# Métodos Numéricos

Prof. Dr. Jonatha Costa

2024

# Organização

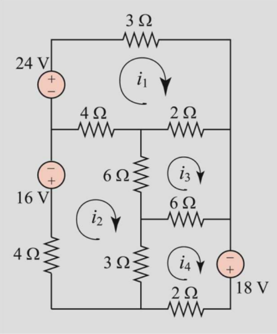
# MN aplicados à Engenharia

- Apresentar conteúdo de Resolução de Equações Lineares
  - Métodos diretos
    - Eliminação de Gauss
    - Eliminação de Gauss-Jordan
    - Fatoração LU
    - Método de Crout
    - Inversa de uma matriz
  - Métodos Iterativos
    - Gauss-Jacobi
    - Gauss-Seidel

# MN aplicados à Engenharia

**Exemplo**: Seja um problema de engenharia que requer a solução de um sistema de equações. Usando a lei de kirchhoff, as correntes $i_1, i_2, i_3$ e $i_4$ podem ser determinadas com a solução do seguinte sistema de quatro equações:

Figura: Circuito Elétrico



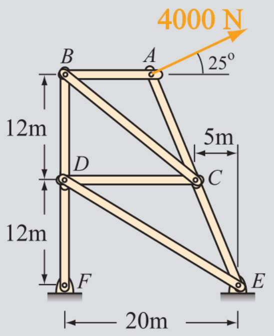

Fonte: GILAT,(2008)

$$\begin{cases} +9i_1 - 4i_2 - 2i_3 = 24 \\ -4i_1 + 17i_2 - 6i_3 - 3i_4 = -16 \\ -2i_1 - 6i_2 + 14i_3 - 6i_4 = 0 \\ -3i_2 - 6i_3 + 11i_4 = 18 \end{cases}$$

# MN aplicados à Engenharia

**Exemplo**: Seja o cálculo de força nos membros de uma treliça. As forças nos oito membros da treliça são determinadas a partir da solução do seguinte sistema de oito equações:

Figura: Dinâmica de forças




Fonte: GILAT,(2008)

$$
\begin{cases}
0,9231F_{AC} = 1690 \\
F_{AB} - 0,7809F_{BC} = 0 \\
F_{CD} + 0,8575F_{DE} = 0 \\
0,3846F_{CE} - 0,3846F_{AC} - 0,7809F_{BC} - F_{CD} = 0 \\
0,9231F_{AC} + 0,6247F_{BC} - 0,9231F_{CE} = 0 \\
-F_{AB} - 0,3846F_{AC} = 3625 \\
0,6247F_{BC} - F_{BD} = 0 \\
F_{BD} - 0,5145F_{DE} - F_{DF} = 0
\end{cases}
$$

# MN aplicados à Engenharia

## Conceituação

Um sistema de $m$ equações e $n$ variáveis é chamado de sistema de equações lineares e tem a seguinte forma genérica:

$$\begin{cases} a_{11}x_1 + a_{12}x_2 + \ldots + a_{1n}x_n = b_1 \\ a_{21}x_1 + a_{22}x_2 + \ldots + a_{2n}x_n = b_2 \\ \vdots \\ a_{n1}x_1 + a_{n2}x_2 + \ldots + a_{nn}x_n = b_n \end{cases}$$

**em que:** $a_{ij}$ são os coeficientes para $1 \leq i \leq m$, $1 \leq j \leq n$, $x_j$ são as variáveis e $b_i$ são as constantes.

A resolução de um sistema linear consiste em calcular os valores de $x_j$, para $j = 1, ..., n$, caso eles existam, que satisfaçam as $m$ equações simultaneamente.

O sistema linear pode ter:

* Mais equações do que incógnitas $(m > n)$;
* Mais incógnitas do que equações $(m < n)$
* O mesmo número de incógnitas e equações $(m = n)$.

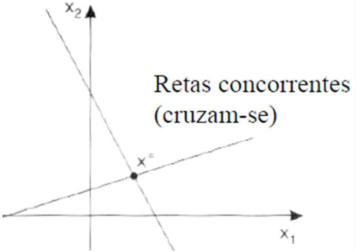
# MN aplicados à Engenharia

*Solução única*

$$\begin{cases} 2x_1 + x_2 = 3 \\ x_1 - 3x_2 = -2 \end{cases}$$

$$x^* = \begin{pmatrix} 1 \\ 1 \end{pmatrix}$$

Figura: Retas concorrentes




Fonte: DIAS,(2019)

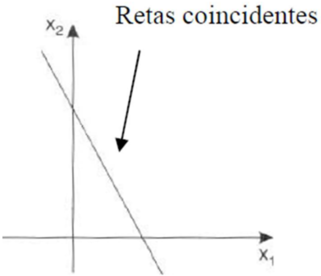
# MN aplicados à Engenharia

*Infinitas Soluções*

$$\begin{cases} 2x_1 + x_2 = 3 \\ 4x_1 - 2x_2 = 6 \end{cases}$$

para o qual $\forall x^* = (\alpha, 3 - 2\alpha)^t$ com $\alpha \in \mathbb{R}$, é solução.

