

# Chapter 4

## Multi-scale modelling of dry granular flows

### 4.1 Introduction

The dynamics of a homogeneous granular flow involve at least three distinct scales: the *microscopic scale*, which is characterised by the contact between grains, the *meso-scale* that represents micro-structural effects such as grain rearrangement, and the *macroscopic scale*, where geometric correlations can be observed (see Figure 4.1). Conventionally, granular flows are modelled as a continuum because they exhibit many collective phenomena. However, on a grain scale, the granular materials exhibit complex solid-like and/or fluid-like behaviour. Recent studies, however, suggest that a continuum law may be unable to capture the effect of inhomogeneities at the grain scale level, such as orientation of force chains, which are micro-structural effects. Discrete element methods (DEM) are capable of simulating these micro-structural effects, however they are computationally expensive. In the present study, a multi-scale approach is adopted, using both DEM and continuum techniques, to better understand the rheology of granular flows and the limitations of continuum models.

### 4.2 Granular column collapse

The collapse of a granular column on a horizontal surface is a simple case of granular flow, however a proper model that describes the flow dynamics is still lacking. Granular flow is modelled as a frictional dissipation process in continuum mechanics but studies showing the lack of influence of inter-particle friction on the energy dissipation and spreading dynamics is surprising. In the present study, the generalised interpolation material point method (GIMPM),

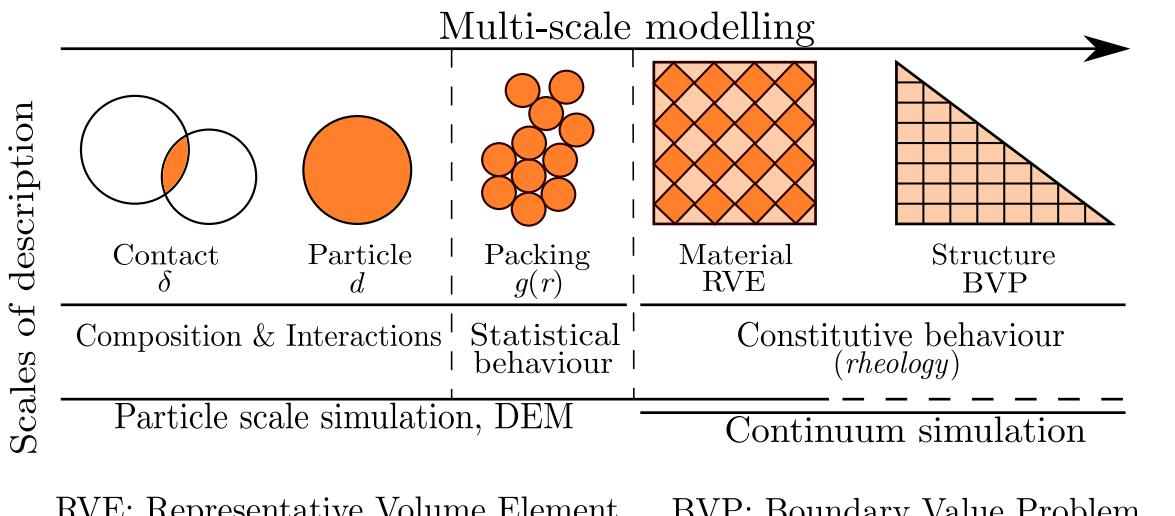


Figure 4.1 Multi-scale modelling of granular materials

- <sup>1</sup> a hybrid Eulerian – Lagrangian approach, is implemented with Mohr-Coloumb failure criterion
- <sup>2</sup> to describe the continuum behaviour of quasi-two dimensional collapse of granular columns.
- <sup>3</sup> The granular column collapse is also simulated using DEM to understand the micro-mechanics
- <sup>4</sup> of the flow.

<sup>5</sup> The granular column collapse experiment involves filling a cylinder of height  $H_0$  and width  $L_0$  with a granular material of mass ‘m’. The granular column is then released *en masse* <sup>6</sup> by quickly removing the cylinder, thus allowing the granular material to collapse onto the <sup>7</sup> horizontal surface, forming a deposit having a final height  $H_f$  and radius  $L_f$ . A computational <sup>8</sup> study on an equivalent two-dimensional configuration (circular disks) was carried out using <sup>9</sup> Discrete Element and Continuum (MPM) approaches.

<sup>10</sup> [Lajeunesse et al. \(2004\)](#) performed axis-symmetric and plane strain tests on granular <sup>11</sup> column collapse. Granular materials when released suddenly on a horizontal surface exhibit <sup>12</sup> transient flow. The mechanism of flow initiation, spreading dynamics and energy dissipation <sup>13</sup> are studied. The experimental configuration used by [Lajeunesse et al. \(2004\)](#) is shown in <sup>14</sup> Figure 4.2. Granular material of mass ‘M’ was poured into a container to form a rectangular <sup>15</sup> heap of length ‘ $L_i$ ’, height ‘ $H_i$ ’ and thickness ‘ $W$ ’. The internal friction angle and the wall <sup>16</sup> friction between the wall and the glass beads measured by [Lajeunesse et al. \(2004\)](#) are listed <sup>17</sup> in Table 4.1. The gate was then quickly removed to release the granular mass that spreads in <sup>18</sup> the horizontal channel until it comes to rest. The final run-out distance ‘ $L_f$ ’ and the collapsed <sup>19</sup> height ‘ $H_f$ ’ were measured. The run-out distance and collapse height were found to exhibit <sup>20</sup> power law relation with the initial aspect ratio ‘ $a$ ’ ( $= H_i/L_i$ ) of the column.

<sup>21</sup> In this study, numerical simulations of the granular column collapse experiments are per-  
formed by varying the initial aspect ratio of the column. Discrete Element Method simulations

## 4.2 Granular column collapse

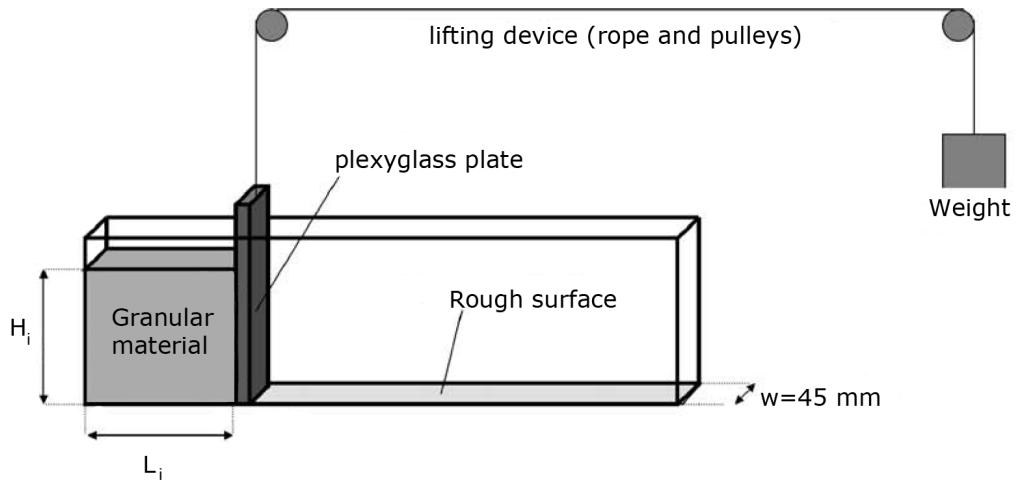


Figure 4.2 Schematic of experimental configuration for 2-D collapse in a rectangular channel, ([Lajeunesse et al., 2004](#))

Table 4.1 Material properties of glass ballotini, ([Lajeunesse et al., 2004](#))

Parameter	Value
Mean diameter	1.15 mm
Repose angle	$22 \pm 0.5^\circ$
Avalanche angle	$27.4 \pm 0.5^\circ$
Wall friction angle	$24.8 \pm 0.2^\circ$

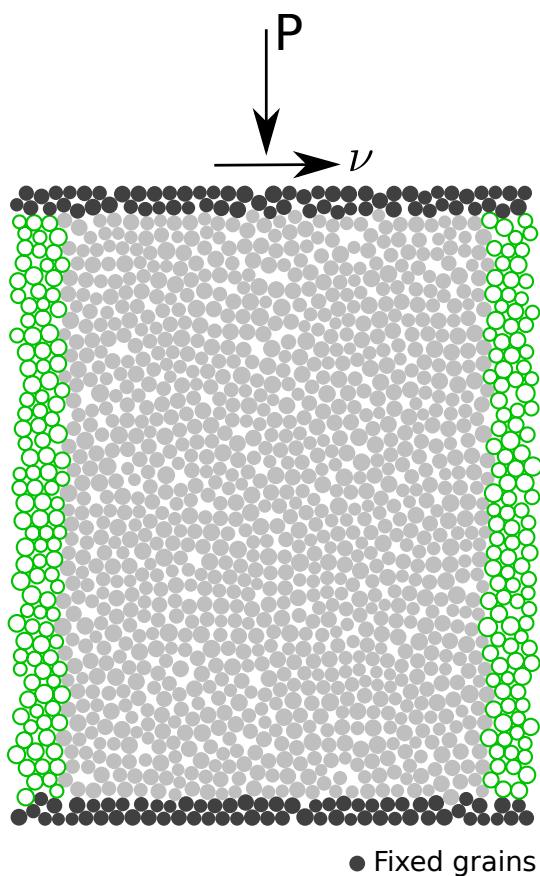


Figure 4.3 Shear test periodic boundary condition

involve modelling the granular column as individual particles. The granular column is prepared by randomly packing poly-disperse grains on a regular lattice and allowing them to undergo free fall due to gravity, forming a randomly packed granular column (see

Krishna Kumar Soundararajan: Add Chapter reference

The grains are allowed to reach a stable equilibrium after undergoing some elastic compressions due to gravity. The overlaps between particles are determined by the stiffness  $k_n$  of the spring in the normal direction. Typically, average overlaps in the range 0.1 to 1.0% are desirable and the spring constant is chosen to produce particle overlaps in this range. The stiffness is determined using the following equation:

$$k_n = \frac{2\pi G}{(1-\nu)[2\ln(\frac{2r}{A})-1]} \quad (4.1) \quad 10$$

$$A = \left[ \frac{2r(1-\nu)f_n}{\pi G} \right]^{\frac{1}{2}} \quad (4.2) \quad 11 \\ 12$$

where  $f_n$  is the normal contact force;  $G$  is the shear modulus;  $\nu$  is the Poisson's ratio and  $r$  is the radius of the grain. A simpler form of stiffness for a spherical grain is defined as:

$$k_n = 4ER \quad (4.3) \quad 15 \\ 16$$

where  $E$  is the Young's modulus of the material and  $R$  is the radius of the grain. The normal damping coefficient  $C_n$  is chosen to give a required coefficient of restitution  $\varepsilon$  (defined as the ratio of the post-collisional to pre-collisional normal component of the relative velocity) for the materials involved:

$$C_n = 2\gamma\sqrt{m_{ij}k_n} \quad (4.4) \quad 21$$

$$\text{where } \gamma = -\frac{\ln(\varepsilon)}{\sqrt{\pi^2 + \ln^2(\varepsilon)}}, \quad \text{and} \quad m_{ij} = \frac{m_i m_j}{m_i + m_j} \quad (4.5) \quad 22 \\ 23$$

10 Discrete Element Method simulations were carried out with columns having different initial aspect ratio 'a', varying from 0.2 to 10. In order to study the effect of crystallisation on the run-out distance, 10 more MD simulations were carried out on granular columns composed of grains arranged on a hexagonal lattice. In order to maintain a threshold amount of grains, in all the cases the columns contain at least 1000 grains, which is the safe lower limit for DEM as suggested by [Oda and Iwashita \(1999\)](#). The micro-mechanical parameters used in this study are presented in Table 4.2. Due to the unsteady nature of the flow, the grains get dispersed on the horizontal plane as discrete bodies start to separate from the main mass, hence the run-out distance is calculated as the position of the farthest grain which has at least one contact with the

Table 4.2 Micromechanical parameters used in Discrete Element Method simulations

Parameter	Value
Young's modulus of glass bead	$70 \times 10^9 \text{ N/m}^2$
Poisson's ratio	0.22 - 0.24
Diameter of glass beads	0.92 to 1.38 mm
Normal and shear stiffness of grains	$1.6 \times 10^8 \text{ N/m}$
Normal and shear stiffness of wall	$4 \times 10^8 \text{ N/m}$
Inter-particle friction coefficient, $\mu$	0.53
Friction coefficient of wall	0.466
Coefficient of restitution, $\Gamma$	0.4

Table 4.3 Parameters used in continuum simulations

Parameters	Value
Number of material points representing an actual particle	4
Material point spacing	0.575 mm
Number of material points per mesh	16
Young's Modulus, E	$1.98 \times 10^6 \text{ Pa}$
Poisson's ratio, $\nu$	0.22 to 0.24
Friction angle, $\phi$	$23.0^\circ$
Dilatancy angle, $\Phi$	$0^\circ$
Density, $\rho$	$1800 \text{ kg/m}^3$
Wall friction	0.466
Time step increment	$1.0 \times 10^{-6}$

<sup>1</sup> main mass. A plane strain collapse of granular column is simulated as a continuum using MPM.  
<sup>2</sup> The effect of number of material points on the accuracy of the simulation was discussed in  
<sup>3</sup> Chapter 4. [Guilkey et al. \(2003\)](#) suggests at least four particles per cell for problems involving  
<sup>4</sup> large deformations. 10 MPM simulations of the granular column collapse were performed using  
<sup>5</sup> Mohr-Coulomb constitutive law by varying the initial aspect ratio, to understand the difference  
<sup>6</sup> between the particle and continuum scale description of granular flows. The parameters used  
<sup>7</sup> for the continuum analyses are presented in Table. 4.3. The Young's modulus of the granular  
<sup>8</sup> assembly is determined by performing a uni-axial compression of the granular column in  
<sup>9</sup> Discrete Element Method.

#### <sup>10</sup> 4.2.1 Deposit morphology

<sup>11</sup> The variation of the normalized final run-out distance,  $\Delta L = (L_f - L_i)/L_i$ , with the initial aspect  
<sup>12</sup> ratio 'a' of the column is presented in Figure 4.4. Similar to the experimental results, a power

## 4.2 Granular column collapse

## 7

law relationship is observed between the normalized run-out distance and the initial aspect ratio of the column. However, the molecular dynamics simulations with random packing of grains overestimate the run-out distance by a factor of 1.2. In the present study, the following scaling law for the run-out is observed.

$$\frac{L_f - L_i}{L_i} \approx \begin{cases} 1.67a, & a \lesssim 2.3 \\ 2.5a^{2/3}, & a \gtrsim 2.3 \end{cases} \quad (4.6)$$

The run-out distance observed in the case of hexagonal packing of grains matches the experimental run-out distance observed by [Lajeunesse et al. \(2004\)](#). However, the Discrete Element Method simulations performed with random packing predict longer run-out distances in comparison with the experimental data. The difference in the run-out distance can be attributed to the variation in the packing of grains in the granular column. Although, experimental data corresponds to granular column collapse in a rectangular channel, the collapse is not a pure two-dimensional collapse as in the case of numerical simulations. This can cause some variation in the run-out distance. [Balmforth and Kerswell \(2005\)](#) observed that the material property affects the final run-out distance and included a pre-factor ‘ $\lambda$ ’ in the scaling law, which is in contrast to the observation made by [Lube et al. \(2005\)](#). The scaling law observed in the present study for the random packing is identical to the scaling law observed by [Lajeunesse et al. \(2004\)](#), except for the pre-factor in the scaling law, indicating a strong correlation between the run-out distance and the material property. [Daerr and Douady \(1999\)](#) also observed the effect of initial packing density and the internal structure on the behaviour of granular flows. The continuum description of the granular column collapse using Material Point Method shows good agreement with the experimental results for columns with lower aspect ratio (‘ $a$ ’  $\lesssim 2.3$ ), however it exhibits a significant increase in the run-out distance beyond the aspect ratio of 2.3. [Bandara \(2013\)](#) also observed a jump in the run-out distance at the initial aspect ratio of 2.

In order to understand the mechanism of the run-out in a granular column collapse, it is essential to study the relation between the final collapsed height of the granular column and its initial aspect ratio. Figure 4.5 shows the variation of the normalized final height with the initial aspect ratio of the column. The final height predicted by the Discrete Element Method and the MPM simulations matches the experimental data for granular columns with aspect ratio below 0.7, which indicates that the initial density of the column has negligible effect on the final collapse height. The scaling of final height of the column with the initial aspect ratio of

1 the column can be written as:

$$2 \quad \frac{H_f}{L_i} \propto \begin{cases} a, & a \lesssim 0.7 \\ 3 \quad a^{2/3}, & a \gtrsim 0.7 \end{cases} \quad (4.7)$$

4 The Material Point Method predicts a higher final height of the column in comparison with  
 5 the particular simulations that should result in shorter run-outs, however it is inconsistent with  
 6 the observations. In case of granular columns with smaller aspect ratios, only a tiny portion of  
 7 the total mass is mobilized and the rest remains static, thus predicting the final collapse height  
 8 accurately. The final height of a column is controlled by the amount of static region in the  
 9 granular column collapse, while the run-out distance is essentially a function of the flowing  
 10 mass. Hence, it is essential to compare the evolution of flow and the internal flow structure  
 11 in the Discrete Element Method and Material Point Method simulations to understand the  
 12 limitations of both the continuum and discrete element approaches.

Figure 4.4 Normalised final runout distance for columns with different initial aspect ratio

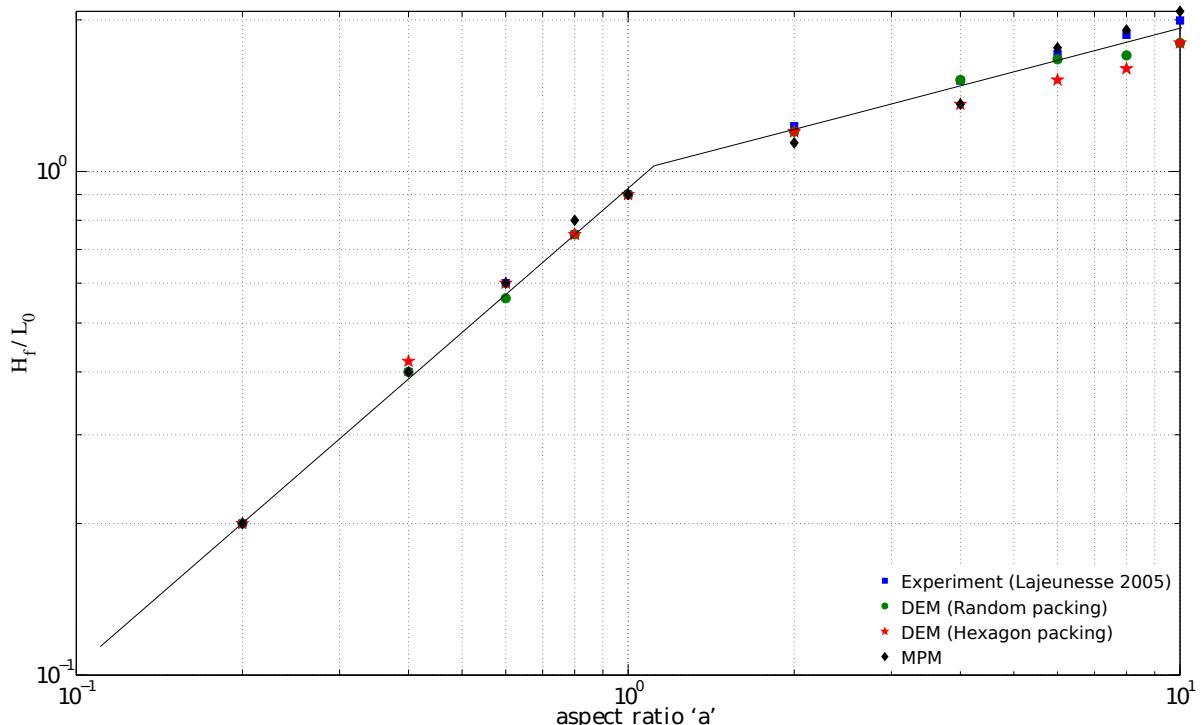


Figure 4.5 Normalised final collapse height for columns with different initial aspect ratio

### 4.2.2 Flow evolution and internal flow structure

For a fixed granular material and substrate properties, the flow dynamics and the final deposit morphology do not depend on the volume of granular material released, but depend only on the aspect ratio ‘ $a$ ’ of the column. A power law relationship is observed between the run-out distance and the initial aspect ratio of the column. A transition in the run-out behaviour at an aspect ratio of 2.3 indicates a change in the flow dynamics. For smaller aspect ratios, the granular mass fails through avalanching of flanks producing a truncated cone-like deposit ( $a' < 0.7$ ) or conical deposit ( $a' > 0.7$ ). At smaller values of aspect ratios, the flow is initiated by failure at the edge of the pile along a well-defined fracture surface. The grains located above the failure surface move “*en masse*” leaving a static region underneath the failure surface. After a transient time of order  $\tau_c$ , defined as  $\sqrt{H_i/g}$ , the flow is fully developed. The velocity profile along the granular column at critical time  $\tau_c$  is presented in Figure 4.6. At critical time, the velocity field depends only on the position of the grain along the sliding mass. The maximum velocity is observed at the front of the flowing mass corresponding to that of a plug flow in horizontal direction. Particulate and continuum simulations yield similar run-out distance at critical time. Unlike particulate simulations, the Material point method predicts that the maximum horizontal velocity occurs at the top of the sliding mass. Behind the fast flowing front, the flow is localized in the mass above the failure surface and the velocity profiles are locally parallel to the failure plane. The flow is composed of upper linear part and a lower exponential tail in the static granular bed. The velocity profile is similar to steady granular surface flow as observed by Lajeunesse et al. (2004).

