CS 542G: The Poisson Problem, Finite Differences

Robert Bridson

November 10, 2008

1 The Poisson Problem

At the end last time, we noticed that the gravitational potential has a zero Laplacian except at the mass points where it is undefined, leading to the possibility of considering it in terms of a **partial differential equation** (PDE). Let's first look at really what happens at the singularity, where the potential has an undefined Laplacian: it turns out we can define it using the Dirac delta function.

The claim is that $\nabla \cdot \nabla -1/\|x\|$ is proportional to the Dirac delta $\delta(x)$. We already saw that $\nabla \cdot \nabla -1/\|x\| = 0$ for $x \neq 0$, so we just have to check the other defining property of the Dirac delta: the integral of $\delta(x)$ is 1. We'll integrate our Laplacian using the Divergence Theorem[1] to avoid dealing with the unpleasantness of the singularity at the origin. In particular, take Ω to be the volume of the unit sphere centred on the origin, with $\partial\Omega$ its boundary (the surface of the sphere). Then:

$$\iiint_\Omega \nabla \cdot \nabla \frac{-1}{\|x\|} = \iint_{\partial\Omega} \left(\nabla \frac{-1}{\|x\|} \right) \cdot \hat{n}$$

Before we worked out that the gradient here is just $x/\|x\|$, and on the surface of this unit sphere it's also obvious that $\hat{n} = x$ and $\|x\| = 1$. Therefore $\nabla -1/\|x\| = x$, and the dot-product with the outward normal is just 1. The integral simplifies to

$$\iiint_\Omega \nabla \cdot \nabla \frac{-1}{\|x\|} = \iint_{\partial\Omega} 1$$

which is just the surface area of the sphere:

$$\iiint_\Omega \nabla \cdot \nabla \frac{-1}{\|x\|} = 4\pi$$

[1] Remember the Divergence Theorem is just one of the multidimensional generalizations of the Fundamental Theorem of Calculus, that the integral of a derivative of a function on an interval is the difference of function values on the endpoints. For the Divergence theorem, the integral $\int_\Omega \nabla \cdot g(x)$ of the divergence of a vector-valued function $g(x)$ over some volume Ω is equal to the boundary integral with the outward normal, $\int_{\partial\Omega} g(x) \cdot \hat{n}$.

Since this integral is a nonzero, finite value, we conclude that we do have something proportional to the Dirac delta:

$$\nabla \cdot \nabla \frac{-1}{\|x\|} = 4\pi\delta(x)$$

We can then rephrase the gravity problem as the following PDE:

$$\begin{aligned} \nabla \cdot \nabla u(x) &= \sum_{i=1}^{n} 4\pi m_i \delta(x - x_i) \\ \lim_{x\to\infty} u(x) &= 0 \qquad \textit{(A so-called boundary condition)} \end{aligned}$$

The actual acceleration on a point mass at x due to the n point masses is then $-G\nabla u$.

This term allows us to fairly easily generalize the n-body gravity problem to a **continuum** problem, where instead of having a finite set of point masses we view mass as spread out continuously through space. Conceptually, we could divide up space into tiny little regions and in each put a point mass with mass equal to the integral of density over that volume: in the limit as the little regions become infinitesimal we are left with the PDE

$$\begin{aligned} \nabla \cdot \nabla u(x) &= 4\pi\rho(x) \\ \lim_{x\to\infty} u(x) &= 0 \end{aligned}$$

where $\phi(x)$ is the mass density at any point in space. We no longer have Dirac deltas or gigantic sums over points (with n large) so in many respects this problem is simpler than the original straightforward point-mass problem!

This is an example of the **Poisson problem**, a special PDE which at its simplest requires the Laplacian of the unknown function u to equal some specified function (in this case $4\pi\rho(x)$). The Poisson problem in different guises comes up in a ridiculous number of different applications: gravity, electrostatics (just replace mass with charge in the above!), many different parts of fluid mechanics, heat diffusion problems, image processing, geometry processing, ...The special case where the specified function is zero comes up in enough places it gets its own name, the **Laplace equation**.

