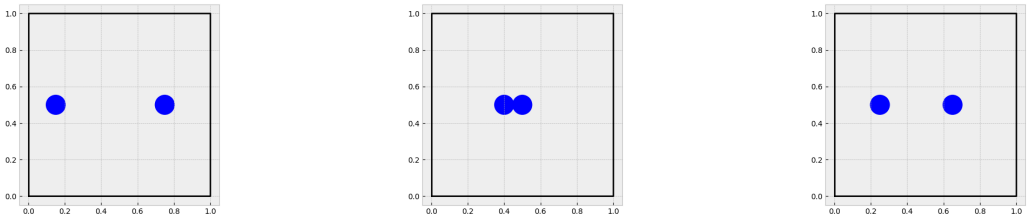
# Results and discussion

The computed results in this thesis have been used to verify the implementation of two different simulation methods used to study many-particle systems, namely an event driven simulation of particles colliding in a box and solving different Langevin equations. As introduced in chapter 3 and Appendix E, there are many variables and parameters for a system of particles needed to perform an event driven simulation, given in Table E.1. Thus, for each system only the essential variables will be stated, such as the number of particles and radius, giving the packing fraction of the system. If not specified otherwise all particles in the system have the same mass $m = 1$, the same radius $r$, the initial positions are uniformly distributed in the box, all initial velocities have the same speed $v_0$ and $t_c = 0$, indicating that the TC model has not been used for the results. For a granular gas an equilibrium state of a molecular gas has been used as the initial values for the velocities of the particles, while for a molecular gas initial velocity vectors given by (3.3) have been used. In order to apply the Euler-Maruyama method to solve SDEs numerically, the variables and parameters given in Table E.2 have been used. As one does not need to take into account the other particles when solving an SDE for $N$ particles simultaneously, all particles will start with the same initial position and the velocities of the particles have been given by (3.3) as for a molecular gas. All results are computed for a three-dimensional system, while some visualizations for a two-dimensional system are provided additionally. The computed results are given as the average of a number of different unique runs, achieved by either computing new initial velocity vectors from (3.3) for a molecular gas, or by shuffling the set of initial velocity vectors for the simulations of a granular gas for each run as discussed in section 3.8. All plots given for a quantity as a function of time has used a stopping criterion given as the maximum time provided in the plot.

During the comparison of the numerical results with the theoretical predictions in chapter 2 the given theoretical expression for $\tau_0$ in (2.45), $D_0$ in (2.52), and $\gamma_0$ in (2.55) will be used. In order to identify some of the reasons why the results differ from theory we have also tried to estimate said parameters from the simulation results by fitting a curve to the datapoints for a given function with unknown parameters. As many of the expressions for the MSD depend in a number of ways on different constants it can be quite challenging to identify possible explanations as to why the results differ from theory.

## 4.1 Event driven simulation of many-particle systems

We will first look at the results for the event driven simulation. The results will start with some simple test cases, before moving on to speed distributions, the evolution of granular temperature and computation of the MSD for both a molecular and a granular gas. We intend to compare the computed speed distribution with the Maxwell-Boltzmann speed distribution in section 2.4, the evolution of granular temperature with Haff's law in section 2.6 and the MSD with the theoretical



**(a)** $t_0 = 0$ **(b)** $t_1 > t_0$ **(c)** $t_2 > t_1$

**Figure 4.1:** Illustration of one of the simple tests used to verify the event driven simulation. Here one can see two particles starting at time $t_0 = 0$, colliding at $t_1 > t_0$ and moving away from each other at $t_2 > t_1$. The illustrated behaviour is as expected, thus making it a useful verification tool. As time is evolved further, the particles will collide again after bouncing off the two vertical walls for RBC.

predictions derived in section 2.8. We have mainly used PBC for all simulations, but we will state which boundary conditions has been used for some of the topics in order to see how the boundary conditions influence the results.

### 4.1.1 Test cases

#### Visual inspection

One of the first verifications of the event driven simulation can be done by looking at how the positions of the particles evolve in time. If they behave as expected, the correct logic has most likely been implemented when computing collision times and using the priority queue to identify the next valid collision. We implemented a few different test cases, e.g. looking at how a particle bounces off each wall before returning to its original position for RBC, how two particles collide with each other when starting on the same line with velocities leading to a head on collision, illustrated in Figure 4.1, and making sure that energy is conserved for $\xi = 1$. Note that when particles obey the law of reflection, a single particle in a rectangular box needs at most four collisions in order to return to its original trajectory in two dimensions. The collision in Figure 4.1 illustrates a few properties which have been introduced earlier. At the moment of collision, illustrated in Figure 4.1b, $R_{12} = r_1 + r_2$. The change in velocities for the particles are only along $\Delta \boldsymbol{x}_{12}$, which in Figure 4.1 is along the $x$-axis, in agreement with the collision rule in Eqs. (2.10) and (2.11).

As the number of particles increases it becomes difficult to rely only on visual inspection since there are many collisions. In order to verify the event driven simulation for many-particle systems one must thus include different simulation statistics in order to get a more systematic approach of studying the dynamics of the system. We would like to note that even though visual inspection quickly becomes redundant in order to make sure the simulation are correct, one can quickly notice incorrect behaviour. The visual inspection was key in the implementation of e.g. PBC.

#### Simulation statistics

The conservation of energy is a useful verification tool for the case of many particles in a molecular gas. The deviation from unity for the average particle energy of a molecular gas ($\xi = 1$) for a simulation of a system of $N = 1000$ particles with radius $r = 0.025L$, giving a packing fraction of $\eta \approx 0.065$, is given in Figure 4.2. The particles in the simulation started from an equilibrium state of a molecular gas. As we can see in Figure 4.2, energy is conserved almost down to machine precision, constituting one of the many necessary verifications of the implementation. The development of the energy for a granular gas will be addressed later in the section about the verification of Haff's law.

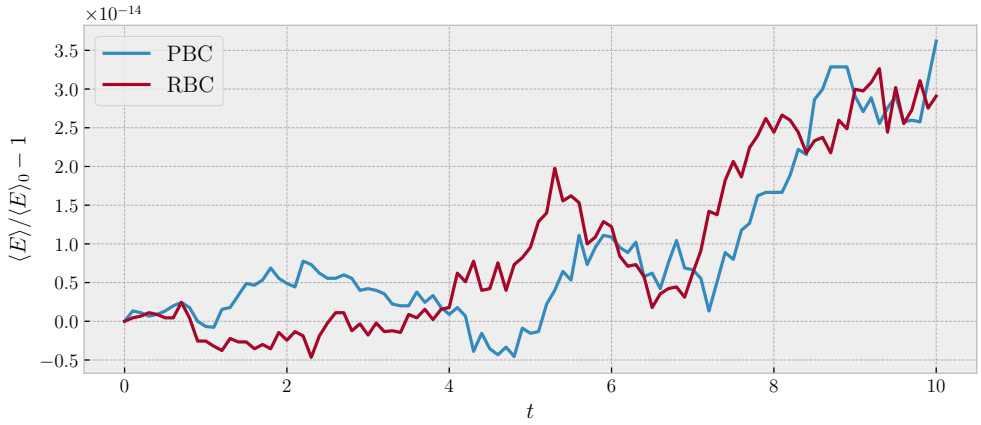



**Figure 4.2:** Plot of the average kinetic particle energy for a simulation of a molecular gas, where $\xi = 1$, as a function of time. The simulation has been done for a system of $N = 1000$ particles with a radius $r = 0.025L$, giving a packing fraction of $\eta \approx 0.065$, starting from a equilibrium state for a molecular gas for both PBC (blue line), and RBC (red line). At all times, we can see from the plot that energy is conserved almost down to machine precision for as the deviation from unity in the plot is $\sim 10^{-14}$. The plot is given as the average of 4 runs with $t_{stop} = 10$ as the stopping criterion. The output timestep $\Delta t = 0.1$ has been used to have a constant resolution in the output of the statistics.

The collision rule is however only correct if there are collisions occurring in the system. The average number of collisions per particle as a function of time for a similar system as in Figure 4.2 is given for a molecular gas in Figure 4.3 and for a granular gas with $\xi = 0.8$ in Figure 4.4. From the statistics of the average number of collisions per particle we can see some interesting differences between a molecular and a granular gas and between the different boundary conditions. As discussed earlier we have suggested based on the experience of others that RBC leads to a higher collision frequency than PBC, which was also the case for the simulations in Figure 4.3 and Figure 4.4. Note that for both boundary conditions we include the pairwise particle collisions and the interactions with the walls, where for RBC we have collisions with the walls, while for PBC we use "walls" to get a periodic system copied in all directions (see section 3.4).

As energy is dissipated for a granular gas, the increase in the average number of collisions per particle slows down as a function of time as we can see in Figure 4.4. Due to a constant coefficient of restitution the system loses energy at every collision, and it is thus natural to see a decrease in the number of collisions per time compared to the case of a molecular gas where energy is not lost. As a result of the decrease in the number of collisions, where the time between two valid collision increases as a function of time, simulations of a granular gas for long times are efficiently performed by the event driven simulation. We can also see from the average number of collisions per particle for a molecular gas in Figure 4.3 that the dynamics of the system seem to be similar for all times, which is expected as energy is conserved and we expect the system to converge towards an equilibrium state given from the Maxwell-Boltzmann speed distribution.

The simple test cases have provided some simple verifications of the event driven simulation, and now we want to see if we can reproduce some known results for both a molecular and a granular gas presented in chapter 2. We will start by looking at the speed distribution of a molecular gas, before studying the evolution of granular temperature for different coefficients of restitution, and then finally studying Brownian motion and the development of the MSD as a function of time for both molecular and granular gases.

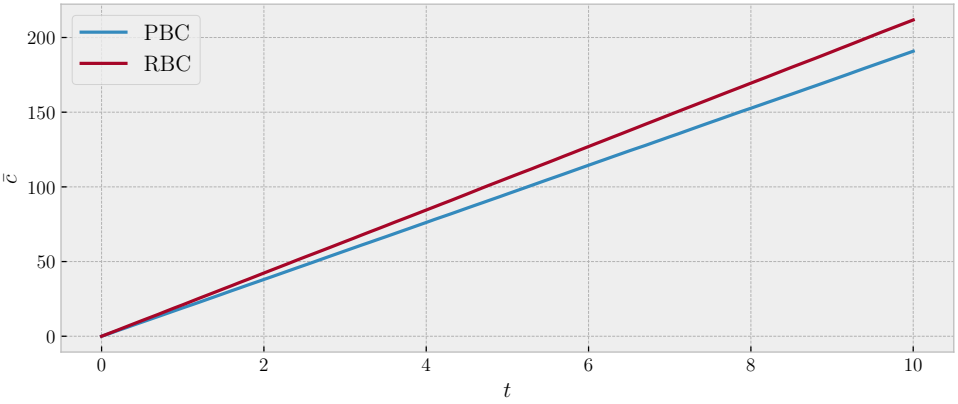



**Figure 4.3:** Plot of the average number of collisions per particle as a function of time for a similar system and parameters as in Figure 4.2. As seen in the plot, the average number of collisions per particle increases in a similar fashion for all times, and is higher for RBC, resulting from a higher collision frequency.

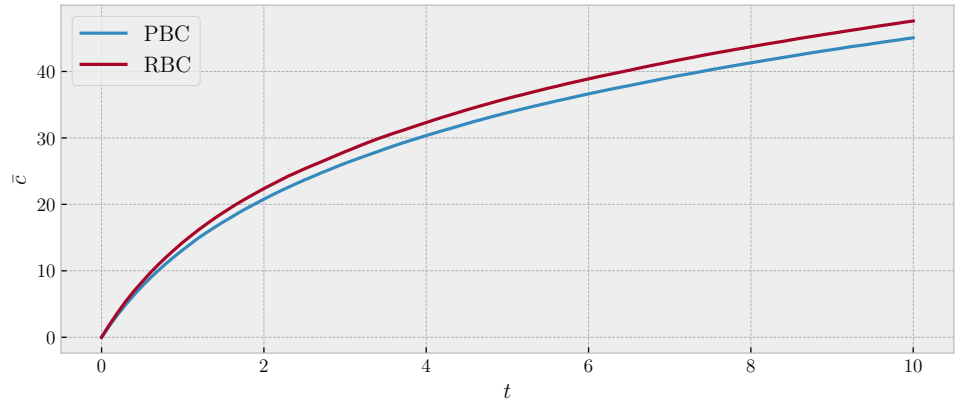



**Figure 4.4:** Similar plot as in Figure 4.3, but for a granular gas, here with $\xi = 0.8$. The dissipation of energy compared to a molecular gas makes the increase in the average number of collisions per particle slow down as a function of time. We still see the same comparison between PBC and RBC as for a molecular gas, where RBC gives a higher collision frequency.

### 4.1.2 Speed distributions

We want to see if we can reproduce the Maxwell-Boltzmann speed distribution in two and three dimensions, which is one of the most well-known results for a molecular gas, familiar to many from introduction courses to thermal and statistical physics. These results are provided for two reasons. The first reason, similar as for all results in this section, is to use known results in order to verify the implemented event driven simulation. The second reason is based on the study of granular gases, where we want to use an equilibrium state for a molecular gas as the initial velocities for a granular gas. In order to compute an equilibrium state we start with a system where the initial velocities are given by (3.3) with the initial speed $v_0 = \sqrt{2}$. For these simulations we have used RBC, as the speed of the particles are a quantity unaffected by the boundary conditions. Additionally we also exploit the fact the RBC gives a higher collision frequency and the simulations are faster as we do not need to copy the system of particles to enforce a periodic system. One should note that only the pairwise particle collisions make the system converge towards an equilibrium state, whereas the interactions with the walls do not. Note that for the purpose of counting the number of collision, we count both types for simplicity, however the majority of the collisions are particle-particle collisions.

#### Two dimensions

In two dimensions we have used a pseudo two-dimensional system by enforcing that all $v_{zi} = 0$ and $z_i = 0.5$. I.e. that all particles have zero velocity along the $z$-axis and all particles are confined to a square region limited by the vertical and horizontal walls for which we simulate a set of hard disks instead of hard spheres. The computed speed distribution in two dimensions is plotted as a histogram in Figure 4.5. The results in Figure 4.5 are given as the average of 20 runs for a system of $N = 2000$ with a radius $r = 0.005L$ giving a two-dimensional packing fraction $\approx 0.16$, starting from the initial system illustrated in Figure 3.7. For these simulations we used a stopping criterion given as $\bar{c}_{stop} = 0.2N$. By comparing the results with the theoretical prediction for the Maxwell-Boltzmann speed distribution in two dimensions in (2.30) one can see that we have successfully managed to reproduce the speed distribution. The agreement is also a reliable measure of expressing that is it possible to conduct two-dimensional simulations with the implemented three-dimensional event driven simulation as discussed in chapter 3.

#### Three dimensions

While in order to do simulations of a two-dimensional system we have to adjust the implemented simulation, performing simulations of a three-dimensional system is more straightforward. The computed speed distribution in three dimensions is plotted as a histogram in Figure 4.6 together with the three-dimensional Maxwell-Boltzmann distribution given in (2.32). For the simulations we have used a system of $N = 1000$ particles with a radius $r = 0.025L$, giving $\eta \approx 0.065$, which is the standard system we will be using for the study of a granular gas later. As for the simulations of a two-dimensional system we have used the same stopping criterion $\bar{c}_{stop} = 0.2N$ and the results are given as the average of 20 different runs. Figure 4.6 shows a good agreement between the computed values and the theoretical predictions, which we see as a verification that the implemented event driven simulation can be used to study many-particle systems of particles colliding in a box.

#### Expectation values

As we have seen for the computed speed distribution in equilibrium, the results have shown good agreement with the Maxwell-Boltzmann speed distribution. In addition to the clear visual agreement, we have also computed the values of the expectation values given in Table 2.2. The computed values are presented in Table 4.1 in addition to the relative error compared to the theoretical expressions.

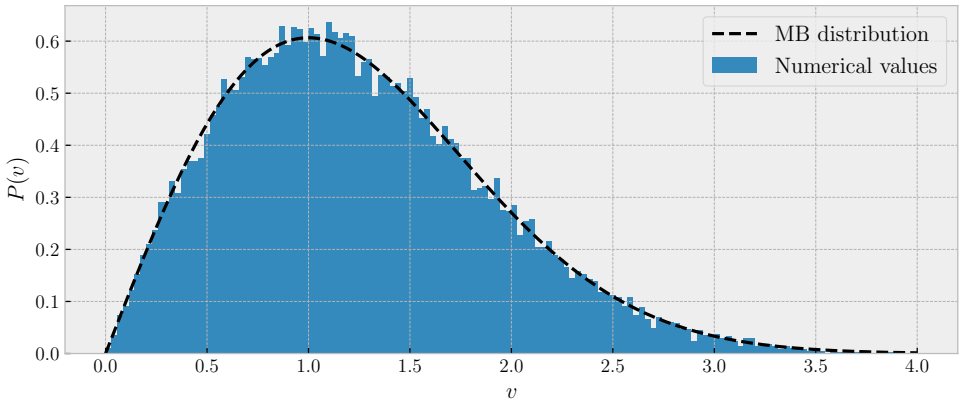



**Figure 4.5:** The computed speed distribution of a two-dimensional molecular gas in equilibrium. The result has been achieved for a system of $N = 2000$ particles with a radius $r = 0.005L$, giving a two-dimensional packing fraction $\approx 0.16$, with a stopping criterion given as $\bar{c}_{stop} = 0.2N$ with RBC. The initial velocities of the particles were given by (3.3) with $\theta = \pi/2$ and $v_0 = \sqrt{2}$ for a pseudo two-dimensional system. The results have been computed as the average of 20 different runs. The two-dimensional Maxwell-Boltzmann speed distribution (black dashed line) is given by (2.30), and the computed results are given as a histogram in blue.

