Finite Difference Solution of the Heat Equation

Adam Powell

22.091 March 13–15, 2002

In example 4.3 (p. 10) of his lecture notes for March 11, Rodolfo Rosales gives the constant-density heat equation as:

$$c_p \rho \frac{\partial T}{\partial t} + \nabla \cdot \vec{q} = \dot{q}, \tag{1}$$

where I have substituted the constant pressure heat capacity c_p for the more general c, and used the notation $\vec{q}$ for the heat flux vector and $\dot{q}$ for heat generation in place of his $\mathbf{Q}$ and s. This is based on the more general equation for enthalpy conservation:

$$\frac{\partial H}{\partial t} + \nabla \cdot \vec{q} = \dot{q}, \tag{2}$$

where H is the enthalpy per unit volume, typically given in J/m^3.

This equation is closed by the relationship between flux and temperature gradient:

$$\vec{q} = -k \nabla T \tag{3}$$

where k is the thermal conductivity in watts/m·K. This results in the equation:

$$c_p \rho \frac{\partial T}{\partial t} - \nabla \cdot (k \nabla T) = \dot{q}, \tag{4}$$

or for constant k,

$$c_p \rho \frac{\partial T}{\partial t} - k \nabla^2 T = \dot{q}, \tag{5}$$

and defining the thermal diffusivity as $\alpha \equiv k/\rho c_p$ (Rosales uses ν):

$$\frac{\partial T}{\partial t} - \alpha \nabla^2 T = \frac{\dot{q}}{\rho c_p}. \tag{6}$$

Equations 4 through 6 are closed and first order in time and second order in space, so they require one boundary condition in time (called an *initial condition*), and surrounding boundary conditions in space—two such conditions in 1-D, for example.

The remainder of this lecture will focus on solving equation 6 numerically using the method of finite differences.

The Finite Difference Method

Because of the importance of the diffusion/heat equation to a wide variety of fields, there are many analytical solutions of that equation for a wide variety of initial and boundary conditions. However, one very often runs into a problem whose particular conditions have no analytical solution, or where the analytical solution is even more difficult to implement than a suitably accurate numerical solution. Here we will discuss one particular method for anaytical solution of partial differential equations called the finite difference method.

The finite difference method begins with the discretization of space and time such that there is an integer number of points in space and an integer number of times at which we calculate the field variable(s), in this case just the temperature. The resulting approximation is shown schematically in figure 1. For simplicity here we'll assume equal spacing of the points x_i in one dimension with intervals of size $\Delta x = x_{i+1} - x_i$, and equal spacing of the timesteps t_n at intervals of $\Delta t = t_{n+1} - t_n$. This simplifies the system considerably, since instead of tracking a smooth function at an infinite number of points, one just deals with a finite number of temperature values at a finite number of locations and times.

Figure 1: Finite difference discretization of space and time. The thin line segments represent the finite difference approximation of the function.

Based on this discretization and approximation of the function, we then write the following approximations of its derivatives in time and space:

$$\left.\frac{\partial T}{\partial t}\right|_{x_i, t_{n+1/2}} \simeq \frac{T_{i,n+1} - T_{i,n}}{\Delta t} \tag{7}$$

$$\left.\frac{\partial T}{\partial x}\right|_{x_{i+1/2}, t_n} \simeq \frac{T_{i+1,n} - T_{i,n}}{\Delta x} \tag{8}$$

We can take the latter of these derivatives one step further, by taking differences of the derivative approximations, to arrive at an approximation for the second derivative:

$$\left.\frac{\partial^2 T}{\partial x^2}\right|_{x_i, t_n} \simeq \frac{\frac{\partial T}{\partial x}\big|_{x_{i+1/2}, t_n} - \frac{\partial T}{\partial x}\big|_{x_{i-1/2}, t_n}}{\Delta x} \simeq \frac{T_{i-1,n} - 2T_{i,n} + T_{i+1,n}}{(\Delta x)^2} \tag{9}$$

Equations 7 and 9 give us sufficient finite difference approximations to the derivatives to solve equation 6.

