Multiscale Modeling in Granular Flow

by

Christopher Harley Rycroft

M.A., University of Cambridge, 2001
C.A.S.M., University of Cambridge, 2002

Submitted to the Department of Mathematics
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

September 2007

Author .....
Department of Mathematics
August 7, 2007

Certified by .....
Martin Z. Bazant
Associate Professor, Thesis Supervisor

Accepted by .....
Alar Toomre
Chairman, Applied Mathematics Committee

Accepted by .....
David Jerison
Chairman, Department Committee on Graduate Students

Multiscale Modeling in Granular Flow

by

Christopher Harley Rycroft

Submitted to the Department of Mathematics
on August 7, 2007, in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy

Abstract

Granular materials are common in everyday experience, but have long-resisted a complete theoretical description. Here, we consider the regime of slow, dense granular flow, for which there is no general model, representing a considerable hurdle to industry, where grains and powders must frequently be manipulated.

Much of the complexity of modeling granular materials stems from the discreteness of the constituent particles, and a key theme of this work has been the connection of the microscopic particle motion to a bulk continuum description. This led to development of the "spot model", which provides a microscopic mechanism for particle rearrangement in dense granular flow, by breaking down the motion into correlated group displacements on a mesoscopic length scale. The spot model can be used as the basis of a multiscale simulation technique which can accurately reproduce the flow in a large-scale discrete element simulation of granular drainage, at a fraction of the computational cost. In addition, the simulation can also successfully track microscopic packing signatures, making it one of the first models of a flowing random packing.

To extend to situations other than drainage ultimately requires a treatment of material properties, such as stress and strain-rate, but these quantities are difficult to define in a granular packing, due to strong heterogeneities at the level of a single particle. However, they can be successfully interpreted at the mesoscopic spot scale, and this information can be used to directly test some commonly-used hypotheses in modeling granular materials, providing insight into formulating a general theory.

Thesis Supervisor: Martin Z. Bazant
Title: Associate Professor

Acknowledgments

As I prepare to leave MIT, I am extremely grateful to the many wonderful people I have met during my time here, without whom this thesis would not have been possible. Firstly, I would like to express my heartfelt gratitude to my advisor, **Martin Bazant**, who is one of the most dedicated and hard-working people I have ever met, and throughout my time at MIT he was a constant source of energy, ideas, and enthusiasm. I am very grateful to fellow graduate students **Jaehyuk Choi** and **Kevin Chu** for their helpful discussions, and to **Pak-Wing Fok** and **Kevin Matulef** with whom I organised the Simple Person's Applied Math Seminar (SPAMS) during the 2005–2006 academic year. I must also not forget the ever charming and ebullient **Ken Kamrin**, and the latter half of this thesis could not have been done without the frequent discussions we had during the many granular conferences and workshops that we attended together.

Throughout my time at MIT, **Arshad Kudrolli** from Clark University provided a brilliant experimental perspective, that gave an oft-needed counter-balance to the atmosphere in the applied math department. I would also like to thank my other qualifiying and thesis committee members **Ruben Rosales**, **Dionisios Margetis**, and **Jean-Christophe Nave** for their numerous helpful suggestions. **Daniel Freedman** provided a lot of helpful advice as my first-year academic mentor, and was largely responsible for my original decision to come to MIT. The applied math administrative assistant, **Shirley Entzminger**, frequently went out of her way to help me out with organizational difficulties. Finally, I would like to thank **Jon Wilkening** at UC Berkeley and **Jim Langer** at UC Santa Barbara who provided much support and encouragement in my final months at MIT, as I brought this thesis to a close, and considered future directions.

I arrived in August 2002 with some trepidation, since it was my first time living in the US, and I knew hardly anyone. However, within several weeks I had made many excellent friends through social events at Tang Hall, including **Vikas Anant**, **Julia Cline**, **Natalija Jovanovic**, **Alexandra Kern**, **Karen Lee**, **Vivian Lei**, **Song-**

Hee Paik, **Mitch Peabody**, **Johnna Powell**, **Sarah Rodriguez**, **Kaity Ryan**, **Ming Tang**, **Katy Thorn**, and **Bruce Wu**. Although many of these people moved away over the following years, they formed the bedrock of my social experiences at MIT, and I could write pages about the many great times that we shared. Also members of this group were **Tyrone Hill**, **Stephen Kohen**, and **Shawn Kuo** who later became my roommates for four years at 374 Washington Street, and helped make it one of the most friendly and relaxed environments that any MIT graduate student could hope for.

As time went by in the first year, I also formed numerous other friendships. While still living in Tang Hall, I became close friends with **Miranda Newbery** and **Josep Dorca Luque** with whom I shared a British connection, and with whom I could share a laugh with about the American pronunciation of "tomato". Around that time, I also met **Tony Lau**, who makes truly awesome smoothies, and who convinced me that California really is the place to be. In the winter of 2003, I went sledding with **Hong Ma**, and subsequently enjoyed many crazy adventures with him, whether it was swimming down New Hampshire rivers, running around Manhattan solving puzzles in the middle of the night, or rowing home-made rafts along the Charles River.

At the start of my second year, I met the incredible **Emma Brunskill**, who became my running partner for four years; she was always overflowing with ideas on every subject, and she was the first to cultivate in me a deep appreciation for the many excellent ice cream stores in Cambridge. Several months later, I befriended **Maggie Lee** and **Rachel Sugal** from UMass Amherst, who frequently provided me with an interesting perspective outside the Boston and Cambridge bubble. In the summer, while hiking in Acadia, I met **Valentina Urbanek** who has been a great friend during my time at MIT. Around that time, I also met **Olivia Cheo**, who makes some of the best chocolate cookies that I have ever tasted, and **Caterina Schweidenback**, who is always overflowing with energy and enthusiasm. I went on many excellent hikes and attended a lot of dinner parties with my friends **Marcus Roper** and **Silas Alben** from Harvard University.

During the winter of my third year, I met **Grace Zheng**, who is one of the most

friendly people I've ever met, and has since become one of my closest friends at MIT. Around the same time, I also became got to know **Fumei Lam**, who shared my love for Apple computers and hardcore runs. During that year I became good friends with **Shan Wu** and **Amy Shi**, and also **Melinda Wong**, who shared my love of photography and independent cinema, and has the best taste in restaurants of anyone I know. During the spring, I became close friends with **Juan Montoya** and **Sushil Kumar**, and along with aforementioned Kaity Ryan we formed the "Fast Furious Transforms" running group. They were always furiously enthusiastic about going running, even if it was pouring with rain or freezing cold.

At the start of my fourth year, I met **Maria Hondele** on a hiking trip, who shared my love of skiing and the outdoors, and in the following winter, I became friends with Harvard Law graduate **Lauren Weldon**, and our interesting conversations changed my perspective on many things. In the summer of 2006, while in hiking Acadia National Park, I met my wonderful girlfriend **Yuhua Hu**. I am extremely grateful to her for her support, and for putting up with me during the many months in 2007 when I spent far too long locked up in my room compiling this thesis. At this point, it also seems appropriate to mention my polar bear **Snowy** who has accompanied me on many trips, and even made it to my thesis defense.

Throughout my time at MIT, my family have provided unwavering encouragement. Over the five years, I have exchanged countless phone calls and emails with my father **Stephen Rycroft**, my mother **Kathryn Potter** and my step-grandfather **Allan Gormley**. In particular I would like to thank my late grandmother, **Phyllis Gormley**. She was the only grandparent that I ever met, and I was her only grandson, which created a very special bond between us. Despite a sixty year age difference, we were able to communicate with a frankness and honesty that would be the envy of many. For a period of almost eight years during my undergraduate and graduate degrees, she wrote me beautifully handwritten letters almost every week, as she had told me that when she was at college, she had always appreciated "a little bit of home". Sadly, she was not around to see her grandson complete his studies at MIT, but I will never forget her uncompromising love and support.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Background

Despite being common in everyday experience, granular materials are poorly understood from a theoretical standpoint, and even in the simplest of situations, they can exhibit surprisingly complex behavior. The last twenty years has seen a resurgence of interest in them from the scientific and engineering community, motivated by the possibility that there is new physics to be discovered [59, 60, 10, 64, 34]. Granular materials are frequently viewed in connection with other systems featuring strong confinement (such as glasses and colloids) and thus the central challenges have a much broader applicability.

Granular materials cannot be characterised as a gas, liquid, or solid, and in certain situations, they can exhibit behavior characteristic of each of these three phases [60]. Perhaps some of the most successful work has been in situations where the granular material can be approximated by one of these phases. The behavior of dilute, collisional granular media has been explained using a version of Boltzmann's kinetic theory of gases, modified to account for inelastic collisions [61]. At the opposite limit, the modern methods of soil mechanics, such as Critical State Theory [118, 94], have had some successes in explaining the bulk stresses in a static granular assembly of densely packed particles.

This thesis is primarily concerned with an intermediate regime, of slow flow in

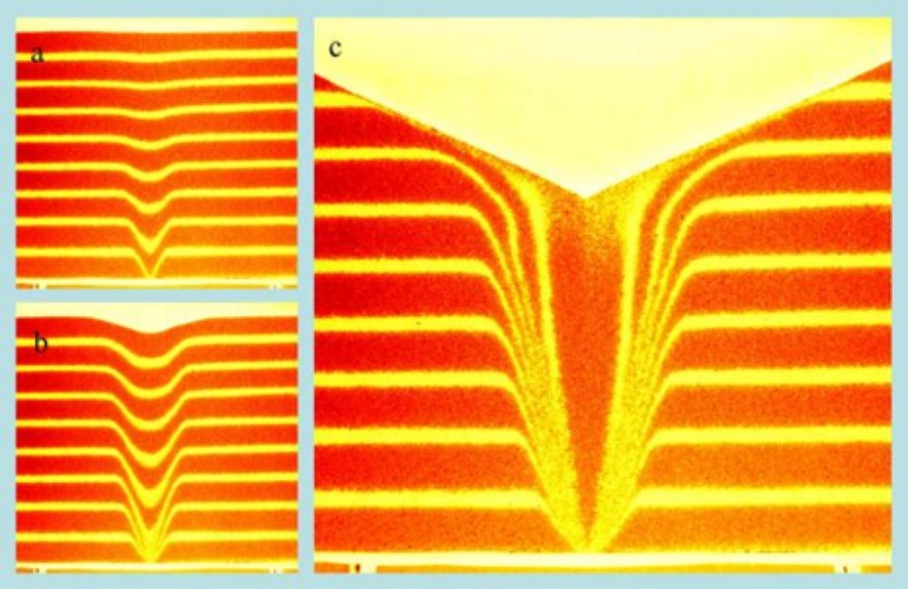


Figure 1-1: Three snapshots of a granular drainage experiment. Red and yellow particles are held between two parallel glass plates and drain from an orifice in the container base. (Samadani and Kudrolli, Clark University.)

dense granular materials. Two everyday examples of this would be drainage of sand in an hourglass, or the flow of seed through a hopper. In these flows, particles form a dense, amorphous packing, and since they interact inelastically, energy is rapidly lost in the system, meaning that individual particles have long-lasting, many-body contacts with their neighbors. Before flow starts, the packing is solid-like and supports stresses, but during flow, the material begins to rearrange and exhibit more liquid-like behavior. The interplay between these two different regimes is poorly understood.

At the microscopic level, particles in a dense granular flow exhibit complex behavior. Experiments have shown that there are large heterogeneities at the level of a single particle [31, 89, 19, 39, 56]. Stresses in granular materials tend not to be evenly distributed, and are often localized along chains of particles, which continually break and reform during flow [57, 139, 138]. This can create challenges for silo design, since the weight of the grains may be focussed on particular points on the silo wall. Even during flow, particles are frequently in long-lasting contact with their neighbors, and thus statistical mechanics approaches are generally invalid, since the system is in an athermal limit, far from equilibrium. Traditional statistical mechanics

assumptions, such as particle collisions occurring randomly from a thermal bath, are generally inappropriate due to strong neighbor correlations.

Given the discrete effects at the level of a single particle, one might expect that the flow profiles would exhibit features at this level too, such as cracks or dislocations. However, the macroscopic properties of slow, dense granular flow in most situations are smooth, and are frequently well-approximated by simple functions. Figure 1-1 shows several snapshots from a quasi-2D granular drainage experiment. The flowing region forms a parabola, and using particle image velocimetry, the mean flow in the vertical direction at different cross sections is well-approximated by a Gaussian; similar results have been seen in many other situations [84, 83, 114, 95, 137, 90, 91, 27]. In a rotating Couette shear cell, the velocity profile forms a boundary layer on the rotating wall, with exponential decay into the bulk [80, 87, 72, 21, 88]. In flow down an inclined plane, the velocity profile as a function of height follows an approximately polynomial dependence [107, 7, 18, 123, 126]. In a split-bottom Couette cell, the velocity profiles were well approximated by an error function [45], with the data being accurate enough to tell this apart from a hyperbolic tangent. It is tantalizing to wonder whether the flow in these different geometries can be explained by a single formalism, in the same way that the Navier-Stokes equations can describe a large class of fluid flows in arbitrary geometries. However, the variety of the functional forms, coupled with the lack of a good microscopic model, make the task of finding such a theory challenging. While there have been theories that explain a specific subset of these geometries, there is no well-accepted overreaching theory. The lack of such a theory is a hurdle to industry, where grains and powders are frequently manipulated.

There is perhaps cause to be optimistic that some of these key questions about granular materials may be answered in the near future. The last ten years has seen the development of many theoretical models for granular flow, with an diverse array of physical postulates, such as "granular eddies" [41], granular temperature-dependent viscosity [116], density-dependent viscosity [80, 21] "shear transformation zones" [43, 44, 75, 73, 74], and partial fluidization [8, 9]. None of these models could be considered a general theory, and typically only explain a narrow subset of possible geometries.

However, it may well be that a complete theory incorporates aspects from some of these current ideas.

The wealth of new theories has been coupled with an increase in the sophistication of experimental techniques. While granular materials are tangible, gaining pertinent experimental information about the details of flows poses many challenges. To observe stresses at the level of a particle, experimentalists have made use of photoelastic discs [82]. Another obvious problem in granular experiments is that it is difficult to visualize the bulk flow, although recently several techniques, such as magnetic resonance imaging [88], diffusing-wave spectroscopy [85], confocal microscopy [142], and index-matching with an interstitial fluid [136, 135, 122, 102], have been used to directly measure properties of flow in the bulk.

Simulation of granular materials presents another promising avenue of investigation. In the past five years, with the large scale deployment of parallel computer clusters in universities and research labs, it has become possible to carry out Discrete Element Method (DEM) simulations on a reasonable scale [1, 144, 129, 26, 123, 24, 48]. Although computationally intensive, the DEM simulations offer the ability to investigate many aspects of a granular flow in complete, three-dimensional detail, without any of the experimental difficulties, and they have made a number of important contributions.

1.2 Previous work at MIT before this thesis

Hopper drainage is perhaps one of the simplest examples of a dense granular flow, since it requires no external forcing and happens by gravity only. Over the past forty years, it has been extensively studied, and a number of continuum models [77, 90, 95, 108, 94, 92] have been proposed. However, these models primarily concentrated on the steady state mean flow, and certain practical situations arise in which one may be interested in the behavior of the constituent particles, such as how they mix and rearrange. Perhaps one of the most interesting examples is in the pebble-bed nuclear reactor, a schematic of which is drawn in figure 1-2. In the pebble-bed reactor, fuel

is inserted as spherical pebbles, which are slowly and continuously drained through a large cylindrical container. In some designs of reactors, including the one shown here, there are two types of pebbles – a core of reflecting moderator pebbles, surrounded by an annulus of fuel pebbles. For this type of design, diffusion of the fuel/moderator interface has considerable implications for reactor design and safety.

In addition to particle diffusion, some of the functional forms seen in granular flow experiments suggest that the velocity field itself may in part be explained by a diffusing quantity. In particular, the drainage experiments, with a gaussian velocity profile whose width scales according to the square root of height, suggest a solution motivated by a diffusion equation. As discussed in following chapter, this possibility was considered by several authors, who postulated the void model [76, 77, 78, 90, 25]. In this model, voids of free space are introduced at the orifice, and diffuse upwards, causing the particles to move correspondingly downwards. Despite its simplicity, the result has been shown to be in reasonably good agreement with the experimental results.

Motivated by this, Choi, Kudrolli, Rosales, and Bazant [28] carried out one of the first experimental studies of particle diffusion in granular drainage. Their results (summarized in section 2.2) show that the regime of slow, dense granular flow is governed by a distinctly non-thermal picture, where particles undergo long-lasting contacts with their neighbors, and the features of the flow are predominately governed by geometry and packing constraints. In particular, they observed that for a large range of hopper drainage experiments, altering the orifice size resulted in a change in the overall flow rate, but did not alter the geometry of the flow profile – the flow velocities were scaled by a constant factor. The amount of diffusion observed was very small, with the length scale of particle diffusion being two to three orders of magnitude smaller than the length scale associated with the overall width of the flow.

A natural next step is to ask whether these results could be explained by a mathematical model, and perhaps the only candidate in the literature is the void description. While originally proposed to explain the mean flow, the void model also gives a microscopic model for particle motion, which was not considered by the original authors.

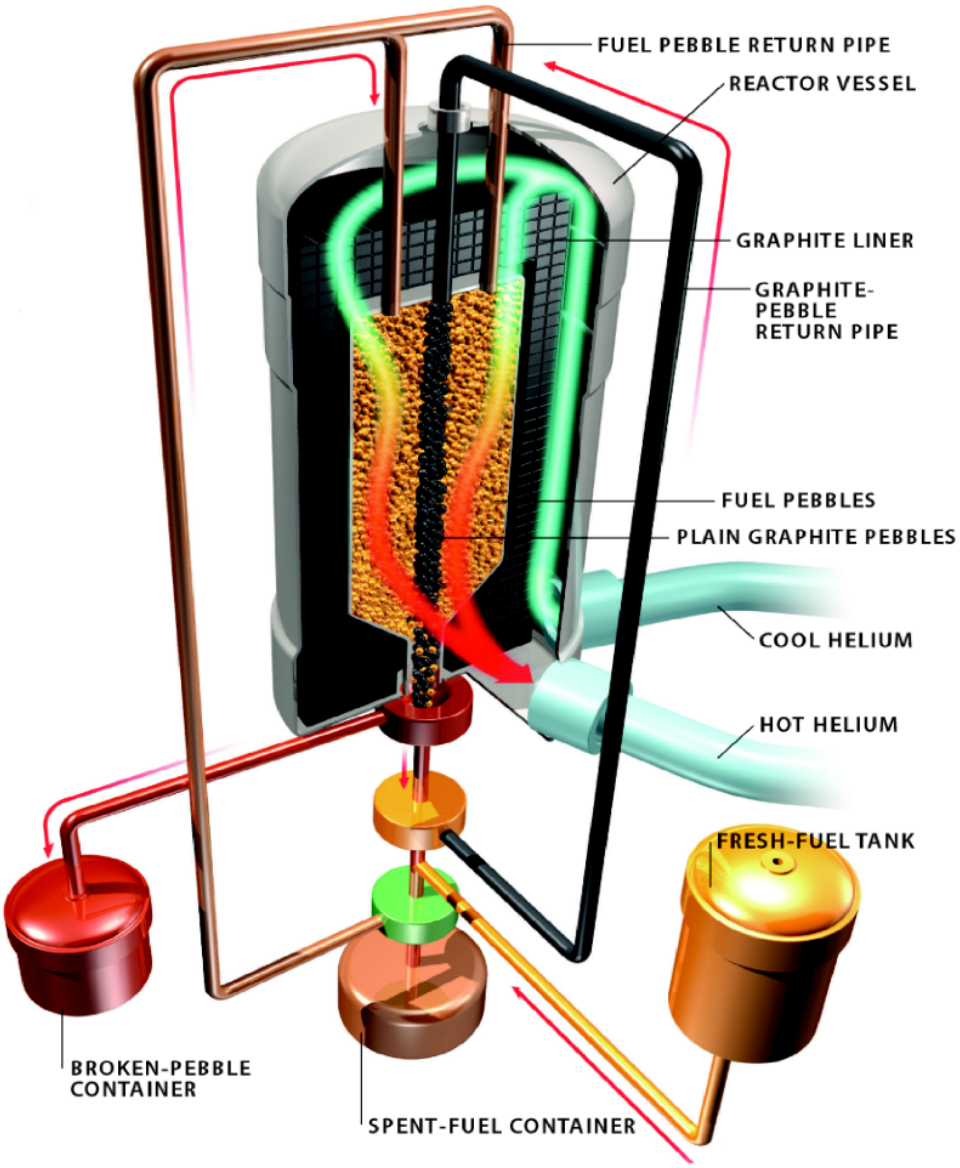


Figure 1-2: A schematic diagram of the pebble-bed nuclear reactor. The evolution of the interface between the two types of particles is critical to the reactor design.

However, despite its successes with the mean flow, the void model greatly overpredicts the amount of particle diffusion, since the length scale of particle diffusion is the same as the diffusive length scale of the flow profile. One of the main deficiencies of the void model is that it does not respect the packing geometry of the granular material: it allows individual particles to move independently of the rest of the packing, but in reality, a single particle may be strongly constrained by its neighbors.

Based on this idea, Bazant proposed the spot model [17] which provides a more realistic microscopic mechanism for particle motion, by moving particles in local groups, corresponding to the fact that they must move co-operatively. The spot model predicts similar mean flows to the void model, but reduces the particle diffusion by several orders of magnitude, consistent with the experimental evidence. It remains simple enough for mathematical analysis, and makes a number of physical predictions. While a complete flow may be made up of many spots, and thus a single spot cannot be directly observed, the theory predicts that spots would cause local, spatial velocity correlations. These were subsequently found by Choi in the granular drainage experiments, providing a validation of the theory.

The spot model also forms the basis of a simulation technique. Choi carried out two dimensional spot model simulations, which were able to capture many aspects of the flow in the hopper experiments, such as the velocity profiles. However, these preliminary simulations had one large flaw: even though the movements of an individual tracer particle appear physically reasonable, the packing arrangements that were generated would exhibit overlapping particles, particularly in regions of high shear. This became referred to as the "density problem". Several simple modifications to the spot model were investigated, such as the use of persistent random walks [55], spot rotations, or extra particle diffusion, and while these led improvements in the, none were successful in creating fully valid particle packings.

1.3 The contribution of this thesis

During the past decade, scientific computation has become increasingly important in applied mathematics, providing insight into many problems that would preclude a simple analytical description. Since granular materials exhibit a complicated interplay between discrete effects at the particle level, and continuum effects and a macroscopic level, they are ideally suited to being studied by computer. At the most general level, this thesis has made use applied computational methods from a variety of different perspectives to examine aspects of granular flow that would be difficult to address in either pure theory or experiment.

1.3.1 Discrete Element Simulation

While Choi was able to show the existence of velocity correlations in experiment, the results had a fundamental drawback that the measurements could only be taken of the particles which were pressed against the front and back walls of the silo, as there was no direct way to imaging the particle rearrangements in the bulk of the flow. As mentioned above, experimental techniques have been developed to probe into the bulk of the granular flow, but these usually provide limited information, and are unsuitable for this case where we would like to know in detail about microscopic rearrangements of the individual particles.

DEM simulation provides an ideal tool for this situation, since it gives complete information about all particles in a granular flow. Using a parallel discrete element code developed at Sandia National Laboratories (discussed in section 2.6), the author ran a full-size simulation of Choi's experimental geometry. With no fitting, the computed velocity profiles matched Choi's experimental results to a high degree of accuracy (section 2.7). Computing the velocity correlations at the surface of the packing yielded similar results to Choi's experiment. However, in the DEM simulation, it was possible to show that velocity correlations were also present in the bulk of the granular flow, validating the use of the spot model in fully three-dimensional granular packings.

1.3.2 Solution to the "density problem"

In addition to carrying out the large-scale DEM simulations above, the author also created a three-dimensional spot simulation of granular drainage. Initial studies of the spot model showed that the density problem was still present in three dimensions. However, the author demonstrated that it could be eliminated by modifying the spot model (discussed in chapter 3) to incorporate an elastic relaxation step, whereby after the bulk spot displacement, any overlapping particles experienced a small repulsion. Surprisingly, it was found that only a small, single-step correction was required to generate non-overlapping packings, and that the addition of this did not significantly alter any of the other aspects of the simulation. The author used this model as a basis for a multiscale simulation of granular drainage that matched many aspects of a corresponding Discrete Element simulation. Since the spot simulation suppresses much of the details of individual particle contacts, and concentrates on the packing geometry only, it required approximately a hundredth of the computational power of DEM, making it a promising technique for applications where one wishes to rapidly gain approximate answers about granular drainage problems.

1.3.3 How do random packings flow?

Random packings have received much attention in the literature, but most of the work has concentrated on the geometry of static assemblies [132, 81, 133, 36], with much of the current work focusing on the jamming transition [134, 99, 100, 37, 79]. However, little attention has been paid to how a random packing of particles will rearrange during flow. One of the most surprising results of the above simulations was that the spot model was able to recreate and track several important signatures of the random packing structure that were seen in DEM. The spot model with relaxation can be viewed as one of the first theoretical models of particle rearrangement in a flowing random packing, and an in-depth study of its stability was carried out. By moving spots around in a periodic box [104] the algorithm can also be used as a method of generating static random packings.

1.3.4 Additional studies of the spot model

Chapter 4 continues with several extensions to the basic spot model. Since spots are thought of as carrying free volume, it is interesting to track and compare the distribution of free space in the two simulations, and some important differences were observed. Carrying this out required the author to construct a three-dimensional Voronoi tessellation algorithm which is described in detail. Also in this chapter, two alternative spot models were considered, to investigate whether the simulation could accurately describe the free surface, and see whether it would work in a two dimensional crystalline flow. Finally, since the spot influence is local, it lends itself well to running on a parallel computer, and two different parallelization schemes were considered.

1.3.5 A simulation study of the pebble-bed reactor geometry

Concurrent with the spot simulation described above, the author carried out an detailed study of granular flow in a pebble-bed nuclear reactor, described in chapter 5. In collaboration with Sandia National Laboratories, several full-size DEM simulations of pebble-bed reactors were carried out, based on the MIT Modular Pebble Bed Reactor (MPBR) [130, 2] design. Unlike the spot model simulations, this study was primarily focused on asking questions of relevance to engineering, such as the residence-time distribution of pebbles draining through the reactor, although many of the results complemented the spot model studies described in other chapters. In addition to the full-size runs, several half-size simulations were carried out to look at the influence of wall friction, and to investigate the feasibility of a bidisperse pebble-bed reactor core.

1.3.6 A general model of dense granular flow

Despite the spot model's successes, it was unclear how to extend it to other geometries to form a general multiscale simulation technique. The notion of diffusing free volume seems very specific to granular drainage. However, Kamrin and Bazant [65, 66] found a method of relating the spot motion to granular stresses, and were able to extend the

applicability to several other geometries, including the Couette shear cell. In chapter 6 we carried out an in-depth test of the SFR to DEM simulations in both the silo and Couette geometries, making the SFR one of the first theoretical models to describe two types of geometry using the same formalism.

1.3.7 Measuring a granular continuum element

Despite the SFR's successes, the description of stresses that it made use of had several troubling features, and we wished to test these directly. However, given the complications of force chains at the local level, it was unclear whether there was any notion of a continuum stress at the particle level. In chapter 7 it was shown that stress, and other material quantities can be interpreted in an approximate sense at the spot scale even though they would be intractable at the level of a a single particle. By carrying out a variety of DEM simulations, and viewing the results as an ensemble of "granular elements" at the spot scale, it was possible to directly test the assumptions of Mohr-Coulomb plasticity at the local level.

Chapter 2

Diffusion and mixing in granular drainage

2.1 Introduction

Hopper drainage is perhaps one of the simplest examples of a dense granular flow, since it requires no external forcing and happens by gravity only. Over the past forty years, it has been extensively studied, and a number of continuum models [77, 90, 95, 108, 94, 92] have been proposed. However, these models primarily concentrated on the steady state mean flow. This chapter documents some early work that was carried out to study other aspects of granular drainage, particularly diffusion and mixing, which has relevance in a number of industrial situations.

2.2 Experimental motivation

The experiment shown in figure 1-1 made use of two different colors of particles. While the movement of the different colors allows us to visualize the mean flow, it also tells us something about diffusion. In frames (a) and (b), there appears to be very little mixing, but by the third frame, some of the colors appear to have blended together. Motivated by this work, Choi *et al.* carried out experiments to specifically address this issue [28]. They carried out experiments of glass beads of diameter $d = 3$ mm in

a quasi two dimensional hopper of size 20 cm $\times$ 2.5 cm $\times$ 2.5 cm, with aluminum side walls, and Plexiglas walls at the front and back. Drainage took place from a slit in the center of the container base, the width of which could be altered to change the overall flow rate. During the flow, the particles at the glass wall were imaged using a high-speed digital camera, capable of taking images at 1000 frames per second.

The steady state mean flow in the experiment is characterized by a parabolic converging region of flow above the orifice, that connects with roughly uniform flow higher up. The measurements of particle diffusion were made in the region of uniform flow, by imaging a $17d \times 87d$ region centered at a height of $150d$ above the slit. Three of their key conclusions were:

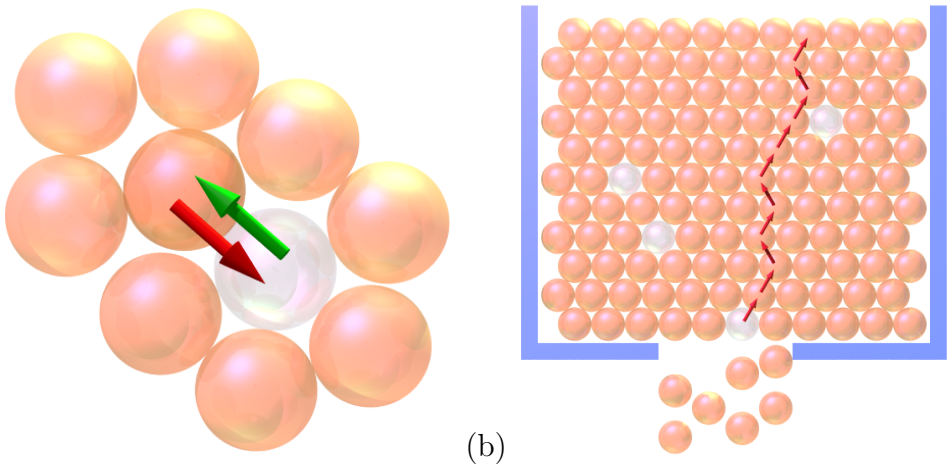
- Particle diffusion is extremely small. Péclet[1] numbers on the order of 300 were observed, meaning that a particle would have to fall twice the height of the silo before diffusing by a single particle diameter.

- The neighbors of a particle strongly persist: a particle at the top of the observation window will typically retain 90% of its neighbors by the bottom.

- Over a large range of different flow rates, the amount of particle diffusion collapsed if plotted as a function of distance dropped, and not time.

These results are very surprising: in gases and liquids, diffusion is facilitated by thermal fluctuations, and the amount of diffusion is proportional to the time. In these experiments, time appears to not be the most significant factor, and altering the flow rate gives the same results, but with time rescaled. It suggests that granular diffusion is distinctly athermal, and that the packing geometry of the particles is the most significant factor.

[1]The Péclet number characterises the ratio of advection to diffusion [105], and is typically defined as $\mathrm{Pe} = LV/D$, where V is the velocity, D is the diffusion constant, and L is a characteristic length scale. In Choi's experiments, this is defined as $\mathrm{Pe} = 2d\Delta h/\langle \Delta x^2 \rangle$, where Δh is the distance dropped, and Δx is the horizontal displacement.



(a) (b)

Figure 2-1: (a) The void model microscopic mechanism: particles movie by exchanging position with voids. (b) The void model for granular flow: particles are imagined as lying in a two dimensional lattice, and move in response to voids diffusing upwards from the container orifice.

2.3 The void model

To explain these results requires a microscopic model for particle motion, and perhaps the only candidate in the literature is the concept of a void, shown in figure 2-1(a). The void mechanism has a long history, and was proposed by Eyring for viscous flow [42] but it has re-appeared in many other situations such as the glass transition [29], shear flow in metallic glasses [128], and compaction in vibrated granular materials [23].

The idea was first postulated in the context of granular materials by Litwiniszyn [77], and was subsequently studied by several other authors [90, 95, 137, 25] (with some related work making use of cellular automata models [15, 16]). Particles are imagined as being on a two-dimensional hexagonal lattice, as shown in figure 2-1(b). The flow is created by introducing voids at the orifice which propagate up through the packing according to a random walk, moving up-left or up-right with equal probability. To find the mean flow in this case, we introduce an infinite square lattice of sites labeled (M, N) in the (x, z) plane, with a horizontal and vertical spacings of d and h respectively. We consider a single void propagating upwards through the lattice, and let $V_N(M)$ be the probability that it is at horizontal site M when it

is in the Nth vertical layer. The random walk description allows us to write down a recurrence relation of the form

$$V_N(M) = \frac{V_{N-1}(M-1) + V_{N-1}(M+1)}{2}.$$

If we now introduce a continuum analogue η such that $V_N(M) = \eta(dM, hN)$ then we see that

$$\eta(x,z) = \frac{\eta(x-d, z-h) + \eta(x+d, z-h)}{2}$$

and by taking the limit $h, d \to 0$, in a way that $2bh = d^2$ where b is a constant, we arrive at a partial differential equation

$$\frac{\partial \eta}{\partial z} = b\frac{\partial^2 \eta}{\partial x^2}.$$

This is a diffusion equation, but with the time replaced by the vertical coordinate z. For the case of a point orifice, which would be represented by an initial condition of the form $\eta(x,0) = \delta(x)$, we would obtain the result

$$\eta(x,z) = \frac{1}{\sqrt{4\pi bz}} e^{x^2/4bz}. \tag{2.1}$$

The density of voids at a point is proportional to the vertical velocity, and hence the void model predicts Gaussian velocity profiles, with a width proportional to $\sqrt{z}$. This has been verified in several experimental studies [84, 83, 114, 27], and appears in good agreement with figure 1-1. The result depends on a single parameter b which is derived from geometrical considerations alone, and it controls the width of the velocity profile.

As described by Nedderman and Tüzün [95], this prediction for the velocity field $\mathbf{v}$ can also be derived entirely from a continuum standpoint, from two reasonable physical postulates:

- The material is incompressible, so that $\nabla \cdot \mathbf{v} = 0$.
- Particles drift towards regions of higher vertical velocity, so that $v_x = -b\frac{\partial v_z}{\partial x}$.

Thus

$$\begin{aligned} 0 &= \nabla \cdot \mathbf{v} \\ &= \frac{\partial v_x}{\partial x} + \frac{\partial v_z}{\partial z} \\ &= \frac{\partial}{\partial x}\left(-b\frac{\partial v_z}{\partial x}\right) + \frac{\partial v_z}{\partial z} \end{aligned}$$

from which we see that

$$\frac{\partial v_z}{\partial z} = b\frac{\partial^2 v_z}{\partial x^2}$$

which is equivalent to the void model continuum prediction.

2.4 Diffusion in the void model

The void model was originally proposed to explain the mean flow in granular drainage, but the microscopic mechanism proposed in figure 2-1 also tells us how particles will move, and allows us to make predictions about the amount of mixing. In this section, we provide a mathematical description of this subject, comparing with experimental results of Choi [28]. Choi's third result about diffusion being proportional to distance dropped is immediately satisfied, since mixing occurs only in response to particle flow, and has no explicit time scale, but it remains to be seen whether the other two results remain true.

Continuing the analysis of the previous section, we introduce a quantity $P_N(M)$ to be the probability that a particle is at horizontal location M when it is in the Nth layer. We assume that the voids do not interact, and pass through the material independently of one another. A particle at (M, N) can move to either $(M-1, N-1)$ or $(M+1, N-1)$ depending on the direction the first void to arrive at (M, N) comes from. Thus the ratio of the probability of moving left to moving right is equal to the ratio of the probability of a void being at $(M-1, N-1)$ to being at $(M+1, N-1)$,

and hence

$$\begin{aligned}
\mathbb{P}((M,N) \to (M-1,N-1)) &= \frac{V_{N-1}(M-1)}{V_{N-1}(M-1)+V_{N-1}(M+1)} \\
&= \frac{V_{N-1}(M-1)}{2V_N(M)} \\
\mathbb{P}((M,N) \to (M+1,N-1)) &= \frac{V_{N-1}(M+1)}{V_{N-1}(M-1)+V_{N-1}(M+1)} \\
&= \frac{V_{N-1}(M+1)}{2V_N(M)}.
\end{aligned}$$

Thus we can write down a recurrence relation of the form

$$P_{N-1}(M) = P_N(M-1)\frac{V_{N-1}(M)}{2V_N(M-1)} + P_N(M+1)\frac{V_{N-1}(M)}{2V_N(M+1)}.$$

Introducing a continuum analogue ρ such that $P_N(M) = \rho(dM, hN)$ gives

$$2\rho(x, z-h) = \eta(x, z-h)\left(\frac{\rho(x-d,z)}{\eta(x-d,z)} + \frac{\rho(x+d,z)}{\eta(x+d,z)}\right).$$

The fastest way to find a continuum equation for ρ is to introduce the quantity $\sigma = \rho/\eta$. We see that

$$2\sigma(x, z-h) = \sigma(x-d,z) + \sigma(x+d,z)$$

and thus in the continuum limit $d, h \to 0$ with $2bh = d^2$, we obtain

$$-\frac{\partial \sigma}{\partial z} = b\frac{\partial^2 \sigma}{\partial x^2}.$$

Substituting back for σ gives

$$\begin{aligned}
\frac{\rho_z}{\eta} - \frac{\rho\eta_z}{\eta^2} &= -b\frac{\partial}{\partial x}\left(\frac{\rho_x}{\eta} - \frac{\eta_x\rho}{\eta^2})\right) \\
&= -b\left(\frac{\rho_{xx}}{\eta} - \frac{2\eta_x\rho_x}{\eta^2} - \frac{\eta_{xx}\rho}{\eta^2} + 2\frac{\eta_x^2\rho}{\eta^3}\right).
\end{aligned}$$

Using the continuum equation $\eta_z = b\eta_{xx}$ gives

$$\begin{aligned}\rho_z &= -b\rho_{xx} + 2b\left(\frac{\eta_x\rho_x}{\eta} + \frac{\eta_{xx}\rho}{\eta} - \frac{\eta_x^2\rho}{\eta^2}\right) \\ &= 2b\frac{\partial}{\partial x}\left(\frac{\rho\eta_x}{\eta}\right) - b\rho_{xx}.\end{aligned}$$

Thus the particle probability density follows an advection-diffusion equation. The amount of advection is proportional to the logarithmic derivative of η, meaning that particles preferentially drift towards regions of faster flow, which appears reasonable. However, this equation also predicts the diffusion of a particle is controlled by the same constant, b, as the diffusion of the voids. This appears to be at odds with Choi's experimental results, since we would expect the particles to diffuse on a much smaller length scale than the voids.

This can be seen more clearly by considering the explicit case of drainage from a point orifice, where η takes the form of equation 2.1. In that case, the advection is given by $\eta_x/\eta = -x/2bz$ and the above PDE becomes

$$\rho_z = -2\frac{\partial}{\partial x}\left(\frac{\rho x}{2z}\right) - b\rho_{xx}.$$

It is natural to consider the initial condition $\rho(x, z_0) = \delta(x - x_0)$, corresponding to a particle initially located at (x_0, z_0). In section A.1 of appendix A, two methods are provided to show that the exact solution of this equation is

$$\rho(x,z) = \frac{1}{\sqrt{4\pi bz\left(1-\frac{z}{z_0}\right)}}\exp\left(\frac{-(x-\frac{x_0 z}{z_0})^2}{4bz\left(1-\frac{z}{z_0}\right)}\right). \tag{2.2}$$

Thus the solution is always a gaussian, with variance $bz(1 - z/z_0)$ and mean x_0z/z_0. A typical solution is shown in figure 2-2. The mean drifts linearly towards the orifice. The variance is initially zero when $z = z_0$, corresponding to the delta function initial condition. It then increases to a maximum value of $bz_0^2/4$ when $z = z_0/2$. As $z \to 0$, the variance begins to decrease, since the particle must exit from the point orifice.

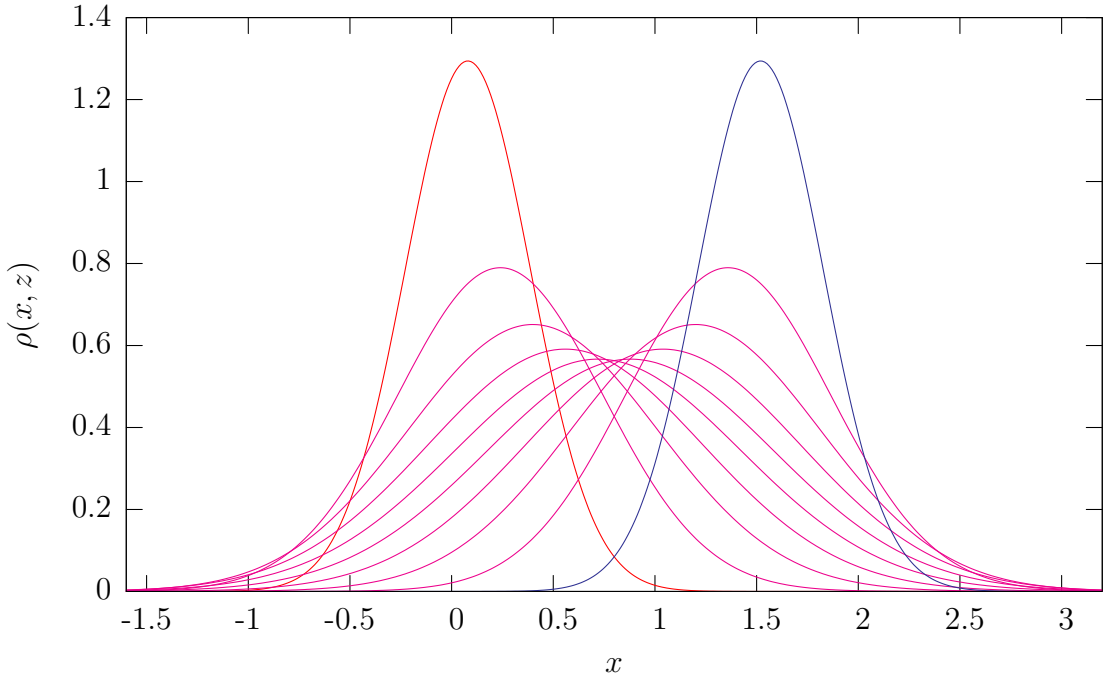


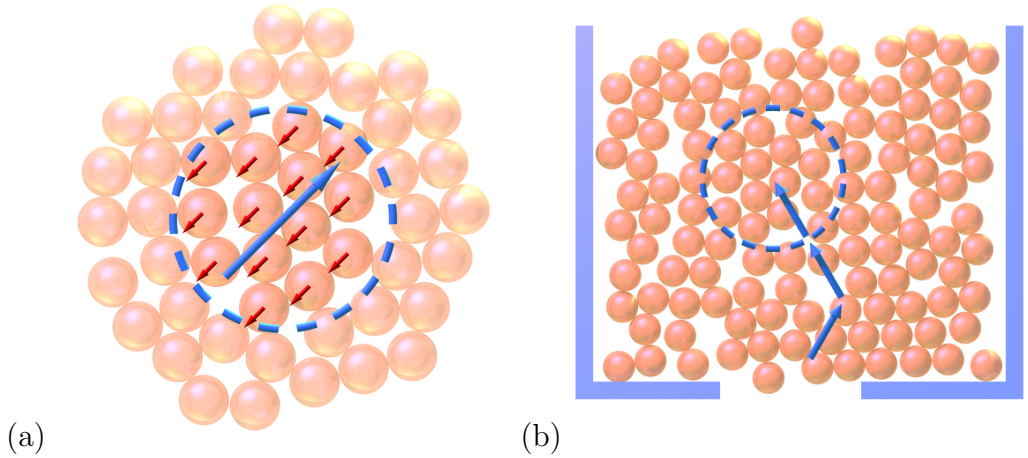
Figure 2-2: A typical solution to equation 2.2 using parameters $x_0 = 1.6$, $z_0 = 2$, and $b = 1$. Cross-sections are plotted for $z = 0.1$ (shown in blue), $z = 0.3, 0.5, \ldots, 1.7$ (shown in magenta), and $z = 1.9$ (shown in red).

The crucial result of figure 2-2 is that the particle's probability density will attain a width comparable of that of the mean flow, which is completely at odds with the experimental results that predict a particle's diffusion should be much more localized. In the experimental geometry of figure 1-1, the void model would predict that the colored bands would rapidly mix in the flowing region, when in fact they remain coherent even after a large amount of flow has taken place. We are therefore led to the conclusion that while the void model predicts a reasonable mean flow, it is based on a flawed model of the microscopic physics.

2.5 The spot model

Although the void model microscopic mechanism appears unrealistic, the spreading gaussian velocity profiles seen in experiment suggest that the notion of a diffusing quantity may still be worthwhile. It is therefore natural to ask if one could formulate an alternative model that would still predict the diffusing velocity profiles, but would have a better microscopic basis. Reconsidering the void mechanism, it is clear that one of its main problems is that particles are allowed to move too independently of one another: after a single void has passed, a particle will lose two of its original neighbors, which seems at odds with the persistent particle cages seen in experiment. In reality, a single particle in a granular packing is strongly constrained by its neighbors, and if it is going to move, it must do so *cooperatively*.

Based on these ideas, Bazant formulated the spot model [17] shown in figure 2-3(a). In this model, which can be applied in both two or three dimensions, particles are no longer constrained to lie on a lattice. They move in response to spots, shown by the blue circle, which correspond to a small amount of excess free volume spread across several particle diameters. When the spot is displaced by an amount, it causes a correlated motion in the opposite direction of all particles within range. In the simplest model, the displacement of each particle $\Delta\mathbf{x}_p$ is a fraction of the spot's displacement $\Delta\mathbf{x}_s$, so that $\Delta\mathbf{x}_p = -w\Delta\mathbf{x}_s$. The diameter of the spot D corresponds to a length scale characteristic of local particle rearrangement. Experimental evidence



(a) (b)

Figure 2-3: The blue circle shown in (a) represents a spot, which corresponds to a small amount of free volume spread across spread across several particle diameters. When the spot is displaced by the blue arrow, it causes a small, correlated motion in the opposite direction of all particles within range, as shown by the red arrows. The size of the particle displacement is typically on the order of a hundredth of a particle diameter, but is shown as much larger here for clarity. Granular drainage is modeled by imagining a container full of off-lattice particles, and then injecting spots at the orifice, which propagate upwards according to a random walk.

suggests that a scale of three to five particle diameters is important in dense granular flow, and we typically take D in this range. In granular drainage, particles will only flow from an orifice of diameter $\sim 3d - 5d$, since for smaller orifices, particles will tend to jam. Granular flows often exhibit boundary layers of slower velocity with a length scale in this region.

In a typical three-dimensional monodisperse granular material made up of spherical particles, the packing fraction is approximately $\phi = 60\%$. For $D = 5d$, the number of particles that are influenced by a single spot is therefore

$$N = \frac{\frac{4}{3}\pi\left(\frac{5d}{2}\right)^3 \times 60\%}{\frac{4}{3}\pi d^3} \approx 75.$$

When the spot moves, it causes N particles to move by a fraction w in the opposite direction, and thus it makes sense to define the volume of the spot as

$$V_s = NwV_p$$

where V_p is the particle volume. When spots enter a region, they cause a drop in the packing fraction, and comparing this drop with experimental data on flowing packings allows us to gain an order-of-magnitude estimate on the volume carried by a spot. In the DEM simulations presented later in this thesis, and in work carried out by other authors [134, 67, 98, 99, 100] it is typical to expect the packing fraction to decrease by 5% from a static to a flowing state. Since spots could overlap, a reasonable estimate is to attribute $\Delta\phi/\phi \approx 1\%$ as an upper bound on the drop attributed with a single spot. In his paper [17], Bazant argued that

$$w = \frac{\Delta\phi}{\phi^2}.$$

However, the current author believes that

$$w = \frac{\Delta\phi}{\phi}$$

is a more appropriate definition, and for a full comparison between these two different perspectives, the reader should refer to section A.2 of appendix A. However, since ϕ is an order one quantity, both approaches would give an order of magnitude estimate for w in the range 10^{-2} to 10^{-3}. Correspondingly, we expect V_s/V_p to have an order of magnitude between 1 and 0.1.

The granular drainage experiment can be modeled by introducing spots at the orifice which propagate upwards according to a random walk, as shown in figure 2-3(b). At each stage, we assume that a spot makes a fixed step Δz_s upwards, and makes a random step $\Delta\mathbf{x}_s$ in the remaining horizontal dimensions. The spot diffusion length is defined by

$$b_s = \frac{\mathrm{Var}(\Delta x_s)}{2d_h\,|\Delta z_s|}$$

where d_h is the number of horizontal dimensions. The particle diffusion length is given by

$$b_p = \frac{\mathrm{Var}(\Delta x_p)}{2d_h\,|\Delta z_p|} = \frac{w^2\,\mathrm{Var}(\Delta x_s)}{2d_h w\,|\Delta z_p|} = w b_s$$

and thus we see that the spot model predicts that particle diffusion occurs on a scale

approximately two to three orders of magnitude smaller than the spot diffusion. The spot model therefore appears to qualitatively explain all the experimental features seen by the Choi experiments. Like the void model, it predicts parabolic velocity profiles, and that diffusion will be driven by geometry, but it also successfully captures the strongly correlated motion and small diffusion of the particles. The spot model remains simple enough for mathematical analysis, and as described by Bazant [17], makes a number of predictions about particle motion in granular flow. Perhaps most importantly, it suggests that particles in a granular flow should exhibit spatial velocity correlations. Consider the spatial velocity correlation tensor

$$C^{\alpha\beta}(\mathbf{r}_1, \mathbf{r}_2) = \frac{\left\langle v^\alpha(\mathbf{r}_1) v^\beta(\mathbf{r}_2) \right\rangle}{\sqrt{\left\langle v^\alpha(\mathbf{r}_1) \right\rangle \left\langle v^\beta(\mathbf{r}_2) \right\rangle}}$$

where $v^\alpha(\mathbf{r})$ is the α component of instantaneous velocity at position $\mathbf{r}$, and the angular brackets refer to time averages. We can simplify this by considering a single horizontal velocity component u, and exploiting translational and rotational symmetry to write

$$C(\mathbf{r}) = C(r) = \frac{\langle u(\mathbf{0}) u(\mathbf{r}) \rangle}{\sqrt{\langle u(\mathbf{0}) \rangle \langle u(\mathbf{r}) \rangle}}.$$

In the spot model, the velocities of two nearby particles will be correlated if they fall within the displacement of the same spot. If we assume that spot displacements occur randomly, then $C(r)$ will be given by the probability that a spot influencing a particle at $\mathbf{0}$ influences a particle a distance r away as well. The correlations in two dimensions are given by

$$C(r) = \begin{cases} 1 - \frac{r\sqrt{1-(r/2R)}}{\pi R} - \frac{2}{\pi} \sin^{-1}\left(\frac{r}{2R}\right) & \text{for } r < 2R \\ 0 & \text{for } r \geq 2R \end{cases}$$

and in three dimensions by

$$C(r) = \begin{cases} \left(1 + \frac{r}{4R}\right)\left(1 - \frac{r}{2R}\right)^2 & \text{for } r < 2R \\ 0 & \text{for } r \geq 2R. \end{cases}$$

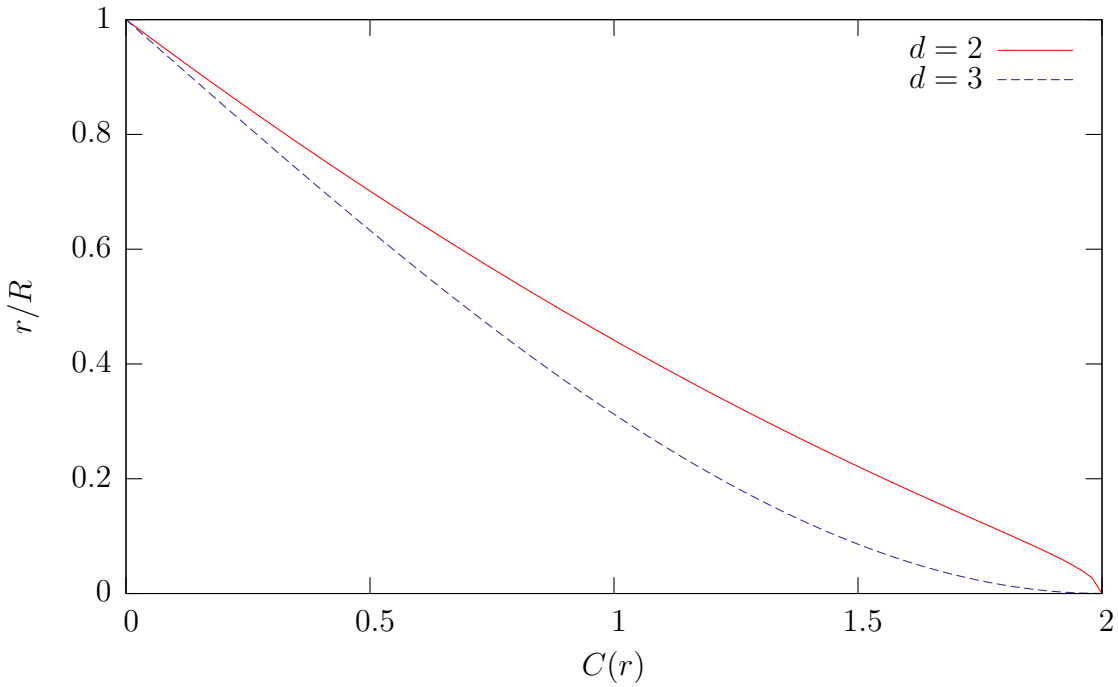


Figure 2-4: Plots of the theoretical velocity correlations for the uniform spot potential, in two and three dimensions.

These are plotted in figure 2-4 – if the spot model for particle motion exists, then we would expect to see correlations that have a decay length similar to the spot size. Choi searched for these correlations in the tall silo experimental geometry, and the results are shown in figure 2-5 – we do indeed see correlations decaying on an intermediate scale, although local ordering effects are visible. Since the measurement is taken at the wall, many particles locally arranged in a hexagonal crystalline formation. It would be expected to see higher correlations at values of $1d$, $\sqrt{3}d$, and $2d$ since particles which are locally crystallized will have very strong geometrical constraints with their neighbors.

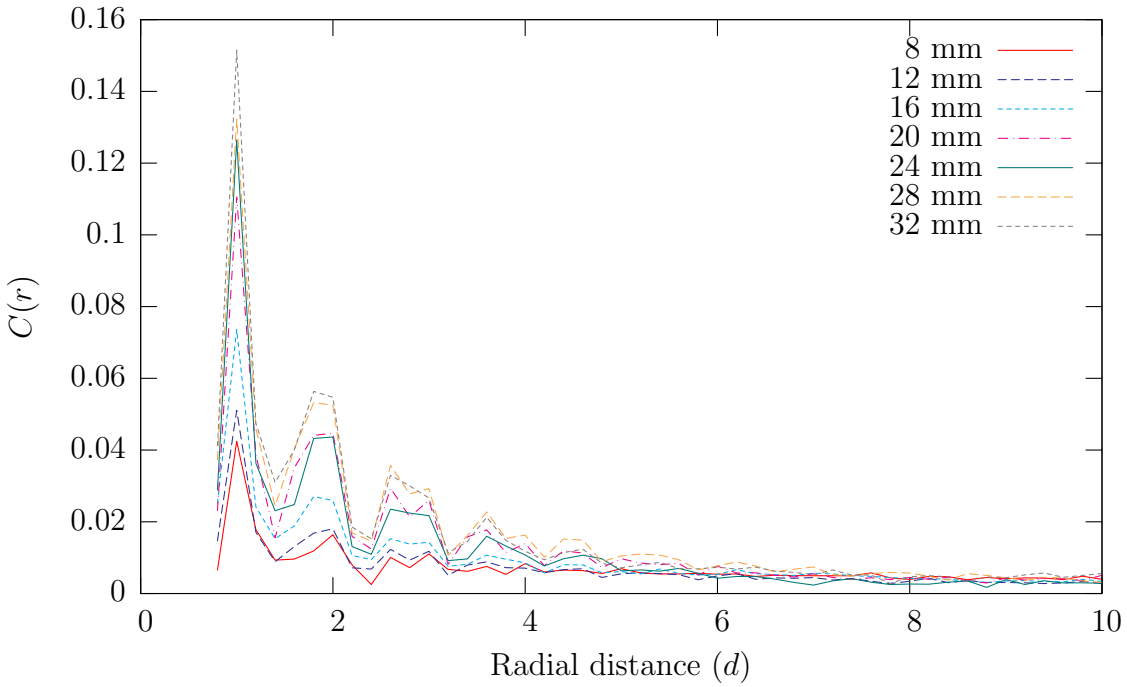


Figure 2-5: Velocity correlations in the x direction for Choi's granular drainage experiment.

2.6 Discrete Element Simulation and comparison to experiment

While the experiments of Choi provided important insight into granular diffusion, they have one unavoidable drawback: measurements can only be made of the particles at the glass wall. It is hoped that the low friction coefficient of the glass causes particles near the wall to have similar velocities to those in the bulk, but the presence of the wall will have a significant effect on the packing geometry. It is a well-known fact that packing properties of monodisperse spherical particles are very different in two and three dimensions, with particles in two dimensions exhibiting a much larger tendency to form hexagonal crystalline arrangements. It is therefore natural to ask whether the velocity correlations of figure 2-5 are also present in the bulk.

Carrying out large-scale particle-based simulations would therefore be particularly advantageous, since they can provide us with complete three-dimensional information

about packing arrangements. However, simulating granular materials is a difficult task that has only become computationally practical in the last five to ten years. The most obvious problem is the large number of particles that must be considered, even for a fairly small system. The simulation of particles made of hard materials is also very difficult. One approach is to model the particles as inelastic hard spheres, which interact according to a coefficient of restitution. Simulations of this type are event-driven, with the code calculating the next time until a collision occurs between any two particles in the system. Simulation studies have used this method to look at dense granular flows [48, 46, 47], but they have only been able to consider a fully flowing packing, since some particles in static regions undergo "inelastic collapse" where they will want to undergo an infinite number of collisions in over a finite time.

An alternative model is to treat particles as interacting via a spring with a very high spring constant, and then simulate the resulting positions using a traditional fixed timestep – this is referred to as a Discrete Element Method (DEM). The high spring constant means that the equations are stiff, and require a very small timestep, but the approach has become computationally feasible in the past five years, particularly since the interactions between particles are short range, which allows the code to be efficiently parallelized, to take advantage of the recent and growing trend toward multicore and parallel computing. In 2001 and 2002 Grest, Landry, Plimpton, and Silbert at Sandia National Laboratories implemented a parallel DEM code called GranFlow, written in Fortran 90. This has since been incorporated into the Large-scale Atomic/Molecular Massively Parallel Simulator (LAMMPS) [1] which is written in C++. GranFlow is used in the work presented here and in chapters 3 and 5, while LAMMPS was used in the later work of chapters 6 and 7. The code has been very successful and has been used by many groups to study jammed granular packings [124, 125, 71], vibrated granular systems [129], and granular flows [123, 126, 24, 112, 113, 26].

The DEM simulations presented in this thesis use of a common set of conventions. In all cases, we make use of a three co-ordinate system (x, y, z), and make measurements in terms of the particle diameter d. The simulations act under gravity

g, pointing in the negative z direction. From this, a natural simulation time unit $\tau = \sqrt{d/g}$ is defined. Given a particle diameter d in terms of a physical length, we can make comparisons to experiment by computing τ in terms of seconds. For example, to make the correspondence to Choi's simulation data, where $d = 3$ mm, we have

$$\tau = \sqrt{\frac{3 \text{ mm}}{9.81 \text{ ms}^{-2}}} = 0.0174 \text{ s}.$$

The mass of the particles is referred to by m, although most of the time, it is not necessary to consider this explicitly, since it is only important in relation to the details of the interaction model.

The contact model is based on that developed by Cundall and Strack [32] to model cohesionless particulates. If a particle and its neighbor are separated by a distance $\mathbf{r}$, and they are in compression, so that $\delta = d - |\mathbf{r}| > 0$, then they experience a force $\mathbf{F} = \mathbf{F}_n + \mathbf{F}_t$, where the normal and tangential components are given by

$$\mathbf{F}_n = f(\delta/d)\left(k_n \delta \mathbf{n} - \frac{\gamma_n \mathbf{v}_n}{2}\right) \tag{2.3}$$

$$\mathbf{F}_t = f(\delta/d)\left(-k_t \Delta \mathbf{s}_t - \frac{\gamma_t \mathbf{v}_t}{2}\right). \tag{2.4}$$

Here, $\mathbf{n} = \mathbf{r}/|\mathbf{r}|$. $\mathbf{v}_n$ and $\mathbf{v}_t$ are the normal and tangential components of the relative surface velocity, and $k_{n,t}$ and $\gamma_{n,t}$ are the elastic and viscoelastic constants, respectively. Two different force models are considered: $f(z) = \sqrt{z}$ for Hertzian particle contacts and $f(z) = 1$ for Hookean particle contacts. $\Delta \mathbf{s}_t$ is the elastic tangential displacement between spheres, obtained by integrating tangential relative velocities during elastic deformation for the lifetime of the contact. If $|\mathbf{F}_t| > \mu |\mathbf{F}_n|$, so that a local Coulomb yield criterion is exceeded, then $\mathbf{F}_t$ is rescaled so that it has magnitude $\mu |\mathbf{F}_n|$, and $\Delta \mathbf{s}_t$ is modified so that equation 2.4 is upheld.

Particle-wall interactions are treated identically, but the particle-wall friction coefficient is set independently. For the current simulations we set $k_t = \frac{2}{7} k_n$, and choose $k_n = 2 \times 10^5 mg/d$. While this is significantly less than would be realistic for glass spheres, where we expect $k_n \sim 10^{10} mg/d$, such a spring constant would be prohibitively computationally expensive, as the time step must have the form $\delta t \propto k_n^{-1/2}$

for collisions to be modeled effectively. Previous simulations have shown that increasing k_n does not significantly alter physical results [71]. We make use of a timestep $\delta t = 10^{-4}\tau$ The normal damping term is set to $\gamma_n = 50\sqrt{g/d}$, and the tangential damping γ_t is set to zero for Hookean contacts, and equal to γ_n Hertzian contacts.

Unless stated otherwise, the Hertzian contact model was employed for the simulations presented in this thesis. The Hertzian model is motivated by the fact that when spheres come into contact, the amount of volume by which they overlap grows faster than linearly, and thus the force should grow faster than linearly too. While it may be more physically desirable to use this model, it was found here and by others that it can sometimes lead to simulation artefacts, such as elastic "breathing modes" propagating through the simulations. Therefore, in cases where precise information about microscopic motion was needed, the Hookean contact model was used.

The simulations were carried out on MIT's Applied Mathematics Computational Laboratory (AMCL), a Beowulf cluster consisting of 16 nodes each with a dual processor. During the simulations, snapshots of all particle positions were saved every 2τ, corresponding to 20,000 integration timesteps. The LAMMPS code is written in C++ and can be run on any number of processors, by decomposing the computational domain into a rectangular grid of subdomains of equal size. Interactions between particles in neighboring domains are handled using message passing. For problems where the particles are split evenly between the subdomains, the LAMMPS code scales very well, and doubling the number of processors can frequently result in almost a doubling of speed.

2.7 Comparison to experiment: velocity profiles

One of the first tasks was to validate the simulations against experiment. A simulation was run where the parameters were chosen as closely as possible to the experiments of Choi. Based on a particle diameter of $d = 3$ mm, a quasi-two dimensional container with walls at $x = \pm 33\frac{1}{3}d$, $y = \pm 4\frac{1}{6}d$ and $z = 0d$ was created. The interparticle friction coefficient and the front and back wall friction coefficients were set to be 0.2

to approximately model a glass/glass contact. The side and base walls were modeled using a friction coefficient of 0.3, to correspond with a glass/metal contact.

Throughout this thesis, initial packings were created by pouring the particles into a container. To pour the particles, an insertion region over a range $z_{\text{low}} < z < z_{\text{high}}$ is created. When the simulation starts, the insertion region is filled with particles up to a packing fraction of 10%, by making random insertion events. When a particle is inserted at a location (x, y, z), its horizontal velocity components are set to zero, and its vertical velocity is set to $v_z = \sqrt{2g(z_{\text{high}} - z)}$ corresponding to the speed it would have attained had it fallen from rest at z_{high}. The x and y coordinates of the insertion events are chosen uniformly. The z coordinates are chosen so that v_z is uniformly distributed. The simulation continues, and when the particles at the top of the insertion region have dropped to z_{low}, another insertion event is carried out, and another batch of particles is created. This process creates the effect of a steady stream of particles being introduced at a constant rate from $z = z_{\text{high}}$. Once the required number of particles has been introduced, the batch insertion events are terminated and simulation continues. Typically, the system is simulated for a long time after the last insertion event to ensure that the particles have come to rest.

For the current simulation, 174,249 particles were poured using $z_{\text{low}} = 333\frac{1}{3}d$ and $z_{\text{high}} = 366\frac{2}{3}d$, to fill the container to a height of $264d$. The drainage process was then carried out by opening a slit of width $5\frac{1}{3}d = 16$ mm in the center of the container base. Figure 2-9(a) shows a snapshot of the system after a certain amount of flow has taken place.

Figure 2-6 shows a comparison of the vertical velocity profiles between the simulation and the experiment for two different cross sections, where the simulation data has been converted into physical units. The match is extremely good, both in terms of the overall shape, and the total flow rate.

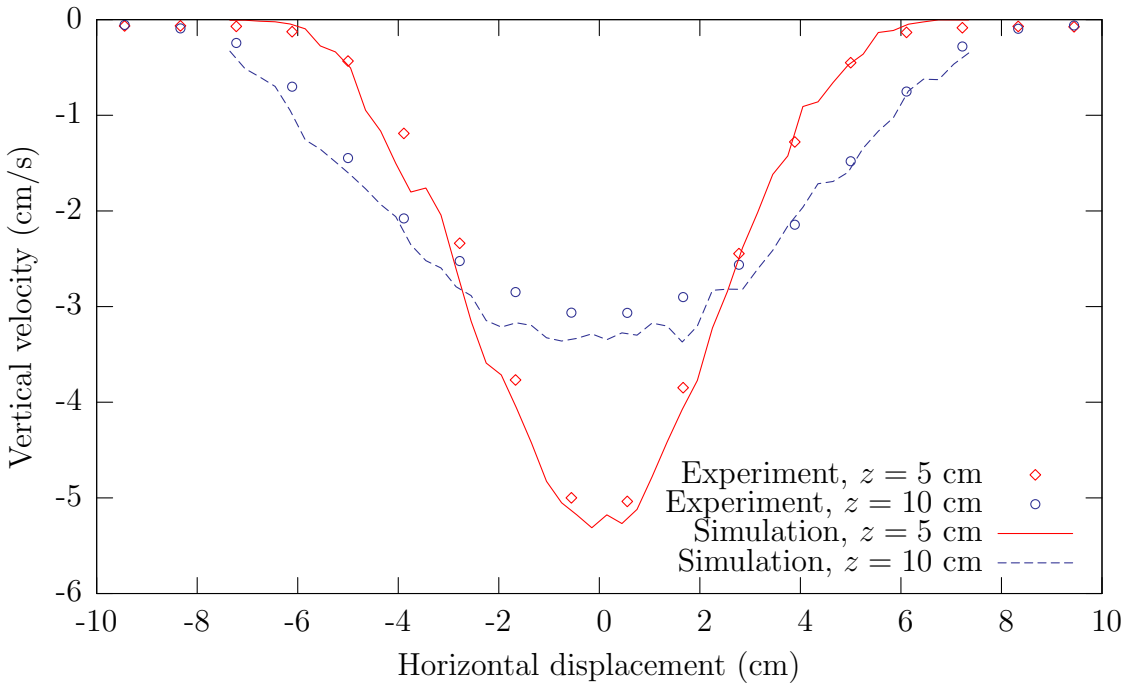


Figure 2-6: Comparison of horizontal profiles of downwards velocity between DEM simulation and Choi's granular drainage experiment.