For columns with lower initial aspect ratios, the run-out distance is proportional to the mass flowing above the failure surface. To understand the amount of mass mobilized during a collapse, the angle of the failure surface has to be studied. Figure 4.6 shows a distinct failure surface when the flow is fully developed at critical time  $\tau_c$ . The angle of the failure surface is found to be about  $55^\circ$ . The failure surface begins from the toe of the column and protrudes inwards at an angle of 50 to  $55^\circ$ . For columns with lower aspect ratios, the formation of the “truncated conical deposit” or “conical deposit” depends only on the initial length of the column, as the angle of the failure surface is found to be independent of the aspect ratio. The failure angle is consistent with the interpretation in terms of *active Coulomb failure* (Lajeunesse et al., 2004), which leads to a predicted failure angle  $\theta_y = 45^\circ + \delta/2$ , where  $\delta$  is the internal friction angle of the granular material. In the present study, the friction angle of the glass beads is  $22^\circ$ , which leads to  $\theta_y = 45^\circ + 22^\circ/2 = 56^\circ$ , which is in good agreement with the numerical simulations and experimental observations by Lajeunesse et al. (2004). Contrary to the suggestion of Lajeunesse et al. (2004), the fracture angle is found to have no direct effect on the transition between the truncated cone and the conical deposit occurring at an aspect ratio

of 0.7. [Schaefer \(1990\)](#) observed the onset of instabilities in a narrow wedges of 56 to 65° for Cambridge type constitutive models that describes granular flows. This observation matches well with the failure angle observed in the present study. The final profile of the collapsed granular column with an initial aspect ratio of 0.4 is shown in Figure 4.7. The failure surface is distinct and the hexagonal dense packing of grains has a steeper failure surface in comparison with the random packing. The variation observed in the angle of the failure surface causes a difference in the amount of mobilized mass above the failure surface, and in turn in the run-out distance. The lower value of run-out distance observed in the case of hexagonal packing of grains can be attributed to the crystallisation effects. crystallisation is the formation of large-scale lattice structures during the flow, resulting in non-generic flow patterns. crystallisation is found to have a significant effect on the final state of the granular column. [Lacaze and Kerswell \(2009\)](#) observed that poly-disperse grains have lesser tendency to crystallize especially in the case of columns with larger aspect ratio.

For larger aspect ratios, the flow is still initiated by a well defined failure surface as can be seen in Figure 4.8. However, in this case the initial granular column is much higher than the top of the failure surface. Due to gravity most of the grains in the column fall in the vertical direction consuming the column along their way. When they reach the vicinity of the failure surface, the flow gets deviated along the horizontal direction releasing a huge amount of kinetic energy gained during the free fall. For larger aspect ratio ( $a > 0.7$ ), the resulting static region is a cone, the final height of the cone, i.e.,  $H_f$  lies above the summit of the failure surface. Hence, a different evolution is observed from that of the axis-symmetric geometry ([Lube et al., 2005](#)), where the final height coincides with the summit of the failure surface forming a truncated conical deposit. [Lajeunesse et al. \(2004\)](#) articulated the variation in the deposit morphology between the axis-symmetric case and the rectangular collapse to be a geometrical effect rather than as an experimental artefact. The final profile of the collapsed granular column with an initial aspect ratio of 6 is presented in Figure 4.9. An initial failure surface starting from the toe end of the column at an angle of about 55° can be observed at critical time  $\tau_c$ . As the collapse of the granular collapse progresses, successive failure planes parallel to the initial failure surface are formed and shear failure occurs along these planes. The presence of several shear bands in the final profile of the collapsed granular column confirms the hypothesis. crystallisation in hexagonal packing causes a significant effect on the run-out distance by forming series of parallel shear bands. However, the Material Point Method fails to capture the formation of shear bands during the collapse. This observation throws light on the mechanics of propagation of shear bands in massive landslides such as the Storegga submarine landslide. The flow behaviour becomes similar to that of columns with lower aspect ratio as the flow starts descending along the failure plane. Regardless of the experimental configuration and the initial aspect ratio of

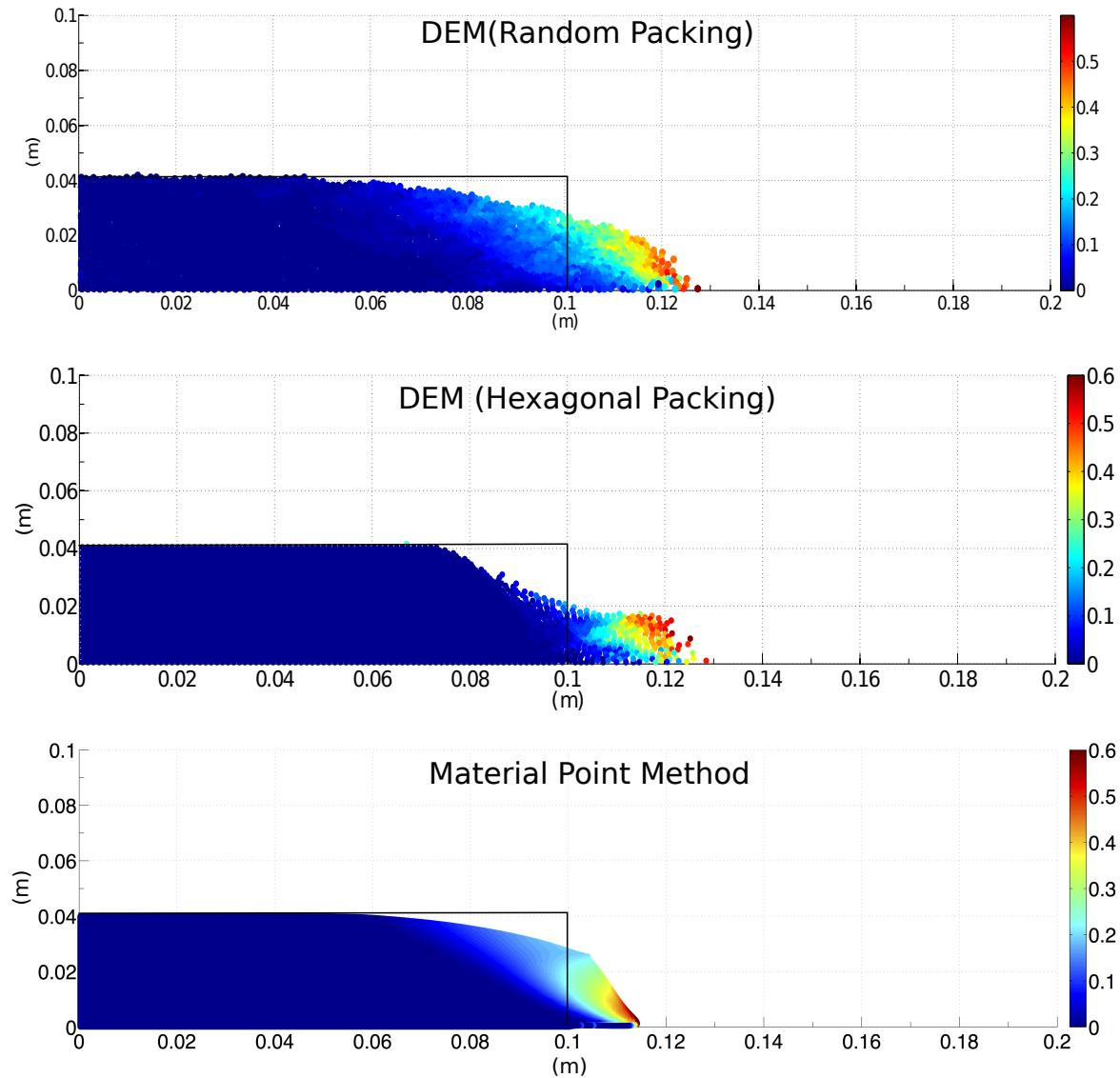


Figure 4.6 Velocity profile of a granular column collapse ( $a' = 0.4 \& t = \tau_c$ )

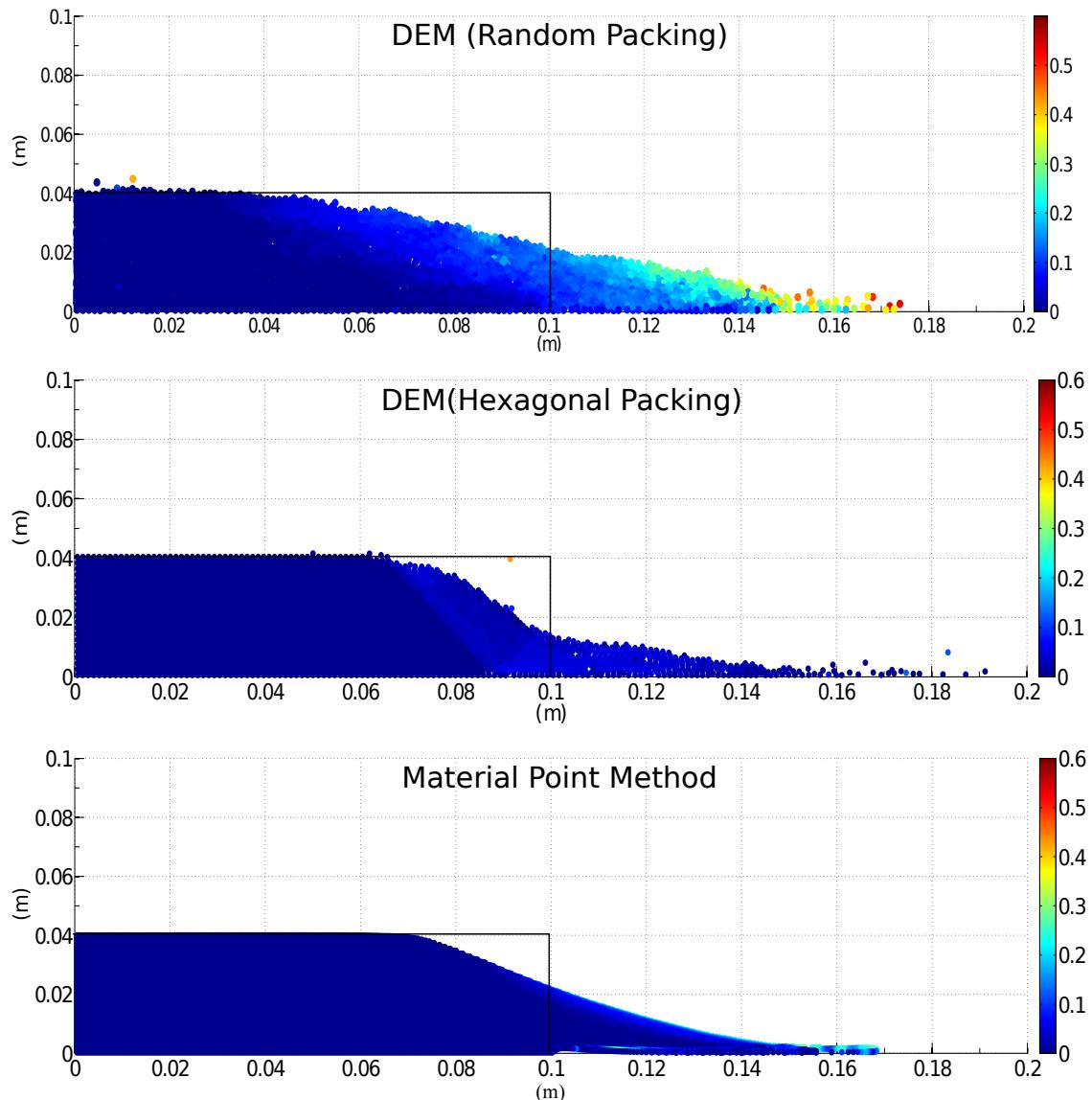


Figure 4.7 Velocity profile of a granular column collapse ( $a' = 0.4$  &  $t = 3 \times \tau_c$ )

the columns, the flow is initiated by a well-defined rupture surface, above which the material slides down leaving a static region underneath the failure plane. Depending on the aspect ratio of the column, two asymptotic behaviours are observed. For smaller aspect ratios, the flow is dominated by friction and by the pressure gradient for larger aspect ratio.

To study the flow dynamics of granular columns with different aspect ratios, the flow front  $L(t)$  and the maximum height of column  $H(t)$  are tracked. The evolution of scaled height ( $H_f/L_i$ ) and the run-out distance  $(L_f - L_i)/L_i$  with time for granular columns with an initial aspect ratio of 0.4 and 6 are presented in Figures 4.10 and 4.11. Time is scaled with respect to the critical time  $\tau_c$ , defined as the time at which the flow is fully mobilized. Three distinct regions can be observed in the flow evolution of granular column collapse regardless of the initial aspect ratio of the column. An initial transient acceleration phase is observed for a time  $0.8\tau_c$ . This phase is followed by a heap movement of granular materials at the foot with a constant spreading velocity  $V$  for about  $2\tau_c$ . When time ' $t$ ' >  $\tau_c$ , the velocity varies linearly with depth in the flowing layer and decreases exponentially with depth near the static layer. This velocity profile is similar to those observed in steady granular surface flows (Lajeunesse et al., 2004). Most of the run-out happens during this phase. The final phase involves deceleration of the flow front and the flow comes to rest after  $0.6\tau_c$ . The spreading of the granular column ceases after a time in the order of  $3\tau_c$  for all values of aspect ratios, however some motion still persists along the free surface behind the flow front for a much longer time due to internal rearrangement, the duration of which can last up to  $t \approx 6\tau_c$ . For smaller aspect ratios, the critical time is evaluated as the point of intersection of the scaled run-out and height. The critical time predicted for both hexagonal and random packing of grains matches the experimental observations. However, the Material Point Method overestimates the critical time by a factor of 1.25, which means that it takes longer for the flow to be fully mobilized. However, the actual run-out duration is short and the granular materials comes to rest abruptly at about  $t = 3\tau_c$ . For columns with larger aspect ratios, the continuum and particulate approaches simulate similar flow evolution behaviour for times up to  $3\tau_c$ , beyond which particulate simulations stabilise and come to rest, while the flow continues to evolve in MPM simulations resulting in larger run-outs than expected. The flow tends to come to rest at time  $t = 6\tau_c$ . The three phases in a granular flow can be distinctly observed in the flow evolution plot for a granular column with initial aspect ratio of 6 (see Figure figure 4.11). For larger aspect ratios, the flow evolution behaviour observed in the case of random packing matches the experimental observation by Lajeunesse et al. (2004). Hexagonal packing predicts longer time for the flow to evolve, which can be attributed to the increase in the internal resistance due to crystallisation of grains. MPM overestimates the critical time by 50%, however has the same value of run-out as the particulate simulations, at time  $t = 3\tau_c$ , beyond which the material continues to flow until it ceases at  $6\tau_c$ . In order to

- <sup>1</sup> understand the flow dynamics in the case of Material Point Method it is important to study the  
<sup>2</sup> effect of different parameters on the deposit morphology.

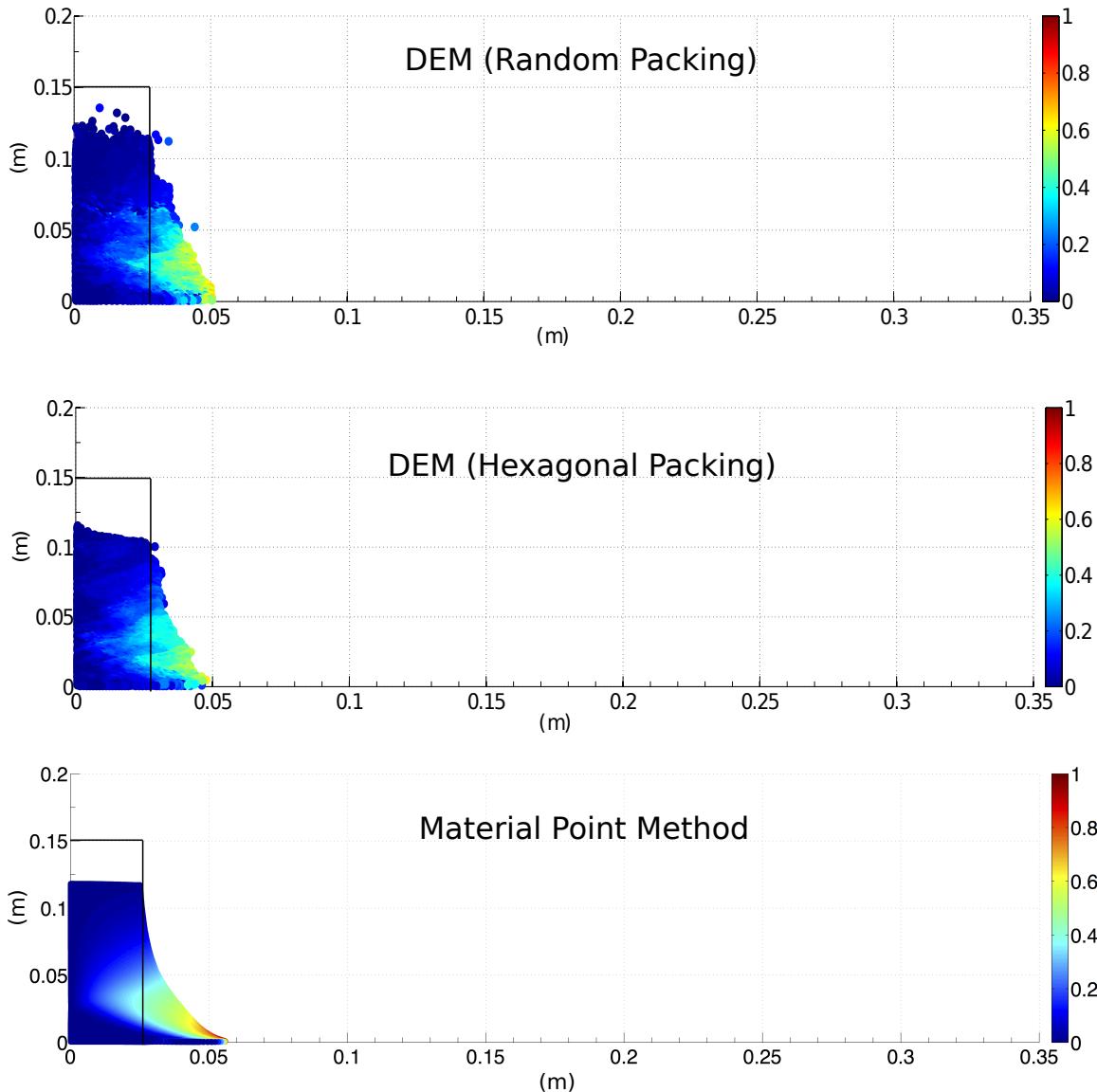


Figure 4.8 Velocity profile of a granular column collapse (' $a' = 6 \& t = \tau_c$ )

### <sup>3</sup> 4.2.3 Energy dissipation mechanism

- <sup>4</sup> The time evolution of the flow exhibited three distinct stages during the collapse of a granular  
<sup>5</sup> column. Studying the energy dissipation mechanism provides useful insight into the flow  
<sup>6</sup> dynamics. shows the time evolution of potential energy ( $E_p$ ) and kinetic energy ( $E_k$ ) normalized