Forward Euler time integration, a.k.a. explicit timestepping

The forward Euler algorithm, also called explicit timestepping, uses the field values of only the previous timestep to calculate those of the next. In this case, this means that the spatial derivatives will be evaluated at timestep n and the time derivatives at $n + \frac{1}{2}$. This algorithm is very simple, in that each new temperature at timestep $n + 1$ is calculated independently, so it does not require simultaneous solution of equations, and can even be performed in a spreadsheet. On the other hand, the forward Euler algorithm is unstable for large timesteps, as we will see shortly, and is not as accurate as other algorithms such as will also be visited presently.

When the finite difference approximations are inserted into equation 6, for 1-D heat conduction we get:

$$\frac{T_{i,n+1}-T_{i,n}}{\Delta t}-\alpha\frac{T_{i-1,n}-2T_{i,n}+T_{i+1,n}}{(\Delta x)^2}=\frac{\dot{q}}{\rho c_p}. \tag{10}$$

We solve this for the temperature at the new timestep $T_{i,n+1}$ to give:

$$T_{i,n+1}=T_{i,n}+\Delta t\left[\alpha\frac{T_{i-1,n}-2T_{i,n}+T_{i+1,n}}{(\Delta x)^2}+\frac{\dot{q}}{\rho c_p}\right]. \tag{11}$$

At this point, it is convenient to define the mesh Fourier number:

$$\mathrm{Fo}_M\equiv\frac{\alpha\Delta t}{(\Delta x)^2}. \tag{12}$$

Since the time to steady state over a distance L is approximately L^2/α, this mesh Fourier number can roughly be thought of as the ratio of the timestep size to the time required to equilibrate one space interval of size Δx. This permits further simplification of equation 11:

$$T_{i,n+1}=T_{i,n}+\mathrm{Fo}_M(T_{i-1,n}-2T_{i,n}+T_{i+1,n})+\frac{\Delta t}{\rho c_p}\dot{q}=(1-2\mathrm{Fo}_M)T_{i,n}+2\mathrm{Fo}_\mathrm{M}\frac{T_{i-1,n}+T_{i+1,n}}{2}+\frac{\Delta t}{\rho c_p}\dot{q}. \tag{13}$$

Ignoring the source term for a moment, this states that the new temperature at x_i is the *weighted average* of the old temperature at that point and the average of its neighbors' old temperatures. If $\mathrm{Fo}_M=0$, then $T_{i,n+1}=T_{i,n}$; if $\mathrm{Fo}_M=\frac{1}{2}$ then $T_{i,n+1}=\frac{1}{2}(T_{i-1,n}+T_{i+1,n})$; otherwise it will be somewhere in between.

But we can see very easily that if $\mathrm{Fo}_M>\frac{1}{2}$, then the algorithm is artificially unstable, as mentioned above. Qualitatively speaking, in this case the new temperature value "overshoots" the average of its neighbors. This is illustrated in figure 2, where it is shown that a small oscillation grows exponentially for $\mathrm{Fo}_M=0.7$. Thus we have a *stability criterion*: for explicit timestepping in one dimension, the mesh Fourier number must be no greater than $\frac{1}{2}$.

Figure 2: "Overshoot" of the average neighboring temperature for $\mathrm{Fo}_M>0.5$; growth of an oscillation for $\mathrm{Fo}_M=0.7$ with T_0 and T_5 fixed as boundary conditions (the dark curve is the initial condition).

This stability criterion limits the timestep size to $\Delta t\leq\frac{(\Delta x)^2}{2\alpha}$. To design a simulation, one can first construct a mesh in space, then choose Δt to exactly satisfy the criterion by using the minimum value of $\frac{(\Delta x)^2}{2\alpha}$ in the simulation. Note that if one wants to make the simulation more accurate by reducing Δx by a factor of 2, one must also reduce Δt by a factor of four, increasing by a factor of eight the total number of points×timesteps at which the field value is calculated.

