Finite Difference Solutions For Parabolic Differential Equations (I.E., Heat Equation)

Jonathan B. Snively
Embry-Riddle Aeronautical University
Contents

• Diffusion and Heat Equations

• Numerical Solutions
The Diffusion Equation

The diffusion equation describes the diffusive flux of some quantity over time. Physically, it is a result of Fick’s law:

\[ \vec{F} = -\kappa \nabla u \]

This flux of the quantity being diffused is proportional to the gradient of the quantity as it varies in space. Therefore, after substitution into a continuity equation:

\[ \frac{\partial u}{\partial t} = -\nabla \cdot (-\kappa \nabla u) = \nabla \cdot (\kappa \nabla u) \]

This general diffusion equation allows for the possibility that the diffusivity \( \kappa \) (with physical units \([m^2/s]\)) may vary spatially – If that is not the case, we can simplify...
The Heat Equation

The “heat equation” describes diffusion where the diffusivity parameter $\kappa$ does not vary spatially:

$$\frac{\partial u}{\partial t} = \kappa \nabla^2 u$$

The heat equation is often used to describe simple cases of thermal or momentum diffusion (i.e., associated with thermal conductivity and molecular viscosity).

This is a parabolic differential equation, for which we can construct simple finite difference solutions.

In some cases, we can also obtain analytical solutions, useful for testing models (and homework assignments!)
Analytical Example

The heat equation can be written in one dimension:

\[
\frac{\partial u}{\partial t} = \kappa \frac{\partial^2 u}{\partial x^2}
\]

Given an initial condition of: \( u(x) = \sin(kx) \)

We can assume a decaying exponential solution:

\[
u(x, t) = e^{-t/\tau} \sin(kx)\]

We find that:

\[
\frac{1}{\tau} e^{-t/\tau} \sin(kx) = k^2 \kappa e^{-t/\tau} \sin(kx)
\]

Therefore

\[
\tau = \frac{1}{k^2 \kappa}
\]

is the characteristic relaxation time.
**Interpreting Diffusion**

*Put simply:* Diffusion results in damping of perturbations over time, smoothing of steep gradients, and removal of small-scale features.

Since the relaxation time is given by:

\[ \tau = \frac{1}{k^2 \kappa} \]

... it is dependent on wavelength such that short wavelengths experience much faster diffusion time-scales!

\[ \tau = \frac{\chi^2}{4\pi^2 \kappa} \]

We will see (later) that molecular diffusion processes in the atmosphere result in significant and rapid damping of small-scale features in the middle and upper atmosphere.
Contents

• Diffusion and Heat Equations

• Numerical Solutions
  • Forward Euler (Explicit)
  • Backward Euler (Implicit)
  • Crank-Nicolson (Implicit)
Forward Euler Discretization

The heat equation can be written in one dimension:

\[
\frac{\partial u}{\partial t} = \kappa \frac{\partial^2 u}{\partial x^2}
\]

Using a forward-in-time Euler method discretization for the LHS, and a centered second-order difference for the RHS:

\[
\frac{u_j^{n+1} - u_j^n}{\Delta t} = \kappa \frac{u_{j+1}^n - 2u_j^n + u_{j-1}^n}{(\Delta x)^2}
\]

Rearranging, we obtain the explicit scheme:

\[
u_j^{n+1} = u_j^n + \frac{\kappa \Delta t}{(\Delta x)^2} (u_{j+1}^n - 2u_j^n + u_{j-1}^n)
\]
The forward Euler method is first order in time, and second order in space. It is an “explicit” method, which considers data points from only the present time-step when calculating the future.

\[
u_{j}^{n+1} = u_{j}^{n} + \frac{\kappa \Delta t}{(\Delta x)^2} (u_{j+1}^{n} - 2u_{j}^{n} + u_{j-1}^{n})\]
Stability of the Explicit Forward Euler Method

To confirm stability, we assume a Fourier mode solution:

\[ u^n_j = \hat{u}^n e^{ikx_j} \]