## 4.2 Granular column collapse

## 15

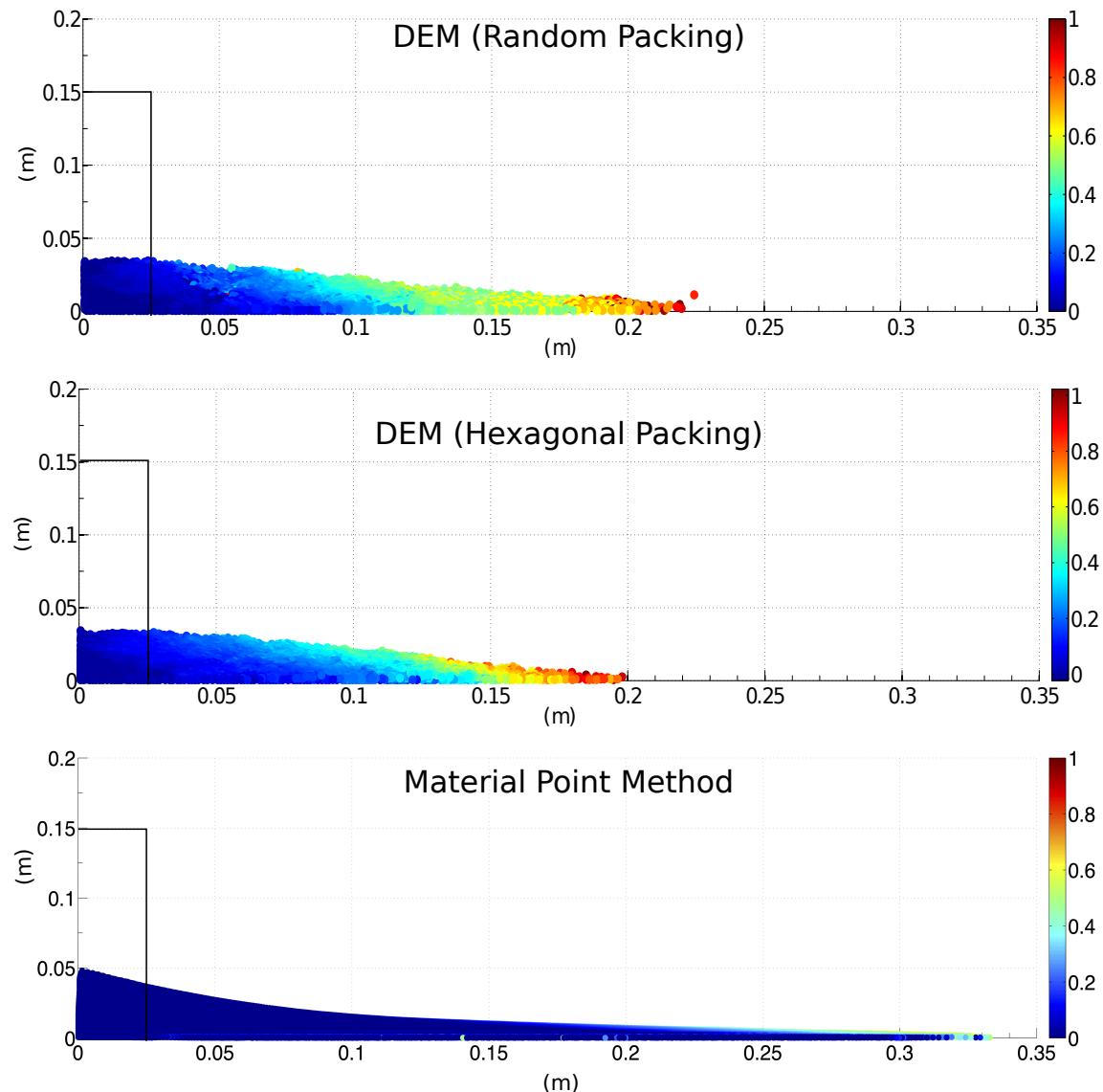
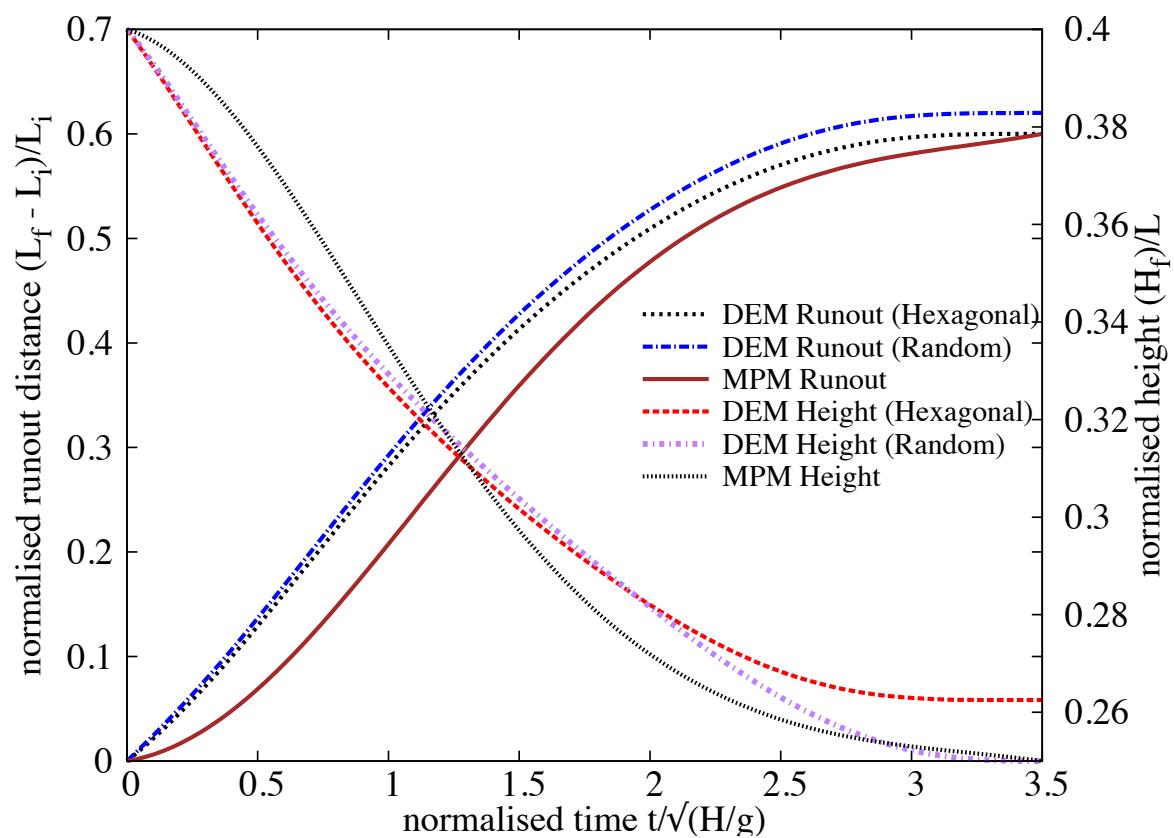
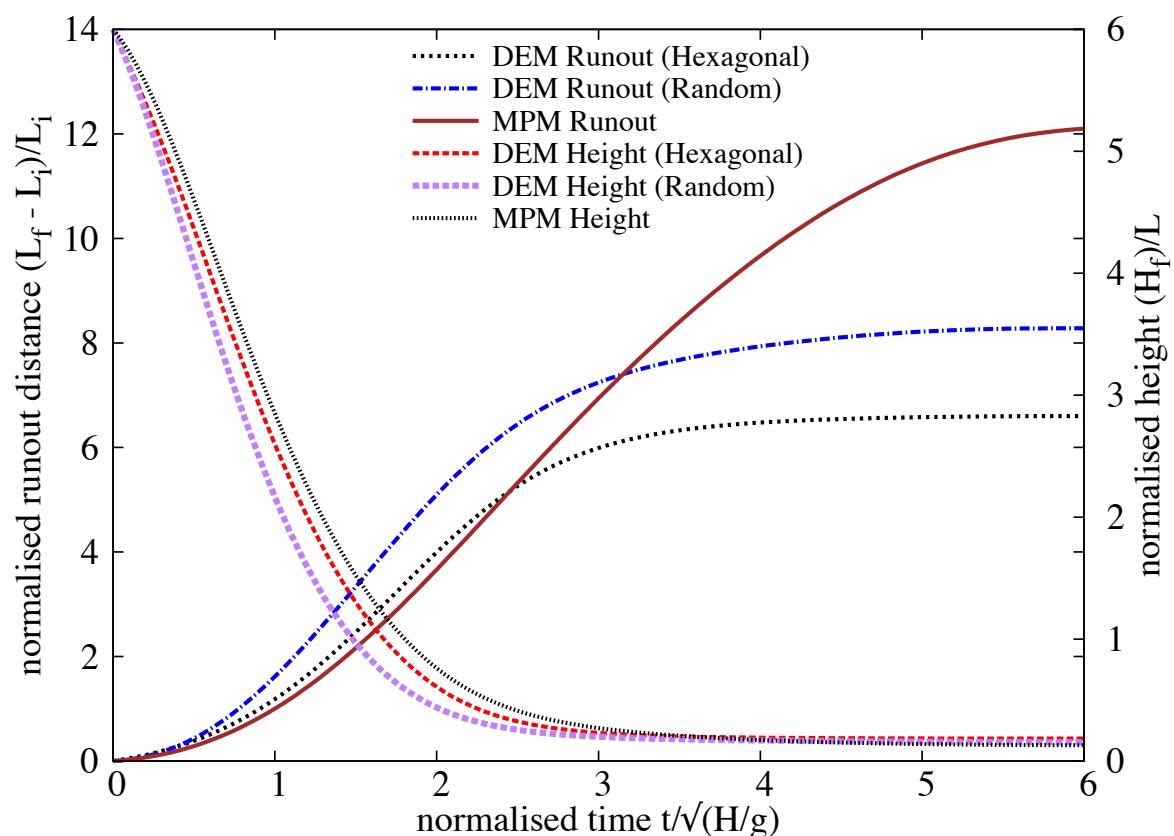


Figure 4.9 Velocity profile of a granular column collapse ( $a' = 6$  &  $t = 3 \times \tau_c$ )

Figure 4.10 Flow evolution of a column with ' $a'$  = 0.4

Figure 4.11 Flow evolution of a column with ' $a'$  = 6

1 by the initial potential energy  $E_o$ .

$$2 \quad E_p = \sum_{p=1}^{N_p} m_p g h_p \quad (4.8)$$

$$3 \quad E_{ki} = \frac{1}{2} \sum_{p=1}^{N_p} m_p v_p^2 \quad (4.9)$$

4 where  $N_p$  is the total number of particles,  $m_p$  is the mass of a particle ‘ $p$ ’,  $h_p$  is the height and  
 5  $v_p$  is the velocity of the particle ‘ $p$ ’. It can be observed from the figure that the initial potential  
 6 energy stored in the particle is converted to kinetic energy which is dissipated as the granular  
 7 material flows down. Three successive stages can be identified in the granular column collapse.

8 In the initial acceleration stage ( $t < 0.8\tau_c$ ), the initial potential energy stored in the grains is  
 9 converted into vertical motion. In the second stage, the grains undergo collisions with the  
 10 bottom plane and/or with neighbouring grains, and the stored potential energy is converted into  
 11 horizontal motion. In the third stage, the grains eventually leave the base area of the column  
 12 and flow sideways. As the process involves collective dynamics of all the particles, it is difficult  
 13 to predict the exact trajectory of a grain, however, the overall dynamics can be explained. To  
 14 explain the dissipation of energy during the collapse, [Staron et al. \(2005\)](#) assumed that the total  
 15 initial potential energy stored in the system is completely dissipated through friction over the  
 16 entire run-out distance as:

$$18 \quad \mu m_o g \times (L_f - L_i) = m_o g H_o \quad (4.10)$$

19 where  $\mu$  is the friction coefficient. The model predicts well the flow dynamics for columns  
 20 with larger aspect ratios, as most of the initial potential energy is dissipated during the collapse  
 21 involving the entire column. However, for columns with smaller aspect ratios, only a portion of  
 22 the mass above the failure surface is involved in the flow. Hence, the energy dissipation should  
 23 involve only the grains lying above the failure surface. A mathematical model, which considers  
 24 the grains lying above the failure surface, will be derived to predict the flow dynamics of the  
 25 granular column collapse for different aspect ratios.

# Chapter 5

## Numerical modelling of fluid–grain interactions

### 5.1 Fluid simulation using lattice Boltzmann method

Grain–fluid systems can be found in many scientific and engineering applications, such as suspensions, fluidised beds, sediment transport, and geo-mechanical problems. In general, the fundamental physical phenomena in these systems are not well understood mainly due to the intricate complexity of grain–fluid interactions and the lack of powerful analysis tools (Han et al., 2007a). In addition to the interaction among soil grains, the motion of soil grains is mainly driven by gravity and the hydrodynamic force exerted by the fluid. The fluid flow pattern can be significantly affected by the presence of soil grains and this often results in a turbulent flow. Hence, the development of an effective numerical framework for modelling both the fluid flow patterns and the grain–fluid interactions is very challenging.

Development of a numerical framework depends crucially on the size of the soil grains relative to the domain/mesh size (Feng et al., 2007). Traditionally, the Navier-Stokes equation is solved by a grid-based Computational Fluid Dynamics (CFD) method, such as the Finite Volume Method, FVM, (Capelatro and Desjardins, 2013) or a mesh-free technique such as Smooth Particle Hydrodynamics (SPH) (Sun et al., 2013). The grid size in FVM or the smooth length in SPH for discretisation of the Navier-Stokes equation is at least an order of magnitude larger than the grain diameter (Xiong et al., 2014).

In situations where the average domain concentration phase is far from dilute, the computational effort is mostly devoted to the grain dynamics. The hydrodynamic forces on the soil grains are applied based on an empirical relation using the domain-averaged local porosity of the soil grains in the grid. As a result, developing a fast fluid hydrodynamics solver

1 is unimportant for dense flows. However, most geo-mechanical problems involve complex  
2 interactions between the solid and the fluid phase. This requires accurate modelling of the fluid  
3 flow pattern. Additionally, geophysical problems, such as submarine landslides and debris flow  
4 have a relatively large simulation domain, which requires parallel computation. Implementing  
5 traditional grid-based CFD methods face great challenges on multi-processor systems (Xiong  
6 et al., 2014). Although mesh-free approaches are free from the problem of parallel scalability,  
7 its modelling accuracy and speed are relatively low when compared to grid-based CFD methods.  
8 Therefore, an accurate, fast and a highly scalable scheme is required to model fluid - grain  
9 systems in geo-mechanics.

10 The Navier-Stokes equation describes the motion of a non-turbulent Newtonian fluid. The  
11 equation is obtained by applying Newton's second law to the fluid motion, together with an  
12 assumption that the fluid stress is the sum of the viscous term, proportional to the gradient of  
13 the velocity, and the pressure term. Conventional CFD methods compute pertinent flow fields,  
14 such as velocity  $u$  and pressure  $p$ , by numerically solving the Navier-Stokes equation in space  $x$   
15 and time  $t$ . Alternatively, the transport equation or the Boltzmann equation, which deals with a  
16 single particle distribution function  $f(x, \xi, t)$  in phase space  $(x, \xi)$  and time  $t$ , can be used to  
17 solve various problems in fluid dynamics.

18 The Lattice Boltzmann Method (LBM) (Chen and Doolen, 1998; Han et al., 2007b; He and  
19 Luo, 1997a,b; Mei et al., 2000; Zhou et al., 2012) is an alternative approach to the classical  
20 Navier-Stokes solvers for fluid flows. LBM works on an equidistant grid of cells, called lattice  
21 cells, which interact only with their direct neighbours. In LBM, the discretisation of continuum  
22 equations is based on microscopic models and mesoscopic continuum theories. LBM is a  
23 special discretising scheme of the Boltzmann equation where the particle distribution functions  
24 (mass fractions) collide and propagate on a regular grid. The important aspect, however, is  
25 the *discretisation of the velocity*, which means that the particle velocities are restricted to a  
26 predefined set of orientations.

27 The theoretical premises of the LB equation are that (1) hydrodynamics is insensitive to  
28 the details of microscopic physics, and (2) hydrodynamics can be preserved so long as the  
29 conservation laws and associated symmetries are respected in the microscopic and mesoscopic  
30 level. Therefore, the computational advantages of LBM are achieved by drastically reducing  
31 the particle velocity space  $\xi$  to only a very few discrete points without seriously degrading  
32 the hydrodynamics (Mei et al., 2000). This is possible because LBM rigorously preserves the  
33 hydrodynamic moments of the distribution function, such as mass density and momentum fluxes,  
34 and the necessary symmetries (He and Luo, 1997a,b). LBM has evolved as a comprehensive  
35 fluid solver and its theoretical aspects link well with the conventional central finite difference  
36 scheme (Cook et al., 2004).

### 5.1.1 Formulation

LBM is a ‘micro-particle’ based numerical time-stepping procedure for the solution of incompressible fluid flows. Consider a 2D incompressible fluid flow with density  $\rho$  and kinematic viscosity  $\nu$ , in a rectangular domain  $D$ . The fluid domain is divided into rectangular grids or lattices, with the same grid length ‘ $h$ ’ in both  $x$ - and  $y$ -directions (see figure 5.1).

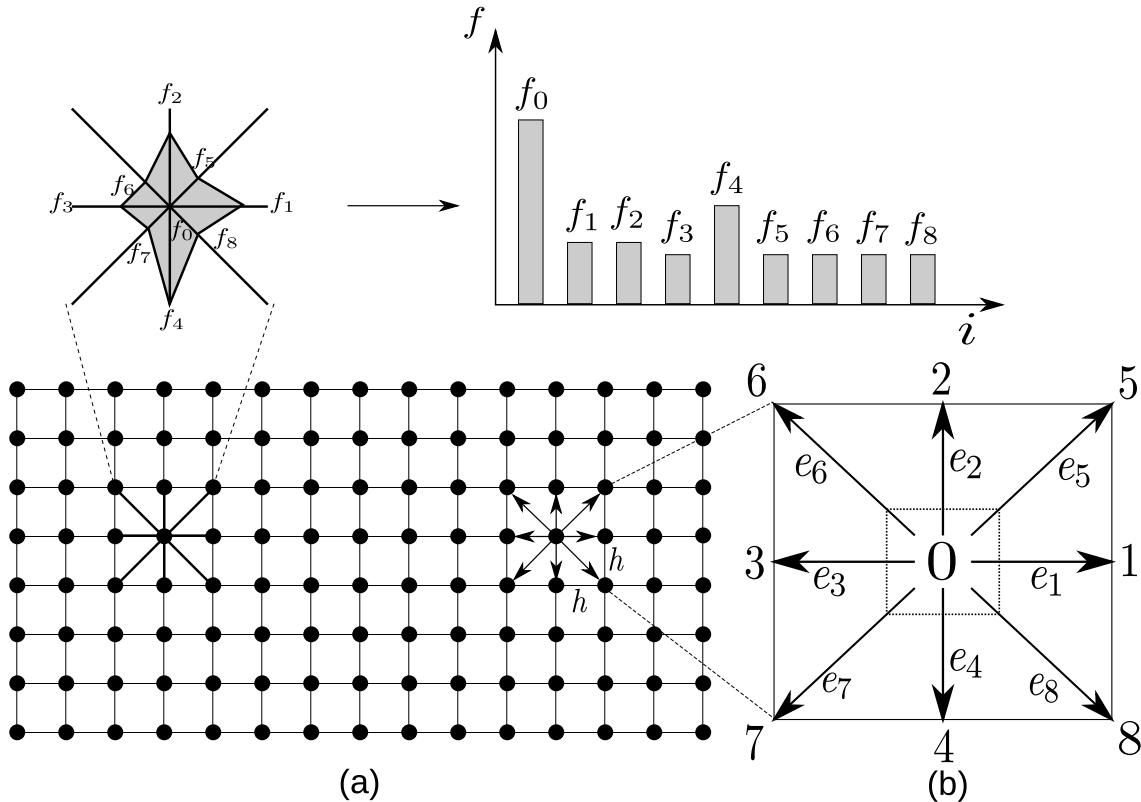


Figure 5.1 The Lattice Boltzmann discretisation and D2Q9 scheme: (a) a standard LB lattice and histogram views of the discrete single particle distribution function/direction-specific densities  $f_i$ ; (b) D2Q9 model.

These lattices are usually classified in the literature using the  $D\alpha Q\beta$ -notation, where  $\alpha$  denotes the space dimensionality and  $\beta$  is the number of discrete velocities (but also including the possibility of having particles at rest) within the momentum discretisation. The most common lattices are the  $D2Q9$  and the  $D3Q19$ -models, see He et al. (1997). The present study focuses on two-dimensional problems, hence the  $D2Q9$  momentum discretisation is adopted.

