

Fast-forward the Sedimentation of Solid Particles in Protoplanetary Disks

Eric Andersson

Lund Observatory
Lund University



2017-EXA109

Degree project of 15 higher education credits
January 2017

Supervisor: Chao-Chin Yang

Lund Observatory
Box 43
SE-221 00 Lund
Sweden

Abstract

Numerical simulations have become an essential tool for studying particle-gas systems. One such system is the protoplanetary disk consisting of vast number of solid particles interacting with the surrounding gas. However, the particles tend to have a non-uniform distribution due to sedimentation, among other processes, creating an imbalanced workload for computers in different regions of the disk. In this thesis we present an adaptive-particle algorithm which relieves this imbalance, reducing the computational task in dense areas by merging particles while allowing sparse areas to be efficiently probed by splitting particles. Using the assignment schemes from the Particle-Mesh methods, including Nearest-Grid-Point and Cloud-In-Cell, we split and merge particles while conserving all important physical properties of the particles on the grid. Coupled to a Runge-Kutta integrator of order three, the algorithm was tested on a simulation model for the sedimentation process of particles in a turbulent gas disk. The theoretical density profile was compared to the numerical ones with and without adapting particles, and they were in good agreement in most cases. Moreover, while the simulations without adaptive particles failed to sample the particle properties at high altitudes of the disk, those with adaptive particles successfully reproduced the density profile in the same region. Finally, the adaptive-particle algorithm maintains a roughly equal number of particles per cell at all time, and thus demonstrates its ability to balance the workload over the entire computational domain.

Populärvetenskaplig beskrivning

Är vi och vår Jord unik i universum? Ända sedan Galileo Galilei byggde den första stjärnkikaren så har vi blickat ut över alla de stjärnor som utgör vårt himlavalv. Människans naturliga nyfikenhet, vilken visat vägen till ekosystemets topp, skapar även en strävan att förstå vår tillvaro här på jorden. Under de senaste årtionderna så har ett flertal rymdfärder gett en ny syn på huruvida vi är ensamma i universum. Vi trodde tidigare att solsystemet var något mycket ovanligt, men det har visat sig att mer än hälften utav alla stjärnor har planeter kring sig. En del utav dessa planeter är mycket lik vår jord.

Så vad har detta för konsekvens för den vetenskap som bedrivs om planeter? Att vi hade så fel om Vintergatans planetsystem har skapat ett intresse för hur planeter skapas. Många teorier har framförts, och den som studeras i min tes förklarar hur planeter föds ur jättelika gasmoln som omger nyfödda stjärnor. Likt pizzadegen som snurras utav pizzabagaren på din lokala pizzeria, så kommer dessa gasmoln att sprida sig i en stor skiva omkring stjärnan. Ur denna skiva så föds de planeter som efter en tid skulle kunna vara hem åt utomjordingar.

En process som är mycket viktig för skivans uppkomst kallas sedimentation. Denna process innebär att små partiklar, så som stenar och damm, samt gas lägger sig i en skiva kring stjärnan. För att förstå denna process så används super-datorer på vilka vi gör beräkningar i flera månader. Ett stort problem är att många vill använda dessa datorer och det är dessutom mycket dyrt att köra dem. Därför föreslår jag i mitt projekt en metod som ska göra beräkningarna lättare för datorer. Jag använder mig utav partiklar som kan ändra sina egenskaper utan att påverka resultatet av beräkningarna. Förändringarna hos partiklarna har potential att göra beräkningarna många gånger snabbare jämfört med tidigare.

Denna metod som vi kallar för anpassningsbara partiklar, kommer inte bara sänka kostnader för forskningen som bedrivs utan även ge möjlighet att detaljerat undersöka hur planeter skapas från början till slut. Detta är något som inte gjorts tidigare då det är för tidskrävande även för världens snabbaste datorer. Detta kanske ger oss möjligheten att en gång för alla svara på huruvida vi är unika eller om det finns andra ute i universum.

Contents

1	Introduction	5
2	Sedimentation of a single particle	7
2.1	Equations of motion	7
2.2	Damped harmonic oscillator	8
2.3	Numerical integration	9
2.4	Validation of the numerical integrator	9
3	Sedimentation of a swarm of particles in turbulent gas	12
3.1	Random process	12
3.2	Equilibrium solution	13
3.3	Particle-Mesh method	14
3.4	Validation of the numerical integration	16
4	Algorithm for adaptive particles	22
4.1	Splitting particles	22
4.2	Merging particles	23
4.2.1	Nearest-Grid-Point	24
4.2.2	Cloud-In-Cell	25
4.2.3	Triangular-Shaped-Cloud	27
5	Numerical simulations using adaptive particles	28
5.1	Nearest-Grid-Point	29
5.2	Cloud-In-Cell	31
5.3	Resolution study	32
6	Summary	38
6.1	Sedimentation of solid particles in protoplanetary disks	38
6.2	Algorithm for adapting particles	39
6.3	Further research	40

List of Figures

2.1	Sedimentation of a single particle in a stationary gas, showing evolution in both position (top panel) and velocity (bottom panel) for three different dimensionless stopping time τ_s . Time is shown in dimensionless units Ωt on the x-axis. Blue crosses represent data points from the third-order Runge-Kutta integration, while the red solid line shows the analytic solution of the corresponding damped harmonic oscillator.	11
3.1	Various density functions and their corresponding weight function from the Particle-Mesh method. The left panel shows how mass is continuously distributed over a finite size for a particle (red dot) with the Nearest-Grid-Point (NGP), Cloud-In-Cell (CIC) and Triangular-Shaped-Cloud (TSC) schemes. The mass assigned to each cell is equal to the area under the respective curve within the cell boundary. The weight function $W(z)$ is shown as a function of particle displacement from cell center in the right panel for NGP, CIC, TSC.	15
3.2	The square of the standard deviation σ_z^2 of the position z as a function of dimensionless time Ωt . The Stokes numbers used were (a) $St = 10$ and (b) $St = 100$. Gravity was turned off, hence the particles undergo a random walk from their initial position, set at $z = 0$ for all particles. The number of particles was 10000 for both plots. The fit was made by optimizing values m , and c for a linear function $\sigma = mt + c$	18
3.3	Histograms showing number of particles at each location around the mid-plane of the disk after reaching equilibrium. Stokes number used was (a) $St=10$ and (b) $St=100$. Simulations were run with $\tau_s = 0.1$ for 10000 particles. The fitted curve is a Gaussian (see Equation 3.3) with zero mean and standard deviation σ_z given in Table 3.1.	20
3.4	The density profile of the particles around the mid-plane, calculated by using three different particle-mesh assignment schemes: Nearest-Grid-Point (NGP), Cloud-In-Cell (CIC) and Triangular-Shaped-Cloud (TSC). Mesh density is displayed on y-axis, while the x-axis shows vertical displacement from the mid-plane. Stokes numbers used were (a) $St = 10$, and (b) $St = 100$. The grid has 64 cells. Histograms of the particle distribution which are normalized are plotted in the background (gray) for comparison.	21

LIST OF FIGURES

5.1	The particle density (top panel) and distribution (bottom panels) profiles from the simulations of adaptive particles with the Nearest-Grid-Point scheme. The plots are snapshots at the last step in the simulation, that is after 5000 iterations. The grid has a resolution of 32 points. Each plot includes the theoretical solution, along with the profiles both with and without adaptation. The four different plots show different ranges $[n_{\text{low}}, n_{\text{high}}]$ used for the adaptation.	34
5.2	The particle density (top panel) and distribution (bottom panels) profiles from the simulations of adaptive particles with the Cloud-In-Cell scheme. The plots are snapshots at the last step in the simulation, that is after 5000 iterations. The grid has a resolution of 32 points. Each plot includes the theoretical solution, along with the profiles both with and without adaptation. The four different plots show different ranges $[n_{\text{low}}, n_{\text{high}}]$ used for the adaptation.	36
5.3	Equilibrium density profiles from simulations with adaptive particles with (a) the Nearest-Grid-Point scheme and (b) the Cloud-In-Cell scheme, using two different resolutions. Also included are the theoretical profiles in red dashed lines. The black line shows simulations using a grid with $n_c = 256$ cells while the blue line shows simulation using $n_c = 32$ cells for the Particle-Mesh method.	37

List of Tables

3.1	Measurements of the diffusion coefficient and the standard deviation of the particle distribution for two Stokes numbers. σ_z is the standard deviation in the positions of the particles, while σ_ρ is the standard deviation of the density distribution on the mesh. Also included is the theoretical scale height H_p calculated by Equation 3.13, which should be consistent with the standard deviation of the particle position.	17
5.1	Measured properties of the particle distribution in simulations with adaptive particles with the Nearest-Grid-Point scheme. σ_{z_p} and σ_ρ (calculated using Equations 5.1 and 5.3 respectively) are the standard deviation in particle position around the mid-plane, which correspond to the scale height of the particles. The last two columns shows $\langle n_i \rangle$, which is the average number of particles in the cells with corresponding standard deviation σ_{n_i} . *The last row lists the measurement obtained when not using the adaptation algorithm for comparison.	30
5.2	Measured properties of the particle distribution in simulations with adaptive particles with the Cloud-In-Cell scheme. σ_{z_p} and σ_ρ (calculated using Equations 5.1 and 5.3 respectively) are the standard deviation in particle position around the mid-plane, which correspond to the scale height of the particles. The last two columns shows $\langle n_i \rangle$, which is the average number of particles in the cells with corresponding standard deviation σ_{n_i} . *The last row lists the measurement obtained when not using the adaptation algorithm for comparison.	31

Chapter 1

Introduction

Numerical simulations is an important tool in all fields of physics. The reason is that most physical systems are nonlinear, giving a convoluted evolution. Such systems are often analytically intractable. However, in order to justify theoretical models we require them to provide definitive predictions, which can be compared to experiments and/or observations. When analytic solutions are not accessible, approximate solutions can be obtained using numerical methods. A numerical method discretizes the differential equations in question so that the equations become algebraic and thus an approximate solution can be obtained. We implement these methods into computer programs in order to probe the time evolution of the system for an extended period of time with high accuracy.

Different physical processes are described by different models, and therefore, we use different numerical methods to simulate them. To describe the dynamics of a fluid, and its interactions with the surroundings, we often use the Eulerian formalism of hydrodynamics. This approach dictates that any property of the fluid, for example, the density, is a function of spatial position and time. It is effective since we do not need to consider the individual atoms in the fluid. Numerical methods that compute such systems often discretizes the space into a fixed grid, on which the properties of the fluid is based.

Methods treating interactions between gas and solid particles has come into focus in many fields of physics. Solid particles can be formulated as either a collisionless fluid or individual Lagrangian particles. The Lagrangian approach traces the exact position and velocity of each particle, therefore, better samples the phase space of the system. However, given the fundamental difference between a Eulerian gas and Lagrangian particles in their interpretation of space, some kind of mapping between a fixed grid and each of the particles is required before one could treat their interactions. This can be achieved by the particle-mesh method (see, e.g., Hockney and Eastwood, 1981).

Although the Lagrangian treatment of particles has its advantages, it sets a high demand on processing power as the system increases in size. Since the number of calculations needed scales with the number of particles in the system, more computing time is required when simulating more particles as well as the need for higher-resolution grids. Nevertheless, this issue can be remedied by utilizing parallel computing.

With parallel computing, multiple processors are used to divide the number of calcu-

lations. One way to divide the calculations is to have each processor covering a subset of the computational domain. The particles will then be handled by the CPU which covers the domain in which it resides. To fully utilize the power of parallel computing in this approach we want to ensure that the particles are roughly equally distributed in the computational domain. However, systems in which the particle distribution is nonuniform are common, and thus the workload on each processor can become wildly different. This can significantly reduce the efficiency of parallel computing.

One system, which has such a nonuniform distribution of particles, is the protoplanetary disk. These disks consist of a vast number of particles surrounded by gas. The gas disk which is supported by its own pressure gradient is thick while particles sediment into a dense layer around the mid-plane of the disk. It is believed that the gas disk is perturbed by turbulence due to the magneto-rotational instability (see, e.g., Hawley et al., 1995; Armitage, 2011). Particle-gas interactions cause some energy to be transferred from the gas to the particles, and hence increase the velocity dispersion of the particles. This gives the particle layer a finite thickness albeit much less than the thickness of the gas disk.

In this project we suggest and test a solution to resolve the issue of the load balancing in particle-gas simulations with an inhomogeneous particle distribution. We introduce an algorithm which adapts the number of particles in different regions to maintain an approximately uniform number density. Particles residing in sparse areas will split, increasing in number, while particles in clusters will merge into larger particles. This ensures that the particles are roughly evenly distributed over the entire computational domain. The goal of the method is to balance the workload while preserving the physical properties these particles represent, and thus improve the efficiency of parallel processing.

In the first part of this thesis, we establish the foundation for the sedimentation process of particles in a protoplanetary disk, both theoretically and numerically. In Chapter 2 we discuss in detail how a single particle behaves when settling towards the mid-plane of a stationary gaseous disk. We expand on this in Chapter 3, and model the sedimentation of a swarm of particles in turbulent gas. We compare the theoretical expectation of the resulting particle distribution with the numerically integrated system to ensure that the validity of our numerical integrator before we introduce the particle-adapting algorithm. We also explain the particle-mesh method in detail, which will become an important ingredient for our algorithm of adapting particles.

Finally, we present the algorithm for the adaptive particles in Chapter 4 and the simulations of particle sedimentation in a turbulent gas with the algorithm in Chapter 5. In Chapter 6 we summarize the results, discuss the implications of the new method and suggest possible further research.