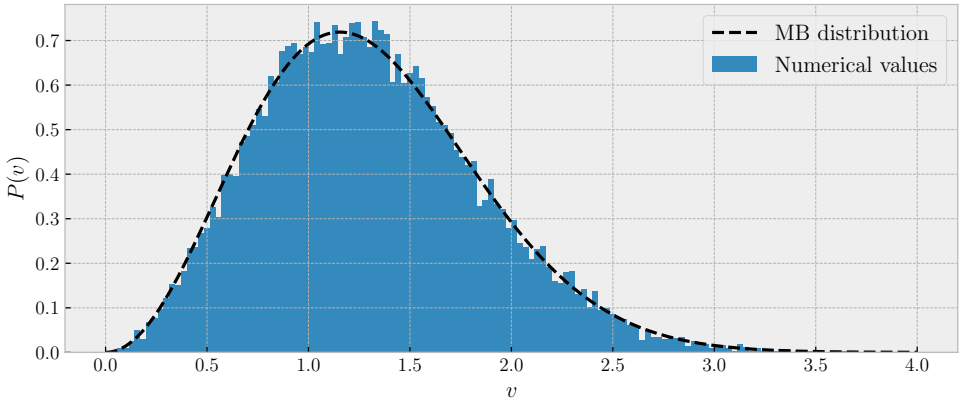



**Figure 4.6:** Similar plot as in Figure 4.5, but for a three-dimensional system of $N = 1000$ particles with a radius $r = 0.025L$, giving $\eta \approx 0.065$. We have used RBC with $\bar{c}_{stop} = 0.2N$ as the stopping criterion, and the results are computed as the average of 20 different runs, where each particle starts with an initial velocity given by (3.3) for $v_0 = \sqrt{2}$. The three-dimensional Maxwell-Boltzmann speed distribution is given by (2.32).

**Table 4.1:** The computed mean values and standard deviation, $\sigma$, of Maxwell-Boltzmann speed distribution in two and three dimensions. The relative error compared to the theoretical expressions in Table 2.2 are given in the table as [...] in %.

| Dimensions | $\langle v \rangle$ | | $\langle v^2 \rangle$ | | $\sigma$ | |
|---|---|---|---|---|---|---|
| 2 | 1.25 | [0.08] | 2 | [0] | 0.65 | [0.31] |
| 3 | 1.31 | [0.17] | 2 | [0] | 0.54 | [0.98] |

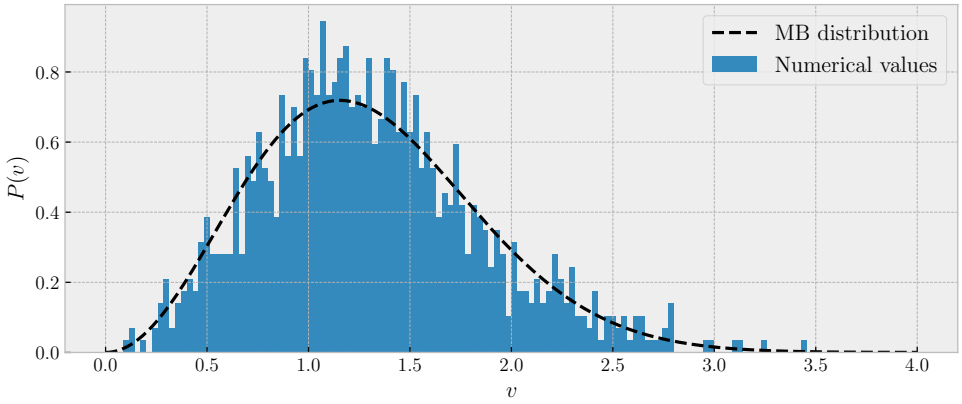



**Figure 4.7:** The initial speed distribution used as the initial velocities of a granular gas. The velocities of the particles are one of the end states of a unique run used to compute the average values in Figure 4.6. The three-dimensional Maxwell-Boltzmann speed distribution is given in black from (2.32). One should note that the average of many states such as illustrated here gives the proper prediction by theory (see Figure 4.6), whereas a unique run for such a low number of particles does not provide an accurate equilibrium state.

Note that the relative error for $\langle v^2 \rangle$ in Table 4.1 is default zero because we use the average particle energy, $\langle E \rangle$, to compute the temperature from the relation in (2.35) and energy is conserved.

#### Equilibrium state

As previously mentioned we want to use an equilibrium state for a molecular gas as the initial velocities for the particles in a granular gas. The results for the speed distribution in Figure 4.6 are given as the average of 20 runs. Since we can not use the average as an initial condition we have used the result of a single run as an initial condition. The speed distribution of the equilibrium state we have used as the initial set of velocities for simulations of a granular gas in given in Figure 4.7, together with the Maxwell-Boltzmann distribution in (2.32). Figure 4.7 does not show the same agreement as the average value, giving an example of why we use average values to get the correct behaviour predicted by theory. Using the initial state in Figure 4.7 we will now look at the evolution of granular temperature, and see if the numerical results follow the prediction by Haff's law in section 2.6.

### 4.1.3 Haff's law

Due to the dissipative interactions, modelled as inelastic collisions, the energy or equivalently the temperature of a granular gas will decay as a function of time. The evolution of granular temperature is known as Haff's law and depends on two main characteristics, the number of collisions and the amount of dissipated energy per collision. Before moving on to the use of PBC, necessary for the

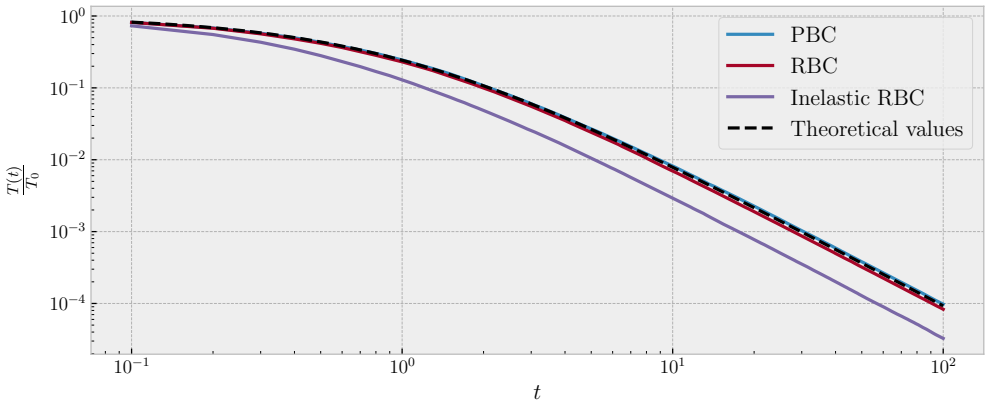



**Figure 4.8:** Plot of the impact on the evolution of the granular temperature from different boundary conditions for a granular gas, PBC (blue line), RBC (red line) and RBC with inelastic walls (purple line). The system is given by $N = 1000$ particles with a radius $r = 0.025L$, giving $\eta \approx 0.065$, colliding with a coefficient of restitution $\xi = 0.8$. The simulation has used $t_{stop} = 100$ as its stopping criterion with an output timestep of $\Delta t = 0.1$, starting from an equilibrium state of a molecular gas. The results in the plot are the average of 4 different runs. The theoretical values (black dashed line) are given from Haff's law in (2.40).

study of the MSD, we will first look at how different boundary conditions affect the dissipation of energy in the simulations. Then we will present some visualizations of two-dimensional systems in order to illustrate some of the impacts from the boundary conditions. In the end, we will see how the numerical results compare to Haff's law in (2.40) for a variety of different coefficients of restitution for simulations with PBC.

### Effect of Boundary conditions

In the presentation of simulation statistics, we saw an increase in the collision frequency for RBC compared to PBC. In light of this discovery we want to explore in detail how the different boundary conditions affect the decay of temperature for a granular gas with $\xi = 0.8$. The system used in the simulations consists of $N = 1000$ particles with a radius $r = 0.025L$ giving $\eta \approx 0.065$ starting from an equilibrium state of a molecular gas. In Figure 4.8 we have plotted the evolution of the granular temperature for three different boundary conditions together with the prediction of Haff's law from (2.40). The boundary conditions used have been PBC, RBC and RBC with inelastic walls. Inelastic walls are modelled by the collision rules in Eqs. (2.22), (2.24) and (2.26), while RBC refers to the simulation with reflecting elastic walls. This terminology is motivated by the clear disagreement between Haff's law and the numerical result for inelastic RBC in Figure 4.8. The disagreement clearly makes the use of inelastic walls a case where the boundaries are too dominating to achieve valid results compared to theory. From the plot in Figure 4.8 one could argue that both PBC and RBC seem to give a good agreement with theory. This seems to be the case, but we can see how the increased collision frequency manifests itself to some degree by providing lower values for RBC compared to PBC.

### Two-dimensional visualizations

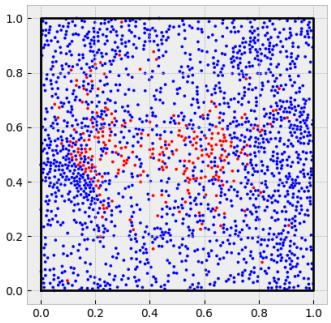
In order to complement the plot in Figure 4.8 we provide some visualizations of a two-dimensional system colliding with $\xi = 0.8$. The initial system is given in Figure 3.7 with $N = 2000$ particles with

a radius $r = 0.005L$ giving a two-dimensional packing fraction $\approx 0.16$. The same initial system has been evolved in time for both PBC and RBC and snapshots of the particle positions at $t = 2.5$ and $t = 10$ are given in Figure 4.9. In the figure we can see some explanation as to why the decay of temperature, and the dynamics in general, differ from PBC to RBC. For RBC clusters start to form close to the walls at an earlier stage than what seems to be case for PBC. For PBC we do eventually see the formation of clusters as well, but on a different size scale than for RBC. An important note to make is that the timescale of Figure 4.9 and Figure 4.8 can not be compared due to different packing fractions and dimensionality. For the simulation results for a three-dimensional system we have seen no indication that the system has left the homogeneous cooling state, while that is clearly the case for the two-dimensional system at long times in Figure 4.9.

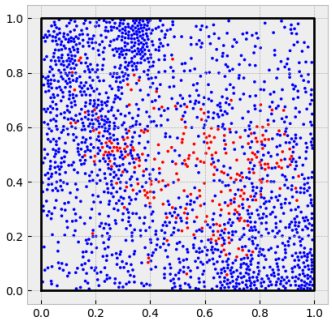
**The evolution of granular temperature**

The evolution of granular temperature for a granular gas is given in Figure 4.10 for a set of coefficients of restitution $\xi \in [0.99, 0.5, 0.3]$. The simulations are performed for a system of $N = 1000$ particles, with a radius $r = 0.025L$, giving $\eta \approx 0.065$ with PBC, the same system as we used to compute speed distributions in three dimensions. From the plot in Figure 4.10 we can see an excellent agreement between the numerical values and the theoretical values given by Haff's law in (2.40). The computed values in Figure 4.10 are given as the average of 4 different runs, where we have for each run computed the average kinetic particle energy. Additionally, we have used the central limit theorem presented in section 2.11 to see that the standard deviation of the mean value given in Figure 4.10 is small enough that we infer that the computed values have converged, i.e. we would not get better results by taking the average of more runs.

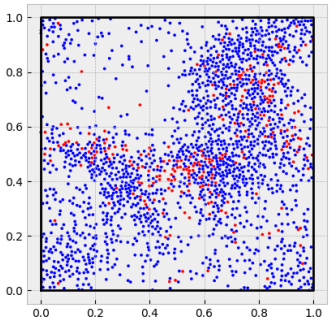
In order to better illustrate how the central limit theorem has been used let us look at the example of the granular gas with $\xi = 0.5$ in Figure 4.10. Note that we have run a total of 4 different runs for each coefficient of restitution. For each of those runs we have computed the evolution of the granular temperature as a function of time. I.e. at each point in time for each of the plots in Figure 4.10 we have 4 different computed values of the same quantity. In the plot only the average of these values are given, as discussed earlier. Let us pick the convenient time $t = 1$ where the dimensionless granular temperature for $\xi = 0.5$ is $\approx 0.1$ as illustrated in Figure 4.10. The essence of how we use the central limit theorem (see section 2.11) is that we assume that all computed granular temperatures in each point in time are drawn from the same distribution. The standard deviation of the 4 different granular temperatures at time $t = 1$ is $\approx 10^{-3}$ (not shown here). As one assumes that the unknown true standard deviation of the distribution of the granular temperature, $\sigma$, at that point in time is equal to the standard deviation of the computed values we get that the standard deviation, and thus an error estimate, of the average value is given by (2.122). We thus compute the error to be $\sigma/\sqrt{4} \approx 0.0005$ for the granular temperature at the time $t = 1$. We see however that this error becomes smaller as we take the average of more runs as the error estimate is $\propto n_r^{-1/2}$. To realize why the average of more runs does not give better results we need to remember that the true error of the computed values is found by comparing with the theoretical expression of Haff's law in (2.40). For the time $t = 1$ for the system with $\xi = 0.5$, the absolute difference between the average numerical value and theoretical value is $\approx 10^{-2}$ (not shown here), giving a relative error of $\approx 0.5\%$. As the true error is higher than the estimation based on the average value, we would get a better estimate of the average value from the computed values by taking the average of more runs, but that average would not result in a better agreement with theory. By using the central limit theorem in the following manner we can be certain that the agreement between the numerical and theoretical values cannot improve without changing and improving the implementation. The exact values of the errors change from one point in time to another as the value for $\sigma$ is different. However, for the plot in Figure 4.10 there is a trend where the true error is larger than the error estimate from the central limit theorem (also not shown here), for which one can infer that the average of more runs would not improve the correspondence between the numerical results and theory.



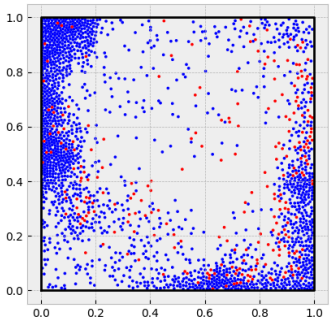
**(a)** $t = 2.5$ with PBC.



**(b)** $t = 2.5$ with RBC.



**(c)** $t = 10$ with PBC.



**(d)** $t = 10$ with RBC.

**Figure 4.9:** An illustration of how the positions of the particles in a granular gas with $\xi = 0.8$ can change in time for PBC and RBC. The initial system for both simulation are given in Figure 3.7. The simulation with PBC are given at $t = 2.5$ and $t = 10$ in Figure 4.9a and Figure 4.9c, while the simulation with RBC are given at the same times in Figure 4.9b and Figure 4.9d. The different figures show that we achieve very different systems for the different boundary conditions, where the clustering occurs near the walls for RBC somewhat earlier than we see clusters forming for PBC. The size of the formed clusters are also on different scales.