LBM discretises the Boltzmann equation in space to a finite number of possible particle spatial positions, microscopic momenta, and time. Particle positions are confined to the lattice nodes. The fluid particles at each node are allowed to move to their eight intermediate neighbours with eight different velocities  $e_i (i = 1, \dots, 8)$ . A particle can remain at its own node,

which is equivalent to moving with zero velocity  $e_o$ . The particle mass is uniform, hence these microscopic velocities and momentum are always effectively equivalent (Han et al., 2007b). Referring to the numbering system shown in figure 5.1, the nine discrete velocity vectors are defined as

$$\begin{cases} e_0 = (0, 0); \\ e_1 = C(1, 0); e_2 = C(0, 1); e_3 = C(-1, 0); e_4 = C(0, -1); \\ e_5 = C(1, 1); e_6 = C(-1, 1); e_7 = C(-1, -1); e_8 = C(1, -1), \end{cases} \quad (5.1)$$

where  $C$  is the lattice speed that is defined as  $C = h/\Delta t$ , and  $\Delta t$  is the discrete time step. The primary variables in LB formulation are called the *fluid density distribution functions*,  $f_i$ , each representing the probable amount of fluid particles moving with the velocity  $e_i$  along the direction  $i$  at each node. The macroscopic variables are defined as functions of the particle distribution function (see figure 5.1)

$$\begin{cases} \rho = \sum_{i=0}^{\beta-1} f_i & \text{(macroscopic fluid density)} \\ \text{and} \\ \overrightarrow{u} = \frac{1}{\rho} \sum_{i=0}^{\beta-1} f_i \overrightarrow{e}_i & \text{(macroscopic velocity)}, \end{cases} \quad (5.2)$$

where  $i \in [0, \beta - 1]$  is an index spanning the discretised momentum space. There are nine fluid density distribution functions,  $f_i (i = 0, \dots, 8)$ , associated with each node in the *D2Q9* model. The evolution of the density distribution function at each time step for every lattice point is governed by

$$f_i(\mathbf{x} + \mathbf{e}_i \Delta t, t + \Delta t) = f_i(\mathbf{x}, t) - \frac{1}{\tau} [f_i(\mathbf{x}, t) - f_i^{eq}(\mathbf{x}, t)] \quad (i = 0, \dots, 8), \quad (5.3)$$

where for any grid node  $\mathbf{x}$ ,  $\mathbf{x} + \mathbf{e}_i \Delta t$  is its nearest node along the direction  $i$ .  $\tau$  is a non-dimensional relaxation time parameter, which is related to the fluid viscosity; and  $f_i^{eq}$  is termed as the equilibrium distribution function that is defined as

$$\begin{cases} f_0^{eq} = w_0 \rho (1 - \frac{3}{2C^2} \mathbf{v} \cdot \mathbf{v}) \\ \text{and} \\ f_i^{eq} = w_i \rho (1 + \frac{3}{C^2} \mathbf{e}_i \cdot \mathbf{v} \frac{9}{2C^2} (\mathbf{e}_i \cdot \mathbf{v})^2 - \frac{3}{2C^2} \mathbf{v} \cdot \mathbf{v}) \quad (i = 0, \dots, 8), \end{cases} \quad (5.4)$$

## 5.1 Fluid simulation using lattice Boltzmann method

## 23

in which,  $w_i$  represents the fixed weighting values:

$$w_0 = \frac{4}{9}, \quad w_{1,2,3,4} = \frac{1}{9}, \quad \text{and} \quad w_{5,6,7,8} = \frac{1}{36}. \quad (5.5)$$

The right-hand side of [eq. 5.3](#) is often denoted as  $f_i(\mathbf{x}, t_+)$  and termed the post collision distribution. LBM ensures conservation of total mass and total momentum of the fluid particles at each lattice node (see [eq. 5.3](#)). The lattice Boltzmann modelling consists of two phases: *collision* and *streaming*. The collision phase computed in the right-hand side of [eq. 5.3](#) involves only those variables that are associated with each node  $\mathbf{x}$ , and therefore is a local operation. The streaming phase then explicitly propagates the updated distribution functions at each node to its neighbours  $\mathbf{x} + e_i \Delta t$ , where no computations are required and only data exchange between neighbouring nodes are necessary. These features, together with the explicit time-stepping nature and the use of a regular grid, make LB computationally efficient, simple to implement and easy to parallelise ([Han et al., 2007b](#)).

The streaming step involves the translation of the distribution functions to their neighbouring sites according to the respective discrete velocity directions, as illustrated in figure [5.2](#) in the *D2Q9* model. The collision step, (see figure [5.3](#)) consists of re-distribution the local discretised Maxwellian equilibrium functions in such a way that local mass and momentum are invariants. In incompressible flows, the energy conservation is equivalent to the momentum conservation ([He et al., 1997](#)).

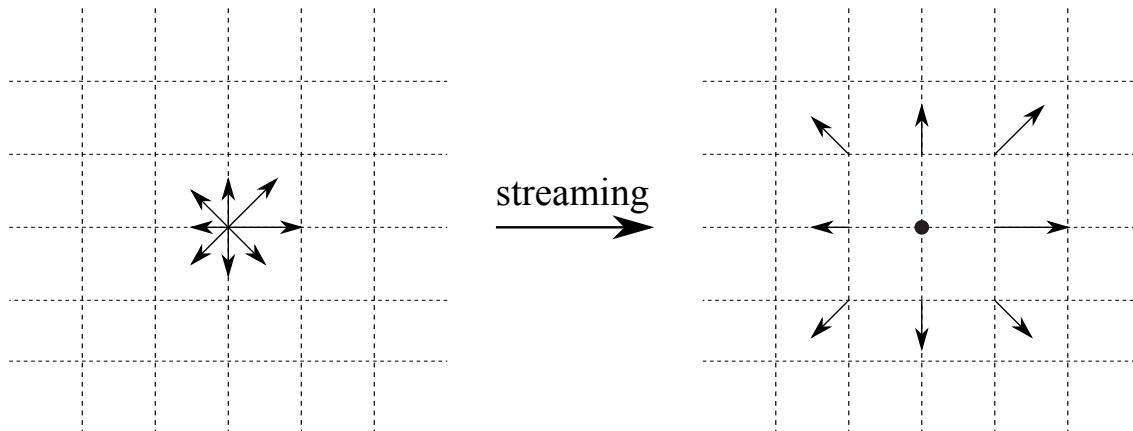


Figure 5.2 Illustration of the streaming process on a *D2Q9* lattice. The magnitude of the distribution functions remains unchanged, but they move to a neighbouring node according to their direction.

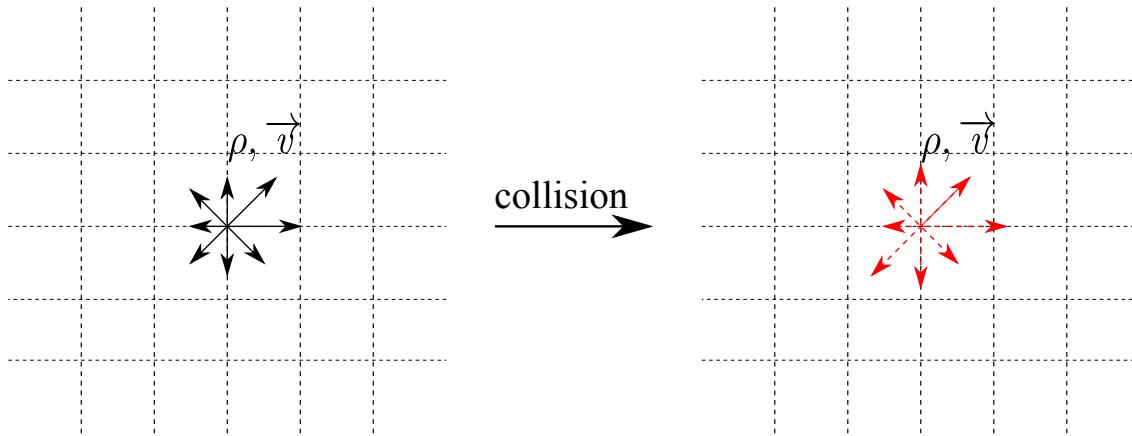


Figure 5.3 Illustration of the collision process on a  $D2Q9$  lattice. The local density  $\rho$  and velocity  $v$  are conserved, but the distribution functions change according to the relaxation to local Maxwellian rule.

<sup>1</sup> The standard macroscopic fluid variables, such as density  $\rho$  and velocity  $v$ , can be recovered  
<sup>2</sup> from the distribution functions as

$$\rho = \sum_{i=0}^8 f_i, \quad \text{and} \quad \rho \mathbf{v} = \sum_{i=0}^8 f_i e_i. \quad (5.6)$$

<sup>4</sup> The fluid pressure field ‘ $p$ ’ is determined by the equation of state

$$p = C_s^2 \rho, \quad (5.7)$$

<sup>6</sup> where  $C_s$  is termed the fluid speed of sound and is related to the lattice speed  $C$  as

$$C_s = C/\sqrt{3}. \quad (5.8)$$

<sup>8</sup> The kinematic viscosity of the fluid  $\nu$  is implicitly determined by the model parameters  $h$ ,  
<sup>9</sup>  $\Delta t$  and  $\tau$  as

$$\nu = \frac{1}{3}(\tau - \frac{1}{2}) \frac{h^2}{\Delta t} = \frac{1}{3}(\tau - \frac{1}{2}) Ch, \quad (5.9)$$

<sup>11</sup> which indicates that these three parameters are related to each other and have to be appropriately  
<sup>12</sup> selected to represent the correct fluid viscosity. An additional constraint to the parameter  
<sup>13</sup> selection is the lattice speed  $C$ , which must be sufficiently large in comparison to the maximum  
<sup>14</sup> fluid velocity  $v_{max}$ , to ensure accuracy of the solution. The ‘computational’ Mach number,  $M_a$ ,  
<sup>15</sup> defined as

$$M_a = \frac{v_{max}}{C}. \quad (5.10)$$

## 5.1 Fluid simulation using lattice Boltzmann method

25

Theoretically, for an accurate solution, the Mach number is required to be  $<< 1$ . In practice,  $M_a$  should be at least smaller than 0.1 (He et al., 1997). From a computational point of view, it is more convenient to choose  $h$  and  $\tau$  as two independent parameters and  $\Delta t$  as the derived parameter

$$\Delta t = (\tau - \frac{1}{2}) \frac{h^2}{3v}. \quad (5.11)$$

It can be observed that  $\tau$  has to be greater than 0.5 (He et al., 1997). Since there is no a priori estimation available to determine appropriate values of  $h$  and  $\tau$ , for a given fluid flow problem and a known fluid viscosity  $v$ , a *trial and error* approach is employed to ensure a smaller *Mach Number*. This is similar to choosing an appropriate Finite Element mesh size, without using automatic adaptive mesh techniques.

## 5.1.2 Lattice Boltzmann - Multi-Relaxation Time (LBM-MRT)

The Lattice Boltzmann Bhatnagar-Gross-Krook (LGBK) method is capable of simulating various hydrodynamics, such as multiphase flows and suspensions in fluid (Succi, 2001; Succi et al., 1989). However, LBM suffers from numerical instability when the dimensionless relaxation time  $\tau$  is close to 0.5. The Lattice Boltzmann Method – Multi-Relaxation Time (LBM-MRT) overcomes the deficiencies of linearised single relaxation LBM-BGK approach, such as the fixed Prandtl number ( $Pr=v/\kappa$ ), where the thermal conductivity ‘ $\kappa$ ’ is unity (Liu et al., 2003). LBM-MRT offers better numerical stability and has more degrees of freedom. In LBM-MRT the advection is mapped onto the momentum space by a linear transformation and the flux is finished within the velocity space (Du et al., 2006).

The lattice Boltzmann equation with multiple relaxation time approximation is written as

$$f_\alpha(\mathbf{x} + \mathbf{e}_i \Delta_t, t + \Delta_t) - f_\alpha(\mathbf{x}, t) = -\mathbf{S}_{\alpha i}(f_i(\mathbf{x}, t) - f_i^{eq}(\mathbf{x}, t)), \quad (5.12)$$

where  $\mathbf{S}$  is the collision matrix. The nine eigen values of  $\mathbf{S}$  are all between 0 and 2 so as to maintain linear stability and separation of scales. This ensures that the relaxation times of non-conserved quantities are much faster than the hydrodynamic time scales. The LGBK model is a special case in which the nine relaxation times are all equal and the collision matrix  $\mathbf{S} = \frac{1}{\tau} \mathbf{I}$ , where  $\mathbf{I}$  is the identity matrix. The evolutionary progress involves two steps, advection and flux:

$$f_\alpha^+(\mathbf{x}, t) - f_\alpha(\mathbf{x}, t) = -\mathbf{S}_{\alpha i}(f_i(\mathbf{x}, t) - f_i^{eq}(\mathbf{x}, t)) \quad (5.13)$$

$$f_\alpha(\mathbf{x} + e_\alpha \Delta_t, t + \Delta_t) = f_\alpha^+(\mathbf{x}, t). \quad (5.14)$$

<sup>1</sup> The advection (eq. 5.13) can be mapped to the momentum space by multiplying with a  
<sup>2</sup> transformation matrix  $\mathbf{M}$ . The evolutionary equation of LBM–MRT is written as

$$\mathbf{f}(\mathbf{x} + \mathbf{e}_i \Delta_t, t + \Delta_t) - \mathbf{f}(\mathbf{x}, t) = -\mathbf{M}^{-1} \hat{\mathbf{S}}(\hat{\mathbf{f}}(\mathbf{x}, t) - \hat{\mathbf{f}}^{eq}(\mathbf{x}, t)), \quad (5.15)$$

<sup>4</sup> where  $\mathbf{M}$  is the transformation matrix mapping a vector  $\mathbf{f}$  in the discrete velocity space  $\mathbb{V} = \mathbb{R}^b$   
<sup>5</sup> to a vector  $\hat{\mathbf{f}}$  in the moment space  $\mathbb{V} = \mathbb{R}^b$ .

$$\hat{\mathbf{f}} = \mathbf{M}\mathbf{f}, \quad (5.16)$$

$$\mathbf{f}(\mathbf{x}, t) = [f_0(\mathbf{x}, t), f_1(\mathbf{x}, t), \dots, f_8(\mathbf{x}, t)]^T. \quad (5.17)$$

<sup>9</sup> The collision matrix  $\hat{\mathbf{S}} = \mathbf{M}\mathbf{S}\mathbf{M}^{-1}$  in moment space is a diagonal matrix:

$$\hat{\mathbf{S}} = \text{diag}[s_1, s_2, s_3, \dots, s_9].$$

<sup>11</sup> The transformation matrix  $\mathbf{M}$  can be constructed via Gram-Schmidt orthogonalisation procedure.  
<sup>12</sup> The general form of the transformation matrix  $\mathbf{M}$  can be written as

$$\mathbf{M} = [|p\rangle, |e\rangle, |e^2\rangle, |u_x\rangle, |q_x\rangle, |u_y\rangle, |q_y\rangle, |p_{xx}\rangle, |p_{xy}\rangle]^T, \quad (5.18)$$

<sup>14</sup> whose elements are,

$$|p\rangle = |e_\alpha|^0 \quad (5.19a)$$

$$|e\rangle_\alpha = Qe_\alpha^2 - b_2 \quad (5.19b)$$

$$|e^2\rangle_\alpha = a_1(Qe_\alpha^4 - b_6) + a_2(Qe_\alpha^4 - b_6) \quad (5.19c)$$

$$|u_x\rangle_\alpha = e_{\alpha,x} \quad (5.19d)$$

$$|q_x\rangle_\alpha = (b_1 e_\alpha^2 - b_3) e_{\alpha,x} \quad (5.19e)$$

$$|u_y\rangle_\alpha = e_{\alpha,y} \quad (5.19f)$$

$$|q_y\rangle_\alpha = (b_1 e_\alpha^2 - b_3) e_{\alpha,y} \quad (5.19g)$$

$$|p_{xx}\rangle_\alpha = de_{\alpha,x}^2 - e_\alpha^2 \quad (5.19h)$$

$$|p_{xy}\rangle_\alpha = e_{\alpha,x} e_{\alpha,y}, \quad (5.19i)$$

<sup>25</sup> where  $d = 2$  and  $Q = 9$ ,  $b_1 = \sum_{\alpha=1}^Q e_{\alpha,x}^2$ ,  $b_2 = \sum_{\alpha=1}^Q e_\alpha^2$ ,  $b_3 = \sum_{\alpha=1}^Q e_\alpha^2 e_{\alpha,x}^4$ ,  $a_1 = ||e^2||^2$ , and  
<sup>26</sup>  $a_2 = \sum_{\alpha=0}^{Q-1} (Qc_\alpha^2 - b_2) \times (Qc_\alpha^4 - b_6)$ .

## 5.1 Fluid simulation using lattice Boltzmann method

27

Explicitly, the transformation matrix can be written as

$$\mathbf{M} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ -4 & -1 & -1 & -1 & -1 & 2 & 2 & 2 & 2 \\ 4 & -2 & -2 & -2 & -2 & 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & -1 & 0 & 1 & -1 & -1 & 1 \\ 0 & -2 & 0 & 2 & 0 & 1 & -1 & -1 & 1 \\ 0 & 0 & 1 & 0 & -1 & 1 & 1 & -1 & -1 \\ 0 & 0 & -2 & 0 & 2 & 1 & 1 & -1 & -1 \\ 0 & 1 & -1 & 1 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{bmatrix}. \quad (5.20)$$

The corresponding equilibrium distribution functions in moment space  $\widehat{\mathbf{f}}^{eq}$  is given as

$$\widehat{\mathbf{f}}^{eq} = [\rho_0, e^{eq}, e^{2eq}, u_x, q_x^{eq}, q_y^{eq}, p_{xx}^{eq}, p_{xy}^{eq}]^T, \quad (5.21)$$

where

$$e^{eq} = \frac{1}{4}\alpha_2 p + \frac{1}{6}\gamma_2(u_x^2 + y_y^2) \quad (5.22a)$$

$$e^{2eq} = \frac{1}{4}\alpha_3 p + \frac{1}{6}\gamma_4(u_x^2 + y_y^2) \quad (5.22b)$$

$$q_x^{eq} = \frac{1}{2}c_1 u_x \quad (5.22c)$$

$$q_y^{eq} = \frac{1}{2}c_2 u_y \quad (5.22d)$$

$$p_{xx}^{eq} = \frac{3}{2}\gamma_1(u_x^2 - u_y^2) \quad (5.22e)$$

$$p_{xy}^{eq} = \frac{3}{2}\gamma_3(u_x u_y). \quad (5.22f)$$

To get the correct hydrodynamic equation, the values of the co-efficients are chosen as  $\alpha_2 = 24$ ,  $\alpha_3 = -36$ ,  $c_1 = c_2 = -2$ ,  $\gamma_1 = \gamma_3 = 2/3$ ,  $\gamma_2 = 18$  and  $\gamma_4 = -18$ . The values of the elements in the collision matrix are:  $s_8 = s_9 = \tau$  and  $s_1 = s_4 = s_6 = 1.0$  and the others vary between 1.0 and 2.0 for linear stability. Through the Chapman-Enskog expansion (Du et al., 2006), the incompressible Navier-Stokes equation can be recovered and the viscosity is given as

$$\nu = c_s^2 \Delta t (\tau - 0.5). \quad (5.23)$$

**<sup>1</sup> 5.1.3 Boundary conditions**

<sup>2</sup> Boundary conditions (BC) form an important part of any numerical technique. In many cases,  
<sup>3</sup> the boundary conditions can strongly influence the accuracy of the algorithm. Velocity and  
<sup>4</sup> pressure are not the primary variables in LBM, hence the standard pressure, velocity, and  
<sup>5</sup> mixed boundary conditions cannot be imposed directly. Alternative conditions in terms of the  
<sup>6</sup> distribution functions are adopted to describe the boundary conditions.

**<sup>7</sup> Periodic boundary condition**

<sup>8</sup> The simplest type of boundary condition is the periodic boundary. In this case, the domain is  
<sup>9</sup> folded along the direction of the periodic boundary pair. For boundary nodes, the neighbouring  
<sup>10</sup> nodes are on the opposite boundary, using the normal referencing of neighbours (see figure 5.1a).  
<sup>11</sup> From the perspective of submarine landslide modelling, the periodic boundary conditions are  
<sup>12</sup> useful for preliminary analysis, as they imply a higher degree of symmetry of the fluid domain.  
<sup>13</sup> Further information on the periodic boundary condition can be found in [Aidun et al. \(1998\)](#).

**<sup>14</sup> No-slip boundary condition**

<sup>15</sup> The most commonly adopted BC in the lattice Boltzmann approach is the no-slip BC, especially  
<sup>16</sup> the simple bounce-back rule, which is quite elegant and surprisingly accurate. The basic idea is  
<sup>17</sup> that the incoming distribution functions at a wall node are reflected back to the original fluid  
<sup>18</sup> nodes, but with the direction rotated by  $\pi$  radians. The bounce-back boundary condition is one  
<sup>19</sup> of the benefits of LBM, as it is trivial to implement and it allows one to effortlessly introduce  
<sup>20</sup> obstacles into the fluid domain. However, the boundary conditions have been proven to be only  
<sup>21</sup> first-order accurate in time and space ([Pan et al., 2006](#)). A straightforward improvement is to  
<sup>22</sup> consider the wall-fluid interface to be situated halfway between the wall and the fluid lattice  
<sup>23</sup> nodes ([Ziegler, 1993](#)). It involves defining the *solid* nodes as those lying within the stationary  
<sup>24</sup> wall regions, and the *fluid* nodes otherwise. Then, if  $i$  is the direction between a fluid node  $n_1$   
<sup>25</sup> and a solid node  $n_2$ , the bounce-back rule requires that the incoming fluid particle from  $n_1$  to  
<sup>26</sup>  $n_2$  be reflected back along the direction it came from, i.e.,

$$\text{27} \quad f_{-i}(\mathbf{x}, t + \Delta t) = f_i(\mathbf{x}, t_+), \quad (5.24)$$

<sup>28</sup> where  $-i$  denotes the opposite direction of  $i$ . The bounce back rule is illustrated in figure 5.4.  
<sup>29</sup> This simple rule ensures that no tangential velocity exists along the fluid-wall interface, thereby  
<sup>30</sup> a non-slip condition is imposed, and can be extended to any shapes or objects in a fluid flow ([Han](#)  
<sup>31</sup> [et al., 2007a; Zou and He, 1997](#)). The slip boundary conditions have similar treatment to the

## 5.1 Fluid simulation using lattice Boltzmann method

29

non-slip condition, except that the distribution functions are reflected in the boundary instead of bounce-back ([Succi, 2001](#)). 1  
2

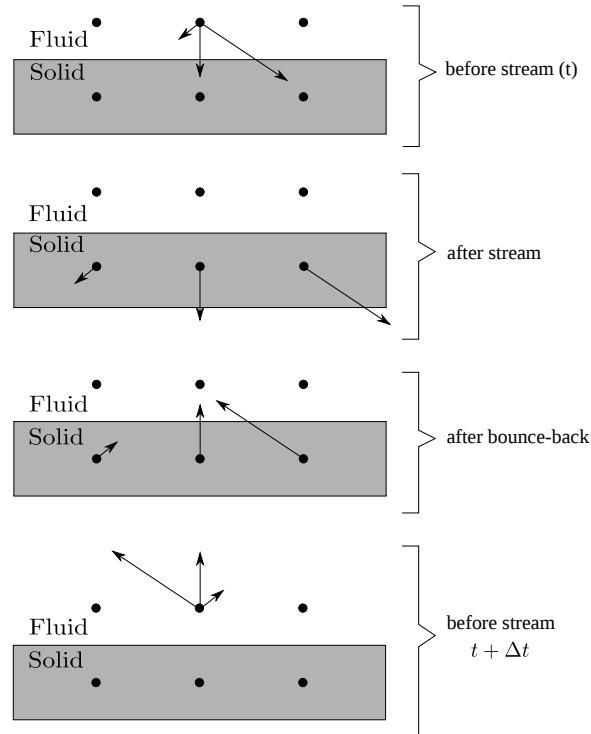


Figure 5.4 Half-way bounce back algorithm for the *D2Q9* model adopted after [Sukop and Thorne \(2006\)](#).