Chapter 2

Sedimentation of a single particle

In this chapter the theory regarding the sedimentation of a single particle is described. It is the process in which solid particles settle onto the mid-plane of a gaseous disk surrounding a newborn star, and this process is the first of many leading to the formation of planetary systems (see, e.g., Armitage, 2010).

2.1 Equations of motion

Before describing the equation of motion for a single particle, some simplifying assumptions are needed. In our system we consider one particle, located at a large distance from the star, slightly above the gas disk. We are only interested in vertical motion and thus ignore the horizontal components.

The particle accelerates toward the mid-plane due to the gravity of the central star. The mass of the system is dominated by the mass of the star, therefore it is assumed that the gravitational force from any other source is negligible. The particle is in orbit around the star which is located at the origin and for simplicity we assume that the radial distance for the particle in cylindrical coordinates is nearly constant. Hence, the important part is only the vertical component of the gravitational acceleration g . The coordinates are r , the radial distance from the star in the mid-plane, and z , the displacement perpendicular to r above the plane. We define θ as the angle between the mid-plane and a vector from the star pointing towards the particle. The vertical component of the gravitational acceleration is then

$$g_z = g \sin(\theta) = -\frac{GM_*}{r^2 + z^2} \frac{z}{\sqrt{r^2 + z^2}} = -\frac{GM_* z}{(r^2 + z^2)^{3/2}}, \quad (2.1)$$

where G is the gravitational constant and M_* is the mass of the star. It is assumed that z is very small compared to r , therefore we may write

$$g_z = -\frac{GM_* z}{r^3} = -\Omega^2 z, \quad (2.2)$$

where Ω is the Keplerian angular frequency with which the particle oscillates about the mid-plane.

2.2. DAMPED HARMONIC OSCILLATOR

Eventually, oscillations fade and the particle settles onto the disk mid-plane. The settling is due to a drag force slowing the particle down. This force arises from its interaction with the surrounding gas and therefore depends on many factors which characterize the gas and the particle. However, for our purposes, we make use of the stopping time t_s , which incorporates all these factors and characterizes the timescale for the reduction of the velocity difference between the particle and its surrounding gas. With this simplifying parameter the drag force can be written as

$$\mathbf{F}_D = m_p \frac{\mathbf{u} - \mathbf{v}}{t_s}, \quad (2.3)$$

where m_p is the mass of the particle, \mathbf{u} is the velocity of the surrounding gas and \mathbf{v} is the velocity of the particle.

Combining the gravitational force and the drag force, we obtain the equation of motion for the particle. Newton's second law gives

$$\frac{d^2 z}{dt^2} = \frac{u_z - v_z}{t_s} - \Omega^2 z, \quad (2.4)$$

where the first term represents the drag force, and the second term is due to the gravity of the star. It is convenient to use a dimensionless version of the stopping time $\tau_s = \Omega t_s$. For simplicity, we assume that the stopping time is constant. The motion of a particle with an initial position above, or below, the mid-plane tends to oscillate about the plane with a damping determined by the dimensionless stopping time τ_s .

2.2 Damped harmonic oscillator

If the particle moves through a stationary gas (i.e. $u_z = 0$), it constantly feels a headwind, and hence undergoes damped harmonic oscillation about the mid-plane of the disk. Using z as the displacement, the general equation describing such motion is

$$\frac{d^2 z}{dt^2} + 2\zeta\omega_0 \frac{dz}{dt} + \omega_0^2 z = 0, \quad (2.5)$$

where ω_0 is the natural angular frequency of the system and ζ is the damping ratio. The behavior of the oscillator depends on the damping ratio. If $\zeta > 1$, then the oscillator is over-damped, and the system decays exponentially without oscillating. If $\zeta < 1$, it is under-damped, and the system oscillates with a decaying amplitude. In the special case of $\zeta = 1$, the oscillator is called critically damped, and the decay occurs gradually but without any oscillations. Comparing Equation 2.4 to the damped harmonic oscillator (HO) we find $\Omega = \omega_0$, and $\tau_s = (2\zeta)^{-1}$. Hence, critical damping occurs when $\tau_s = 1/2$ while under-damped for $\tau_s > 1/2$, and over-damped for $\tau_s < 1/2$. Particles are strongly coupled to the gas if $\tau_s \ll 1/2$, and loosely coupled if $\tau_s \gg 1/2$.

The harmonic oscillator described by Equation 2.5 has an analytic solution which is used to validate the numerical solution of Equation 2.4 below.

2.3 Numerical integration

To solve the differential Equation 2.4 numerically, we use the Runge-Kutta method of third order (RK3). This integrator is a well established numerical solver, which can be used to compute nonlinear, time-dependent functions. The third-order scheme has a local truncation error of order $\mathcal{O}(h^3)$, where h is the time step of the integration. This method will give enough accuracy while keeping the number of calculations low.

Suppose that we have a set of dynamical variables $u^{(n)}$ at time t_n , and we seek the values of these variables $u^{(n+1)}$ at time $t_{n+1} = t_n + h$. This problem can be solved by using the RK3 scheme, which is generally written as

$$\begin{aligned} w_i &= \alpha_i w_{i-1} + hF(t_{i-1}, u_{i-1}), \\ u_i &= u_{i-1} + \beta_i w_i, \end{aligned} \tag{2.6}$$

where $i = 1, 2, 3$ and

$$\begin{cases} u^{(n)} = u_0 \\ u^{(n+1)} = u_3 \end{cases} \tag{2.7}$$

In Equations 2.6, u_i and w_i are values obtained from intermediate calculations, α_i and β_i are constants and $F(t, u)$ is the first derivative of u with respect to time. The constants α_i and β_i are chosen such that the accuracy of the scheme is of third order.

The RK3 method is used to solve first-order differential equations. However, Equation 2.4 is a second-order differential equation. Nevertheless, Equation 2.4 can be transformed into a set of first-order equations:

$$\begin{aligned} \frac{dz}{dt} &= v_z, \\ \frac{dv_z}{dt} &= \frac{u_z - v_z}{t_s} - \Omega^2 z, \end{aligned} \tag{2.8}$$

to which the RK3 method can be applied.

Hence, we implemented the RK3 method in Python3 to integrate the system of Equations 2.8 which describes the position and velocity of a particle influenced by the stellar gravity and the gas drag. For the constant coefficients, β_i and α_i , in Equation 2.6 we use those derived by Williamson (1980).

2.4 Validation of the numerical integrator

We first validated the RK3 integrator for a single particle moving through a stationary gas, i.e., $u_z = 0$. Without loss of generality we chose $\Omega = 1$, and ran tests using various cases of the dimensionless stopping time τ_s . The RK3 integrator implemented in Python3 was tested using different step-sizes h in order to investigate the stability of the numerical method. We then compared the numerical solution of Equation 2.4 to the corresponding analytic solution of Equation 2.5.

2.4. VALIDATION OF THE NUMERICAL INTEGRATOR

Figure 2.1 shows the numerical solution for three different regimes of dimensionless stopping time: (a) under-damped $\tau_s > 1/2$, (b) over-damped $\tau_s < 1/2$ and (c) critically damped $\tau_s = 1/2$. In comparison to the numerical integration we over-plotted the analytic solution of Equation 2.5 for each damping regime. We found that the integrator gave an accurate solution if the step-size was chosen to be small enough. It was deduced empirically that if the step-size is chosen as $h \leq \tau_s$, then the integrator is stable for cases in which $\tau_s < 1$. Therefore, we used $h = \tau_s$ for simulations in this regime. For cases in which $\tau_s > 1$ we found that the solution diverges if $h > 1$. When $h \sim 1$ the numerical solution has the same period as the analytic solution, however, decreases in amplitude more rapidly. Therefore, we chose $h = 0.1$ for simulations with $\tau > 1$.

2.4. VALIDATION OF THE NUMERICAL INTEGRATOR

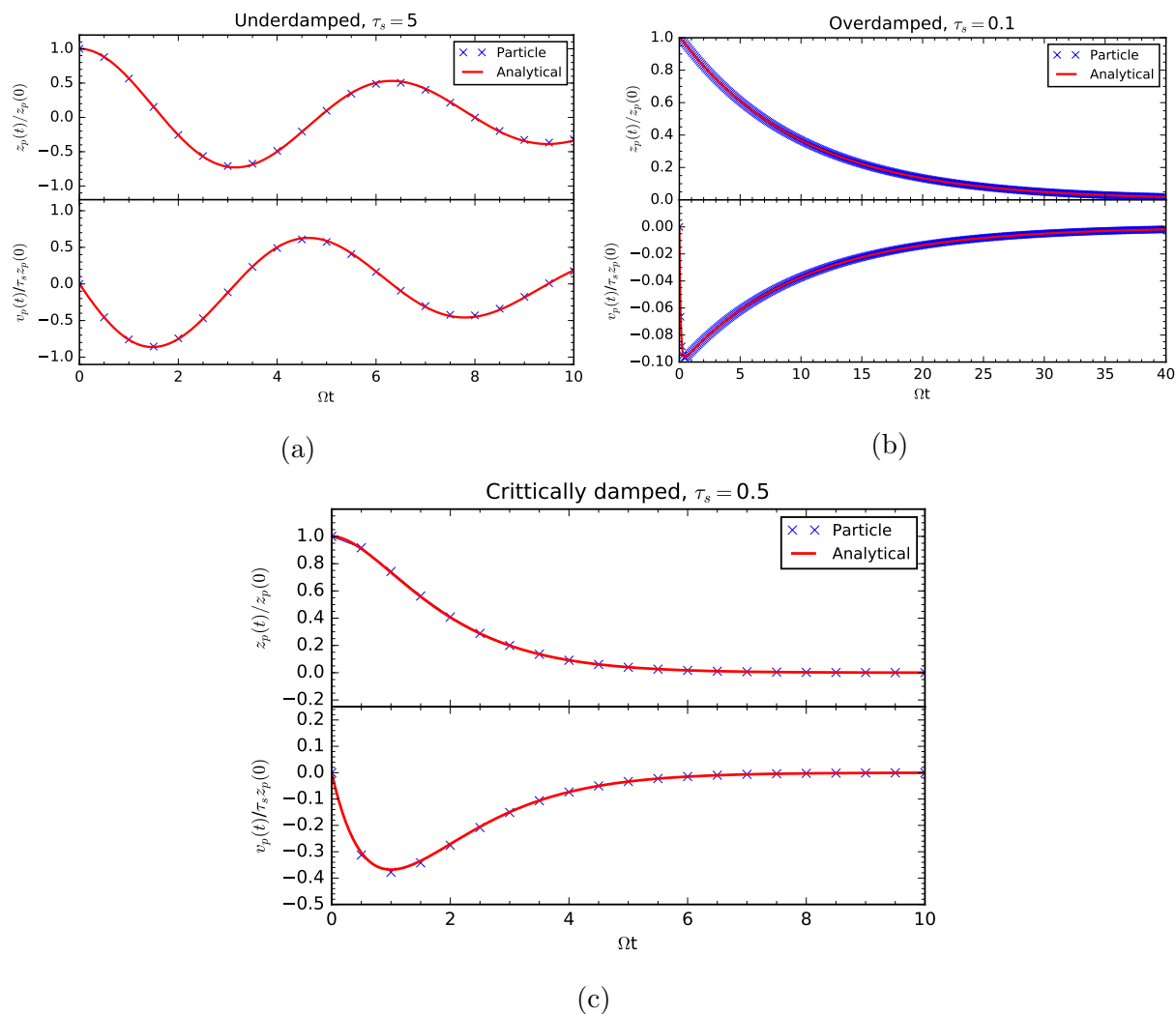


Figure 2.1: Sedimentation of a single particle in a stationary gas, showing evolution in both position (top panel) and velocity (bottom panel) for three different dimensionless stopping time τ_s . Time is shown in dimensionless units Ωt on the x-axis. Blue crosses represent data points from the third-order Runge-Kutta integration, while the red solid line shows the analytic solution of the corresponding damped harmonic oscillator.

Chapter 3

Sedimentation of a swarm of particles in turbulent gas

3.1 Random process

The drag force on a particle, described by the first term in Equation 2.4, is determined by two factors: the relative velocity between the particle and the gas and how well the particle is coupled to the gas. Random gas motions are introduced to a differentially rotating disk due to magnetohydrodynamic turbulence (see, e.g., Balbus and Hawley, 1998). These motions hinder the particles from fully settling onto the mid-plane, which results in an equilibrium state for the distribution of the particles over height.

We define the Stokes number as

$$St \equiv \frac{\tau_s}{\tau_e}, \quad (3.1)$$

where τ_e is the dimensionless characteristic time for eddy current turnovers in the turbulent gas. The Stokes number is a measure of how well a particle is coupled to fluctuations in turbulent gas. The dynamics of a particle-gas system depends strongly on the Stokes number, discussed in detail by Youdin and Lithwick (2007). In this thesis, we focus only on the regime in which $St \gg 1$ and $\tau_s \ll 1$. This implies $\tau_e \ll \tau_s \ll 1$, and therefore, the influence of the turbulent gas on the particles is impulsive compared to the response time of the particles.