Figura: Retas coincidentes




Fonte: DIAS,(2019)

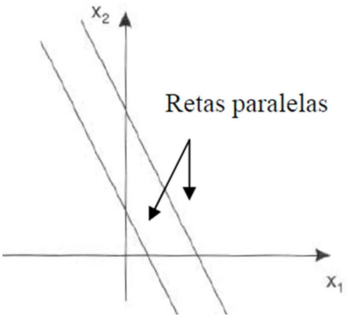
# MN aplicados à Engenharia

*Nenhuma solução*

$$\begin{cases} 2x_1 + x_2 = 3 \\ 4x_1 + 2x_2 = 2 \end{cases}$$

Determinante de $A$ é nulo.
$det(A) = 0 \rightarrow \nexists\ x^*\ in\ \mathbb{R}$.

Figura: Retas Parelelas




Fonte: DIAS,(2019)

# MN aplicados à Engenharia

*As operações elementares entre equações de um sistema linear são:*

1. Trocar as equações de posição
2. Multiplicar uma ou mais equações por constantes (chamamos múltiplos de equações):
3. Somar o múltiplo de uma equação por outra e aplicar uma operação elementar entre equações em um sistema linear implicará no sempre mesmo resultado $(x_1, x_2, ..., x_n)$.

# MN aplicados à Engenharia

- O sistema pode ser escrito na forma de um produto matricial $Ax = b$, em que as matrizes são definidas por:

$$[\mathbf{A}] = \begin{vmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \dots & a_{mn} \end{vmatrix}, [\mathbf{x}] = \begin{vmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{vmatrix} [\mathbf{b}] = \begin{vmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{vmatrix}$$

Em que:

- $A$ é a matriz $(m, n)$ dos coeficientes
- $x$ é o vetor das variáveis($n$ linhas)
- $b$ é o vetor das constantes($m$ linhas), termos independentes

- Obter a solução de $Ax = b$ implica em se obter os escalares $x_1, x_2 \dots, x_n$ que permitam escrever $b$ como combinação linear das $n$ colunas de $A$.

$$[\mathbf{b}] = x_1 \begin{bmatrix} a_{11} \\ a_{21} \\ \vdots \\ a_{m1} \end{bmatrix} + x_2 \begin{bmatrix} a_{12} \\ a_{22} \\ \vdots \\ a_{m2} \end{bmatrix} + \dots + x_n \begin{bmatrix} a_{1n} \\ a_{2n} \\ \vdots \\ a_{mn} \end{bmatrix}$$

# MN aplicados à Engenharia

## Métodos de soluções para Sistemas Lineares

**Métodos Diretos** - São aqueles que fornecem uma solução exata, a menos que existam erros de arredondamento.

- $x = A^{-1}b$
- Eliminação de Gauss;
- Pivotamento
- Fatoração LU

**Métodos Indiretos** - São aqueles que geram uma sequência de vetores x(k) a partir de uma aproximação inicial x(0).

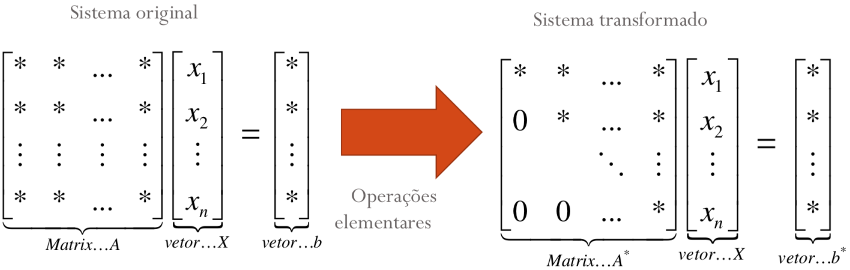
- Método Iterativo de Gauss;
- Método Iterativo de Gauss-Jacobi.

# MN aplicados à Engenharia

## Método da Eliminação de Gauss

Este método consiste em transformar o sistema linear original $Ax = b$ em um sistema linear equivalente $A * x = b*$ com matriz dos coeficientes **triangular superior**.

Figura: Eliminação de Gauss




Fonte: DIAS,(2019)

# MN aplicados à Engenharia

## Método da Eliminação de Gauss

*Considerações*

- Supor que a matriz $A$ seja quadrada $m = n$ e não singular.
- Adotado as seguintes notações
  - $i = 1, 2, ..., m$ (i-ésima linha);
  - $j = 1, 2, ..., n$ (j-ésima coluna);
  - $k = 1, ...$ (k-ésima etapa da eliminação)
  - $a_{ij}^{(k)}$ e $b_i^{(k)}$

Uma **matriz é singular** se e somente se seu determinante é nulo.

# MN aplicados à Engenharia

## Método da Eliminação de Gauss

*Procedimento:*

Para cada fase $k = 1, 2, \ldots, n$ da eliminação (ou pivoteamento):

- Determinar o pivô $a_{kk}^{(k)} \neq 0$ (ou não muito pequeno).
- Eliminar (zerar) os elementos da coluna $a_{ik}^{(k)}$ abaixo da $k$-ésima linha do pivô, para $i = k + 1, \ldots, n$.
- Determinar uma constante $m_{ik}$, de modo que, ao multiplicá-la pela $k$-ésima linha do pivô e subtrair com a $i$-ésima linha, esse elemento deverá ser zerado.

