MAP 2210 – Aplicações de Álgebra Linear

1º Semestre - 2020

Prof. Dr. Luis Carlos de Castro Santos

lsantos@ime.usp.br

Objetivos

Formação básica de álgebra linear aplicada a problemas numéricos. Resolução de problemas em microcomputadores usando linguagens e/ou software adequados fora do horário de aula.

NÃO ESQUEÇA DE INICIAR A GRAVAÇÃO

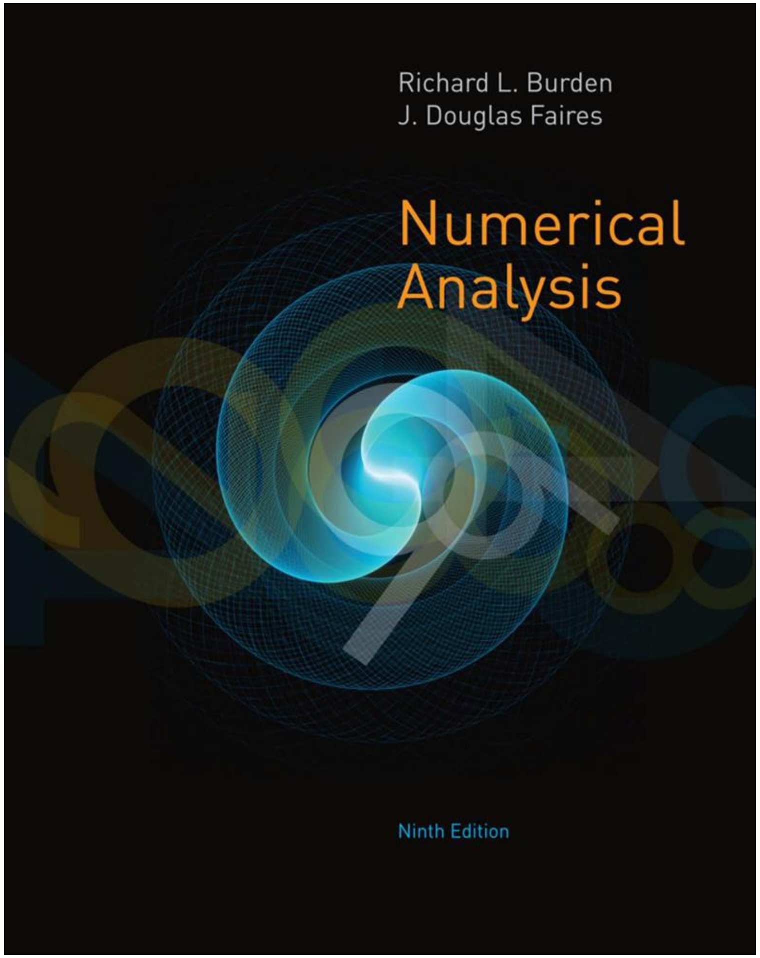
MAP 2210 – Aplicações de Álgebra Linear
1º Semestre - 2020

Prof. Dr. Luis Carlos de Castro Santos
lsantos@ime.usp.br

Objetivos

Formação básica de álgebra linear aplicada a problemas numéricos. Resolução de problemas em microcomputadores usando linguagens e/ou software adequados fora do horário de aula.

MAP2210



Numerical Analysis

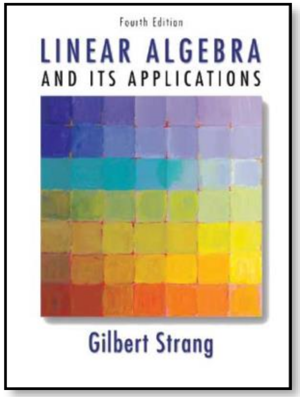
NINTH EDITION

Richard L. Burden
Youngstown State University

J. Douglas Faires
Youngstown State University

MAP2210

+



Fourth Edition
LINEAR ALGEBRA
AND ITS APPLICATIONS
Gilbert Strang

+

6.6 Special Types of Matrices

We now turn attention to two classes of matrices for which Gaussian elimination can be performed effectively without row interchanges.

