Compressible Subsonic Flow on a Staggered Grid

by

Michael Patrick Bonner

B.Sc., California Polytechnic State University, San Luis Obispo, 2002

A THESIS SUBMITTED IN PARTIAL FULFILMENT OF
THE REQUIREMENTS FOR THE DEGREE OF

Master of Science

in

The Faculty of Graduate Studies

(Computer Science)

The University Of British Columbia

October, 2007

Abstract

This work focuses on numerically modelling the dynamics of a single phase fluid at varying densities and pressures. We explore the potential of incompressible flow simulation methods in modelling compressible flow, with an eye towards computer animation applications. The methods developed capture the interesting thermodynamic effects of compressible flow, and reduce to the standard Marker and Cell incompressible flow Poisson matrix in the incompressible limit. The method works well in modelling flows in the subsonic range that normal incompressible techniques do not capture and where compressible methods are inefficient. We have also investigated adapting these techniques to granular elastic-plastic flow.

Table of Contents

List of Figures

Chapter 1

Introduction - Background on Fluid and Material Simulation

This work focuses on modelling the dynamics of a single fluid flow. How fast a fluid is flowing determines how a fluid is modeled. Scientists use a dimensionless number called the Mach number, defined as

$$M = \frac{v}{c} \tag{1.1}$$

to quantify the speed of the flow (v) relative to the speed of sound (c) in the fluid. Depending on the Mach number desired to reproduce, completely different solution techniques tracking different properties may be required.

We can use the Mach number to classify three regions of flow. Flows with $M < 1$ are subsonic, $M = 1$ sonic, and flows with a Mach number greater than one are supersonic. Fluids begin to behave radically different when objects in the fluid or parts of the fluid approach or exceed the speed of sound in the fluid. The partial differential equations that govern the behaviour of each flow region may loosely be classified as elliptic, parabolic, and hyperbolic. For this work we are primarily interested in subsonic flow and elliptical PDEs, but it is important to note that higher speed flows require drastically different solution techniques. For this work we are concerned with a further delineation of subsonic flow, that between incompressible flow and compressible flow. Typically, flows with Mach numbers less than 0.3 can be modeled as incompressible, while flows with Mach numbers greater than 0.3 must be modeled as compressible. However, this is merely a rule of thumb; compressible effects have been noted with Mach numbers as low as 0.1, but it depends on changes in pressure and density relative to the speed of sound [15].

Although flows and solution methods can differ drastically they all start with the same underlying Navier-Stokes equations, defined continuously for a differential element. We begin with conservation of momentum, written

in non-conservative velocity form as

$$\frac{\partial u}{\partial t} + u \cdot \nabla u + \frac{1}{\rho}\nabla p - \frac{1}{\rho}\nabla \cdot \sigma_v - F_j = 0, \tag{1.2}$$

the continuity equation

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho u) = 0, \tag{1.3}$$

and conservation of energy

$$\rho(\frac{\partial E}{\partial t} + u \cdot \nabla E) - \nabla \cdot (K_h \nabla T) + p\nabla \cdot u = 0. \tag{1.4}$$

Here u is the velocity vector, ρ is density, p is pressure, σ_v is the viscous stress tensor, F represents body forces such as gravity, E is internal energy, and K_h is a heat conduction coefficient. Note that this system is not closed; an equation of state that relates two or more of the variables—such as pressure and density or pressure and temperature—like the ideal gas law, is needed. In many cases, certain terms in the equations can be neglected because their physical and/or numerical effects are negligible; later we will be dropping the viscous stress term and heat conduction term in particular, since numerical viscosity from the treatment of advection terms ($u \cdot \nabla u$ or $u \cdot \nabla E$) will dominate them.

Equations 1.2 through 1.4 are described in a continuous setting. In order to solve them we must discretize our computational domain in a discrete way and solve the equations at points in our domain. There are two ways to discretize the domain. The Lagrangian approach attaches a mesh or unstructured particles to the fluid, i.e. the discrete elements move with the fluid velocity. In the Eulerian approach the domain is voxelized into fixed discrete elements and fluid is free to move between them: fluid properties are tracked at nodes between cells or at cell centers. Lagrangian models are not as popular for fluids as they are for solids. Obviously a fluid can move and distort a great deal—if you imagine a mesh moving with a fluid, the mesh will become entangled beyond recognition. For this reason regular staggered or collocated grids, irregular grids, or adaptive grids are usually used to model fluids in an Eulerian setting. We will see in the subsequent sections that the type of grid used is also dependent on the type of flow being modeled.

Since we are interested in producing data sets to be used in production computer graphics environments, one of our most stringent constraints is time. We seek visually believable simulations as quickly as possible. In

contrast, engineers and physicists care that the simulation is correct, not that it just looks correct. So long as the results "look" correct, the accuracy that physicists and engineers obsess over is often sacrificed for improvements in efficiency and run time. We also need a complete representation of our domain; physicists and engineers will often take advantage of symmetry in order to reduce the problem size, figuring why simulate the entire domain if a subset will do? Also, the time frame in which important physics take place is usually not macroscopic as it is in a movie; nobody wants to animate something that is hardly perceivable. Therefore, scientific simulations usually span a much shorter time span than those needed for computer graphics applications which may run for several seconds. Conversely, physicists may be interested most in the first several milliseconds of an explosion.

We will now introduce the different flow regimes we are interested in and the modelling techniques that are used to model a particular flow regime.

1.1 Incompressible Flow

If the relative speeds within a flow are low enough (typically Mach number less than 0.3), thermodynamic effects and density changes due to changes in pressure become negligible. If density is constant and mass is conserved so is volume. This condition is expressed mathematically as the divergence of velocity is zero

$$\nabla \cdot u = 0 \tag{1.5}$$

Essentially what goes into a differential volume must exit it simultaneously. Coupling this equation with conservation of momentum 1.2 makes the system fully determined, without need of the energy equation or an equation of state, and yields extremely efficient simulations. Pioneering research in the computational modelling of incompressible flow began in the 1950's. The primitive variable approach[1] favoured in computer graphics was pioneered by Harlow and Welch who developed the Marker and Cell method (MAC) in 1965 [27], Chorin who developed the artificial compressibility method in 1967 [14], Patankar and Spalding who developed the semi-implicit method for pressure linked equations (SIMPLE) in 1972 [43], and Issa who developed the pressure implicit with splitting of operators (PISA) method in 1985 [31]. The computer graphics industry has primarily focused on the work of

[1] Primitive variable refers to using velocity and pressure as the chief unknowns, as opposed to other techniques such as vorticity methods; readers can consult [15] for a more in-depth consideration.

Harlow and Welch and the use of MAC grids. You will see in the subsequent chapter that it is ideally suited for efficient solutions of incompressible flow problems. MAC grids store the field variables in a staggered fashion, whereas collocated grids store all data at cell centers. Collocated methods for incompressible flow that use central differences are sensitive to checkerboard instabilities. This problem can be dealt with by using complicated filtering schemes, but they have the disadvantage of adding additional numerical dissipation on coarse grids. Therefore, collocated methods are not popularly used for applications in computer graphics. MAC methods typically treat the advection term explicitly but the pressure term as an implicit constraint—and thus can use time steps restricted by the material velocity only, not the speed of sound (which, for low Mach numbers, is obviously desirable).

A vast amount of research in graphics has started with simple incompressible flow and tweaked it with the goal that some physical process which is normally difficult to model can be approximated: this is the general spirit behind this work. Starting with Foster and Metaxes in 1996 [24], the amount of graphics research in modelling the Navier-Stokes with incompressible flow has included semi-Lagrangian advection [49], vorticity confinement [20, 41], surface tracking [19], [18], [7], surface tension [33], surface flow [50], dynamic meshes [32], fire [41], viscosity [12, 45], viscoelastic modelling [25], coupling with rigid and deformable solids [26], efficient implementations [33], [30], smooth-particle hydrodynamics ([38, 48], irregular boundaries [9], multiple fluid flow [34], large bodies of water [35], and of course, explosions [22, 40, 46, 54] , and sand [10, 56].

1.2 Compressible Flow

While in low speed flows, temperature, energy, and changes in density may be insignificant, and thus efficient incompressible flow solvers can be used, for higher speed scenarios more complex compressible methods must be used. Consider a small element of fluid, v, under pressure, p, from all sides. If we increase the pressure by a tiny amount, dp, the element will compress by a corresponding amount, $d\rho$. The compressibility of a fluid is defined as $\tau = \frac{d\rho}{\rho dp}$. Compressibility measures how much a static material deforms subjected to a unit of pressure. Under a change in pressure the corresponding change in density is then

$$d\rho = \rho \tau dp \tag{1.6}$$

Compressibility is a material property; water has a value of $5 \times 10^{-}10$ while air has a value of $1 \times 10^{-}5$ [1]. Air is easier to compress than water—not surprising. Things become interesting as the fluid is subjected to forces or loads and begins to move. It can be seen plainly in equation 1.6 that if large pressure changes are present, than the change in density is significant. This is usually the case for flows with Mach speeds greater than 0.3, which in air corresponds to a flow speed of $100\frac{m}{s}$. Further flow classification begins within the envelope of compressible flow: subsonic flow ($0.3 \geq M \geq 0.8$), transonic flow ($0.8 \geq M \geq 1.2$), supersonic flow ($1.2 \geq M \geq 5$), and hypersonic ($M > 5$). Each region warrants special consideration and may benefit from specialized methods. Typically in all cases, in order to accurately capture the interesting effects of compressible flow, the conservative variables ρ, ρu, and ρE, are used—in fact it's generally considered essential to use these variables, and the conservative form of the equations, to capture shock waves at transonic and higher speeds. In contrast, incompressible methods track the non-conservative variables p and v; while the focus of this thesis will be on modelling the compressible effects present in sub-sonic flow that incompressible simulators cannot approximate, we will find that the non-conservative variables will do fine.

In order to take into account the compressibility and capture the variations in density we now need all of the equations listed in the previous chapter (1.2 to 1.4)[2] and an equation of state, such as the ideal gas law ($pv = nRT$), that relates pressure, density, and temperature[3].

As airplanes approached the speed of sound and as scientists began to model explosions accurately computational fluid dynamics has evolved from modelling incompressible flows to modelling fully compressible fluid dynamics. In supersonic flows containing shocks, information about conditions downstream of a shock cannot propagate back upstream past the shock; this effect produces large gradients in energy and pressure. Understandably higher resolution solution techniques are needed. Currently, two methods for solving the hyperbolic partial differential equation conservation laws involved in compressible flow are commonly used. The first and most common are Riemann solvers. Riemann solvers capture shock formation and evolution by solving simplified scalar conservation laws with constants for $x < 0$ and $x > 0$ (the shock face or front is located at $x = 0$). In order to reduce the PDEs to separate scalar equations the variables must be collocated so that

[2]Though as noted modified to use the conservative variables, and written in "conservation law" form.

[3]This is the traditional approach. We will see later other approaches exist, although they may not be as physically accurate.

the flux matrix can be diagonalized. The second and less popular approach, but still extensively researched, is epitomized by the Lax-Friedrichs method. Variables are again collocated but every other time step is staggered by half a grid cell. The staggering allows central differences to be used without concerns about stability, so long as a CFL-like condition is obeyed. Lax-Friedrichs methods avoid Riemann problems and do not require the matrix to be diagonalized. The latter method is closest in spirit to the approach taken in this thesis.