## Pressure and velocity boundary condition

The pressure (Dirichlet) boundary condition can be imposed in lattice Boltzmann by specifying a fluid density at the pressure boundary ([Zou and He, 1997](#)). To impose a pressure boundary along the y-direction (for example, consider the left hand side inlet boundary in figure 5.5), a density  $\rho = \rho_{in}$  is specified from which the velocity is computed. The vertical component of the velocity on the boundary is set as zero,  $u_y = 0$ . After streaming,  $f_2, f_3, f_4, f_6$ , and  $f_7$  are known,  $u_x$  and  $f_1, f_5, f_8$  are to be determined from [eq. 5.2](#) as 3  
4  
5  
6  
7  
8  
9

$$f_1 + f_5 + f_8 = \rho_{in} - (f_0 + f_2 + f_3 + f_4 + f_6 + f_7) \quad (5.25) \quad 10$$

$$f_1 + f_5 + f_8 = \rho_{in} u_x + (f_3 + f_6 + f_7) \quad (5.26) \quad 11$$

$$f_5 - f_8 = f_2 - f_4 + f_6 - f_7, \quad (5.27) \quad 12$$

<sup>1</sup> Consistency of equations (5.25) and (5.26) gives

$$\text{2} \quad u_x = 1 - \frac{[f_0 + f_2 + f_4 + 2 * (f_3 + f_6 + f_7)]}{\rho_{in}}. \quad (5.28)$$

<sup>3</sup> The bounce-back rule for the non-equilibrium part of the particle distribution normal to the  
<sup>4</sup> inlet is used to find  $f_1 - f_1^{eq} = f_3 - f_3^{eq}$ . The values of  $f_5$  and  $f_8$  can be obtained from  $f_1$ :

$$\begin{aligned} \text{5} \quad f_1 &= f_3 + \frac{2}{3}\rho_{in}u_x \\ \text{6} \quad f_5 &= f_7 - \frac{1}{2}(f_2 - f_4) + \frac{1}{6}\rho_{in}u_x \\ \text{7} \quad f_8 &= f_6 + \frac{1}{2}(f_2 - f_4) + \frac{1}{6}\rho_{in}u_x. \end{aligned} \quad (5.29)$$

<sup>9</sup> The corner node at inlet needs some special treatment. Considering the bottom node at inlet  
<sup>10</sup> as an example, after streaming,  $f_3, f_4, f_7$  are known;  $\rho$  is defined, and  $u_x = u_y = 0$ . The particle  
<sup>11</sup> distribution functions  $f_1, f_2, f_5, f_6$ , and  $f_8$  are to be determined. The bounce-back rule for the  
<sup>12</sup> non-equilibrium part of the particle distribution normal to the inlet and the boundary is used to  
<sup>13</sup> find

$$\text{14} \quad f_1 = f_3 + (f_1^{eq} - f_3^{eq}) = f_3 \quad (5.30)$$

$$\text{15} \quad f_2 = f_4 + (f_1^{eq} - f_3^{eq}) = f_4. \quad (5.31)$$

<sup>17</sup> Using these we can compute

$$\text{18} \quad f_5 = f_7 \quad (5.32)$$

$$\text{19} \quad f_6 = f_8 = \frac{1}{2}[\rho_{in} - (f_1 + f_2 + f_3 + f_4 + f_5 + f_6 + f_7 + f_8)]. \quad (5.33)$$

<sup>21</sup> Similar procedure can be applied to the top inlet node and the outlet nodes. Von Neumann  
<sup>22</sup> boundary conditions constrain the flux at the boundaries. A velocity vector  $u = [u_0 \ v_0]^T$  is  
<sup>23</sup> specified, from which the density and pressure are computed based on the domain. The velocity  
<sup>24</sup> boundary condition can be specified in a similar way (Zou and He, 1997). The pressure  
<sup>25</sup> and velocity boundary conditions contribute additional equation(s) to determine the unknown  
<sup>26</sup> distribution functions. In the case of velocity boundary, the boundary condition equation is  
<sup>27</sup> sufficient to determine the unknown distribution functions in the D2Q9 model, however the  
<sup>28</sup> pressure boundary conditions require additional constitutive laws to determine the unknown  
<sup>29</sup> distribution functions.

Table 5.1 LBM parameters used in simulating laminar flow through a circular pipe.

Parameter	Value
Density $\rho$	1000 kg m <sup>-3</sup>
Relaxation parameter $\tau$	0.51
Kinematic viscosity	$1 \times 10^{-6}$ m <sup>2</sup> s <sup>-1</sup>
Grid resolution ‘h’	1 <sup>-2</sup> m
Number of steps	50,000
Error in predicting horizontal velocity	0.009 %

## 5.2 Validation of the lattice Boltzmann method

To verify the incompressible LBM model implemented in the above section, numerical simulation of a transient development of steady state Poiseuille flow in a straight channel is performed. At  $t = 0$ , the LBM water particles ( $\rho = 1000\text{kg/m}^3$ ) are simulated to flow through a channel of width ‘H’ (= 0.4 m) and simulation length ‘L’ (2.5H) under constant body force. Periodic boundary conditions are applied at either end of the channel and the pressure gradient is set to zero, which simulates the condition of a continuous flow of fluid in a closed circular pipe. The length ‘L’ has no effect on the simulation as no streamwise variation is detected in the solution. The parameters adopted in LBM simulation are presented in table 5.1. Sufficient time is allowed for the flow to travel beyond the required development length so that the flow is laminar (Durst et al., 2005). The development length  $X_D$  required for a flow to be fully laminar is

$$X_D/H = [(0.619)^{1.6} + (0.0567R_e)^{1.6}]^{1/1.6}, \quad (5.34)$$

where  $R_e$  is the Reynolds number. The velocity profile at steady state is presented in figure 5.5. A maximum horizontal velocity of 0.037863 m s<sup>-1</sup> is observed along the center-line of the channel. The maximum horizontal velocity is compared with the closed-form based on the Haygen-Poiseuille flow equation for no-slip boundary condition (Willis et al., 2008)

$$U_x = \frac{\Delta P}{2\mu L} \left[ \frac{H^2}{4} - y^2 \right], \quad (5.35)$$

where  $v_x$  is the horizontal velocity (m/s);  $\Delta P$  is the pressure gradient,  $\mu$  dynamic viscosity of the fluid. LBM predicts the maximum horizontal velocity within an error of 0.009 %.

In order to further validate the accuracy of the lattice Boltzmann code, the transient development of the Poiseuille’s flow is compared with the CFD simulation performed using ANSYS Fluent. Finite Volume Method is a common CFD technique, which involves solving

- <sup>1</sup> the governing partial differential equation (Navier-Stokes) over the discretised control volume.  
<sup>2</sup> This guarantees the conservation of fluxes over a particular control volume. The finite volume  
<sup>3</sup> equations yield governing equations of the form

<sup>4</sup>

$$\frac{\partial}{\partial t} \int \int \int Q dV + \int \int F dA = 0, \quad (5.36)$$

<sup>5</sup>

- <sup>6</sup> where  $Q$  is the vector of conserved variables,  $F$  is the vector of fluxes in the Navier-Stokes  
<sup>7</sup> equation,  $V$  is the volume of control volume element, and  $A$  is the surface area of the control  
<sup>8</sup> volume element.

<sup>9</sup> A 2D rectangular plane of length 1 m and height 0.04 m is discretised into 400 cells  
<sup>10</sup> of size  $1^{-2}$  m (see figure 5.6). A constant velocity is applied at the inlet. Water ( $\rho =$   
<sup>11</sup>  $998.2 \text{ kg/m}^3$ , viscosity ' $\eta'$  =  $1 \times 10^{-3} \text{ Ns/m}^2$ ) is allowed to flow through the channel and  
<sup>12</sup> it develops into a fully laminar flow. The least squares approach was adopted to solve the  
<sup>13</sup> gradient, and a maximum of 100 iteration steps were carried out until the solution converged.

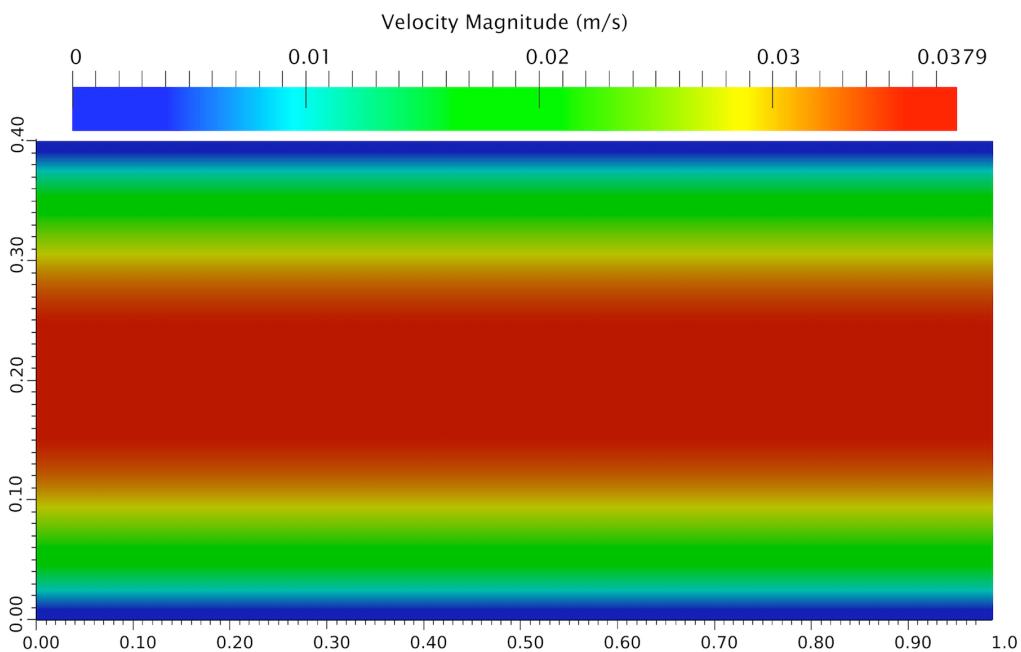


Figure 5.5 Velocity profile: LBM Simulation of a laminar flow through a channel.

<sup>14</sup> The velocity profile obtained from the CFD simulation at cross-section ‘L/4’ is shown  
<sup>15</sup> in figure 5.7. Figure 5.8 compares the development of computed velocity profiles with the  
<sup>16</sup> analytical solution. At normalised time  $t = 1$ , the flow approaches steady state. It can be  
<sup>17</sup> observed that LBM has excellent agreement with CFD and the analytical solution at various  
<sup>18</sup> stages of flow evolution.

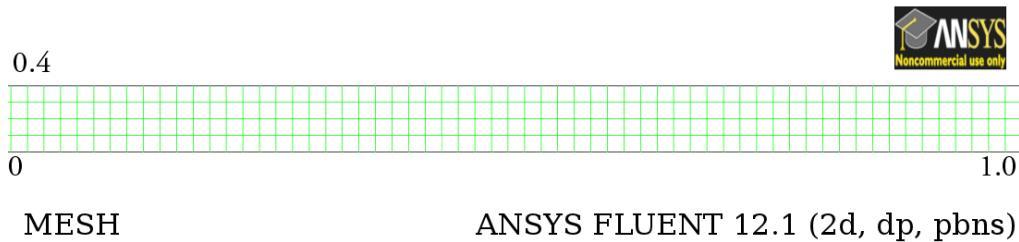


Figure 5.6 Finite Volume mesh used in the CFD analysis of laminar flow through a channel.

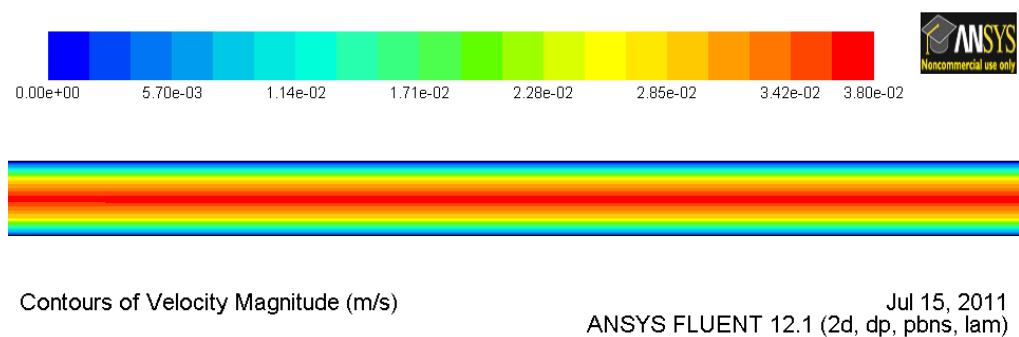


Figure 5.7 Velocity profile: CFD analysis of laminar flow through a channel.

In order to study the capability of the lattice Boltzmann technique to simulate fluid–solid interaction, LB simulation of a fluid flow around a rectangular obstacle is compared with the CFD technique. A solid wall of height ‘H/2’ is placed at length ‘L/4’ in the channel. Bounce-back algorithm is employed to model the fluid-wall interaction in LBM. In the CFD model, the control volume is discretised into 10,000 cells. A constant velocity is applied in the inlet and the horizontal velocity profile is recorded. Both, CFD and LBM simulations were performed to study the influence of a solid wall on the fluid flow behaviour.

The horizontal velocity profile obtained after 50,000 LBM iterations is presented in figure 5.9. LBM is able to capture the velocity shedding around the edges of the wall. The velocity profile obtained from the CFD analysis is presented in figure 5.10. The horizontal velocity profile at ‘L/4’ at  $t = 1$  is shown in figure 5.11. The maximum horizontal velocity from the CFD analysis is 0.3% higher in comparison with the LBM simulation. The discrepancy in the horizontal velocity profile (figure 5.11) can be attributed to the relaxation parameter used in the LBM, which is obtained by a trial and error procedure. The velocity profile obtained from the LBM simulation compares qualitatively with the FE analysis performed by [Zhong and Olson \(1991\)](#). Thus, it can be concluded that the lattice Boltzmann method is a suitable form of numerical representation of the Navier-Stokes equation to model fluid – solid interactions.

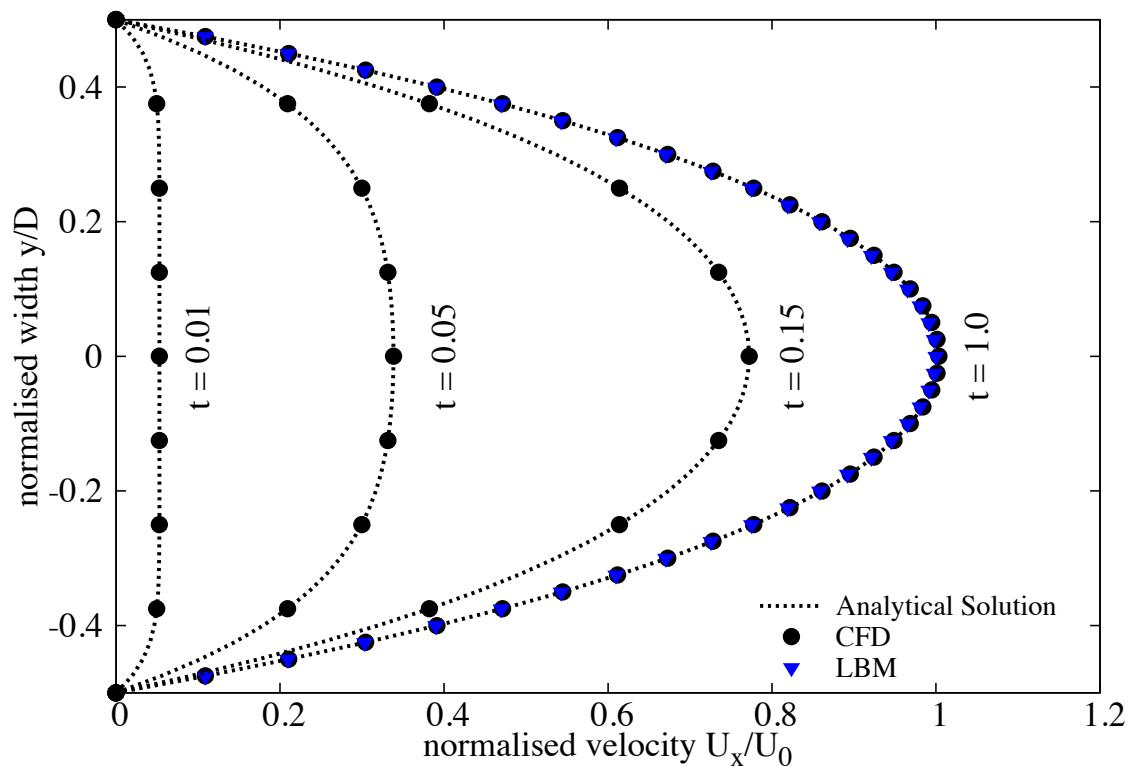


Figure 5.8 Development of the Poiseuille velocity profile in time: comparison between LBM simulation, CFD simulation and the analytical solution. Time is made dimensionless by  $H/U_0$ .

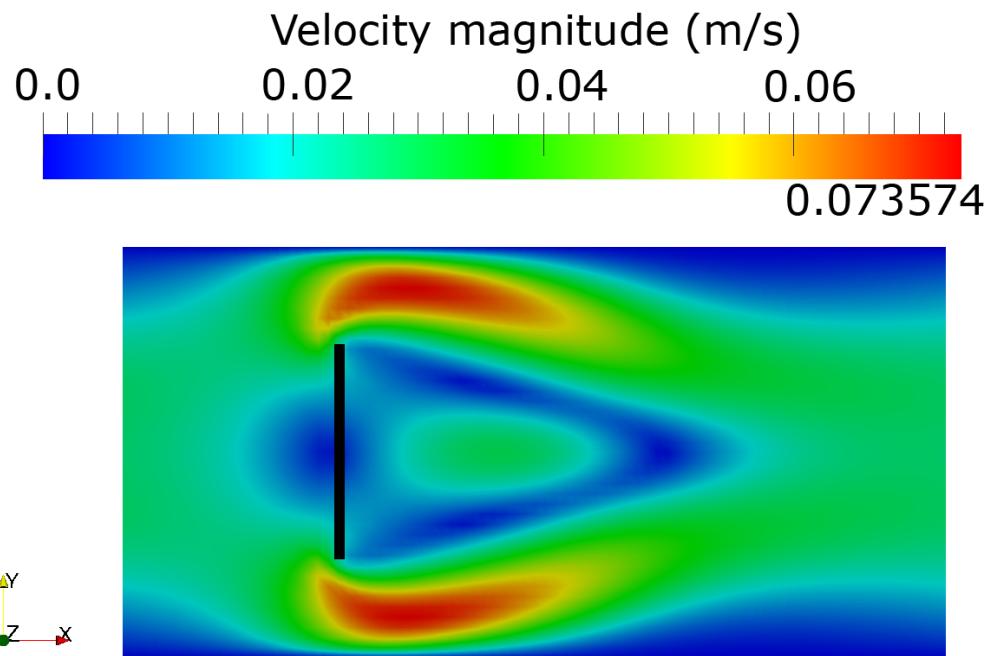


Figure 5.9 LBM simulation of velocity profile for a laminar flow through a pipe with an obstacle at L/4.

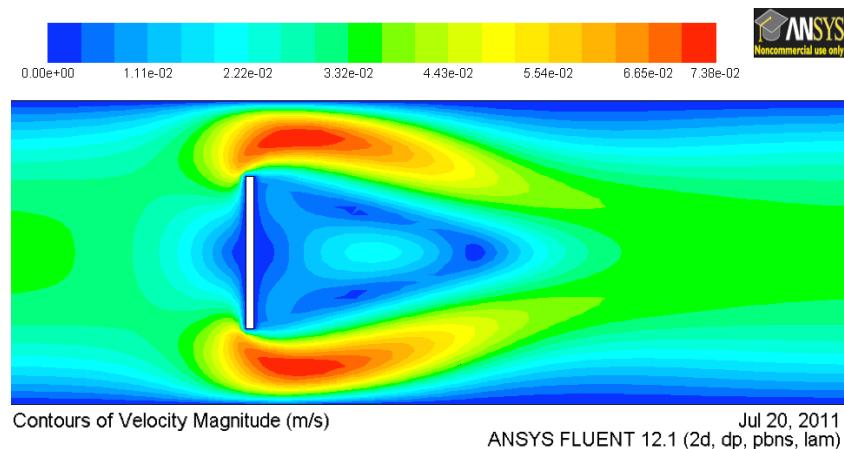


Figure 5.10 CFD simulation of velocity contour for a laminar flow through a pipe with an obstacle at L/4.

