Molecular Dynamics Simulation of Two Possible Mixing Configurations

Jonathan Hunt

June 10, 2005

Abstract

This report examines recent progress on understanding flow in small channels. In particular, it explores the relationship between wettability and slip length and the application of this understanding to microfluidics. Molecular dynamics simulation is utilised to explore two geometric configurations and changes in flow velocities with varying wettability. It is found that within these two configurations no significant change in flow velocities are made by varying wettability and so their application to mixing is of limited usefulness.

Contents

Chapter 1

Introduction

Imagine being told at the end of World War II that scientists expect to be able to make wires only hundreds of nanometres wide and that they also expect to make transistors and other devices that can control the flow of electrons through these wires on the same size scale. Over the coming decades that exactly what was accomplished (*The Integrated Circuit*, n.d.). The ability to manufacture miniture electronic devices has had a profound impact on history. In fact, the ease of numerical computation that these devices allowe is what makes a large part of this report feasible.

However, even as work continues to improve electronics there is much else that people are interested in miniaturizing. Feynman (1959) gave a very forward thinking talk for the time - demonstrating that there was no reason, in principle, why machines could not be miniaturized. Unfortunately, why we still believe much that Feynman talked is still considered, in principle,

doable it has proven very difficult.

Microfluidics is one area of miniaturization technology that is coming of age. At the broadest definition microfluidics refers to fluid flow through small channels. The scale of small depends on the topic of discussion. Although you may not have heard of microfluidics, it is quite likely you have utilised technology from this field. If you have ever had the misfortune to use an inkjet printer[1] you were using a device that relies of microfluidic technology to precisely control the ink flow. However, microfluidics provide much more potential than just printing. The ability to manipulate small amounts of liquids in a small space could potentially be as influential, or more, than manipulating electrons on integrated circuits.[2] Potential applications include DNA sequencing, blood monitoring and drug delivery (Coch, Evans, & Brunnschweiler, 2000) to the more general concept of taking any chemical reaction currently performed on relatively large quantities of material and miniaturizing it - allowing portability, better control over chemical reactions and smaller assays.

Obviously, there is much scientific and commercial interest in microfluidics and indeed microfluidics are being applied already in some of the applications mentioned. Fabrication techniques for microfluidic devices have

[1] The author recommends laser printers for their currently superior contrast and lack of smudging.

[2] The idea of a comparison between the emerging field of nanofluidics and the early days of computers is not my own - it comes from a talk given by Shawn Hendy at Massey University

been developed (Coch et al., 2000, ch. 5) - some are modifications of silicon based techniques used in integrated circuit construction. However, there is still much that is not understood about life in the small scale. Traditional theories on liquid flow tend to break down for small channels where surface effects and boundary conditions become much more influential on the fluid flow. The small size of the channels tends to ensure the flow occurs at low Reynold's numbers and thus turbulent flow is uncommon on the small scale. This can pose a problem when attempting to mix two fluid flows. Unlike electrons which behave relatively linearly as the size of the channel is reduced, fluid flow at small length scales can appear quite different from that of the bulk fluid. Traditional fluid control devices, such as pumps, valves, etc., do not tend to scale well and it may not always be feasible to construct them on small length scales.

An important property of fluid flow is slip length. Roughly defined as a measure of how much the fluid is flowing at the edges of a channel [3] Obviously, in small channels the slip length may be quite influential on the fluid flow and is much interest. In classical fluid mechanics the slip length is considered to be zero (no fluid flow at the edge). This report outlines some previous work that appears to demonstrate non-zero slip lengths at the boundary in small channels and some applications of this. It attempts to give relevant background on the fluid flow and in particular on the use

[3] A more refined definition of this property is given later.

of molecular dynamics simulation - the use of computers - to give insight on the problem. Finally, simulation of two geometric configurations is discussed along with the results and applications.

Chapter 2

Background on Molecular Dynamics Simulation

2.1 History of Molecular Dynamics Simulation

Molecular dynamics simulation is a relatively recent development of an ancient idea. Given a system, made of many parts, that is known to be in a certain initial state and given information about the interaction between these parts - the forces they exert on one another - one ought then,in principle, to be able to calculate the future development of the system from the initial state. Unfortunately, while this can be applied with ease to systems of one or two parts, the solution of a vast array of physical problems using Newton's laws of motion are a tribute to this, up until the 1950's, it was not possible to straightforwardly apply deterministic laws to systems of large

numbers of parts. Even then it was not until the 1970's with digital computers becoming much more powerful and common that molecular simulation began to receive widespread attention (Haile, 1992). Ercolessi (1997) and Allen and Tildesley (1989) both contain more information on the history of molecular dynamics and references to some of the early work in the field.

2.2 What is Molecular Dynamics

Molecular dynamics (MD) simulation can be concisely defined as the use of computers to numerically solve Newton's laws of motion for atoms or molecules (Lauga, Brenner, & Stone, 2005).

$$m_i \frac{d^2 \mathbf{r}_i}{dt^2} = \sum_j \mathbf{F}_{ij} \tag{2.1}$$

where $\mathbf{r_i}$ and m_i are the position and mass respectively of atom or molecule i and $\mathbf{F_{ij}}$ is the intermolecular or interatomic force between molecules or atoms i and j.

Unlike Monte Carlo techniques molecular dynamics simulation is, in principal, completely deterministic (Ercolessi, 1997). Given the initial conditions the equations are solved to calculate a trajectory in phase space. Haile (1992, ch. 1) determines there to be three steps to MD simulation: model the individual particles, simulate the movements of large numbers of the individual particles and analyse the simulation data for statistical results. Thus the

calculation of the trajectory is not, by itself, of much interest. However, it allows one to obtain statistical results on the collective behaviour of the particles of the system. It is also relevant to point out that deterministic and predictable are not the same thing (Haile, 1992, ch. 1). A system may be determinstic in that the final state of the system is completely determined by the initial state while stating a system is predictable implies that the relationship between the initial state and final state is simple enough that it is possible to calculate a final state from any initial state.

2.3 Philosophy of Molecular Dynamics

Molecular dynamics simulation can be thought of as computer experiments. The physical sciences have always had experiment and theory. In the past scientific theory was, by necessity, produced by reducing complex systems to simpler systems and analyzing them. Until the advent of computers the subsystems needed to be simple enough to be able to be mathematically modelled in a way that was solvable analytically. This often requires taking several approximations and even then may produce solutions only for a few simple cases (Ercolessi, 1997).

The introduction of computers gives a new option - computer experiments. This allows more complex mathematical models for which there is no known analytic solution to be tested using simulation. It also allows models to be tested in ways that were not possible previously - such as de-

termining the predicted melting point of a material modelled with a given potential (Ercolessi, 1997).

2.4 Interaction Potential

Molecular dynamics simulation requires a system to model. Newton's laws of motion (2.1) are then solved for the system approximately using finite difference methods. Obviously, in order to solve (2.1) one needs to know the masses of all the particles in the system and the initial positions of all particles. But very importantly, a model must be made describing the interactions between particles. This is an important factor in determining how closely a simulation will match the physical system simulation. Usually a potential function much be made for the potential of the system.

$$V(\mathbf{r}_1, \ldots, \mathbf{r}_N) \tag{2.2}$$

and from this the forces on the particles of the system can be calculated as:

$$\mathbf{F}_i = -\nabla_{\mathbf{r}_i} V(\mathbf{r}_1, \ldots, \mathbf{r}_N) \tag{2.3}$$

since there are no dissipative forces acting on the particle. Generally, V will depend on the relative positions of the particles to one another, not their absolute positions. In many simple cases the potential is calculated as the

sum of pair-wise potential interactions:

$$V(\mathbf{r}_1, \ldots, \mathbf{r}_N) = \sum_i \sum_{j>i} \phi(|\mathbf{r}_j - \mathbf{r}_i|) \tag{2.4}$$

The sum over $j > i$ implies that each atom pair is summed once. It is now recognised that using pairwise potential is a poor approximation in many areas (Ercolessi, 1997), however because of their historical usage and simplicity pairwise potentials continue to get much use - although many-body potentials are also in common use.

2.4.1 Lennard-Jones Potential

The Lennard-Jones potential - a pairwise-potential - is not rigourously derivable from theory or experimental results. However, it has been shown to be a good approximation for classical Newtonian fluids. It is given as:

$$\phi(r_{ij}) = 4\varepsilon\left[\left(\frac{\sigma}{r_{ij}}\right)^{12} - \left(\frac{\sigma}{r_{ij}}\right)^{6}\right] \tag{2.5}$$

where r_{ij} is the distance between two atoms i and j and ε and σ are parameters of the system that are chosen to fit the experimental data. The $(\sigma/r)^{12}$ term dominates when the two atoms are close together providing a repulsive interaction while the $(\sigma/r)^6$ gives an attractive potential for atoms separated by long distances. One of the primary advantages of the Lennard-Jones potential is that it is a relativity easy function to calculate - an important

consideration for practical implementation of the MD simulation. Ercolessi (1997) contains a discussion on when the LJ potential is an good approximation to the real world - namely in rare gases with weak interactions such as Ar or Kr. Allen and Tildesley (1989, p. 21) contains a useful table of parameters for the LJ-potential for real gases.