As mentioned before, incompressible flow methods employ operator splitting to separate an explicit and implicit part of the solution. This allows much larger time steps to be used which means simulations can be produced quicker. This is a huge advantage in a production computer graphics environment. In contrast, the Riemann solvers and Lax-Friedrichs methods use fully explicit methods. In supersonic flows the time step restrictions are from the material velocity, not from the speed of sound. There is no reason to treat the advection and pressure separately. However, since these compressible methods are explicit, the conservation equations are very stiff in the low speed limit. The usual solution for stiffness is to treat pressure implicitly, but because the spatial discretization is highly non-linear and potentially non-smooth, doing this for a Riemann solver is extremely difficult. Lax-Friedrichs type methods also do not fare well with this flow regime and have a similar stiffness problem—but since the space and time discretizations are tightly integrated in staggering, there is no obvious way to make these methods implicit. On the other hand, incompressible solvers cannot model flows where the divergence of velocity is not zero. Another concern when modelling compressible flow is that a region of the flow will be moving slow enough to be theoretically incompressible. When the difference between flow speeds and the speed of sound is very large it can lead to a poorly conditioned system which may not converge using iterative procedures. This problem has been extensively researched in [44], [13], and elsewhere. In fact flowfield-dependent variation methods have been developed motivated by the difficulty in modelling very low velocities on one side of a shock and the very high velocities on the other side of a shock.

In contrast to the large amount of work done in the computer graphics community on incompressible flow, compressible flow and the conservative Navier-Stokes system has received little attention. A surprising fact noting the prevalence of explosions in movies and video games. Yngve el al. [55] have produced the most notable work modelling the full compressibility of a shocked fluid. The work used two simplifying assumptions used by engineers and although they did capture the effects of temperature and the

conservation of internal energy, the method is inefficient as the flow speed decreases [22].

The methods developed for this thesis model both low and relatively high speed flows and capture some of the visual effects of compressible flow. Because neither standard compressible or incompressible techniques handle flow speeds in the sub-sonic range above $M = 0.3$ we have aimed our research at this flow regime and are not primarily concerned with conservation of energy and momentum or with modelling shock creation and propagation. Specifically, we will never attempt to conserve energy, instead we make simplifying assumptions that allow us to forgo energy conservation and to provide a robust and quick method.

1.3 Elasto-Plastic Flow

Another type of flow we are interested in, elasto-plastic flow, isn't actually fluid flow. In particular we are interested in the flow of sand. The dynamic behaviour of sand under load resembles the behaviour of a fluid, but, unlike a fluid and more like a classic elastic-plastic material such as steel, sand stops flowing—it has a strength associated with it. Classic elastic-plastic materials will yield or deform under load. If the strength of the material is not exceeded, the deformation or *strain*, is recoverable: upon unloading the material will return to its initial configuration. If the material's strength is exceeded, it deforms plastically—plastic strain is not recoverable. If we consider a pile of sand we know that if it is too steep it will fail and flow—some sort of sand inter-grain strength has been exceeded—and it will continue to flow until it reaches a stable configuration. Once the grains move, that "deformation" is not recoverable: they will not return to their original configuration. This fact makes classic plasticity an obvious choice for modelling the dynamics of granular materials—clearly, regular fluids do not behave this way. Elastic-plastic techniques like those in [39] were developed by adapting known methods for dealing with classic problems in solid mechanics to things that flow like sand. Unfortunately, the deformations seen in typical solid mechanics problems are relatively small, and thus numerical work has focused on mesh-based Lagrangian techniques—which suffer from entanglement and require re-meshing in order to handle realistic sand flows. For this reason, Eulerian techniques, which are underdeveloped, are very attractive for sand.

Continuum methods that calculate the dynamics of granular materials are very similar to the compressible and incompressible fluid dynamic models

mentioned in the previous sections[4]. Instead of solving for v and p or ρ, ρu, and ρE, they solve for v, ϵ (strain), and σ (stress). The models used consist of partial differential equation conservative laws for mass, energy and momentum. The most prominent differences with the Navier-Stokes system of equations we have seen are present in the conservation of momentum, which now relate stress and strain. One models used is

$$T_{i,j} = \sigma(\delta_{i,j} + k\frac{V_{i,j}}{|V|})$$

where $V_{i,j} = -(\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y})$ is the strain rate tensor[5], $|V|^2 \equiv \sum V_{i,j}^2$, k is a material constant, and δ is the angle of internal friction. This equation, and others like it, essentially replaces the viscous term in the Navier-Stokes equation with a shear rate independent term. This rate independence makes these equations generally more difficult to solve than the Navier-Stokes equation. These conservative laws are combined with a yield condition such as Mohr-Coulomb or Drucker-Prager and a flow rule. Normal plastic materials flow in a direction in strain-space that corresponds with the gradient of the yield condition (a so-called "associated" flow rule); however, doing this would make the dilation part of plastic strain to be non-zero, an unphysical case for granular dynamics, and thus a more complicated "non-associative" flow rule must be used.

Another popular technique for modelling elastic-plastic flow of sand is to use a discrete element method (DEM). DEMs use Newton's law to model the interaction of individual grains or clumps of sand. The simulations progress by detecting "contacts" between particles and then calculating the resultant forces. The proper and efficient detection of contacts is key to the success of discrete element methods. Even with an optimal technique, DEMs track individual particles; with sand this can means tracking the interaction of millions of grains of sand, clearly, a daunting computational task. In applications where knowing the exact location of particles is paramount this expense is worthwhile; this is not the case with computer graphics and DEM methods are often too costly. Bell et. al. used discrete elements in [10] specifically aimed at producing effects for movies, but their sand grains appear too large, most likely the effect of trying to reduce the problem to a manageable size.

[4] At least, they are similar to fluids when considering low-speed, high-density flows; in contrast rapid "gaseous" flows are modeled using kinetic theory methods.

[5] The negative sign in the strain rate tensor is there because sand spreads apart when subjected to a tensile load.

Both elastic-plastic and discrete element techniques require more resources and are significantly more computationally complex than incompressible simulation methods. Treating the dynamics of flowing sand using an incompressible flow technique is an attractive alternative; the work of Zhu and Bridson [56] was in this spirit. Although the method is more efficient than competing techniques and does produce realistic simulations in many cases, for some basic granular physics their results are incorrect. In particular, the effects of internal friction are not accounted for in the pressure calculation and could lead to inconsistencies when pressure is corrected to account for zero divergence. Additionally, cohesion is handled incorrectly; when subjected to a tensile load the grains of sand do not come apart as in real life. **In Chapter Four we introduce a method that attempts to improve Zhu's work and ultimately move toward a quick and accurate Eulerian granular dynamics model.**

We have introduced three types of fluid dynamics; modelling each traditionally requires wildly different techniques. In order to capture the physics of compressible subsonic and elastic-plastic flow we have modified the simplest and most efficient techniques, those used to model incompressible flow—while maintaining the ability to accurately and efficiently model incompressible flow.

Chapter 2

Review of Incompressible Flow Simulation

We mentioned before that if the relative speed of flow is low enough temperature and pressure effects on density can be neglected. Essentially, density can be regarded as constant throughout our fluid. This allows us to subject our system to the following constraint:

$$\nabla \cdot u = 0 \tag{2.1}$$

$\nabla\cdot$ is the divergence operator. This equation simply states that all points in our velocity field must be divergence-free. In simpler terms, what goes into a point must go out and vice versa.

Assuming that a flow is incompressible allows density to be treated as a constant—this eliminates the need to use the continuity equation. Furthermore, as we will see in the following sections viscosity can also be neglected. This essentially leads us with equation 1.2 and 2.1; we will see how these equations are combined to model fluid dynamics efficiently.

2.1 The Marker and Cell (MAC) Grid

The Navier-Stokes system of equations are defined continuously for all points within a volume of fluid. Practically, this is not solvable; it becomes necessary to solve the equations using finite differences defined on a Eulerian grid where our domain is divided into a finite set of voxels of equal volume. The Navier-Stokes PDEs are sampled at cell centers and cell faces in the domain. For reasons that will be clear later we will use a Marker and Cell (MAC) grid [27]. A two-dimensional MAC grid cell (figure 2.1) defines the x component of a cell's velocity vector on the vertical faces of the cell and the y component on the horizontal faces. Pressure and density are defined at the center of the MAC cells. We use a half index notation, so for cell (i, j) in two dimensions we have $\rho_{i,j}$ and $p_{i,j}$ at the center, and $u_{i-\frac{1}{2},j}$, $u_{i+\frac{1}{2},j}$, $v_{i,j-\frac{1}{2}}$,

and $v_{i,j+\frac{1}{2}}$ normal to the faces. Most of the graphics research mentioned in the incompressible section of Chapter One uses a staggered MAC grid.

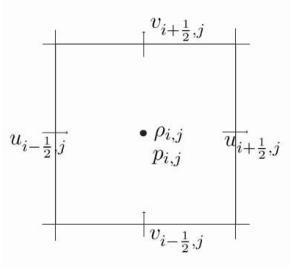

Figure 2.1: A 2D Marker and Cell (MAC) cell with density and pressure defined at the cell center and the vertical component of velocity defined on horizontal faces and the horizontal component of velocity defined on the vertical cell faces.

2.2 Operator Splitting

In order to solve our PDEs on our grid we need to introduce the idea of operator splitting. Consider the continuity equation: $\rho_t + \nabla \cdot (\rho u) = 0$. Expanding the $\nabla\cdot$ operator yields $\rho\nabla\cdot u + u\nabla\rho$. The idea of operator splitting is just substitution: solve for the $u\nabla\rho$ as $\tilde{\rho}_t = -u\nabla\rho$ and then solve $\rho_t = \tilde{\rho} - \rho\nabla \cdot u$. We do this with equation 1.2 and equation 1 by solving for the advection term first, that is $\tilde{u}_t = -u\nabla u$. In the momentum equation it is advantageous to treat the advection part of the equation with an explicit technique, for reasons of efficiency, accuracy, and simplicity [11]. We will see in the subsequent sections that the advection algorithm commonly used behaves well for arbitrarily large time steps. Treating the pressure term in the momentum equation implicitly produces a well conditioned system. If in contrast the pressure term were to be dealt with explicitly, the CFL condition[6] would require that the speed of sound be taken into account when determining a safe time step—doing so is extremely inefficient for low speed flow.

[6]The CFL condition will be defined later, but for now it is important to know that it is a restriction on the size of the time steps taken during the simulation.

2.3 Semi-Lagrangian Advection

In graphics a first order accurate[7] technique called semi-Lagrangian advection [49] is used to determine the advection term in our conservation of momentum equation. We are seeking the value or quantity, in this case u, that will be present at this quantity's current location at the end of the current time step. This is done by determining a velocity vector at our position x (interpolated from nearby u and v velocity components). The vector is then negated and multiplied by Δt. We trace back to the location $(x - \Delta t\, \vec{v})$ where we estimate the fluid came from in our domain, and then interpolate (linearly, bi-linearly, and tri-linearly in 1d, 2d, and 3d) from nearby values to determine our final value (see figure 2.2 for an illustration of this process). This technique is stable with arbitrarily large time steps—a great quality for a computer graphics application. Unfortunately, this process involves averaging operations that have the effect of smoothing out high frequency (or sharp) features in our data set. In some situation this is acceptable because it can be shown to be mathematically similar to viscosity [11]. The excessive dissipation has a positive side effect: we are now justified in dropping the $\nu \nabla \cdot \nabla u$ term from our momentum equation. Semi-Lagrangian advection still damps out too much detail for many applications, which has prompted the research in more complex methods, e.g. [20, 41].