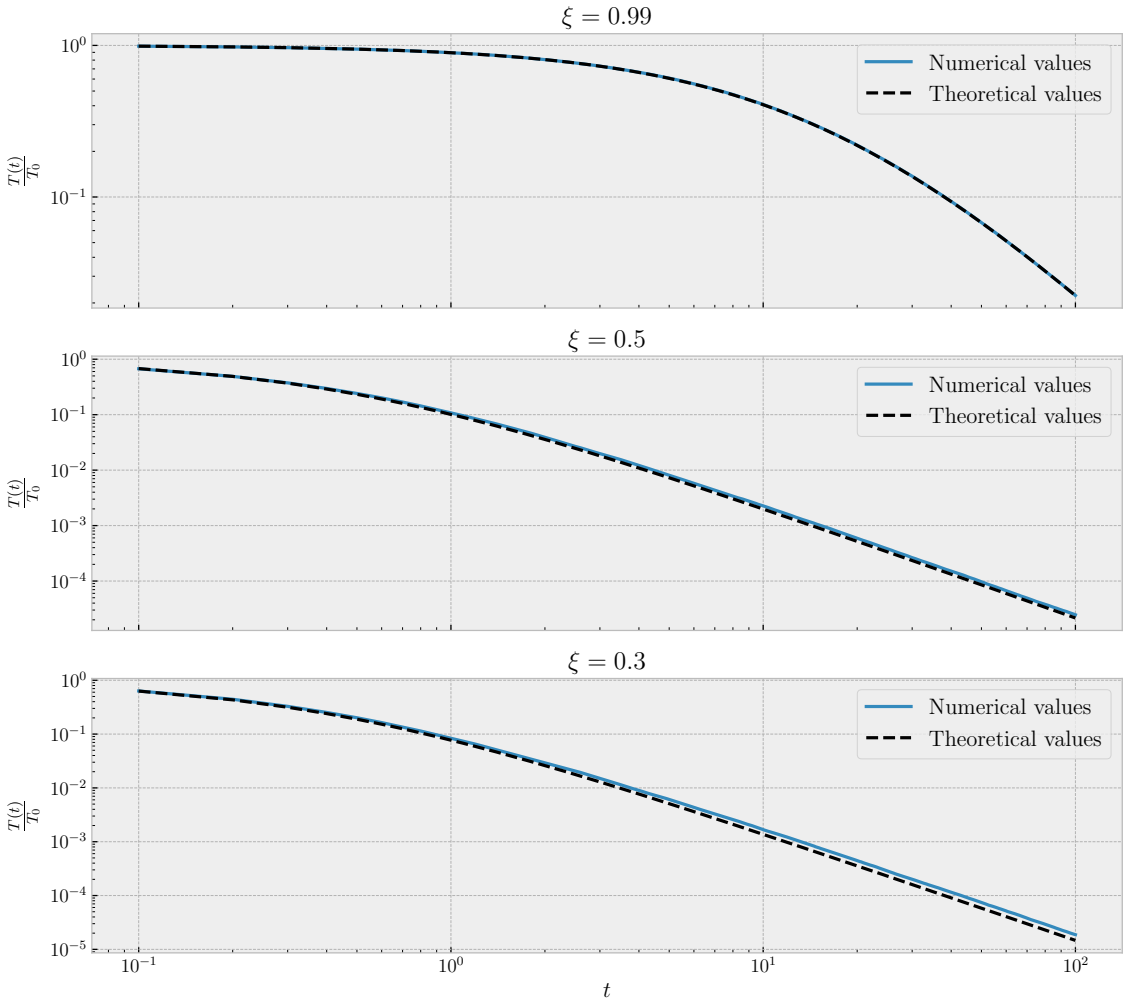



**Figure 4.10:** Plot of the evolution of granular temperature for a granular gas in systems with $\xi \in [0.99, 0.5, 0.3]$. For all the simulations we have used a system of $N = 1000$ particles, with a radius $r = 0.025L$, providing $\eta \approx 0.065$ with PBC. The computed results (blue line) have been achieved with a stopping criterion of $t_{stop} = 100$ with an output timestep $\Delta t = 0.1$, and are given as the average of 4 runs starting from an equilibrium state of a molecular gas. The theoretical prediction (black dashed line) is given by Haff's law in (2.40).

We do however see that the computed results and Haff's law becomes more distinguishable for lower values of the coefficient of restitution, where the computed values are somewhat higher than theory. At an early phase for the work done for this thesis we tried to compare Haff's law with the simulation results for two-dimensional systems with RBC. While this also gave good agreement, the numerical values diverged from the theoretical values for long times, which most likely was a result of the formation of clusters. These simulations results are not shown here, but the disagreement between the numerical and theoretical values were somewhat consistent with the time scale of the clusters forming in Figure 4.9. I.e., when we compared plots of the granular temperature and the particle positions we saw that that clusters had formed or were beginning to form when the numerical and theoretical values diverged. For the time scale used in the simulations, it seems from the plot in Figure 4.10 that in three dimensions we are still in the homogeneous cooling state, for which Haff's law is valid and the derivations of the MSD expressions in section 2.8 are valid. As a result of the previous argument we can expect valid results for the simulations for the MSD, the essential topic of Brownian motion.

### 4.1.4 Brownian motion

In the study of Brownian motion, we have used the same system of $N = 1000$ particles with a radius of $r = 0.025L$, giving a packing fraction of approximately 0.065, equal to the system used in [15], to conduct studies of the MSD as a function of time. By using different values of the coefficient of restitution we can conduct studies of a molecular gas ($\xi = 1$) or a granular gas ($\xi < 1$). The event driven simulation is performed by starting from an equilibrium state, with the speed distribution given in Figure 4.7. The theoretical predictions of the MSD in section 2.8 are derived for a one-dimensional system, and in order to compare with the three-dimensional event driven simulation we have multiplied the theoretical values with 3 due to the symmetry properties of the MSD presented in (2.56). The described system is used in order to compare results for the MSD of a granular gas with the results in Fig. 2 in [15], which uses the same packing fraction[1].

#### Molecular gas

The computed value for the MSD as a function of time for a molecular gas is given, together with the theoretical prediction from (2.68), in Figure 4.11. The numerical values in Figure 4.11 are given as the average of 4 different runs, where the MSD has been computed for all the particles in each run. In addition, in Figure 4.11 we provide an error estimate presented as error bars given by the central limit theorem in (2.122), where we compute the standard deviation of the mean computed value. The error bars are computed from the central limit theorem (see section 2.11) by computing the standard deviation of the computed values at each point in time, $\sigma_i$, and assuming that the computation of the MSD at each point in time is drawn from the same distribution with a standard deviation equal to the computed values. Thus, the error estimate in each point in time have been computed as $\sigma_i/\sqrt{4}$, and the error estimate of the mean MSD is thus a function of time. From the plot in Figure 4.11 we can discuss the validity of the theoretical expression in (2.68). The first thing we should note is that Figure 4.11 exhibits the asymptotic behaviour predicted for a molecular gas in (2.69), where we can see a ballistic periodic where the MSD $\propto t^2$ for $t < 10^{-1}$, and a linear dependence on time for $t > 1$. The second thing to emphasize is that on a logarithmic scale the computed MSD seems to be inseparable from the theoretical predictions. If we had chosen to plot the values on a non-logarithmic scale we would see a slight disagreement between the slope of the computed values and the theoretical values, which would make the values more distinguishable

[1]The results in [15] are based on a system where $L = 40$. Hence we cannot directly compare the results due to the explicit dependence of $n = N/L^3$ in the parameters $\tau_0, D_0$ and $\gamma_0$. Even though we have used the same relation between the particle size and wall size, they will for the same initial average energy get a much longer ballistic period. We have however compared to see that the theoretical predictions for the expression with $L = 40$ matches the plot in Fig. 2 in [15] to be certain that we use an equivalent foundation to compare the numerical values with theory.

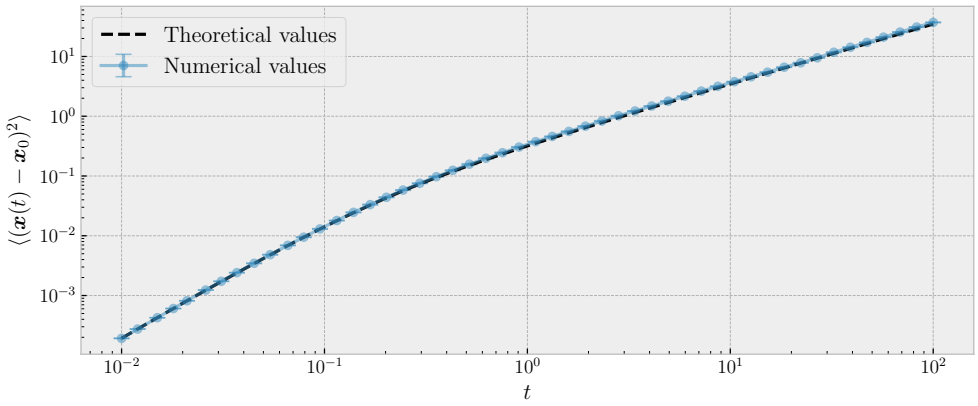



**Figure 4.11:** Plot of the MSD as a function of time, $t$, for a molecular gas of a system of $N = 1000$ particles with a radius of $r = 0.025L$. For the plot we have used a stopping criterion of $t_{stop} = 100$ and used logarithmically spaced times to compute the MSD on suitable times. The computed values (blue line with error bars) are given as the average of 4 different runs, and the error is given as the standard deviation computed from the central limit theorem in section 2.11. From the errorbars we can see that the average value have converged. The theoretical predictions (black dashed line) for the MSD of a molecular gas are given by (2.68). On a small timescale the plot indicates an MSD $\propto t^2$, before changing to $\propto t$, which is regarded as normal diffusion.

for long times due to the linear dependence. Logarithmic scales are preferred to easily see the asymptotic behaviour, especially for short time scales. We refer to Appendix F for some additional elaboration and treatment of this issue, where we also provide the same plot as in Figure 4.11 using non-logarithmic axes. Regardless of the scale used for the plots, we do see the good agreement between theory and the computer simulations making it plausible that the event driven simulation of many-particle systems seems to exhibit the correct dynamics for a molecular gas.

### Comparison with earlier results

The agreement for the MSD as a function of time found for a system with PBC here is better than what we achieved for the two-dimensional systems with RBC used in the specialization project [1]. The inspiration of the use of RBC was based on work in [24] and due to simplicity. Another explanation as to why the agreement have improved can be the trouble of the convergence of the integral of the velocity autocorrelation function for two-dimensional systems [see 11, p. 162]. As discussed in [11, p. 162] the decay of temperature for a granular gas seems to help the convergence in two dimensions, but that is not the case for a molecular gas. Hence the change to three dimensions, and the change to PBC, have improved the simulations results to an extent where we achieve satisfactory agreement with theory. In light of the better agreement found between the numerical results and theory achieved for this thesis, it seems natural to assume that the choice of boundary conditions affected the system more than we realized at first.

### Granular gas

For the study of a granular gas, we expect a different behaviour of the MSD due to the dissipation of energy. Due to the general collision rule in Eqs. (2.10) and (2.11) all we need to study a granular gas compared to a molecular gas is to change the value of $\xi$ in the implementation. For a system of a granular gas we have achieved the results for the MSD with $\xi = 0.8$ given in Figure 4.12, and with

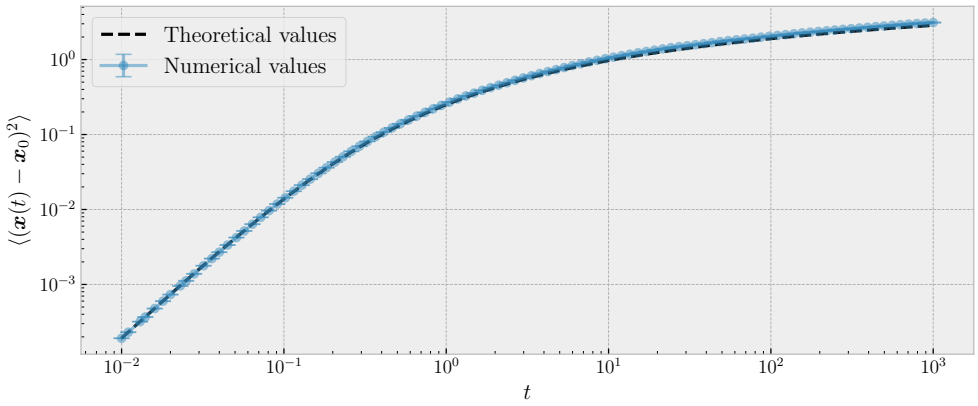



**Figure 4.12:** Similar plot, parameters and system as in Figure 4.11, but for a granular gas with $\xi = 0.8$. In the plot we can see a ballistic period where the MSD $\propto t^2$, before changing a logarithmic dependence on time, one of the properties of ultraslow diffusion. The theoretical values (black dashed line) are given by (2.79).

$\xi = 0.3$ given in Figure 4.13. The computed results are given for the same system and parameters as for a molecular gas in Figure 4.11. The theoretical prediction of the MSD of a granular gas is given in (2.79), where the asymptotic behaviour is presented in (2.80). By comparing the computed average values and the theoretical predictions in Figure 4.12 and Figure 4.13 we see that the plots show a good agreement with theory, and the standard deviation is small enough that we consider the results to be converged based on a similar argument as presented for the results of the MSD of a molecular gas and the evolution of the granular temperature. I.e. that the agreement for the MSD would not improve compared to the theoretical values by taking the average of more runs. The latter is clear from Figure 4.13 where the error bars are smaller than the difference between the numerical and the theoretical values. We can also clearly see that we achieve the correct asymptotic behaviour, which gives us a ballistic period before the MSD changes to a logarithmic dependence on time, one of properties of ultraslow diffusion as described in section 2.8. We would like to again emphasize as earlier that the agreement seems better on a logarithmic scale than on a linear scale, as discussed in Appendix F. We do however as earlier see that we achieve a valid agreement between theory and the simulations. The plots of the MSD show the same behaviour as achieved in [15]. Due to the high degree of agreement between the computed values of the MSD and the theoretical predictions for a granular gas it seems that an event driven simulation of pairwise collisions captures the dynamics of a granular gas at the particle volume density we have studied. Hence we can argue that the implemented simulation method is correct.

As we saw in the results for the evolution of granular temperature in Figure 4.10, there was a increasing disagreement between the computed values and the theoretical predictions for lower values of the coefficient of restitution. The same was observed from the results of the MSD of a granular gas in Figure 4.12 and Figure 4.13. Through numerical experimentation (not shown) during the work done for this thesis we have run simulations for a number of different $\xi \in [0.1, 0.99]$ and for us it seems like a clear trend where the disagreement is higher for lower values of $\xi$. In order to explain why we see this trend we have looked into a few properties of the expressions for the MSD, a function of the constants $\tau_0, D_0$ and $\gamma_0$. As one can see in the comparison to Haff's law in Figure 4.10, the simulation results gives an accurate approximation of the characteristic decay time of the granular temperature. The long term behaviour of the results of the MSD are given by the diffusivity of the system, and we have thus decided to see how the model for the diffusivity in (2.51)

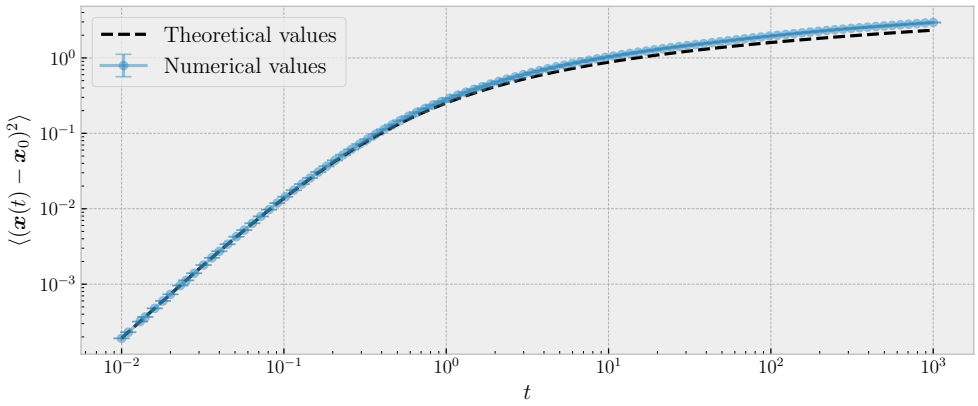



**Figure 4.13:** Similar plot as in Figure 4.12, for the same system with a coefficient of restitution $\xi = 0.3$.

corresponds to the simulation results.

### Diffusivity

The use of diffusion properties to study the case of particles colliding in a box, either elastically or inelastically, has given some important verification tools. As mentioned in section 2.8 it is possible to explain the long term behaviour of the MSD as the integral of the diffusion coefficient, given as a three-dimensional expression of the relation in (2.64). We have used said relation to go the other way, where we from simulation results of the MSD have computed the diffusivity as a function of time. The derivative of the MSD has been computed from the following first order finite difference approach. First we define the forward difference, where the derivative of the quantity $u$ at a time $t_i$, denoted by $u_i$, is approximately given by

$$\left.\frac{du}{dt}\right|_{t_i} \approx \frac{u_{i+1} - u_i}{t_{i+1} - t_i}, \tag{4.1}$$

where $t_{i+1}$ is the next point in time and we have ignored terms of the order $\mathcal{O}((t_{i+1} - t_i)^2)$ and higher. The backward difference is given in a similar fashion by

$$\left.\frac{du}{dt}\right|_{t_i} \approx \frac{u_i - u_{i-1}}{t_i - t_{i-1}}, \tag{4.2}$$

where $t_{i-1}$ is the previous point in time and we have ignored terms of the order $\mathcal{O}((t_i - t_{i-1})^2)$ and higher. For a constant resolution in time, where the difference in time between two values are given by $\Delta t$, the expression for the forward and backward finite difference in (4.1) and (4.2) are simplified and we could compute a central difference scheme in a straightforward manner [see 37, pp. 45–52]. For the computation of the MSD on logarithmically spaced times we have a non-uniform resolution in time, and we have computed the derivative of the MSD as the average of the forward and backward finite difference in (4.1) and (4.2). At the end points where the forward (backward) difference is not defined we have used the backward (forward) difference to compute the values.