The stochastic impulses cause particles to random walk in velocity space. Without any other forces, the distribution of particles then follows the diffusion equation

$$\frac{\partial f}{\partial t} = \frac{\partial}{\partial z} \left(\mathcal{D} \frac{\partial f}{\partial z} \right), \quad (3.2)$$

where \mathcal{D} is the diffusion coefficient assumed to be constant and $f = f(t, z)$ is the distribution function. Undergoing this diffusion process, an initial delta function will broaden over time and become a normal distribution, which is described by the normal distribution

3.2. EQUILIBRIUM SOLUTION

function (see, e.g., Yang et al., 2009)

$$f(z, t) = \frac{1}{\sigma(t)\sqrt{2\pi}} \exp\left[-\frac{z^2}{2\sigma^2(t)}\right], \quad (3.3)$$

where $\sigma(t)$ is the standard deviation as a function of time. Substituting Equation 3.3 into Equation 3.2, one finds the standard deviation of the system,

$$\sigma(t) = \sqrt{2\mathcal{D}t}, \quad (3.4)$$

with a time dependence of $t^{1/2}$.

3.2 Equilibrium solution

When the number of particles is so large that their collective dynamics can be approximated by a Eulerian fluid description, the transport of the particles in a turbulent gas can be described as follows (Dubrulle et al., 1995). Consider a small, virtual box bounding a fluid element, through which particles flow. Since mass is conserved, the particle density ρ_p in the box must change according to the continuity equation, including a source term from the turbulent diffusion,

$$\frac{\partial \rho_p}{\partial t} + \vec{\nabla} \cdot (\rho_p \mathbf{v}) = \mathcal{D} \frac{\partial}{\partial z} \left[\rho_g \frac{\partial}{\partial z} \left(\frac{\rho_p}{\rho_g} \right) \right], \quad (3.5)$$

where ρ_g is the density of the gas. The second term on the left hand side of Equation 3.5, $\vec{\nabla} \cdot (\rho_p \mathbf{v})$, is the net flux flowing through the box. When $\tau_s \ll 1$, the velocities of the particles are asymptotically close to the terminal velocity. The terminal velocity is obtained from Equation 2.4 with $\frac{d^2 z}{dt^2} = 0$ and $u_z = 0$,

$$0 = -\frac{v_z}{t_s} - \Omega^2 z \quad \Rightarrow \quad v_z = -z\Omega^2 t_s \quad (3.6)$$

Because we are only interested in the vertical component, we find the flux

$$\vec{\nabla} \cdot (\rho_p \mathbf{v}) = \frac{\partial}{\partial z} (\rho_p v_z) = -\frac{\partial}{\partial z} (z\Omega^2 t_s \rho_p) \quad (3.7)$$

By inserting Equation 3.7 into Equation 3.5 we obtain the transport equation,

$$\frac{\partial \rho_p}{\partial t} - \frac{\partial}{\partial z} (z\Omega^2 t_s \rho_p) = \mathcal{D} \frac{\partial}{\partial z} \left[\rho_g \frac{\partial}{\partial z} \left(\frac{\rho_p}{\rho_g} \right) \right]. \quad (3.8)$$

In Section 3.1 it was found that with only the diffusion process, the particles are distributed with a normal distribution. Therefore, we use the same distribution as a trial function for Equation 3.5, i.e.,

$$\frac{\rho_p}{\rho_g} = \left(\frac{\rho_p}{\rho_g} \right)_0 \exp\left[\frac{-z^2}{2H_p^2}\right], \quad (3.9)$$

in which H_p is the scale height of the particles. By inserting Equation 3.9 into Equation 3.8 and taking the equilibrium limit ($\frac{\partial \rho_p}{\partial t} = 0$), we find

$$0 - \frac{\partial}{\partial z} \left(z \Omega^2 t_s \rho_g \left(\frac{\rho_p}{\rho_g} \right)_0 \exp \left[\frac{-z^2}{2H_p^2} \right] \right) = \mathcal{D} \frac{\partial}{\partial z} \left[\rho_g \frac{\partial}{\partial z} \left(\left(\frac{\rho_p}{\rho_g} \right)_0 \exp \left[\frac{-z^2}{2H_p^2} \right] \right) \right] \quad (3.10)$$

$$\Leftrightarrow \left(\frac{z^2 \Omega^2 t_s \rho_p}{H_p^2} - \Omega^2 t_s \rho_p \right) \left(\frac{\rho_p}{\rho_g} \right)_0 \exp \left[\frac{-z^2}{2H_p^2} \right] = \mathcal{D} \left(\frac{z^2 \rho_p}{H_p^4} - \frac{\rho_p}{H_p^2} \right) \left(\frac{\rho_p}{\rho_g} \right)_0 \exp \left[\frac{-z^2}{2H_p^2} \right] \quad (3.11)$$

$$\Leftrightarrow \left(\frac{z^2 \Omega^2 t_s \rho_p}{H_p^2} - \Omega^2 t_s \rho_p \right) \left(\frac{\rho_p}{\rho_g} \right) = \mathcal{D} \left(\frac{z^2 \rho_p}{H_p^4} - \frac{\rho_p}{H_p^2} \right) \left(\frac{\rho_p}{\rho_g} \right). \quad (3.12)$$

For Equation 3.12 to be identity, then $\mathcal{D} = \Omega^2 t_s H_p^2$. The scale height of the particles at equilibrium is therefore

$$H_p = \sqrt{\frac{\mathcal{D}}{\Omega \tau_s}}. \quad (3.13)$$

3.3 Particle-Mesh method

In Chapter 2, we numerically evolve the position and velocity of each individual particle. However, we now require a method which computes the ensemble properties of particles in continuous space, such as particle density. A numerical tool that accomplishes this is the Particle-Mesh (PM) method. Given a fixed grid of cells, this method assigns the density of a property to each cell from the distribution of individual particles. The PM method is described in detail by Hockney and Eastwood (1981). Nevertheless, for completeness we derive the theory here, which will be needed in our algorithm for adaptive particles in Chapter 4.

In the following, the method is exemplified by assigning the masses of individual particles onto a uniform grid in one dimension. For a one dimensional system the mass density (hereafter referred to as density) in a cell, indexed i , of size Δz is defined by $\rho_i = m_i / \Delta z$, where m_i is the total mass inside that cell. Therefore, the essence of the PM method is to, in some way, distribute the mass of each particle into cells before the density can be computed. This is incorporated in a weight function $W(z)$. For a system of N particles, each with individual mass m_j , the density ρ_i assigned to cell i is in general

$$\rho_i = \frac{1}{\Delta z} \sum_{j=1}^N W(z_{p,j} - z_i) m_j. \quad (3.14)$$

Equation 3.14 can be interpreted as the mass contribution of the j -th particle at position $z_{p,j}$ to cell i located at z_i is $W(z_{p,j} - z_i) m_j$.

The rest of the question is to find an expression for the weight function. The simplest weight function is the Nearest-Grid-Point (NGP) method, in which the mass of the particle

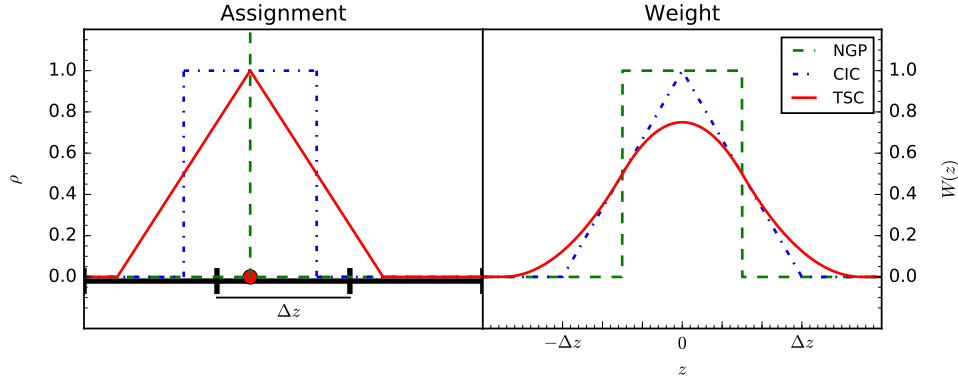


Figure 3.1: Various density functions and their corresponding weight function from the Particle-Mesh method. The left panel shows how mass is continuously distributed over a finite size for a particle (red dot) with the Nearest-Grid-Point (NGP), Cloud-In-Cell (CIC) and Triangular-Shaped-Cloud (TSC) schemes. The mass assigned to each cell is equal to the area under the respective curve within the cell boundary. The weight function $W(z)$ is shown as a function of particle displacement from cell center in the right panel for NGP, CIC, TSC.

is assigned directly to a cell if the particle is within the cell boundary. This results in the weight function

$$W_{NGP}(z) = \begin{cases} 1, & |z| \leq \frac{\Delta z}{2} \\ 0, & \text{otherwise} \end{cases} \quad (3.15)$$

which is illustrated in Figure 3.1. However, the error in this assignment is of the order of cell size.

A more accurate PM method is the Cloud-In-Cell (CIC) method. This method uniformly distributes the mass of each particle to within one half of the cell size from the particle in both directions. The fraction of mass per unit length at a distance z from the particle is then

$$S_{CIC}(z) = \frac{1}{\Delta z} \begin{cases} 1, & |z| \leq \frac{\Delta z}{2} \\ 0, & \text{otherwise} \end{cases}. \quad (3.16)$$

The mass fraction assigned to a cell is $S_{CIC}(z)$ averaged over the entire cell, which gives the weight function

$$W_{CIC}(z) = \begin{cases} 1 - \frac{|z|}{\Delta z}, & |z| \leq \Delta z \\ 0, & \text{otherwise} \end{cases}. \quad (3.17)$$

The third, most accurate PM method is the ‘Triangular-Shaped-Cloud’ (TSC). This method assigns the mass using a triangular distribution, which divides the mass on three different cells. The TSC is derived from a set of quadratic equations which are chosen in such a way that the accuracy of the assignment is second order in cell size. Consider a

particle located inside cell 0. The three neighboring cells $(-1, 0, 1)$ will have some fraction of the mass assigned to them, and in order to conserve mass we require

$$W_{-1} + W_0 + W_1 = 1. \quad (3.18)$$

The accuracy of the TSC also requires the weight functions to be the quadratic functions

$$\begin{aligned} W_0(z) &= az^2 + b, \\ W_1(z) &= cz^2 + dz + e, \\ W_{-1}(z) &= W_1(-z), \end{aligned} \quad (3.19)$$

where a, b, c, d, e are constants and z is the displacement of the particle with respect to the center of the respective cell.

To reduce fluctuations in the interactions between particles which are located close to each other, we apply a smoothness criterion. This criterion requires that variations in the the weight functions, and their derivatives to be continuous at the boundary. The quadratic functions, given in Equations 3.19, automatically satisfies this for $-\Delta z/2 < z < \Delta z/2$. To enforce this for the particle when it moves to cells $(0,1,2)$, we set the following conditions:

$$\begin{aligned} W_0(\Delta z/2) &= W_1(\Delta z/2), \\ W_{-1}(\Delta z/2) &= 0, \end{aligned} \quad (3.20)$$

$$\begin{aligned} \frac{d}{dz} W_{-1}(\Delta z/2) &= 0, \\ \frac{d}{dz} W_0(\Delta z/2) &= -\frac{d}{dz} W_1(\Delta z/2), \end{aligned} \quad (3.21)$$

Using these boundary conditions, as well as the conservation of mass set by Equations 3.18 to solve for the coefficients given in Equation 3.19, we obtain the TSC weight function, as shown in Figure 3.1,

$$W_{TSC}(z) = \begin{cases} \frac{3}{4} - \left(\frac{z}{\Delta z}\right)^2, & |z| \leq \frac{\Delta z}{2} \\ \frac{1}{2} \left(\frac{3}{2} - \frac{|z|}{\Delta z}\right)^2, & \frac{\Delta z}{2} \leq |z| \leq \frac{3\Delta z}{2} \\ 0, & \text{otherwise} \end{cases} \quad (3.22)$$

3.4 Validation of the numerical integration

In order to validate our numerical integration for a swarm of particles in a turbulent gas we updated the RK3 scheme used in Chapter 2, to work for a set of N particles. The turbulence was simulated by setting u_z in Equation 2.4 to be a stochastic variable drawn from a normal distribution $N(0, \sigma_t^2)$, where σ_t is the magnitude of the turbulent velocity. In all our simulations we used $\sigma_t = 0.1$. This results in small kicks delivered to the particles by the gas in every iteration. By choosing a step-size $h = \tau_e$ we mimic the eddy turnovers. The integrator remains stable since we assume $\tau_e \ll \tau_s$.

3.4. VALIDATION OF THE NUMERICAL INTEGRATION

Table 3.1: Measurements of the diffusion coefficient and the standard deviation of the particle distribution for two Stokes numbers. σ_z is the standard deviation in the positions of the particles, while σ_ρ is the standard deviation of the density distribution on the mesh. Also included is the theoretical scale height H_p calculated by Equation 3.13, which should be consistent with the standard deviation of the particle position.

St	τ_s	$\mathcal{D}[\times 10^5]$	$\sigma_z[\times 10^{-3}]$	$\sigma_\rho[\times 10^{-3}]$	$H_p[\times 10^{-3}]$
10	0.1	1.99382895	14.3265	14.3014	14.1203008
100	0.1	0.197588355	4.43637	4.42002	4.4450912

Measuring the diffusion coefficient

Before we can use Equation 3.13 to find the theoretical scale height of the particles at equilibrium, we need to know the diffusion coefficient \mathcal{D} of the system. Hence, we first set up a system of $N = 10000$ particles initially located at $z = 0$, equivalent to the distribution of a delta function. The acceleration resulting from the stellar gravity was turned off, thus creating freely random walking particles. As described in Section 3.1, the distribution of these particles would then diffuse into a normal distribution $N(0, \sigma^2)$ with standard deviation $\sigma(t) = \sqrt{2\mathcal{D}t}$.