Corollary 6.29 Let A be a symmetric $n \times n$ matrix for which Gaussian elimination can be applied without row interchanges. Then A can be factored into LDL^t, where L is lower triangular with 1s on its diagonal and D is the diagonal matrix with $a_{11}^{(1)}, \ldots, a_{nn}^{(n)}$ on its diagonal. ■

Example 3 Determine the LDL^t factorization of the positive definite matrix

$$A = \begin{bmatrix} 4 & -1 & 1 \\ -1 & 4.25 & 2.75 \\ 1 & 2.75 & 3.5 \end{bmatrix}.$$

$$A = LDL^t = \begin{bmatrix} 1 & 0 & 0 \\ -0.25 & 1 & 0 \\ 0.25 & 0.75 & 1 \end{bmatrix} \begin{bmatrix} 4 & 0 & 0 \\ 0 & 4 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & -0.25 & 0.25 \\ 0 & 1 & 0.75 \\ 0 & 0 & 1 \end{bmatrix}.$$ ■

Algorithm 6.5 is easily modified to factor the symmetric matrices described in Corollary 6.29. It simply requires adding a check to ensure that the diagonal elements are nonzero. The Cholesky Algorithm 6.6 produces the LL^t factorization described in Corollary 6.28.

Corollary 6.28 The matrix A is positive definite if and only if A can be factored in the form LL^t, where L is lower triangular with nonzero diagonal entries. ∎

Example 4 Determine the Cholesky LL^t factorization of the positive definite matrix

$$A = \begin{bmatrix} 4 & -1 & 1 \\ -1 & 4.25 & 2.75 \\ 1 & 2.75 & 3.5 \end{bmatrix}.$$

Example 4

Determine the Cholesky LL^t factorization of the positive definite matrix

$$A = \begin{bmatrix} 4 & -1 & 1 \\ -1 & 4.25 & 2.75 \\ 1 & 2.75 & 3.5 \end{bmatrix}.$$

Solution The LL^t factorization does not necessarily has 1s on the diagonal of the lower triangular matrix L so we need to have

$$A = \begin{bmatrix} a_{11} & a_{21} & a_{31} \\ a_{21} & a_{22} & a_{32} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} = \begin{bmatrix} l_{11} & 0 & 0 \\ l_{21} & l_{22} & 0 \\ l_{31} & l_{32} & l_{33} \end{bmatrix} \begin{bmatrix} l_{11} & l_{21} & l_{31} \\ 0 & l_{22} & l_{32} \\ 0 & 0 & l_{33} \end{bmatrix}$$

$$= \begin{bmatrix} l_{11}^2 & l_{11}l_{21} & l_{11}l_{31} \\ l_{11}l_{21} & l_{21}^2 + l_{22}^2 & l_{21}l_{31} + l_{22}l_{32} \\ l_{11}l_{31} & l_{21}l_{31} + l_{22}l_{32} & l_{31}^2 + l_{32}^2 + l_{33}^2 \end{bmatrix}$$

Thus

$$a_{11}: \quad 4 = l_{11}^2 \implies l_{11} = 2, \qquad a_{21}: \quad -1 = l_{11}l_{21} \implies l_{21} = -0.5$$

$$a_{31}: \quad 1 = l_{11}l_{31} \implies l_{31} = 0.5, \qquad a_{22}: \quad 4.25 = l_{21}^2 + l_{22}^2 \implies l_{22} = 2$$

$$a_{32}: \quad 2.75 = l_{21}l_{31} + l_{22}l_{32} \implies l_{32} = 1.5, \qquad a_{33}: \quad 3.5 = l_{31}^2 + l_{32}^2 + l_{33}^2 \implies l_{33} = 1,$$

and we have

$$A = LL^t = \begin{bmatrix} 2 & 0 & 0 \\ -0.5 & 2 & 0 \\ 0.5 & 1.5 & 1 \end{bmatrix} \begin{bmatrix} 2 & -0.5 & 0.5 \\ 0 & 2 & 1.5 \\ 0 & 0 & 1 \end{bmatrix}.$$

■

ALGORITHM 6.6

Cholesky

To factor the positive definite $n \times n$ matrix A into LL^t, where L is lower triangular:

INPUT the dimension n; entries a_{ij}, for $1 \leq i, j \leq n$ of A.

OUTPUT the entries l_{ij}, for $1 \leq j \leq i$ and $1 \leq i \leq n$ of L. (*The entries of* $U = L^t$ *are* $u_{ij} = l_{ji}$, *for* $i \leq j \leq n$ *and* $1 \leq i \leq n$.)

Step 1 Set $l_{11} = \sqrt{a_{11}}$.