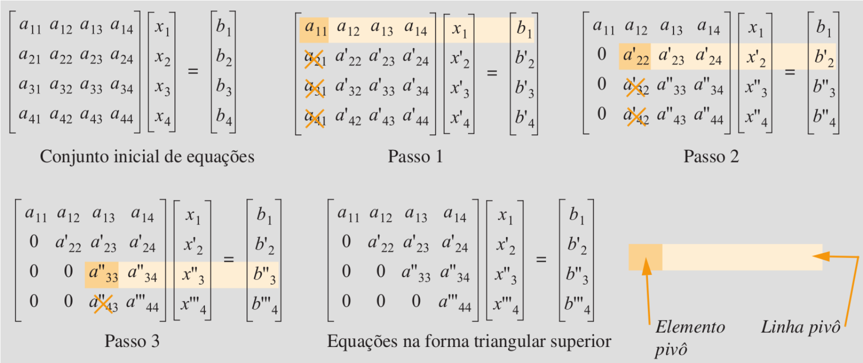
$$a_{ik}^{(k)} - m_{ik} a_{kk}^{(k)} = 0 \qquad\qquad m_{ik} = \frac{a_{ik}^{(k)}}{a_{kk}^{(k)}}$$

# MN aplicados à Engenharia

## Método da Eliminação de Gauss

*Procedimento:*

Figura: Eliminação de Gauss - Procedimento




Fonte: DIAS,(2019)

# MN aplicados à Engenharia

Figura: Pivotação

Após o Passo 1, a segunda equação tem um elemento pivô zero

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ 0 & 0 & a'_{23} & a'_{24} \\ 0 & a'_{32} & a'_{33} & a'_{34} \\ 0 & a'_{42} & a'_{43} & a'_{44} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} b_1 \\ b'_2 \\ b'_3 \\ b'_4 \end{bmatrix}$$

Usando pivotação, troca-se a segunda equação pela terceira.

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ 0 & a'_{32} & a'_{33} & a'_{34} \\ 0 & 0 & a'_{23} & a'_{24} \\ 0 & a'_{42} & a'_{43} & a'_{44} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} b_1 \\ b'_3 \\ b'_2 \\ b'_4 \end{bmatrix}$$

Fonte: GILAT (2008)

**Método da Eliminação de Gauss com Pivotação**

- Se durante o procedimento de Eliminação de Gauss uma equação pivô tiver um elemento pivô nulo e o sistema de equações tiver solução, uma equação com um elemento pivô diferente de zero sempre poderá ser encontrada.

# MN aplicados à Engenharia

Resolva o sistema linear: $\begin{cases} 10x_1 + 2x_2 + x_3 = 7 \\ x_1 + 5x_2 + x_3 = -8 \\ 2x_1 + 3x_2 + 10x_3 = 6 \end{cases}$

- Forma matricial Ax=b

$$\begin{vmatrix} 10 & 2 & 1 \\ 1 & 5 & 1 \\ 2 & 3 & 10 \end{vmatrix} . \begin{vmatrix} x1 \\ x2 \\ x3 \end{vmatrix} = \begin{vmatrix} 7 \\ -8 \\ 6 \end{vmatrix}$$

(1) **Método de Gauss**

$$\begin{vmatrix} 10 & 2 & 1 & 7 \\ 1 & 5 & 1 & -8 \\ 2 & 3 & 10 & 6 \end{vmatrix} \begin{matrix} m_{21} = \frac{A(2,1)}{A(1,1)} = \frac{1}{10} = 0,1 \\ L2 = L2 - m_{21} * L1 \\ \Longrightarrow \end{matrix} \begin{vmatrix} \boxed{10} & 2 & 1 & 7 \\ 0 & 4,8 & 0,9 & -8,7 \\ 2 & 3 & 10 & 6 \end{vmatrix} \Longrightarrow$$

$$\begin{vmatrix} \boxed{10} & 2 & 1 & 7 \\ 0 & 4,8 & 0,9 & -8,7 \\ 2 & 3 & 10 & 6 \end{vmatrix} \begin{matrix} m_{31} = \frac{A(3,1)}{A(1,1)} = \frac{2}{10} = 0,2 \\ L3 = L3 - m_{31} * L1 \\ \Longrightarrow \end{matrix} \begin{vmatrix} \boxed{10} & 2 & 1 & 7 \\ 0 & 4,8 & 0,9 & -8,7 \\ 0 & 2,6 & 9,8 & 4,6 \end{vmatrix} \Longrightarrow$$

$$\begin{vmatrix} 10 & 2 & 1 & 7 \\ 0 & \boxed{4,8} & 0,9 & -8,7 \\ 0 & 2,6 & 9,8 & 4,6 \end{vmatrix} \begin{matrix} m_{32} = \frac{A(3,2)}{A(2,2)} = \frac{2,6}{4,8} = 0,5416 \\ L3 = L3 - m_{32} * L2 \\ \Longrightarrow \end{matrix} \begin{vmatrix} 10 & 2 & 1 & 7 \\ 0 & \boxed{4,8} & 0,9 & -8,7 \\ 0 & 0 & 9,31 & 9,31 \end{vmatrix}$$

Logo:

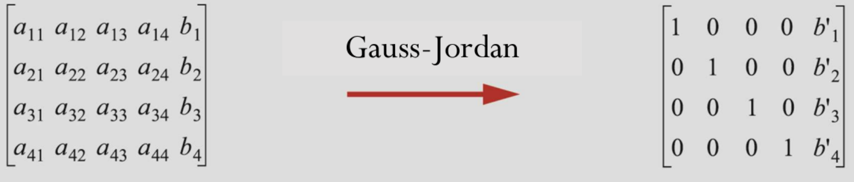
$$\begin{cases} x_3 = 9,31/9,31 \\ x_2 = (-8,7 - 0,9 * x3)/4,8 \\ x_1 = (7 - 1 * x3 - 2 * x2)/10 \end{cases} \qquad \begin{matrix} x_1 = 1 \\ x_2 = -2 \\ x_3 = 1 \end{matrix}$$

# MN aplicados à Engenharia

## Método de Eliminação de Gauss-Jordan

- A equação pivô é normalizada com a **divisão de todos os seus termos pelo coeficiente pivô**. Isso faz com que o coeficiente pivô seja igual a 1.
- A equação pivô é utilizada na eliminação dos elementos fora da diagonal principal em TODAS as demais equações, ou seja, o processo de eliminação aplica-se às equações acima e abaixo da equação pivô.
- A manipulação da equação pivô segue a mesma estrutura de Gauss para a obtenção dos elementos elementos fora da **diagonal**.

