Computer-Assisted Problem-Solving / Numerical Methods

Differential Equations and $A\vec{u} = \vec{f}$

version: January 31st, 2017

Legend: Method, Theory, Example, Advanced, Appendix

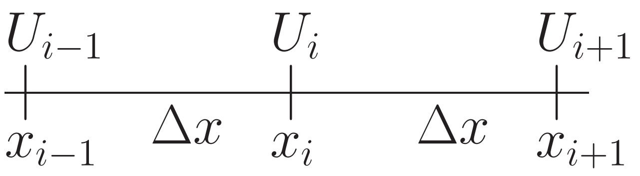
Method

Finite Difference method

Computational mesh on $[a\ b]$:

points $x_i = a + (i-1) * \Delta x$, $\quad i = 1 \cdots N+1$

with mesh size $\Delta x = (b-a)/N$



Taylor series around x_i:

$$U(x_i+\Delta x) = U(x_i) + U'(x_i)\frac{\Delta x}{1} + U''(x_i)\frac{(\Delta x)^2}{2} + U'''(x_i)\frac{(\Delta x)^3}{6} + \text{h.o.t.} \cdots$$

$$U(x_i-\Delta x) = U(x_i) - U'(x_i)\frac{\Delta x}{1} + U''(x_i)\frac{(\Delta x)^2}{2} - U'''(x_i)\frac{(\Delta x)^3}{6} + \text{h.o.t.} \cdots$$

In terms of values U_{i-1}, U_i, U_{i+1} at x_{i-1}, x_i, x_{i+1}

$$U_{i+1} = U_i + U_i'\frac{\Delta x}{1} + U_i''\frac{(\Delta x)^2}{2} + U_i'''\frac{(\Delta x)^3}{6} + \text{h.o.t.} \cdots$$

$$U_{i-1} = U_i - U_i'\frac{\Delta x}{1} + U_i''\frac{(\Delta x)^2}{2} - U_i'''\frac{(\Delta x)^3}{6} + \text{h.o.t.} \cdots$$

Taylor series $\Longrightarrow$ expressions for $U'(x)$:

$$U'(x_i) \;=\; \frac{U_{i+1} - U_i}{\Delta x} - U_i''\frac{(\Delta x)}{2} - U_i'''\frac{(\Delta x)^2}{6} + \text{h.o.t.}\cdots$$

$$U'(x_i) \;=\; \frac{U_i - U_{i-1}}{\Delta x} + U_i''\frac{(\Delta x)}{2} - U_i'''\frac{(\Delta x)^2}{6} + \text{h.o.t.}\cdots$$

Accuracy 1st order: halving $\Delta x \Longrightarrow$ error/2

Expression for $U''(x)$ (combine the series):

$$U''(x_i) = \frac{U_{i+1} - 2U_i + U_{i-1}}{(\Delta x)^2} + \mathcal{O}(\Delta x^2)$$

Accuracy 2nd order: halving $\Delta x \Longrightarrow$ error/4

By means of expressions for $U'(x)$ and $U''(x)$:
ODE $\Longrightarrow$ matrix-vector system

Example

Example Linear System

Continuous 2nd order ODE on $[a\ b]$

$$U''(x) + x^2\, U(x) = \sin(x)$$

Boundary conditions: **left** $U(a) = \tilde{U}_a$
right $U(b) = \tilde{U}_b$

Computational mesh on $[a\ b]$**:**

points $x_i = a + (i-1) * \Delta x$**,** $\quad i = 1 \cdots N+1$

with mesh size $\Delta x = (b-a)/N$

Boundary points:

$$\begin{aligned} i = 1: \quad U_1 &= \tilde{U}_a \\ i = N+1: \; U_{N+1} &= \tilde{U}_b \end{aligned}$$

ODE in internal points x_i **(**$i = 2, \ldots, N$**):**

$$U''(x_i) + x_i^2\, U_i = \sin(x_i)$$

$$\frac{U_{i+1} - 2U_i + U_{i-1}}{(\Delta x)^2} + x_i^2\, U_i = \sin(x_i)$$

$$\frac{1}{(\Delta x)^2} U_{i-1} + \left(-\frac{2}{(\Delta x)^2} + x_i^2\right) U_i + \frac{1}{(\Delta x)^2} U_{i+1} = \sin(x_i)$$

Matrix-Vector system:

$$\begin{pmatrix} 1 & & & & & & & & \\ L & D & R & & & & & & \\ & L & D & R & & & & & \\ & & \ddots & \ddots & \ddots & & & & \\ & & & L & D & R & & & \\ & & & & \ddots & \ddots & \ddots & & \\ & & & & & L & D & R & \\ & & & & & & L & D & R \\ & & & & & & & & 1 \end{pmatrix} \begin{pmatrix} U_1 \\ U_2 \\ U_3 \\ \vdots \\ U_i \\ \vdots \\ U_{N-1} \\ U_N \\ U_{N+1} \end{pmatrix} = \begin{pmatrix} \tilde{U}_a \\ \sin(x_2) \\ \sin(x_3) \\ \vdots \\ \sin(x_i) \\ \vdots \\ \sin(x_{N-1}) \\ \sin(x_N) \\ \tilde{U}_b \end{pmatrix}$$

with:

$$\begin{aligned} L &= \frac{1}{(\Delta x)^2} \\ D &= \frac{-2}{(\Delta x)^2} + x_i^2 \\ R &= \frac{1}{(\Delta x)^2} \end{aligned}$$