2.4 Operator Discretization

Before we consider solutions to our PDEs we need to look at the discretization of the gradient operator (∇p) and the divergence operator ($\nabla \cdot u$) on a MAC grid. Since our velocities are stored at cell faces and pressure at cell centers it is easy to solve for the pressure gradient across a cell face. If we had used a collocated grid we could have used a second order accurate central scheme such as $\nabla p = \frac{p_{i+1} - p_{i-1}}{2\Delta x}$. Unfortunately, this scheme ignores information stored at location i (were we are calculating ∇p and u_t) which can lead to poor numerical results—known as a checkerboard instability [11]. The staggered positions of our unknowns in the MAC grid yield a second order accurate central scheme without introducing inaccuracy or instability. Ultimately the gradient is approximated as $\nabla p = \frac{p_{i+1} - p_i}{\Delta x}$. Similarly, if we consider our constraint $\nabla \cdot u = 0$ for cell i, this equation in two dimensions is

[7] A second order accurate version has been developed, but the simple version is presented here.

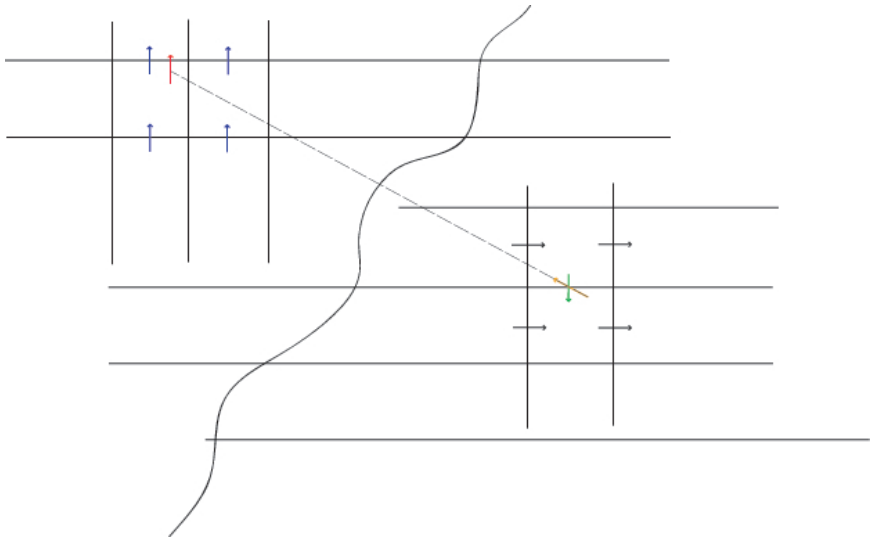

Figure 2.2: Illustration of semi-Lagrangian advection for a horizontal velocity unknown.

$$\nabla \cdot u_i = (u_{i-\frac{1}{2}} - u_{i+\frac{1}{2}} + v_{i-\frac{1}{2}} - v_{i+\frac{1}{2}})/\Delta x \tag{2.2}$$

which, like the gradient operator, is second-order accurate. It can be shown that if the divergence is calculated on a collocated grid the same problems present with the pressure gradient arise.

2.5 Boundary Conditions

The most simple and most common boundary conditions used in computer graphics simulations are solid wall boundaries and free surface boundaries. In the inviscid case solid wall boundaries involve simply setting the normal velocity on a boundary to zero or $\vec{u} \cdot \hat{n} = 0$. This is trivial for grid aligned solids on a MAC grid since velocities are stored perpendicular to cell faces. The issue of accurately handling curved un-grid aligned boundaries is more difficult, but recent work by Batty et. al. in [9], in which the pressure projection step is rephrased as a kinetic energy minimization, has proven to be accurate and efficient. In computer graphics, air is usually not modeled since it is so much lighter than water and has little effect on the dynamics of our fluid flow. Instead of actually calculating the pressures in the air it

is taken to be at atmospheric pressure, or, since only differences in pressure are important, simply zero. This is the free surface boundary condition.

2.6 Finite Difference Solution

As was mentioned before, density is constant and the continuity equation can be neglected. If we use a forward Euler approximation to u_t, substitute in $\tilde{u}$ (section 2.3), and neglect viscosity (section 2.3), the conservation of momentum equation becomes[8]

$$u^{n+1} = \tilde{u} + \frac{1}{\rho}\nabla p \tag{2.3}$$

Now applying the constraint $\nabla \cdot u = 0$ and re-arranging yields:

$$\nabla \cdot u^{n+1} = \nabla \cdot \tilde{u} + \frac{1}{\rho}\nabla \cdot \nabla p$$

$$\frac{1}{\rho}\nabla \cdot \nabla p = \nabla \cdot \tilde{u} \tag{2.4}$$

We now have a system of linear equations in pressure, p, involving the Laplacian $\nabla \cdot \nabla$ or ∇^2, the divergence of the gradient of pressure. Once discretized as described above, a single row in the linear system[9] looks like

$$\frac{\delta t^2}{\Delta x^2 \rho_i}\big(- p_{i,j-1} - p_{i-1,j} + 4p_{i,j} - p_{i+1,j} + p_{i,j+1}\big) = (\tilde{u}_{i-\frac{1}{2},j} - \tilde{u}_{i+\frac{1}{2},j} + \tilde{v}_{i,j-\frac{1}{2}} - \tilde{v}_{i,j+\frac{1}{2}})/\Delta x \tag{2.5}$$

All other entries in the ith row are zero—it is a sparse system of equations. The Laplacian operator discretized using our MAC grid is symmetric positive semidefinite[10], and in fact, if at least one pressure value is constrained to zero, positive definite. Moreover it is a diagonally dominant M-matrix, i.e. the off-diagonal entries are non-positive and the row-sums are non-negative. Large amounts of research have gone into solving this particular system and

[8] In order to ensure the final system is symmetric the equation has been divided through by ρ.

[9] Note that we have multiplied both sides of 2.4 by -1 to make the matrix positive definite. A matrix is positive definite (PD) if $x^T Ax > 0 \; \forall \; x \in R^n$.

[10] A matrix is positive semidefinite (PSD) if $x^T Ax \geq 0 \; \forall \; x \in R^n$.

systems like it very quickly. We solve for p and then correct the intermediate (advection only) velocities according to $u^{n+1} = \tilde{u} - \nabla p$.

The standard algorithm proceeds by advecting velocities and other quantities according to section 2.3; solving for pressure according to 2.5; and enforcing boundary conditions and updating secondary quantities with the new velocity field.

As was seen in the previous sections the amount of computer graphics research in incompressible flow is staggering (as referenced in section 1.1). However, the question of modelling explosions well is still largely open and the question of modelling granular flow has been largely untouched. The following chapters will present methods that use a MAC grid, conservation of momentum, and rather than a divergence restraint, include conservation of mass, and an equation of state to close the Navier-Stokes system of equations. We feel that this approach allows us to quickly and efficiently capture some important aspects of compressible and granular flow.

Chapter 3

Compressible Flow on Staggered Grids

Our hope in modelling compressible flow on a staggered grid is to capture some of the visual effects of thermodynamics while maintaining the computational efficiency of incompressible methods. Explosions produce the dynamic loads required to make density changes visually important. Significant effort has been spent in the computer graphics community in modelling explosions; that effort has almost exclusively focused on modifying incompressible methods. Many production applications use heuristic rules; many physics based techniques replace or adapt a heuristic with more physically motivated heuristics. Starting with Reeves in 1983 [47] particle systems have been used exhaustively to model fire effects and explosions in computer graphics. Mazarak et. al. [37], Martins et. al. [36], and Neff [40] used analytical approximations to experimental data to model the blast wave associated with an explosion. Feldman et. al. [22] model the blast wave using incompressible techniques but place a constraint on the divergence of velocity, that is $\nabla \cdot u = 0$ becomes $\nabla \cdot u = \phi$. Rasmussen et. al. [46] use a tiled Kolmogorov velocity field combined with standard incompressible methods to produce appealing large explosions. The methods mentioned all require extensive artistic control and experienced users to achieve appealing results. Only Yngve et. al. [55], as mentioned in Chapter One, attempts to model the fully compressible Navier-stokes system of equations. Their method is inefficient compared to incompressible methods—our hope has been to produce the most visually and physically accurate explosions in the computer graphics community.

3.1 ICE

Seeking inspiration for adapting incompressible techniques as well as the MAC grid towards modelling compressible flow led us, once again, to work by Harlow. In 1968 Harlow and Amsdan published "Numerical Calcula-

tions of Almost Incompressible Flow" and refined this work in "A Numerical Fluid Dynamics Calculation Method for All Flow Speeds" [28] published in 1972. In these two papers Harlow and Amsden described an Implicit Continuous-fluid Eulerian (ICE) technique for modelling single phase fluid dynamic problems. They start with the differential equations for conservation of momentum and of mass, and solve them using central differences defined on a MAC grid. The momentum discretization includes both previous time step (multiplied by $1 - \phi$) and forward time step pressures (multiplied by ϕ). The scalar ϕ (with values between 0 and 1) determines the time centering of the pressure term. Similarly the continuity equation includes both previous and forward time values, this time weighted by $\theta - 1$ and θ. These two equations with unknown densities and velocities are combined to create a system for pressure that includes the product of the scalars ϕ and θ. The resulting linear system is positive definite and reduces to a Poisson matrix as the speed of sound approaches infinity: the incompressible limit. This system is actually better conditioned than the incompressible Poisson matrix due to the presence of an additional p_i term making it more diagonally dominant.

ICE was unique because flows occurring over a wide range of Mach numbers could be modeled. It incorporated the two scalars ϕ and θ, that determine how explicit or implicit the solution technique is. If the two scalars are set to zero the method is fully explicit; if both scalars are 1. the method is fully implicit. In essence, changing the two independent of one another in between the two extremes becomes a knob a simulator can turn to tailor the outcome to their desires. Our experience in implementing the method presented was that the large number of parameters made the simulations fussy. We felt many of the parameters were not necessary to meet our goals. ICE also supported arbitrary equations of state—we felt a simple linear EOS would suffice. In the end we did not feel there was a great need to have an arbitrary amount of explicitness versus implicitness. Instead, we developed a much simpler method that more closely resembles classical computer graphics incompressible flow methods. Advection and pressure are fully separated into an explicit and implicit step; doing this allows any advection scheme to be used, further adding to the overall robustness of the method.

3.2 A Simplified ICE Technique

Like Harlow and Amsden, our initial developments used velocity and density as our primary variables. In one dimension the method begins with the

conservation of mass and momentum we have seen before:

$$\frac{\partial \rho}{\partial t} + \frac{\partial \rho u}{\partial x} = 0 \tag{3.1}$$

$$\frac{\partial \rho u}{\partial t} + \frac{\partial (\rho u^2)}{\partial x} = -\frac{\partial p}{\partial x} \tag{3.2}$$

$$p = c^2 \rho \tag{3.3}$$

Equation 3.2 is an equation of state (EOS), where c is the speed of sound. The EOS is needed to close the system, as equations 3.1 and 3.2 are not enough to fully determine the variables present.

