CIP Scheme

Higher-order interpolation for the upwind region $x_{j-1} \le x \le x_j$

Characteristics: f and f_x are the dependent variables given independently for the interpolation

→ Hermite Interpolation

$$F_j(X) = a_j X^3 + b_j X^2 + f_{x,j} X + f_j$$

where $X = x - x_j$

4 Matching Conditions: $f_j = F(x_j)$, $f_{x,j} = F_x(x_j)$

$f_{j-1} = F(x_{j-1})$, $f_{x,j-1} = F_x(x_{j-1})$

CIP Scheme

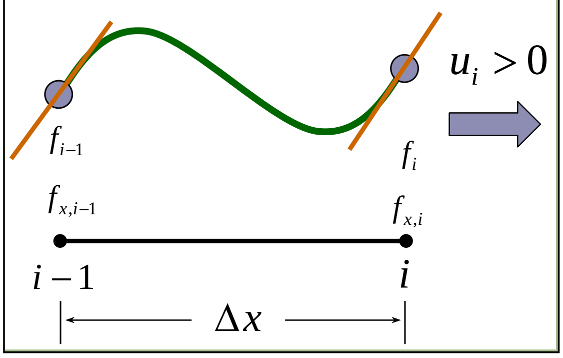
2 Matching Conditions:

$f_{j-1} = F(-\Delta x)$, $f_{x,j-1} = F_x(-\Delta x)$

$$a_i = \frac{f_i + f_{i-1}}{\Delta x^2} - 2\frac{f_i - f_{i-1}}{\Delta x^3}$$

$$b_i = \frac{2f_i + f_{i-1}}{\Delta x} - 3\frac{f_i - f_{i-1}}{\Delta x^2}$$

$F(x) = ax^3 + bx^2 + f_x x + f$



At $t = t^{n+1}$

$$f_j^{n+1} = F_j^n(-u\Delta t) = a_j(-u\Delta t)^3 + b_j(-u\Delta t)^2 + f_{x,j}^n(-u\Delta t) + f_j^n$$

$$f_{x,j}^{n+1} = \frac{\partial}{\partial x}F_j^n(-u\Delta t) = 3a_j(-u\Delta t)^2 + 2b_j(-u\Delta t) + f_{x,j}^n$$

Source Code

Implicit Scheme

The explicit scheme includes only one unknown f^{n+1} in the discretized form, and easy to solve.

The implicit schemes are derived for t^{n+1} , and includes and more than one unknown value in the discretized form. Normally it becomes coupled equation.

$$
\begin{aligned}
f_j^{n+1} &= f_j^n - u \frac{f_j^{n+1} - f_{j-1}^{n+1}}{\Delta x} \Delta t \\
&= f_j^n - C\left(f_j^{n+1} - f_{j-1}^{n+1}\right)
\end{aligned}
$$

Source Code

Crank-Nicolson Scheme

When we can use f^{n+1} in the right-hand side of the discretized form, we improve the time accuracy to $O(\Delta t^2)$

We change the time of the right-hand side from t^n to $t^{n+1/2}$

$$
\frac{\phi_j^{n+1} - \phi_j^n}{\Delta t} + u \frac{1}{2}\left(\frac{\phi_{j+1}^{n+1} - \phi_{j-1}^{n+1}}{2\Delta x} + \frac{\phi_{j+1}^{n} - \phi_{j-1}^{n}}{2\Delta x} \right) = 0
$$

von Neumann's stability analysis:

$$
\delta\phi_j^{n+1} / \delta\phi_j^n = \frac{1 - \frac{1}{2} iC \sin k\Delta x}{1 + \frac{1}{2} iC \sin k\Delta x} \quad \rightarrow \quad \left| \delta\phi_j^{n+1} / \delta\phi_j^n \right| = 1
$$

Generalized
Crank-Nicolson Scheme

Weight average between ***n-th*** and ***n+1-th*** value

$$\frac{\phi_j^{n+1}-\phi_j^n}{\Delta t}+u\left(\lambda\frac{\phi_{j+1}^{n+1}-\phi_{j-1}^{n+1}}{2\Delta x}+(1-\lambda)\frac{\phi_{j+1}^{n}-\phi_{j-1}^{n}}{2\Delta x}\right)=0$$

$\lambda=0$: FTCS Scheme (perfect explicit)

$\lambda=1/2$: Crank–Nicolson Scheme

$\lambda=1$: perfect implicit

Generalized
Crank-Nicolson Scheme

von Neumann's stability analysis:

$$\left|\delta\phi_j^{n+1}/\delta\phi_j^n\right|=\sqrt{\frac{1+(1-\lambda)^2C^2\sin^2 k\Delta x}{1+\lambda^2C^2\sin^2 k\Delta x}}$$

$\lambda > 1/2$: stable
$\lambda < 1/2$: unstable

Implicit Upwind Scheme

When we solve the advection equation implicitly,
if we calculate it from the upwind region to the down-stream region, we do not have to solve the matrix.

