Heat Flow via Crank–Nicolson

An Improved Leap Frog

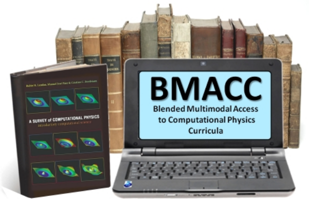
Rubin H Landau

Sally Haerer, Producer-Director

Based on *A Survey of Computational Physics* by Landau, Páez, & Bordeianu

with Support from the National Science Foundation

Course: **Computational Physics II**



Problem: How Does a Bar Cool?

Insulated Metallic Bar Touching Ice

- Aluminum bar, $L = 1$ m, w along x
- Insulated along length, ends in ice ($T = 0$ C)
- Initially $T = 100$ C
- How does temperature vary in space and time?

CN Improved Algorithm: Split t Step $\Rightarrow \partial_t$ FD $\rightarrow$ CD

Problem: Making the First Step Knowing Only $T(x, t = 0)$

- *Split time step* for ∂_t at $t + \Delta t/2$

$$\frac{\partial T}{\partial t}\left(x, t + \frac{\Delta t}{2}\right) \simeq \frac{T(x, t+\Delta t) - T(x,t)}{\Delta t} + O(\Delta t^2) \tag{1}$$

- Bad as FD for $t = t + \Delta t$, Good at $t + \Delta t/2$
- CD ∂_x^2 at $t = t + \Delta t/2$

$$\frac{\partial^2 T}{\partial x^2}\left(x, t + \frac{\Delta t}{2}\right) \simeq \frac{1}{2(\Delta x)^2} \times \tag{2}$$

$$[T(x - \Delta x,\ t + \Delta t) + \ T(x + \Delta x,\ t + \Delta t) - 2T(x,\ t + \Delta t)$$

$$+T(x - \Delta x,\ t) + T(x + \Delta x,\ t) - 2T(x,\ t) + O(\Delta x^2)\Big]$$

Split-Time Discrete Form of Heat Equation

Derivatives in Terms of Differences

$$\frac{\partial T(x,t)}{\partial t} = \frac{K}{C\rho}\frac{\partial^2 T(x,t)}{\partial x^2} \tag{3}$$

$$T_{i,j+1} - T_{i,j} = \frac{\eta}{2}\left[T_{i-1,j+1} - 2T_{i,j+1} + T_{i+1,j+1} + T_{i-1,j} - 2T_{i,j} + T_{i+1,j}\right] \tag{4}$$

$$x = i\Delta x, \quad t = j\Delta t, \quad \eta = \frac{K\Delta t}{C\rho\,\Delta x^2} \tag{5}$$

- Future in terms of present (grid values only, yet mid time):

$$-T_{i-1,j+1} + \left(\frac{2}{\eta} + 2\right) T_{i,j+1} - T_{i+1,j+1} = T_{i-1,j} + \left(\frac{2}{\eta} - 2\right) T_{i,j} + T_{i+1,j} \tag{6}$$

- *Implicit* solution: solve all lattice sites simultaneously
- Leap frog = *explicit*, permits single time step

Arrange Discrete Heat Equation as Matrix Equation

$$\begin{pmatrix} (\frac{2}{\eta}+2) & -1 & & & \\ -1 & (\frac{2}{\eta}+2) & -1 & & \\ & -1 & (\frac{2}{\eta}+2) & -1 & \\ & & \ddots & \ddots & \ddots \end{pmatrix} \begin{pmatrix} T_{1,j+1} \\ T_{2,j+1} \\ T_{3,j+1} \\ \vdots \end{pmatrix} = \begin{pmatrix} T_{0,j+1} + T_{0,j} + (\frac{2}{\eta}-2)\, T_{1,j} + T_{2,j} \\ T_{1,j} + (\frac{2}{\eta}-2)\, T_{2,j} + T_{3,j} \\ T_{2,j} + (\frac{2}{\eta}-2)\, T_{3,j} + T_{4,j} \\ \vdots \end{pmatrix}$$