Solve $A\vec{u} = \vec{f} \Longrightarrow$

U_i **known in points** $x_i \quad (i = 1 \cdots N+1)$

Method

Solving Tridiagonal Matrices

Tridiagonal system $A\vec{u} = \vec{f}$

$A = a_{ij}$ **tridiagonal if** $a_{ij} = 0$ **for** $|i - j| > 1$

$$A = \begin{pmatrix} a_1 & c_1 & 0 & & 0 \\ b_2 & a_2 & c_2 & & 0 \\ 0 & b_3 & a_3 & c_3 & 0 \\ \vdots & & \ddots & \ddots & \vdots \\ 0 & & & b_n & a_n \end{pmatrix}$$

LU factorisation with such a matrix:

$$A = LU = \begin{pmatrix} \alpha_1 & 0 & 0 & & 0 \\ b_2 & \alpha_2 & 0 & & 0 \\ 0 & b_3 & \alpha_3 & & 0 \\ \vdots & & \ddots & \ddots & \vdots \\ 0 & & & b_n & \alpha_n \end{pmatrix} \begin{pmatrix} 1 & \gamma_1 & 0 & & 0 \\ 0 & 1 & \gamma_2 & & 0 \\ \vdots & & \ddots & \ddots & \vdots \\ \vdots & & & 1 & \gamma_{n-1} \\ 0 & & & 0 & 1 \end{pmatrix}$$

Thus

$$a_1 = \alpha_1 \quad \alpha_1 \gamma_1 = c_1$$

$$a_i = \alpha_i + b_i \gamma_{i-1} \quad i = 2 \cdots n$$

$$\alpha_i \gamma_i = c_i \quad i = 2 \cdots n - 1$$

Hence

$$\alpha_1 = a_1 \quad \gamma_1 = c_1 / \alpha_1$$

$$\alpha_i = a_i - b_i \gamma_{i-1} \quad \gamma_i = c_i / \alpha_i \quad i = 2 \cdots n - 1$$

$$\alpha_n = a_n - b_n \gamma_{n-1}$$

LU factorisation is now available

Then solve $L\vec{v} = \vec{f}$ **and** $U\vec{u} = \vec{v} \implies A\vec{u} = \vec{f}$

Example:

$$\begin{pmatrix} 2 & 1 & 0 & 0 \\ 1 & 4 & 1 & 0 \\ 0 & 1 & 4 & 1 \\ 0 & 0 & 1 & 2 \end{pmatrix} = \begin{pmatrix} 2 & 0 & 0 & 0 \\ 1 & \frac{7}{2} & 0 & 0 \\ 0 & 1 & \frac{26}{7} & 0 \\ 0 & 0 & 1 & \frac{45}{26} \end{pmatrix} \begin{pmatrix} 1 & \frac{1}{2} & 0 & 0 \\ 0 & 1 & \frac{2}{7} & 0 \\ 0 & 0 & 1 & \frac{7}{26} \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Operations for LU:
$n-1$ **divisions,** $n-1$ **multipl.,** $n-1$ **subtract.**
$\Longrightarrow$ **total for LU** $\sim \mathcal{O}(3n)$

Solve $L\vec{v}=\vec{f}$**:**

$$\begin{aligned} v_1 &= f_1/\alpha_1 \\ v_i &= (f_i - b_i v_{i-1})/\alpha_i \quad i = 2\cdots n \end{aligned}$$

Solve $U\vec{u}=\vec{v}$**:**

$$\begin{aligned} u_n &= v_n \\ u_i &= v_i - \gamma_i u_{i+1} \quad i = n-1\cdots 1 \end{aligned}$$

Operations for solution:
n **divisions,** $2(n-1)$ **multipl.,** $2(n-1)$ **subtract.**
$\Longrightarrow$ **total for solving** $\sim \mathcal{O}(5n)$

Solution costs similar to LU **construction:**
both $\sim \mathcal{O}(n)$

General band matrices:
p **lower and** q **upper bands (with** $p, q \ll n$**)**
LU $\sim \mathcal{O}(2npq)$ **flops**
Solving $\sim \mathcal{O}(2np + 2nq)$

Modern PC ideally can do 100G flops, super computers can handle 100P flops

Example with 10^{+9} operations/second

Tri-Diagonal Matrix Algorithm (TDMA)

task	$n = 1,000$	$n = 10,000$	$n = 100,000$
LU	**3 μsec**	**30 μsec**	**300 μsec**
solve	**5 μsec**	**50 μsec**	**500 μsec**

Complete LU + solution (Matlab: `u = A\f;`)

task	$n = 1,000$	$n = 10,000$	$n = 100,000$
row reduction	**0.7 sec**	**11 min**	**185 hours**
solve	**0.001 sec**	**0.1 sec**	**10 sec**

Use of `u=inv(A)*f;` takes even more time.

Therefore: use matrix structure $\Longrightarrow$ large reduction in computer time!

Efficient storage of band matrix A:

$$A_{i,j} = \begin{pmatrix} a_1 & c_1 & 0 & & 0 \\ b_2 & a_2 & c_2 & & 0 \\ 0 & b_3 & a_3 & c_3 & 0 \\ \vdots & & \ddots & \ddots & \vdots \\ 0 & & & b_n & a_n \end{pmatrix} \qquad \longrightarrow \qquad A(k,l) = \begin{pmatrix} 0 & a_1 & c_1 \\ b_2 & a_2 & c_2 \\ b_3 & a_3 & c_3 \\ \vdots & \vdots & \vdots \\ b_n & a_n & 0 \end{pmatrix}$$

$i, j = 1 \ldots n$ $\qquad$ $k = 1 \ldots n$, $l = 1, 2, 3$