$$\phi_j^{n+1} = \phi_j^n - u\frac{\phi_j^{n+1} - \phi_{j-1}^{n+1}}{\Delta x}\Delta t$$
$$= \phi_j^n - C\left(\phi_j^{n+1} - \phi_{j-1}^{n+1}\right)$$

Implicit Upwind Scheme

When we calculate the advection equation on the grid ***j***, ϕ_{j-1}^{n+1} has been calculated. The unknown variable is only ϕ_j^{n+1} .

$$\phi_j^{n+1} = \frac{\phi_j^n + C\phi_{j-1}^{n+1}}{1+C}$$

von Neumann's stability analysis:

$$\left|\delta\phi_j^{n+1} / \delta\phi_j^n\right| = \frac{1}{\sqrt{\left(1 + C - C\cos k\Delta x\right)^2 + C^2\sin^2 k\Delta x}} < 1$$

For Diffusion Equation

Implicit Scheme

$$\frac{\partial \phi}{\partial t} = \kappa \frac{\partial^2 \phi}{\partial x^2} \Rightarrow \frac{\phi_j^{n+1} - \phi_j^n}{\Delta t} = \kappa \frac{\phi_{j+1}^{n+1} - 2\phi_j^{n+1} + \phi_{j-1}^{n+1}}{\Delta x^2}$$

$$\phi_j^{n+1} = \phi_j^n + \mu\left(\phi_{j+1}^{n+1} - 2\phi_j^{n+1} + \phi_{j-1}^{n+1}\right)$$

where $\mu = \frac{\kappa \Delta t}{\Delta x^2}$

von Neumann's stability analysis:

$$\delta\phi_j^{n+1} / \delta\phi_j^n = \frac{1}{1 + 2\mu(1 - \cos k\Delta x)} < 1$$

Features of Implicit Scheme

The ***n***-step values can affect ***n+1***-step values of all the grid points. The traveling velocity of the numerical information is infinity.

- No CFL restriction
- Stabilization for the scheme
- CPU time consuming

Elliptic Equation

Typical equation: Poisson Equation $$\Delta\phi = \rho$$

$$\frac{d^2\phi}{dx^2} = \rho \qquad \frac{\partial^2\phi}{\partial x^2} + \frac{\partial^2\phi}{\partial y^2} = \rho \qquad \frac{\partial^2\phi}{\partial x^2} + \frac{\partial^2\phi}{\partial y^2} + \frac{\partial^2\phi}{\partial z^2} = \rho$$

Very important equation in science and engineering

- Maxwell's Equation for Scalar Potential
- Incompressible Navier-Stokes Equation
- Steady State of Diffusion (Thermal Conduction) Equation

Poisson Equation

1-dimensional case:

$$\frac{\phi_{j+1} - 2\phi_j + \phi_{j-1}}{\Delta x^2} = \rho_j$$

2-dimensional case:

$$\frac{\phi_{i+1,j} - 2\phi_{i,j} + \phi_{i-1,j}}{\Delta x^2} + \frac{\phi_{i,j+1} - 2\phi_{i,j} + \phi_{i,j-1}}{\Delta y^2} = \rho_{i,j}$$

Poisson Equation

N x N linear coupling equations

$$
\begin{aligned}
\phi_1 \qquad\qquad\qquad\qquad\qquad\qquad &= 0 \\
\phi_1 - 2\phi_2 + \phi_3 &= \rho_2 \Delta x^2 \\
\phi_2 - 2\phi_3 + \phi_4 &= \rho_3 \Delta x^2 \\
\phi_3 - 2\phi_4 + \phi_5 &= \rho_4 \Delta x^2 \\
\cdots &= \cdots \\
\phi_{j-1} - 2\phi_j + \phi_{j+1} &= \rho_j \Delta x^2 \\
\cdots &= \cdots \\
\phi_{N-2} - 2\phi_{N-1} + \phi_N &= \rho_{N-1} \Delta x^2 \\
\phi_N &= 0
\end{aligned}
$$

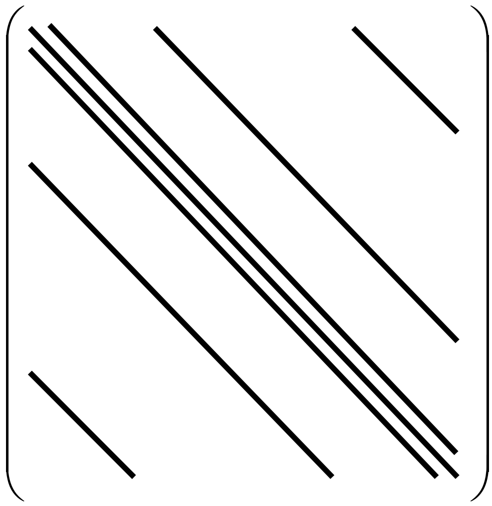
Poisson Equation

$$
\begin{pmatrix}
1 & & & & & & & \\
1 & -2 & 1 & & & & & \\
 & \cdot & \cdot & \cdot & & & & \\
 & & 1 & -2 & 1 & & & \\
 & & & \cdot & \cdot & \cdot & & \\
 & & & & & 1 & -2 & 1 \\
 & & & & & & & 1
\end{pmatrix}
\begin{pmatrix}
\phi_1 \\ \phi_2 \\ \\ \phi_j \\ \\ \phi_{N-1} \\ \phi_N
\end{pmatrix}
= \Delta x^2
\begin{pmatrix}
\rho_1 \\ \rho_2 \\ \\ \rho_j \\ \\ \rho_{N-1} \\ \rho_N
\end{pmatrix}
$$

Gauss Elimination Method is not available, because the matrix size is too large.

