Solving PDEs with PGI CUDA Fortran
Part 6: More methods for more partial differential equations

Outline

Heat equation in 1D: implicit and Crank-Nicolson schemes. Heat equation in more dimensions: alternating-direction implicit method. Multigrid method. Wave equation in 1D and 2D: strings and drums.

Heat equation in 1D: more schemes

A symbol for the difference operator

$$\delta_x^2 u_j^n \equiv u_{j-1}^n - 2u_j^n + u_{j+1}^n$$

FTCS scheme with Dirichlet boundary conditions

$$u_j^{n+1} = u_j^n + \beta\delta_x^2 u_j^n, \qquad \beta = dt/dx^2$$
$$u_j^{n+1} = u_j^n + \beta\left(u_{j-1}^n - 2\beta u_j^n + u_{j+1}^n\right)$$

Features: 1st-order accurate in time, 2nd-order in space, conditionaly stable ($\beta <= 1/2$)

BTCS scheme (backward-time centered-space)
implicit formula

$$u_j^{n+1} = u_j^n + \beta\delta_x^2 u_j^{n+1}, \qquad \beta = dt/dx^2$$

$$\begin{pmatrix} 1+2\beta & -\beta & & & \\ -\beta & 1+2\beta & -\beta & & \\ & . & . & . & \\ & & -\beta & 1+2\beta & -\beta \\ & & & -\beta & 1+2\beta \end{pmatrix} \begin{pmatrix} u_1^{n+1} \\ u_2^{n+1} \\ . \\ u_{J-1}^{n+1} \\ u_J^{n+1} \end{pmatrix} = \begin{pmatrix} u_1^n + \beta u_0 \\ u_2^n \\ . \\ u_{J-1}^n \\ u_J^n + \beta u_{J+1} \end{pmatrix}$$

Features: 1st-order accurate in time, 2nd-order in space, unconditionaly stable (i.e., for any dt)
Each time step requires direct solution to a linear algebraic system with tridiagonal matrix of size J x J.

Crank-Nicolson scheme (CN)
implicit formula with an average of FTCS and BTCS schemes on the right-hand side

$$u_j^{n+1} = u_j^n + \frac{\beta}{2}\left(\delta_x^2 u_j^n + \delta_x^2 u_j^{n+1}\right)$$

$$\begin{pmatrix} 1+\beta & -\beta/2 & & \\ -\beta/2 & 1+\beta & -\beta/2 & \\ & . & . & . \\ & & -\beta/2 & 1+\beta \end{pmatrix} \begin{pmatrix} u_1^{n+1} \\ u_2^{n+1} \\ . \\ u_J^{n+1} \end{pmatrix} = \begin{pmatrix} 1-\beta & \beta/2 & & \\ \beta/2 & 1-\beta & \beta/2 & \\ & . & . & . \\ & & \beta/2 & 1-\beta \end{pmatrix} \begin{pmatrix} u_1^n \\ u_2^n \\ . \\ u_J^n \end{pmatrix} + \begin{pmatrix} \beta u_0 \\ 0 \\ . \\ \beta u_{J+1} \end{pmatrix}$$

Features: 2nd-order accurate in both time and space, unconditionally stable
Each time step requires direct solution to a linear algebraic system with tridiagonal matrix of size J x J.

Heat equation in 2D: FTCS, BTCS and CN schemes

Difference operators $\quad \delta_x^2 u_{jk}^n \equiv u_{j-1,k}^n - 2u_{jk}^n + u_{j+1,k}^n, \quad \delta_y^2 u_{jk}^n \equiv u_{j,k-1}^n - 2u_{jk}^n + u_{j,k+1}^n$

FTCS scheme $\quad u_{jk}^{n+1} = u_{jk}^n + \beta\left(\delta_x^2 u_{jk}^n + \delta_y^2 u_{jk}^n\right)$

BTCS scheme $\quad u_{jk}^{n+1} = u_{jk}^n + \beta\left(\delta_x^2 u_{jk}^{n+1} + \delta_y^2 u_{jk}^{n+1}\right)$

CN scheme $\quad u_{jk}^{n+1} = u_{jk}^n + \frac{\beta}{2}\left(\delta_x^2 u_{jk}^n + \delta_y^2 u_{jk}^n + \delta_x^2 u_{jk}^{n+1} + \delta_y^2 u_{jk}^{n+1}\right)$

For implicit BTCS and CN schemes, the matrix is J^2 x J^2, sparse and band diagonal (tridiagonal with fringes).
Direct solution is possible with special methods.