Regardless of how well LJ potential fits actual materials it has now become an important model system that researchers study when they are primarily interested in the basic issues rather than the properties of specific materials. Thus, there is a large and growing body of literature on the LJ potential.

The simulations reported here make use of the modified Lennard-Jones potential which is given as:

$$\phi(r_{ij}) = 4\varepsilon\Big[\Big(\frac{\sigma}{r_{ij}}\Big)^{12} - c_{ij}\Big(\frac{\sigma}{r_{ij}}\Big)^{6}\Big] \tag{2.6}$$

where c_{ij} is an additional interaction cofficient. Then from 2.3 the forces acting on a particle i is given as:

$$\mathbf{F}_i = -\sum_{i\neq j}\frac{\partial\phi(r_{ij})}{\partial r_{ij}} = \sum_{i\neq j} 48\varepsilon\Big[\Big(\frac{\sigma}{r_{ij}}\Big)^{12} - \frac{c_{ij}}{2}\Big(\frac{\sigma}{r_{ij}}\Big)^{6}\Big] \tag{2.7}$$

2.5 Integration Algorithms

Once a potential function is chosen, the equations of motion for the system (2.1) must be approximately solved. The most common methods are finite difference methods where time is discretised into steps, each a finite time Δt. A simple way is a discrete Taylor expansion about time t:

$$\begin{aligned} \mathbf{r_i}(t+\Delta t) &= \mathbf{r_i}(t) + \Delta t \dot{\mathbf{r}}_\mathbf{i}(t) + \frac{1}{2}(\Delta t)^2 \ddot{\mathbf{r}}_\mathbf{i}(t) + O\left(\Delta t^2\right) \\ \dot{\mathbf{r}}_\mathbf{i}(t+\Delta t) &= \dot{\mathbf{r}}_\mathbf{i}(t) + \Delta t \ddot{\mathbf{r}}_\mathbf{i}(t) + O\left(\Delta t^2\right) \\ \ddot{\mathbf{r}}_\mathbf{i}(t+\Delta t) &= \frac{\mathbf{F}_i}{m_i} \end{aligned} \tag{2.8}$$

There are two sources of error from these equations. Truncation error results from terminating the Taylor expansions at a finite number of points while round-off error results from all the errors involved in using a finite timestep. Of more concern than the error from one timestep is the accumulated error of multiple timesteps - global error. Haile (1992, p. 151) outlines in more depth the details of error calculations. There are many variations that may be made to (2.8) - such as using the results of the previous step values $(t-\Delta t)$ to increase accuracy.

2.5.1 Predictor-Corrector Algorithms

Generally, a predictor-corrector algorithm involves two steps. First the predicted positions $\bar{\mathbf{r}}_i$ at a later time are calculated from the current positions

using some finite-difference method.

$$\bar{\mathbf{r}}_i(t+\Delta t) = f\Big(\mathbf{r}_i(t), \dot{\mathbf{r}}_i(t), \ddot{\mathbf{r}}_i(t)\Big) \tag{2.9}$$

These predicted values are then used to calculate the predicted accelerations $\bar{\ddot{\mathbf{r}}}_i$ and velocities $\bar{\dot{\mathbf{r}}}_i$. This information is then used to make a correction on positions and velocities (Rahman, 1964).

$$\begin{aligned} \mathbf{r}_i(t+\Delta t) &= g\Big(\mathbf{r}_i(t), \dot{\mathbf{r}}_i(t), \ddot{\mathbf{r}}_i(t), \bar{\mathbf{r}}_i(t+\Delta t), \bar{\dot{\mathbf{r}}}_i(t+\Delta t), \bar{\ddot{\mathbf{r}}}_i(t+\Delta t)\Big) \\ \dot{\mathbf{r}}_i(t+\Delta t) &= h\Big(\mathbf{r}_i(t), \cdots, \ddot{\mathbf{r}}_i(t), \bar{\mathbf{r}}_i(t+\Delta t), \cdots, \bar{\ddot{\mathbf{r}}}_i(t+\Delta t)\Big) \end{aligned} \tag{2.10}$$

This prediction-corrector step can be repeated until the predicted values are within a given error of the corrected values. Gear (1971, ch. 9) contains much more information on the mathematical background and several commonly used predictor-corrector algorithms. Predictor-corrector algorithms are now not so commonly used in MD simulation (Allen, 2004). In general, the computation of the forces is the most computational intensive task and so low-order methods that allow long timesteps are necessary. The predictor-corrector step has the disadvantage of requiring more than one force calculation per timestep.

2.5.2 Verlet Algorithm

The Verlet algorithm (Verlet, 1967) and variations of this method are probably the most common time integration algorithm in use in MD today. The Verlet algorithm is found by taking the Taylor expansion about $t + \Delta t$ and $t - \Delta t$ and rearranging to give:

$$\mathbf{r}_i(t + \Delta t) = -\mathbf{r}_i(t - \Delta t) + 2\mathbf{r}_i(t) + \Delta t^2 \ddot{\mathbf{r}}_i(t) \tag{2.11}$$

The Verlet equation is correct to $O(\Delta t^4)$ but it suffers one drawback. The velocities are never explicitly calculated, however, often the velocities of the particles are of much interest. Of course, one could calculate them using a Taylor expansion but an improved method is commonly used known as the Velocity Verlet algorithm (Swope, Andersen, Berenes, & Wilson, 1981) which is mathematically equivalent but provide greater numerical precision and involve the explicit calculation of velocities.

$$\begin{aligned} \mathbf{r}_i(t + \Delta t) &= \mathbf{r}_i(t) + \Delta t \dot{\mathbf{r}}_i(t) + \frac{1}{2}\Delta t^2 \ddot{\mathbf{r}}_i(t) \\ \dot{\mathbf{r}}_i(t + \Delta t) &= \mathbf{r}_i(t) + \frac{1}{2}\Delta t \Big[\ddot{\mathbf{r}}_i(t) + \ddot{\mathbf{r}}_i(t + \Delta t)\Big] \end{aligned} \tag{2.12}$$

The algorithm only requires storage of r, v and a for each particle when implemented in a two-step form (Allen & Tildesley, 1989). First the positions

at $t + \Delta t$ are calculated and then the velocities at $t + \frac{1}{2}\Delta t$ using

$$\dot{\mathbf{r}}_i(t + \frac{1}{2}\Delta t) = \dot{\mathbf{r}}_i(t) + \frac{1}{2}\Delta t\ddot{\mathbf{r}}_i(t) \tag{2.13}$$

and then the accelerations at time $t + \Delta t$ is computed and the velocities recalculated at $t + \Delta t$ using

$$\dot{\mathbf{r}}_i(t + \Delta t) = \dot{\mathbf{r}}_i(t + \frac{1}{2}\Delta t) + \frac{1}{2}\Delta t\ddot{\mathbf{r}}_i(t + \frac{1}{2}\Delta t) \tag{2.14}$$

2.6 Computational Considerations

2.6.1 Potential Truncation and Bounding Boxes

As mentioned previously, usually the most time consuming part of molecular dynamics simulation is the calculations of the forces on each particle which must be done each timestep from (2.3). As first stated this operation is of order N^2 where N is the number of particles in the system. An obvious simplication is the use of Newton's second law to allow the interaction between particles to be calculated only once per pair.

Another approximation commonly employed to reduce computational costs is a cutoff radius r_c. This takes advantage of the asymtotic approach to zero with increasing distance of most potential functions by setting the potential to zero for $r_{ij} \geq r_c$. One problem with this approach is that this causes a discontinuity in the potential function (Allen & Tildesley, 1989)

which implies that the properties of the system will no longer be identical with one without the cutoff. However, a cutoff radius is commonly used and the results can usually be generalised to the original system (Verlet, 1968).

The use of a cutoff radius allows the use of computational technique known as bounding boxes that is in common use and is described by Verlet (1967). A variant of this method is described here. By dividing the simulation space up into boxes of side length d each timestep, such as in figure 2.1, it is then clear that provided $d \geq r_c$ all the non-zero interactions on any particle in a box can be found in the interactions between particles internal to the box and interactions between each particle and the particles in all neighbouring boxes. The limit on d ensures that this area contains all particles within length r_c of any particle in the central box. The task of assigning particles to the box is of order N and provided r_c is significantly smaller than the size of the system this method significantly speeds computation.