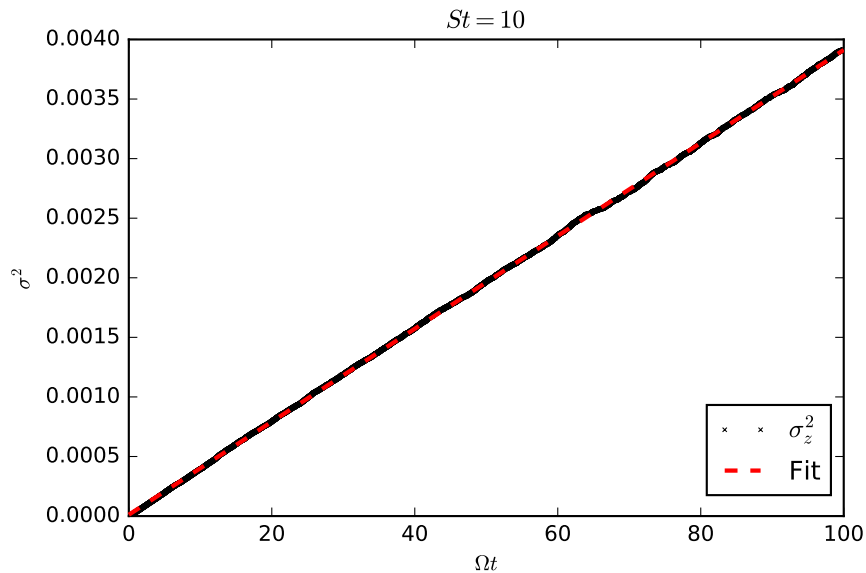
Figure 3.2 shows the square of the standard deviation σ_z^2 in position for $N = 10000$ particles as a function of time t . The resulting curve is indeed linear in time and we fitted a straight line $\sigma^2 = mt + c$ in order to compute the diffusion coefficient $\mathcal{D} = m/2$. This was tested for two different coupling regimes, $St = 10$ and $St = 100$, for particles with dimensionless stopping time $\tau_s = 0.1$. The measured value of the diffusion coefficient and the resulting scale height of particles calculated by Equation 3.13 in each case are listed in Table 3.1.

Measuring the scale height

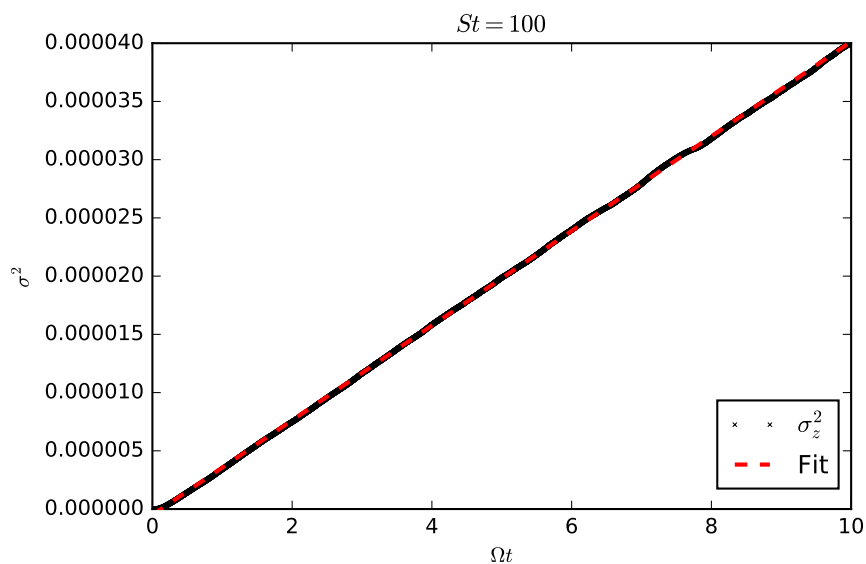
After obtaining the theoretical scale height of the particles, we were in a position to validate our numerical simulations of particle sedimentation in turbulent gas. Under the influence of stellar gravity the particles settle onto the mid-plane of the disk. However, the drag force from the turbulent gas eventually reaches dynamical equilibrium with the gravitational force. This gives the disk the scale height H_p . We let the parameters governing the turbulence remain the same as previously, that is having velocity kicks u_z drawn from a normal distribution with zero mean and $\sigma_t = 0.1$. The step size of the iterations was set to $h = \tau_e$ and the dimensionless stopping time was kept at $\tau_s = 0.1$. We reintroduced the stellar gravity and set up the simulation with 10000 particles, initially distributed evenly on the vertical axis z .

In Equation 3.9 we see that the equilibrium distribution is expected to be a Gaussian shape, with standard deviation $\sigma = H_p$. Therefore, we fitted a Gaussian curve, given by Equation 3.3, to the final distribution of the particles. Figure 3.3 shows this distribution

3.4. VALIDATION OF THE NUMERICAL INTEGRATION



(a)



(b)

Figure 3.2: The square of the standard deviation σ_z^2 of the position z as a function of dimensionless time Ωt . The Stokes numbers used were (a) $St = 10$ and (b) $St = 100$. Gravity was turned off, hence the particles undergo a random walk from their initial position, set at $z = 0$ for all particles. The number of particles was 10000 for both plots. The fit was made by optimizing values m , and c for a linear function $\sigma = mt + c$.

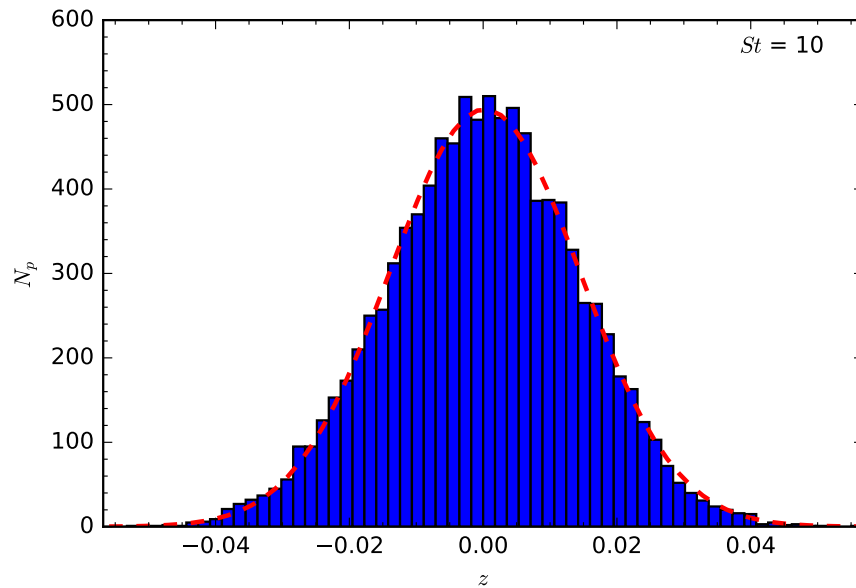
3.4. VALIDATION OF THE NUMERICAL INTEGRATION

in a histogram with 64 bins together with the fitted curve. We found that the the curve fits the distribution well for both $St = 10$ and $St = 100$. From the fitted Gaussian we obtained the standard deviation σ_z , recorded in Table 3.1. This measured scale height is in good agreement with the theoretical scale height.

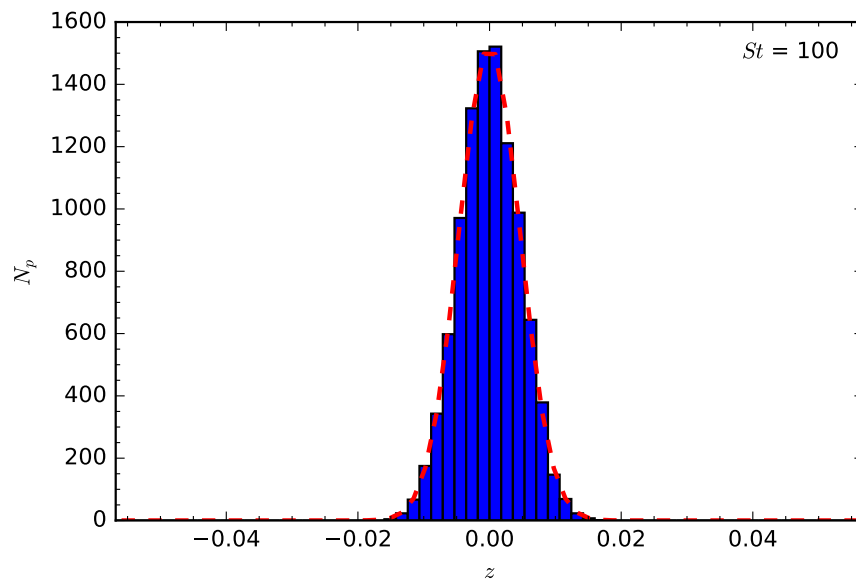
Moreover, we needed to validate if the implementation of the PM method had been successful. Therefore, we created a mesh of 64 cells, which was the same as the number of bins used in the histograms of Figure 3.3. We then used the assignment methods NGP, CIC and TSC to assign the mass onto this grid. A total mass of $M_{tot} = 1$ was divided equally among all particles. The resulting density profile is presented in Figure 3.4 for both $St = 10$ and $St = 100$. The histograms shown in Figure 3.3 were rescaled to have a total area of 1 and plotted in the background for comparison. We found that the NGP follow the histogram exactly, which is to be expected since the assignment of NGP is analogous to histogram binning. The CIC and the TSC follow the histogram as well, but the resulting distribution is smoother. This is due to the intrinsic smoothing of the assignment over multiple cells, and thus gives a better Gaussian shape. In Table 3.1 we list the standard deviations σ_ρ for the density profiles using the most accurate TSC method.

The particle distribution of our simulations have a clear Gaussian shape, therefore, are in agreement with the theoretical model. However, the Gaussian distribution has 99.7% of its distribution within three standard deviations. Therefore, in Figure 3.3(a), which probes an area that is roughly $\pm 4H_p$ in size, the outer 1/4 of the distribution is probed by less than ~ 30 particles, while 10000 particles in one dimension will amount to an intractable workload in three dimensions even in current high-performance computing. Furthermore, the number density ratio between the mid-plane and the four scale heights reaches ~ 3000 , illustrating the issue of unbalanced workload. In some hydrodynamic codes, such as PENCIL, parallel processing divides different spatial regions among multiple processors, in order to decrease the run time of the simulations. To prevent any processor from idling, each processor needs to have similar workload. In the following chapter we present an algorithm which homogenizes the particle distribution in number, thus balances the workload for the parallel computing.

3.4. VALIDATION OF THE NUMERICAL INTEGRATION



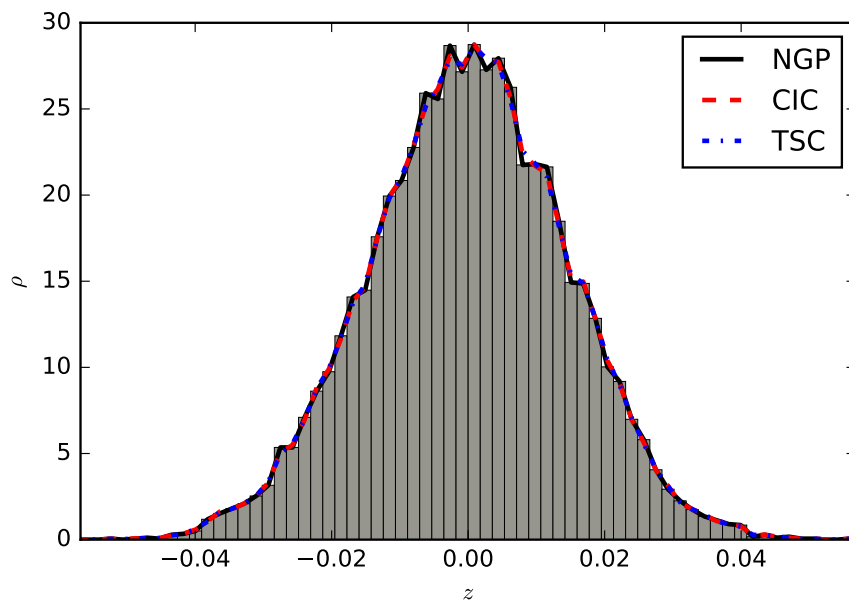
(a)



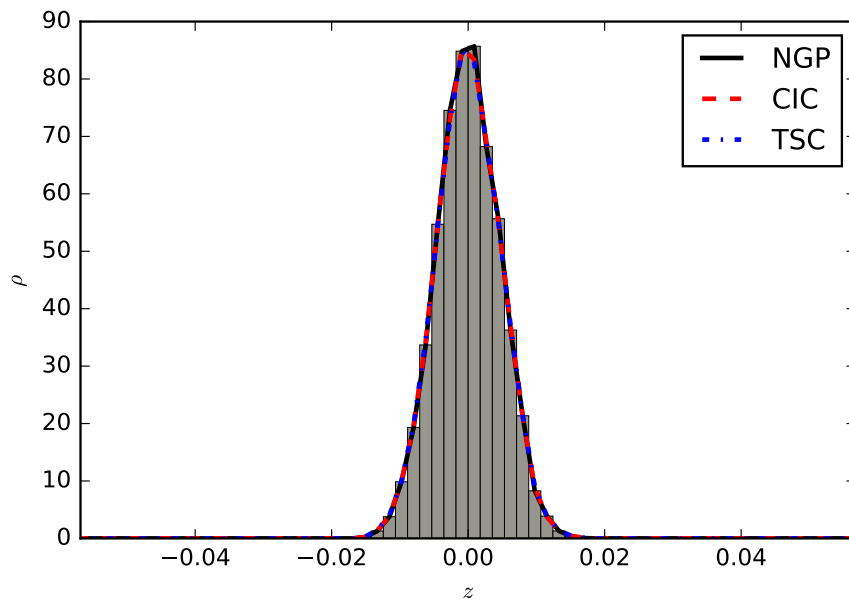
(b)

Figure 3.3: Histograms showing number of particles at each location around the mid-plane of the disk after reaching equilibrium. Stokes number used was (a) $St=10$ and (b) $St=100$. Simulations were run with $\tau_s = 0.1$ for 10000 particles. The fitted curve is a Gaussian (see Equation 3.3) with zero mean and standard deviation σ_z given in Table 3.1.