2.8 Velocity correlations

Velocity correlations were also compared to experiment. However, since searching for velocity correlations requires a large amount of data over which to gain a statistical average, it was chosen to carry out these measurements in a smaller simulation, which could be run for longer. Since the velocity correlations are a local effect, they should not be dependent on the precise geometry used. Walls were placed at $x = \pm 25d$, $y = \pm 4d$, and $z = 0$, and approximately 55,000 particles were poured from a height of $170d$ to fill the container up to a height of $110d$. For this simulation, $\mu = \mu_w = 0.5$, which is a commonly-used figure when running the DEM code. To create the flow, a circular hole of radius $4d$ centered on $x = y = 0$ was opened. The simulation was treated as periodic in the z direction, so that particles falling out at $z = 0$ would be reintroduced at $z = 181d$, creating a continuous flow. For making measurements of velocity correlations, there were two regions of interest: the particles near the front wall, whose centers satisfy $y > 3.2d$, and the particles in the central slice, whose

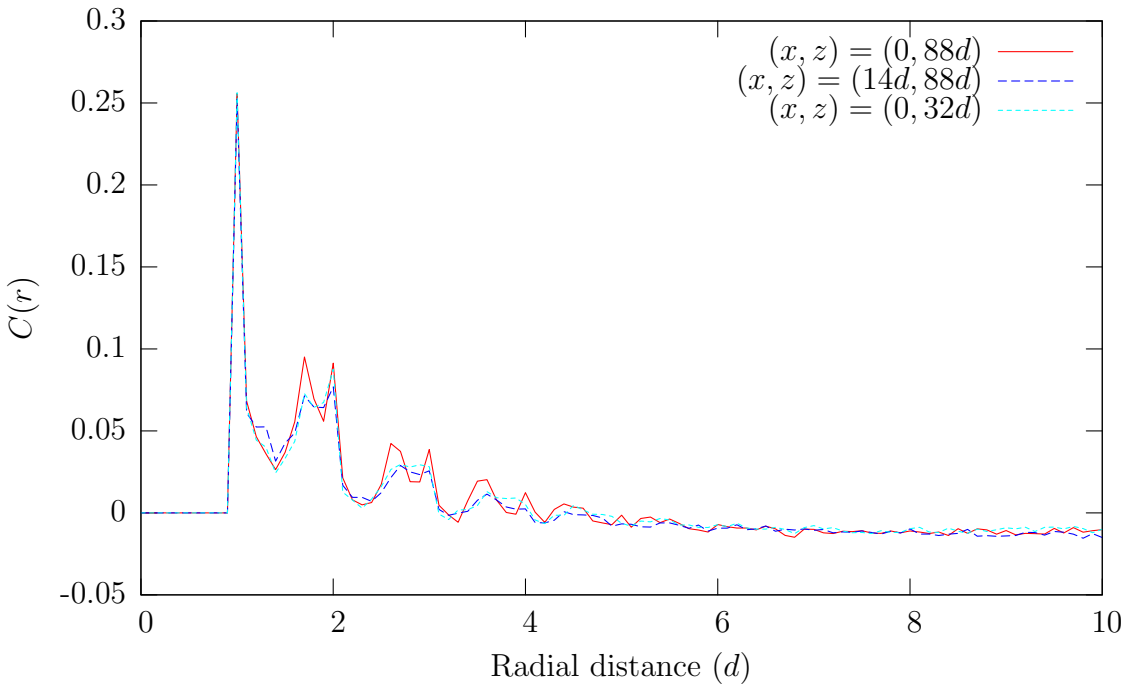


Figure 2-7: Velocity correlations at the boundary of a DEM simulation, computed in square test regions of side length $16d$ centered at three different locations. The functional form closely matches Choi's experimental data.

centers satisfy $|y| < 2d$.

The simulation was first run to calculate a mean background velocity, by averaging the instantaneous particle velocities in cubic grid of side length $1d$. Since the system is symmetric in the x and y directions, this can be exploited to improve the accuracy of the computed field by a factor of four. Once computed, the mean velocity at a specific point could be found by linearly interpolating from the eight neighboring cells.

A run was then carried out to calculate velocity correlations in three different test squares of side length $16d$, both at the boundary and in the bulk. One test square was centered on $(x, z) = (0, 88d)$, high in the central region of the container where the flow is approximately uniform. In addition, an off-axis test region at $(x, z) = (14d, 88d)$, plus a lower test region at $(x, z) = (0, 32d)$, were tested. We computing the correlation statistics, the previously computed mean velocity field is subtracted, so that the results capture the local variations.

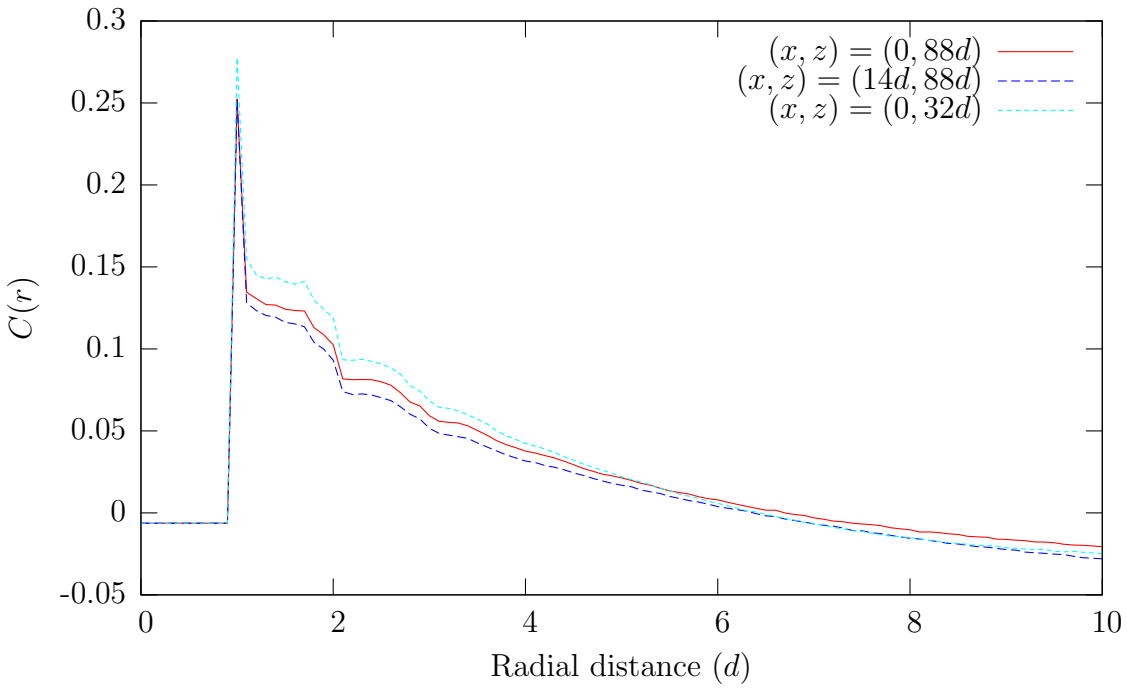


Figure 2-8: Velocity correlations at in the bulk slice of a DEM simulation, computed in square test regions of side length $16d$ centered at three different locations.

Figure 2-7 shows the computed velocity correlations in the boundary slice. In all three test regions, the functional form closely matches Choi's correlation data, and captures the same lattice effects. Along with the mean flow data presented in the previous section, this provides further evidence that the simulations are a faithful reproduction of experiment.

Figure 2-8 shows the velocity correlations in the same three test regions, but looking in the boundary slice. We see a decaying correlation, similar in magnitude and decay length to the result at the boundary. However, as expected, the local ordering effects are much weaker: there is a strong signature at $r \approx d$, corresponding to particles in contact, but for larger separations, the computed function is smooth. We can thus infer that the spot model for correlated particle motion appears to be a general phenomenon in dense granular flow, that holds both at the boundaries and in the bulk.

Furthermore, we see that for the three different test regions, the structure of the correlations, and their overall correlation length, are remarkably similar. It suggests

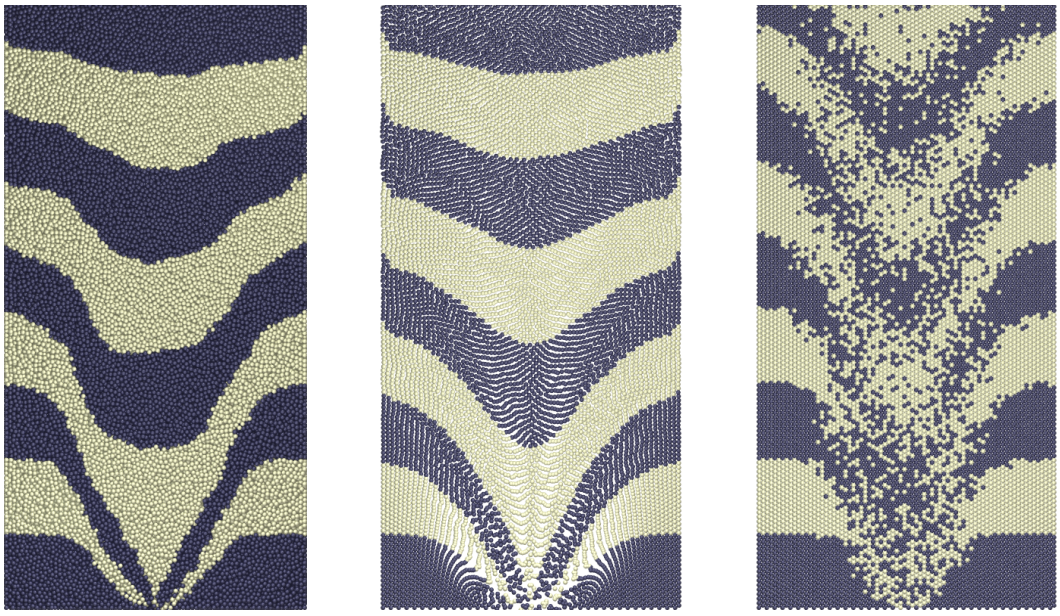


Figure 2-9: A snapshot of a DEM simulation of granular drainage in Choi's experimental geometry (left), compared to 2D simulations of the spot model (center) and void model (right), all shown at $t = 200\tau$. The total flow rates of the spot and void models were calibrated to match DEM. The spot size and diffusion rate were calibrated using the same process discussed in the following chapter.

that the process by which particles flow is similar throughout the container, and thus it may be a reasonable approximation to treat spots as having a fixed radius. The evidence also supports the later work in chapter 7 where we make use of a fixed-size granular element throughout a granular simulation.

2.9 Comparison of DEM, spot and void simulations

Figure 2-9 shows a snapshot from the DEM simulation of Choi's geometry, and compares it to simulations in two dimensions of the spot and void models. The simulation of the void model shows an unrealistic amount of diffusion, and the interfaces between the colored layers is rapidly smeared out, agreeing with the theoretical predictions of previous sections.

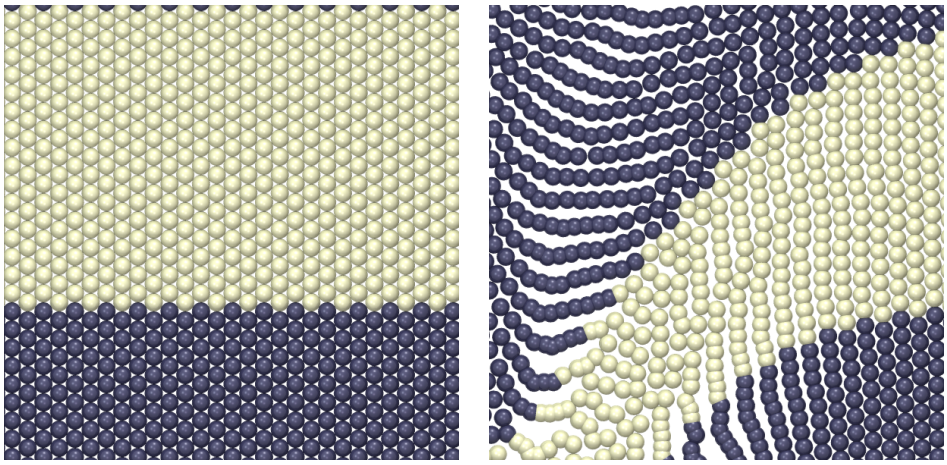
The spot model is a significantly better match, and the interfaces show an amount

of diffusion which matches that seen in the DEM. Although there are some discrepancies, the spot model tracks the overall flow profile reasonably well. However, this image shows one very significant drawback of the spot model as a simulation technique: looking near the orifice, we see that many particles have become overlapped. This is seen more clearly in the progression of close-up images shown in figure 2-10.

This should be expected, since in the spot model of microscopic particle motion shown in figure 2-3(a), there is nothing explicitly enforcing packing constraints. In this figure, it is clear that particles near the edge of the spot may end up overlapping with their neighbor by small amount, which may become larger and larger as more spots pass through. The above figures show that the problem is largest in areas undergoing a large amount of shear.

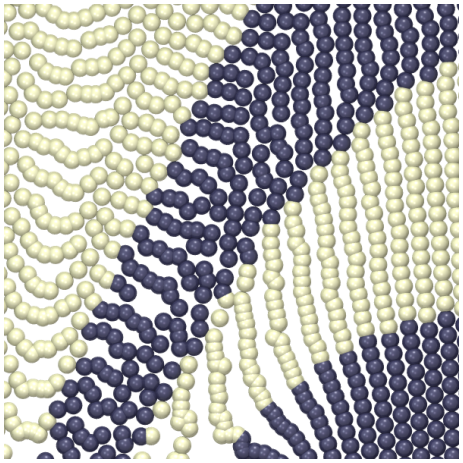
This represents a significant hurdle in using this model as a simulation technique. The basic model of particle motion is good for a mathematical analysis, and can make reasonable predictions about the statistics of motion of a single particle, but it cannot be used to generate reasonable predictions about the motions of all particles in the system. This problem is not present in the void model, since by enforcing particles to lie on a lattice they will never become overlapped.

Several modifications to the spot model microscopic motion were proposed to solve this problem. Making the spots carry out persistent random walks [55], adding a diffusive term to the particle motion, and adding a rotation were all tested. While some of these techniques mitigated the effect of the overlap, none of them were able to create fully valid packings. This is unsurprising, since it should be expected that in order to maintain a valid packing of particles, one must take into account the geometry of packing structure at a local level. In the following chapter, a solution to the density problem taking this approach will be described.

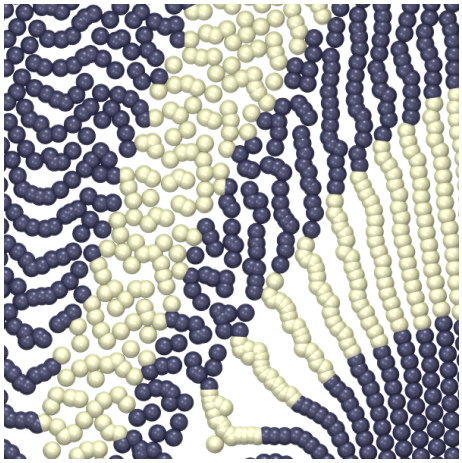


$t = 0$ s

$t = 1.74$ s



$t = 3.50$ s



$t = 5.25$ s

Figure 2-10: Four close-up snapshots of the spot model simulation from figure 2-9, taken in the region $0 < x < 7.5$ cm, 2.5 cm $< z < 10$ cm. The overlapping particles highlight the "density problem" of the basic spot model.

Chapter 3

Dynamics of Random Packings[1]

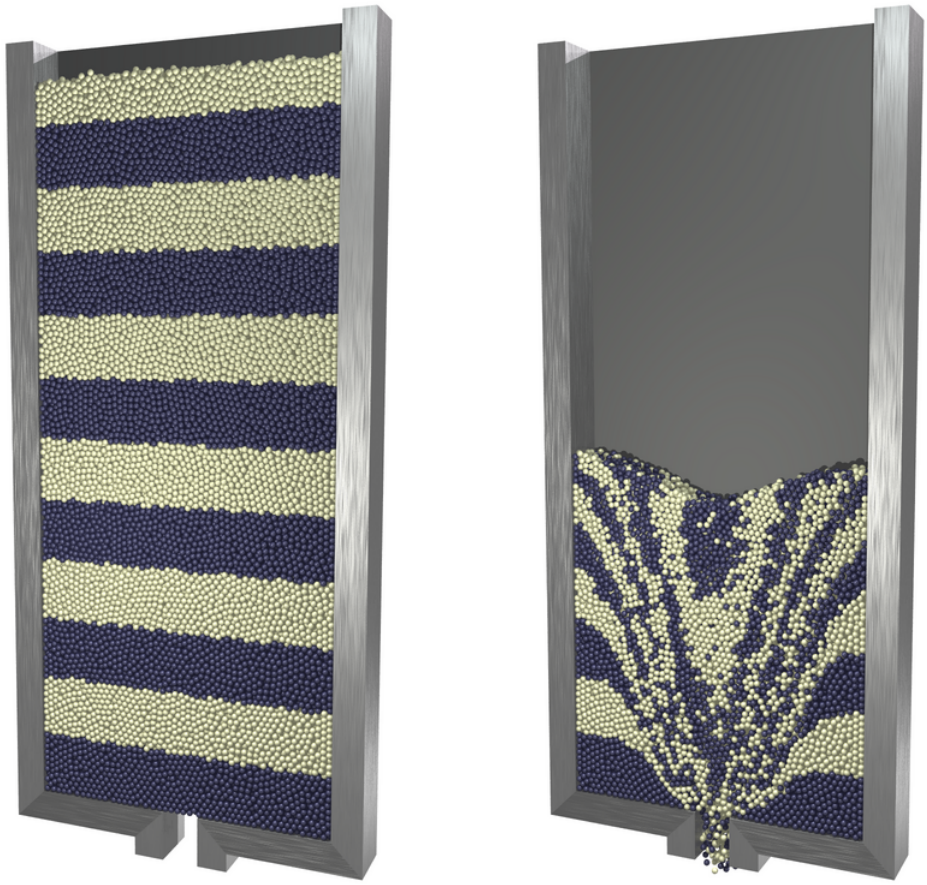
3.1 Introduction

In the previous chapter, it was shown that the spot model can provide a reasonable mathematical model of mean flow and particle motion in granular drainage. However, since particles are off-lattice, and there is nothing to prevent them from overlapping, it quickly generates unphysical packings.

One of the primary motivations of the spot model comes from a consideration of the local packing geometry, and it would therefore be advantageous if the spot model could be refined to correctly simulate the local packing rearrangements. We therefore proposed the model shown in figure 3-2, whereby each spot-induced block displacement (a) is followed by a relaxation step (b), in which the affected particles and their nearest neighbors experience a soft-core repulsion (with all other particles held fixed). The net displacement in (c) involves a cooperative local deformation, whose mean is roughly the block motion in (a). It is not clear *a priori* that this procedure can produce realistic flowing packings, and, if so, whether the relaxation step dominates the simple dynamics from the original model.

To answer these questions, we calibrated and tested the spot model against a large-scale DEM simulation of granular drainage, shown in figure 3-1. Simulations are

[1]This chapter is based on reference [112], *Dynamics of Random Packings in Granular Flow*, published in Physical Review E in 2006. See `http://pre.aps.org/` for more details.



(a) (b)

Figure 3-1: A simulation of the experiment in Ref. [28] by discrete element simulations. (a) First, 55,000 glass beads are poured into a quasi-two-dimensional silo (8 beads deep) and let come to rest. (b) Slow drainage occurs after a slit orifice is opened. (The grains are identical, but colored by their initial height.)

advantageous in this case since three-dimensional packing dynamics cannot easily be observed experimentally. We begin by running the DEM simulation, briefly described in section 3.2. We then calibrate the free parameters in the spot model by measuring various statistical quantities from the DEM simulation, as described in 3.3. In section 3.4, we describe the computational implementation of the spot model, before carrying out a detailed comparison to DEM in section 3.5.

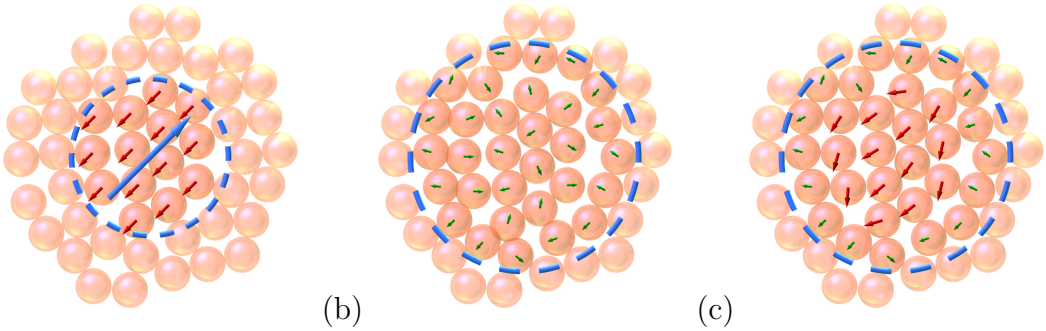


Figure 3-2: The mechanism for structural rearrangement in the spot model. The random displacement $\mathbf{r}_s$ of a diffusing spot of free volume (dashed circle) causes affected particles to move as a block by an amount $\mathbf{r}_p$ (a), followed by an internal relaxation with soft-core repulsion (b), which yields the net cooperative motion (c). (The displacements, typically 100 times smaller than the grain diameter, are exaggerated for clarity.)

3.2 DEM Simulation method

The basic geometry is equivalent to that used in the velocity correlation study in section 2.8, except that we consider a single drainage of the container, with no recycling of particles using periodic coordinates. The silo has width $50d$ and thickness $8d$ with side walls at $x = \pm 25d$ and front and back walls at $y = \pm 4d$, all with friction coefficient $\mu = 0.5$. The initial packing is generated by pouring $N = 55{,}000$ particles in from a fixed height of $z = 170d$ and allowing them to come to rest under gravity, filling the silo up to $H_o \approx 110d$. We also studied a taller system with $N = 135{,}000$ generated by pouring particles in from a height of $z = 495d$, which fills the silo to $H_o \approx 230d$. We refer to these systems by their initial height H_o. Drainage is initiated by opening a circular orifice of width $8d$ centered at $x = y = 0$ in the base of the silo ($z = 0$). A snapshot of all particle positions is recorded every 2×10^4 time steps ($\delta t = 1.75 \times 10^{-6}$ s). Once particles drop below $z = -10d$, they are removed from the simulation.

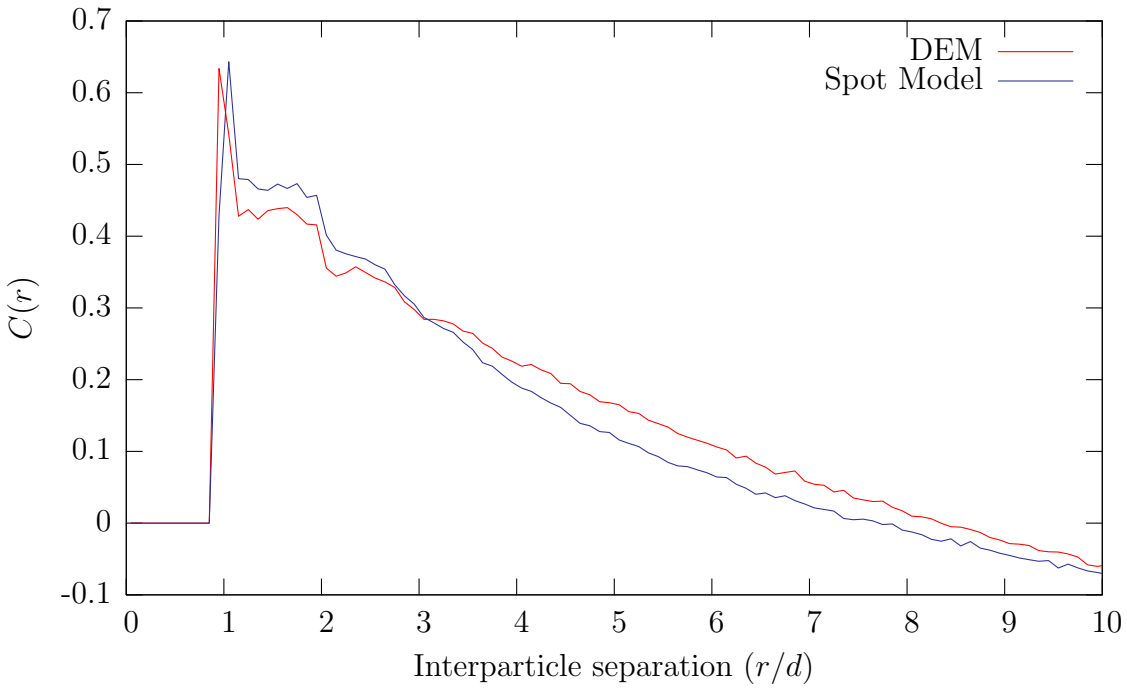


Figure 3-3: Comparison of velocity correlations calculated over the time period 0.52 s $< t <$ 1.57 s. Calculations are based on particle velocity fluctuations about the mean flow in a $16d \times 16d$ region high in the center of the container. For $H_o = 110d$.

3.3 Calibration of the model

We begin by calibrating the spot radius R_s by examining velocity correlations and comparing to the theoretical predictions of figure 2-4. The velocity correlations in the DEM simulation are shown in figure 3-3. Since the shapes of the two curves do not match, partly due to relaxation effects, we fit the simulation data to a simple decay, $C(r) = \alpha e^{-r/\beta}$ with $\beta = 1.87d$. We also fit a simple decay of the same form to the theoretical prediction, finding $\beta = 0.72R_s$, so we infer $R_s = 2.60d$ as the spot radius. Thus a grain has significant dynamical correlations with neighbors up to three diameters away.

Next, we infer the dynamics of spots, postulating independent random walks as a first approximation. We assume that spots drift upward at a constant mean speed, $v_s = \Delta z_s/\Delta t$, (determined below), opposite to gravity, while undergoing random horizontal displacements of size Δx_s in each time step Δt. The spot diffusion length, $b_s = \mathrm{Var}(\Delta x_s)/2\Delta z_s$, is obtained from the spreading of the mean flow away from the

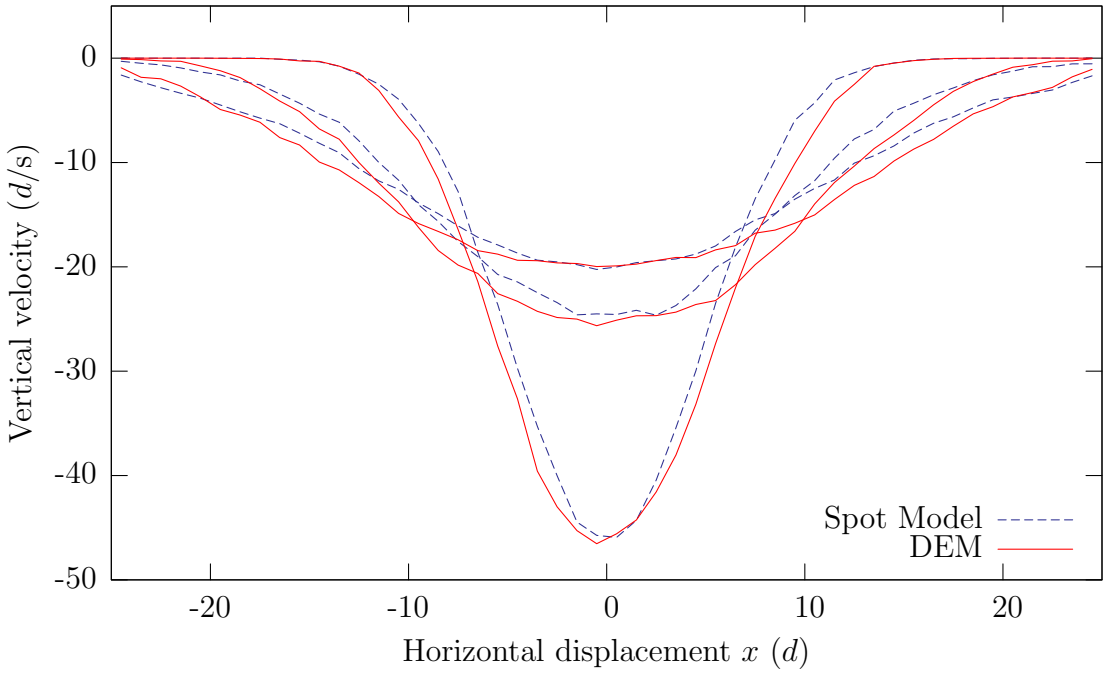


Figure 3-4: Comparison of the mean velocity profile, for three different heights calculated over the time period 4.37 s $< t <$ 5.25 s once steady flow has been established. The spot model successfully predicts a Gaussian velocity profile near the orifice and the initial spreading of the flow region with increasing height, although the DEM flow becomes more plug-like higher in the silo.

orifice. In DEM simulations, the horizontal profile of the vertical velocity component is well described by a Gaussian, whose variance grows linearly with height, as shown in Fig. 3-4. Applying linear regression gives $\mathrm{Var}(u_z) = 2.28zd + 1.60d^2$, which implies $b_s = 2.28d/2 = 1.14d$. To reproduce the spot diffusion length, we chose $\Delta z_s = 0.1d$ and $\Delta x_s = 0.68d$.

The typical excess volume carried by a spot can now be obtained from a single bulk diffusion measurement. From the previous chapter, we know that the particle diffusion length, b_p, is given by

$$b_p = \frac{\mathrm{Var}(\Delta x_p)}{2\Delta z_p} = \frac{\mathrm{Var}(w\Delta x_s)}{2w\Delta z_s} = wb_s.$$

We measure b_p in the DEM simulation by tracking the variance of the x displacements of particles that start high in the silo as a function of their distance dropped. We

find $b_p = 2.86 \times 10^{-3} d$ and thus $w = 2.50 \times 10^{-3}$. During steady flow in the DEM simulation, a typical packing fraction of particles is 57.9%, so a spot with radius $R_s = 2.60d$ influences on average 81.7 other particles. Thus we find that a spot carries roughly 20% of a particle volume: $V_s = 81.7 V_p w = 0.205 V_p$.

The three spot parameters so far (radius, R_s, diffusion length, b_s, and influence factor, w) suffice to determine the geometrical features of a steady flow, such as the spatial distribution of mean velocity and diffusion, but two more are needed to introduce time dependence. The first is the mean rate of creating spots at the orifice (for simplicity, according to a Poisson process). In the DEM simulation, particles exit a rate of mean rate of 4.40×10^3 s^{-1}, so spots carrying a typical volume $V_s = 0.205 V_p$ should be introduced at a mean rate of $\nu_s = 2.15 \times 10^4$ s^{-1}. The second remaining spot parameter is the vertical drift speed, or, equivalently, the mean waiting time between spot displacements, Δt, which can be inferred from the drop in mean packing fraction during flow. In the DEM simulation, we find that there are initially 9,400 particles in the horizontal slice, $50d < z < 70d$, which drops to 8,850 during flow. Choosing the spot waiting time to be $\Delta t = 8.68 \times 10^{-4}$ s reproduces this decrease in density in the spot simulation. The spot drift speed is thus $v_s = 0.1d/\Delta t = 115d/\text{s} = 34.5$ cm/s, which is roughly ten times faster than typical particle speeds in Fig. 3-4.

3.4 Spot model simulation

Having calibrated the five parameters (R_s, b_s, w, ν_s, v_s), we can test the spot model by carrying out drainage simulations starting from the same static initial packing as for the DEM simulations. For efficiency, a standard cell method (also used in the parallel DEM code) is adapted for the spot simulations. The container is partitioned into a grid of $10 \times 3 \times N_z$ cells, each responsible for keeping track of the particles within it, with $N_z = 30$ for $H_o = 110d$ and $N_z = 60$ for $H_o = 230d$. When a spot moves, only the cells influenced by the spot need to be tested, and particles are transferred between cells when necessary. Without further optimization, the multiscale spot simulation runs over 100 times faster than the DEM simulation.

The flow is initiated as spots are introduced uniformly at random positions on the orifice (at least R_s away from the edges) at random times according to a Poisson process of rate ν_s. (The waiting time is thus an exponential random variable of mean ν_s^{-1}.) Once in the container, spots also move at random times with a mean waiting time, $\Delta t = v_s/\Delta z_s$. Spot displacements in the bulk are chosen randomly from four displacement vectors, $\Delta\mathbf{r}_s = (\pm\Delta x_s, 0, \Delta z_s), (0, \pm\Delta x_s, \Delta z_s)$, with equal probability, so spots perform directed random walks on a body-centered cubic lattice (with lattice parameter $2\Delta z_s = 0.2d$). We make this simple choice to accelerate the simulation because more complicated, continuously distributed and/or smaller spot displacements with the same drift and diffusivity give very similar results. Spot centers are constrained not to come within d of a boundary; this distance was an arbitrary choice, and altering it allows the boundary layer velocities in the simulation to be tuned; a careful study of this issue remains a subject for future work. Once a spot reaches the top of the packing, it is removed from the simulation.

The particles in the simulation move passively in response to spot displacements without any lattice constraints. Consider a spot initially located at $\mathbf{r}_s$, being displaced by an amount $\Delta\mathbf{r}_s$. When it moves, all particles within a ball of radius R_s are displaced by an amount $-w\Delta\mathbf{r}_s$. Two different formulations were considered for the positioning of this ball:

1. Center the ball at the end of the spot displacement, at $\mathbf{r}_s + \Delta\mathbf{r}_s$
2. Center the ball at the midpoint of the spot displacement, at $\mathbf{r}_s + \Delta\mathbf{r}_s$

These two formulations are somewhat analogous to different definitions of stochastic differentials, with the first resembling the Itō formulation and the second being closer to the Stratinovich form [17, 111]. Since the diameter of the spot is larger than the spot step size, one might expect that the differences between the two approaches would be negligible in the simulation, but it turns out that it can have an important effect. The first definition is the more theoretically straightforward of the two, and also appears to be directly analogous to the void model: if a void moves from location A to location B, then the particle displacement should be centered on B. However,

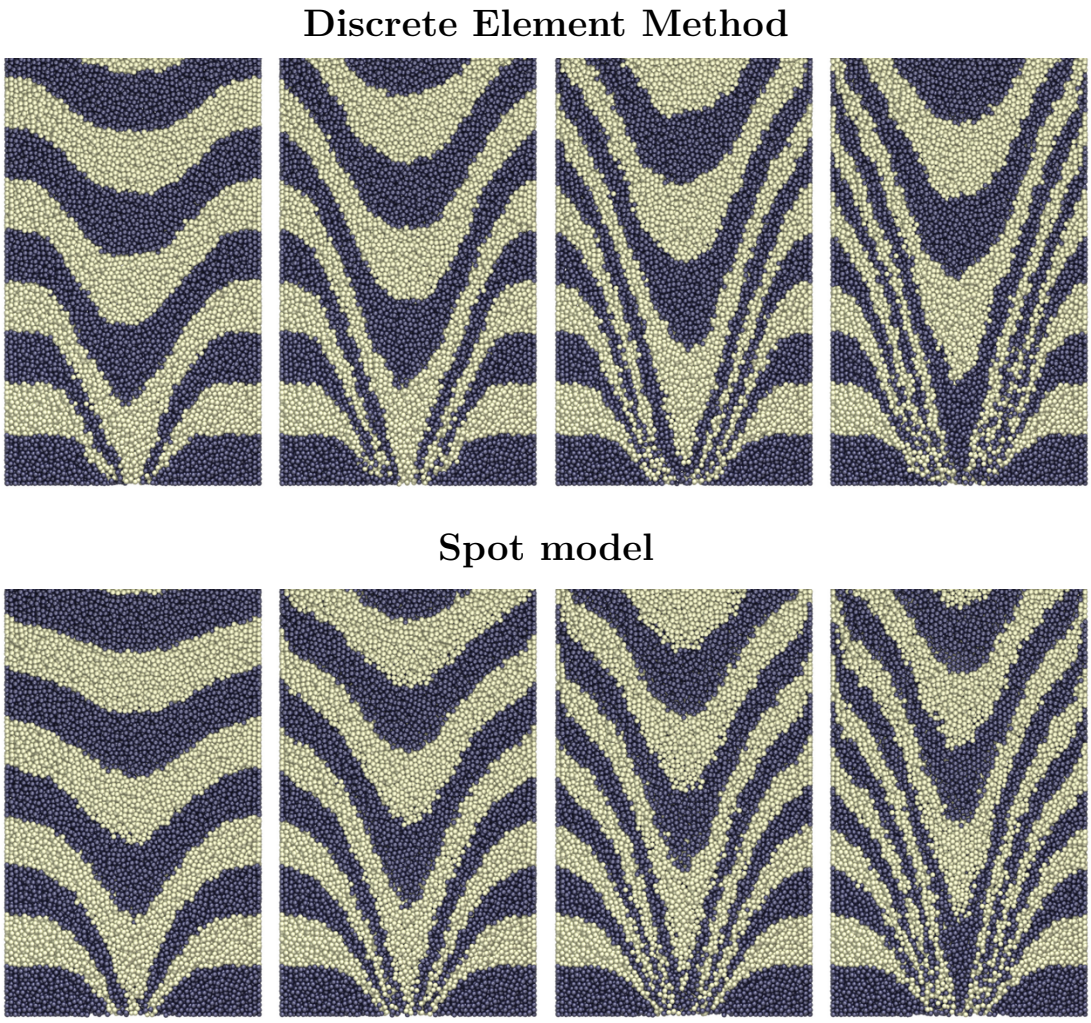


Figure 3-5: Time evolution of the random packing (from left to right) in DEM (top) and the spot simulation (bottom), for the $H_o = 230d$ system, starting from the same initial state. Each image is a vertical slice through the center of the silo near the orifice well below the free surface.

spot simulations using this method created some undesirable results, with the particles appearing to peel away from the walls.

To see why this may happen, it is helpful to consider the horizontal random walk motion of a spot. For simplicity, consider the x component of the spot motion, and assume that in this direction the spot is constrained to lie on lattice points labeled from 1 to N. Using the random walk prescription described above, we see that a spot at an internal lattice point n has a $\frac{1}{4}$ probability of moving to $n-1$, and a $\frac{1}{4}$ probability of moving to $n+1$. There is also a $\frac{1}{2}$ probability of staying at n, if the spot's motion at this step was in the y direction, leading to a zero displacement in the x direction. A spot at the boundary lattice point N has a $\frac{1}{4}$ probability of moving to $N-1$ and a $\frac{3}{4}$ probability of staying at N.

High in the container, the spots will be uniformly distributed on the N lattice sites. Particles will move rightwards if a spot at a lattice site from 2 to N moves leftwards. Using the Itō approach, the particle displacements from these spot motions will be centered on the final positions of the spots, namely lattice sites 1 to $N-1$. By a similar argument one can see that leftwards particle motion will be centered on lattice sites 2 to N. We therefore see that there is a disparity between the ranges over which the two motions are applied: rightwards particle motion occurs up to lattice site $N-1$, but leftwards motion occurs up to lattice site N. This difference causes a inward-pointing net displacement of particles near the boundary, causing them to peel away from the walls.

The Stratinovich approach resolves this problem, since both the leftwards and rightwards particle displacements are centered on the half-lattice points $1\frac{1}{2}, 2\frac{1}{2}, \ldots, N-\frac{1}{2}$. The approach is roughly analogous to thinking of a spot continuously moving from $\mathbf{r}_s$ to $\mathbf{r}_s + \Delta\mathbf{r}_s$, which continuously displaces the particles as it goes. Because of these advantages, the approach was employed for the simulations presented here. However, a more careful study of this issue is definitely needed. The above argument is specific to precise description of the random walk process, and for different spot motions, it may be that the Itō formulation is more appropriate.

To preserve realistic packings, we carry out a simple elastic relaxation after each

spot-induced block motion, as in Fig. 3-2(b). All particles within a radius $R_s + 2d$ of the midpoint of the spot displacement exert a soft-core repulsion on each other, if they begin to overlap. Rather than relaxing to equilibrium or integrating Newton's laws, however, we use the simplest possible algorithm: each pair of particles separated by less than d moves apart with identical and opposite displacements, $(d - r)\alpha$, for some constant $\alpha < 1$. Similarly, a particle within $d/2$ of a wall moves away by a displacement, $(\frac{d}{2} - r)\alpha$. Particle positions are updated simultaneously once all pairings are considered, but those within the shell, $R_s + d < r < R_s + 2d$, more than one diameter away from the initial block motion, are held fixed to prevent long-range disruptions.

It turns out that, due to the cooperative nature of spot model, only an extremely small relaxation is required to enforce packing constraints, mainly near spot edges where some shear occurs. Here, we choose $\alpha = 0.8$ and find that the displacements due to relaxation are typically less than 25% of the initial block displacement, which is at the scale of 1/10,000 of a particle diameter: $0.25w\Delta r_s \approx 2 \times 10^{-4} d$. Due to this tiny scale, the details of the relaxation do not seem to be very important; we have obtained almost indistinguishable results with $\alpha = 0.6$ and $\alpha = 1.0$ and also with more complicated energy minimization schemes. As such, we do not view the soft-core repulsion as introducing any new parameters.

3.5 Results

The spot and DEM simulations are compared using snapshots of all particle positions taken every 2×10^4 time steps. As shown in Figure 3-5, the agreement between the two simulations is remarkably good, considering the small number of parameters and physical assumptions in the spot model. It is clear *a posteriori* that the relaxation step, in spite of causing only minuscule extra displacements, manages to produce reasonable packings during flow, while preserving the realistic description of the mean velocity and diffusion in the basic spot model. Only one parameter, b_s, is fitted to the mean flow, but we find that the entire velocity profile is accurately reproduced

in the lower part of the container, as shown in Fig. 3-4, although the flow becomes somewhat more plug-like in DEM simulation higher in the container. Similarly, we fit w to the particle diffusion length in middle of the DEM simulation, $b_p = 2.86 \times 10^{-3} d$, without accounting for the elastic relaxation step, so it is reassuring that the same measurement in the spot simulation yields a similar value, $b_p = 2.73 \times 10^{-3} d$.

The most surprising findings concern the agreement between the DEM and spot simulations for various *microscopic* statistical quantities. First, we consider the radial distribution function, $g(r)$, which is the distribution of inter-particle separations, scaled to the same quantity in a ideal gas at the same density. For dense sphere packings, the distribution begins with a large peak near $r = d$ for particles in contact and smoothly connects smaller peaks at typical separations of more distant neighbors, while decaying to unity. As shown in Fig. 3-6, the functions $g(r)$ from the spot and DEM simulations are nearly indistinguishable, across the entire range of neighbors for the $H_o = 110d$ system. This cannot be attributed entirely to the initial packing because each simulation evolves independently through substantial drainage and shearing.

Next, we consider the three-body correlation function, $g_3(\theta)$, which gives the probability distribution for "bond angles" subtended by separation vectors to first neighbors (defined by separations less than the first minimum of $g(r)$ at $1.38d$). For sphere packings, $g_3(\theta)$ has a sharp peak at 60° for close-packed triangles, and another broad peak around $110 - 120^\circ$ for larger crystal-like configurations. In Fig. 3-7, we reach the same conclusion for $g_3(\theta)$ as for $g(r)$: The spot and DEM simulations evolve independently from the initial packing to nearly indistinguishable steady states.

The striking agreement between the spot and DEM simulations seems to apply not only to structural, but also to dynamical, statistical quantities. Returning to Fig. 3-3, we see that the two simulations have very similar spatial velocity correlations. Of course, the spot size, R_s, in the Spot model (without relaxation) was fitted roughly to the scale of the correlations in the DEM simulation, but the multiscale spot simulation also manages to reproduce most of the fine structure of the correlation function.

At much longer times, however, the random packings are no longer indistinguish-

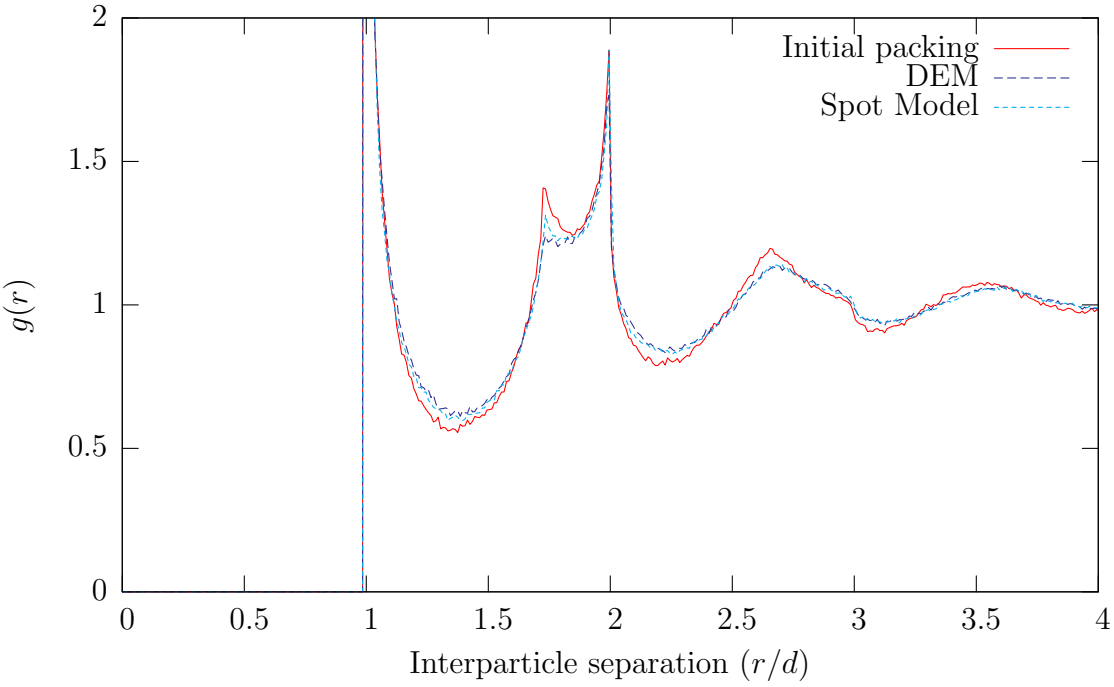


Figure 3-6: Comparison of radial distribution functions for particles in the region $-15d < x < 15d$, $15d < z < 45d$ for $H_o = 110d$ system. Three curves are shown on each graph, the first calculated from the initial static packing (common between the two simulations), and the second and third calculated for over the range 1.04 s $< t <$ 1.40 s.

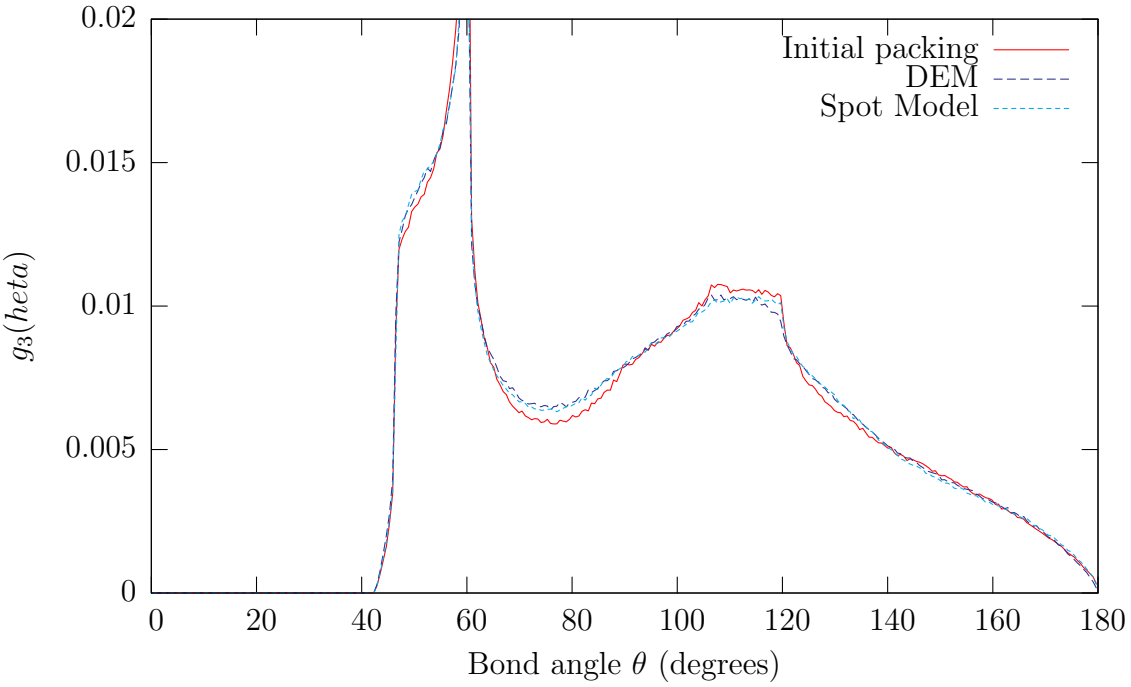


Figure 3-7: Comparison of bond angles for particles in the region $-15d < x < 15d$, $15d < z < 45d$ for $H_o = 110d$ system. Three curves are shown on each graph, the first calculated from the initial static packing (common between the two simulations), and the second and third calculated for over the range 1.04 s $< t <$ 1.40 s.

able, as a small tendency for local close-packed ordering appears the spot simulation. As shown in Fig. 3-8, the spot simulation develops enhanced crystal-like peaks in $g(r)$ at $r = \sqrt{3}d$, $2d$, ... The number of particles involved, however, is very small ($\sim 2\%$), and the effect seems to saturate, with no significant change between 8s and 16s. This is consistent with even longer spot simulations in systems with periodic boundary conditions, which reach a similar, reproducible steady state (at the same volume fraction) from a variety of initial conditions [104]. In all cases, the spot algorithm never breaks down (e.g. due to jamming or instability), and unrealistic packings with overlapping particles are never created.

The structure of the flowing steady state is fairly insensitive to various details of the spot algorithm. For example, changing the relaxation parameter (in the range $0.6 \leq \alpha \leq 1.0$), rescaling the spot size (by $\pm 25\%$), and using a persistent random walk (for smoother spot trajectories), all have no appreciable effect on $g(r)$. On the other hand, decreasing the vertical spot step size (in the range $0.025d \leq \Delta z \leq 0.1d$) tends to inhibit spurious local ordering and reduce the difference in $g(r)$ between the spot and DEM simulations (e.g. measured by the L_2 norm). Therefore, our spot algorithm appears to "converge" with decreasing time step (and increasing computational cost), analogous to a finite-difference method, although this merits further study.

3.6 Conclusions

Our results suggest that *flowing* dense random packings have some universal geometrical features. This would be in contrast to static dense random packings, which suffer from ambiguities related to the degree of randomness and definitions of jamming [134, 99]. The similar packing dynamics in spot and DEM simulations suggest that geometrical constraints dominate over mechanical forces in determining structural rearrangements, at least in granular drainage. Some form of the spot model may also apply to other granular flows and perhaps even to glassy relaxation, where localized, cooperative motion also occurs [35, 142].

The spot model provides a simple framework for the multiscale modeling of liquids

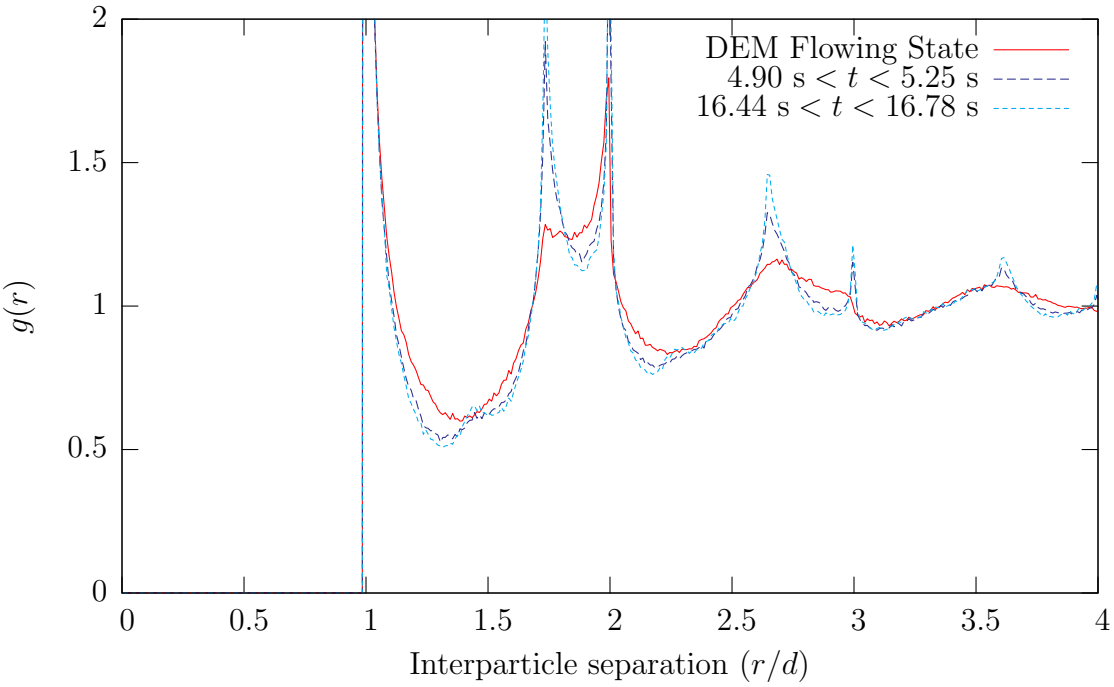


Figure 3-8: Evolution of the radial distribution function $g(r)$ for $H_o = 230d$ in the region $-15d < x < 15d$, $15d < z < 45d$. The spot simulation (dashed curves) reaches a somewhat different steady state from the DEM simulation (solid curve), after a large amount of drainage has taken place.

and glasses, analogous to dislocation dynamics in crystals. Our algorithm, which combines an efficient, "coarse-grained" simulation of spots with limited, local relaxation of particles, runs over 100 times faster than fully particle-based DEM for granular drainage. On current computers, this means that simulating one cycle of a pebble-bed reactor [130] can take hours instead of weeks, although a general theory of spot motion in different geometries is still lacking. In any case, we have demonstrated that dense random-packing dynamics can be driven entirely by the motion of simple, collective excitations.

Chapter 4

Further studies of the spot model

4.1 Do voids exist?

Other models of granular and glassy materials often make use of the concept of particle-sized voids in the material. Motion can then be described by a single neighboring particle jumping into the free space. As a test of these theories, it is interesting to ask whether such voids actually exist. In experiment, confirming or disproving the existence of these is difficult, but in the DEM and spot simulations this can easily be carried out.

A code was written to analyze the free space size distribution for the granular packing. The entire container is first covered with a fine grid with spacing $\lambda = \frac{1}{50}d$, and at each point on the grid the maximum size of sphere that can fit there is calculated. Of course, we may miss larger voids which are centered at points not on the grid. However, by geometrical considerations, we see that if the maximum radius found on the lattice is r, then the maximum radius for a void anywhere in the packing is bounded above by $r + \sqrt{3}\lambda/2$.

For the initial random packing, the maximum size of sphere that could be found had radius $0.466207d$. Using the above calculations the absolute maximum size for a void in the packing is $0.483527d$, and thus no particle sized voids exist in the packing.

Furthermore, of the 36 regions in the packing where a void of radius $0.4d$ or greater can be fitted, 31 of these lie within a particle diameter of the walls, and of

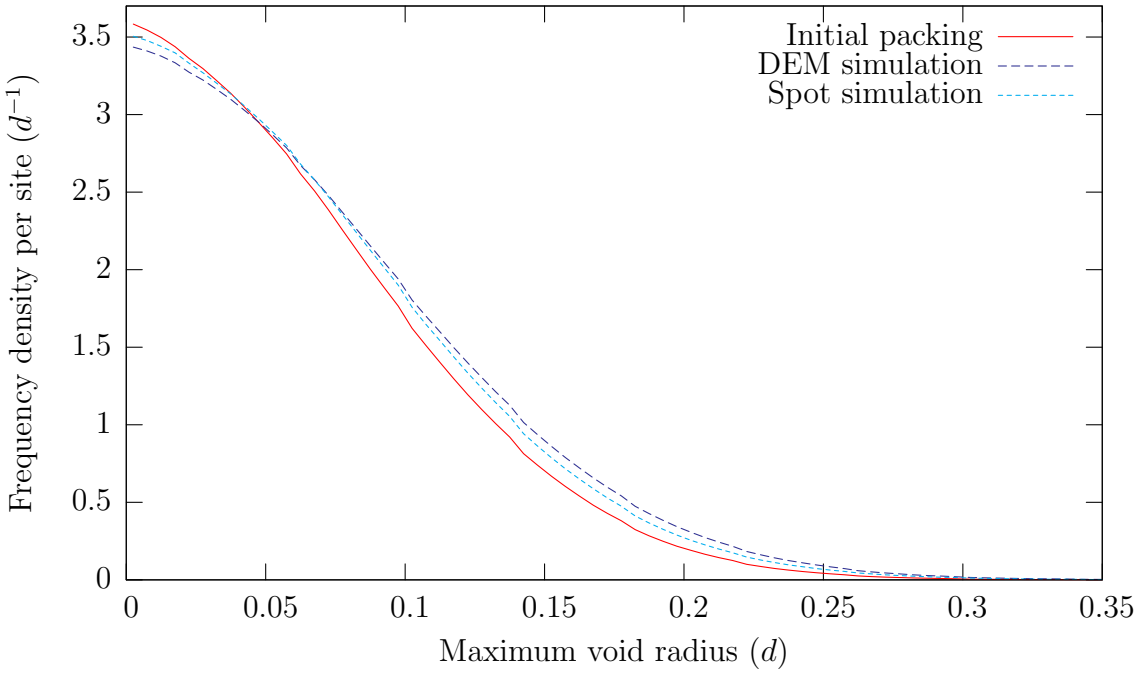


Figure 4-1: Distribution of radii of free spaces for the DEM and spot simulations, time averaged over the interval $80\tau \leq t < 100\tau$.

Time (τ)	80	82	84	86	88	90	92	94	96	98
MD	0	3	2	0	2	1	2	1	1	1
Spot	0	0	0	0	1	1	1	1	1	1

Table 4.1: Number of particle-sized free spaces present in the spot and DEM simulations for ten simulation snapshots starting from $t = 80\tau$.

the remaining five, the largest void that can be fitted has radius $0.421792d$. Thus in a fully three dimensional granular packing, the likelihood of finding significant areas of empty space may be even less.

The above code was also applied to analyze the free space size distribution in the flowing random packings generated by the spot and DEM simulations. Since this requires testing many frames, the grid spacing was doubled to $\frac{1}{25}d$. Also, attention was restricted to the strip $20d < z < 60d$ to avoid orifice and free surface effects. Furthermore, to reduce lattice effects, only voids whose nearest contact was with a particle and not a wall were counted.

Figure 4-1 shows the distribution of hole sizes in the DEM and spot simulations during the flow. The vertical axis has been scaled to represent the frequency density

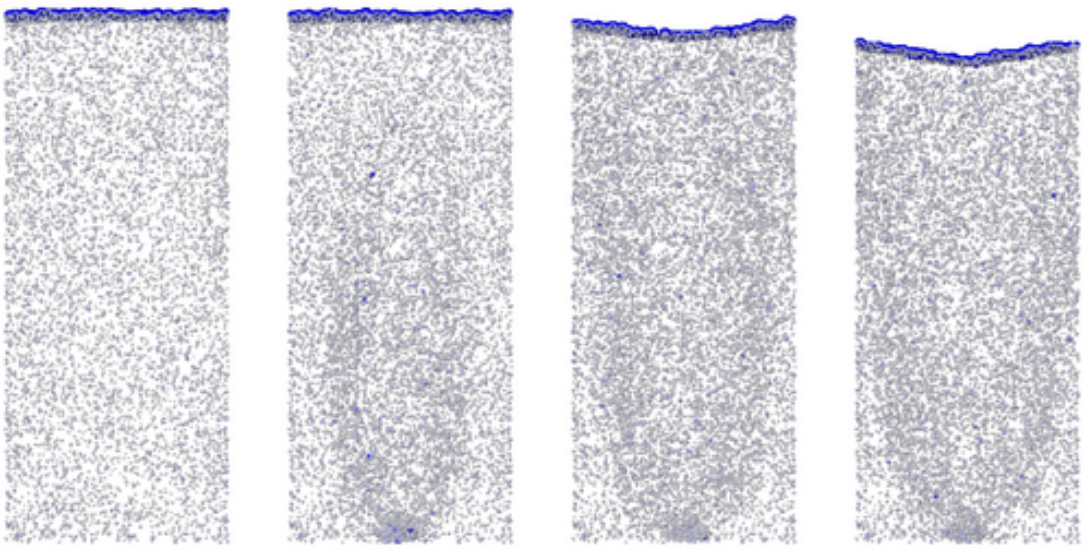


Figure 4-2: Free space plots for the DEM simulation for $t = 0, 30\tau, 60\tau, 90\tau$ showing all voids with radius $d/4$ or larger. Small voids are plotted in white, larger voids are plotted in blue, and the blue line at the top represents the free surface. Images generated using *Raster3D* [86].

of finding a hole of a given size per test site. Thus, the total area under the graph is approximately 40%, which is in rough agreement with proportion of free space in a typical granular packing.

As expected, the flow causes a general increase in the size of free spaces in the packing. This alteration in the distribution takes around twenty-five frames to occur, and remains roughly in equilibrium from then on. During the flow it is also possible to find voids in the material that are more than particle in radius – table 4.1 shows the number of such for ten simulation snapshots starting from $t = 80\tau$. Compared to the amount of flow, the number of particle-sized empty spaces is extremely small, providing further evidence that the void-based models are a very poor description of the microscopic particle dynamics. Also, every single void in the table was located either at the front or back wall of the container, suggesting that their effect may be even less in a fully three dimensional simulation.

Since the position of free space is very important to the spot model, it was decided to go further and explicitly track its position during the flow. The previous program was modified to dump the positions of all voids that have radii greater than $d/25$

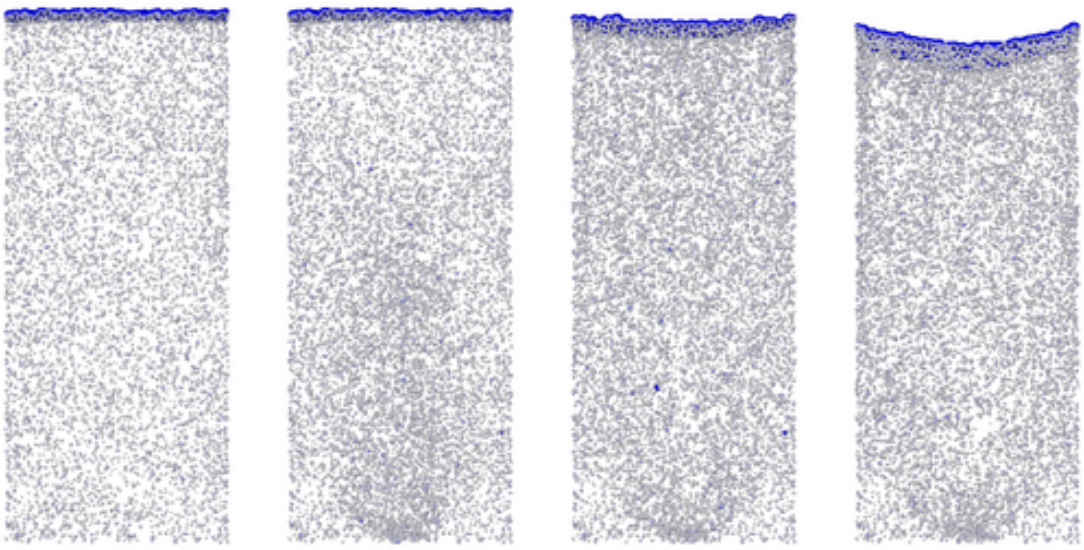


Figure 4-3: Free space plots for the spot simulation for frames $t = 0, 30\tau, 60\tau, 90\tau$.

to a file. These were then rendered, coloring small voids white, and larger voids progressively more blue. The results for four frames covering the transition from the static packing to the steadily flowing state are shown in figures 4-2 and 4-3 for the DEM and spot simulations respectively.

In both simulations, we see that the amount of free space increases first in the neighborhood around the orifice before spreading out into a roughly parabolic region, as expected. However, one very interesting feature of the DEM simulation is that the free space tends to be concentrated in two bands to either side of the orifice. This feature is completely absent in the spot simulation – according to the spot model, there should be a direct correspondence between free volume and downwards velocity. The result suggests that free volume may be better associated with shear, and a more complete and quantitative comparison of density fluctuations in the two models is considered in the following section.

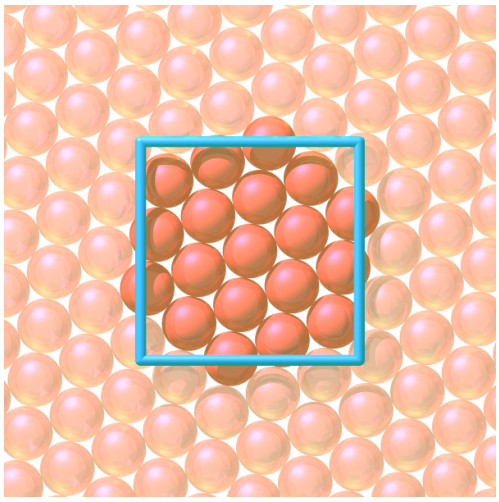
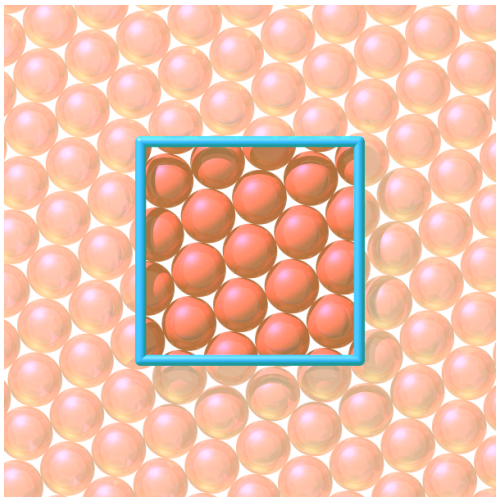
(a)  (b) 

Figure 4-4: Two simple methods of estimating packing fraction in a two dimensional packing, based on a $4d$ square test region. In (a), the packing fraction is computed by counting up the volume of all particles whose centers are in the test region, and dividing by the square's area. In (b), particles which overlap the boundary of the region contributed only their volume which is within the square.