A typical PDE such as the Poisson or Laplace problem also needs boundary conditions to be well-posed. In the case of gravity, this took the form of requiring the potential u to decay to zero out at infinity. More commonly, the PDE is only satisfied on some bounded domain Ω of space, and the boundary conditions arise at the boundary $\partial\Omega$ of that region—for example, in image processing the domain is probably the rectangle of the image, and the boundary is the outer edge of the rectangle. Usually either the value of the function is specified there, called a **Dirichlet** condition (our gravity problem essentially has a Dirichlet condition), or the normal component of the gradient is specified, called a **Neumann** condition. They

could even be mixed:

$$\begin{aligned} \nabla \cdot \nabla u(x) &= f \qquad \text{in } \Omega \\ u(x) &= g \qquad \text{for } x \in D, \text{ the Dirichlet part of } \partial\Omega \\ \nabla u(x) \cdot \hat{n} &= h \qquad \text{for } x \in N, \text{ the Neumann part of } \partial\Omega \end{aligned}$$

Other more complicated, possibly nonlinear, boundary conditions occasionally crop up, but we won't get into them. In this course we'll focus on just the Dirichlet case, which in some ways is the simplest.

Getting back to gravity, this provides with an alternative method of approximating gravitational acceleration—and extending numerical methods to the continuum case. We split up space into a grid or some other nice division; estimate the density of matter ρ in each grid cell (perhaps just by summing the masses of points in each cell, divided by the volume of the cell); then approximately solve the PDE on that grid. This can get around having to deal with unstructured point sets—we transfer the problem to a nice grid of our own choosing instead—and the whole business of building trees for Barnes-Hut etc. This can be thought of as a **Particle-in-Cell** method, where we start with particles (mass points), transfer their properties (mass) to a grid, solve a problem on the grid, then use that to update the particles (find the gradient to get their accelerations). We mentioned the Fast Multipole Method before as a generalization of Barnes-Hut: one of the other tricks commonly used with FMM is exactly this sort of thing, solving the problem of interacting distant clusters on a regular grid instead of on unstructured trees.

Going the other way, the derivation from gravity gives us another tool: if we have a Poisson problem from some other application, we could solve it by approximating the right-hand-side of the PDE with a sum over a set of n chosen "mass" points, and then use something like Barnes-Hut to get an approximate answer efficiently. This sidesteps the need of creating a grid or mesh, which sometimes can be painful.

2 Finite Differences

Now that we've argued it can be worthwhile to directly approximate the solution of a PDE, we're left with the question of how to do it. The simplest tool at our disposal is something called **Finite Differences**, which are immediately derived from Taylor series.

We begin by laying down a regular Cartesian grid covering the domain. For the moment we'll assume the domain is in fact rectangular, but there are ways to work around the cases where it's not which we'll mention later in passing. Through an appropriate translation and/or rotation of coordinates, the coordinates of grid point i, j, k will be $x_{ijk} = (i\Delta x, j\Delta x, k\Delta x)$, where Δx is the grid spacing (which

I'm assuming is equal on each axis, so grid cells are cubes). We'll store approximate values of the solution u at these grid points, letting us represent in a 3D array for example; we can always use interpolation to estimate u between grid points. We'll call u_{ijk} the approximate value at x_{ijk}.

(Aside on terminology: this process of taking a continuous function in the original PDE and replacing it with a finite set of discrete values is called **discretization**. The term also refers to all of the following process of taking the continuous PDE and replacing it with a finite set of discrete equations.)

Writing the Laplacian operator out with partial derivatives, the Poisson problem is:

$$\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} + \frac{\partial^2 u}{\partial z^2} = f$$

plus boundary conditions. We need some way to approximate these derivatives just from the values of u on the grid. Let's focus on the first, the second derivative of u, and forget about the other dimensions for the time being.

As our first try, let's estimate the first derivative u' in 1D, from grid points $u_i = u(x_i) = u(i\Delta x)$. Taylor series show:

$$u_{i+1} = u_i + u_i'\Delta x + \frac{1}{2}u_i''\Delta x^2 + O(\Delta x^3)$$

Rearranging this and simplifying the error term gives:

$$u_i' = \frac{u_{i+1} - u_i}{\Delta x} + O(\Delta x)$$

This is a finite difference: the quantity $(u_{i+1} - u_i)/\Delta x$ which only depends on grid values is a **first-order accurate** estimate of the first derivative at x_i. We say first-order because the exponent of Δx in the $O(\Delta x)$ error term is one.

