Tridiagonal Matrix Algorithm

MATLAB Implementation

Tamas Kis | kis@stanford.edu

Tamas Kis
https://github.com/tamaskis

Contents

tridiagonal

Solves the tridiagonal linear system $\mathbf{A}\mathbf{x} = \mathbf{d}$ for $\mathbf{x}$ using the tridiagonal matrix algorithm (i.e. the Thomas algorithm).

Syntax

```
x = tridiagonal(A,d)
```

Description

`x = tridiagonal(A,d)` solves the tridiagonal linear system $\mathbf{A}\mathbf{x} = \mathbf{d}$ for $\mathbf{x} \in \mathbb{R}^n$, where $A \in \mathbb{R}^{n\times n}$ is a tridiagonal matrix and $\mathbf{d} \in \mathbb{R}^n$ is a vector.

Examples

Example 1

Solve the tridiagonal linear system $\mathbf{A}\mathbf{x} = \mathbf{d}$ for $\mathbf{x}$, where

$$\mathbf{A} = \begin{bmatrix} 1 & 2 & 0 & 0 & 0 \\ 3 & 4 & 5 & 0 & 0 \\ 0 & 6 & 7 & 8 & 0 \\ 0 & 0 & 9 & 1 & 2 \\ 0 & 0 & 0 & 3 & 4 \end{bmatrix}, \quad \mathbf{d} = \begin{bmatrix} 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{bmatrix}$$

■ ***SOLUTION***

Entering $\mathbf{A}$ and $\mathbf{d}$ into MATLAB,

```
% defines tridiagonal matrix A
A = [1,2,0,0,0;
     3,4,5,0,0;
     0,6,7,8,0;
     0,0,9,1,2;
     0,0,0,3,4];

% defines vector d
d = [1;
     2;
     3;
     4;
     5];
```

To solve the tridiagonal linear system for $\mathbf{x}$,

```
x = tridiagonal(A,d)
```

This yields the result

```
x =
```

```
   -0.7229
    0.8614
    0.1446
   -0.3976
    1.5482
```

Links

MATLAB® Central's File Exchange:
https://www.mathworks.com/matlabcentral/fileexchange/85438-tridiagonal-matrix-algorithm-thomas-alg-tridiagonal

GitHub®:
https://github.com/tamaskis/tridiagonal-MATLAB

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

---

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

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

However, 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 extremely inconvenient from an algorithmic standpoint; therefore, I defined them here as ranging from a_1 to a_{n-1}, and this is also reflected in Algorithm 1.

define as [1]

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

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

Algorithm 1: Tridiagonal matrix algorithm (Thomas algorithm).

Given: $\mathbf{A}, \mathbf{x}, \mathbf{d}$

// determines n (where $\mathbf{A} \in \mathbb{R}^{n\times n}$)
$n = \text{size}(\mathbf{A}, 1)$

Preallocate vectors of size $n \times 1$ to store $\mathbf{b}$ and $\mathbf{x}$.
Preallocate vectors of size $(n-1) \times 1$ to store $\mathbf{a}$ and $\mathbf{c}$.

// extracts $\mathbf{a}$ from $\mathbf{A}$
for $i = 2$ **to** n **do**
  $a_{i-1} = A_{i,i-1}$
end

// extracts $\mathbf{b}$ from $\mathbf{A}$
for $i = 1$ **to** n **do**
  $b_i = A_{i,i}$
end

// extracts $\mathbf{c}$ from $\mathbf{A}$
for $i = 2$ **to** n **do**
  $c_{i-1} = A_{i-1,i}$
end

// forward elimination
for $i = 1$ **to** n **do**
  $w = a_{i-1}/b_{i-1}$
  $b_i = b_i - wc_{i-1}$
  $d_i = d_i - wd_{i-1}$
end

// backward substitution
$x_n = d_n/b_n$
for $i = n-1$ **to** 1 **by** -1 **do**
  $x_i = (d_i - c_i x_{i+1})/b_i$
end

return $\mathbf{x}$

References

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

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

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