In one dimension we begin by expressing continuity explicitly as

$$\rho_i^{n+1} = \rho_i^n + \frac{\Delta t}{\Delta x}\big[\frac{1}{2}(\rho_{i-1}^n + \rho_i^n)u_{i-\frac{1}{2}}^n - \frac{1}{2}(\rho_i^n + \rho_{i+1}^n)u_{i+\frac{1}{2}}^n\big] \tag{3.4}$$

and velocity as

$$u_{i+\frac{1}{2}}^{n+1} = u_{i+\frac{1}{2}}^n + \frac{1}{\frac{1}{2}(\rho_i^n + \rho_{i+1}^n)}\frac{\Delta t}{\Delta x}\big[c^2(\rho_i^n - \rho_{i+1}^n) + \rho u_i^{n2} - \rho u_{i+1}^n)^2\big] \tag{3.5}$$

Notice that $c^2(\rho_i^n - \rho_{i+1}^n)$ is pressure via a linear equation of state (equation 3.2). The convective fluxes, $(u_i^n)^2$ and $(u_{i+1}^n)^2$ (which are needed at cell centers), are upwinded as

$$u_i^2 = \begin{cases} u_{i-\frac{1}{2},j}^2 & \text{if } (u_{i-\frac{1}{2}} + u_{i+\frac{1}{2}}) < 0 \\ u_{i+\frac{1}{2},j}^2 & \text{if } (u_{i-\frac{1}{2}} + u_{i+\frac{1}{2}}) \geq 0 \end{cases} \tag{3.6}$$

This is a first order upwinding approach, but could easily be replaced by a higher resolution method or semi-Lagrangian formulas.

This explicit step is limited by

$$\Delta t = \alpha \frac{\Delta x}{c + \max(|\vec{u}|)} \tag{3.7}$$

This is an expression of what is known as the *Courant-Friedrichs-Lewy condition* (CFL condition). The CFL condition states that the time step must be small enough in order to make the numerical domain of dependence of a solution point include the true physical domain of dependence, i.e. not miss any significant propagation of information. In our case this is partly dictated by fluid velocity: obviously we do not want anything to be advected

a distance greater than a cell length as we would lose the density associated with that cell. In addition, the sound speed c appears in the denominator of equation 3.7, since in this explicit treatment of pressure the CFL condition dictates that sound waves must not travel further than a cell length in one time step. Using p^{n+1} instead of p^n in 3.5 and solving for p implicitly removes the sound speed c from equation 3.7. Doing this is attractive for any flow where the flow speed is significantly slower than the speed of sound and is the approach taken in the next section.

Solving for pressure implicitly requires the operator splitting technique presented in Chapter Two. The previous equation for u^{n+1} becomes, as the pressure is now dealt with implicitly, an equation for an intermediate velocity value that only includes advection, $\tilde{u}$:

$$\tilde{u}_{i+\frac{1}{2}} = u^n_{i+\frac{1}{2}} + \frac{1}{\frac{1}{2}(\rho^n_i + \rho^n_{i+1})} \frac{\Delta t}{\Delta x} \left[\rho {u^n_i}^2 - \rho {u^n_{i+1}}^2\right] \tag{3.8}$$

The u^2 terms are upwinded as before in equation 3.6. Like velocity, we calculate an intermediate density that includes the advection term from the continuity equation

$$\tilde{\rho_i} = \rho^n_i + \frac{\Delta t}{\Delta x}[(\rho u)_{i-\frac{1}{2}} - (\rho u)_{i+\frac{1}{2}}] \tag{3.9}$$

where now the $\rho_{i\pm\frac{1}{2}}$ densities are upwinded as

$$\rho_{i+\frac{1}{2}} = \begin{cases} \rho_{i+1} & \text{if } u_{i+\frac{1}{2}} < 0 \\ \rho_i & \text{if } u_{i+\frac{1}{2}} \geq 0 \end{cases}$$

Using these advected values the final velocity and density are

$$u^{n+1}_{i+\frac{1}{2}} = \tilde{u}_{i+\frac{1}{2}} + \frac{1}{\frac{1}{2}(\tilde{\rho}^n_i + \tilde{\rho}^n_{i+1})} \frac{\Delta t}{\Delta x} (p^{n+1}_i - p^{n+1}_{i+1}) \tag{3.10}$$

$$\rho^{n+1}_i = \tilde{\rho_i} + \frac{\Delta t}{\Delta x} \left[(\frac{1}{2}(\tilde{\rho_{i-1}} + \tilde{\rho_i})\Delta u_{i-\frac{1}{2}} - \frac{1}{2}(\tilde{\rho_i} + \tilde{\rho_{i+1}})\Delta u_{i+\frac{1}{2}} \right] \tag{3.11}$$

where $\Delta u_{i-\frac{1}{2}}$ is simply $u^{n+1}_{i+\frac{1}{2}} - \tilde{u}_{i+\frac{1}{2}}$ from equation 3.10. After substituting the equation for Δu into the equation for ρ^{n+1}_i and then substituting that into our constitutive relation $p = c^2\rho$ gives us our linear system to solve for

in p^{n+1}. This system is tridiagonal in one dimension with non-zero bands in two and three dimensions. Moreover it is positive definite, and an M-matrix. From the layout of the matrix we can see that when the speed of sound goes to infinity (the incompressible limit) we are left with the incompressible Poisson matrix.
If we let $\alpha = \frac{c^2 \Delta t^2}{\Delta x^2}$ a single row of the matrix (in one dimension) looks like

$$\Big[\alpha p_{i-1} + (1 + 2\alpha)p_i - \alpha p_{i+1}\Big] = c^2 \tilde{\rho}_i \tag{3.12}$$

This system is identical to the linear system formed in the original ICE method without the dependence on ϕ and θ. Again it reduces to the MAC matrix in the limit, but it is better conditioned because of the additional p_i term.

3.3 Adding Simple Thermodynamics

The previous method performed well but we felt it could easily be extended in order to better capture thermodynamic expansion and contraction effects. We begin like most compressible flow simulators by using the equation of state $p = nRT$ (the ideal gas law), instead of $p = c^2\rho$ which does not relate pressure to temperature. We assumed that viscosity and heat conduction were negligible in our flow and that the flow was isentropic: entropy is conserved. It can be shown [8] for an ideal isentropic gas flow that

$$p = e^{\frac{S}{c_v}} \rho^\gamma \tag{3.13}$$

where S is entropy and c_v is the specific heat, and γ is the heat capacity ratio (which is constant for an ideal gas, with $\gamma \approx 1.4$ for air). Since entropy is conserved, the first term, $e^{\frac{S}{c_v}}$, is constant; for our purposes we will refer to it is as the entropy factor or F. Furthermore, since entropy remains constant the material derivative of F is zero—this means it can be advected like density in [21]. For air at regular conditions in an isentropic setting, the temperature can be shown to be:

$$T = F\rho^{\gamma - 1} \tag{3.14}$$

We manipulate the entropy factor field by adding heat sources during the calculation. The simplest heat source we used was to set $F = \frac{T_{target}}{\rho^{\gamma-1}}$ where T_{target} was a target temperature for a given cell. Temperatures were set to T_{target} for cells located within a heat source region of the domain.

Using the preceding simplifications we get the following equations[11] (which are the non-conservative form of the Navier-Stokes system of equations)

$$\vec{u}^{n+1} = \tilde{u} - \frac{\Delta t}{\tilde{\rho}} \nabla p \tag{3.15}$$

$$\rho^{n+1} = \tilde{\rho} - \Delta t \tilde{\rho} \nabla \cdot \vec{u}^{n+1} \tag{3.16}$$

$$p^{n+1} = RT\rho^{n+1} \tag{3.17}$$

Notice that in this equation of state 3.17 the sound speed is RT—it depends on temperature. By using a "lagged" T calculated from intermediate explicit values of F and ρ, the equations are linear, but no longer uniform throughout the domain.

The finite difference solution begins similarly by computing the intermediate density and velocity values that account for advection:

$$\tilde{\rho}_{i,j} = \rho^n_{i,j} - \frac{\Delta t}{\Delta x}\left(u^n_{i,j}(\nabla\rho)_x + v^n_{i,j}(\nabla\rho)_y\right) \tag{3.18}$$

and

$$\tilde{u}_{i+\frac{1}{2},j} = u^n_{i+\frac{1}{2},j} - \frac{\Delta t}{\Delta x}\left(u_{i+\frac{1}{2},j}(\nabla u)_x + v_{i+\frac{1}{2},j}(\nabla u)_y\right) \tag{3.19}$$

where $u^n_{i,j}$ and $v^n_{i,j}$ are averaged to the cell centers from the neighbouring two face values and $v_{i+\frac{1}{2},j}$ is averaged from the neighbouring four vertical velocity values. The $(\nabla\rho)_x$ values are calculated as:

$$(\nabla\rho)_x = \begin{cases} (\rho^n_{i+1,j} - \rho^n_{i,j}) & \text{if } u_{i,j} < 0 \\ (\rho^n_{i,j} - \rho^n_{i-1,j}) & \text{if } u_{i,j} \geq 0 \end{cases} \tag{3.20}$$

Similarly the $(\nabla u)_x$ values from 3.19 are calculated as:

$$(\nabla u)_x = \begin{cases} (u^n_{i+\frac{3}{2},j} - u^n_{i+\frac{1}{2},j}) & \text{if } u_{i+\frac{i}{2},j} < 0 \\ (u^n_{i+\frac{1}{2},j} - u^n_{i-\frac{1}{2},j}) & \text{if } u_{i+\frac{1}{2},j} \geq 0 \end{cases} \tag{3.21}$$

Like density, our entropy factor (F) can be advected around in the same velocity field as:

$$\tilde{F}_{i,j} = F^n_{i,j} - \frac{\Delta t}{\Delta x}\left(u^n_{i,j}(\nabla F)_x + v^n_{i,j}(\nabla F)_y\right) \tag{3.22}$$

[11] Where again, the ˜ values are intermediate values that account for advection.

The temperature value used in the equation of state is calculated as $T_{i,j} = \tilde{F}_{i,j}\tilde{\rho}_{i,j}^{\gamma-1}$. Since our EOS is a function of the temperature, using ρ^{n+1} instead of ρ^n would produce a more difficult, although more accurate, non-linear problem to solve. We felt using ρ^n and having a linear system in pressure, rather than a non-linear system in density (ρ^γ), was a reasonable and attractive simplification.