Step 2 For $j = 2, \ldots, n$, set $l_{j1} = a_{j1}/l_{11}$.

Step 3 For $i = 2, \ldots, n-1$ do Steps 4 and 5.

Step 4 Set $l_{ii} = \left(a_{ii} - \sum_{k=1}^{i-1} l_{ik}^2\right)^{1/2}$.

Step 5 For $j = i+1, \ldots, n$

set $l_{ji} = \left(a_{ji} - \sum_{k=1}^{i-1} l_{jk} l_{ik}\right) / l_{ii}$.

Step 6 Set $l_{nn} = \left(a_{nn} - \sum_{k=1}^{n-1} l_{nk}^2\right)^{1/2}$.

Step 7 OUTPUT (l_{ij} for $j = 1, \ldots, i$ and $i = 1, \ldots, n$);
STOP.

The LDL^t factorization described in Algorithm 6.5 requires

$$\frac{1}{6}n^3 + n^2 - \frac{7}{6}n \text{ multiplications/divisions} \quad \text{and} \quad \frac{1}{6}n^3 - \frac{1}{6}n \text{ additions/subtractions.}$$

The LL^t Cholesky factorization of a positive definite matrix requires only

$$\frac{1}{6}n^3 + \frac{1}{2}n^2 - \frac{2}{3}n \text{ multiplications/divisions} \quad \text{and} \quad \frac{1}{6}n^3 - \frac{1}{6}n \text{ additions/subtractions.}$$

This computational advantage of Cholesky's factorization is misleading, because it requires extracting n square roots. However, the number of operations required for computing the n square roots is a linear factor of n and will decrease in significance as n increases.

If the Cholesky factorization given in Algorithm 6.6 is preferred, the additional steps for solving the system $A\mathbf{x} = \mathbf{b}$ are as follows. First delete the STOP statement from Step 7. Then add

Step 8 Set $y_1 = b_1/l_{11}$.

Step 9 For $i = 2, \ldots, n$ set $y_i = \left(b_i - \sum_{j=1}^{i-1} l_{ij} y_j\right) \Big/ l_{ii}$.

Step 10 Set $x_n = y_n/l_{nn}$.

Step 11 For $i = n-1, \ldots, 1$ set $x_i = \left(y_i - \sum_{j=i+1}^{n} l_{ji} x_j\right) \Big/ l_{ii}$.

Step 12 OUTPUT (x_i for $i = 1, \ldots, n$);
STOP.

Steps 8–12 require $n^2 + n$ multiplications/divisions and $n^2 - n$ additions/ subtractions.

Retomando o exemplo da aula anterior, agora usando a decomposição de Cholesky:

b.
$$\begin{aligned} 4x_1 + 2x_2 + 2x_3 &= 0, \\ 2x_1 + 6x_2 + 2x_3 &= 1, \\ 2x_1 + 2x_2 + 5x_3 &= 0. \end{aligned}$$

$$A = \begin{bmatrix} 4 & 2 & 2 \\ 2 & 6 & 2 \\ 2 & 2 & 5 \end{bmatrix} \qquad b = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}$$

Seguindo os passos do algoritmo:

INPUT the dimension n; entries a_{ij}, for $1 \leq i, j \leq n$ of A.

$$n = 3, a_{11} = 4, a_{21} = 2, a_{31} = 2$$

$$a_{22} = 6, a_{32} = 2$$

$$a_{33} = 5$$

OUTPUT the entries l_{ij}, for $1 \leq j \leq i$ and $1 \leq i \leq n$ of L. (*The entries of* $U = L^t$ *are* $u_{ij} = l_{ji}$, *for* $i \leq j \leq n$ *and* $1 \leq i \leq n$.)

$$l_{11}, l_{22}, l_{33}, l_{21}, l_{31}, l_{32}$$

Step 1 Set $l_{11} = \sqrt{a_{11}}$.

Step 2 For $j = 2, \ldots, n$, set $l_{j1} = a_{j1}/l_{11}$.

Step 1 $\quad l_{11} = \sqrt{a_{11}} = \sqrt{4} = 2$

Step 2

$j = 2$

$$l_{21} = \frac{a_{21}}{l_{11}} = \frac{2}{2} = 1$$

$j = 3$

$$l_{31} = \frac{a_{31}}{l_{11}} = \frac{2}{2} = 1$$

Step 3 For $i = 2, \ldots, n-1$ do Steps 4 and 5.