4.2 Computing packing fraction using a Voronoi tessellation

4.2.1 The problems of accurately tracking free volume

While the snapshots shown in figures 4-2 and 4-3 were originally created to search for voids, they suggest that the distribution of free volume in the spot and DEM simulations may be considerably different, and since spots carry free volume, and it would be beneficial to accurately track this quantity. Unfortunately, accurately measuring the distribution of free volume in the simulation is difficult, since the changes in packing fraction may be very small. In a monodisperse static granular material, the structure will be similar to a Random Close Packing (RCP) of spheres, with a packing fraction of 63%. During flow, this may decrease to approximately 58%, a change of only 5%. If we were interested in a time-averaged packing fraction, or the packing fraction in a large spatial region, then we could get reasonably accurate results on this scale by averaging over a large amount of particle data. However, to potentially track spots, we would like to measure the instantaneous packing fraction

in regions on the same size as the spot, and given the particle discreteness at this level, there is not enough data to achieve an accuracy level of less than 5%.

To illustrate this, consider the two dimensional regular hexagonal packing of particles shown in figure 4-4, where the orientation of the packing has been offset by 10° from the vertical. The packing fraction of this lattice should be a constant, with value

$$\frac{\pi(d/2)^2}{6 \times d \times (d/\sqrt{3})} = \frac{\pi}{2\sqrt{3}} = 90.689968\%.$$

Suppose now that we wish to estimate the packing fraction based on the particles in the blue square, which has side length $4d$. A first attempt would be to count up the number of particle centers which lie within the box (as shown in figure 4-4(a)) and then estimate the packing fraction by dividing the area of those particles by the area of the square. Unfortunately this leads to very large errors: by moving the box around by small amounts on the lattice, it is possible to have between 17 and 21 particle centers within the box, leading to a packing fraction estimate between 83% and 103%. A second approach is shown in figure 4-4(b). In this scheme, particles which are at the edges of the square contribute only their area which is within the square to the density estimate. This gives better accuracy, but as shown in figure 4-5 can still give errors on the order of 1%.

It is possible to remove this error by making use of Voronoi cells. In a given packing, the Voronoi cell [140] for a particle is defined as all the volume which is closer to that particle than any other. Using this prescription, it is possible to define a packing fraction at the level of a single particle, as the ratio of a particle's volume to the volume of its Voronoi cell. A local estimate on the scale of a spot can be obtained by averaging over the particles and cells in a small region.

In two dimensions, the Voronoi cells are polygons, the sides of which are the perpendicular bisectors between neighboring particles. In the current example, the Voronoi cells would be regular hexagons. Thus, as shown in figure 4-6(a), the packing fraction could be computed by summing up the volume of all the Voronoi cells in a small region. Regardless of precisely how the cluster of particles is chosen, the method

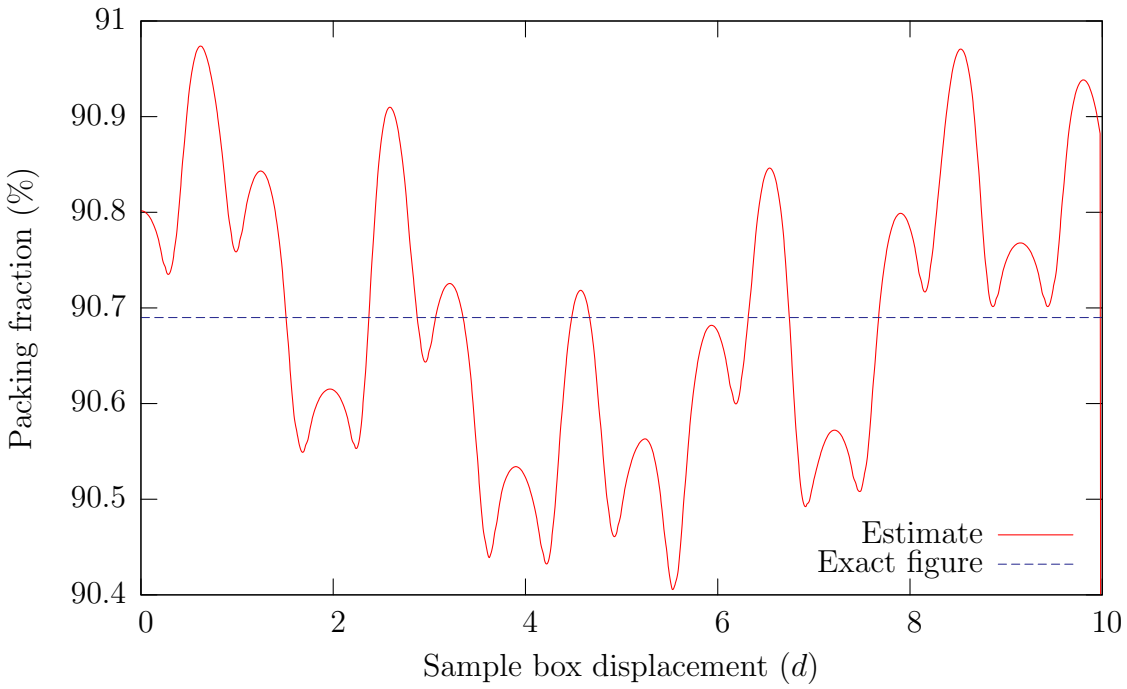


Figure 4-5: Graph showing how the local density estimate would change if the test region in figure 4-4(b) is continuously moved upwards.

will always give the exact answer for the packing fraction of $\pi/2\sqrt{3}$.

Although this technique was demonstrated on a regular packing, it will also work on a random packing as well (figure 4-6(b)). The method seems theoretically advantageous, since to estimate packing fraction, it respects the packing structure, rather than cutting particles.

4.2.2 Computation of three-dimensional Voronoi cells

The computation of Voronoi cells is a well-studied problem, particularly in the computer science literature [11]. Several methods are readily available, and one is built into the popular mathematical package Matlab, which makes use of the dual Delaunay triangulation, computed via Qhull [3, 14]. However, to provide greater control and flexibility, it was chosen to create an algorithm specifically for the computation in granular simulation, that could directly compute cells individually.

In a three dimensional random packing, the Voronoi cell of a particular particle will be a convex irregular polyhedron, the faces of which are the perpendicular bisectors of

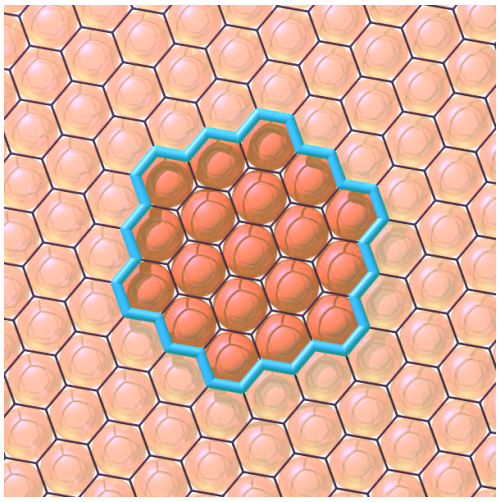
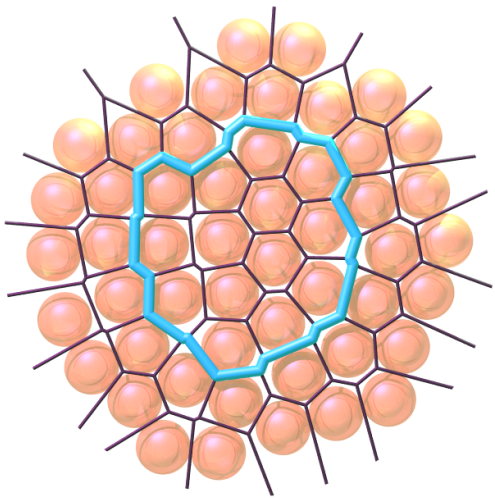
(a)  (b) 

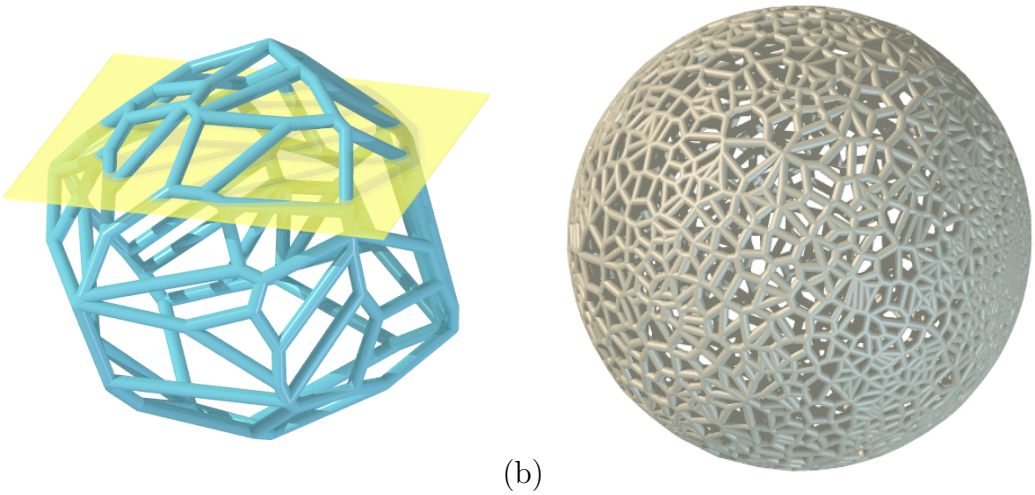
Figure 4-6: The density of the local region can be accurately found by computing the Voronoi cells of all the particles, and then dividing the particle volume be the Voronoi cell region in a small cluster. For the two dimensional hexagonal packing, the Voronoi cells are all regular hexagons, and the computation yields the exact packing fraction of $\pi/2\sqrt{3}$. The algorithm can also be applied to an arbitrary amorphous packing (b), in which case the Voronoi cells will be irregular polygons (in two dimensions) or polyhedra (in three dimensions).

its neighbors. While in practice, we expect the Voronoi cells will be reasonably simple polyhedra, in a general situation, there is no theoretical limit on how complicated they may be. For example, consider a single particle surrounded by a densely packed spherical shell of particles, all at some large radial separation R. The Voronoi cell will be approximately a sphere with radius $R/2$, with many facets due to the particles in the shell. As R increases, the number of facets will increase without limit. While a Voronoi cell of this type may be unlikely to occur in practice, it highlights the fact that our algorithm cannot make assumptions about the form of the polyhedra – we must be able to handle completely arbitrary convex polyhedra.

To compute a Voronoi cell for a particle with position $\mathbf{x}_p$, the algorithm takes the following approach:

1. Surround the particle with an initial Voronoi cell consisting of the entire container volume.

2. Set $r_{\text{test}} = \frac{d}{2}$.



(a) (b)

Figure 4-7: The basic computational challenge in computing three dimensional Voronoi cells is to take an irregular polyhedron, and recompute its vertices based on cutting off a plane (a). The algorithm developed in this thesis can be tested on arbitrarily complicated polyhedra, such as this approximation to a sphere, made by cutting many planes all of unit distance from the origin (b).

3. Find all the neighboring particles with positions $\mathbf{x}$ in the spherical shell defined by $r_{\text{test}} < |\mathbf{x}_p - \mathbf{x}| < r_{\text{test}} + d$.

4. For each of these particles, cut the Voronoi cell by the plane which is the perpendicular bisector between $\mathbf{x}$ and $\mathbf{x}_p$.

5. Compute the maximal distance R of a vertex of the Voronoi cell to its center $\mathbf{x}_p$.

6. If $2R > r_{\text{test}}$, then increase r_{test} by d, and go back to step 3. If $2R \leq r_{\text{test}}$, the computation is complete.

By looping over concentric spherical shells, we can efficiently test those particles nearest $\mathbf{x}_p$ which will most likely create the facets of the Voronoi cell. When we know that all particles which are left in the packing are further than $2R$ from the particle, then we can be sure that the cutting planes of those particles (which at the very least, are a distance of R from $\mathbf{x}_p$) would not intersect the Voronoi cell, and thus do not need to be tested.

The remaining computational challenge comes from item 4, which is shown schematically in figure 4-7(a): we must be able to computationally represent an arbitrary convex polyhedra, and be able to recompute the polyhedra based on cutting off an arbitrary plane. To represent the polyhedra, three pieces of information were stored:

- A list of polyhedra vertices $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N$.
- A table of edges: for each vertex i, store the three connected vertices $e(0,i)$, $e(1,i)$, and $e(2,i)$. The three edges are stored in a handed order, so that they trace around the vertex according a right hand rule with respect to an outward-pointing normal.
- A relational table describing how vertices are connected back to each other: for each vertex i, store three additional numbers $l(0,i), l(1,i), l(2,i)$ that satisfy the property
$$e(l(j,i), e(j,i)) = i.$$

The first and second items are obvious requirements, although it should be noted that in this representation, vertices always have three edges. The case of four or more edges (as would occur in a octahedron or icosahedron) is treated as degenerate, since it requires that the random cutting planes precisely intersect with an existing vertex. In practice this is not a problem, since due to numerical inaccuracy, any intersection between four edges is represented by two proximate, connected vertices.

The third requirement listed above does not appear obvious, but is useful during computation. A common task during the plane-cutting involves breaking and reforming edges. Suppose we consider a vertex i, that is connected to a vertex j, so that $e(\lambda, i) = j$. We know that one of the edges of vertex j points back to vertex i, but without the relational table, we do not know which one, without searching over the three elements $e(0,j), e(1,j), e(2,j)$. The relational table tells us which edge this is immediately.

Given this representation of a polyhedron, we now consider cutting it by an arbitrary plane, $\mathbf{x} \cdot \mathbf{n} = a$. We want to remove any part of the polyhedra which intersects

the half-space $\mathbf{x} \cdot \mathbf{n} > a$.

The algorithm is described in detail below. The process starts by picking a vertex, and moving across the polyhedra to search for an edge which intersects the plane (items 1 to 6). To do this, the property of convexity is exploited. If the plane intersects the cell, then one of the three neighbors to a vertex will always be closer to the plane; the only time this fails is if the cell and the plane are disjoint. Once an intersected edge is found, the algorithm traces around intersected facets to find all intersected edges (items 7 to 9). On each intersected edge, a new point is created at the intersection point, and these points are then connected to create a new facet (items 10 to 11). The algorithm then deletes all the old vertices which were in the removed half-space $\mathbf{x} \cdot \mathbf{n} > a$ (items 12 to 13).

1. Pick an arbitrary vertex i.

2. Let $a_i = \mathbf{x}_i \cdot \mathbf{n}$. If $a_i > a$, skip to step 5.

3. Compute $a_j = \mathbf{x}_j \cdot \mathbf{n}$ for $j = e(0,i), e(1,i), e(2,i)$. If all three computed values are less than a_i, the half-space does not intersect the polyhedron; exit. Otherwise, pick a j such that $a_j > a_i$.

4. If $a_j > a$, skip to step 7. Otherwise redefine i to be j, and go back to step 3.

5. Compute $a_j = \mathbf{x}_j \cdot \mathbf{n}$ for $j = e(0,i), e(1,i), e(2,i)$. If all three computed values are all more than a_i, the polyhedron lies wholly within the half-space; exit. Otherwise, pick a j such that $a_j < a_i$.

6. If $a_j > a$, redefine i to be j, and go back to step 5. Otherwise, swap i and j.

7. We have now found an edge, between i and j, that intersects the cutting plane, so that $a_i < a < a_j$, meaning that i is not in the half-space to be removed, and j is. Consider an arrow from i pointing j.

8. Trace around the facet to the right of the arrow, by following the arrow and going right at every vertex, until reaching a point which lies outside the half-space. This point is on an intersected edge.

9. Starting from this intersected edge, trace around the next facet, until another intersected edge is found. Repeat this process, finding all the intersected edges, until returning to the edge between i and j. Define $(i_1, j_1), (i_2, j_2), \ldots, (i_n, j_n)$ to be all the intersected edges, where all the i vertices lie outside the half-space, and all the j vertices lie inside the half space.

10. Define new vertices k_α at the points where the edges (i_α, j_α) intersect the cutting plane.

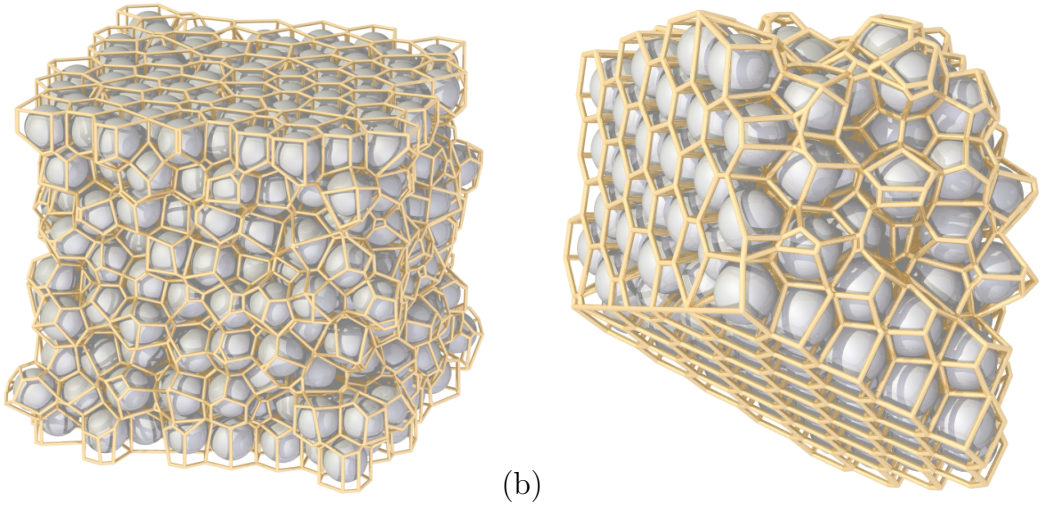
11. For each $\alpha = 1, \ldots, n$ redefine the edge from i_α to j_α as going from i_α to k_α. Connect k_α to $k_{(\alpha+1)}$ for $\alpha = 1, \ldots, (n-1)$, and connect k_n to k_1.

12. Search the vertices $j_1, j_2, \ldots, j_n$, and find all the vertices which are connected to these; call these $j_{n+1}, \ldots, j_m$.

13. Delete vertices $j_1, j_2, \ldots, j_m$.

Before using this in the granular simulation, the algorithm was tested by constructing many arbitrary polyhedra. The cell used to make figure 4-7(a) was created using the algorithm. Figure 4-7(b) represents a difficult test of the algorithm, creating an approximation to a sphere by intersecting many planes all of the form $\mathbf{x} \cdot \hat{\mathbf{n}} = d$, where $\hat{\mathbf{n}}$ is a random unit vector. This polyhedron was then intersected with the plane $x = 0$, which removes approximately half of the vertices, and is a good test of the deletion step.

Figure 4-8(a) shows the algorithm results in a local region when applied to the DEM simulation data from figure 3-5. In this picture, the front wall has been rotated to the top, and the Voronoi cells mesh together to form a planar surface. This demonstrates one of the advantages of the direct computation: it is very easy to correctly mesh the cells to walls by cutting with the appropriate planes. Figure 4-8(b) shows an example of this, where cells have been cut to correctly handle a packing at an oblique corner. Figure 4.2.2 is a more complicated example, looking up from below at a conical funnel.



(a) (b)

Figure 4-8: A small region of particles and their computed Voronoi cells, taken from the DEM simulation data of chapter 3, and rotated so that the front wall is facing upwards. The plane-cutting algorithm can also be applied to handle packings of particles at oblique boundaries (b).

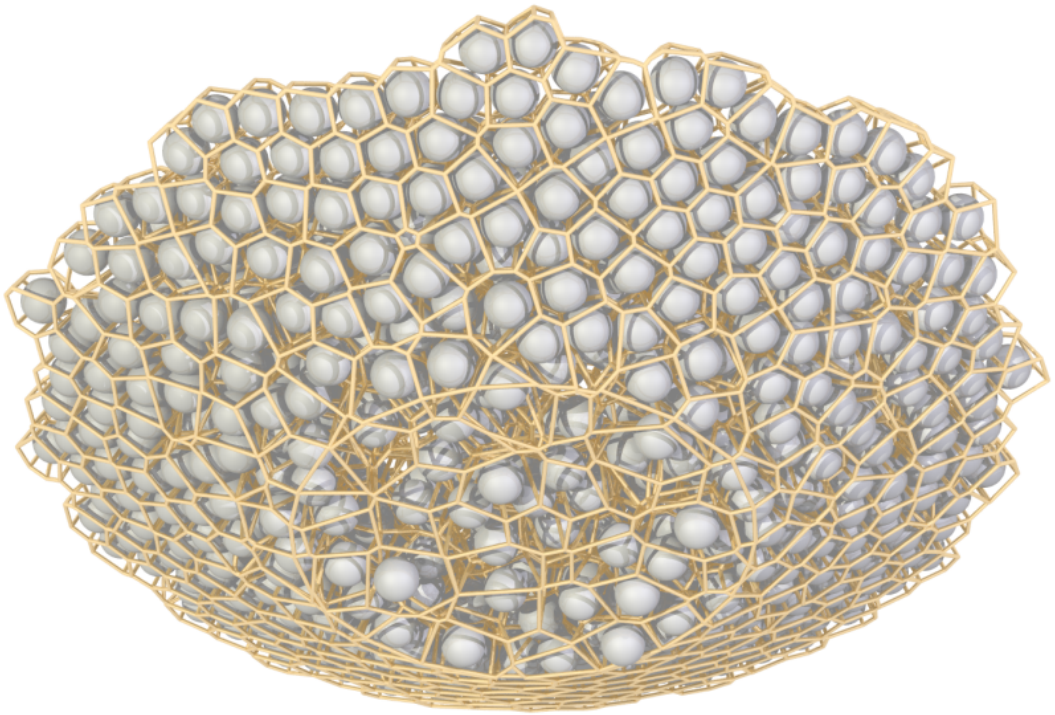


Figure 4-9: An example result of the Voronoi cell algorithm, looking up from below at particles falling out of a conical funnel (using data from chapter 5). For the Voronoi cells next to the wall, the cells are cut by approximating the conical wall as locally planar, and cutting the cell with that plane. This yields very satisfactory results, although small discrepancies are visible, particularly at the circular interface between the cone and the exit pipe.

4.2.3 Local density computation

To calculate the local density requires one additional computational step: we must be able to take the computed polyhedra of the previous section and find their volume. This is done by the following procedure:

1. Pick an arbitrary vertex $\mathbf{v}$.
2. Choose a facet, whose vertices are $\mathbf{x}_0, \mathbf{x}_1, \ldots, \mathbf{x}_n$.
3. Compute the volume of the pyramid whose base is the facet, and whose apex is $\mathbf{v}$, by dividing it into tetrahedra with vertices $(\mathbf{v}, \mathbf{x}_0, \mathbf{x}_\alpha, \mathbf{x}_{\alpha+1})$ for $\alpha = 1, 2, \ldots, (n-2)$.
4. Return to step 2, and repeat over all facets.

An excellent test of this algorithm was to compute the Voronoi cells for an entire simulation snapshot, and then check that the total volume of all the Voronoi cells equalled the volume of simulation container. Calculating the Voronoi cells for all 55,000 particles in the container took approximately 9.5 s on a 2GHz Athlon machine. Thus, as a rough guide, the algorithm computes 5000 cells per second although many factors such as system geometry or computer architecture could significantly alter this.

Using this algorithm, the local packing fraction at a point was defined by finding all the particles whose centers were within a distance $s = 2.2d$ of the point. Figures 4-10 show snapshots of local packing fraction in the DEM and spot simulations from the previous chapter. In both simulations, the initial packing fraction is approximately 63%, although we can see that at the boundaries the computed packing fraction decreases, since the Voronoi cells which are pressed against the walls have slightly more volume.

The snapshots highlight very large differences between the packing fraction distributions in the two simulations. The upwards velocity of the drop in density is roughly equal, as would be expected from the calibration of spot velocity. However in the DEM simulation, the drop in density takes place in two bands either side of

the central region, which correspond to the particles undergoing the highest amount of shear, and is a direct observation of the well-know concept of shear dilation. The presence of these bands is in good agreement with the spatial distribution of free volume that was seen in 4-2.

In the spot simulation, the largest density drop is located in the central region above the orifice, in the region of highest velocity. This suggests a fundamental problem with the formulation of the spot model up to this point: since spots each move at constant velocity, the only way to make a packing move faster is to send more spots through, resulting in a larger density drop. The above results suggest that density drop may better be linked to shear, and not velocity. This subject will be returned to in later chapters.

4.3 Alternative spot models

4.3.1 A model of the free surface

Describing the behavior of the free surface of a granular pile has received much attention from the literature. Typically the free surfaces of dry granular materials will form at a critical angle, known as the angle of repose. If perturbed, they will tend to reform at this angle again. This became a useful metaphor for "Self-Organized Criticality", a notion applicable in describing many natural processes [12, 13]. Recently, work on the evolution of the free surface has concentrated on the rotating drum geometry [109, 68, 40, 101, 22].

In granular drainage, the free surface also tends to form heaps with slopes at the critical angle. This can be seen in an hourglass, where the free surface takes the form of an indented cone. Particles on the surface of the cone fall radially inwards to the cone's apex, before moving downwards through the packing. While the spot model was originally proposed in the context of bulk flow, it is interesting to see whether it could be used in any way to describe the free surface also. In the previous simulations the free surface was ignored, and the figures only shown the bulk region, well below

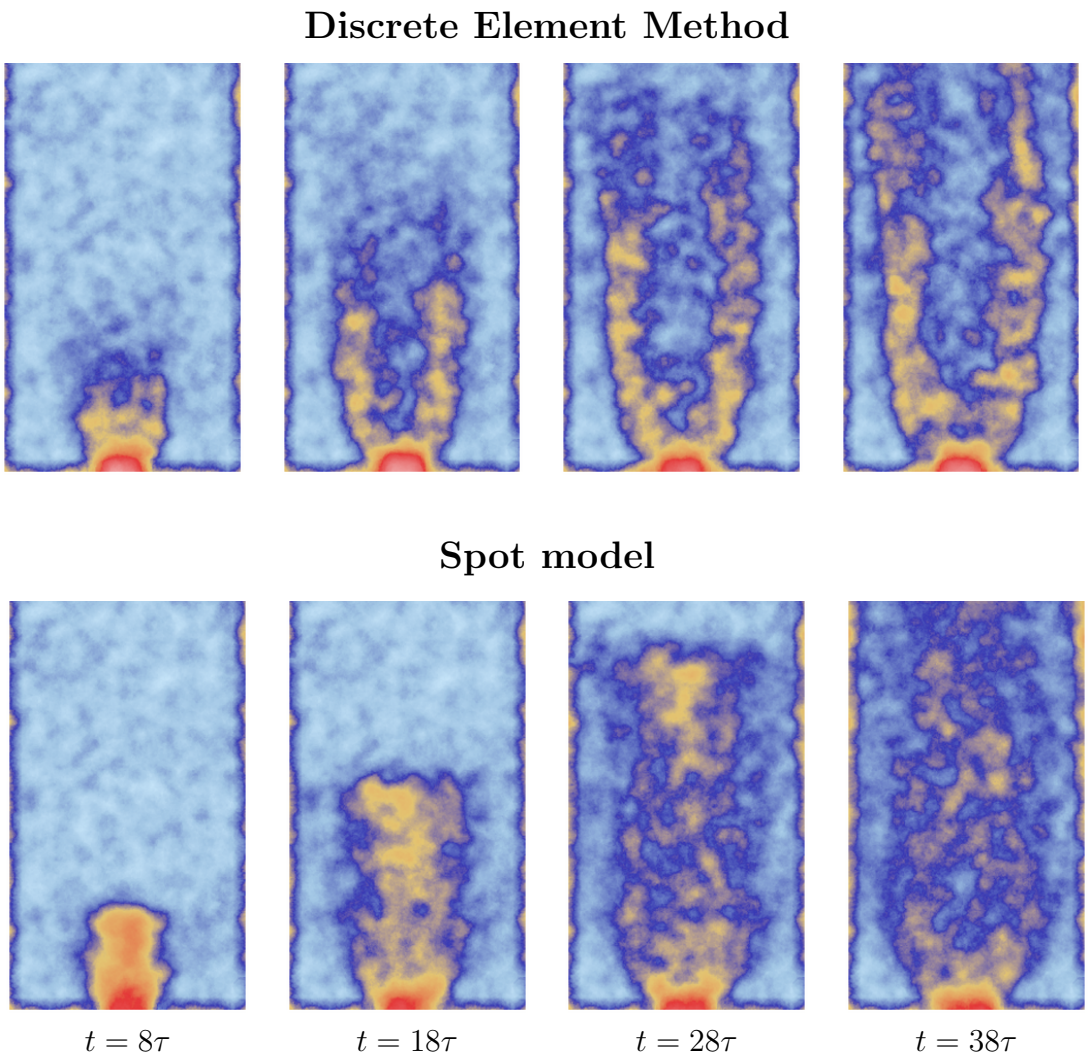


Figure 4-10: Plots of instantaneous local packing fraction computed using the Voronoi algorithm for the two simulations shown figure 3-5. Volume fractions of 50%, 57%, 60%, and 63% are shown using the colors of red, yellow, dark blue, and cyan respectively. Colors are smoothly graded between these four values to show intermediate volume fractions.

the top of the packing. The approach of the previous chapters was for the spots to carry out a completely unbiased random walk, and as shown in figure 4-11, this does not capture the behavior seen in DEM. Particles at the free surface essentially follow the parabolic velocity field in the bulk, which is inappropriate and does not capture the particle avalanching.

In the void model, the evolution of the free surface has been addressed, by proposing a very simple modification to the walk process [25]. In the bulk of the packing, when a void always generally has two particles in the lattice points above it, and in this situation it picks one at random. However, in the case when only one of these two sites is filled with a particle, the void always moves in the direction of the particle. A void is only removed from the simulation when both of the sites above it are vacant. This simple modification suffices to create heaps and avalanching at the free surface as when a void reaches the heap, it travels diagonally upwards along the heap surface.

This, in effect, creates a simple biasing of the random walk: there are two locations it can jump to, and it chooses randomly amongst the available options. This idea can be adapted to the spot model. Suppose that a spot can potentially move to N locations, and it would influence p_i particles if it moved to position i. Let $q = \sum_{i=1}^{N} p_i$. If $q = 0$ then remove the spot from the simulation. Otherwise, let

$$\mathbb{P}(\text{Spot moves to } i) = \frac{p_i}{q}.$$

In the bulk, where the density of particles is approximately constant, this does not alter the process by a large amount, but at the free surface, spots will bias their motion to create heaps. As shown in the left snapshot in figure 4-12 this correction dramatically improves the free surface behaviour in the spot model. However, it can also be seen that the packing of the particles at the free surface in this model is unphysical: individual particles are floating, and not in contact with their neighbors. There is no explicit gravity in the spot model, but in the bulk, particles are kept in contact by geometrical constraints from their neighbors. At the free surface, the particle rearrangement during avalanching causes them to separate.

This can be corrected by a further modification of the spot model. In the previous implementation, when a spot moves by $\mathbf{r}_s$, then the particles experience a displacement $-w\mathbf{r}_p$, where w is a fixed quantity. Suppose a spot is going to influence p particles, each of volume V_p. If spots are thought of as carrying a completely fixed amount of free volume V_s, then another possible approach would be to let $w = V_s/pV_p$, so the spot's influence is divided equally among the particles in range. In the bulk, where the particles are roughly at constant density, this modification has little effect. However, at the free surface, where p is lower, the spots give a larger downwards push, and stop particles from drifting upwards.

When this model was implemented, an additional constraint was needed: if $p < 20$, then the spot displacement was calculated based on $p = 20$. This removes the possibility of a spot only in range of a few particles giving an extremely large push to them. Physically, one can think of this constraint as saying that spots start to partially "evaporate" once they get close to the limit of the free surface.

The results of this model, shown in the right snapshot of figure 4-12, appear very promising. It has all the qualitative features of the DEM simulation, and the particles near the free surface remain packed. The angle of repose of the free surface does not precisely match DEM, but this could be tuned by altering the details of the random walk weighting. We leave this as a promising direction for further study.

4.3.2 A two dimensional spot simulation with relaxation

In figure 2-9, a spot model without relaxation was carried out on a two-dimensional regular hexagonal packing, and it is interesting to see whether how a spot model with relaxation will work for this case. Particles in a two-dimensional packing have even stronger geometrical constraints, and have a tendency to crystallize in monodisperse situations, and it is not clear *a priori* whether the spot model microscopic mechanism will be able to handle this case.

Figure 4-13 show two snapshots from a simulation of this type. The spot simulation has no problem running in this geometry, and generates valid, non-overlapping packings, although several of the features are quite surprising. Large areas of crystal-

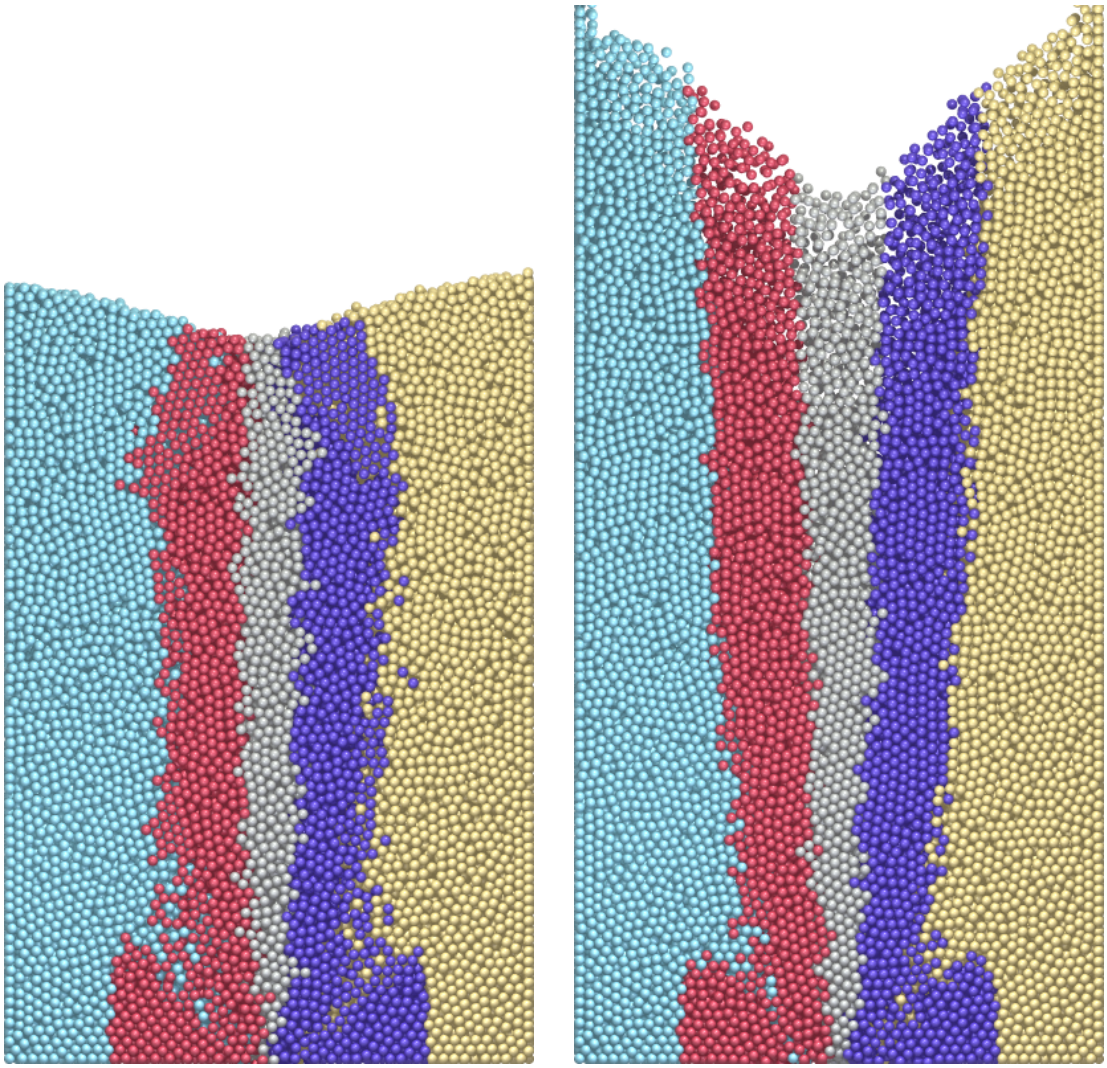


Figure 4-11: Snapshots of the DEM (left) and spot (right) simulation of chapter 3 taken at $t = 300\tau$ showing the evolution of the free surface. The colored particles were initially in equally-spaced columns of width $10d$, and serve as a guide to the eye. The right image shows that if spots follow a completely unbiased walk, then the free surface is not accurately modeled.

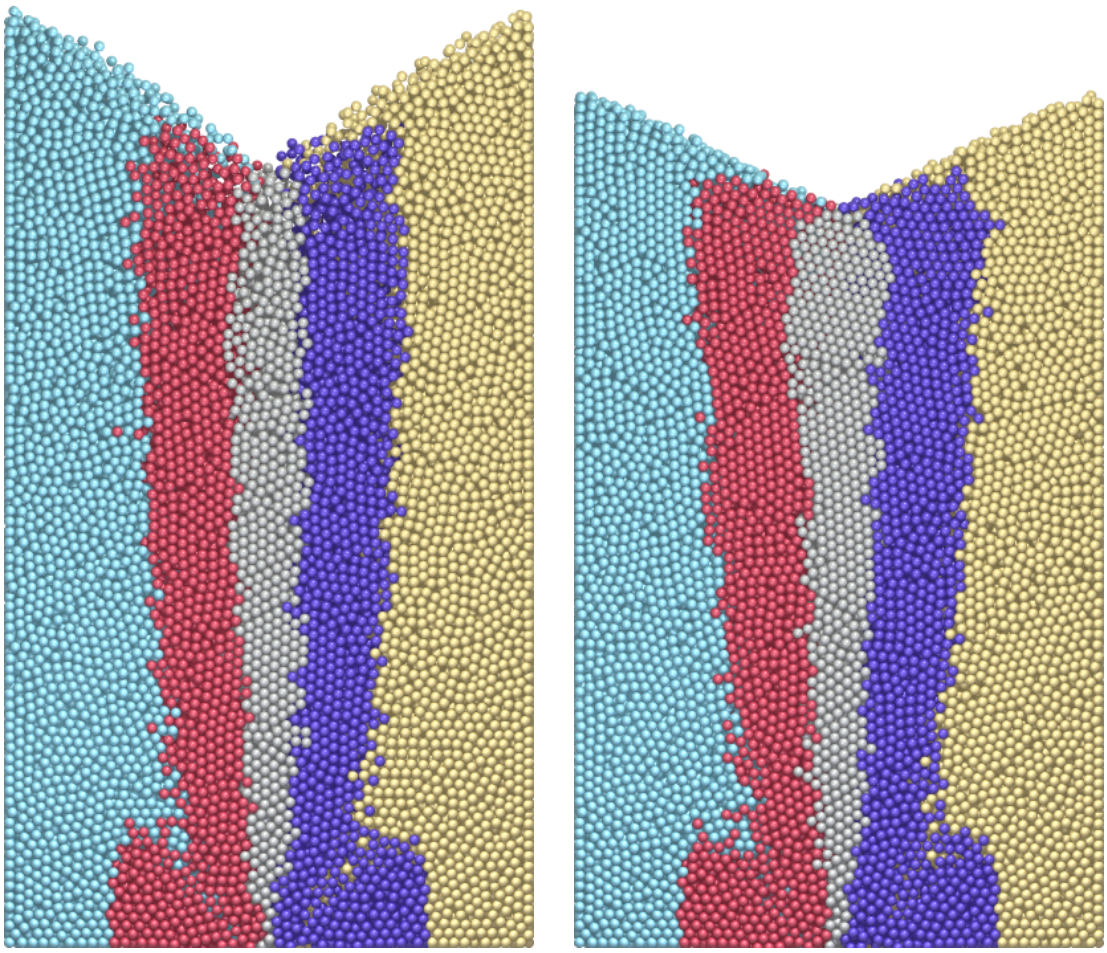


Figure 4-12: Two spot models with a modified random walk process that can qualitatively predict the features of the free surface seen in DEM, while leaving the bulk flow largely unaltered. In the left image, the random walk process was biased so that spots preferentially move towards regions with more particles. In the right image, the influence of the spot was also modified so that spots in range of fewer particles give them a larger displacement.

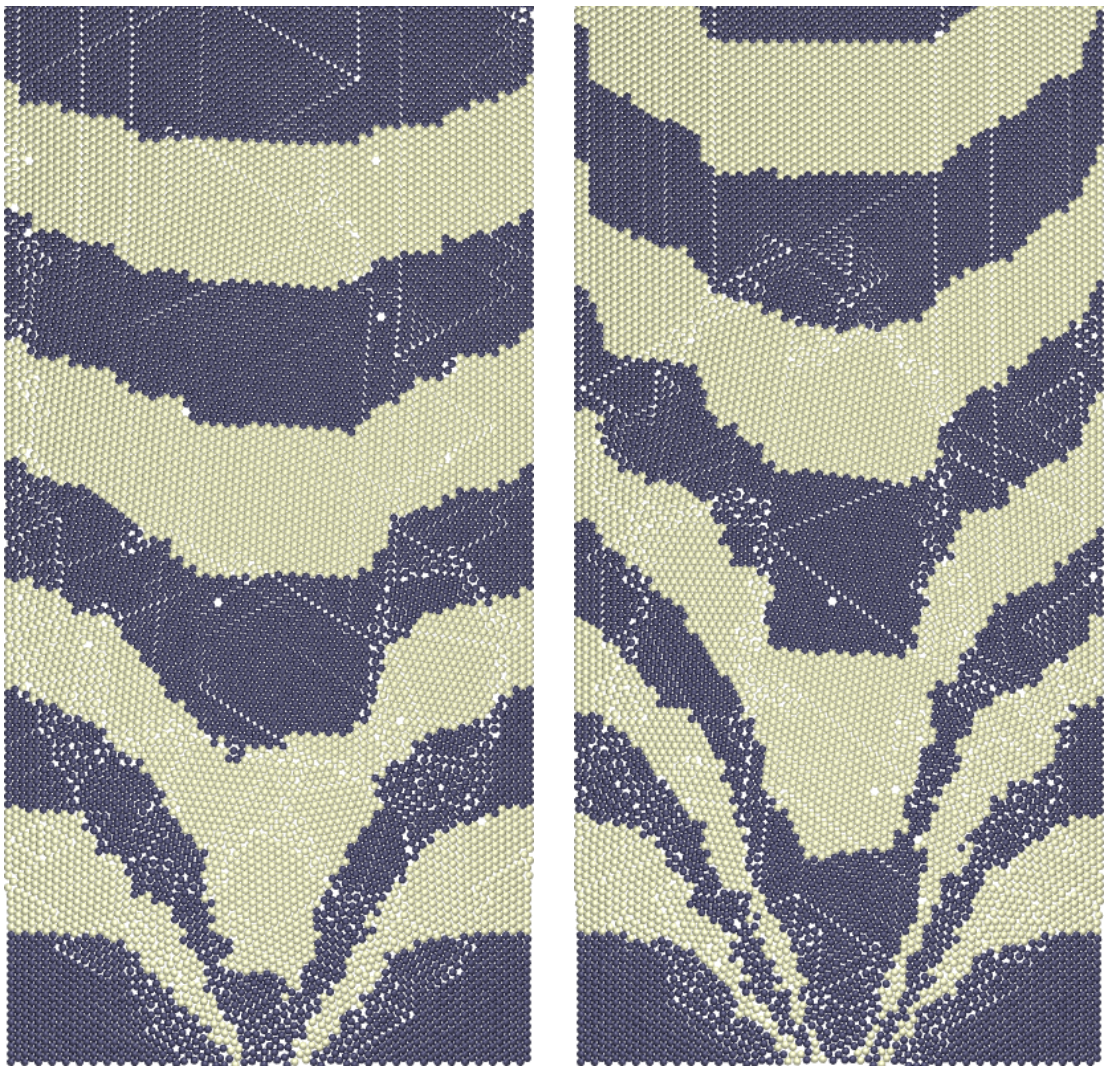


Figure 4-13: Snapshots of a spot simulation with relaxation on a two-dimensional regular hexagonal packing, taken at $t = 200\tau$ (left) and $t = 400\tau$ (right). Extended, linear dislocation defects are visible.

lized particles are visible, and the boundaries of these regions form linear dislocation defects. Many of these defects are extended over tens of particles, and it is interesting that the spot motion (which is only ever applied on an scale of $\sim 5d$) can produce these. Voids in the crystalline structures are also apparent and these play a large role on the rearrangement of the packing, by nucleating dislocations.

4.4 Parallelizing the spot model

4.4.1 Introduction

One of the major advantages of the spot model as a simulation technique is its speed. In the simulations above, the spot model typically executes many times faster on a single processor than a DEM simulation on twenty processors. However, since the spot model microscopic mechanism is local, and contains no long-range forces, it lends itself well to parallelization, allowing for still faster simulations, or the ability to handle much larger systems.

In this section, we consider two possible methods of parallelizing the spot model simulations considered in the previous section. The codes are written using the Message-Passing Interface (MPI), which is a popular library for development of parallel codes suitable for Beowulf clusters. It provides low-level routines for passing data between nodes, for node synchronization, and also for accurate parallel timing. All the codes discussed in this section were run on the AMCL, allowing for an in-depth comparison of running times.

The following subsection gives an overview of the serial spot model code, and highlights which parts need to be parallelized. In subsection 4.4.3 a parallel implementation using a master/slave architecture is discussed, while in subsection 4.4.4, an alternative code making use of a more distributed architecture is presented.

4.4.2 Overview of the serial code

The serial version of this code was written in object oriented C++, and is discussed in detail in appendix C. The bulk of the code is built into the `container` class, which has a constructor of the form

```
container::container(float minx, float maxx, float miny,
                     float maxy, float minz, float maxz,
                     int xn, int yn, int zn);
```

which initializes a simulation box. The first six parameters set the size of the box. The volume is divided up into smaller regions, each of which handles the particles within

that region, and the number of subdivisions in each direction is given by the remaining three parameters, `nx`, `ny`, and `nz`. The container holds and manages all particles in the simulation. Simple routines such as `void put(vec &p, int n)` place a single particle with numerical identifier `n` and position `p` into the simulation, assigning it to the correct region. Other routines such as `void dump(char *filename)` dump the particle positions out to a file.

However, the two most important routines for the simulation are the ones responsible for the spot motion and the elastic relaxation. The routine `void spot(vec &p, vec &v, float r)` displaces the particles within a radius `r` of position `p` by an amount `v`. The routine `void relax(vec &p, float r, float s, float force, float damp, int steps)` carries out an elastic relaxation on those particles within radius `s` of position `p`, with those between radius r and s held fixed to prevent long-range disruptions. The quantities `force` and `damp` set the forcing and damping used in the relaxation, and the relaxation process is carried out `steps` times. Further details are listed in appendix C.

The relaxation step is typically the most computationally intensive part of the calculation. For the simulations in the previous chapter, we generally make use of just a single relaxation step, where `force=0.8`, `damp=0`, and `steps=1`, and for these parameters, the entire container can be drained in approximately twelve hours. In those simulations, the single step relaxation scheme was found to be sufficient, but one could easily imagine situations where a more complicated relaxation process was carried out (perhaps taking into account friction). Situations involving a more complex relaxation steps would benefit the most from parallelization, and thus in the timing comparisons given below, it is also helpful to consider a five step relaxation scheme using `force=0.6`, `damp=0.2`, and `steps=5`. The serial version of this code takes approximately two days to drain the entire container.

4.4.3 Parallelization using a master/slave model

Overview

In the serial spot simulation code, the elastic relaxation step is the computational bottleneck, since it requires analyzing all pairs of neighboring particles within a small volume. In a parallel version of the code, we would ideally like to distribute this computational load across many processors, and since each relaxation event occurs is a local area, we can pass out different relaxation jobs to different processors.

As a first attempt at this, a code was written using a master/slave configuration. During the computation, the entire state of the system (particle positions and spot positions) is held on the master node. The master node then sequentially passes out jobs to the slave nodes for computation, before receiving them back. Since the spot motion events occur at random positions within the container, multiple relaxation events can be computed simultaneously.

In the typical spot algorithm, every spot displacement is followed by an elastic relaxation at that location. For this implementation, it therefore made sense to combine the `spot()` and `relax()` routines into a general routine `void spotrelax(vec &p, vec &v, float sr, float r, float s, float force, float damp, int steps)` which carries out a displacement of `v` with radius `sr` at location `p`, and follows it with an elastic relaxation, using the same parameters as those described for `relax()`. For the parallel version of the code, the `spotrelax()` routine first sends out the parameters of the relaxation job to an available slave node, and then sends across all the particles (typically on the order of several hundred) which can be potentially be influenced. The slave carries out the relaxation, and waits until it receives a new relaxation job, at which point it passes the completed job back to the master node.

A job to the slaves takes the form of four MPI messages. A short header message is first sent, containing the number of particles that are involved in the relaxation and the number of fixed particles in the outer shell. Three buffers are then sent containing the fixed particle positions, the moving particle positions, and the numerical identifiers of the moving particles. The completed jobs are passed to the master node using two

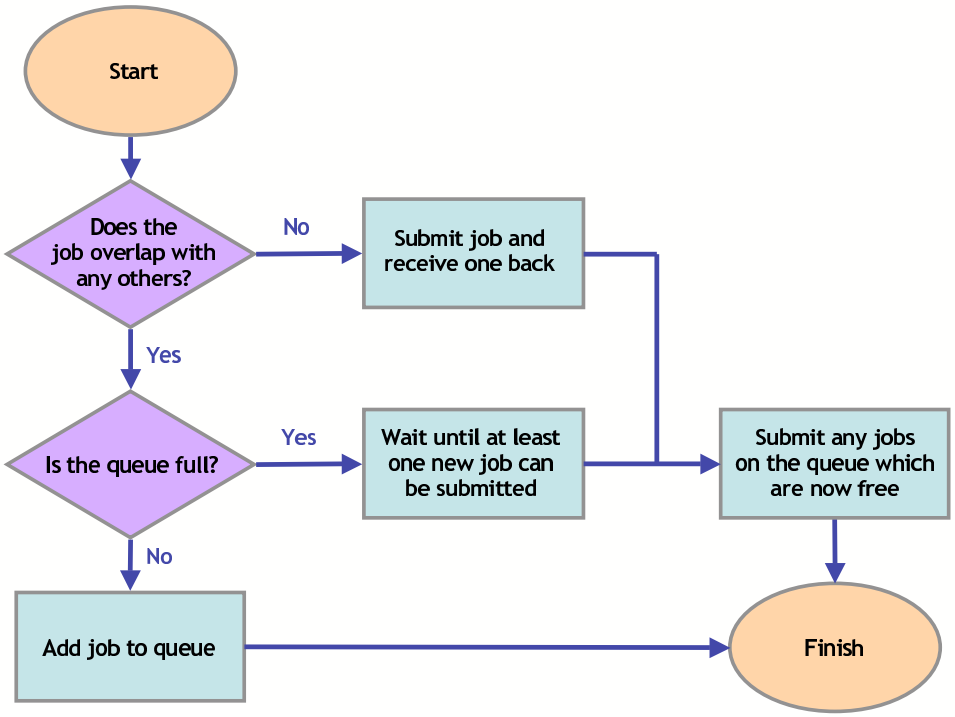


Figure 4-14: Flow chart showing the queueing system used in the master/slave parallelization method.

MPI messages, one containing the numerical identifiers of the displaced particles, and one containing the positions of the displaced particles; the master node already knows how big these messages will be by remembering from when it sent out the job. To save on time the `MPI_Irecv` command is used, so that the new job is transmitted to a slave concurrently with the old job being sent back. However, the message passing in this algorithm could definitely be improved further, and reducing the total number of messages could significantly improve efficiency.

During a typical flow of the system, spot displacements will occur at different positions and as such, their effects can be computed independently. However sometimes cases will arise in which a spot motion takes place in a position which overlaps with a job already being computed on a slave node. A simple resolution to this problem is to wait until the conflicting job is finished before submitting the new one. However, in many cases this may result in a large amount of wasted computer time.

A queueing system was therefore implemented, and this is shown in detail in figure 4-14. If a job overlaps with any others which are already being computed by the slave

nodes, then an attempt is made to queue it. If the queue is not full, then the job is added to the queue, and the routine exits. If the queue is full, then jobs are pulled back from the nodes until at least one job from the queue can be submitted, and the original job is then be added to the queue. Each time a job is submitted to the slave nodes, the queue is scanned to see if any new jobs are now free; when the routine exits, it is guaranteed that every job which is stored on the queue cannot at that time be submitted.

Another routine, `void queueflush()`, was written to flush all the jobs from the queue and from the slave nodes; this routine must be run before the particle positions are saved to file.

Timing results

To test the algorithm, simulations were carried out using different numbers of slave nodes, for both the one-step relaxation process, and the five-step relaxation process. The first sixty snapshots of the drainage simulation were calculated, and the times to generate each snapshot (using `MPI_Wtime`) were logged to a file. Graphs of the progress of the simulation for the two relaxation schemes are shown in figures 4-15 and 4-16.

From the graphs, we see that the rate of computation is initially very rapid, before reaching a steady state after around fifteen frames. Since spots are injected at the orifice during computation, it takes several frames for them to propagate through the container, before the total number in the container reaches a steady state and the number being introduced at the orifice roughly balances the number reaching the top of the packing. In order to assess the rate of computation, linear regression was applied during the steady state region from frame thirty to frame sixty. Figure 4-17 and table 4.2 show the rates of computation compared to those from the serial code. Unfortunately, for the single step relaxation code, the parallel jobs are all slower than the serial code, suggesting that the amount of time spent communicating the particle information to the slaves is larger than the amount of time for the actual computation. However, for the five step relaxation method, a significant speedup is seen over the

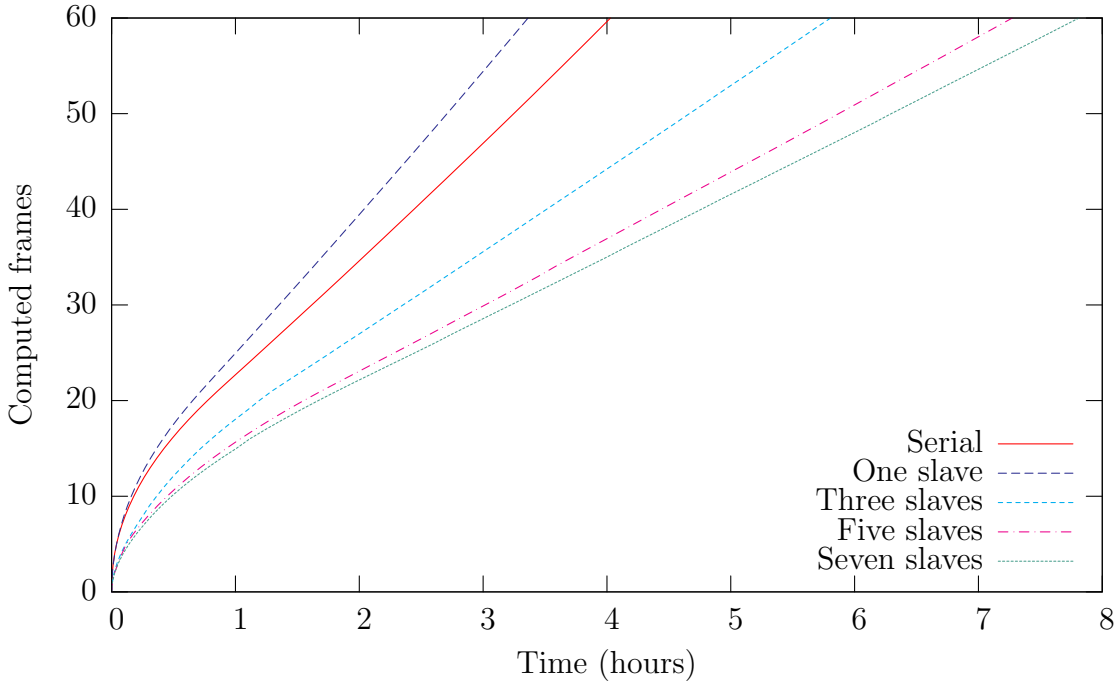


Figure 4-15: Plots showing the progression of the simulation, in terms of the number of frames computed as a function of time, for the single step relaxation method.

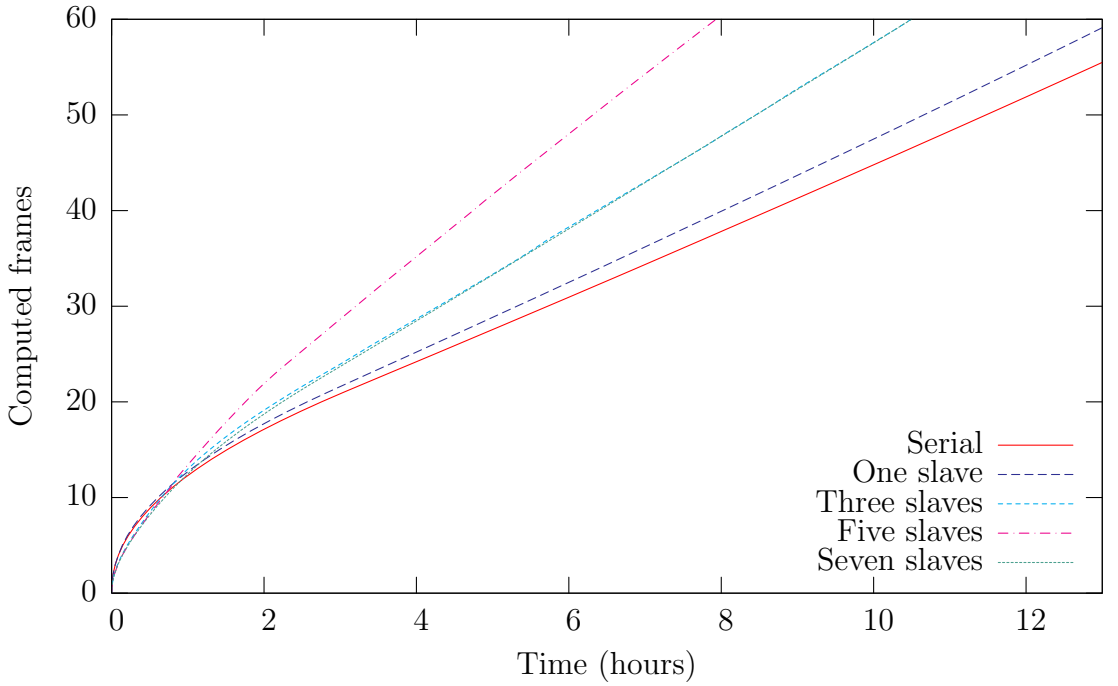


Figure 4-16: Plots showing the progression of the simulation, in terms of the number of frames computed as a function of time, for the five step relaxation method.

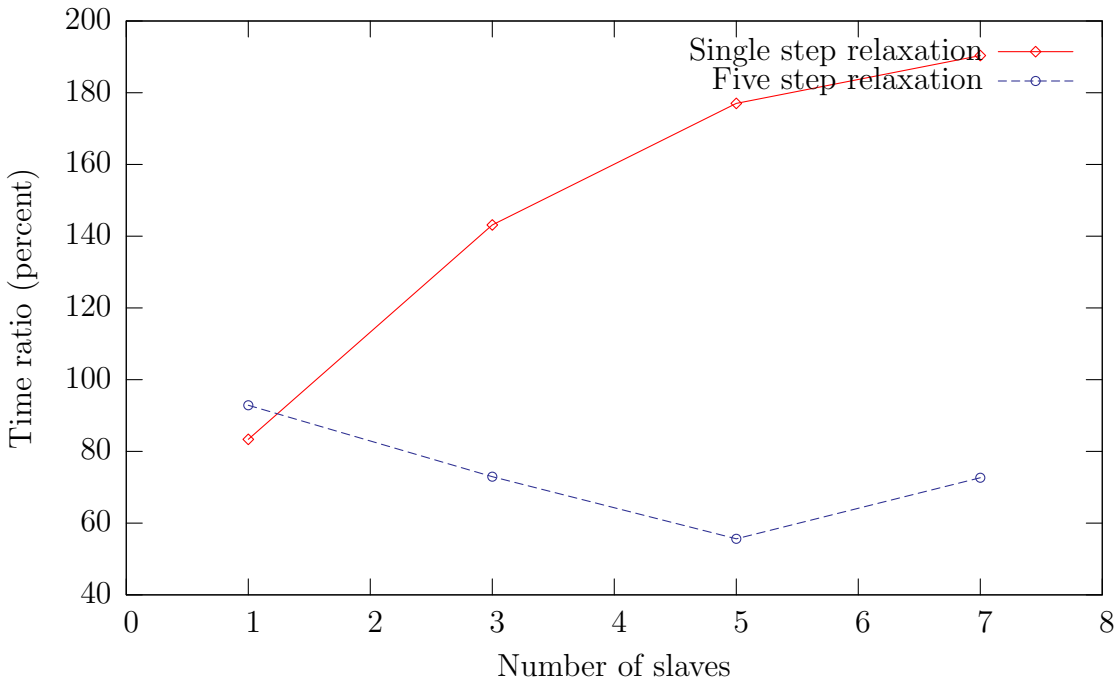


Figure 4-17: Computation times for the master/slave spot algorithm for different numbers of slave processors, shown as a ratio to those for the serial code.

serial code, since the amount of computation required for each relaxation event is much larger.

The results show the problem of creating a parallel algorithm using a master/slave architecture. In this algorithm, too much stress is placed on the master node, and very poor scalability with the number of nodes is achieved, as the slaves often stand idle waiting for the master node to pass them work. The distributed approach, discussed in the following subsection, therefore seems much more advantageous.

However, the above algorithm could potentially work very well on a shared memory machine. In that case, the master node would only need to handle the queue and transfer minimal job information to the slaves, since they would be able to access the particle positions directly. This could be implemented in MPI, but it could also be ideally suited to running in Cilk [20].

Slaves	One step time (s)	Ratio	Five step time (s)	Ratio
(Serial)	289	100%	1020	100%
1	241	83.3%	948	92.9%
3	414	143.1%	744	72.9%
5	512	177.0%	568	55.6%
7	551	190.4%	741	72.6%

Table 4.2: Times for the computation of a single frame using the master/slave algorithm, for the single step relaxation and the five step relaxation. The ratios show the amount of time taken for a run compared to the time of execution of the serial version.

4.4.4 A distributed parallel algorithm

Overview

The parallelization method described in the previous section had the significant drawback that large amounts of data needed to be transmitted to and from the master node. For the case of a single-step elastic relaxation, the master node cannot copy out the jobs to the slave nodes rapidly enough. A second parallelization scheme was therefore considered, in which the container is divided up between the slaves, with each slave holding the particles in that section of the container. A master node holds the position of the spots and computes their motion. When a spot moves, the master node tells the corresponding slave node to carry out a spot displacement of the particles within it. Only the position and displacement carried by the spot need to be transmitted to the slave, significantly reducing the amount of communication. Furthermore, many spot motions can be submitted to each slave simultaneously in a long worklist, minimizing the number of messages that need to be sent.

However, the drawback with this method is that sometimes a spot's region of influence may overlap with areas managed by other slaves. In that case, each slave must transmit particles to the slave carrying out the computation, and then receive back the displaced particles once the computation is carried out. Note however that since this communication happens between slaves, and not between the master and the slave, the workload is much more evenly distributed. The information about when slaves must pass their particles to neighboring slaves for computation can be passed

to them by the master node as a type of job in the worklist.

Details of the worklist

Since sending very small MPI messages is inefficient, the master node sends slaves a worklist of spot motions and other jobs that need to be carried out as an array of 1024 floating point numbers. The slaves sequentially process this information, and then wait for the master node to supply a new set of tasks. Each entry in the worklist begins with a type:

- **1** – Add a particle.
- **2** – Carry out overlapping spot motion.
- **3** – Carry out non-overlapping spot motion.
- **4** – Pass particles to a neighboring slave.
- **5** – Send all particle positions back to the master.
- **6** – Send the total number of particles to the master.
- **7** – Finish the simulation.
- **0** – End of worklist; wait for new tasks.

After the job type, subsequent numbers may follow to describe the specifics of the job. For job type 1 to add a particle, four numbers follow, giving the particle's position and a numerical label. Job types 5, 6, and 7 have no subsequent numbers. For the dump and count operations, the slaves scan all the particles they have, and then send the data to the master node for analysis.

For job type 3, to initiate a spot motion, twelve numbers follow, giving the position and displacement of the spot, plus the details of the relaxation. For job type 2, an additional six numbers are sent, describing a grid of processors which need to be contacted in order to see if they have any particles contributing to the spot motion. Each of these processors is sent a job type 4, telling it to package up all particles which

could potentially be involved in a spot motion, and send them off to a neighboring processor. The slave must then wait until the processor carrying out the spot motion sends back the positions of all particles which have been displaced.

Passing particles between slaves

In this algorithm, we expect that communication between nodes will be significant bottleneck. To minimize the amount of communication, the code looks at all possible ways to create a grid of slaves using the specified number of processors, and chooses the one which minimizes the shared surface area between nodes. For the test cases considered here, this always results in a grid of the form $1 \times 1 \times n$ for n slave nodes. However the cases of $2 \times 1 \times 2$, $2 \times 1 \times 4$, and $3 \times 1 \times 3$ nodes were also investigated.

Two different methods for passing particles between slaves were considered. For the first method, two MPI messages were used: each slave node first sends the number of particles it is passing to the node carrying out the computation, and it then sends a buffer containing the particle positions and their numerical identifiers. The slave deletes the positions of the particles which can be affected by the relaxation, but keeps copies of the particles which remain fixed in the outer relaxation shell.

Since the receiving node knows exactly how many particles to expect from each processor from the first message, it can efficiently store the incoming particle positions sequentially in a single block of memory, before adding the particles to the ones from its own region for the relaxation calculation. Once the relaxation is completed, the two-message format is used to communicate the particles back to the slave nodes; only the moving particles need to be transferred, since the receiving node kept a copy of all the fixed particles. Figure 4-18 shows a frequency plot of the number of particles passed and returned by this procedure.