In the explicit version, ρ^{n+1} is simply taken to be $\tilde{\rho}$ and u^{n+1} is updated using our equation of state 3.17, as:

$$u_{i+\frac{1}{2}}^{n+1} = \tilde{u}_{i+\frac{1}{2}} - R\frac{\Delta t}{\Delta x}\left(T_{i,j}\tilde{\rho}_{i,j} - T_{i+1,j}\tilde{\rho}_{i+1,j}\right) \tag{3.23}$$

In the implicit version a linear system in pressure is built by substituting

$$u_{i+\frac{1}{2}}^{n+1} = \tilde{u}_{i+\frac{1}{2}} - \frac{\Delta t}{\Delta x \rho_{i+\frac{1}{2},j}}\left(p_{i,j}^{n+1} - p_{i+1,j}^{n+1}\right) \tag{3.24}$$

into

$$\rho_{i,j}^{n+1} = \tilde{\rho}_{i,j} - \frac{\tilde{\rho}_{i,j}\Delta t}{\Delta x}\left(u_{i+\frac{1}{2},j}^{n+1} - u_{i-\frac{1}{2},j}^{n+1} + v_{i,j+\frac{1}{2}}^{n+1} - v_{i,j-\frac{1}{2}}^{n+1}\right) \tag{3.25}$$

$\rho_{i,j}^{n+1}$ in 3.25 is taken to be $\frac{p_{i,j}^{n+1}}{RT(i,j)}$. In order to ensure the system is symmetric we divide 3.25 by $\Delta t \rho_{i,j}$. A row in the resulting linear system looks like:

$$\begin{aligned}\frac{\Delta t}{\Delta x^2}\Bigg[&- \frac{p_{i+1,j}}{\tilde{\rho}_{i+\frac{1}{2},j}} - \frac{p_{i-1,j}}{\tilde{\rho}_{i-\frac{1}{2},j}} - \frac{p_{i,j+1}}{\tilde{\rho}_{i,j+\frac{1}{2}}} - \frac{p_{i,j-1}}{\tilde{\rho}_{i,j-\frac{1}{2}}} + \\ &\left(\frac{1}{RT_{i,j}\tilde{\rho}_{i,j}} + \frac{1}{\tilde{\rho}_{i-\frac{1}{2},j}} + \frac{1}{\tilde{\rho}_{i+\frac{1}{2},j}} + \frac{1}{\tilde{\rho}_{i,j-\frac{1}{2}}} + \frac{1}{\tilde{\rho}_{i,j+\frac{1}{2}}}\right)p_{i,j}\Bigg] = \\ &\frac{1}{\Delta t} - \left(\tilde{u}_{i+\frac{1}{2},j}^{n+1} - \tilde{u}_{i-\frac{1}{2},j}^{n+1} + \tilde{v}_{i,j+\frac{1}{2}}^{n+1} - \tilde{v}_{i,j-\frac{1}{2}}^{n+1}\right)\end{aligned} \tag{3.26}$$

The $\rho_{i\pm\frac{1}{2},j}$ values are averaged to the cell faces from the adjacent cells. This system is symmetric, positive definite, and reduces in the incompressible limit to the incompressible Poisson matrix. Although it is positive definite the incorporation of the averaged densities into the matrix makes it more difficult to deal with than the previous system. The potential for ill-conditioning is now present.

This leads to the following solution algorithm:

- Advect ρ, entropy factor, and velocity according to 3.19-3.22

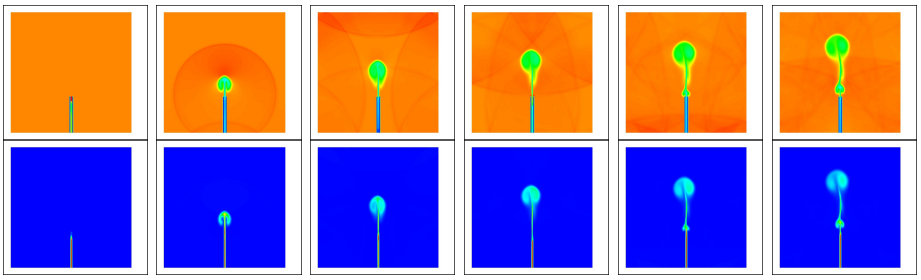

Figure 3.1: Results obtained using the method presented in section 3.3. A heat source is located in the middle of the domain in between two walls. The plots on top are of the density profile while the bottom plots depict the temperatures. The pressure wave can be seen in the density plot and, as expected, travels much faster than the plume of hotter, less dense air.

- Add heat sources (increase entropy factor in a specified region) and calculate temperature as $T_{i,j} = \tilde{F}_{i,j}\tilde{\rho}_{i,j}^{\gamma-1}$
- Determine the pressure according to 3.26
- Update ρ and velocity according to 3.25 and 3.24. Let $F^{n+1} = \tilde{F}$

Additional steps pertaining to boundary conditions and solids inside the domain have been neglected for simplicity and will be addressed in section 3.5.

3.4 A Change in Primary Variables

The previous methods, although simple, are not guaranteed to conserve momentum. This fact prompted us to investigate a change in primary variables from velocity to momentum, and using the conservative form of the Navier-Stokes equations to perfectly conserve momentum. In contrast to calculating fluxes when needed, velocities are now calculated by dividing the momentum by a density averaged between the two cells that share the face where the velocity is being calculated. In the presence of shocks this calculation would produce unphysical spikes in the velocity field because of the sharp gradients present when shocks occur. As before, our focus is on subsonic flow. Velocities are calculated without concern for the presence of sharp discontinuities in the density field. The method is presented in two dimensions because the upwinding terms require more attention.

Our explicit formulation begins as before by calculating densities as

$$\tilde{\rho}_{i,j} = \rho^n_{i,j} + \frac{\Delta t}{\Delta x}\big((\rho u)^n_{i-\frac{1}{2},j} - (\rho u)^n_{i+\frac{1}{2},j} + (\rho v)^n_{i,j-\frac{1}{2}} - (\rho v)^n_{i,j+\frac{1}{2}}\big) \tag{3.27}$$

Since we store momentum values, ρu directly on cell faces, it is possible to calculate equation 3.27 without any upwinding. Unfortunately, we found this gives rise to spurious oscillations, and so found the following upwind scheme for calculating $(\rho u)_{i\pm\frac{1}{2},j}$ and $(\rho v)_{i,j\pm\frac{1}{2}}$.

$$(\rho u)_{i+\frac{1}{2},j} = \begin{cases} u_{i+\frac{1}{2},j}\rho_{i+1,j} & \text{if } u_{i+\frac{1}{2},j} < 0 \\ u_{i+\frac{1}{2},j}\rho_{i,j} & \text{if } u_{i+\frac{1}{2},j} \geq 0 \end{cases} \tag{3.28}$$

necessary. The velocity $u_{i+\frac{1}{2}}$ in 3.28 is calculated as mentioned previously:

$$u_{i+\frac{1}{2},j} = \frac{(\rho u)_{i+\frac{1}{2},j}}{\frac{1}{2}(\rho_{i,j} + \rho_{i+1,j})} \tag{3.29}$$

and has the previously mentioned problems in the presence of shocks.

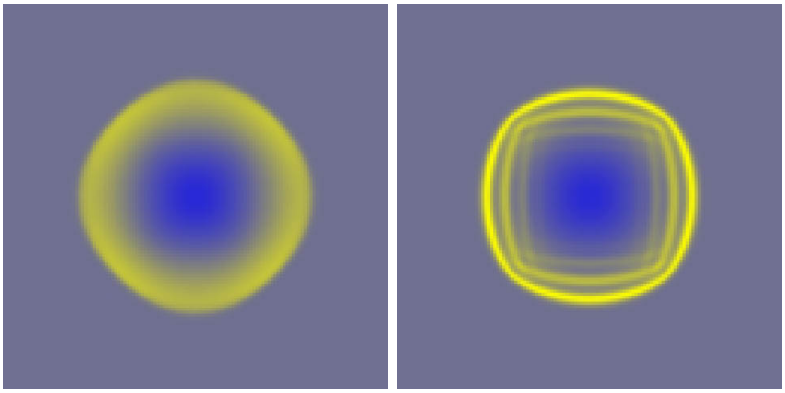

Figure 3.2: Results at t=0.005 for two simulations in which a circular region of higher density (one order of magnitude) is brought to equilibrium. The figure on the left uses the upwinding scheme in equation 3.28, while the figure on the right uses the zipped scheme (eq. 3.32) for the $\frac{(\rho u)^2}{\rho}$ and the upwinded terms for the $v(\rho u)$ terms.

In contrast to the previous method where we would update velocity, we

now update our momentum in the horizontal direction with

$$(\tilde{\rho u})_{i+\frac{1}{2},j} = \frac{\Delta t}{\Delta x}\Big(\frac{(\rho u)^{n}_{i,j}{}^{2}}{\rho^{n}_{i,j}} - \frac{(\rho u)^{n}_{i+1,j}{}^{2}}{\rho^{n}_{i+1,j}} + (v\rho u)^{n}_{i+\frac{1}{2},j+\frac{1}{2}} - (v\rho u)^{n}_{i+\frac{1}{2},j-\frac{1}{2}}\Big) \tag{3.30}$$

The $(v\rho u)_{i+\frac{1}{2},j\pm\frac{1}{2}}$ values are calculated at the horizontally-aligned red points in 3.3 and are upwinded according to:

$$(v\rho u)_{i+\frac{1}{2},j+\frac{1}{2}} = \begin{cases} v_{i+\frac{1}{2},j+\frac{1}{2}}(\rho u)_{i+\frac{1}{2},j+1} & \text{if } v_{i+\frac{1}{2},j+\frac{1}{2}} < 0 \\ v_{i+\frac{1}{2},j+\frac{1}{2}}(\rho u)_{i+\frac{1}{2},j} & \text{if } v_{i+\frac{1}{2},j+\frac{1}{2}} \geq 0 \end{cases} \tag{3.31}$$

where $v_{i+\frac{1}{2},j+\frac{1}{2}}$ is calculated as $\frac{(v_{i,j+\frac{1}{2}}+v_{i-1,j+\frac{1}{2}})}{2}$. The $(\rho u)^2_{i,j}$ and $(\rho u)^2_{i+1,j}$ terms are calculated at the vertically-aligned red points in 3.3 and are upwinded as in 3.6.

We are considering the flux into the cell of size Δx by Δy centered around position $(i+\frac{1}{2}, j)$. These flux locations are the red dots in figure 3.3, while the green arrows are the positions to which we average v values, and the blue arrows are the momentum values we use to upwind $(\rho u)^2$ terms. The $(v\rho u)$ and $\frac{(\rho u)^2}{\rho}$ terms reduce to ρu when multiplied by $\frac{\Delta t}{\Delta x}$.

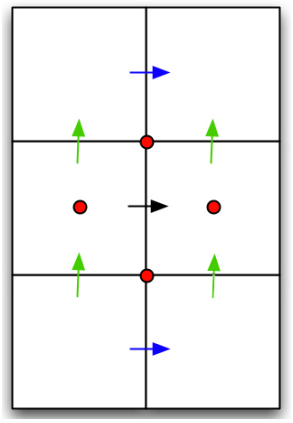

Figure 3.3: Illustration of the cells and vectors involved in the upwinding scheme used in equations 3.31 and 3.6.

Harlow and Amsden describe a different way to calculate the $(\rho u)^2$ terms, a form they called zip fluxes:

$$(\rho u)^2_{i,j} = (u_{i-\frac{1}{2},j})(u_{i+\frac{1}{2},j}) \tag{3.32}$$

Similarly the v and the ρu values in equation 3.31 can become $v_{i,j} = (v_{i,j-\frac{1}{2}} v_{i,j+\frac{1}{2}})^{\frac{1}{2}}$ and $((\rho u)_{i-\frac{1}{2},j}(\rho u)_{i+\frac{1}{2}})^{\frac{1}{2}}$. However, this too gives rise to unstable oscillations without adding artificial viscosity, which we have tried to avoid.