Step 4 Set $l_{ii} = \left(a_{ii} - \sum_{k=1}^{i-1} l_{ik}^2\right)^{1/2}$.

Step 5 For $j = i+1, \ldots, n$

set $l_{ji} = \left(a_{ji} - \sum_{k=1}^{i-1} l_{jk} l_{ik}\right) / l_{ii}$.

Step 3 $i = 2,2$

Step 4 $l_{22} = \sqrt{a_{22} - l_{21}^2} = \sqrt{5}$

Step 5 $j = 3,3$

$$l_{32} = \frac{a_{32} - l_{31} l_{21}}{l_{22}} = \frac{2 - 1.1}{\sqrt{5}} = \frac{1}{\sqrt{5}}$$

Step 6 Set $l_{nn} = \left(a_{nn} - \sum_{k=1}^{n-1} l_{nk}^2\right)^{1/2}$.

Step 6 $n = 3$

$$l_{33} = \sqrt{a_{33} - l_{31}^2 - l_{32}^2} = \sqrt{5 - 1 - \frac{1}{5}} = \sqrt{\frac{19}{5}}$$

Step 7 OUTPUT (l_{ij} for $j = 1, \ldots, i$ and $i = 1, \ldots, n$);

$$l_{11} = 2, l_{22} = \sqrt{5}, l_{33} = \sqrt{\frac{19}{5}}, l_{21} = 1, l_{31} = 1, l_{32} = \frac{1}{\sqrt{5}}$$

Na forma matricial:

$$\mathrm{L} = \begin{bmatrix} 2 & 0 & 0 \\ 1 & \sqrt{5} & 0 \\ 1 & \frac{1}{\sqrt{5}} & \sqrt{\frac{19}{5}} \end{bmatrix} \quad L^T = \begin{bmatrix} 2 & 1 & 1 \\ 0 & \sqrt{5} & \frac{1}{\sqrt{5}} \\ 0 & 0 & \sqrt{\frac{19}{5}} \end{bmatrix}$$

Com a decomposição disponível pode-se partir para a solução do sistema

Partindo do sistema

$$Ax = \mathrm{b}$$

pela decomposição $LL^T x = b$

Utiliza-se uma sequência de varíaveis intermediárias que aproveitam a estrutura das matrizes

$$Ly = b \qquad y = \begin{bmatrix} 0 \\ \frac{1}{\sqrt{5}} \\ -\frac{1}{\sqrt{95}} \end{bmatrix}$$

$$L^T x = y \qquad x = \begin{bmatrix} \frac{3}{38} \\ \frac{4}{19} \\ -\frac{1}{19} \end{bmatrix}$$

9. Modify the Cholesky Algorithm as suggested in the text so that it can be used to solve linear systems, and use the modified algorithm to solve the linear systems in Exercise 7.

b.
$$\begin{aligned}
4x_1 + x_2 + x_3 + x_4 &= 0.65,\\
x_1 + 3x_2 - x_3 + x_4 &= 0.05,\\
x_1 - x_2 + 2x_3 &= 0,\\
x_1 + x_2 + 2x_4 &= 0.5.
\end{aligned}$$

Band Matrices

The last class of matrices considered are *band matrices*. In many applications, the band matrices are also strictly diagonally dominant or positive definite.

Definition 6.30

An $n \times n$ matrix is called a **band matrix** if integers p and q, with $1 < p, q < n$, exist with the property that $a_{ij} = 0$ whenever $p \le j - i$ or $q \le i - j$. The **band width** of a band matrix is defined as $w = p + q - 1$. ■

The number p describes the number of diagonals above, and including, the main diagonal on which nonzero entries may lie. The number q describes the number of diagonals below, and including, the main diagonal on which nonzero entries may lie. For example, the matrix

$$A = \begin{bmatrix} 7 & 2 & 0 \\ 3 & 5 & -1 \\ 0 & -5 & -6 \end{bmatrix}$$

is a band matrix with $p = q = 2$ and bandwidth $2 + 2 - 1 = 3$.

The definition of band matrix forces those matrices to concentrate all their nonzero entries about the diagonal. Two special cases of band matrices that occur frequently have $p = q = 2$ and $p = q = 4$.