For a molecular gas we expect to see a constant diffusion coefficient as we see a linear trend for the long term behaviour of the MSD. The diffusivity for the system in Figure 4.11 is given as a function of time together with a unity reference in Figure 4.14. As expected we do see a fairly constant diffusion coefficient after the time $t \approx 0.3$, which is the point where we start to see the linear

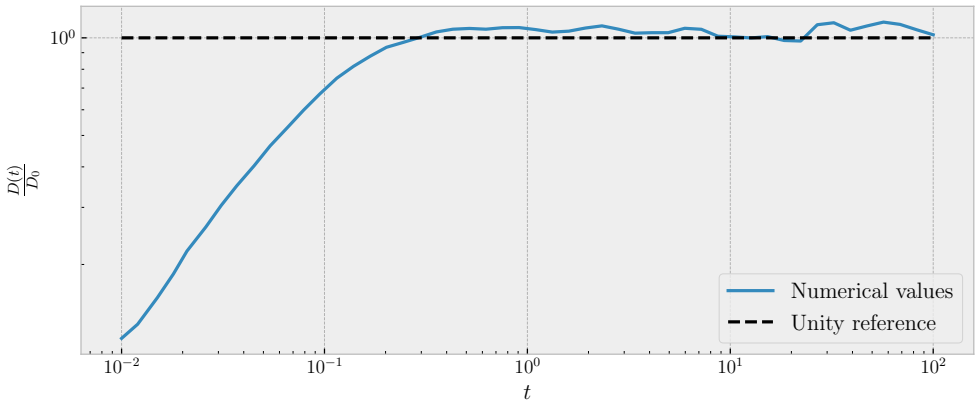



**Figure 4.14:** Plot of the diffusivity of a molecular gas for the system in Figure 4.11. We have computed the diffusivity from a three-dimensional expression of the relation in (2.64), where we have used the mean of the forward and backward difference in (4.1) and (4.2) to compute the diffusivity. From the plot we can see that the diffusivity becomes to some extent constant when the MSD display the linear trend in Figure 4.11. The behaviour is as expected as the expression for the MSD as the integral of the diffusivity is only valid for long term behaviour.

trend of the MSD in Figure 4.11. The plot of the diffusivity for a molecular gas in Figure 4.14 offers an explanation as to why the numerical values of the MSD are slightly higher than the theoretical expression, discussed in Appendix F. If the simulation results are consistent with a higher diffusion coefficient compared to theory for the used set of particle parameters, it seems to give a natural explanation as to why the results for the MSD are somewhat higher. The reason being that the MSD at long times is given as the integral of the diffusivity.

For a granular gas, where the collisions are inelastic, we expect to see a diffusion coefficient with a time dependence given by (2.51). The time dependence of the diffusivity is consistent with the long term dependence of time for the MSD of a granular gas, with a logaritmic dependence on time. The diffusivity of a granular gas with $\xi = 0.8$ for the system in Figure 4.12 is given as a function of time in Figure 4.15. An equivalent plot for a granular gas with $\xi = 0.3$, the system in Figure 4.13, is presented in Figure 4.16. For the plots we infer that the model used for the diffusivity in this thesis is in agreement with the simulation results for the MSD. We do however, as earlier, see that the agreement is lower for lower values of the coefficient of restitution. In addition, the trend of a higher diffusivity than we should expect as we saw for a molecular gas, is also present here. For us it seems clear that the trend for a granular gas can be a manifestation that the model for the dynamics of a granular gas based on kinetic theory [11] is a better fit for higher values of the coefficient of restitution. The previous argument is hard to use for anything other than to collect thoughts concerning the results for the MSD. We have not been able to locate any published results for the MSD of a granular gas with a constant and low coefficient of restitution to see if they experienced some of the same behaviour.

To explain why we achieve a higher diffusivity than expected is a complex and difficult topic. It can be as simple as a result of numerical errors resulting from number representation, but that seems very unlikely. There is also a possibility that the reduction of the molecular dynamics to only pairwise particle collision results in an overestimation of some of the diffusion properties. Even though we have done simulations for packing fractions where there should be few multi-particle interactions, it is difficult to be certain that we capture the complete correct dynamics of a gas with this assumption, as mentioned in [11, pp. 5–7]. It should also be mentioned again that the use of a finite number of

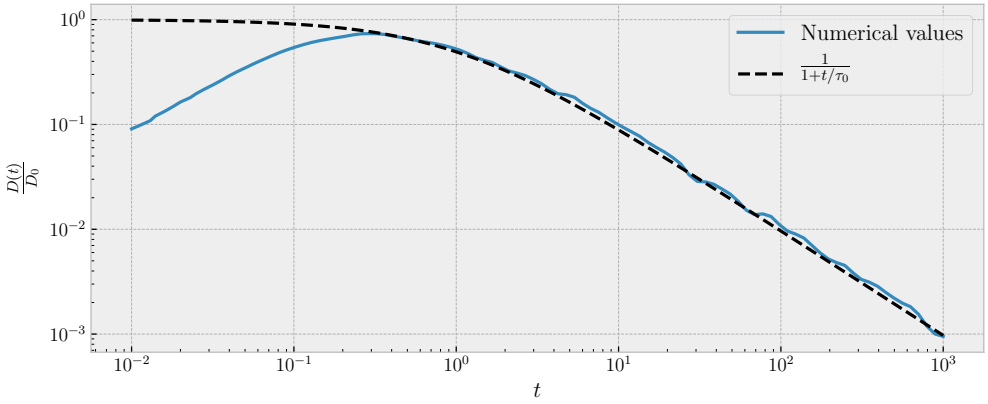



**Figure 4.15:** Similar plot as Figure 4.14 for the system of a granular gas with $\xi = 0.8$ addressed in Figure 4.12. From the plot we can see that the diffusivity of a granular gas follows the prediction in (2.51) when the MSD in Figure 4.12 has entered the region with a logarithmic dependence on time.

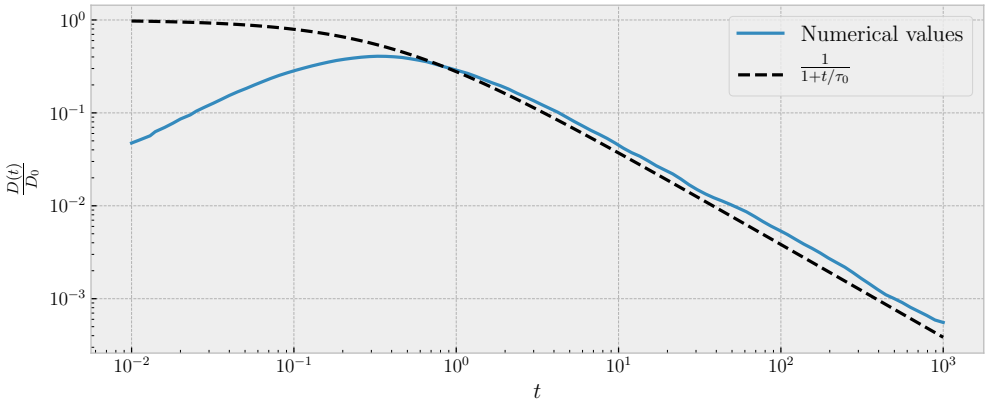



**Figure 4.16:** Similar plot as Figure 4.14 for the system of a granular gas with $\xi = 0.3$ addressed in Figure 4.13. Compared to the result for the granular gas with $\xi = 0.8$ in Figure 4.15, we do see the same behaviour for the diffusivity, but the numerical values do not show the same agreement with the theory in (2.51).

particles for numerical simulations makes it impossible to simulate an infinite system, which is often the idealization used for theory. Additionally, we have only subtracted the mean velocity along each axis at the beginning of each simulation. As time is incremented it is possible that the system at some point starts to move with a velocity in a direction. The last case would however probably give contributions to the MSD in a manner that would make the agreement between the numerical and theoretical values worse that we have seen. Due to the limited applications of such a simple model as used in this thesis, with a constant coefficient of restitution, it should maybe be left as an open topic.

## 4.2 Numerical solutions of SDEs describing Brownian motion

In addition to the results from the event driven simulation we want to see what kind of results we can achieve by solving SDEs describing Brownian motion numerically. By solving such equations, with similar results as the event driven simulation, we can verify the use of Langevin dynamics to model the dynamics of a granular system. We will start with the underdamped Langevin equation, where the solution as derived in section 2.9 exhibits the same velocity autocorrelation function as a molecular gas. Systems with the same velocity autocorrelation function should provide equivalent results for the MSD. We will continue by looking at UDSBM, a model used to approximate a granular gas. The solution to UDSBM does not provide the same velocity autocorrelation function as a granular gas, but a granular gas and the solution to UDSBM produce expressions for the MSD which converge to the same expression in the limit $\gamma_0\tau_0 \gg 1$. They do regardless exhibit the same asymptotic behaviour in (2.80). We have solved the SDEs numerically by applying the Euler-Maruyama approximation, introduced in section 2.9. In the end we will look at the time averaged MSD in order to see if we manage to capture the ergodic property, i.e. to see if the time averaged MSD is equal to the ensemble MSD, of the different particles we try to model using SDEs. Hence we expect a non-ergodic system for the solution to UDSBM, and an ergodic system for the solution to the underdamped Langevin equation. The non-ergodic system given by UDSBM is a result of UDSBM modelling a granular gas, which is a non-ergodic system [see 15].

In the implemented iterative scheme for the Euler-Maruyama method we solve the SDE for $N = 1000$ equal particles at the same time by using vector operations. We have then taken the average of several runs, as we did for the results from the event driven simulations. As we solve an SDE for a single particle without considering the other particles we can compute the MSD for a system of a varying degree of number of particles by using the correct values for the diffusion and friction coefficients. In that regard we pretend to have a system of $N = 1000$ equal particles and compute the coefficients corresponding to this system while we in the iterative scheme solve the SDE for each particle without considering the others. We have however used the same parameters and systems as for the results of the event driven simulation to produce comparable results. The diffusivity and friction coefficients, together with the velocity, determine the drift and diffusion ($a$ and $b$) coefficients which separate the different SDEs describing Brownian motion. The chosen value for mass, radius, number of particles and the coefficient of restitution determines the constants $\tau_0, \gamma_0$ and $D_0$, but are not used explicitly in the iterative schemes.

### 4.2.1 Underdamped Langevin equation

In order to solve the Underdamped Langevin equation we have used the iterative scheme given in (2.92). For the initial values of the positions and the velocities we have used that all $N = 1000$ particles start with the same position $X = [0.5, 0.5, 0.5]$ and the velocity of each particle is given by (3.3) for $v_0 = \sqrt{2}$. The random force is approximated as a Wiener process, and each particle thus gets a unique trajectory. We start by using the iterative scheme to compute the velocity and position at each timestep, before computing the MSD as a function of time at each timestep. As discussed earlier in section 2.9, the velocities of the particles are updated at each timestep, making it necessary to use a low value for the timestep to get the correct short time dependence of the MSD. For the solution to the underdamped Langevin equation we achieved an MSD as a function of time given in Figure 4.17, computed with the timestep $\Delta t = 0.01$ and given as the average of 4 runs. If we compare the results for the underdamped Langevin equation with the results from the event driven simulation (see Figure 4.11) we see a similar excellent agreement between theory and numerical values in Figure 4.17. Actually, we will see that the results for the numerical solution to the SDEs exhibit a trend where the agreement between theory and numerical results are somewhat better than what was the case for the event driven simulations.

For a molecular gas we assume a constant friction coefficient, $\gamma_0$, and diffusivity, $D_0$, computed

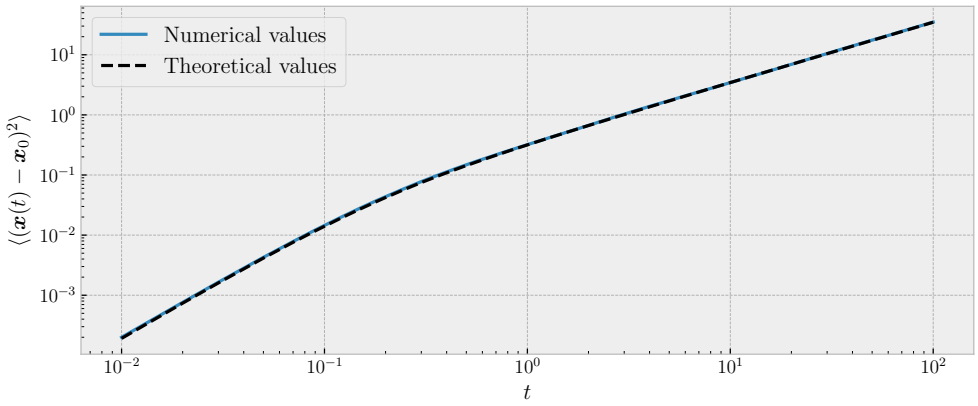



**Figure 4.17:** Plot of the MSD for the solution of the underdamped Langevin equation by applying the Euler-Maruyama method as a function of time. The computed results (blue line) are given for a set of $N = 1000$ particles with radius $r = 0.025L$, and are presented as the average of 4 different runs, computed with $\Delta t = 0.01$. The theoretical values (black dashed line) are given by the expression for a molecular gas given in (2.79). The plot gives similar results as the event driven simulation of molecular gas in Figure 4.11, with a ballistic period before changing to a linear dependence of time for the MSD.

from (2.55) and (2.52). We can thus see from the coefficients of the drift and diffusion in (2.91) that the only time dependence is given by the velocity. That is not the case for UDSBM, but we can use the same implementation by adding a time dependent friction coefficient and diffusivity.

### 4.2.2 UDSBM

We now turn to the case of UDSBM, used to model the dynamics of the particles in a granular gas. In order to solve UDSBM as an SDE we have used the iterative scheme given in (2.110). For the case of UDSBM we used the same initial conditions as for the solution to the underdamped Langevin equation. The dependence of time for the friction coefficient and the diffusivity in (2.54) and (2.51) are used to compute the correct coefficients $a$ and $b$ given by (2.109) at each timestep. The results for the MSD computed from the numerical solution to UDSBM are given for a system with $\xi = 0.8$ in Figure 4.18 and a system with $\xi = 0.5$ in Figure 4.19. For the computation we have used the timestep $\Delta t = 0.01$ and the results are given as the average of 4 runs, and we compare the results with the theoretical prediction from (2.113). As expected, we can see in Figure 4.18 and Figure 4.19 that we achieve the same asymptotic behaviour of the MSD as we saw for a granular gas, with an initial ballistic period before a logarithmic dependence on time, associated with systems exhibiting ultraslow diffusion properties. For higher values of the coefficient of restitution, e.g. $\xi = 0.8$, the MSD of the solution to UDSBM is a good match with the equivalent event driven simulation for a granular gas. This is clear by comparing the plots in Figure 4.12 and Figure 4.18. In addition to the simulations presented here we have also gotten similar results for the one-dimensional system (not shown here) of UDSBM given in [see 25, pp. 8–9]. By comparing the obtained results with others, we can be certain that we have implemented the Euler-Maruyama approximation correctly.

One thing to note is mismatch between theory and computed values for the MSD of the numerical solution to UDSBM with parameters corresponding to $\xi = 0.5$. This is not a result of an increasing error for lower values of the coefficient of restitution, but is a result of the difference between using the correct initial condition for the speed as used in the derivation of the velocity autocorrelation function for UDSBM given in Appendix C. For the simulation we have used $v_0 = \sqrt{2}$, in agreement with the event driven simulations of a granular gas. As it turns out the mismatch is a result of

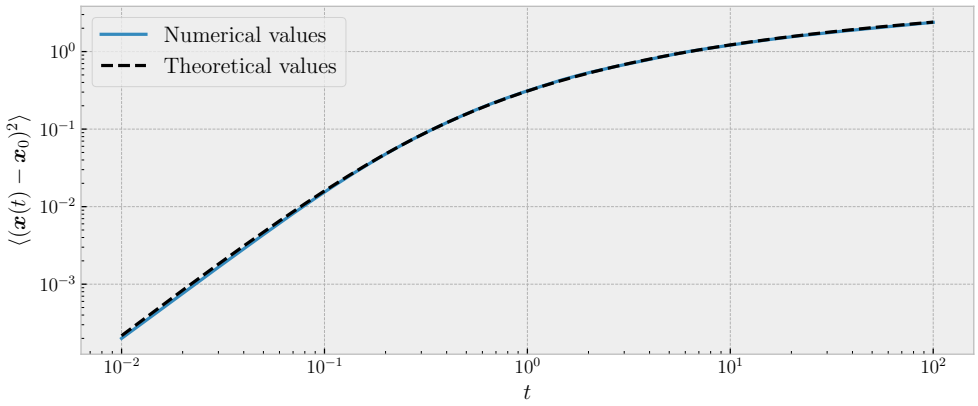



**Figure 4.18:** Plot of the MSD for the solution to UDSBM by applying the Euler-Maruyama method in (2.110) as a function of time. The constants $\tau_0, \gamma_0$ and $D_0$ have been computed for a system of 1000 particles with a radius $r = 0.025L$ and a coefficient of restitution $\xi = 0.8$. The computed values (blue line) are given as the average of 4 different runs, computed with $\Delta t = 0.01$. The theoretical values (black dashed line) are given by (2.113), which are similar to the expression for a granular gas for $\gamma_0\tau_0 \gg 1$. From the plot we can see the same asymptotic behaviour for UDSBM as we saw for a granular gas in Figure 4.12, with a ballistic period before the MSD scales logarithmically with time, as expected for a system with ultraslow diffusion properties.