Figura: Gauss-Jordan




Fonte: GILAT,(2008)

# MN aplicados à Engenharia

Resolva o sistema linear: $\begin{cases} 10x_1 + 2x_2 + x_3 = 7 \\ x_1 + 5x_2 + x_3 = -8 \\ 2x_1 + 3x_2 + 10x_3 = 6 \end{cases}$

(2) **Método de Gauss-Jordan**

$$\left|\begin{array}{cccc} 10 & 2 & 1 & 7 \\ 1 & 5 & 1 & -8 \\ 2 & 3 & 10 & 6 \end{array}\right| \rightarrow \left|\begin{array}{cccc} 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & -2 \\ 0 & 0 & 1 & 1 \end{array}\right| \rightarrow \begin{cases} x_1 = 1 \\ x_2 = -2 \\ x_3 = 1 \end{cases}$$

# MN aplicados à Engenharia

## Fatoração LU

A base deste método, assim como o método da eliminação de Gauss, é o uso de uma propriedade elementar de sistemas de equações lineares que estabelece o seguinte:

- A solução de um sistema linear $Ax = b$ não se altera se o submetivermos a uma sequência de operações tais como:
  - Multiplicação de uma equação (linha) por uma constante não nula;
  - Soma do múltiplo de uma equação a outra;
  - Troca de posição de duas ou mais equações.
- Seja o sistema linear $Ax = b$, este processo de fatoração consiste em decompor a matriz A em um produto de dois ou mais fatores, na forma:

$$[A] = [L][U]$$

  em que:
  - L = Matriz triangular inferior (Decomposição LU)
  - U = Matriz triangular superior (Eliminação de Gauss)

# MN aplicados à Engenharia

## Fatoração LU

- Seja o sistema linear: $\begin{cases} Ax = b \\ LUx = b \end{cases}$
- Considerando, $Ux = y$ temos então dois sistemas: $Ly = b$
- Decomposição de Matriz $[a]$:

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \\ a_{41} & a_{42} & a_{43} & a_{44} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ m_{21} & 1 & 0 & 0 \\ m_{31} & m_{32} & 1 & 0 \\ m_{41} & m_{42} & m_{43} & 1 \end{bmatrix} \begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ 0 & a_{22} & a_{23} & a_{24} \\ 0 & 0 & a_{33} & a_{34} \\ 0 & 0 & 0 & a_{44} \end{bmatrix}$$

* *Os elementos diagonais de $[L]$ são todos iguais a 1 e os elementos abaixo desta são os multiplicadores $m_{ij}$ que multiplicam a equação pivô quando ela é usada para eliminar os elementos abaixo do coeficiente pivô no método de Gauss;*

* *A matriz triangular superior $[U]$ é a matriz de coeficientes $[a]$ obtida ao final do procedimento de Eliminação de Gauss.*

# MN aplicados à Engenharia

Resolva o sistema linear: $\begin{cases} 10x_1 + 2x_2 + x_3 = 7 \\ x_1 + 5x_2 + x_3 = -8 \\ 2x_1 + 3x_2 + 10x_3 = 6 \end{cases}$

## (3) Fatoração LU

- Aplicar o método de Eliminação de Gauss na Matriz A para obtenção de L e U;
  - Então, para a matriz A dada, tem-se:

$$L = \begin{pmatrix} 1 & 0 & 0 \\ 0,1 & 1 & 0 \\ 0,2 & 0,54 & 1 \end{pmatrix} \; e \; U = \begin{pmatrix} 10 & 2 & 1 \\ 0 & 4,8 & 0,9 \\ 0 & 0 & 9,31 \end{pmatrix}$$

- Resolver os dois sistemas lineares equivalentes obtidos pelas substituição $A = LU$
  - $Ly = b; L = \begin{vmatrix} 1 & 0 & 0 \\ 0,1 & 1 & 0 \\ 0,2 & 0,54 & 1 \end{vmatrix} . \begin{vmatrix} y_1 \\ y_2 \\ y_3 \end{vmatrix} = \begin{vmatrix} 7 \\ -8 \\ 6 \end{vmatrix} \rightarrow y = \begin{vmatrix} 7 \\ -8,7 \\ 9,31 \end{vmatrix}$
  - $Ux = y; U = \begin{vmatrix} 10 & 2 & 1 \\ 0 & 4,8 & 0,9 \\ 0 & 0 & 9,31 \end{vmatrix} . \begin{vmatrix} x_1 \\ x_2 \\ x_3 \end{vmatrix} = \begin{vmatrix} 7 \\ -8,7 \\ 9,31 \end{vmatrix} \rightarrow x = \begin{vmatrix} 1 \\ -2 \\ 1 \end{vmatrix}$

