MSDM5004
Numerical Methods and Modeling in Science
Spring 2024

Lecture 5

Prof Yang Xiang
Hong Kong University of Science and Technology

Tridiagonal linear system: Thomas algorithm

$$
\begin{bmatrix}
d_1 & c_1 & & & & & \\
a_2 & d_2 & c_2 & & & & \\
 & a_3 & d_3 & c_3 & & & \\
 & & a_4 & d_4 & c_4 & & \\
 & & & \ddots & \ddots & \ddots & \\
 & & & & a_{n-1} & d_{n-1} & c_{n-1} \\
 & & & & & a_n & d_n
\end{bmatrix}
\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ \vdots \\ x_{n-1} \\ x_n \end{bmatrix}
=
\begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ b_4 \\ \vdots \\ b_{n-1} \\ b_n \end{bmatrix}
$$

A sufficient condition for finding the solution by Gaussian elimination:

Diagonally dominant

$|d_{ij}| > |a_{ij}| + |c_{ij}|$ in the case of the tridiagonal matrix.

Forward elimination

$$d_1' = d_1$$

$$d_2' = d_2 - \frac{a_2}{d_1'}c_1$$

$$\dots$$

$$d_i' = d_i - \frac{a_i}{d_{i-1}'}c_{i-1}$$

Accordingly

$$b_1' = b_1$$

$$b_2' = b_2 - \frac{a_2}{d_1'}b_1'$$

$$b_i' = b_i - \frac{a_i}{d_{i-1}'}b_{i-1}'$$

$$\begin{bmatrix} d_1 & c_1 & & & & & & \\ a_2 & d_2 & c_2 & & & & & \\ & a_3 & d_3 & c_3 & & & & \\ & & a_4 & d_4 & c_4 & & & \\ & & & \ddots & \ddots & \ddots & & \\ & & & & & a_{n-1} & d_{n-1} & c_{n-1} \\ & & & & & & a_n & d_n \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ \vdots \\ x_{n-1} \\ x_n \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ b_4 \\ \vdots \\ b_{n-1} \\ b_n \end{bmatrix}$$

$$\begin{bmatrix} d_1' & c_1 & & & & & \\ & d_2' & c_2 & & & & \\ & & d_3' & c_3 & & & \\ & & & d_4' & c_4 & & \\ & & & & \ddots & \ddots & \\ & & & & & d_{n-1}' & c_{n-1} \\ & & & & & & d_n' \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ \vdots \\ x_{n-1} \\ x_n \end{bmatrix} = \begin{bmatrix} b_1' \\ b_2' \\ b_3' \\ b_4' \\ \vdots \\ b_{n-1}' \\ b_n' \end{bmatrix}$$

Backward substitution

$$x_n = \frac{b_n'}{d_n'}$$

$$x_{n-1} = \frac{b_{n-1}' + c_{n-1}x_n}{d_{n-1}'}$$

$$\dots$$

$$x_i = \frac{b_i' + c_i x_{i+1}}{d_i'}$$

The Thomas Algorithm

1. Forward elimination

$$d_i = d_i - \frac{a_i}{d_{i-1}} c_{i-1},$$

$$b_i = b_i - \frac{a_i}{d_{i-1}} b_{i-1},$$

for $i = 2, 3, \cdots, N$.

2. Backward substitution

$$x_n = \frac{b_n}{d_n}$$

$$x_i = \frac{b_i + c_i x_{i+1}}{d_i}, \quad i = n-1, n-2, \cdots, 2, 1.$$

Chapter 7

Iterative Methods for Solving Linear Systems

1. Iterative methods

Solve linear system $A\mathbf{x} = \mathbf{b}$

by iterations

$$\mathbf{x}^{(k)} = T\mathbf{x}^{(k-1)} + \mathbf{c},$$

Advantages

- Iterative techniques are seldom used for solving linear systems of small dimension since the time required for sufficient accuracy exceeds that required for direct techniques such as Gaussian elimination.
- For large systems with a high percentage of 0 entries, however, these techniques are efficient in terms of both computer storage and computation.

sparse matrices

2. Jacobi method

For each $k \geq 1$, generate the components $x_i^{(k)}$ of $\mathbf{x}^{(k)}$ from the components of $\mathbf{x}^{(k-1)}$ by

$$x_i^{(k)} = \frac{1}{a_{ii}} \left[\sum_{\substack{j=1 \\ j \neq i}}^{n} \left(-a_{ij} x_j^{(k-1)} \right) + b_i \right], \qquad \text{for } i = 1, 2, \ldots, n.$$

Example

The linear system $A\mathbf{x} = \mathbf{b}$ given by

$$
\begin{aligned}
E_1 &: \quad 10x_1 - x_2 + 2x_3 = 6, \\
E_2 &: \quad -x_1 + 11x_2 - x_3 + 3x_4 = 25, \\
E_3 &: \quad 2x_1 - x_2 + 10x_3 - x_4 = -11, \\
E_4 &: \quad 3x_2 - x_3 + 8x_4 = 15
\end{aligned}
$$

Use Jacobi's iterative technique to find approximations $\mathbf{x}^{(k)}$ to $\mathbf{x}$ starting with $\mathbf{x}^{(0)} = (0, 0, 0, 0)^t$