We can then divide by \( e^{ikx_j} \) to obtain:

\[ \hat{u}^{n+1} e^{ikx_j} = \hat{u}^n e^{ikx_j} + \frac{\kappa \Delta t}{(\Delta x)^2} \hat{u}^n (e^{ikx_j+1} - 2e^{ikx_j} + e^{ikx_j-1}) \]

\[ \hat{u}^{n+1} = \hat{u}^n + \frac{\kappa \Delta t}{(\Delta x)^2} \hat{u}^n (e^{i\Delta x} - 2 + e^{-i\Delta x}) \]

\[ \hat{u}^{n+1} = \hat{u}^n \left( 1 + \frac{2\kappa \Delta t}{(\Delta x)^2} \left( \frac{1}{2} e^{i\Delta x} + \frac{1}{2} e^{-i\Delta x} - 1 \right) \right) \]

\[ g(\Delta x, \Delta t, \kappa) \]
Stability of Forward Euler

The amplification factor \( g(\Delta x, \Delta t, \nu) \) can then be expressed:

\[
g = 1 + \frac{2\kappa \Delta t}{(\Delta x)^2} (\cos(k \Delta x) - 1)
\]

Or, in simplified form:

\[
g = 1 - \frac{4\kappa \Delta t}{(\Delta x)^2} \sin^2 \left( \frac{k \Delta x}{2} \right)
\]

The amplitude of the amplification factor remains \( \leq 1 \) for:

\[
\left| 1 - \frac{4\kappa \Delta t}{(\Delta x)^2} \sin^2 \left( \frac{k \Delta x}{2} \right) \right| \leq 1 \quad \text{...or} \quad \frac{2\kappa \Delta t}{(\Delta x)^2} \leq 1
\]

Thus:

\[
\Delta t \leq \frac{(\Delta x)^2}{2\kappa}
\]
Courant-Friedrichs-Lewy Stability Conditions

For finite difference solutions of diffusion problems, the non-dimensional Courant-Friedrichs-Lewy (CFL) number is given by:

\[ \text{CFL} = \frac{\kappa \Delta t}{(\Delta x)^2} \]

For the explicit forward Euler method, stability requires CFL ≤ 0.5!

In many cases, this is a very severe time constraint, such that alternate methods are often pursued, specifically implicit methods which are often stable for any time step.

Caveat: Implicit methods will require more computational time per step, but fewer steps!
Backward Euler Discretization

Dr. Zettergren is a fan of the following simple solution:

Using a backward-in-time Euler discretization for the LHS, and a centered second-order difference for the RHS:

\[
\frac{u^n_j - u^{n-1}_j}{\Delta t} = \kappa \frac{u^n_{j+1} - 2u^n_j + u^n_{j-1}}{(\Delta x)^2}
\]

Rearranging, we obtain the *implicit* scheme:

\[
u^{n+1}_j = u^n_j + \frac{\kappa \Delta t}{(\Delta x)^2} (u^{n+1}_{j+1} - 2u^{n+1}_j + u^{n+1}_{j-1})
\]

This is the first “implicit” solution that we have considered!
The backward Euler method is also first order in time, and second order in space. As an implicit method, solutions for grid points in the future time step are dependent on each other in addition to the present step.

\[
    u_{j}^{n+1} = u_{j}^{n} + \frac{\kappa \Delta t}{(\Delta x)^2} (u_{j+1}^{n+1} - 2u_{j}^{n+1} + u_{j-1}^{n+1})
\]
Crank-Nicolson Method

The backward and forward Euler methods can be combined in the Crank-Nicolson method, incorporating both present and future time-step data in the solution:

\[
 u_{j}^{n+1} = u_{j}^{n} + \frac{\kappa \Delta t}{2(\Delta x)^{2}} (u_{j+1}^{n+1} - 2u_{j}^{n+1} + u_{j-1}^{n+1})
\]

\[
 + \frac{\kappa \Delta t}{2(\Delta x)^{2}} (u_{j+1}^{n} - 2u_{j}^{n} + u_{j-1}^{n})
\]
Stability of the Implicit Crank-Nicolson Method