Since sending short MPI messages is inefficient, a second communication method was also investigated, whereby particle positions were sent in a single fixed-length buffer. Particle positions are read from the buffer sequentially, and the buffer end is marked with a -1 entry. In the case when a processor receives a buffer completely full of particle positions, it waits for a subsequent one to be sent. A buffer length of 1024

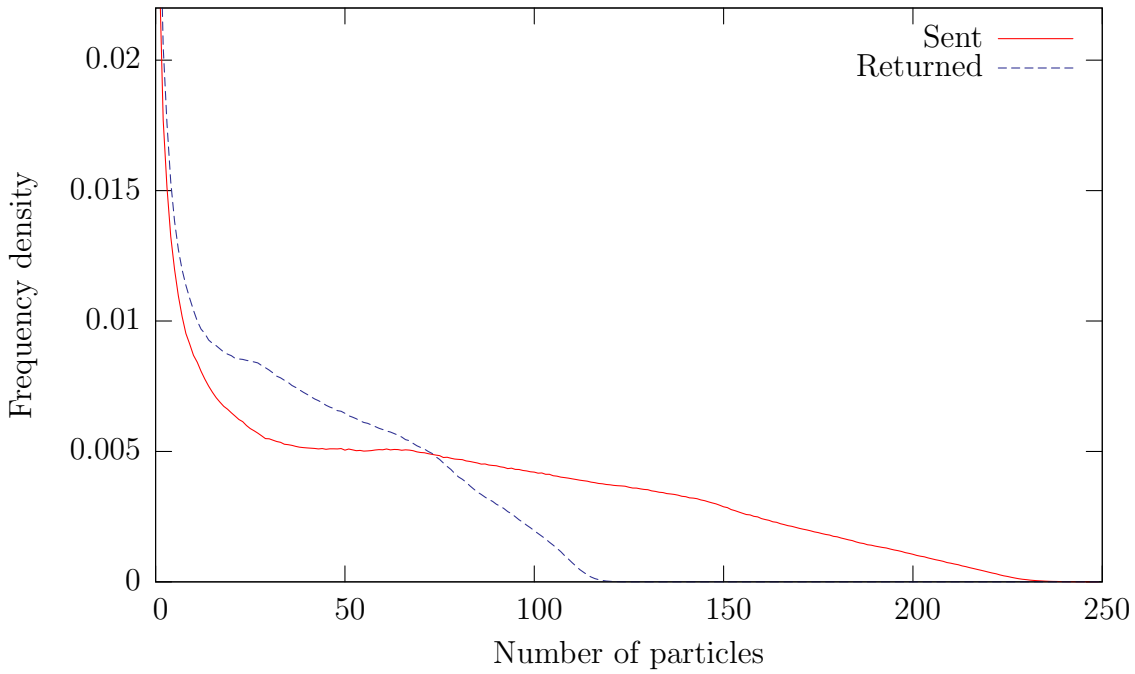


Figure 4-18: Frequency plots of the number of particles transmitted between slaves nodes in the distributed parallel algorithm. The slave nodes send all particles involved in the relaxation, but receive back only the displaced ones, and not the ones which remain fixed during the calculation.

(which can hold 256 particles) was used, which with reference to figure 4-18 should be ample for most inter-processor communications. In addition, a buffer length of 512 was investigated, which should allow most receiving operations to be done in one message, and most send operations to be done in two.

4.4.5 Timing results

Figure 4-19 and table 4.3 show how the computation speed scales as a function of the number of slave nodes. We see much better results than for the master/slave algorithm, and even with a single step relaxation we see a marked speedup over the serial code, with the simulation running almost twice as rapidly when five or seven slave nodes are used.

For a larger number of processors with a single step relaxation, we begin to see a small slowdown. This may be an indication that inter-processor communication

eventually becomes the most significant factor in computation. While the actual time for the communication may play a role, perhaps a more serious problem may be due to job distribution. Suppose we are running on two slave nodes, and that the master node finds that three spots move on the first slave, then one spot moves on the overlap between slaves, and then three spots move on the second slave. If this occurs, then the second slave will have to wait until the overlapped job is completed before it can execute its three jobs, meaning that the total time for the computation will be similar to executing in serial.

Normally we expect that jobs will be more evenly distributed between nodes, so that computation can be executed in parallel. However, it may be that for larger number of processors, bottlenecking of jobs becomes more frequent. We will be guaranteed that one slave will always be running, but other slaves can potentially have multiple dependencies on others, particularly as the number of slaves and the number of overlapping jobs increases.

In certain situations, a reordering of jobs could significantly improve the flow of the algorithm. In the example above, if the three jobs on the second node were independent of the overlapping job, they could be executed while the first node was processing its own local jobs. This could perhaps be implemented using a queueing system similar to that used in the master/slave algorithm.

Figure 4-20 shows the results of the simulations using a fixed message length, compared to the serial code and those for the variable message length, We considered the case of a $1 \times 1 \times 7$ node configuration using the single step relaxation process. From the figure, we see that the 1024 length buffer results in a slower computation, with a time ratio of 126.7% compared to the variable length case. However, the simulation using a 512 length buffer is faster, with a time ratio of 88.6% compared to the variable length case, and time ratio of 55.7% compared to the serial case.

4.4.6 Conclusion

For both of the algorithms considered, significant speedups over the serial version of the algorithm were possible for certain situations. However, the time of the sim-

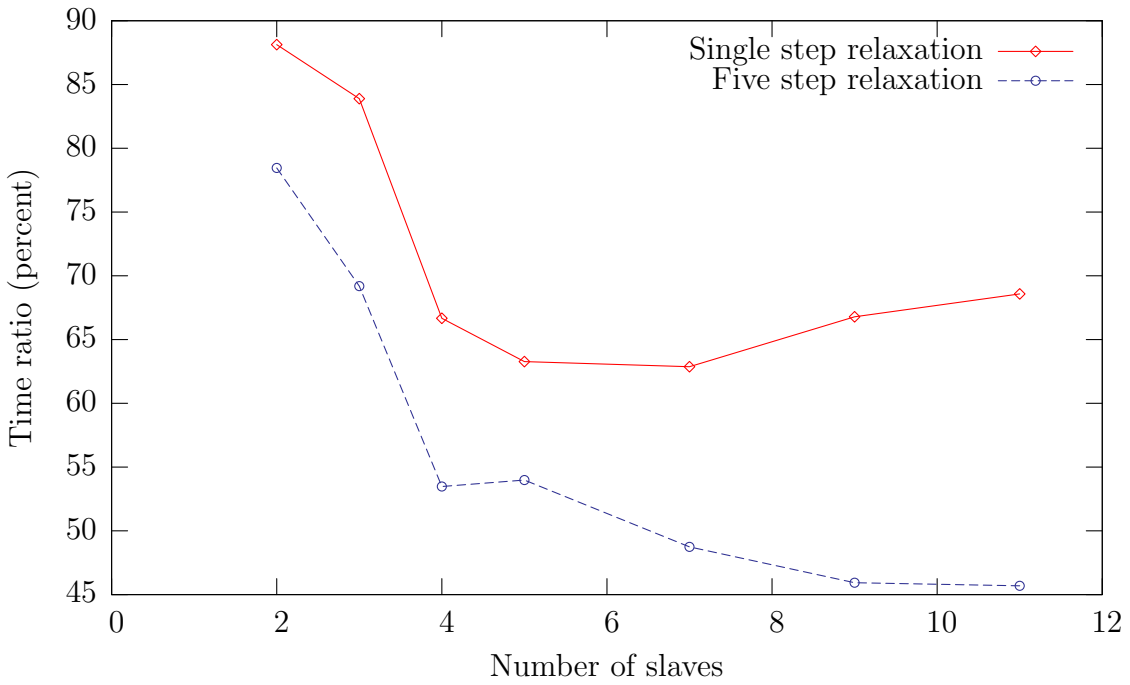


Figure 4-19: Computation times for the distributed parallel spot algorithm for different numbers of slave processors (using the $1 \times 1 \times n$ grid configuration) shown as a ratio to those for the serial code.

Slaves	Processor grid	One step time (s)	Ratio	Five step time (s)	Ratio
(Serial)		289	100%	1020	100%
2	$1 \times 1 \times 2$	254	88.1%	801	78.4%
3	$1 \times 1 \times 3$	242	83.9%	706	69.2%
4	$1 \times 1 \times 4$	193	66.7%	546	53.5%
5	$1 \times 1 \times 5$	156	54.0%	551	54.0%
7	$1 \times 1 \times 7$	182	62.9%	497	48.7%
9	$1 \times 1 \times 9$	193	66.8%	467	45.9%
11	$1 \times 1 \times 11$	198	68.6%	466	45.7%
4	$2 \times 1 \times 2$	326	112.6%	828	56.6%
8	$2 \times 1 \times 4$	272	94.0%	664	65.1%
9	$3 \times 1 \times 3$	320	110.8%	769	42.3%

Table 4.3: Times for the computation of a single frame for different configurations of slave processes, for the single step relaxation, and for the five step relaxation. The ratios show the amount of time taken for a run compared to the time of execution of the serial version.

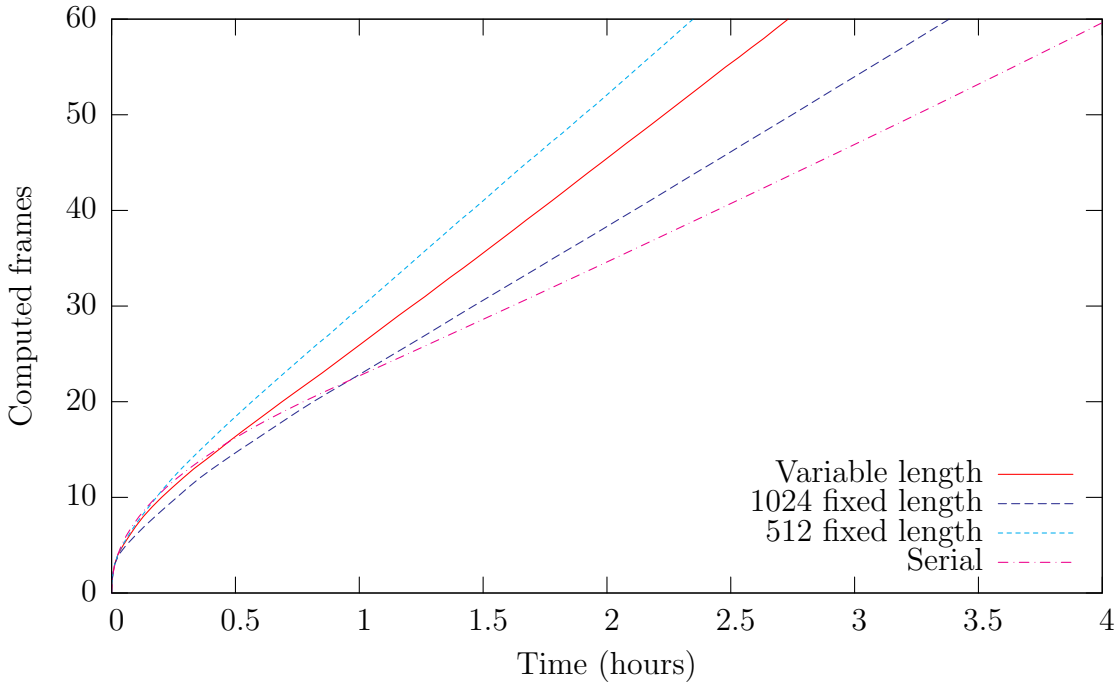


Figure 4-20: Plots showing the progression of computation for the simulations using a fixed message length, for a $1 \times 1 \times 7$ node configuration with a single step relaxation process. Plots for the serial code and the corresponding variable message length simulation are shown for comparison.

ulations is definitely far from the optimal behavior for parallel algorithms, where we would like the time taken to scale inversely with the number of processors. In both of the algorithms, inter-processor communication and job distribution caused significant losses in the effectiveness of the algorithm. Queueing and reordering jobs, and improving the efficiency of message passing were both considered, and have both resulted in speedup improvements.

However, it is hoped that with more work, significant advancements could still be made in both of these algorithms. For the master/slave method, where the efficiency of the master node is paramount, improvements in the handling of the queue could speed up the computation. For the distributed method, implementing a job reordering scheme would be beneficial. In both schemes, improving the format of the messages between nodes could be investigated further. The tests with fixed length messages of 512 numbers showed improvements over the variable length messages, and a more thorough tuning of the length of this buffer could be beneficial.

Making more advanced use of MPI could also help. In places, efforts were made to make use of synchronous sending of messages, using the commands `MPI_Irecv` and `MPI_Wait` to exchange data between two processors concurrently. However, more advanced MPI functionality was not considered, and in places send operations were used which waited for confirmation of their send when they did not need to.

Other types of algorithms could also result in better performance. A fully distributed algorithm, in which particles and spots are all shared between slaves, may exhibit better load-balancing. This possibility was not considered here, since it was unclear how to run an event-driven simulation of this type across many nodes. However, if the random waiting-time distribution for spot motion was replaced with moving at every fixed time interval, the slaves could more easily manage the spots in their region without reference to other nodes.

Another possible algorithm would involve calculating overlapping spot motions using all of the slaves involved, rather than first transferring all the particles to one of the slaves for computation. Each slave could handle the relaxation of the particles in its region, and then send messages to the neighboring slaves to handle interactions

between particles on different slaves. While this could result in more messages being sent, far fewer particles overall would actually need to be transferred, since only the skin of particles exactly at interface between regions would need to be communicated.

Chapter 5

Pebble-Bed Simulation[1]

5.1 Introduction

5.1.1 Background

A worldwide effort is underway to develop more economical, efficient, proliferation resistant, and safer nuclear power [4]. A promising Generation IV reactor design is the uranium-based, graphite moderated, helium-cooled very high temperature reactor [49], which offers meltdown-proof passive safety, convenient long-term waste storage, modular construction, and a means of nuclear-assisted hydrogen production and desalination. In one embodiment, uranium dioxide is contained in microspheres dispersed in spherical graphite pebbles, the size of billiard balls, which are very slowly cycled through the core in a dense granular flow [97, 130]. Control rods are inserted in graphite bricks of the core vessel, so there are no obstacles to pebble flow.

The pebble-bed reactor (PBR) concept, which originated in Germany in the 1950s, is being revisited by several countries, notably China [110] (HTR-10 [58]) and South Africa [97] (PBMR [5]), which plan large-scale deployment. In the United States, the Modular Pebble Bed Reactor (MPBR) [130, 2] is a candidate for the Next Generation Nuclear Plant of the Department of Energy. A notable feature of MPBR (also

[1]This chapter is based on reference [113], *Analysis of Granular Flow in a Pebble-Bed Nuclear Reactor*, published in Physical Review E in 2006. See `http://pre.aps.org/` for more details.

present in the original South African design) is the introduction of graphite moderator pebbles, identical to the fuel pebbles but without the uranium microspheres. The moderator pebbles form a dynamic central column, which serves to flatten the neutron flux across the annular fuel region without placing any fixed structures inside the core vessel. The annular fuel region increases the power output and efficiency, while preserving passive safety. In the bidisperse MPBR, the moderator pebbles are smaller to reduce the permeability of the central column and thus focus helium gas on the outer fuel annulus. The continuous refueling process is a major advantage of pebble-bed reactors over other core designs, which typically require shutting down for a costly dismantling and reconstruction. The random cycling of pebbles through a flowing core also greatly improves the uniformity of fuel burnup.

In spite of these advantages, however, the dynamic core of a PBR is also a cause for concern among designers and regulators, since the basic physics of dense granular flow is not fully understood. Indeed, no reliable continuum model is available to predict the mean velocity in silos of different shapes [27], although the empirical Kinematic Model [90, 95, 94] provides a reasonable fit near the orifice in a wide silo [137, 84, 114, 28]. A complete statistical theory of dense granular flow is still lacking. The classical kinetic theory of gases has been successfully applied to dilute granular flows [115, 61, 108], in spite of problems with inelastic collisions [64], but it clearly breaks down in dense flows with long-lasting, frictional contacts [93, 28], as in pebble-bed reactors. Plasticity theories from soil mechanics might seem more appropriate [94], but they cannot describe flows in silos of arbitrary shape and often lead to violent instabilities [117, 106].

For now, experiments provide important, although limited, information about dense granular flows. Many experiments have been done on drainage flows in quasi-2D silos where particles are tracked accurately at a transparent wall [84, 83, 114, 28, 27]. Experimental studies of more realistic geometries for PBR have mostly focused on the porosity distribution of static packings of spheres [50, 119], which affects helium gas flow through the core [30, 141, 143].

As a first attempt to observe pebble dynamics experimentally in a reactor model,

the slow flow of plastic beads has recently been studied in 1:10 scale models of MPBR in two different ways [63]: the trajectories of colored pebbles were recorded (by hand) along a Plexiglas wall in a half-core model, and a single radioactive tracer pebble in the bulk was tracked in three dimensions in a full-core model. Very slow flow was achieved using a screw mechanism at the orifice to approximate the mean exit rate of one pebble per minute in MPBR. These experiments demonstrate the feasibility of the dynamic central column and confirm that pebbles diffuse less than one diameter away from streamlines of the mean flow. However, it is important to gain a more detailed understanding of pebble flow in the entire core to reliably predict reactor power output, fuel efficiency, power peaking, accident scenarios using existing nuclear engineering codes [131, 51].

5.1.2 Discrete-Element Simulations

Simulations are ideally suited to provide complete, three-dimensional information in a granular flow. Some simulations of the static random packing of fuel pebbles in a PBR core have been reported [38, 103], but in the last few years, large-scale, parallel computing technology has advanced to the stage where it is now possible to carry out simulations of continuous pebble flow in a full-sized reactor geometry using the Discrete Element Method (DEM). In this chapter, we present DEM simulations which address various outstanding issues in reactor design, such as the sharpness of the interface between fuel and moderator pebbles (in both monodisperse and bidisperse cores), the horizontal diffusion of the pebbles, the geometry dependence of the mean streamlines, the porosity distribution, wall effects, and the distribution of "residence times" for pebbles starting at a given height before exiting the core.

Our simulations are based on the MPBR geometry [130, 2], consisting of spherical pebbles with diameter $d = 6$cm in a cylindrical container approximately 10m high and 3.5m across. In this design there is a central column of moderating reflector pebbles, surrounded by an annulus of fuel pebbles. The two pebble types are physically identical except that the fuel pebbles contain sand-sized uranium fuel particles. Particles are continuously cycled, so that those exiting the container are reintroduced at the

top of the packing. In order to efficiently maintain the central column, a cylindrical guide ring of radius $r_{\text{in}} = 14.5d$ extends into the packing to $z = 140d$. Reflector pebbles are poured inside, while fuel pebbles are poured outside, and the guide ring ensures that two types do not mix together at the surface. Figure 5-1 shows the two main geometries that were considered; for much of this analysis, we have concentrated on the case when the exit funnel is sloped at thirty degrees, but since this angle can have a large effect on the pebble flow, we also consider the case of the when the funnel is sloped at sixty degrees. In both cases the radius of the opening at the bottom of the funnel is $r_{\text{exit}} = 5d$.

In MPBR, as in most pebble-bed reactors, the drainage process takes place extremely slowly. Pebbles are individually removed from the base of the reactor using a screw mechanism, at a typical rate of one pebble per minute, and the mean residence time of a pebble is 77 days. Carrying out a DEM simulation at this flow rate would make it infeasible to collect enough meaningful data. However, the results of the previous chapters, and the experimental work of Choi [28], show that for granular drainage, altering the overall flow rate does not alter the geometry of the flow profile – the flow velocities are scaled by a constant factor. Furthermore, geometric properties of the flow, such as particle diffusion, are unaffected by the overall flow rate. We therefore chose to study a faster flow regime in which pebbles drain from the reactor exit pipe under gravity. Our results can be related directly to the reactor design by rescaling the time by an appropriate factor.

As well as the two full-scale simulations described above, we also considered a half-size geometry in order to investigate how various alterations in the makeup of the reactor would affect the flow. In particular, we examined a series of bidisperse simulations, in which the diameter of moderator particles in the central column was reduced. As explained in section 5.8, this has the effect of reducing the gas permeability of the central column, thus focusing the helium coolant flow on the hottest region of the reactor core, in and around the fuel annulus. The purpose of the simulations is to test the feasibility of the bidisperse PBR concept, as a function of the size ratio of moderator and fuel pebbles, with regard to the granular flow. It is not

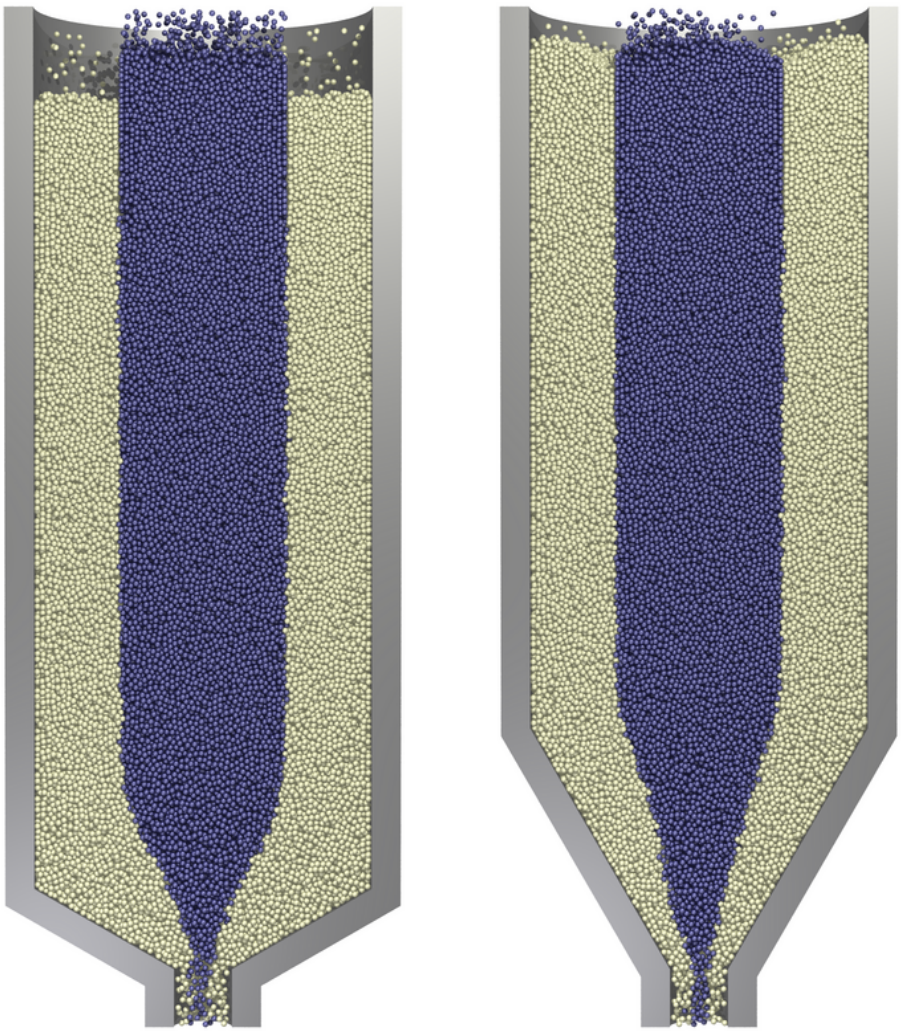


Figure 5-1: Snapshots of vertical cross-sections of the simulations for the two geometries considered in this report. We make use of a cylindrical coordinate system (r, θ, z) where $z = 0$ at the orifice. At the base of the container there is a small exit pipe of radius $r_{\text{exit}} = 5d$ that extends upwards to $z = 10d$. This connects to a conical funnel region, which has slope thirty degrees (left) or sixty degrees (right). The conical wall connects to a cylindrical wall of radius $r_{\text{out}} = 29d$, at $z = 23.86d$ and $z = 51.57d$ for the thirty and sixty degree reactor geometries respectively. Particles are poured into the container up to a height of approximately $z = 160d$. A cylindrical wall at $r_{\text{in}} = 14.5d$ extends down into the packing to a height of $z = 140d$ to keep the two types of pebbles mixing at the surface.

clear *a priori* under what conditions the dynamic column will remain stable with little interdiffusion of moderator and graphite pebbles.

To study this issue, we made a sequence of three runs using a half-size reactor geometry. (The smaller core size is needed since the number of smaller pebbles increases as the inverse cube of the diameter ratio.) The geometry is similar to that used above, except that the radius of the cylindrical container is decreased to $15d$, with the guide ring at $r_{\text{in}} = 7.5d$. The radius of the exit pipe is decreased to $r_{\text{exit}} = 4d$. In the experiments, we keep the diameter of the fuel pebbles fixed at d, and use d, $0.8d$, and $0.5d$ for the diameters of the moderator pebbles. The same geometry was also used to study the effect of wall friction, by making an additional run with the particle/wall friction coefficient $\mu_w = 0$.

This chapter is organized as follows. In section 5.2, we discuss the simulation technique that was used and briefly describe its implementation. This is followed with some basic analysis of the velocity profiles and a comparison to the Kinematic Model in section 5.3. We study diffusion around streamlines in section 5.4 and the distribution of porosity and local ordering in section 5.5. Next, in section 5.6 we examine the residence-time distribution of pebbles in the reactor, which is related to fuel burnup, and in section 5.7 we show that wall friction plays an important role. In section 5.8 we analyze the bidisperse PBR concept with half-size reactor simulations for a range of pebble-diameter ratios, focusing on the mean flow, diffusion, and mixing. We conclude in section 5.9 by summarizing implications of our study for reactor design and the basic physics of granular flow.

5.2 Models and Methods

Unlike all other simulations in this thesis, the DEM simulations presented here were carried out at Sandia Labs, since the MIT AMCL was too small to handle the number of particles used in the fully three dimensional system. For the monodispersed simulation, the spheres have diameter $d = 6$ cm, mass $m = 210$ g and interparticle friction coefficient $\mu = 0.7$, flowing under the influence of gravity $g = 9.81$ ms^{-2}. For

the bidispersed systems, the moderator particles have diameter $0.8d$ or $0.5d$. The particle-wall friction coefficient $\mu_w = 0.7$ except in one case where we model a frictionless wall, $\mu_w = 0.0$.

The initial configurations are made by extending the inner cylinder from $140d$ to the bottom of the container, adding a wall at the bottom of the container to stop particles from draining, and pouring in moderator pebbles into the inner cylinder and fuel pebbles between the inner and outer cylinders until the reactor was loaded. The bottom wall is then removed, the inner cylinder is raised to $140d$, and particles are allowed to drain out of the container. As noted above, particles are recycled with moderator particles reinserted within the inner cylinder, and fuel particles between the inner and outer cylinders. All results presented here are after all the particles have cycled through the reactor at least once. The number of moderator and fuel particles was adjusted slightly from the initial filling so that the level at the top of the reactor is approximately equal. For the full scale simulation with a thirty degree outlet, the total number of pebbles is 440,000 with 105,011 moderator pebbles and 334,989 fuel pebbles, while for the sixty degree outlet, the total number of pebbles is 406,405 with 97,463 moderator and 308,942 fuel pebbles. For the former case, a million steps took approximately 13 hours on 60 processors on Sandia's Intel Xenon cluster.

For the bidispersed simulations the total number of pebbles is 130,044, 160,423, and 337,715 for the diameter of the moderator particles equal to d, $0.8d$ and $0.5d$ respectively. As the diameter of the moderator pebbles is decreased the number of particles required rapidly increases, since it scales according to the inverse of the diameter cubed.

A snapshot of all the particle positions is recorded every $5\tau = 0.39$ s. For the thirty degree reactor geometry we collected 1,087 successive snapshots, totaling 24.9Gb of data, while for the sixty degree reactor geometry, we collected 881 successive snapshots, totaling 18.7Gb of data. A variety of analysis codes written in Perl and C++ were used to sequentially parse the snapshot files to investigate different aspects of the flow. We also created extended data sets, with an additional 440 snapshots for

the thirty degree geometry, and 368 snapshots for the sixty degree geometry, for examining long residence times in section 5.6.

5.3 Mean-Velocity Profiles

5.3.1 Simulation Results

Since we have a massive amount of precise data about the positions of the pebbles, it is possible to reconstruct the mean flow in the reactor with great accuracy. However care must be taken when calculating velocity profiles to ensure the highest accuracy. Initial studies of the data showed that crystallization effects near the wall can create features in the velocity profile at a sub-particle level, and we therefore chose a method that could resolve this.

By exploiting the axial symmetry of the system, one only needs to find the velocity profile as a function of r and z. The container is divided into bins and the mean velocity is determined within each. A particle which is at $\mathbf{x}_n$ at the nth timestep and at $\mathbf{x}_{n+1}$ at the $(n+1)$th timestep, makes a velocity contribution of $(\mathbf{x}_{n+1} - \mathbf{x}_n)/\Delta t$ in the bin which contains its midpoint, $(\mathbf{x}_{n+1} + \mathbf{x}_n)/2$.

In the z direction, we divide the container into strips $1d$ across. However, in the r direction we take an alternative approach. Since the number of pebbles between a radius of r and $r + \Delta r$ is proportional to $r\Delta r$, dividing the container into bins of a fixed width is unsatisfactory, as the amount of data in bins with high r would be disproportionately large. We therefore introduce a new coordinate $s = r^2$. The coordinate s covers the range $0 < s < r_{\text{out}}^2$, and we divide the container into regions that are equally spaced in s, of width $1d^2$. The number of pebbles in each bin is therefore roughly equal, allowing for accurate averaging in the bulk and high resolution at the boundary.

This result yields extremely accurate velocity profiles in the cylindrical region of the tank. However, it fails to capture crystallization effects in the conical region: since the particles are aligned with the slope of the walls are averaged over a strip in

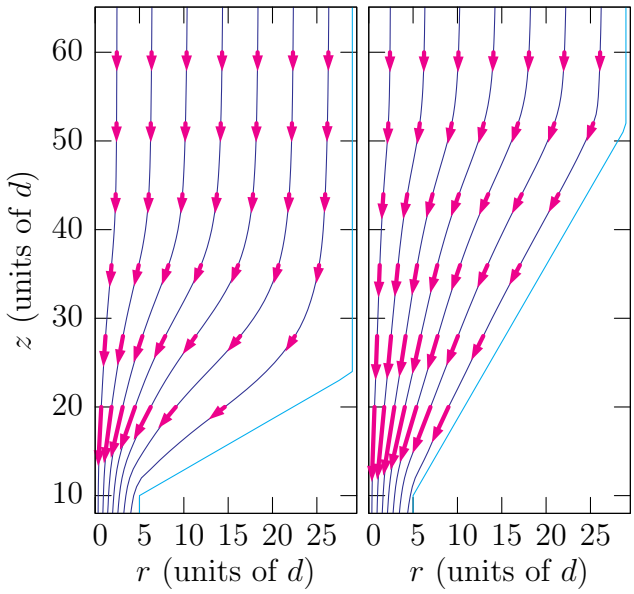


Figure 5-2: Computed streamlines of the mean flow in the 30° (left) and 60° reactor geometries. Arrows are proportional to the velocity vectors in selected horizontal slices.

z of width $1d$, any effects are smeared out across several bins. We therefore scaled the radial coordinate to what it would be if the particle was in the center of the strip. Specifically, if the radius of the container is given by $R(z)$, a particle at (r_n, z_n) is recorded as having radial coordinate $r_n R(z)/R(z_n)$. In the cylindrical region of the tank this has no effect, while in the conical region, it effectively creates trapezoid-shaped bins from which it is easy to see crystallization effects which are aligned with the wall.

The streamlines of the mean flow are shown in Fig. 5-2 in the two geometries. Streamlines are computed by Lagrangian integration of the DEM velocity field, starting from points at a given height, equally spaced in radius. In each geometry, there is a transition from a nonuniform converging flow in the lower funnel region to a nearly uniform plug flow in the upper cylindrical region, consistent with the standard engineering picture of silo drainage [94]. In the wider funnel, there is a region of much slower flow near the sharp corner at the upper edge of the funnel. Our results for both geometries are quite consistent with particle-tracking data for quasi-2D silos of similar shapes [27] and half-cylinder models of the MPBR core [63], which provides

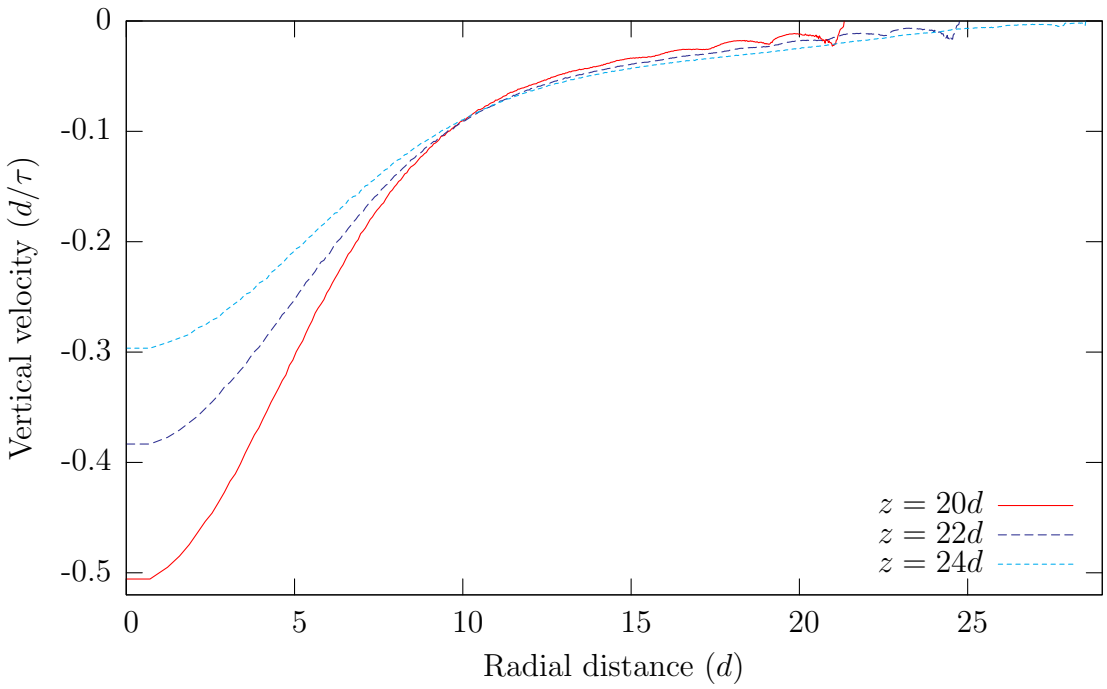


Figure 5-3: Velocity profiles for the thirty degree reactor geometry for several low cross-sections.

an important validation of our simulations.

We now look more closely at horizontal slices of the velocity field. Figure 5-3 shows several velocity profiles for the thirty degree case in the narrowing section of the container. As expected, we see a widening of the velocity profile as z increases. We can also see lattice effects, spaced at $\sqrt{3}d$ apart, due to to particles crystallizing on the conical wall section.

Figure 5-4 shows similar plots for several heights in the upper region of the container. At these heights, the velocity profile is roughly uniform across the container. However a boundary layer of slower velocities, several particle diameters wide, still persists. The average velocities of particles touching the boundary is between one half and two thirds that of particles in the bulk; it is expected that this behavior is very dependent on particle-wall friction; this issue is studied in more detail in section 5.7.

High in the container, results for the sixty degree geometry are very similar to the thirty degree case (and thus are not shown). However, as would be expected, a

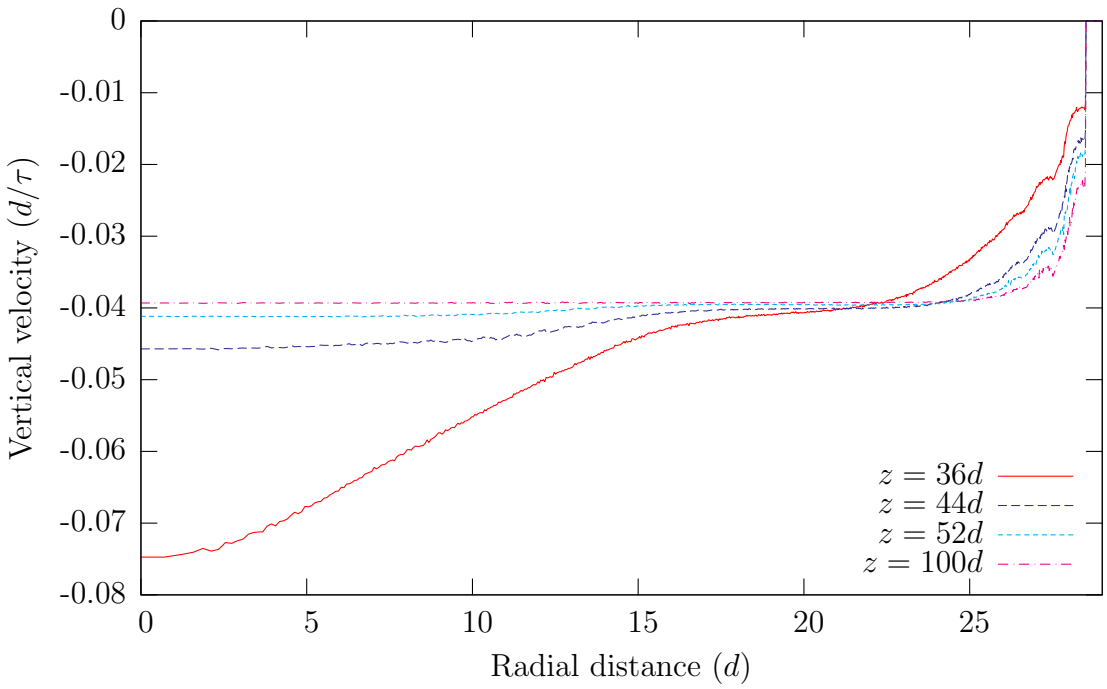


Figure 5-4: Velocity profiles for the thirty degree reactor geometry for several high cross-sections.

significantly different crossover from parabolic flow to plug-like flow in the lower part of the tank is observed, as shown in figure 5-6.

5.3.2 Comparison with the Kinematic Model

In section 2.3 we introduced the Kinematic Model, which is perhaps the only continuum theory available for the mean flow profile in a slowly draining silo. The vertical velocity v satisfies

$$\frac{\partial v}{\partial z} = b\nabla_{\perp}^{2} v, \tag{5.1}$$

where the vertical coordinate z acts like "time". Boundary conditions on Eq. (5.1) require no normal velocity component at the container walls, except at the orifice, where v is specified (effectively an "initial condition"). As described in Appendix B, this boundary-value problem can be accurately solved using a standard Crank-Nicholson scheme for the diffusion equation.

Consistent with a recent experimental study of quasi-2D silos [27], we find reason-

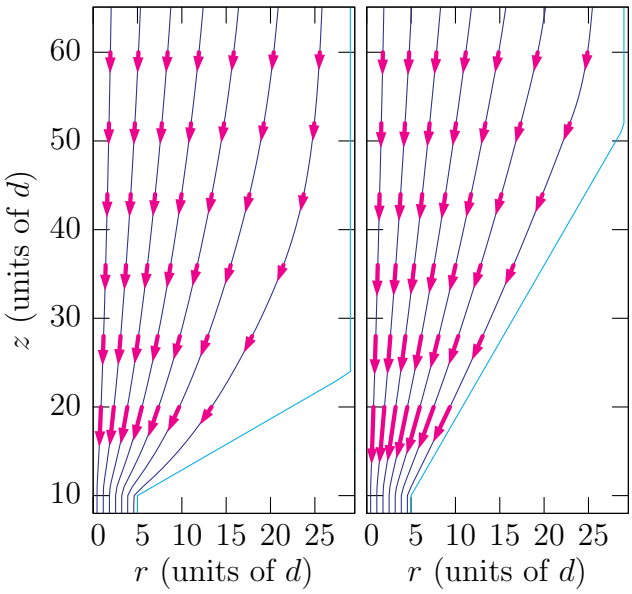


Figure 5-5: Streamlines of the mean flow in the 30° (left) and 60° reactor geometries for the numerical solution of the Kinematic Model. Arrows are proportional to the velocity vectors in selected horizontal slices.

able agreement between the Kinematic Model predictions and the DEM flow profiles, but the effect of the container geometry is not fully captured. In the converging flow of the funnel region, the streamlines are roughly parabolic, as predicted by the Kinematic Model and found in many experiments [137, 84, 114, 28, 27]. For that region, it is possible to choose a single value ($b = 3d$) to achieve an acceptable fit to the DEM flow profiles for both the 30° and 60° funnel geometries, as shown in figure 5-6.

In spite of the reasonable overall fit, the Kinematic Model has some problems describing the DEM results. It fails to describe the several particle thick boundary layer of slower velocities seen in the DEM data. In the original model, b depends only on the properties of the granular material, but we find that it seems to depend on the geometry; the best fit to the 30° DEM data is $b \approx 2.5d$, while the best fit for the 60° DEM data is $b \approx 3.0d$. Such discrepancies may partly be due to the boundary layers, since in the lower section of the container the conical walls may have an appreciable effect on the majority of the flow. We also find that the Kinematic Model fails to capture the rapid transition from converging flow to plug flow seen in the DEM data. This is shown clearly by comparing the streamlines for the Kinematic Model in figure

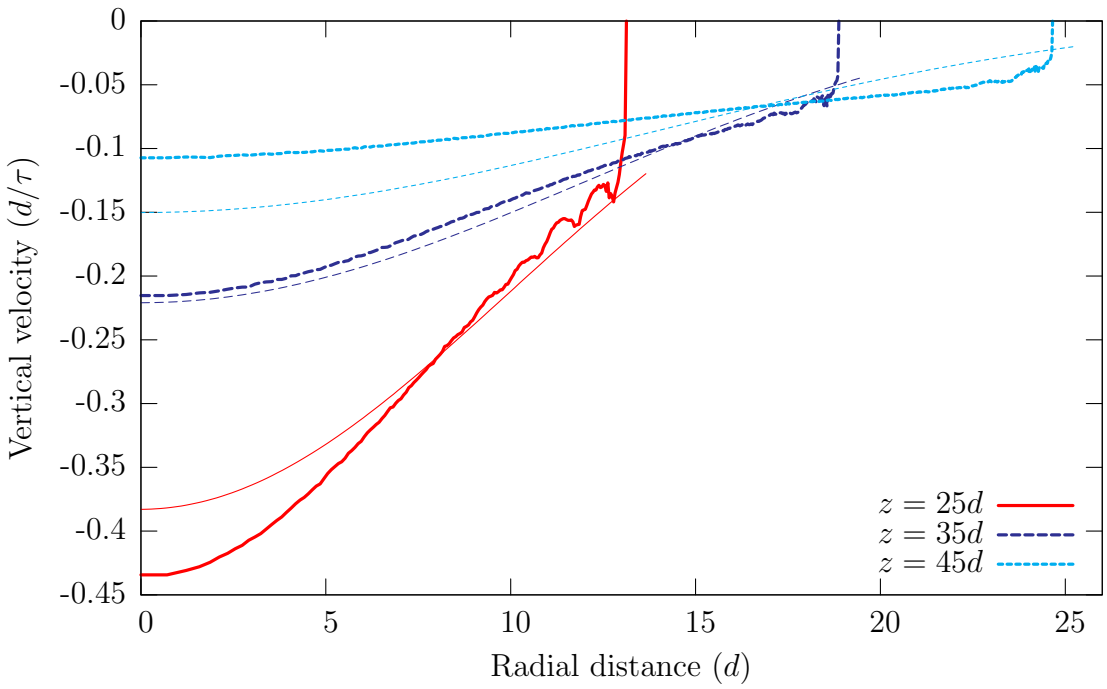


Figure 5-6: Velocity profiles for the 60° reactor geometry (heavy lines), with a comparison to the Kinematic Model for $b = 3d$ (thin lines).

5-5 with those for DEM. Streamlines for the Kinematic Model are roughly parabolic, and no single value of b can capture the rapid change from downward streamlines to converging streamlines seen in DEM.

The difficulty in precisely determining b is also a common theme in experiments, although recent data suggests that a nonlinear diffusion length may improve the fit [27]. Perhaps a more fundamental problem with the Kinematic Model is that it cannot easily describe the rapid crossover from parabolic converging flow to uniform plug flow seen in both geometries our DEM simulations; we will return to this issue in section 5.5.

5.4 Diffusion and Mixing

Nuclear engineering codes for PBR core neutronics typically assume that pebbles flow in a smooth laminar manner along streamlines, with very little lateral diffusion [131, 51]. Were such significant diffusion to occur across streamlines, it could alter the

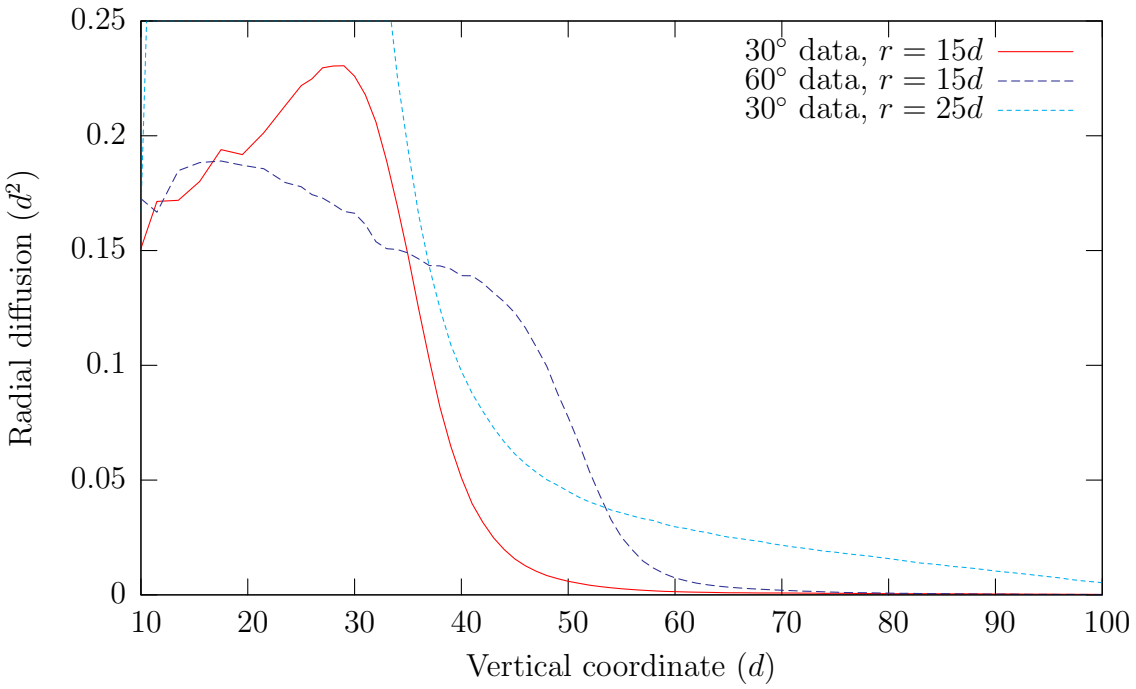


Figure 5-7: Radial diffusion of particles about streamlines of the mean flow as a function of height, z, in both reactor geometries for pebbles starting at $z = 110d$ in an annulus of radius $r = 15d$, at the edge of the dynamic central column in MPBR. For the 30° geometry, we also show data for pebbles near the wall at $r = 25d$.

core composition in unexpected ways. In the MPBR design with a dynamic central column [2], diffusion leads to the unwanted mixing of graphite pebbles from the central reflector column with fuel pebbles from the outer annulus, so it must be quantified.

Particle-tracking experiments on quasi-2D silos [28] and half-cylinder MPBR models [63] have demonstrated very little pebble diffusion in slow, dense flows, but the observations were made near transparent walls, which could affect the flow, e.g. due to ordering (see below). Three-dimensional tracking of a radioactive tracer in a cylindrical MPBR model has also shown very little diffusion, at the scale of a single pebble diameter for the duration of the flow [63]. Here, we take advantage of the complete information on pebble positions in our DEM simulations to study core diffusion and mixing with great accuracy.

We collected extensive statistics on how much pebbles deviate from the mean-flow streamlines during drainage. Consistent with theoretical concepts [17], experiments have demonstrated that the dynamics are strongly governed by the packing geometry,

so that diffusion can most accurately be described by looking at the mean-squared horizontal displacement away from the streamline, as a function of the distance dropped by the pebble (not time, as in molecular diffusion), regardless of the flow rate. Motivated by the importance of quantifying mixing at the fuel/moderator interface in the dynamic central column of MPBR, we focus on tracking pebbles passing through $z = 110d$ with $|r - 15d| < 0.16d$. The variance of the r coordinate of the particles as they fall to different heights in z can be calculated. From this, we can determine the amount of radial diffusion, defined as the increase in the variance of r of the tracked particles from the variance at the initial height.

The diffusion data for both reactor geometries is shown in figure 5-7. We see that for large values of z in the cylindrical part of the container, the pebbles undergo essentially no diffusion; this is to be expected, since we have seen that in this area the packing is essentially plug-like, and particles are locked in position with their neighbors. However for lower values of z the amount of radial spreading begins to increase, as the particles experience some rearrangement in the region corresponding to converging flow. Note however that the scale of this mixing is very small, and is much less than a pebble diameter. The height where the amount of diffusion begins to increase is approximately $z = 35d$ in the 30° geometry and $z = 50d$ in the 60° geometry. In the 30° geometry, this transition is significantly above the height of the interface between conical and cylindrical walls, while in the 60° geometry, the transition is almost level with the interface. This suggests that while the container geometry may play a role in diffusion and velocity profiles, it is a lower-order effect. For very small values of z, there is a decrease in the variance of the radial coordinate, since the pebbles must converge on the orifice as they exit the container.

We applied a similar analysis for different initial values of r, and found very similar results over the range $0 < r < 25d$. However, for particles close to the container boundary, very different behavior is observed, as shown by the third line in figure 5-7 for particles with $|r - 25d| < 0.10d$. In this region, the particles undergo rearrangement, and this causes a (piecewise) linear increase in the mean-squared displacement with distance dropped, which corresponds to a constant local diffusion

length. There is also evidence of a sharp transition in the boundary-layer diffusion length, which increases significantly as pebbles pass the corner into the converging-flow region of the funnel.

5.5 Packing Statistics

5.5.1 Pebble Volume Fraction

Pebble-bed experiments [50, 119] and simulations [38, 103] of static sphere packings in cylinders have revealed that there are local variations in porosity near walls, at the scale of several pebble diameters, but there has been no such study of flowing packings, averaging over dynamic configurations. Similar findings would have important implications for helium flow in the core, since the local gas permeability is related to the porosity [30, 141, 143].

First, we study the distribution of local volume fraction (% of volume occupied by pebbles) throughout the container, averaged in time. (The porosity is one minus the volume fraction.) We make use of the Voronoi algorithm discussed in the previous chapter to examine the packing fraction in vertical cross sections through the containers. Figure 5-8 shows density snapshots for cross sections through the thirty degree and sixty degree reactor geometries, based on computing the local density at a particular point by averaging over the Voronoi densities of particles within a radius of $2.2d$. Figure 5-9 shows density plots over the entire flow of the data, but using a smaller averaging radius of $0.8d$. Many interesting features are visible, which corroborate our other results. High in the center of the container, we see that the local packing fraction is mostly close to 63%, suggesting that the plug-like region is in a nearly jammed and rigid state. This is consistent with our earlier reactor data showing nearly uniform plug flow with no significant diffusion or mixing.

We also observe two annular lines of lower density propagating down from the guide ring, which form due to wall effects on the guide ring itself (see below) and are advected downward. The fact that these subtle artifacts of the guide-ring constraints

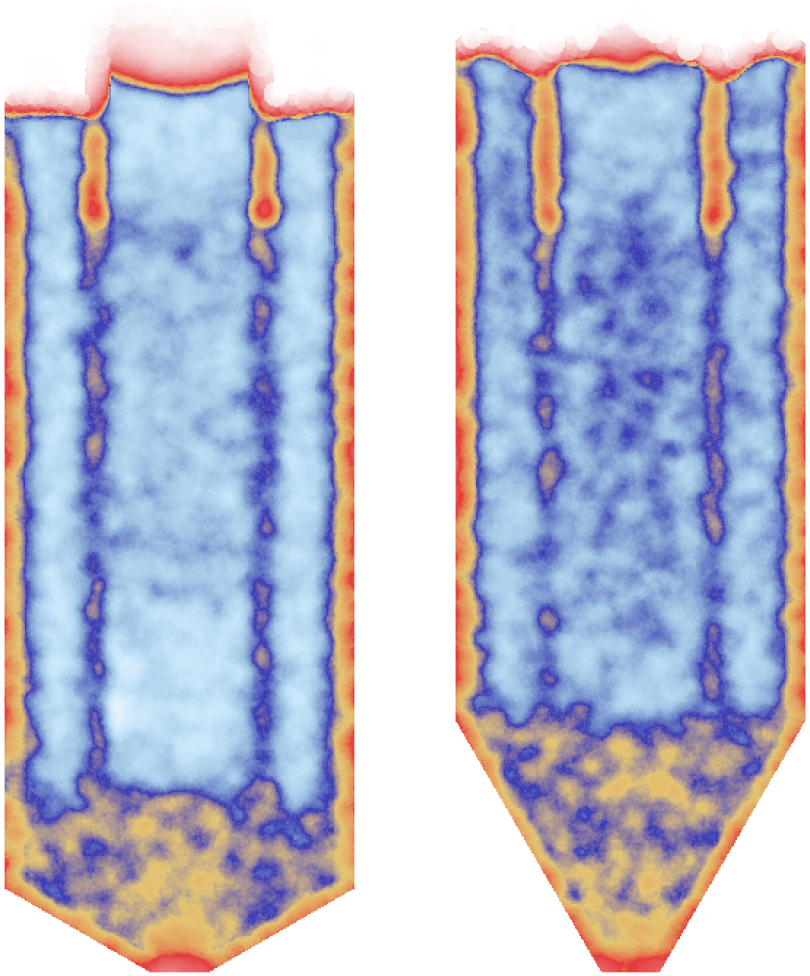


Figure 5-8: Plots of local volume fraction (1 − porosity) in a vertical cross section for the thirty degree reactor geometry (left) and the sixty degree reactor geometry (right), calculated using a Voronoi cell method. Volume fractions of 50%, 57%, 60%, and 63% are shown using the colors of red, yellow, dark blue, and cyan respectively. Colors are smoothly graded between these four values to show intermediate volume fractions. High in the bulk of the container, the packing fraction is approximately 63%, apart from in a small region of lower density at $r_{\text{in}} = 14.5d$, corresponding to packing defects introduced by the guide ring. In both geometries a sharp reduction in density is observed in a region above the orifice, where particles in the parabolic flow region are forced to undergo local rearrangements.

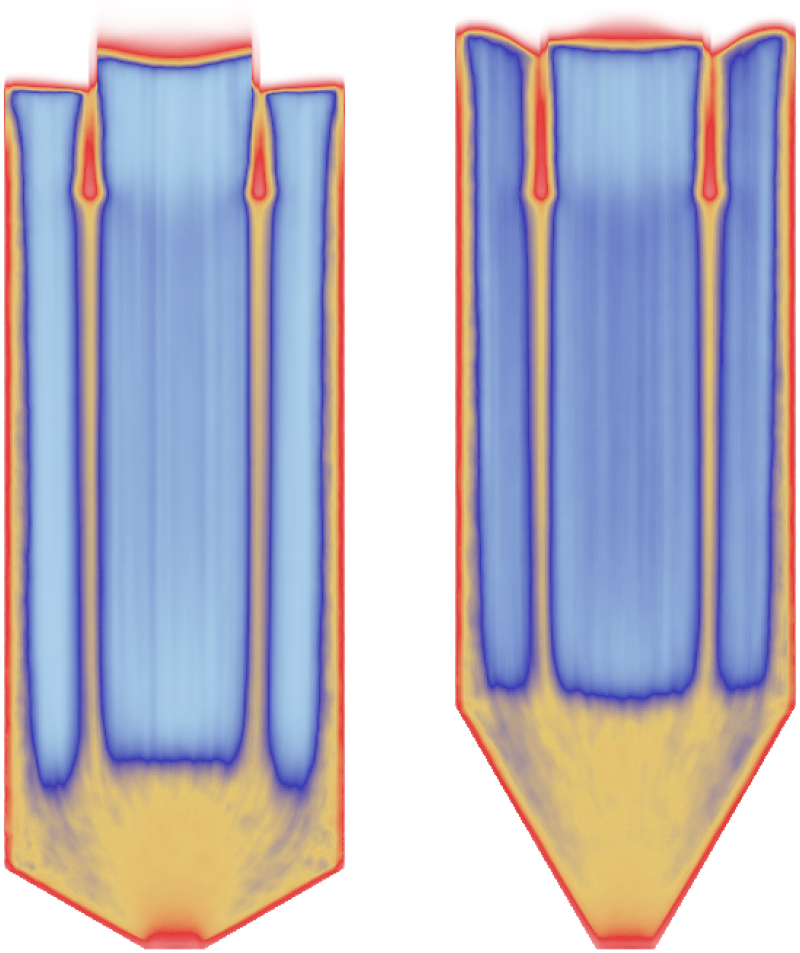


Figure 5-9: Time-averaged plots of the local volume fraction, using the same color scheme as figure 5-8.

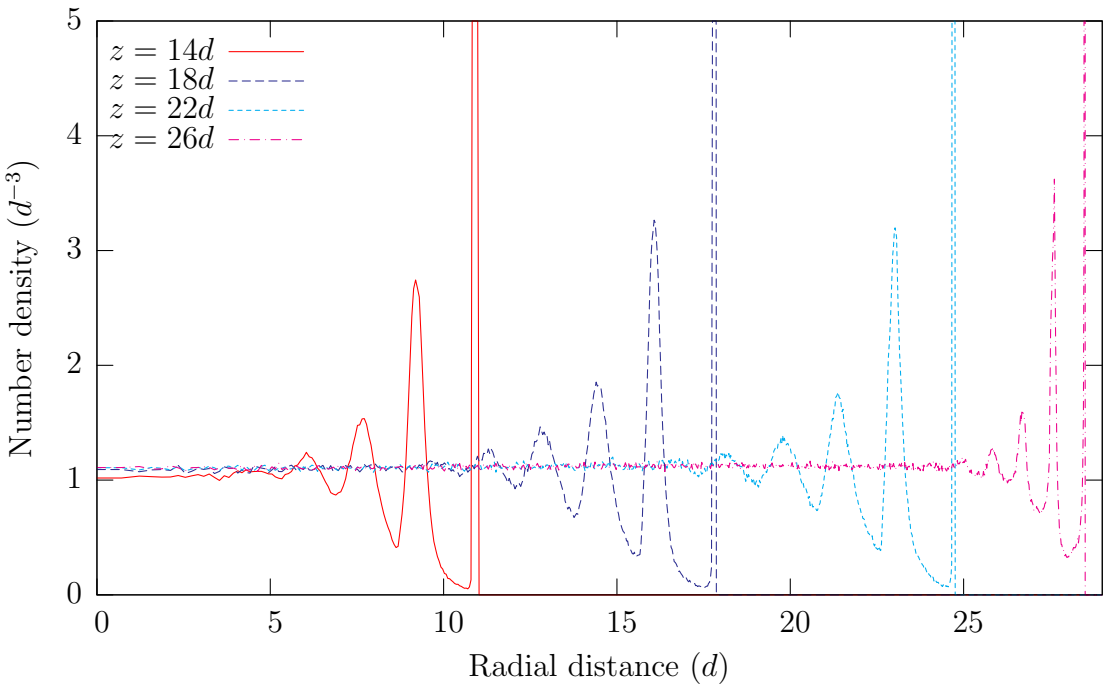


Figure 5-10: Plots of number density (defined as the number of particle centers per unit volume) in the 30° reactor geometry for several low cross sections.

are felt far down in the flow further demonstrates that very little diffusion or shearing occurs in the upper region. There are also similar lower-density regions along the walls, related to partial crystallization described in more detail below.

It is also clear in both geometries, especially the 30° model, that there is a fairly sharp transition between the upper region of nearly rigid plug flow and a less dense lower region of shear flow in the funnel. Similar features are in the velocity profiles described above, but the transition is much more sharp, at the scale of at most a few particles, for the local packing fraction. These sudden variations in material properties and velocities are reminiscent of shock-like discontinuities in Mohr-Coulomb plasticity theories of granular materials [94, 117]. It seems no such existing theory can be applied to the reactor flows, but our results suggest that plasticity concepts may be useful in developing a continuum theory of dense granular flow [65].

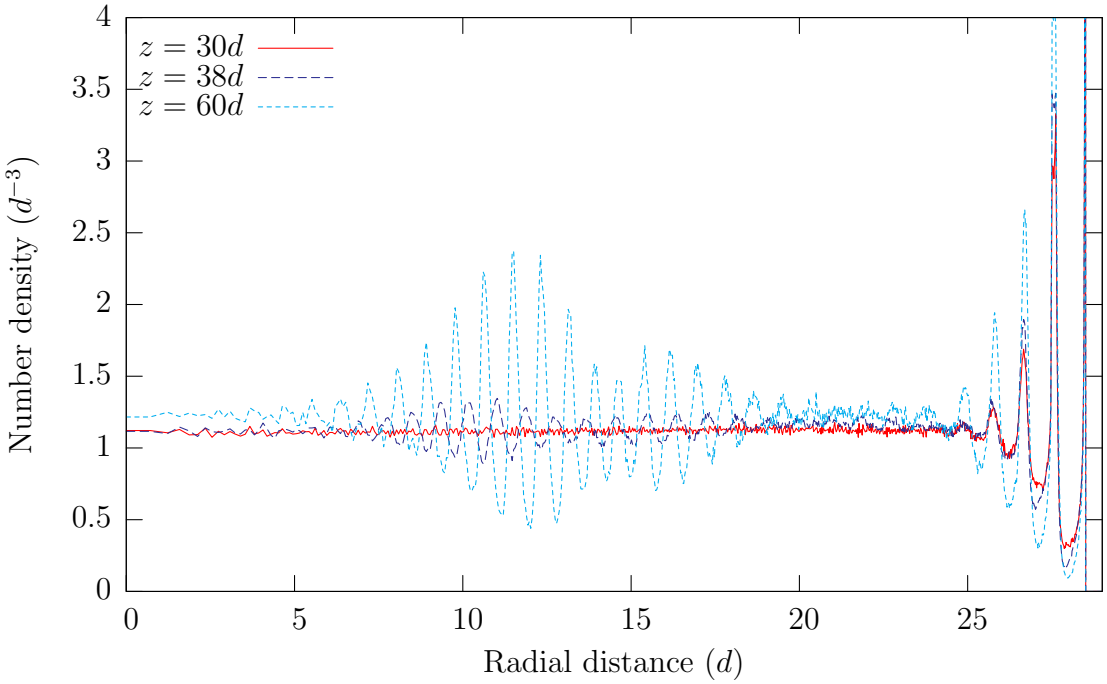


Figure 5-11: Number density plots in the 30° reactor geometry for several high cross sections.

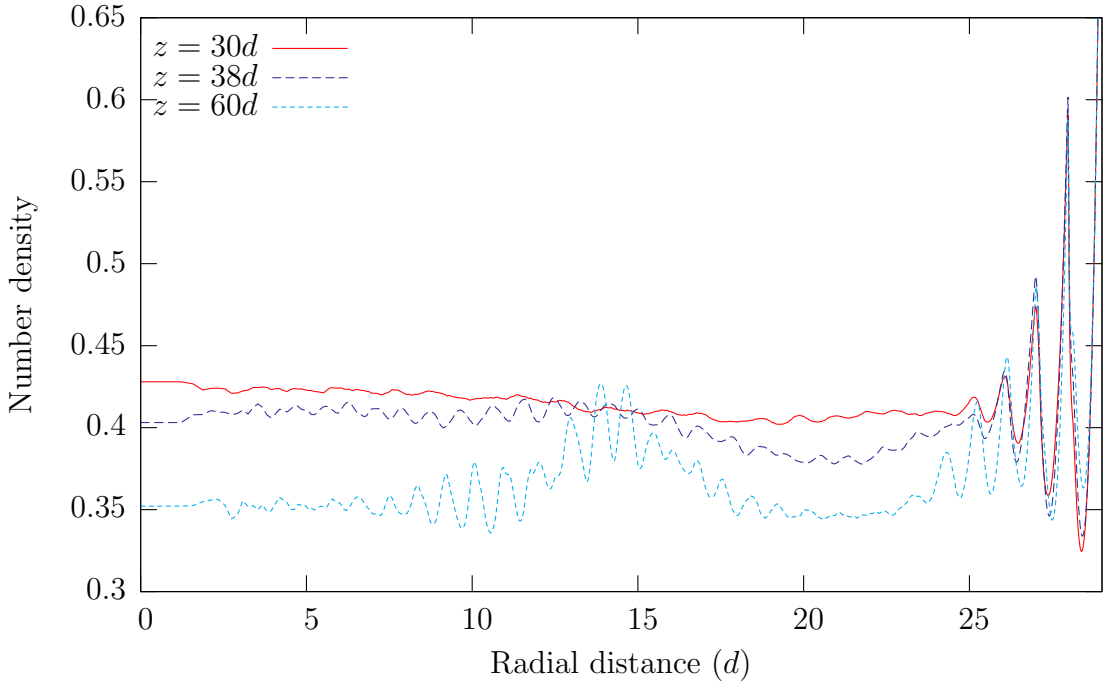


Figure 5-12: Horizontal profiles of porosity at different heights in the 30° reactor geometry.

5.5.2 Local Ordering and Porosity

As noted above, previous simulation studies of local ordering near walls have focused on static packings in simplified cylindrical geometries (without the funnel, outlet pipe, or guide ring) [38, 103], while we compute average statistics for slowly flowing packings in realistic full-scale reactor models. To take a closer look at ordering near walls, we study the number density profile in horizontal slices at different heights. The container is divided into bins in the same way as discussed previously and the number density in a bin is obtained by counting the number of times a particle center lies within that bin.

Figure 5-10 shows a sequence of number density profiles for several low values of z in the thirty degree reactor geometry. At all four heights, lattice effects are clearly visible and quite similar to those observed in experiments [50, 119] and other simulations [38, 103]. For the lowest three heights, these peaks are roughly $\sqrt{3}d$ apart, corresponding to particles crystallized against the conical wall, while for the highest value of z, these effects are roughly $1d$ apart, due to particles being crystallized against the cylindrical wall. The above graph also shows that in the middle of the container, no lattice effects are present.

However, this situation changes dramatically higher up in the container, as shown in Fig. 5-11. As z increases from $30d$ to $60d$, the interior of the packing goes from being disordered to having a strong radial ordering, centered at around $z = 12d$. The reason for this ordering is due to the presence of the guide ring high in the container, which keeps the fuel and moderator pebbles separate. The ring, placed at $r_{\text{in}} = 14.5d$ in the container, creates radial crystallization, which can then propagate very far downward, since the packing is plug-like for most of the cylindrical part of the reactor. At much lower heights, around $z = 40d$, this radial ordering is broken, as the particles are forced to reorganize once they enter the parabolic region of flow.

To make a direct connection with the modeling of gas flow, we show horizontal slices of the porosity at different heights in figure 5-12. The porosity is measured here by intersecting the spheres with annular cylindrical bins to compute the fraction

of each bin volume not occupied by pebbles. The features noted above appear in the porosity and alter the local permeability, which enters continuum descriptions of helium gas flow in the core [30, 141, 143].

5.6 Residence-Time Distribution

5.6.1 Predictions of the Kinematic Model

The statistical distribution of fuel burnup is closely related to the distribution of pebble residence times in the reactor core, differing only due to nonuniform sampling of the neutron flux profile. Since the upper pebble flow is essentially a uniform plug flow, the distribution of residence times is the same (up to a constant time shift) as the distribution of waiting times for pebbles starting at a given height in the core to exit through the orifice, and we concentrate on these distributions in this section. However, we conclude by examining the residence times for particles to pass through the entire container, to investigate the effects of the guide ring and the outer walls.

We have seen that there is very little pebble diffusion, so fluctuations in the residence time are primarily due to hydrodynamic dispersion in the mean flow. We have also seen that the Kinematic Model gives a reasonable description of the mean flow profile in the conical funnel region, where most of the shear and hydrodynamic dispersion occur. Therefore, we can approximate the residence-time distribution by the distribution of times to travel along different streamlines of the mean flow, starting from different radial positions, r_0, at a given height z_0. Below we will compare such predictions, based on our numerical solutions to the Kinematic Model, to our DEM simulations for the two reactor geometries.