Using equation 3.2 the final step in our explicit formulation is to update our momentum values as:

$$(\rho u)^{n+1}_{i+\frac{1}{2}} = (\tilde{\rho u})_{i+\frac{1}{2},j} - \frac{\Delta t c^2}{\Delta x}\big(\tilde{\rho}_{i+1,j} - \tilde{\rho}_{i,j}\big) \tag{3.33}$$

For the time being we ignored the pressure effects on density and simply let $\rho^{n+1} = \tilde{\rho}$.

As in the explicit formulation, we solve equation 3.27 and equation 3.30 that account for the advection of density and momentum. Our next time step density and momentum are:

$$\rho^{n+1}_{i,j} = \tilde{\rho}_{i,j} - \frac{\Delta t}{\Delta x}\big((\Delta\rho u)_{i-\frac{1}{2},j} - (\Delta\rho u)_{i+\frac{1}{2},j} + (\Delta\rho v)_{i,j-\frac{1}{2}} - (\Delta\rho v)_{i,j+\frac{1}{2}}\big) \tag{3.34}$$

$$(\rho u)^{n+1}_{i+\frac{1}{2},j} = (\tilde{\rho u})_{i+\frac{1}{2},j} - \frac{\Delta t}{\Delta x}\big(p^{n+1}_{i+1,j} - p^{n+1}_{i,j}\big) \tag{3.35}$$

where $(\Delta\rho u)_{i-\frac{1}{2},j} = (\rho u)_{i-\frac{1}{2},j} - (\tilde{\rho u})_{i-\frac{1}{2},j}$ (from equation 3.35). As opposed to the previous technique from section 3.2 in which our equation for ρ^{n+1}_i was substituted into our constitutive relationship, we now substitute $p^{n+1} = c^2\rho^{n+1}$ into equation 3.35 which is then substituted into equation 3.34. This produces a linear system in ρ^{n+1}_i as opposed to p^{n+1}. With $\alpha = c^2(\frac{\Delta t}{\Delta x})^2$ a row in this linear system looks like

$$\Big[-\alpha\rho_{i,j-1} - \alpha\rho_{i-1,j} + (1+4\alpha)\rho_{i,j} - \alpha\rho_{i+1,j} - \alpha\rho_{i,j+1}\Big] = \tilde{\rho}_i \tag{3.36}$$

Since ρ and pressure are linearly related, this system is identical to the one in the previous formulation. It is sparse, symmetric, and positive definite and reduces, in the limit, to the incompressible Poisson matrix.

The solution algorithm is identical to the algorithm listed in section 3.2, only now the variables and upwinding procedures differ.

3.5 Boundary Conditions and Grid Aligned Solids

Both solid wall and periodic boundary conditions were used with the preceding methods. Values needed for periodic boundaries are taken from the other side of the domain when needed. The solid wall boundaries and interior grid aligned solids were modeled by skipping density calculations (setting them to zero) and setting the velocity or momentum values into and out of solid cells to be zero. For cells bordering on solid cells, the matrix entries in equation 3.36 must account for the pressure in the wall equalling the pressure within the current cell. For example, for a cell on the top boundary, its corresponding row in the matrix becomes:

$$\Big[-\alpha\rho_{i,j-1} - \alpha\rho_{i-1,j} + (1+3\alpha)\rho_{i,j} - \alpha\rho_{i+1,j} \Big] = \tilde{\rho}_i \tag{3.37}$$

The only difference being that the $\rho_{i,j+1}$ entry is gone and $(1+4\alpha)\rho_{i,j}$ has become $(1+3\alpha)\rho_{i,j}$, because $p_{i,j+1} = p_{i,j}$.

3.6 The Incompressible Limit

As we saw in Chapter Two, enforcing the divergence of velocity to be zero results in the following linear system of equations:

$$\nabla \cdot \nabla p = \nabla \cdot \tilde{u} \tag{3.38}$$

The systems shown in 3.12, 3.26, and 3.36 all reduce as the speed of sound approaches infinity to the incompressible Poisson matrix: the $\nabla \cdot \nabla$ matrix in 3.38. This means that these methods will model incompressible flows as well as the compressible flows they have been designed to model. Since we track variations in density using the continuity equation and use it to construct our linear systems, the right hand side vectors differ from the vectors in 3.38. The incompressible methods assume density to be constant–a reasonable, but incorrect, assumption. Changes in pressure can actually lead to changes in density, if for no other reason than numerical error. If an incompressible simulator did track these minuscule deviations in density space and included a drift correction term, the incompressible system would resemble the ones presented in the previous sections. In contrast, our models directly use the continuity equation to track and respond to changes in density in a fully physical way.

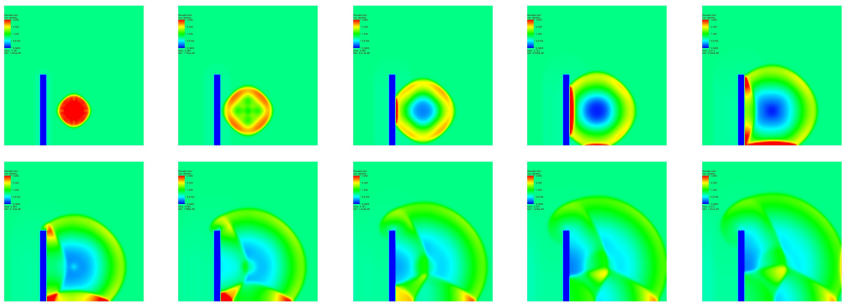

Figure 3.4: Results of a simulation run in which a circular area of higher density (one order of magnitude) is brought to equilibrium.

3.7 Implementation and Results

The preceding methods were implemented using C++ in two dimensions with both solid wall and periodic boundaries for a square domain that included solid objects. The momentum formulation was also implemented in three dimensions. A preconditioned conjugate gradient solver and a sparse matrix class were used to solve and represent the linear systems presented in 3.12, 3.26, and in 3.36. Solutions are usually found in a single iteration when a warm start is used.

Figure 3.3 shows the results of a simulation that included the heat source defined in section 3.3. The heat source is located in between the walls in the center of the domain and ends at approximately three quarters of the wall height. It continually sets the temperature in the heat source region to $1000K$. The figure contains density profiles (top plots) and temperature profiles (bottom plots). The initial pressure wave can been seen in the density plots, and, as expected, travels much faster than the plume of hot air which rises out of the heat source.

In figure 3.6 a circular region within the center of the domain with density one order of magnitude larger than the surrounding cells is brought to equilibrium. On the left portion of the domain is a wall. This simulation used the method presented in section 3.4 and the upwinding scheme for fluxes presented in 3.31 and 3.6 (as opposed to the zipped procedure). Notice the diffraction and reflection waves in the last frames of the figure. Although these results look fine, often the results obtained when using momentum values directly in equation 3.27 as opposed to the upwinding scheme presented in 3.28 were incorrect. At other times the simulations simply crashed. Essentially, the upwinding scheme is, as expected, more stable as we begin to

push the limits of the simulation. For the same initial conditions, if we begin to increase the CFL factor from 0.1 to 1.0, the simulation using 3.27 will produce unphysical results at a lower CFL factor. Although, in the cases where this occurs we are also pushing the theoretical limits of the method, unphysical results occur when the Mach number approaches and exceeds 1.0. This failure is inconsequential since we freely admit our method will not produce physically accurate results for supersonic flow.

The zipped fluxes presented in 3.32 are second order accurate, and with this accuracy comes less stability. As can be seen in figure 3.4, the results obtained using the zipped fluxes are more detailed, but these simulations are more likely to crash or produce small time-steps. Like the semi-lagrangian techniques, the first order accurate upwinding scheme adds diffusion and allows us to neglect viscosity. Incorporating viscosity would make the zipped method more stable, but it would add considerable computational cost. Harlow and Amsden favour the higher order schemes combined with viscosity because they favour more physically accurate simulations. We are willing to sacrifice physical accuracy for an improvement in run time. In practice, neither upwinded nor zipped fluxes worked better all of the time. As we changed the CFL factor and/or the initial density gradient, one method would produce more spherical results than the other, or in some cases, especially when directly using the momentum values in equation 3.27, the simulation would become unstable and crash.

Chapter 4

Towards Eulerian Granular Flow

Granular material dynamics, like those of sand, are essentially macroscopic fluid flows. At the microscopic level the molecules that make up a regular fluid collide and interact, transferring kinetic energy from one molecule to another. The time and length scales over which a fluid is modeled cover an enormous number of molecular interactions. Individual collisions are averaged over these significantly larger scales: modelling the flow as a continuum makes sense. Granular flows also behave in this way, but the time and length scales are significantly closer to the scales that are significant in individual grain interactions. Treating these flows as a continuum is not as straightforward or as plausible; some phenomena, such as force "bridging" where force is transferred through discrete chains of grains in contact and not averaged through the volume, are out of reach of the continuum approach. That said, the continuum approach has been successfully applied in engineering to solve soil mechanics problems, for example, and appears to be the most promising approach[12] to efficiently solve large-scale problems.

We have all seen sand flow on a dune or a beach like it was water, yet sand always settles in a way that water never does. The primary reason for this response is the presence of friction between the individual grains of sand. As mentioned in Chapter Two, scientists who model granular flow often use mesh-free methods like discrete elements where the response of every single grain is tracked, or with Lagrangian mesh-based elasto-plastic continuum methods that are strictly limited to small amounts of flow. We feel that an Eulerian continuum approach would be an ideal third possibility for applications in computer graphics or as quick analysis tools for engineers.

While this research was ultimately unsuccessful in effectively simulating granular material with an Eulerian continuum formulation, we here report on our first steps towards this goal. Rather than tackle the full problem

[12] As opposed to the previously mentioned elasto-plastic Lagrangian or discrete element techniques

with a Mohr-Coulomb constitutive law, which relates shear stress and strain rate in the continuum to inter-particle friction, we restricted attention to just the pressure/density relationship of an idealized frictionless granular material. This is in line with the previous chapters' work on improved handling of pressure/density changes in low speed flow, but also exposes a key distinction of granular flow from fluid flow: the presence of inequality constraints, expressed as complementarity conditions.

Our continuum model for sand is created by averaging properties onto a MAC grid. The density in a cell essentially measures the number of sand particles in a cell, and the pressure measures the normal contact force between them. This approximation is obviously isotropic; direction and orientation of individual contacting particles is lost. The cells that represent an unsettled flowing region of sand may have density values less than the cells of a settled region of sand. As a simplified first approximation, when the density is less than the settled value, we take the pressure (the average normal contact force) to be zero. Similarly, if the sand in an area of our domain has settled, its density will exactly equal some ideal settled density and the pressure will be allowed to be positive to resist applied compressive load (e.g. from gravity). This relationship is known as a complementarity relationship; one quantity (pressure) is positive while the other (the difference between current density and the settled density) is zero, but they are never positive at the same time. Mathematically this is expressed as

$$0 \leq p \perp \rho_{settled} - \rho \geq 0 \tag{4.1}$$

This constraint replaces the divergence-free condition of incompressible flow (where p can be both positive and negative, but density is always held constant), or the equation of state relating pressure and density in compressible flow.

