In numerical linear algebra, the **tridiagonal matrix algorithm**, also known as the **Thomas algorithm** (named after Llewellyn Thomas), is a simplified form of Gaussian elimination that can be used to solve tridiagonal systems of equations. A tridiagonal system for n unknowns may be written as

$$a_i x_{i-1} + b_i x_i + c_i x_{i+1} = d_i,$$

where $a_1 = 0$ and $c_n = 0$.

$$\begin{bmatrix} b_1 & c_1 & & & 0 \\ a_2 & b_2 & c_2 & & \\ & a_3 & b_3 & \ddots & \\ & & \ddots & \ddots & c_{n-1} \\ 0 & & & a_n & b_n \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} d_1 \\ d_2 \\ d_3 \\ \vdots \\ d_n \end{bmatrix}.$$

For such systems, the solution can be obtained in $O(n)$ operations instead of $O(n^3)$ required by Gaussian elimination. A first sweep eliminates the a_i's, and then an (abbreviated) backward substitution produces the solution. Examples of such matrices commonly arise from the discretization of 1D Poisson equation and natural cubic spline interpolation; similar systems of matrices arise in tight binding physics or nearest neighbor effects models.

Thomas' algorithm is not stable in general, but is so in several special cases, such as when the matrix is diagonally dominant (either by rows or columns) or symmetric positive definite;[1][2] for a more precise characterization of stability of Thomas' algorithm, see Higham Theorem 9.12.[3] If stability is required in the general case, Gaussian elimination with partial pivoting (GEPP) is recommended instead.[4]

Method

The forward sweep consists of modifying the coefficients as follows, denoting the new coefficients with primes:

$$c'_i = \begin{cases} \dfrac{c_i}{b_i} & ; \quad i = 1 \\ \dfrac{c_i}{b_i - a_i c'_{i-1}} & ; \quad i = 2, 3, \dots, n-1 \end{cases}$$

and

1
2
3
4

$$d'_i = \begin{cases} \dfrac{d_i}{b_i} & ; \quad i = 1 \\ \dfrac{d_i - a_i d'_{i-1}}{b_i - a_i c'_{i-1}} & ; \quad i = 2, 3, \dots, n. \end{cases}$$

The solution is then obtained by back substitution:

$$x_n = d'_n$$

$$x_i = d'_i - c'_i x_{i+1} \qquad ; \; i = n-1, n-2, \ldots, 1.$$

Derivation

The derivation of the tridiagonal matrix algorithm is a special case of Gaussian elimination.

Suppose that the unknowns are $x_1, \ldots, x_n$, and that the equations to be solved are:

$$\begin{aligned} b_1 x_1 + c_1 x_2 &= d_1; & i &= 1 \\ a_i x_{i-1} + b_i x_i + c_i x_{i+1} &= d_i; & i &= 2, \ldots, n-1 \\ a_n x_{n-1} + b_n x_n &= d_n; & i &= n. \end{aligned}$$

Consider modifying the second ($i = 2$) equation with the first equation as follows:

$$(\text{equation 2}) \cdot b_1 - (\text{equation 1}) \cdot a_2$$

which would give:

$$(a_2 x_1 + b_2 x_2 + c_2 x_3) b_1 - (b_1 x_1 + c_1 x_2) a_2 = d_2 b_1 - d_1 a_2$$

$$(b_2 b_1 - c_1 a_2) x_2 + c_2 b_1 x_3 = d_2 b_1 - d_1 a_2$$

where the second equation immediately above is a simplified version of the equation immediately preceding it. The effect is that x_1 has been eliminated from the second equation. Using a similar tactic with the **modified** second equation on the third equation yields:

$$(a_3 x_2 + b_3 x_3 + c_3 x_4)(b_2 b_1 - c_1 a_2) - ((b_2 b_1 - c_1 a_2) x_2 + c_2 b_1 x_3) a_3 = d_3 (b_2 b_1 - c_1 a_2) - (d_2 b_1 - d_1 a_2) a_3$$

$$(b_3(b_2b_1-c_1a_2)-c_2b_1a_3)x_3+c_3(b_2b_1-c_1a_2)x_4 = d_3(b_2b_1-c_1a_2)-(d_2b_1-d_1a_2)a_3.$$

This time x_2 was eliminated. If this procedure is repeated until the n^{th} row; the (modified) n^{th} equation will involve only one unknown, x_n. This may be solved for and then used to solve the $(n-1)^{th}$ equation, and so on until all of the unknowns are solved for.

Clearly, the coefficients on the modified equations get more and more complicated if stated explicitly. By examining the procedure, the modified coefficients (notated with tildes) may instead be defined recursively:

$$\tilde{a}_i = 0$$

$$\tilde{b}_1 = b_1$$

$$\tilde{b}_i = b_i\tilde{b}_{i-1} - \tilde{c}_{i-1}a_i$$

$$\tilde{c}_1 = c_1$$

$$\tilde{c}_i = c_i\tilde{b}_{i-1}$$

$$\tilde{d}_1 = d_1$$

$$\tilde{d}_i = d_i\tilde{b}_{i-1} - \tilde{d}_{i-1}a_i.$$

To further hasten the solution process, $\tilde{b}_i$ may be divided out (if there's no division by zero risk), the newer modified coefficients, each notated with a prime, will be:

$$a'_i = 0$$

$$b'_i = 1$$

$$c'_1 = \frac{c_1}{b_1}$$

$$c'_i = \frac{c_i}{b_i - c'_{i-1}a_i}$$

$$d'_1 = \frac{d_1}{b_1}$$

$$d'_i = \frac{d_i - d'_{i-1}a_i}{b_i - c'_{i-1}a_i}.$$

This gives the following system with the same unknowns and coefficients defined in terms of the original ones above:

$$\begin{array}{lll} x_i + c'_i x_{i+1} = d'_i & ; & i = 1, \ldots, n-1 \\ x_n = d'_n & ; & i = n. \end{array}$$

The last equation involves only one unknown. Solving it in turn reduces the next last equation to one unknown, so that this backward substitution can be used to find all of the unknowns:

$$x_n = d'_n$$

$$x_i = d'_i - c'_i x_{i+1} \qquad ; \ i = n-1, n-2, \ldots, 1.$$

Variants

In some situations, particularly those involving periodic boundary conditions, a slightly perturbed form of the tridiagonal system may need to be solved:

$$\begin{aligned} a_1 x_n + b_1 x_1 + c_1 x_2 &= d_1, \\ a_i x_{i-1} + b_i x_i + c_i x_{i+1} &= d_i, \qquad i = 2, \ldots, n-1 \\ a_n x_{n-1} + b_n x_n + c_n x_1 &= d_n. \end{aligned}$$

In this case, we can make use of the Sherman-Morrison formula to avoid the additional operations of Gaussian elimination and still use the Thomas algorithm. The method requires solving a modified non-cyclic version of the system for both the input and a sparse corrective vector, and then combining the solutions. This can be done efficiently if both solutions are computed at once, as the forward portion of the pure tridiagonal matrix algorithm can be shared.

In other situations, the system of equations may be **block tridiagonal** (see block matrix), with smaller submatrices arranged as the individual elements in the above matrix system(e.g., the 2D Poisson problem). Simplified forms of Gaussian elimination have been developed for these situations.[5]

The textbook *Numerical Mathematics* by Quarteroni, Sacco and Saleri, lists a modified version of the algorithm which avoids some of the divisions (using instead multiplications), which is beneficial on some computer architectures.

References

-
-
-

Category:Numerical linear algebra

[5]