5.6.2 An Analytical Formula

We can obtain a simple, exact formula for the residence-time distribution in a somewhat different geometry using the Kinematic Model, as follows. The similarity solution to Eq. (5.1) for a wide, flat bottomed silo draining to a point orifice at $z = 0$

is

$$u(r,z) = -\frac{Qr}{2bz^2}e^{-r^2/4bz} \tag{5.2}$$

$$v(r,z) = \frac{Q}{bz}e^{-r^2/4bz} \tag{5.3}$$

where u and v are the radial (horizontal) and downward velocity components and Q is a constant proportional to the total flow rate through the orifice. (This is just the classical Green function for the diffusion equation in two dimensions, where z acts like "time".) A slightly more complicated solution is also possible for a parabolic silo, but let us focus on the simplest case of Eqs. (5.2)-(5.3), which is a good approximation for a wide parabolic funnel, where the velocity near the walls is small, i.e. $R > \sqrt{4bz_0}$. A more detailed analysis is not appropriate here, since a simple analytical solution does not exist for the actual reactor geometry of a conical funnel attached to straight cylinder.

For the flow field in Eqs. (5.2)-(5.3), the trajectory of a Lagrangian tracer particle along a streamline is given by

$$\frac{dr}{dt} = u(r,z), \quad r(t=0) = r_0 \tag{5.4}$$

$$\frac{dz}{dt} = -v(r,z), \quad z(t=0) = z_0 \tag{5.5}$$

Combining these equations and integrating, we find that the streamlines are parabolae, $z/z_0 = (r/r_0)^2$, and that the residence time for a pebble starting at (r_0, z_0) is

$$\tau_0(r_0,z_0) = \frac{bz_0^2}{2Q}e^{r_0^2/4bz_0}. \tag{5.6}$$

Now we consider pebbles that are uniformly distributed at a height z_0 in a circular cross section of radius R in the flow field Eqs. (5.2)-(5.3. The probability distribution

for the residence time of those pebbles is

$$p(\tau|z_0,R) = \int_0^R \delta(\tau-\tau_0(r_0,z_0))\frac{2\pi r_0 dr_0}{\pi R^2} \tag{5.7}$$

$$= \begin{cases} 0 & \text{for } \tau < \tau_{min}(z_0) \\ 4bz_0/R^2\tau & \text{for } \tau_{min} < \tau < \tau_{max} \\ 0 & \text{for } \tau > \tau_{max}(z_0,R) \end{cases} \tag{5.8}$$

where

$$\tau_{min} = \tau_0(0,z_0) = \frac{bz_0^2}{2Q} \tag{5.9}$$

$$\tau_{max} = \tau_0(R,z_0) = \frac{bz_0^2}{2Q}e^{R^2/4bz_0} \tag{5.10}$$

Once again, this solution is strictly valid for an infinitely wide and tall silo draining to a point orifice, and it is roughly valid for a parabolic funnel, $z/z_0 = (r/R)^2$, as an approximation of a conical funnel in the actual reactor geometry. We can further approximate the effect of a nearly uniform flow of speed v_0 to describe the upper cylindrical region by simply adding $(z-z_0)/v_0$ to the residence time for a starting point $z > z_0$.

Although this analysis is for a modified geometry, we will see that it captures the basic shape of the residence-time distributions from the DEM simulations in a simple formula (5.8). The probability density is sharply peaked near the shortest residence time, τ_{min}, corresponding to pebbles near the central axis traveling the shortest distance at the largest velocity. The longer distance and (more importantly) the smaller velocity at larger radial positions cause strong hydrodynamic dispersion, resulting a fat-tailed residence-time density which decays like $1/t$, up to a cutoff τ_{max}.

5.6.3 Simulation Results

For the DEM reactor simulations, we calculate the distribution of times it takes for particles to drop from several different values of z_0, adding in a weighting factor to take into account that shorter residence times are preferentially observed in the data

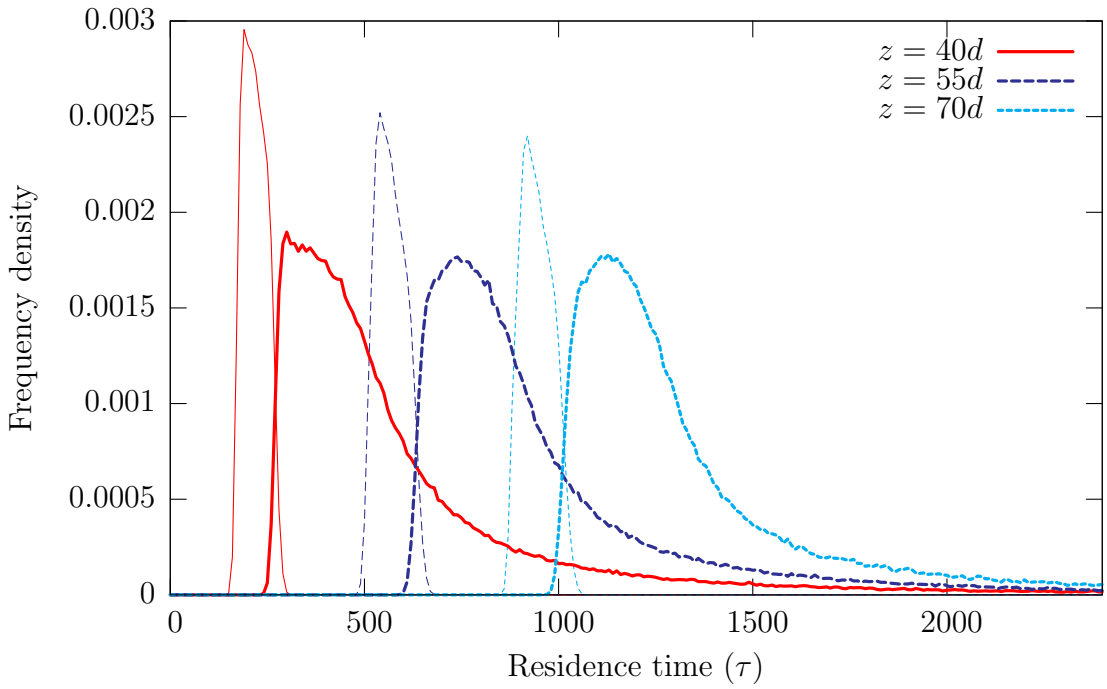


Figure 5-13: Residence-time probability densities for the time it takes particles to drop from a specific height z out of the container, for the 30° reactor geometry for fuel pebbles (heavy lines) and for moderator pebbles (thin lines).

set.

Since we are primarily interested in the radioactive burnup, we concentrate on the residence times for the fuel pebbles, but for comparison, we also report results for the moderator pebbles. Figure 5-13 shows the residence-time probability densities for pebbles starting at $z = 40d, 55d, 70d$ to exit the container for the 30° reactor geometry. The distributions for the moderator pebbles are quite narrow, showing all particles exit over a short time window. In contrast, the distributions for the fuel pebbles exhibit fat tails, as expected qualitatively from the Kinematic Model approximation (5.8) for a parabolic geometry. A closer analysis of the data confirms that the longest waiting times are associated with pebbles passing close to the walls, especially near the corner between the conical and cylindrical wall sections, although there are no completely stagnant regions.

Figure 5-14 shows corresponding plots for the 60° reactor geometry. In general, the residence-time densities have similar shapes as for the 30° geometry, but they are

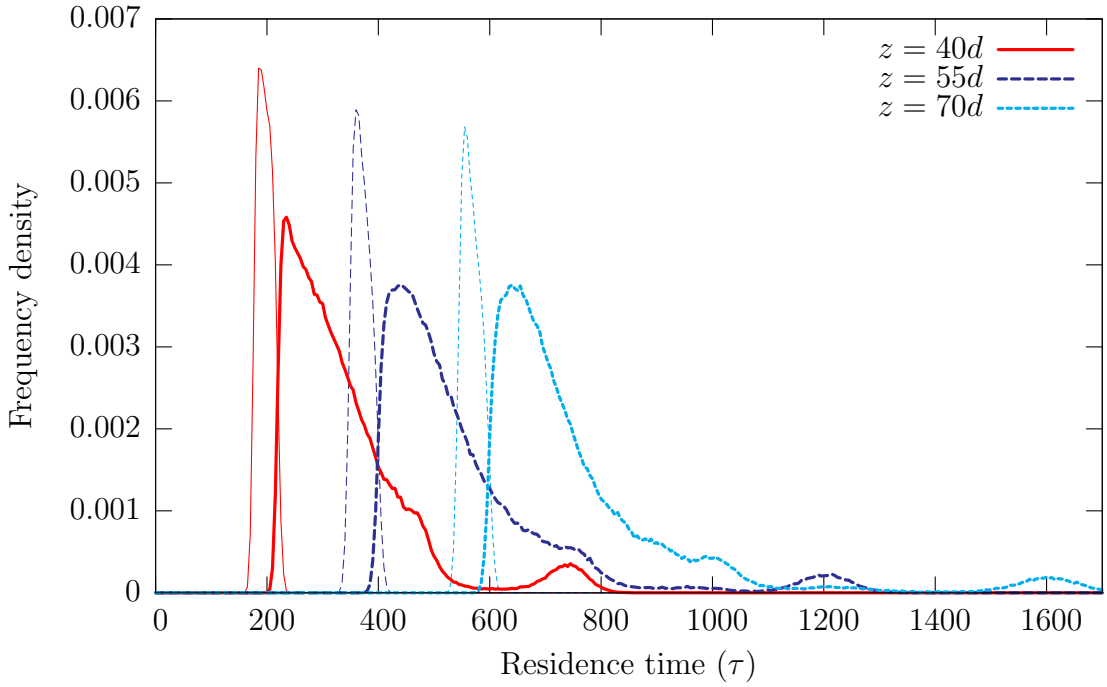


Figure 5-14: Residence-time probability densities for the time it takes particles to drop from a specific height z out of the container, for the 60° reactor geometry for fuel pebbles (heavy lines) and for moderator pebbles (thin lines).

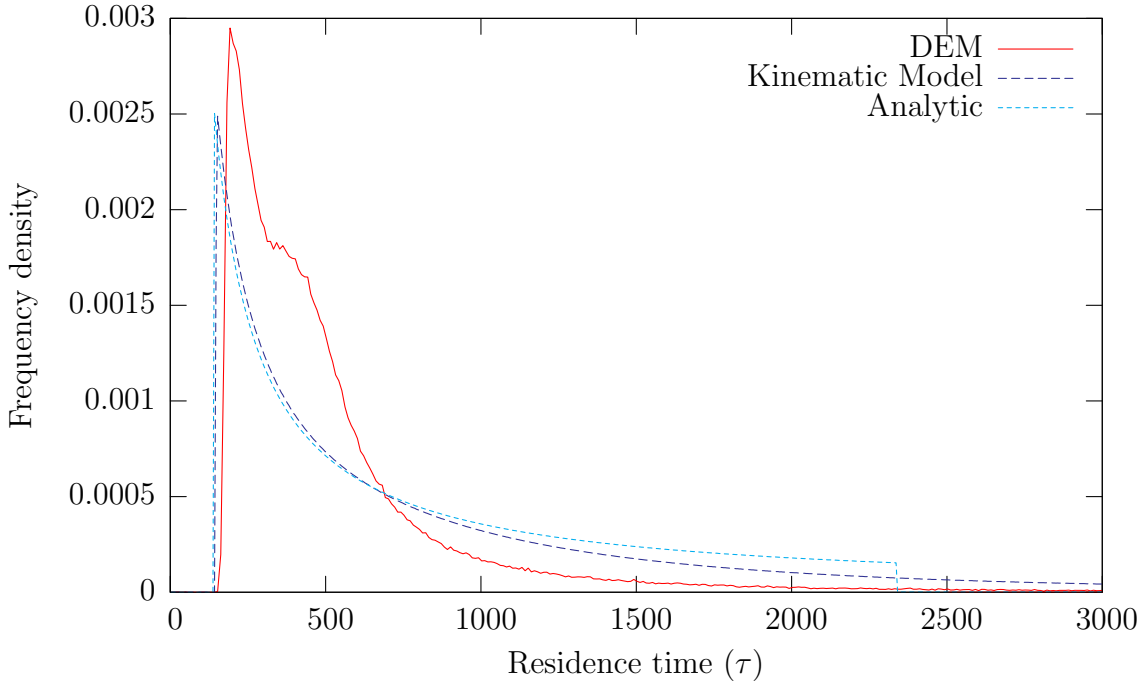


Figure 5-15: Comparison of the residence time distributions between DEM simulation, numerical solution of the Kinematic Model, and the analytic formula.

much narrower and exhibit a small secondary peak far into the tail. Examining movies shows that this extra peak is due to a boundary layer of particles, roughly one-pebble thick, touching the 60° conical wall sliding down at a speed lower than the nearby bulk. This extra source of hydrodynamic dispersion could not be easily captured by a continuum model for the mean flow. A simple way to eliminate it would be to replace add an outer annulus of moderator pebbles (controlled by another guide ring at the top), which would flow more slowly along the walls, leaving the fuel pebbles in a more uniform flow with smaller fluctuations. Another possibility would be to reduce the wall friction, which makes the flow more uniform, as discussed in the following section.

Figure 5-15 investigates the accuracy of the Kinematic Model in predicting the DEM residence-time distribution. The total residence-time distribution for both fuel and moderator pebbles to exit the reactor from $z = 40d$ in the 30° geometry is shown, and is compared with two predictions from the Kinematic Model, one making use of the analytic formula (5.8), and one making use of the numerical solution of the velocity profile. We use of the value $b = 2.5d$ and calibrate the total flow to match the total flow from the DEM data. Both the numerical solution and the analytic formula can roughly capture the overall shape of the DEM distribution, although neither achieves a good quantitative agreement, particularly in the tails. Since the analytic formula assumes all streamlines are parabolic, it fails to take into account the slow-moving particles that stay close to the wall, and it therefore predicts a cut-off in the residence time distribution which is much shorter than some of the observed residence times in the DEM simulation. The numerical solution of the Kinematic Model accounts for this and provides a better match, although it is clear that a model correctly accounting for the flow of pebbles near the container walls may be required in order to achieve high accuracy.

5.6.4 Residence times for the entire container

We also considered the distribution of times for the particles to pass through the entire container. While the flow in the upper part of the reactor is essentially plug-

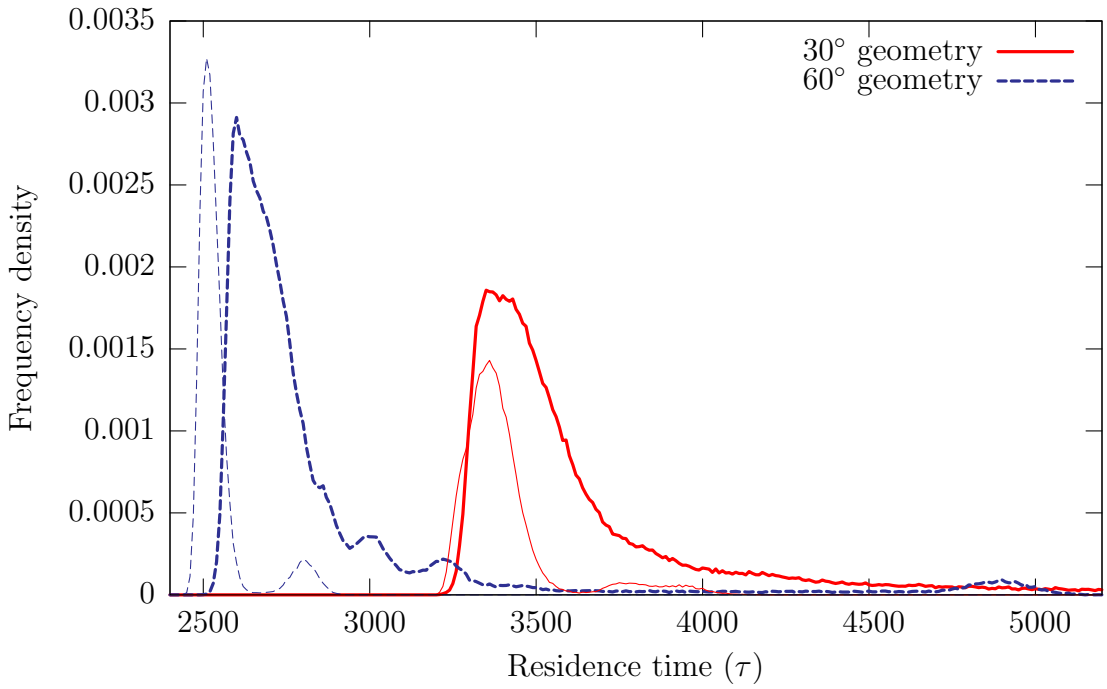


Figure 5-16: Distribution of times to pass through the entire container for fuel pebbles (heavy lines) and moderator pebbles (thin lines).

like, boundary effects near the container walls and on the guide ring can have an appreciable effect on the pebble residence times, which we study here. Since it takes a long time for particles to pass through the entire container we made use of the two extended data sets, consisting of 1,427 snapshots for the thirty degree geometry and 1,249 snapshots for the sixty degree geometry.

Figure 5-16 shows the time distributions for pebbles to pass through the entire container. Apart from a large positive time shift, the curves are similar in form to those in Fig. 5-13 and Fig. 5-14. However, for both geometries, we see second small peaks in the distributions for the moderator pebbles, corresponding to a slow-moving boundary layer of pebbles touching the guide ring. The sixty degree curve for the fuel pebbles also exhibits several undulations corresponding to multiple layers of pebbles crystallized against the outer wall, each moving at different speeds.

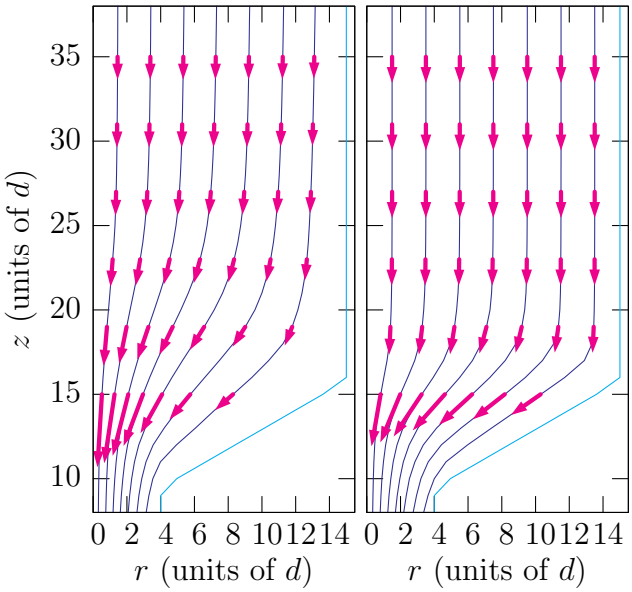


Figure 5-17: Streamlines for the half-size, monodisperse geometries with wall friction (left) and without wall friction (right). Arrows are proportional to the velocity vectors in selected horizontal slices.

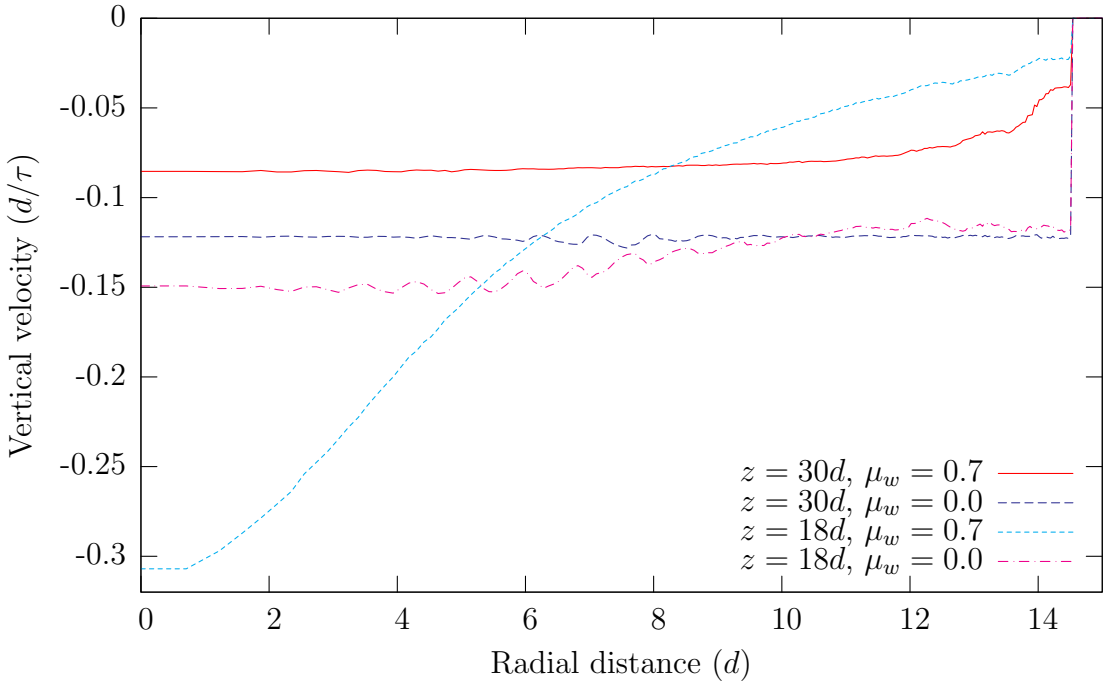


Figure 5-18: Comparison of velocity profiles for simulations with and without wall friction for two different heights.

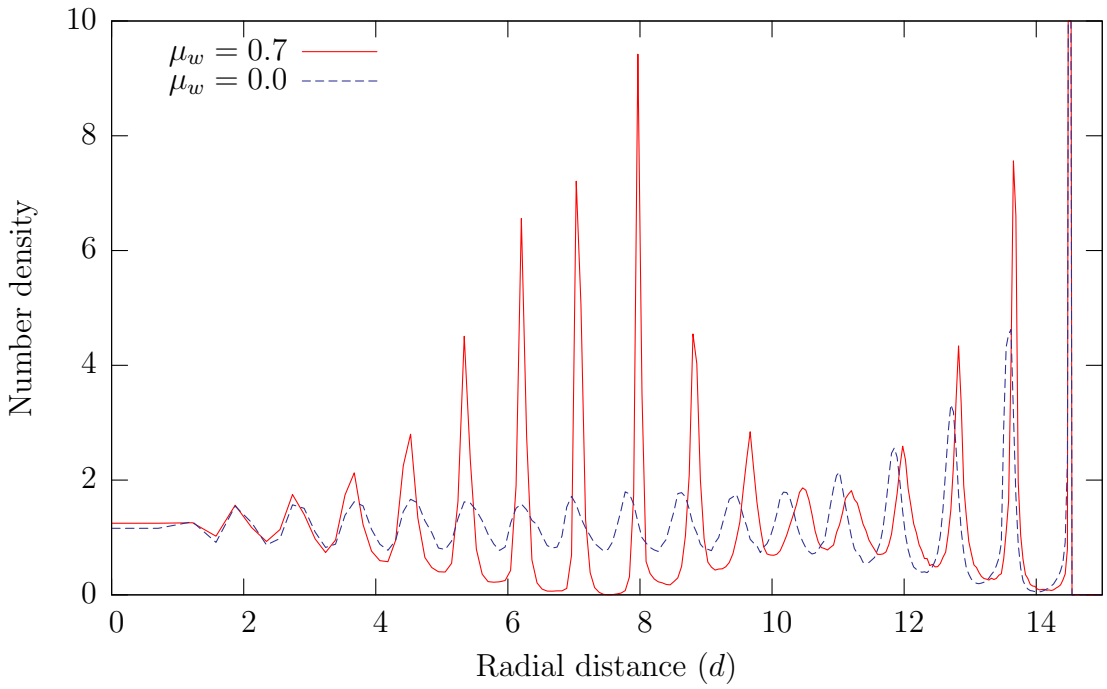


Figure 5-19: Comparison of number density profiles at $z = 60d$ for simulations with and without wall friction.

5.7 Wall friction

The behavior of pebbles near the walls is of significant interest to reactor design, and to look into this further, we investigated the effect of wall friction by comparing two simulations runs in the half-size geometry, with wall friction coefficients $\mu_w = 0$ and $\mu_w = 0.7$. All other aspects of the simulation, including the interparticle interactions, were kept the same.

Figures 5-17 and 5-18 show comparisons of flow streamlines and velocity profiles respectively for the two simulations. We see that the $\mu_w = 0$ simulation results in a significantly larger flow speed, with a mass flow rate of $104m\tau^{-1}$, as opposed to $59.6m\tau^{-1}$ for $\mu_w = 0.7$. As would be expected, removing wall friction also removes the boundary layer of slower velocities at the wall, creating an almost perfectly uniform velocity profile high in the reactor. This also has the effect of increasing radial ordering effects, and we can see from figure 5-19 that the number density profile is more peaked close to the wall. Figure 5-19 also shows that the radial ordering created by the guide ring is also significantly enhanced. While this is due in part to the more plug-like flow

allowing packing effects to propagate further down, it is also due to the frictionless guide ring initially creating radial ordering. Thus it may be possible to tune the material properties of the guide ring (or the roughness of its walls) to enhance or reduce the radial ordering effects.

Removing wall friction also has the effect of increasing radial ordering effects near the wall. Perhaps most surprisingly, removing wall friction results in a significant alteration of the flow in the *interior* of the packing, as shown by the two velocity profiles in figure 5-18 for $z = 18d$. While both velocity profiles must converge upon the orifice, we see that the velocity profile for the $\mu_w = 0.7$ case is significantly more curved than that for $\mu_w = 0$. This also has the effect of preferentially speeding up the relative flux of fuel pebbles: with wall friction, the fuel pebbles make up 71.5% of the total mass flux, but without wall friction, this increases to 74.7%.

5.8 Bidispersity

5.8.1 The Bidisperse PBR Concept

The two-pebble design of MPBR with a dynamic central moderator column has various advantages over a solid graphite central column (as in the revised PBMR design). For example, it flattens the neutron flux profile, while preserving a very simple core vessel without any internal structures, which would be subjected to extreme radiation and would complicate the granular flow. It also allows the widths of the moderator column and fuel annulus to be set "on the fly" during reactor operation, simply by adjusting the guide ring at the top.

A drawback of the dynamic moderator column, however, is its porosity, which allows the passage of the helium-gas coolant, at the highest velocity (along the central axis). In most PBR designs, high-pressure helium gas is introduced from a reservoir above the core, through holes in the graphite bricks which make up the core vessel. The gas then flows through the core and exits through holes in the graphite bricks of the conical funnel to another reservoir at high temperature. To improve the thermal

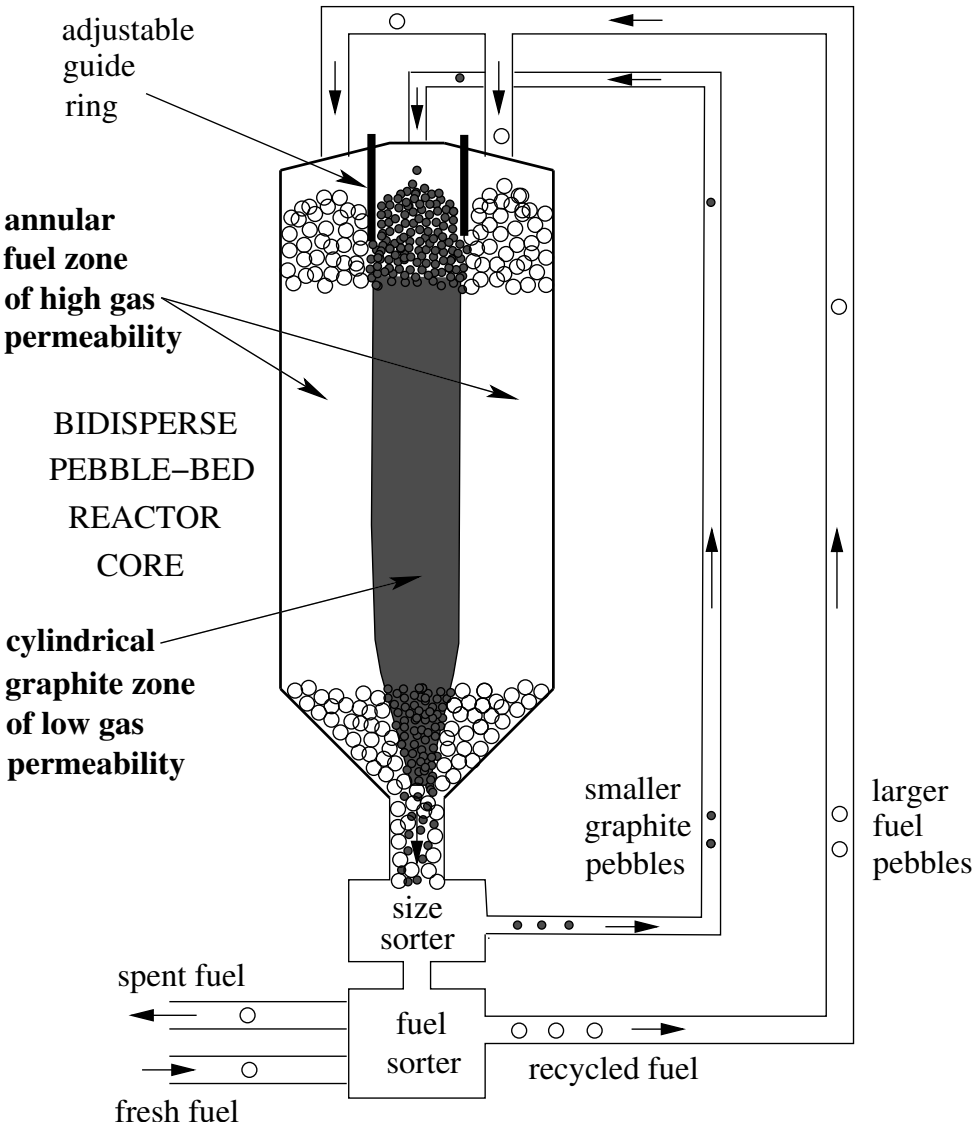


Figure 5-20: Schematic diagram of the pebble flow in a bidisperse MPBR design.

efficiency and power output, it would be preferable to focus the gas flow on the fuel annulus and the interface with the moderator column, where the most heat is generated. This is automatically achieved with a solid graphite column, but there is a very simple way to shape the gas flow in a similar way with a dynamic column, while preserving its unique advantages.

The idea is to make the graphite moderator pebbles in the central column smaller than the fuel pebbles in the outer annulus, as shown in Fig. 5-20. (This also helps with sorting of fuel and moderator pebbles as they exit the core.) In standard continuum models of flow in porous media [30, 141, 143], the permeability of the packing scales with the square of the pebble diameter (or pore size), so reducing the diameter of the moderator pebbles can greatly reduce the gas flow (e.g. by a factor of four for half-diameter pebbles). This argument holds everywhere that the packing is statistically the same, in the monodisperse packings of the fuel annulus and the moderator column, which have the same porosity. At the interface between the two regions, we have seen in Figures 5-8 and 5-12 that the porosity is enhanced for a monodisperse core due to the guide ring, although a bidisperse interface will have different structure. In summary, if helium gas is introduced outside the guide ring in a bidisperse core, it can be made to pass almost entirely through the fuel annulus and the interface with the moderator column.

5.8.2 Simulation Results

The only question regarding the feasibility of the bidisperse core is the stability of the central column over time and the possibility of enhanced diffusion of the small moderator pebbles into the annulus of larger fuel pebbles. In other systems, such as rotating drums [96, 70, 69], vibrated buckets [121, 120], and draining silos [114], bidisperse granular materials display a tendency to segregate (rather than mix) during dynamics, but there is currently no general theory which could be applied to our reactor geometry. Therefore, our DEM simulations provide a useful means to address this important question.

Figure 5-21 shows snapshots of vertical cross sections for the three different bidis-

perse simulations that were run in the half-size geometry. As shown in the diagram, the central column remains stable and coherent in all three cases, and very little mixing between the two types of pebbles is visible. Figure 5-22 shows a comparison of the velocity profiles from the three simulations for two different heights. It is reassuring to see that the bidisperse simulations do not significantly differ from the monodisperse simulation, although we do see a slightly higher overall flow rate in the bidisperse systems: we see total mass flow rates of $59.6m\tau^{-1}$, $60.8m\tau^{-1}$, and $65.0m\tau^{-1}$ for the monodisperse, 0.8:1, and 0.5:1 simulations respectively.

The velocity profiles are slightly more curved in the bidisperse central core; this is particularly apparent in the 0.5:1 simulation. This leads to a small cusp in the velocity profile near the interface between the two types of particles which may lead to adverse mixing effects. The faster flow also leads to a significantly larger turnaround of the moderator pebbles. In the monodisperse system, the moderator pebbles comprise 28.5% of the total mass flux, but this is increased to 31.7% in the 0.8:1 bidisperse simulation, and 42.6% in the 0.5:1 bidisperse simulation.

To investigate the amount of mixing of the central column, we used a technique similar to that described in section 5.4. At $z = 110d$ all moderator particles with $r > 8d$ are marked, and their radial diffusion is then calculated as a function of z. The results are shown in figure 5-23: in the cylindrical section of the packing, there is very little difference between the three simulations, but in the area of convergent flow, we see that bidispersity leads to significantly more mixing. However, even for the 0.5:1 simulation, the scale of diffusion is still smaller than a single particle diameter, and essentially the central column remains stable.

Due to computational limitations, we were unable to investigate smaller size ratios in the reactor geometries, so we carried out simulations in a smaller container with a 0.3:1 size ratio (figure 5-24) and found dramatically different behavior: During drainage, the central column became unstable, and the small particles penetrated many particle diameters into the packing of larger particles. We expect that there is a fundamental crossover in behavior simply due to geometry of amorphous packings, when the moderator pebbles become small enough to pass through the gaps between

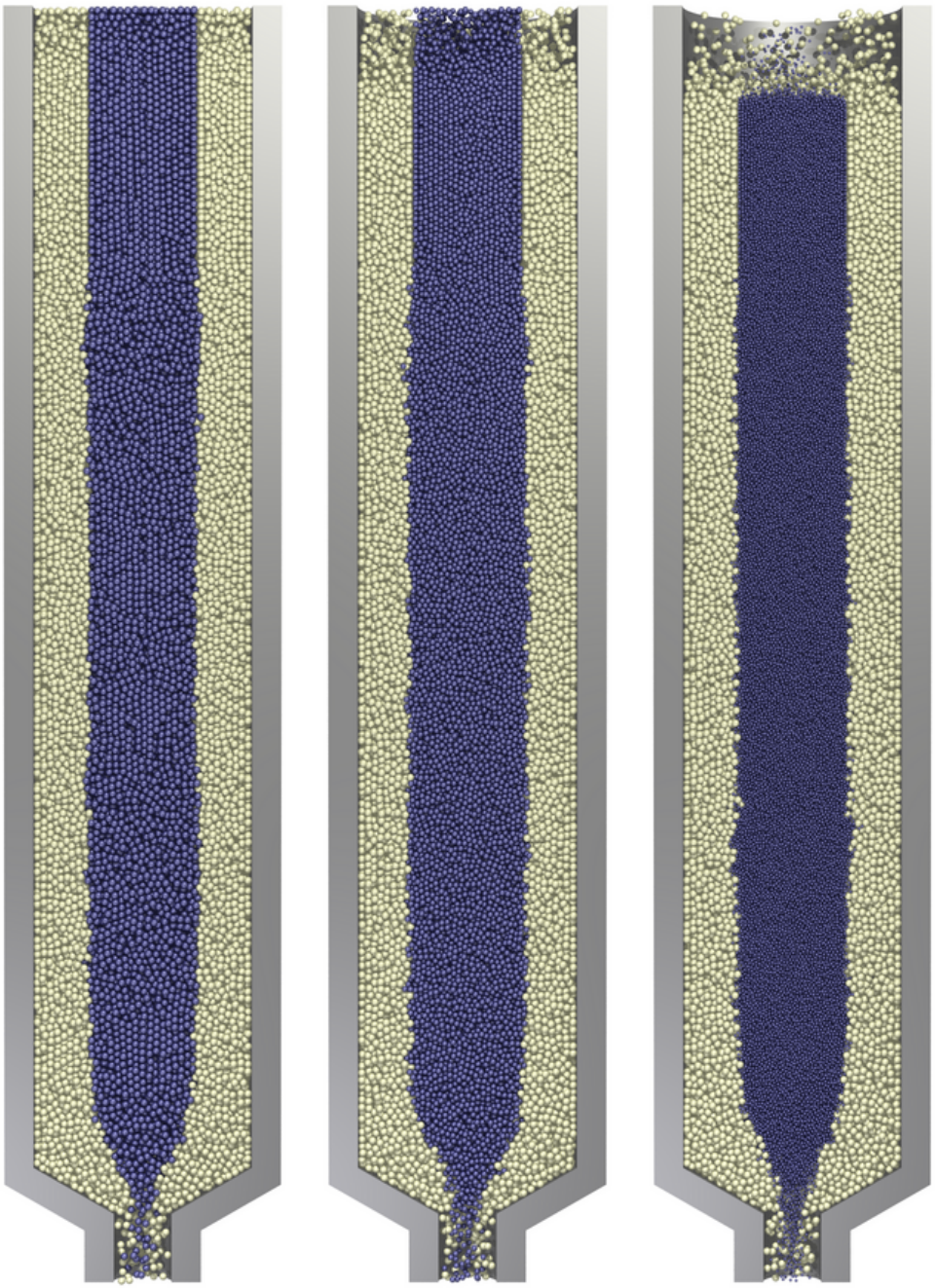


Figure 5-21: Snapshots of vertical cross-sections for the bidisperse simulations. From left to right, the moderator pebbles have diameters $1d$, $0.8d$, and $0.5d$ while the fuel pebbles are of constant size $1d$.

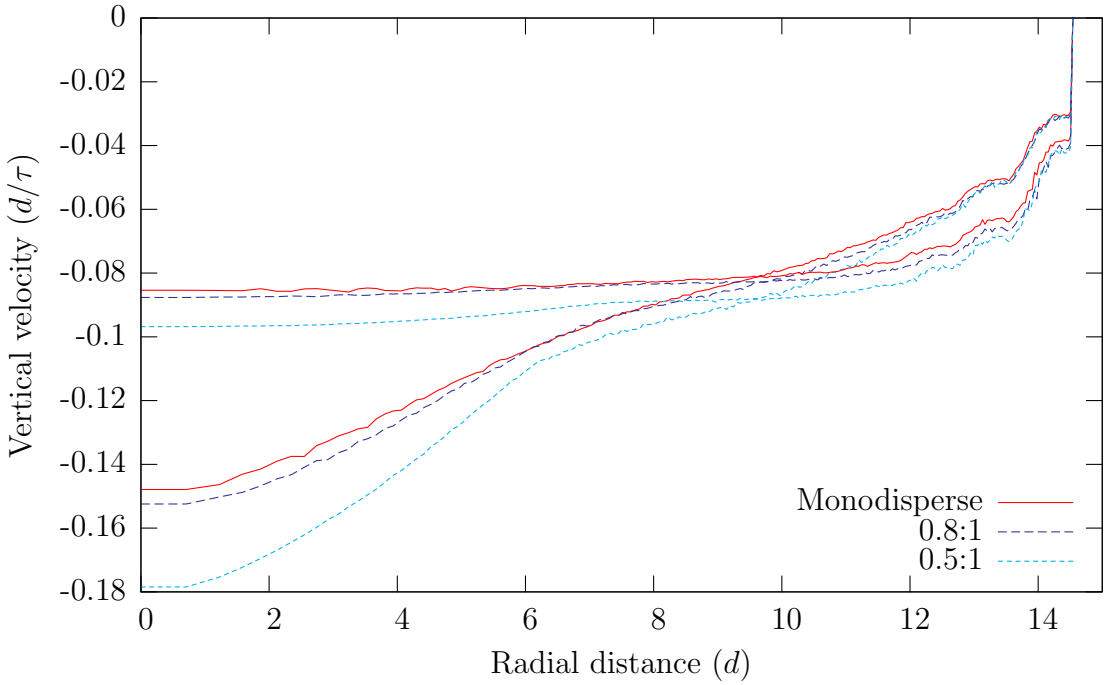


Figure 5-22: Comparison of velocity profiles for the three bidisperse simulations. The three flatter curves are calculated at $z = 30d$ in the plug-like flow region while the other three were taken at $z = 22d$ in the parabolic flow region.

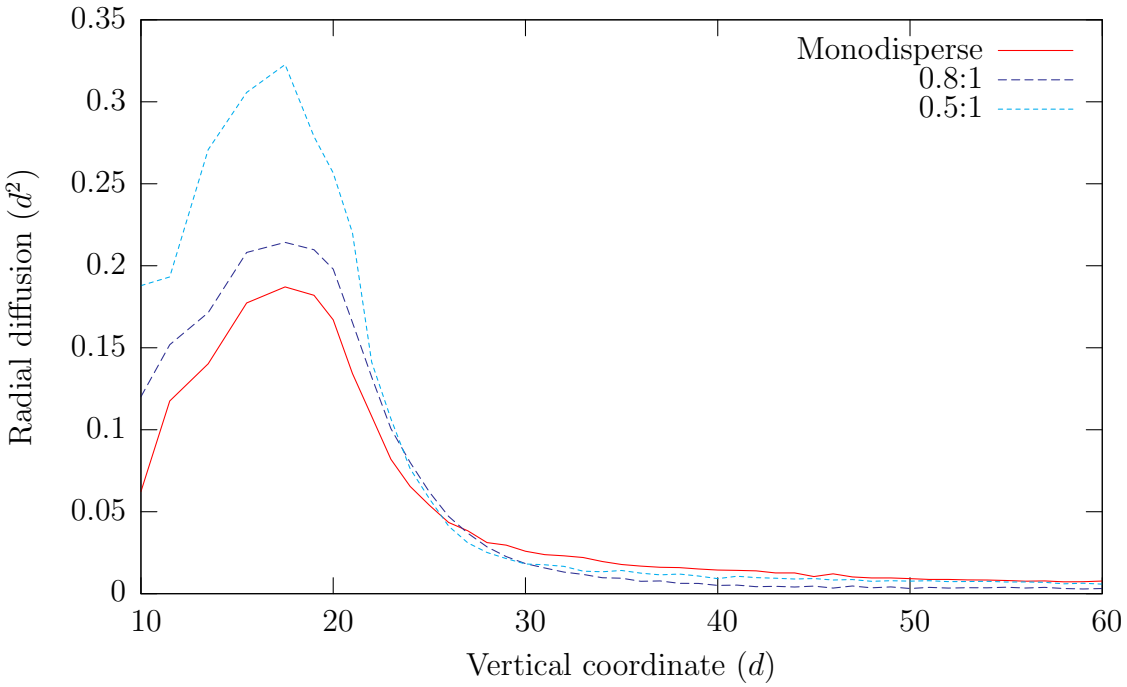


Figure 5-23: Comparison of particle diffusion for the three bidisperse simulations.

the densely packed fuel pebbles. An in-depth study of this phenomenon remains a subject of future work. For now, we can safely recommend a diameter ratio of 0.5:1, which reduces the dynamic central column's permeability by a factor of four without introducing any significant diffusion of moderator pebbles into the fuel annulus.

5.9 Conclusions

5.9.1 Pebble-Bed Reactor Core Design

Using DEM simulations, we have analyzed many aspects of granular flow in pebble-bed reactor cores of direct relevance for design and testing. We close by summarizing some key conclusions.

The mean flow profile exhibits a smooth transition from a nearly uniform plug flow in the upper cylindrical region to a nonuniform, converging flow in the lower funnel region, consistent with recent experiments [27, 63]. There are no stagnant regions in the 30° and 60° conical funnels considered in this study, although the flow is slower near the corner at the top of the funnel, especially in the former case. Moreover, the wider 30° funnel has a boundary mono-layer of slower pebbles partially crystallized on the wall.

The only available continuum theory for such flows, the simple Kinematic Model [76, 90, 95, 94], gives a reasonable qualitative picture of the flow profiles, although it cannot capture discrete boundary-layer effects. As in other experiments on similar geometries [27], the Kinematic Model does not quantitatively predict the dependence of the flow profile on geometry. We suggest that it be used to get a rough sense of the flow profile for a given core geometry prior to (much more computationally expensive) DEM simulations and/or experiments.

We have quantified the degree of pebble mixing in the core. Although there is some horizontal diffusion in the funnel region, pebbles depart from the streamlines of the mean flow by less than one pebble diameter prior to exiting the core.

We have demonstrated that the "mixing layer" between the central moderator

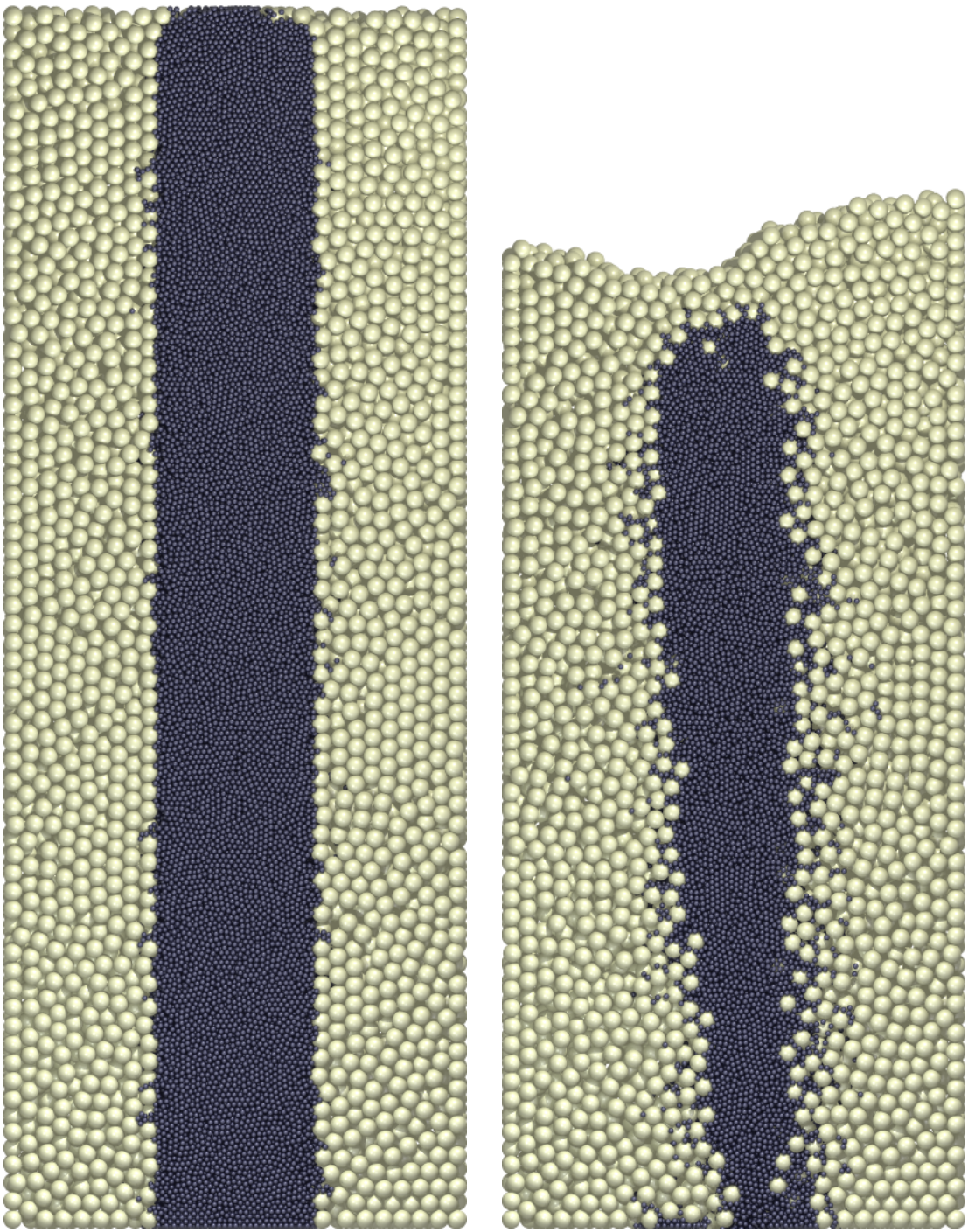


Figure 5-24: Simulation snapshots of a bidisperse drainage experiment with size ratio 0.3:1 in a rectangular silo of width $25d$ and depth $4d$. The left image is taken before drainage taken place, and even at this stage, some of the smaller particles have penetrated into the packing of larger particles. During drainage (right) some of the small particles penetrate deep into the side packings. Small particles are also visible on the container base, having fallen through the gaps between the large particles.

column and the outer fuel annulus, which appears in prior models [51], can be reduced to the thickness of one pebble diameter by separating moderator and fuel pebbles with a guide ring at the ceiling (to eliminate mixing by surface avalanches), consistent with experiments on MPBR models [63]. We conclude that the dynamic central column of moderator pebbles is a sound concept, which should not concern regulators.

We have constructed Voronoi tessellations of our flowing packings to measure the profile of volume fraction (or porosity) and found some unexpected features which would affect coolant gas flow through the core. The bulk of the core, in the plug-flow region of the upper cylinder, has a volume fraction near the jamming point (63%), but there is a sharp transition to less dense packings ($55-60\%$) in the funnel region, due to shear dilation. We also observe lower volume fractions in this range at the moderator/fuel interface in the upper cylinder, below the guide ring, and lower volume fractions ($50-55\%$) against the walls. These narrow regions of increased porosity (and thus, increased permeability) would allow faster helium gas flow.

We have also studied local ordering in the flowing packings and find evidence for partial crystallization within several pebble diameters of the walls, consistent with previous experiments [50, 119] and simulations [38, 103]. Such ordering on the walls of the guide ring, then advected down through the core, is responsible for the increased porosity of the moderator/fuel interface.

We have varied the wall friction in our DEM simulations and observe that it can affect the mean flow, even deep into the bulk. Reducing the wall friction increases radial ordering near the walls and makes the flow profile more uniform.

Since diffusion is minimal, the probability distribution of pebble residence times is dominated by advection in the mean flow. Therefore, we have made predictions using the Kinematic Model, numerically for the conical-funnel reactor geometries, and analytically for a wide parabolic funnel. The model predicts a fat-tailed ($\sim 1/t$) decay of the residence-time density due to hydrodynamic dispersion in the funnel region.

Our DEM simulations predict that the 60° conical funnel results in a narrower residence-time distribution than the 30° funnel, which has more hydrodynamic dis-

persion. The steeper 60° funnel also exhibits a boundary layer of slower, partially crystallized pebbles near the wall which lead to an anomalous bump far in the tail of residence-time distribution. These results have important implications for non-uniformity in the burnup of fuel pebbles.

We have introduced the concept of a bidisperse core with smaller moderator pebbles in the dynamic central column than in the outer fuel annulus, in order to focus the helium gas flow on the fuel. Our DEM simulations demonstrate that there is negligible pebble mixing at the interface for diameter ratios as small as 0.5:1, for which the permeability of the moderator column is reduced by a factor of four. We conclude that the bidisperse MPBR design is sound and will produce a stable moderator-pebble column of greatly reduced gas permeability.

A natural next step would be to combine our full-scale DEM model for the pebble flow with existing computational approaches to reactor core physics [131, 51], which rely on pebble flow as an empirical input. More accurate studies of gas flow in the core could also be done, starting from our complete pebble packings, or the average quantities such as the porosity. With such computational tools, one should be able to reliably test and develop new reactor designs.

5.9.2 Basic Physics of Dense Granular Flow

We have noted a number of favorable comparisons between our simulations and experiments in similar geometries [63, 27, 50, 119], which provides further validation of the Discrete-Element Method as a realistic means of simulating granular materials. As such, it is interesting to consider various implications of our results for the theories of dense granular flow, since the simulations probe the system at a level of detail not easily attained in experiments.

Our conclusions about the Kinematic Model are similar to those of a recent experimental study [27]: The model describes the basic shape of the flow field in the converging region, but fails to predict the nearly uniform plug flow in the upper region with vertical walls or the precise dependence on the funnel geometry. It also cannot describe boundary-layers due to partial crystallization near walls or incorporate wall

friction, which we have shown to influence the entire flow profile.

On the other hand, there is no other continuum model available for dense silo drainage, except for Mohr-Coulomb plasticity solutions for special 2D geometries, such as a straight 2D wedge without any corners [94], so it is worth trying to understand the relative success of the Kinematic Model for our 3D reactor geometries and how it might be improved. A cooperative microscopic mechanism for random-packing dynamics, based on "spots" of diffusing free volume, has recently been proposed, which yields the mean flow of the Kinematic Model as the special case of independent spot random walks with uniform upward drift from the orifice (due to gravity) [17]. Under the same assumptions, the spot model has also been shown to produce rather realistic simulations of flowing packings in wide silos (compared to DEM simulations) [112], where the Kinematic Model is known to perform well [137, 84, 114, 28]. This suggests that some modification of the spot dynamics, such as spot interactions and/or nonuniform properties coupled to mechanical stresses, and an associated modification of the Kinematic Model in the continuum limit, may be possible to better describe general situations.

From a fundamental point of view, perhaps the most interesting result is the profile of Voronoi volume fraction (or porosity) in our flowing random packings in Figure 5-8. Although the mean velocity in Figure 5-2 shows a fairly smooth transition from the upper plug flow to the lower converging flow, the volume fraction reveals a sharp transition (at the scale of $1-3$ particles) from nearly jammed "solid" material in the upper region (63%) to dilated, sheared "liquid" material in the lower region (57-60%). The transition line emanated from the corners between the upper cylinder and the conical funnel. We are not aware of any theory to predict the shape (or existence) of this line, although it is reminiscent of a "shock" in the hyperbolic equations of 2D Mohr-Coulomb plasticity [94].

Our measurements of diffusion and mixing provide some insights into statistical fluctuations far from equilibrium. Consistent with the experiments in wide quasi-2D silos [28], we find that diffusion is well described geometrically as a function of the distance dropped, not time (as in the case of thermal molecular diffusion). As a clear

demonstration, there is essentially no diffusion as pebbles pass through the upper core, until they cross the transition to the funnel region, where the diffusion remains small (at the scale of one pebble diameter) and cooperative in nature. The behavior in the funnel is consistent with the basic spot model [17], but a substantial generalization would be needed to describe the transition to the upper region of solid-like plug flow, perhaps using concepts from plasticity theory [65].

We view silo drainage as a fundamental unsolved problem, as interesting and important as shear flow, which has received much more attention in physics. The challenge will be to find a single theory which can describe both shear cells and draining silos. Our results for pebble-bed reactor geometries may provide some useful clues.

Chapter 6

Testing the Stochastic Flow Rule[1]

6.1 Introduction

In chapters 3 and 4 it was shown that the spot model simulation could reproduce many features of granular drainage to a high degree of accuracy. However, a key drawback is that it appears difficult to generalize to other forms of granular flow. While the concept of a local relaxation appears general, the concept of a diffusing free volume appears to be very specific to granular drainage, where one can make an easy identification with particles exiting, and free volume being injected. In other situations, such as a shear cell, or plate dragging, the motion of free volume appears a lot less clear.

It also seems difficult to generalize the specific random walk process that was used. While the approximately-gaussian velocity profiles in granular drainage, or the error function profiles seen in shear zone experiments [45] seem to warrant explanation by a diffusing object, other types of granular flows show different functional forms which are less associated with diffusion. In Couette cell experiments, velocity profiles have been shown to be exponential, while on inclined planes the velocity profiles show a polynomial dependence, and it appears more difficult to explain these quantities in terms of a diffusing object. Even in granular drainage, the results of the previ-

[1]This chapter is based on reference [66], *The Stochastic Flow Rule: A Multi-Scale Model for Granular Plasticity*, published in Modelling and Simulation in Materials Science and Engineering, 2007. See `http://www.iop.org/journals/msmse/` for more details.

ous two chapters suggest that the spot model random walk process may be a large oversimplification.

One of the attractions of the spot model for granular drainage is its motivation from mainly geometrical reasoning. However, it seems that a general model for granular flow must ultimately be grounded in mechanical concepts. Formulating a mechanical model for dense granular flow has an extremely long history, dating back to Coulomb, who proposed the "Ideal Coulomb Material": an idealized infinitesimal element of material that would fail under certain stress conditions. Using this microscopic picture, Mohr-Coulomb plasticity theory was developed [94, 118]. This theory has become widely-accepted, but cannot be considered at a complete theory of granular materials, since it fails to provide an accurate prediction of many granular flows, and it is also numerically ill-posed, sometimes leading to shocks – this is discussed in more detail in the following section.

While physical ideas should play a role in a general theory, the results of the previous sections suggest that particle discreteness plays an integral role in granular flow. Perhaps Mohr-Coulomb plasticity's largest failure is that it assumes a continuum throughout. Based on these ideas, Kamrin and Bazant formulated the Stochastic Flow Rule (SFR) [65], attempting to rectify traditional plasticity theories by accounting for the particle discreteness. The SFR provides a physical basis for the spot model simulations considered in previous chapters, and predicts several of the free parameters that were originally calibrated in chapter 3. In addition, it also correctly predicts the exponential flow profile seen in the annular Couette cell, and at the time of writing, we are not aware of any other model, continuum or discrete, which can describe both of these cases, even qualitatively.

In this chapter, the continuum predictions of the SFR are tested. In section 6.4 the velocity profiles for a wide silo and an annular Couette cell are computed, and in section 6.5 these are directly compared with DEM simulations in these two geometries.

6.2 Continuum theories for two dimensional stress

Coulomb proposed the "Ideal Coulomb Element" as a representation of two dimensional infinitesimal element of a dry cohesionless granular material. The element is subject to a shear stress τ and a normal stress σ. The material is rigid (perfectly plastic) until failure occurs when

$$|\tau/\sigma| > \mu, \tag{6.1}$$

where μ is an internal friction coefficient which is a constant for the material. It is also convenient to define a friction angle $\phi = \tan^{-1}\mu$.

From this microscopic picture of an element, we would like to form a continuum model, by using momentum balance. However, to get a closed system of equations, additional assumptions are needed. An obvious problem stems from equation 6.1 being an inequality, corresponding to the fact granular materials have a memory, and may have internal state variables that were set by their method of preparation.

To make progress, we consider the Critical State Theory of soils [118]. In this theory, the stresses in a soil during flow converge on a "critical state line" in which the pressure and the deviatoric stress tensor $\mathbf{D_0} = \mathbf{D} - \mathbf{I}(\operatorname{tr}\mathbf{D})/3$ are linearly related. This Drucker-Prager yield criterion is similar to the three dimensional analog of the Mohr-Coulomb yield function. Thus, if we wish to compute stresses in a flowing environment, we assume the *Mohr-Coulomb incipient yield hypothesis*, that $|\tau/\sigma| = \mu$ everywhere.

Let us focus entirely on 2D geometries (plane strain) complete with a 2D stress tensor $\mathbf{T}$ defined in the plane of the flow. By requiring the yield criterion to be met at all points, the limit-state assumption implies the following constraint: $\frac{1}{\sqrt{2}}|\mathbf{T}_0| = \sin\phi\left(\frac{1}{2}\operatorname{tr}\mathbf{T}\right)$, where $\mathbf{T}_0$ is the deviatoric stress tensor and $|\mathbf{A}| = \sqrt{\mathbf{A}:\mathbf{A}}$. A simple way to uphold this constraint is to rewrite the stress field in terms of two stress parameters (the Sokolovskii variables [127]): the average pressure p, and the "stress-angle" ψ denoting the angle from the horizontal to the major principal stress. The

components of the 2D stress tensor are then

$$\begin{aligned} T_{xx} &= -p(1+\sin\phi\cos 2\psi) && (6.2)\\ T_{yy} &= -p(1-\sin\phi\cos 2\psi), && (6.3)\\ T_{xy}=T_{yx} &= -p\sin\phi\sin 2\psi && (6.4) \end{aligned}$$

from which it can be seen that $p = -\operatorname{tr}\mathbf{T}/2$ and $\tan 2\psi = 2T_{12}/(T_{11}-T_{22})$. The convection-free 2D momentum balance law $\boldsymbol{\nabla}\cdot\mathbf{T}+\mathbf{F}_{\text{body}}=\mathbf{0}$ then reduces to the two variable system of hyperbolic PDEs

$$\begin{aligned} F^x_{\text{body}} &= (1+\sin\phi\cos 2\psi)p_x - 2p\sin\phi\sin 2\psi\ \psi_x \\ &\quad +\sin\phi\sin 2\psi\ p_y + 2p\sin\phi\cos 2\psi\ \psi_y && (6.5)\\ F^y_{\text{body}} &= \sin\phi\sin 2\psi\ p_x + 2p\sin\phi\cos 2\psi\ \psi_x \\ &\quad +(1-\sin\phi\cos 2\psi)p_y + 2p\sin\phi\sin 2\psi\ \psi_y. && (6.6) \end{aligned}$$

The directions along which the yield criterion is met, the slip-lines, form at the angles $\psi\pm\epsilon$ from the horizontal where $\epsilon=\pi/4-\phi/2$.

To compute a velocity field, we need to specify a flow rule, which relates the deformation rate tensor $\mathbf{D}$ to the stress tensor, specifying how the material will flow during failure. A common assumption here is *coaxiality* which assumes that the eigenvectors of $\mathbf{T}$ and $\mathbf{D}$ are aligned. This assumption comes from a belief that the material is isotropic (even at the local scale), and thus if the material is pushed on a certain set of axes, the response will also be aligned with those axes. In this situation, it says that when a material element fails, it fails equally along both slip lines at the same time.

Unfortunately, the resulting system of equations for $\mathbf{T}$ is extremely difficult to solve. In many situations, and even in some simple geometries, the equations predict shocks [94], which require sophisticated mathematical techniques to solve [52, 53, 54]. Given the large debate about forces in granular packings at the microscopic level, the presence of shocks in this theory may not be fatal, and perhaps these shocks

characterize a physical concept, such as particle arching. However, what is perhaps more troubling is that with the assumption of a coaxial flow rule, these theories would predict discontinuities in the flow field also, and from all experiments and simulations considered in this thesis, granular flows have always exhibited smooth flow fields.

6.3 The Stochastic Flow Rule

The Stochastic Flow Rule proposed by Kamrin and Bazant [65, 66] attempts to resolve the above problems by assuming continuum stresses, but proposing that the flow made up of the superposition of many discrete plastic flow events. They remove the assumption of coaxiality, suggesting that it does not make sense for a discrete chunk of granular material to deform along both slip planes. Instead, they suggest that at each instant, the material will pick one slip plane randomly. Motivated by geometrical considerations, they suggest that the size for these plastic deformation events happens on the scale of spot.

Thus, they postulate that to generalize the spot model to an arbitrary geometry, one can first calculate the Mohr-Coulomb stresses, and then generate flow by imagining that spots carry out random walks on the lattice of Mohr-Coulomb slip lines. Since the flow is generated by the spots which are a diffusing quantity, the predicted flow field will be much smoother than that predicted by Mohr-Coulomb plasticity, with discontinuities blurred out by diffusion.

To model this mathematically, they assume that the spot concentration follows a Fokker-Planck equation

$$\frac{\partial \rho_s}{\partial t} + \boldsymbol{\nabla} \cdot (\mathbf{D}_1 \rho_s) = (\boldsymbol{\nabla} \otimes \boldsymbol{\nabla}) : (\mathbf{D}_2 \rho_s). \tag{6.7}$$

In steady-state flow, this simplifies to the time-independent drift-diffusion equation,

$$\boldsymbol{\nabla} \cdot (\mathbf{D}_1 \rho_s) = (\boldsymbol{\nabla} \otimes \boldsymbol{\nabla}) : (\mathbf{D}_2 \rho_s) \tag{6.8}$$

where $\mathbf{A} : \mathbf{B} = A_{ij} B_{ij}$. Once solved, the mean particle velocity field $\mathbf{u}$ can be found

by superposing the effects of all spots on all the particles. As discussed in previous chapters, this can be calculated by convolving the spot influence function $w(\mathbf{r}, \mathbf{r}')$ with the negative spot flux vector

$$\mathbf{u}^* = -\mathbf{D}_1\rho_s + \boldsymbol{\nabla} \cdot (\mathbf{D}_2\rho_s). \tag{6.9}$$

Thus the particle velocity $\mathbf{u}$ is

$$\mathbf{u} = \int w(\mathbf{r}, \mathbf{r}')\mathbf{u}^*(\mathbf{r}')d\mathbf{r}'. \tag{6.10}$$

The influence function, which in simple language describes the spot's shape, should have a characteristic width of three to five particle diameters, so that spots match known correlation length data. In many situations, $\mathbf{u}^*$ is close to $\mathbf{u}$ since the convolution with w tends only to smooth out sharp changes in the spot flux density.

The remaining question is to derive the spot drift $\mathbf{D}_1$ and diffusion $\mathbf{D}_2$ from mechanical principles from the underlying stress field. To begin, several assumptions are made, such as proposing that the diffusion is isotropic so that $\mathbf{D}_2 = D_2\mathbf{I}$, and that all length scales are set to the spot length scale, so that $|\mathbf{D}_1| = L_s/\Delta t$, and $D_2 = L_s^2/2\Delta t$. All that remains is to find the direction of the spot drift, which can be calculated from looking at local stress imbalances; for a full treatment, the reader should refer to [65] and [66].

6.4 Solutions for the flow in two simple geometries

6.4.1 Wide silo

For the wide, 2D silo with smooth walls and a flat bottom, the stress balance equations can be solved analytically, giving

$$\psi = \frac{\pi}{2}, \qquad p = \frac{f_g(z_m - z)}{1 + \sin\phi} \tag{6.11}$$

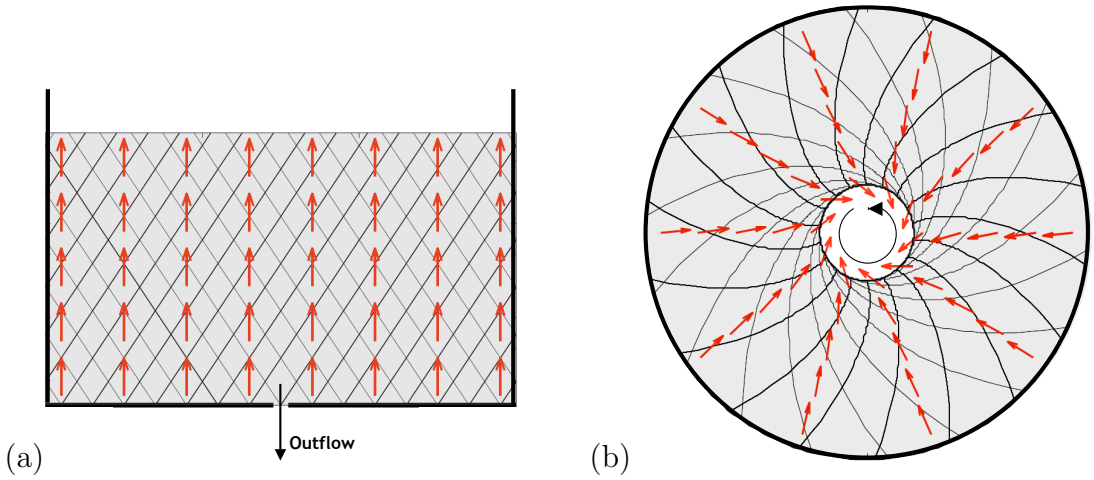


Figure 6-1: Slip-line fields (from Mohr-Coulomb Plasticity) and the spot drift field (from the SFR) displayed for (a) a wide silo draining under gravity, and (b) shearing in an annular Couette cell (no gravity).

where z is the distance from the silo bottom, z_m is the distance from the bottom to the free surface, and f_g is the material's weight density. As described in [66] this results in a regular lattice of diagonal slip lines, with a spot drift vector pointing uniformly upwards. Thus, we recover a spot density which satisfies a diffusion equation

$$\frac{\partial \rho_s}{\partial z} = \frac{L_s}{2} \boldsymbol{\nabla}^2 \rho_s. \tag{6.12}$$

as in the formulation in the previous chapters, and for a point orifice we obtain

$$\rho_s \approx \frac{\exp(-x^2/4bz)}{\sqrt{4\pi bz}} \tag{6.13}$$

where the diffusion parameter is given by $b = L_s/2$. This matches the previous analysis, but here the diffusion parameter no longer needs to be fitted; it is given directly by the spot length. Using the typical range of spot lengths to be $3d < L_s < 5d$, the typical range of b values is predicted to be $1.5d$ to $2.5d$, which compares very well to the results of previous chapters. In the literature [95, 114, 27, 137, 84], granular drainage experiments have predicted a range for b between $1.3d$ and $3.5d$, in good agreement with the prediction.

