Tridiagonal Matrices: Thomas Algorithm

W. T. Lee
MS6021, Scientific Computation, University of Limerick

The Thomas algorithm is an efficient way of solving tridiagonal matrix systems. It is based on LU decomposition in which the matrix system $Mx = r$ is rewritten as $LUx = r$ where $L$ is a lower triangular matrix and $U$ is an upper triangular matrix. The system can be efficiently solved by setting $Ux = \rho$ and then solving first $L\rho = r$ for $\rho$ and then $Ux = \rho$ for $x$. The Thomas algorithm consists of two steps. In Step 1 decomposing the matrix into $M = LU$ and solving $L\rho = r$ are accomplished in a single downwards sweep, taking us straight from $Mx = r$ to $Ux = \rho$. In step 2 the equation $Ux = \rho$ is solved for $x$ in an upwards sweep.

1. STAGE 1

In the first stage the matrix equation $Mx = r$ is converted to the form $Ux = \rho$. Initially the matrix equation looks like:

$$
\begin{bmatrix}
  b_1 & c_1 & 0 & 0 & 0 \\
  a_2 & b_2 & c_2 & 0 & 0 \\
  0 & a_3 & b_3 & c_3 & 0 \\
  0 & 0 & a_4 & b_4 & c_4 \\
  0 & 0 & 0 & a_5 & b_5 & c_5 \\
  0 & 0 & 0 & 0 & a_6 & b_6
\end{bmatrix}
\begin{bmatrix}
  x_1 \\
  x_2 \\
  x_3 \\
  x_4 \\
  x_5 \\
  x_6
\end{bmatrix}
= 
\begin{bmatrix}
  r_1 \\
  r_2 \\
  r_3 \\
  r_4 \\
  r_5 \\
  r_6
\end{bmatrix}
$$

Row 1

$$b_1x_1 + c_1x_2 = r_1$$

Divide through by $b_1$

$$x_1 + \frac{c_1}{b_1}x_2 = \frac{r_1}{b_1}$$

Rewrite:

$$x_1 + \gamma_1x_2 = \rho_1, \quad \gamma_1 = \frac{c_1}{b_1}, \quad \rho_1 = \frac{r_1}{b_1}$$

Row 2

$$a_2x_1 + b_2x_2 + c_2x_3 = r_2$$

Use $a_2$ times row 1 of the matrix to eliminate the first term

$$a_2 \left( x_1 + \gamma_1x_2 = \rho_1 \right)$$

We can rewrite this as

$$x_3 + \gamma_2x_4 = \rho_3, \quad \gamma_2 = \frac{c_2}{b_2 - a_2\gamma_1}, \quad \rho_3 = \frac{r_3 - a_2\rho_1}{b_2 - a_2\gamma_1}$$
Row 4.  
\[ a_4 x_3 + b_4 x_4 + c_4 x_5 = r_4 \]

Use \( a_4 \) times row 3 of the matrix to eliminate the first term

\[ a_4 \begin{pmatrix} x_3 + \gamma_3 x_4 \end{pmatrix} = \rho_3 \]

\[
\begin{array}{c|c}
\hline
a_4 x_3 & b_4 x_4 + c_4 x_5 = r_4 \\
\hline
\end{array}
\]

\[ (b_4 - a_4 \gamma_3) x_4 + c_4 x_5 = r_4 - a_4 \rho_3 \]

Divide through by \( (b_4 - a_4 \gamma_3) \) to get

\[ x_4 + \frac{c_4}{b_4 - a_4 \gamma_3} x_5 = \frac{r_4 - a_4 \rho_3}{b_4 - a_4 \gamma_3} \]

We can rewrite this as

\[ x_4 + \gamma_4 x_5 = \rho_4, \quad \gamma_4 = \frac{c_4}{b_4 - a_4 \gamma_3}, \quad \rho_4 = \frac{r_4 - a_4 \rho_3}{b_4 - a_4 \gamma_3} \]

Row 5.  
\[ a_5 x_4 + b_5 x_5 + c_5 x_6 = r_5 \]

Use \( a_5 \) times row 4 of the matrix to eliminate the first term

\[ a_5 \begin{pmatrix} x_4 + \gamma_4 x_5 \end{pmatrix} = \rho_4 \]

\[
\begin{array}{c|c}
\hline
a_5 x_4 & b_5 x_5 + c_5 x_6 = r_5 \\
\hline
\end{array}
\]

\[ (b_5 - a_5 \gamma_4) x_5 + c_5 x_6 = r_5 - a_5 \rho_4 \]

Divide through by \( (b_5 - a_5 \gamma_4) \) to get

\[ x_5 + \frac{c_5}{b_5 - a_5 \gamma_4} x_6 = \frac{r_5 - a_5 \rho_4}{b_5 - a_5 \gamma_4} \]

We can rewrite this as

\[ x_5 + \gamma_5 x_6 = \rho_5, \quad \gamma_5 = \frac{c_5}{b_5 - a_5 \gamma_4}, \quad \rho_5 = \frac{r_5 - a_5 \rho_4}{b_5 - a_5 \gamma_4} \]
Row 5: \( x_5 + \gamma_5 x_6 = \rho_5 \).
Rearrange to get: \( x_5 = \rho_5 - \gamma_5 x_6 \).