Tridiagonal Matrices

Matrices of bandwidth 3 occurring when $p = q = 2$ are called **tridiagonal** because they have the form

$$A = \begin{bmatrix} a_{11} & a_{12} & 0 & \cdots & \cdots & 0 \\ a_{21} & a_{22} & a_{23} & \ddots & & \vdots \\ 0 & a_{32} & a_{33} & a_{34} & \ddots & 0 \\ \vdots & \ddots & \ddots & \ddots & \ddots & a_{n-1,n} \\ 0 & \cdots & 0 & & a_{n,n-1} & a_{nn} \end{bmatrix}.$$

The factorization algorithms can be simplified considerably in the case of band matrices because a large number of zeros appear in these matrices in regular patterns. It is particularly interesting to observe the form the Crout or Doolittle method assumes in this case.

To illustrate the situation, suppose a tridiagonal matrix A can be factored into the triangular matrices L and U. Then A has at most $(3n-2)$ nonzero entries. Then there are only $(3n-2)$ conditions to be applied to determine the entries of L and U, provided, of course, that the zero entries of A are also obtained.

Suppose that the matrices L and U also have tridiagonal form, that is,

$$L = \begin{bmatrix} l_{11} & 0 & \cdots & \cdots & 0 \\ l_{21} & l_{22} & \ddots & & \vdots \\ 0 & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & l_{n,n-1} & l_{nn} \end{bmatrix} \quad \text{and} \quad U = \begin{bmatrix} 1 & u_{12} & 0 & \cdots & 0 \\ 0 & 1 & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & 0 \\ \vdots & & \ddots & \ddots & u_{n-1,n} \\ 0 & \cdots & \cdots & 0 & 1 \end{bmatrix}.$$

There are $(2n-1)$ undetermined entries of L and $(n-1)$ undetermined entries of U, which totals $(3n-2)$, the number of possible nonzero entries of A. The 0 entries of A are obtained automatically.

The multiplication involved with $A = LU$ gives, in addition to the 0 entries,

$$a_{11} = l_{11};$$

$$a_{i,i-1} = l_{i,i-1}, \quad \text{for each } i = 2, 3, \ldots, n; \tag{6.13}$$

$$a_{ii} = l_{i,i-1}u_{i-1,i} + l_{ii}, \quad \text{for each } i = 2, 3, \ldots, n; \tag{6.14}$$

and

$$a_{i,i+1} = l_{ii}u_{i,i+1}, \quad \text{for each } i = 1, 2, \ldots, n-1. \tag{6.15}$$

A solution to this system is found by first using Eq. (6.13) to obtain all the nonzero off-diagonal terms in L and then using Eqs. (6.14) and (6.15) to alternately obtain the remainder of the entries in U and L. Once an entry L or U is computed, the corresponding entry in A is not needed. So the entries in A can be overwritten by the entries in L and U with the result that no new storage is required.

Example 5

Determine the Crout factorization of the symmetric tridiagonal matrix

$$\begin{bmatrix} 2 & -1 & 0 & 0 \\ -1 & 2 & -1 & 0 \\ 0 & -1 & 2 & -1 \\ 0 & 0 & -1 & 2 \end{bmatrix},$$

and use this factorization to solve the linear system

$$\begin{aligned} 2x_1 - x_2 &= 1, \\ -x_1 + 2x_2 - x_3 &= 0, \\ - x_2 + 2x_3 - x_4 &= 0, \\ - x_3 + 2x_4 &= 1. \end{aligned}$$

Solution The LU factorization of A has the form

$$A = \begin{bmatrix} a_{11} & a_{12} & 0 & 0 \\ a_{21} & a_{22} & a_{23} & 0 \\ 0 & a_{32} & a_{33} & a_{34} \\ 0 & 0 & a_{43} & a_{44} \end{bmatrix} = \begin{bmatrix} l_{11} & 0 & 0 & 0 \\ l_{21} & l_{22} & 0 & 0 \\ 0 & l_{32} & l_{33} & 0 \\ 0 & 0 & l_{43} & l_{44} \end{bmatrix} \begin{bmatrix} 1 & u_{12} & 0 & 0 \\ 0 & 1 & u_{23} & 0 \\ 0 & 0 & 1 & u_{34} \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} l_{11} & l_{11}u_{12} & 0 & 0 \\ l_{21} & l_{22} + l_{21}u_{12} & l_{22}u_{23} & 0 \\ 0 & l_{32} & l_{33} + l_{32}u_{23} & l_{33}u_{34} \\ 0 & 0 & l_{43} & l_{44} + l_{43}u_{34} \end{bmatrix}.$$