# MN aplicados à Engenharia

## Fatoração LU - Método de Crout

As matrizes L e U são da forma

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \\ a_{41} & a_{42} & a_{43} & a_{44} \end{bmatrix} = \begin{bmatrix} L_{11} & 0 & 0 & 0 \\ L_{21} & L_{22} & 0 & 0 \\ L_{31} & L_{32} & L_{33} & 0 \\ L_{41} & L_{42} & L_{43} & L_{44} \end{bmatrix} \begin{bmatrix} 1 & U_{12} & U_{13} & U_{14} \\ 0 & 1 & U_{23} & U_{24} \\ 0 & 0 & 1 & U_{34} \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Resolvendo o produto L e U, tem-se:

$$= \begin{bmatrix} L_{11} & L_{11}U_{12} & L_{11}U_{13} & L_{11}U_{14} \\ L_{21} & L_{21}U_{12} + L_{22} & L_{21}U_{13} + L_{22}U_{23} & L_{21}U_{14} + L_{22}U_{24} \\ L_{31} & L_{31}U_{12} + L_{32} & L_{31}U_{13} + L_{32}U_{23} + L_{33} & L_{31}U_{14} + L_{32}U_{24} + L_{33}U_{34} \\ L_{41} & L_{41}U_{12} + L_{42} & L_{41}U_{13} + L_{42}U_{23} + L_{43} & L_{41}U_{14} + L_{42}U_{24} + L_{43}U_{34} + L_{44} \end{bmatrix}$$

# MN aplicados à Engenharia

## Fatoração LU - Método de Crout

Igualando os elementos correspem quentes em ambos os lados da equação, pode-se encontrar os elementos das matrizes $[L]$ e $[U]$, como sendo:

- Na primeira linha:

$$L_{11} = a_{11}; U_{12} = \frac{a_{12}}{L_{11}}; U_{13} = \frac{a_{13}}{L_{11}}; U_{14} = \frac{a_{14}}{L_{11}}$$

- Na segunda linha:

$$L_{21} = a_{21}; L_{22} = a_{22} - L_{21}U_{12};$$
$$U_{23} = \frac{a_{23} - L_{21}U_{13}}{L_{22}}; U_{24} = \frac{a_{24} - L_{21}U_{14}}{L_{22}}$$

E assim sucessivamente!

# MN aplicados à Engenharia

## Inversa de uma Matriz

- A inversa de uma matriz quadrada $[a]$ é a matriz $[a]^{-1}$ tal que o produto das duas matrizes fornece a identidade.

$$[A][A]^{-1} = [I]$$

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \\ a_{41} & a_{42} & a_{43} & a_{44} \end{bmatrix} \cdot \begin{bmatrix} x_{11} & x_{12} & x_{13} & x_{14} \\ x_{21} & x_{22} & x_{23} & x_{24} \\ x_{31} & x_{32} & x_{33} & x_{34} \\ x_{41} & x_{42} & x_{43} & x_{44} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

- A solução da identidade é obtida através da solução das quatro equações:

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \\ a_{41} & a_{42} & a_{43} & a_{44} \end{bmatrix} \cdot \begin{bmatrix} x_{11} \\ x_{21} \\ x_{31} \\ x_{41} \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} \qquad \begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \\ a_{41} & a_{42} & a_{43} & a_{44} \end{bmatrix} \cdot \begin{bmatrix} x_{12} \\ x_{22} \\ x_{32} \\ x_{42} \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix}$$

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \\ a_{41} & a_{42} & a_{43} & a_{44} \end{bmatrix} \cdot \begin{bmatrix} x_{13} \\ x_{23} \\ x_{33} \\ x_{43} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix} \qquad \begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \\ a_{41} & a_{42} & a_{43} & a_{44} \end{bmatrix} \cdot \begin{bmatrix} x_{14} \\ x_{24} \\ x_{34} \\ x_{44} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}$$

# MN aplicados à Engenharia

## Método Iterativos

Consiste em colocar cada incógnita $x_i$ em função das outras variáveis, conforme segue:

Figura: Iterativos

$$
\begin{aligned}
a_{11}x_1 + a_{12}x_2 + a_{13}x_3 + a_{14}x_4 &= b_1\\
a_{21}x_1 + a_{22}x_2 + a_{23}x_3 + a_{24}x_4 &= b_2\\
a_{31}x_1 + a_{32}x_2 + a_{33}x_3 + a_{34}x_4 &= b_3\\
a_{41}x_1 + a_{42}x_2 + a_{43}x_3 + a_{44}x_4 &= b_4
\end{aligned}
$$

(a)

Escrevendo as equações em uma forma explícita $\longrightarrow$

$$
\begin{aligned}
x_1 &= [b_1 - (a_{12}x_2 + a_{13}x_3 + a_{14}x_4)]/a_{11}\\
x_2 &= [b_2 - (a_{21}x_1 + a_{23}x_3 + a_{24}x_4)]/a_{22}\\
x_3 &= [b_3 - (a_{31}x_1 + a_{32}x_2 + a_{34}x_4)]/a_{33}\\
x_4 &= [b_4 - (a_{21}x_1 + a_{42}x_2 + a_{43}x_3)]/a_{44}
\end{aligned}
$$

(b)

Fonte: GILAT,(2008)

Destacam-se os métodos:

- **Jacobi** - os valores das incógnitas no lado direito da equação são atualizados todos de uma vez no final de cada iteração.
- **Gauss-Seidel** - em que o valor de cada incógnita é atualizado (e usado no cálculo da nova estimativa das demais incógnitas dentro da mesma iteração) assim que se calcula uma nova estimativa para essa incógnita.