In two space dimensions plus time, with equal grid spacing $\Delta y = \Delta x$, and indices i and j in the x- and y-directions, equation 13 becomes:

$$T_{i,j,n+1} = (1 - 4\mathrm{Fo}_M)T_{i,j,n} + 4\mathrm{Fo_M}\frac{T_{i-1,j,n} + T_{i+1,j,n} + T_{i,j-1,n+1} + T_{i,j+1,n+1}}{4} + \frac{\Delta t}{\rho c_p}\dot{q}. \tag{14}$$

Now this weighted average overshoots the average of the previous neighboring temperatures and causes exponential growth of oscillations for $\mathrm{Fo}_M > \frac{1}{4}$, so the stability criterion must be modified accordingly.

In three dimensions, that criterion becomes $\mathrm{Fo}_M \leq \frac{1}{6}$.

Example: cooling of an HDPE sheet A high-density polyethylene sheet 1 cm thick is cooled from 150° C to the ambient temperature of 20°C, with cooling fans creating sufficient heat transfer to assume the surface temperature of the sheet rapidly reaches the ambient temperature. We would like to calculate the temperature profile across the sheet. HDPE has the following properties:

- Thermal conductivity: $k = 0.64$ W/m-K
- Density: $\rho = 920$ kg/m^3
- Heat capacity: $c_p = 2300$ J/kg-K

These properties give us the thermal diffusitivy $\alpha = k/\rho c_p = 3.02 \times 10^{-7}\,\frac{\mathrm{m}^2}{\mathrm{s}}$, so the timescale of cooling the whole sheet is approximately

$$t \simeq \frac{L^2}{\alpha} = \frac{(0.01\mathrm{m})^2}{3.02 \times 10^{-7}\,\frac{\mathrm{m}^2}{\mathrm{s}}} = 331\text{seconds}. \tag{15}$$

If we divide the sheet thickness into five intervals each 2mm thick, the timestep given by the mesh Fourier number of one half is

$$\Delta t = \frac{\mathrm{Fo}_M(\Delta x)^2}{\alpha} = \frac{\frac{1}{2}(0.002\mathrm{m})^2}{3.02 \times 10^{-7}\,\frac{\mathrm{m}^2}{\mathrm{s}}} = 6.61\text{seconds}. \tag{16}$$

A spreadsheet with these properties and timestep calculation is on the 22.091 website. That spreadsheet has a temperature array including the initial and boundary conditions, and equation 13 in the interior. It also has a separate array simulating half the thickness with $\Delta x = 1$mm, and a symmetry boundary condition at $x = 0.5$ cm.

(Semi-)Implicit timestepping

Instabilities are a pain, and often we want to use larger timesteps which are constrained by the physics, not the numerics. Toward this end, we can use the backward Euler algorithm, also called implicit timestepping, which differs from forward Euler (equation 10) in that the spatial derivatives are calculated in the new timestep.

$$\frac{T_{i,n+1} - T_{i,n}}{\Delta t} - \alpha\frac{T_{i-1,n+1} - 2T_{i,n+1} + T_{i+1,n+1}}{(\Delta x)^2} = \frac{\dot{q}}{\rho c_p}. \tag{17}$$

Unfortunately, the new temperature field values may not be calculated independently, so one must solve simultaneous equations in order to make this work.