Heat equation in more dimensions: alternating-direction implicit (ADI) method

2D: splitting the time step into 2 substeps, each of lenght t/2

$$\begin{aligned} u_{jk}^{n+1/2} &= u_{jk}^n + \frac{\beta}{2}\left(\delta_x^2 u_{jk}^{n+1/2} + \delta_y^2 u_{jk}^n\right) \\ u_{jk}^{n+1} &= u_{jk}^{n+1/2} + \frac{\beta}{2}\left(\delta_x^2 u_{jk}^{n+1/2} + \delta_y^2 u_{jk}^{n+1}\right) \end{aligned}$$

3D: splitting the time step into 3 substeps, each of length t/3

$$\begin{aligned} u_{jkl}^{n+1/3} &= u_{jkl}^n + \frac{\beta}{3}\left(\delta_x^2 u_{jkl}^{n+1/3} + \delta_y^2 u_{jkl}^n + \delta_z^2 u_{jkl}^n\right) \\ u_{jkl}^{n+2/3} &= u_{jkl}^{n+1/3} + \frac{\beta}{3}\left(\delta_x^2 u_{jkl}^{n+1/3} + \delta_y^2 u_{jkl}^{n+2/3} + \delta_z^2 u_{jkl}^{n+1/3}\right) \\ u_{jkl}^{n+1} &= u_{jkl}^{n+2/3} + \frac{\beta}{3}\left(\delta_x^2 u_{jkl}^{n+2/3} + \delta_y^2 u_{jkl}^{n+2/3} + \delta_z^2 u_{jkl}^{n+1}\right) \end{aligned}$$

All substeps are implicit and each requires direct solutions to J independent linear algebraic systems with tridiagonal matrices of size J x J.

Example: ADI method for heat equation in 2D and 3D

Wave equation

a quantity travelling over the domain

a partial differential equation (2nd-order in time t, 2nd-order in spatial variables X) for a function u(t, X)

1D (one-dimensional) case: X = x, 2D case: X = x,y, 3D case: X = x,y,z

General form: $\partial_t^2 u(t,X) = c^2 \Delta u(t,X)$

in 3D: $\partial_t^2 u(t,x,y,z) = c^2(\partial_x^2 + \partial_y^2 + \partial_z^2) u(t,x,y,z)$

Initial conditions: $u(t_0,X) = u_0(X), \quad \partial_t u(t_0,X) = v_0(X)$

Boundary conditions: $u(t,X_B) = u_B(t,X_B)$ on the boundary

i.e., the initial value problem (IVP) for the hyperbolic partial differential equation

Discretized wave equation in 1D

1D wave equation

$$\partial_t^2 u(t,x) = c^2 \partial_x^2 u(t,x)$$

can be rewritten into the form of two equations of the 1st-order in time

$$\begin{aligned} (\partial_t + c\partial_x) u(t,x) &= v(t,x) \\ (\partial_t - c\partial_x) v(t,x) &= 0 \end{aligned}$$

Discretization grids

$$\begin{aligned} t_n &= t_0 + n\,dt, & dt &= (t_N - t_0)/N \\ x_j &= x_0 + j\,dx, & dx &= (x_J - x_0)/J \\ u_j^n &\approx u(t_n, x_j), & v_j^n &\approx v(t_n, x_j) \end{aligned}$$

Explicit FTBS scheme (forward-in-time, backward-in-space)