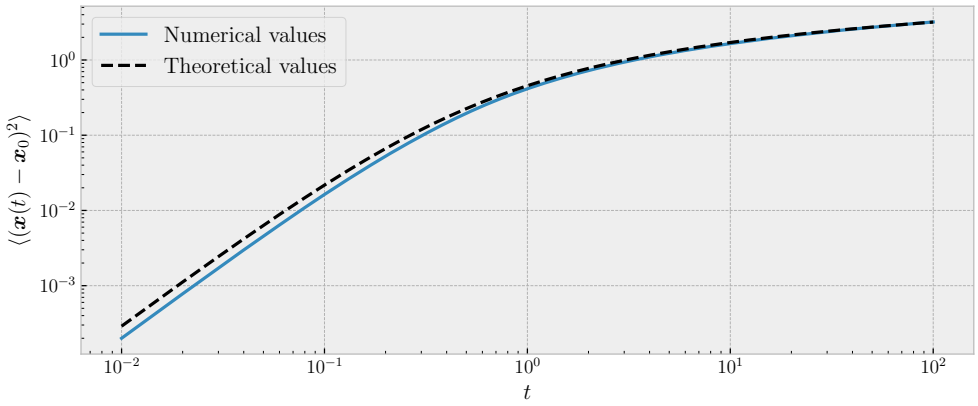



**Figure 4.19:** Similar plot as Figure 4.18, but for a system with $\xi = 0.5$.

the fact that the limit $\gamma_0\tau_0 \gg 1$ is not true for low values of the coefficient of restitution. In fact, based on the expression for $\tau_0$ in (2.45), $\gamma_0$ from (2.55) and $D_0$ from (2.52)[2] we can see that $\gamma_0\tau_0 \propto (1+\xi)^2/(1-\xi^2)$, and is thus an increasing function of $\xi$. For high values of the coefficient of restitution we will thus experience a better agreement with theory by using $v_0 = \sqrt{2}$, as seen in the results, compared to the correct value from theory $v_0 = \sqrt{(D_0\gamma_0^2\tau_0)/(\gamma_0\tau_0 - 1)}$ (see Appendix C). From the plot in Figure 4.19 we can also see that the long term behaviour of the MSD converges to the correct behaviour regardless of the correct initial conditions from the theory.

[2]It might seem strange at first to refer to $D_0$ when talking about $\tau_0\gamma_0$, but remember that we use $D_0$ to compute $\gamma_0$ due to the relation between the diffusion coefficient and the friction coefficient in (2.53).

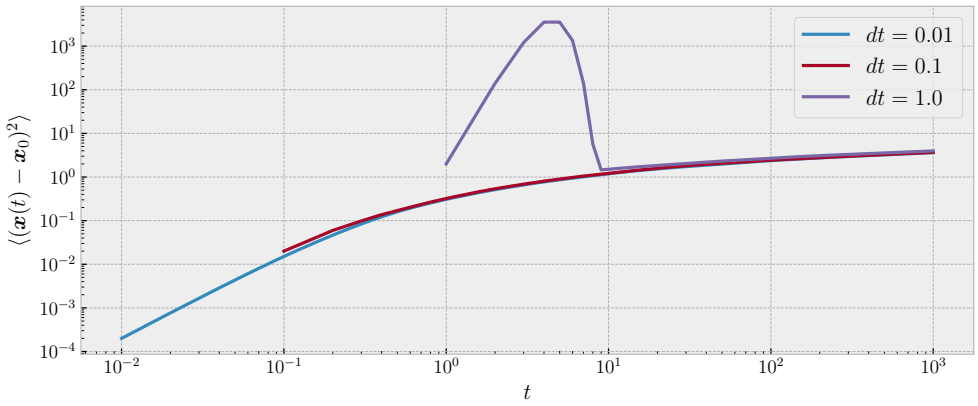



**Figure 4.20:** Similar plot as Figure 4.18, but the computed values are given for three different solutions with a different timestep. In the plot we can see the solution for $\Delta t = 0.01$ (blue line), the solution computed with $\Delta t = 0.1$ (red line) and the solution with $\Delta t = 1$ (purple line). In the plot we can see that all solutions converges towards the same results, but in order to get the correct short term behaviour we need a low value for the timestep. The results are given as the average of 4 runs.

For the plots in Figure 4.18 and Figure 4.19 we have used $\Delta t = 0.01$ in the results. The choice of the timestep is crucial in more than one way. First of all we do need a low value to capture the short term behaviour of the MSD. Additionally, due to only updating the velocity at each timestep we need to update a given number of times in order to capture the correct dynamics. We could never expect to see the logarithmic dependence from the correct time for the MSD by choosing a $\Delta t \gg 1$ simply because the velocities of the particles need a number of updates before starting to display the ulstraslow diffusion properties. In order to display the effect of the choice of $\Delta t$ we have in Figure 4.20 plotted the MSD of the solution to UDSBM for the parameters matching $\xi = 0.8$ for different choices of $\Delta t \in [0.01, 0.1, 1]$. As we can see from Figure 4.20 all the different simulations manage to capture the long term behaviour of the MSD eventually, whereas the computed values provide a better agreement with theory for smaller values of the timestep as expected.

There is some interesting behaviour regarding the solution with $\Delta t = 1$ we would like to address. Remember that we approximate the random term, modelling the particle collisions, as a Wiener process, and the difference in a Wiener process is $\sim \mathcal{N}(0, \Delta t)$. For high values of the timestep, we can thus get changes in the velocity of a high magnitude, that will vary greatly from simulation to simulation. Additionally, when we do not update the position and velocities that often we will thus get values for the MSD not corresponding with any theoretical predictions. We are actually a bit surprised that the MSD converges to the theoretical predictions for the long term behaviour. As a matter of fact we did also try higher values for the timestep (not shown here). The results showed even stranger behavior, before converging to a logarithmic dependence not consistent with the values proposed by theory.

### 4.2.3 Ergodicity

As a result of using the iterative Euler-Maruyama method to solve the SDEs describing Brownian motion for a total of $N$ particles at the same time, we have access to all particle positions at all times. This is actually an easy way to get memory problems by combining a high $N$ and $t_{stop}$ with a low $\Delta t$. We have used the solutions to the SDEs to display the ergodicity found for the solution to underdamped Langevin equation, whereas due to the non-ergodic behaviour of a granular gas we expect to find non-ergodic behaviour for the solution to UDSBM. Hence we want to compare the

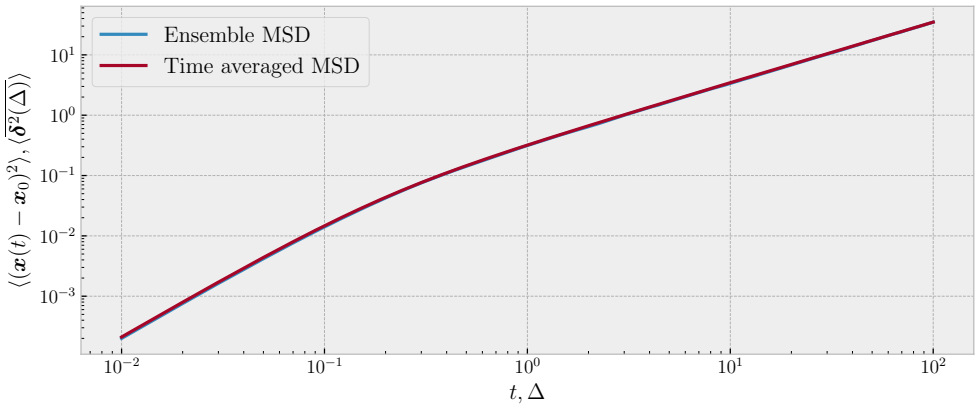



**Figure 4.21:** Plot of the ensemble (blue line) and time averaged MSD (red line) for the numerical solution to the underdamped Langevin equation as a function of time, $t$, and lag time, $\Delta$, respectively. The computed values have been obtained for a system of $N = 1000$ particles with a radius $r = 0.025L$ giving the constants $\tau_0, \gamma_0$ and $D_0$, which are used in the computation of the $a$ and $b$ coefficients each timestep. The values presented have been computed with $\Delta t = 0.01$, simulated until $t_{stop} = 100$, and are given as the average of 4 different runs. The time averaged MSD has been computed from (2.115) with Simpson's rule by using all particle positions at each timestep. From the plot we can see that the ensemble and the time averaged MSD are identical, as expected for an ergodic system.

ensemble MSD with the time averaged MSD given from (2.115).

A molecular gas, and thus the solution to the underdamped Langevin equation, is a system for which we expect to find the ergodic property, given in (2.114), where the time averaged MSD as a function of the lag time, $\Delta$, is equal to the ensemble MSD, which we for most of this report have only referred to as the MSD. In order to compute the time averaged MSD we have computed the three-dimensional equivalent of the expression in (2.115), where the $t$ is equal to the $t_{stop}$ used in the numerical solution to the SDE. We have computed the time averaged MSD with the same resolution for $\Delta$ as we have for the positions, namely $\Delta t$, using Simpson's rule from SciPy[3]. The ensemble and the time averaged MSD of the solution to the underdamped Langevin equation as a function of $\Delta$ and $t$ respectively is given in Figure 4.21 for the same system as we used to provide the simulation results for the solution to the underdamped Langevin equation. As we can see in Figure 4.21 the ensemble and the time averaged MSD are identical, as expected for an ergodic system. The topic of ergodicity for the underdamped Langevin equation is also discussed briefly in [25, p. 3].

A granular gas, and thus the solution to UDSBM, is a system which we expect not to exhibit the ergodic property in (2.114). The non-ergodic behaviour of granular gases is known, and is the main topic of [15]. For parameters similar to a granular gas with $\xi = 0.8$, the ensemble and the time averaged MSD of the solution to UDSBM are given in Figure 4.22. As we can see clearly from Figure 4.22, UDSBM is a non-ergodic system where the ensemble and the time averaged MSD are different. We also observe that the time averaged MSD depends on $t_{stop}$, as illustrated by comparing two different simulations in Figure 4.23. In addition from the results we can see the same behaviour as achieved in the study performed in [15] and [25] for higher values of $t_{stop}$ in regards of the development of the time averaged MSD as a function of the lag time.

[3]See `https://docs.scipy.org/doc/scipy/reference/generated/scipy.integrate.simps.html` for the documentation.

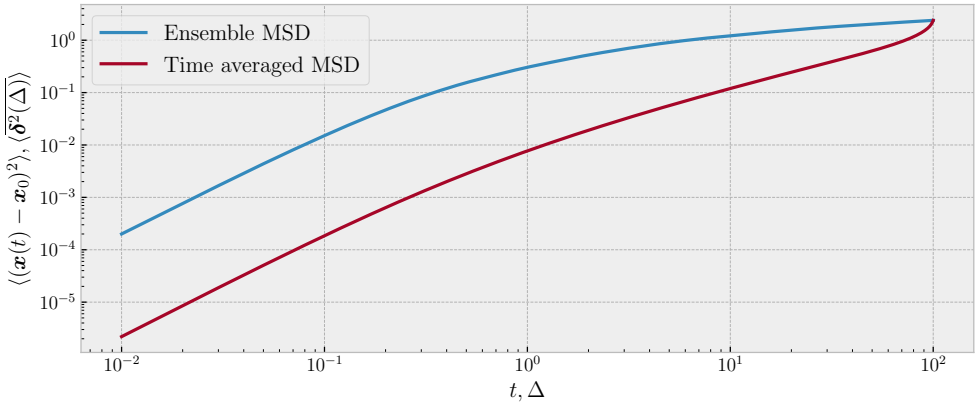



**Figure 4.22:** Similar plot as Figure 4.21, but for the solution to UDSBM with the constants $\tau_0, \gamma_0$ and $D_0$ corresponding to a coefficient of restitution $\xi = 0.8$. From the plot we can see that the ensemble and the time averaged MSD are different. Even though the time averaged MSD start with the same quadratic dependence on time as the ensemble MSD, the time averaged MSD does not become logarithmic.

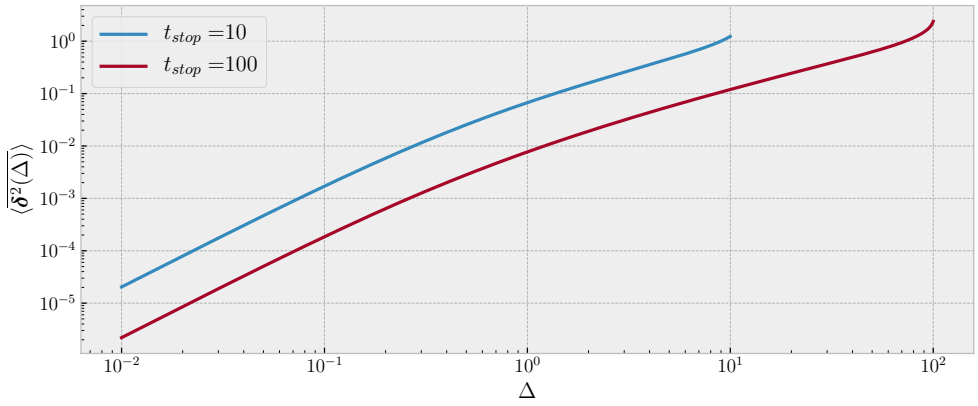



**Figure 4.23:** Plot of the effect of $t_{stop}$ on the time averaged MSD for numerical solution to UDSBM. In the plot we have given the simulation results for $t_{stop} = 10$ (blue line) and the simulation results for $t_{stop} = 100$ (red line). From the plot we can see that they are different, but do scale somewhat similarly in the region where we can compare the two. The difference for larger values of $t_{stop}$ is similar as the results given in Fig. 2 in [15].

# Chapter 5

# Further Work

There exist a variety of different topics suitable for further work based on this thesis both in the area of granular gas dynamics, and in other fields. These topics can roughly be sorted in three different categories. We will start by mentioning other aspects of granular gas dynamics, which already have been briefly mentioned throughout the theory in chapter 2 and the numerical modelling in chapter 3. We will then present some possible improvements to the implemented event driven simulation, before moving to the more open category of different uses for a similar type of event driven simulation.

There are some different topics in the area of granular gas dynamics which could be interesting to explore more in detail. One of these is the case of viscoelastic particles, where the coefficient of restitution is a function of the relative velocity. Due to the dependence of the relative velocity we get a different dependence of time for the decay of granular temperature, given by Haff's law [see 11, pp. 51–54]. As the temperature evolves differently, we also achieve a different dependence of time for the MSD of a granular gas, where a granular gas of viscoelastic particles exhibits a propertiy known as subdiffusion where the MSD $\propto t^{1/6}$ [see 11, pp. 142–143]. For such a system we see a clustering phase as for a constant coefficient of restitution, before we get a cluster dissolution as the coefficient of restitution approaches unity for low values of the relative velocity [see 5, pp. 181–184].

The topic of clustering and how a granular gas behaves when it is not longer in the homogeneous cooling state is another topic which can, and should, be studied further in order to gain more insight into the dynamics of a granular gas. In addition to the molecular dynamics simulation of a granular gas, it is possible to use Monte Carlo simulations to study a granular gas by looking at a hydrodynamic description used to solve the Boltzmann equation [see 5, pp. 191–210]. As it turns out, the use of hydrodynamic description may be necessary to fully describe a system of many particles, commonly applied in computational fluid dynamics [see 37]. The Boltzmann equation, hydrodynamic descriptions and the use of transport phenomenon for granular gases are presented in [3, 11], which is a natural place to start if one wants to study structure formation or use Monte Carlo simulations to study the hydrodynamics description of a granular gas. It is also possible to study a granular gas with a force description, where one can implement a molecular dynamics simulation and solve Newton's equation of motion. For such a simulation one could use different potentials to model the interactions between the particles. For an introduction to theory for more complex granular systems, such as driven systems, see [3]. Molecular dynamics simulations are, in spite of a simple conceptual idea, a very diverse topic and there exists a lot of literature which can serve both as an introduction and as a recipe for efficient algorithms, see e.g. [16].

From the way we have used SDEs to model similar dynamics as we have seen in the event driven simulation, it seems natural to assume that by adding additional forces or terms to the Langevin equations one can study both the interacting forces between the particles in a granular gas as well as the addition of external forces. The use of a general Langevin equation to model some granular systems is discussed in [5, pp. 293–301].

In order to improve the implemented event driven simulation one must first have a good understanding of the basis of the event driven simulation. We have implemented the event driven simulation with a priority queue for all future collisions, and the essence of the simulation is using the priority queue with the collisions rule in order to perform the correct sequence of collisions. By splitting the system into a set of boxes we can use the boxes to only store the future collisions for particles in the same or the neighbouring box. As long as the box is larger than the particle diameter we only need to consider the particles in the same box and the neighbouring boxes. This is the same principle as we used in the implementation of the PBC. The same procedure can be used for smaller boxes inside the system in order to not compute when particles far away will collide. We can then implement PBC by knowing which are the correct neighbouring boxes for the edge boxes. Note that we would have to compute all future collisions every time a particle cross into another box. The use of many boxes is discussed in the improved algorithm in [see 5, pp. 160–168], together with a few other possible improvements. This scheme can also be used to perform an event driven simulation in parallel as discussed in [38]. Such a modification can be necessary for simulations of many particles as the number of collisions in the priority queue can cause memory problems if we add all collisions as we have done in the implementation.

One additional improvement is to use a priority queue based on a more complex heap than what is the case for heapq in Python. It is also possible to explore other data structures to save the collisions, where it is possible to delete invalid entries, such as the collisions we have to discard when comparing the collision count. As such structures quickly become non-trivial we are somewhat uncertain that it is possible to use a better one for the implementation. It might still be worth to explore, but one will most likely get a much better improvement by exploring options to get fewer entries in the priority queue as discussed above. For a possible way to use an approach where we delete invalid entries see [16, pp. 391–417]. The use of several boxes in a system can also make use of different times in order to minimize the number of times we update the e.g. position of all the particles in order to minimize the error resulting from number representation. For simple speed enhancement, it is possible to write an equivalent program using a compiled programming language such as C or Fortran[1].