\[
\begin{pmatrix}
1 & \gamma_1 & 0 & 0 & 0 & 0 \\
0 & 1 & \gamma_2 & 0 & 0 & 0 \\
0 & 0 & 1 & \gamma_3 & 0 & 0 \\
0 & 0 & 0 & 1 & \gamma_4 & 0 \\
0 & 0 & 0 & 0 & 1 & \gamma_5 \\
0 & 0 & 0 & 0 & 0 & 1
\end{pmatrix}
\begin{pmatrix}
x_1 \\
x_2 \\
x_3 \\
x_4 \\
x_5 \\
x_6
\end{pmatrix}
=
\begin{pmatrix}
\rho_1 \\
\rho_2 \\
\rho_3 \\
\rho_4 \\
\rho_5 \\
\rho_6
\end{pmatrix}
\]

Row 4: \( x_4 + \gamma_4 x_5 = \rho_4 \).
Rearrange to get: \( x_4 = \rho_4 - \gamma_4 x_5 \).

\[
\begin{pmatrix}
1 & \gamma_1 & 0 & 0 & 0 & 0 \\
0 & 1 & \gamma_2 & 0 & 0 & 0 \\
0 & 0 & 1 & \gamma_3 & 0 & 0 \\
0 & 0 & 0 & 1 & \gamma_4 & 0 \\
0 & 0 & 0 & 0 & 1 & \gamma_5 \\
0 & 0 & 0 & 0 & 0 & 1
\end{pmatrix}
\begin{pmatrix}
x_1 \\
x_2 \\
x_3 \\
x_4 \\
x_5 \\
x_6
\end{pmatrix}
=
\begin{pmatrix}
\rho_1 \\
\rho_2 \\
\rho_3 \\
\rho_4 \\
\rho_5 \\
\rho_6
\end{pmatrix}
\]

Row 3: \( x_3 + \gamma_3 x_4 = \rho_3 \).
Rearrange to get: \( x_3 = \rho_3 - \gamma_3 x_4 \).

\[
\begin{pmatrix}
1 & \gamma_1 & 0 & 0 & 0 & 0 \\
0 & 1 & \gamma_2 & 0 & 0 & 0 \\
0 & 0 & 1 & \gamma_3 & 0 & 0 \\
0 & 0 & 0 & 1 & \gamma_4 & 0 \\
0 & 0 & 0 & 0 & 1 & \gamma_5 \\
0 & 0 & 0 & 0 & 0 & 1
\end{pmatrix}
\begin{pmatrix}
x_1 \\
x_2 \\
x_3 \\
x_4 \\
x_5 \\
x_6
\end{pmatrix}
=
\begin{pmatrix}
\rho_1 \\
\rho_2 \\
\rho_3 \\
\rho_4 \\
\rho_5 \\
\rho_6
\end{pmatrix}
\]

Row 2: \( x_2 + \gamma_2 x_3 = \rho_2 \).
Rearrange to get: \( x_2 = \rho_2 - \gamma_2 x_3 \).

\[
\begin{pmatrix}
1 & \gamma_1 & 0 & 0 & 0 & 0 \\
0 & 1 & \gamma_2 & 0 & 0 & 0 \\
0 & 0 & 1 & \gamma_3 & 0 & 0 \\
0 & 0 & 0 & 1 & \gamma_4 & 0 \\
0 & 0 & 0 & 0 & 1 & \gamma_5 \\
0 & 0 & 0 & 0 & 0 & 1
\end{pmatrix}
\begin{pmatrix}
x_1 \\
x_2 \\
x_3 \\
x_4 \\
x_5 \\
x_6
\end{pmatrix}
=
\begin{pmatrix}
\rho_1 \\
\rho_2 \\
\rho_3 \\
\rho_4 \\
\rho_5 \\
\rho_6
\end{pmatrix}
\]

Row 1: \( x_1 + \gamma_1 x_2 = \rho_1 \).
Rearrange to get: \( x_1 = \rho_1 - \gamma_1 x_2 \).

\[
\begin{pmatrix}
1 & \gamma_1 & 0 & 0 & 0 & 0 \\
0 & 1 & \gamma_2 & 0 & 0 & 0 \\
0 & 0 & 1 & \gamma_3 & 0 & 0 \\
0 & 0 & 0 & 1 & \gamma_4 & 0 \\
0 & 0 & 0 & 0 & 1 & \gamma_5 \\
0 & 0 & 0 & 0 & 0 & 1
\end{pmatrix}
\begin{pmatrix}
x_1 \\
x_2 \\
x_3 \\
x_4 \\
x_5 \\
x_6
\end{pmatrix}
=
\begin{pmatrix}
\rho_1 \\
\rho_2 \\
\rho_3 \\
\rho_4 \\
\rho_5 \\
\rho_6
\end{pmatrix}
\]

At this point \( x \), the solution to the matrix equation, is fully determined.

III. IN PRACTICE

The Thomas algorithm is used because it is fast and because tridiagonal matrices often occur in practice. (This argument is slightly circular because people often manipulate the problems they are working on to reduce them to solving a tridiagonal matrix problem.) Although it is rare, the algorithm can be unstable if \( b_i - a_i \gamma_{i-1} \) is zero or numerically zero for any \( i \). This will occur if the tridiagonal matrix is singular, but in rare cases can occur if it is non-singular. The condition for the algorithm to be stable is

\[
\| b_i \| > \| a_i \| + \| c_i \|
\]

for all \( i \). The matrix problems which result from the discretisation of partial differential equations nearly all satisfy this criterion.

If the algorithm is numerically unstable then you must rearrange the equations: known as pivoting. Standard LU decomposition algorithms for full or banded matrices include pivoting. (But first you should check to make sure you have not made a mistake in formulating the problem.)