Solution

We first solve equation E_i for x_i, for each $i = 1, 2, 3, 4$, to obtain

$$\begin{aligned}
x_1 &= \frac{1}{10}x_2 - \frac{1}{5}x_3 + \frac{3}{5},\\
x_2 &= \frac{1}{11}x_1 + \frac{1}{11}x_3 - \frac{3}{11}x_4 + \frac{25}{11},\\
x_3 &= -\frac{1}{5}x_1 + \frac{1}{10}x_2 + \frac{1}{10}x_4 - \frac{11}{10},\\
x_4 &= -\frac{3}{8}x_2 + \frac{1}{8}x_3 + \frac{15}{8}.
\end{aligned}$$

$$\boxed{\begin{aligned}
E_1:&\quad 10x_1 - x_2 + 2x_3 = 6,\\
E_2:&\quad -x_1 + 11x_2 - x_3 + 3x_4 = 25,\\
E_3:&\quad 2x_1 - x_2 + 10x_3 - x_4 = -11,\\
E_4:&\quad 3x_2 - x_3 + 8x_4 = 15
\end{aligned}}$$

From the initial approximation $\mathbf{x}^{(0)} = (0, 0, 0, 0)^t$ we have $\mathbf{x}^{(1)}$ given by

$$
\begin{aligned}
x_1^{(1)} &= \qquad\qquad \frac{1}{10}x_2^{(0)} - \frac{1}{5}x_3^{(0)} \qquad\qquad + \frac{3}{5} = 0.6000,\\
x_2^{(1)} &= \frac{1}{11}x_1^{(0)} \qquad\qquad + \frac{1}{11}x_3^{(0)} - \frac{3}{11}x_4^{(0)} + \frac{25}{11} = 2.2727,\\
x_3^{(1)} &= -\frac{1}{5}x_1^{(0)} + \frac{1}{10}x_2^{(0)} \qquad\qquad + \frac{1}{10}x_4^{(0)} - \frac{11}{10} = -1.1000,\\
x_4^{(1)} &= \qquad\qquad - \frac{3}{8}x_2^{(0)} + \frac{1}{8}x_3^{(0)} \qquad\qquad + \frac{15}{8} = 1.8750.
\end{aligned}
$$

Additional iterates, $\mathbf{x}^{(k)} = (x_1^{(k)}, x_2^{(k)}, x_3^{(k)}, x_4^{(k)})^t$, are generated in a similar manner and are presented in Table 7.1.

Table 7.1

k	0	1	2	3	4	5
$x_1^{(k)}$	0.0000	0.6000	1.0473	0.9326	1.0152	0.9890
$x_2^{(k)}$	0.0000	2.2727	1.7159	2.053	1.9537	2.0114
$x_3^{(k)}$	0.0000	−1.1000	−0.8052	−1.0493	−0.9681	−1.0103
$x_4^{(k)}$	0.0000	1.8750	0.8852	1.1309	0.9739	1.0214

6	7	8	9	10
1.0032	0.9981	1.0006	0.9997	1.0001
1.9922	2.0023	1.9987	2.0004	1.9998
−0.9945	−1.0020	−0.9990	−1.0004	−0.9998
0.9944	1.0036	0.9989	1.0006	0.9998

Remark: The exact solution is $x=(1,2,-1,1)^t$.

Linear system $\mathbf{A}\mathbf{x} = \mathbf{b}$

$$A = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{bmatrix}$$

write $A = D - L - U$ where

$$D = \begin{bmatrix} a_{11} & 0 & \cdots & 0 \\ 0 & a_{22} & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & a_{nn} \end{bmatrix} \qquad L = \begin{bmatrix} 0 & \cdots & \cdots & 0 \\ -a_{21} & \ddots & & \vdots \\ \vdots & \ddots & \ddots & \vdots \\ -a_{n1} & \cdots & -a_{n,n-1} & 0 \end{bmatrix}$$

and

$$U = \begin{bmatrix} 0 & -a_{12} & \cdots & -a_{1n} \\ \vdots & \ddots & \ddots & \vdots \\ \vdots & & \ddots & -a_{n-1,n} \\ 0 & \cdots & \cdots & 0 \end{bmatrix}$$

The equation $A\mathbf{x} = \mathbf{b}$, or $(D - L - U)\mathbf{x} = \mathbf{b}$, is then transformed into

$$D\mathbf{x} = (L + U)\mathbf{x} + \mathbf{b}$$

and, if D^{-1} exists, that is, if $a_{ii} \neq 0$ for each i, then

$$\mathbf{x} = D^{-1}(L + U)\mathbf{x} + D^{-1}\mathbf{b}$$

This results in the matrix form of the Jacobi iterative technique:

$$\mathbf{x}^{(k)} = D^{-1}(L + U)\mathbf{x}^{(k-1)} + D^{-1}\mathbf{b}, \quad k = 1, 2, \ldots$$

Assumption: $a_{ii} \neq 0$, for each $i = 1, 2, \ldots, n$.

This formulation based on D, L, U is only for analysis.

Use the formulation on page 8 in codes.

Introducing the notation $T_j = D^{-1}(L + U)$ and $\mathbf{c}_j = D^{-1}\mathbf{b}$ gives the Jacobi technique the form

$$\mathbf{x}^{(k)} = T_j\mathbf{x}^{(k-1)} + \mathbf{c}_j$$

Jacobi