2.6.2 Periodic Boundary Conditions

Periodic boundary conditions are commonly employed in MD-simulation. Essentially, even with the advances in computation it is still unfeasible to simulate a system with more than about few thousand or hundred thousand atoms. Unless what is of interest is the surface effects this does not model a macroscopic system which may have on the order of 10^{23} particles (Ercolessi, 1997) where edge effects are largely insignificant. The solution is periodic

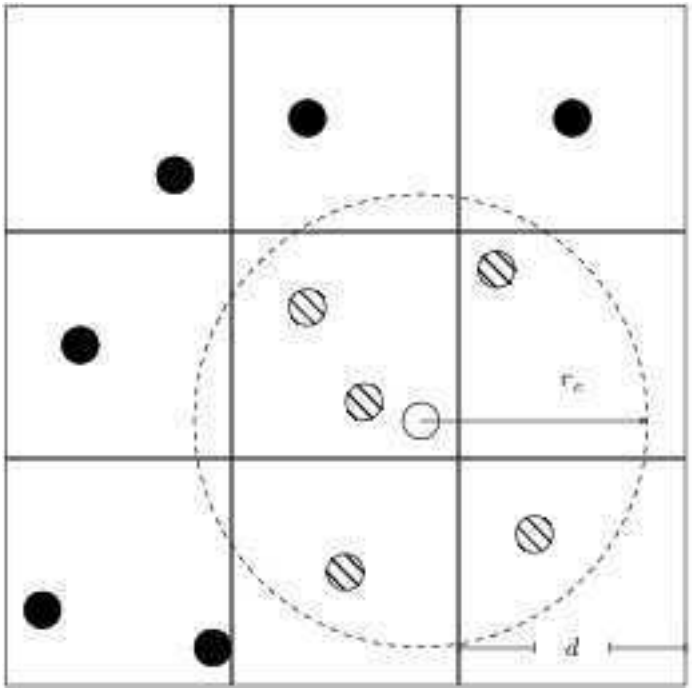


Figure 2.1: Bounding Boxes over a system

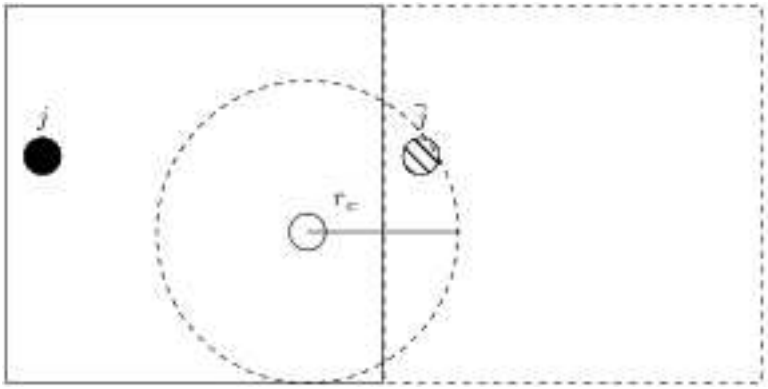


Figure 2.2: Periodic Boundary Conditions . Particle i interacts with the closest image of j

boundary conditions. By enclosing all the particles in a cube of side-length l and then viewing the system as this cube replicated infinitely many times in every direction then each particle i in the system represents and infinite set of particles at positions

$$\mathbf{r}_i + al\mathbf{i} + bl\mathbf{j} + cl\mathbf{k}, (a, b, c = -\infty, \infty) \tag{2.15}$$

where a, b, c are integers. Provided the $l \geq 2r_c$ then the interaction between each particle pair i and j is the interaction between i and the closest image j as seen in figure 2.2. The inequality $l \geq 2r_c$ is known as the minimum image criteria and ensures that only the closest image of j needs to be considered. Any particle exiting the cube is replaced by an identical particle entering the cube on the opposite face. Periodic boundary conditions need not be imposed in every direction.

2.7 Measurements of Statistical Properties

We describe the measurement of some of the statistical properties that are utilised in the simulations described later. For a more complete description of the possible properties that can be measured see Haile (1992).

2.7.1 Temperature

The total kinetic energy at a given timestep in the simulation can easily be calculated as (Ercolessi, 1997):

$$K = \frac{1}{2} \sum_i m_i |\mathbf{v}_i|^2 \tag{2.16}$$

where i is summed over all particles. Using the equipartition formula the temperature at a given timestep is given as:

$$T = \frac{2}{3} \frac{K}{N k_B} \tag{2.17}$$

2.7.2 Pressure

The pressure of the system can be calculated from the virial equation - a derivation can be found in Ercolessi (1997, p. 25) - which is:

$$P = \frac{N k_b T}{V} + \frac{1}{DV} \left\langle \sum_{i=1}^{N} \mathbf{r}_i \cdot \mathbf{F}_i \right\rangle \tag{2.18}$$

where N is the number of particles in the system, V the volume, k_B is Boltzmann's constant, $\mathbf{r}_i$ is the position of each particle, $\mathbf{F}_i$ is the total force acting on the particle and D is the number of degrees of freedom on the system. If the interactions are described by pairwise potentials then (2.18) can be written as:

$$P = \frac{Nk_bT}{V} + \frac{1}{DV}\left\langle \sum_i \sum_{j>i} \mathbf{r}_i \cdot \frac{d\phi}{dr}\Big|_{r_{ij}} \right\rangle \tag{2.19}$$

2.7.3 Space Dependant Variables

A simple technique utilised in this report is to divide the simulation space into n boxes. After first allowing a given number of timesteps in order for the system to reach equilibrium, statistical data is collected for each box. In this report, the average velocity and the total number of particles in each box was calculated and averaged each timestep. At the completion of the simulation this gives information on the density of particles and velocities as a function of position.

Chapter 3

Slip in Newtonian Fluids

3.1 Interfaces

Interfaces are the boundaries between different phases of a system. As one becomes interested in fluid flow through small channels the interface effects become of much more interest than for fluid flow in bulk liquids. An important interface property is the surface free energy. The purely thermodynamically definition of surface energy γ is (Aveyard & Haydon, 1973):

$$\gamma = \left(\frac{\partial F}{\partial \alpha}\right)_{T,V} \tag{3.1}$$

where F is the Helmholtz free energy, α is the the area of the surface with constant temperature and volume. The surface temperature is generally considered independant of the curvature of the surface but this is known to be false for curvatures with radius $r < 100$nm. The work done per unit

area in separating two surfaces i and j initially in contact and separating them outside the range of the intermolecular forces is given as (Rowlinson & Widom, 1982):

$$H_{ij} = \int_0^{d_{ij}} F(l)dl = -\frac{1}{4}\rho_i\rho_j \int_0^{d_{ij}} ru_ij(r)dr \tag{3.2}$$

where ρ_i, ρ_j are the densities, $u(l)$ is the intermolecular potential between particles of type i and j a distance l apart and d_{ij} is the distance where the intermolecular potential becomes neglible. If the work done in separation is given by H_{ij} and it produces two free surfaces the surface energy is given as:

$$\gamma_{ij} = \frac{1}{2}H_{ij} \tag{3.3}$$

For a drop of liquid in equilibrium with vapour on a surface (as seen in figure 3.1) Young's equation gives:

$$\gamma_{lv} = \gamma_{ls} + \gamma_{sv}cos\theta_c \tag{3.4}$$

where $\gamma_{lv}, \gamma_{ls}, \gamma_{sv}$ are the surface energies of the liquid-vapour, liquid-solid and solid-vapour interfaces, respectively, and θ_c is the contact angle which will be discussed shortly. The work in separating the liquid from the solid is given by (3.2). But in this case the separation does not produce a free surface so (3.3) is not valid. Instead, because in the separation two new surfaces of

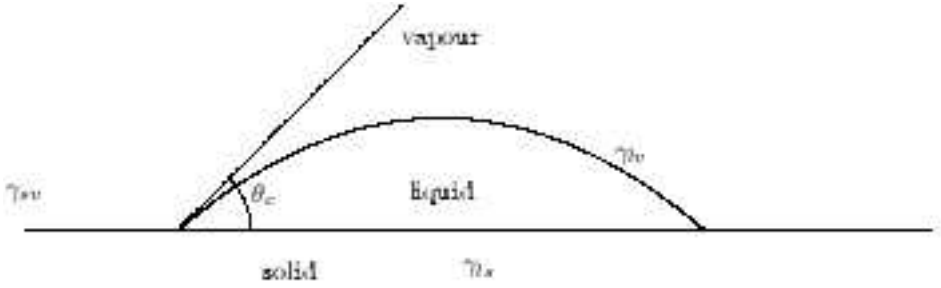


Figure 3.1: Liquid drop in equilibrium with vapour on a solid surface - adapted from Aveyard and Haydon (1973).

liquid-vapour and solid-vapour are formed thus the surface energy is given as:

$$\gamma_{ls} = \gamma_{lv} + \gamma_{sv} - H_{ls} \tag{3.5}$$

Wettability and Contact Angle

A measure of the wetting of a surface is the contact angle θ_c defined as the angle made between a surface and the interface of vapour and liquid phases in equilibrium on the surface - see figure 3.4. The angle can obviously only be between 0 and π. The closer to 0 the more wetting the surface is said to be. An angle of π is known as non-wetting while angles > 0 are known as partial wetting. The modified Lennard-Jones potential provides a convenient way of controlling the wetting of a surface. Using the definition of the LJ-potential

(2.6) and (3.2), (3.3), (3.4), (3.5) gives:

$$\cos\theta_c = -1 + 2\frac{\rho_f c_{fs}}{\rho_f c_{ff}} \tag{3.6}$$

where ρ_f is the density of the fluid and c_{fs}, c_{ff} are the fluid-solid and fluid-fluid interaction coefficients, respectively.

3.2 Boundary Slip Length

In solving the Navier-Stokes equations which are valid for viscous, incompressible fluids for fluid flow between two flat plates the boundary conditions for the velocity in the flow direction at the surfaces is given as (Hendy, Jasperse, & Burnell, n.d.):

$$u(\pm w) = +- \lambda\frac{\partial u}{\partial z}(\pm w) \tag{3.7}$$

where u is velocity in the flow direction, the two plates are separated along the z direction at positions $\pm w$ and λ has units of length and is known as the slip length. The slip length may be thought of as the distance below the surface where the no-slip condition is satisfied - see 3.2. This equation was introduced by Navier and is still used a today. It is not valid for a curved surface.