As we saw in the introduction in chapter 1, there are numerous fields where one can use an event driven simulation. Here we will focus on the different areas where the same event driven simulation as we have used in the study of granular gases can be used. The first we would like to discuss is the area of studying the spread of disease in a population. As a matter of fact, we have used the implemented event driven simulation to do some basic studies of how many people need to be at rest[2] in a two-dimensional system before a disease does not reach the entire population, and keep the immediately need for medical help at a given time under some control. This simulation was performed by infecting a random particle with a disease, and let future collision partners be infected if they are healthy. All particles could spread the disease a time after being infected before recovering and becoming immune. Even for such a simplified simulation we did see great effect for social distancing as a mean to flatten the curve. It could be interesting to use such an event driven simulation to see how different social changes can affect the spread of disease. Whereas it is hard to see other uses for the exact same event driven simulation, the use of time driven simulations are used both in the area of active matter and molecular dynamics in general. The area of solving SDEs numerically, and creating simulation tools have many different applications, and we would be glad if the work in this thesis inspired others to use similar approaches to study other problems.

[1] It should here be mentioned that in Python one can often use somewhat simple tricks of e.g. using a list of both integers and floats, which does not make the code directly translatable, but there should exist similar equivalent libraries to get the same straightforward implementation. One can implement the same parallelization scheme as we have done with Joblib by using OpenMP for instance.

[2] We modelled these people using social distancing by not allowing them them to move. This was achieved by giving them a large mass such that particles bounced off them almost like a wall interaction. We also used a collision rule where we never let the people at rest achieve a non-zero velocity.

# Chapter 6

# Conclusion

To conclude, the work done in this master's thesis has been mainly to implement and verify two simulation methods to study many-particle systems, with a set of particles colliding in a cubic box. The main objective was to create an event driven simulation in order to perform molecular dynamics simulations. In addition, we have used Langevin equations to approximate the dynamics of different types of particles in order to see if we can get the same type of results for Langevin dynamics as we did for the event driven simulation. The results presented in chapter 4 show an excellent agreement with theoretical predictions, both for the case of speed distributions, the evolution of granular temperature and how the MSD evolves in time for both a molecular and a granular gas. The Langevin equations of interest to model a molecular and a granular gas were the underdamped Langevin equation and UDSBM respectivly. We solved these SDEs describing Brownian motion by implementing the simple, yet powerful, Euler-Maruyama method. By comparing the use of an event driven simulation and the numerical solution to SDEs we see that both methods seem to capture the correct dynamics for both a molecular and a granular gas. One of the major accomplishments has been to verify that a granular gas exhibits what is known as anomalous diffusion, while a molecular gas exhibits normal diffusion as predicted by Einstein in [23]. As a result of the agreement between numerical values and theory we infer that the reduction of a molecular dynamics simulation to a series of successive pairwise particle collision is very useful tool to study many-particle systems for the packing fractions studied in this project based on predictions from kinetic theory. The implementation of both RBC and PBC makes it possible to choose the most appropriate for different applications. Based on the results achieved throughout this thesis we are aware that one should prefer PBC to RBC, in order to achieve results representing kinetic theory for granular gases, which is derived without considering boundary conditions.

In comparison with the code used for the specialization project [1], the code used for the master's thesis is a vast improvement. The choice of studying three-dimensional systems with PBC instead of two-dimensional systems with RBC has made a great impact for the agreement between simulation results and theoretical predictions. By making it possible to do simulations in parallel we also explore some of the strengths of modern computers, namely the number of available cores. The use of HPC has also been an educational experience, as it is natural to use HPC to conduct simulations of systems with a higher number of particles than we have used for this thesis. In general this experience has introduced a number of different interesting topics, which was unknown prior to this project, such as granular gases, the use of SDEs and the need for general structured code which can be utilized by others if desirable. It is easy to implement a new and different event driven simulation using the same structure as we have done, inspired by the flow chart in Figure 3.1.

As we did not study the MSD of a granular gas in [1] it is difficult to know if it was possible to obtain good results for a granular gas with the implemented simulation at that time. Compared to the work in [1] the work done in this thesis is more complete, both in terms of the implementation

and the theoretical predictions used to compare with the numerical results. However, it must also be noted that the work done in [1] was a foundation which the current implementation and thesis greatly benefited from.

While we have only looked at a few topics for a granular gas, some of which who have been studied in detail by many authors, we do acknowledge that in order to study some of the more complex topics, such as cluster formation and long term behaviour of granular gases, one needs to have a great understanding of some of the more basic concepts. As per the intention of this thesis, we wanted to build from scratch simulation tools which are used by several, but not discussed in great detail. Considering this, we have succeeded both in creating a reliable and varied simulation tool, and in understanding and learning about new interesting topics. In that regard, we hope that this thesis can inspire others to work in the interesting field of granular materials, or to use event driven simulations to study different topics, may it be granular gases or other such as the spread of disease. The work done for this thesis can also be used as a basis for a further study in the area of granular gas dynamics by looking at the suggested improvements and further topics discussed in chapter 5.

# References


[1] Aleksander Gjersvoll. "Granular gas dynamics". Specialization project, Department of Physics, NTNU. Unpublished, but can be provided upon request. Dec. 2019.

[2] Stefan Luding and Thorsten Pöschel. *Granular Gases*. Springer, 2001.

[3] Vicente Garzó. *Granular Gaseous Flows : A Kinetic Theory Approach to Granular Gaseous Flows*. 1st ed. Cham: Springer International Publishing : Imprint: Springer, 2019.

[4] R. A Bagnold. *The Physics of Blown Sand and Desert Dunes*. Dordrecht: Springer Netherlands, 1974.

[5] Thorsten Pöschel and Thomas Schwager. *Computational granular dynamics: Models and algorithms*. Springer, 2005.

[6] P. G. de Gennes. "Granular matter: a tentative view". In: *Rev. Mod. Phys.* 71 (2 Mar. 1999), S374–S382. DOI: 10.1103/RevModPhys.71.S374.

[7] Igor S. Aranson and Lev S. Tsimring. "Patterns and collective behavior in granular media: Theoretical concepts". In: *Rev. Mod. Phys.* 78 (2 June 2006), pp. 641–692. DOI: 10.1103/RevModPhys.78.641.

[8] Xiaobo Nie, Eli Ben-Naim, and Shiyi Chen. "Dynamics of Freely Cooling Granular Gases". In: *Phys. Rev. Lett.* 89 (20 Oct. 2002), p. 204301. DOI: 10.1103/PhysRevLett.89.204301.

[9] I. Goldhirsch and G. Zanetti. "Clustering instability in dissipative gases". In: *Phys. Rev. Lett.* 70 (11 Mar. 1993), pp. 1619–1622. DOI: 10.1103/PhysRevLett.70.1619.

[10] PC Hemmer. *Termisk fysikk*. 2nd ed. Fagbokforlaget Vigmostand & Bjørke AS, 2002.

[11] Nikolai V. Brilliantov and Thorsten Pöschel. *Kinetic Theory of Granular Gases*. Oxford University Press, 2004.

[12] Alain Barrat and Emmanuel Trizac. "Molecular dynamics simulations of vibrated granular gases". In: *Phys. Rev. E* 66 (5 Nov. 2002), p. 051303. DOI: 10.1103/PhysRevE.66.051303.

[13] Prasenjit Das, Sanjay Puri, and Moshe Schwartz. "Clustering and velocity distributions in granular gases cooling by solid friction". In: *Phys. Rev. E* 94 (3 Sept. 2016), p. 032907. DOI: 10.1103/PhysRevE.94.032907.

[14] Wen-Guang Wang et al. "Experimental and numerical study on energy dissipation in freely cooling granular gases under microgravity". In: *Chinese Physics B* 27.8 (Aug. 2018), p. 084501. DOI: 10.1088/1674-1056/27/8/084501.

[15] Anna Bodrova et al. "Quantifying non-ergodic dynamics of force-free granular gases". In: *Phys. Chem. Chem. Phys.* 17 (34 2015), pp. 21791–21798. DOI: 10.1039/C5CP02824H.

[16] D. C Rapaport. *The Art of Molecular Dynamics Simulation*. 2nd ed. Cambridge University Press, 2004.

[17] Christian Scholz and Thorsten Pöschel. "Velocity Distribution of a Homogeneously Driven Two-Dimensional Granular Gas". In: *Phys. Rev. Lett.* 118 (19 May 2017), p. 198003. DOI: 10.1103/PhysRevLett.118.198003.

[18] Stefan Luding and Sean McNamara. "How to handle the inelastic collapse of a dissipative hard-sphere gas with the TC model". In: *Granular Matter*. Vol. 1. Springer, 1998, pp. 113–128.

[19] Sean McNamara and W. R. Young. "Inelastic collapse in two dimensions". In: *Phys. Rev. E* 50 (1 July 1994). DOI: 10.1103/PhysRevE.50.R28.

[20] Patrick Billingsley. *Probability and Measure*. 3rd ed. Wiley, 1995.

[21] Jens O. Andersen. *Introduction to Statistical Mechanics*. Fagbokforlaget Vigmostand & Bjørke AS, 2012.

[22] P. K. Haff. "Grain flow as a fluid-mechanical phenomenon". In: *Journal of Fluid Mechanics* 134 (1983), pp. 401–430. DOI: 10.1017/S0022112083003419.

[23] Albert Einstein. "Über die von der molekularkinetischen Theorie der Wärme geforderte Bewegung von in ruhenden Flüssigkeiten suspendierten Teilchen. (German) [On the Motion of Small Particles Suspended in Liquids at Rest Required by the Molecular-Kinetic Theory of Heat]". In: *Annalen der Physik* 322.8 (1905), pp. 549–560. DOI: https://doi.org/10.1002/andp.19053220806.

[24] Christian Henrique, George Batrouni, and Daniel Bideau. "Diffusion process in Two-Dimensional Granular Gases". In: *Granular Gases*. Springer, 2001. Chap. I, pp. 140–149.

[25] Anna S Bodrova et al. "Underdamped scaled Brownian motion:(non-) existence of the overdamped limit in anomalous diffusion". In: *Scientific reports* 6 (2016), p. 30520. DOI: https://doi.org/10.1038/srep30520.

[26] Ralf Metzler et al. "Anomalous diffusion models and their properties: non-stationarity, non-ergodicity, and ageing at the centenary of single particle tracking". In: *Phys. Chem. Chem. Phys.* 16 (44 2014), pp. 24128–24164. DOI: 10.1039/C4CP03465A.

[27] Anna Bodrova et al. "Intermediate Regimes in Granular Brownian Motion: Superdiffusion and Subdiffusion". In: *Phys. Rev. Lett.* 109 (17 Oct. 2012), p. 178001. DOI: 10.1103/PhysRevLett.109.178001.

[28] Anna S Bodrova et al. "Ultraslow scaled Brownian motion". In: *New Journal of Physics* 17.6 (June 2015), p. 063038. DOI: 10.1088/1367-2630/17/6/063038.

[29] S. C. Lim and S. V. Muniandy. "Self-similar Gaussian processes for modeling anomalous diffusion". In: *Phys. Rev. E* 66 (2 Aug. 2002), p. 021114. DOI: 10.1103/PhysRevE.66.021114.

[30] Hadiseh Safdari et al. "Aging underdamped scaled Brownian motion: Ensemble- and time-averaged particle displacements, nonergodicity, and the failure of the overdamping approximation". In: *Phys. Rev. E* 95 (1 Jan. 2017), p. 012120. DOI: 10.1103/PhysRevE.95.012120.

[31] Ioannis Karatzas and Steven E. Shreve. *Brownian Motion and Stochastic Calculus*. 2nd ed. Springer, 1998.

[32] R Kubo. "The fluctuation-dissipation theorem". In: *Reports on Progress in Physics* 29.1 (Jan. 1966), pp. 255–284. DOI: 10.1088/0034-4885/29/1/306.

[33] Don S. Lemons and Anthony Gythiel. "Paul Langevin's 1908 paper "On the Theory of Brownian Motion" ["Sur la théorie du mouvement brownien," C. R. Acad. Sci. (Paris) 146, 530–533 (1908)]". In: *American Journal of Physics* 65.11 (1997), pp. 1079–1081. DOI: 10.1119/1.18725.

[34] Peter E. Kloeden and Eckhard Platen. *Numerical Solution of Stochastic Differential Equations*. Springer, 1999.

[35] Domokos Szász. "Boltzmann's ergodic hypothesis, a conjecture for centuries?" In: *Studia Scientiarum Mathematicarum Hungarica* 31.1 (1996), pp. 299–322.

[36] Thomas H. Cormen et al. *Introduction to algorithms*. 3rd ed. The MIT Press, 2009.

[37] John C. Tannehill, Dale A. Anderson, and Richard H. Pletcher. *Computational Fluid Dynamics and Heat Transfer*. 2nd ed. Taylor & Francis, 1997.

[38] S. Miller and S. Luding. "Event-driven molecular dynamics in parallel". In: *Journal of Computational Physics* 193.1 (2004), pp. 306–316. ISSN: 0021-9991. DOI: https://doi.org/10.1016/j.jcp.2003.08.009.

# Appendices

# Appendix A

# Coefficient of restitution for an elastic collision in one dimension

One possible way of deriving the expression for the coefficient of restitution is to look at the equations for an elastic one-dimensional collision, the system illustrated in Figure 2.1. Conservation of momentum and kinetic energy gives the following relations

$$m_i v_i + m_j v_j = m_i v_i' + m_j v_j', \tag{A.1a}$$

$$\frac{1}{2} m_i v_i^2 + \frac{1}{2} m_j v_j^2 = \frac{1}{2} m_i v_i'^2 + \frac{1}{2} m_j v_j'^2, \tag{A.1b}$$

where we, as in the report, have used $'$ to denote the velocities after the collision. By rearranging terms in the conservation laws we get

$$m_i(v_i - v_i') = m_j(v_j' - v_j), \tag{A.2a}$$

$$m_i(v_i^2 - v_i'^2) = m_j(v_j'^2 - v_j^2). \tag{A.2b}$$

The expression in equation (A.2b) can also be written in the following way

$$m_i(v_i + v_i')(v_i - v_i') = m_j(v_j' - v_j)(v_j' + v_j). \tag{A.3}$$

By dividing the expression in (A.3) with the expression in (A.2a) we achieve the following relation for the velocities

$$v_i + v_i' = v_j' + v_j. \tag{A.4}$$

The expression in (A.4) can be rearranged to be given by the relative velocities in the following expression

$$v_j' - v_i' = -(v_j - v_i). \tag{A.5}$$

For an elastic collision, $\xi = 1$, and the expression for the coefficient of restitution can therefore be retrieved as the ratio of the relative velocities

$$\xi = 1 = -\frac{v_j' - v_i'}{v_j - v_i}. \tag{A.6}$$

# Appendix B

# Energy dissipation of an inelastic collision

An important characteristic of an inelastic collision is the amount of dissipated energy from a single collision between two particles. A priori we know that the dissipated energy should follow a few intuitive properties. We know that the expression for the energy dissipated is dependent on the coefficient of restitution, and the expression reduces to zero for a elastic collision, given by $\xi = 1$. In addition, we should expect to see that the energy dissipated is a result of the energy of the particles before the collision. The latter is simply a result of that we can only lose part of what we already have. The energy dissipated from a single particle-particle collision can be derived from the the post-collision velocities given in (2.10) and (2.11). If the particles have equal mass, we end up with the following collision rule

$$\boldsymbol{v}_i' = \boldsymbol{v}_i + \left( (1+\xi) \frac{\Delta \boldsymbol{v}_{ij} \cdot \Delta \boldsymbol{x}_{ij}}{2R_{ij}^2} \right) \Delta \boldsymbol{x}_{ij}, \tag{B.1a}$$

$$\boldsymbol{v}_j' = \boldsymbol{v}_j - \left( (1+\xi) \frac{\Delta \boldsymbol{v}_{ij} \cdot \Delta \boldsymbol{x}_{ij}}{2R_{ij}^2} \right) \Delta \boldsymbol{x}_{ij}. \tag{B.1b}$$

The kinetic energy of the particles before the collision is given as

$$E = \frac{1}{2} m (v_i^2 + v_j^2), \tag{B.2}$$

while the kinetic energy after the collision is given as

$$E' = \frac{1}{2} m (v_i'^2 + v_j'^2). \tag{B.3}$$

The difference in the energy prior to and after the collision is equal to the dissipated energy $\Delta E'$. Hence we can give the following expression for the dissipated energy

$$\Delta E' = E' - E = \frac{1}{2} m (v_i'^2 + v_j'^2 - v_i^2 - v_j^2). \tag{B.4}$$

In order to simplify the expression in (B.4) we will need the expression for the square speed of the particles after the collision, $v_i'^2$ and $v_j'^2$. Starting from (B.1a) and (B.1b) we get the following expressions

$$v_i'^2 = v_i^2 + \Delta \boldsymbol{x}_{ij}^2 \left( (1+\xi) \frac{\Delta \boldsymbol{v}_{ij} \cdot \Delta \boldsymbol{x}_{ij}}{2R_{ij}^2} \right)^2 + (1+\xi) \frac{\Delta \boldsymbol{v}_{ij} \cdot \Delta \boldsymbol{x}_{ij}}{R_{ij}^2} (\boldsymbol{v}_i \cdot \Delta \boldsymbol{x}_{ij}), \tag{B.5a}$$

$$v_j'^2 = v_j^2 + \Delta \boldsymbol{x}_{ij}^2 \left( (1+\xi) \frac{\Delta \boldsymbol{v}_{ij} \cdot \Delta \boldsymbol{x}_{ij}}{2R_{ij}^2} \right)^2 - (1+\xi) \frac{\Delta \boldsymbol{v}_{ij} \cdot \Delta \boldsymbol{x}_{ij}}{R_{ij}^2} (\boldsymbol{v}_j \cdot \Delta \boldsymbol{x}_{ij}). \tag{B.5b}$$

By inserting the square speeds in (B.5a) and (B.5b) into (B.4) we get the following result for the dissipated energy

$$\begin{aligned}
\Delta E' &= \frac{1}{2} m \left( 2\Delta \boldsymbol{x}_{ij}^2 \left( (1+\xi) \frac{\Delta \boldsymbol{v}_{ij} \cdot \Delta \boldsymbol{x}_{ij}}{2R_{ij}^2} \right)^2 + (1+\xi) \frac{\Delta \boldsymbol{v}_{ij} \cdot \Delta \boldsymbol{x}_{ij}}{R_{ij}^2} (\boldsymbol{v}_i \cdot \Delta \boldsymbol{x}_{ij} - \boldsymbol{v}_j \cdot \Delta \boldsymbol{x}_{ij}) \right) \\
&= \frac{1}{2} m \left( 2\Delta \boldsymbol{x}_{ij}^2 \left( (1+\xi) \frac{\Delta \boldsymbol{v}_{ij} \cdot \Delta \boldsymbol{x}_{ij}}{2R_{ij}^2} \right)^2 - (1+\xi) \frac{(\Delta \boldsymbol{v}_{ij} \cdot \Delta \boldsymbol{x}_{ij})^2}{R_{ij}^2} \right) \\
&= \frac{1}{2} m \left( (1+\xi) \left( \frac{\Delta \boldsymbol{v}_{ij} \cdot \Delta \boldsymbol{x}_{ij}}{R_{ij}} \right)^2 \left( \frac{1}{2}(1+\xi) - 1 \right) \right) \\
&= -\frac{1}{4} m \left( (1-\xi^2) \left( \frac{\Delta \boldsymbol{v}_{ij} \cdot \Delta \boldsymbol{x}_{ij}}{R_{ij}} \right)^2 \right),
\end{aligned} \tag{B.6}$$

where the expressions in (2.2) has been used to simplify the end result.