### 5.3 Turbulence in lattice Boltzmann method

The above formulation of lattice Boltzmann has been successfully applied to many fluid flow problems, however it is restricted to flows with low Reynolds number. Modelling fluids with low viscosity like water and air remains a challenge, necessitating very small values of  $h$ , and/or  $\tau$  very close to 0.5 (He et al., 1997). The standard lattice Boltzmann can deal with laminar

1

2

3

4

5

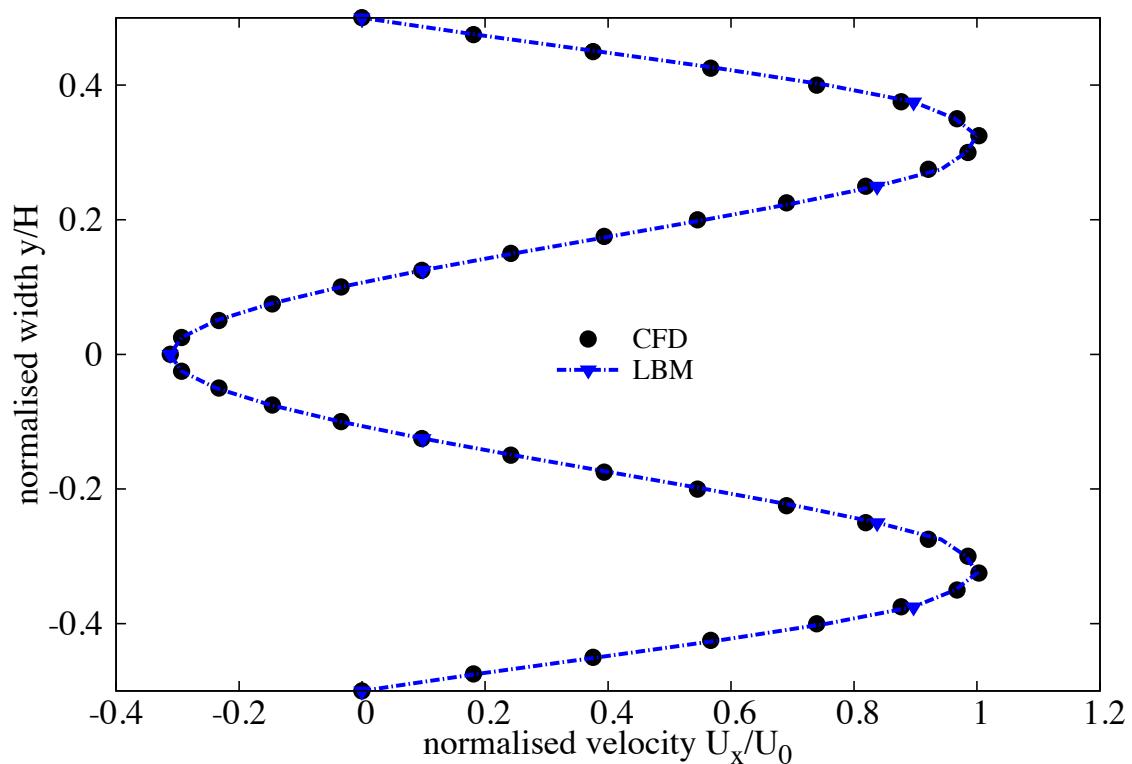


Figure 5.11 LBM and CFD simulation of a flow around an obstacle.

flows, while practical problems with small kinematic viscosity are often associated with flows having large Reynolds numbers, i.e. flows which are unsteady or turbulent in nature. The turbulent flows are characterised by the occurrence of eddies with multiple scales in space, time and energy.

The large eddy simulation (LES) is the most widely adopted approach to solve turbulent flow problems. It directly solves the large scale eddies, which carry the predominant portion of the energy, and the smaller eddies are modelled using a sub-grid approach. The separation of scales is achieved by filtering of the Navier-Stokes equations, from which the resolved scales are directly obtained. The unresolved scales are modelled by a one-parameter Smagorinski sub-grid methodology, which assumes that the Reynolds stress tensor is dependent only on the local strain rate (Smagorinsky, 1963). It involves parametrising the turbulent energy dissipation in the flows, where the larger eddies extract energy from the mean flow and ultimately transfer some of it to the smaller eddies which, in turn, pass the energy to even smaller eddies, and so on up to the smallest scales. At the smallest scale, the eddies convert the kinetic energy into the internal energy of the fluid. At this scale, the viscous friction dominates the flow (Frisch and Kolmogorov, 1995).

In the Smagorinsky model, the turbulent viscosity  $\nu$  is related to the strain rate  $S_{ij}$  and a filtered length scale ‘ $h$ ’ as follows

$$S_{ij} = \frac{1}{2}(\partial_i u_j + \partial_j u_i) \quad (5.37)$$

$$\nu_t = (S_c h)^2 \bar{S} \quad (5.38)$$

$$\bar{S} = \sqrt{\sum_{i,j} \tilde{S}_{i,j} \tilde{S}_{i,j}}, \quad (5.39)$$

where  $S_c$  is the Smagorinsky constant, which is close to 0.03 (Yu et al., 2005). The effect of the unresolved scale motion is taken into account by introducing an effective collision relaxation time scale  $\tau_t$ , so that the total relaxation time  $\tau_*$  is written as

$$\tau_* = \tau + \tau_t, \quad (5.40)$$

where  $\tau$  and  $\tau_t$  are respectively the standard relaxation times corresponding to the true fluid viscosity  $\nu$  and the turbulence viscosity  $\nu_t$ , defined by a sub-grid turbulence model. The new

1 viscosity  $\nu_*$  corresponding to  $\tau_*$  is defined as

$$2 \quad \nu_* = \nu + \nu_t \\ 3 \quad = \frac{1}{3}(\tau_* - \frac{1}{2})C^2\Delta t = \frac{1}{3}(\tau + \tau_t - \frac{1}{2})C^2\Delta t \quad (5.41)$$

$$4 \quad \nu_t = \frac{1}{3}\tau_t C^2\Delta t. \quad (5.42) \\ 5$$

6 The Smagorinski model is easy to implement and the lattice Boltzmann formulation remains  
 7 unchanged, except for the use of a new turbulence-related viscosity  $\tau_*$ . The component  $s_1$  of  
 8 the collision matrix becomes  $s_1 = \frac{1}{\tau + \tau_t}$ .

9 The effectiveness of LBM-LES model in simulating unsteady flows is verified by modelling  
 10 the Kármán vortex street. In fluid dynamics, a Kármán vortex street is a repeating pattern of  
 11 vortices caused by unsteady separation of fluid flow around circular obstacles. A vortex street  
 12 will only be observed above a limiting value of Reynolds number of 90. The Reynolds number  
 13 is computed based on the cylinder diameter ‘D’ and the mean flow velocity  $U$  of the parabolic  
 14 inflow profile:

$$15 \quad Re = \frac{UD}{\nu}. \quad (5.43)$$

16 LBM particles are simulated to flow through a 2D rectangular channel with an aspect ratio  
 17 ‘L/H’ of 2.5. A cylinder of diameter ‘d’ = 0.27H is placed at H/2. The pressure gradient at  
 18 the inlet and the outlet is varied to create flows with different mean velocities. Numerical  
 19 simulations of vortex shedding behind a circular obstacle are carried out for three different  
 20 fluid flow regimes (Reynolds number of 55, 75, and 112). The fully developed fluid flows for  
 21 different Reynolds numbers are shown in figure 5.12. It can be observed from figure 5.12 that  
 22 the von Kármán vortex street can only be observed at high a Reynolds number of 112 ( $Re >$   
 23 90), which shows the ability of the LBM turbulence model to capture instabilities in fluid flow.

24 One important quantity taken into account in the present analysis is the Strouhal number  $St$ ,  
 25 a dimensionless number describing oscillating unsteady flow dynamics. The Strouhal number  
 26 is computed from the cylinder diameter D, the measured frequency of the vortex shedding f,  
 27 and the maximum velocity  $U_{max}$  at the inflow plane:

$$28 \quad St = \frac{fD}{U_{max}}. \quad (5.44)$$

29 The characteristic frequency  $f$  is determined by a spectral analysis (Fast Fourier Transform -  
 30 FFT) of time series of the fluid pressure. Table 5.2 shows that the Strouhal numbers computed  
 31 from LBM simulations have a very good agreement with FVM results obtained by Breuer et al.  
 32 (2000). This shows the ability of LBM-LES in capturing unsteady flow dynamics.

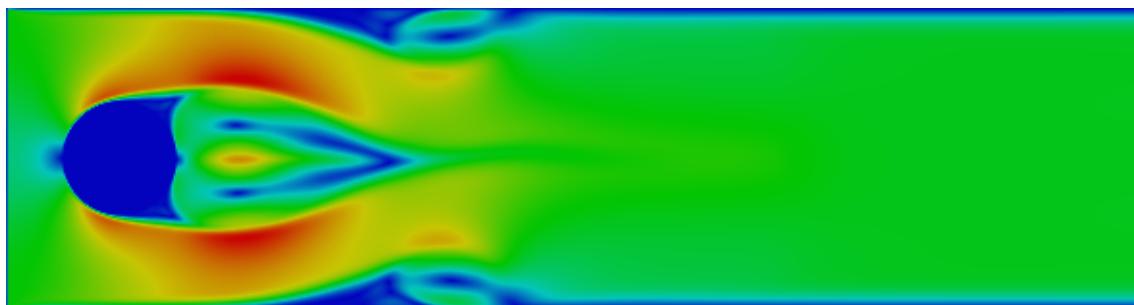
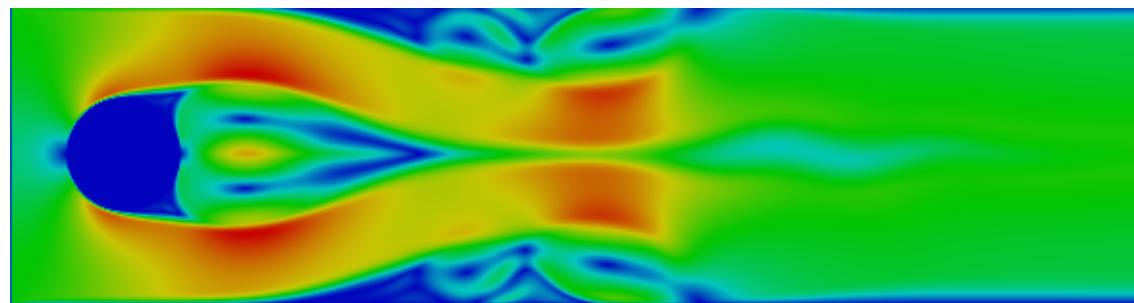
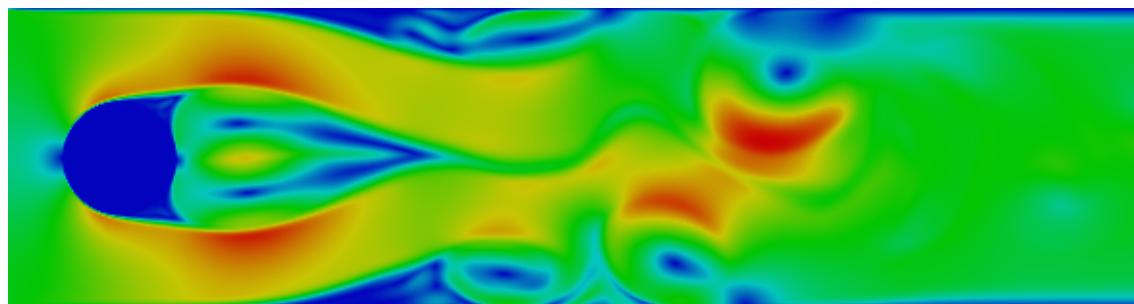
(a)  $\text{Re} = 55$ (b)  $\text{Re} = 75$ (c)  $\text{Re} = 112$ 

Figure 5.12 Kármán vortex street

Table 5.2 Computed Strouhal number for fluid flows with different Reynolds number

Reynolds number	Strouhal number	
	LBM	FVM
55	0.117	0.117
75	0.128	0.129
112	0.141	0.141

\* FVM results are from [Breuer et al. \(2000\)](#)

## 5.4 Coupled LBM and DEM for fluid-grain interactions

Modelling fluid–grain interactions in submarine landslides requires the ability to simulate the interactions at the dynamic fluid – solid boundaries. In principle, the conventional FE and FVM based approaches for solving the Navier-Stokes equations with moving boundaries and/or structural interaction ([Bathe and Zhang, 2004](#)) can be applied to particle fluid interaction problems. The common feature of these approaches is to model the interaction between the fluid and the solid to a high degree of accuracy. However, the main computational challenge is the need to continuously generate new geometrically adapted meshes to circumvent severe mesh distortion, which is computationally very intensive ([Han et al., 2007b](#)).

The lattice Boltzmann approach has the advantage of accommodating large particle sizes and the interaction between the fluid and the moving grains can be modelled through relatively simple fluid - grain interface treatments. Further, employing DEM to account for the grain/grain interaction naturally leads to a combined LB – DEM solution procedure. The Eulerian nature of the lattice Boltzmann formulation, together with the common explicit time step scheme of both LBM and DEM makes this coupling strategy an efficient numerical procedure for the simulation of fluid – grain systems.

LBM – DEM technique is a powerful predictive tool for gaining insights into many fundamental physical phenomena in fluid-solid systems. Such a coupled methodology was first proposed by ([Cook et al., 2004](#)) for simulating fluid-grain systems dominated by fluid-grain and grain-grain interactions. To capture the actual physical behaviour of the fluid-grain system, it is essential to model the boundary condition between the fluid and the grain as a non-slip boundary condition, i.e. the fluid velocity near the grain should be similar to the velocity of the grain boundary. The soil grains in the fluid domain are represented by lattice nodes. The discrete nature of the lattice will result in stepwise representations of the surfaces, which are otherwise circular, this is neither accurate nor smooth, unless sufficiently small lattice spacing is adopted.

## Modified bounce back rule

To accommodate the movement of solid particles in the commonly adopted bounce-back rule (see section 5.1.3), Ladd (1994) modified the ‘no-slip’ rule for a given boundary link  $i$  to be

$$f_i(\mathbf{x}, t + \Delta t) = f_i(\mathbf{x}, t_+) - \alpha_i e_i \cdot v_b \quad (\alpha_i = 6w_i \rho / C_s^2), \quad (5.45)$$

where  $f_i(\mathbf{x}, t_+)$  is the post collision distribution at the fluid or solid boundary node  $\mathbf{x}$ , and  $v_b$  is the velocity at the nominal boundary point at the middle of the boundary link  $i$

$$\mathbf{v}_b = \mathbf{v}_c + \boldsymbol{\omega} \times (\mathbf{x} + e_i \Delta t / 2 - \mathbf{x}_c), \quad (5.46)$$

in which  $v_c$  and  $\boldsymbol{\omega}$  are the translational and angular velocities at the mass centre of the solid particle, respectively.  $\mathbf{x}_c$  and  $\mathbf{x} + e_i \Delta t / 2$  are the coordinates of the centre and the nominal boundary point, respectively. The impact force on the soil grain from the link is defined as

$$\mathbf{F}_i = 2[f_i(\mathbf{x}, t_+) - \alpha_i e_i \cdot v_b] / \Delta t. \quad (5.47)$$

The corresponding torque  $\mathbf{T}_i$ , produced by the force with respect to the centre of the particle is computed as

$$\mathbf{T}_i = \mathbf{r}_c \times \mathbf{F}_i (\mathbf{r}_c = \mathbf{x} + \mathbf{e}_i \Delta t / 2 - \mathbf{x}_c). \quad (5.48)$$

Then the total hydrodynamic force and torque exerted on the particle can be calculated by summing up the forces and torques from all the related boundary links:

$$\begin{aligned} \mathbf{F} &= \sum_i \mathbf{F}_i \\ \mathbf{T} &= \sum_i \mathbf{T}_i. \end{aligned} \quad (5.49)$$

Ladd and Verberg (2001) described a methodology that minimises the oscillations resulting from soil grains crossing lattices at a very high speed. The methodology involves combining several extensions for the fluid simulation like the treatment of moving curved boundaries with the scheme of Yu et al. (2003) and a fluid/grain force interaction method with the momentum exchange method of Ladd and Verberg (2001). The simulation of the moving curved grain surfaces results in the intersection of links between two nodes at arbitrary distances (Iglberger et al., 2008). These distance values are referred to as delta values:

$$\delta = \frac{\text{Distance between fluid node and soil surface}}{\text{Distance between fluid node and soil node}} \in [0, 1]. \quad (5.50)$$

1 For each pair of a fluid and grain node, a delta value has to be calculated. Delta values of zero  
 2 are not possible as the nodes on the surface are considered as solid nodes. The algorithm for  
 3 computation of the  $\delta$  value is presented in [Iglberger et al. \(2008\)](#). Figure 5.13 shows the three  
 4 possible situations for delta values between 0 and 1. The fluid particles in LBM are always  
 5 considered to be moving at the rate of one lattice per time step ( $\delta\mathbf{x}/\delta t$ ), for delta values smaller  
 6 than 0.5. For  $\delta$  values larger than 0.5, the fluid particles would come to rest at an intermediate  
 7 node  $\mathbf{x}_i$ . In order to calculate the reflected distribution function in node  $\mathbf{x}_f$ , an interpolation  
 8 scheme has to be applied. The linear interpolation scheme of [Yu et al. \(2003\)](#) is used in the  
 9 present study, which uses a single equation, irrespective of the value of  $\delta$  being smaller or  
 10 larger than 0.5, to the reflected distribution function that is computed as

$$\begin{aligned} 11 \quad f_{\bar{\alpha}}(\mathbf{x}_f, t + \delta t) = & \frac{1}{1 + \delta} \cdot [(1 - \delta) \cdot f_{\alpha}(\mathbf{x}_f, t + \delta t) + \delta \cdot f_{\alpha}(\mathbf{x}_b, t + \delta t) \\ 12 \quad & + \delta \cdot f_{\bar{\alpha}}(\mathbf{x}_{f2}, t + \delta t) - 2w_a \rho_w \frac{3}{c^2} e_a \cdot \mathbf{u}_w], \end{aligned} \quad (5.51)$$

14 where  $w_{\alpha}$  is the weighting factor,  $\rho_w$  is the fluid density in node  $\mathbf{x}_f$ , and  $\mathbf{u}_w$  is the velocity at the  
 15 bounce-back wall. In order to couple the fluid-grain interaction, the LBM approach is extended  
 16 by adopting a force integration scheme, to calculate the fluid force acting on the grain surface,  
 17 and the momentum exchanged method described earlier. The physical force acting on grain  
 18 agglomerates is calculated as the sum over all fluid/grain node pairs, resulting in

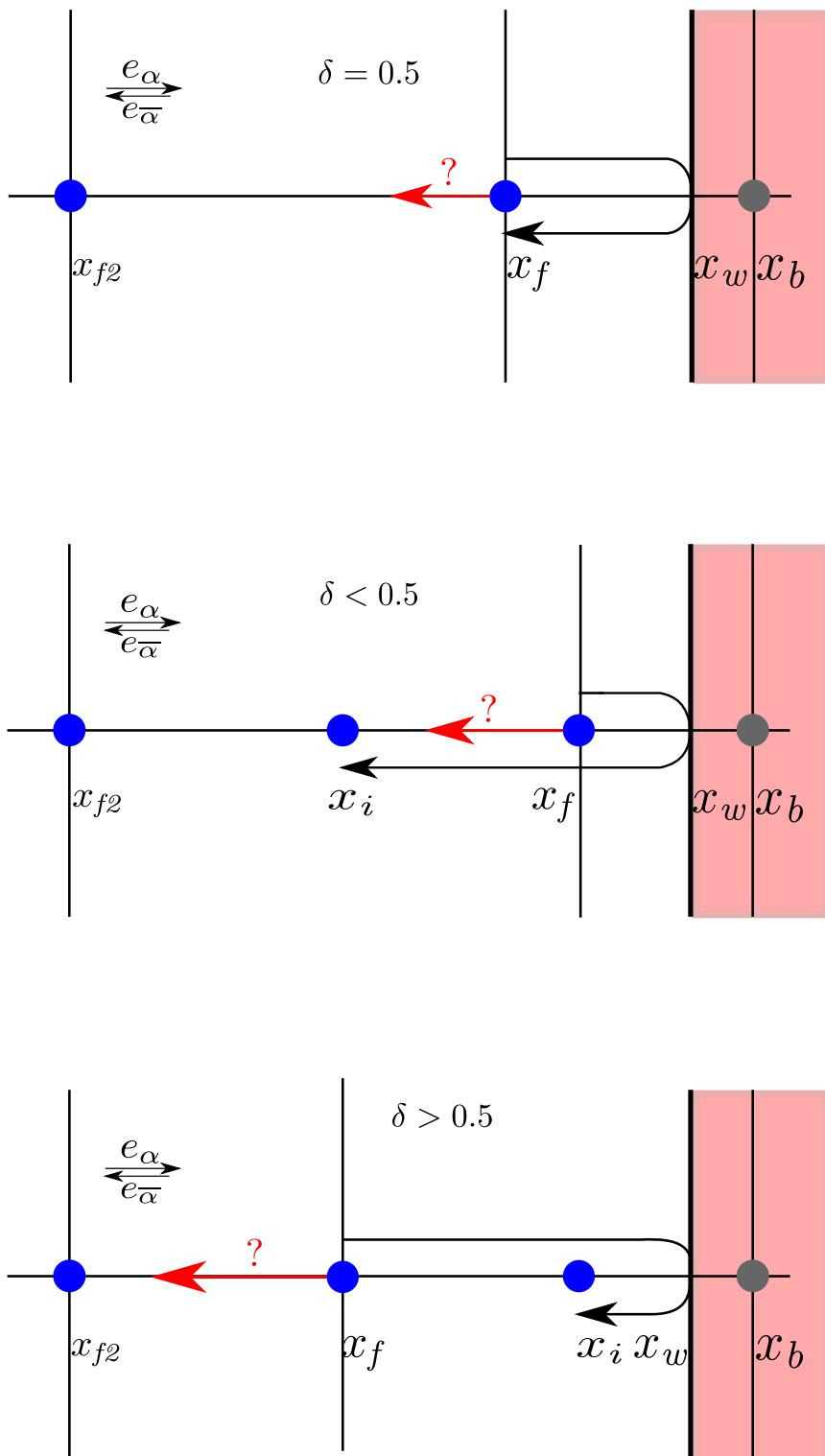
$$19 \quad F = \sum_{\mathbf{x}_b} \sum_{\alpha=1}^{19} \mathbf{e}_{\alpha} [f_{\alpha}(\mathbf{x}_b, t) + f_{\bar{\alpha}}(\mathbf{x}_f, t)] \delta\mathbf{x}/\delta t. \quad (5.52)$$

20 After the force calculations, the coupled rigid body physics can be simulated in order to move  
 21 the grains / grain-agglomerates according to the applied forces. The total hydrodynamic forces  
 22 and torque exerted on a grain can be computed as ([Cook et al., 2004](#); [Noble and Torczynski,](#)  
 23 [1998](#))

$$24 \quad \mathbf{F}_f = Ch \left[ \sum_n \left( \beta_n \sum_i f_i^m e_i \right) \right] \quad (5.53)$$

$$25 \quad \mathbf{T}_f = Ch \left[ \sum_n (\mathbf{x}_n - \mathbf{x}_c) \times \left( \beta_n \sum_i f_i^m e_i \right) \right]. \quad (5.54)$$

27 The summation is over all lattice nodes covered by the soil grain, and  $\mathbf{x}_n$  represents the  
 28 coordinate of the lattice node  $n$ .