A similar Taylor series shows:

$$u_{i-1} = u_i + u_i'(-\Delta x) + \frac{1}{2}u_i''(-\Delta x)^2 + O(\Delta x^3)$$

and thus another first-order finite difference is:

$$u_i' = \frac{u_i - u_{i-1}}{\Delta x} + O(\Delta x)$$

Note that both differences we've seen are **one-sided**: we used the value of u at x_i and to one side, either u_{i+1} or u_{i-1}, to approximate the derivative at x_i.

We can get somewhat higher accuracy with a **central finite difference**, where we use values from both sides to estimate the derivative in the centre. Subtracting the two Taylor series above gives:

$$u_{i+1} - u_{i-1} = 2u_i'\Delta x + O(\Delta x^3)$$

with the second order term cancelling out (it's easy to see the third order term unfortunately doesn't cancel). This cancellation of terms is common with central differences, usually lending them higher accuracy. Rearranging, we get a more accurate approximation of the first derivative:

$$u_i' = \frac{u_{i+1} - u_{i-1}}{2\Delta x} + O(\Delta x^2)$$

This is **second-order accurate**, since the exponent in the error term is two.

Let's now turn to estimating the second derivative, u''. Adding the two Taylor series from above gives a different cancellation:

$$u_{i+1} + u_{i-1} = 2u_i u_i'' \Delta x^2 + O(\Delta x^4)$$

We haven't written it out, but the third order terms (in fact, any odd order term) cancels as well, giving the $O(\Delta x^4)$ remainder here. Rearranging this gives:

$$u_i'' = \frac{u_{i+1} - 2u_i + u_{i-1}}{\Delta x^2} + O(\Delta x^2)$$

This is a second-order accurate central finite difference estimate.

Finally, putting this together in multiple dimensions, we get an approximation for the Poisson problem at grid point x_{ijk}:

$$\frac{u_{i+1jk} - 2u_{ijk} + u_{i-1jk}}{\Delta x^2} + \frac{u_{ij+1k} - 2u_{ijk} + u_{ij-1k}}{\Delta x^2} + \frac{u_{ijk+1} - 2u_{ijk} + u_{ijk-1}}{\Delta x^2} = f_{ijk} + O(\Delta x^2)$$

That is, if we take the exact solution and plug it into this finite difference equation, we're left with an error term of $O(\Delta x^2)$. In the limit as we refine the grid, letting $\Delta x \to 0$, the error goes to zero: we call such a method **consistent**.

The finite difference method for solving the PDE turns this around. We'll set

$$\frac{u_{i+1jk} - 2u_{ijk} + u_{i-1jk}}{\Delta x^2} + \frac{u_{ij+1k} - 2u_{ijk} + u_{ij-1k}}{\Delta x^2} + \frac{u_{ijk+1} - 2u_{ijk} + u_{ijk-1}}{\Delta x^2} = f_{ijk}$$

without the $O()$ error term as a set of equations that our unknown grid values of u must satisfy at every grid point, and hope that the discrete solution is an accurate approximation to the true solution. Right now this is just a hope—the fact that the true solution approximately satisfies the discrete equations does not imply the discrete solution has to approximate the true solution, though next time we will prove this is indeed the case for this particular scheme.

We said this equation should hold at every grid point, i.e. for every i, j, k. However, obviously we'll have a bit of a problem at the edges of the grid. If $i = 0$, for example, the equation refers to $u_{i-1,j,k}$ which doesn't lie in the grid anymore. It's here where the boundary conditions will enter into the discretization. For Dirichlet conditions, at least treated the simplest possible way, we simply replace every instance of a

discrete u which is not in the actual domain (either off the edge of the grid, or in the grid but not in the domain due to the domain not being rectangular) with the value prescribed to it by the Dirichlet boundary condition.

We only write down equations at grid points where we have an unknown variable to solve for. This gives one equation for each unknown, and inspection of the equation above shows they are all linear equations. This boils down to a linear system, which we again hope will be solvable: next time we'll try to establish this properly.