# MN aplicados à Engenharia

## Método Iterativos

- O processo de solução começa com a escolha de valores iniciais para as incógnitas (primeira solução estimada).
- Na primeira iteração, a primeira solução assumida é substituída no lado direito das equações, e os novos valores calculados para as incógnitas formam a segunda solução estimada.
- Na segunda iteração, a segunda solução é substituída de volta nas equações para que novos valores sejam obtidos para as incógnitas, e isso constitui a terceira solução estimada.
- As iterações continuam da mesma forma e, quando o método "dá certo", as soluções obtidas durante as iterações sucessivas convergem para a solução real.

Destacam-se os métodos:

- **Jacobi** - os valores das incógnitas no lado direito da equação são atualizados todos de uma vez no final de cada iteração.
- **Gauss-Seidel** - em que o valor de cada incógnita é atualizado (e usado no cálculo da nova estimativa das demais incógnitas dentro da mesma iteração) assim que se calcula uma nova estimativa para essa incógnita.

# MN aplicados à Engenharia

## Método Iterativos

Em um sistema com $n$ equações:

- As equações explícitas para as incógnitas $[x_i]$ são:

$$x_i = \frac{1}{a_{ii}}\left[b_i - \left(\sum_{j=1, j\neq i}^{n} a_{ij}x_j\right)\right] \tag{1}$$

  $i = 1, 2, \ldots, n$

- Uma condição suficiente para a convergência ocorre se, em cada uma das linhas da matriz de coeficientes $[a]$, o valor absoluto do elemento diagonal for maior que a soma dos valores absolutos dos elementos fora da diagonal.

$$|a_{ii}| \leq \sum_{j=1, j\neq i}^{n} |a_{ij}|$$

* Essa condição é suficiente, mas não necessária para a convergência do método iterativo e, quando ocorre a matriz $[a]$ é classificada como diagonalmente dominante.

# MN aplicados à Engenharia

## Método Iterativos de Jacobi

- Um valor inicial é escolhido para cada uma das incógnitas, $x_1^{(1)}, x_2^{(1)}, \ldots, x_n^{(1)}$ assumindo-se que o valor inicial de todas seja zero, caso não haja informações iniciais a respeito.
- A segunda estimativa da solução,$x_1^{(2)}, x_2^{(2)}, \ldots, x_n^{(2)}$, é calculada com a substituição do lado direito da equação 1 de modo que se tem:

$$x_i^{(2)} = \frac{1}{a_{ii}} \left[ b_i - \left( \sum_{j=1, j\neq i}^{n} a_{ij} x_j^{(1)} \right) \right] \ , \ i = 1, 2, \ldots, n$$

- Em geral, a $(k+1)$-ésima estimativa da solução é calculada a partir da k-ésima estimativa usando:

$$x_i^{(k+1)} = \frac{1}{a_{ii}} \left[ b_i - \left( \sum_{j=1, j\neq i}^{n} a_{ij} x_j^{(k)} \right) \right] \ , \ i = 1, 2, \ldots, n$$

- As iterações continuam até que as diferenças entre os valores obtidos nas iterações sucessivas sejam pequenas ou que o valor absoluto do erro relativo estimado de todas as incógnitas for menor que um valor $\varepsilon$ predeterminado:

$$\left| \frac{x_i^{(k+1)} - x_i^{(k)}}{x_i^{(k)}} \right| \leq \varepsilon \ , \ i = 1, 2, \ldots, n$$

# MN aplicados à Engenharia

Resolva o sistema linear:

$$\begin{cases} 10x_1 + 2x_2 + x_3 = 7 \\ x_1 + 5x_2 + x_3 = -8 \\ 2x_1 + 3x_2 + 10x_3 = 6 \end{cases}$$

(4) **Método de Jacobi**

- Descrição de equações $x_i$:
  - $x_1 = (7 - (2x_2 + x_3))/10$
  - $x_2 = (-8 - (x_1 + x_3))/5$
  - $x_3 = (6 - (2x_1 + 3x_2))/10$
  - Ponto de partida: $x_1 = 0, x_2 = 0, x_3 = 0$
- Segunda iteração $x_i^{(k=2)}$:
  - $x_1^{(2)} = (7 - (2.0 + 0))/10 = 0,7$
  - $x_2^{(2)} = (-8 - (0 + 0))/5 = -1,6$
  - $x_3^{(2)} = (6 - (2.0 + 3.0))/10 = 0,6$
  - Após a segunda iteração: $x_1 = 0,7, x_2 = -1,6, x_3 = 0,6$
- Terceira iteração $x_i^{(k=3)}$:
  - $x_1^{(3)} = (7 - (2.(-1,6) + 0,6))/10 = 0,96$
  - $x_2^{(3)} = (-8 - (0,7 + 0,6))/5 = -1,86$
  - $x_3^{(3)} = (6 - (2.(0,7) + 3.(-1,6))/10 = 0,94$
  - Após a terceira iteração: $x_1 = 0,96, x_2 = -1,86, x_3 = 0,94$
- Quarta iteração $x_i^{(k=4)}$:
  - $x_1^{(4)} = (7 - (2.(-1,86) + 0,94))/10 = 0,978$
  - $x_2^{(4)} = (-8 - (0,96 + 0,94))/5 = -1,98$
  - $x_3^{(4)} = (6 - (2.(0,96) + 3.(-1,86))/10 = 0,966$
  - Após a quarta iteração: $x_1 = 0,978, x_2 = -1,98, x_3 = 0,966$
- ...

