Tridiagonal Matrix Algorithm (Thomas Algorithm)

Tamas Kis | tamas.a.kis@outlook.com | https://github.com/tamaskis

Contents

1 TRIDIAGONAL MATRIX ALGORITHM (THOMAS ALGORITHM)

A tridiagonal linear system is one of the form

$$\begin{bmatrix} b_1 & c_1 & & & & \\ a_1 & b_2 & c_2 & & & \\ & a_2 & \ddots & \ddots & & \\ & & \ddots & \ddots & c_{n-2} & \\ & & & a_{n-2} & b_{n-1} & c_{n-1} \\ & & & & a_{n-1} & b_n \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_{n-1} \\ x_n \end{bmatrix} = \begin{bmatrix} d_1 \\ d_2 \\ \vdots \\ d_{n-1} \\ d_n \end{bmatrix}$$

We can define the $\mathbf{x}$ and $\mathbf{d}$ vectors as

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_{n-1} \\ x_n \end{bmatrix}, \quad \mathbf{d} = \begin{bmatrix} d_1 \\ d_2 \\ \vdots \\ d_{n-1} \\ d_n \end{bmatrix}$$

and the $n \times n$ **triadiagonal matrix**[1], $\mathbf{A}$, as

$$\mathbf{A} = \begin{bmatrix} b_1 & c_1 & & & & \\ a_1 & b_2 & c_2 & & & \\ & a_2 & \ddots & \ddots & & \\ & & \ddots & \ddots & c_{n-2} & \\ & & & a_{n-2} & b_{n-1} & c_{n-1} \\ & & & & a_{n-1} & b_n \end{bmatrix} \tag{1}$$

Now we can write the tridiagonal linear system as

$$\mathbf{A}\mathbf{x} = \mathbf{d} \tag{2}$$

where $\mathbf{A} \in \mathbb{R}^{n \times n}$ and $\mathbf{x}, \mathbf{d} \in \mathbb{R}^n$.

The **tridiagonal matrix algorithm** (also known as the **Thomas algorithm**) is an algorithm that can efficiently solve the tridiagonal linear system (given by Eq. (2)) for $\mathbf{x}$. This algorithm uses three vectors, $\mathbf{a}$, $\mathbf{b}$, and $\mathbf{c}$, which we define as [1]

$$\mathbf{a} = \begin{bmatrix} a_1 \\ \vdots \\ a_{n-1} \end{bmatrix}, \quad \mathbf{b} = \begin{bmatrix} b_1 \\ \vdots \\ b_n \end{bmatrix}, \quad \mathbf{c} = \begin{bmatrix} c_1 \\ \vdots \\ c_{n-1} \end{bmatrix}$$

[1] In many references, a tridiagonal matrix is defined with the convention

$$\mathbf{A} = \begin{bmatrix} a_1 & b_1 & & & & \\ c_1 & a_2 & b_2 & & & \\ & c_2 & \ddots & \ddots & & \\ & & \ddots & \ddots & b_{n-2} & \\ & & & c_{n-2} & a_{n-1} & b_{n-1} \\ & & & & c_{n-1} & a_n \end{bmatrix}$$

When dealing with the tridiagonal matrix algorithm, a convention similar to the one in Eq. (1) is used almost exclusively. However, the convention that most sources have has the a_i's ranging from a_2 to a_n, which is inconvenient from a programming standpoint; therefore, I defined them here as ranging from a_1 to a_{n-1}. This convention is also reflected in Algorithm 1.

The tridiagonal matrix algorithm is shown below [1–3].

Algorithm 1: `tridiagonal`

Tridiagonal matrix algorithm (Thomas algorithm).

Given:

- $\mathbf{A} \in \mathbb{R}^{n \times n}$ - tridiagonal matrix
- $\mathbf{d} \in \mathbb{R}^{n}$ - vector

Note:

- $\mathbf{A}$ and $\mathbf{d}$ define the tridiagonal linear system $\mathbf{Ax} = \mathbf{d}$.

Procedure:

1. Determine n (where $\mathbf{A} \in \mathbb{R}^{n \times n}$).
2. Preallocate vectors of size $n \times 1$ to store $\mathbf{b}$ and $\mathbf{x}$.
3. Preallocate vectors of size $(n-1) \times 1$ to store $\mathbf{a}$ and $\mathbf{c}$.
4. Extract $\mathbf{a}$ from $\mathbf{A}$.

for $i = 2$ **to** n $a_{i-1} = A_{i,i-1}$

endExtract $\mathbf{b}$ from $\mathbf{A}$.

for $i = 1$ **to** n

$b_i = A_{i,i}$

end

6. Extract $\mathbf{c}$ from $\mathbf{A}$.

for $i = 2$ **to** n $c_{i-1} = A_{i-1,i}$

endForward elimination.

for $i = 1$ **to** n

$w = a_{i-1}/b_{i-1}$

$b_i = b_i - wc_{i-1}$

$d_i = d_i - wd_{i-1}$

end

8. Backward substitution.

for $i = n-1$ **to** 1 **by** -1

$x_i = (d_i - c_i x_{i+1})/b_i$

end

Return:

- $\mathbf{x} \in \mathbb{R}^{n}$ - solution of the tridiagonal linear system $\mathbf{Ax} = \mathbf{d}$

REFERENCES

[1] James Hateley. *Linear Systems of Equations and Direct Solvers*. MATH 3620 Course Reader (Vanderbilt University). 2019.

[2] *Tridiagonal matrix algorithm*. Wikipedia. https://en.wikipedia.org/wiki/Tridiagonal_matrix_algorithm (accessed: January 9, 2021).

[3] *Tridiagonal matrix algorithm – TDMA (Thomas algorithm)*. CFD Online. https://www.cfd-online.com/Wiki/Tridiagonal_matrix_algorithm_-_TDMA_(Thomas_algorithm) (accessed: January 9, 2021).