FD1 for time: $\partial_t u_j^n \approx (u_j^{n+1} - u_j^n)/dt$

FD1 for space: $\partial_x u_j^n \approx (-u_{j-1}^n + u_j^n)/dx$

Features: low accuracy, stability for $\frac{c\,dt}{dx} \le 1$ (Courant-Friedrichs-Lewy condition)

PDEs in the matrix form:

$$\partial_t \begin{pmatrix} u \\ v \end{pmatrix} = \begin{pmatrix} -c\partial_x & 1 \\ 0 & c\partial_x \end{pmatrix} \begin{pmatrix} u \\ v \end{pmatrix}$$

Discretized equations:

$$\begin{pmatrix} u \\ v \end{pmatrix}^{n+1} = \begin{pmatrix} I - \gamma A & \delta I \\ 0 & I + \gamma A \end{pmatrix} \begin{pmatrix} u \\ v \end{pmatrix}^n, \quad \gamma = \frac{c\,dt}{dx},\ \delta = dt,\ A = \begin{pmatrix} 1 & 0 & 0 & \cdot \\ -1 & 1 & 0 & \cdot \\ 0 & -1 & 1 & \cdot \end{pmatrix}$$

and I is the identical matrix

Explicit FTCS scheme (forward-in-time, centered-in-space)

FD1 for time: $\partial_t u_j^n \approx (u_j^{n+1} - u_j^n)/dt$

FD2 for space: $\partial_x u_j^n \approx (-u_{j-1}^n + u_{j+1}^n)/(2dx)$

Features: unstable for any dt, i.e., FTCS scheme inappropriate for the wave equation

Implicit Crank-Nicolson scheme

implicit formula with an average of FTBS and BTBS schemes on the right-hand side

$$\begin{pmatrix} u \\ v \end{pmatrix}^{n+1} = \begin{pmatrix} u \\ v \end{pmatrix}^{n} + \begin{pmatrix} -\gamma A & \delta I \\ 0 & \gamma A \end{pmatrix} \left[\begin{pmatrix} u \\ v \end{pmatrix}^{n} + \begin{pmatrix} u \\ v \end{pmatrix}^{n+1} \right], \quad \gamma = \frac{c\,dt}{dx},\ \delta = dt,\ A = \begin{pmatrix} 1 & 0 & 0 & \cdot \\ -1 & 1 & 0 & \cdot \\ 0 & -1 & 1 & \cdot \end{pmatrix}$$

Features: higher accuracy, unconditional stability (i.e., for any dt)

Example: travelling waves

domain $t \geq t_0 = 0,\ x \geq x_0 = 0$

initial condition $u(0,x) = u_0(x), \quad v(0,x) = 0$

boundary condition $u(t,0) = 0$

analytical solution $u(t,x) = u_0(t - cx)$

Links and references

PDEs

Koev P., Numerical Methods for Partial Differential Equations, 2005
http://dspace.mit.edu/bitstream/handle/1721.1/56567/18-336Spring-2005/OcwWeb/Mathematics/18-336Spring-2005/CourseHome/index.htm

Lehtinen J., Time-domain numerical solution of the wave equation, 2003
http://www.cs.unm.edu/~williams/cs530/wave_eqn.pdf

Piché R., Partial Differential Equations, 2010
http://math.tut.fi/~piche/pde/index.html

Press W. H. et al., Numerical Recipes in Fortran 77: The Art of Scientific Computing, Second Edition, Cambridge, 1992
Chapter 19.2: Diffusive initial value problems
Chapter 19.3: Initial value problems in multidimensions
Chapter 19.6: Multigrid methods for boundary value problems
http://www.nr.com, PDFs available at http://www.nrbook.com/a/bookfpdf.php

Spiegelman M., Myths and Methods in Modelling, 2000
http://www.ldeo.columbia.edu/~mspieg/mmm/

Wave equation on GPU

Michéa D. and Komatitsch D., Accelerating a three-dimensional finite-difference wave propagation code using GPU graphics cards, 2010