6.4.2 Annular Couette cell

In annular Couette flow, material fills the region between two rough cylinders and is sheared by rotating the inner cylinder while holding the outer stationary. To solve for the stresses, we first convert the stress balance equations to polar coordinates (r, θ) and require that p and ψ obey radial symmetry. This simplifies to

$$\frac{\partial \psi^*}{\partial r} = -\frac{\sin 2\psi^*}{r(\cos 2\psi^* + \sin \phi)}, \qquad \frac{\partial \eta}{\partial r} = -\frac{2 \sin \phi}{r(\cos 2\psi^* + \sin \phi)} \tag{6.14}$$

where $\eta = \log p$ and $\psi^* = \psi + \frac{\pi}{2} - \theta$. As described in [66], this must be solved numerically using fully rough inner wall boundary conditions. As shown in figure 6-1 this gives a drift field that points inward, but gradually opposes the motion of the inner wall. The competition between diffusion, and an inwards-point diffusion leads to an approximately exponential velocity profile.

To find the velocity field $\mathbf{u}$ requires convolving the solution for spot drift $\mathbf{u}^*$ with the spot influence function. To obtain a solution close to the walls requires a specification of how spots interact with the wall, and we consider two separate hypotheses:

1. Assume the $\mathbf{u}^*$ equals the wall velocity wherever spots overlap with the wall.

2. View the region inside the wall as a bath of non-diffusive spots which cause particles to move in a manner which mimics the rigid wall motion.

The second hypothesis creates velocity profiles which decay faster. Regardless of which hypothesis is chosen, the bulk behavior remains the same.

6.5 Comparing SFR Predictions to DEM Simulations

6.5.1 The silo geometry

We considered DEM simulations of a quasi-2D silo with plane walls at $x = \pm 75d, z = 0$ with friction coefficient $\mu_w = 0.5$, and made the y direction periodic with width $8d$. To generate an initial packing, 90,000 spherical particles with contact friction coefficient $\mu_c = 0.5$ were poured in from a height of $z = 130d$ at a rate of $378\tau^{-1}$. After all particles are poured in at $t = 238\tau$, the simulation is run for an additional 112τ in order for the particles to settle. After this process has taken place, the particles in the silo come to a height of approximately $z = 62.2d$. To initiate drainage, an orifice in the base of the container is opened up over the range $-3d < x < 3d$, and the particles are allowed to fall out under gravity; figure 6-2 shows a typical simulation snapshot during drainage.

We collected 282 snapshots every 2τ, and made use of this information to construct velocity cross sections. A particle with coordinates $\mathbf{x}_n$ at the nth timestep and $\mathbf{x}_{n+1}$ at the $(n+1)$th timestep makes a velocity contribution of $(\mathbf{x}_{n+1} - \mathbf{x}_n)/\Delta t$ at the point with coordinates $(\mathbf{x}_n + \mathbf{x}_{n+1})/2$. This data can then be appropriately binned to create a velocity profile; we considered bins of size d in the x direction, and created velocity profiles for different vertical slices $|z - z_s| < d/2$.

Since the SFR makes predictions about the velocity profile during steady flow, we choose a time interval $t_1 < t < t_2$ over which the velocity field is approximately constant. Choosing this interval requires some care, since if t_1 is too small then initial transients in the velocity profile can have an effect, and if t_2 is too large, then the free surface can have an influence. For the results reported here, we chose $t_1 = 120\tau$ and $t_2 = 200\tau$.

Figure 6-3 compares the SFR predictions for this environment to the DEM simulation. The displayed simulation data uses a particle contact friction of $\mu_c = 0.3$. Since the typical range of L_s from velocity correlations in simulations [112] and exper-

iments [65] is $3d$ to $5d$, we choose $L_s = 4d$ to generate the approximate SFR solution. We emphasize that this parameter is not fitted. In this geometry, the slip-lines are symmetric about the drift direction causing both $\mathbf{D}_1$ and D_2 to become independent of the internal friction. In prior simulations we have found that particle contact friction has some effect on the flow [113], and analogously the internal friction should play some role in the determination of b. Here, the loss of friction dependence comes from our simplification that $\mathbf{D}_2$ is isotropic. A less simple but more precise definition for $\mathbf{D}_2$ would anisotropically skew the spot diffusion tensor as a function of internal friction: the slip-lines, which we model as the directions along which a spot can move (roughly), intersect at a wider angle as internal friction is increased.

Even so, our simple model captures many of the features of the flow and accurately portrays the dominant behavior. The downward velocity, especially at $z = 10d$, strongly matches the predicted Gaussian. Perhaps a more global demonstration of the underlying stochastic behavior in the SFR is evident in Figure 6-4, where a linear relationship can be seen between the mean square width of v_z and the height, indicating that the system variables are undergoing a type of diffusive scaling. The SFR solution also predicts this linear relationship and, in particular, that the slope should equal $2b = L_s$. The agreement shown in Figure 6-4 for such a typical L_s value is a strong indicator that the role of the correlation length in the flow has been properly accounted for in the SFR.

6.5.2 The Couette geometry

For the Couette geometry, we considered five different interparticle friction coefficients, $\mu_c = 0.1, 0.3, 0.5, 0.7, 0.9$, and for each value an initial packing was generated using a process similar to that for the silo. We consider a large cylindrical container with a side wall at $r = 64d$ with friction coefficient $\mu_w = \mu_c$, and a base at $z = 0$ with friction coefficient $\mu_w = 0$. For each simulation, 160,000 particles are poured in from a height of $z = 48d$ at a rate of $4,848\tau^{-1}$. After all particles are introduced at $t = 33\tau$, the simulation is run for an additional time period of 322τ to allow the particles to settle. After this process has taken place, the initial packings are approx-

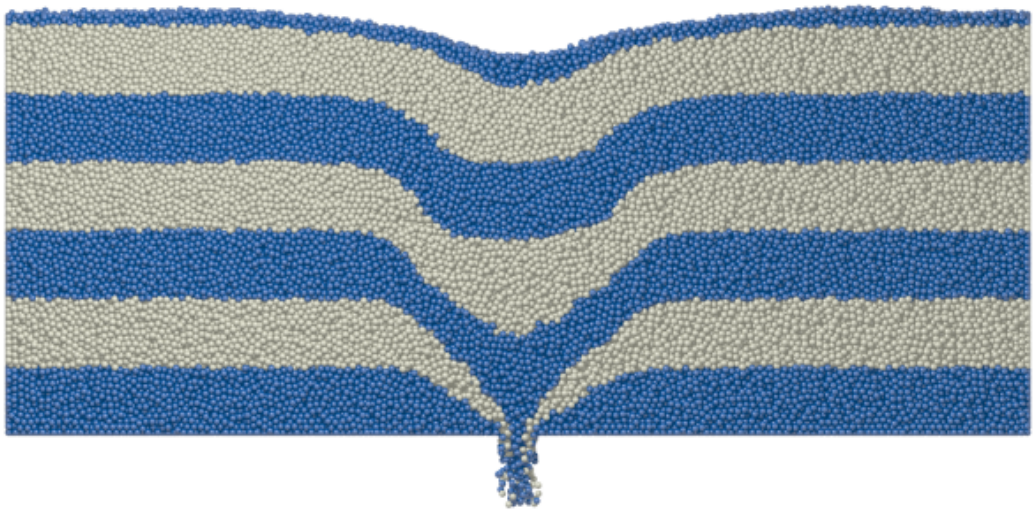


Figure 6-2: A typical snapshot of the silo system during drainage, taken at $t = 60\tau$. The colored bands are initially spaced $10d$ apart, and highlight the deformations that occur during flow.

imately $11.5d$ thick. Packings with $\mu_c = 0.9$ are approximately 2% thicker than those with $\mu_c = 0.1$.

The shearing simulations take place in an annulus between two radii, r_{in} and r_{out}. A rough outer wall is created by freezing all particles which have $r > r_{\text{out}}$; any forces or torques that these particles experience are zeroed during each integration step. Similarly, all particles between $r_{\text{in}} - 4d$ and r_{in} are forced to rotate with a constant angular velocity ω around $r = 0$. Particles with $r < r_{\text{in}} - 4d$ are deleted from the simulation and an extra cylindrical wall with friction coefficient $\mu_w = \mu_c$ is introduced at $r = r_{\text{in}} - 4d$ to prevent stray surface particles from falling out of the shearing region. A typical run is shown in Figure 6-5.

These simulations of the Couette geometry caused some problems with finding an efficient load-balancing scheme for the parallelization. In the silo geometries considered in this thesis, the particles are usually equally distributed in a box-shaped region of space, so when the simulation is divided into a rectangular grid of subdomains, each processor gets approximately the same number of particles, leading to good load-balancing. However, if an annular geometry is subdivided into rectangular grid of subdomains, some the processors may have many more particles than others,

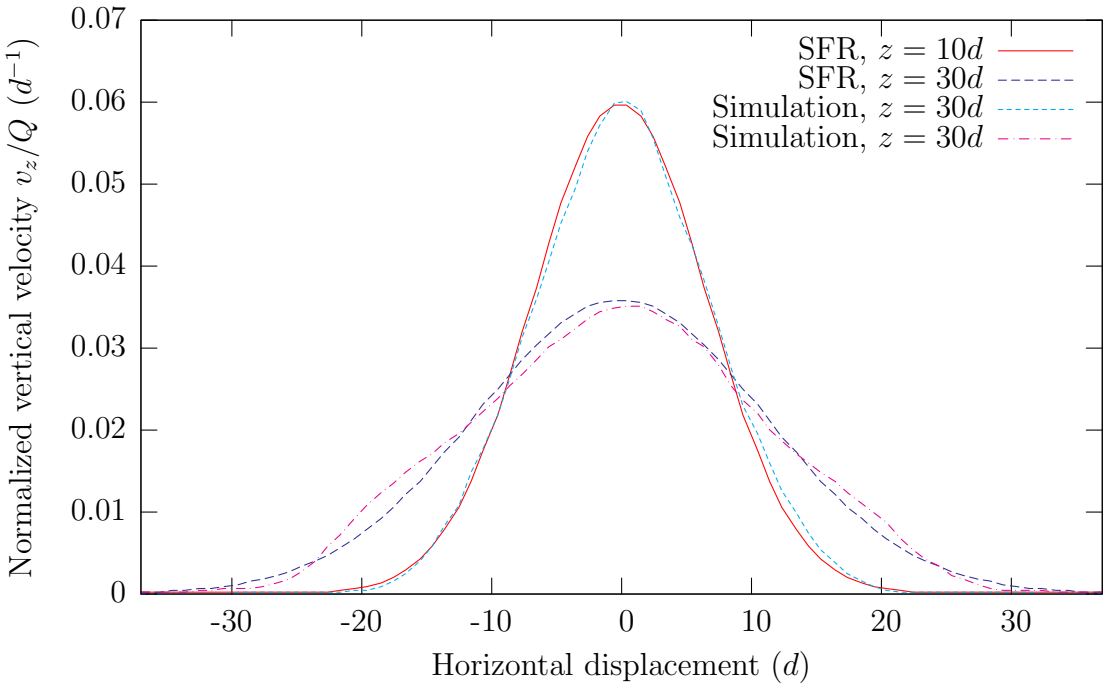


Figure 6-3: Mean downward velocity profiles at two different heights in a wide silo. DEM results are plotted against the SFR predictions at heights of $z = 10d, 30d$.

and since the speed of the code is determined by the speed of the slowest node, this is undesirable. We therefore exploited the radial symmetry of the geometry, and made a 2×2 grid of processors with boundaries at $x = y = 0$, so that each computational subdomain contains approximately a quarter of the total particles. The twenty nodes of the AMCL were then utilised by running five different simulations concurrently.

From the five initial packings, we carried out eight different shear cell simulations to investigate the effects of friction, angular velocity, and inner wall radius. To investigate the effect of friction, five runs were carried out with $\omega = 0.01\tau^{-1}$ and $r_{\text{in}} = 40d$, for $\mu_c = 0.1, 0.3, 0.5, 0.7, 0.9$. To examine the effect of the inner wall radius, an additional run with $r_{\text{in}} = 30d$ and $r_{\text{out}} = 50d$ was carried out, with $\mu_c = 0.3$ and $\omega = 0.01\tau^{-1}$ kept constant. To look at the effect of angular velocity, a further two runs with $\omega = 0.05\tau^{-1}$ and $\omega = 0.2\tau^{-1}$ were carried out, with $\mu_c = 0.5$ and $r_{\text{in}} = 40d$ kept constant. For each simulation, we collected 561 snapshots. For the runs where

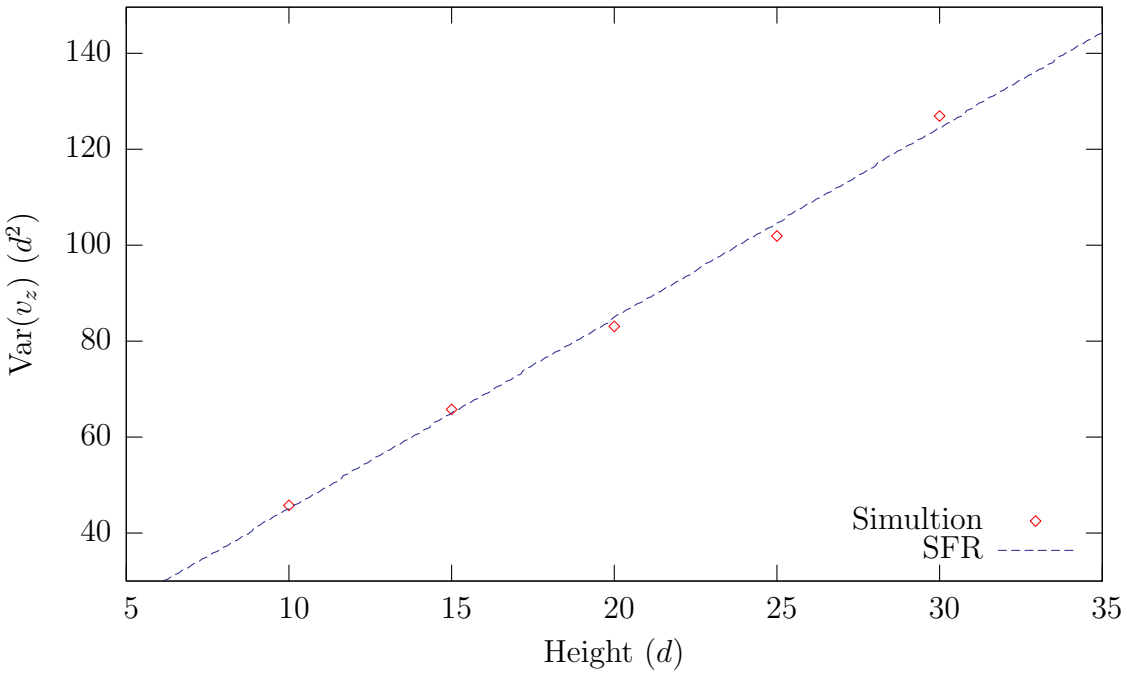


Figure 6-4: Mean square width of the downward velocity across horizontal cross-sections in the DEM simulation, compared to SFR predictions.

$r_{\text{in}} = 40d$, approximately $108,350$ particles were simulated, corresponding to 2.0Gb of data. For the run with $r_{\text{in}} = 30d$, $88,657$ particles were simulated, corresponding to 1.7Gb of data.

The simulation results show a good empirical agreement with previous experimental work on shear cells [80, 87, 72, 21, 88]. In all cases, we see an angular velocity profile that falls off exponentially from the inner cylinder, with a half-width on the order of several particle diameters. Near the inner wall, the flow deviates from exponential, which is an effect seen in some prior studies but is more dramatic here. Following methods similar to those used in silo simulation, we used the snapshots to construct an angular velocity profile. We used bins of size $d/2$ in the radial direction, and we looked at velocity profiles in different vertical slices $z_{\text{low}} < z < z_{\text{high}}$.

Since the simulation geometry is rotationally symmetric, our angular velocity profile can most generally be a function of r, z and t. Ideally, we hope that ω is

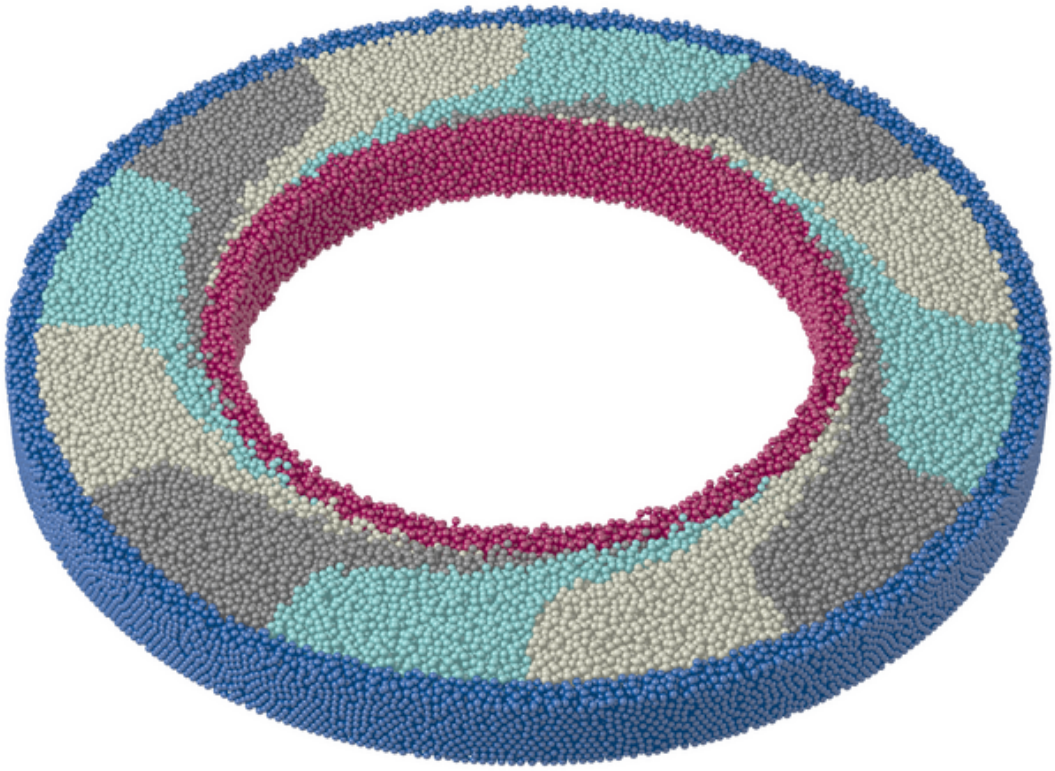


Figure 6-5: A typical snapshot of the annular Couette cell during shearing based on the simulation with $r_{\text{in}} = 40d$, $\omega = 0.01\tau^{-1}$, and $\mu_c = 0.5$. Dark blue particles (with $r > r_{\text{out}}$) are frozen during the simulation, while dark red particles (with $r < r_{\text{in}}$) are rotated with angular velocity ω. The particles between r_{in} and r_{out} undergo shearing. The colored bands of grey, cream, and cyan were initially radial, and in this snapshot, after fifty frames, the deformations can be clearly seen.

primarily a function of r, with only a very weak dependence on z and t, but we began by studying the effects of these other variables. To determine the dependence on time, the velocity profiles were calculated over many different time intervals. As would be expected, the simulation had to be run for small amount of time before the velocity profile would form; this happens on a time scale of roughly 50τ, and the results suggest a longer time is needed for the cases with low friction. However, the data also shows time-dependent effect happening on a longer scale: as the shearing takes place, there is a small but consistent migration of particles away from the rotating wall, which has a small effect on the velocity profile. This effect does eventually appear to saturate, but because of this, we chose to discard the simulation data for $t < 500\tau$ and calculate velocity profiles based on the time window $500\tau < t < 1100\tau$.

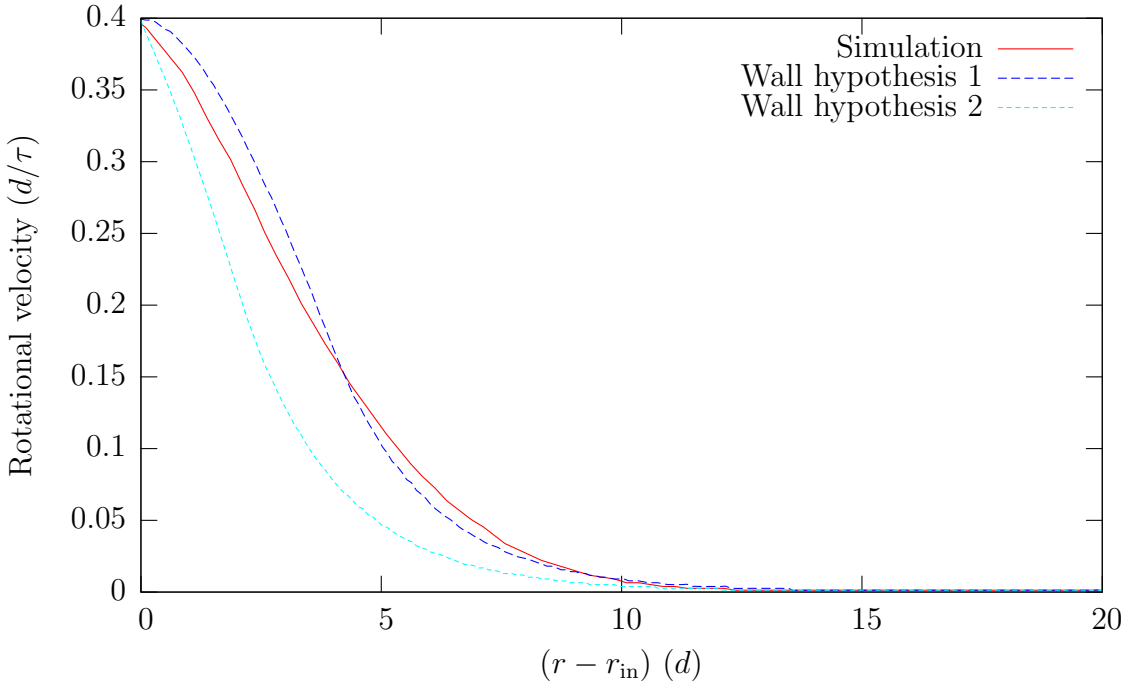


Figure 6-6: SFR solutions for the two different wall hypotheses in the annular Couette geometry ($r_{\text{in}} = 40d, r_{\text{out}} = 60d$) are compared to the simulation results. Both use the same parameters that were used in the silo geometry ($L_s = 4d, \mu_c = 0.3$) and the bead internal friction angle is set at the typical value 25°.

To investigate the angular velocity dependence on the height, we calculated the velocity profiles in five different slices $z_h < z < z_h + d$ for $z_h = d, 3d, 5d, 7d, 9d$. Near the inner rotating wall, the velocity profiles show very little dependence on height. However, in the slow-moving region close to the fixed outer wall, large differences can be seen, with particles in the lowest slice moving approximately 30% slower than those in the central slice, and those in the top slice moving approximate 30% faster. The three central slices show differences of at most 10%, and we therefore chose to use the range $3d < z < 8d$.

The SFR treats the correlation length as a material property independent of the flow geometry or other state variables. To see how well this notion is upheld, we solve the SFR in the annular Couette geometry using the same correlation length ($L_s = 4d$) that was used in the displayed silo prediction, figure 6-4. It is then compared to a simulation of annular flow which uses the same grain properties ($\mu_c = 0.3$).

Results from Figure 6-6 show that regardless of the wall hypothesis, the SFR prediction captures the same qualitative features of the simulation. The SFR and simulation both predict a flatter range near the inner wall, followed by exponential decay. Near the inner wall, it does appear that wall hypothesis 1 ("no slip condition" for spots) gives a closer match to the simulation.

The SFR, when applied to the annular flow geometry, does predict a slight dependence of the flow on the internal friction. We emphasize that internal friction is not the same quantity as particle contact friction μ_c, though we believe if the contact friction is increased, inevitably, the internal friction must be as well. Spherical grains almost always have internal friction angles in the range $\phi = 20^\circ$ to $\phi = 30^\circ$, so to represent this range as best as possible, we simulated flows varying the particle contact friction from $\mu_c = 0.1$ to $\mu_c = 0.9$. Figure 6-7 displays velocity profiles for five different values of friction. In the semi-log format, profiles appear almost linear over the range $45d < r < 58d$ indicating a very good fit to an exponential model of velocity.

Since the curves in the figure are very close, and exhibit some experimental noise, it is difficult to discern any small differences in the widths of the velocity profiles.

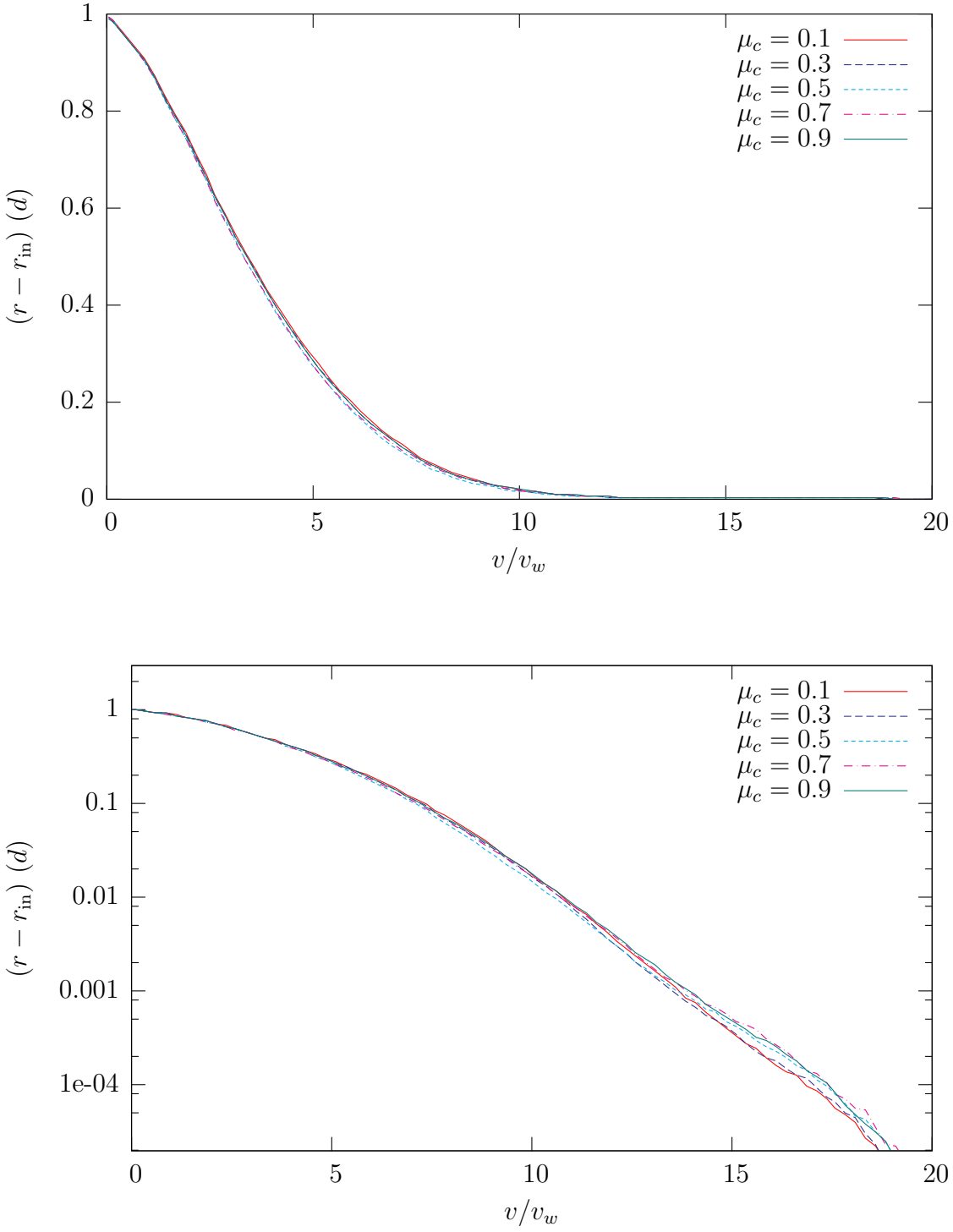


Figure 6-7: Velocity profiles for five different values of μ_c, where $r_{\text{in}} = 40d$ and $\omega = 0.01\tau^{-1}$, plotted on a linear scale (top) and on a semi-logarithmic scale (bottom).

μ_c	b
0.1	$0.974d$
0.3	$0.983d$
0.5	$1.033d$
0.7	$1.046d$
0.9	$1.032d$

ϕ	b
20°	$1.026d$
22°	$1.038d$
24°	$1.052d$
26°	$1.069d$
28°	$1.084d$
30°	$1.102d$

Table 6.1: (Left) Simulation: Half-widths of the shearing velocity profiles for different values of μ_c, calculated by fitting the functional form $f(r) = a - (\log 2)r/b$ to $\log v/v_w$ over the range $45d < r < 58d$. (Right) SFR: Fits the predictions to the same form and uses $L_s = 3d$.

However, table 6.1 shows the results of applying linear regression to extract a half-width for each velocity profile. We see differences on the order of 5%, roughly in line with the SFR. More importantly, the trend of increasing flow width with increasing friction is seen in both.

When the inner wall radius is decreased, Figure 6-8 indicates that the shear band shrinks but the decay behavior in the tail changes only minimally. The SFR predicts this qualitative trend as well, but significantly underestimates the size of the shear-band decrease.

In agreement with past work on Couette flow [80, 21], we too find that the normalized flow profile is roughly unaffected by the shearing rate (see Figure 6-9). As previously discussed, this behavior is in agreement with the SFR, which always permits flow fields to be multiplied by a constant.

6.6 Conclusion

The crucial principles motivating the Stochastic Flow Rule have been presented, and its validity has been assessed by checking analytical predictions for silo and annular Couette flow against discrete-element simulations. Using the same parameters for both cases, without any fitting, the SFR manages good predictions for these two very different flow geometries, which it seems cannot be described, even qualitatively, by any other model. The model was also shown to capture the "diffusive" type flow

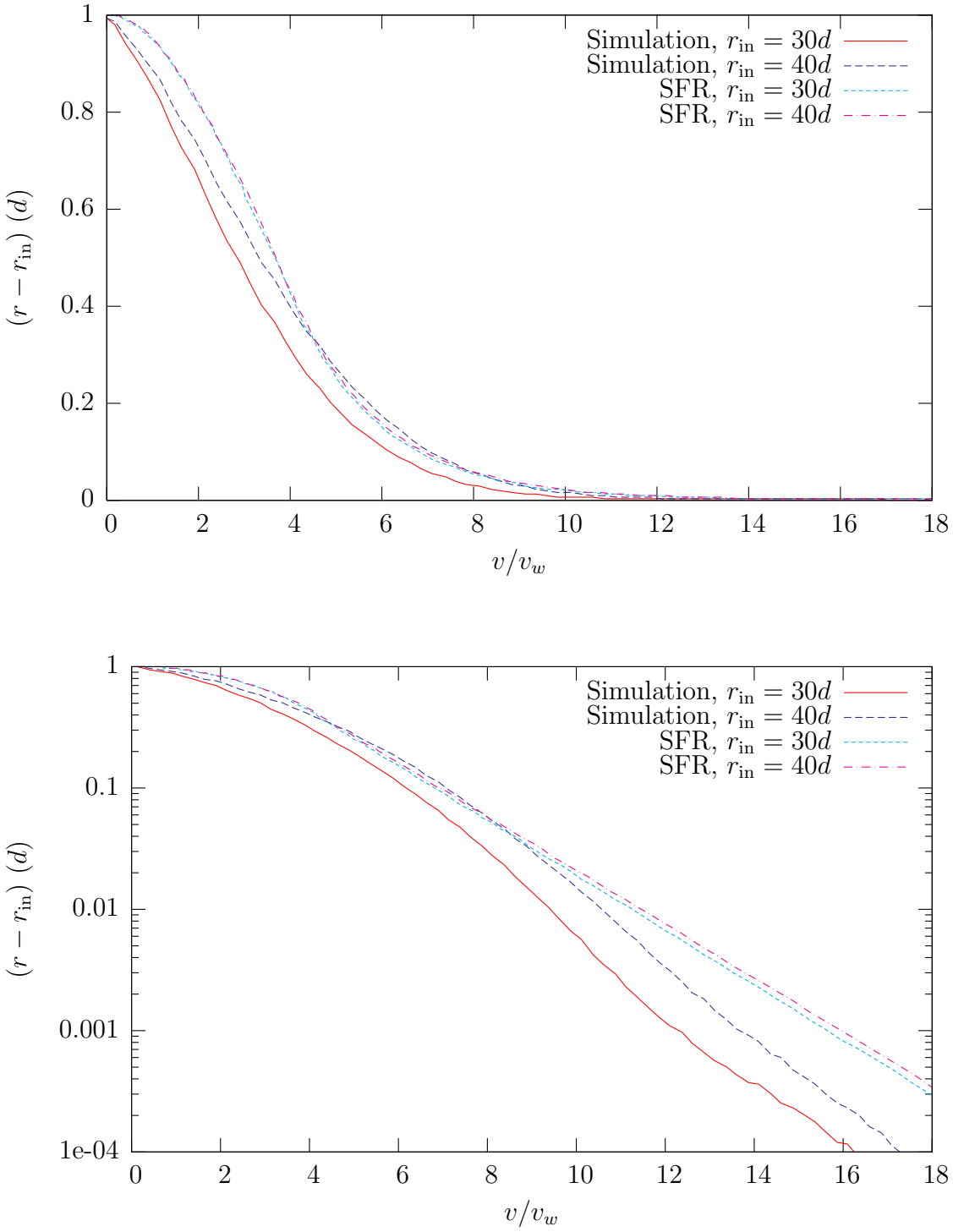


Figure 6-8: DEM simulation results for the flow profile compared to SFR predictions, for two different values of r_{in}. For solidarity with the silo flow predictions, the SFR solutions use $L_s = 4d$.

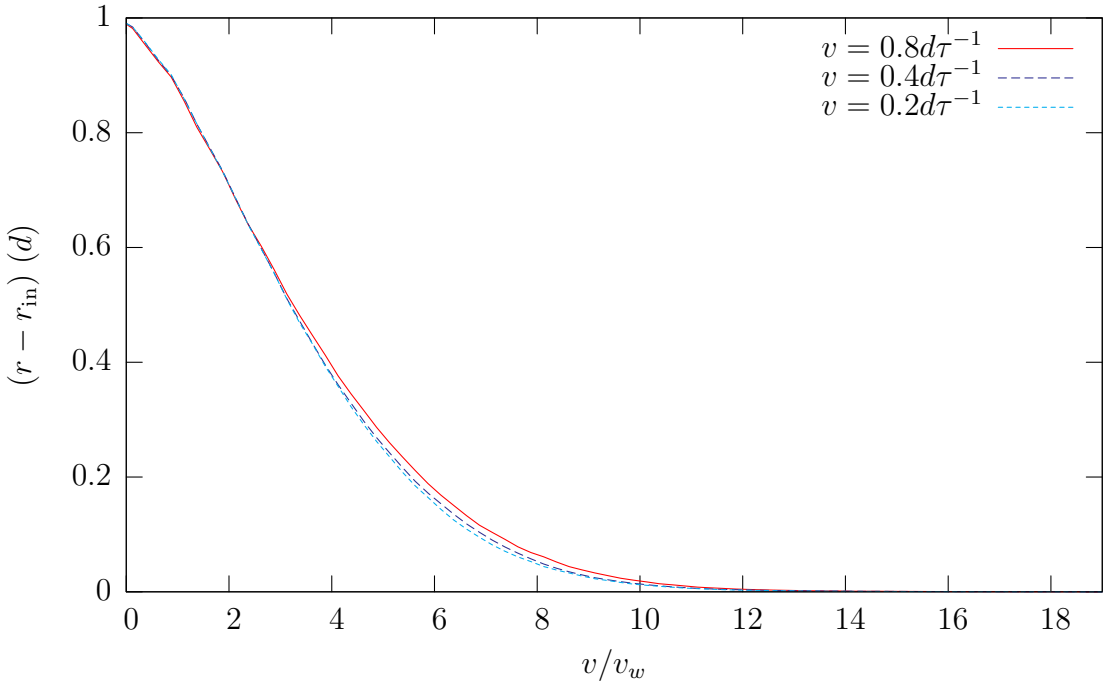


Figure 6-9: Velocity profiles for three different angular velocities, with $\mu_c = 0.5$ and $r_{\text{in}} = 40d$. The time windows over which the velocities are computed are scaled according to the angular velocity.

properties unique to granular materials such as Gaussian downward velocity in the draining silo and exponentially decaying velocity in the annular Couette cell.

Our simulations also indicate that the slight changes in flow brought on by varying the inter-particle contact friction in the annular flow geometry match the trends the SFR predicts when the internal friction angle is appropriately varied. The trend is also captured when the inner wall radius is varied, though the quantitative agreement is not as strong. In agreement with past studies on annular Couette flow, and in validation of one of the first principles behind the SFR, we find in our simulations that the flow rate does not significantly affect the normalized flow profile over a significant range of rates.

Chapter 7

Measuring a granular continuum element

7.1 Introduction

In the previous chapter, it was shown that the Stochastic Flow Rule could extend the applicability of the spot model to other geometries. As the first theory to predict the silo drainage and annular Couette flow from the same underlying principles, it represents a significant advance. However, to achieve this, it made use of the Mohr-Coulomb stresses, and as previously mentioned, there are several aspects of this theory which are troubling, with the theory predicting shocks, even for some simple geometries.

Much of the problem with Mohr-Coulomb Plasticity comes from its treatment of the microscopic element of granular element. As previously described, the additional assumption of Mohr-Coulomb Incipient Yield, stating that the entire packing achieved critical value of μ everywhere, was employed to create a closed system of equations. Ideally, we would like to be able to directly test whether these assumptions about a microscopic granular element are valid.

Recently, a variety of other continuum theories for granular materials have been proposed, which attempt to more precisely capture the underlying physics of a granular element. The Shear Transformation Zones theory developed by Langer *et al.* [43,

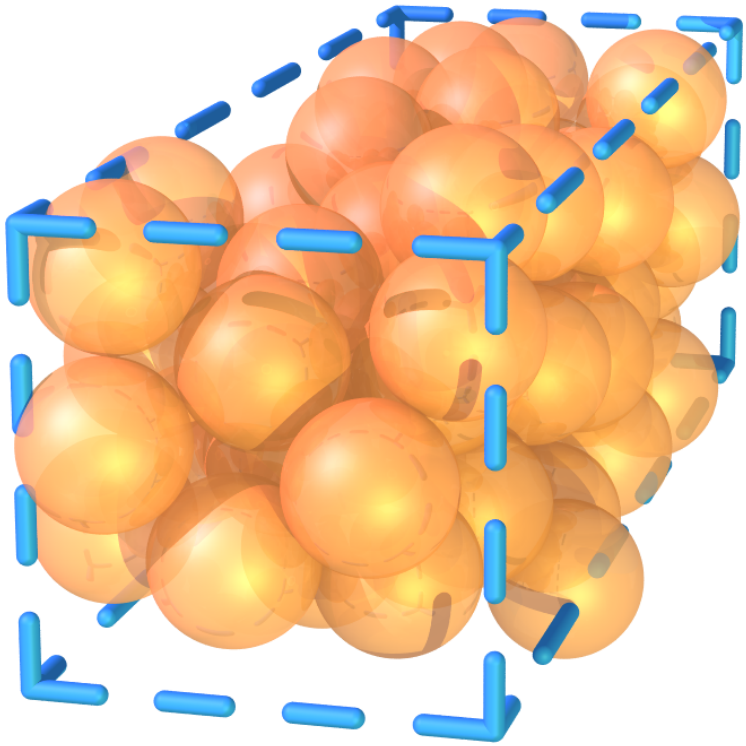


Figure 7-1: A typical $2.5d \times 8d \times 2.5d$ cell of particles from a discrete element method simulation, proposed as an approximate granular element for this study.

75, 73, 74] provides a mechanism for giving granular materials a memory of their shearing history. The partial fluidization model developed by Aranson and Tsimring [8, 9] attempts to separate the solid-like and liquid-like properties of a granular material via an order parameter. From studies of inclined planes, Pouliquen has proposed a flow rule for dense granular flows [62]. However, none of these theories have been developed into a complete continuum description of a granular flow. Like the Mohr-Coulomb theory, additional hypotheses of the granular physics have been employed to create a closed theory, which are then used to generate macroscopic predictions. Testing the macroscopic predictions provides an indirect way to evaluate the original microscopic model.

This chapter demonstrates an alternative approach. Instead of looking at macroscopic properties, we ask the question of whether is possible to directly view and test the properties of a granular element. However, at first sight, it is unclear whether a granular element can be meaningfully defined. In a dense amorphous packing, associ-

ating the notion of a granular element with a single particle appears unclear. As previously noticed, granular materials have complex force networks which are extremely non-homogeneous, with stresses frequently being concentrated on a fractal-like networks of particles. It does not appear that these forces can be directly associated with a continuum notion of stress.

To make progress, we refer back to the results of previous chapters. In chapter 3 it was shown that even with an approximate random-walk model of granular flow, it was possible to generate realistic flowing random packings via the spot mechanism. It suggests that we may only need a physical model to be valid down to the scale of a spot, since the spot model microscopic mechanism will always be able to correct the microscopic packing statistics at the local level. This motivates the definition of a granular element on a mesoscopic length scale, as shown in figure 7-1. Notions of a continuum variables at a single particle level may not make sense, but at the mesoscopic scale, it is possible that they can be defined statistically.

To test this, large-scale DEM simulations were carried out in a variety of dense granular flows. For simplicity, attention to was restricted to quasi-2D flows, where one dimension is periodic with period $8d$, but this still allows for the consideration of a fully three dimensional stress tensor. For each simulation considered, the flow was divided into mesoscopic granular elements, on the scale of $2.5d \times 8d \times 2.5d$, and material quantities were computed for each of these elements. The primary aim has been to show that an approximate granular continuum element can be defined at this scale. Once this is established, the data has been used to test and develop some of the fundamental ideas behind continuum granular flow theories at the local level, by viewing the simulations as an ensemble of approximate granular elements.

7.2 DEM Simulation

Since our overall aim was to extract information about the inherent properties of a granular material, and not its behavior in a particular situation, we considered a variety of different DEM simulations, with different geometries and forcings. To

minimize the effects of elastic modes in the DEM simulations, the Hookean contact model was employed.

Our initial particle packings were generated using a pouring process. We considered a tall silo with base at $z = 0$ and walls at $x = \pm 25d$, and poured in 55,000 particles from $z = 160d$ at a constant rate of $123\tau^{-1}$ to fill the silo to an approximate height of $z = 114d$. We also considered a wide silo, with walls at $x \pm 75d, z = 0$, and poured in 100,000 particles from $z = 160d$ at a constant rate of $379\tau^{-1}$, making a packing up to approximately $z = 69d$. For both situations, the initial packing fraction is approximately 63.5%.

In some situations considered, flow occurs passively in response to gravity. However, we also considered several situations using an external force. In all these cases, this was done by freezing certain particles (so that all forces and torques were zeroed) and then moving them at a constant velocity. For the initial study, three geometries were considered:

- Drainage from the tall silo, through a $6d$-wide slit in the center of the base.
- Drainage from the wide silo, through a $6d$-wide slit in the center of the base.
- Deformation of the wide silo by pushing, created by freezing all particles with centers initially satisfying $z < 7.5d$, and then moving those with $x > 0$ upwards at a constant speed of $0.2d\tau^{-1}$.

In addition, two simulations were carried out to investigate the effect of arresting a developed flow:

- Stopping drainage in the wide silo, by plugging the hole after 50τ.
- Stopping the pushing deformation after 25τ, when the right side of the packing been raised by $5d$.

Finally, to investigate the precise role of strain and strain rate, we considered one additional simulation:

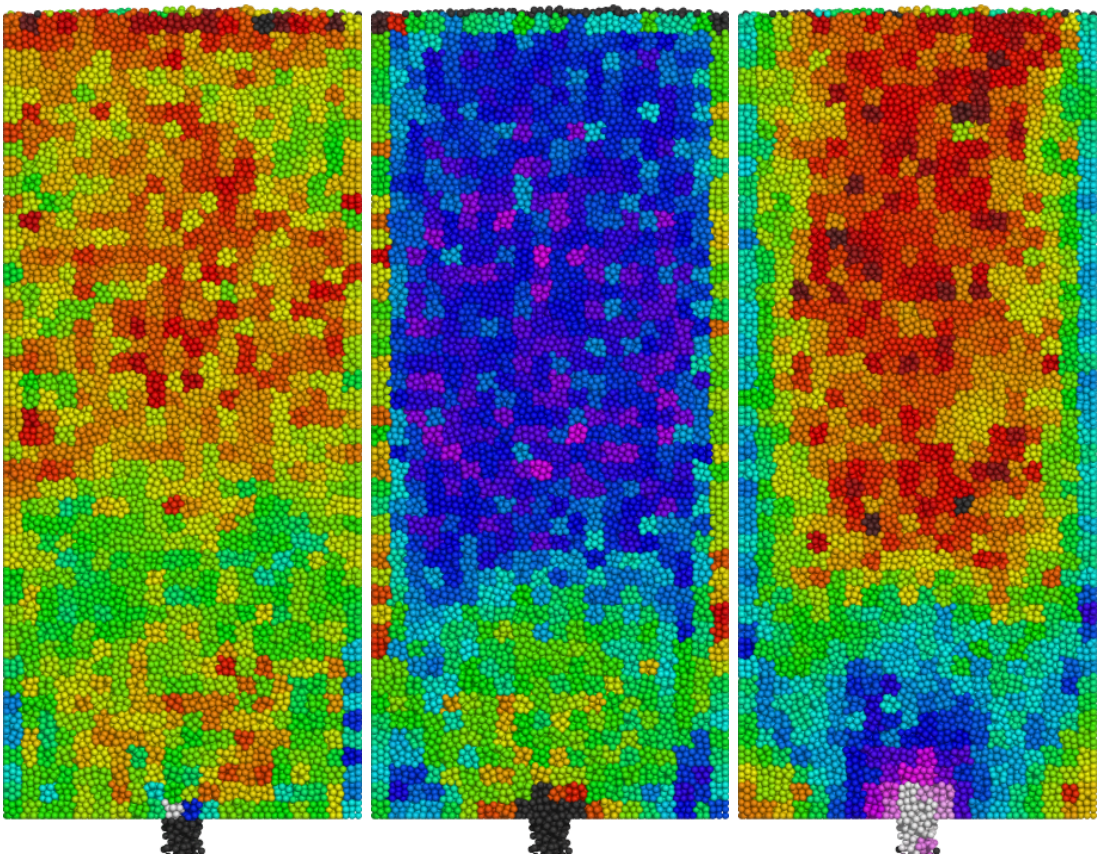


Figure 7-2: Three computed material quantities (μ, packing fraction ϕ, and magnitude of deviatoric strain rate $|\mathbf{D_0}|$) in the tall silo drainage simulation, shown at $t = 25\tau$. All the particles in each computational cell are colored according to the computed material parameters for that cell, using the color scheme of figure 7-3.

- Shearing the wide silo, by freezing all particles whose centers lie within $5d$ of any wall, and then moving those with $x \lesseqgtr 0$ with a velocity of $\pm 0.5 d\tau^{-1}$ in the x direction.

7.3 Computation of material parameters

For each simulation, a snapshot of all particle positions was recorded at fixed intervals of 0.2τ. In addition to this information, numerous material quantities were calculated in a grid of cells of size $2.5d \times 8d \times 2.5d$, and the precise details of these measurements are described below.

As discussed in chapters 4 and 5 we have previously computed local packing frac-

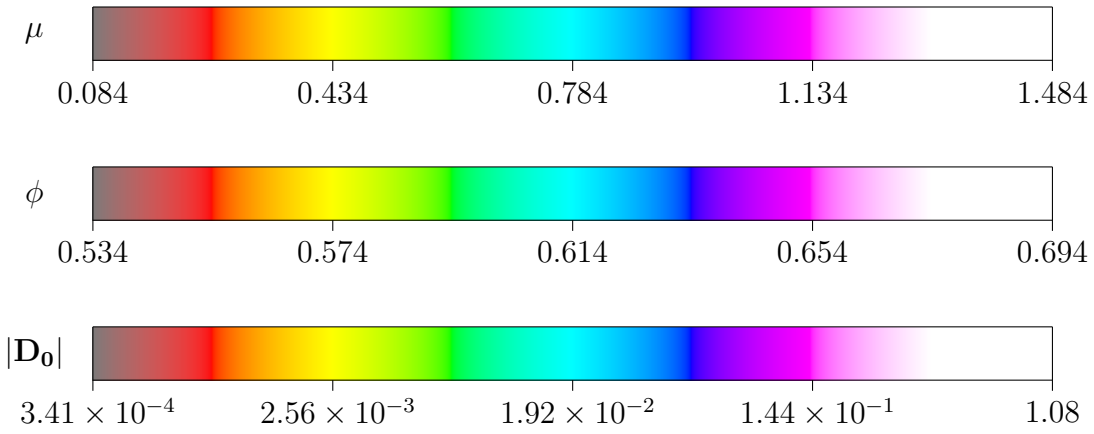


Figure 7-3: Color scheme for the quantities plotted in figures 7-2, 7-4, 7-5, and 7-12. μ and ϕ are dimensionless quantities, while $|\mathbf{D_0}|$ has units of τ^{-1}.

tion using a Voronoi cell method. However, for reasons of computational convenience, we computed it in each cell by evaluating the precise fraction of the cell volume which is occupied by particles. Particles which lie across the boundary of the cell make a fractional contribution, which provides much higher accuracy than only counting particles whose centers lie in the cell. Since the cell is periodic in the y direction, the two cases that must be considered are when the particle intersects a face or edge of the box. Consider a sphere defined by $x^2 + y^2 + z^2 < d^2$. By elementary integration, the proportion of the sphere's volume in the region $x < a$ is

$$\frac{(2d - a)(d + a)^2}{4d^3}$$

It can also be shown that the proportion of the sphere in the region $x < a, y < b$ (where $a^2 + b^2 < d^2$) is

$$\begin{aligned}
&\frac{2ab\sqrt{d^2 - a^2 - b^2}}{3d^3} \\
&+\frac{b}{d^3}\left(d^2 - \frac{b^2}{3}\right)\left(\sin^{-1}\left(\frac{a}{\sqrt{d^2 - b^2}}\right) + \frac{\pi}{2}\right) \\
&+\frac{a}{d^3}\left(d^2 - \frac{a^2}{3}\right)\left(\sin^{-1}\left(\frac{b}{\sqrt{d^2 - a^2}}\right) + \frac{\pi}{2}\right) \\
&+\frac{1}{3}\left(\sin^{-1}\left(\frac{d^4 - (a^2 + b^2)d^2 - a^2b^2)}{(b^2 - d^2)(a^2 - d^2)}\right) + \frac{\pi}{2}\right).
\end{aligned}$$

The stress tensor in each cell is calculated by looking at the forces between particles in contact. If there are N particles in the cell, and particle l has a total of N_l contacts, then an approximate stress tensor can be defined by

$$T_{ij} = \frac{1}{V} \sum_{l=1}^{N} \sum_{k=1}^{N_l} \Delta x_i^{(k,l)} F_j^{(k,l)}$$

where $\mathbf{F}^{k,l}$ is the force of the kth contact on particle l, and $\mathbf{\Delta x}$ is the separation vector from the point of center of the particle to the point of contact. V is the volume of the cell, and ensures that the stress tensor has the correct units of energy per unit volume, or equivalently force per unit area. This definition is appropriate for a single particle, and taking the average over many particles improves accuracy. This definition of T_{ij} is not explicitly symmetric, but it can be shown that it is approximately symmetric, by considering torques. The total torque on a particle, obtained by summing over all contacts, must add up to the particle's rotational inertia times the particle's angular acceleration. Since the rotational inertia is proportional to d^2, and the particle radius is supposed to be a differentially small quantity, the sum of torques must approach zero. By considering th kth component of torque, we see that

$$\begin{aligned} 0 &\approx \sum_{l=1}^{N} \sum_{k=1}^{N_l} \epsilon_{ijk} x_i^{(k,l)} F_j^{(k,l)} \\ &= \epsilon_{ijk} \sum_{l=1}^{N} \sum_{k=1}^{N_l} x_i^{(k,l)} F_j^{(k,l)} \\ &= \epsilon_{ijk} T_{ij} \end{aligned}$$

and thus $\mathbf{T}$ is approximately symmetric. From this, we can define a pressure $p = \frac{1}{3} \operatorname{tr} \mathbf{T}$ and a deviatoric stress tensor $\mathbf{T_0} = T - \frac{1}{3} p \mathbf{I}$.

In this chapter, we have a fully three-dimensional stress tensor, and thus the simple definition of μ as shear stress divided by normal stress employed in chapter 6 is no longer appropriate. There are three stress directions, with eigenvalues $\lambda_1 < \lambda_2 < \lambda_3$, and from this several definitions of μ exist, two of which are:

- **Drucker-Prager:** For this definition, the normal stress is identified with the

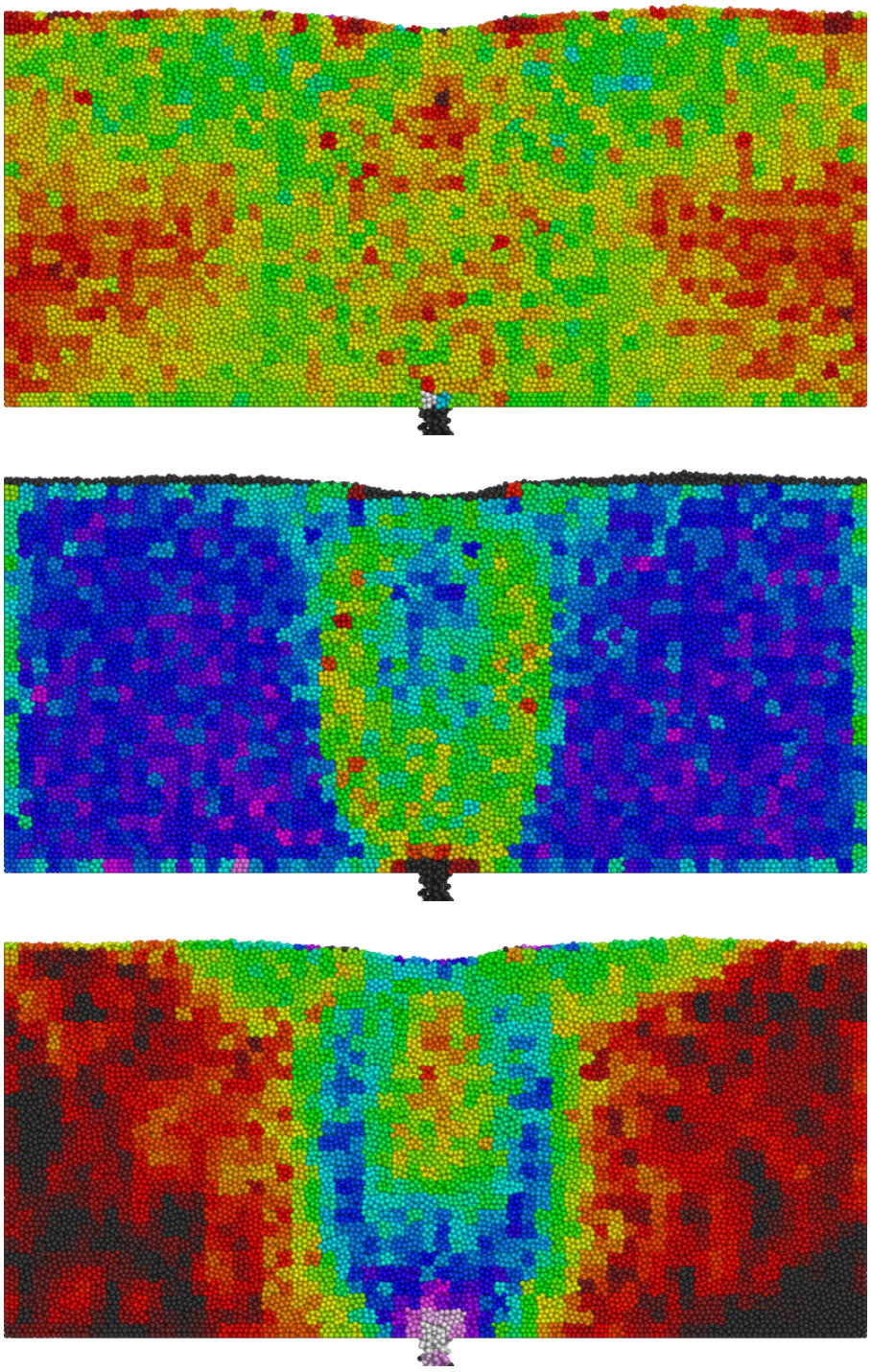


Figure 7-4: Three computed material quantities (μ, top; packing fraction ϕ, middle; magnitude of deviatoric strain rate $|\mathbf{D_0}|$, bottom) in the wide silo drainage simulation at $t = 40\tau$, using the color scheme in figure 7-3.

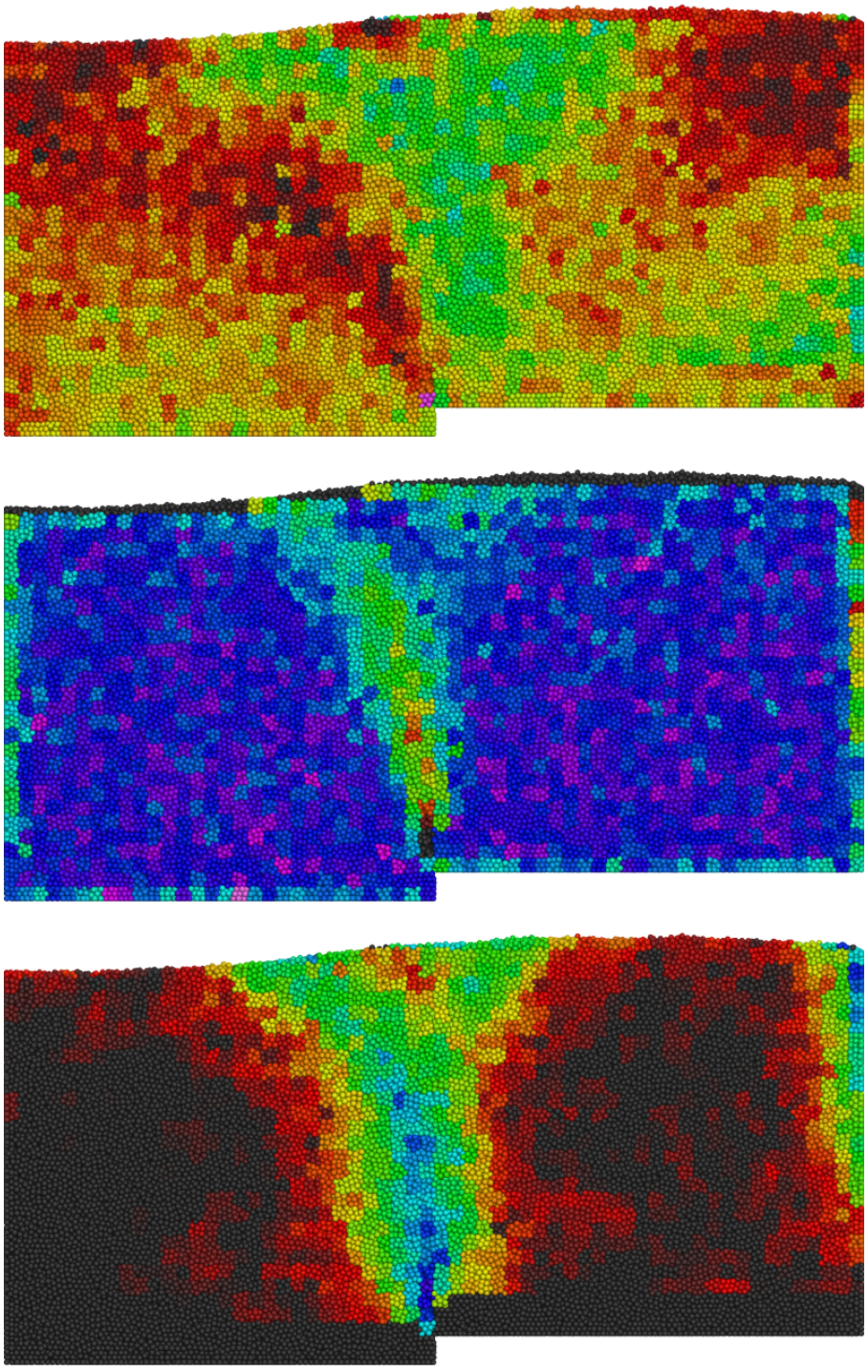


Figure 7-5: Three computed material quantities (μ, top; packing fraction ϕ, middle; magnitude of deviatoric strain rate $|\mathbf{D_0}|$, bottom) in the wide silo pushing simulation at $t = 25\tau$, using the color scheme in figure 7-3.

pressure p, and the shear stress is identified with the magnitude of $\mathbf{D_0}$, divided by a normalizing factor $\sqrt{2}$, so that

$$\mu = \frac{|\mathbf{D_0}|}{p\sqrt{2}}.$$

In terms of the eigenvalues, we have $p = (\lambda_1 + \lambda_2 + \lambda_3)/3$ and

$$\mu = \frac{\sqrt{(\lambda_1 - p)^2 + (\lambda_2 - p)^2 + (\lambda_3 - p)^2}}{p\sqrt{2}}.$$

- **Mohr-Coulomb:** For this definition, the intermediate stress λ_2 is treated as irrelevant, and μ is based on a two-dimensional computation in the plane of the minimal and maximal stresses, where there is an effective stress tensor

$$\mathbf{T}_{\text{eff}} = \begin{pmatrix} \lambda_1 & 0 \\ 0 & \lambda_3 \end{pmatrix}.$$

Rotating the stress tensor by 45° gives

$$\mathbf{T}'_{\text{eff}} = \frac{1}{2}\begin{pmatrix} \lambda_1 + \lambda_3 & \lambda_1 - \lambda_3 \\ \lambda_3 - \lambda_1 & \lambda_1 + \lambda_3 \end{pmatrix}.$$

In this form, we can see that it makes sense to define a normal stress as $p_{\text{eff}} = \operatorname{tr} T = (\lambda_1 + \lambda_3)/2$ and a shear stress as $(\lambda_3 - \lambda_1)/2$. Thus we can define

$$\mu = \frac{\lambda_3 - \lambda_1}{\lambda_3 + \lambda_1}.$$

It was found that both definitions of μ would give reasonable results, although it did appear that in many situations, the value of the intermediate stress had little effect, and thus for the results presented here, the Mohr-Coulomb definition was adopted. Although the stress tensors computed here were fully three-dimensional, they are strongly constrained due to the periodicity in the y direction, and a more in-depth study of the merits of the two definitions would be better carried out in fully three

dimensional, non-periodic systems where there would be greater variations in the stress tensor.

To calculate the strain rate tensor in a cell, we consider the least squares regression problem

$$\mathbf{v} = \mathbf{M}\mathbf{x} + \mathbf{v}^0.$$

Here $\mathbf{x}$ and $\mathbf{v}$ are the instantaneous positions and velocities of all particles within the cell. We find the average cell velocity $\mathbf{v}^0$ and the velocity gradient by minimizing the sum of squares of residuals. Since the simulation is periodic in the y direction, we enforce that the second column of M is zero, as the velocity should not have an explicit dependence on the periodic coordinate. Note however that the velocity in the y direction does play a role, and the elements M_{21}, M_{23} are allowed to be non-zero, and may, for example, be significant in the shearing simulation. The regression problem can be broken down into three separate equations each of which has three parameters, M_{i1}, M_{i3}, and v_i^0. The parameter v_i^0 can be expressed in terms of the means of the $\mathbf{x}$ and $\mathbf{v}$, leaving two components in M for each equation. This can be solved algebraically, without any need to result to a linear matrix solver. The strain rate tensor is then defined as the symmetric part of M, namely

$$\mathbf{D} = \frac{\mathbf{M} + \mathbf{M}^{\mathrm{T}}}{2}.$$

From this, the deviatoric strain rate tensor is defined as $\mathbf{D_0} = \mathbf{D} - \frac{1}{3}(\operatorname{tr}\mathbf{D})\mathbf{I}$. In most cases we expect $\mathbf{D_0} \approx \mathbf{D}$ since the density of the granular material does not fluctuate by a large amount, making the contribution to $\mathbf{D}$ from shearing larger than that from dilation. For some of the analysis, we made use of a normalized strain rate $|\mathbf{D_0}|/\sqrt{p}$. For the results presented here, we used instantaneous particles velocities throughout. However, since these instantaneous quantities may be subject to simulation artefacts, such as particle rattling, or elastic modes, we also considered calculating strain rate based on velocities computed by interpolating between particle positions between successive frames, effectively giving an averaged velocity on the scale of 0.2τ. A plot of $\log|\mathbf{D_0}|$ for the two different computation methods appears linear to a high degree

of accuracy, indicating little physical distinction between them.

7.4 Stress, strain rate and packing fraction

Figure 7-2 shows plots of the Mohr-Coulomb parameter μ, the local packing fraction ϕ, and the magnitude of deviatoric strain rate $|\mathbf{D_0}|$ for the tall silo drainage simulation. Near the orifice, there is a converging region of flow, that has roughly parabolic streamlines. Higher in the container, there is a transition to uniform flow, where the particles drop like a plug, behave like a solid, and experience little rearrangement. The plot of $|\mathbf{D_0}|$ supports these results. High in the container, very little rearrangement is seen, while there is a sharp transition to high values when the packing must undergo deformation to pass through the orifice.

It is clear from looking at the plot of local packing fraction, that ϕ and $|\mathbf{D_0}|$ are closely correlated. In the upper region, where particles are falling like a plug, the packing fraction remains constant. In the converging region, the packing fraction decreases, as the particles must have more free space in order to geometrically rearrange. This verifies the well-known concept of shear-dilation, and is studied quantitatively in later sections.

Of the three plots, μ exhibits the largest fluctuations, which we attribute to the fact that it is a ratio between two computed quantities. However, large variations over the range 0.2 to 0.6 can be clearly seen. This immediately calls the Mohr-Coulomb incipient yield hypothesis into question, which would predict that μ would be constant everywhere, achieving a value of approximately 0.45 to 0.55. At first sight, the spatial differences in μ appear confusing, and not directly correlated with the other two plots. However, regions of higher μ exist at the interface between the plug-like region, and the converging flow region, suggesting that a large μ my be required in order to first achieve material failure; this point will be expanded on in section 7.5.