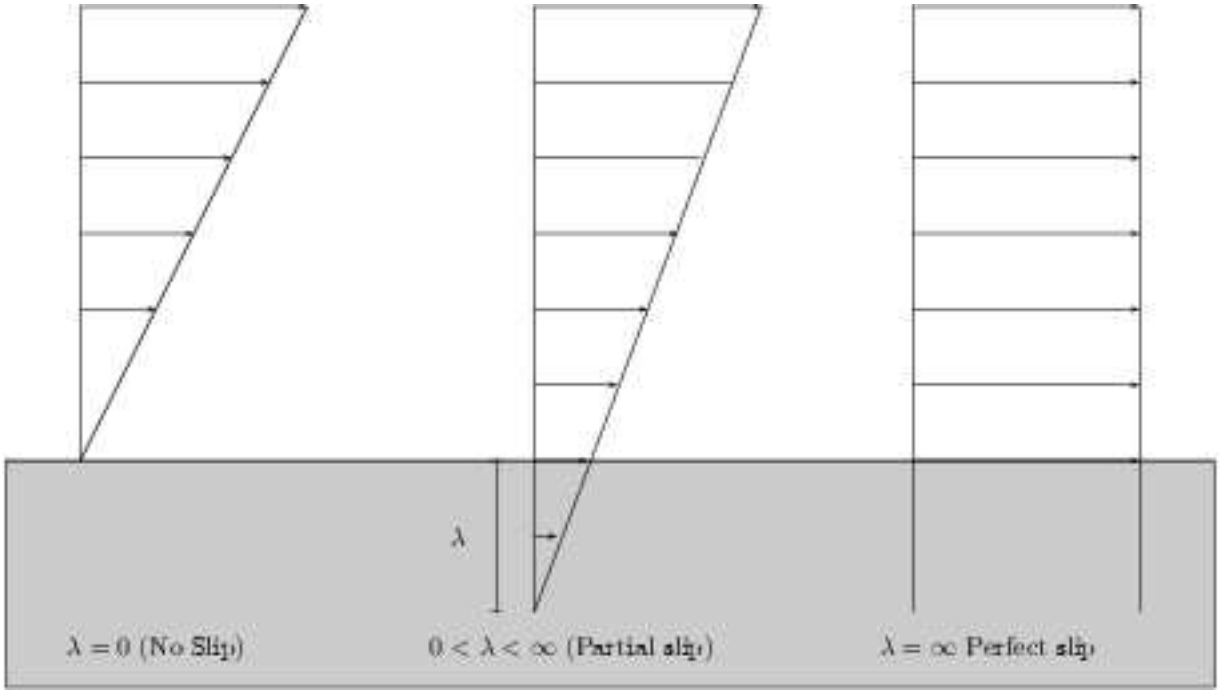


Figure 3.2: Interpretation of Slip Length - Adapted from Lauga et al. (2005)

3.2.1 Controlling the Slip Length

The slip length is of much interest. In order to control and study it is is necessary to know what factors in a system are important in determining the slip length. Obviously slip length is very related to fluid-surface interactions so as expected factors influencing this interaction will influence slip length. Lauga et al. (2005, p. 11) outlines several known factors affecting slip length. Surface roughness, roughness-induced dewetting, dissolved gases, shear rate, electrical properties, pressure gradients and surface wetting are all known to play a role. This report concentrates primarily on surface wetting.

In early eras, the slip condition was discussed and it became generally

accepted that the no-slip boundary condition ($\lambda = 0$) was true for Newtonian liquids. This assumption gave agreement with experimental results (Lauga et al., 2005). It is well known that non-Newtonian fluids such as polymer solutions have finite slip (Lauga et al., 2005). However, developments in experimental methods, new numerical and theoretical results along with an increasing interest with flow in small channels has reopened this ancient question of slip length in Newtonian liquids. Lauga et al. (2005, p. 3) provides useful clarification on the various meanings of the types of slip.

Slip is known to exist in non-Newtonian fluids, gas flow in devices with dimensions on the order of the mean free path and in contact line motion (Lauga et al., 2005). However, the experimental existence of slip and characterisation of it in Newtonian fluids is a current area of research. Lauga et al. (2005, p. 5) outlines several experimental methods that are used determine slip length. The most common are indirect methods - which do not directly measure the slip boundary condition but rather infer it from some other more macroscopic qualities of the system. Recent developments have also allowed local methods - which attempt to directly measure the boundary condition using such things as fluorescence cross-correlations or using small particles as tracers with optical techniques being used to measure the boundary condition. However, each method has limitations and it is not straightforward to measure boundary conditions.

3.2.2 Previous Studies on Varying Slip Length

A previous study (Barrat & Bocquet, 1999a, 1999b) used molecular dynamics simulation to observe the changes in slip length resulting from a partial-wetting situation in fluid flow between two parallel plates. A modified Lennard-Jones potential was used although a more complex method (Nijmeijer, Bruin, Bakker, & Leeuwen, 1990) than (3.6) was used to determine the wetting angle for each simulated potential. Barrat and Bocquet points out that most previous studies - experimental and theoretical - on slip lengths of confined liquids was with total wetting conditions. In their study MD simulations were performed for wetting angles of greater than 90°. The minimum pressure required to to force a fluid down the the non-wetting channel was calculated as:

$$P_0 = 2\left(\gamma_{FS} - \gamma_{SV}\right)/h \tag{3.8}$$

where γ_{FS}, γ_{SV} are the surface energies for the fluid-surface and surface-vapour interfaces and h is the width of the channel. Simulation of fluid between two infinite surfaces resulted in the profiles shown in Figure 3.3 showing density layering. Using both Poisueille and Couette methods flow was induced between the surfaces. A quadratic was fitted to the resulting flow velocity profiles to allow determination of slip. Large (over 15 molecular diameters) slip lengths were found for surfaces with high contact angles.

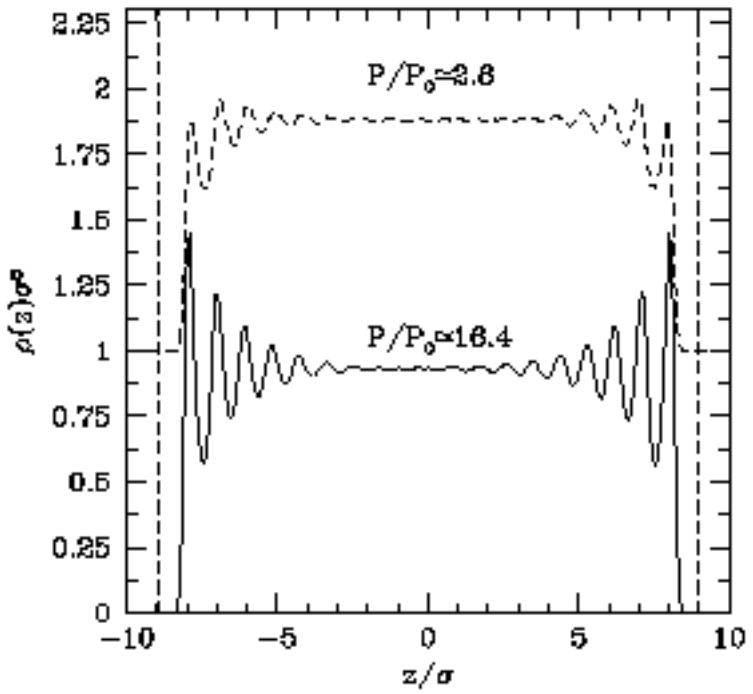


Figure 3.3: Density profile results from (Barrat & Bocquet, 1999a)

The effective permeability of a single capillary of diameter h with slip length σ was approximated as:

$$K_{eff} = K_0 \left(1 + 8\frac{\sigma}{h}\right) \tag{3.9}$$

where K_0 is the permeability of the pore of equal size with no slip. It was pointed out that in nanopores the effective slip length may be much larger than K_0. An approximate theoretical expression for the slip length was also derived and shown to be in good agreement with their results. This expression is not given here but gives low slip length for high pressures and also predicts slip length to decrease with increasing contact angle.

After first verifying some of the results (Jasperse, 2005) of Barrat and Bocquet, Hendy et al. (n.d.) considered a similar situation of fluid flow between parallel plates introducing the interesting idea of patterned variation in wettability of the surfaces. As mentioned earlier, it is expected that the variations in wettability would cause variations in slip length. Approximate solutions to the Navier-Stokes equations were found for the flow of Newtonian fluid with varying slip length and shown to induce a transverse flow that enhances the mixing of the fluid. Molecular dynamics simulations of these conditions were also performed and shown to be in good qualitative agreement with the previous theory although the simulations showed transverse velocites 2 to 3 times that predicted by theory. Hendy et al. also

mention that their results have application in microfluidic devices. In particular, there is interest in creating mixing in small channels and patterned wettability may have application.

Chapter 4

Simulation and Results

In this chapter I describe the simulations performed and outline results of interest. The starting point for the simulations were similar to those used by Barrat and Bocquet - namely fluid flow between two plates. The simulations were performed using a modified version of MDflow - as developed by Hendy et al..