3.4. VALIDATION OF THE NUMERICAL INTEGRATION



(a)



(b)

Figure 3.4: The density profile of the particles around the mid-plane, calculated by using three different particle-mesh assignment schemes: Nearest-Grid-Point (NGP), Cloud-In-Cell (CIC) and Triangular-Shaped-Cloud (TSC). Mesh density is displayed on y-axis, while the x-axis shows vertical displacement from the mid-plane. Stokes numbers used were (a) $St = 10$, and (b) $St = 100$. The grid has 64 cells. Histograms of the particle distribution which are normalized are plotted in the background (gray) for comparison.

Chapter 4

Algorithm for adaptive particles

The numerical simulations of sedimenting particles in turbulent gas presented in Chapter 3 concentrates a large number of particles around the mid-plane, and has few or even no particles away from the mid-plane. At each time step each particle requires an update of position and velocity. This causes an imbalance in the workload in different spatial regions. For numerical simulations, which uses different processors to compute different regions, this causes some of the processors to be idle while the others are still finishing the calculations. In this chapter we propose a solution to this issue, namely an algorithm that makes the distribution of the particles roughly uniform over the entire computational domain. The algorithm also ensures that the essential physical properties sampled by the particles are conserved.

4.1 Splitting particles

As is concluded in Chapter 3, we often end up with cells in which few or no particles reside. The numerical simulation loses the ability to accurately sample properties in areas in which few particles reside. To alleviate this issue an algorithm which splits the particles in cells where their number is low is required.

The algorithm splits every particle into n_{pair} number of pairs. To do this the algorithm first compares the number of particles $n_{p,i}$, in cells $i = 1, 2, 3, \dots, n_{cells}$ where n_{cells} is the total number of cells, with a parameter n_{low} . If $n_{p,i} < n_{low}$ it splits every particle into n_{pairs} pairs defined by

$$n_{pair} = \left\lceil \frac{n_{low}}{2n_{p,i}} \right\rceil. \quad (4.1)$$

This results in a particle number which is greater than but close to n_{low} , hence, increase the probability to probe the regions which are sparse in particles.

The new particles are assigned with new masses, positions and velocities in such a way that we conserve density ρ , momentum density p and energy density e , on the mesh. The mass of the original particle, defined m_p , is divided equally among the $j = 1, 2, 3, \dots, 2n_{pair}$

new particles giving each a mass

$$m_j = \frac{m_p}{2n_{pair}} \quad (4.2)$$

In order to conserve the density field, new particles need to be placed at the position z_p of the original particle

$$z_j = z_p, \quad (4.3)$$

since no shift from the original position could conserve both the center of mass and the assigned density in the neighboring cells.

Following the appointment of new positions and masses we need to give the new particles new velocities. If we want to conserve the energy of the system then we require that the particles receive the same velocity as the original particle. However, in a system for which particle evolution is deterministic this is not preferable. In such a system, two particles, starting with the same initial condition, follow the exact same path and thus give no additional information compared to having one particle probing their path. This defeats the point of splitting. Therefore, we introduce small stochastic kicks to each pair of the particles when one splits, giving new velocities

$$v_{new} = \begin{cases} v_j + dv \\ v_j - dv \end{cases} \quad (4.4)$$

where dv is a realization value from a normal distribution with zero mean and variance σ_{kick}^2 , in which σ_{kick}^2 is chosen to be small. By kicking each pair of particles with equal magnitude in opposite directions we ensure the conservation of momentum p , however, introduce energy to the system.

Nevertheless, in the simulation of the sedimenting particles described in Chapter 3 the system is stochastic. In each iteration we add to the particles uncorrelated random kicks from the gas. Therefore, we can choose $\sigma_{kick} = 0$, hence, conserve the energy of the system while any pair of split particles will not follow the same path.

4.2 Merging particles

To make the particle distribution more uniform, particles in dense areas need to merge. With time the new particles, created in the high altitude regions, also settle onto the mid-plane. This will create a larger number of particles around the mid-plane, hence, slow down the process further. The solution of this problem is to merge the particles in cells where a large number of particles reside.

We require that the merging algorithm conserves all the physical properties in all the affected cells in the particle-mesh assignment. For mesh cell denoted i and particle denoted j , this implies the conservation of density ρ , momentum density p , and kinetic energy density e ,

$$\rho_i = \sum_{j=1}^N W(z_i - z_{p,j}) \frac{m_j}{\Delta z}, \quad (4.5)$$

$$p_i = \sum_{j=1}^N W(z_i - z_{p,j}) \frac{m_j v_j}{\Delta z}, \quad (4.6)$$

$$e_i = \frac{1}{2} \sum_{j=1}^N W(z_i - z_{p,j}) \frac{m_j v_j^2}{\Delta z}, \quad (4.7)$$

where W is the particle-mesh weight function, z_i is the position of the cell center, $z_{p,j}$ is the initial particle position, Δz is the cell-size. Different weight functions affect a different number of cells, and hence a unique merging process is needed for each method. In this work we focus on three different methods, namely NGP, CIC, and TSC.

4.2.1 Nearest-Grid-Point

The simplest particle-mesh method is the NGP whose weight function is expressed in Equation 3.15. This method only affects the properties in the cell in which the particles resides, without loss of generality denoted by index $i = 0$. This requires us to satisfy the three equations ρ_0, p_0, e_0 only in the cell in which the particles are merged. These equations have unknown variables m_j, z_j, v_j , with $j = 1, 2, 3 \dots n$, where n is the number of merged particles. This can be achieved by merging more than two particles into $n = 2$ particles.

We first need to assign the masses and the positions to the merged particles. In NGP, as long as the total mass of the system is unchanged and the merged particles remain in the same cell, the density in the cell ρ_0 is conserved. Therefore, we have one degree of freedom to choose the new masses, and we choose the symmetric way of dividing the total mass M of the original particles in the cell equally into the two merged particles

$$m_1 = m_2 = \frac{M}{2}. \quad (4.8)$$

As for the positions, even though the merged particles can be placed anywhere inside the same cell, we choose the center of mass z_{cm} of the original particles as the position for the merged particles, i.e.,

$$z_1 = z_2 = z_{cm}. \quad (4.9)$$

An expression for the new velocities can be derived starting from the equation of total momentum,

$$P_{tot} = m_1 v_1 + m_2 v_2 \quad (4.10)$$

$$\Rightarrow v_1 = \frac{P_{tot} - m_1 v_1}{m_2} = \frac{P_{tot} - \frac{M}{2} v_{z,1}}{\frac{M}{2}} = \frac{2}{M} P_{tot} - v_1. \quad (4.11)$$

Using this relation between the velocities and the masses from Equation 4.8, an expression for the total kinetic energy E_{tot} becomes

$$\begin{aligned} E_{tot} &= \frac{1}{2} (m_1 v_1^2 + m_2 v_2^2) = \frac{M}{4} (v_1^2 + v_2^2) = \frac{M}{4} \left(v_1^2 + \left(\frac{2}{M} P_{tot} - v_1 \right)^2 \right) \\ &= \frac{M}{4} \left(v_1^2 + \left(\frac{4P_{tot}^2}{M^2} - \frac{4P_{tot}}{M} v_1 + v_1^2 \right) \right) = \frac{M}{2} \left(v_1^2 - \frac{2P_{tot}}{M} v_1 + \frac{2P_{tot}^2}{M^2} \right), \end{aligned} \quad (4.12)$$

thus we have

$$v_1^2 - \frac{2P_{tot}}{M}v_1 + \frac{2P_{tot}^2}{M^2} - \frac{2E_{tot}}{M} = 0. \quad (4.13)$$

Therefore, by using the quadratic formula, we obtain an expression for the velocities of the two merged particles,

$$v_{\pm} = \frac{P_{tot}}{M} \pm \sqrt{\left(\frac{P_{tot}}{M}\right)^2 - \frac{2P_{tot}^2}{M^2} + \frac{2E_{tot}}{M}} = \frac{P_{tot} \pm \sqrt{2ME_{tot} - P_{tot}^2}}{M}. \quad (4.14)$$

However, this only holds if the discriminant is non-negative, that is if $2E_{tot}M - P_{tot}^2 \geq 0$.

It can be proved that the discriminant is always non-negative by using the Cauchy inequality, which states that the magnitude of the scalar product of two vectors is always less than or equal to the product of their lengths,

$$|\mathbf{a} \cdot \mathbf{b}| \leq |\mathbf{a}||\mathbf{b}| \quad (4.15)$$

From the definitions of momentum and energy we obtain the expressions

$$\begin{aligned} P_{tot}^2 &= \left(\sum_j m_j v_j\right)^2 \\ 2ME_{tot} &= \left(\sum_j m_j\right)\left(\sum_j m_j v_j^2\right) \end{aligned} \quad (4.16)$$

Using the vectors $\mathbf{a} = (\sqrt{m_j}, \sqrt{m_j})$, and $\mathbf{b} = (\sqrt{m_j}|v_j|, \sqrt{m_j}|v_j|)$ it can be shown that

$$\begin{aligned} (\mathbf{a} \cdot \mathbf{b})^2 &\leq |\mathbf{a}|^2 |\mathbf{b}|^2 \\ &\Leftrightarrow \\ \left(\sum_j m_j v_j\right)^2 &\leq \left(\sum_j m_j |v_j|\right)^2 \leq \left(\sum_j m_j\right)\left(\sum_j m_j |v_j|^2\right) \\ &\Leftrightarrow \\ P_{tot}^2 &\leq 2ME_{tot}, \quad Q.E.D. \end{aligned} \quad (4.17)$$

4.2.2 Cloud-In-Cell

The CIC weight function affects three different cells when assigning a property. Therefore, when merging particles in a selected cell we must consider conserving the physical quantities in both the selected cell, and the two neighboring cells on each side. Our method for satisfying this is to merge particles in a given cell into 6 particles. One pair is placed at the center of the cell, and the other pairs are placed at each edge of the selected cell. The mass is then divided among the particles in such a way that the densities in all three cells are preserved.

The density of each cell is given by Equation 4.5 with the weight function W_{CIC} given by Equation 3.17. There are three pairs for which masses are allocated, which we denote

4.2. MERGING PARTICLES

m_C mass contributed to the center cell in which the merging particles resides and m_L (m_R) for the mass contributed to the cell on the left hand side (right hand side) of the center cell. The side cells have contributions from half the mass of the side pairs, while the center cell has the entire mass of the center pair, plus half the mass of each side pair. This gives densities,

$$\rho_L = \frac{m_L}{2\Delta z}, \quad (4.18)$$

$$\rho_C = \frac{m_L + 2m_C + m_R}{2\Delta z}, \quad (4.19)$$

$$\rho_R = \frac{m_R}{2\Delta z}, \quad (4.20)$$

where we use the same index convention for the densities ρ_L, ρ_C, ρ_R as we do for the mass. Inverting Equations 4.18–4.20 gives

$$m_L = 2\rho_L\Delta z, \quad (4.21)$$

$$m_R = 2\rho_R\Delta z, \quad (4.22)$$

$$m_C = (\rho_C - \rho_L - \rho_R) \Delta z. \quad (4.23)$$

The assignment of the masses and velocities for each pair of particles then follows the same procedure as in the case of NGP (Section 4.2.1), giving

$$m_{L,\pm} = \frac{m_L}{2}, \quad (4.24)$$

$$m_{C,\pm} = \frac{m_C}{2}, \quad (4.25)$$

$$m_{R,\pm} = \frac{m_R}{2}. \quad (4.26)$$

and

$$v_{L,\pm} = \frac{P_{tot,L} \pm \sqrt{m_L E_{tot,L} - P_{tot,L}^2}}{m_L}, \quad (4.27)$$

$$v_{C,\pm} = \frac{P_{tot,C} \pm \sqrt{m_C E_{tot,C} - P_{tot,C}^2}}{m_C}, \quad (4.28)$$

$$v_{R,\pm} = \frac{P_{tot,R} \pm \sqrt{m_R E_{tot,R} - P_{tot,R}^2}}{m_R}. \quad (4.29)$$

The values of P_{tot} and E_{tot} for each pair of particles are obtained following the same inversion procedure as for the densities in Equations 4.18–4.23.

The proof that the velocities in Equations 4.27–4.29 are always real is no longer obvious. The reason is that the new properties are assigned by considering several cells. This gives cross terms in Equations 4.16 and the proof in Section 4.2.1 is no longer valid. However, in the case of CIC the same proof applies to the side pair of the particles because their total properties only depend on those from the side cells (see Equations 4.18 and 4.20), with an additional weight, 1/2 in this case. As for the center pair of particles, we found in practice that Equation 4.28 always give a real value in our simulations.

