Tridiagonal Systems

Charles Kankelborg

Rev. January 15, 2009

1 The Problem

A matrix equation $M\mathbf{u} = \mathbf{r}$ is just a way of writing $N$ linear, algebraic equations. The $i$th equation is:

$$
\sum_j M_{ij} u_j = r_i.
$$

(1)

The standard problem is to solve for $\mathbf{u}$ given $M$ and $\mathbf{r}$. A square matrix is said to be tridiagonal when

$$
M_{ij} = 0, \quad |i - j| > 1.
$$

(2)

For example,

$$
\begin{pmatrix}
  b_1 & c_1 & 0 & 0 \\
  a_2 & b_2 & c_2 & 0 \\
  0 & a_3 & b_3 & c_3 \\
  0 & 0 & a_4 & b_4 \\
\end{pmatrix}
\begin{pmatrix}
  u_1 \\
  u_2 \\
  u_3 \\
  u_4 \\
\end{pmatrix}
=
\begin{pmatrix}
  r_1 \\
  r_2 \\
  r_3 \\
  r_4 \\
\end{pmatrix}.
$$

(3)

2 Algorithm

Have a look Numerical Recipes Section 2.4. The Numerical Recipes function tridag is easily understood without LU decomposition.

To solve the $4 \times 4$ system in equation 3 for $\mathbf{u}$, we begin by using row 1 to eliminate $a_2$ from row 2. Similarly, we eliminate all the $a_i$ row by row.

Here is how the tridag implementation works. The first row stands for the equation:
\[ b_1 u_1 + c_1 u_2 = r_1. \]

Let \( \beta_1 = b_1 \),

\[ v_1 = \frac{r_1}{\beta_1}, \]  

\[ \gamma_2 = \frac{c_1}{\beta_1} \] (6)

If we divide row 1 by \( \beta_1 \), its equation becomes

\[ u_1 + \gamma_2 u_2 = v_1. \] (7)

The equation for the second row is:

\[ a_2 u_1 + b_2 u_2 + c_2 u_3 = r_2. \]

Let \( \beta_2 = b_2 - a_2 \gamma_2 \),

\[ v_2 = \frac{(r_2 - a_2 v_1)}{\beta_2}, \]

\[ \gamma_3 = \frac{c_2}{\beta_2}. \]

We now subtract \( a_2 \) times row 1 from row 2. Dividing the result by \( \beta_2 \), our new row 2 is written:

\[ u_2 + \gamma_3 u_3 = v_2. \] (8)

This process may be repeated over the range \( i = 2, 3, \ldots N \) using the recursion formulas

\[ \gamma_i = \frac{c_{i-1}}{\beta_{i-1}}, \] (9)

\[ \beta_i = b_i - a_i \gamma_i, \] (10)

\[ v_i = \frac{r_i - a_i v_{i-1}}{\beta_i}. \] (11)
The result is that the matrix equation is rewritten in upper triangular form. For our $4 \times 4$ example,

$$
\begin{pmatrix}
1 & \gamma_2 & 0 & 0 \\
0 & 1 & \gamma_3 & 0 \\
0 & 0 & 1 & \gamma_4 \\
0 & 0 & 0 & 1
\end{pmatrix}
\begin{pmatrix}
u_1 \\
u_2 \\
u_3 \\
u_4
\end{pmatrix}
=
\begin{pmatrix}
v_1 \\
v_2 \\
v_3 \\
v_4
\end{pmatrix}.
$$

(12)

This problem is trivially solved by starting from the bottom and back substituting as you go up:

\begin{align}
 u_N &= v_N, \\
 u_i &= v_i - \gamma_{i+1} u_{i+1}.
\end{align}

(13) (14)

### 3 How to Break the Algorithm

If $\beta_i = 0$, for some $i$, the program quits to avoid a division by zero. The authors of *Numerical Recipes* assure us that this doesn’t happen very often. For example, it can be shown that $\beta_i = 0$ cannot occur in the calculation of $y''_i$ for the cubic spline.

### 4 Applications

Tridiagonal matrices come up in a variety of contexts:

- Cubic splines (*NR* § 3.3)
- A concluding step in diagonalization of matrices (*NR* §§ 11.0, 11.2)
- Implicit differencing schemes for diffusive PDEs (*NR* § 19.2)
- Cyclic reduction methods for PDE BV problems (*NR* § 19.4)

### 5 MATLAB and Octave

In MATLAB or Octave, the code $u = M \backslash x$ will solve the matrix equation for $u$. The backslash operator is smart enough (as of MATLAB 7.5, and Octave 2.9) to test whether $M$ is tridiagonal. If so, then a fast algorithm like the one described above is used. This all requires that your matrix is stored as a sparse matrix; see, e.g., *spdiags*. 

3