4.1 Simulated Systems

The modified Lennard Jones potential was used (2.6) with a cutoff of 3σ. As is common the molecular diameter σ and energy ϵ were taken to be unity and from this a system of units is given. Also results reported here are given in such Lennard-Jones units and no attempt is made to transfer them to physical units - Jasperse (2005, p. 6) provides a basis for converting LJ units to estimated SI units. The simulations were integrated using the velocity

Verlet algorithm (2.12) using a timestep of $\Delta t = 0.01$LJu. All simulations were performed at a temperature of unity and pressure of about 1.7LJu. Each simulation was integrated for 5×10^4 timesteps to allow equilibriation and a further 5×10^4 timesteps where statistics were collected on the system. In all simulations a Poiseulle flow (see section 4.3) of 0.01LJu was generated in the positive x direction.

Two geometric systems were considered. In one system two surfaces of fixed atoms arranged in a face-centered cubic lattice of reduced density 0.9 (as per Barrat and Bocquet (1999b)) was filled with a liquid initially in a base-centered cubic configuration (as per Jasperse (2005)). However, a cylinder of fixed atoms along the z direction was placed at the origin of the system - see figure 4.1. The cylinder of radius r was generated by filling a box with a height equal to the space between the surfaces and width and length equal to $2r$ with a face-centered lattice. Then all atoms in this box outside radius r were eliminated. This method of constructing a cylinder may not be valid. At the dimensions that were simulated the cylinder radius was on the order of 3σ so the inherent roughness of the cylinder surface from the atomic size pits may be significant. Obviously, the fluid in the position occupied the cylinder was removed to prevent the system from exploding. Periodic boundary conditions were imposed in the x and y directions (the walls contains the system in the z direction) so in fact what is simulated is a fluid flow past a 'honeycomb' of cylinders as shown in figure 4.2. This

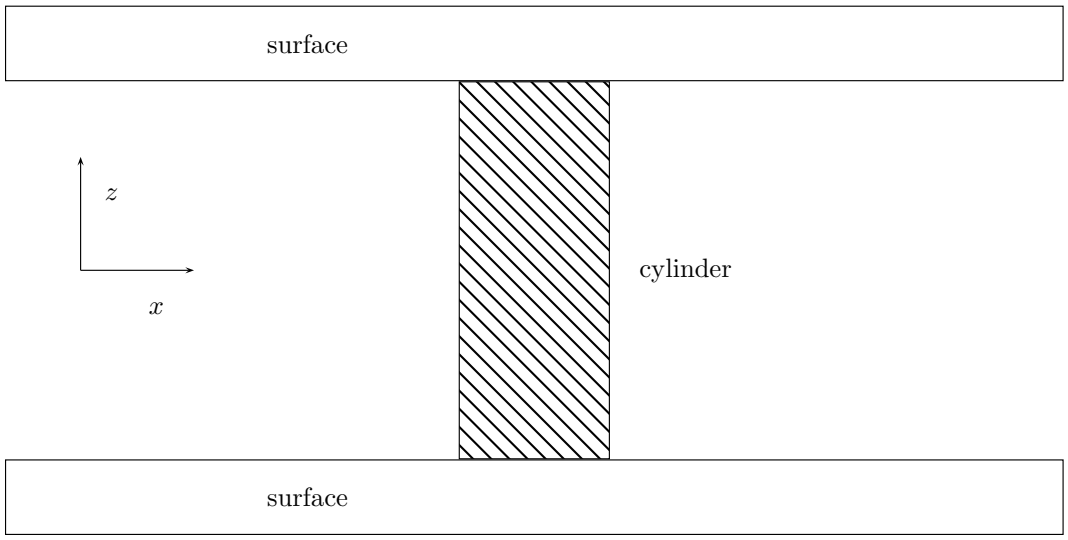


Figure 4.1: Fluid Flows in the positive x direction between two surfaces and obstructed by a centrally placed cylinder along the z axis

system will be referred to as the z-cylinder system.

The other geometric system considered was a similar configuration but with the cylinder being generated along the y direction in a similar manner. The periodic boundary conditions make this like flowing between two infinite surfaces but obstructed regularly with infinite cylinders along the y direction. This will be referred to as the y-cylinder system.

For the y-cylinder systems two simulations were performed. The interaction coefficients between fluid particles $c_{ff} = 1.2$ and the the $c_{fs} = 0.7$ for the fluid-surface interactions. In the first simulation the interaction coefficient between the cylinder atoms and fluid was set to $c_{fc} = 0.7$ as well. In

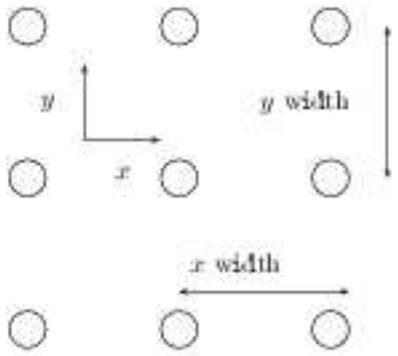


Figure 4.2: Periodic Boundary Conditions make the system equivalent to an infinite array of cylinders with fluid flowing past them

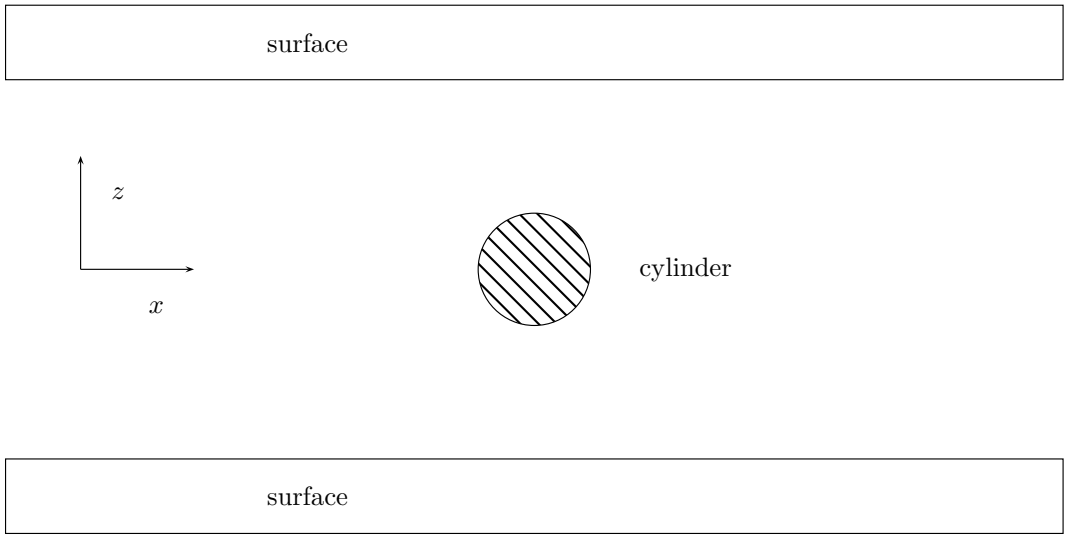


Figure 4.3: Fluid Flows in the positive x direction between the two surfaces and obstructed by a centrally place cylinder along the y axis

the second simulation the coefficient was set according to:

$$c_{fc} = \begin{cases} 0.9 & \text{if } z > 0 \\ 0.5 & \text{if } z \leq 0 \end{cases} \tag{4.1}$$

Of interest was any differences in flow caused by the varying wettability - particularly any increase in v_z that might cause increased mixing.

For the z-cylinder systems three simulations were performed. The interaction coefficients for fluid-fluid and fluid-surface particles was the same as for the y-cylinder system. The simulation was performed with $c_{fc} = 0.7$ and two further simulations were performed for c_{fc} controlled by:

$$c_{fc} = \begin{cases} 0.9 & \text{if } x > 0 \\ 0.5 & \text{if } x \leq 0 \end{cases} \tag{4.2}$$

$$c_{fc} = \begin{cases} 0.5 & \text{if } x > 0 \\ 0.9 & \text{if } x \leq 0 \end{cases} \tag{4.3}$$

Of interest was differences in flow caused by varying wettability.

4.2 Temperature Control

It is common in many types of MD-simulation to provide some means of regulating the tempature. In a perfectly energy conserving system this would not be necessary but because most integration algorithms are not

completely energy conserving it is often necessary to provide some means of keeping the tempature constant throughout the simulation. Also in the simulations described here there is additional energy loss as the surface atoms are non-moving thus interactions with the surfaces are not energy conserving. Jasperse (2005) found that if temperature was not regulated this effect would cause the liquid to freeze.

Temperature scaling is a simple method of maintaining tempature. In this method the temperature T at the end of each timestep is calculated using (2.17). The velocities of each atom are then rescaled to maintain a desired temperature T_0 using

$$\mathbf{v}_i' = \mathbf{v}_i \sqrt{\frac{T_0}{T}} \tag{4.4}$$

where $\mathbf{v}_i'$ is new velocity of atom i after rescaling the existing velocity $\mathbf{v}_i$.

While easy to implement velocity scaling does not correspond to any physically realisable situation. There are various alternative methods for temperature control. In this report the Nosé-Poincaré method was used as described in Bond, Leimkuhler, and Laird (1999). This method is the same as that used by Hendy et al..