4.2.3 Triangular-Shaped-Cloud

TSC assignment affects three different cells. Therefore, when merging particles we tried the same method as for the CIC, that is merging the particles in a cell into three particle pairs and then placing them at the center and edges of the selected cell. However, unlike for the CIC the outer cells are now affected by the center particle pair and thus all three cells are coupled to each other. The solution for the new masses is found from the linear equations describing the density given by Equations 4.5. In matrix form we write the equations as

$$\begin{bmatrix} 1/2 & 1/8 & 0 \\ 1/2 & 3/4 & 1/2 \\ 0 & 1/8 & 1/2 \end{bmatrix} \begin{bmatrix} m_L \\ m_C \\ m_R \end{bmatrix} = \begin{bmatrix} \rho_L \\ \rho_C \\ \rho_R \end{bmatrix} \Delta z \quad (4.30)$$

By solving this linear system of equations we obtain the mass of each particle pair,

$$\begin{bmatrix} m_L \\ m_C \\ m_R \end{bmatrix} = \begin{bmatrix} 1/2 & 1/8 & 0 \\ 1/2 & 3/4 & 1/2 \\ 0 & 1/8 & 1/2 \end{bmatrix}^{-1} \begin{bmatrix} \rho_L \\ \rho_C \\ \rho_R \end{bmatrix} \Delta z = \begin{bmatrix} 5/2 & -1/2 & 1/2 \\ -2 & 2 & -2 \\ 1/2 & -1/2 & 5/2 \end{bmatrix} \begin{bmatrix} \rho_L \\ \rho_C \\ \rho_R \end{bmatrix} \Delta z \quad (4.31)$$

This procedure can also be applied to both the momentum and the energy of the system.

However, as seen in Equation 4.31, the mass assigned to a side pair is the linear combination of the densities all three cells. For certain particle distribution this result in a negative mass assignment. In such cases Equations 4.27–4.29 do not give real velocities. This problem remained unsolved, therefore, we could not apply this merging algorithm for the TSC.

Chapter 5

Numerical simulations using adaptive particles

In this chapter we focus on validating the numerical simulations of the sedimentation process using adaptive particles with the NGP and the CIC particle-mesh methods. We compare the resulting scale-heights of steady-state sedimented disks with different adaptation parameters. The most important parameters are the range $[n_{\text{low}}, n_{\text{high}}]$, which represent the number of particles allowed in a mesh-cell. For the most interesting cases we present plots showing the particle density and number distributions in comparisons to non-adapting simulations. In the last section of this chapter we perform a resolution study with the adaptive particles.

The adaptive particles were tested on particle sedimentation in a gaseous protoplanetary disk, as described in Chapter 3. The particles dimensionless stopping time $\tau_s = 0.1$, and a Stokes number of $St = 10$. For this setup we expected the equilibrium scale height to be $H_p = 14.12$, as measured in Section 4.3. The particles were initially distributed uniformly over $\pm 4H_p$ around the mid-plane ($z = 0$) of the disk. The turbulence in the gas was introduced to the particles as small stochastic velocity kicks given by a normal distribution with zero mean and standard deviation $\sigma_{\text{kick}} = 0.1$. The total mass of the system was chosen to be $M_{\text{tot}} = 1$ in arbitrary units.

The algorithm adapts the number of particles to remain within a range $[n_{\text{low}}, n_{\text{high}}]$ per cell. If a grid-cell contains $n_{p,i} < n_{\text{low}}$ particles, it splits the particles in pairs until $n_{p,i} \geq n_{\text{low}}$ is satisfied. The upper limit n_{high} sets the number of particles allowed in a cell before merging them. The purpose of the algorithm is to create a balanced particle distribution, but, still keep the particle number low. Therefore, we initialized the system with n_{high} particles per cell so that initially merging is preferred to splitting. When the algorithm splits particles they are given a slight kick dv in order to ensure that they take separate paths after splitting. The sedimentation model in this thesis already includes random kicks from the turbulence. For this reason we set $dv = 0$ in all our simulations, hence, conserving the energy of the system throughout the simulations.

In all our simulations we updated the position and velocity of all particles using the Runge-Kutta method, described in Chapter 2. After the equilibrium state was reached

the scale height of the particles was measured by computing the standard deviation of the particle distribution for 1000 additional iterations. The final scale height was obtained by taking the mean value of the 1000 measurements. This reduced the noise in the measurement and gave a more accurate result for the scale height.

The particles change their masses due to splitting and merging, and therefore it is no longer trivial to measure the scale-height. In Chapter 3 we measured the scale height of the system by calculating the standard deviation in the position of the particles. However, when we adapt the particles the standard deviation of the distribution needs to be weighted by the mass of each particle. For N discrete particles, with individual masses m_j , this gives a scale height equal to

$$\sigma_{z_p} = \sqrt{\frac{\sum_j^N m_j (z_{p,j} - \bar{z}_p)^2}{\sum_j^N m_j}}, \quad (5.1)$$

where \bar{z}_p is the center of mass for all particles given by

$$\bar{z}_p = \frac{\sum_j^N m_j z_j}{\sum_j^N m_j}. \quad (5.2)$$

In order to validate the PM method used for adapting the particles we also computed the standard deviation of the density profile on the grid. This was done with an equation similar to Equations 5.1 and 5.2 given by

$$\sigma_\rho = \sqrt{\frac{\int (z - \bar{z}_\rho)^2 \rho(z) dz}{\int \rho(z) dz}}, \quad (5.3)$$

and

$$\bar{z}_\rho = \frac{\int z \rho(z) dz}{\int \rho(z) dz}, \quad (5.4)$$

where the integrals cover the entire computational domain.

5.1 Nearest-Grid-Point

We start by focusing on in Figure 5.1b with a range set by $n_{\text{low}} = 4$ and $n_{\text{high}} = 8$. Included in this plot is density and distribution profiles of particles both with and without adaptive particles. The feature in the density profile (upper panel) which distinguishes the simulation with adaptive particles (black line) most prominently from the non-adapting particles (blue dotted dash-line) is that the profile stretches to the boundary of the computational domain. Comparing the black line to the analytic solution (red dash-line) we find good agreement in the entire computational domain. The adaptive particles correctly sample the low-density regions, where the non-adaptive particles cannot reach. However, if we instead focus on the area close to the mid-plane ($z = 0$), where both simulations agree with the analytic solution, we find that the black line is noisier than the blue dotted dash-line.

5.1. NEAREST-GRID-POINT

Table 5.1: Measured properties of the particle distribution in simulations with adaptive particles with the Nearest-Grid-Point scheme. σ_{z_p} and σ_ρ (calculated using Equations 5.1 and 5.3 respectively) are the standard deviation in particle position around the mid-plane, which correspond to the scale height of the particles. The last two columns shows $\langle n_i \rangle$, which is the average number of particles in the cells with corresponding standard deviation σ_{n_i} . *The last row lists the measurement obtained when not using the adaptation algorithm for comparison.

n_{low}	n_{high}	$\sigma_\rho [\times 10^{-3}]$	$\sigma_{z_p} [\times 10^{-3}]$	$\langle n_i \rangle$	$\sigma_{n_i} / \langle n_i \rangle$
2	6	14.2091	14.2001	2.93	0.43
2	8	15.6017	15.4130	3.68	0.28
3	4	12.7855	12.7375	3.56	0.17
3	6	14.6384	14.6131	4.16	0.20
4	6	13.2716	13.2112	4.87	0.17
4	8	14.7710	14.7641	5.50	0.24
4	12	13.5717	13.5219	6.93	0.30
4	16	14.3693	14.3263	8.47	0.24
4	32	13.4920	13.4632	14.78	0.43
8	12	13.4195	13.3736	9.03	0.22
-	-	14.2187*	14.1887*	6.25*	-

As shown in the lower panel in Figure 5.1b, the region close to the mid-plane have a larger abundance of non-adaptive particles. This explains why the adaptive particles have an increased noise close to the mid-plane. The noise is determined mainly by Poisson noise, proportional to $\sqrt{n_{p,i}}$, where $n_{p,i}$ is number of particles in cell i . On the other hand, no non-adaptive particles reside at high altitudes and thus no particle properties are available there, as shown by the steep decent in the density profile of the simulation without adapting particles.

We now turn our attention to how n_{low} changes the behavior of the system. In Figure 5.1a, we present the density and distribution profiles when choosing $n_{\text{low}} = 2$, while keeping $n_{\text{high}} = 8$. This decreases the number of particles which are splitted. The density profile shows that with too few splits the probability for cells void of particles becomes significantly higher. Nevertheless, the scale height of the resulting distribution in this case remains reasonably close to the theoretical H_p , as shown in Table 5.1. Further tests showed that using $n_{\text{low}} = 3$ gave a density profile similar to $n_{\text{low}} = 4$. By comparing the measurements listed in Table 5.1 we find that the scale heights of the particles agree with the theoretical scale height when using different values of n_{low} . However, the average number of particles in the cells increases for larger n_{low} . This is expected since a larger n_{low} forces the number of particles to be larger in each cell.

Finally, we discuss the effects of changing n_{high} . Since the algorithm will create two new particles in the event of a merger it is not meaningful to set $n_{\text{high}} \leq 2$. In Figure 5.1c shows the density and distribution profiles of adapting particles using $n_{\text{low}} = 4$ and

$n_{\text{high}} = 12$. The density profile deviates from the theoretical one in the outermost regions even for the numerical simulation using adaptive particles. However, in the distribution profile we find that there are still particles present in this region. It is due to large particles absorbing large quantities of mass. Since every merger results in only two particles with the NGP method, a large n_{high} will change the number of particles in a cell, as well as their masses, drastically with every merger. This creates a few very massive particles which are close to the mid-plane, since this is where mergers are most frequent. Nevertheless, the measurements listed in Table 5.1 agree reasonably well with the theoretical H_p regardless of n_{high} . We find that by increasing n_{high} the average number of particles in the cells is also increased. However, the overall fluctuation in particle number increases as is seen in the measurements of $\sigma_{n_i}/\langle n_i \rangle$ for different n_{high} .

5.2 Cloud-In-Cell

Table 5.2: Measured properties of the particle distribution in simulations with adaptive particles with the Cloud-In-Cell scheme. σ_{z_p} and σ_ρ (calculated using Equations 5.1 and 5.3 respectively) are the standard deviation in particle position around the mid-plane, which correspond to the scale height of the particles. The last two columns shows $\langle n_i \rangle$, which is the average number of particles in the cells with corresponding standard deviation σ_{n_i} . *The last row lists the measurement obtained when not using the adaptation algorithm for comparison.

n_{low}	n_{high}	$\sigma_\rho^2 [\times 10^{-3}]$	$\sigma_{z_p}^2 [\times 10^{-3}]$	$\langle n_i \rangle$	$\sigma_{n_i}/\langle n_i \rangle$
3	8	13.0894	13.0109	4.91	0.33
3	12	13.0574	12.9875	6.68	0.36
4	6	15.2010	15.1468	5.12	0.22
4	8	14.9124	14.8504	5.76	0.30
4	12	13.2053	13.1318	7.29	0.31
4	16	12.6863	12.6026	8.50	0.36
4	32	13.5891	13.7035	14.78	0.43
8	12	14.8093	14.7496	9.82	0.27
-	-	14.2187*	14.1887*	6.25*	-

When we apply the CIC adaptive particles to the sedimentation simulation we find density profiles which are significantly closer the theoretical density profile. The upper panel in Figure 5.2b shows the density profile from a simulation using the parameters $n_{\text{low}} = 4$ and $n_{\text{high}} = 8$. Also shown in the figure are the analytic solution and the profile from the simulation without adaptive particles. Similar to the NGP simulations we find that using the adaptive particles gives good agreement in the regions far from the mid-plane. On the other hand, the CIC gives a much less noisy profile near the mid-plane than the NGP.

Figure 5.2 shows that changing n_{low} and n_{high} when using the CIC does not change the results drastically. Note that since the CIC merging algorithm results in six particles in every merger we constrain $n_{\text{high}} > 6$. Otherwise, merging would increase the number of particles in a cell. Similar to the density profiles for the NGP method we find that increasing both n_{low} and n_{high} simultaneously gives a better agreement to the analytic solution. This is seen by comparing the profile using the adaptive particles given by the black line, to the red dash-line giving the analytic solution in Figure 5.2c.

Comparing the density profiles with the NGP scheme in Figure 5.1 to those with CIC scheme in Figure 5.2 we find that the CIC is better in reproducing the Gaussian density profiles when using adaptive particles. This was expected since the number of particles is dependent on n_{low} and n_{high} rather than differences in splitting or merging algorithm. However, if comparing the average number of particles in cells $\langle n_i \rangle$ for both NGP and CIC methods, shown in Tables 5.1 and 5.2, we find a similar number of particles per cell. Therefore, the reduced noise is not from an increase in particles but from the increased spatial accuracy of the CIC.

