SDS 335 Mid-Project Review

Member names: Joe Garcia, Danny Nguyen, James Sullivan

Group TACC IDs: jag7548, dnguy3n, jsull3

In order to gain insight into a fluid dynamical system via simulation we are required foremost to use a numerical technique to solve the problem. While a numerical method does not provide a true solution we can still find an appropriate approximation. We plan to use a finite difference method in order to approximate the function. In using a finite difference method, we need to generate a grid, discretize, and then solve simultaneous algebraic equations. There are various types of grid generations we can use. However, we will focus on a simple quad and triangular grid in two dimensions. This is done in order to gain an understanding of the advantage of using different grid setups. We do this in two dimensions for simplicity, though may extend it to three or use some shape other than a box in the future. Next, we will discretize using the Taylor series approximation. Depending on the order of the derivatives in our dynamic equation, we will need to solve up to a certain order in the Taylor series expansion. For instance, if a term in the equation required solving a second order derivative then we will need a second order Taylor expansion. We will for the most part ignore higher order terms, but still expect to observe reasonable results for a simulation based in Taylor approximations in this manner.

The simple quad grid uses two for loops to divide up the space of our simulation (the box) in x and y with a prescribed resolution (e.g. if our box has length L, we divide the x axis into n segments of size L/n and do the same for y). The triangular grid is achieved through first performing shear mapping on the matrix that contains the vertex coordinates of the quad grid

and then splitting the resulting parallelograms into triangles. While checking to see if a point in the simulation is within a box in the quad grid is relatively simple (e.g. with an if statement in x and one in y), for the triangle grid we are working on a more detailed method. This entails taking a given point in the simulation and drawing a line from the point to the center of the given triangle and checking so see if this line intersects the lines formed by the edges of the triangle.

The monte carlo sampling options are simple/stratified random sampling, systematic sampling, and importance sampling. Simple sampling is the same as the type used in the pi approximation homework, but using sequential methods. We intend to parallelize the simple sampling as we did in the pi approximation over different chunks, and this is what is called stratified random sampling. Systematic random sampling is similar, but is more efficient and samples every kth element from the total number of elements in the sample size. Importance sampling accounts for more extreme values of the physical parameter of importance by sampling more around them than in places where variation in the parameter is less. This will be especially significant for our problem for any sinks or sources we choose to place in our physical problem.

In terms of what we still have to finish, there are two main points, namely developing a simplified Navier-Stokes equation solver and incorporating all parts of our program to solve a certain physical problem. For the solver, we intend to use crude monte carlo in each grid element to estimate the average value of the physical quantity of interest in that grid element (e.g. temperature, velocity). We will use the explicitly discretized Navier-Stokes equations for unsteady viscous flow evaluated at equidistantly spaced points in our grid. This is a parabolic partial differential equation that is second order in space and first order in time. We can use the

forward and backwards difference methods discussed for computing derivatives in class to discretize the spatial derivatives of the Taylor approximations to the function of interest at the central point of the grid (e.g. a simplified equation of state or an averaged velocity of flow value.). Using these discretized derivatives, we can find the discretized second spatial derivatives for each coordinate direction, which will give us something like the discretized laplacian in two dimensions. We can prescribe the timestep to account for the change in time for the time derivative in the PDE, and the only remaining unknown is the Taylor approximation after it has been evolved one timestep, which is the quantity we desire.

Having accounted for the evolution of our system, the final step is to combine everything into one program that is suited to solve a specific problem. First we must choose our problem, suppose the simple one of placing a hot object as a temperature source in the corner of a box and exploring how the temperature has changed after a set time period. At the beginning of this program we will declare all the relevant variables and choose the relevant geometry setup, which will be a 2D box in this case. Then we will choose the grid setup (either quad or triangle) and resolution as well as the sampling method (simple/stratified, systematic or importance) and sample size. Then we will create an initial temperature matrix that describes the distribution throughout the box, which in our case will be constant throughout to some arbitrary but relatively low value, except at the source point which should account for the hot object. Then we will put the initial temperature matrix through the monte carlo Navier-Stokes solver and execute the solver a number of times corresponding to the time resolution we choose. In terms of results for this simple problem, we expect to see the system reach an equilibrium state after a certain amount of time as the temperature gradient is essentially only pointing in one direction, and will

output the degree to which we have reached equilibrium through some parameter which reads the matrix.

Our ultimate goal is to compare the speed of convergence to equilibrium based on changes in the input options specified before the solver is executed. We expect to see faster convergence with higher time resolution, higher grid resolution, and a higher total number of monte carlo sampling points. However, the interesting part comes in when we vary the options we have already created with regard to the grid type and sampling method under the same resolutions and sample sizes. Any substantial variation based on changing these factors would be most interesting and is what we most hope to talk about when we present our results as we finish the project.