4.3 Creating a Flow

In order to cause the fluid to flow through the system a pressure difference across the system generating a Poiseulle flow, or movement of the surfaces to generate a Couette flow is required. In this report only Poiseuille flows were generated. In solving (2.14) an extra term is added of $\frac{1}{2}f\Delta t$ with corresponds to applying a constant external force f to all fluid molecules. This is corresponds to a pressure gradient of:

$$\frac{dp}{dx} = f\rho_f \tag{4.5}$$

where ρ_f is the density of the fluid.

4.4 Results

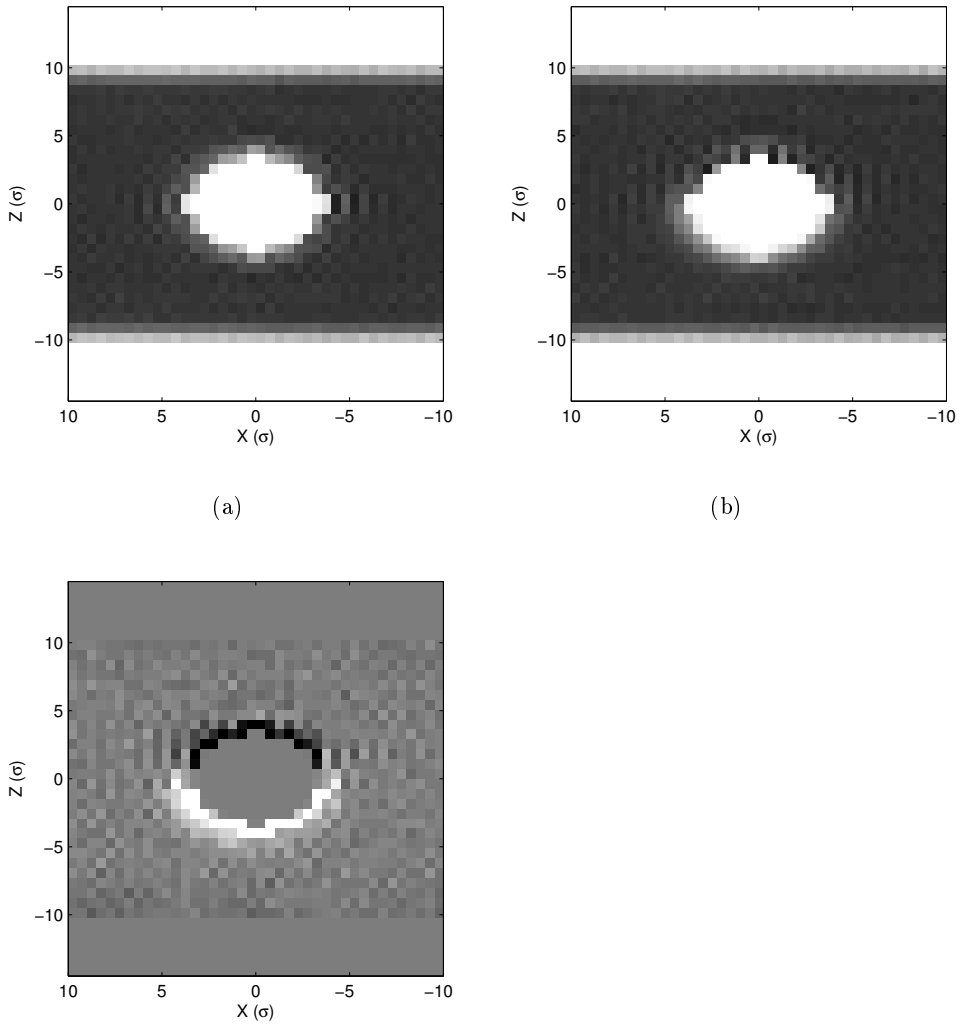
4.4.1 System with Y-Cylinder

Figure 4.4 indicates changes in density with wettability for the y-direction cylinder. As expected there is an increase in density near the wettable side of the cylinder when wettability is varied across the cylinder. However, there was no significant changes found in the velocities which appeared largly unaffected by the varying wettability. Figure 4.5 shows the velocities in the flow direction near the cylinder. There does not appear to be any systematic differences due to the wettability. Other plots of transverse velocities showed

no common trend of change between varying wettability.

4.4.2 System with Z-Cylinder

Figure 4.6 shows the variations in the average density of particles while modifying the wettability of the cylinder. As expected there is an increase in density near more wettable portions of the cylinder. In similarity with the y-cylinder system no significant changes in flow patterns were found with changes in wettability. No attempt was made to find slip lengths between cylinders. However, it is apparent from the velocity data that no significant variation in slip length between cylinders results from varying wettability. Figure 4.7 shows the flow velocity versus y position. Again no significant change is seen with varying wettability. No significant variation in flow velocities were found with varying wettability.



(a)

(b)

(c)

Figure 4.4: Plots showing the density of fluid particles for the y-cylinder systems as a function of x and z positions. The densities are averaged over the y dimension. Darker regions indicate areas of greater density. (a) indicates densities for a cylinder with uniform wettability. (b) indicates density for a cylinder with greater wettability for $z > 0$. (c) highlights the change between (a) and (b) with darker areas indicating areas where density increased and light areas indicating the reverse.

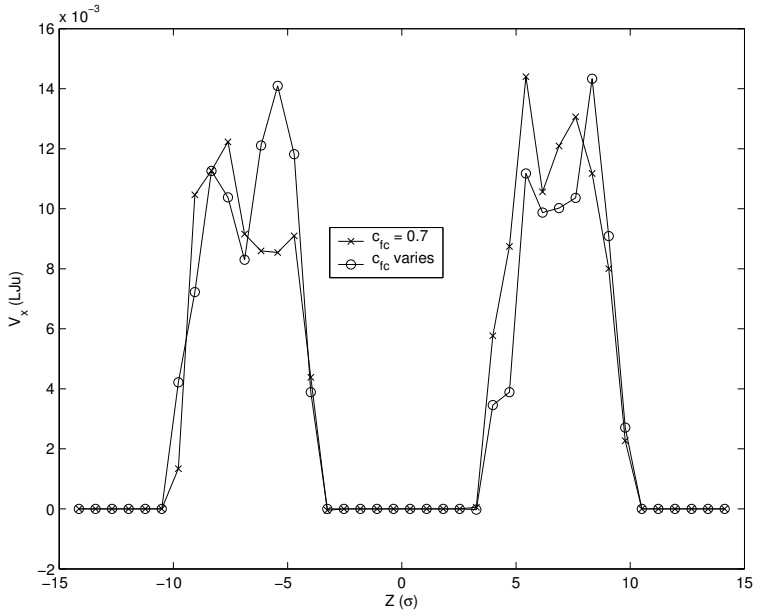
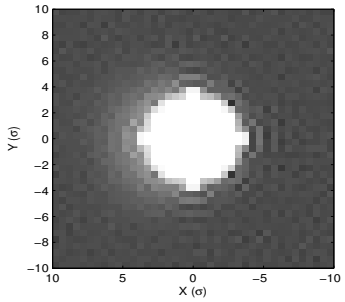
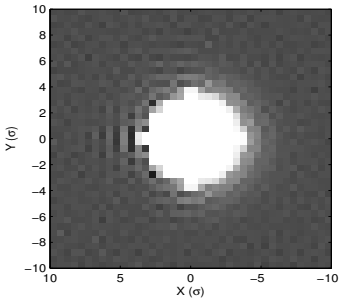


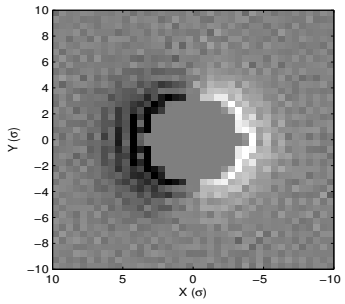
Figure 4.5: The average velocity in the flow direction versus z position as averaged across the y dimension for $x < 0.5$ and $x > -0.5$.



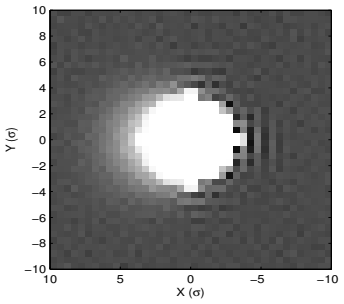
(a)



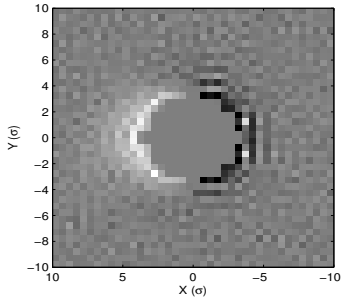
(b)



(c)



(d)



(e)

Figure 4.6: Plots showing the density of fluid particles for the z-cylinder systems as a function of x and y position. The densities are averaged over the z dimension. Darker regions indicate areas of greater density. (a) indicates densities for a cylinder with uniform wettability. (b) indicates density for a cylinder with a greater wettabilitiy for $x > 0$. (c) highlights the change between (a) and (b) with darker ares indicating areas where density increased and light areas indicating the reverse. (d) indicates density for a cylinder with a greater wettability for $x < 0$. (e) highlights the change between (a) and (d).