Figure 5.13 Bounce back boundaries for different values of  $\delta$

When grains are not in direct contact among themselves, but are driven by the fluid flow and body force, i.e. gravity, their motion can be determined by Newton's equation of motion

$$m\mathbf{a} = \mathbf{F}_f + m\mathbf{g} \quad (5.55)$$

$$J\ddot{\theta} = \mathbf{T}_f, \quad (5.56)$$

where  $m$  and  $J$  are respectively the mass and the moment of inertia of a grain,  $\ddot{\theta}$  is the angular acceleration,  $\mathbf{g}$  is the gravitational acceleration,  $\mathbf{F}_f$  and  $\mathbf{T}_f$  are respectively the hydrodynamic forces and torque. The equation can be solved numerically by an explicit numerical integration, such as the central difference scheme.

The interaction between the soil grains, and the soil grains with the walls are modelled using the DEM technique. To solve the coupled DEM–LBM formulation, the hydrodynamic force exerted on soil grains and the static buoyancy force are considered by reducing the gravitational acceleration to  $(1 - \rho/\rho_s)\mathbf{g}$ , where  $\rho_s$  is the density of the grains. When taking into account all forces acting on an element, the dynamic equations of DEM can be expressed as

$$m\mathbf{a} + c\mathbf{v} = \mathbf{F}_c + \mathbf{F}_f + m\mathbf{g}, \quad (5.57)$$

where  $\mathbf{F}_c$  denotes the total contact forces from other elements and/or the walls, and  $c$  is a damping coefficient. The term  $c\mathbf{v}$  represents a viscous force that accounts for the effect of all possible dissipation forces in the system including energy lost during the collision between grains. Considering a linear contact model

$$\mathbf{F}_c = k_n\delta, \quad (5.58)$$

where  $k_n$  is the normal stiffness and  $\delta$  is the overlap, the critical time step associated with the explicit integration is determined as (He et al., 1997)

$$\Delta t_{cr} = 2(\sqrt{1 + \xi^2} - \xi)/\omega, \quad (5.59)$$

where  $\omega = \sqrt{k_n/m}$  is the local contact natural frequency and  $\xi = c/2m\omega$  is the critical damping ratio. The actual time step used for the integration of the Discrete Element equations is

$$\Delta t_D = \lambda \Delta t_{cr}. \quad (5.60)$$

The time step factor  $\lambda$  is chosen to be around 0.1 to ensure both stability and accuracy (He et al., 1997).

When combining the Discrete Element modelling of the grain interactions with the LB formulation, an issue arises. There are now two time steps:  $\Delta t$  for the fluid flow and  $\Delta t_D$  for the particles. Since  $\Delta t_D$  is normally smaller than  $\Delta t$ ,  $\Delta t_D$  is slightly reduced to a new value  $\Delta t_s$  so that  $\Delta t$  and  $\Delta t_s$  have an integer ratio  $n_s$

$$\Delta t_s = \frac{\Delta t}{n_s} \quad (n_s = [\Delta t / \Delta t_D] + 1). \quad (5.61)$$

This results in a sub-cycling time integration for the Discrete Element part. At every step of the fluid computation,  $n_s$  sub-steps of integration are performed for the Discrete Element Method (5.57) using the time step  $\Delta t_s$ . The hydrodynamic force  $\mathbf{F}_f$  is unchanged during the sub-cycling.

### 5.4.1 Draft, kiss and tumbling: Sedimentation of two grains

In multiphase flows, fundamental mechanisms of fluid – grain and grain – grain interactions are very important for accurately predicting the flow behaviours. The sedimentation of two circular grains in a viscous fluid serves as the simplest problem to study these two types of interactions, and many experimental and numerical studies have been carried out to investigate this behaviour (Komiwas et al., 2005; Wang et al., 2014). Fortes (1987) observed experimentally that in the sedimentation of two grains under gravity in a Newtonian fluid, the two grains would undergo the draft, kiss and tumbling (DKT) phenomenon.

The *draft*: grain 2 is first placed within the hydrodynamic drag above grain 1. As the hydrodynamic drag of grain 1 is a depression zone, grain 2 is attracted inside. The *kiss*: grain 2 increases its vertical velocity until it touches grain 1. The horizontal velocity of grain 1 increases and its vertical velocity decreases below that of grain 2. *Tumbling*: grain 2 having the same horizontal velocity and higher vertical velocity than grain 1, overtakes grain 1.

LBM-DEM simulation of two grains under gravity in a viscous Newtonian fluid reproduces the draft, kiss and tumble effect (see figure 5.14). They are in agreement with the experimental description of the DKT effect. For better understanding of the DKT effect, the time history of three distances between the grains (normalised to the diameter of the grain D) are tracked i.e., the difference in the transverse coordinates  $\delta_x/D$  and longitudinal coordinates  $\delta_y/D$  of the two grain centres, and the gap between the two surfaces  $\delta = \sqrt{\delta_x^2 + \delta_y^2} - 1$  (see figure 5.15c).

As shown in figure 5.14, grain 1 trails grain 2. As grain 2 approaches the depression zone, corresponding to negative fluid pressure behind grain 1, the velocity of the trailing grain increases as the grains approach closer, this is in agreement with the experimental description of the draft. Grain 2 increases its vertical velocity more than grain 1 until it touches grain 1. The kiss happens at a normalised time  $(t / \sqrt{(D/g)}) = 25$ . At this stage, the gap  $\delta$  between the

1 grains is zero, but the actual gap is about one lattice spacing for the LBM collision model. After  
2 this time, the vertical velocity of grain 1 decreases and its horizontal velocity increases as the  
3 grains tumble. At this stage, the grains still remain in contact, i.e., the gap remains unchanged  
4  $\delta = 0$ . Subsequently, the two grains separate and move away from each other. Figure 5.15b  
5 shows that the terminal velocities of the two grains are in good agreement with the terminal  
6 velocity of a single grain found by an independent simulation and calculated using the empirical  
7 Schiller and Nauman formula ([Komiwas et al., 2005](#)).

## 8 5.5 GP-GPU Implementation

9 Graphics Processing Unit (GPU) is a massively multi-threaded architecture that is widely  
10 used for graphical and now non-graphical computations. Today's GPUs are general purpose  
11 processors with support for an accessible programming interface. The main advantage of GPUs  
12 is their ability to perform significantly more floating point operations (FLOPs) per unit time  
13 than a CPU. General Purpose computations on GPUs (GPGPUs) often achieve speedups of  
14 orders of magnitude in comparison with optimised CPU implementations.

15 A GPU consists of several *Streaming Multiprocessors* (SMs). Each SM contains 32 CUDA  
16 processors. Each CUDA processor has a fully pipelined integer arithmetic logic unit (ALU)  
17 and a floating point unit (FPU). The FPU complies with the IEEE 754-2008 industry standard  
18 for floating-point arithmetic, capable of double precision computations. The SM schedules  
19 work in groups of 32 threads called warps. Each SM features two warp schedulers and two  
20 instruction dispatch units, allowing two warps to be issued and executed concurrently. Each  
21 thread has access to both L1 and L2 caches, which improves the performance for programs  
22 with random memory access.

23 The occupancy rate of the SPs, i.e. the ratio between the number of threads run and the  
24 maximum number of executable threads, is an important aspect to take into consideration for  
25 the optimisation of a CUDA kernel. Even though a block may only be run on a single SM, it is  
26 possible to execute several blocks concurrently on the same SM. Hence, tuning the execution  
27 grid layout allows one to increase the occupancy rate. Nevertheless, reaching the maximum  
28 occupancy is usually not possible, as the threads executed in parallel on one SM have to share  
29 the available registers ([Obrecht and Kuznik, 2011](#)).

30 Many-core processors are promising platforms for intrinsically parallel algorithms such as  
31 the lattice Boltzmann method. Since the global memory for GPU devices shows high latency  
32 and LBM is data intensive, the memory access pattern is an important issue for achieving good  
33 performances. Whenever possible, global memory loads and stores should be coalescent and  
34 aligned, but the propagation phase in LBM can lead to frequent misaligned memory accesses.

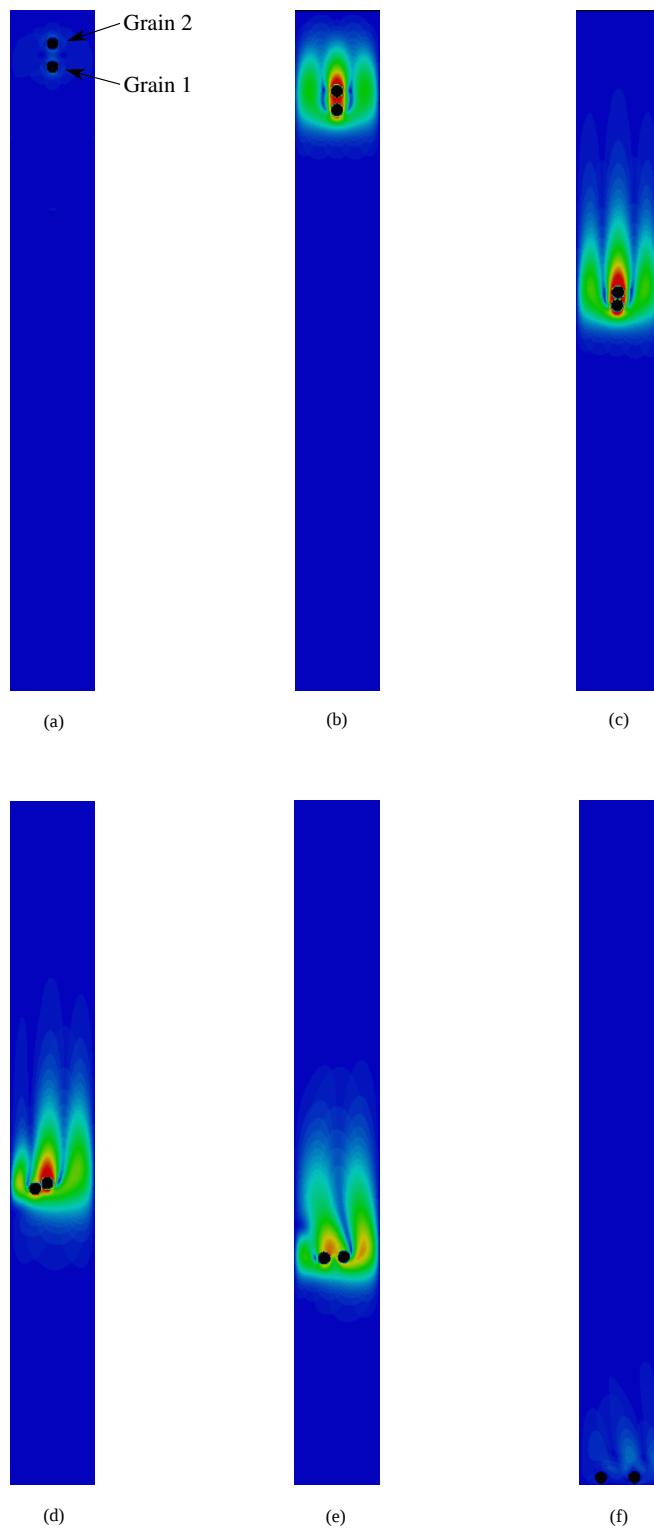


Figure 5.14 Time series of draft, kiss and tumble of two grains during sedimentation in a viscous fluid.

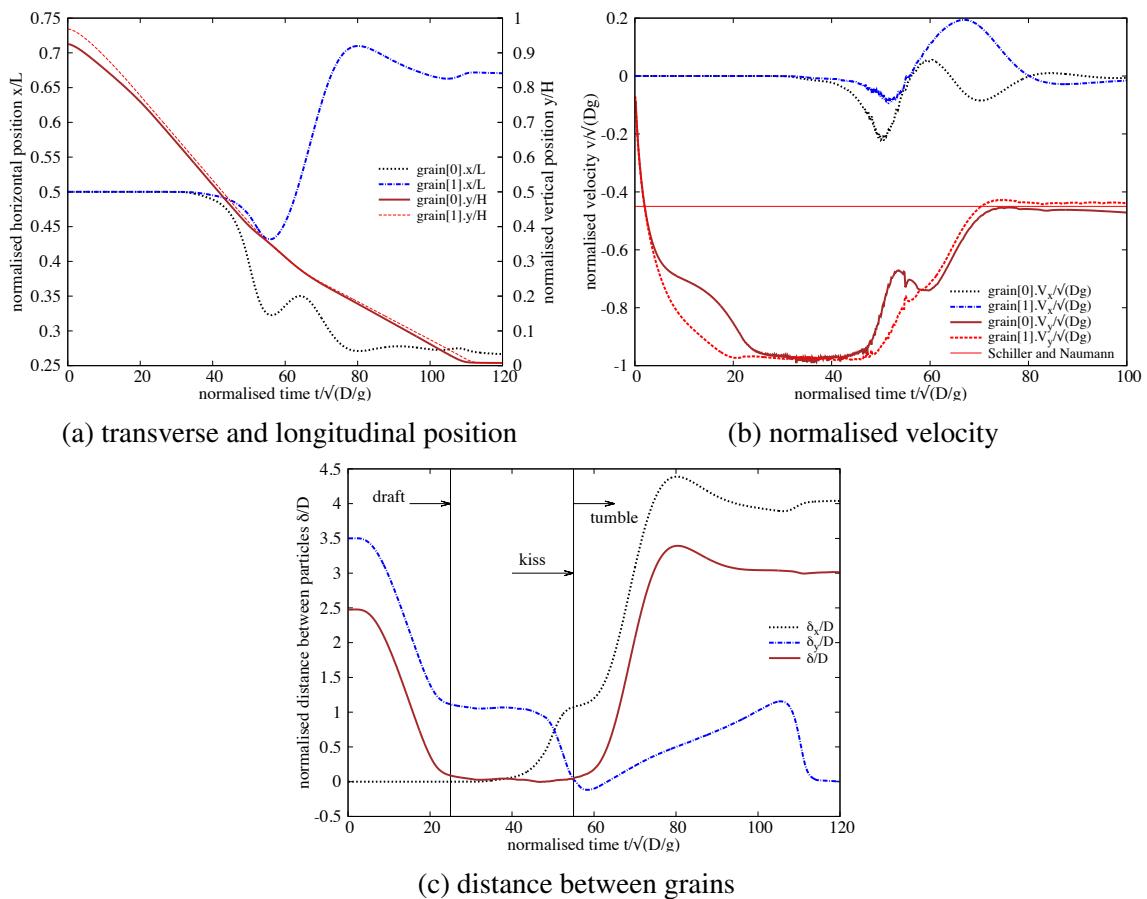


Figure 5.15 Time history of two circular grains during sedimentation.

Also, the data transfer between the host and the device is very expensive. In the present study, the LBM implementation follows carefully chosen data transfer schemes in global memory.

There are three ways to accelerate GPGPU applications: (a) Using ‘drop-in’ libraries, (b) using directives by exposing parallelism, and (c) using dedicated GPGPU programming languages. OpenACC (Open Accelerators) is an open GPU directives programming standard for parallel computing on heterogeneous CPU/GPU systems. Unlike conventional GPU programming languages, such as CUDA, OpenACC uses directives to specify parallel regions in the code and performance tuning works on exposing parallelism. OpenACC targets a host-directed execution model where the sequential code runs on a conventional processor and computationally intensive parallel pieces of code (kernels) run on an accelerator such as a GPU (see figure 5.16).

The original GPGPU LBM – DEM code was implemented in C using OpenACC API v1.0, which was released in November 2011. The current implementation in C++ uses OpenACC API v2.0a ([OpenACC-Members, 2013](#)) and has two compute constructs, the kernels construct and the parallel construct. LBM – DEM implementation predominantly uses the OpenACC gang and vector parallelism. The LBM – DEM code runs sequential and computationally less intensive functions on the CPU, OpenMP multi-threading is used when possible. Computationally intensive functions are converted to a target accelerator specific GPU parallel code. Schematics of a heterogeneous CPU/GPU system is shown in figure 5.16.

OpenACC offers kernel and parallel constructs to parallelise algorithms on CUDA kernels. The loop nests in a kernel construct are converted by the compiler into parallel kernels that run efficiently on a GPU. There are three steps to this process. The first is to identify the loops that can be executed in parallel. The second is to map that abstract loop parallelism onto a concrete hardware parallelism. In OpenACC terms, gang parallelism maps to grid-level parallelism (equivalent to a CUDA blockIdx), and vector parallelism maps to thread-level parallelism (equivalent to a CUDA threadIdx). The compiler normally maps a single loop across multiple levels of parallelism using strip-mining. Finally, in step three the compiler generates and optimizes the actual code to implement the selected parallelism mapping.

An OpenACC parallel construct creates a number of parallel threads that immediately begin executing the body of the parallel construct redundantly. When a thread reaches a work-sharing loop, that thread will execute some subset of the loop iterations, depending on the scheduling policy as specified by the program or at the runtime. The code generation and optimization for a parallel construct is essentially the same as for the kernel construct. A key difference is that unlike a kernel construct, the entire parallel construct becomes a single target parallel operation, aka a single CUDA kernel. Both constructs allow for automatic vectorization within the loops ([Wolfe, 2012](#)).