4.1 Complementarity

Complementarity problems are a broad class of problems, with the simplest case being the linear complementarity problem (LCP). In its simplest form, the LCP can be stated as the goal of determining $p \in R^n$ such that

$$p \geq 0 \tag{4.2}$$

$$Ap + b \geq 0 \tag{4.3}$$

$$p^T(Ap + b) = 0 \tag{4.4}$$

which can also be written as before, $0 \leq p \perp Ap + b \geq 0$ [16]. To clarify, this can be thought of as a relationship where two quantities cannot be greater than zero at the same time. A vector p that satisfies the first two inequalities in 4.2 is said to be feasible. Another way of expressing this is to let $w = Ap + b$, then for all $i = 1, ..., n$, $p_i w_i = 0$. Depending on the matrix A and vector b the LCP may have zero, one, or an infinite number of solutions. This complementarity relationship has been used to describe phenomena in a large number of fields including economics, civil engineering, and computer graphics. LCPs are a specialized version of a mixed complementarity problem: a more general formulation that combines the previous complementarity problem, possible with a nonlinear relationship between the two non-negative quantities, with additional equations that do not involve complementarity[2, 51]. LCPs are not the only complementarity problem, but since we have a linear system we will only mention LCPs.

The LCP can also be expressed as the following inequality quadratic program (QP):

$$\begin{aligned} \underset{x}{\text{minimize}} \quad & x^T(Ax + b) \\ \text{subject to} \quad & Ax + b \geq 0 \\ \text{subject to} \quad & x \geq 0 \end{aligned} \tag{4.5}$$

If the matrix is positive definite the problem has a unique solution. If the matrix A is positive semidefinite, it can be shown that the problem has a solution (potentially non-unique). Using this expression of the LCP can be advantageous since QP problems have been extensively researched over a long period of time. Additionally, they do not necessarily suffer from the same shortcomings as competing LCP algorithms.

Frequently in video games or in movie animation it is necessary to model the dynamic response of contacting rigid bodies. Rigid body contact (as it's known) can be formulated as a linear complementarity or mixed complementarity problem. If we consider two rigid bodies the idea is easy to conceptualize. When two bodies are not in contact there is no impulse (force integrated over time) between them, but the distance is greater than zero. When two bodies are in contact the force between them is positive, while the distance separating them is zero. If we solve for the position and velocity of our rigid bodies using the following scheme

$$v^{n+1} = v^n + \sum_{i=0}^{n} h_i + \Delta t \; g \tag{4.6}$$

$$x^{n+1} = x^n + \Delta t \, v^{n+1} \tag{4.7}$$

the following LCP can be formulated $0 \leq h \perp |x_i - x_j| \geq 0$ where $|x_i - x_j|$ represents the distance between two rigid bodies[13].

A great deal of research [2, 5, 6, 51, 53] has begun with this simple idea. An obvious concern is that this analysis does not consider frictional forces that develop between two contacting bodies. Incorporating frictional constraints into our LCP is not as easy to visualize and is beyond the scope of this work. The interested reader can consult [2, 5, 29, 51] for details concerning frictional constraints. The simple argument begins with Coulomb friction. If the tangential friction impulse is less than a yield value $I_y = \mu I_N$ where μ is the coefficient of friction and I_N is the normal impulse between the two bodies, i.e. if $I_y - I > 0$, then the two contacting bodies will not slide past each other. On the other hand, if $I = I_y$, then the friction impulse I cannot exceed I_y and there is tangential movement between the two bodies. This argument is analogous to our initial argument for approximating Mohr-Coulomb friction (the tensor stress/strain version of Coulomb's law) on an Eulerian grid phrased as a complementarity problem.

4.2 Finite Difference Solution

The linear complementarity problem using pressure as the driving variable was initially built by discretizing, like before, the non-conservative momentum equation 1.2 and the continuity equation 1. We again employed operator splitting to separate the advection of density and velocity, but this time we used semi-Lagrangian advection to calculate $\tilde{\rho}$ and $\tilde{u}$. The equations for density and velocity become:

$$\rho_{i,j}^{n+1} = \tilde{\rho}_{i,j} - \Delta t \tilde{\rho}_{i,j} \left(\frac{u_{i+\frac{1}{2},j}^{n+1} - u_{i-\frac{1}{2},j}^{n+1}}{\Delta x} + \frac{v_{i,j+\frac{1}{2}}^{n+1} - v_{i,j-\frac{1}{2}}^{n+1}}{\Delta x} \right) \tag{4.8}$$

$$u_{i+\frac{1}{2},j}^{n+1} = \tilde{u}_{i+\frac{1}{2},j} - \frac{\Delta t}{\rho_{i+\frac{1}{2},j}} \left(\frac{p_{i+1,j} - p_{i,j}}{\Delta x} \right) \tag{4.9}$$

The equations for the next time step velocities were then substituted into the equation for the next time step density. This density was then subtracted from $\rho_{settled}$ to create the linear system in pressure for the complementarity problem $p \geq 0 \perp \rho_{max} - \rho \geq 0$. After substitution a row in this system looks

[13] This is not the only way to formulate this general contact complementarity idea; a great deal of research has gone into this and no method has shown to be the optimal solution

like:

$$\rho_{max} - \left[\tilde{\rho}_{i,j}\frac{\Delta t^2}{\Delta x^2}\left(\frac{p_{i+1,j}}{\tilde{\rho}_{i+\frac{1}{2},j}} + \frac{p_{i-1,j}}{\tilde{\rho}_{i-\frac{1}{2},j}} + \frac{p_{i,j+1}}{\tilde{\rho}_{i,j+\frac{1}{2}}} + \frac{p_{i,j-1}}{\tilde{\rho}_{i,j-\frac{1}{2}}} - \frac{p_{i,j}}{\tilde{\rho}_{i,j}}\right)\right] - \frac{\Delta t}{\Delta x}\tilde{\rho}_{i,j}(\tilde{u}_{i+\frac{1}{2},j} - \tilde{u}_{i-\frac{1}{2},j} + \tilde{v}_{i,j+\frac{1}{2}} - \tilde{v}_{i,j-\frac{1}{2}}) \geq 0 \tag{4.10}$$

Here, as before, the $\tilde{\rho}_{i\pm\frac{1}{2},j}$ and $\tilde{\rho}_{i,j\pm\frac{1}{2}}$ values are averaged to cell faces from adjacent cell densities. The density that appears in equation 4.8 and 4.9 are the same if the density in equation 4.8 is brought inside the parentheses by using face averaged density. If this is done the densities cancel in the linear system.

Similar to the method just outlined, we constructed the same system using the initial velocity-based compressible discretization outlined in Chapter Three: ignore the constitutive relationship, substitute the equation for velocity into the equation for density, and a system that is identical to the one here is obtained. Since the advection is calculated in a different way, the right hand side vector of the problem is not the same and the two discretizations produce different results.

The linear system constructed in 4.10 is positive semidefinite; the +1 term in the diagonal $(4\alpha + 1)p_{i,j}$ from our previous systems is missing. As a result this system is singular, with a null-space corresponding to constant pressure values. Positive semidefinite (PSD) systems do not admit unique solutions to the LCP, but do have unique $w = Ap + b$ vectors (known as the w-uniqueness property); that is, if p_1 and p_2 are solutions to the LCP, then $Ap_1 = Ap_2$ [16]. This result means that although multiple feasible pressure fields may exist, the density profile is unique. This singularity could be resolved by constraining one pressure (for instance the pressure value corresponding to the lowest density) to zero and thus eliminating it from the matrix sent to our LCP algorithms. A regularization approach, where a small amount is added along the diagonal, could also be used to improve the conditioning of the system.

4.3 Implementation

The sand simulator using the LCP solution for pressures was implemented in Matlab. We attempted to solve the LCP presented in section 4.2 using the Path solver [17, 23], an implementation of Lemke's algorithm [16], and using a Gauss-Seidel method modified to use a clamped Newton step. For

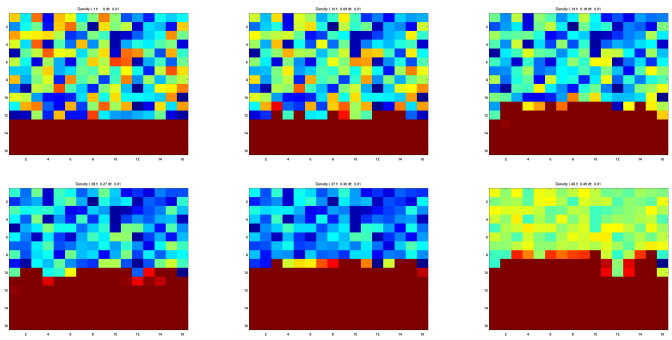

Figure 4.1: Plots showing sand settle under gravity.

our application, the rich history and availability of robust QP methods is attractive, but primarily we need a method that scales well. The performance of Lemke degrades as the problem size increases, a trait we would have liked to have avoided by using a QP method. Also, Lemke's method can not take advantage of previous solutions (a warm start) as certain QP methods can. Despite its drawbacks Lemke's method is commonly used since, if the matrix is positive definite, it will find the solution if it exists. For this reason, we used it primarily with test problems (which were never passed). We assumed that the problem was poorly posed or an error existed if Lemke's algorithm failed to find a solution.

Although the matrix characteristics point to the existence of a solution or solutions, the algorithms used frequently did not converge. The solutions also depended on the initial vector used; often a zero vector did not work. The system constructed from the method outline in section 3.2 behaved similarly. We suspect cells with densities approaching zero are to blame, but the question as to why the method failed is still an open one. Developing an algorithm that correctly solves this LCP requires that this question is answered.

We do feel the idea is sound and could lead to significant improvement in quickly modelling granular dynamics, our stated goal. Unfortunately, we did not achieve this. Hopefully, spending additional time—specifically to develop a specialized QP algorithm for this problem—could lead to accurate and reliable solutions to this LCP. Once the pressure LCP is solved the framework can easily be extended to correctly model the physics between settled grains of sand since there are no restrictions placed on the divergence of velocity like in Zhu's work.

Bibliography

[1] John D. Anderson. *Modern Compressible Flow With Historical Perspective*. McGraw-Hill, Inc., New York, NY, USA, 2003.

[2] M. Anitescu, F. Potra, and D. Stewart. Time-stepping for three-dimensional rigid body dynamics. *Computer Methods in Applied Mechanics and Engineering*, 177(3–4):183–197, 1999.

[3] M. Anitescu and F.A. Potra. Formulating dynamic multi-rigid-body contact problems with friction as solvable linear complementarity problems. *Nonlinear Dynamics*, 14(3):231–247, 1997.

[4] Mihai Anitescu, James F. Cremer, and Florian A. Potra. Formulating 3d contact dynamics problems. *Reports on Computational Mathematics*, 80, 1995.

[5] Mihai Anitescu and Gary D. Hart. A constraint-stabilized time-stepping approach for rigid multibody dynamics with joints, contact, and friction. *International Journal for Numerical Methods in Engineering*, 2002.

[6] David Baraff. Fast contact force computation for nonpenetrating rigid bodies. In *SIGGRAPH '94: Proceedings of the 21st annual conference on Computer graphics and interactive techniques*, pages 23–34, New York, NY, USA, 1994. ACM Press.