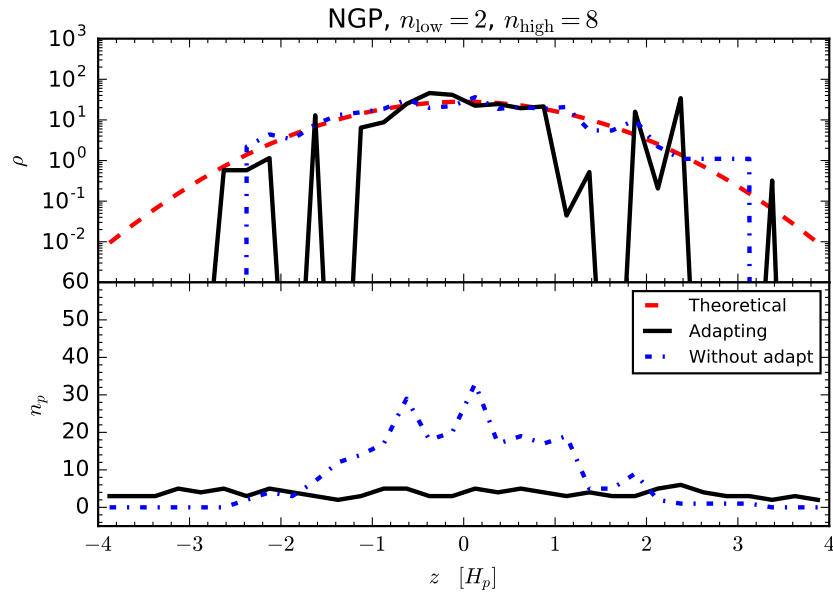
5.3 Resolution study

Finally, we study the effect of resolution on the adaptive-particle algorithm. We set up simulations using a grid-resolution of 256 and comparing them to those using a resolution of 32. We expected that the results would become better with the increase in resolution as the accuracy of the PM assignment depends on the cell size. We focused on the case of $n_{\text{low}} = 4$ and $n_{\text{high}} = 8$. The system was initialized with $256n_{\text{high}}$ particles and run for 5000 iterations for both NGP and CIC.

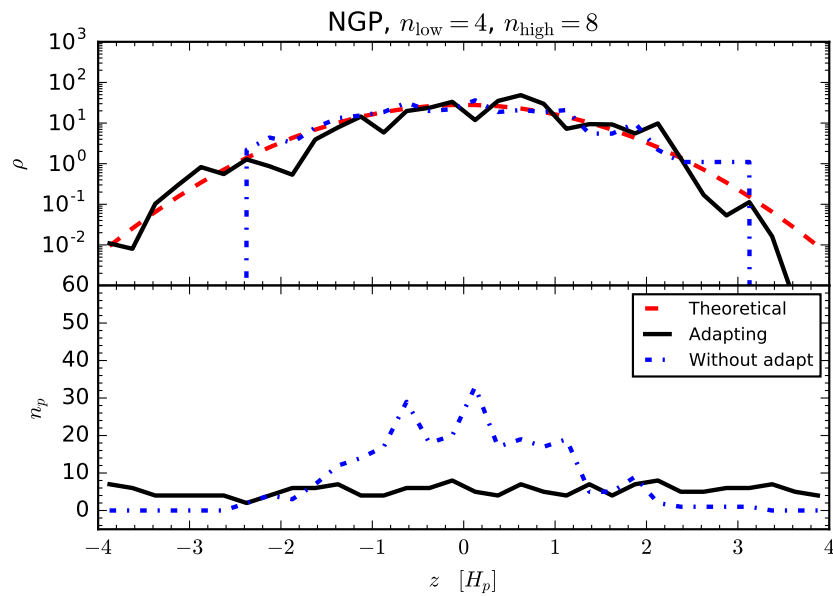
Figure 5.3a shows the density profiles for simulations using the NGP adaptive particles. The profile with mesh resolution of 256 (black line) shows large noise spikes compared to the one with a mesh resolution of 32 (blue line). This effect comes from the fact that the cells are much smaller for the higher resolution. With a small cell size the particles easily move in and out of the cells in each iteration. This results in large fluctuations between neighboring cells. This effect can be compared to using a histogram with a large number of bins. Nevertheless, increase in resolution does not effect the scale height of the particles.

For the CIC density profile shown in Figure 5.3b we find that increasing the resolution does not increase the noise level of the profile. The reason is again that the CIC considers particles in multiple cells when assigning mass to the mesh grid, and thus gives a smoother profile with higher positional accuracy. This demonstrates the significantly better performance of the CIC adaptive particles than that of the NGP adaptive particles..

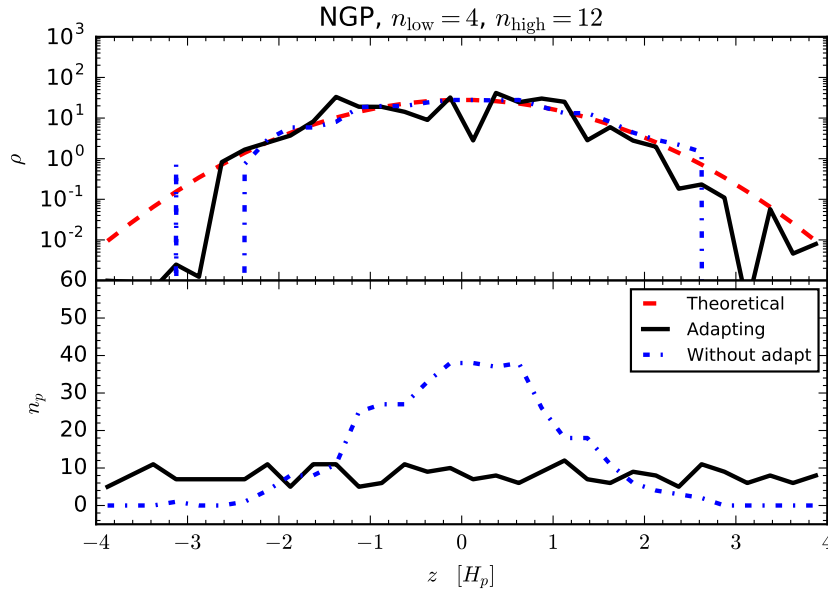
5.3. RESOLUTION STUDY



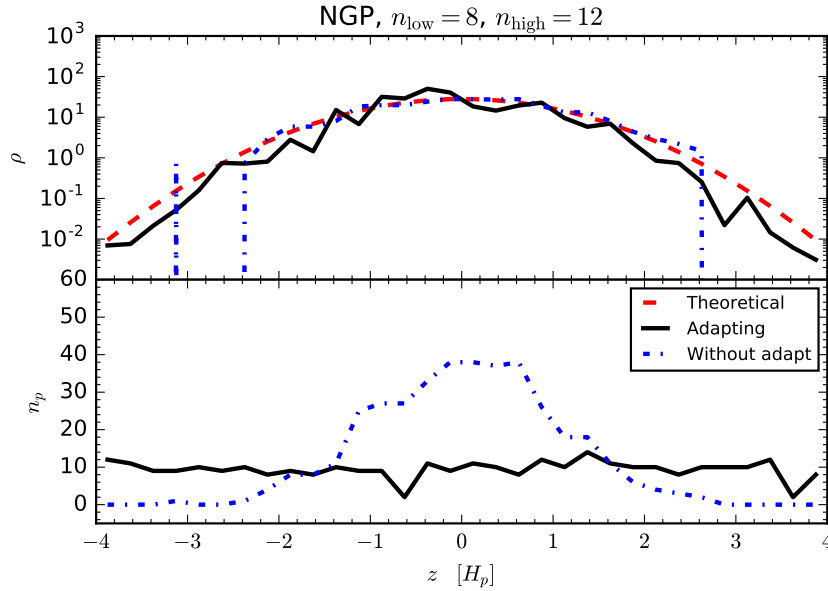
(a)



(b)



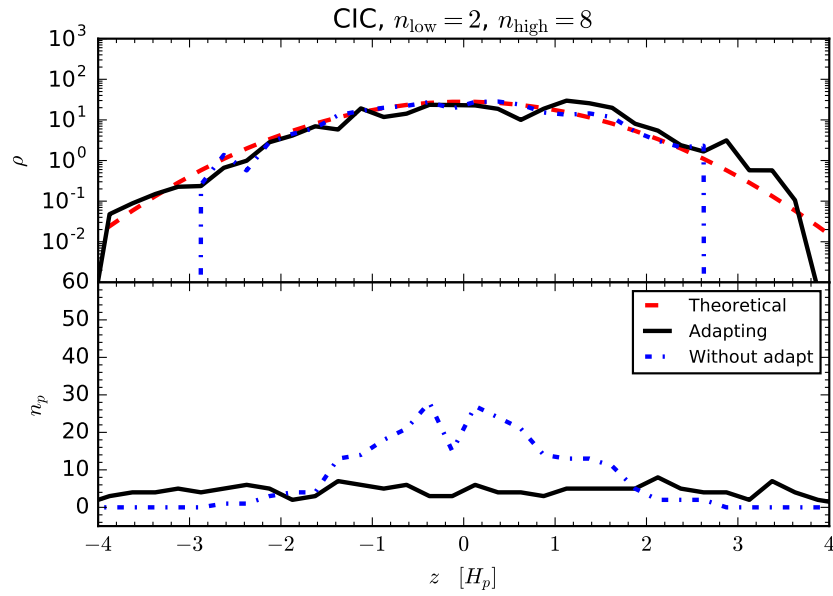
(c)



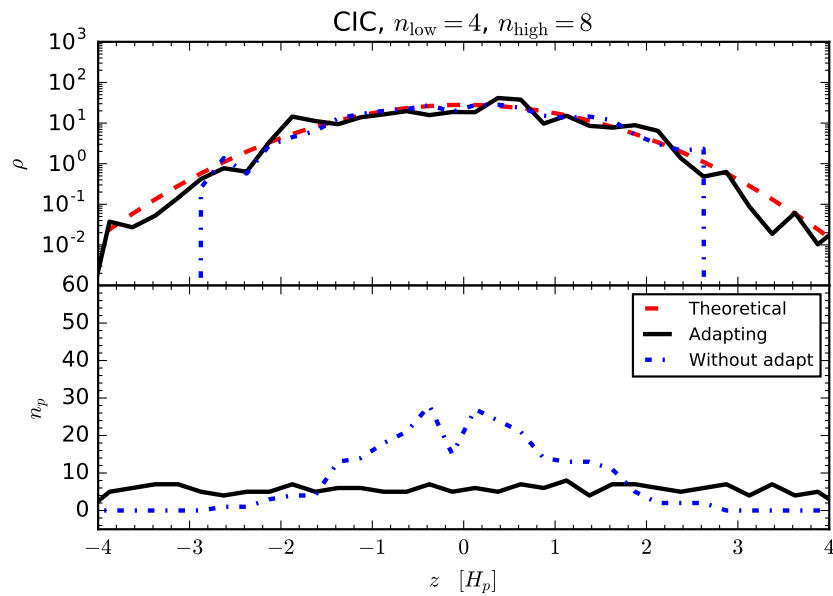
(d)

Figure 5.1: The particle density (top panel) and distribution (bottom panels) profiles from the simulations of adaptive particles with the Nearest-Grid-Point scheme. The plots are snapshots at the last step in the simulation, that is after 5000 iterations. The grid has a resolution of 32 points. Each plot includes the theoretical solution, along with the profiles both with and without adaptation. The four different plots show different ranges $[n_{\text{low}}, n_{\text{high}}]$ used for the adaptation.

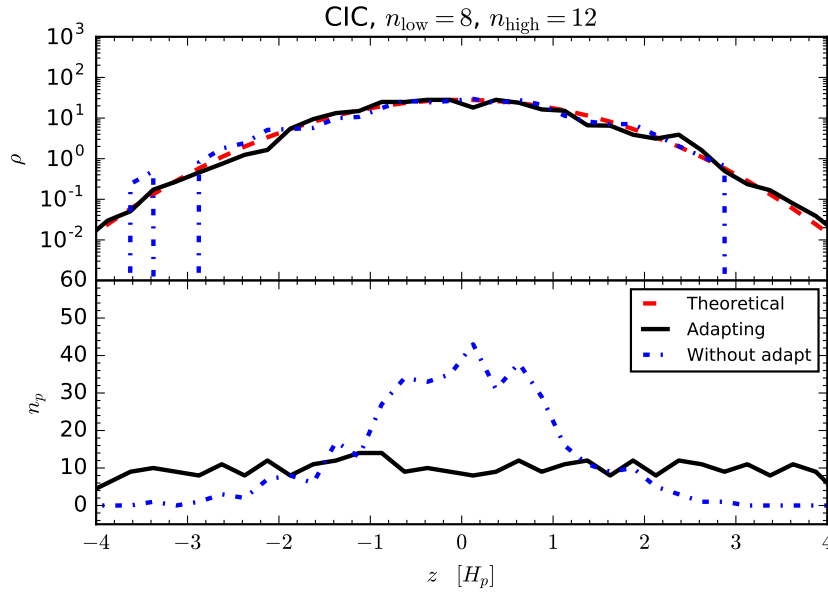
5.3. RESOLUTION STUDY



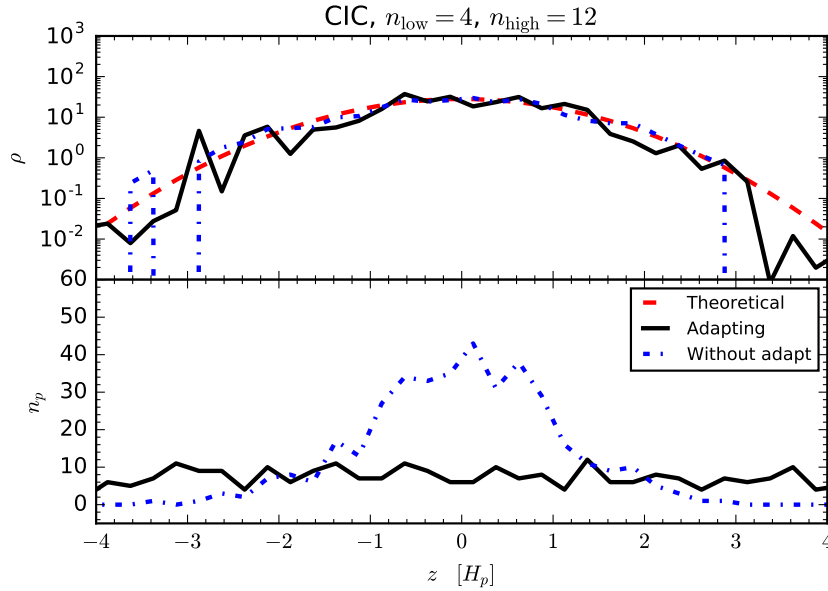
(a)