# MN aplicados à Engenharia

## Método Iterativos de Gauss-Seidel

- No método de Gauss-Seidel, valores iniciais são assumidos para as incógnitas $x_2, x_3, \ldots, x_n$, (**exceto** $x_1$), assumindo-se que o valor inicial de todas seja zero, caso não haja informações iniciais a respeito.
- $x_1$ é obtido pela substituição dos valores assumidos na equação 1.
- Em seguida, o novo valor de $x_2$ é obtido pela substituição dos valores assumidos na equação 1 para $i = 2$, e assim por diante até que $i = n$, para concluir a **primeira iteração.**
- Em seguida, a segunda iteração começa com $i = 1$, em que um novo valor é calculado para $x_1$, e assim por diante.
- No método de Gauss-Seidel, **os valores atuais das incógnitas são utilizados no cálculo do novo valor da próxima incógnita.**

# MN aplicados à Engenharia

## Método Iterativos de Gauss-Seidel

- A aplicação da equações (1) no método de Gauss-Seidel leva à fórmula iterativa:

$$x_1^{(k+1)} = \frac{1}{a_{11}}\left[b_1 - \sum_{j=2}^{j=n} a_{1j}x_j^{(k)}\right]$$

$$x_i^{(k+1)} = \frac{1}{a_{ii}}\left[b_1 - \left(\sum_{j=1}^{j=i-1} a_{ij}x_j^{(k+1)} + \sum_{j=i+1}^{j=n} a_{ij}x_j^{(k)}\right)\right] , \; i = 2, 3, \ldots, n-1$$

$$x_n^{(k+1)} = \frac{1}{a_{nn}}\left[b_n - \sum_{j=1}^{j=n-1} a_{nj}x_j^{(k+1)}\right]$$

- Note que os valores das incógnitas na iteração $k+1$, são calculados obtidos na iteração $k+1$ para $j \leq i$ e usando os valores usando os valores para $j \leq i$.
- O critério de interrupção das iterações é o mesmo utilizado no método de Jacobi.
- O método de Gauss-Seidel converge mais rápido do que o método de Jacobi e requer menos memória computacional quando programado

# MN aplicados à Engenharia

Resolva o sistema linear:

$$\begin{cases} 10x_1 + 2x_2 + x_3 = 7 \\ x_1 + 5x_2 + x_3 = -8 \\ 2x_1 + 3x_2 + 10x_3 = 6 \end{cases}$$

(5) **Método de Gauss-Seidal**

- Descrição de equações $x_i$:
  - $x_1 = (7 - (2x_2 + x_3))/10$
  - $x_2 = (-8 - (x_1 + x_3))/5$
  - $x_3 = (6 - (2x_1 + 3x_2))/10$
  - Ponto de partida: $x_1 = 0, x_2 = 0, x_3 = 0$
- Segunda iteração $x_i^{(k=2)}$:
  - $x_1^{(2)} = (7 - (2.0 + 0))/10 =$ 0,7
  - $x_2^{(2)} = (-8 - ($ 0,7 $+ 0))/5 =$ -1,74
  - $x_3^{(2)} = (6 - (2.$ 0,7 $+ 3.$ -1,74 $))/10 = -0,982$
  - Após a segunda iteração: $x_1 = 0,7, x_2 = -1,74, x_3 = 0,982$
- Terceira iteração $x_i^{(k=3)}$:
  - $x_1^{(3)} = (7 - (2.(-1,74) + 0,982))/10 =$ 0,9498
  - $x_2^{(3)} = (-8 - ($ 0,9498 $+ 0,982))/5 =$ -1,9863
  - $x_3^{(3)} = (6 - (2.$ 0,7 $+ 3.$ -1,9863 $))/10 = -0,062$
  - Após a terceira iteração: $x_1 = 0,9498, x_2 = -1,9863, x_3 = 1,005$
- Quarta iteração $x_i^{(k=4)}$:
  - $x_1^{(4)} = (7 - (2. - 1,9863 + 1,005))/10 =$ 0,9966
  - $x_2^{(4)} = (-8 - ($ 0,9966 $+ 1,005))/5 =$ -2,000
  - $x_3^{(4)} = (6 - (2.$ 0,9966 $+ 3.$ -2,000 $))/10 = 1,000$
  - Após a quarta iteração: $x_1 = 0,9966, x_2 = -2,000, x_3 = 1,000$

# MN aplicados à Engenharia

## Resultados Jacobi e Gauss-Seidal

Figura: Comparativo de iterações para **erro de 0,001**

```
***** Método Iterativo de Jacobi *****

k          x1            x2            x3
 2      0.700000     -1.600000      0.600000
 3      0.960000     -1.860000      0.940000
 4      0.978000     -1.980000      0.966000
 5      0.999400     -1.988800      0.998400
 6      0.997920     -1.999560      0.996760
 7      1.000236     -1.998936      1.000284
 8      0.999759     -2.000104      0.999634
 9      1.000057     -1.999878      1.000079
10      0.999968     -2.000027      0.999952

***** Método Iterativo de Gauss-Seidal *****

k          x1            x2            x3
 2      0.700000     -1.740000      0.982000
 3      0.949800     -1.986360      1.005948
 4      0.996677     -2.000525      1.000822
 5      1.000023     -2.000169      1.000046
 6      1.000029     -2.000015      0.999999
```

Fonte: AUTOR,(2020)

Caso erro seja superado antes de 20 iterações o *script* interrompe as iterações!