Figures 7-4 and 7-5 show the same three plots for the wide drainage and wide pushing simulations. Although these are different situations, a number of the same conclusions can be drawn. Again, we see a clear differentiation between solid-like

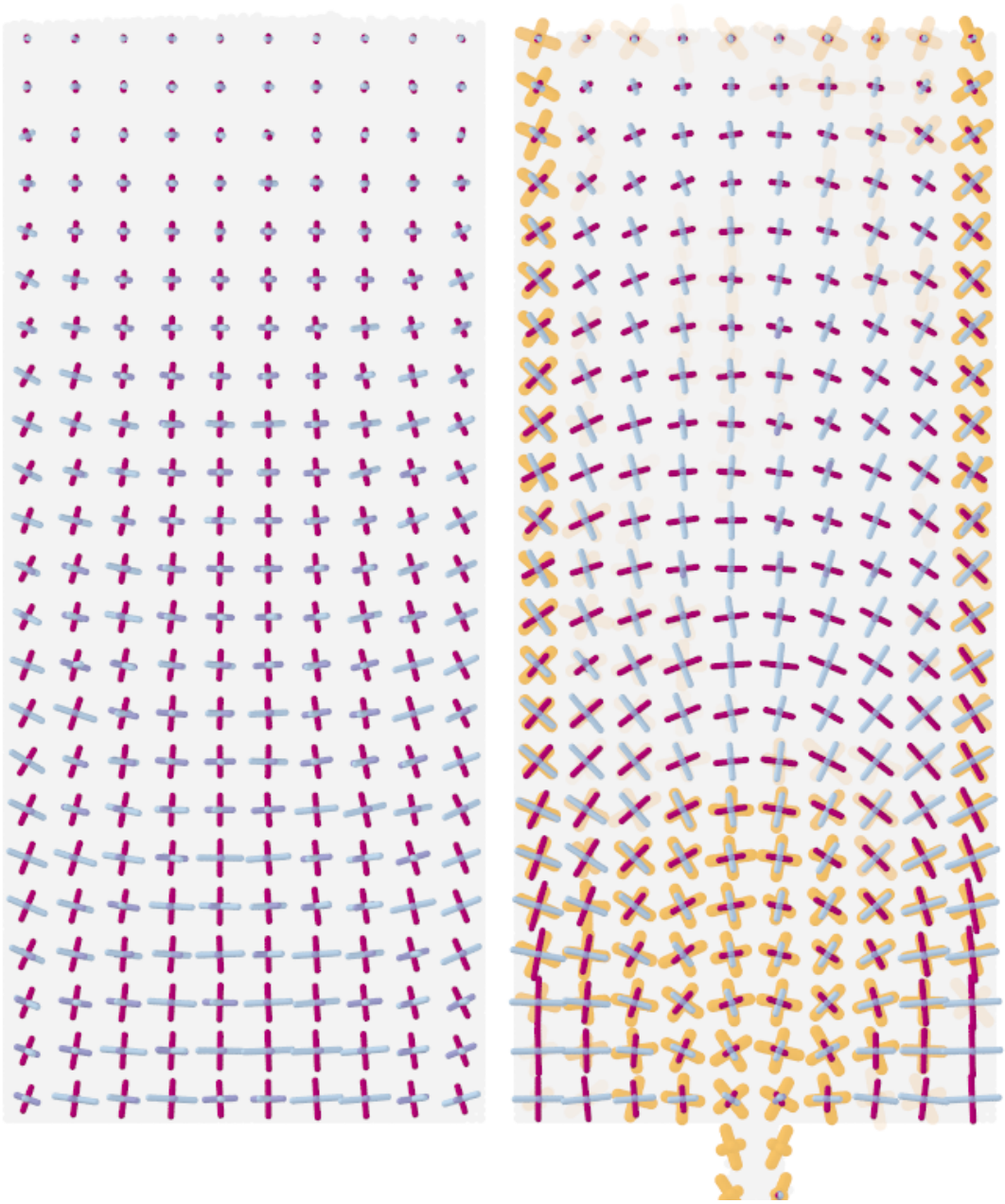


Figure 7-6: Plots of the directions and magnitudes of the eigenvectors of the deviatoric stress and strain rate tensors, calculated instantaneously in $5d \times 8d \times 5d$ boxes with no time-averaging, for the tall silo before drainage (left), and during drainage (right). The maximal stress eigenvector is shown in purple, with the other two eigenvectors being shown in blue. In the regions where deformation is occurring, the maximal and minimal eigenvectors of the strain rate tensors are plotted in orange. A high degree of alignment between the two tensors can be seen.

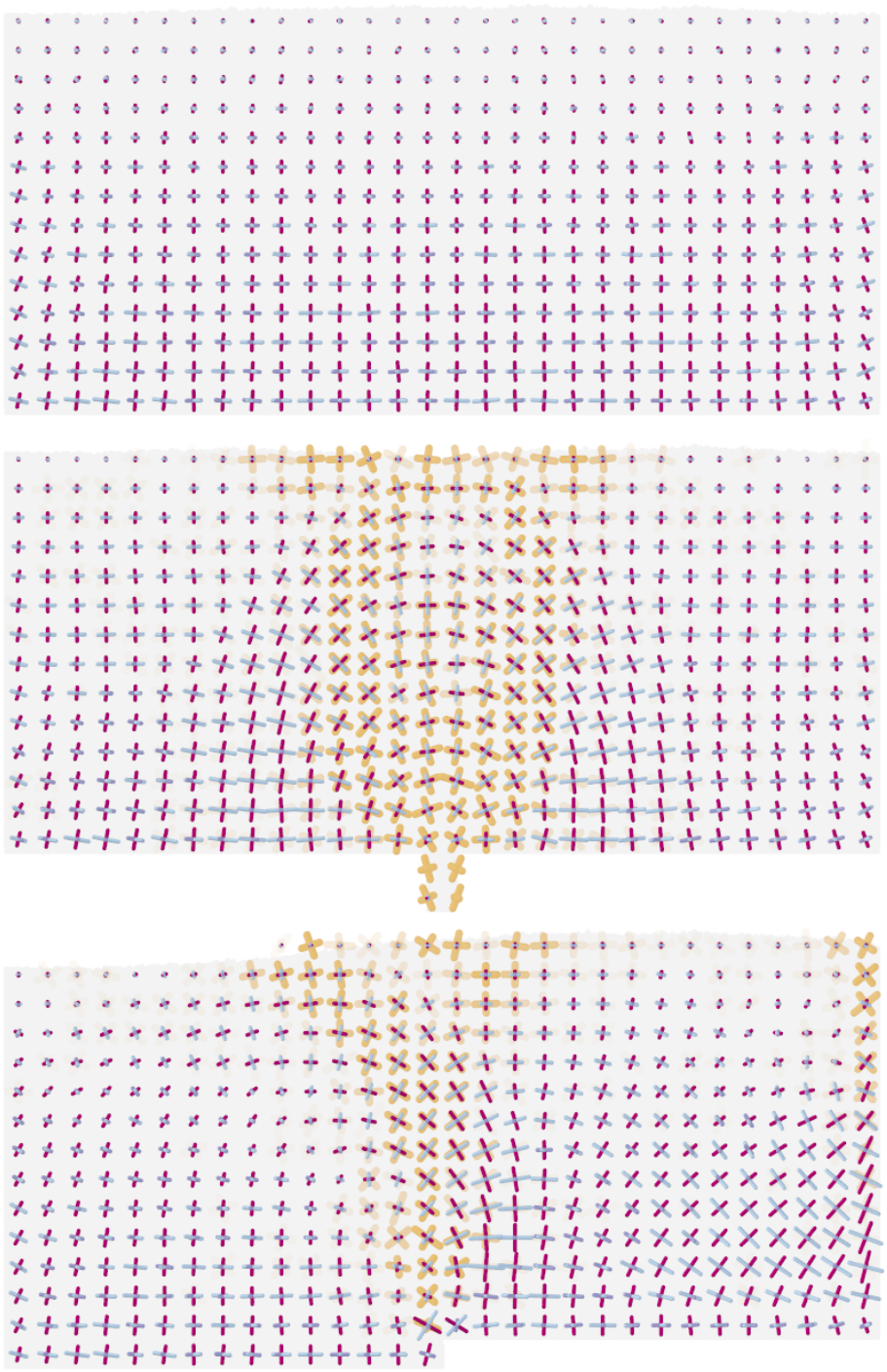


Figure 7-7: Plots of the directions and magnitudes of the eigenvectors of the deviatoric stress and strain rate tensors, calculated instantaneously in $5d \times 8d \times 5d$ boxes with no time-averaging, shown for the initial packing (top), during drainage (middle), and during pushing (bottom). The same color scheme is used as in figure 7-6. Again, a high degree of alignment between the two tensors can be seen.

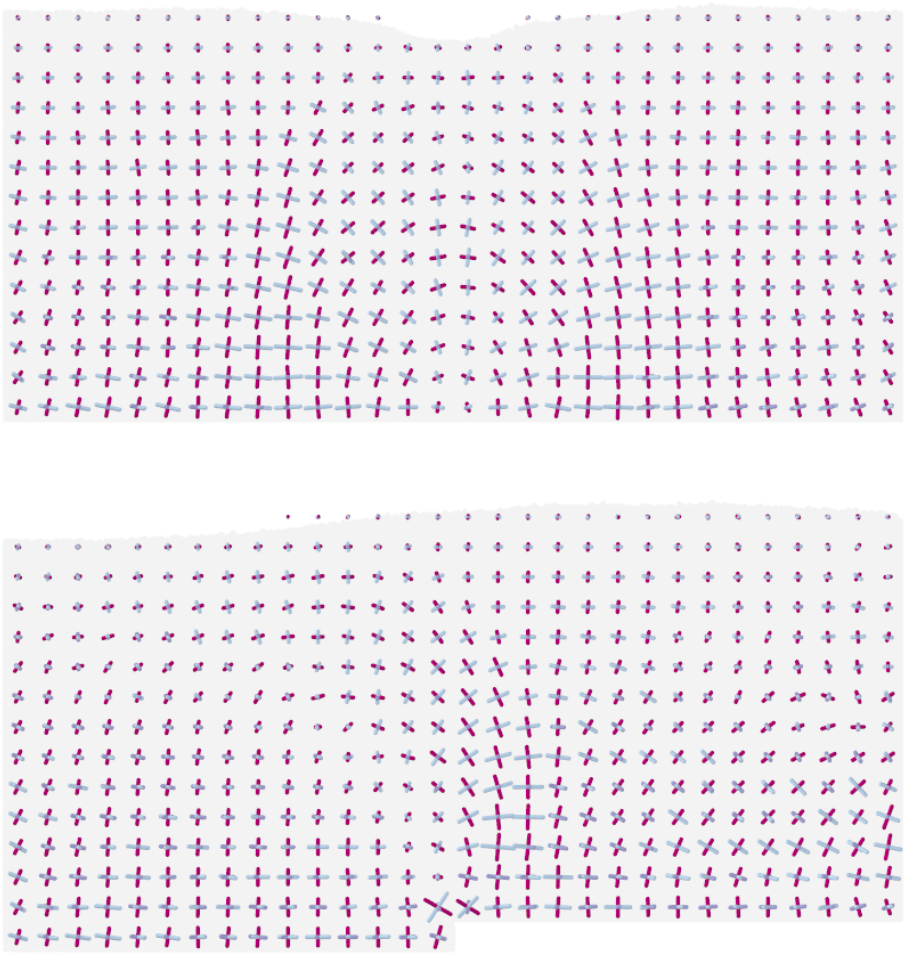


Figure 7-8: Plots of the directions and magnitudes of the eigenvectors of the deviatoric stress tensors, calculated instantaneously in $5d \times 8d \times 5d$ boxes with no time-averaging, after the drainage (left) and pushing processes were arrested. The same color scheme is used as in figure 7-6. The stress lines closely resemble the stress lines during the corresponding flowing states.

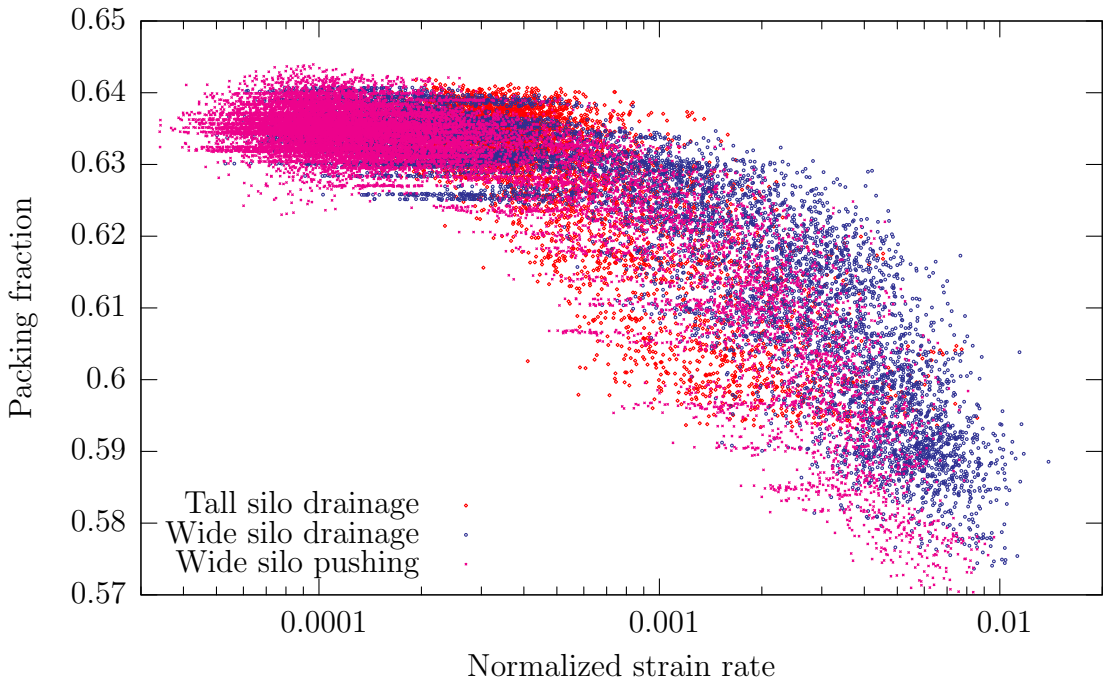


Figure 7-9: Normalized strain rate versus packing fraction for three different simulations. Each point corresponds to the strain rate and packing fraction of a computational cell at a particular instant. The results for computational cells next to the walls exhibit more complex behavior and are not included.

regions with low strain rate and high packing fraction, and liquid-like regions with high strain rate and low packing fraction. Again, the regions of highest μ appear to be at the interfaces between the solid-like and liquid-like regions. Since these correlations hold locally across a variety of different experimental geometries, it gives us reason to believe that they tell us something inherent about the granular material, that is not tied to a particular geometry.

Figures 7-6 and 7-7 show the directions and magnitudes of the eigenvectors of the deviatoric stress tensor for three different simulations, compared with their initial states. For these images, the stress tensor was calculated on a $5d \times 8d \times 5d$ grid, by averaging the computed stress tensor in 2×2 blocks of cells. Even though these stresses were computed using local, instantaneous data, we can see that in all situations, the stress tensors are smooth, and exhibit none of the shocks predicted by plasticity theory. For regions undergoing deformation, the eigenvectors of the strain rate tensor are also shown. We see that in all areas where there is appreciable strain

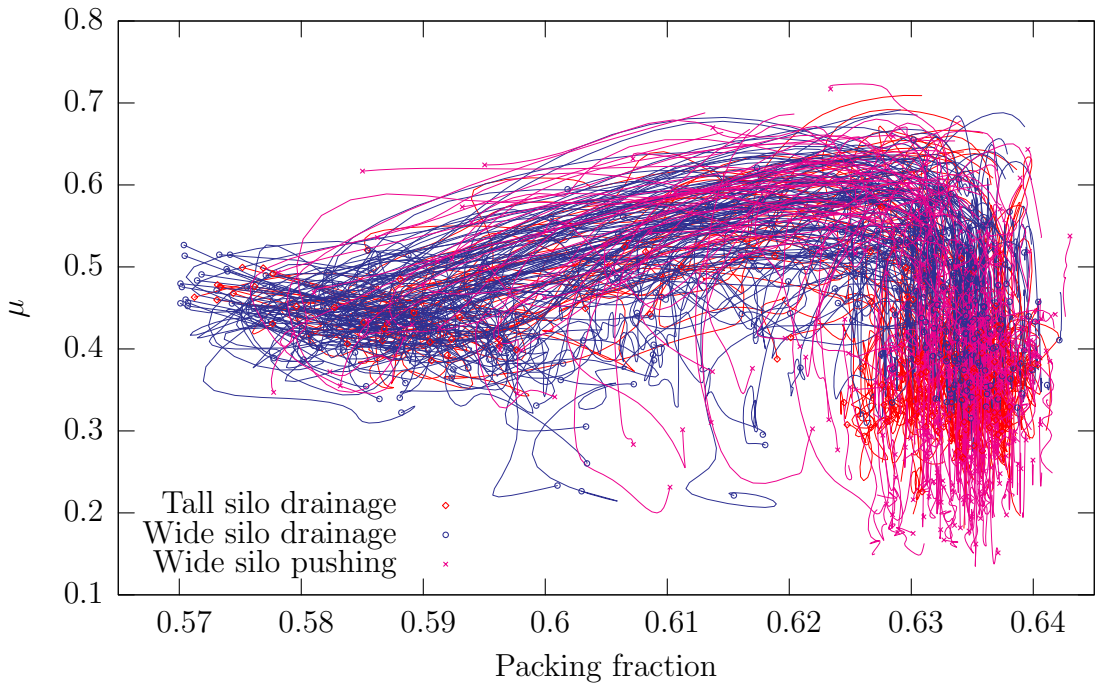


Figure 7-10: Packing fraction versus μ for three different simulations. Each line corresponds to the history of a granular element being advected during the flow of the material. The points which are plotted correspond to the end of the tracers. The tracers' trajectories are Gaussian-smoothed over a time scale of twenty frames to improve accuracy.

rate, there is a strong alignment between the two sets of eigenvectors, even at the local, instantaneous level. If the plots are time-averaged over a window of twenty frames, then the agreement becomes almost perfect. Averaging over progressively larger time windows also allows us to verify coaxiality as far into the granular packing as one can reasonably define a strain rate tensor. These results suggest that while the solid-like/liquid-like distinction is useful, it may be that the concept of liquid-like only makes sense when coupled with an appropriate time window.

In the two wide simulations, the flow process was arrested, and the resulting stress tensors are shown in figure 7-8, The stresses during flow essentially frozen in place, apart from small variations, presumably due to the fact that the material's weight must be redistributed as the packing alters from flowing to static. The stress lines showed no indication of reverting back to their initial state.

7.5 Evolution of material parameters

The plots shown in figures 7-2, 7-4, and 7-5 suggest that a number of correlations exist between the various material parameters at the local level. In this section we investigate these correlations in detail, by viewing all the material cells from the different experiments as an ensemble of approximate "granular elements". Seeking statistical signatures of the material parameters over the ensemble allows us to infer the behaviour of a granular element.

As a first example, consider figure 7-9(a), showing a plot of normalized strain rate $|\mathbf{D_0}|/\sqrt{p}$ against packing fraction. Each point on the graph corresponds to the instantaneous computed values of a material element from one of the simulations, during a period when steady flow has developed. We see an approximate collapse of the points from the three simulations, suggesting that the correlation we are viewing is a property inherent to the granular material, and not tied to any particular simulation configuration. This result has been observed by other authors, such as in the two dimensional simulations of da Cruz *et al.* [33], but we believe this is the first direct verification in three dimensions, at the local level.

While instructive, the above approach will only allow us to search for direct correlations between variables. In reality, we expect that the material parameters are related in a more complicated way, and that differential relationships may exist. Furthermore, we expect that each granular element may have a "memory", and that there may be internal degrees of freedom which we are not measuring, which relate to its history of deformation and stress. Because of these complications, it is natural to switch from an Eulerian to Lagrangian approach: rather than viewing our data set as an ensemble of over different spatial positions and times, we treat it as an ensemble of different spatial elements which evolve over time. Each granular element corresponds to an approximate trace in the phase space of material parameters over its history during the simulation.

To correctly implement this, we must also take into account that a granular cell may move during the simulation. We therefore initially introduced a number of tracer

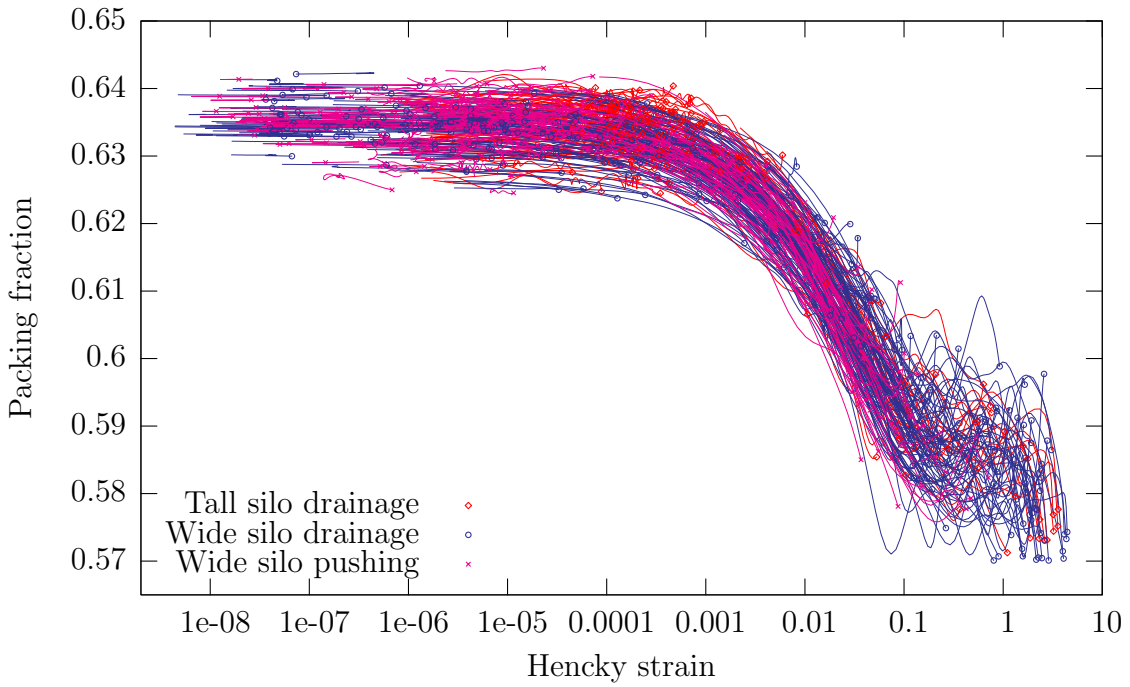


Figure 7-11: Strain versus packing fraction for three different simulations. Each line corresponds to the history of a granular element being advected during the flow of the material. The points which are plotted correspond to the end of the tracers. The tracers' trajectories are Gaussian-smoothed over a time scale of twenty frames to improve accuracy.

positions on a $5d \times 5d$ lattice. During the simulation, the tracer positions are advected according the average background velocity of the particles. For each tracer, a history of a granular element is created by linearly interpolating all the material parameters from the underlying raw simulation data on the $2.5d \times 2.5d$ lattice.

Figure 7-9 shows sample of tracers in a plot of μ against packing fraction ϕ. The plot shows us that, while μ is not directly correlated to ϕ, it plays an important role in the failure of a granular element. The majority of tracers start off on the right side of the graph, at the initial packing fraction of 63.5%. Those tracers corresponding to failing elements take a path in an inverted U-shape, first attaining a value of μ of approximately 0.55 before starting to decrease in packing fraction. A material element will dilate only if a critical value of μ is first attained. This directly relates to the behavior seen in figures 7-2, 7-4, and 7-5, where it was noted that areas of larger μ were located at the interface between solid-like and liquid-like regions. Also

visible in 7-10 is tendency for dilated elements which experience a low value of μ to recompact slightly, suggested by the fact that the majority of the paths in the lower left portion of the graph are moving rightwards.

In the Lagrangian approach it also makes sense to consider the total strain experienced by a material element. Strain is calculated from the simulation data by using the snapshots of all particle positions. At $t = 0$, a $5d \times 8d \times 5d$ box of particles is labeled, centered on each material tracer. Let $\mathbf{x}^i$ be the initial particle positions in a particular box. A deformation gradient tensor $\mathbf{F}$ and an overall drift $\mathbf{x}^0$ can then be found by considering the least squares regression problem

$$\mathbf{x} = \mathbf{F}\mathbf{x}^i + \mathbf{x}^0$$

where $\mathbf{x}$ are the current particle positions. Due to the periodicity, we enforce that $M_{12} = M_{32} = 0$, and $M_{22} = 1$. This problem is very similar to the previous calculations of a deformation tensor, and can be done analytically. We can then compute a polar decomposition $\mathbf{F} = \mathbf{R}\mathbf{U}$ where $\mathbf{R}$ is a rotation matrix, and $\mathbf{U}$ is a symmetric positive definite matrix. The concept of a strain is not well-defined, and from $\mathbf{U}$ there are several ways to define strain-like quantity. For the work presented here, we make use of the Hencky strain $\mathbf{E} = \log \mathbf{U}$, which has the additional property that strains are additive. However, this choice is arbitrary, and the same conclusions could be drawn from any of the other definitions.

Figure 7-11 shows a plot of strain against packing fraction for the three simulations. Again, we see a very clear collapse, perhaps even better than the plot for strain rate. This will be developed in the following section.

7.6 The precise mechanism of shear dilation

Figure 7-9 suggested a strong correlation between strain rate and packing fraction in the three simulations, and this has been observed by other authors in the physics community. However, in figure 7-11, we saw a clear correlation between strain and

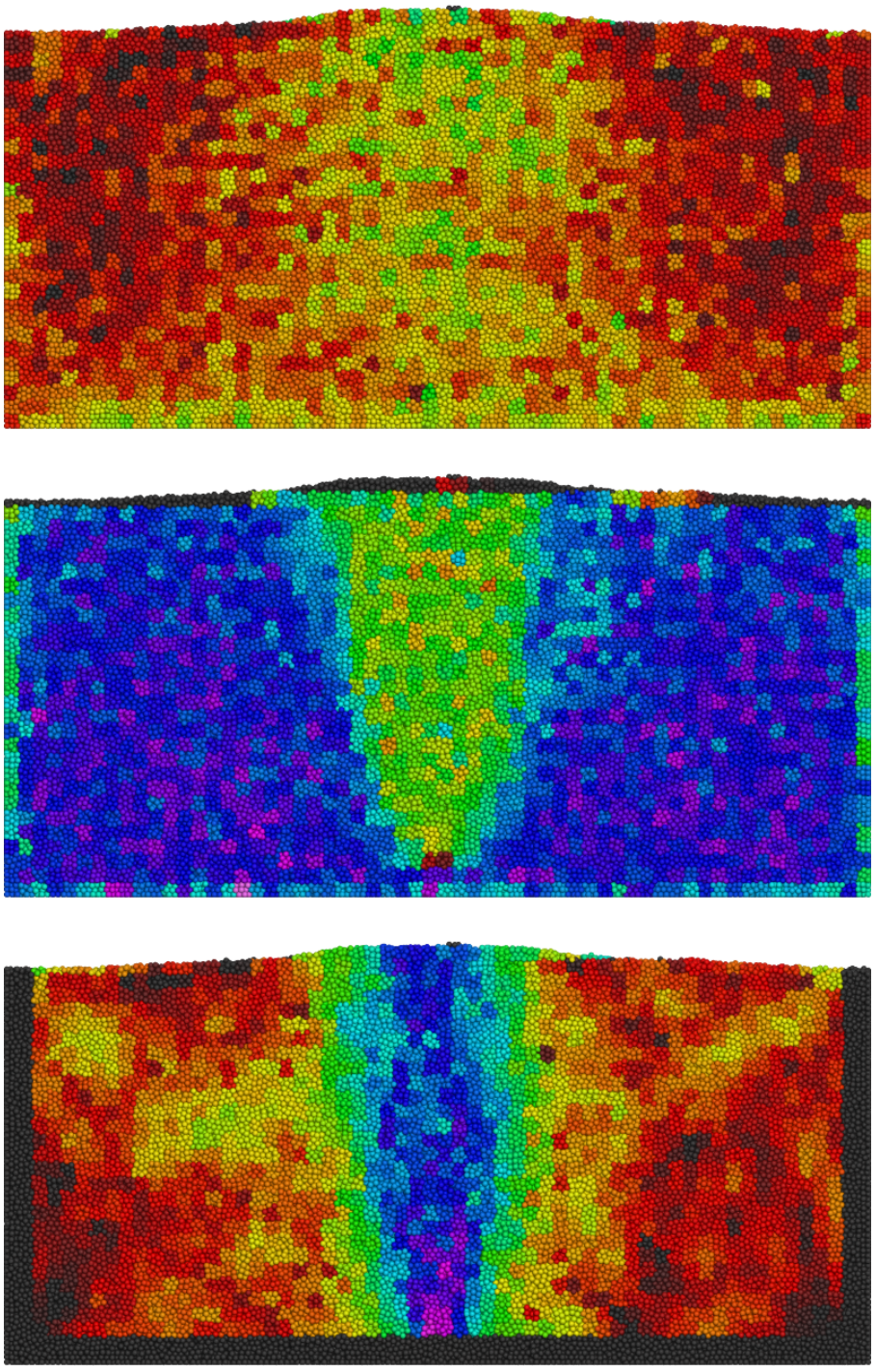


Figure 7-12: Three computed material quantities (μ, top; packing fraction ϕ, middle; magnitude of deviatoric strain rate $|\mathbf{D_0}|$, bottom) in the wide silo shearing simulation at $t = 200\tau$, using the color scheme in figure 7-3.

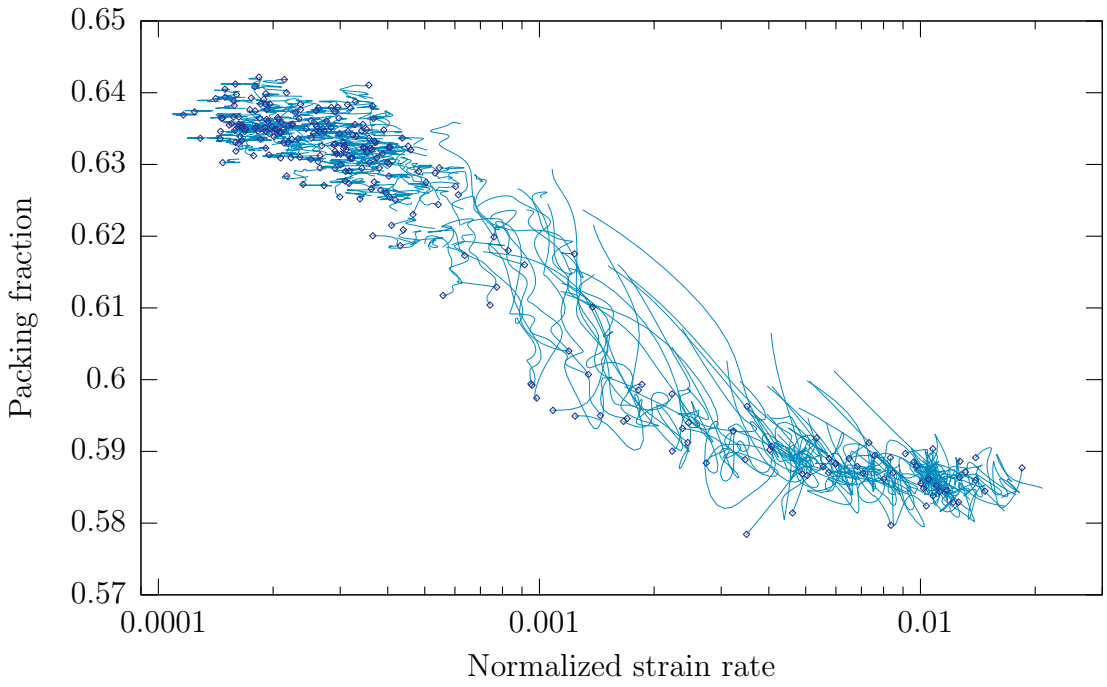


Figure 7-13: Lagrangian tracer plot for strain rate versus packing fraction for the long shearing simulation.

packing fraction, a point of view which has also been expressed in the literature, particularly from an engineering standpoint. Strain and strain rate are very different, and it appears that only one of these connections should exist. The situation is complicated by the fact for a single simulation snapshot, the regions of high strain may be loosely correlated with those of high strain rate.

The shearing experiment discussed in section 7.2 provides the ideal test environment for resolving this paradox. Since each granular element only moves in the y direction, it should experience a constant strain rate, while strain will be constantly increasing. Figure 7-12 shows snapshots of μ, packing fraction, and strain rate in this simulation. To infer long-time behavior, the simulation was run for ten times as long as the other runs, with snapshots saved at intervals of 2τ.

Figures 7-13 and 7-14 show plots of strain rate and strain against packing fraction. In these plots, a distinction has been made between the ends of the tracers (shown in dark blue), and the tracers themselves (shown in light blue). In the strain rate plot, we see that while th tracers are somewhat spread out, the end points are all on

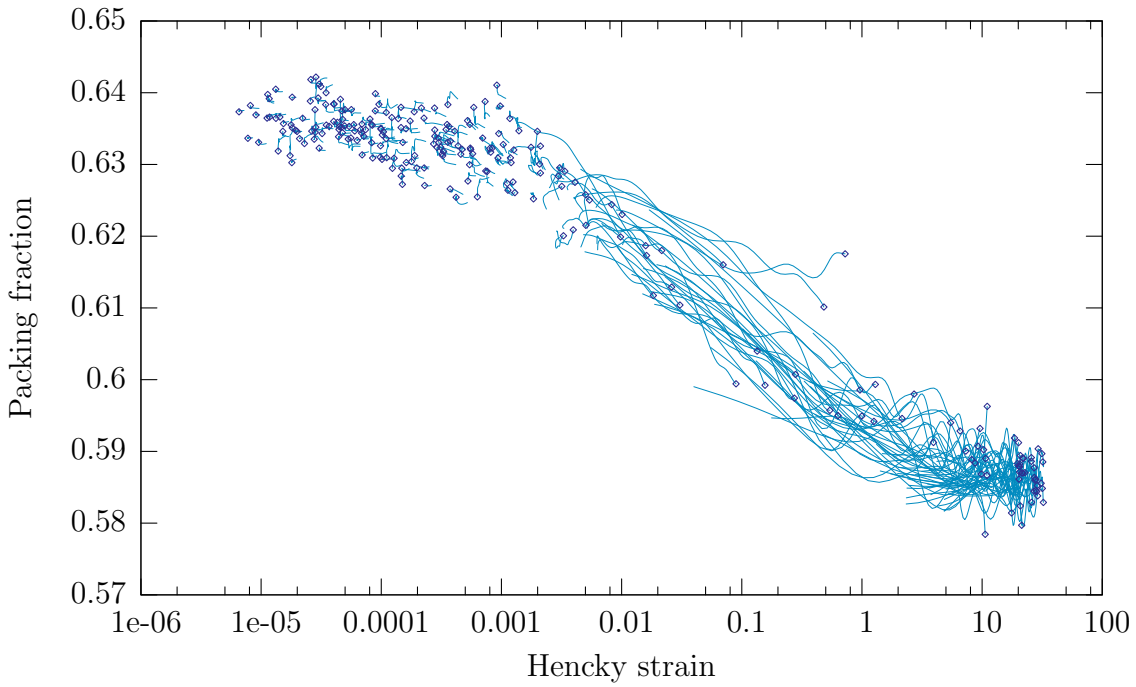


Figure 7-14: Lagrangian tracer plot for strain versus packing fraction for the long shearing simulation.

a fairly tight monotonically decreasing line. This suggests that strain rate may be instrumental in determining the granular material's final packing fraction: any granular element which experiences a constant strain rate for long enough will ultimately end up on this line.

By contrast, in the plot of strain against packing fraction, the tracers, and not the end points, appear to show the best collapse. Coupled with the previous data, this suggests that strain may be instrumental in the dilation process itself, while the final packing fraction will be set. In order for this to be consistent with the previous result, we would expect that some tracers at low strain rates would eventually peel off the main curve of tracers in the strain graph, once they had found their steady state. Testing for this is difficult, since one would have to run the simulation for many times longer. However there are two tracers in figure 7-14 which appear to be starting to peel away.

Looking at different shearing velocities in this system could provide more data to confirm this. Unfortunately, a Lagrangian tracer plot of strain rate against packing in

the drainage and pushing simulations is quite noisy and uninstructive, since it seems that few elements experience a constant shear rate for a long enough time to reach their steady state.

7.7 Conclusion

The above results show that is it possible to make a direct connection between continuum theories and particle-based simulations of dense granular flow at the local level. Motivated by previous work on co-operative particle motion, we have shown that it is possible to define an approximate granular element at the level of several particle diameters, and while material quantities computed at this scale exhibit statistical variation, they are clear enough to provide many insights into the rheology of granular materials.

We have been able to answer a variety of questions about how a granular element will behave, and since our conclusions have been shown for several different simulations, it is hoped that these properties are inherent in the granular material itself, and not tied to any particular geometry. We have tested several of the fundamental ideas behind Mohr-Coulomb plasticity theory, showing that coaxiality appears to be well-satisfied, while the Mohr-Coulomb incipient yield hypothesis appears invalid. However, we have shown that μ still plays a critical role, and we have explicitly demonstrated how it can control failure and dilation.

Consistent with the results of da Cruz [33], we have shown a link between strain rate and packing fraction. However, while their results were carried out in 2D, we have shown this result locally in 3D. Our results also show that while strain rate is important in determining the steady state which a packing will reach, the total strain plays an important role in determining the process by which the deformation takes place.

The above method provides us with an ideal tool for testing and developing continuum theories of granular materials. In the future we hope to directly examine recently proposed ideas in for plasticity theories, such as Pouliquen's flow rule [62].

Examining fully 3D situations may provide useful clues, since it may allow us greater control and variation in the stress tensor. In this work, we only considered yielding as a function of μ, which is defined in terms of the maximum and minimum stress eigenvalue. A useful study would be to check the role, if any, of the intermediate stress eigenvalue, a question which is currently unresolved.

We also hope to make direct connections to other continuum, non-plasticity, theories for granular materials such as STZ theory, or partial fluidization. These theories are frequently characterized by having a microscopic continuum quantity, such as STZ populations $n_{\pm}$ or partial fluidization parameter ρ, and part of the challenge in investigating these parameters will be to understand how they directly relate to the microscopic packing structure. Another promising and related avenue to pursue is to better understand the role of total strain. While strain appeared to play a crucial role in interpreting our data, it is an undesirable quantity to use in an eventual continuum theory of granular materials, since it is always tied to an initial reference state. While granular materials exhibit memory effects, it is also important that they are able to "forget" their initial reference state after a long period of rearrangement.

Chapter 8

Conclusion

The spot model was initially conceived as a model for particle diffusion, but in this thesis, we have shown that it has a much wider applicability. Coupled with relaxation dynamics, it can be used as the basis of a multiscale model of granular flow. It is a completely new type of model, and it has provided a point of view of granular materials which is unique among the current literature.

In evaluating the successes of the spot simulations, it is perhaps helpful to view them as comprised of two separate ideas: a kinematic description of the mean flow, coupled to the spot model microscopic motion. In terms of the description of the mean flow, the simulations presented here were a good match to DEM and experiment, but it has also been seen that there are some deficiencies. Gaussian velocity profiles which spread diffusively appear only to be approximation to the flow in these situations, and the kinematic description appears to violate some fundamental physical principles (such as frame indifference). However, it should still be viewed as surprising that a model with a single parameter motivated entirely by geometry could work so well, and despite its lack of physics, it provides better answers in some hopper flows than much more complicated models.

Perhaps the largest successes of the spot simulations are in its model of microscopic motion. Regardless of the precise details of the implementation, the concept of local, co-operative motion in dense granular flows appears to have very general validity. Whereas the study of static random packings has been considered by many authors,

this work represents the first theoretical model of a flowing random packing, and it has opened up the possibility of asking completely new questions about the nature of amorphous packings. The simulations suggest that in dense granular materials, geometry plays a crucial role, and it seems that any general theory should account for it. One of the failings of traditional plasticity approaches may originate in their continuum formulation, which does not take into account the way in which amorphous packings reorganize. This was one of the underlying principles of the Stochastic Flow Rule, which was explicitly shown in this thesis to provide a good match in two very different geometries.

The notion of strong geometrical constraints has also suggested that correct scale in which to look at physical quantities is the scale of the spot. In the final chapter of this thesis, we showed for the first time the notion of an explicit computational granular element. Using this we were able to directly test continuum hypotheses about granular materials, that would have been difficult to prove in experiment.

8.1 Future work

The spot model simulations presented here represent a new simulation technique for granular and amorphous systems, and as such there are a great number of possibilities for developing this work further. The current spot model based on random walks is suited to handle situations in industry which require fast, approximate answers about granular mixing, and one promising direction of research is to investigate ways in which the algorithm could be tuned and optimized. The details of the relaxation step were shown to be unimportant in creating the flowing random packings, and it would be useful to see if this step could be simplified while retaining the majority of the physical results. Possible modifications could be relaxing in a smaller radius, or relaxing only after several spot bulk displacements have occurred.

Other issues of practical importance could also be addressed. In this work, the spot centers were allowed to come within a distance of d of the wall, but tuning this parameter could create more realistic boundary layers. Tuning the biasing of the spot

motion could also affect the development of the avalanching at the free surface. Using ideas from the Stochastic Flow Rule, it should also be reasonably straightforward to create a spot simulation to describe the annular Couette geometry. A careful analysis of the spot motion and its relation to the exponentially decaying velocity profile could yield important physical insight. Adapting the spot model to handle polydisperse packings, or those with irregular shapes, could also be a promising direction. It is possible that the spot picture could show that some of the segregation behavior of polydisperse granular materials [120, 121, 69] has a geometrical origin.

Perhaps the most exciting aspect of the work presented here is in creating a general multiscale model of granular flow. One can envisage a simulation of particles in a granular packing whereby a continuum description of stresses is employed to generate macroscopic flow, but a relaxation model is used to correct geometric packing constraints at a microscopic level. A model of this type could exhibit feedback, whereby information about the amount of particle reordering is coupled back into the macroscopic description, creating a truly multiscale model for granular flow. Since the spot simulations do not require the complex inter-particle forces to be considered, such a model could be several orders of magnitude faster than DEM, providing a practical simulation technique for industry, while also yielding important insight into the basic physics of granular flow.

Appendix A

Miscellaneous void model and spot model calculations

A.1 Exact solution for a particle PDF in the void model

In chapter 2 it was shown that the PDF $\rho(x,z)$ of a particle diffusing in the void model follows the equation

$$\rho_z = 2b\frac{\partial}{\partial x}\left(\frac{\rho\eta_x}{\eta}\right) - b\rho_{xx}.$$

where $\eta(x,z)$ is the void probability density. In this section, we find the exact solution for a single particle in the presence of voids diffusing for a point orifice at the origin, corresponding to

$$\eta(x,z) = \frac{1}{\sqrt{4\pi bz}}e^{-x^2/4bz}. \tag{A.1}$$

The initial condition $\rho(x,z_0) = \delta(x-x_0)$ is used, corresponding to a particle initially located at (x,z). Two solutions are provided, the first using the method of characteristics, and the second using substitution. While the second method is more direct, the first method may have broader applicability. Although not addressed here, it appears that the first approach could handle situations where the two diffusion parameters

occurring in the PDE are different, as would occur in a spot model formulation.

A.1.1 Solution using the method of characteristics

From equation A.1, we see that

$$\frac{\eta_x}{\eta} = -\frac{x}{2by}$$

and therefore the equation for ρ becomes

$$z\rho_z = -bz\rho_{xx} - \frac{\partial}{\partial x}\left(\rho x\right).$$

To solve this equation, we first take the Fourier transform with respect to x, which yields

$$z\tilde{\rho}_z = -bz(ik)^2\tilde{\rho} - (ik)\left(i\frac{\partial}{\partial k}\right)\tilde{\rho},$$

$$z\tilde{\rho}_z - k\tilde{\rho}_k = bzk^2\tilde{\rho}.$$

This is a first order partial differential equation, so we can apply the method of characteristics. The characteristics are given by

$$\frac{dz}{z} = -\frac{dk}{k} = \frac{d\tilde{\rho}}{\tilde{\rho}bzk^2}$$

from which we obtain

$$kz = C \tag{A.2}$$

and

$$\frac{\partial\tilde{\rho}}{\partial k} = -b\tilde{\rho}kz = -b\tilde{\rho}C \qquad \Longrightarrow \qquad \tilde{\rho} = De^{-k^2zb}$$

for some constants C and D. Thus the general solution is

$$\tilde{\rho}(k,z) = F(kz)e^{-k^2zb}.$$

Taking the Fourier transform of the initial condition gives $\tilde{\rho}(k, y_0) = e^{-ikx_0}$ and hence

$$\begin{aligned} F(kz_0)e^{-k^2 z_0 b} &= e^{-ix_0 k} \\ F(kz_0) &= e^{-ix_0 k + k^2 z_0 b} \\ F(\lambda) &= \exp\left(\frac{\lambda^2 b - ix_0\lambda}{z_0}\right). \end{aligned}$$

Thus

$$\tilde{\rho}(k, z) = \exp\left(-bk^2 z\left(1 - \frac{z}{z_0}\right) - i\frac{x_0 k z}{z_0}\right)$$

and taking the inverse Fourier transform gives

$$\begin{aligned} \rho(x, z) &= \frac{1}{2\pi}\int_{-\infty}^{\infty} dk\, \exp\left(-bk^2\left(1 - \frac{z}{z_0}\right) + ik\left(x - \frac{x_0 z}{z_0}\right)\right) \\ &= \frac{1}{\sqrt{4\pi b z\left(1 - \frac{z}{z_0}\right)}} \exp \frac{-\left(x - \frac{x_0 z}{z_0}\right)^2}{4bz\left(1 - \frac{z}{z_0}\right)}. \end{aligned}$$

Thus we see that the probability density function of the particle position is a gaussian for all values of y, with mean and variance

$$\mu = \frac{x_0 z}{z_0}, \qquad \sigma^2 = 2bz\left(1 - \frac{z}{z_0}\right).$$

Although not considered here, this method of solution would potentially work in some situations where the two diffusion constants occurring in equation A.1 were not equal. In that case, the k occurring in equation A.2 would be taken to some power rather than occurring linearly.

A.1.2 Solution via substitution

The derivation of equation A.1 made use of an auxiliary quantity, $\sigma = \rho/\eta$, which satisfied

$$\sigma_z = -b\sigma_{xx},$$

representing a quantity diffusing in the opposite direction to the voids. It is therefore possible to find ρ by first solving the simpler equation for σ and then substituting back. The initial condition for ρ implies that

$$\sigma(x, z_0) = \sqrt{4\pi b z_0} e^{x_0^2/4bz_0} \delta(x - x_0)$$

which gives

$$\begin{aligned}
\sigma(x,z) &= \sqrt{\frac{z_0}{z_0 - z}} \exp\left(\frac{x_0^2}{4bz_0} - \frac{(x-x_0)^2}{4b(z_0-z)}\right) \\
&= \sqrt{\frac{z_0}{z_0 - z}} \exp\left(\frac{x_0^2 z_0 - x_0^2 z - x^2 z_0 + 2x x_0 z_0 - x_0^2 z_0}{4bz_0(z_0-z)}\right) \\
&= \sqrt{\frac{z_0}{z_0 - z}} \exp\left(\frac{-x_0^2 z - x^2 z_0 + 2x x_0 z_0}{4bz_0(z_0-z)}\right).
\end{aligned}$$

Hence

$$\begin{aligned}
\rho(x,z) &= \eta(x,z)\sigma(x,z) \\
&= \frac{1}{\sqrt{4\pi bz\left(1 - \frac{z}{z_0}\right)}} \exp\left(\frac{-x_0^2 z - x^2 z_0 + 2x x_0 z_0}{4bz_0(z_0-z)} - \frac{x^2}{4bz}\right) \\
&= \frac{1}{\sqrt{4\pi bz\left(1 - \frac{z}{z_0}\right)}} \exp\left(\frac{-x_0^2 z - x^2 z_0^2 + 2x x_0 z z_0}{4bz_0 z(z_0-z)}\right) \\
&= \frac{1}{\sqrt{4\pi bz\left(1 - \frac{z}{z_0}\right)}} \exp \frac{-\left(x - \frac{x_0 z}{z_0}\right)^2}{4bz\left(1 - \frac{z}{z_0}\right)}
\end{aligned}$$

which matches the solution from the previous method. This method of solution can be extended to handle arbitrary void and particle densities, since it only ever requires the solution of a simple diffusion equation.

A.2 Two different interpretations of the spot density drop

Suppose we consider a monodisperse granular materials composed of spherical particles with volume V_p, at a packing fraction ϕ. We consider spots, which when displaced by an amount $\mathbf{\Delta r_s}$ cause motions of particles in the opposite direction of size $\mathbf{\Delta r_p} = -w\mathbf{\Delta r_s}$. As described in chapter 2, if a spot influences N particles, then it makes sense to attribute a volume of

$$V_s = NwV_p$$

to it. In his original description of the spot model [17], Bazant proposed that it was possible to get an estimate of the size of w by looking at the drop in packing fraction $\Delta\phi$ that occurs when a granular material goes from a static state to flowing state. Bazant proposed the formula

$$w = \frac{\Delta\phi}{\phi^2}. \tag{A.3}$$

and by attributing $\Delta\phi/\phi = 1\%$ to the presence of a single spot, arrived at an estimate of $w \approx 0.017$. Attributing the density drop to the presence of several spots which are overlapped would lower the estimate of w, and thus a range of $w = 10^{-2}$ to $w = 10^{-3}$ seemed appropriate.

The justification for equation A.3 comes from considering a region of particles of volume V_0, which would be precisely filled by a single spot, and would contain N particles. Note that $V_0 \neq V_s$, since the former is the volume occupied by a spot, whereas the latter is a measure of the physical influence carried by the spot, and typically $V_s \ll V_0$. Initially, we have

$$\begin{aligned} \phi &= \frac{\text{Volume occupied by particles}}{\text{Total volume}} \\ &= \frac{NV_p}{V_0} \end{aligned} \tag{A.4}$$

Bazant proposed that when a spot enters the region it expands by the spot's vol-

ume. The total volume increases by V_s, while the amount of volume occupied by the particles remains the same. Thus

$$\begin{aligned}
\phi - \Delta\phi &= \frac{\text{Volume occupied by particles}}{\text{Total volume}} \\
&= \frac{NV_p}{V_0 + V_s} \\
&= \frac{NV_p}{V_0}\left(1 + \frac{V_s}{V_0}\right)^{-1} \\
&\approx \frac{NV_p}{V_0}\left(1 - \frac{V_s}{V_0}\right) \\
&= \frac{NV_p}{V_0} - \frac{NV_pV_s}{V_0^2} \\
&= \phi - \frac{\phi(NwV_p)}{V_0} \\
&= \phi - w\phi^2
\end{aligned}$$

from which equation A.3 directly follows.

Bazant's approach has the advantage that one can think directly about how a cluster of N particles will expand when a spot is introduced. However the current author believes that one step of the above argument is inappropriate. When the spot is introduced, the total volume is increased by V_s. The current author feels that it is more appropriate to say that when a spot is introduced, the particle volume is decreased by V_s, since when the spot volume was defined, it was directly in terms of how it would displace the particle volume in the opposite direction, and not in terms of the interstitial space.

Using this argument, equation A.4 still holds, but we have

$$\begin{aligned}
\phi - \Delta\phi &= \frac{\text{Volume occupied by particles}}{\text{Total volume}} \\
&= \frac{NV_p - V_s}{V_0} \\
&= \frac{NV_p}{V_0} - \frac{V_s}{V_0} \\
&= \phi - \frac{NwV_p}{V_0} \\
&= \phi - w\phi
\end{aligned}$$

and hence

$$w = \frac{\Delta\phi}{\phi} \tag{A.5}$$

which differs from Bazant's result by a factor of ϕ.

It is reasonable to suggest that the details of the process by which a spot enters a packing will have an effect on the amount of dilation, and thus it could be argued that both approaches could be valid under different physical assumptions. However, it is possible to construct an argument where purely by considering a spot moving in the bulk of a granular material, one can obtain a result which is consistent with A.5.

To do this, it is helpful to consider spots having the shape of a cube with side length L, as there is nothing in the above volume arguments which requires that spots be spherical. Consider a box with dimensions $2L \times L \times L$ which is initially filled with particles at a uniform packing fraction ϕ. The box is divided into a left half and a right half. Imagine that spot is initially located in the right side, and that it is displaced a distance L into the left of the box. The particles will be displaced to the right by a distance wL, and thus a box of particles of dimensions $wL \times L \times L$ is displaced from the left half of the box to the right half. Before the spot moves, the packing fraction in the left half of the box is given by

$$\begin{aligned} \phi &= \frac{\text{Volume occupied by particles}}{\text{Total volume}} \\ &= \frac{NV_p}{L^3} \end{aligned}$$

and after the spot moves, the packing fraction in the left half of the box is given by

$$\begin{aligned} \phi - \Delta\phi &= \frac{\text{Volume occupied by particles}}{\text{Total volume}} \\ &= \frac{NV_p - \phi L^3 w}{L^3} \\ &= \phi - w\phi. \end{aligned}$$

This argument was created purely by considering a spot motion in the bulk, with no ambiguities about how a spot would enter at a free surface. As the spot moves to

the left of the box, we see that is displaces a volume of V_s particle volume out, rather than introducing V_s interstitial space. We thus conclude that equation A.5 is more appropriate.

For the current purposes, where we wish to gain an order of magnitude estimate for w, the difference between equations A.3 and A.5 is not that significant, since the additional ϕ term is an order 1 quantity. However, the result does suggest that it is more appropriate to think of spots as carrying negative particle volume, as opposed to extra interstitial space, which is an important physical distinction.

Appendix B

Numerical solution of the Kinematic Model in the cylindrical reactor geometry

In the Kinematic Model for drainage the vertical downward velocity u in the container is assumed to follow a diffusion equation of the form

$$\frac{\partial v}{\partial z} = b\nabla_{\perp}^{2} v$$

where $\nabla_{\perp}^{2}$ is the horizontal Laplacian. By exploiting the axial symmetry, v can be treated as a function of z and r only. In cylindrical coordinates the Laplacian is

$$\begin{aligned} \frac{\partial v}{\partial z} &= b\frac{1}{r}\frac{\partial}{\partial r}\left(r\frac{\partial v}{\partial r}\right) \\ &= b\frac{\partial^2 v}{\partial r^2} + b\frac{1}{r}\frac{\partial v}{\partial r}. \end{aligned}$$

The radial velocity component is given by

$$u = b\frac{\partial v}{\partial r}$$

and by enforcing that the velocity field at the wall must be tangential to the wall, we can obtain boundary conditions for solving v.

To solve the above equation in a cylinder is straightforward, since we can make use of a rectangular grid. The boundary condition reduces to $v_r = 0$ at the wall. However, to solve this equation in the reactor geometry, we must also consider the complication of the radius of the wall, R, being a function of z. To ensure accurate resolution in the numerical solution of v at the wall, we introduce a new coordinate $\lambda = r/R(z), \eta = z$, which then allows us to solve for u over the range $0 < \lambda < 1$. Under this change of variables, the partial derivatives transform according to

$$\begin{aligned} \frac{\partial}{\partial r} &= \frac{1}{R(\eta)}\frac{\partial}{\partial \lambda} \\ \frac{\partial}{\partial z} &= \frac{\partial}{\partial y} - \frac{\lambda R'(\eta)}{R(\eta)}\frac{\partial}{\partial \lambda}. \end{aligned}$$

In the transformed coordinates

$$R^2 v_\eta = \frac{b}{\lambda} v_\lambda + b v_{\lambda\lambda} + \lambda R R' v_\lambda.$$

To ensure differentiability at $r = 0$, we use the boundary condition

$$\left.\frac{\partial v}{\partial \lambda}\right|_{\lambda=0} = 0, \tag{B.1}$$

and by ensuring zero normal velocity at the wall we find that

$$\left.\frac{\partial v}{\partial \lambda}\right|_{\lambda=1} = -\frac{v R' R}{b}. \tag{B.2}$$

To numerically solve this partial differential equation, we make use of the implicit Crank-Nicholson integration scheme. We write $v_j^n = v(j\Delta\lambda, n\Delta\eta)$, and solve in the range $j = 0, 1, \ldots, N$ where $N = \Delta\lambda^{-1}$. Away from the end points, the Crank-

Nicholson scheme tells us that

$$\begin{aligned}\frac{v_j^{n+1}-v_j^n}{\Delta\eta} &= \frac{b}{2\Delta\lambda^2R^2}\big(v_{j+1}^{n+1}-2v_j^{n+1}+v_{j-1}^{n+1}+v_{j+1}^n\\ &\quad -2v_j^n+v_{j-1}^n\big)+\left(\frac{b}{4j\Delta\lambda^2R^2}+\frac{jR'}{4R}\right)\\ &\quad \times\big(v_{j+1}^{n+1}-v_{j-1}^{n+1}+v_{j+1}^n-v_{j-1}^n\big),\end{aligned}$$

where all references to R and R' are evaluated at $\eta=\Delta\eta(j+\frac{1}{2})$. If $j=0$, then by reference to equation B.1, we find that

$$\frac{v_0^{n+1}-v_0^n}{\Delta\eta}=\frac{b}{\Delta\lambda^2R^2}\left(v_1^{n+1}-v_0^{n+1}+v_1^n-v_0^n\right).$$

Similarly, for $j=N$, by reference to equation B.2, we see that effectively

$$\frac{v_{N+1}^n-v_{N-1}^n}{2\Delta\lambda}=-\frac{v_N^nR'R}{b}$$

and hence

$$\begin{aligned}\frac{v_N^{n+1}-v_N^n}{\Delta\eta} &= \frac{b}{\Delta\lambda^2R^2}\left(v_{N-1}^{n+1}-v_N^{n+1}+v_{N-1}^n-v_N^n\right)\\ &\quad -\left(\frac{(2N+1)R'}{2R}+\frac{R'^2}{2b}\right)\left(v_N^{n+1}+v_N^n\right).\end{aligned}$$

If we write $\mathbf{v}^n=(v_0^n,v_1^n,\ldots,v_N^n)^T$, then the above numerical scheme can be written in the form $S\mathbf{v}^{n+1}=T\mathbf{v}^n$ where S and T are tridiagonal matrices; this system can be efficiently solved by recursion in $O(N)$ time. The above scheme was implemented in C++, and gives extremely satisfactory results, even with a relatively small number of gridpoints.

Appendix C

The spot model simulation code

C.1 Overview

This appendix documents the computer codes that were developed to carry out the spot model simulations in chapter 3. Although the simulations presented in this thesis concentrated wholly on the case of granular drainage, the spot model microscopic mechanism introduced in figure 3-2 could potentially be applied to many other situations, and it was therefore chosen to develop the code in a method that could be easily adapted to other purposes.

The code is written as a library of routines in object-oriented C++, that aims to provide a flexible environment for the general task of managing particles in a container. Routines exist for creating particles in the container, outputting a snapshot of particles to a file, and carrying out diagnostic tests (such as the calculation of the radial distribution function). Most importantly, the library provides routines to carry out the spot motion and elastic relaxation.

Since simulations of granular materials involve a large number of particles, efficiency was a key consideration in the development of the library. The simplest method of storing a total of N particle positions would be as a single, unsorted list. However, finding all the particles influenced by a single spot would require looking over the entire list, which would be an $O(N)$ calculation. To avoid this, the library divides the simulation up into a rectangular grid of regions, and each region keeps a list of

all particles within it. When applying a spot motion, only the particles in regions overlapping the spot need to be tested, significantly reducing the computational complexity. This arrangement of data does bring with it some computational challenges. For example, when a spot is moved, the code must first find which regions to test, and it must also deal with the possible migration of particles within regions. However, these computational difficulties can be packaged away as low-level routines in the library. Note that the neighbor list technique [6], popular with in other molecular simulations, is not employed.

In the following section, the class structure and code organization are discussed. In section C.3 the individual routines are described in detail. Section C.4 provides two simple examples of using the routines in the code, and section C.6 contains the code listings.

C.2 Code structure

The code presented in this chapter is a trimmed-down version of the original library, which concentrates on the core routines only. For reasons of clarity and economy, routines that were either experimental, or created for a specific purpose, have been omitted. The version of the library presented here consists of four files: "vec.hh", "vec.cc", "container.hh", and "container.cc" which are listed in subsections C.6.1 to C.6.4.

C.2.1 Vector manipulation

Since the spot model simulation requires frequent manipulation of vectors, a simple vector structure named `vec` was created, which is listed in "vec.hh" and "vec.cc". This structure consists of three floating point numbers `x`, `y`, and `z` to represent three coordinates, which can be accessed and set independently. A simple example code would be:

```
#include <cstdio>
#include <iostream>
```

```
using namespace std;            // Standard C++ header

#include "vec.cc"               // Load the structure

int main() {
  vec a;                        // Create two vectors
  a.x=3;a.y=4;a.z=5;            // Set coordinates
  a.z-=4;                       // Subtract 4 from z co-ord

  cout c.x << " " << c.y;
  cout << " " << c.z << endl;   // Output to screen
}
```

A powerful feature of the C++ language is the ability to "overload" the conventional arithmetic operators (`+`, `-`, `*`, `/`) when they are applied to new structures. Routines were therefore constructed so that these operators took on their usual meaning when applied to `vec` objects. In addition simple routines were created to carry out typical vector operations, such as creating a scalar product, or finding the magnitude of a vector. The code listing below provides some examples:

```
#include <cstdio>
#include <iostream>
using namespace std;            // Standard C++ header

#include "vec.cc"               // Load the structure

int main() {
  vec a(1,2,3),b(2,3,1),c;      // Create two vectors
  float d,e;                    // Initialize two numbers
  c=a+b;                        // Add a and b together
  c/=3;                         // Divide the result by 3
  d=sp(a,b);                    // Scalar product of a and b
  e=radsq(a);                   // Magnitude squared of a

  cout c.x << " " << c.y << " " << c.z << endl;
  cout d << " " << e << endl;   // Output to screen
}
```

The spot model code makes extensive use of the `vec` structure. It considerably simplifies the source code, and lowers the chance of introducing the common coding error of accidentally transposing `x`, `y`, and `z` in vector manipulations. It also provides a convenient method of passing vectors (or references to them) between functions.

All low level vector operations are specified using the `inline` command to improve performance.

C.2.2 The container class hierarchy

The files "container.cc" and "container.hh" contain the bulk of the spot model library. There are three main classes that form a hierarchy. At the lowest level is the `particle` structure, containing all the data associated with a single particle. In the code presented here, this contains a position vector `v` and a numerical label `n`, but in a general case it could contain other information, such as the particle diameter, mass, or rotation characteristics.

At the middle layer is the `region` class, which represents a particular box-shaped region of particles in the container. This contains an fixed-size array of particle elements, and several low-level routines for accessing and handling the particles.

The `container` class is at the top layer, and it is the most important for running a simulation. It represents a box-shaped container of particles, and it has a constructor of the form:

```
container::container(float minx, float maxx, float miny,
                     float maxy, float minz, float maxz,
                     int xn, int yn, int zn);
```

The first six arguments set the bounds of the container in each coordinate. The following three arguments set how many regions in each direction the container is to be subdivided into. When a container object is created, it initializes an array of `xn*yn*zn` regions. The container object has a large number of functions associated with it, which are described in detail in the following section.

C.3 Spot model library commands

C.3.1 Particle input

```
void put(int n,vec &p);
void put(int n,float x,float y,float z);
```

This function adds a particle to a container. Two versions of the routine are provided, with one accepting a vector and the other accepting three individual coordinates. The code finds the region which contains this position and gives the particle to that region.

```
void import();
```

This function reads in particle positions from the standard input. Each particle is represented by one line containing four numbers separated by spaces. The first number is the particle's numerical label, and the following three following three are the particle coordinates. The code sorts the particles into the correct spatial regions.

```
void importrestart(fstream &rf);
```

This function reads in a snapshot of particle positions from an open file stream `rf` that are stored in the GranFlow snapshot format. The code sorts the particles into the correct spatial regions.

```
void clear();
```

This function clears the container of particles, by zeroing the particle counts in all the regions.

C.3.2 Particle output

```
void dump(char *filename);
void dumpraster(char *filename);
void dumppov(char *filename);
```

These functions dump a snapshot of all the particles in the container to a file. The `dump` routine outputs a text file with one line of information about each particle. The `dumpraster` and `dumppov` create input files for the Raster3D and POV-Ray raytracers respectively.

```
void dumprestart(int t, fstream &of);
```

This function writes a snapshot of all particle positions in the GranFlow snapshot format to an open file stream `of`. The timestep in the GranFlow header is given by the integer `t`.

```
void regionshot(float minx,float maxx,float miny,
                float maxy,float minz,float maxz);
```

This function outputs all the particle positions within a specific subregion of the container to the standard output.

C.3.3 Spot motion and elastic relaxation

```
int count(vec &p,float r);
```

This function returns the number of particles that are within a distance `r` of the vector `v`. It is useful for telling whether a spot is still within the particle packing. To carry this out, the code first finds which regions the test area overlaps with, and then tests all the particles within those regions.

```
void spot(vec &p,vec &v,float r);
```

This function carries out a spot displacement `v` on all particles which are within a distance `r` of position `p`. The code first calculates which regions the test area overlaps with, and then tests all the particles within those regions, moving the particles which are within range. If a particle is within range, it is moved. If a particle's new position is no longer within its original region, then it is migrated to the region it now occupies. The affected regions are tested in such an order to ensure that any particles which migrate always do so to a region which was already tested. In a one-dimensional situation, this would mean that if the displacement vector pointed to the left, then the regions would be tested from left to right. This avoids the possibility of a migrated particle being displaced more than once.

```
void spotg(vec &p,vec &v,float r,float l);
```

This function carries out a Gaussian-smoothed spot displacement centered on a position `p`. Particles which are a distance `s` away from `p` experience a displacement of `v*exp(-s*s/(2*l*l))`. Particles outside a cutoff radius `r` experience no displacement.

```
void relaxslow(vec &p,float r,float s
               float force,float damp,int steps);
```

This function applies an elastic relaxation on a sphere of radius `s`, centered at `p`. Particles with radii between `s` and `r` fixed to avoid long range disruption. The magnitude of the correcting force is given by `force`. After each step, a velocity damping of magnitude `damp` is applied. Vertical walls also contribute a force, but the container bottom does not, to prevent jamming at the orifice. Using pointers, particle positions are updated in situ and never copied out. A final step is applied to move any particles that moved into a different region – this is done in reverse to prevent memory conflicts. Memory for the buffers is declared as static to prevent unnecessary memory reallocation each time the routine is called.

```
void relax(vec &p,float r,float s,
           float force,float damp,int steps);
```

This function carries out exactly the same task as `relaxslow`, except that during the relaxation process, the particles are spatially sorted into an $n \times n \times n$ grid. This significantly speeds up the calculation, since to find the neighbors of a particle requires a search over only a small subregion of the grid, and not a loop over all the affected particles. The value of n can be set in the code by declaring the quantity `rgrid`. The default value is `rgrid=6`, which was found to give the best results in the hopper drainage simulations considered here. The quantity `rbuf` sets the memory allocation for each grid element, and the default value is 80. The results of this function should agree with the results of `relaxslow` up to rounding error.