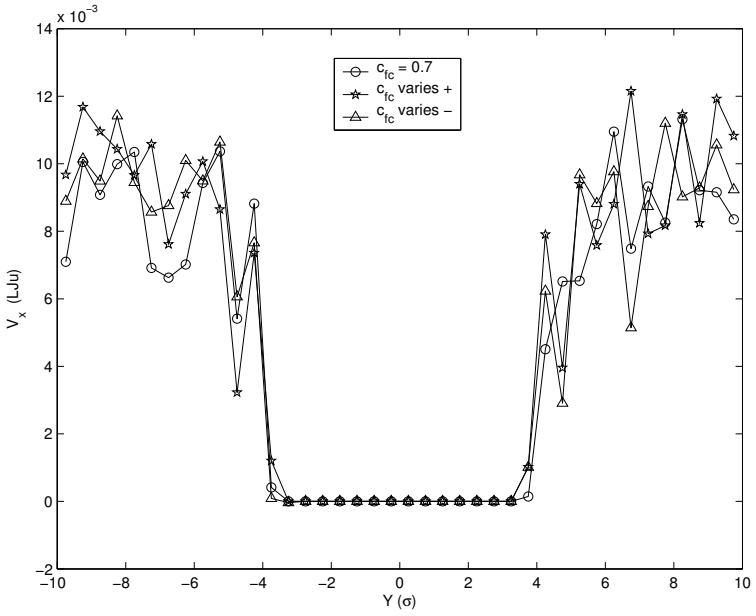


Figure 4.7: The average velocity in the flow direction as a function of y positions as averaged across the z dimension for $x < 0.5$ and $x > 0.5$

Chapter 5

Conclusion

It was known from previous studies that variations in wettability cause variations in slip length. It was hoped that in the two geometries introducing variatiations in wettability might cause transverse velocities, which would be useful for mixing or in other applications. However, in the two geometries considered variations in the wettability did not appear to cause significant variation in either transverse or flow velocities. This suggests that the two systems considered are not good choices for mixing. It is possible that further simulations corresponding to lower pressure flows may find different results as it was previously noted by (Barrat & Bocquet, 1999b) that the wettability had a more significant impact on slip lengths in lower pressure situations.

Acknowledgements

My thanks to Shawn Hendy for the suggestion of these problems and the introduction to fluid dynamics. Also for the use of the computer code for MDFlow written in conjunction with Martijn Jasperse. Thanks also to Ulrich Zuelicke for helpful insights and encouragement along the way.

References

Allen, M. P. (2004). Computational soft matter: From synthetic polymers to proteins, lecture notes. In N. Attig, K. Binder, H. Grubmüller, & K. Kremer (Eds.), (Vol. 23, p. 1-28). John von Neumann Institue for Computing. (Retrieved May 29th, 2005, from `http://www.fz-juelich.de/nic-series/volume23/allen.pdf`)

Allen, M. P., & Tildesley, D. J. (1989). *Computer simulation of liquids.* Oxford Science Publications.

Aveyard, R., & Haydon, D. A. (1973). *An introduction to the principles of surface chemistry.* Cambridge University Press.

Barrat, J.-L., & Bocquet, L. (1999a). Influence of wetting properties on hydrodynamic boundary conditions at a fluid/solid interface. *Faraday Discussions*, *112*, 119-127.

Barrat, J.-L., & Bocquet, L. (1999b). Large slip effect at a nonwetting fluid-solid interface. *Physical Review Letters*, *82*(23), 4671-4674.

Bond, S. D., Leimkuhler, B. J., & Laird, B. B. (1999). The Nosé-Poincaré

method for constant temperature molecular dynamics. *Journal of Computational Physics*, *151*, 114-134.

Coch, M., Evans, A., & Brunnschweiler, A. (2000). *Microfluidic technology and applications.* Research Studies Press Ltd.

Ercolessi, F. (1997, June). *A molecular dynamics primer.* International School for Advanced Studies, Trieste, Italy. (Retrieve April 5th, 2005 from `http://www.fisica.uniud.it/~ercolessi/md/md.pdf`)

Feynman, R. P. (1959). *There's plenty of room at the bottom.* Originally presented as a talk on December 29, 1959 at California Institute of Technology. (Retrieved 8th June, 2005, from http://www.feynman.com/)

Gear, C. W. (1971). *Numerical initial value problems in ordinary differential equations.* Prentice-Hall.

Haile, J. M. (1992). *Molecular dynamics simulation.* Wiley.

Hendy, S. C., Jasperse, M., & Burnell, J. (n.d.). *The effect of patterned slip on micro and nanofluidic flows.* (Accepted for publication in Physical Review E.)

The integrated circuit. (n.d.). (Retrieved 8th June 2005, from `http://nobelprize.org/physics/educational/integrated_circuit/history/`)

Jasperse, M. (2005, Febuary). *A molecular dynamics investigation of slip length* (Tech. Rep.). Industrial Research Ltd.

Lauga, E., Brenner, M. P., & Stone, H. A. (2005). Handbook of experimen-

tal fluid dynamics. In J. Foss, C. Tropea, & A. Yarin (Eds.), (chap. Microfluidics: The No-Slip Boundary Condition). Springer.

Nijmeijer, M., Bruin, C., Bakker, A., & Leeuwen, J. van. (1990). Wetting and drying. *Physical Review A*, *42*(10), 6052-6058.

Rahman, A. (1964). Correlations of motion of atoms in liquid argon. *Physical Review*, *136*(2A), A405-A411.

Rowlinson, J., & Widom, B. (1982). *Molecular theory of capillarity.* Oxford University Press.

Swope, W. C., Andersen, H. C., Berenes, P. H., & Wilson, K. R. (1981). A computer simulation method for the calculation of equilibrium constants for the formation of physical clusters of molecules: Application to small water clusters. *Journal of Chemical Physics*, *76*(1), 637-649.

Verlet, L. (1967). Computer "experiments" on classical fluids. i. thermodynamical properties of lennard-jones molecules. *Physical Review*, *159*(1), 98-103.

Verlet, L. (1968). Computer "experiments" on classical fluids. ii. equilibrium correlation functions. *Physical Review*, *165*(1), 201-214.

Xmol xyz file format. (n.d.). (Retrieved 1st June 2005, from `http://hackberry.chem.trinity.edu/IJC/Text/xmolxyz.html`)

Appendix A

User's Guide for MDFlow

This guide assumes some familiarity with Unix commands and installation procedures. If you are not sure of a step consult the appropriate documentation for your system. It also assumes some familarity with molecular dynamics simulation terms. If you are unsure try reading chapter 2.

A.1 Capabilities

MDFlow was originally written to simulate molecular dynamics flow between two plates of varying wettability. Although there was a parallel execution version written this guide is for the single processor version. Because of this is it not suitable for MD simulation of large systems. It was modified to support changes in the setup geometry and support for static multi-atom molecules was added although it does not yet allow the movement of molecules consisting of more than one atom.

It utilises the velocity Verlet algorithm (see section 2.5.2) to integrate the positions of the atoms which interact utilising the modified Lennard-Jones potential as described in section 2.4.1. It is not a general purpose fluid simulator and indeed would best be suited for simulating modifications on the simulations described in this report. Simulations are performed in two stages - first a equilibration stage where no statistics are collected and then a simulation stage where data is collected.

A.2 Requirements

This program was tested on several Linux systems - all based on the x86 architecture. Although there are no known reasons why it could not be compiled on other architectures it has not been tested. The program was compiled using the excellent Intel C++ Compiler.[1] Some problems were encountered using an earlier version (8.0) of the compiler so it is advised to use at least version 8.1. The GNU compiler could probably be used, however, initial tests showed the Intel compiler generated a significantly faster executable. Along with a working compiler and standard C++ library (including template classes) the program also makes use of GNU make[2],

[1] Obtainable from `http://www.intel.com/software/products/compilers/` and free for non-commercial use.

[2] Obtainable from `http://www.gnu.org/software/make/` under the GNU General Public License (GPL).

GNU Bison[3] and GNU Flex[4] and standard Unix shell scripts. Latex[5] is used to generate the documentation for the project so it is advised to install latex as well. MATLAB[6] is used for displaying the results of the molecular dynamics simulation. All the scripts relating to this are located in `matlab/`. There is also runtime requirement for a C preprocessor named `cpp` to be in the current path - the GNU or Intel C preprocessor are both acceptable here.

A.3 Installation

MDFlow is distributed as a gzipped tar file. Copy this file to the planned installation directory and execute the following commands:

```
$ tar xzf mdflow-[version].tar.gz
$ cd mdflow-[version]
$ make release
```

Provided you have all the requirements installed this should build a working executable named `flowsim-release` in the `bin/` directory.