Thus

$$a_{11}: \quad 2 = l_{11} \implies l_{11} = 2, \qquad a_{12}: \quad -1 = l_{11}u_{12} \implies u_{12} = -\tfrac{1}{2},$$

$$a_{21}: \quad -1 = l_{21} \implies l_{21} = -1, \qquad a_{22}: \quad 2 = l_{22} + l_{21}u_{12} \implies l_{22} = -\tfrac{3}{2},$$

$$a_{23}: \quad -1 = l_{22}u_{23} \implies u_{23} = -\tfrac{2}{3}, \qquad a_{32}: \quad -1 = l_{32} \implies l_{32} = -1,$$

$$a_{33}: \quad 2 = l_{33} + l_{32}u_{23} \implies l_{33} = \tfrac{4}{3}, \qquad a_{34}: \quad -1 = l_{33}u_{34} \implies u_{34} = -\tfrac{3}{4},$$

$$a_{43}: \quad -1 = l_{43} \implies l_{43} = -1, \qquad a_{44}: \quad 2 = l_{44} + l_{43}u_{34} \implies l_{44} = \tfrac{5}{4}.$$

This gives the Crout factorization

$$A = \begin{bmatrix} 2 & -1 & 0 & 0 \\ -1 & 2 & -1 & 0 \\ 0 & -1 & 2 & -1 \\ 0 & 0 & -1 & 2 \end{bmatrix} = \begin{bmatrix} 2 & 0 & 0 & 0 \\ -1 & \frac{3}{2} & 0 & 0 \\ 0 & -1 & \frac{4}{3} & 0 \\ 0 & 0 & -1 & \frac{5}{4} \end{bmatrix} \begin{bmatrix} 1 & -\frac{1}{2} & 0 & 0 \\ 0 & 1 & -\frac{2}{3} & 0 \\ 0 & 0 & 1 & -\frac{3}{4} \\ 0 & 0 & 0 & 1 \end{bmatrix} = LU.$$

Solving the system

$$L\mathbf{z} = \begin{bmatrix} 2 & 0 & 0 & 0 \\ -1 & \frac{3}{2} & 0 & 0 \\ 0 & -1 & \frac{4}{3} & 0 \\ 0 & 0 & -1 & \frac{5}{4} \end{bmatrix} \begin{bmatrix} z_1 \\ z_2 \\ z_3 \\ z_4 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 1 \end{bmatrix} \quad \text{gives} \quad \begin{bmatrix} z_1 \\ z_2 \\ z_3 \\ z_4 \end{bmatrix} = \begin{bmatrix} \frac{1}{2} \\ \frac{1}{3} \\ \frac{1}{4} \\ 1 \end{bmatrix},$$

and then solving

$$U\mathbf{x} = \begin{bmatrix} 1 & -\frac{1}{2} & 0 & 0 \\ 0 & 1 & -\frac{2}{3} & 0 \\ 0 & 0 & 1 & -\frac{3}{4} \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} \frac{1}{2} \\ \frac{1}{3} \\ \frac{1}{4} \\ 1 \end{bmatrix} \quad \text{gives} \quad \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}. \quad \blacksquare$$

Em termos práticos os problemas com matrizes tridigonais são tratados como implementações dedicadas da eliminação de Gauss usando a alocação das diagonais como vetores.

Tomando como exemplo um sistema de dimensão 4, onde as diagonais são armazenadas na forma de vetores, a, b, c e ou lado direito no vetor d:

$$\begin{bmatrix} b_1 & c_1 & 0 & 0 & d_1 \\ a_2 & b_2 & c_2 & 0 & d_2 \\ 0 & a_3 & b_3 & c_3 & d_3 \\ 0 & 0 & a_4 & b_4 & d_4 \end{bmatrix}$$

A eliminação da primeira coluna: $L_2 - \left(\frac{a_2}{b_1}\right) L_1$

altera somente os valores dos coeficientes b_2 e d_2

$$\color{red}{b_2} = b_2 - \left(\frac{a_2}{b_1}\right) c_1$$

$$\color{red}{d_2} = d_2 - \left(\frac{a_2}{b_1}\right) d_1$$

Armazenando os resultados nas mesmas posições

$$\begin{bmatrix} b_1 & c_1 & 0 & 0 & d_1 \\ 0 & b_2 & c_2 & 0 & d_2 \\ 0 & a_3 & b_3 & c_3 & d_3 \\ 0 & 0 & a_4 & b_4 & d_4 \end{bmatrix}$$