C.3.4 Diagnostic routines

```
void regioncount();
```

This function prints a list of the regions and the number of particles that each contains.

```
void gofr(int *b);
```

This command calculates the frequency distribution of interparticle separations in the entire container. The results for separations from radii 0 to $8d$ and stores the results in a 4,000 element matrix `b`. Calculating $g(r)$ requires normalizing the resulting distribution by an appropriate factor. In an infinite container, the normalizing factor is proportional to r^2, although for constrained geometries it is more complicated.

```
void bondangle(int *b,float r);
```

This function calculates the bond angle distribution in the entire container, based on defining two particles as neighboring if their centers are a distance `r` apart. All triplets of particles are considered, and the results are stored in a 4,000 element matrix `b`.

```
void spaces(float xs,float ys,float zs,
            float mnz,float mxz,int *b);
```

This function computes the free space size distribution in the strip of the container in the range `mnz<z<mxz`. The region is covered by an `xs` by `ys` by `zs` grid, and at each point in the grid which is not inside a particle, the maximum radius of void that will fit at that location is computed. The frequency distribution of radii over the range 0 to $1.5d$ is stored in a 300 element array `b`.

C.4 Example: A test of the relaxation scheme

The code listed in subsection C.6.5 was written as a simple test of the spot model library. It creates a container object using dimensions $-25d < x < 25d$, $-4d < y <$

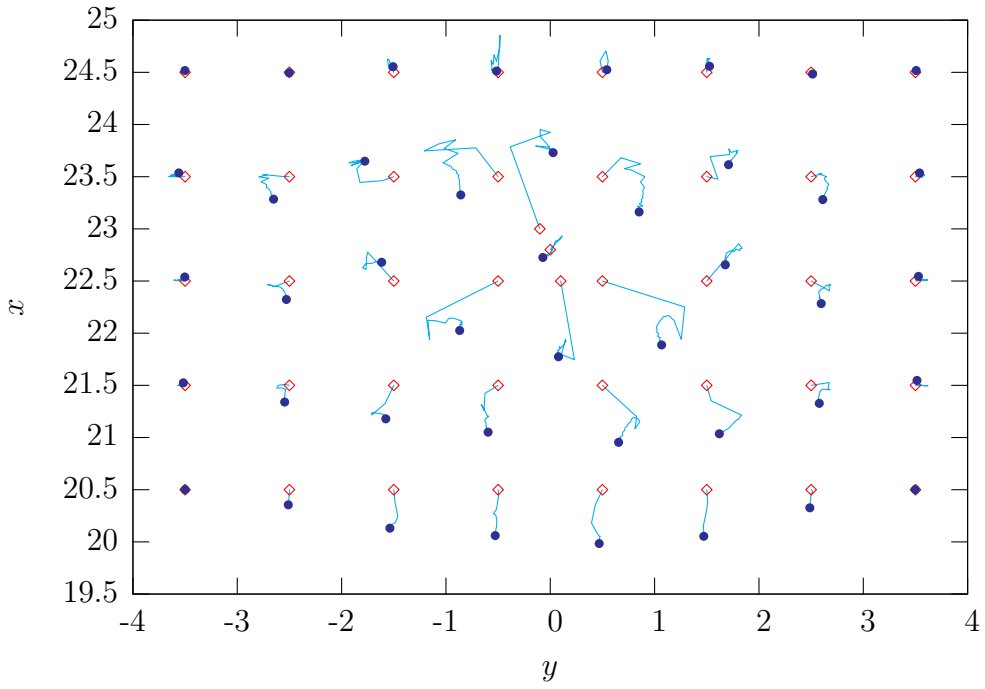


Figure C-1: Results of the "relaxtest.cc" example program. A rectangular grid of particles is initialized, along with three extra at $(23, -0.1, 10)$, $(22.8, 0, 10)$, and $(22.5, 0.1, 10)$. Their initial positions are shown by the red diamonds. A twenty step relaxation process is carried out, shown by the light blue lines, until the particles reach their final positions, shown by the dark blue circles.

$4d$, and $0 < z < 150d$, and then introduces a two dimensional rectangular grid of particles in the (x, y) plane at $z = 10d$. Three extra particles are then positioned at $(23, -0.1, 10)$, $(22.8, 0, 10)$, and $(22.5, 0.1, 10)$. A twenty step relaxation process is then carried out, using `force=0.7` and `damp=0.4`. The results of the relaxation process after n steps are outputted to text files named `relax`n, where $n = 0, 1, \ldots, 20$.

Figure C-1 shows the initial particle positions, final particle positions, and the traces of the particle trajectories. The particles in the square lattice are pushed apart to accommodate for the other particles. After twenty steps, the maximum overlap of particles in the grid is $0.06d$.

C.5 Example: A spot model simulation code

The spot simulation code that was used to generate the results of chapter 3 is listed in subsection C.6.6. First, standard header files and the spot model library are loaded. In the next section, all the free parameters are defined. Four of these (the spot insertion rate, the spot move rate, the displacement scale factor, and the spot radius) directly relate to the calibrated parameters. In addition to these, the parameters `xst`, `yst`, and `zst` set the size of the random walk step, to ensure that the walk has the correct diffusion parameter. The parameter `clu` sets the number of spots that are introduced at each insertion event; in the current simulation, only one spot was introduced at each time, but this allows for the possibility of introducing spots in groups. The parameter `max` sets the size of the array for the spot positions, and is chosen to be large enough to handle when the simulation reaches steady state, and the number of spots being inserted is balanced by the number of spots exiting at the free surface. The parameters `xbu` and `ybu` affect how close the spot centers are allowed to come to the side walls, and can be used to adjust the boundary layer behavior. After these declarations, the code defines a simple function `rnd()` which returns a floating point number in the range 0 to 1.

The main program then starts, and after declaring some variables, the code initializes the container object, with size $-25d < x < 25d, -4d < y < 4d, 0 < z < 150d$, divided into a $10 \times 3 \times 30$ grid of regions. Particle positions from a DEM simulation are then imported from the standard input. An output file stream is then open.

The main loop of the simulation then starts. The simulation runs from `t=0` to `t=500`, and outputs a snapshot every two time units. The simulation is event driven, with insertion events happening at a rate `ipr`, and movement of existing spots happening at a rate of `mpr`. Based on there being `n` spots in the container, the code first calculates the time to the next event, by sampling from an exponential distribution with rate `ipr+n*mpr`. If a snapshot needed to be recorded between this event and the previous one, then it is done so here.

Based on the number of spots in the container, the code then randomly decides

whether the current event is an insertion or a move. To begin with, when there are no spots in the container, the event will always be an insertion. The code then initializes `clu` new spots at the orifice, storing their positions in the vector array `s`. Their z coordinate is set to `-rad/3`, and their x and y coordinates are chosen uniformly in the range from $-1.5d$ to $1.5d$.

If a movement event occurs, then the code first chooses an existing spot at random to be moved. If its z coordinate is less than `rad/3`, it is treated as being in the orifice it moves upwards only by `zst`, with no horizontal motion. It is first moved half of the way, and then a spot motion and relaxation are applied, centered on this intermediate position. After this motion, the spot is moved the rest of the way.

If the z coordinate of the moving spot is bigger than `rad/3` then it is treated as being in the bulk of the container, undergoing the random walk. The displacement of the spot is stored in `v` and is chosen randomly from has four possible directions: it can either take a positive or negative step in the x direction, or a positive and negative step in the y direction. If the motion would take the spot outside its range, then the displacement vector is truncated as necessary. The spot is then moved half-way, its influence is applied, and it is moved the rest of the way.

When the spot's vertical coordinate reaches $130d$, the spot is deleted. This is done by the command `s[b]=s[--n]`, which lowers the number of spots by one, and copies the last element of the array over the spot being deleted, to ensure that the remaining spots form a contiguous block of memory.

C.6 Spot model code listings

C.6.1 vec.hh

```
// Fast spot model code — vector object header file
//
// Author  : Chris H. Rycroft
// Version : 2.0
// Date    : July 26th 2004

#ifndef VEC_HH
#define VEC_HH
```

```
class vec {
  public:
    float x,y,z;
    vec () {};
    inline vec (float a, float b, float c);
    inline vec operator+ (vec p);
    inline vec operator- (vec p);
    inline vec operator* (float a);
    inline vec operator/ (float a);
    inline void operator+= (vec p);
    inline void operator-= (vec p);
    inline void operator*= (float a);
    inline void operator/= (float a);
    inline vec operator* (vec a);
};

struct vptr {
  int n;
  vec *q;
};

struct intrec {
  int lx,ly,lz,ux,uy,uz;
};

inline float radsq(vec &r,vec &s);
inline float radsq(vec &r);
inline float sp(vec &r,vec &s);
inline float stp(vec &r,vec &s,vec &t);

#endif
```

C.6.2 vec.cc

```
// Fast spot model code - vector object routines
//
// Author  : Chris H. Rycroft
// Version : 2.0
// Date    : July 26th 2004

#ifndef VEC_CC
#define VEC_CC

#include "vec.hh"

inline vec::vec (float a,float b,float c) {
   x=a;y=b;z=c;
};

inline vec vec::operator+ (vec p) {
  vec t;
  t.x=x+p.x;t.y=y+p.y;t.z=z+p.z;
  return t;
};

inline vec vec::operator- (vec p) {
  vec t;
  t.x=x-p.x;t.y=y-p.y;t.z=z-p.z;
  return t;
};

inline vec vec::operator* (float a) {
  vec t;
  t.x=x*a;t.y=y*a;t.z=z*a;
  return t;
```

```
};

inline vec vec::operator/ (float a) {
  vec t;
  t.x=x/a;t.y=y/a;t.z=z/a;
  return t;
};

inline void vec::operator+= (vec p) {
  x+=p.x;y+=p.y;z+=p.z;
};

inline void vec::operator-= (vec p) {
  x-=p.x;y-=p.y;z-=p.z;
};

inline void vec::operator*= (float a) {
  x*=a;y*=a;z*=a;
};

inline void vec::operator/= (float a) {
  x/=a;y/=a;z/=a;
};

inline float radsq(vec &r,vec &s) {
  return (r.x-s.x)*(r.x-s.x)+(r.y-s.y)*(r.y-s.y)+(r.z-s.z)*(r.z-s.z);
};

inline float radsq(vec &r) {
  return r.x*r.x+r.y*r.y+r.z*r.z;
};

inline float sp(vec &r,vec &s) {
  return r.x*s.x+r.y*s.y+r.z*s.z;
};

inline vec vec::operator* (vec a) {
  vec t;
  t.x=y*a.z-z*a.y;t.y=z*a.x-x*a.z;t.z=x*a.y-y*a.x;
  return t;
};

inline float stp(vec &r,vec &s,vec &t) {
  return r.x*s.y*t.z+r.y*s.z*t.x+r.z*s.x*t.y-r.z*s.y*t.x-r.y*s.x*t.z-r.x*s.z*t.y;
};

#endif
```

C.6.3 container.hh

```
// Fast spot model code - container object header file
//
// Author  : Chris H. Rycroft
// Version : 2.0
// Date    : July 26th 2004

#ifndef CONTAINER_HH
#define CONTAINER_HH

#ifndef maxpoints
#define maxpoints 100
#endif

struct particle {
  vec p;
```

```
  int n;
};

struct particlev {
  particle data;
  vec v;
};

struct particlevdiag : particlev {
  vec op;
};

class region {
  public:
    region();
    void dump(fstream &of);
    void dumpraster(fstream &of);
    void dumppov(fstream &of);
    void dumprestart(fstream &of);
    void put(int n,vec &p);
    particle data[buf];
    int num;
};

class container {
  public:
    container(float minx,float maxx,float miny,float maxy,
              float minz,float maxz,int xn,int yn,int zn);
    void dump(char *filename);
    void dumpraster(char *filename);
    void dumppov(char *filename);
    void dumprestart(int t, fstream &of);
    void import();
    void importrestart(fstream &rf);
    void regioncount();
    void regionshot(float minx,float maxx,float miny,float maxy,float minz,float maxz);
    void put(int n,vec &p);
    void put(int n,float x,float y,float z);
    void spot(vec &p,vec &v,float r);
    void spotg(vec &p,vec &v,float r,float l);
    int count(vec &p,float r);
    void relax(vec &p,float r,float s,float force,float damp,int steps);
    void relaxslow(vec &p,float r,float s,float force,float damp,int steps);
    void gofr(int *b);
    void bondangle(int *b,float r);
    void spaces(float xs,float ys,float zs,float mnz,float mxz,int *b);
    void clear();
    region *reg;
  private:
    float xmin,xmax,ymin,ymax,zmin,zmax;
    float xsp,ysp,zsp;
    int nx,ny,nz;
    int blocks;
    inline int myint(float co);
    inline intrec boundbox(vec &p,float r);
};

#endif
```

C.6.4 container.cc

```
// Fast spot model code — container object routines
//
// Author  : Chris H. Rycroft
// Version : 2.0
```

```
// Date     : July 26th 2004

#ifndef CONTAINER_CC
#define CONTAINER_CC

#include "container.hh"

#ifndef rgrid
#define rgrid 6
#endif

#ifndef rbuf
#define rbuf 80
#endif

// Region constructor - sets initial number of particles to zero.
region::region() {
  num=0;
};

// Dumps all particles in a region to an open file stream.
void region::dump(fstream &of) {
  int j=0;
  while(j<num) {
    of << data[j].n << " " << data[j].p.x << " ";
    of << data[j].p.y << " " << data[j++].p.z << endl;
  }
};

// Dumps all particles in a region to an open file stream.
void region::dumprestart(fstream &of) {
  int j=0;
  while(j<num) {
    of << data[j].n << " 1 1 " << data[j].p.x << " ";
    of << data[j].p.y << " " << data[j++].p.z << endl;
  }
};

// Dumps all particles in a region to an open file stream.
void region::dumpraster(fstream &of) {
  int j=0;
  while(j<num) {
    of << "2" << endl;
    of << data[j].p.x << " " << data[j].p.y << " " << data[j].p.z;
    if (data[j++].n==0) of << " 0.5 0.25 0.25 0.40" << endl;
    else of << " 0.5 1.00 1.00 0.80" << endl;
  }
};

// Dumps all particles in a region to an open file stream.
void region::dumppov(fstream &of) {
  int j=0;
  while(j<num) {
    of << "sphere{<" << data[j].p.x << "," << data[j].p.y;
    of << "," << data[j].p.z << ">,0.5}" << endl;
    j++;
  }
};

// Adds a particle to a region.
void region::put(int n,vec &p) {
  if (num<=buf) {
    data[num].n=n;
    data[num++].p=p;
  }
};

// Initializes a particle container object.
```

```
container::container(float minx,float maxx,float miny,float maxy,
                     float minz,float maxz,int xn,int yn,int zn) {
  xmin=minx;xmax=maxx;
  ymin=miny;ymax=maxy;
  zmin=minz;zmax=maxz;
  nx=xn;ny=yn;nz=zn;
  xsp=float(nx)/(xmax-xmin);
  ysp=float(ny)/(ymax-ymin);
  zsp=float(nz)/(zmax-zmin);
  blocks=nx*ny*nz;
  reg=new region[nx*ny*nz];
};

// Dumps particle positions and characteristic to a specified file.
void container::dump(char * filename) {
  int i;
  fstream file;
  file.open(filename,fstream::out|fstream::trunc);
  for(i=0;i<blocks;i++) {
    reg[i].dump(file);
  }
  file.close();
};

// Dumps a particle snapshot to an open restart file stream.
void container::dumprestart(int t,fstream &of) {
  int i,j=0;
  for(i=0;i<blocks;i++) {
    j+=reg[i].num;
  }
  of << "␣ITEM:␣TIMESTEP" << endl << t << endl;
  of << "␣ITEM:␣NUMBER␣OF␣ATOMS" << endl << j << endl;
  of << "␣ITEM:␣BOX␣BOUNDS" << endl;
  of << xmin << "␣" << xmax << endl;
  of << ymin << "␣" << ymax << endl;
  of << zmin << "␣" << zmax << endl;
  of << "␣ITEM:␣ATOMS" << endl;
  for(i=0;i<blocks;i++) {
    reg[i].dumprestart(of);
  }
};

// Dumps particle positions to a file readable by the raster3D raytracer.
void container::dumpraster(char * filename) {
  int i;
  fstream file;
  file.open(filename,fstream::out|fstream::trunc);
  file << "@header.r3d" << endl;
  for(i=0;i<blocks;i++) {
    reg[i].dumpraster(file);
  }
  file.close();
};

// Dumps particle positions to a file readable by the POVRay raytracer.
void container::dumppov(char * filename) {
  int i;
  fstream file;
  file.open(filename,fstream::out|fstream::trunc);
  file << "#declare␣container=union{" << endl;
  for(i=0;i<blocks;i++) {
    reg[i].dumppov(file);
  }
  file << "}" << endl;
  file.close();
};

// Adds a particle to the container, placing it in the correct region.
```

```
// (Vector input)
void container::put(int n,vec &p) {
  int mx,my,mz;
  mx=myint((p.x-xmin)*xsp);
  my=myint((p.y-ymin)*ysp);
  mz=myint((p.z-zmin)*zsp);
  if (mx>=0&&mx<nx&&my>=0&&my<ny&&mz>=0&&mz<nz) {
    reg[mx+nx*(my+ny*mz)].put(n,p);
  }
};

// Adds a particle to the container, placing it in the correct region.
// (Three scalar input)
void container::put(int n,float x,float y,float z) {
  int mx,my,mz;
  vec p;
  mx=myint((x-xmin)*xsp);
  my=myint((y-ymin)*ysp);
  mz=myint((z-zmin)*zsp);
  if (mx>=0&&mx<nx&&my>=0&&my<ny&&mz>=0&&mz<nz) {
    p.x=x;p.y=y;p.z=z;
    reg[mx+nx*(my+ny*mz)].put(n,p);
  }

};

//Reads a particle position file from stdin and puts them into the container.
void container::import() {
  int n;vec p;
  cin >> n >> p.x >> p.y >> p.z;
  while (!cin.eof()) {
    put(n,p);
    cin >> n >> p.x >> p.y >> p.z;
  }
};

//Reads in a snapshot from an open restart file stream.
void container::importrestart(fstream &rf) {
  int i,d,e,n;float f;vec p;char q[256];
  for(i=0;i<blocks;i++) reg[i].num=0;
  for(i=0;i<3;i++) rf.getline(q,256);
  rf >> n;
  for(i=0;i<6;i++) rf.getline(q,256);
  for(i=0;i<n;i++) {
    rf >> d >> e >> f >> p.x >> p.y >> p.z;
    put(d,p);
  }
  rf.getline(q,256);
};

// Move a spot in the container, position p, velocity v, radius r.
// Each particle is moved individually and shifted into a new region if
// necessary. Only the regions which can be affected are tested. Regions are
// tested in the correct order to avoid moving a particle twice.
void container::spot(vec &p,vec &v,float r) {
  int ax,ay,az,i,j,k,fx,fy,fz;
  vec t;intrec a;
  bool rx,ry,rz;
  a=boundbox(p,r);r*=r;
  rx=(v.x<0);ry=(v.y<0);rz=(v.z<0);
  for (az=rz?a.lz:a.uz;rz?(az<=a.uz):(az>=a.lz);az+=rz?1:-1) {
    for (ay=ry?a.ly:a.uy;ry?(ay<=a.uy):(ay>=a.ly);ay+=ry?1:-1) {
      for (ax=rx?a.lx:a.ux;rx?(ax<=a.ux):(ax>=a.lx);ax+=rx?1:-1) {
        i=ax+nx*(ay+ny*az);
        j=0;
        while(j<reg[i].num) {
          if (radsq(reg[i].data[j].p,p)<r) {
            t=reg[i].data[j].p+v;
```

```
            fx=myint((t.x-xmin)*xsp);
            fy=myint((t.y-ymin)*ysp);
            fz=myint((t.z-zmin)*zsp);
            if (fx==ax&&fy==ay&&fz==az) reg[i].data[j++].p=t;
            else {
              if (fx>=0&&fx<nx&&fy>=0&&fy<ny&&fz>=0&&fz<nz) {
                k=fx+nx*(fy+ny*fz);
                reg[k].data[reg[k].num].n=reg[i].data[j].n;
                reg[k].data[reg[k].num++].p=t;
              }
              reg[i].data[j]=reg[i].data[--reg[i].num];
            }
          }
          else j++;
        }
      }
    }
  }
};

// Move a gaussian spot in the container, position p, velocity v, radius r.
// Each particle is moved individually and shifted into a new region if
// necessary. Only the regions which can be affected are tested. Regions are
// tested in the correct order to avoid moving a particle twice.
void container::spotg(vec &p,vec &v,float r,float l) {
  int ax,ay,az,i,j,k,fx,fy,fz;r*=r;l=-1/(l*l*2);float s;
  vec t;intrec a;
  bool rx,ry,rz;
  a=boundbox(p,r);
  rx=(v.x<0);ry=(v.y<0);rz=(v.z<0);
  for (az=rz?a.lz:a.uz;rz?(az<=a.uz):(az>=a.lz);az+=rz?1:-1) {
    for (ay=ry?a.ly:a.uy;ry?(ay<=a.uy):(ay>=a.ly);ay+=ry?1:-1) {
      for (ax=rx?a.lx:a.ux;rx?(ax<=a.ux):(ax>=a.lx);ax+=rx?1:-1) {
        i=ax+nx*(ay+ny*az);
        j=0;
        while(j<reg[i].num) {
          s=radsq(reg[i].data[j].p,p);
          if (s<r) {
            t=reg[i].data[j].p+v*exp(s*l);
            fx=myint((t.x-xmin)*xsp);
            fy=myint((t.y-ymin)*ysp);
            fz=myint((t.z-zmin)*zsp);
            if (fx==ax&&fy==ay&&fz==az) reg[i].data[j++].p=t;
            else {
              if (fx>=0&&fx<nx&&fy>=0&&fy<ny&&fz>=0&&fz<nz) {
                k=fx+nx*(fy+ny*fz);
                reg[k].data[reg[k].num].n=reg[i].data[j].n;
                reg[k].data[reg[k].num++].p=t;
              }
              reg[i].data[j]=reg[i].data[--reg[i].num];
            }
          }
          else j++;
        }
      }
    }
  }
};

// Applies elastic relaxation on a sphere of radius s, centered at p. Particles
// with radii between s and r fixed to avoid long range distruption. The
// magnitude of the correcting force is given by "force". After each step, a
// velocity damping of magnitude "damp" is applied. Vertical walls also
// contribute a force, but the container bottom does not, to prevent jamming at
// the orifice. Using pointers, particle positions are updated in situ and
// never copied out. A final step is applied to move any particles that moved
// into a different region - this is done in reverse to prevent memory
// conflicts. Memory for the buffers is static to prevent lots of memory
```

```
// reallocation each time the routine is called.
void container::relaxslow(vec &p,float r,float s,
                          float force,float damp,int steps) {
  static particle * e[1024];
  static particle * c[1024];
  static vec v[1024];
  static int b[1024];
  vec d;intrec a;float t;
  int ax,ay,az,ct=0,cs=0,i,j;
  a=boundbox(p,s);
  for (az=a.lz;az<=a.uz;az++) {
    for (ay=a.ly;ay<=a.uy;ay++) {
      for (ax=a.lx;ax<=a.ux;ax++) {
        i=ax+nx*(ay+ny*az);
        for(j=0;j<reg[i].num;j++) {
          t=radsq(reg[i].data[j].p,p);
          if (t<s*s) {
            if (t<r*r) {
              e[ct]=&(reg[i].data[j]);
              v[ct].x=0;v[ct].y=0;v[ct].z=0;
              b[ct++]=i;
            }
            else c[cs++]=&(reg[i].data[j]);
          }
        }
      }
    }
  }
  for(ax=0;ax<steps;ax++) {
    for(i=0;i<ct;i++) {
      for(j=i+1;j<ct;j++) {
        d=e[i]->p-e[j]->p;
        t=radsq(d);
        if (t<1) {t=sqrt(t);d*=force*(1-t)/t;v[i]+=d;v[j]-=d;}
      }
      for(j=0;j<cs;j++) {
        d=e[i]->p-c[j]->p;
        t=radsq(d);
        if (t<1) {t=sqrt(t);d*=force*(1-t)/t;v[i]+=d;}
      }
      if (e[i]->p.x<xmin+0.5) v[i].x-=force*(e[i]->p.x-xmin-0.5);
      if (e[i]->p.x>xmax-0.5) v[i].x-=force*(e[i]->p.x-xmax+0.5);
      if (e[i]->p.y<ymin+0.5) v[i].y-=force*(e[i]->p.y-ymin-0.5);
      if (e[i]->p.y>ymax-0.5) v[i].y-=force*(e[i]->p.y-ymax+0.5);
//    if (e[i]->p.z<zmin+0.5) v[i].z-=force*(e[i]->p.z-zmin-0.5);
//    if (e[i]->p.z>zmax-0.5) v[i].z-=force*(e[i]->p.z-zmax+0.5);
    }
    for(i=0;i<ct;i++) {e[i]->p+=v[i];v[i]*=damp;}
  }
  for (i=ct-1;i>=0;i--) {
    ax=myint((e[i]->p.x-xmin)*xsp);
    ay=myint((e[i]->p.y-ymin)*ysp);
    az=myint((e[i]->p.z-zmin)*zsp);
    j=ax+nx*(ay+ny*az);
    if (j!=b[i]) {
      if (ax>=0&&ax<nx&&ay>=0&&ay<ny&&az>=0&&az<nz)
        reg[j].data[reg[j].num++]=*e[i];
      *e[i]=reg[b[i]].data[--reg[b[i]].num];
    }
  }
}

// More efficient relaxion scheme
#define rgridc rgrid*rgrid*rgrid
void container::relax(vec &p,float r,float s,float force,float damp,int steps) {
  static particlev b[rgridc][rbuf];
  static int c[rgridc],e[rgridc],cc[rgridc];
  int ax,ay,az,bx,by,bz,i,j,k,l,m;vec d;intrec a;float t;
```

```
for (i=0;i<rgridc;i++) {c[i]=0;e[i]=rbuf;}
a=boundbox(p,s);
for (az=a.lz;az<=a.uz;az++) {
  for (ay=a.ly;ay<=a.uy;ay++) {
    for (ax=a.lx;ax<=a.ux;ax++) {
      i=ax+nx*(ay+ny*az);j=0;
      while(j<reg[i].num) {
        d=reg[i].data[j].p-p;
        t=radsq(d);
        if (t<s*s) {
          bx=int((d.x+s)/(2*s)*rgrid);
          by=int((d.y+s)/(2*s)*rgrid);
          bz=int((d.z+s)/(2*s)*rgrid);
          k=bx+rgrid*(by+rgrid*bz);
          if (t<r*r) {
            b[k][c[k]].data=reg[i].data[j];
            b[k][c[k]].v.x=0;b[k][c[k]].v.y=0;b[k][c[k]++].v.z=0;
            reg[i].data[j]=reg[i].data[--reg[i].num];
          }
          else b[k][--e[k]].data.p=reg[i].data[j++].p;
        }
        else j++;
      }
    }
  }
}
i=0;
while(i<steps) {
  for(j=0;j<rgridc;j++) {
    for(k=0;k<c[j];k++) {
      d=b[j][k].data.p-p;
      a.lx=int((d.x-1+s)/(2*s)*rgrid);if (a.lx<0) a.lx=0;if (a.lx>=rgrid) a.lx=rgrid-1;
      a.ly=int((d.y-1+s)/(2*s)*rgrid);if (a.ly<0) a.ly=0;if (a.ly>=rgrid) a.ly=rgrid-1;
      a.lz=int((d.z-1+s)/(2*s)*rgrid);if (a.lz<0) a.lz=0;if (a.lz>=rgrid) a.lz=rgrid-1;
      a.ux=int((d.x+1+s)/(2*s)*rgrid);if (a.ux<0) a.ux=0;if (a.ux>=rgrid) a.ux=rgrid-1;
      a.uy=int((d.y+1+s)/(2*s)*rgrid);if (a.uy<0) a.uy=0;if (a.uy>=rgrid) a.uy=rgrid-1;
      a.uz=int((d.z+1+s)/(2*s)*rgrid);if (a.uz<0) a.uz=0;if (a.uz>=rgrid) a.uz=rgrid-1;
      for(az=a.lz;az<=a.uz;az++) {
        for(ay=a.ly;ay<=a.uy;ay++) {
          for(ax=a.lx;ax<=a.ux;ax++) {
            l=ax+rgrid*(ay+rgrid*az);
            for(m=rbuf-1;m>=e[l];m--) {
              d=b[j][k].data.p-b[l][m].data.p;
              t=radsq(d);
              if (t<1) {t=sqrt(t);d*=force*(1-t)/t;b[j][k].v+=d;}
            }
            if(j<l) {
              for(m=0;m<c[l];m++) {
                d=b[j][k].data.p-b[l][m].data.p;
                t=radsq(d);
                if (t<1) {t=sqrt(t);d*=force*(1-t)/t;b[j][k].v+=d;b[l][m].v-=d;}
              }
            }
            else if (j==l) {
              for(m=0;m<k;m++) {
                d=b[j][k].data.p-b[l][m].data.p;
                t=radsq(d);
                if (t<1) {t=sqrt(t);d*=force*(1-t)/t;b[j][k].v+=d;b[l][m].v-=d;}
              }
            }
          }
        }
      }
      if (b[j][k].data.p.x<xmin+0.5) b[j][k].v.x-=force*(b[j][k].data.p.x-xmin-0.5);
      if (b[j][k].data.p.x>xmax-0.5) b[j][k].v.x-=force*(b[j][k].data.p.x-xmax+0.5);
      if (b[j][k].data.p.y<ymin+0.5) b[j][k].v.y-=force*(b[j][k].data.p.y-ymin-0.5);
      if (b[j][k].data.p.y>ymax-0.5) b[j][k].v.y-=force*(b[j][k].data.p.y-ymax+0.5);
    }
```

```
      cc[j]=c[j];
    }
    if(++i<steps) {
      for(j=0;j<rgridc;j++) {
        k=0;
        while(k<cc[j]) {
          b[j][k].data.p+=b[j][k].v;
          d=b[j][k].data.p-p;
          b[j][k].v*=damp;
          bx=int((d.x+s)/(2*s)*rgrid);
          by=int((d.y+s)/(2*s)*rgrid);
          bz=int((d.z+s)/(2*s)*rgrid);
          l=bx+rgrid*(by+rgrid*bz);
          if(l!=j) {
            b[l][c[l]++]=b[j][k];
            b[j][k]=b[j][--c[j]];
            if(c[j]>=cc[j]) k++;else cc[j]--;
          }
          else k++;
        }
      }
    }
  }
  for(j=0;j<rgridc;j++) {
    for(k=0;k<c[j];k++) {
      b[j][k].data.p+=b[j][k].v;
      ax=myint((b[j][k].data.p.x-xmin)*xsp);
      ay=myint((b[j][k].data.p.y-ymin)*ysp);
      az=myint((b[j][k].data.p.z-zmin)*zsp);
      i=ax+nx*(ay+ny*az);
      if (ax>=0&&ax<nx&&ay>=0&&ay<ny&&az>=0&&az<nz)
        reg[i].data[reg[i].num++]=b[j][k].data;
    }
  }
};

// Special int function required to pick out the correct regions.
// Returns int(i) if i>0 and int(i)-1 if i<0 to give consistent stepping from
// positive to negative. Thus myint(-1.5,-0.5,0.5,1.5)=-2,-1,0,1.
inline int container::myint(float co) {
  return int((co<0)?int(co)-1:int(co));
}

// Counts the particles in a spherical region.
int container::count(vec &p,float r) {
  intrec a;
  int ax,ay,az,i,j,ct=0;
  a=boundbox(p,r);
  for (az=a.lz;az<=a.uz;az++) {
    for (ay=a.ly;ay<=a.uy;ay++) {
      for (ax=a.lx;ax<=a.ux;ax++) {
        i=ax+nx*(ay+ny*az);
        for(j=0;j<reg[i].num;j++) {
          if (radsq(reg[i].data[j].p,p)<r*r) ct++;
        }
      }
    }
  }
  return ct;
};

// Finds g(r) for the entire container, for small separations.
void container::gofr(int *b) {
  int i,j,k,l,m,ax,ay,az;vec t;
  float r;intrec a;
  for(i=0;i<blocks;i++) {
    for(j=0;j<reg[i].num;j++) {
      a=boundbox(reg[i].data[j].p,3);
```

```
      for (az=a.lz;az<=a.uz;az++) {
        for (ay=a.ly;ay<=a.uy;ay++) {
          for (ax=a.lx;ax<=a.ux;ax++) {
            k=ax+nx*(ay+ny*az);
            for(l=0;l<reg[k].num;l++) {
              r=radsq(reg[k].data[l].p,reg[i].data[j].p);
              if (r<9/*&&k!=i&&l!=j*/) {m=int(sqrt(r)*500);b[m]++;}
            }
          }
        }
      }
    }
  }
};

// Finds the bond angle distribution for the entire container.
void container::bondangle(int *b,float r) {
  int i,j,k,l,m,n,w,ax,ay,az,bx,by,bz;
  float s,t,rr=r*r;intrec a;vec u,v;
  for(i=0;i<blocks;i++) {
    for(j=0;j<reg[i].num;j++) {
      a=boundbox(reg[i].data[j].p,r);
      for (az=a.lz;az<=a.uz;az++) {
        for (ay=a.ly;ay<=a.uy;ay++) {
          for (ax=a.lx;ax<=a.ux;ax++) {
            k=ax+nx*(ay+ny*az);
            for(l=0;l<reg[k].num;l++) {
              u=reg[k].data[l].p-reg[i].data[j].p;s=radsq(u);
              if (s<rr&&(k!=i||l!=j)) {
                for (bz=a.lz;bz<=a.uz;bz++) {
                  for (by=a.ly;by<=a.uy;by++) {
                    for (bx=a.lx;bx<=a.ux;bx++) {
                      m=bx+nx*(by+ny*bz);
                      for(n=0;n<reg[m].num;n++) {
                        v=reg[m].data[n].p-reg[i].data[j].p;t=radsq(v);
                        if (t<rr&&(m!=k||n!=l)&&(m!=i||n!=j)) {
                          w=int(acos(sp(u,v)/sqrt(s*t))*477.4648292756860073066512 9);
                          b[w]++;
                        }
                      }
                    }
                  }
                }
              }
            }
          }
        }
      }
    }
  }
};


// Computes the distribution of hole sizes for the container.
void container::spaces(float xs,float ys,float zs,
                       float mnz,float mxz,int *b) {
  vec p;float xp,yp,zp,r,s,t;intrec a;int ax,ay,az,j,k;float lar;
  xp=(xmax-xmin)/xs;
  yp=(ymax-ymin)/ys;
  zp=(mxz-mnz)/zs;
  for(p.z=mnz+zp/2;p.z<mxz;p.z+=zp) {
    for(p.y=ymin+yp/2;p.y<ymax;p.y+=yp) {
      for(p.x=xmin+xp/2;p.x<xmax;p.x+=xp) {
        r=(p.x-xmin>1.4975)?1.495:p.x-xmin;if (r>xmax-p.x) r=xmax-p.x;
        if (r>p.y-ymin) r=p.y-ymin;if (r>ymax-p.y) r=ymax-p.y;
        if (r>p.z-zmin) r=p.z-zmin;if (r>zmax-p.z) r=zmax-p.z;
        a=boundbox(p,r);t=4;
```

```
        for (az=a.lz;az<=a.uz;az++) {
          for (ay=a.ly;ay<=a.uy;ay++) {
            for (ax=a.lx;ax<=a.ux;ax++) {
              k=ax+nx*(ay+ny*az);
              for(j=0;j<reg[k].num;j++) {
                s=radsq(reg[k].data[j].p,p);
                if (t>s) t=s;
              }
            }
          }
        }
        t=sqrt(t)-0.5;
        if (t>0) {j=int(((t<r)?t:r)*200);b[j]++;}
        lar=(t<r)?t:r;if (lar>0.40) cout << lar << endl;
// Use this line if you want to isolate boundary effects
//        if (t>0&&t<r) {j=int(t*200);b[j]++;}
      }
    }
  }
};

// Finds all the regions that lie within radius r of p.
inline intrec container::boundbox(vec &p,float r) {
  intrec a;
  a.lx=int((p.x-r-xmin)*xsp);if (a.lx<0) a.lx=0;if (a.lx>=nx) a.lx=nx-1;
  a.ly=int((p.y-r-ymin)*ysp);if (a.ly<0) a.ly=0;if (a.ly>=ny) a.ly=ny-1;
  a.lz=int((p.z-r-zmin)*zsp);if (a.lz<0) a.lz=0;if (a.lz>=nz) a.lz=nz-1;
  a.ux=int((p.x+r-xmin)*xsp);if (a.ux<0) a.ux=0;if (a.ux>=nx) a.ux=nx-1;
  a.uy=int((p.y+r-ymin)*ysp);if (a.uy<0) a.uy=0;if (a.uy>=ny) a.uy=ny-1;
  a.uz=int((p.z+r-zmin)*zsp);if (a.uz<0) a.uz=0;if (a.uz>=nz) a.uz=nz-1;
  return a;
};

// Diagnostic routine to output the number of particles in each region.
void container::regioncount() {
  int ax,ay,az,i,ct=0;
  for (az=0;az<nz;az++) {
    for (ay=0;ay<ny;ay++) {
      for (ax=0;ax<nx;ax++) {
        i=ax+nx*(ay+ny*az);
        cout << "(x,y,z)=(" << ax << "," << ay << "," << az
             << ") : " << reg[i].num << " particles" << endl;
        ct+=reg[i].num;
      }
    }
  }
  cout << ct << " particles in total" << endl;
};

//Diagnostic routine to output the all the particles in a specific
//subregion to standard output.
void container::regionshot(float minx,float maxx,float miny,
                           float maxy,float minz,float maxz) {
  intrec a;int ax,ay,az,i,j;
  a.lx=int((minx-xmin)*xsp);if (a.lx<0) a.lx=0;if (a.lx>=nx) a.lx=nx-1;
  a.ly=int((miny-ymin)*ysp);if (a.ly<0) a.ly=0;if (a.ly>=ny) a.ly=ny-1;
  a.lz=int((minz-zmin)*zsp);if (a.lz<0) a.lz=0;if (a.lz>=nz) a.lz=nz-1;
  a.ux=int((maxx-xmin)*xsp);if (a.ux<0) a.ux=0;if (a.ux>=nx) a.ux=nx-1;
  a.uy=int((maxy-ymin)*ysp);if (a.uy<0) a.uy=0;if (a.uy>=ny) a.uy=ny-1;
  a.uz=int((maxz-zmin)*zsp);if (a.uz<0) a.uz=0;if (a.uz>=nz) a.uz=nz-1;
  for (az=a.lz;az<=a.uz;az++) {
    for (ay=a.ly;ay<=a.uy;ay++) {
      for (ax=a.lx;ax<=a.ux;ax++) {
        i=ax+nx*(ay+ny*az);
        for(j=0;j<reg[i].num;j++) {
          if (reg[i].data[j].p.x>minx&&reg[i].data[j].p.x<maxx&&
              reg[i].data[j].p.y>miny&&reg[i].data[j].p.y<maxy&&
              reg[i].data[j].p.z>minz&&reg[i].data[j].p.z<maxz) {
```

```
            cout << reg[i].data[j].n << " " << reg[i].data[j].p.x << " ";
            cout << reg[i].data[j].p.y << " " << reg[i].data[j].p.z << endl;
          }
        }
      }
    }
  }
};

//Clears the container of all particles.
void container::clear() {
 for(int i=0;i<blocks;i++) reg[i].num=0;
}

#endif
```

C.6.5 relaxtest.cc

```
// Spot elastic relaxation test
//
// Author  : Chris H. Rycroft
// Version : 1.0
// Date    : April 6th 2004
//
// This code uses the spot model routines to initialize a small group of
// particles that is too densely packed, and then apply several elastic
// relaxation steps.

// Standard library includes
#include <cstdio>
#include <iostream>
#include <fstream>
#include <cmath>
using namespace std;

// Spot model object routines
#define buf 2048
#include "vec.cc"
#include "container.cc"

int main () {
  float x,y;int i,j;
  container con(-25,25,-4,4,0,150,10,3,30);     // Initialize a container
  char filename[8];vec p(23,0,10);
  for(i=0;i<=20;i++) {
    j=0;
    for(x=24.5;x>20;x-=1) {
      for(y=-3.5;y<4;y+=1) {                    // Make a small regular
        con.put(j++,x,y,10);                    // grid of particles
      }
    }
    con.put(j++,23,-0.1,10);                    // Add three extra particles
    con.put(j++,22.8,0,10);
    con.put(j,22.5,0.1,10);
    con.relax(p,10,10,0.7,0.4,i);               // Apply a relaxation
    sprintf(filename,"relax%d",i);
    con.dump(filename);                         // Dump out positions
    con.clear();
  }
};
```

C.6.6 spot15.cc

```
// Fast spot model code — main routine
//
// Author  : Chris H. Rycroft
// Version : 1.0
// Date    : April 6th 2004
//
// This program uses the spot container routines to make a simple event driven
// spot simulation. Spots are introduced in clusters at the orifice and
// propagated upwards according to a simple continuous time random walk.
// Elastic relaxation is applied after every spot movement.

// Standard library includes
#include <cstdio>
#include <iostream>
#include <fstream>
#include <cmath>
using namespace std;

// Spot model object routines
#define buf 2048
#include "vec.cc"
#include "container.cc"

// Spot model parameters
#define max 16000       //Maximum number of spots
#define clu 1           //Number of spots in a cluster
#define ipr 375.774     //Spot insert rate
#define mpr 27.9540     //Spot move rate
#define rad 2.60266     //Spot radius
#define xst 0.675910    //x step
#define yst 0.675910    //y step
#define zst 0.1         //z step
#define scf 399.341     //Displacement scale factor
#define xbu 1           //Buffer amount of spot center from x walls
#define ybu 1           //Buffer amount of spot center from y walls

// Convenient random number routines
inline double rnd() {
  int r=rand();
  while(r==0) r=rand();
  return (double) rand()/RAND_MAX;
};

int main () {
  double a,t=0;
  vec u(0,0,0),v,s[max],w;
  float xa,xs,ya,ys,sc;
  int i=-1,j,n=0,b,xp,xm,yp,ym,rn;
  char buffer[20];
  fstream file;
  container con(-25,25,-4,4,0,150,10,3,30);     // Initialize container object
  con.import();                                 // Import particles from stdin
  file.open("spot15snap",fstream::out|fstream::trunc);
  while(t<1000) {
    t+=-log(rnd())/(ipr+mpr*n);                 // Work out time to next event
    j=int(t/2);
    if (i<j) con.dumprestart(j*20000,file);
    i=j;
    a=rnd()*(ipr+mpr*n);                        // Decide whether to introduce
    if (a<mpr*n) {                              // or move a spot
      b=int(a/mpr);
      if (s[b].z<rad/3) {
        s[b].z+=zst/2;u.z=-zst/scf;con.spot(s[b],u,rad);
        con.relax(s[b],rad+1,rad+2,0.8,0,1);
        s[b].z+=zst/2;
      } else {
        if (s[b].z<=130) {
          if (s[b].x+xst>25-xbu) xa=(25-xbu-s[b].x)/2;else xa=xst/2;
```

```
        if (s[b].x-xst<xbu-25) xs=(25-xbu+s[b].x)/2;else xs=xst/2;
        if (s[b].y+yst>4-ybu) ya=(4-ybu-s[b].y)/2;else ya=yst/2;
        if (s[b].y-yst<ybu-4) ys=(4-ybu+s[b].y)/2;else ys=yst/2;
        rn=rand()%4;
        if (rn<2) {v.x=(rn<1)?xa:-xs;v.y=0;}
        else {v.x=0;v.y=(rn<3)?ya:-ys;}
        v.z=zst/2;
        w=v/(-scf/2);
        s[b]+=v;                                // Move the spot halfway
        con.spot(s[b],w,rad);                   // Apply its influence
        con.relax(s[b],rad+1,rad+2,0.8,0,1);    // Apply elastic relaxation
        s[b]+=v;                                // Move it the rest of the way
      } else s[b]=s[--n];
    }
  } else {
    b=clu+n;
    if (b<=max) while(n<b)
    {
      s[n].x=rnd()*3-1.5;                       // Introduce up to clu new
      s[n].y=rnd()*3-1.5;                       // spots at the orifice
      s[n++].z=-rad/3;
    }
  }
 }
 file.close();
};
```

Bibliography

[1] http://lammps.sandia.gov/.

[2] http://web.mit.edu/pebble-bed.

[3] http://www.qhull.org/.

[4] See, e.g., http://gif.inel.gov, http://nuclear.inel.gov.

[5] http://www.pbmr.com.

[6] M. P. Allen and D. J. Tildesley. *Computer Simulation of Liquids*. Oxford University Press, 1987.

[7] C. Ancey. *Phys. Rev. E*, 65:011304, 2002.

[8] I. S. Aranson and L. S. Tsimring. Continuum description of avalanches in granular media. *Phys. Rev. E*, 64:020301, 2001.

[9] I. S. Aranson and L. S. Tsimring. Continuum theory of partially fluidized granular flows. *Phys. Rev. E*, 65:061303, 2002.

[10] I. S. Aranson and L. S. Tsimring. Patterns and collective behavior in granular media: Theoretical concepts. *Rev. Mod. Phys.*, 78:641–692, 2006.

[11] F. Aurenhammer. Voronoi diagrams – a survey of a fundamental geometric data structure. *ACM Computing Surveys*, 23(3):345–405, 1991.

[12] P. Bak. *How nature works: the science of self-organized criticality*. Copernicus, New York, 1996.

[13] P. Bak, C. Tang, and K. Wiesenfeld. *Phys. Rev. A*, 38(1):364, 1988.

[14] C. B. Barber, D. P. Dobkin, and H. T. Huhdaanpaa. The quickhull algorithm for convex hulls. *ACM Transactions on Mathematical Software*, 22(4):469–483, 1996.

[15] G. W. Baxter and R. P. Behringer. Cellular automata models of granular flow. *Phys. Rev. A*, 42:1017–1020, 1990.

[16] G. W. Baxter and R. P. Behringer. Cellular automata models for the flow of granular materials. *Physica D*, 52:465–471, 1991.

[17] M. Z. Bazant. The spot model for random-packing dynamics. *Mechanics of Materials*, 38:717–731, 2006.

[18] G. Berton, R. Delannay, P. Richard, N. Taberlet, and A. Valance. *Phys. Rev. E*, 68:051303, 2003.

[19] D. L. Blair, N. Mueggenburg, A. M. Marshall, H. M. Jaeger, and S. R. Nagel. Force distributions in three-dimensional granular assemblies: Effects of packing order and interparticle friction. *Phys. Rev. E*, 63:041304, 2001.

[20] R. D. Blumofe, C. F. Joerg, B. C. Kuszmaul, C. E. Leiserson, K. H. Randall, and Y. Zhou. Cilk: an efficient multithreaded runtime system. In *PPOPP '95: Proceedings of the fifth ACM SIGPLAN symposium on Principles and practice of parallel programming*, pages 207–216, New York, NY, USA, 1995. ACM Press.

[21] L. Bocquet, W. Losert, D. Schalk, T. C. Lubensky, and J. P Gollub. *Phys. Rev. E*, 65:011307, 2001.

[22] D. Bonamy, F. Daviaud, and L. Laurent. *Phys. Rev. Lett.*, 89:034301–1, 2002.

[23] T. Boutreux and P. G. de Gennes. Compaction of granular mixtures: a free volume model. *Physica A*, 244:59–67, 1997.

[24] Robert C. Brewster, James W. Landry, Gary S. Grest, and Alex J. Levine. Breakdown of bagnold scaling in cohesive granular flows. *Phys. Rev. E*, 72:061301, 2005.

[25] H. Caram and D. C. Hong. Random-walk approach to granular flows. *Phys. Rev. Lett.*, 67:828–831, 1991.

[26] X. Cheng, J. B. Lechman, A. Fernandez-Barbero, G. S. Grest, H. M. Jaeger, G. S. Karczmar, M. E. Mobius, and S. R. Nagel. Three-dimensional shear in granular flow. *Phys. Rev. Lett.*, 96:038001, 2006.

[27] J. Choi, A. Kudrolli, and M. Z. Bazant. Velocity profile of gravity-driven dense granular flow. *J. Phys.: Condensed Matter*, 17:S2533–S2548, 2005.

[28] J. Choi, A. Kudrolli, R. R. Rosales, and M. Z. Bazant. Diffusion and mixing in gravity driven dense granular flows. *Phys. Rev. Lett.*, 92:174301, 2004.

[29] M. H. Cohen and D. Turnbull. Molecular transport in liquids and glasses. *J. Chem. Phys.*, 31:1164, 1959.

[30] Y. Cohen and A. B. Metzner. Wall effects in laminar flow of fluids through packed beds. *AIChE Journal*, 27:705–714, 1981.

[31] S. N. Coppersmith, C. h. Liu, S. Majumdar, O. Narayan, and T. A. Witten. Model for force fluctuations in bead packs. *Phys. Rev. E*, 53:4673–4685, 1996.

[32] P. A. Cundall and O. D. L. Strack. A discrete numerical model for granular assemblies. *Geotechnique*, 29:47, 1979.

[33] F. da Cruz, S. Emam, M. Prochnow, J. Roux, and F. Chevoir. Rheophysics of dense granular materials: Discrete simulation of plane shear flows. *Phys. Rev. E.*, 72:021309, 2005.

[34] P. G. de Gennes. Granular matter: a tentative view. *Rev. Mod. Phys.*, 71:S374–S382, 1999.

[35] C. Donati, J. F. Douglas, W. Kob, S. J. Plimpton, P. H. Poole, and S. C. Glotzer. Stringlike cooperative motion in a supercooled liquid. *Phys. Rev. Lett.*, 80:2338–2341, 1998.

[36] A. Donev, I. Cisse, D. Sachs, E. A. Variano, F. H. Stillinger, R. Connelly, S. Torquato, and P. M. Chaikin. Improving the density of jammed disordered packings using ellipsoids. *Science*, 303:990–993, 2004.

[37] A. Donev, S. Torquato, F. H. Stillinger, and R. Connelly. Jamming in hard sphere and disk packings. *J. Appl. Phys.*, 95:989–999, 2004.

[38] C. du Toit. The numerical determination of the variation in the porosity of pebble-bed code. In *Proceedings of the Conference on High-Temperature Reactors, Petten, NL*, Vienna, Austria, April 2002. International Atomic Energy Agency.

[39] S. F. Edwards. The equations of stress in a granular material. *Physica A*, 249:226–231, 1998.

[40] T. Elperin and A. Vikhanski. *Europhys. Lett.*, 42:619, 1998.

[41] D. Ertaş and T. C. Halsey. Granular gravitational collapse and chute flow. *Europhys. Lett.*, 60:931–937, 2002.

[42] H. Eyring. Viscosity, plasticity, and diffusion as examples of absolute reaction rates. *J. Chem. Phys.*, 4:283–291, 1936.

[43] M. L. Falk and J. S. Langer. Dynamics of viscoplastic deformation in amorphous solids. *Phys. Rev. E*, 57:7192–7205, 1998.

[44] M. L. Falk, J. S. Langer, and L. Pechenik. Thermal effects in the shear-transition-zone theory of amorphous plasticity: Comparisons to metallic glass data.

[45] D. Fenistein and M. van Hecke. Wide shear zones in granular bulk flow. *Nature*, 425:256, 2003.

[46] A. Ferguson and B. Chakraborty. Stress and large-scale spatial structures in dense, driven granular flows. *Phys. Rev. E*, 73(1):011303, 2006.

[47] A. Ferguson and B. Chakraborty. Spatially heterogenous dynamics in dense, driven granular flows. *Europhysics Letters (EPL)*, 78(2):28003, 2007.

[48] A. Ferguson, B. Fisher, and B. Chakraborty. Impluse distributions in dense granular flows: Signatures of large-scale spatial structures. *Europhysics Letters*, 66(2):277–283, 2004.

[49] K. Fukuda, J. Kendall, J. Kupitz, D. Matzner, E. Mulder, P. Pretorius, A. Shenoy, S. Shiozawa, W. Simon, Y. Sun, P. Uselton, and Y. Xu. Current status and future development of modular, high temperature, gas-cooled reactor technology. Technical Report IAEA-TECDOC–1198, International Atomic Energy Agency, Vienna, Austria, 2001.

[50] J. S. Goodling, R. I. Vachon, W. S. Stelpflug, and S. J. Ying. Radial porosity distribution in cylindrical beds packed with spheres. *Powder Technology*, 35:23–29, 1983.

[51] H. D. Gougar, W. K. Terry, and A. M. Ougouag. Matrix formulation of pebble circulation in the pebbed code. In *Proceedings of the Conference on High-Temperature Reactors, Petten, NL*, Vienna, Austria, April 2002. International Atomic Energy Agency.

[52] P. A. Gremaud and J. V. Matthews. On the computation of steady hopper flows: I. stress determination for coulomb materials. *J. Comput. Phys.*, 166:63–83, 2001.

[53] P. A. Gremaud, J. V. Matthews, and M. O'Malley. On the computation of steady hopper flows: Ii. von mises materials in various geometries. *J. Comput. Phys.*, 200:639–653, 2004.

[54] P. A. Gremaud, J. V. Matthews, and D. G. Schaeffer. On the computation of steady hopper flows: Iii. model comparisons. *J. Comput. Phys.*, 219:443–454, 2006.

[55] C. Guáqueta. Computer simulations of a stochastic model for granular drainage, 2003.

[56] C. h. Liu, S. R. Nagel, D. A. Schecter, N. Coppersmith, S. Majumdar, O. Narayan, and T. A. Witten. Force fluctuations in bead packs. *Science*, 269:513–515, 1995.

[57] D. W. Howell, R. P. Behringer, and C. T. Veje. Stress fluctuations in a 2D granular couette experiment: A continuous transition. *Phys. Rev. Lett.*, 82(26):5241–5244, 1999.

[58] S. Hu and R. Wang. Power operation commissioning tests of htr-10. In *Proceedings of the 2nd International Topical Meeting on High Temperature Reactor Technology, Beijing, China*. Institute for Nuclear and New Energy Technology, September 22–24 2004.

[59] H. M. Jaeger and S. R. Nagel. Physics of the granular state. *Science*, 255:1523–1531, 1992.

[60] H. M. Jaeger, S. R. Nagel, and R. P. Behringer. Granular solids, liquids, and gases. *Rev. Mod. Phys.*, 68:1259–1273, 1996.

[61] J. T. Jenkins and S. B. Savage. A theory for the rapid flow of identical, smooth, nearly elastic particles. *J. Fluid Mech.*, 130:187–202, 1983.

[62] P. Jop, Y. Forterre, and O. Pouliquen. A constitutive law for dense granular flows. *Nature*, 441:727–730, 2006.

[63] A. Kadak and M. Z. Bazant. Pebble-flow experiments for pebble-bed reactors. In *Proceedings of the 2nd International Topical Meeting on High Temperature Reactor Technology, Beijing, China*. Institute for Nuclear and New Energy Technology, September 22–24 2004.

[64] L. P. Kadanoff. Built upon sand: Theoretical ideas inspired by the flow of granular materials. *Rev. Mod. Phys.*, 71:435–444, 1999.

[65] K. Kamrin and M. Z. Bazant. Stochastic flow rule for granular materials. *Phys. Rev. E*, 75:041301, 2007.

[66] K. Kamrin, C. H. Rycroft, and M. Z. Bazant. The stochastic flow rule: A multi-scale model for granular plasticity. *Modelling. Simul. Mater. Sci. Eng.*, 15:S449–S464, 2007.

[67] A. R. Kansal, S. Torquato, and F. H. Stillinger. Diversity of order and densities in jammed hard-particle packings. *Phys. Rev. E*, 66:041109, 2002.

[68] D. V. Khakhar, J. J. McCarthy, T. Shinbrot, and J. M. Ottino. *Phys. Fluids*, 9:31, 1997.

[69] Z. S. Khan and S. W. Morris. Subdiffusive axial transport of granular materials in a long drum mixer. *Phys. Rev. Lett.*, 94:048002, 2005.

[70] Z. S. Khan, W. A. Tokaruk, and S. W. Morris. Oscillatory granular segregation in a long drum mixer. *Europhys. Lett.*, 66:212, 2004.

[71] J. W. Landry, G. S. Grest, L. E. Silbert, and S. J. Plimpton. Confined granular packings: structure, stress, and forces. *Phys. Rev. E*, 67:041303, 2003.

[72] M. Latzel, S. Luding, H. J. Herrmann, D. W. Howell, and R.P. Behringer. Comparing simulation and experiment of a 2D granular couette shear device. *Euro. Phys. Journ. E.*, 11:325–333, 2003.

[73] A. Lemaître. A dynamical approach to glassy materials. 2002.

[74] A. Lemaître. Origin of a repose angle: Kinetics of rearrangements for granular materials. *Phys. Rev. Lett.*, 89:064303, 2002.

[75] A. Lemaître. Rearrangements and dilatency for sheared dense materials. *Phys. Rev. Lett.*, 89:195503, 2002.

[76] J. Litwiniszyn. Statistical methods in the mechanics of granular bodies. *Rheol. Acta*, 2/3:146, 1958.

[77] J. Litwiniszyn. *Bull. Acad. Pol. Sci.*, 9:61, 1963.

[78] J. Litwiniszyn. The model of a random walk of particles adapted to researches on problems of mechanics of loose media. *Bull. Acad. Pol. Sci.*, 11:593, 1963.

[79] A. J. Liu and S. R. Nagel. Jamming is not just cool any more. *Nature (London)*, 396:21, 1998.

[80] W. Losert, L. Bocquet, T. C. Lubensky, and J. P. Gollub. Particle dynamics in sheared granular matter. *Phys. Rev. Lett.*, 85:1428–1431, 2000.

[81] B. D. Lubachevsky and F. H. Stillinger. Geometric properties of random disk packings. *J. Stat. Phys.*, 60:561–583, 1990.

[82] T. S. Majumdar and R. P. Behringer. Contact force measurements and stress-induced anisotropy in granular materials. *Nature*, 435:1079–1082, 2005.

[83] A. Medina, J. Andrade, and C. Trevino. Experimental study of the tracer in the granular flow of a 2D silo. *Physics Letters A*, 249:63–68, 1998.

[84] A. Medina, J. A. Cordova, E. Luna, and C. Trevino. Velocity field measurements in granular gravity flow in a near 2D silo. *Physics Letters A*, 220:111–116, 1998.

[85] M. Menon and D. J. Durian. Diffusing-wave spectroscopy of dynamics in a three-dimensional granular flow. *Science*, 275:1920–1922, 1997.

[86] E. A. Merritt and D. J. Bacon. Raster3D: Photorealistic molecular graphics. *Meth. Enzymol.*, 277:505–524, 1997.

[87] G. D. R. Midi. On dense granular flows. *Euro. Phys. Journ. E.*, 14:341–365, 2004.

[88] D. E. Mueth, G. F. Debregeas, G. S. Karczmar, P. J. Eng, S. R. Nagel, and H. M. Jaeger. Signatures of granular microstructure in dense shear flows. *Nature*, 406:385–388, 2000.

[89] D. M. Mueth, H. M. Jaeger, and S. R. Nagel. Force distribution in a granular medium. *Phys. Rev. E*, 57:3164–3169, 1998.

[90] J. Mullins. Stochastic theory of particle flow under gravity. *J. Appl. Phys.*, 43:665, 1972.

[91] J. Mullins. Experimental evidence for the stochastic theory of particle flow under gravity. *Powder Technology*, 9:29, 1974.

[92] J. Mullins. Critique and comparison of two stochastic theories of gravity-induced particle flow. *Powder Technology*, 23:115–119, 1979.

[93] V. V. R. Natarajan, M. L. Hunt, and E. D. Taylor. *J. Fluid Mech.*, 304:1, 1995.

[94] R. M. Nedderman. *Statics and Kinematics of Granular Materials*. Nova Science, 1991.

[95] R. M. Nedderman and U. Tüzün. Kinematic model for the flow of granular materials. *Powder Technology*, 22:243, 1979.

[96] M. Newey, J. Ozik, S. M. van der Meer, E. Ott, and W. Losert. Band-in-band segregation of multidisperse granular mixtures. *Europhys. Lett.*, 66:205, 2004.

[97] D. Nicholls. The pebble-bed modular reactor. *Nuclear News*, 44, 2001.

[98] C. S. O'Hern, S. A. Langer, A. J. Liu, and S. R. Nagel. *Phys. Rev. Lett.*, 88:075507, 2002.

[99] C. S. O'Hern, L. E. Silbert, A. J. Liu, and S. R. Nagel. Jamming at zero temperature and zero applied stress: The epitome of disorder. *Phys. Rev. E*, 68:011306, 2003.

[100] G. Y. Onoda and E. G. Liniger. Random loose packing of uniform spheres and the dilatancy onset. *Phys. Rev. Lett.*, 64:2727, 1990.

[101] A. V. Orpe and D. V. Khakhar, 2001.

[102] A. V. Orpe and A. Kudrolli. Velocity correlations in dense granular flows observed with internal imaging. *Phys. Rev. Lett.*, 98:238001, 2007.

[103] A. M. Ouguag, J. J. Cogliati, and J.-L. Kloosterman. Methods for modeling the packing of fuel elements in pebble-bed reactors. In *Proceedings of the Topical Meeting on Mathematics and Computation, Avignon, France*, LaGrange Park, IL, September 2005. American Nuclear Society.

[104] J. Palacci. Computer simulations of granular materials with periodic boundary conditions using the spot model. Technical report, MIT Department of Mathematics, 2005.

[105] S. Patankar. *Numerical Heat Transfer and Fluid Flow*. Taylor & Francis, 1980.

[106] E. B. Pitman and D. G. Schaeffer. Stability of time dependent compressible granular flow in two dimensions. *Commun. Pure. Appl. Math.*, 40:421–447, 1987.

[107] O. Pouliquen. *Phys. Fluids*, 11:542, 1999.

[108] J. R. Prakash and K. K. Rao. *J. Fluid Mech.*, 225:21, 1991.

[109] J. Rajchenbach. *Phys. Rev. Lett.*, 65:2221, 1990.

[110] S. Reiss. Let a thousand reactors bloom. *Wired Magazine*, 12.09, 2004.

[111] H. Risken. *The Fokker-Planck Equation*. Springer, 1996.

[112] C. H. Rycroft, M. Z. Bazant, G. S. Grest, and J. W. Landry. Dynamics of random packings in granular flow. *Phys. Rev. E*, 73:051306, 2006.

[113] C. H. Rycroft, G. S. Grest, J. W. Landry, and M. Z. Bazant. Analysis of granular flow in a pebble-bed nuclear reactor. *Phys. Rev. E*, 74:021306, 2006.

[114] A. Samadani, A. Pradhan, and A. Kudrolli. Size segregation of granular matter in silo drainage. *Phys. Rev. E*, 60:7203–7209, 1999.

[115] S. B. Savage. Gravity flow of cohesionless granular materials in chutes and channels. *J. Fluid Mech.*, 92:53–96, 1979.

[116] S. B. Savage. *J. Fluid Mech.*, 377:1, 1998.

[117] D. G. Schaeffer. Instability in the evolution equations decsribing incompressible granular flow. *J. Diff. Eq.*, 66:19–50, 1987.

[118] A. Schofield and C. Wroth. *Critical State Soil Mechanics*. McGraw-Hill, 1968.

[119] A. J. Sederman, P. Alexander, and L. F. Gladden. Structure of packed beds probed by magnetic resonance imaging. *Powder Technology*, 117:255–269, 2001.

[120] T. Shinbrot. The brazil nut effect – in reverse. *Nature*, 429:352–353, 2004.

[121] T. Shinbrot and F. J. Muzzio. Reverse buoyancy in shaken granular beds. *Phys. Rev. Lett.*, 81:4365, 1998.

[122] S. Siavoshi, A. V. Orpe, and A. Kudrolli. *Phys. Rev. E*, 73:010301(R), 2006.

[123] L. E. Silbert, D. Ertaş, G. S. Grest, T. C. Halsey, D. Levine, and S. J. Plimpton. Granular flow down an inclined plane: Bagnold scaling and rheology. *Phys. Rev. E*, 64(5):051302, Oct 2001.

[124] L. E. Silbert, D. Ertaş, G. S. Grest, T. C. Halsey, and D. Levine. Geometry of frictionless and frictional sphere packings. *Phys. Rev. E*, 65:031304, 2002.

[125] L. E. Silbert, G. S. Grest, and J. W. Landry. Statics of the contact network in frictional and frictionless granular packings. *Phys. Rev. E*, 66:061303, 2002.

[126] L. E. Silbert, J. W. Landry, and G. S. Grest. Granular flow down a rough inclined plane: transition between thin and thick piles. *Phys. Fluids*, 15:1, 2003.

[127] V. V. Sokolovskii. *Statics of Granular Materials*. Pergamon/Oxford, 1965.

[128] F. Spaepen. A microscopic mechanism for steady state inhomogeneous flow in metallic glasses. *Acta Metallurgica*, 25:407–415, 1977.

[129] J. Sun, F. Battaglia, and S. Subramaniam. Dynamics and structures of segregation in a dense, vibrating granular bed. *Phys. Rev. E*, 74:061307, 2006.

[130] David Talbot. The next nuclear power plant. *MIT Technology Review*, pages 54–59, Jan/Feb 2002.

[131] W. K. Terry, H. D. Gougar, and A. M. Ougouag. Direct deterministic method for neutronics analysis and computation of asymptoptic burnup distribution in a recirculating pebble-bed reactor. *Annals of Nuclear Energy*, 29:1345–1364, 2002.

[132] S. Torquato. *Random Heterogeneous Materials*. Springer, New York, 2003.

[133] S. Torquato and F. H. Stillinger. Local density fluctuations, hyperuniformity, and order metrics. *Phys. Rev. E*, 68:041113, 2003.

[134] S. Torquato, T. M. Truskett, and P. G. Debenedetti. Is random close packing of spheres well defined? *Phys. Rev. Lett.*, 84:2064, 2000.

[135] J.-C. Tsai and J. P. Gollub. Slowly sheared dense granular flows: Crystallization and nonunique final states. *Phys. Rev. E*, 70:031303, 2004.

[136] J.-C. Tsai, G. A. Voth, and J. P. Gollub. Internal granular dynamics, shear-induced crystallization, and compaction steps. *Phys. Rev. Lett.*, 91:064301, 2003.

[137] U. Tüzün and R. M. Nedderman. Experimental evidence supporting the kinematic modelling of the flow of granular media in the absence of air drag. *Powder Technolgy*, 23:257, 1979.

[138] B. Utter and R. P. Behringer. Transients in sheared granular matter. *Eur. Phys. J. E*, 14:373–380, 2004.

[139] C. T. Veje, D. W. Howell, and R. P. Behringer. Kinematics of a two-dimensional granular couette experiment at the transition to shearing. *Phys. Rev. E*, 59:739–745, 1999.

[140] G. Voronoi. Nouvelles applications des paramètres continus à la théorie des formes quadratiques. *Journal f̈ur die Reine und Angewandte Mathematik*, 133:97–198, 1908.

[141] D. Vortmeyer and J. Schuster. Evaluation of steady flow profiles in rectangular and circular packed beds by a variational method. *Chemical Engineering Science*, 38:1691–1699, 1983.

[142] E. R. Weeks, J. C. Crocker, A. C. Levitt, A. Schofield, and D. A. Weitz. Three-dimensional direct imaging of structural relaxation near the colloidal glass transition. *Science*, 287:627–631, 2000.

[143] S. M. White and C. L. Tien. Analysis of flow channelling near the wall in packed beds. *Wärme- und Stoffübertragung*, 21:291–296, 1987.

[144] H. P. Zhu and A. B. Yu. Steady-state granular flow in a three-dimensional cylindrical hopper with flat bottom: microscopic analysis. *J. Phys. D*, 37:1497, 2004.