Sparse Matrix:

3-D Poisson Equation



$$\begin{pmatrix} \phi_{1,1,1} \\ \phi_{2,1,1} \\ \\ \phi_{i,j,k} \\ \\ \phi_{N-1,N,N} \\ \phi_{N,N,N} \end{pmatrix} = \Delta x^2 \begin{pmatrix} \rho_{1,1,1} \\ \rho_{2,1,1} \\ \\ \rho_{i,j,k} \\ \\ \rho_{N-1,N,N} \\ \rho_{N,N,N} \end{pmatrix}$$

For example, N_x x N_y x N_z = 100x100x100
Double precision 8 byte x $(1000000)^2$ = 8×10^{12} = 8 TB

c.f. TSUBAME GPU : 12TB

Matrix Solver

Relaxation Method:

- Point Jacobi
- Gauss-Seidel
- SOR
- ICCG (Incomplete Conjugate Gradient)
- ILUCR (Incomplete LU Conjugate Residual)
- BiCGStab (Bi-conjugate Conjugate Gradient Stabilize)

■ Advantage : Memory and CPU time
■ Disadvantage : No guarantee to be solved
limited types of matrix

Point Jacobi Method

Expecting iterative convergence:

$$\frac{\phi_{j+1}^{n} - 2\phi_{j}^{n+1} + \phi_{j-1}^{n}}{\Delta x^2} = \rho_j$$

The n+1-th value is calculated by the n-th value.

$$\phi_j^{n+1} = \frac{1}{2}\left(\phi_{j+1}^{n} + \phi_{j-1}^{n} - \Delta x^2 \rho_j\right)$$

If $\phi_j^{n+1} \approx \phi_j^{n}$, ϕ_j^{n} is the solution of Poisson equation.

Point Jacobi Method

Starting from the initial value ϕ_j^0,

$$\phi_j^1 = \frac{1}{2}\left(\phi_{j+1}^{0} + \phi_{j-1}^{0} - \Delta x^2 \rho_j\right)$$

Iterative calculations give ϕ_j^1, ϕ_j^2, ϕ_j^3, . . . ϕ_j^n, ϕ_j^{n+1}

If $\left|\phi_j^{n+1} \approx \phi_j^{n}\right| < \varepsilon$ is satidfied, the iteration has been converged.

Point Jacobi Method

von Neumann's stability analysis
for the iterative process:

Assuming the perturbation $\phi_j^n = \delta\phi^n \, e^{ik\cdot j\Delta x}$

$$\delta\phi^{n+1} / \delta\phi^n = \frac{1}{2}\left(e^{ik\Delta x} + e^{-ik\Delta x}\right) = \cos k\Delta x$$

The iteration process is stable, but slow.

Actual stability depends on the source term.

Point Jacobi Method

Introduction of the relaxation factor : ω

$$\phi_j^{n+1} = (1-\omega)\phi_j^n + \omega\frac{1}{2}\left(\phi_{j+1}^n + \phi_{j-1}^n - \Delta x^2 \rho_j\right)$$

Stability analysis $\delta\phi^{n+1} / \delta\phi^n = (1-\omega) + \omega\cos k\Delta x \; < \; 1$

$$(0 \le \omega \le 1)$$

Small $\delta\phi^{n+1} / \delta\phi^n$ means rapid decrease of the error.

$\omega = 1$ is found to be the fastest convergence of Jacobi Iteration method.

SOR Method

Any faster convergence technique ?

When we calculate ϕ_j^{n+1} , if ϕ_{j-1}^{n+1} has been calculated, it is better convergence to use ϕ_{j-1}^{n+1} .

$$\phi_j^{n+1} = \frac{1}{2}\left(\phi_{j+1}^{n} + \phi_{j-1}^{n+1} - \Delta x^2 \rho_j\right)$$

This iteration method is called “Gauss-Seidel” method.

SOR Method

Introduction of the relaxation factor : ω

$$\phi_j^{n+1} = (1-\omega)\phi_j^{n} + \omega\frac{1}{2}\left(\phi_{j+1}^{n} + \phi_{j-1}^{n+1} - \Delta x^2 \rho_j\right)$$

The stability analysis shows that the iteration process is stable for $0 < \omega < 2$.

Acceleration of the iteration : $1 < \omega < 2$.

SOR (Successive Over-relaxation) Method

The fastest convergence is achieved for $\omega \approx 1.82$.