A.4 Simulation Setup

To begin a simulation, first make a directory inside the mdflow directory. Before the simulation can be realized the initial layout of the system needs

[3] Obtainable from `http://www.gnu.org/software/bison/bison.html` under the GPL.
[4] Obtainable from `http://www.gnu.org/software/flex/` under the GPL.
[5] Obtainable from `http://www.latex-project.org/` under the Latex Public License.
[6] Commercial Software produced by MathWorks (`http://www.mathworks.com/`)

to be described. This is done using 4 files `fluidmol.xyz`, `solidmol.xyz`, `clustermol.xyz` and `input.sim`. MDFlow allows three types of molecules in the system - fluid, solid and clusters. The three xyz files should be in standard XMol xyz format described in *XMol XYZ File Format* (n.d.). The type of atoms making up the molecules are of no consequence, the file is simply used to the describe the positions of each atom in the molecule. If no xyz file is found for a given molecule type it is assumed the molecule is made up of a single atom and a warning is emitted.

The most important part of the simulation is the `input.sim` file which describes the parameters of the simulated system to mdflow. This file is first preprocessed using a C preprocessor so standard C preprocessor commands[7] are acceptable. Once preprocessed the input file is expected to consist only of blank lines and lines in the following format:

$$parameter = state$$

Below is an example input file

```
/**
 * Demonstration simulation file
 */
```

[7] See `http://gcc.gnu.org/onlinedocs/cpp/` for documentation on C preprocessor commands.

```
// Simulation parameters
      neq =  300    // Equilibration loops
      nsim = 600   // Simulation loops
      nsteps = 100 // Timesteps per loop
      surfmove = 0 // Enable surface movement
      rmax = 3.0   // Cutoff radius
      nose_coupling = 0.2  // Thermostat coupling


// Physical parameters
      temp0 = 1.0  // Desired temperature
      flowf = 0.01 // Poiseuille flow force
      surface_density = 0.9 //  Desired surface
         packing density
      fluid_density = 1.0 // Desire fluid packing
         density (can be used to control pressure)
      css = 1.0 //  Surface-surface interaction
         coefficient
      cfs = 0.7 //  Fluid-surface interaction
         coefficient
      cff = 1.2 //  Fluid-fluid interaction
         coefficient
```

```
        csc = 1.0 // Surface-cluster interaction
           coefficient
        cfc = 0.7 // Fluid-cluster interaction
           coefficient
        ccc = 1.0 // Cluster-cluster interaction
           coefficient

        cylinderdensity = 0.9

        cmoves = 0 // Clusters move
        cflows = 0 // Clusters flow (only makes sense
           if clustersmove =1

        vmode = 1 // Variation method ( 0=none, 1=step,
            2=sin )
        vphase = 0  // Phase shift between top and
           bottom pattern

// No patterning
        vamp_top = 0.0 // Deviation in fluid-surface
           coefficient on top surface
        vlambda_top = 20 //   Wavelength of coefficient
```

```
        variation on top surface

    vamp_bot = 0.0 //  Deviation in fluid-surface
       coefficient on bottom surface
    vlambda_bot =  20  //  Wavelength of
       coefficient variation on botton surface

    vamp_xcylinder = 0.2 // Deviation in cluster-
       fluid coefficient. Add to the +x half,
       subtracted from -x
    vamp_ycylinder = 0.0
    vamp_zcylinder = 0.0

// Geometry
    x = 30  //Channel depth (x)
    y = 30  // Channel width (y)
    z = 30   // Plate separation (z)
    plate_thickness = 2 //Plate thickness (layers)

    layout = 3 // 1 - random geometry, 2 - z-
       cylinder, 3 y-cylinder
    cylinderradius = 2
```

```
// Output
      equ_output_freq = 50 // Output a positions file
          every n timesteps during equilibration
      sim_output_freq = 50 // Output a positions file
          every n timesteps during simulation



      xbins = 60
      ybins = 60
      zbins = 50

      resume_postdata = 1  // When resuming either
         start a fresh (all zero version of
                        // the velocity, density
                           data or load the
                           version that was saved.
```

A.4.1 General Simulation Parameters Parameters

nsteps Number of timesteps per 'simulated' step. The internal timestep is set at $\Delta t = 0.01$. This parameter modifies how many internal timesteps are considered to make up one 'simulated' step. A value of 100 is suggested.

neq Number of 'simulated' steps to spend equilibriating the system.

nsim Number of 'simulated' steps to spend during the simulation.

equ_output_freq Every n timesteps during the equibriation phase a file is generated detailing all atom positions and velocities. This parameter sets n.

sim_output_freq Every n timesteps during the simulation phase a file is generated detailing all atom positions and velocities. This parameter sets n .

xbins,ybins,zbins During the simulation the velocity and density statistics are collected in the manner described in section 2.7.3. These three variables allow control of how many separate sections in the x,y and z directions, respectively that the statistics are collected for.

A.4.2 Geometric Simulation Parameters

These parameters affect the initial geometric setup. As mentioned earlier, MDFlow understands 3 types of molecules, surface molecules, fluid molecules

and cluster molecules. Note that any flow force specified will be in the x direction.

x,y,z These simulations govern the size of the simulated system. The z dimension specified will be enlarged slightly to provide room for the surface plates.

platethickness These specify how many atomic layers thick to make the surface plates. A warning will be generated if this is less than the cutoff radius.

surface_density Specifies the density of the face-centered cubic surfaces generated. A value of 1 is maximal density.

fluid_density Specifies the density of the base-centered cubic initially generated for the fluid positions. This can be modified to change the number of fluid molecules in the system and so the pressure of the system.

layout This specifies the layout type. A value of 1 gives random layout, 2 gives z cylinder and 3 gives y cylinder. These layouts are described below.

Random Layout

Random layout is a setting that is more useful for testing the system than actual simulation. At random cluster molecules are substitue for fluid molecules.

clusterdensity Controls the frequency of substitutions. Must be between 0 to 1 where 0 means no substitutions take place. Setting to 0 can be used to generate a system with no clusters.

Cylinder Layout

The cylinder layout is used to generate a central cylinder in the system of cluster molecules. The cylinder is a face-centrered cubic. A fuller description of the system is given in section 4.1.

cylinderradius This specifies the the number of atoms along the radius of the central cylinder.

cylinderdensity This specifies the density of the face-centered cubic generated for the cylinder. A value of 1 specifies a maximum density lattice and lower values specify lower density lattices.

A.4.3 Dynamic Parameters

temp0 Specifies the initial temperature and the desired temperature that the system is regulated to maintain using a Nosé-Poincaré thermostat (see section 4.2) in Lennard Jones units. Note that only the y direction velocities are regulated and equipartition of energy is relied upon to maintain the other temperatures.

nose_coupling Specifies the coupling of the thermostat.

flowf The flow force acting all fluid particles. In LJ units. See 4.3.

css The surface-surface interaction coefficient (for LJ potential).

cfs The fluid-surface interaction coefficient (for LJ potential).

cff The fluid-fluid interaction coefficient (for LJ potential).

cfc The fluid-cluster interaction coefficient (for LJ potential).

csc The surface-cluster interaction coefficient (for LJ potential).

ccc The cluster-cluster interaction coefficient (for LJ potential).

surfmoves If non-zero the surface molecules are permitted to move (not well-tested).

cmoves If non-zero the cluster molecules are permitted to move (not well-tested).

cflows If non-zero the cluster molecules also have the flow force applied to them. Obviously, this setting only makes sense if clusters are permitted to move.

The key capability of MDflow is the possibility of varying the interaction coefficients with position. The following settings allow this to be done.

vamp_(x,y,z)cylinder When layout is the cylinder type then interactions between clusters and the fluid are affected by this setting. In the x direction, any interaction between fluid and clusters, for clusters initially

in the positive x direction have this value added to the interaction coefficient, while the value is subtracted for clusters initially in the negative x direction. The other directions behave similarly.

vmode MDFlow also provides three possibles modes of varying the interaction coefficient of the surface and fluid with respect to z position. When 0 the coefficient is not varied. The following parameters control the variation:

vamp_(top,bottom) Specifies the maximum variation in the coefficient.

vlambda_(top,bottom) Specifies the wavelength of the variations in LJ units.

vphase Specifies the phase shift between variations on bottom surface and top surface.

A.5 Running the Simulation and Utilizing the Results

In general `flowsim` is not run directly - rather it is recommended the provided script `dosim` be utilized. This runs the simulation at lower priority to allow interactive processes to continue, logs to a file and ensures the simulation will not be interrupted if the hangup signal is received. Note: On receipt of a shutdown signal or `SIGUSR1`, and at the end of the equilibration and

simulation phases the programme will attempt to write the current state of the simulation to a file. This allows the simulation to be resumed at a later stage using `resumesim` with two arguments - the filename of the resume data and the number of steps to continue simulating. Other helpful commands are `stopsim` which attempts to stop the simulating in a manner that leaves a data file for resumation later and `stopsimnosav` which immediately stops the running simulation (any unsaved simulation will be lost).

Once the simulation is completed the log file `output.log` will contain transcripts of the simulation including temperature and pressure at each simulation step. The `postdata.dat` contains the average space-resolved data collected in the manner described in section 2.7.3. The format of the files is (per line):

```
(xpos) (ypos) (zpos) (vx) (vy) (vz) (d)
```

The positions give the center of the box where the average velocity per molecule (LJ units) and density (Given as number of particles in box) were collected and averaged with time. The file `matlab/loaddata.m` loads data from `postdata.dat` into a MATLAB structure. There are further scripts for displaying the data in various ways. However, since these routines are quite specific to the simulation they are documented only in source code comments and not detailed here.