(b)

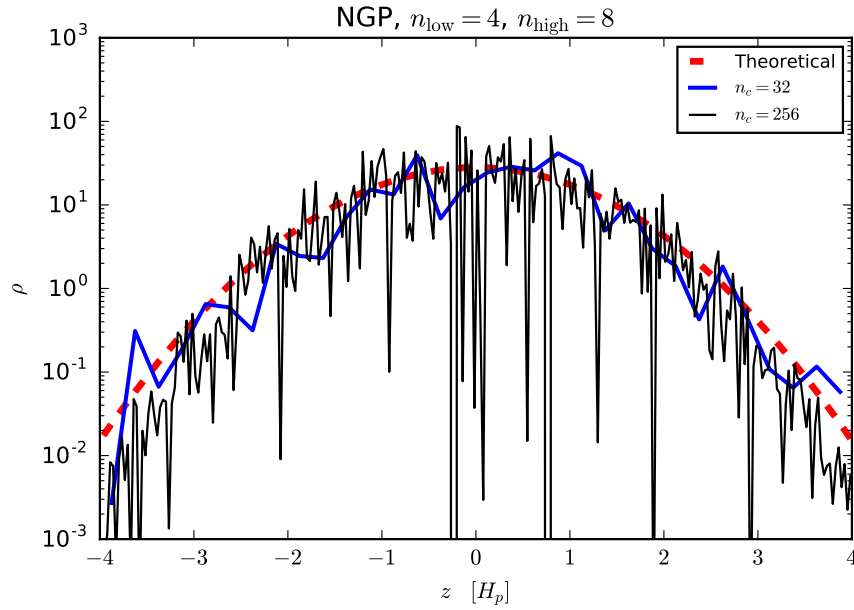


(c)

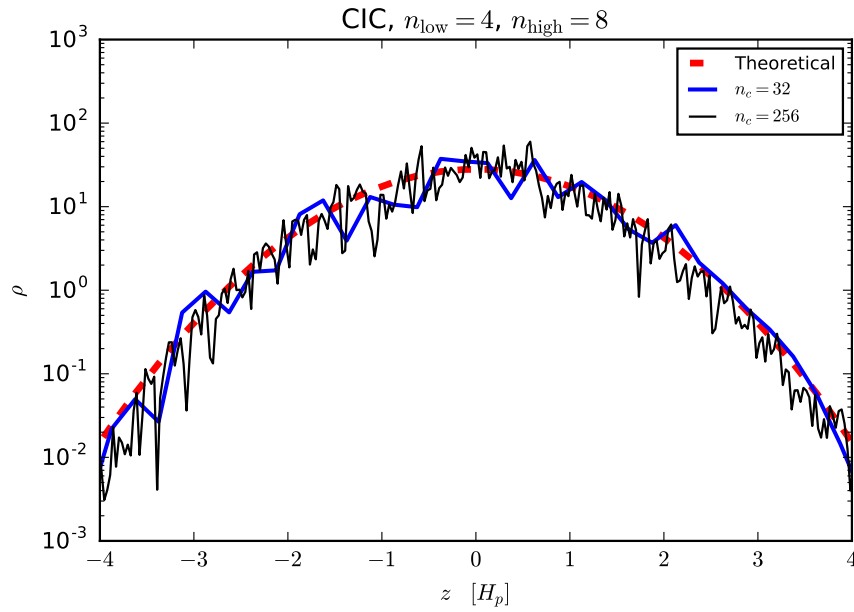


(d)

Figure 5.2: The particle density (top panel) and distribution (bottom panels) profiles from the simulations of adaptive particles with the Cloud-In-Cell scheme. The plots are snapshots at the last step in the simulation, that is after 5000 iterations. The grid has a resolution of 32 points. Each plot includes the theoretical solution, along with the profiles both with and without adaptation. The four different plots show different ranges $[n_{\text{low}}, n_{\text{high}}]$ used for the adaptation.



(a)



(b)

Figure 5.3: Equilibrium density profiles from simulations with adaptive particles with (a) the Nearest-Grid-Point scheme and (b) the Cloud-In-Cell scheme, using two different resolutions. Also included are the theoretical profiles in red dashed lines. The black line shows simulations using a grid with $n_c = 256$ cells while the blue line shows simulation using $n_c = 32$ cells for the Particle-Mesh method.

Chapter 6

Summary

6.1 Sedimentation of solid particles in protoplanetary disks

In the first part of this thesis we investigated the sedimentation of Lagrangian solid particles in a protoplanetary disk, both theoretically and numerically. The dynamics of the particles was described by two force terms: stellar gravity in the direction perpendicular to the mid-plane of the protoplanetary disk and drag force from interactions with the gas in the disk. The drag force was described using a dimensionless stopping time τ_s , which dampened the motion of the particles, and a term \mathbf{u} introducing stochastic kicks to the particles. The stopping time was made dimensionless by multiplying it with the Keplerian angular frequency Ω which was set to 1 in all our simulations. The kicks, drawn from a normal distribution with zero mean and standard deviation $\sigma = 0.1$, describe how energy introduced to the gas via the magnetorotational instability, is transferred to the particles. A similar approach was used by Carballido et al. (2006). The advantage of using stochastic kicks is that they simulate the turbulence which is otherwise modeled with magneto-hydrodynamics.

For the numerical simulations we used a third-order Runge-Kutta method to integrate the differential equations describing the system. Implementing it using parameter values derived by Williamson (1980), we found that the method is stable and can accurately reproduce the analytic solution if the step-size is chosen with the criterion $h \leq \tau_s$ for $\tau_s < 1$ and $h < 1$ for $\tau_s > 1$. However, when running simulations using $\tau_s > 1$ for extended periods of time the step size h needed to be set smaller. Otherwise we lost the amplitude in the oscillation too quickly with time in each iteration, which is known as diffusive error.

An advantage of our model is that we are able to derive an analytic solution for the density profile of the particles at the equilibrium state. However, the diffusion coefficient present in the transport equation is an unknown quantity. Nevertheless, we were able to measure this diffusion coefficient in our simulations and thus find quantitatively the analytic density profile. The analytic density profiles allowed us to evaluate our numerical simulations, especially those with the adaptive particles.

In numerical simulations of the sedimentation process a limiting factor is the non-uniform distribution of particles. The density profile of particles follow a normal distribution around the mid-plane, and thus most particles are located close to the mid-plane. Simulation codes using parallel computing usually divide the computational domain into smaller subsets and distribute all calculation in the subsets to different processors. Therefore, different processors may receive different workloads when particles are not uniformly distributed, and thus some processors become idle when waiting for others to finish the calculations. This brought us to the algorithm of adaptive particles which aims to balance the workload over the computational domain.

6.2 Algorithm for adapting particles

The main goal of this thesis was to develop an algorithm for adapting particles. The algorithm operates in two situations: merging particles in dense regions and splitting particles in sparse regions. In the algorithm, we carefully maintain the particle properties on a fixed grid assigned by the particle-mesh method. The algorithm was successfully implemented for Nearest-Grid-Point and Cloud-In-Cell assignment of the Particle-Mesh method. However, we have not succeeded in implementing the merging method for the more accurate Triangular-Shaped-Cloud method. The reason was that conserving energy and momentum with our particle placement resulted in assignment of negative mass or imaginary velocity to the particles.

With the adaptive particles we were able to distribute the particles roughly uniformly in the entire computational domain. The algorithm uses a range set by n_{low} and n_{high} . If the number of particles in a cell is outside this range it will adapt the particles to be within the range. We found that using a small range made the measurements more stable, while increasing both parameters would increase the accuracy. For the CIC method the algorithm can keep the number of particles in all cells nearly equal with a variation of $\sim 30\%$, and smaller if the range was decreased. This ensures that the entire computational domain has a balanced workload.

With particles present in the entire domain we were able to sample the particle properties even in the regions at high altitudes above and below the mid-plane. For both the NGP and the CIC methods the simulated density profiles followed the theoretical one the entire computational domain. In our simulations we used a space which stretched to four scale heights away from the mid-plane. The numerical simulations performed without adaptive particles could only probe a region ± 2.5 scale heights from the mid-plane.

A limitation of the adaptive particles is that the splitting method introduces energy to the system. A deterministic system does not separate particles with the same initial conditions. We obtain no additional information about such a system by only splitting particles because they take the same path after the split. Therefore, we need to add small kicks to split the particles so that they can diverge from each other.

In our model the particles feel the turbulence via stochastic kicks in each time step, and thus we did not require any additional kicks when splitting particles in our system.

Therefore, we have not validated the algorithm for splitting particles with kicks. We speculate that the algorithm remain valid as long as the energy introduced by these kicks is small compared to the kinetic energy transferred from the turbulent gas to the particles, which requires further study by properly modeling the turbulent gas on a fixed grid.

6.3 Further research

Due to limited amount of time for this project we did not manage to devise a reliable algorithm to merge particles using the Triangular-Shaped-Cloud (TSC) method. Youdin and Johansen (2007) showed that the accuracy of the assignment is increased significantly with the TSC. Hence, further attempts to implement the algorithm for this method is of interest.

As discussed above, the numerical simulations we used to validate the adaptive particles is not suitable for studying the effects of kicks when splitting the particles. For this purpose the algorithm needs to be implemented in a particle-gas simulation that includes gas turbulence on a fixed grid. Magnetohydrodynamics can be used to introduce the magnetorotational instability which is believed to be the origin of the gas turbulence (see, e.g., Balbus and Hawley, 1998; Armitage, 2011). Turbulent gas velocity on a fixed grid would impart particles at the same location with the same drag force, and thus require the kicks in the splitting method would become necessary.

In this project we investigate the sedimentation process for protoplanetary disks in only one dimension, that is the direction perpendicular to the mid-plane of the disk. However, if the algorithm is implemented into a 3D simulation of a protoplanetary disk, one could investigate how it affects other mechanisms associated with these disks, e.g. the streaming instability (e.g., Youdin and Johansen, 2007; Yang and Johansen, 2014). The current implementation of the adaptive particles requires six particles to be created in every merger for CIC accuracy. Because the particles are positioned at the cell center and boundaries, and hence the number of particles in a CIC merger scales as $2*3^D$ where D is the number of dimensions. For a 3D code this imply creating 54 particles in every merger. Given that the density ratio of particles between the mid-plane and the four scale heights amounts to 3000, a workload of 54 particles per cell remains advantageous in boosting the computational efficiency and sampling the low-density regions of particles.

Acknowledgements

First, and foremost, I want to acknowledge my supervisor Chao-Chin Yang, without whom this bachelor thesis would have become a major embarrassment for my academic career. The list of things Chao-Chin did for me during this project is long. Some examples are: improved my general knowledge about physics, staying up late correcting my drafts, taught me invaluable coding advise, tried helping me write a perfect thesis, watered the flowers in my office and last but not least became a good friend.

I also want to thank my friend Vassily Kornienko for proof reading, questioning my statements, enlightening discussions, 24 liters of apple vine and keeping me sane during the most challenging times. Other people I want to acknowledge for helping me are Linn Eriksson, Aron ten Brinke, Neige Frankel.

Bibliography

- Armitage, P. J.
2010. *Astrophysics of Planet Formation*.
- Armitage, P. J.
2011. Dynamics of Protoplanetary Disks. *ARA&A*, 49:195–236.
- Balbus, S. A. and J. F. Hawley
1998. Instability, turbulence, and enhanced transport in accretion disks. *Reviews of Modern Physics*, 70:1–53.
- Carballido, A., S. Fromang, and J. Papaloizou
2006. Mid-plane sedimentation of large solid bodies in turbulent protoplanetary discs. *MNRAS*, 373:1633–1640.
- Dubrulle, B., G. Morfill, and M. Sterzik
1995. The dust subdisk in the protoplanetary nebula. *Icarus*, 114:237–246.
- Hawley, J. F., C. F. Gammie, and S. A. Balbus
1995. Local Three-dimensional Magnetohydrodynamic Simulations of Accretion Disks. *ApJ*, 440:742.
- Hockney, R. W. and J. W. Eastwood
1981. *Computer Simulation Using Particles*.
- Williamson, J. H.
1980. Low-Storage Runge-Kutta Schemes. *Journal of Computational Physics*, 35:48–56.
- Yang, C.-C. and A. Johansen
2014. On the Feeding Zone of Planetesimal Formation by the Streaming Instability. *ApJ*, 792:86.
- Yang, C.-C., M.-M. Mac Low, and K. Menou
2009. Planetesimal and Protoplanet Dynamics in a Turbulent Protoplanetary Disk: Ideal Unstratified Disks. *ApJ*, 707:1233–1246.

BIBLIOGRAPHY

Youdin, A. and A. Johansen

2007. Protoplanetary Disk Turbulence Driven by the Streaming Instability: Linear Evolution and Numerical Methods. *ApJ*, 662:613–626.

Youdin, A. N. and Y. Lithwick

2007. Particle stirring in turbulent gas disks: Including orbital oscillations. *Icarus*, 192:588–604.