# MN aplicados à Engenharia

## *Script Octave/Matlab* Jacobi e Gauss-Seidal

Figura: *Script* comparativo de iterações com **Erro de 0,001**

```
% Sistema Linear Iteração de Jacobi e Gauss-Seidel.
clc; clear all;   format bank
%*******************************************
fprintf('\n***** Método Iterativo de Jacobi *****\n\n');
x1=0;x2=0;x3=0;x4=0; n=20; err=0.0001;
a=0;b=0;c=0;
disp('k          x1             x2               x3  ');

for(k=2:n)
  a=(7-(2*x2+x3))/10;           % Atualiza a e usa Xi anterior
  b=(-8-(x1+x3))/5;             % Atualiza b e usa Xi anterior
  c=(6-(2*x1+3*x2))/10;         % Atualiza c e usa Xi anterior
  if((abs(x1-a) < err) && (abs(x2-b))<err && (abs(x3-c))<err) break end;
  x1=a;x2=b;x3=c;
  fprintf('%2.0f \t%-8.6f \t%-8.6f \t%-8.6f \t\n',k,x1,x2,x3);
end

%****************************************
%break
%*******************************************
fprintf('\n***** Método Iterativo de Gauss-Seidal *****\n\n')
x1=0;x2=0;x3=0;x4=0;
disp('k          x1             x2               x3  ');
for(k=2:n)
a=x1; x1=(7-(2*x2+x3))/10;          % Guarda x1(k-1) em a;
b=x2; x2=(-8-(x1+x3))/5;            % Guarda x2(k-1) em b;
c=x3; x3=(6-(2*x1+3*x2))/10;        % Guarda x3(k-1) em c;
if((abs(x1-a) < err) && (abs(x2-b))<err && (abs(x3-c))<err) break end;
fprintf('%2.0f \t%-8.6f \t%-8.6f \t%-8.6f \t\n',k,x1,x2,x3);
```

Fonte: AUTOR,(2020)

# MN aplicados à Engenharia

## *Script Python* Jacobi e Gauss-Seidal

Figura: *Script* comparativo de iterações com **Erro de 0,001**

```
print('\n****** Método Iterativo de Jacobi ******\n')
x1,x2,x3,x4 = 0,0,0,0
a,b,c,err = 0,0,0, 0.001
n=20
print(' k \t  x1 \t  x2 \t  x3')

for k in range(1,n):
    a=(7-(2*x2+x3))/10;          # Atualiza a e usa Xi anterior
    b=(-8-(x1+x3))/5;            # Atualiza b e usa Xi anterior
    c=(6-(2*x1+3*x2))/10;        # Atualiza c e usa Xi anterior
    if((abs(x1-a) < err) and (abs(x2-b))<err and (abs(x3-c))<err):
        break
    x1,x2,x3=a,b,c
    print('%2.d \t%.3f \t%.3f \t%.3f\n'%(k,x1,x2,x3))

print('\n***** Método Iterativo de Gauss-Seidal *****\n\n')
x1,x2,x3,x4 = 0,0,0,0
print(' k \t  x1 \t  x2 \t  x3')

for k in range(1,n):
    a , x1 = x1 , (7-(2*x2+x3))/10          # Guarda x1(k-1) em a;
    b , x2 = x2 , (-8-(x1+x3))/5;           # Guarda x2(k-1) em b;
    c , x3 = x3 , (6-(2*x1+3*x2))/10;       # Guarda x3(k-1) em c;
    if((abs(x1-a) < err) and (abs(x2-b))<err and (abs(x3-c))<err):
        break
    print('%2.d \t%.3f \t%.3f \t%.3f\n'%(k,x1,x2,x3))
```

Fonte: AUTOR,(2020)

# MN aplicados à Engenharia

## Recursividade Octave / MatLab

Sejam as matrizes: $A_{(n\ x\ n)}$ e $b_{(n\ x\ 1)}$, tal que $Ax = b$;

- $x = A\backslash b$ - A divisão à esquerda - *equivalente a x=**linsolve(A,b)***
- $x = b^t/A^t$ - A divisão à direita
- **x= inv(A)*B** ou $x = A^{-1} * b$
- **rref([A b])** - Solução pelo método de Gauss-Jordan
- **[L,U,P]=lu(A)** - Solução pela fatoração LU
  *L - é uma matriz triangular inferior;*
  *U - é uma matriz triangular superior;*
  *P - é a matriz de permutação.*

# MN aplicados à Engenharia

## Recursividade Python

Sejam as matrizes: $A_{(n \ x \ n)}$ $e$ $b_{(n \ x \ 1)}$, tal que $Ax = b$;

- **Utilizando o comando solve :** $solve(A, b)$
  $s1 = np.linalg.solve(A, b)$
- **Utilizando o comando** $inv(A)'$ **e** $'@'$ **:** $inv(A)@b$
  $s2 = np.linalg.inv(A)@b$
- **Utilizando o comando** $`inv(A)'$ **e** $`np.dot$ **:** $dot(inv(A), b)$
  $s3 = np.dot(np.linalg.inv(A), b)$
- **Utilizando o comando** $lu(A)$ **do** $scipy$ **:** $P, L, U = lu(A)$
  from scipy.linalg import lu
  $P, L, U = lu(A)$

# Exercícios

- **Veja a lista de exercícios na web**
- **Veja a lista de códigos em:** `https://github.com/jonathacosta/NM`