To confirm stability, we assume a Fourier mode solution:

\[ \hat{u}^{n+1} e^{ikx_j} = \hat{u}^n e^{ikx_j} + \frac{\kappa \Delta t}{2(\Delta x)^2} \hat{u}^{n+1} (e^{ikx_{j+1}} - 2e^{ikx_j} + e^{ikx_{j-1}}) \]

\[ + \frac{\kappa \Delta t}{2(\Delta x)^2} \hat{u}^n (e^{ikx_{j+1}} - 2e^{ikx_j} + e^{ikx_{j-1}}) \]

We can then divide by \( e^{ikx} \) to obtain:

\[ \hat{u}^{n+1} = \hat{u}^n + \frac{\kappa \Delta t}{2(\Delta x)^2} \hat{u}^{n+1} (e^{ik\Delta x} - 2 + e^{-ik\Delta x}) \]

\[ + \frac{\kappa \Delta t}{2(\Delta x)^2} \hat{u}^n (e^{ik\Delta x} - 2 + e^{-ik\Delta x}) \]
Stability of the Implicit Crank-Nicolson Method

\[ \hat{u}^{n+1} \left( 1 - \frac{\kappa \Delta t}{2(\Delta x)^2} \left( e^{ik\Delta x} - 2 + e^{-ik\Delta x} \right) \right) = \hat{u}^n \left( 1 + \frac{\kappa \Delta t}{2(\Delta x)^2} \left( e^{ik\Delta x} - 2 + e^{-ik\Delta x} \right) \right) \]

The amplification factor \( g(\Delta x, \Delta t, v) \) can then be expressed:

\[ g = \frac{1 - \frac{2\kappa \Delta t}{(\Delta x)^2} \sin^2 \left( \frac{k\Delta x}{2} \right)}{1 + \frac{2\kappa \Delta t}{(\Delta x)^2} \sin^2 \left( \frac{k\Delta x}{2} \right)} \]

The amplification factor \( g(\Delta x, \Delta t, v) \) is always \( \leq 1 \)! Therefore, the implicit Crank-Nicolson method is unconditionally stable (albeit not unconditionally accurate).
Implicit Method Implementation

As shown, the implicit methods result in systems involving a tri-diagonal matrix (which can be solved easily using “textbook” techniques or, simply, MATLAB)

\[
\begin{align*}
    u_j^{n+1} - \frac{\kappa \Delta t}{2(\Delta x)^2} (u_{j+1}^{n+1} - 2u_j^{n+1} + u_{j-1}^{n+1}) &= \\
    u_j^n + \frac{\kappa \Delta t}{2(\Delta x)^2} (u_{j+1}^n - 2u_j^n + u_{j-1}^n)
\end{align*}
\]

In the form shown above, the “future” time step has been separated from the “present”.
Implicit Method Implementation

\[
\begin{align*}
    u_j^{n+1} &= -\frac{\kappa \Delta t}{2(\Delta x)^2} (u_{j+1}^{n+1} - 2u_j^{n+1} + u_{j-1}^{n+1}) \\
    u_j^n &= \frac{\kappa \Delta t}{2(\Delta x)^2} (u_{j+1}^n - 2u_j^n + u_{j-1}^n)
\end{align*}
\]

Already Known – “Store” as \( d_j \)

Set new constant \( C = \frac{\kappa \Delta t}{(\Delta x)^2} \)

Reconstruct system in terms of \( C \):

\[
\begin{align*}
    u_j^{n+1} &= \frac{C}{2} (u_{j+1}^{n+1} - 2u_j^{n+1} + u_{j-1}^{n+1}) = d_j \\
    -\frac{C}{2} u_{j+1}^{n+1} + (1 + C)u_j^{n+1} - \frac{C}{2} u_{j-1}^{n+1} &= d_j
\end{align*}
\]

This system is ready to solve!