Da mesma forma a eliminação da segunda coluna: $L_3 - \left(\frac{a_3}{b_2}\right) L_2$

altera somente os valores dos coeficientes b_3 e d_3

$$b_3 = b_3 - \left(\frac{a_3}{b_2}\right) c_2$$

$$d_3 = d_3 - \left(\frac{a_3}{b_2}\right) d_2$$

Após esse passo tem-se

$$\begin{bmatrix} b_1 & c_1 & 0 & 0 & d_1 \\ 0 & b_2 & c_2 & 0 & d_2 \\ 0 & 0 & b_3 & c_3 & d_3 \\ 0 & 0 & a_4 & b_4 & d_4 \end{bmatrix}$$

Finalmente eliminando a última coluna: $L_4 - \left(\frac{a_4}{b_3}\right) L_3$

de forma análoga altera apenas os coeficientes b_4 e d_4

$$b_4 = b_4 - \left(\frac{a_4}{b_3}\right) c_3$$

$$d_4 = d_4 - \left(\frac{a_4}{b_3}\right) d_3$$

Terminada a eliminação

$$\begin{bmatrix} b_1 & c_1 & 0 & 0 & d_1 \\ 0 & b_2 & c_2 & 0 & d_2 \\ 0 & 0 & b_3 & c_3 & d_3 \\ 0 & 0 & 0 & b_4 & d_4 \end{bmatrix}$$

A solução pode ser encontrada por substituição reversa já que a matriz resultante já é triangular superior:

$$x_4 = \frac{d_4}{b_4}$$

$$x_3 = \left(\frac{d_3 - c_3 d_4}{b_3}\right)$$

$$x_2 = \left(\frac{d_2 - c_2 d_3}{b_2}\right)$$

$$x_1 = \left(\frac{d_1 - c_2 d_2}{b_1}\right)$$

Ou, utilizando as mesmas posições do vetor do lado direito

$$d_4 = \frac{d_4}{b_4}$$

$$d_3 = \left(\frac{d_3 - c_3 d_4}{b_3}\right)$$

$$d_2 = \left(\frac{d_2 - c_2 d_3}{b_2}\right)$$

$$d_1 = \left(\frac{d_1 - c_2 d_2}{b_1}\right)$$

Um exemplo da implementação em Python desse algoritmo teria forma

Tridiagonal Matrix Algorithm solver in Python

TDMAsolver.py

```python
import numpy as np

## Tri Diagonal Matrix Algorithm(a.k.a Thomas algorithm) solver
def TDMAsolver(a, b, c, d):
    '''
    TDMA solver, a b c d can be NumPy array type or Python list type.
    refer to http://en.wikipedia.org/wiki/Tridiagonal_matrix_algorithm
    and to http://www.cfd-online.com/Wiki/Tridiagonal_matrix_algorithm_-_TDMA_(Thomas_algorithm)
    '''
    nf = len(d) # number of equations
    ac, bc, cc, dc = map(np.array, (a, b, c, d)) # copy arrays
    for it in xrange(1, nf):
        mc = ac[it-1]/bc[it-1]
        bc[it] = bc[it] - mc*cc[it-1]
        dc[it] = dc[it] - mc*dc[it-1]

    xc = bc
    xc[-1] = dc[-1]/bc[-1]

    for il in xrange(nf-2, -1, -1):
        xc[il] = (dc[il]-cc[il]*xc[il+1])/bc[il]

    return xc
```

https://gist.github.com/cbellei/8ab3ab8551b8dfc8b081c518ccd9ada9

Use os problemas abaixo para encontrar a solução tanto a mão quanto em Python:

b.
$$\begin{aligned} 2x_1 - x_2 \qquad &= 5, \\ -x_1 + 3x_2 + x_3 &= 4, \\ x_2 + 4x_3 &= 0. \end{aligned}$$

c.
$$\begin{aligned} 2x_1 - x_2 \qquad\qquad &= 3, \\ x_1 + 2x_2 - x_3 \qquad &= 4, \\ x_2 - 2x_3 + x_4 &= 0, \\ x_3 + 2x_4 &= 6. \end{aligned}$$

Fim...

AULA 07