As a verification of the expression for the dissipated energy in (B.6) we must see if the expected properties are present in the result. The first property which stands out is that for an elastic collision, with $\xi = 1$, we get $\Delta E' = 0$. We can also see that for an inelastic collision, where $\xi < 1$, we obtain $\Delta E' < 0$, indicating that energy has dissipated. The last term gives the square component of the relative velocity, $\Delta \boldsymbol{v}_{ij}$ along $\boldsymbol{n}$, where we refer to Figure 2.3 for an illustration. The relative velocity is of the same order as the average velocity [see 22, p. 410]. The previous argument for a system of particles with equal mass makes it possible to state that

$$\Delta E' \propto (1-\xi^2) E. \tag{B.7}$$

The statement in (B.7) is used in the derivation of the evolution of energy for a granular gas, commonly known as Haff's law (see section 2.6), where we use that the average dissipated energy for a collision is proportional to the average particle energy, and thus the granular temperature.

# Appendix C

# The velocity autocorrelation function of UDSBM

Here we will derive the velocity autocorrelation function of UDBSM, which is used to derive an expression for the MSD of particles where the dynamics is determined by the second order ODE in (2.106). The derivation presented here is similar as the one given for the underdamped Langevin equation presented in the chapter 2, but is more complex due to the time-dependent coefficients present in UDSBM compared to the underdamped Langevin equation. UDSBM is given by the following ODE for the velocity

$$\frac{dv(t)}{dt} = -\gamma(t)v(t) + \sqrt{2D(t)}\gamma(t)\Gamma(t) \tag{C.1}$$

Eq. (C.1) can be handled as a linear ODE, where the help variable $H$ is given as

$$H = \int_0^t dt'(-\gamma(t)) = -\gamma_0 \int_0^t \frac{dt'}{1 + t'/\tau_0} = -\gamma_0\tau_0 \ln(1 + t/\tau_0).$$

Multiplying (C.1) with the integration factor $\exp(-H) = (1 + t/\tau_0)^{\gamma_0\tau_0}$ gives us

$$\frac{dv(t)}{dt}(1 + t/\tau_0)^{\gamma_0\tau_0} + \gamma_0 v(t)(1 + t/\tau_0)^{\gamma_0\tau_0 - 1} = \sqrt{2D(t)}\gamma(t)\Gamma(t)(1 + t/\tau_0)^{\gamma_0\tau_0}, \tag{C.2}$$

where we now see that the left hand side of (C.2) is equal to $\frac{d}{dt}(v(t)(1 + t/\tau_0)^{\gamma_0\tau_0})$. We can trus rewrite (C.2) into the following form

$$\frac{d}{dt}[v(t)(1 + t/\tau_0)^{\gamma_0\tau_0}] = \sqrt{2D(t)}\gamma(t)\Gamma(t)(1 + t/\tau_0)^{\gamma_0\tau_0}. \tag{C.3}$$

Integrating (C.3) with $v(t = 0) = v_0$ as an initial condition leads to the following expression

$$v(t)(1 + t/\tau_0)^{\gamma_0\tau_0} - v_0 = \int_0^t dt' \sqrt{2D(t')}\gamma(t')\Gamma(t')(1 + t'/\tau_0)^{\gamma_0\tau_0}. \tag{C.4}$$

The expression in (C.4) can be rewritten as a solution for the velocity given as

$$v(t) = v_0(1 + t/\tau_0)^{-\gamma_0\tau_0} + (1 + t/\tau_0)^{-\gamma_0\tau_0} \int_0^t dt' \sqrt{2D(t')}\gamma(t')\Gamma(t')(1 + t'/\tau_0)^{\gamma_0\tau_0}. \tag{C.5}$$

Again, due to the random term we can not give an analytical expression for $v(t)$. Using the moments of $\Gamma(t)$ in (2.88) we can still derive, as for the underdamped Langevin euqation, an expression for the

velocity autocorrelation function. The velocity autocorrelation function $\langle v(t_1)v(t_2)\rangle$ is then found by using the expression in (C.5) giving

$$\begin{aligned}\langle v(t_1)v(t_2)\rangle =&\langle\Big[v_0(1+t_1/\tau_0)^{-\gamma_0\tau_0}+(1+t_1/\tau_0)^{-\gamma_0\tau_0}\int_0^{t_1}dt'\sqrt{2D(t')}\gamma(t')\Gamma(t')(1+t'/\tau_0)^{\gamma_0\tau_0}\Big]\\&\cdot\Big[v_0(1+t_2/\tau_0)^{-\gamma_0\tau_0}+(1+t_2/\tau_0)^{-\gamma_0\tau_0}\int_0^{t_2}dt''\sqrt{2D(t'')}\gamma(t'')\Gamma(t'')(1+t''/\tau_0)^{\gamma_0\tau_0}\Big]\rangle.\end{aligned} \tag{C.6}$$

Due to the first moment of $\Gamma(t)$ in (2.88a) the cross terms are equal to zero, reducing (C.6) to

$$\begin{aligned}\langle v(t_1)v(t_2)\rangle =&\langle v_0^2\rangle(1+t_1/\tau_0)^{-\gamma_0\tau_0}(1+t_2/\tau_0)^{-\gamma_0\tau_0}+(1+t_1/\tau_0)^{-\gamma_0\tau_0}(1+t_2/\tau_0)^{-\gamma_0\tau_0}\\&\cdot\int_0^{t_1}dt'\int_0^{t_2}dt''\langle\Gamma(t')\Gamma(t'')\rangle\sqrt{2D(t')}\gamma(t')(1+t'/\tau_0)^{\gamma_0\tau_0}\sqrt{2D(t'')}\gamma(t'')(1+t''/\tau_0)^{\gamma_0\tau_0}.\end{aligned} \tag{C.7}$$

Inserting the second moment of $\Gamma(t)$ in (2.88b) into (2.99) and using the definition of the Dirac delta function in a integral simplifies the expression for the velocity autocorrelation function further. We obtain the following

$$\begin{aligned}\langle v(t_1)v(t_2)\rangle =&\langle v_0^2\rangle(1+t_1/\tau_0)^{-\gamma_0\tau_0}(1+t_2/\tau_0)^{-\gamma_0\tau_0}\\&+(1+t_1/\tau_0)^{-\gamma_0\tau_0}(1+t_2/\tau_0)^{-\gamma_0\tau_0}\int_0^{t_1}dt'2D(t')\gamma^2(t')(1+t'/\tau_0)^{2\gamma_0\tau_0},\end{aligned} \tag{C.8}$$

where we have chosen to use the Dirac delta for the integral of $dt''$ in order to ensure an expression for which $t_2 \geq t_1$ as needed for (2.59). Inserting the definition of $\gamma(t)$ from (2.54) and $D(t)$ from (2.51) we get the following integral

$$\begin{aligned}\langle v(t_1)v(t_2)\rangle =&\langle v_0^2\rangle(1+t_1/\tau_0)^{-\gamma_0\tau_0}(1+t_2/\tau_0)^{-\gamma_0\tau_0}\\&+2D_0\gamma_0^2(1+t_1/\tau_0)^{-\gamma_0\tau_0}(1+t_2/\tau_0)^{-\gamma_0\tau_0}\int_0^{t_1}dt'(1+t'/\tau_0)^{2\gamma_0\tau_0-3}.\end{aligned} \tag{C.9}$$

By performing the integral in (C.9) we get an expression for the velocity autocorrelation function given as

$$\langle v(t_1)v(t_2)\rangle = (1+t_1/\tau_0)^{-\gamma_0\tau_0}(1+t_2/\tau_0)^{-\gamma_0\tau_0}\left(\langle v_0^2\rangle+\frac{D_0\gamma_0^2\tau_0}{\gamma_0\tau_0-1}\left((1+t_1/\tau_0)^{2\gamma_0\tau_0-2}-1\right)\right). \tag{C.10}$$

If we choose the very convenient initial conditions where $\langle v_0^2\rangle = \frac{D_0\gamma_0^2\tau_0}{\gamma_0\tau_0-1}$, the expression in (C.10) simplifies to

$$\langle v(t_1)v(t_2)\rangle = \frac{D_0\gamma_0^2\tau_0}{\gamma_0\tau_0-1}(1+t_1/\tau_0)^{\gamma_0\tau_0-2}(1+t_2/\tau_0)^{-\gamma_0\tau_0}, \tag{C.11}$$

which is the expression used in (2.111) for the velocity autocorrelation function for UDSBM. The initial conditions coincide with the equipartition theorem for $\gamma_0\tau_0 \gg 1$. If we call the constant in (C.11) $T_0/m$ and insert $t_1 = t_2 = t$ we achieve a similar dependence on time as Haff's law in (2.40). Note that for the expression in (2.111) we have used that $D_0\gamma_0 = T_0/m$ compared to the expression in (C.11).

# Appendix D

# Computation of the time until a particle-particle collision

Computing if and when a particle will collide with the other particles is the most time consuming part of the implemented event driven simulation. A naive approach would for a given particle, iterate through all possible collisions partners and compute if and when the particles will collide. This would require a for-loop which would be done two times for each collision between two particles, which is a process occurring numerous times when one tries to study the case of particles colliding in a box. A more efficient approach can be computed by using vector operations, where we do operations on an entire array using third-party packages in Python made for scientific computing in Python, namely SciPy and NumPy. These packages provide wrappers to compiled and thus much faster programming languages which reduce run time and provide the opportunity to avoid certain for-loops, for instance the one presented in the naive approach.

The code utilized to compute the time until a particle collide with all other particles is given in Listing 1, and illustrates some of the elegant optimizations one can achieve by using Python. The function time_at_collision_particles is efficiently used to compute the quantity given in (2.20) by using NumPy arrays and using the SciPy package to compute the norm of a vector. The function belongs to the ParticleBox class, giving it access to the object parameters, which includes the position, velocity, mass and radius of all particles with the shape given in Table E.1. The function takes two input parameters, the particle number and the simulation time. The particle number is used to identify the parameters of the particle in focus and the simulation time is used to scale the time until collisions to the correct time in the simulation, for usage in the event driven simulation. The implementation is based on the following steps

- Compute $\Delta \boldsymbol{x}_{ij}$, $\Delta \boldsymbol{v}_{ij}$, $R_{ij}^2$, $\Delta \boldsymbol{v}_{ij}^2$, $\Delta \boldsymbol{x}_{ij} \cdot \Delta \boldsymbol{v}_{ij}$ and $d$ between the particle given by the input parameter and all other particles in the system.
- Assume that the particle does not collide with any other particle by creating an array where all elements are equal to $\infty$. This array is used to store the time until all collisions.
- Compute a boolean array which are true for the particles that the particle will collide with, i.e. to identify the particles where $d > 0$ and $\Delta \boldsymbol{x}_{ij} \cdot \Delta \boldsymbol{v}_{ij} < 0$.
- For all particles that the particle will collide with, identifiable from the boolean array, compute the earliest non-negative collision given in the last case of (2.20).
- Compute the time of collision in the simulation by adding the simulation time, provided as an input.

```python
import numpy as np
from scipy.linalg import norm

def time_at_collision_particles(self, particle_number, simulation_time):
        """
            Function that computes the time until a particle collides with all other particles
        :param particle_number: the index of a particle in order to retrieve and/or update the particle data
        :param simulation_time: is a float of the simulation time, used to get time for collisions in the simulation
        :return: the time when particle particle_number will collide with all of the other particles
        """
        # difference from particle particle_number to all other particles
        delta_x = self.positions - np.tile(self.positions[particle_number, :], reps=(len(self.positions), 1))
        # difference in velocity from particle particle_number to all other particles
        delta_v = self.velocities - np.tile(self.velocities[particle_number, :], reps=(len(self.velocities), 1))
        r_squared = (self.radii[particle_number] + self.radii) ** 2  # array of center to center distances
        dvdx = np.sum(delta_v*delta_x, axis=1)  # dot product between delta_v and delta_x
        dvdv = np.sum(delta_v*delta_v, axis=1)  # dot product between delta_v and delta_v
        d = dvdx ** 2 - dvdv * (norm(delta_x, axis=1) ** 2 - r_squared)  # help array quantity
        time_until_collisions = np.ones(self.N)*np.inf  # assume no particles is going to collide
        boolean = np.logical_and(dvdx < 0, d > 0)  # both these conditions must be valid particle-particle collision
        # check if there exist some valid particle-particle collisions for particle particle_number
        if np.sum(boolean) > 0:
            # compute time until collision
            time_until_collisions[boolean] = -1 * ((dvdx[boolean] + np.sqrt(d[boolean])) / (dvdv[boolean]))
        return time_until_collisions + simulation_time
```

**Listing 1:** Implementation of how to compute the time until a particle collides with all other particles in the system in Python.

Using the packages the computation of the quantities in the first step become very efficient. As seen in Listing 1, there is no need for any direct for-loop in Python. This computation is still the bottleneck of the simulation, but it is much more efficient than the introduced naive approach.

The approach used in Listing 1 can not directly be used for PBC as we need 27 copies of the system for different values of $k, l, m$ as discussed in section 3.4. The code for this procedure is similar as the one given in Listing 1, but with a few modifications. First we need to create an array of all the positions for all the particles in the 27 copies. This have been done with a simple for-loop through a set of offsets, given by unique values for $k, l, m \in [-1, 0, 1]$ and using the following expression for the position in the infinite system from (3.1). We then use the same procedure as described in Listing 1, but use the new set of positions instead of the original positions saved in the class. We must also take 27 copies of the $\Delta \boldsymbol{v}_{ij}$ and $R_{ij}^2$ since they are the same for all copied systems compared to particle $i$. By getting the correct shape and size of the arrays, the rest of the computation in Listing 1 is as before.

In the implementation we have a function that uses the computation functions before adding the collisions to the priority queue, and based on a simple boolean value it can use the version for either PBC or RBC for the computation of the time until future particle collisions. Note that even though there are more copies for a three-dimensional system than a two-dimensional system, the use of PBC can be considerably faster since there are fewer collision occurring due to the increased number of possible trajectories. For fewer collision, we have to compute if and when a particle collides with all other particles fewer times, which is the most time consuming part, thus making some simulations faster. This is naturally a function of the packing fraction of the system as well, but for some occasions we have experienced faster three-dimensional simulations than we experienced in two dimensions.

# Appendix E

# Numerical setup

In this appendix we present a detailed introduction to the numerical setup used for the event driven simulations and the numerical solution of SDEs describing Brownian motion. The appendix is a complement to chapter 3 and is most relevant for the interested reader. We will present all the parameters and variables used in the implementation of the two simulation methods, starting with the event driven simulation before moving on to the numerical solution of SDEs.