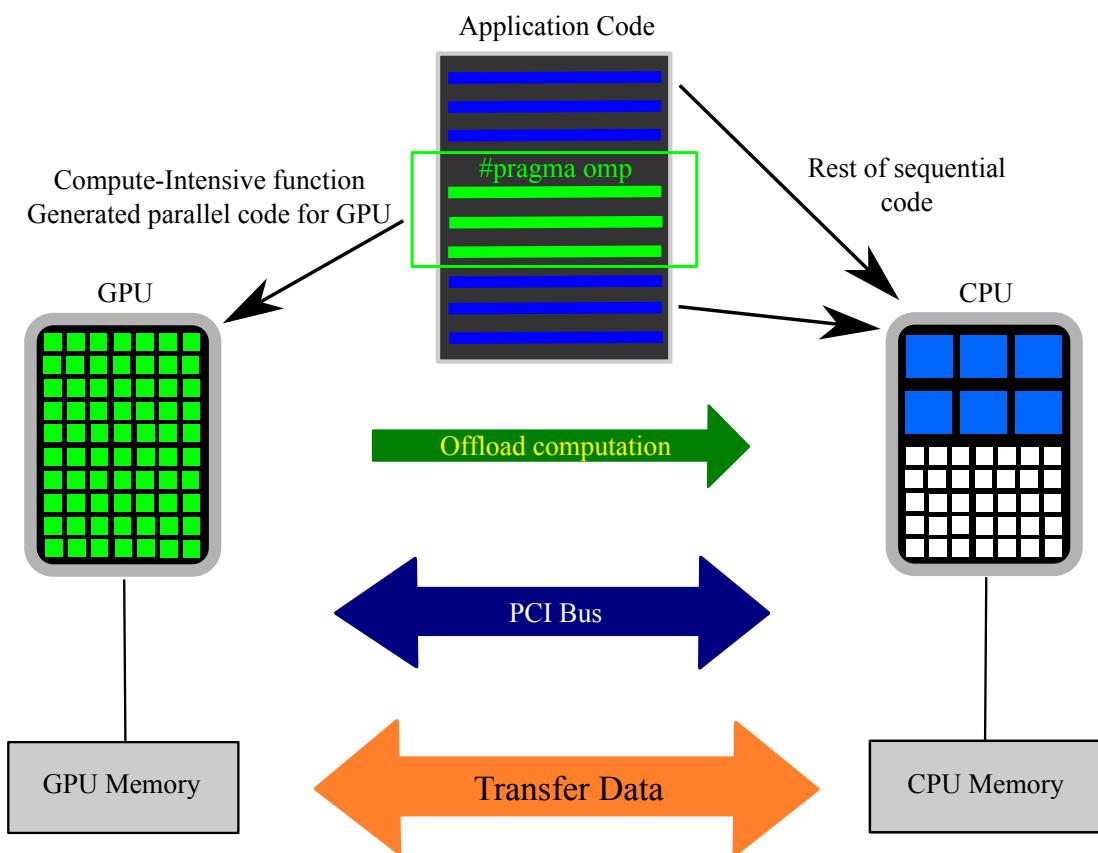


Figure 5.16 Schematics of a heterogeneous CPU/GPU system.

An excerpt from the LBM-DEM code showing the OpenACC GPU implementation of the hydrodynamic force computation is presented in Listing 5.1. The kernels loop construct tells the compiler to map the body of the following loop into an accelerator kernel. The GPU implementation uses a two-dimensional grid splitting the iterations across both the vector and gang modes. The kernel is mapped to a vector mode mapped (aligned with CUDA threadidx% $x$ ) with a vector length (thread block size) of 128. The kernel is also mapped to gang parallelism, aligned to CUDA blockidx% $x$ , to avoid partition camping by mapping the stride-1 loop to the  $x$  dimension. The compiler strip-mines the loop into chunks of 256 iterations, mapping the 256 iterations of a chunk in vector mode across the threads of a CUDA thread block, and maps the  $n/256$  chunks in gang mode across the thread blocks of the CUDA grid. The consecutive iterations ( $i$  and  $i+1$ ), which refer to contiguous array elements ( $f_{hf}[i]$  and  $f_{hf}[i+1]$ ), are mapped to adjacent CUDA threads in the same thread block, to optimize for coalesced memory accesses.

Listing 5.1 OpenACC GPU implementation of the hydrodynamic force computation.

```

1 //OpenACC Kernels copy data between the host and the device
2 #pragma acc kernels
3 copyout(fhf1 [0: nbgrains ], f hf2 [0: nbgrains ], f hf3 [0: nbgrains ])
4 copyin(obst [0:][0:], g [0: nbgrains ], ey [0:], f [0:][0:][0:], ex [0:])
5 //Create individual threads for each DEM grain
6 #pragma acc parallel for
7 for ( i=0; i<nbgrains ;i++) {
8     // Reset hydrodynamic forces to zero at the start of time step
9     f hf1 [ i]=f hf2 [ i]=f hf3 [ i ]=0. ;
10    // Iterate through all lattice nodes
11    for (y=0; y<ly;y++) {
12        for (x=0; x<lx;x++) {
13            if (obst [x ][y]==i) {
14                // generate code to execute the iterations in parallel with
15                // no synchronization
16                #pragma acc for independent
17                for (iLB=1; iLB<Q; iLB++) {
18                    next_x=x+ex[iLB];
19                    next_y=y+ey[iLB];
20                    if (iLB<=half) halfq=half ;
21                    else halfq= -half;

```

```

1   22      if (obst[next_x][next_y]!=i) {
2   23          fnx=(f[x][y][iLB+halfq]+f[next_x][next_y][iLB])*ex[iLB+halfq];
3   24          fny=(f[x][y][iLB+halfq]+f[next_x][next_y][iLB])*ey[iLB+halfq];
4   25          fhf1[i]=fhf1[i]+fnx;
5   26          fhf2[i]=fhf2[i]+fny;
6   27          fhf3[i]=fhf3[i]-fnx*(y-(g[i].x2-wall_bottom_y)/dx)
7   28                  +fny*(x-(g[i].x1-wall_left_x)/dx);
8   29      }
9   30  }
10  31  }
11  32  }
12  33  }
13  34 }
```

14       Memory transaction optimisation is more important than computation optimisation. Registers  
 15       do not give rise to any specific problem apart from their limited amount. Global memory,  
 16       being the only one accessible by both the CPU and the GPU, is the critical path as it suffers  
 17       from high latency. However, this latency is mostly hidden by the scheduler which stalls inactive  
 18       warps until data are available. For data intensive LBM, this aspect is generally the limiting  
 19       factor ([Obrecht and Kuznik, 2011](#)). To optimise the global memory transactions, the memory  
 20       access is coalesced and aligned, as explained above. The memory transactions between the  
 21       host and the target through a PCI bus are kept to a minimum.

22       A two-dimensional fluid – grain system, which consists of 7.2 million LBM nodes and  
 23       2500 DEM grains is used to demonstrate the ability of the GPGPU LBM – DEM code. The  
 24       wall time required to compute 100 iterations of the given LBM – DEM problem is compared  
 25       for executions running on a single CPU thread, multi-threaded CPU (using OpenMP) and  
 26       the GPGPU implementations (see table 5.3). The speed-up of parallel implementations are  
 27       measured against the single CPU thread execution time. OpenMP parallelised multi-threaded  
 28       CPU execution running on 12 cores achieved a speed-up of 13.5x in comparison to a serial  
 29       implementation. GPGPU implementation using OpenACC delivered an impressive 126x speed-  
 30       up in comparison to a single thread CPU execution and about 10 times quicker than a CPU  
 31       parallel code. In other words, a simulation that would have ordinarily taken 126 days to  
 32       compute, could now be finished in a day using a GPU.

33       Scalability is an important criterion when developing high-performance computing codes.  
 34       Scalability in GPUs is measured in terms of SM utilisation. It is important to distribute  
 35       sufficient work to all SMs such that on every cycle the warp scheduler has at least one warp  
 36       eligible to issue instructions. In general, sufficient warps on each SM should be available to

Table 5.3 GPU vs CPU parallelisation

Execution	Computational Time (s)	Speedup
CPU 1 OpenMP thread	2016	–
CPU 2 OpenMP threads	1035	1.5 x
CPU 4 OpenMP threads	660	3.0 x
CPU 12 OpenMP threads	150	13.5 x
GPU OpenACC	16	126.0 x

# Wall time for 100 iteration for 7.2 Million LBM nodes and 2500 DEM grains.

\* CPU OpenMP threads - 6 core Intel Xeon @ 3.3GHz

† GPU threads - GeForce GTX 580 - 512 CUDA cores

hide instruction and memory latency and to provide a variety of instruction types to fill the execution pipeline. Figure 5.17 shows the scalability of GP-GPU implementation as the LBM domain size is increased from 500,000 to 9 million nodes. With increase in LBM nodes the computation time increases linearly with a slope of about 2, which shows that the LBM-DEM implementation algorithm scales with the domain size.

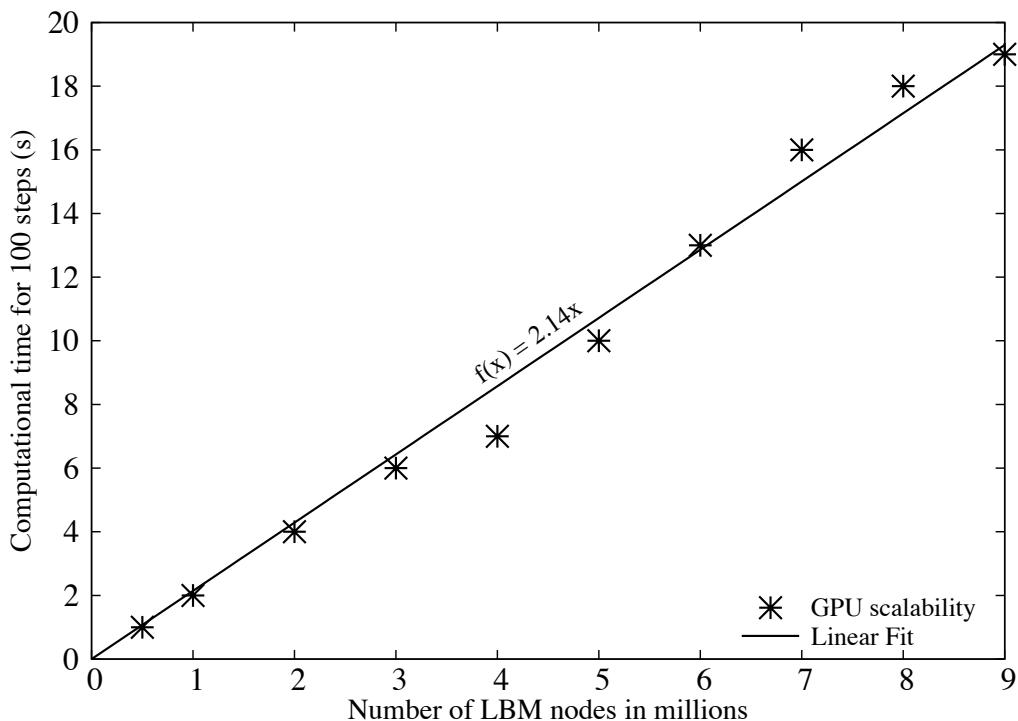


Figure 5.17 GPU scalability with increase in LBM nodes

A two-dimensional coupled LBM-DEM technique is developed to understand the local rheology of granular flows in fluid. The coupled LBM-DEM technique offers the possibility to

1 capture the intricate microscale effects such as the hydrodynamic instabilities. The Smagorinsky  
2 turbulence model is implement in LBM to capture the unsteady flow dynamics in underwater  
3 granular avalanches. The GPGPU implementation of the coupled LBM – DEM technique  
4 offers the capability to model large scale fluid – grain systems, which are otherwise impossible  
5 to model using conventional computation techniques. Efficient data transfer mechanisms that  
6 achieves coalesced global memory ensures that the GPGPU implementation scales linearly  
7 with the domain size.

# References

- Aidun, C., Lu, Y., and Ding, E. (1998). Direct analysis of particulate suspensions with inertia using the discrete Boltzmann equation. *Journal of Fluid Mechanics*, 373(-1):287–311. 1  
2  
3
- Balmforth, N. J. and Kerswell, R. R. (2005). Granular collapse in two dimensions. *Journal of Fluid Mechanics*, 538:399–428. 4  
5
- Bandara, S. (2013). *Material Point Method to simulate Large Deformation Problems in Fluid-saturated Granular Medium*. PhD thesis, University of Cambridge. 6  
7
- Bathe, K. and Zhang, H. (2004). Finite element developments for general fluid flows with structural interactions. *International Journal for Numerical Methods in Engineering*, 60(1):213–232. 8  
9  
10
- Breuer, M., Bernsdorf, J., Zeiser, T., and Durst, F. (2000). Accurate computations of the laminar flow past a square cylinder based on two different methods: lattice-Boltzmann and finite-volume. *Journal of Heat and Fluid Flow*, 21(2):186–196. 11  
12  
13
- Capecelatro, J. and Desjardins, O. (2013). An Euler–Lagrange strategy for simulating particle-laden flows. *Journal of Computational Physics*, 238(0):1–31. 14  
15
- Chen, S. and Doolen, G. G. D. (1998). Lattice Boltzmann method for fluid flows. *Annual review of fluid mechanics*, 30(1):329–364. 16  
17
- Cook, B., Noble, D., and Williams, J. (2004). A direct simulation method for particle-fluid systems. *Engineering Computations*, 21(2/3/4):151–168. 18  
19
- Daerr, A. and Douady, S. (1999). Two types of avalanche behaviour in granular media. *Nature*, 399(6733):241–243. 20  
21
- Du, R., Shi, B., and Chen, X. (2006). Multi-relaxation-time lattice Boltzmann model for incompressible flow. *Physics Letters A*, 359(6):564–572. 22  
23
- Durst, F., Ray, S., Unsal, B., and Bayoumi, O. (2005). The development lengths of laminar pipe and channel flows. *Journal of fluids engineering*, 127:1154. 24  
25
- Feng, Y. T., Han, K., and Owen, D. R. J. (2007). Coupled lattice Boltzmann method and discrete element modelling of particle transport in turbulent fluid flows: Computational issues. *International Journal for Numerical Methods in Engineering*, 72(9):1111–1134. 26  
27  
28
- Fortes, A. (1987). Nonlinear mechanics of fluidization of beds of spherical particles. . . of *Fluid Mechanics*. 29  
30

- <sup>1</sup> Frisch, U. and Kolmogorov, A. (1995). *Turbulence: the legacy of AN Kolmogorov*. Cambridge Univ Pr.
- <sup>3</sup> Guilkey, J., Harman, T., Xia, A., Kashiwa, B., and McMurtry, P. (2003). An Eulerian-Lagrangian approach for large deformation fluid structure interaction problems, Part 1: algorithm development. *Advances in Fluid Mechanics*, 36:143–156.
- <sup>6</sup> Han, K., Feng, Y., and Owen, D. (2007a). Coupled lattice Boltzmann and discrete element modelling of fluid-particle interaction problems. *Computers & Structures*, 85(11-14):1080–1088.
- <sup>9</sup> Han, S., Zhu, P., and Lin, Z. (2007b). Two-dimensional interpolation-supplemented and Taylor-series expansion-based lattice Boltzmann method and its application. . . in *Nonlinear Science and Numerical Simulation*, 12(7):1162–1171.
- <sup>12</sup> He, X. and Luo, L. L.-S. (1997a). Theory of the lattice Boltzmann method: From the Boltzmann equation to the lattice Boltzmann equation. *Physical Review E*, 56(6):6811.
- <sup>14</sup> He, X. and Luo, L.-S. (1997b). A priori derivation of the lattice Boltzmann equation. *Physical Review E*, 55(6):R6333–R6336.
- <sup>16</sup> He, X., Zou, Q., Luo, L. S., and Dembo, M. (1997). Analytic solutions of simple flows and analysis of nonslip boundary conditions for the lattice Boltzmann BGK model. *Journal of Statistical Physics*, 87(1):115–136.
- <sup>19</sup> Iglberger, K., Thurey, N., and Rude, U. (2008). Simulation of moving particles in 3D with the Lattice Boltzmann method. *Computers & Mathematics with Applications*, 55(7):1461–1468.
- <sup>21</sup> Komiwas, V., Mege, P., Meimon, Y., and Herrmann, H. (2005). Simulation of granular flow in a fluid applied to sedimentation. *Granular Matter*, 8(1):41–54.
- <sup>23</sup> Lacaze, L. and Kerswell, R. R. (2009). Axisymmetric Granular Collapse: A Transient 3D Flow Test of Viscoplasticity. *Physical Review Letters*, 102(10):108305.
- <sup>25</sup> Ladd, A. J. (1994). Numerical simulations of particulate suspensions via a discretized Boltzmann equation. Part 1. Theoretical foundation. *Journal of Fluid Mechanics*, 271:285–309.
- <sup>27</sup> Ladd, A. J. C. and Verberg, R. (2001). Lattice-Boltzmann Simulations of Particle-Fluid Suspensions. *Journal of Statistical Physics*, 104(5):1191–1251.
- <sup>29</sup> Lajeunesse, E., Mangeney-Castelnau, A., and Vilotte, J. P. (2004). Spreading of a granular mass on a horizontal plane. *Physics of Fluids*, 16(7):2371.
- <sup>31</sup> Liu, S. H., Sun, D. A., and Wang, Y. (2003). Numerical study of soil collapse behavior by discrete element modelling. *Computers and Geotechnics*, 30(Compex):399–408.
- <sup>33</sup> Lube, G., Huppert, H. E., Sparks, R. S. J., and Freundt, A. (2005). Collapses of two-dimensional granular columns. *Physical Review E - Statistical, Nonlinear, and Soft Matter Physics*, 72(4):1–10.
- <sup>36</sup> Mei, R., Shyy, W., Yu, D., and Luo, L.-S. (2000). Lattice Boltzmann Method for 3-D Flows with Curved Boundary. *Journal of Computational Physics*, 161(2):680–699.

- Noble, D. and Torczynski, J. (1998). A lattice-Boltzmann method for partially saturated computational cells. *International Journal of Modern Physics C-Physics and Computer*, 9(8):1189–1202. 1  
2  
3
- Obrecht, C. and Kuznik, F. (2011). A new approach to the lattice Boltzmann method for graphics processing units. *Computers & Mathematics* .... 4  
5
- Oda, M. and Iwashita, K. (1999). *Mechanics of granular materials: an introduction*. Taylor & Francis. 6  
7
- OpenACC-Members (2013). The OpenACC Application Programming Interface Version 2.0 (June, 2013; Corrected: August, 2013). Technical report, OpenMP Standard Group. 8  
9
- Pan, C., Luo, L., and Miller, C. (2006). An evaluation of lattice Boltzmann schemes for porous medium flow simulation. *Computers & fluids*, 35(8-9):898–909. 10  
11
- Schaefer, D. G. (1990). Instability and ill-posedness in the deformation of granular materials. *International Journal for Numerical and Analytical Methods in Geomechanics*, 14(4):253–278. 12  
13  
14
- Smagorinsky, J. (1963). General circulation experiments with the primitive equations. *Monthly weather review*, 91(3):99–164. 15  
16
- Staron, L., Radjai, F., and Vilotte, J. P. (2005). Multi-scale analysis of the stress state in a granular slope in transition to failure. *The European physical journal. E, Soft matter*, 18(3):311–20. 17  
18  
19
- Succi, S. (2001). *The lattice Boltzmann equation for fluid dynamics and beyond*. Oxford University Press. 20  
21
- Succi, S., Foti, E., and Higuera, F. (1989). Three-Dimensional Flows in Complex Geometries with the Lattice Boltzmann Method. *Europhysics Letters (EPL)*, 10(5):433–438. 22  
23
- Sukop, M. and Thorne, D. (2006). *Lattice Boltzmann modeling: An introduction for geoscientists and engineers*. Springer Verlag. 24  
25
- Sun, X., Sakai, M., and Yamada, Y. (2013). Three-dimensional simulation of a solid–liquid flow by the DEM–SPH method. *Journal of Computational Physics*, 248(0):147–176. 26  
27
- Wang, L., Guo, Z., and Mi, J. (2014). Drafting, kissing and tumbling process of two particles with different sizes. *Computers & Fluids*. 28  
29
- Willis, A., Peixinho, J., Kerswell, R., and Mullin, T. (2008). Experimental and theoretical progress in pipe flow transition. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 366(1876):2671. 30  
31  
32
- Wolfe, M. (2012). PGInsider August 2012: OpenACC Kernels and Parallel Constructs. 33
- Xiong, Q., Madadi-Kandjani, E., and Lorenzini, G. (2014). A LBM–DEM solver for fast discrete particle simulation of particle–fluid flows. *Continuum Mechanics and Thermodynamics*. 34  
35  
36

- <sup>1</sup> Yu, D., Mei, R., Luo, L., and Shyy, W. (2003). Viscous flow computations with the method of  
<sup>2</sup> Lattice Boltzmann equation. *Progress in Aerospace Sciences*, 39(5):329–367.
- <sup>3</sup> Yu, H., Girimaji, S., and Luo, L. (2005). Lattice Boltzmann simulations of decaying homoge-  
<sup>4</sup> neous isotropic turbulence. *Physical Review E*, 71(1):016708.
- <sup>5</sup> Zhong, Q. and Olson, M. (1991). Finite element–algebraic closure analysis of turbulent  
<sup>6</sup> separated-reattaching flow around a rectangular body. *Computer methods in applied me-  
<sup>7</sup> chanics and engineering*, 85(2):131–150.
- <sup>8</sup> Zhou, H., Mo, G., Wu, F., Zhao, J., Rui, M., and Cen, K. (2012). GPU implementation of  
<sup>9</sup> lattice Boltzmann method for flows with curved boundaries. *Computer Methods in Applied  
<sup>10</sup> Mechanics and Engineering*, 225-228(null):65–73.
- <sup>11</sup> Ziegler, D. (1993). Boundary conditions for Lattice Boltzmann simulations. *Journal of  
<sup>12</sup> Statistical Physics*, 71(5):1171–1177.
- <sup>13</sup> Zou, Q. and He, X. (1997). On pressure and velocity boundary conditions for the Lattice  
<sup>14</sup> Boltzmann BGK model. *Physics of Fluids*, 9(6):1591–1598.