[7] Adam W. Bargteil, Tolga G. Goktekin, James F. O'brien, and John A. Strain. A semi-lagrangian contouring method for fluid simulation. *ACM Trans. Graph.*, 25(1):19–38, 2006.

[8] G. K. Batchelor. *An Introduction to Fluid Mechanics*. Cambridge University Press, 1967.

[9] Christopher Batty, Florence Bertails, and Robert Bridson. A fast variational framework for accurate solid-fluid coupling. In *SIGGRAPH '07: ACM SIGGRAPH 2007 papers*, page 100, New York, NY, USA, 2007. ACM Press.

[10] Nathan Bell, Yizhou Yu, and Peter J. Mucha. Particle-based simulation of granular materials. In *SCA '05: Proceedings of the 2005 ACM SIGGRAPH/Eurographics symposium on Computer animation*, pages 77–86, New York, NY, USA, 2005. ACM Press.

[11] Robert Bridson, Ronald Fedkiw, and Matthias Muller-Fischer. Fluid simulation: Siggraph 2006 course notes fedkiw and muller-fischer presenation videos are available from the citation page. In *SIGGRAPH '06: ACM SIGGRAPH 2006 Courses*, pages 1–87, New York, NY, USA, 2006. ACM Press.

[12] Mark Carlson, Peter J. Mucha, and Greg Turk. Rigid fluid: animating the interplay between rigid bodies and fluid. In *SIGGRAPH '04: ACM SIGGRAPH 2004 Papers*, pages 377–384, New York, NY, USA, 2004. ACM Press.

[13] D. Choi and C.L. Merkle. The application of preconditioning for viscous flows. *J. Comp. Phys.*, 105:203–223, 1993.

[14] A. J. Chorin. A numerical method for solving incompressible viscous flow problems. *J. Comp. Phys.*, 2:12–26, 1967.

[15] T. Chung. *Computational Fluid Dynamics*. Cambridge University Press, London, 2002.

[16] Richard W. Cottle, Jong-Shi Pang, and Richard E. Stone. *The Linear Complementarity Problem*. Academic Press,Inc., San Diego, CA, USA, 1992.

[17] S. P. Dirkse and M. C. Ferris. The path solver: A non-monotone stabilization scheme for mixed complementarity problems. *Optimization Methods and Software*, pages 123–156, 1995.

[18] Douglas Enright, Ronald Fedkiw, Joel Ferziger, and Ian Mitchell. A hybrid particle level set method for improved interface capturing. *J. Comput. Phys.*, 183(1):83–116, 2002.

[19] Douglas Enright, Stephen Marschner, and Ronald Fedkiw. Animation and rendering of complex water surfaces. In *SIGGRAPH '02: Proceedings of the 29th annual conference on Computer graphics and interactive techniques*, pages 736–744, New York, NY, USA, 2002. ACM Press.

[20] Ronald Fedkiw, Jos Stam, and Henrik Wann Jensen. Visual simulation of smoke. In *SIGGRAPH '01: Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, pages 15–22, New York, NY, USA, 2001. ACM Press.

[21] Ronald Fedkiw, Jos Stam, and Henrik Wann Jensen. Visual simulation of smoke. In *SIGGRAPH '01: Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, pages 15–22, New York, NY, USA, 2001. ACM Press.

[22] Bryan E. Feldman, James F. O'Brien, and Okan Arikan. Animating suspended particle explosions. In *SIGGRAPH '03: ACM SIGGRAPH 2003 Papers*, pages 708–715, New York, NY, USA, 2003. ACM Press.

[23] Michael C. Ferris and Todd S. Munson. Interfaces to path 3.0: Design, implementation and usage.

[24] Nick Foster and Dimitri Metaxas. Realistic animation of liquids. *Graph. Models Image Process.*, 58(5):471–483, 1996.

[25] Tolga G. Goktekin, Adam W. Bargteil, and James F. O'Brien. A method for animating viscoelastic fluids. In *SIGGRAPH '04: ACM SIGGRAPH 2004 Papers*, pages 463–468, New York, NY, USA, 2004. ACM Press.

[26] Eran Guendelman, Andrew Selle, Frank Losasso, and Ronald Fedkiw. Coupling water and smoke to thin deformable and rigid shells. In *SIGGRAPH '05: ACM SIGGRAPH 2005 Papers*, pages 973–981, New York, NY, USA, 2005. ACM Press.

[27] F. Harlow and J. Welch. Numerical calculation of time-dependent viscous incompressible flow of fluid with free surface. *Phys. Fluids*, 8:2182–2189, 1965.

[28] F. H. Harlow and A. A. Amsden, editors. *A numerical fluid dynamics calculation method for all flow speeds*, 1972.

[29] Gary D. Hart and Mihai Anitescu. A hard-constraint time-stepping approach for rigid multibody dynamics with joints, contact, and friction. In *TAPIA '03: Proceedings of the 2003 conference on Diversity in computing*, pages 34–41, New York, NY, USA, 2003. ACM Press.

[30] Ben Houston, Michael B. Nielsen, Christopher Batty, Ola Nilsson, and Ken Museth. Hierarchical rle level set: A compact and versatile deformable surface representation. *ACM Trans. Graph.*, 25(1):151–175, 2006.

[31] R. Issa. Solution of the implicitly discretized fluid flow equations by operator splitting. *J. Comp. Phys.*, 62:40–65, 1985.

[32] Bryan M. Klingner, Bryan E. Feldman, Nuttapong Chentanez, and James F. O'Brien. Fluid animation with dynamic meshes. In *Proceedings of ACM SIGGRAPH 2006*, August 2006.

[33] Frank Losasso, Frédéric Gibou, and Ron Fedkiw. Simulating water and smoke with an octree data structure. In *SIGGRAPH '04: ACM SIGGRAPH 2004 Papers*, pages 457–462, New York, NY, USA, 2004. ACM Press.

[34] Frank Losasso, Tamar Shinar, Andrew Selle, and Ronald Fedkiw. Multiple interacting liquids. *ACM Trans. Graph.*, 25(3):812–819, 2006.

[35] Frank Losasso, Tamar Shinar, Andrew Selle, and Ronald Fedkiw. Multiple interacting liquids. *ACM Trans. Graph.*, 25(3):812–819, 2006.

[36] C. Martins, J. Buchanan, and J. Amanatides. Animating real-time explosions. *The Journal of Visualization and Computer Animation*, 13(2):133–145, 2002.

[37] Oleg Mazarak, Claude Martins, and John Amanatides. Animating exploding objects. In *Proceedings of the 1999 conference on Graphics interface '99*, pages 211–218, San Francisco, CA, USA, 1999. Morgan Kaufmann Publishers Inc.

[38] Matthias Müller, Barbara Solenthaler, Richard Keiser, and Markus Gross. Particle-based fluid-fluid interaction. In *SCA '05: Proceedings of the 2005 ACM SIGGRAPH/Eurographics symposium on Computer animation*, pages 237–244, New York, NY, USA, 2005. ACM Press.

[39] G. C. Nayak and O. C. Zienkiewicz. Elasto-plastic stress analysis. a generalization for various constitutive relations including strain softening. *Int. J. Num. Meth. Eng.*, 5:113–135, 1972.

[40] Michael Neff and Eugene Fiume. A visual model for blast waves and fracture. In *Proceedings of the 1999 conference on Graphics interface*

'99, pages 193–202, San Francisco, CA, USA, 1999. Morgan Kaufmann Publishers Inc.

[41] Duc Quang Nguyen, Ronald Fedkiw, and Henrik Wann Jensen. Physically based modeling and animation of fire. In *SIGGRAPH '02: Proceedings of the 29th annual conference on Computer graphics and interactive techniques*, pages 721–728, New York, NY, USA, 2002. ACM Press.

[42] J. Nocedal and S.J. Wright. *Numerical Optimization*. Springer-Verlag, New York, NY, USA, 1999.

[43] S.V. Patankar and D.B. Spalding. A calculation procedure for heat, mass and momentum transfer in three-dimensional parabolic flows. *Int. J. Heat Mass Transfer*, 15:1787–1806, 1972.

[44] R. Peyret and H. Viviand. Pseudo-unsteady methods for inviscid or viscous flow computations. Plenum, NY, USA, 1985. C. Casi (em).

[45] N. Rasmussen, D. Enright, D. Nguyen, S. Marino, N. Sumner, W. Geiger, S. Hoon, and R. Fedkiw. Directable photorealistic liquids. In *SCA '04: Proceedings of the 2004 ACM SIGGRAPH/Eurographics symposium on Computer animation*, pages 193–202, Aire-la-Ville, Switzerland, Switzerland, 2004. Eurographics Association.

[46] Nick Rasmussen, Duc Quang Nguyen, Willi Geiger, and Ronald Fedkiw. Smoke simulation for large scale phenomena. In *SIGGRAPH '03: ACM SIGGRAPH 2003 Papers*, pages 703–707, New York, NY, USA, 2003. ACM Press.

[47] W. T. Reeves. Particle systemsa technique for modeling a class of fuzzy objects. *ACM Trans. Graph.*, 2(2):91–108, 1983.

[48] Premoze S., Tasdizen T., Bigler J., Lefohn A., and Whitaker R. Particlebased simulation of fluids. In *In Comp. Graph. Forum (Eurographics Proc.)*, volume 22, pages 401–410, New York, NY, USA, 2003. ACM Press.

[49] Jos Stam. Stable fluids. In *SIGGRAPH '99: Proceedings of the 26th annual conference on Computer graphics and interactive techniques*, pages 121–128, New York, NY, USA, 1999. ACM Press/Addison-Wesley Publishing Co.

[50] Jos Stam. Flows on surfaces of arbitrary topology. In *SIGGRAPH '03: ACM SIGGRAPH 2003 Papers*, pages 724–731, New York, NY, USA, 2003. ACM Press.

[51] D.E. Stewart. Rigid-body dynamics with friction and impact. *SIAM Review*, pages 3–39, 2000.

[52] D.E. Stewart and J. C. Trinkle. An implicit time-stepping scheme for rigid body dynamics with inelastic collisions and coulomb friction. In *International Journal for Numerical Methods in Engineering*, pages 2673–2691, The Atrium, Southern Gate, Chichester P019 8SQ, England, 1996. Wiley.

[53] D.E. Stewart and J. C. Trinkle. An implicit time-stepping scheme for rigid body dynamics with coulomb friction. In *IEEE International Conference on Robotics and Automation*, pages 162–196, 2000.

[54] Daiki Takeshita, Shin Ota, Machiko Tamura, Tadahiro Fujimoto, Kazunobu Muraoka, and Norishige Chiba. Particle-based visual simulation of explosive flames. In *PG '03: Proceedings of the 11th Pacific Conference on Computer Graphics and Applications*, page 482, Washington, DC, USA, 2003. IEEE Computer Society.

[55] Gary D. Yngve, James F. O'Brien, and Jessica K. Hodgins. Animating explosions. In *SIGGRAPH '00: Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, pages 29–36, New York, NY, USA, 2000. ACM Press/Addison-Wesley Publishing Co.

[56] Yongning Zhu and Robert Bridson. Animating sand as a fluid. *ACM Trans. Graph.*, 24(3):965–972, 2005.