Both forward Euler and backward Euler time integration are first-order accurate in the timestep size, that is, the error is proportional to Δt. A straightforward mixture of the two called Crank-Nicholson time integration, or semi-implicit timestepping, improves the accuracy to second order by averaging the spatial derivatives in the old and new timesteps in a "trapezoid rule" fashion:

$$\frac{T_{i,n+1} - T_{i,n}}{\Delta t} - \alpha \frac{T_{i-1,n} - 2T_{i,n} + T_{i+1,n} + T_{i-1,n+1} - 2T_{i,n+1} + T_{i+1,n+1}}{2(\Delta x)^2} = \frac{\dot{q}_{i,n} + \dot{q}_{i,n+1}}{2\rho c_p}. \tag{18}$$

For small timesteps, this gives much better accuracy than the explicit and (fully) implicit algorithms. However, it is not as good as implicit timestepping for very long timesteps. To take an extreme example, if one uses a timestep of 330 seconds in the above polyethylene cooling simulation, the implicit scheme will converge to the steady-state temperature profile, as it should be, whereas the semi-implicit scheme will leave the temperatures somewhere between the original and steady-state profiles.

3-D Cahn-Hilliard Demonstration

At the end of class, a brief demonstration of Cahn-Hilliard dynamics was shown, which is part of the `Illuminator` graphics library. That library and its notation are described in detail at `http://lyre.mit.edu/ powell/illumi` What follows is a brief review of the math using conventional vector notation.

The free energy of a body Ω which separates naturally into two phases is given by a homogeneous part and a "gradient penalty" term:

$$\mathcal{F} = \int_\Omega \left(frac\alpha 2 \left|\nabla C\right|^2 + \beta \Psi(C) \right) dV, \tag{19}$$

where the second term is the homogeneous part, and α and β are constants which scale the terms to approximately fit the real system. This gradient penalty term gives the system a diffuse interface instead of a sharp one, to reflect the physical reality that the interface is never perfectly sharp. The diffuse nature of real interfaces results in such phenomena as solute trapping during rapid solidification, and the finite size of domains at the very beginning of spinodal decomposition phase separation, the latter of which this gradient penalty formulation was first used to describe.

The homogeneous free energy has a common tangent at the equilibrium compositions of the two phases. Here a simplified homogeneous free energy is used with the functional form:

$$\Psi = C^2(1-C)^2, \tag{20}$$

which is a simple fourth-order polynomial with minima at $C = 0$ and $C = 1$, and a local maximum at $C = 0.5$, so this material separates into phases with concentrations at the minima.

At the interface, C varies smoothly between 0 in one phase and 1 in the other. In that transition, the homogeneous term is higher than the surroundings, so its minimization will drive the interface to become thinner. However, as it becomes thinner, the gradient becomes steeper, and the gradient penalty term rises; minimizing that term provides a driving force to make the interface thicker, with a more gradual transition between phases. The system reaches an equilibrium with an interface of finite thickness ϵ and total energy per unit area γ where

$$\epsilon \sim \sqrt{\frac{\alpha}{\beta}} \text{ and } \gamma \sim \sqrt{\alpha\beta}. \tag{21}$$

These thermodynamics relate to the kinetics of the system through the chemical potential μ. The diffusion equation can be written in terms of chemical potential and a mobility κ (replacing the diffusivity D) as follows:

$$\frac{\partial C}{\partial t} = \nabla \cdot (\kappa \nabla \mu). \tag{22}$$

We must close this equation by expressing μ in terms of C. The chemical potential comes from the free energy; because of the gradient penalty, we must use variational principles to express it, which can be written:

$$\mu = \frac{\delta \mathcal{F}}{\delta C} = -\alpha \nabla^2 C + \beta \Psi'(C). \tag{23}$$

With this chemical potential from, and constant κ, α and β, equation 22 becomes

$$\frac{\partial C}{\partial t} = \kappa \left(\alpha \nabla^2 \nabla^2 C + \beta \nabla^2 \Psi'(C) \right). \tag{24}$$

There are very few analytical solutions for this equation, so we turn to numerics to solve it. This is a fourth-order nonlinear equation, but with ∇^2 operators which we can approximate on a finite difference grid using equation 9. The demonstration called `chts` shown in class (which comes with `Illuminator`) solves this using semi-implicit timestepping in 3-D, and displays the result as a set of contour surfaces of constant concentration.