## E.1 Event driven simulation

The set of all parameters used in an event driven simulation is presented in Table E.1. We will now go through all parameters in detail and explain how they are used in the implementation in order to perform the event driven simulation presented in section 3.2. The problem indicator is used to choose which problem to study, e.g. system statistics, speed distributions or the MSD. The number of particles, the coefficient of restitution and the initial speed can be chosen freely. The radii of all the particles have been set to be equal for all the particles, the same procedure used as for the mass. The mass is automatically set equal to unity, while we must choose the radius of the particles. A set of initial positions and velocities are loaded from created initial values. For the simulation to work, is is important that the number of particles and the radius match some of the created initial values. If we want to look at a new system with a different number of particles or packing fraction, we must create a new set of initial values first. As initial values we use an equilibrium state of velocities or initial velocity vectors given by (3.3) and uniformly distributed positions in the box. The stopping criterion, output timestep, $t_c$, the number of cores and the number of runs are given as simulation parameters. The average number of collisions are updated during each collision in order for it to be used as a stopping criterion if desirable, while the average particle energy is only updated during each output. The collisions count, simulation time, the number of crossings, the positions of all the particles and the velocity of the involved particle(s) are updated at each valid collision. Note that the number of crossings are only updated for collisions with "walls", where the velocity of the particle is not updated for PBC, whereas it is never updated for RBC. Each element in the priority queue is given as the tuple presented in equation (3.2). Note that we have not given the choice of boundary conditions as a parameter. We have instead specified which boundary conditions to use for different applications. E.g. if you want to run simulations to compute the MSD, you will thus automatically use PBC. The choice of boundary conditions can however quickly be altered if desirable. Note that in the implementation we use elastic walls by default, where $\xi = 1$ is the collision rule with the wall regardless of the collision rule in the system.

The initial priority queue and the initial average particle energy is based on initial positions and velocities. When the simulation lets time evolve the queue is expanded with new possible collisions after each valid collision. One should note that the number of elements in the queue, $n_q$, is not

**Table E.1:** Table containing all the parameters and variables in an event driven simulation. The collision count, priority queue, the number of crossings, average number of collisions, and simulation time are not given by an initial value, but updated during each valid collision. In the implementation we have used an output timestep, $\Delta t$, to decide the time between outputs from the simulation. The average particle energy is updated for each output. During a collision all positions and the velocity of the involved particle(s) are updated. In the table, $^\mathsf{T}$ is used to indicate that we use the transpose of the given matrices in order to have the indicated shape. Only one stopping criterion is used for a given simulation, but it can be based on time, the average number of collisions or the average particle energy in the system. The problem indicator, $p$, is a help variable used to decide which problem to study, e.g. system statistics, MSD or speed distribution.

| Parameter | Symbol | Shape |
|---|---|---|
| Problem indicator | $p$ | 1 |
| Number of particles | $N$ | 1 |
| Coefficient of restitution | $\xi$ | 1 |
| Initial speed | $v_0$ | 1 |
| Radii | $[r_1, \ \ldots, r_N]$ | $(N)$ |
| Masses | $[m_1, \ \ldots, m_N]$ | $(N)$ |
| Stopping criterion | $\bar{c}_{stop}$, $t_{stop}$ or $\langle E \rangle_{stop}$ | 1 |
| Output timestep | $\Delta t$ | 1 |
| Duration of contact | $t_c$ | 1 |
| Number of cores | $n_c$ | 1 |
| Number of runs | $n_r$ | 1 |
| Simulation time | $t$ | 1 |
| Average number of collisions | $\bar{c}$ | 1 |
| Average particle energy | $\langle E \rangle$ | 1 |
| Collision count | $[c_1, \ \ldots, c_N]$ | $(N)$ |
| Positions | $\begin{bmatrix} x_1 & \ldots & x_N \\ y_1 & \ldots & y_N \\ z_1 & \ldots & z_N \end{bmatrix}^\mathsf{T}$ | $(N, 3)$ |
| Velocities | $\begin{bmatrix} v_{x1} & \ldots & v_{xN} \\ v_{y1} & \ldots & v_{yN} \\ v_{z1} & \ldots & v_{zN} \end{bmatrix}^\mathsf{T}$ | $(N, 3)$ |
| Number of crossings | - | $(N, 3)$ |
| Priority queue | - | $(n_q)$ |

an indication one can use to say something about the system other than that as time evolves there exist a lot of possible collisions. The parameters describing all particles have a shape where we easily can update and extract the desired particle parameters by using the particle number as the index. This is necessary to update the correct particle velocities during collisions and to update the collision count. Note that in order to make sure that the system evolves as expected we have implemented a feature such that the simulation provides output with a resolution in time given by the output timestep $\Delta t$. The output consist of the simulation time, the number of elements in the priority queue, the average particle energy and the average number of collisions. This feature is also used to compute quantities such as MSD with a constant time resolution. Choosing an output timestep equal to zero will make the MSD study use the logarithmically spaced times output version discussed in section 3.2.

## E.2 Numerical solution of SDEs

The set of all parameters used in order to solve SDEs describing Brownian motion with the Euler-Maruyama method is given in Table E.2. As discussed in section 2.9 the dynamics of a molecular and a granular gas can be given as two different Langevin equations, the underdamped Langevin equation in (2.87) and UDSBM in (2.106) respectively. Both these SDEs can be solved numerically with the same iterative scheme given in (2.86), where the two schemes differ from the time dependence of the diffusivity and the friction coefficient for UDSBM.

As before, we will go through the parameters in Table E.2 in order to explain how they are used in the implementation. Similarly as for the event driven simulation we use the problem indicator to choose which problem to solve[1]. We want to solve the SDE for a number of particles with a given mass, radius, initial speed, in a system with a given coefficient of restitution. In the implementation we have assumed that all particles are equal. When solving an SDE describing the dynamics of a particle, we are only looking at one particle. That particle can not see any other particles, and the dynamics is determined by a friction and a random term. We can thus use the same initial positions for all the particles, which have been chosen to be $\boldsymbol{X}_0 = [0.5, 0.5, 0.5]$. The power of vector operations let us do the iterative scheme for all particles at the same time in order to save time. Note that for the SDE we do not have to worry about any of the other particles or the boundaries of the system since we want to use PBC. The random force at each timestep is given as a Wiener process along each axis for each particle. We then update the position and the velocity from the general iterative Euler-Maruyama scheme for Brownian motion in (2.86) at each timestep. For a given initial speed we have chosen an initial velocity vector for each particle as given in (3.3). After we have reached the stopping criterion, given as a maximum time limit we can then compute the ensemble MSD in a similar fashion as for the event driven simulation.

Note that whereas the position and the velocities in the event driven simulation got updated during each collision, the position and the velocities in the numerical solution to an SDE is only updated each timestep and not between timesteps. In order to achieve result which show the theoretical predictions it is thus necessary to use such a low value for $\Delta t$ that the short time behaviour of the MSD becomes correct. For the results in chapter 4 we have used $\Delta t = 10^{-2}$.

[1] In the implementation solving SDEs is actually only one problem. The correct SDE is solved based on the choice of $\xi$.

**Table E.2:** Table containing all the parameters and variables used in order to solve SDEs describing Brownian motion with the Euler-Maruyama method. The positions and velocities at each timestep are updated as given in (2.86) for a three-dimensional system, while the simulation time is updated by adding $\Delta t$ at each timestep. The random force each timestep is approximated as a Wiener process. The information about the particles are only used to compute the diffusivity and the friction coefficient as a function of time. In the table, $^\mathsf{T}$ is used to indicate that one uses the transpose of the given matrices in order to have the indicated shape. The stopping criterion is given as a maximum time limit. Compared to earlier for the event driven simulation the position and velocities are only updated at each timestep and not between timesteps as well.

| Parameter | Symbol | Shape |
|---|---|---|
| Problem indicator | $p$ | 1 |
| Number of particles | $N$ | 1 |
| Coefficient of restitution | $\xi$ | 1 |
| Initial speed | $v_0$ | 1 |
| Radius of the particles | $r$ | 1 |
| Mass of the particles | $m$ | 1 |
| Stopping criterion | $t_{stop}$ | 1 |
| Timestep value | $\Delta t$ | 1 |
| Number of runs | $n_r$ | 1 |
| Time | $t$ | 1 |
| Positions | $\begin{bmatrix} X_{x1} & \dots & X_{xN} \\ X_{y1} & \dots & X_{yN} \\ X_{z1} & \dots & X_{zN} \end{bmatrix}^\mathsf{T}$ | $(N, 3)$ |
| Velocities | $\begin{bmatrix} Y_{x1} & \dots & Y_{xN} \\ Y_{y1} & \dots & Y_{yN} \\ Y_{z1} & \dots & Y_{zN} \end{bmatrix}^\mathsf{T}$ | $(N, 3)$ |
| Random force each timestep | $\begin{bmatrix} \Delta W_{x1} & \dots & \Delta W_{xN} \\ \Delta W_{y1} & \dots & \Delta W_{yN} \\ \Delta W_{z1} & \dots & \Delta W_{zN} \end{bmatrix}^\mathsf{T}$ | $(N, 3)$ |

# Appendix F

# MSD of event driven simulations on a non-logarithmic scale

In order to illustrate the asymptotic behaviour of the MSD in the study of Brownian motion for both a molecular and a granular gas in chapter 4 we used a logarithmic scale. Logarithmic scales have both advantages and disadvantages we would like to discuss. The most common use of logarithmic scales is to look at quantities evolving over different order of magnitudes. For which we can use to capture both short and long term behaviour of the MSD as a function of time. On the other hand we get a different impression of the results by looking at the plots in Figure F.1 and Figure F.2 compared to the equivalent plots on a logarithmic scale in Figure 4.11 and Figure 4.12. Although the correct long term behaviour can be seen from the plots in Figure F.1 and Figure F.2, it is not possible to see whether the short term behaviour of the MSD as a function of time is correct or not. In order to explain the difference between theoretical predictions and the numerical values we note that the MSD is plotted for a relatively long time, meaning that small errors in the measured quantities can increase as a function of time.

Let us first look at the results for a molecular gas. The long term behaviour of the MSD should be given as $6D_0t$ for a three dimensional system as seen from (2.68). We then get a linear trend with the slope $6D_0$. From Figure F.1 we see that the disagreement is a result of that the slope of the numerical values are slightly higher than $6D_0$, which makes the difference increase as a function of time. For the simulation results we achieved a slope of $\approx 0.364$ compared to the theoretical value $\approx 0.350$, giving a relative error of 4%. Thus we argue that we manage quite reasonable results for such a complex procedure as studying a many-particle system.

Finally, we want to give the same treatment to the results for a granular gas. We see from the Figure F.2 that we achieve similar behaviour as for a molecular gas. There is a disagreement arising in the region where we see the transition from the ballistic period to the logarithmic dependence. Then we see a slight increasing difference where the MSD is logarithmic. The logarithmic growth is given with the slope $6D_0\tau_0$ as seen from (2.79). For the simulation results for $\xi = 0.8$ we get a slope of $\approx 0.420$ compared to the theoretical value $\approx 0.451$, giving a relative error of 7%. Even for a higher relative error we do not see the same growing difference due to the logarithmic growth of the values. The higher error could be a result of more constants in the theoretical expression, making the expression more prone to possible errors in the estimation of such parameters from the simulation results.

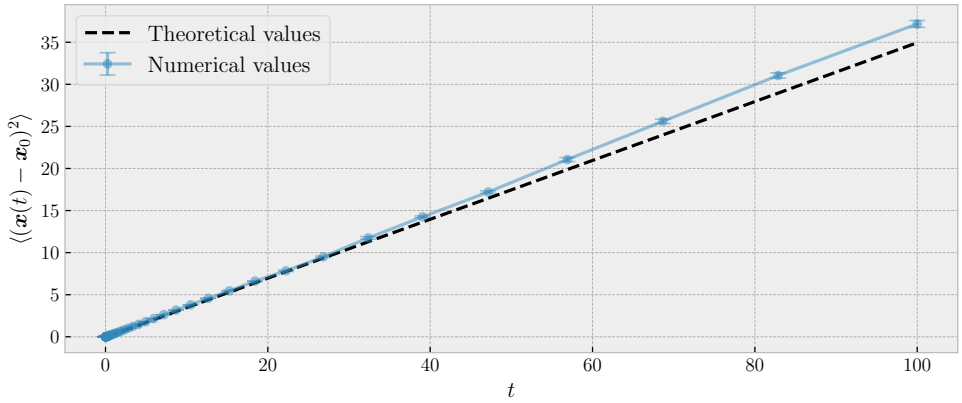



**Figure F.1:** Same plot as Figure 4.11, but with non-logarithmic axes.

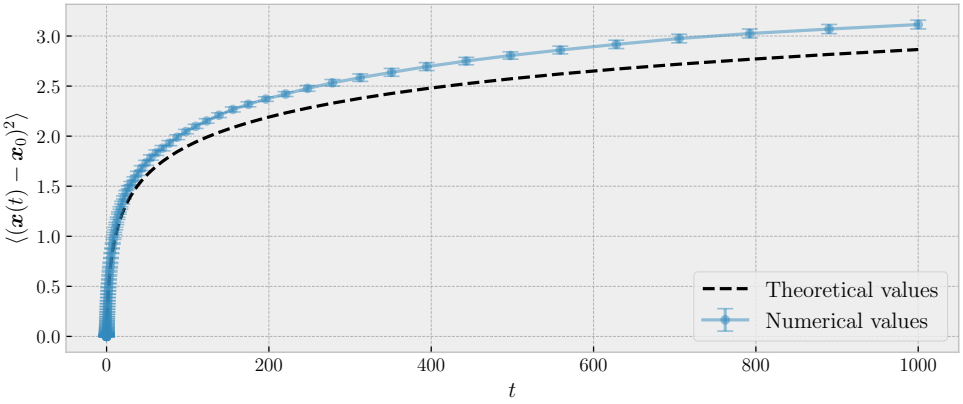



**Figure F.2:** Same plot as Figure 4.12, but with non-logarithmic axes.

# List of Symbols

Here we present a list of the most important symbols used in the report. In the list we give the symbol, what is represents and the page number where the symbol is first introduced.

$\langle E \rangle$ Average kinetic particle energy. 19

$\phi$ Azimuth angle used for polar and spherical coordinates. 17

$k_B$ Boltzmann constant. 16

$\tau_0$ Characteristic time decay of the granular temperature. 21

$\xi$ Coefficient of restitution. 7

$a$ Coefficient modelling the strength of the friction term in stochastic differential equations modelling Brownian motion. 30

$b$ Coefficient modelling the strength of the random term in stochastic differential equations modelling Brownian motion. 30

$\alpha$ Constant value used to illustrate general behaviour of expectation values and diffusion properties. 18

$g_2(\zeta)$ Contact value of the equilibrium correlation function for hard spheres. 21

$d$ Help quantity used in the computation of the time until a particle-particle collision. 12

$\zeta$ Diameter of the particles. 20

$\Delta \boldsymbol{x}_{ij}$ Vector from the center of particle $i$ to the center of particle $j$. 8

$\Delta \boldsymbol{v}_{ij}$ Relative velocity for particle $i$ and $j$. 8

$D(t)$ Self-diffusion coefficient, referred to as diffusion coefficient, or the diffusivity. 23

$R_{ij}$ Distance between the centers of the particles. 8

$t_c$ Duration of contact used in the TC model. 51

$\boldsymbol{F}_i$ Force acting upon particle $i$. 4

$\gamma(t)$ Friction coefficient, and the inverse velocity autocorrelation time, sometimes also referred to as the damping coefficient. 24

$H$ Help quantity used in the derivation of the velocity autocorrelation function for stochastic differential equations. 33

$\theta$ Inclination angle used for spherical coordinates. 18

$\Delta$ Lag time used in the computation of the time averaged mean squared displacement. 36

$L$ Length of the system. 39

$m_i$ Mass of particle $i$. 4

$\tau_c(t)$ Mean collision time. 21

$\mu$ Mean value of a distribution. 18

$\mathcal{N}(\mu, \sigma^2)$ Normal distributed parameter with mean value $\mu$ and variance $\sigma^2$. 31

$N$ Number of particles. 4

$n$ Number density. 20

$c_i$ Number of collisions for particle $i$. 55

$n_r$ The number of runs used in the simulations. 55

$k, l, m$ Set of unique integers giving the offset of one of the copies of the system used to implemented periodic boundary conditions. 47

$\eta$ Packing fraction of the system, also referred to as the particle volume density. 21

$x(t)$ Position in one dimension. 22

$\boldsymbol{x}_i(t)$ Position in multiple dimensions for particle $i$. 4

$X_t$ Position at time $t$ as a stochastic variable. 31

$P(v)$ Probability density function for the speed distribution. 17

$r_i$ Radius of particle $i$. 10

$\Gamma(t)$ Random term used to model the interactions with the Brownian particle from colliding with other particles. 30

$\sigma$ Standard deviation of a distribution. 18

$T$ Temperature. 16

$t$ Time. 4

$\Delta t$ Timestep value. Used to give the resolution in time for the output timestep of the event driven simulation and the timestep in the iterative schemes used to solve stochastic differential equation numerically. 31

$\Delta t^*$ Time until a collision for a particle with another object. 13

$v(t)$ Velocity in one dimension. Is also used to give the speed, denoted by dropping the time notation. 7

$\boldsymbol{v}_i(t)$ Velocity in multiple dimensions for particle $i$. 4

$f_v(v_x)$ Velocity distribution of the Maxwell-Boltzmann distribution. 16

$Y_t$ Velocity at time $t$ as a stochastic variable. 31

$V$ Volume of the system. 20

$W_t$ Wiener process. 31

Aleksander Gjersvoll

Simulation methods for granular gas dynamics in periodic three-dimensional systems

NTNU
Norwegian University of
Science and Technology