- LHS future T's = $j+1$
- RHS present T's = j
- RHS ends @ future, OK as BC
- IC: $j = 0$, hot bar, cold ends
- Solve matrix $\rightarrow j = 1$ all x=i
- Advance t, repeat step
- C-N = ↑ precise, stable, work
- vonN stability: all Δt, Δx OK:

$$\xi(k) = \frac{1 - 2\eta \sin^2(k\Delta x/2)}{1 + 2\eta \sin^2(k\Delta x/2)}$$

Special Solution of Tridiagonal Matrix Equations ⊙

$$\begin{pmatrix} d_1 & c_1 & 0 & 0 & \cdots & \cdots & \cdots & 0 \\ a_2 & d_2 & c_2 & 0 & \cdots & \cdots & \cdots & 0 \\ 0 & a_3 & d_3 & c_3 & \cdots & \cdots & \cdots & 0 \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ 0 & 0 & 0 & 0 & \cdots & a_{N-1} & d_{N-1} & c_{N-1} \\ 0 & 0 & 0 & 0 & \cdots & 0 & a_N & d_N \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ \ddots \\ x_{N-1} \\ x_N \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \\ b_3 \\ \ddots \\ b_{N-1} \\ b_N \end{pmatrix}$$

- Libes good for stnd linear eq

$$[A]\,\mathbf{x} = \mathbf{b}$$

- Yet $[A]$ = tridiagonal
- $\Rightarrow \exists$ more robust, faster solution
- $A_{i,j} \Rightarrow N^2$ words, access
- Tridiagonal $\Rightarrow$ only 3 vectors

$$\{d_i\}_{i=1,N},\ \{c_i\}_{i=1,N},\ \{a_i\}_{i=1,N}$$

- **Single subscript $\Rightarrow$ 3N − 2**

Soltn: Coef matrix → *upper triangular*, diagonals =1

- Start: divide 1st eqtn by d_1, subtract $a_2 \times$ 1st eqtn:

$$\begin{pmatrix} 1 & \frac{c_1}{d_1} & 0 & 0 & \cdots & \cdots & \cdots & 0 \\ 0 & d_2 - \frac{a_2 c_1}{d_1} & c_2 & 0 & \cdots & \cdots & \cdots & 0 \\ 0 & a_3 & d_3 & c_3 & \cdots & \cdots & \cdots & 0 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} \frac{b_1}{d_1} \\ b_2 - \frac{a_2 b_1}{d_1} \\ b_3 \end{pmatrix}$$

- Next: divide 2nd eqtn by 2nd diagonal element

$$\begin{pmatrix} 1 & \frac{c_1}{d_1} & 0 & 0 & \cdots & \cdots & \cdots & 0 \\ 0 & 1 & \frac{c_2}{d_2 - a_2 \frac{c_1}{a_1}} & 0 & \cdots & & \cdots & 0 \\ 0 & a_3 & d_3 & c_3 & \cdots & \cdots & \cdots & 0 \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} \frac{b_1}{d_1} \\ \frac{b_2 - a_2 \frac{b_1}{d_1}}{d_2 - a_2 \frac{c_1}{d_1}} \\ b_3 \end{pmatrix}$$

- Repeat steps → upper triangular form

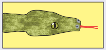
$$\begin{pmatrix} 1 & h_1 & 0 & 0 & \cdots & 0 \\ 0 & 1 & h_2 & 0 & \cdots & 0 \\ 0 & 0 & 1 & h_3 & \cdots & 0 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} p_1 \\ p_2 \\ p_3 \end{pmatrix}$$

- Back substitution ⇒ explicit solution:

$$x_i = p_i - h_i x_{i-1}; \quad i = n-1, n-2, \ldots, 1, \quad x_N = p_N$$

Crank–Nicolson Implementation

HeatCNTridiag.py



CODE

1. Solve C-N linear equations using libe, esp for tridiagonal
2. Check stability for Δx and Δt
3. Contoured surface plot of $T(x, t)$
4. Compare precision and speed: leap-frog *vs* Crank–Nicolson
5. Assume stable, very small Δt = accurate